

Pendragon Forms
Version 5.1

Copyright Information

Copyright © 2005-2007 Pendragon Software Corporation. All rights reserved.

This documentation may be printed by licensee for personal use. Except for the foregoing, no part of this documentation may be reproduced or transmitted in any form by any means, electronic or mechanical, including photocopying or recording on any information storage and retrieval system, without prior written permission from Pendragon Software Corporation.

Pendragon Software

Pendragon is a registered trademark, and the dragon logo is a trademark of Pendragon Software Corporation.

Palm

Palm is a registered trademark, and Treo and Zire are trademarks of Palm Inc.

PalmSource

PalmSource, Inc., PalmSource, Palm OS, Palm Powered, Graffiti, HotSync and certain other trademarks and logos appearing herein, are trademarks or registered trademarks of PalmSource, Inc. or its affiliates or of its licensor, Palm Trademark Holding Company, in the United States, France, Germany, Japan, the United Kingdom, and other countries.

All other brands and product names may be trademarks or registered trademarks of their respective holders.

Contents

1. Getting Started.....	1
Installing Pendragon Forms	1
Installing Pendragon Forms on a Palm OS Handheld.....	6
Setting up a Wireless Palm OS Handheld.....	7
Installing Pendragon Forms on a Pocket PC Handheld	8
Synchronizing a Pocket PC Device.....	10
Performing a Network Client Installation	11
2. Quick Guide to Using Forms	13
3. Planning Your Data Collection System	24
Who Will be Using the Forms?	24
How many records can be stored on a handheld device?	25
What are the Limits of Form Designs and Records?.....	30
Do users need to share both form designs and records?	31
Has synchronization been tested?	35
How Often Should Users Synchronize?	36
Who is responsible for making backups?.....	37
4. Managing Form Designs	38
Making Changes to a Frozen Form.....	38
Copying a Form Design.....	39
Organizing Form Designs into Categories	40
Recycle Bin	41
Printing a Form Design.....	42
Deleting a Form from the PC.....	42
5. Synchronization Rules.....	43
The Synchronization Process.....	43
The Synchronization Cache	44
Users and User Groups.....	45
How to Delete Forms.....	50
Limiting the number of records on the Handheld	51
6. Using Forms on the Handheld	55
Installing Pendragon Forms on the Handheld	55
Pendragon Forms Main Menu (Forms List).....	57
Entering New Records	57

Layout View	58
Field View	59
Record View	59
Reviewing Records on the Handheld	60
Displaying Several Fields on the Review Screen	61
Quick Sum and Average	63
Filtering Records	64
Sorting Records	65
Deleting Records and Forms from the Handheld	67
To delete a form from a Pocket PC handheld:	69
Password-Protecting a Form Design	69
Setting Forms Preferences	70
Using the AutoNavigate Feature	72
Switching on Field Numbers (for Debugging Purposes)	73

7. Field Types 74

Available Field Types	75
Freeform Text Field	87
Numeric Field	90
Currency Field	92
Option 1 of 5 Field	94
Popup List Field	95
MultiSelection List Field	97
Date & Time Field	100
Date Field	102
Time Field	103
Yes or No Checkbox Field	104
Time Checkbox Field	106
Completion Checkbox Field	107
Section Field	108
Jump to Section Field	111
Lookup List Field	113
Cascading Lookup	119
Lookup to Another Form (Lookup to a Reference Form)	123
Lookup to a Read-Only Reference Form	127
Subform Field	133
Cascading Completion Checkbox from Parent to Subform	139
Single Subform Field	142
Signature Field	145
Button Field	148
Slider Field	150
Sketch Field	153

Image Field.....	155
Custom Control Field.....	162

8. Advanced Form Designer 165

The Form Designer Window.....	165
Field Tab.....	167
Data Tab.....	168
Visual Tab.....	169
Sizing Tab.....	180
Script Tab.....	184

9. Advanced Field Properties 185

Column Name	186
Autodefault	187
Primary Key.....	188
Required.....	189
Display Key	190
Default Value	191
Integer	192
Minimum Value & Maximum Value	193
Max Length.....	194
Pattern.....	194
Lookup List.....	195
Do Not Upload to PC.....	196
Non-Printing	196

10. Form Properties..... 197

Form Properties.....	197
Data Persistence.....	198
Access Rights.....	199
Category.....	200
Freezing a Form	201
Advanced Form Properties.....	202
Defaults Tab.....	202
Views Tab.....	204
Behavior Tab.....	207
Security Tab.....	210
Synchronization Tab	214
Additional Download Criteria	214
Beaming Tab.....	223
Changing Advanced Form Properties on a Frozen Form.....	225
Columns Tab.....	226

11. Managing Data on the PC	227
Viewing and Editing Data in the Database	227
Alternative Methods for Viewing Data	229
Deleting a Record.....	230
Exporting Data to Microsoft Excel	230
Exporting Data to ASCII (CSV File).....	230
Creating a Report in Microsoft Word	231
Creating Queries and Reports in Microsoft Access.....	235
12. Importing & Exporting.....	236
Import Button	236
Importing from a previous Pendragon Forms database	237
Importing Data from a Different Form Design.....	239
Importing & Exporting Form Designs	240
Importing & Exporting Lookup Lists.....	241
Importing & Exporting Data	242
Creating an ASCII (CSV) File	243
Creating a Form from a CSV File	244
13. Scripting Reference	245
Format of a Script.....	246
How Scripting Works & Limits of Scripts	246
Field Labels	247
Adding Comments to Scripts	248
Platform-Specific Scripts	249
Event Procedures.....	250
Variables.....	255
Functions.....	257
Scientific Functions	260
Operators	260
Statements	262
Assignment Statements	262
Conditional Statements	265
Action Statements	270
Scripting Errors.....	341
Using Scientific Functions in Scripts	343

14. Scripting Examples	347
Using Scripts to Perform Calculations	347
Calculation Example 1: Performing a simple calculation	348
Working with Dates.....	354
Branching	357
Using Scripts in Button Fields	364
Using a Script in a MultiSelection field	366
Using Scripts with Parent forms and Subforms	367
Using Scripts to do a Cascading Lookup	370
15. Using Bar Codes	373
Bar Code Scanners	373
Using a Microvision Flic Cordless Bar Code Scanner	374
Using the Socket Secure Digital Scan Card 3E	379
Creating a field for receiving bar code input	380
Using Scripts to Control Bar Code Scans	381
Scanning from any field on a form.....	385
Repeating Forms	386
Performing Reference Lookups.....	388
Scanning Multiple Bar Codes into the Same Record	393
Controlling which Bar Code Symbolologies can be scanned	394
Checking the Type of Bar Code Scanned	397
Troubleshooting Bar Codes	398
16. Printing from the Handheld	399
Serial Port Printing	399
Infrared or BlueTooth Printing.....	399
Printer Driver Software	399
Using Scripts to Control Printer Output	400
Printing a Single Form.....	405
Printing a Single Form using PrintBoy and HTML.....	408
Printing a Parent and Subform	412
Printing a Block of Text.....	416
Printing a Multi-Selection List Field	417
17. Working with Images.....	419
Decorative (Static) Images	420
Attached (Dynamic) Images	422
Attaching Images to Records on the PC	423
Attaching Images to Records on the Handheld	426
Taking a Photo on a Treo Smartphone	430

Drawing Non-Attached Image Files.....	433
Saving Images out of the Forms Manager Database.....	437

18. Internet and Serial Communications 439

Wireless / Serial Scripting Statements.....	440
E-Mailing a Record.....	441
Communicating with a Web Server.....	443

19. Beaming Records 445

Issue 1: Software required to beam records.....	446
Issue 2: Receiving handheld must allow beam receiving.....	446
Issue 3: Select primary key field(s) in your form design.....	447
Issue 4: Create a Button field containing a beaming script.....	447
Issue 5: Beaming Rules.....	451
Issue 6: Choose whether to mark received records as modified.....	452
Issue 7: Disable Record Overwrite Protection.....	452
Issue 8: Switch on Merge Changes.....	453
Memory Requirements when Beaming Records.....	454

20. Working with Multiple Forms 455

Example: Taking Orders on the Handheld.....	456
Advanced Scripting.....	460
Performing a Cascading Lookup to Another Form.....	461
Performing a Double Cascading Lookup.....	463
Updating Records in a Reference Form.....	466

21. Creating a Custom Main Menu 471

Features of a Custom Main Menu Form.....	471
Item 1: Set the Custom Main Menu form to Auto-Execute.....	472
Item 2: The form name must include the word MENU.....	474
Creating Custom Menu Options.....	475
Using a Custom Main Menu form to Filter Records.....	484

22. Linking to an External Access Database 486

Step 1: Creating a form from an existing database table.....	491
Step 2: Editing the form design for use on the Handheld.....	493
Step 3: Controlling which records go to the Handheld.....	498
Step 4: Linking the form to the external Access database.....	500
Step 5: Sending the form to the Handheld.....	504
Step 6: Viewing data on the PC.....	505
Troubleshooting Tips.....	505

23. Linking to an ODBC Database.....	506
Option 1: Creating a Linked Table in Access	506
Option 2: Mapping directly to an ODBC table	508
Troubleshooting Tips	508
Security concerns when using ODBC databases.....	509
24. Backing Up the Forms Database	510
Backing up the Database	510
Backing up Form Designs	510
Recovering Forms Designs	511
Backing up Data within a Form	511
VFS Backup on the Handheld	512
Warnings about using VFS Backup	513
Checking if VFS Backup is Enabled on the Handheld	514
Restoring Records from External Media	514
Purging Old Records from External Media	517
25. Using Forms in a Multi-User Environment.....	518
Entering a Multi-User Code	518
Setting up Users and User Groups	518
Sharing Records Across Handhelds.....	519
Choosing the right Multi-User Installation.....	520
Security concerns in a Multi-User environment.....	521
Appendix A: Sample Forms.....	524
Appendix B: Troubleshooting	526
Problems with Sending Form Designs or Data to the Handheld	527
Form Designer Error Messages	530
Problems with Freezing a Form Design	531
Problems with Distributing a Form	534
Problems with Lookup Lists.....	535
Problems with Subforms or Single Subforms	536
Problems with Working with Multiple Forms	539
Problems with Linking to an External Database	541
Synchronization Error Messages	545
Problems with Sending Data from the Handheld to the PC	556
Technical Support.....	558

Appendix C: What's New in Forms 5.1 559

Appendix D: Pendragon SyncServer..... 560

What is Pendragon SyncServer? 560
How Pendragon SyncServer Works with Pendragon Forms 561
SyncServer Cache 563
SyncServer Compression..... 564
Pendragon SyncServer Maintenance..... 565
SyncServer Client Connection Error Messages 572

Appendix E: GPS Custom Control..... 574

Adding the GPS Control to Your Form 574
Installing the GPS Control 576
Pairing a Handheld with a Bluetooth-enabled GPS Receiver 577
Using the GPS Control on the Handheld..... 578
Advanced GPS Features..... 579

Sample Forms

Pendragon Forms ships with some sample form designs, intended to illustrate the various features of the product.

You can send any of the sample form designs to the handheld.

See Appendix A for a list of the sample forms.

1. Getting Started

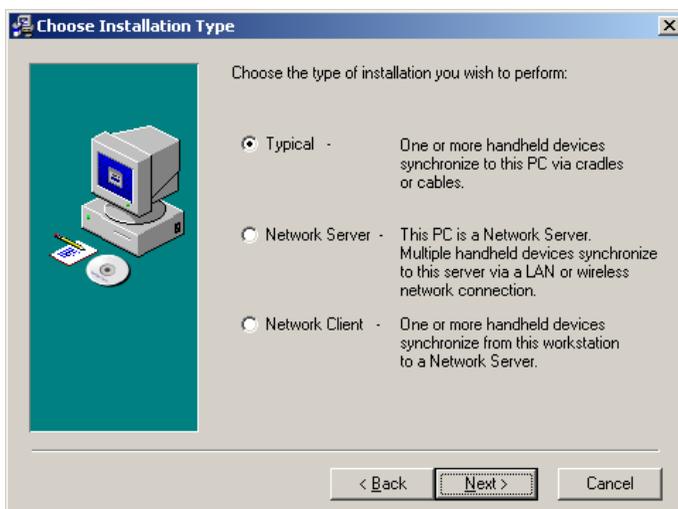
Installing Pendragon Forms

Installing Pendragon Forms 5.1 consists of two steps:

1. Installing the Pendragon Forms 5.1 desktop components on your PC.
2. Installing the Pendragon Forms 5.1 runtime on a Palm OS or a Pocket PC handheld.

Installing Pendragon Forms on your PC

1. Insert the CD-ROM into the CD-ROM drive.
If the CD-ROM program does not start, click the Start button and choose Run.
Type **D:\SETUP**, where **D** is the drive letter referring to your CD-ROM drive.
2. Choose *Install Pendragon Forms 5.1*.
3. If you are upgrading from an earlier version of Forms, install to the same folder where Forms was installed - typically the C:\Program Files\Forms3 folder. Your original Forms database will be renamed, and when you open the new Forms Manager you will be prompted to import all of your form designs and data. If you are not prompted to do the import, do a manual import - see page 237.
DO NOT UNINSTALL THE EARLIER VERSION OF FORMS.
4. Follow the onscreen prompts. You will be prompted for a type of installation:



Choose a **Typical** installation if you are installing on a PC that will synchronize handhelds via a direct cradle or cable connection.

Choose a **Network server** installation if you are installing on a server that will synchronize handheld devices connecting from other workstations or connecting wirelessly.

A Network server installation must be done on a server with a fixed TCP/IP address.

Choose a **Network client** installation if handhelds at this workstation will synchronize to Forms on a remote server. Go to page 11 if you are doing a Network client install.

5. After selecting a type of installation, you will be prompted to choose which version of Microsoft Access you have, if any. If you do not have a version of Microsoft Access, you will be prompted to install a runtime version of Access 2003.



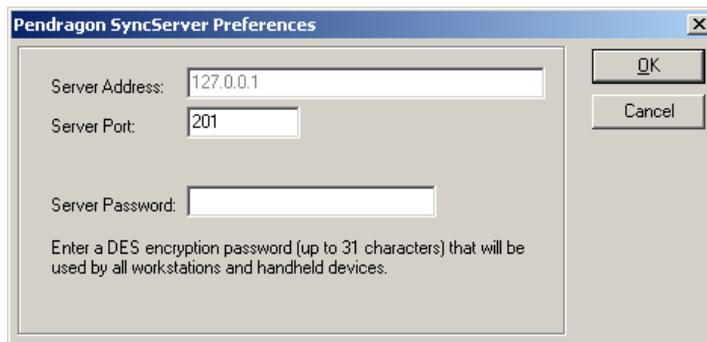
6. You will be prompted to enter the product Unlock Code. This serial number is located on the inside back cover of your manual, or in the case of the electronic version of the product, the Unlock Code will have been e-mailed to you.
7. You will be prompted for the directory folder where Forms will be installed. It is recommended that you keep the default folder of C:\Program Files\Forms3 especially if you are upgrading from a prior version of Forms.
8. You will be prompted for the name of the Start Group for the software. Keep the default group name of Pendragon Forms 5.1. You will then be prompted to install the software.

9. Once Pendragon Forms is installed, you will see a Pendragon SyncServer Service Manager screen. Pendragon Forms 5.1 ships with a single-connection version of Pendragon SyncServer, which is the technology that enables the handheld devices to synchronize to the PC. If you are using the single-connection version of SyncServer, leave all fields on this screen blank and continue to the next screen. If you have purchased a SyncServer license you can enter your SyncServer License information on this screen.



10. The next screen is the Pendragon SyncServer Preferences screen. The Server Address field contains a special IP address that always points to the local PC. The default Server Port is 201, meaning that SyncServer communicates on port 201. Keep this default for a **Typical** installation. If this is a **Network server** installation, keep port 201 unless your network administrator assigns a different server port number.

For both Typical and Network Server installations, if you have firewall software on the PC, you will have to open a port in your firewall to allow SyncServer communications on port 201. Programs such as Zone Alarm and Norton Internet Worm Protection (in Norton Antivirus) are firewall programs. Add exceptions in your firewall to allow the programs PFSERVER.EXE and PFSERVICE.EXE to accept incoming network connections.



Important: If you are going to synchronize across a network, it is strongly recommended

that you enter a Server Password. Make a note of the password that you have selected, as this password will also be used on client installations and on handheld devices. If you need to reset the Server Password at a later date, see page 570.

If you are synchronizing at one PC, entering a Server Password is optional. If you leave the Server Password blank, also leave it blank on the handheld(s).

11. If you have a Pocket PC handheld device, you will see this dialog. Choose Yes to queue the Forms 5.1 runtime to be installed on the Pocket PC device the next time the device is synchronized.



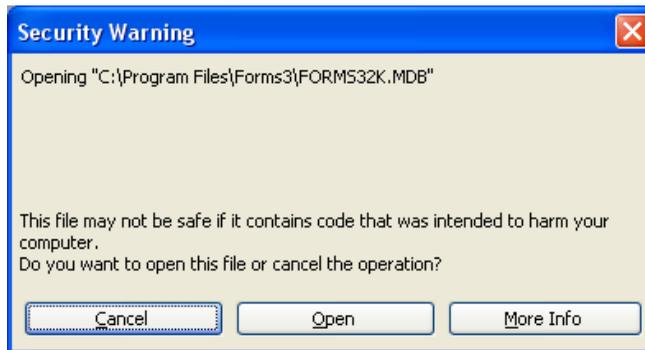
12. If you have a Palm OS handheld device, you will see this dialog. Click on the name of each Palm OS device on which you want to install the Forms 5.1 runtime. (You will need additional Forms licenses to run the software on multiple handhelds.) The software will be installed the next time each device is synchronized.



TIP:
If you have a Palm OS device that will be synchronizing wirelessly, check the Install S-Link checkbox to install the S-Link application as well as the Forms 5.1 application. S-Link is used to initiate wireless synchronization - see page 7.

13. After Pendragon Forms has been installed, your PC will be re-started. Once the PC has re-started, click Start...Programs...Pendragon Forms...Pendragon Forms Manager to begin creating forms.

Note to Microsoft Access 2003 users: You may see the security dialog shown below when you open the Forms Manager. This security mechanism can be disabled for all Access databases, but not for individual databases. For best security, leave the security mechanism in place and click the Open button each time you want to open the Pendragon Forms Manager.



14. The very first time that you open the Forms Manager, you will be prompted to enter a User Name. Type the handheld user name. If you choose not to enter a handheld user at this time, you can go to the User List and enter the handheld user name at a later time - see page 46.

PALM OS

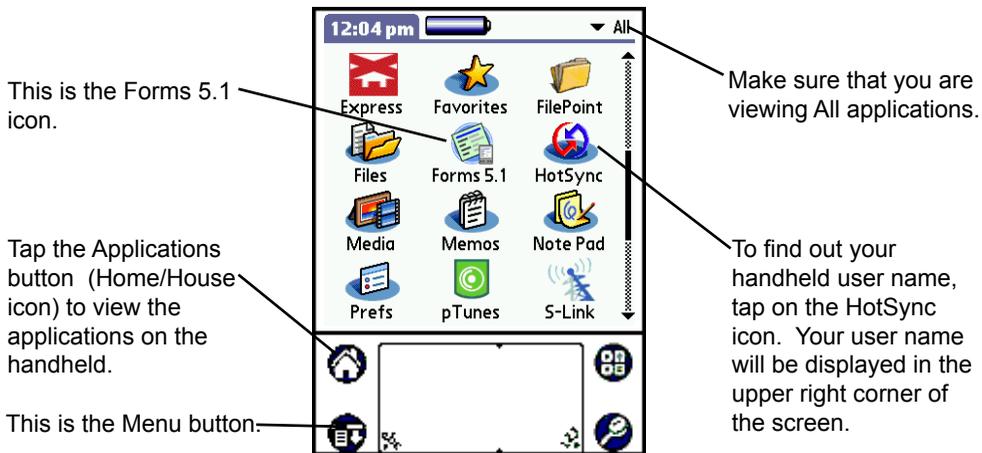
If you do not know the handheld user name of a Palm OS device, tap the HotSync icon on the handheld and the user name should appear in the upper right corner of the screen.

PPC

If you do not know the handheld user name of a Pocket PC device, tap Start...Settings on the handheld. In the Settings window, tap the System tab. Tap the About icon, and in the About screen, tap the Device ID tab. The Device name field contains the user name that Pendragon Forms will use.

Installing Pendragon Forms on a Palm OS Handheld

If you have just installed Pendragon Forms on a PC, the Forms runtime software should have been queued to be installed on the Palm OS handheld. Synchronize the handheld, then tap the Applications button (Home/House icon) and look for the Forms 5.1 icon.



If the Forms 5.1 icon is not on the handheld, do the following to install Forms 5.1:

1. On the PC, click on Start... Programs...Pendragon Forms 5.1...Install Forms 5.1 on Palm OS Device.
2. A Select Palm User(s) window will be displayed. Click on the name of a handheld user to select that user.
 - If the handheld is going to synchronize wirelessly to a server, also check the S-Link box to install the S-Link application that is needed for wireless synchronization (see page 7).Click the Done button.

If you have a Pendragon Forms license code for more than one user, you can select more than one handheld in the list. To de-select a name, click on the name a second time.

TIP: If you do not know what your handheld user name is, tap the HotSync icon on the handheld and the user name should appear in the upper right corner of the screen.

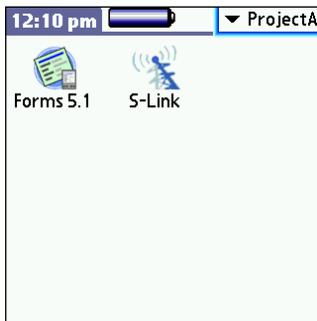
3. Press the button to synchronize the handheld. The Pendragon Forms program FORMS5.PRC will be installed on the handheld. Tap the Applications button (Home/House icon) on the handheld, and look for the Forms 5.1 icon.
4. Once the Forms 5.1 icon is on the handheld, you can begin creating forms. See page 13.

Setting up a Wireless Palm OS Handheld

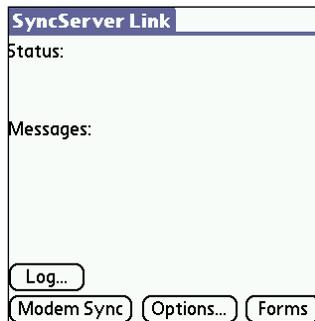
If you have a wireless Palm OS device such as a Palm Treo 650 or Treo 700p with data service, you can synchronize Pendragon Forms wirelessly to a network server. Pendragon Forms 5.1 must be installed on a network server with a fixed IP address.

In addition to the Forms 5.1 application, another program called S-Link is installed on the handheld. S-Link is used to initiate wireless synchronization of Pendragon Forms. See page 6 for instructions on installing Forms 5.1 and S-Link on the handheld.

S-Link must be configured in order to do a wireless synchronization.



Step 1:
Tap the S-Link icon to run S-Link.



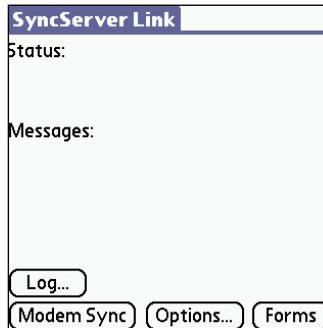
Step 2:
Tap the Options button on the SyncServer Link screen.



Step 3:
Verify that the Server IP Address, Server Port and User name are correct. The User name must be in the Forms Manager User List (page 46).



Step 4:
If you created a server password (see page 3-4), tap in the Password field and enter the server password. The Password field is case sensitive.



Step 5:
To initiate wireless synchronization, tap the Modem Sync button in S-Link. See page 545 if any problems.

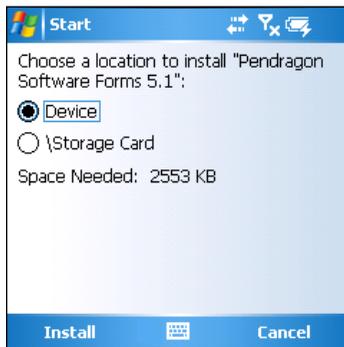
Tip:
New in Forms 5.1, instead of requiring users to tap the S-Link icon to initiate synchronization, you can add a Sync button to the Forms Main Menu on the handheld. See page 71.

Installing Pendragon Forms on a Pocket PC Handheld

If you have just installed Pendragon Forms on a PC, the Forms runtime software should have been queued to be installed on the next Pocket PC handheld that sits in the cradle.

Alternatively, you can install the software on a Pocket PC device by clicking (on the desktop PC): Start...Programs...Pendragon Forms 5.1...Install Forms 5.1 on Pocket PC Device.

When you sit the Pocket PC device in the cradle, or attach the device to the synchronization cable, Pendragon Forms will be installed onto the Pocket PC.



Follow the onscreen prompts on the handheld to confirm that you want to install Pendragon Forms. If you have an external storage card in the handheld, you will have the option of installing Pendragon Forms to the storage card. The default is to install to the device (the program runs faster if installed to the device).

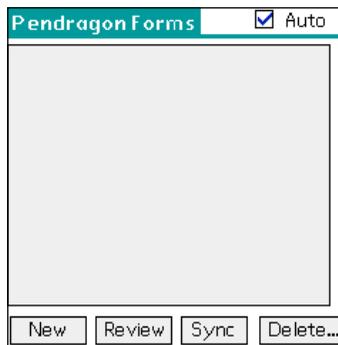
After Pendragon Forms is first installed on the handheld, a Launch screen will be displayed. Tap the Launch button to start Pendragon Forms.

After the initial installation process, to access Forms 5.1 on the Pocket PC handheld you can tap Start...Programs. The Forms 5.1 icon will be present on the Programs screen.

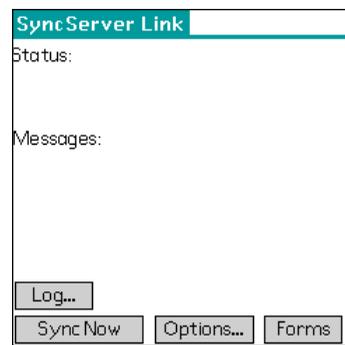
You will need to set up the Pocket PC device to synchronize Pendragon Forms.



Step 1:
Tap the Forms 5.1 icon to run the program.



Step 2:
Tap the Sync button on the Forms Main Menu screen.



Step 3:
On the SyncServer Link screen, tap the Options button.

Step 4:

Verify that the Server IP Address, Server Port and User name are correct.

The default Server address is PC, which works for synchronizing via a cradle or cable directly to a PC on which Forms is installed. **IMPORTANT:** If the Pocket PC device is going to synchronize from a client PC to a server, or is going to synchronize wirelessly to a server, you will need to change the Server address to the IP address of the server.

The default communications port is port 201. Leave this default if you are synchronizing to a standalone PC. If you are synchronizing to a server, verify that your network administrator is allowing Pendragon SyncServer communications on port 201. If another port is being used for Pendragon SyncServer, enter that port number instead of 201.

The User Name must be added to the Forms Manager User List (page 46).

Step 5:

If you created a Server Password when Pendragon Forms was first installed on the PC or server (see page 3), then tap in the Password field and enter the password. The Password field is case sensitive.

If you cannot remember the password, you can reset the password on the PC/server and use the new password on the handheld. See page 570.

On the Server Settings screen, the Auto-Sync checkbox is checked by default. This means that when the handheld user sits the handheld in the cradle, Pendragon Forms will synchronize.

If you switch off Auto-Sync, then you can manually initiate synchronizing Pendragon Forms by tapping the Sync button on the Forms Main Menu.

Tap Done on the Server Settings screen to return to the SyncServer Link screen. Tap the Sync Now button to synchronize Forms.

Synchronizing a Pocket PC Device

Automatic Synchronization

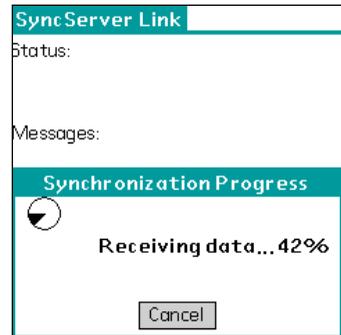
If Forms 5.1 is configured to automatically synchronize in the cradle (see page 9), then if you sit the handheld in the cradle, Forms 5.1 will automatically begin to synchronize.

Note: Forms 5.1 can synchronize whether you set up a Standard or a Guest Partnership in ActiveSync. Forms 5.1 only uses ActiveSync to obtain a TCP/IP connection for synchronization. If you use a Guest Partnership, you will need to set up the Guest Partnership before Forms can synchronize.



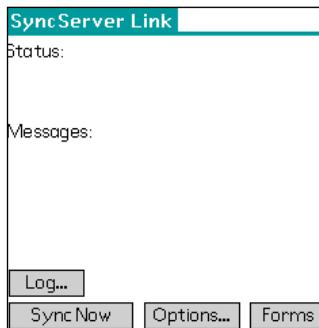
If Auto-Sync in Cradle is switched on, Forms will start synchronizing when the handheld is placed in the cradle.

At the end of the synchronization, tap the Forms button to return to the Forms Main Menu.



Manual Synchronization

To initiate Forms synchronization manually, tap the Sync button on the Forms Main Menu. On the SyncServer Link screen, tap the Sync Now button. At the end of synchronization, tap the Forms button to return to the Forms Main Menu.

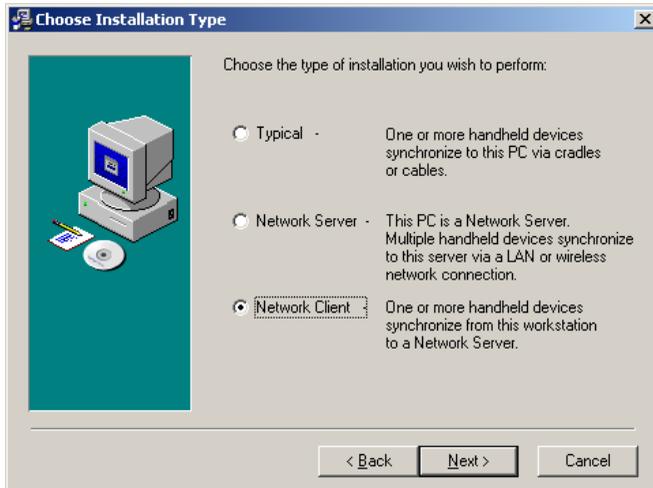


Tip:
Users with wireless Pocket PC devices should do manual synchronization.

Users synchronizing via a cradle should sit the handheld in the cradle before starting manual synchronization.

Performing a Network Client Installation

If handheld users are going to synchronize to their local workstations, but send their Pendragon Forms data to a server, you need to do a Network Server installation of Pendragon Forms on the server (see page 1-2), and then do Network Client installations of Pendragon Forms on each of the workstation PC's.



During installation, select to do a Network Client installation.

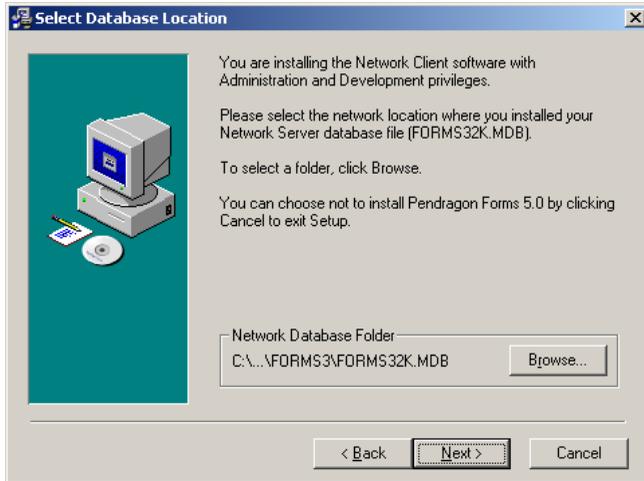
Note: Pendragon Forms should first be installed on the server as a Network Server installation before you do Network Client installations on the workstations.



You will be asked whether you want to install the Developer Tools.

Choose **Yes** if the user at this workstation will need to access the Forms Manager database to design forms and view data.

Choose **No** if the user at this workstation only needs to synchronize their handheld device, and will not need to design forms and view data on the server.



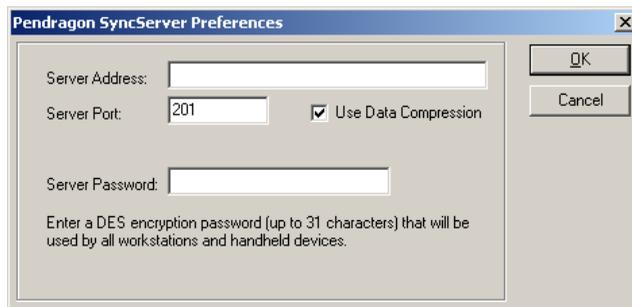
If you choose to allow access to the Developer Tools, you will be prompted to select the location and name of the Forms Manager Database.

Click the Browse button and select the Forms Manager database (FORMS32K.MDB) that was installed on the server.

You will be prompted to enter the Server IP address, Server Port and Server Password of the server where the Pendragon Forms database is located. The Server Password was created when Pendragon Forms was installed on the server (see page 3).

Note 1: If the Server Address displays 127.0.0.1, this is NOT correct for a Network Client installation. That address means the local PC which would be the client, not the server. Ask your network administrator for the server IP address and server communications port number.

Note 2: You can also find out the server IP address by going to the server and clicking Start...Run. Type **cmd** and click OK. A Command window will be displayed. Type **IPCONFIG** and press Enter. The IP address of the machine at which you are typing will be displayed. If you are at the server PC, the IP address will be the server IP address.



Finally, you will be prompted to install on the handheld device(s). Select the Palm OS devices that will be synchronizing from this client workstation, or select to install on a Pocket PC device. Synchronize the handheld device(s). See pages 8-9 for additional Pocket PC setup information.

2. Quick Guide to Using Forms

This chapter gives you an overview of how Pendragon Forms works.

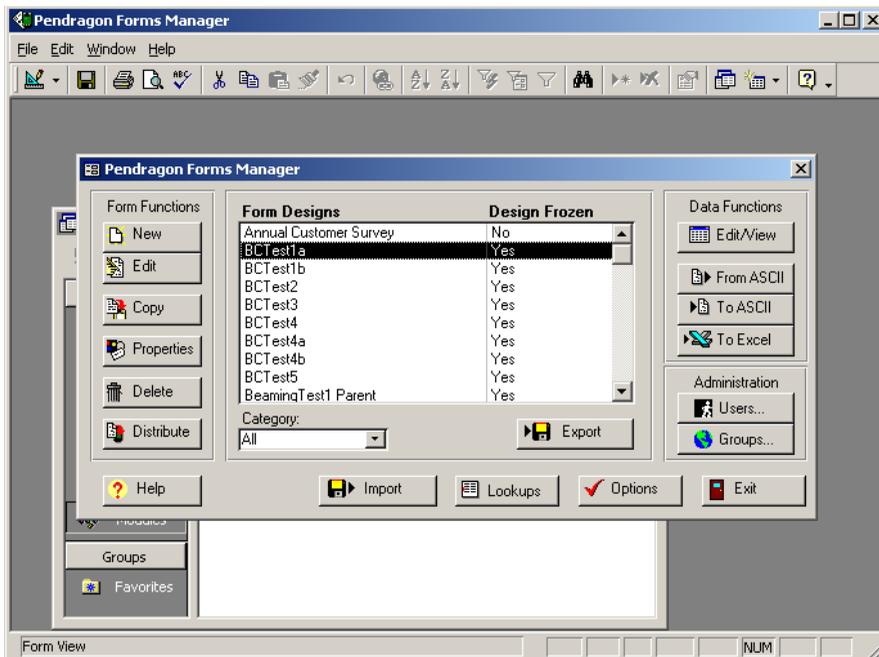
On the PC, open the Forms Manager by clicking Start...Program Files...Pendragon Forms 5.1... Pendragon Forms Manager.

The Forms Manager window in the foreground and is where you will design forms and view data.

To create a new form, click the New button.

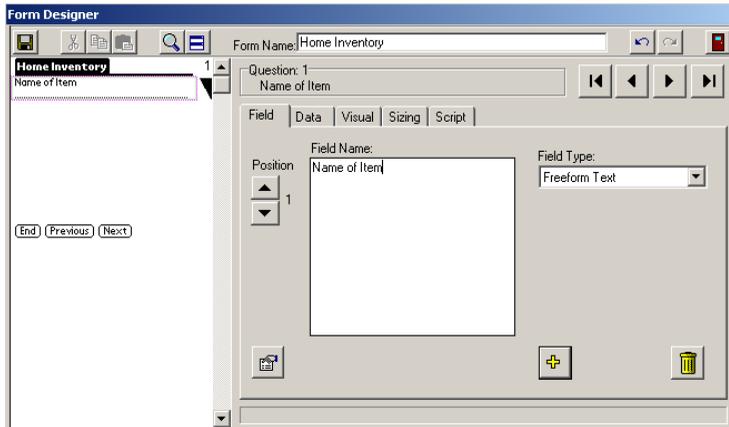


If you are editing an existing form, click the name of the form to highlight it, and then click the Edit button.



When you click the New or Edit button, the Form Designer window is displayed. The Form Designer window is where you design your form.

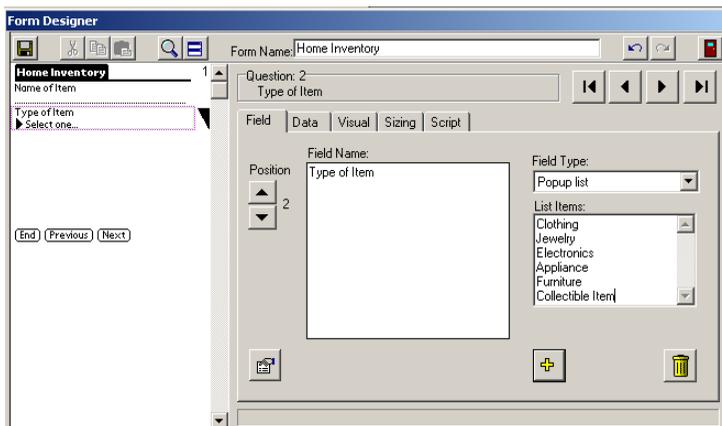
- Fill in the Form Name field at the top of the window to give your form a name.
- Click in the Field Name field and type the name of the first field on your form.
- In the Field Type field, select the type of data that the handheld user can enter in this field.



To add another field to the form, click the  button.

Or, if you are on the last field of the form, click the right arrow button to create a field. 

There are 23 field types that you can use. Several field types require you to enter additional information. For example, a Popup List field, shown below, requires you to enter the options that you want the user to see in the Popup List. See Chapter 7, *Field Types*, starting on page 74 for details about the 23 field types that you can use.

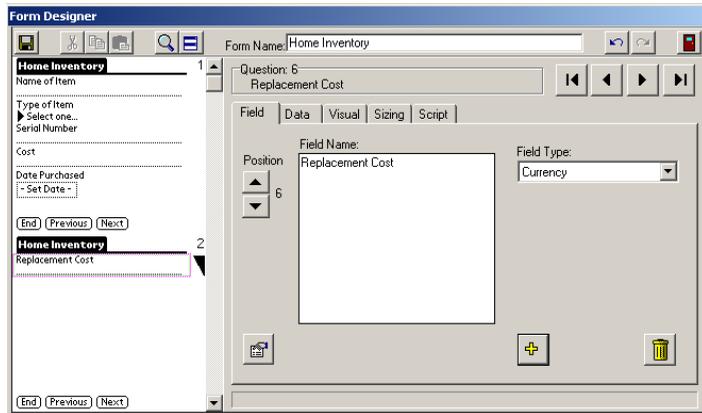


As you add fields to your form design, the Preview Area on the left side of the Form Designer window shows you a preview of what your form will look like on the handheld.

If your form design is several screens long, you can scroll up and down the Preview Area to find a particular screen.

Click on a field in the Preview area to select that field to edit.

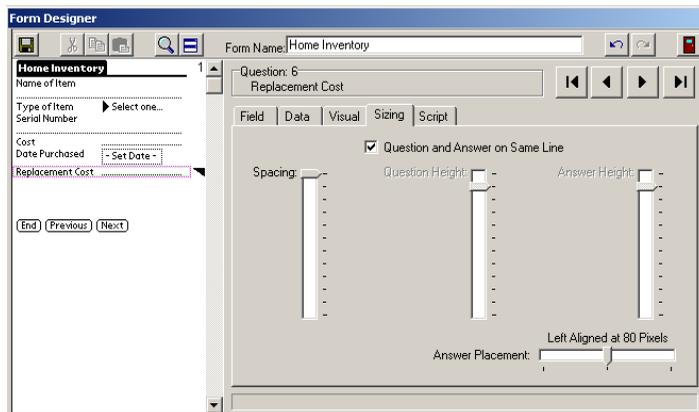
You can also use these buttons  to move from field to field.



The tabs in the Form Designer: Field, Data, Visual, Sizing and Script, apply to the selected field.

The Sizing tab lets you control how a field is displayed on the handheld screen. To reduce the screen space that a field occupies, check the Question and Answer on Same Line checkbox. The field name (question) and the handheld user's response (answer) will take up one line on screen.

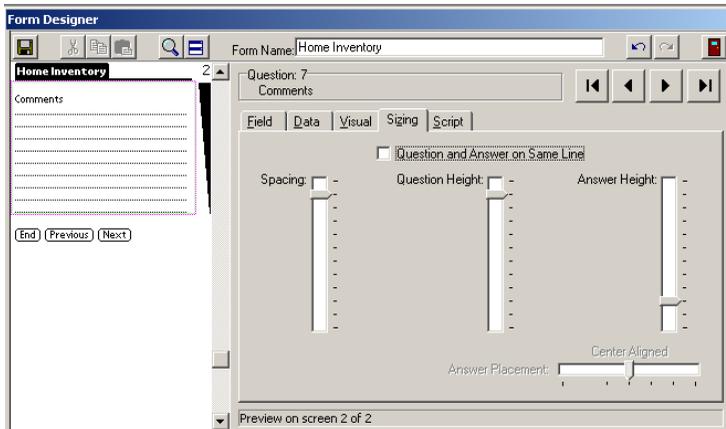
The Answer Placement control lets you position the answer to the left, middle or right of the screen.



Some fields display better if they take up more screen space, not less.

If the question and answer are on different lines, you can add Spacing before the question, and adjust the Question Height and the Answer Height as needed.

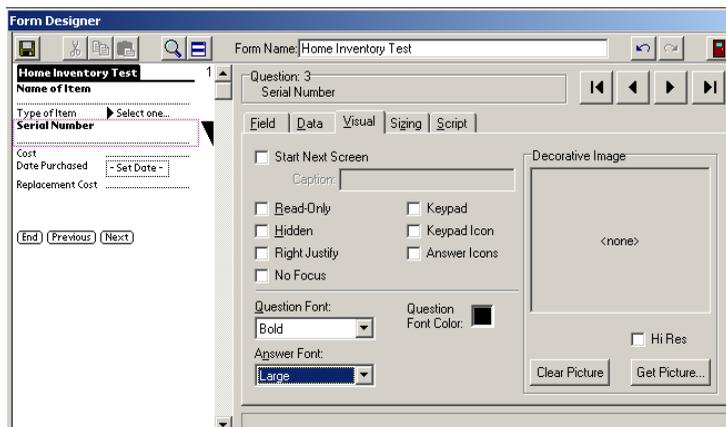
The question and answer components of a field cannot be larger than a single screen.



The Visual tab in the Form Designer window allows you to change the font for the question and answer components of the current field.

Fonts can be Standard, Bold, Large, or Large and Bold.

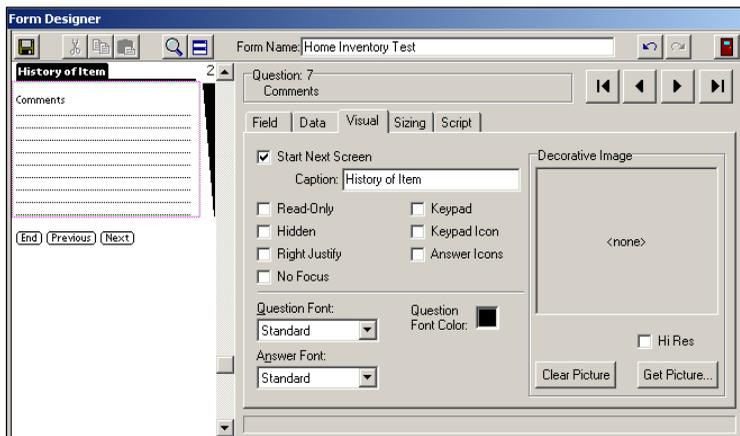
If you are designing for a color handheld device, you can also make the question font a color other than black.



If you have a field that you want to start at the top of a screen, you can check the Start Next Screen checkbox on the Visual tab.

In the Caption field, you can type an optional header for that screen. If you leave the Caption field blank, the default header is the form name.

To learn more details of the Form Designer window, see Chapter 8, *Advanced Form Designer*, starting on page 165.



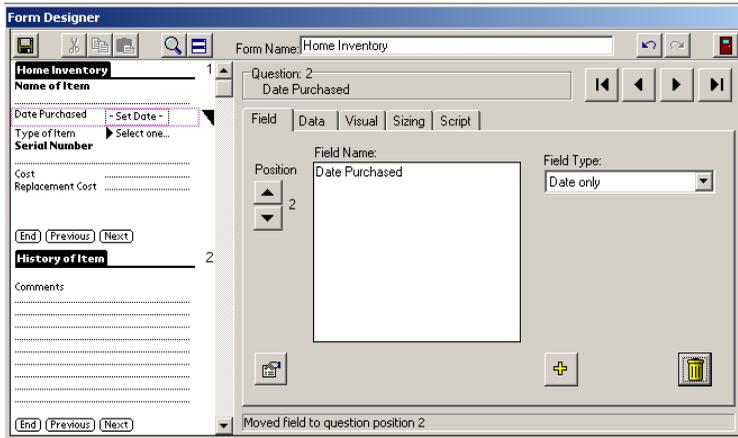
IMPORTANT:

If you are creating a form with a lot of fields, click the  Save button in the Form Designer window often, to save your work regularly.

To re-position a field, click on the field in the Preview Area to select that field.



Then click the Up or Down arrows to move the field to the previous or next position on the form.

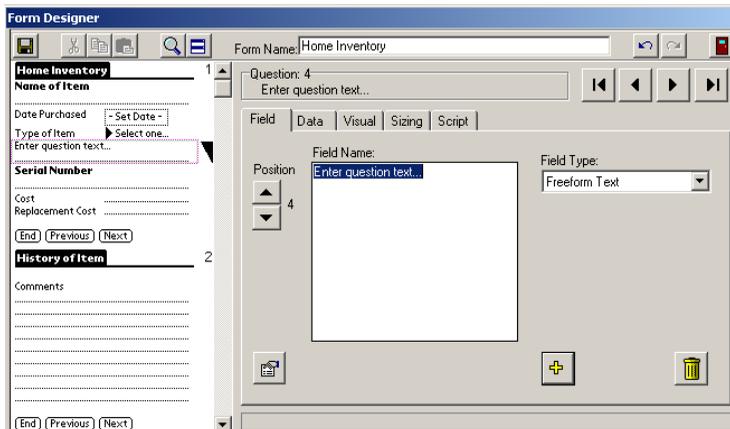


To insert a field in the middle of a form, click on a field in the Preview Area to select the field.

Click the  button to add a new field after the selected field. Remember to type a field name and select a field type for the new field.

To delete a field, click on the field in the Preview Area to select the field. Then click the Garbage Can button.

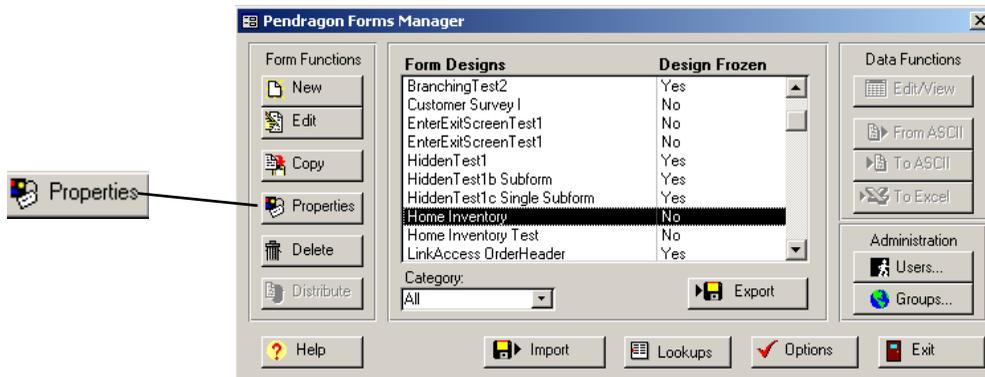
To save your form and exit the Form Designer, click the Exit button. Choose Yes when prompted to save the form.



The next step in the form design process is to select form properties and freeze the form design.

Freezing a form design creates a database table within the Forms Manager database for storing all the records associated with a form. You cannot add, delete or move fields once the form is frozen.

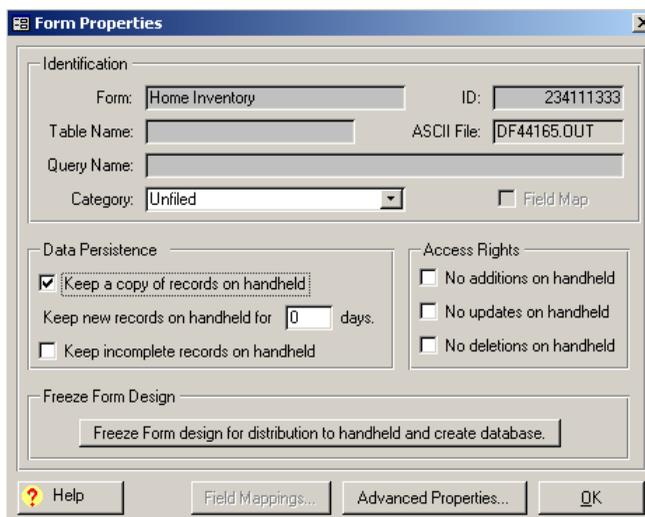
To freeze a form, click on the name of the form to highlight it, and then click the Properties button.



In the Data Persistence section of the Form Properties window, choose how long you want to keep records on the handheld after synchronization.

Check the Keep a Copy of Records on Handheld checkbox if you want the handheld to have a copy of the records after synchronization.

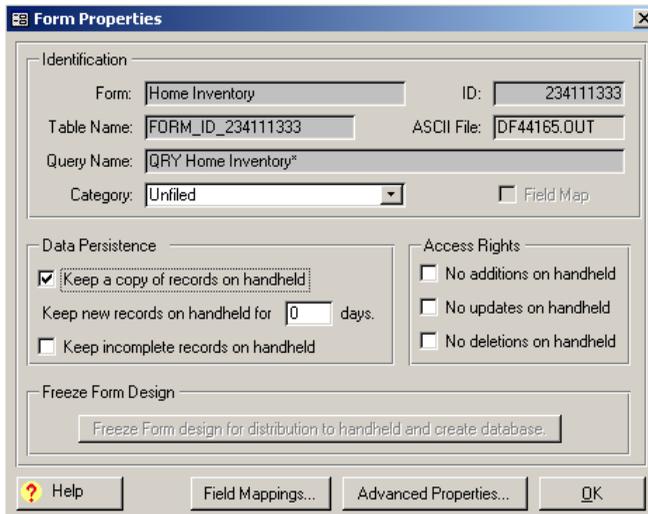
To freeze the form design, click the button labeled Freeze Form Design for distribution to handheld and create database.



Once the form is frozen, the Table Name field in the Form Properties window displays the name of the database table that was created for storing records for this form.

The ID field is a unique Form ID number. A form on the handheld has to match the Form ID on the PC for the form to be able to synchronize. It is therefore very important not to delete a form design from the PC if you are still using the form on the handheld.

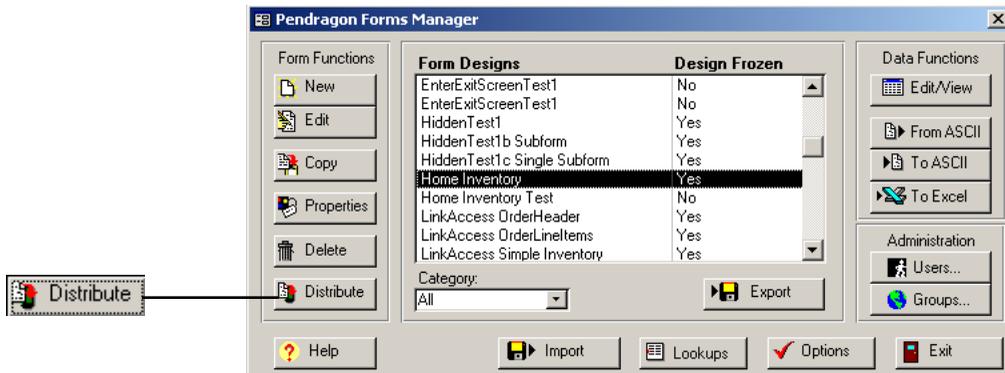
Click the OK button to close the Form Properties window.



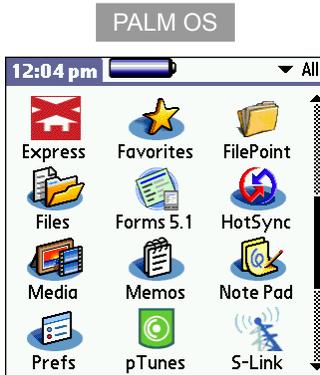
Once you have frozen a form, the next step is to distribute the form to the handheld device.

Click on the name of the form in the Forms Manager window, then click the Distribute button.

The form will be added to the Default User Group, and members of that user group will receive the form design on their next synchronization.



If this is the first form that you are sending to the handheld, make sure that the Forms application is first installed on the handheld.



To check if Forms is installed on a Palm OS device, tap the Applications button (Home/House icon) and tap in the upper right corner of the screen to view All applications. Look for the Forms 5.1 icon.

If Forms is not installed on the handheld, refer to page 6 for installation instructions.

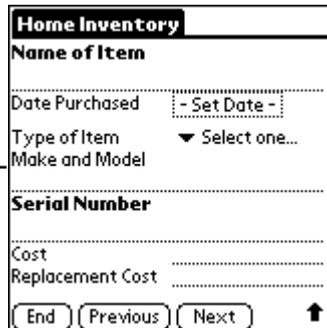


Synchronize the Palm OS device to receive any new or changed form designs. (If you have just synchronized to install the Forms 5.1 application, you will need to synchronize a second time to receive the form designs.)

If a HotSync Problem dialog is displayed, see Appendix B, Troubleshooting, page 545.



After synchronization, tap on the Forms 5.1 icon to view the Forms Main Menu. To fill in a form, tap on the name of the form and then tap the New button. This creates a new record for the form.



To fill in a field, tap in the answer part of the field and then select or write your answer.

If a form has more than one screen, tap the Next button to move to the next screen.

Tap the End button to save and exit a record.

PPC



To check if Forms is installed on a Pocket PC device, tap Start...Programs. Look for the Forms 5.1 icon in the Programs window.

If Forms is not installed on the handheld, refer to pages 8-9 for installation instructions.



Tap the Forms 5.1 icon to open the Forms application.

Then sit the handheld in the cradle (or connect a synchronization cable). Pendragon Forms should initiate the synchronization.

If Forms does not begin synchronizing, you can manually start synchronization - see page 10.

If you experience synchronization problems, see page 545.

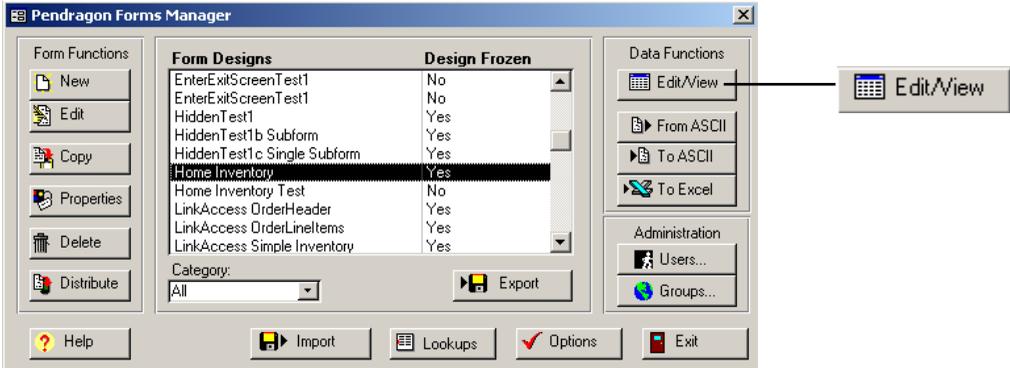


After synchronization, tap the Forms 5.1 icon to view the Forms Main Menu. To create a new record, tap on the name of the form, then tap the New button.

Tap in a field and use the onscreen keyboard to enter data for that field. Tap the Next button at the bottom of the screen to move to the next screen. Tap the End button to save and exit the record.

To send your data from the handheld to the PC, synchronize the handheld.

To view your data on the PC, open the Forms Manager program. Click on the name of your form, and then click the Edit/View button.



In the database table containing the records for your form, each field is a column in the database table, and each record is a separate row of the table.

Pendragon Forms adds four fields to the front of the database table, that the program uses internally: RecordID, UnitID, UserName and TimeStamp.

The Timestamp is the creation date and time of the record.

The UserName is the user name of the handheld device that most recently changed the record.

IMPORTANT: Never delete or change columns in the database table, or the form will not be able to synchronize to the database table.

RecordID	UnitID	UserName	TimeStamp	NameOfItem	DatePurchased
0	0	Sarah Jane	3/29/2003 5:38:01 PM	DVD player	1/19/2003
0	0	Sarah Jane	3/30/2003 4:51:13 PM	Tea set	8/21/1950
0	0	Sarah Jane	3/30/2003 4:53:35 PM	Gold Charm Bra	12/24/1962
*	0	0	No one	3/30/2003 4:55:57 PM	

Record: 1 of 3

3. Planning Your Data Collection System

Before you design a form in Pendragon Forms, it is a good idea to think about the entire data collection system that you are going to implement. You need to know the answers to the following questions in order to design a data collection system that works for your organization:

- Who will be using the forms?
- How many records can be stored on a handheld device?
- What are the limits of form designs?
- Do users need to share both form designs and records?
- Has synchronization been tested?
- How often should users synchronize?
- Who will be responsible for making backups to protect the data and the form designs?

This chapter looks at these questions, to show how the answers affect your form design and data collection system.

Who Will be Using the Forms?

It is important to think of the audience who will be using your forms on handheld devices. Instead of concentrating on making the data easy to analyze once it is on the PC, focus on making the form easy to use on a handheld device. If the handheld users find a form easy to use, they will be more likely to enter data correctly.

To help improve the accuracy of data entry, try to minimize the use of Text fields. If there is a question on your form for which there are several known responses, use a Popup List or a Lookup List field to let the handheld user select a response. The user will save time by not having to write text, and the data will be more accurate because it will not be prone to typographical errors.

How many records can be stored on a handheld device?

A handheld device has a limited amount of memory, and when you design your data collection system, it is important to think about how many records you want to keep on the handheld, and how to manage removing records from the handheld. If you remove records from the handheld on a regular basis, you will avoid exceeding the available memory on the handheld.

Data Persistence Options

The Form Properties window in the Pendragon Forms Manager gives you several Data Persistence options for removing records from the handheld. To access the Form Properties window, click the name of the form in the Forms Manager, and then click the Properties button.

The default Data Persistence option is that no checkboxes are checked. This means that after a successful synchronization, all records are removed from the handheld.

You can change the Data Persistence option to:

- Keep a copy of records on the handheld indefinitely.
- Keep records on the handheld for a limited time, such as 7 days or 30 days.
- If you put a Completion Checkbox field on your form (see page 107), you can choose to Keep incomplete records on handheld. This means records stay on the handheld until the user checks the Completion Checkbox field to request removal of the record on the next synchronization.

To read more about the Form Properties, see Chapter 10, *Form Properties*, page 197.

How much space does Pendragon Forms require on the handheld?

The Pendragon Forms program itself takes up a certain amount of space on the handheld, and then each form design will take up a certain amount of space. However, it is the records in your form that will take up the most space on the handheld. If you are using a Pocket PC device, or if you are wirelessly synchronizing a Palm OS device, storage space will also be required during synchronization.

The following table illustrates approximate storage space requirements of Pendragon Forms:

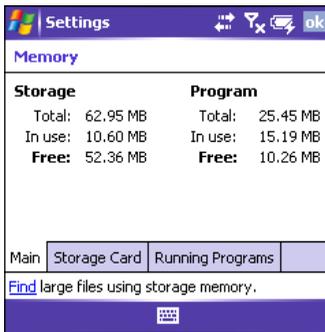
	PALM OS	PPC
Program size before any form designs:	370 KB (kilobytes)	3 MB (Megabytes) of storage space, plus 10-20 MB of program memory.
Large form design: (e.g. 100 fields, no graphics)	35KB	35KB
Data in one (1) field:	45 bytes approx.	45 bytes approx.
Data in one (1) record of a 20-field form:	20 x 45 = 900 bytes	20 x 45 = 900 bytes
Data in 1000 records of a 20-field form:	900 bytes x 1000 records = 900,000 bytes or close to 1MB	900 bytes x 1000 records = 900,000 bytes or close to 1MB
Additional storage space required during synchronization:	If a Palm OS device is synchronizing wirelessly or via an attached modem, then synchronization is done via the SyncServer S-Link program on the handheld. During synchronization, S-Link requires as much free memory as the Forms database on the handheld occupies. If the Forms database on the handheld occupies 2 MB, S-Link will need another 2 MB free in order to synchronize the handheld.	During synchronization, all Pocket PC devices require as much free memory as the Forms database on the handheld occupies in storage memory. Therefore, if the Forms database on the handheld occupies 2 MB, the program requires another 2 MB free in order to synchronize the handheld. This applies even for local cradle/cable synchronization of Pocket PC devices.



PALM OS

To check how much space is being used by Pendragon Forms on a Palm OS handheld, tap the Applications button (the Home/House icon) on the handheld. Then tap the Menu button (the drop-down menu icon below or near the House icon). On the App menu, tap the Info menu option.

The Info screen on the handheld shows you how much free space is available on the handheld, and how much space Forms 5.1 is taking up.



PPC

Pocket PC devices do not indicate specifically how much memory each application is using, but you can check how much free memory is available on the device.

To check available memory, tap the Start button, then tap Settings. In the Settings window, tap the System tab and scroll down to look for the Memory icon. Tap the Memory icon to display how much memory is in use and how much memory is available.

Storage Limit Per Form Design

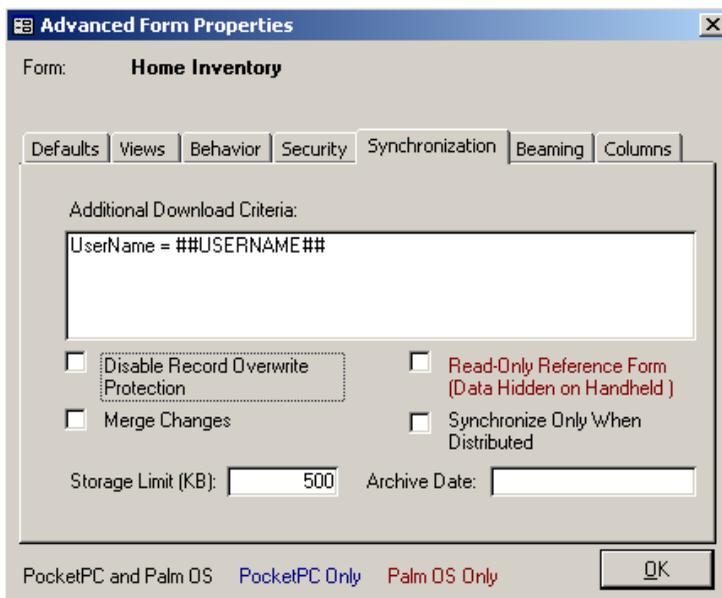
To prevent the handheld from being completely filled up with records, Pendragon Forms has a default Storage Limit per form design of 500KB (kilobytes). This means that by default, you can only maintain up to 500KB worth of records on the handheld per form design. (Note: you can collect more than 500KB of new records on the handheld, but after synchronization only up to 500KB of records will be sent back to the handheld.)

If you have a lot of free space available on the handheld, you can choose to increase the Storage Limit for a particular form design, to store more than 500KB.

The Storage Limit is an Advanced Form Property. To access the Advanced Form Properties window, click the name of the form in the Forms Manager, then click the Properties button. In the Form Properties window, click the Advanced Properties button. Click the Synchronization tab to access the Storage Limit field.

The default value in the Storage Limit field is 500, which stands for 500KB. You can change this value to 1000, 1500 or higher as needed. Close the Advanced Form Properties window and the Form Properties window, and then synchronize the handheld to receive the increased number of records.

To learn more about other Advanced Form Properties that can be used to control the number of records on the handheld, see page 214.



Wireless device considerations

If you make an implementation decision to use wireless handheld devices instead of synchronizing via a cradle or cable, it is important to note that the synchronization speed of a wireless device is not the same as a local synchronization. This means that if you put the same amount of data on a wireless device as on a handheld that is synchronizing locally, the wireless device may take longer to synchronize.

Cradle speeds:

Local serial sync = 56,000 bps (bits per second)

Local USB sync = 400,000 bps minimum

Wireless speeds:

GSM phone = 9600 bps

GPRS (hi-speed GSM upgrade) = typical 32,000 bps; max 64,000 bps

EDGE, 1xRTT/CDMA 2000 phone = 100,000 bps

HSDPA, EV-DO phone = >1 Mbps

WIFI = 10 Mbps

If the synchronization time is taking longer than you would like, you can consider the following:

- a) Reduce the number of records that are synchronizing on the handheld.
If there are records that the user does not need to keep on the handheld, remove them. See Data Persistence on page 198 and Additional Download Criteria on page 214 for options on how to select which records are removed.

- b) If you are using certain forms to maintain fixed reference information on the handheld, you can switch off synchronization for just those form designs.
See Advanced Form Properties, page 222.

In addition to synchronization speed, another consideration is the extra memory that is required on the handheld to do wireless synchronization. On Palm OS devices, wireless synchronization is performed via the Pendragon SyncServer S-Link program. During synchronization, S-Link requires an amount of free memory equal to the size of the Forms database on the handheld. If the Forms database takes up 1MB on the handheld, then S-Link will need an additional 1MB to synchronize. Pocket PC devices always require this extra memory for synchronization, whether the synchronization is wireless or not. Palm OS devices only require the extra free memory for wireless or modem synchronization, not for cable/cradle synchronization.

What are the Limits of Form Designs and Records?

Pendragon Forms has some limits that apply to the size of your form designs, and the size of the records that can be created.

The maximum number of fields that you can have on a form is 250 fields.

If you need to create a form with more than 250 fields, you can create two forms. To create an application that allows the user to start filling out one form and then jump to a second form to finish, consider a parent form and a single subform. See page 142 for information on single subforms.

There are other limits on the handheld and in the Microsoft Access database on the PC that may restrict you to fewer than 250 fields in your form design.

Microsoft Access has some limits on the size of form designs that can be viewed on the desktop. Pendragon form designs with a large number of Signature, Sketch, Image or Text fields (with more than 255 characters) may run into this limitation which will prevent the form design from being frozen.

There is a 64KB limit on the handheld for the actual size of the form design, including field names, field types and scripts. If you create a lot of MultiSelection Lists or Popup Lists with many items in each list, or you have a lot of scripts on your form, you may exceed this size limit for the form design. If you run into this problem, the form design will not be downloaded to the handheld. The solution in this case is to create smaller forms.

Finally, Microsoft Access has limitations on the size of a record that can be stored in a database table. If you have a lot of Text fields on your form, and the handheld user enters the maximum number of characters in each Text field, you may run into the problem of a record being too large to fit in the Access database. Records that exceed this limit will generate a "Record is too large" message during synchronization. In this case, the only solution is to delete or abbreviate text in the offending record.

Do users need to share both form designs and records?

Pendragon Forms is optimized to facilitate users sharing the same form design, but not the same records. An example of this is a work order system in which work orders are assigned to various handheld users, who complete their assignments independently of each other.

In some instances, however, you may want handheld users to share records as well as sharing form designs. For example, a group of physicians may share the same patients, and would need the patient information to be available on each physician's handheld device.

By default, Pendragon Forms does two things to make it easy for separate handhelds to maintain separate records:

1. The default primary key that is created when a form is frozen consists of three fields: the UnitID, the handheld UserName and the creation Timestamp of the record. The primary key is unique for each record, and by including the handheld UserName in the default primary key, Pendragon Forms ensures that if two users were to create a record on their handhelds at exactly the same time (i.e. with the same Timestamp), their two records would still be considered different because each record would have a different UserName.
2. The default Advanced Form Property that determines which records from the PC are sent to the handheld, is called the Additional Download Criteria. The Additional Download Criteria is pre-set so that only the records assigned to a user are sent to that user's handheld device.

RecordID	UnitID	UserName	TimeStamp	NameOfItem	DatePurchased
0	0	Sarah Jane	3/29/2003 5:38:01 PM	DVD player	1/19/2003
0	0	Sarah Jane	3/30/2003 4:51:13 PM	Tea set	8/21/1950
0	0	Sarah Jane	3/30/2003 4:53:35 PM	Gold Charm Bra	12/24/1962
*	0	No one	3/30/2003 4:55:57 PM		

Record: 1 of 3

You can see the default primary key fields UnitID, UserName and Timestamp, when you click on a frozen form in the Forms Manager, and then click the Edit/View button.

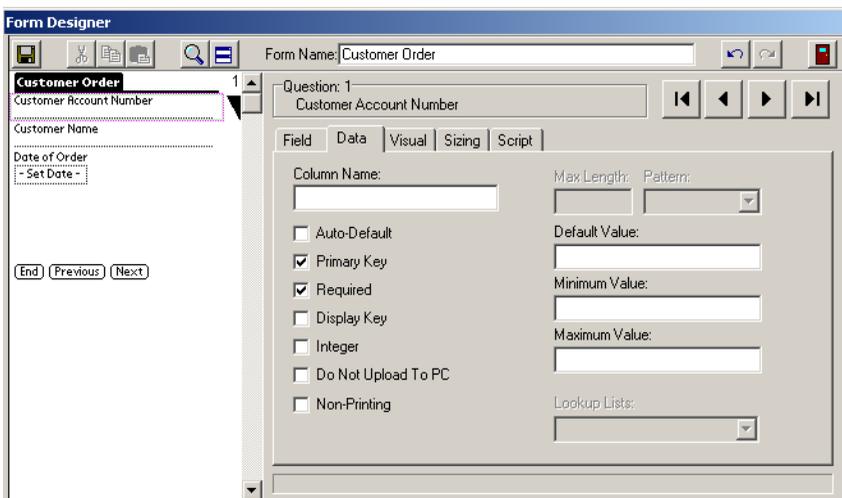
If you want users to be able to share records, here is what you will need to do to overcome the default settings that prevent the sharing of records:

1. Select your own primary key when you design your form, before freezing the form.

With the default primary key (UnitID, UserName and Timestamp), if one user creates a record, the UserName of that user is assigned to the record. If the record is also on a second user's handheld, and that user changes the record, the UserName of the second user will be stored in the record. When both users synchronize, duplicate records will be created, because the primary keys of the two records will be different since the two UserNames are different.

To avoid the problem of duplicate records, select your own primary key that does not depend on the UserName. The primary key has to be one or more fields that in combination will be different for every record. For instance, in the Customer Order form design shown below, the Customer Account Number is not unique if the same customer places more than one order. However if the Customer Account Number plus the Date of Order is unique for every order, then those two fields can be the primary key fields.

To make a field a primary key, click the Data tab in the Forms Designer, and check the Primary Key checkbox. The Required checkbox is automatically checked as well, because primary key fields cannot be left blank. See Advanced Field Properties, page 188 for more information on primary keys.



2. Remove the Additional Download Criteria that causes records to be sent to a handheld only if the UserName of the record matches the UserName of the handheld.

To access the Additional Download Criteria, click the name of the form in the Forms Manager, then click the Properties button, then click the Advanced Form Properties button. In the Advanced Form Properties window, click the Synchronization tab.

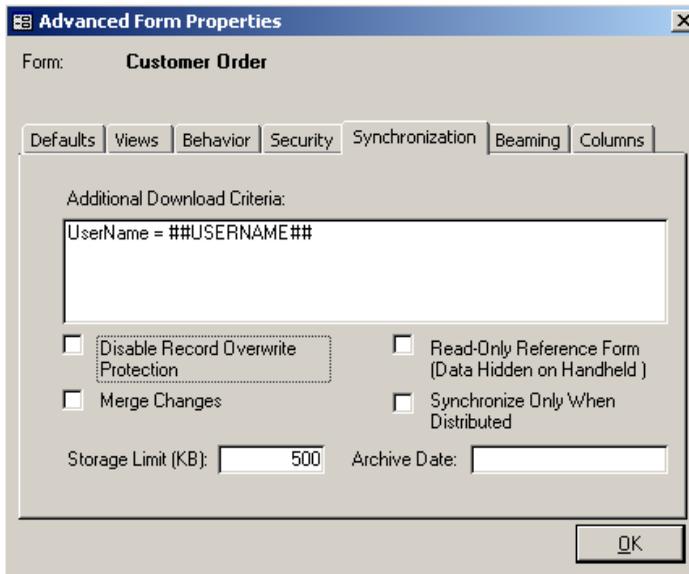
The default Additional Download Criteria setting is:

USERNAME = ##USERNAME##

which means that the UserName field of the record has to match the Username of the handheld device for the record to be sent to the handheld.

To allow all users to receive the record, delete the default download criteria.

For further information on Additional Download Criteria, see page 214.



To allow users to receive a record no matter what the UserName, erase the default Additional Download Criteria so that the field is blank.

A final consideration when sharing records across handhelds is the order in which the handhelds will synchronize back to the PC. During synchronization of Pendragon Forms, new or changed records on the handheld overwrite the existing record on the PC. If two handhelds modify the same record, the first handheld to synchronize will upload its data to the PC, but then the second handheld to synchronize will overwrite the data from the first handheld. This can be a problem if the handheld with less-up-to-date information synchronizes last.

If users who change the same record are actually changing different fields of the record, it is possible to set an Advanced Form Property to merge changes. That way the first person who synchronizes will upload their changes to the PC, and then if the second person has changed different fields on the form, their changes will be merged into the record on the PC when they synchronize. See page 220 for information on merging changes. However, if the users change the same field of the same record, then the last user to synchronize will again overwrite the first, even with Merge Changes switched on.

If you do not want to risk one user accidentally overwriting another's record, then you need to plan your form design so that users never modify the same record, and instead just create their own records. In the case of medical personnel sharing patient records, for example, you can create a parent form for storing the basic patient information, and a subform for storing the details of a patient visit. Each handheld user can access the parent form record but not modify it, and creates a separate subform record to log their observations when they see the patient. Since users are creating new subform records, they are not modifying the same record and will not overwrite each other. For information on using parent forms and subforms, see page 133.

Has synchronization been tested?

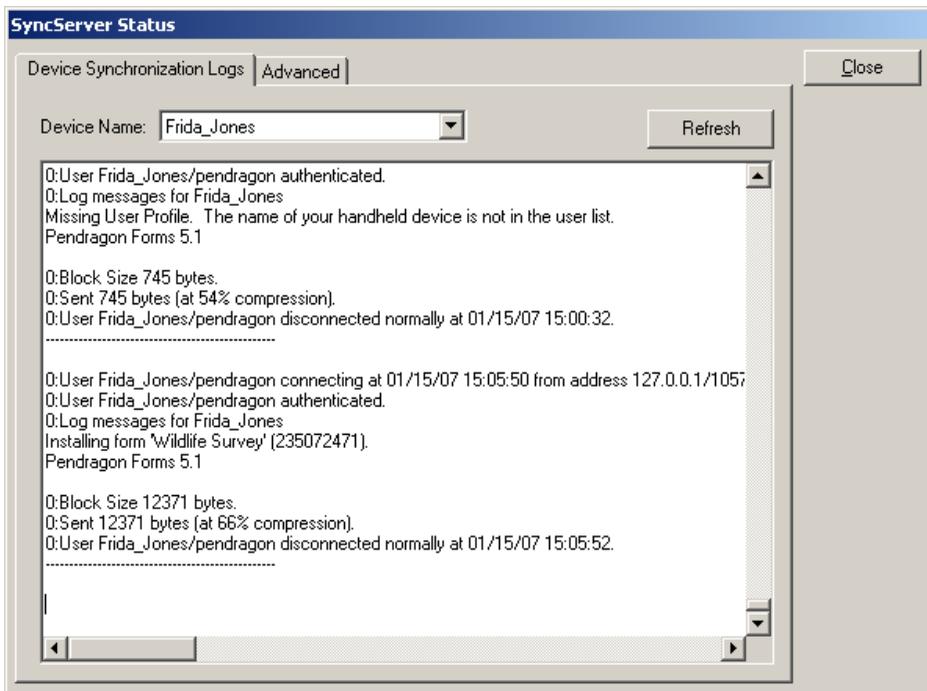
Putting a form design on a handheld device is only half of your data collection system. You need to make sure that records entered on the handheld can synchronize successfully to the Forms Manager database (or an external database) on the PC, before you deploy your application.

During synchronization, Pendragon Forms 5.1 uses Pendragon SyncServer technology to send data from the handheld to the PC and vice versa.

If Pendragon Forms encounters a problem that prevents data from going to or from the handheld during synchronization, an error message will be generated in the SyncServer Log. You will need to investigate and solve any synchronization problems before deploying your application.

To view the SyncServer synchronization log, click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools.

Refer to page 545 for information on synchronization error messages.

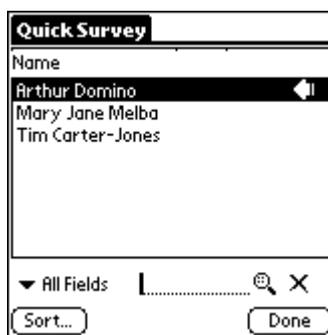


Detecting problem records on the handheld

After synchronization, if there are any records on the handheld that could not be uploaded to the PC, these records will be marked with an arrow on the Review screen of the handheld.

To look at the records on the handheld, tap the name of the form on the handheld, then tap the Review button. The records that could not be uploaded to the PC will be marked with an arrow.

Check the synchronization Log to determine why the record could not be uploaded to the PC. Do not collect any more data on the handheld until the problem is resolved.



How Often Should Users Synchronize?

It is recommended that users synchronize at least once a day if they are creating or modifying records on the handheld. The longer that they wait to synchronize, the higher the risk of losing data due to a mishap with the handheld device.

Pendragon Forms is stored in RAM on the handheld device, which means that losing battery power or accidentally dropping the device would cause a loss of your form designs and data. By synchronizing at least once a day, you will reduce the amount of data that is on the handheld but has not yet been backed up to the PC. Losing the handheld would mean recovering one day's worth of data collection, not weeks'.

If you cannot risk losing a full day's worth of records, consider synchronizing more than once per day. If you are going to be in an environment where you cannot synchronize once per day, consider using a handheld device that supports backing up data onto external media. See pages 512-514.

Who is responsible for making backups?

The Pendragon Forms Manager database on the PC is where all of your form designs and data are stored. If you lose everything on your handheld device and you still have the database on the PC, you can recover all of your form designs, and all data up to the last time you synchronized.

Warning: The handheld is NOT a backup for the data on the PC!

If you lose the Pendragon Forms Manager database on the PC but you still have Pendragon Forms on the handheld, you will NOT be able to synchronize your forms just by re-installing Pendragon Forms on the PC. This is because each form design has a unique Form ID#, and there needs to be a match between the Form ID# of the form on the handheld, and the Form ID# of a form design in the database in order to synchronize.

As part of your data collection process, you need to backup the Pendragon Forms Manager database regularly, and even daily. If you are going to rely on electronic forms instead of paper forms, you need to take steps to protect your only copy of your data.

In version 5.1, the Pendragon Forms Manager database is named FORMS32K.MDB.

The Forms Manager database is typically stored in the C:\Program Files\Forms3 folder.

Make a backup copy of the Forms database regularly, even daily, and store the backup on a zip disk or CD-ROM that is separate from the hard drive where Forms is installed.

Refer to page 510 for additional options for making backups of your form designs and data. See page 511 for information on recovering form designs. See pages 512-514 for information on performing backups on the handheld using external media.

4. Managing Form Designs

Your form designs may need to change over time. When you first create a form and send it to the handheld to test, you may discover that you need to change the form design in one way or another in order to make the form more user-friendly for the handheld users.

This chapter looks at the options available for changing form designs and organizing form designs within the Forms Manager.

Making Changes to a Frozen Form

After a form design has been frozen, there are some elements of the form design that can be changed, and some that cannot.

Items that can be changed after a form has been frozen include:

- Form names.
- Field names.
- Selection of which Lookup List to reference in a Lookup List field.
- Selection of which form name to reference in a Subform or Single Subform field.
- Some Advanced Field Properties. (Unavailable field properties will be grayed out.)
- Form Properties and Advanced Form Properties.
- Scripts.
- Selection of which form name to reference in a Lookup List field.

If you make a change to a form, the form must be re-distributed and sent to the handheld in order for the changes to take effect. To re-distribute a form, click on the name of the form in the Forms Manager, then click the Distribute button. Then synchronize the handheld to receive the changes.

Things that cannot be changed after a form has been frozen include:

- Adding or deleting fields from the form.
- Changing the items in a Popup List or MultiSelection List.
- Changing the field type used in a given field.

If you need to change one of these, you can copy the form design and modify the copy.

Copying a Form Design

You may need to copy a form design if:

- a. You want to use one form as the starting point for the design of a different form.
- b. You need to modify a form which is frozen. If you copy a form which is frozen, the copy will not be frozen, and can be modified as necessary. When you make a copy of a form, only the form design is copied, not the data associated with the original form.

To Copy a Form:

1. Click on the name of the form, then click the Copy button. You will be asked to confirm that you want to make a new copy of the selected form. Click Yes.
2. The name of the copy will be the name of the original form plus an asterisk (*) at the end of the name. Click on the Edit button to change the name of the new form, as well as to change, add or delete fields on the form.
3. When the copy is created, the database column names will remain the same as in the original form design. If you need to import the data from the original form into the copied form, you can do so by freezing the copy, then clicking on the Import button (see page 239).

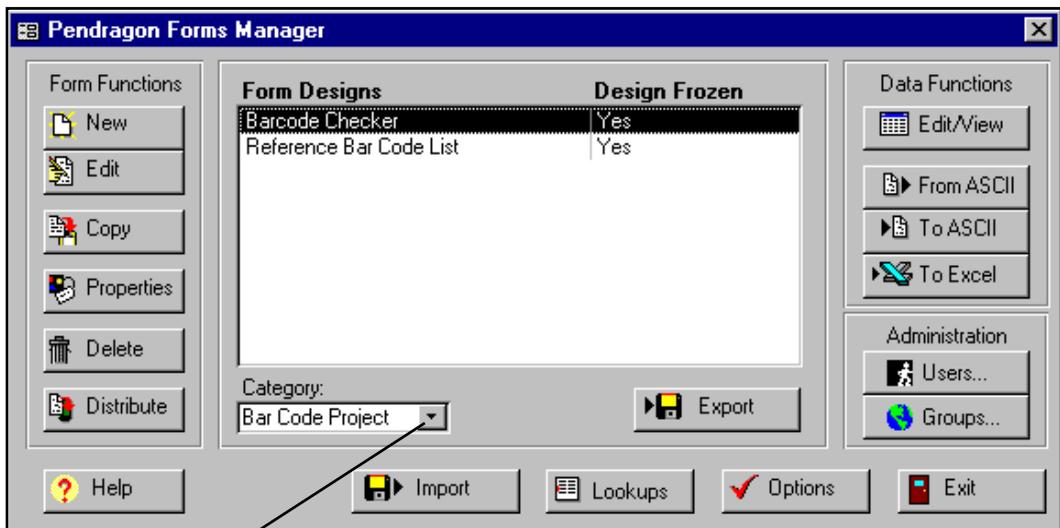
If you encounter any problems when trying to freeze a copied form, see Appendix B, *Troubleshooting*, page 531.

Tip:

Avoid having multiple form designs with exactly the same name. If you are copying a form to modify an existing form, and you plan to use the same form name, rename your old form immediately to avoid confusion. Alternatively, consider using version numbers in your form designs, e.g.: Inspection v1.0, Inspection v2.0 etc.

Organizing Form Designs into Categories

If you are working on a project that involves several form designs, it may be useful to group the forms into one category for ease of use.



To view the form designs in a given category, select the category here.

To assign a form design to a category:

1. Click the name of a form, then click the Properties window. (See picture on next page.)
2. In the Category field, type the name of a new category. Or, if you have already created a category, click the arrow to select that category from the list.

If you are viewing a category and then you click the New button to create a new form design, the new form will not appear in the list until you assign the form to the category.

The category **All** allows you to view all form designs that you have created, regardless of category.

The **Samples** category is a special category containing Pendragon Forms sample form designs. Form designs in the Samples category do not appear in the list when you choose the All category.

Either type a new category or select from a list of existing categories.

The screenshot shows the 'Form Properties' dialog box with the following details:

- Identification:**
 - Form: Reference Bar Code List
 - ID: 136577676
 - Table Name: FORM_ID_136577676
 - ASCII File: 824028C.OUT
 - Query Name: QRY Reference Bar Code List
 - Category: Bar Code Project (selected from a dropdown menu)
 - Field Map
- Data Persistence:**
 - Keep a copy of records on PalmPilot
 - Keep new records on PalmPilot for 0 days.
 - Keep incomplete records on PalmPilot
- Access Rights:**
 - No additions on PalmPilot
 - No updates on PalmPilot
 - No deletions on PalmPilot
- Freeze Form Design:**
 - Freeze Form design for distribution to PalmPilot and create database.
- Buttons:** Help, Field Mappings..., Advanced Properties..., OK

Recycle Bin

The Recycle Bin is a special category. When you delete a form design from the PC, the form is re-assigned to the Recycle Bin category. If the form design is still on the handheld, the next time you synchronize, data will still be sent to the PC and then the form will be removed from the handheld.

If you need to retrieve a form design from the Recycle Bin, view the Recycle Bin category, then click the name of the form and click the Properties button. Assign the form to a different category (for example the Unfiled category).

A form on the handheld will not synchronize unless the Form ID# on the handheld matches the Form ID# in the database. The Recycle Bin provides some protection from accidentally deleting a form from the PC that is still being used on the handheld. Only if you are certain that a form design is not on the handheld should you delete form designs from the Recycle Bin category. To delete a form and its data from the Recycle Bin, view the Recycle Bin category, then click the name of the form and click the Delete button in the Forms Manager.

Printing a Form Design

As you design a form, you may want to print the form design so that you can see the order of the fields on the form.

To Print a Form Design:

1. Click on the name of the form.
2. Click the File menu. Select a report option:
 - Form Design Report displays 9 fields per page.
 - Detailed Form Design Report displays 2 fields per page and includes scripts. If you are using this option, you may want to set your printer properties to print 2 pages per paper page, so that you reduce the amount of paper that is used.
 - Screen Preview Report displays 4 fields per page and shows what the handheld screen will look like in Field View (displaying one record at a time.)
3. A print preview window will appear. You can click the < and > arrows at the bottom of the window to move from page to page. Press CTRL P to print the report.

Deleting a Form from the PC

In order to synchronize, the Form ID# of a form on the handheld must exactly match the Form ID# of a form in the Pendragon Forms Manager database. If you permanently delete a form from the database on the PC, you will lose your ability to synchronize that form, even if the form is still on the handheld.

Before deleting any form designs, refer to *How to Delete Forms*, in the chapter on Synchronization Rules, page 50.

If you delete a form design from the PC by accident, refer to the Recycle Bin, page 41.

5. Synchronization Rules

Pendragon Forms 5.1 has bidirectional synchronization, meaning that during synchronization, records that are created on the handheld will be sent to the PC, and records that are entered on the PC will be sent to the handheld. The program is designed to be centrally managed on the PC, which means that the selection of which forms and data go to which handheld is controlled on the PC.

The Synchronization Process

During synchronization, Pendragon SyncServer executes the following sequence:

1. All form designs, records and Lookup Lists are uploaded from the handheld into the desktop's memory.
2. The Forms32k.mdb database file is opened and the handheld user is verified as an active user, authorized to synchronize Pendragon Forms.
3. For each form which should be synchronized with the handheld:
 - The database on the desktop is opened. (May be an external database.)
 - Changes are uploaded to the database.
New records from the handheld are inserted into the database.
Changed records from the handheld will update existing database records.
(If a Changed record on the handheld cannot find an existing record on the desktop, an insert into the database will be performed.)
 - All records on the handheld are marked for deletion, except records which were unable to be placed into the database.
 - If the form design is New or has been modified and re-distributed, it is marked for installation to the handheld.
 - A query is run to determine which records should be sent to the handheld.
Records are selected based on form properties and download criteria.
 - Forms which are to be deleted, along with their records, are marked for deletion on the handheld.
4. The Last Sync date is updated in the User List on the desktop.
5. Deleted form designs and records are removed from the handheld. Form designs, records and Lookup Lists are sent from the desktop to the handheld.

The Synchronization Cache

In Forms 5.1, Pendragon SyncServer technology is built into Pendragon Forms to control the synchronization process. Pendragon SyncServer maintains a synchronization cache on the PC to keep track of the data stored on each handheld device. To make synchronization faster, Pendragon SyncServer only sends records to the handheld if the record is not already in the cache.

The cached copy of the handheld data is stored on the PC or server in the C:\Program Files\Forms3\Cache folder. A separate file is stored in this folder for each handheld device.

When a form is re-distributed, the cached data for that form is no longer used. This means that synchronization will be slower when a form is re-distributed, as all the records designated for each handheld are re-sent to the handheld. If a form is not re-distributed, then the caching mechanism speeds up the synchronization since records that are already on the device do not have to be re-sent from the PC.

The cache does not store the images, signatures or sketches that are attached to records. This means that changing only an image, signature or sketch on the desktop will not cause the change to show up on the handheld unless another field that is not an image, signature or sketch is also changed.

The cache does not affect new or changed records on the handheld. New or changed records are always sent from the handheld to the PC during synchronization.

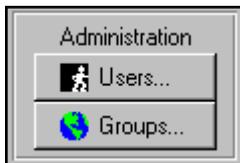
For further information on Pendragon SyncServer, see Appendix D, Pendragon SyncServer, starting page 560.

Users and User Groups

The Pendragon Forms Manager maintains a list of active handheld devices that are to be synchronized.

When the Pendragon Forms Manager is first opened on the PC or server, you are prompted to enter a handheld user name, and this user name is added to the User list.

If a multi-user license for Pendragon Forms is activated, additional users can be added by clicking on the **Users** button in the Pendragon Forms Manager.



Once a handheld unit has been listed in the Users list, the **Groups** button is used to associate handheld users with form designs.

Through the use of Users and User Groups, it is possible to centrally select a form to be sent to several handhelds. Each handheld in the group will receive the form whenever next that handheld synchronizes.

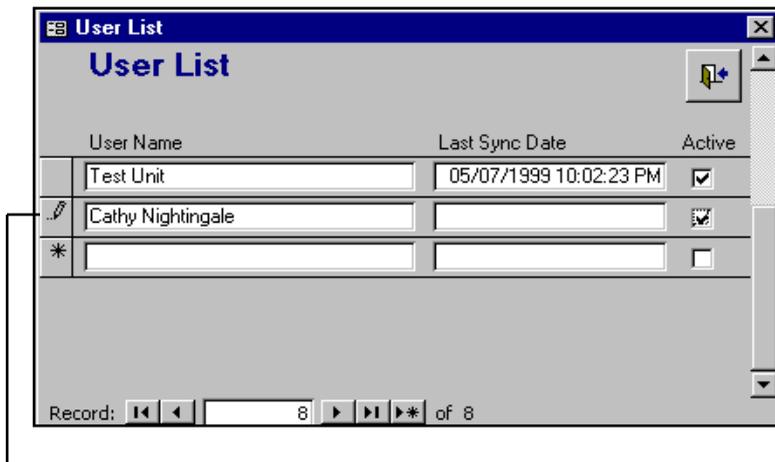
It is also possible to centrally select a form for removal from various handhelds, by removing the form from a User Group. The next time that the handhelds in that group synchronize, the form will be removed.

Details on Users and User Groups are explained on the pages which follow.

Adding Handheld Users to the User List

To add a user:

1. Type the handheld user name in the User Name field.
If you do not know the handheld user name of the device, see step 14 on page 5.
Use the same upper and lower case letters and spaces in the Pendragon Forms User List as you see on the handheld.
2. Check the Active box to make the user active. Close the User List window.
3. Add the user to a User Group - either the Default Group or another group. (See next page.)



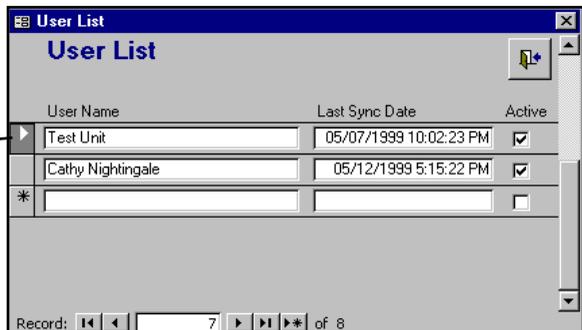
To save a record, click the pencil icon until it changes to a solid black triangle. Or you can press TAB until the cursor moves to the next row.

Deleting Users

If a handheld unit is no longer being used, you can either un-check the Active box to make the user inactive, or you can choose to delete the user from the list of users.

To delete a user:

1. Click to the left of a row to highlight that row.
2. Press the Delete button on the keyboard.



Creating User Groups

There are two special user groups:

- **Default Group**

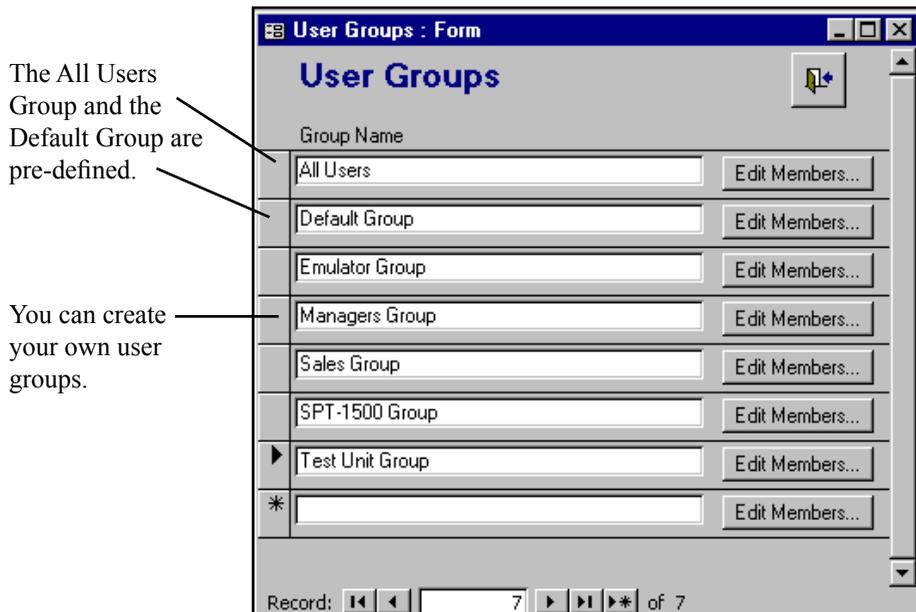
When you click the Distribute button to send a form to the handheld, if the form is not yet assigned to a user group, the form is added to the Default Group. Handheld users who are in the Default Group will receive the form the next time they synchronize.

If you are using Pendragon Forms in a single-user capacity, you do not need to create any other User Groups. When you first install Pendragon Forms, you are prompted to enter a handheld user name. This user name is automatically added to the Default Group.

If you are using Pendragon Forms in a multi-user capacity, you can add all of your users to the Default Group if everyone uses the same forms. Or you can create your own groups if you want to assign different forms to different groups of users.

- **All Users Group**

The All Users Group is not used at this time.



To create a new user group:

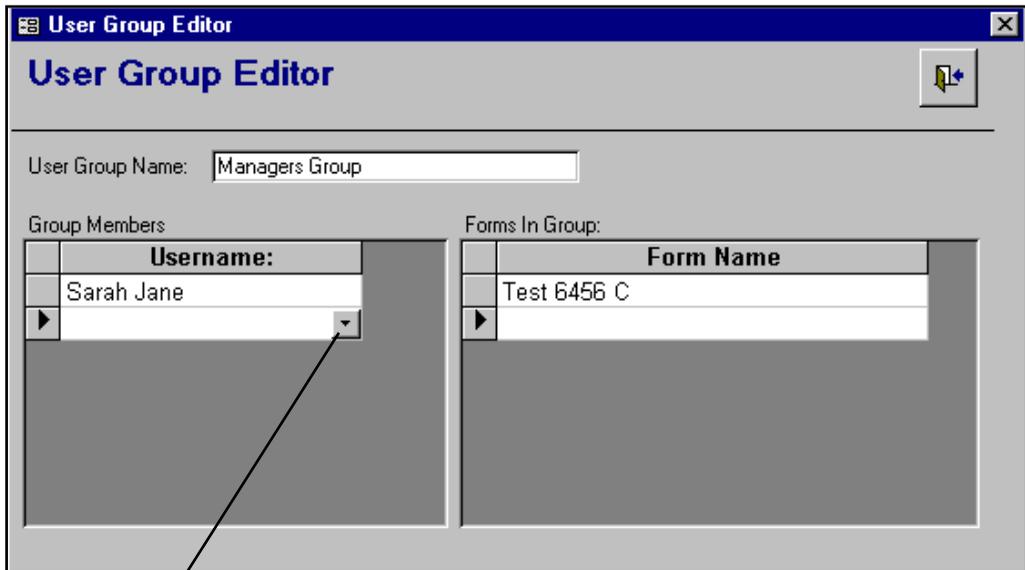
1. Click in an empty row and type a name for the user group.
2. Click the Edit Members button to select which users belong to the user group, and which forms will be sent to this group. (See next page.)

Adding users and forms to a User Group

Once you have created a User Group, you can select which users belong to this group, and which forms members of the group will receive.

To add users and forms to a User Group:

1. Click the Groups button in the Forms Manager, then click the Edit Members button for a selected group.



2. To add a user to this group, click the arrow in a blank row of the Username list to select a user. Only users who have been added to the User list can be selected for membership in a group.
3. To add a form for this group, click the arrow in a blank row of the Form Name list to select a form. Only forms which are frozen and have been distributed will appear in the list.

To delete a user or a form from a group:

1. Click in the gray cell to the left of the User Name or Form Name, to select that row.
2. Press the Delete key on the keyboard.

How User Groups affect Synchronization of Form Designs

During synchronization, Pendragon SyncServer checks the following:

Is the User listed as Active in the User List?

If a handheld unit is not listed as active in the User List, then Pendragon Forms will not synchronize during the synchronization.

Which form designs should be added to the handheld?

If a new form is distributed and added to a user group, then all handheld units in that user group will receive the new form.

Which form designs should be updated on the handheld?

If a form which is already in a user group is modified (for example, by changing field names or scripts), then the form will need to be re-distributed in order to be updated on the handheld. When the handheld synchronizes, the last synchronization date will be compared with the date of re-distribution of the form. If the last synchronization date is before the re-distribution date, then the updated form design will be sent to the handheld.

To re-distribute a form, click on the name of the form and then click the Distribute button.

Which form designs should be removed from the Handheld?

If a form is deleted from a user group, then when handheld users in that group synchronize their devices, any records in that form on the handheld are first sent to the database, and then both the form design and the records for that form are removed from the handheld.

WARNING: If a form is deleted from the database on the PC, handheld units will not have anywhere to send records, and consequently neither the form design nor the records for that form will be synchronized or removed from the handheld. If the form and its data on the handheld is not needed, the form can be manually deleted from the handheld. If the data is needed, it may be possible to retrieve the form design (see page 511) to retrieve the data. See page 50 for the proper procedure for deleting forms.

How to Delete Forms

The following is the proper sequence for deleting forms from the handheld:

1. In the Pendragon Forms Manager, click on the Groups button, then click the Edit Members button of the user group containing the form to be deleted.
 - The Default Group contains your form designs if you have not created any groups.
2. In the User Group Editor, click in the gray cell to the left of the form name, then press the Delete button on the keyboard to delete the form design from the group.
3. Wait until each handheld has had an opportunity to synchronize. During each synchronization, records for the form will be sent to the PC, and then both the form design and the associated records will be removed from the handheld.

NOTE: In a multi-user scenario it may take several days before each handheld user has had an opportunity to synchronize.

4. After each handheld has had an opportunity to synchronize, you can either:
 - Keep the form and its data in the Pendragon Forms database.
 - Delete both the form design and its data by clicking on the name of the form in the Forms Manager, then clicking the Delete button. Deleting the form design will re-assign the form to the Recycle Bin category (see page 41). You may want to wait a while before deleting the form design from the Recycle Bin, in the event that there is a handheld user who is still entering data in the form.

Troubleshooting Tips on Deleting forms

- If you delete a form manually from the handheld and the form returns after synchronization, you may want to check that the form has been removed from the Default user group and any other user groups to which your handheld unit belongs.
- If you delete a form from the PC by clicking the Delete button in the Forms Manager, but you still have new records on the handheld, the records will be able to synchronize to the PC only as long as the form design remains in the Recycle Bin (see page 41). If you delete a form from the Recycle Bin, you will have to attempt to retrieve the form design first before you can retrieve the data from the handheld. See page 511.

Limiting the number of records on the Handheld

The database on the PC can store many more records than the handheld. During the synchronization process, the database is queried to determine which records should be sent to the handheld. Form Properties and Advanced Form Properties offer several ways to limit the number of records which are kept on the handheld, as described here.

Data Persistence

Data Persistence options on the Form Properties window are the easiest way to limit the number of records that stay on the handheld. You can choose to:

- Remove all records from the handheld during synchronization.
- Keep a copy of records on the handheld.
- Keep records on the handheld for a certain number of days.
- Keep incomplete records on the handheld. (Requires a Completion Checkbox field on the form.)

See page 198 for information on how to set Data Persistence options.

The screenshot shows the 'Form Properties' dialog box with the following fields and options:

- Identification:**
 - Form: Customer Order
 - ID: 234712307
 - Table Name: (empty)
 - ASCII File: DFD6CF3.OUT
 - Query Name: (empty)
 - Category: Unfiled
 - Field Map: (unchecked)
- Data Persistence:**
 - Keep a copy of records on handheld
 - Keep new records on handheld for 0 days.
 - Keep incomplete records on handheld
- Access Rights:**
 - No additions on handheld
 - No updates on handheld
 - No deletions on handheld
- Freeze Form Design:**
 - Freeze Form design for distribution to handheld and create database.

Buttons at the bottom: Help, Field Mappings..., Advanced Properties..., OK.

The Advanced Form Properties window has more options for limiting the number of records on the handheld:

Storage Limit

There is a built-in storage limit of 500KB for each form design, meaning that up to 500KB of records for a given form will be sent to the handheld. If you need to keep a large number of records on the handheld, you can increase the Storage Limit - see page 222.

Archive Date

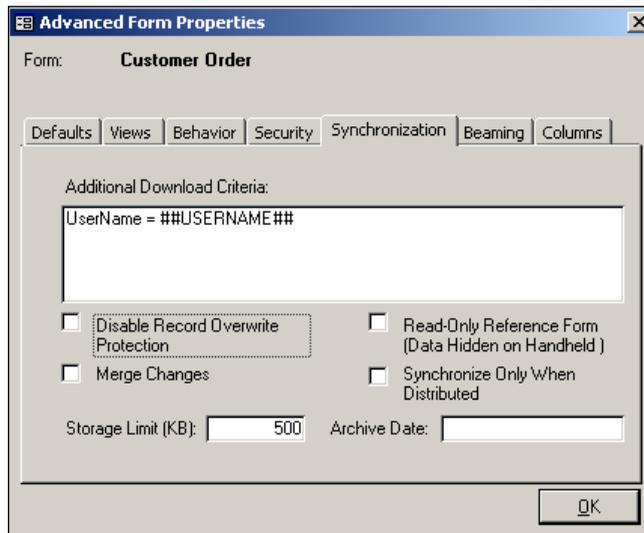
If you do not want to receive the earliest records on the handheld, you can periodically set an Archive Date. Records before the Archive Date will not be placed on the handheld. See page 222 for information on how to set an Archive Date.

Additional Download Criteria

Additional Download Criteria lets you specify criteria for selecting which records are sent to the handheld. The default of `UserName = ##USERNAME##` means that records must be assigned to a handheld user to be sent to that user's handheld.

If you are linking to an external database, you will need to specify Additional Download Criteria to choose which records from your database are sent to the handheld.

See page 214 for more information on Additional Download Criteria.



Switching Off Synchronizing Forms

PALM OS

On a Palm OS device, if you don't want to synchronize Pendragon Forms every time you synchronize the handheld, you can use the HotSync Manager program to switch on or off synchronizing Pendragon Forms.

To switch on or off synchronizing Forms:

1. Click the HotSync Manager icon (red & blue arrows icon in the Windows task tray), and choose Custom.
2. In the Custom window, select your handheld user name.
3. In the list of handheld applications, click on Pendragon Forms to select it.
4. Click the Change button.
 - Select the Do Nothing option to turn off synchronizing Forms on the next HotSync data transfer only.
 - Check the Set as Default box if you want to switch off synchronizing Forms indefinitely.
 - Select the Synchronize option to return to synchronizing Forms.

IMPORTANT:

In order to backup data that is on the handheld, you must set Forms to synchronize.

PPC

On a Pocket PC device, you can set Pendragon Forms to automatically synchronize when the handheld is in the cradle, or you can set Pendragon Forms to synchronize when you manually initiate synchronization. (See pages 8-10).

To switch off synchronizing Pendragon Forms on a Pocket PC, set the synchronization to manual.

IMPORTANT:

In order to backup data that is on the handheld, you must set Forms to synchronize automatically or you must do a manual synchronization.

Switching off Synchronizing Only One Form Design

If you have a form that is for reference purposes only, that handheld users do not add to or update, you can choose to set the form to Synchronize Only When Distributed. This speeds up synchronization, as the form will be ignored during synchronization unless the form has been re-distributed. For information on setting this Advanced Form Property, see page 222.

IMPORTANT:

If users are adding records or modifying records of a form, you should not switch on Synchronize Only When Distributed. Otherwise, records on the handheld will not be backed up.

6. Using Forms on the Handheld

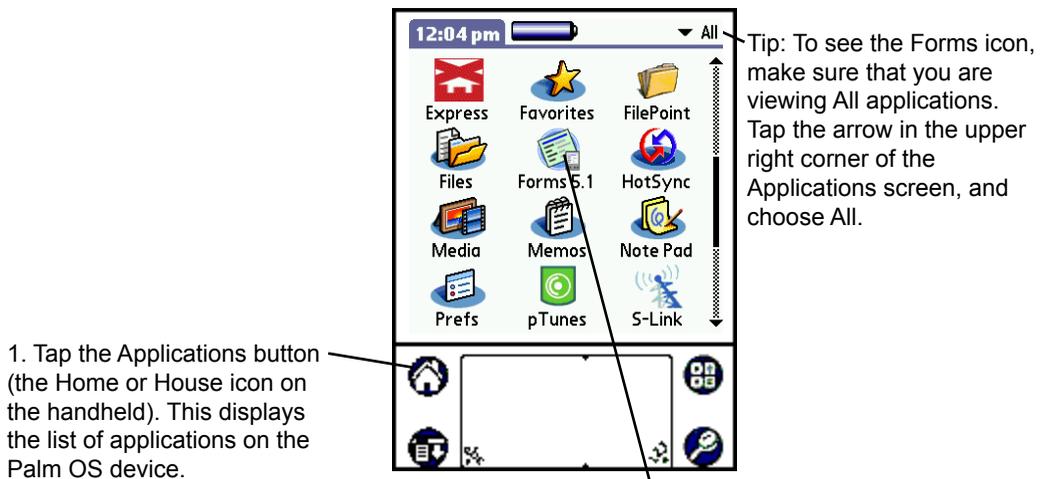
Installing Pendragon Forms on the Handheld

Before you can run Pendragon Forms on the handheld, the program must first be installed on the handheld. After installation, there will be a Forms 5.1 icon on the handheld. Tap the Forms icon to run Pendragon Forms.

PALM OS

To install Pendragon Forms on a Palm OS device, start on the PC and click Start...Programs...Pendragon Forms 5.1...Install Forms 5.0 on Palm OS device. Select the name of the Palm OS device on which you want to install Pendragon Forms, then click Done. Synchronize the handheld once to install the program, and synchronize a second time to receive the form designs for that device.

To run the Pendragon Forms program, tap the Forms 5.1 icon.



1. Tap the Applications button (the Home or House icon on the handheld). This displays the list of applications on the Palm OS device.

2. Tap the Forms 5.1 icon to run the Pendragon Forms program on the Palm OS device.

PPC

To install Pendragon Forms on a Pocket PC device, start on the PC and click Start...Programs...Pendragon Forms 5.1...Install Forms 5.1 on Pocket PC device. The application will be installed on the next Pocket PC device that you synchronize. Synchronize the handheld once to install the program, and follow the prompts on the handheld after synchronization.

See the instructions on pages 8-9 for instructions on setting up the Pocket PC device to synchronize Pendragon Forms.

To run Pendragon Forms on the handheld, tap Start...Programs.. and then tap the Forms 5.1 icon.

1. Tap the Start button on the Pocket PC device.

2. Tap Programs to view the programs installed on the device.

3. Tap the Forms 5.1 icon to run Pendragon Forms.

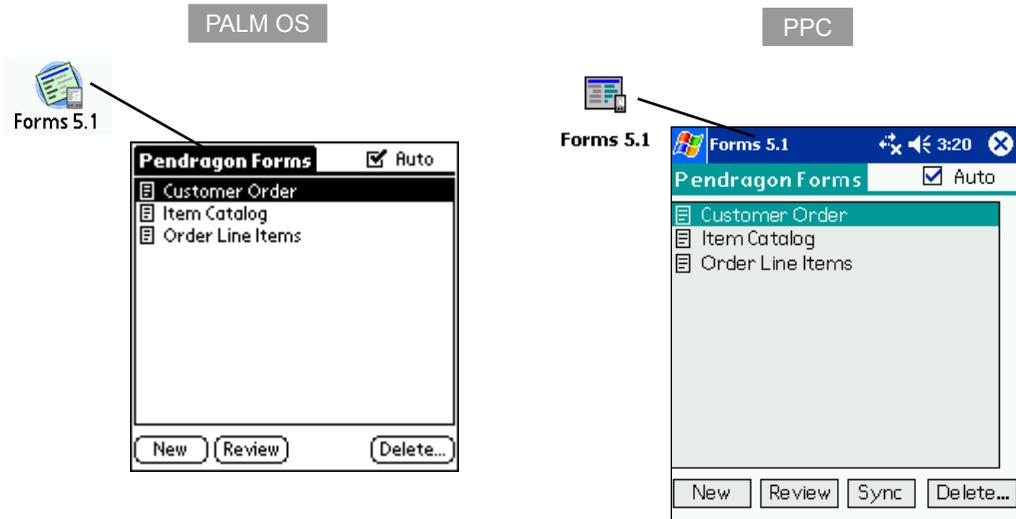
Pocket PC devices may also provide shortcuts to programs that you have recently run. If you have run Pendragon Forms recently, you can tap the Start button and there will either be a Forms shortcut icon, or Forms will be listed in the Start Menu as a recently used program. Tap on the Forms icon or Forms in the menu to run Pendragon Forms.



This is the Forms shortcut icon.

Pendragon Forms Main Menu (Forms List)

When you tap the Forms icon, a list of forms that have been sent to the handheld is displayed. This screen is called the Forms List, and is the Main Menu screen for Pendragon Forms.



Entering New Records

Each instance of filling in a form creates a record.

To create a new record on the handheld:

1. Tap the name of a form.
2. Tap the **New** button.

By default all forms are displayed in Layout View, which displays several fields per screen. The layout of each screen is set by the person who designed the form on the PC.

For backward compatibility with earlier versions of Pendragon Forms, Palm OS and Pocket PC devices can also display forms in Field View (one field per screen) and Record View (eleven fields per screen in a two-column format). Only older form designs should use Field View and Record View. To set which view is used on the handheld, see Advanced Form Properties, page 202.

Layout View

Layout View is the default for creating new records and reviewing existing records. Layout View allows the designer of the form to decide how many fields are displayed per screen and how much space each field occupies on the screen. Layouts can be changed even after the form is frozen, as long as the form is re-distributed to the handheld.

Handheld users can tap in the answer area of a field to enter the answer for that particular field.

The Next and Previous Buttons at the bottom of the screen take the user to the next or previous screen of the form. The End button saves and exits the record.

PALM OS

PPC

Customer Order

Customer Name
Grover's Ice Cream

Customer Account Number
63124

Date of Order [4/26/03 3:45 pm]

Customer Purchase Order Number
003156

Phone Number
847-555-7895

Fax Number
847-555-7995

[End] [Previous] [Next]

Forms 5.1 2:24

Customer Order

Customer Name
Grover's Ice Cream

Customer Account Number
63124

Date of Order [4/26/03 3:45 pm]

Customer Purchase Order Number
003156

Phone Number
847-555-7895

Fax Number
847-555-7995

[End] [Previous] [Next]

End	Previous	Next												
123	1	2	3	4	5	6	7	8	9	0	=	↵		
Tab	q	w	e	r	t	y	u	i	o	p	[]		
CAP	a	s	d	f	g	h	j	k	l	;	'	↵		
Shift	z	x	c	v	b	n	m	,	.	/	↵	↵		
Ctl	á	ü	`	\							↓	↑	←	→

To enter data, tap in the answer portion of a field and use Graffiti or the handheld keyboard to write in the field.

Tap the End button to save and exit the record.

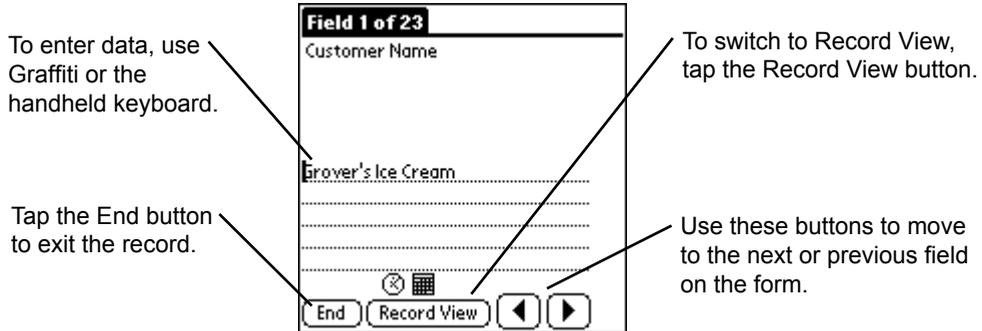
Tap the Next button to move to the next screen of the form. Tap the Previous button to move to the previous screen.

On Pocket PC devices, you can tap the keyboard icon to pop up or to hide the onscreen keyboard.

When the onscreen keyboard is up, the title bar of the application is hidden. To re-display the title bar, tap the keyboard icon to hide the keyboard.

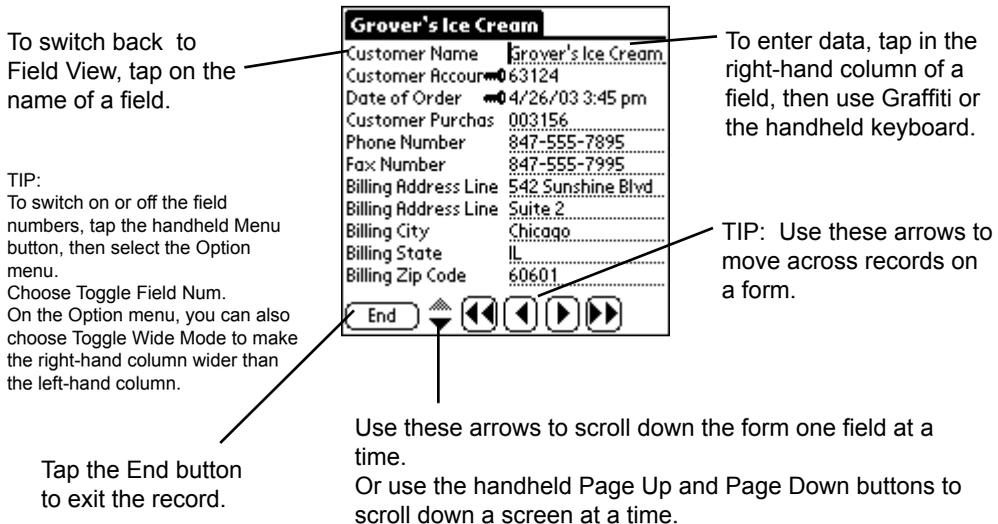
Field View

For backward compatibility with early versions of Pendragon Forms, form designs can also be displayed in Field View. Field View displays only one field at a time on screen. Right and Left arrow buttons allow the user to move to the next or previous field on the form.



Record View

For backward compatibility with early versions of Pendragon Forms, forms can also be displayed in Record View. Record View is a two-column format that displays field names in the left-hand column and responses in the right-hand column. An entire field name may not be visible. The handheld user can tap in the right-hand column to enter a response for a given field.

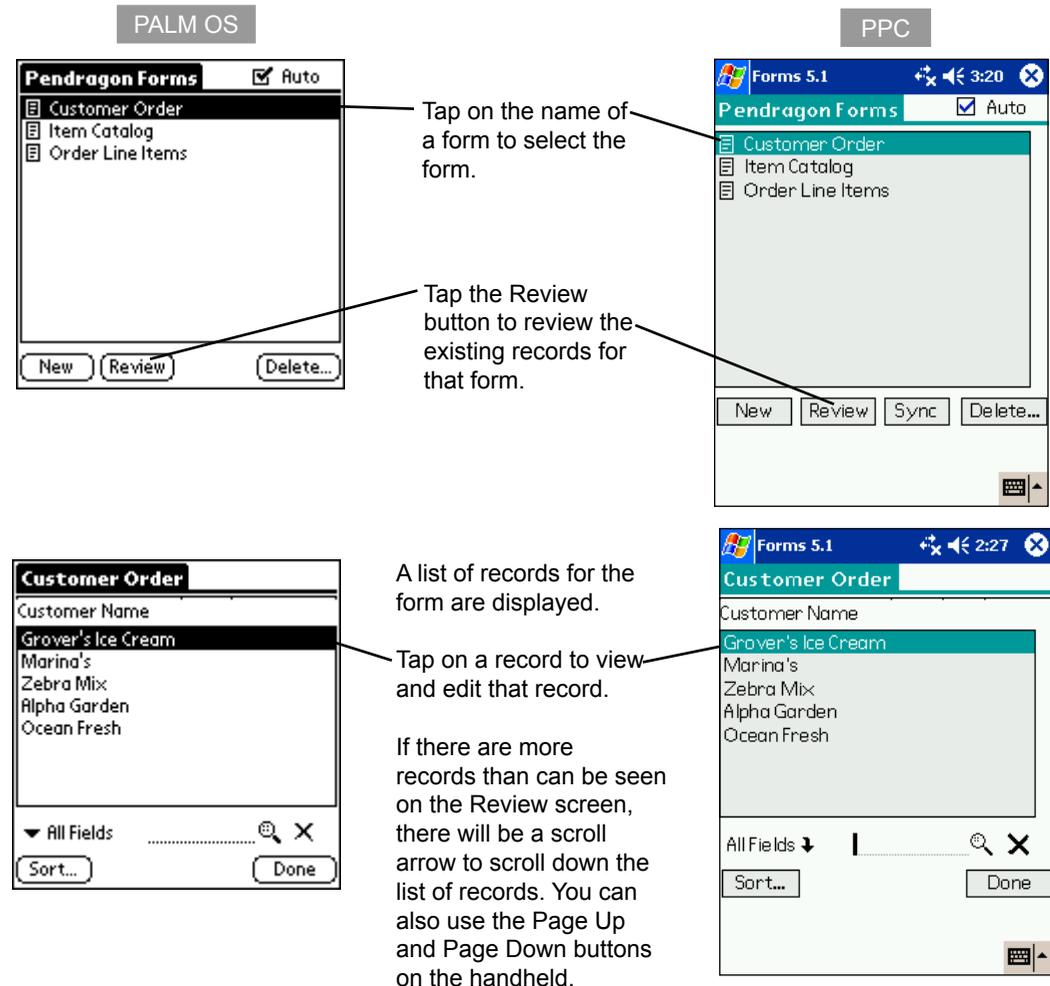


Reviewing Records on the Handheld

To modify an existing record, you need to review the records on the handheld and select a record to edit. Records created on the PC and sent to the handheld are considered existing records.

To review records on the handheld:

1. Tap on the name of a form to highlight it.
2. Tap the Review button. A list of existing records for that form will be displayed. The Display Key field (typically the first field) of each record is shown on screen.
3. Tap on a record to select that record and make changes.



Displaying Several Fields on the Review Screen

Sometimes it is easier to select the correct record if the handheld user can see several fields on the Review screen.

Handheld users can select additional fields to display on the Review screen, as shown here. Alternatively, the person designing the form can set an Advanced Form Property to pre-select up to three fields to display on the Review screen - see page 226.

PALM OS

Tap in the blank column heading of the 2nd column position.

Customer Order

Customer Name	
Grover's Ice Cream	
Marina's	
Zebra Mix	
Alpha Garden	
Ocean Fresh	

▼ All Fields [Search] [X]
Sort... Done

Select a second field to view.

Customer Order

Customer	Fax Number
Grover's	Billing Address Line 1
Marina's	Billing Address Line 2
Zebra Mix	Billing City
Alpha Gar	Billing State
Ocean Fre	Billing Zip Code

▼ All Fields [Search] [X]
Sort... Done

Customer Order

Customer Name	Billing Address Line
Grover's Ice Cream	542 Sunshine Blv
Marina's	24 Sweet Avenue
Zebra Mix	500 Zoo Parkwa
Alpha Garden	2 Alpha Base
Ocean Fresh	67 Pier 5

▼ All Fields [Search] [X]
Sort... Done

PPC

Tap in the blank column heading of the 2nd column position.

Forms 5.1 Customer Order

Customer Name	
Grover's Ice Cream	
Marina's	
Zebra Mix	
Alpha Garden	
Ocean Fresh	

All Fields [Search] [X]
Sort... Done

Select a second field to view.

Forms 5.1 Customer Order

Customer	Fax Number
Grover's	Billing Address Line 1
Marina's	Billing Address Line 2
Zebra Mix	Billing City
Alpha Gar	Billing State
Ocean Fre	Billing Zip Code

All Fields [Search] [X]
Sort... Done

Forms 5.1 Customer Order

Customer Name	Billing Address Line
Grover's Ice Cream	542 Sunshine Blv
Marina's	24 Sweet Avenue
Zebra Mix	500 Zoo Parkwa
Alpha Garden	2 Alpha Base
Ocean Fresh	67 Pier 5

All Fields [Search] [X]
Sort... Done

Handheld devices can display up to four columns on the Review screen.

PALM OS

Customer Na	Billing Add	Order Total
Grover's Ice Cr	542 Sunshil	\$15.49
Marina's	24 Sweet	\$23.80
Zebra Mix	500 Zoo P	\$25.90
Alpha Garden	2 Alpha Ba	\$71.15
Ocean Fresh	67 Pier 5	\$35.40

▼ All Fields | | 🔍 ✕
Sort... Done

PPC

Customer Na	Billing A	Order	Date of
Grover's Ice C	542 Sun	\$15.49	4/26
Marina's	24 Swee	\$23.80	5/18
Zebra Mix	500 Zoo	\$67.55	5/18
Alpha Garden	2 Alpha	\$71.15	5/11
Ocean Fresh	67 Pier 5	\$35.40	5/18

All Fields ↓ | | 🔍 ✕
Sort... Done

To display a third column, tap in the column heading of the 3rd column position and select a third field to view.

Similarly, tap in the 4th column heading position to select a fourth column to view.

You can re-size columns by dragging the column header divider lines, to make a column wider or smaller.

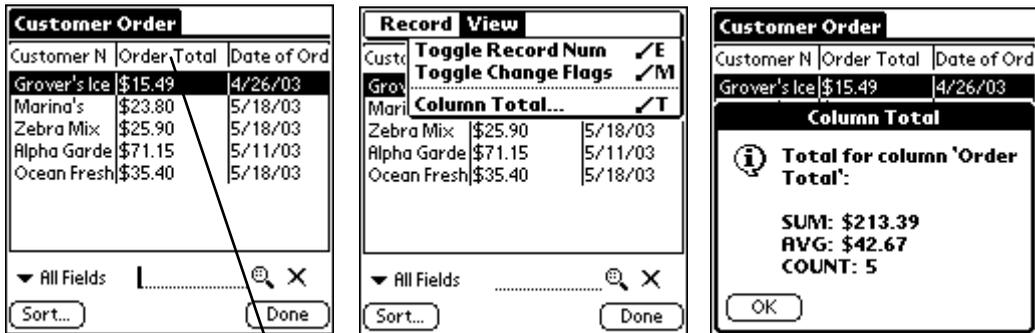
NOTE:
To revert to displaying one column only, make the 2nd, 3rd and 4th columns the same field as the 1st column.

Quick Sum and Average

From the list of records on the Review screen, there is a quick way to view the sum of all the records for one column of data. In addition to the sum of a column, the average (mean) value in that column and the number of records (Count) in the form are also displayed.

NOTE: The sum, average and count are across all currently filtered records.

PALM OS

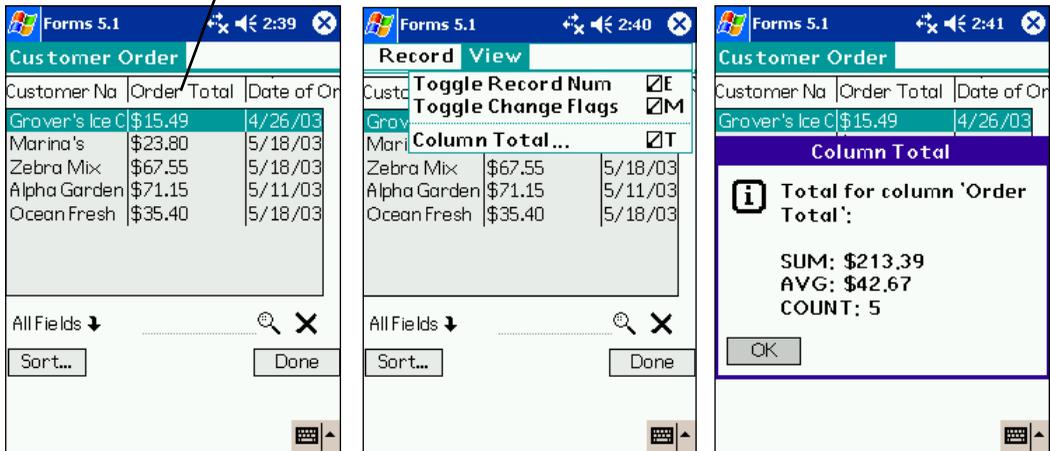


Display the column that you want to sum in the 2nd column position.

On Palm OS devices, tap the handheld Menu button (the drop-down menu icon below the Home icon).
 On Pocket PC devices, tap the title bar where the form name is displayed.
 On the View menu of both devices, select Column Total.

SUM = sum of all values in this column.
 AVG = Mean value.
 COUNT = Total number of records.

PPC

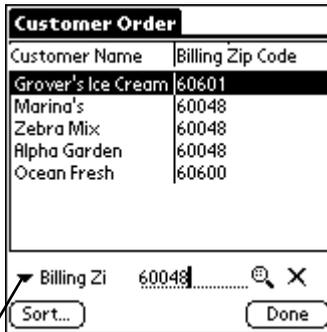


Filtering Records

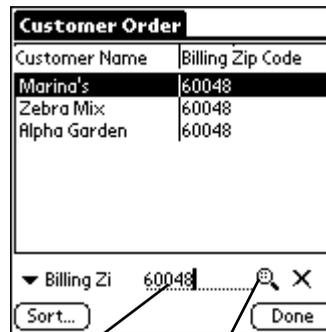
On the Review screen, the Filter feature allows you to display only those records that match the search criteria that you specify.

You can search in just one field, or across all fields on a form.

PALM OS



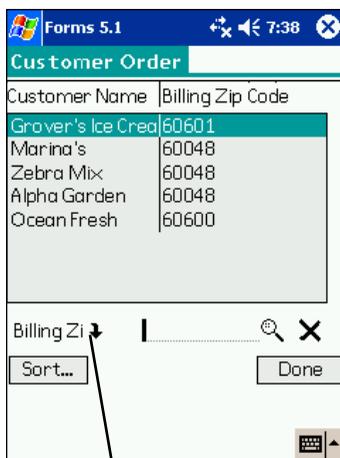
On a Palm OS device, tap the arrow underneath the list of records, and select a field that will be used to filter the records.



On the dotted line, type the search criteria for which you are looking.

Tap the Search icon (magnifying glass icon) to perform the search and filter the records. To switch off a filter, tap the X icon.

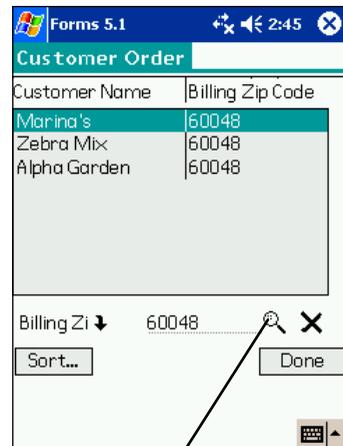
PPC



Tap here to select a field on which to filter.



Pop up the onscreen keyboard, and type a search criteria.



Tap the Search icon to perform the search. To switch off the filter, tap X.

Sorting Records

The Sort feature on the Review screen allows you to select a field to sort by. You can then choose to sort the records in this field in Ascending order (A..Z) or Descending order (Z..A).

The default sort order after synchronization is to display the records in the natural sort order from the PC. The natural sort order is in Ascending order by primary key.

If you have not selected a primary key, the TimeStamp (the creation date and time of the record) is used as part of the default primary key.

New records appear at the bottom of the list of records, and are placed in the correct sort order only after synchronization.

You can only sort by one field at a time.

PALM OS

Customer Order	
Customer Name	Billing Address Line
Grover's Ice Cream	542 Sunshine Bly
Marina's	24 Sweet Avenue
Zebra Mix	500 Zoo Parkwa
Alpha Garden	2 Alpha Base
Ocean Fresh	67 Pier 5

▼ All Fields | | 🔍 X

Sort... Done

Tap the Sort button to select a field to sort by.

Customer Order	
Customer Name	Billing Address Line
Grover's Ice Cream	542 Sunshine Bly
Marina's	24 Sweet Avenue

Sort Records

Sort Order: A->Z Z->A

Sort On: ▼ Customer Nam

OK Cancel

Select a Sort Order (A->Z) or (Z->A), and select a field to Sort On. Then tap OK.

Customer Order	
Customer Name	Billing Address Line
Alpha Garden	2 Alpha Base
Grover's Ice Cream	542 Sunshine Bly
Marina's	24 Sweet Avenue
Ocean Fresh	67 Pier 5
Zebra Mix	500 Zoo Parkwa

▼ All Fields | | 🔍 X

Sort... Done

In this example, records have been sorted from A to Z in a field called Customer Name.

NOTE 1:

To revert to sorting records in the order in which they were created, choose to sort on the TimeStamp field.

NOTE 2:

If sorting seems to take a long time - such as minutes - on a Palm OS device, then instead of sorting records on the handheld, you may want to sort records as they are sent from the PC - see Additional Download Criteria, page 214.

PPC



Tap the Sort button to select a field to sort by.



Select a Sort Order (A->Z) or (Z->A), and select a field to Sort On. Then tap OK.



In this example, records have been sorted from A to Z in a field called Customer Name.

NOTE 1:

To revert to sorting records in the order in which they were created, choose to sort on the TimeStamp field.

NOTE 2:

If sorting seems to take a long time - such as minutes - , then instead of sorting records on the handheld, you may want to sort records as they are sent from the PC - see Additional Download Criteria, page 214.

Deleting Records and Forms from the Handheld

Pendragon Forms is designed to be centrally managed from the PC.

- To delete a form and all its records from the handheld, remove the form from the User Group to which the handheld belongs, and then synchronize the handheld. See page 50.
- To delete records, but not the form design from the handheld, you need a criteria on the PC that determines which records to remove from the handheld. (See *Limiting the number of records on the Handheld*, page 51.) Without such criteria, records that are manually deleted from the handheld will re-appear after synchronization.

When should you delete records and forms manually from the handheld?

NOTE: Manually deleting a form or record from the handheld will not delete the form or record the desktop.

You may want to manually delete a record if:

- The record was sent from the PC and then incorrect information was entered on the handheld. If you manually delete the record and then synchronize, you will receive a fresh copy of the record from the PC.

You may want to manually delete a form if:

- The form was deleted from the desktop before it was removed from your handheld. If you do not need the data in the form, you can delete the form from the handheld. Note that when you delete a form design from the handheld, all the associated records for that form are also deleted.

PALM OS

To delete a record from a Palm OS handheld:



To delete a record on a Palm OS device, tap on a form to select it, then tap the Delete button.

A list of records for that form will be displayed. Tap on a record and then tap the Delete button.

Travel Planning Form		
Customer Nam	Purpo	Starting Date a
Edmond Richar	Annual	1/31/05 8:30
Frank Rywell	Vacati	3/15/05 9:00
Sally Desmond	Buying	1/1/05 3:08
Jimmy Wicket	Confer	1/1/05 4:00

▼ All Fields

Sort... Delete... Done

PALM OS

To delete a form from a Palm OS handheld:



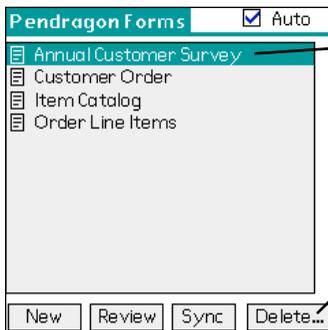
To delete a form from a Palm OS device, tap the handheld Menu button (the drop-down menu icon below the Home/House icon). On the Options menu, tap Delete Form Designs.



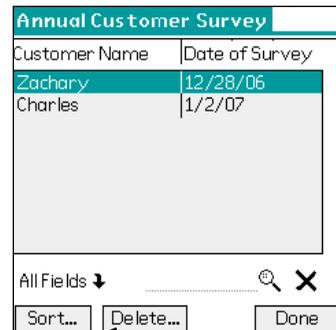
A list of form designs is displayed. Tap on the name of a form to delete, then tap the Delete button.

PPC

To delete a record from a Pocket PC handheld:



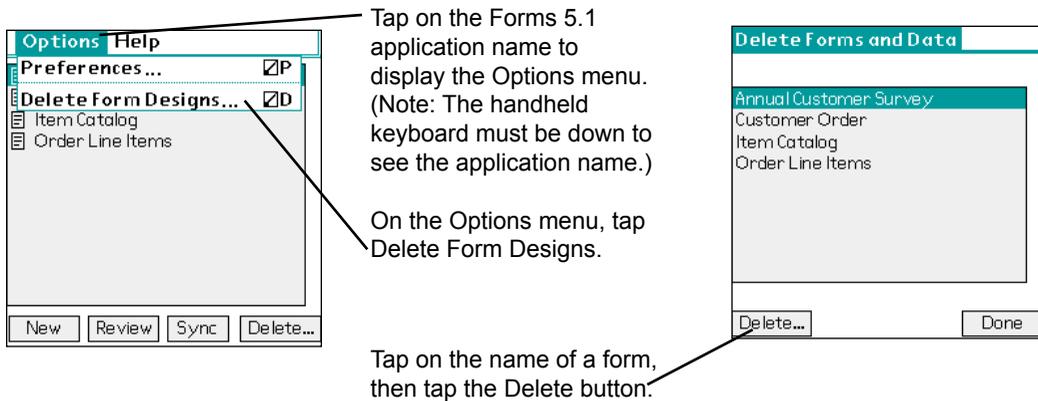
Tap on the name of a form, then tap the Delete button.



A list of records is displayed for that form. Tap on a record and then tap the Delete button.

PPC

To delete a form from a Pocket PC handheld:



Password-Protecting a Form Design

You can password-protect a form design so that the handheld user has to enter a password to access the form on the handheld. Once the password is entered, the handheld user can access the form for as long as Pendragon Forms is the active application and the handheld device remains powered up. If the user switches applications or if the handheld device is powered off, the user will need to enter the password the next time the form is accessed again.



Setting a password on a form design is an Advanced Form Property. See page 213 for details.

Warning: Password-protecting a form design does not encrypt the data within the form. This means that a person with advanced knowledge of the Palm OS or Pocket PC device may be able to access the data.

Setting Forms Preferences

The Forms Preferences screen sets global options that apply to all forms.

To access the Forms Preferences screen:

On Palm OS devices, tap the Forms icon, and then tap the handheld Menu button (the drop-down menu icon on the handheld). Tap the Options menu and then select the Preferences option.

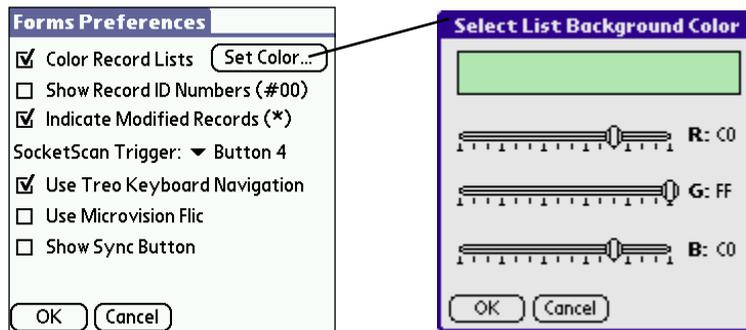
On Pocket PC devices, tap the Pendragon Forms title bar. On the Options menu, tap Preferences.

The Forms Preferences are:

- **Color Record Lists** PALM OS

This option applies to Palm OS devices only. The default is that this checkbox is checked. If you are using a monochrome device, this option does not apply, even when checked. On Color handheld devices, if Color Record Lists is switched on, then when the user reviews records for a form, the records will appear on lines of alternating green and white colors. This helps the user differentiate between records.

If you tap the Set Colors button, you can change the default color used on the review screen. The default setting is Red (R) = C0 (or CC), Green (G) = FF and Blue (B) = C0 (or CC). If you change the default, do not make the color too dark, or it will be difficult to read the text of the records that appear on that color.



- **Show Record ID Numbers**

This option is switched off by default. If this option is on, users will see a Record ID number next to each record when they review records. The Record ID numbers count records across all forms, not within a given form, and so end-users do not usually need this information. However, Record ID numbers are used by Pendragon Software technical support.

- **Indicate Modified Records (*)**

This option is switched on by default. When the user reviews records, any new or modified records will have an asterisk (*) next to the record. This helps to identify to the user which records will be uploaded to the PC on the next synchronization.

- **SocketScan Trigger** PALM OS

This option applies to Palm OS devices only. The SocketScan Trigger option sets which handheld button will be used to activate a Socket Communications Secure Digital Scan Card plugged into the SD slot of a Palm OS device. The default is Button 4, which is typically the right-most physical button on the handheld device.

On some handhelds, Button 4 may be the lower right of the physical buttons on the device. See page 384 for a way to set the trigger button via a script.

- **Use Treo Keyboard Navigation** PALM OS

This option only applies to Palm OS devices, and is switched on by default. The Treo Keyboard Navigation options allows the handheld user to navigate on the screen using the 5-way navigation control on Palm Z22, TX, Treo 600, Treo 650 and Trep 700p devices.

- **Use Microvision Flic** PPC

This option applies to Pocket PC devices only. By default, this option is switched off. When the Microvision Flic option is switched on, then whenever the Pocket PC user enters the Pendragon Forms application on the handheld, the user will be prompted to connect via Bluetooth to a Microvision Flic bar code scanner.

If only specific form(s) on the handheld use bar codes, it may be better to prompt for connection to a Microvision Flic device on a form by form basis, instead of setting a global form preference which applies to all forms. To enable an individual form to connect with a Microvision Flic scanner, add the word [FLIC] in uppercase letters and in square brackets to a field name on the form. See page 376 for details.

- **Show Sync Button**

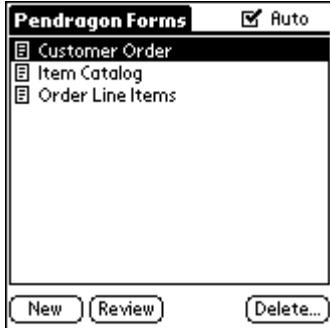
On Pocket PC devices, this option is switched ON by default. Pocket PC devices typically need to have a Sync button visible in the Forms Main Menu for end-users to tap to synchronize the handheld. If the Sync button is hidden, end-users will need to tap Start...Programs and then tap the S-Link application on the handheld in order to synchronize.

On Palm OS devices, this option is switched OFF by default. Palm OS devices that synchronize via cradle or cable can synchronize by pressing the synchronization button on the cable or cradle, and so these users do not need to have a Sync button. For Palm OS users who synchronize wirelessly, switching on Show Sync Button will provide a Sync button for users to access the S-Link application from the Forms application.

Using the AutoNavigate Feature

The AutoNavigate (Auto) checkbox appears in the upper right corner of the Forms Main Menu screen. AutoNavigate is switched on by default. To switch off AutoNavigate, un-check the Auto checkbox. To switch off AutoNavigate permanently for a given form, see the Advanced Form Property of Disable AutoNavigation, page 208.

AutoNavigate primarily affects Field View. When AutoNavigate is switched on, then in Field View, making a selection in certain fields automatically advances the user to the next field on the form. Popup Lists, Lookup Lists, Yes/No Checkboxes, Option 1 of 5, Date, Date & Time and bar code Text fields will advance you to the next field on the form if AutoNavigate is switched on.



If the next field on the form is a Popup List, Lookup List, Date or Date & Time field, then the list or calendar is automatically displayed if the record is a new record. If you review an existing record, selection fields will not automatically be displayed. This enables the existing contents of the field to be reviewed before the handheld user decides whether to make a change.

AutoNavigate has a limited effect in Layout View. If a selection field is the last field on a screen, making a selection will automatically advance the user to the next screen. AutoNavigate does not automatically pop up any lists or calendars in Layout View.

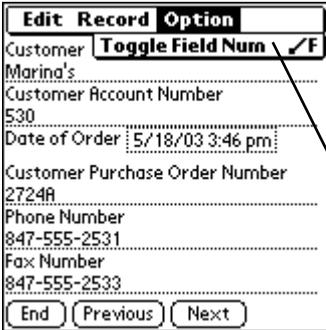
Switching on Field Numbers (for Debugging Purposes)

If you are in a record on the handheld, you can choose to switch on field numbers to help you debug a problem, particularly a scripting problem, on the handheld.

Field numbers will appear at the start of each field name. Note that in some fields, such as a field on which the question and answer are on the same line, adding the field numbers may cause a part of the field name to be out of view. This is why it is generally recommended that you switch on field numbers only when debugging a form. If you want field numbers permanently in your form, put numbers in the field names as you design the form. (Note, however, that database columns in Microsoft Access cannot begin with a number, so the database table that is created when you freeze the form will contain the letter A in front of any field that starts with a number. The handheld user will not see the added letter A.)

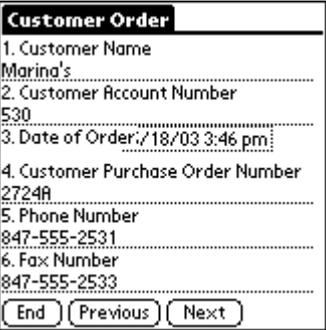
Field numbers can help you troubleshoot a scripting problem. You can look at the script in the Form Designer window on the PC, and then look on the handheld to see whether the script is referencing the correct field numbers.

PALM OS

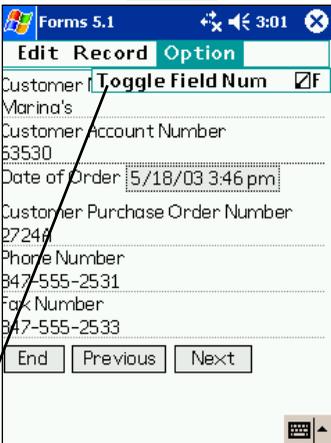


On Palm OS devices, tap the handheld Menu button (the drop-down menu icon below the Home/House icon on the handheld).

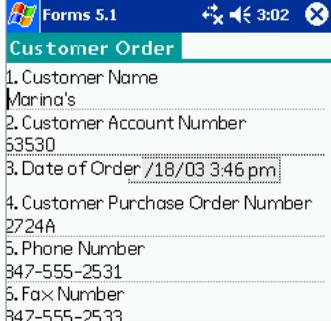
On the Option menu, tap Toggle Field Num to display field numbers.



PPC



On Pocket PC devices, hide the keyboard, then tap on the name of the form (the line just below the Windows title bar) to view the menu bar. On the Option menu, tap Toggle Field Num.



If you tap Toggle Field Num a second time, the field numbers will be switched off.

7. Field Types

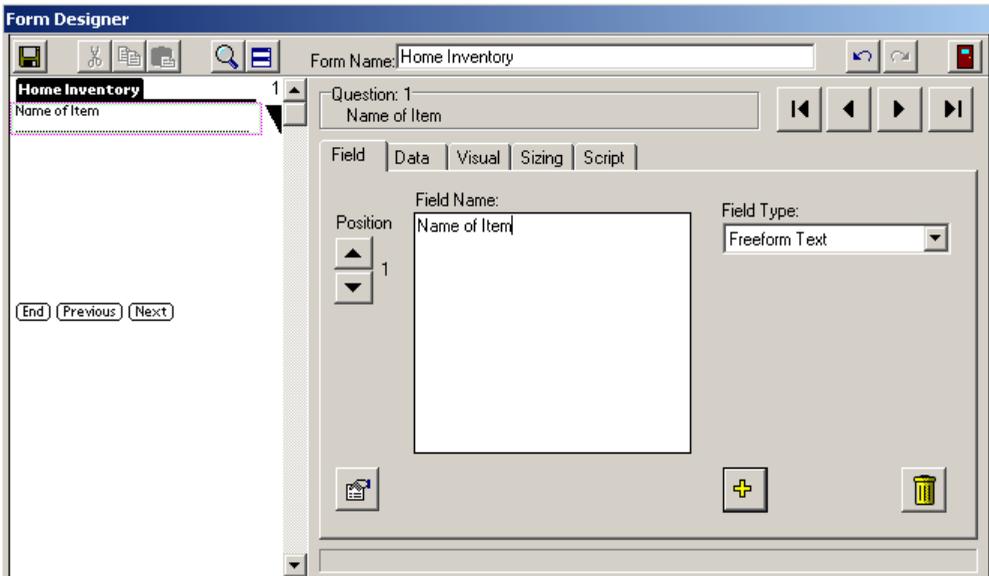
In the Pendragon Forms Manager, the Form Designer window is where you create your form design. To access the Form Designer window, either click the **New** button in the Forms Manager window to create a new form design, or select a form and click on the **Edit** button to edit an existing form design.

Every field on your form must have a field name and a field type.

- A field name can be approximately 255 characters in length. When viewing a form on the handheld in Record View, only the information from the first line or up to the first colon (:) on the first line will be displayed.
- A field type determines the kind of data that you want to allow the handheld user to input when entering records on a form.

This chapter looks at the 23 different field types that are available in Pendragon Forms.

For a quick overview of using the Form Designer, see Chapter 2, page 14. A detailed look at the Form Designer starts on page 165.



Available Field Types

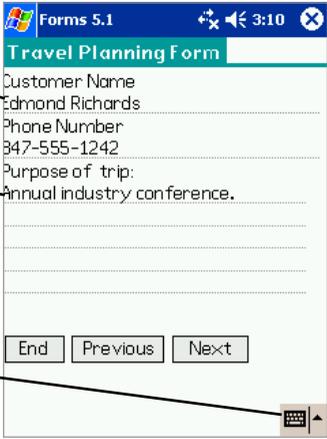
For each field or question on your form, you can select a field type. Field types let you choose the type of data that you want the handheld user to enter in a field. By limiting the type of data entry, you can improve the speed and accuracy of data entry on the handheld.

The following pictures illustrate how various field types look on Palm OS and Pocket PC devices. Each field type is described in more detail later in this chapter, starting on page 87.

PALM OS



PPC



Text field.

Text field with answer height of several lines.

On Pocket PC devices, popping up the onscreen keyboard will hide the title bar and the name of the form.

Text field.

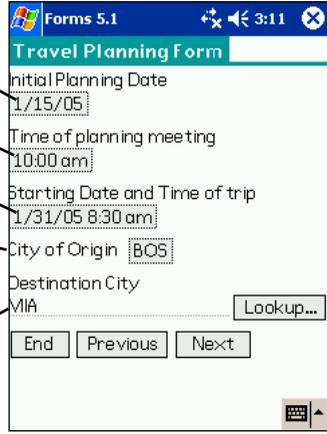
Text field with answer height of several lines.

On Pocket PC devices, popping up the onscreen keyboard will hide the title bar and the name of the form.

PALM OS



PPC



Date field.

Time field.

Date & Time field.

Lookup List with question and answer on same line.

Lookup List with answer on a separate line.

Date field.

Time field.

Date & Time field.

Lookup List with question and answer on same line.

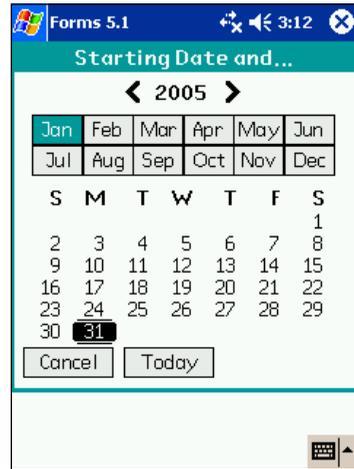
Lookup List with answer on a separate line.

PALM OS

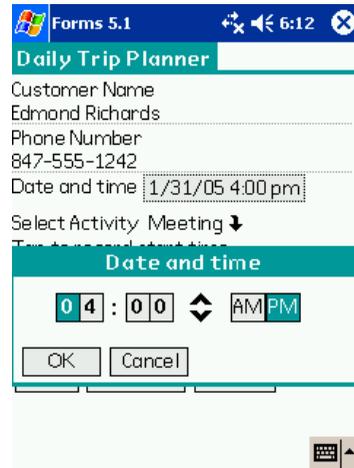


Tapping in a Date field or a Date & Time field displays a calendar to select a year, month and day.

PPC



Time fields, and Date & Time fields display a clock to allow the user to select a time.



PALM OS

Tapping in a Lookup List field displays a list of items for the user to select one item.

Lookup Lists are created separately from the form design, and can be changed even after a form has been frozen.

PPC

Option 1 of 5 field.

Yes/No Checkbox.

Popup List field.

MultiSelection List.

PALM OS

Tapping in a Popup List field displays a list of items for the user to select one item.

Popup Lists freeze with the form design, meaning that you cannot add or change the items in the list after the form is frozen.

PPC

Numeric field.

Currency field.

Currency amounts are entered in cents. For instance, 39500 is \$395.00.

You can add a numeric keypad to make it easier for the handheld user to enter numbers in Numeric, Currency and Text fields.

PALM OS

A Slider field is a type of numeric integer field.

Tap the Slider bar to position the marker.

A Signature field allows you to tap and record a signature.

PPC

Tapping in a Signature field displays a box in which a person can sign.

PALM OS

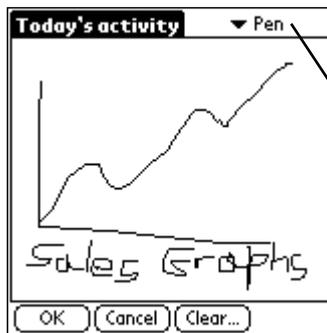
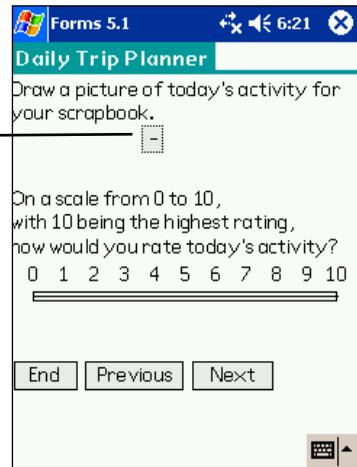


A Sketch field allows you to draw a picture.

If a Sketch field is empty, the Sketch box displays a hyphen (-).

If a Sketch field contains a picture, the sketch box displays the word Image.

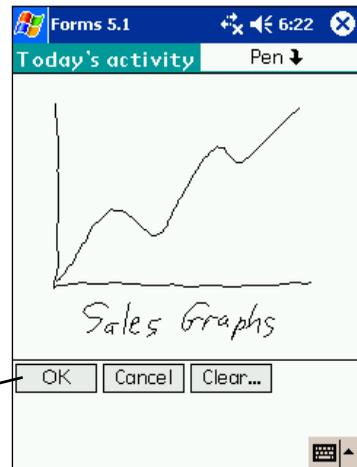
PPC



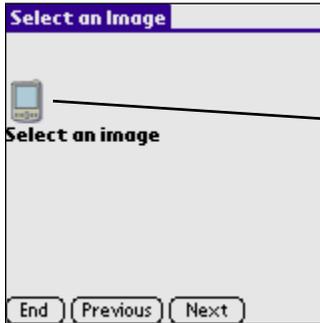
Tapping in a Sketch field displays a screen for drawing a picture.

In the upper right corner you can select to draw with a Pen or Marker, or you can choose Eraser if you need to erase something. To erase the whole picture, tap the Clear button.

To save a sketch tap OK.



PALM OS

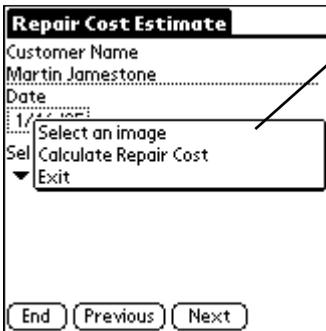
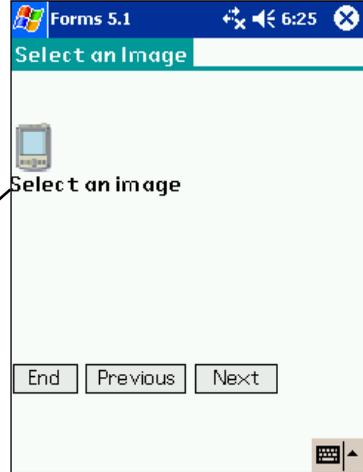


A Section field can be used as a topic heading.

Decorative images can be added to Section fields to help illustrate what the fields in this section of the form are about.

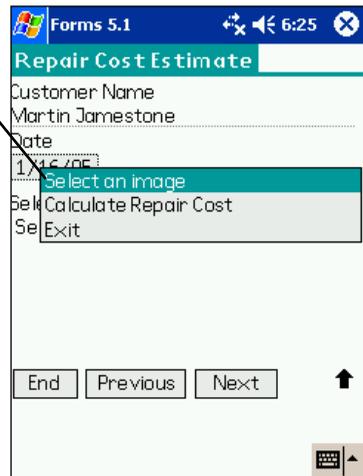
You can provide text instructions in a Section field to help the handheld user complete the fields that follow the Section field.

PPC



A Jump to Section field allows you to list Section fields that the user can jump to.

Another way to branch to a particular field on a form is to use scripts.



PALM OS

Single Subform field.

Subform field.

Single Subform fields and Subform fields allow the handheld user to jump from the parent form to another form (the subform) to fill in additional data.

Tapping on the Single Subform or Subform field jumps the user to the subform.

PPC

In a Single Subform field, the user can only create one new record for each parent record.

A Single Subform provides a way to extend the length of a form, which is limited to 255 fields.

PALM OS

In a Subform field, the user can create many subform records for each parent record.

This is useful for applications such as recording patient visits, sales calls, or customer orders.

PPC

Tapping on a Subform field displays a review screen of existing subform records associated with the parent record.

Tap an existing record to display that record.

Tap the Add button to create a new subform record.

When a subform record is exited, the user is returned to the subform review screen.

Tapping Done on the subform review screen returns the user to the parent form.

PALM OS

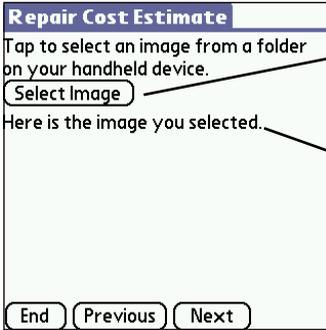
A Time Checkbox field looks like a Yes/No Checkbox.

However, when the checkbox is checked, the current date and time is stored in the field. The handheld user cannot see the date, but it is visible on the PC.

PPC

You can use scripts to perform calculations. The calculations can occur automatically, or you can use a Button field to allow the user to choose when the calculation is performed.

PALM OS



A Button field is also used to trigger a script to attach an image to a Forms record.

An Image field is a place-holder for the image that will be attached.

PPC

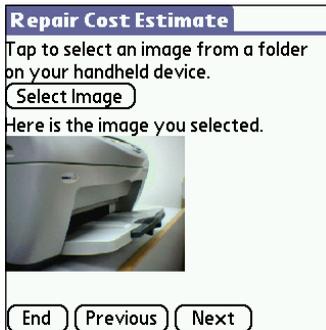


When the Button is tapped, a script allows the handheld user to view the directory structure and select an image file.

On Palm-branded Palm OS 5.4 devices, and on Pocket PC devices, the original image file can be in memory or can be on an external media storage card.



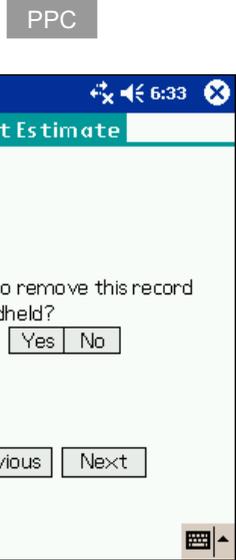
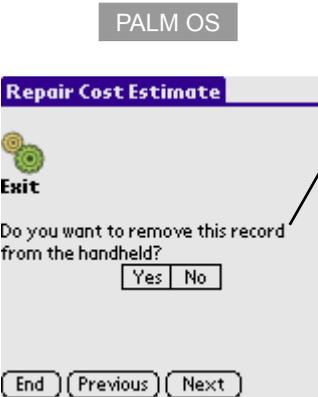
Tapping on an image makes a copy of the image that will be attached to the Pendragon Forms record.



After an image has been attached to the Forms record, the image can be viewed in the record.

You can also use Image fields to attach images to records on the PC, then download the records to the handheld.





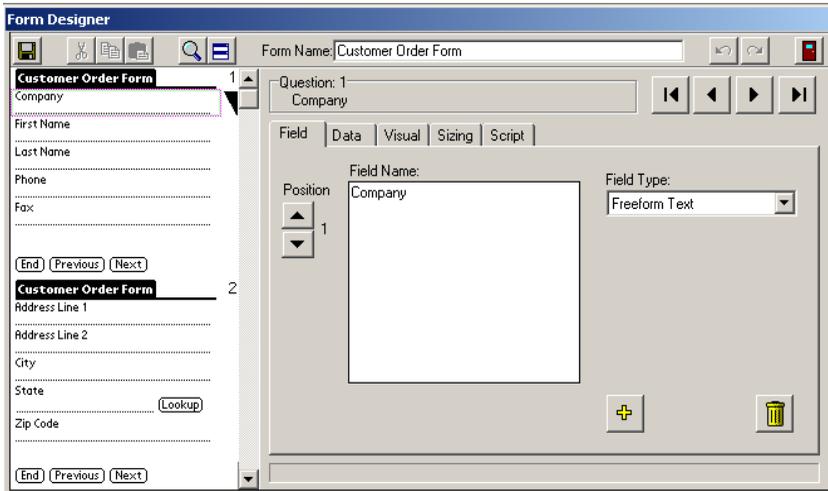
A Completion Checkbox field looks like a Yes/No Checkbox.

If the user taps Yes, the record is removed from the handheld during the next synchronization.

For other ways to limit the number of records on the handheld, see page 51.

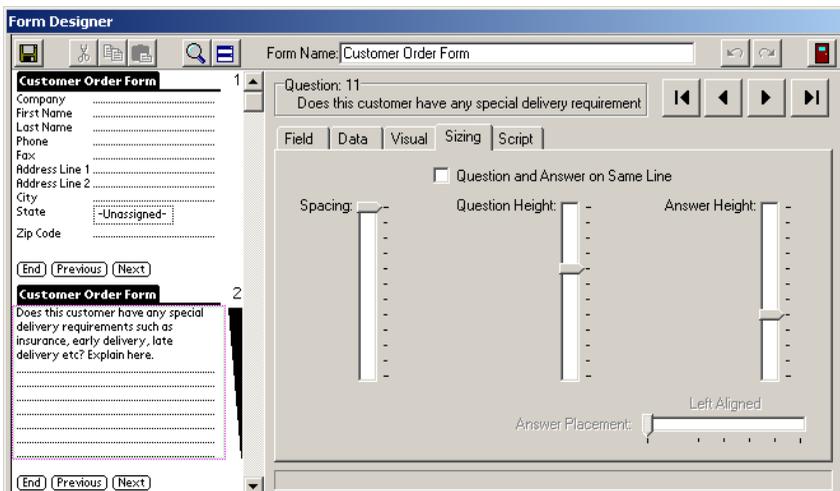
Freeform Text Field

A Freeform Text field, also called a Text field, allows the handheld user to enter any alpha, numeric or other character.



For each field on your form, use the Sizing tab in the Form Designer to size the question or answer portion of the field to match the amount of data that is needed.

For example, a field such as a name can fit the question and answer on the same line, but a field for a comment may need the question or answer to be several lines long.



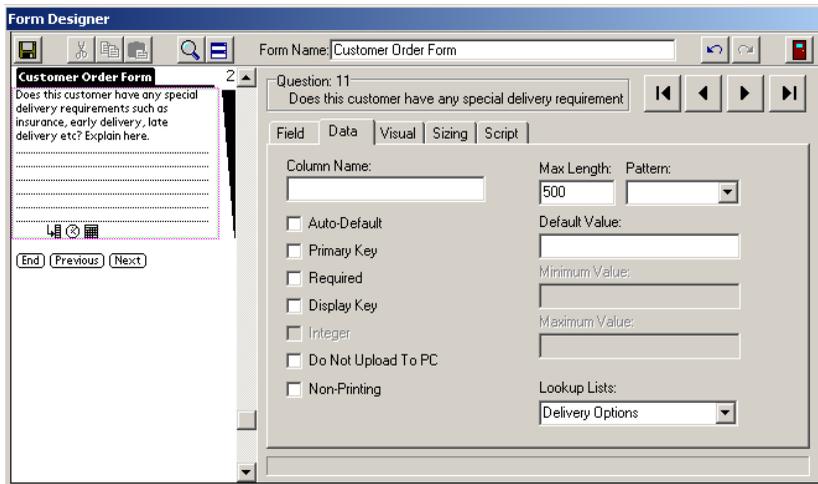
Advanced Field Properties of Text Fields

The Data tab in the Form Designer window shows the Advanced Field Properties that are available for a Text field.

The default maximum length of a Text field is 255 characters, which is enough for most Text fields. If you need to allow more characters or limit users to fewer characters, you can click on the Data tab for the field and type a number from 1 to 2000 in the Max Length field.

NOTE: The Max Length field cannot be changed after a form is frozen.

For information on other Advanced Field Properties, see Chapter 9, *Advanced Field Properties*, page 185.



You can select a Lookup List to be associated with a Text field. If users frequently need to type specific phrases, you can put these phrases in a Lookup List, and the users can select a phrase to insert into the Text field to avoid hand-writing. See page 113 for information on creating a Lookup List.

To associate a Lookup List with a Text field, click the Data tab for that Text field, and select a list in the Lookup List field. Click the Sizing tab and make the answer height bigger, so that you can see the Lookup List icon that appears below the Text field. If the cursor is blinking in the Text field, the handheld user can tap the Lookup List icon to add items from the list to the Text field.

Features of Text Fields

Customer Order Form

Company Chopper Rescue
 First Name P.
 Last Name McOtter
 Phone 847-555-9632
 Fax 847-555-9634
 Address Line 1 8 Seashore Drive
 Address Line 2 Suite 5
 City Chicago
 State IL
 Zip Code 60606

End Previous Next

On the handheld, use Graffiti or the built-in keyboard to enter data in a Text field.

Forms 5.1 4:39

Customer Order Form

Company Chopper Rescue
 First Name P.
 Last Name McOtter
 Phone 847-555-9632
 Fax 847-555-9634
 Address Line 1 8 Seashore Drive
 Address Line 2 Suite 5
 City Chicago
 State IL
 Zip Code 60606

Customer Order Form

Does this customer have any special delivery requirements such as insurance, early delivery, late delivery etc? Explain here.
 Late Afternoon Delivery is preferred after 4:47 pm
 No signature is required.

End Previous Next

If answer icons are available, you can:

- a) Tap the clock icon at the bottom of a Text field to add the current time to the field.
- b) Tap the calendar icon at the bottom of a Text field to add a date and time to the field.
- c) Tap the Lookup List icon to select an item from a Lookup List and paste it into the Text field.

Bar Code Inventory

Scan Bar Code
 04963406
 Item Name
 Coca-Cola 12-oz can
 Listed Quantity 36
 Actual Quantity 42

+-	7	8	9
Del	4	5	6
00	1	2	3
	000	0	.

End Previous Next

A numeric keypad can be used to enter data in Text, Numeric or Currency fields. If the cursor is not blinking in a field, tap in the field before using the numeric keyboard.

Text fields are the only fields that can accept bar code input. See Chapter 15, *Using Bar Codes*, page 373.

Forms 5.1 4:45

Bar Code Inventory

Scan Bar Code
 04963406
 Item Name
 Coca-Cola 12-oz can
 Listed Quantity 36
 Actual Quantity 42

+-	7	8	9
Del	4	5	6
00	1	2	3
	000	0	.

PALM OS

On Palm OS devices, you can copy a name and phone number from the built-in Address Book to a Text field. Tap the handheld Menu button and select the Lookup option. Select a name from the Address Book and tap the Add button.

History of Item

Comments
 Purchased from
 Jones, Karina 847-555-2646 W

End Previous Next

Numeric Field

A Numeric field allows the handheld user to enter a numeric character using an onscreen numeric keypad, Graffiti, or the built-in keyboard.

The maximum length of a Numeric field is 15 digits. A decimal point and a minus sign can be added in addition to the 15 digits.

Exponents can be used to represent very large or very small numbers.

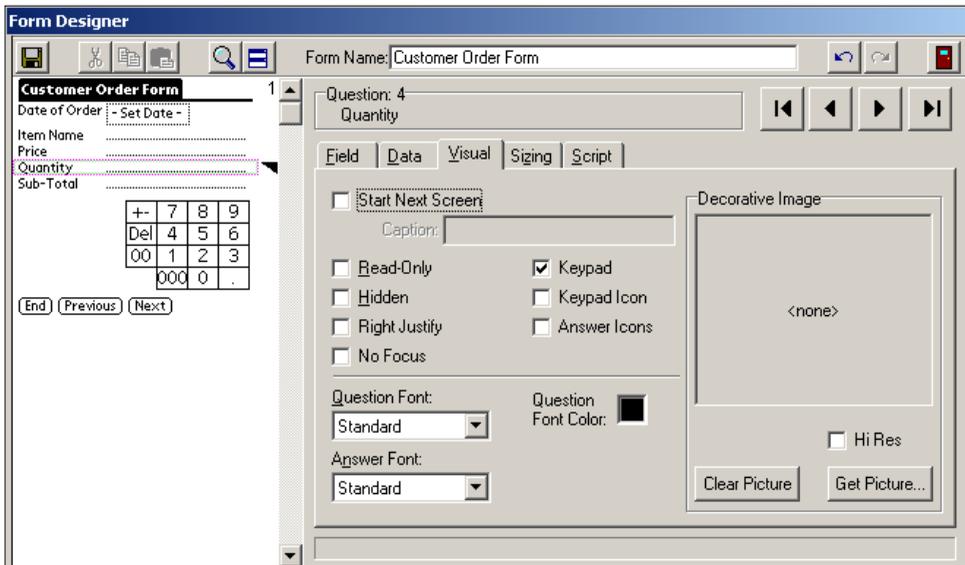
For example:

1,500,000 can be entered as 1.5e6,

0.0025 can be entered as 2.5e-3

The maximum positive exponent (for a large number) is e308, and minimum negative exponent (for a small number) is e-324.

To display an on-screen keypad for a Numeric field, click the Visual tab for that field, and check the Keypad checkbox.



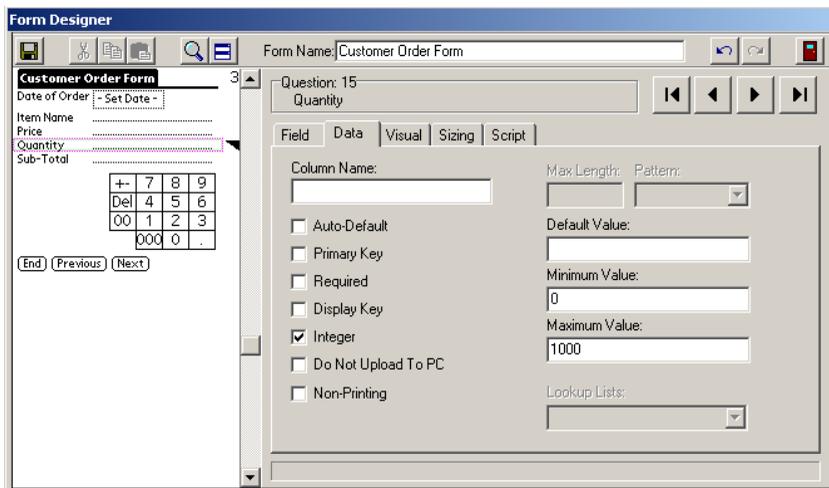
Only one numeric keypad will be displayed per screen, and all the Text, Numeric and Currency fields on that screen can use the same keypad. On the handheld, if the cursor is not blinking in a field, the user should tap in the field to move the cursor to that field before using the numeric keypad.

Advanced Field Properties of Numeric Fields

On the Data tab of a Numeric field, you can check the Integer checkbox if you want the handheld user to enter integers (whole numbers) only. Note that if you choose integers, you are limited to numbers between + or - 2,147,483,647 (effectively 9 digits).

If you want to restrict the handheld user to entering numbers within a numeric range, you can set the maximum (Max) and minimum (Min) values of the allowable range. You must fill in both the Max and the Min values to specify the range.

For information on other Advanced Field Properties of Numeric fields, see page 185.



Features of Numeric Fields

Average Test Score 3	
Student Name	Sandra
Test score 1	85
Test score 2	91
Test score 3	95
Total points	271
Average Score	90.33

+-	7	8	9
Del	4	5	6
00	1	2	3
000	0	.	

End Previous Next

On the handheld, enter numeric characters using the onscreen numeric keypad. You can also use Graffiti or the built-in handheld keyboard.

Numeric fields can contain calculations. See Chapter 14, *Scripting Examples*, page 347.

Average Test Score 3	
Student Name	Sandra
Test score 1	85
Test score 2	91
Test score 3	95
Total points	271
Average Score	90.33

+-	7	8	9
Del	4	5	6
00	1	2	3
000	0	.	

End Previous Next

Currency Field

A Currency field allows the handheld user to enter a number with 2 decimal places.

Currency values are stored as a number of cents. The decimal point is automatically added. Commas are automatically added after every thousand.

The maximum length of a currency field is 9 significant digits including the decimal places, that is, + / - \$9,999,999.99

If a larger number is necessary, use a Numeric field.

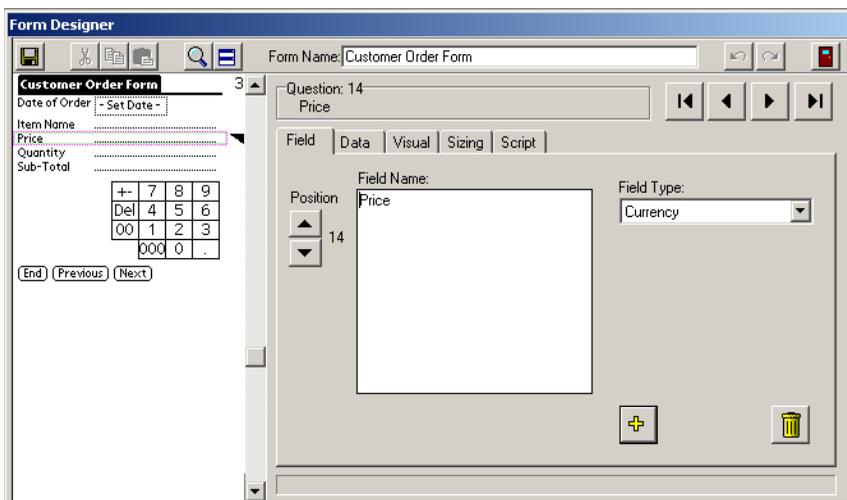
PALM OS

On Palm OS devices, the onscreen keypad or Graffiti can be used to enter data in a Currency field. The handheld's built-in keyboard cannot be used, except in the case of Palm OS devices that have a physical keyboard such as the Treo 650 or Treo 700p. To display the onscreen keypad, click the Visual tab for the Currency field and check the Keypad checkbox.

PPC

On Pocket PC devices, you can use either the onscreen keypad or the Pocket PC device's keyboard to enter Currency values. To display the onscreen keypad, click the Visual tab for the Currency field and check the Keypad checkbox.

Note: Pendragon Forms version 5.1 supports 9 significant digits including decimal places. This is a difference from Forms version 5.0 which supported 16 significant digits on Pocket PC devices.



Features of Currency Fields

Customer Order Form			
Date of Order	4/15/03		
Item Name	Rotor 5000		
Price	\$259.95		
Quantity	2		
Sub-Total	\$519.90		
	+-	7	8
	Del	4	5
	00	1	2
		000	0
			.
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>			

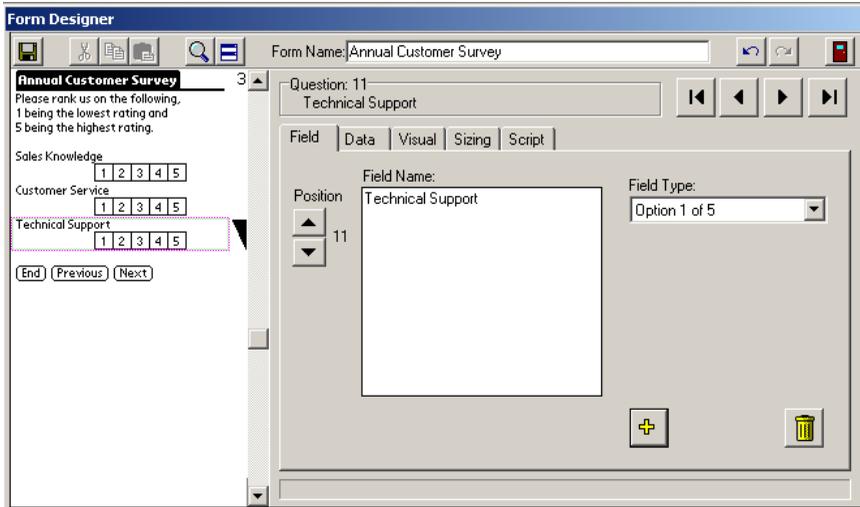
Currency fields are stored as a whole number of cents, and the Currency field displays the number with the appropriate decimal places. For example, entering 25995 in a Currency field will display as \$259.95.

Forms 5.1 Customer Order Form			
Date of Order	4/15/03		
Item Name	Rotor 5000		
Price	\$259.95		
Quantity	2		
Sub-Total	\$519.90		
	+-	7	8
	Del	4	5
	00	1	2
		000	0
			.
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>			

Calculations can be performed in Currency fields. If the result of a Currency calculation is also placed in a Currency field, the result will be displayed correctly. See Chapter 14, *Scripting Examples*, page 348-349.

Option 1 of 5 Field

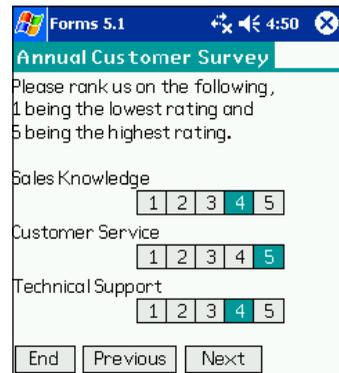
The Option 1 of 5 field displays five checkboxes labeled 1 2 3 4 5, and the handheld user can check one box to make a selection.



Features of Option 1 of 5 Fields



On the handheld, tap a box in the Option 1 of 5 field to select that number.



Option 1 of 5 can be configured to have fewer than five checkboxes, or to label the boxes with a different character. To do this, add a character sequence in square brackets at the end of the field name in the Form Designer. Example: Overall Grade [ABCD]

Note that after synchronization, options on the PC still appears as the numbers 1-5.

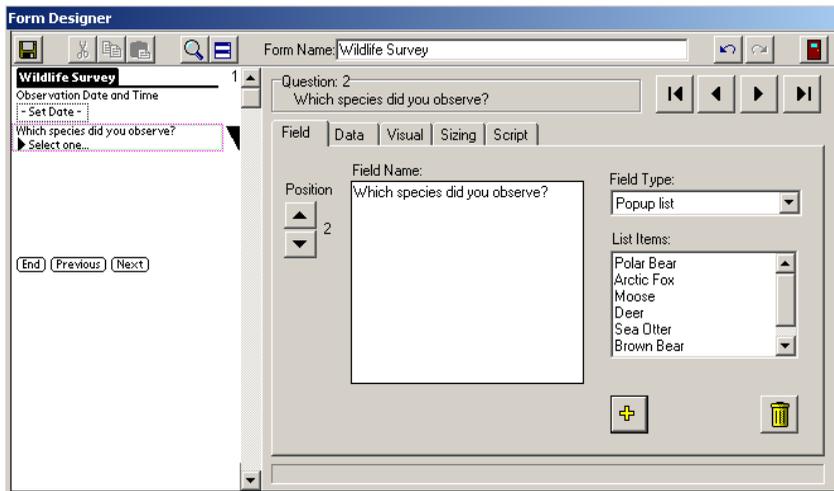
If you prefer to record the alpha characters, use a Popup List field instead of an Option 1 of 5 field.

Popup List Field

A Popup List field allows the handheld user to select only one item from a list of items.

In the Form Designer window, type each item in the list on a separate line in the List Items section of the screen. Each item in the list cannot exceed 30 characters. The maximum length of a Popup List is 512 characters. A carriage return (pressing Enter) counts as 2 characters.

NOTE: When you freeze a form, Popup List entries are also frozen and cannot be changed. If you need a list which can be modified after the form is frozen, use a Lookup List field instead of a Popup List.

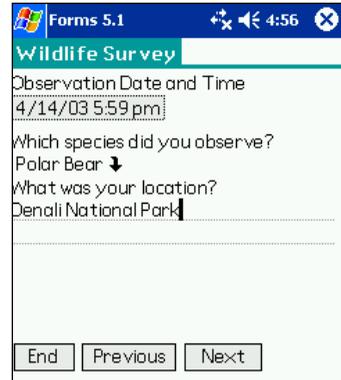
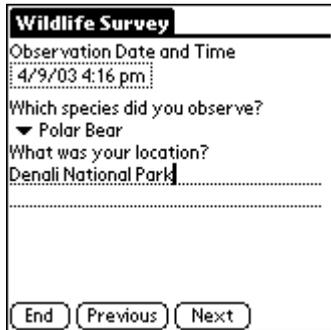


Features of Popup Lists



Tap in a Popup List field and make a selection from the list.

You can create a cascading Popup/Lookup relationship, whereby a selection made in a Popup List determines which Lookup List is displayed in the next field. See Cascading Lookup, page 119.

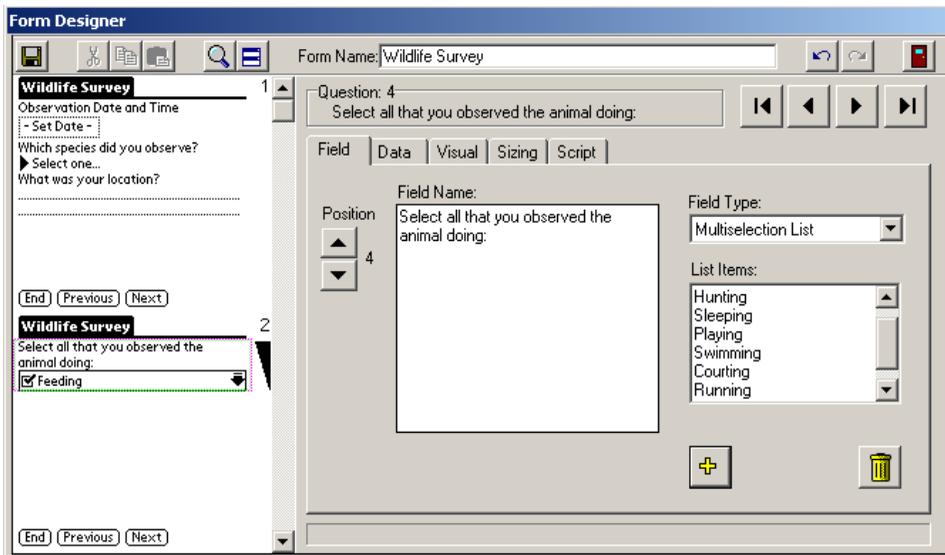


MultiSelection List Field

A MultiSelection List allows the handheld user to select one or more items from a list. A checkbox appears next to each item in the list, and the user can choose which boxes to check.

In the Form Designer window, type each item in the list on a separate line in the List Items field. Each item in the list can be up to 30 characters.

The maximum number of items that can be entered in a MultiSelection List is 32 items.

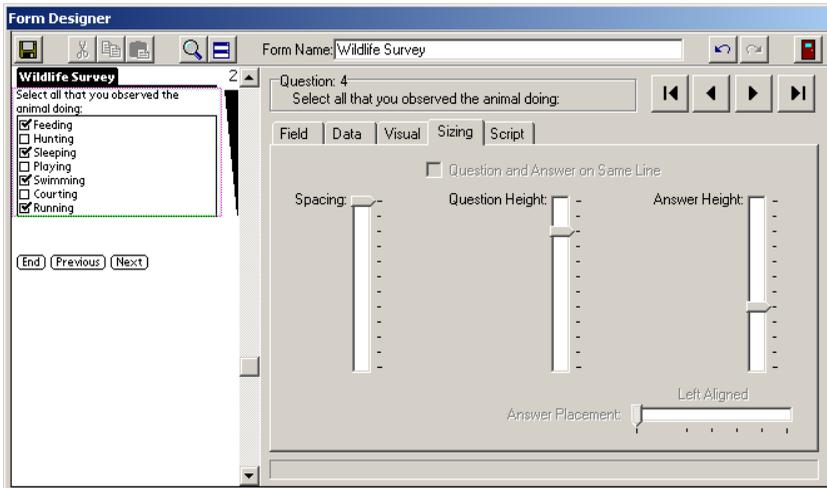


NOTE: When you freeze a form design, the MultiSelection List entries are frozen and cannot be changed. If you need to change a list on a regular basis, you may want to consider using a Text field with a Lookup List instead of a MultiSelection List. (See page 88.)

Internally, a MultiSelection List is stored as a binary number, with each bit position in the number representing one of the options in the list. To use a MultiSelection field in a script, see page 366.

When you create a MultiSelection List field, click the Sizing tab in the Form Designer to re-size the answer height of the field so that the user can see as many options as can fit on the handheld screen.

If the list has more entries than can be seen, a scroll arrow in the lower right corner of the list, or the handheld Page Up and Page Down buttons can be used to scroll down.



When data comes back to the PC, all selected items will be listed in the field, separated by semicolons.

ObservationDa	WhichSpecies	WhatWasYour	SelectAllThatYouObserv
4/9/2003 4:16:00 PM	Polar Bear	Denali National Park	Feeding;Swimming
6:01:00 PM	Sea Otter	Oceanfront	Feeding;Sleeping;Playing; Swimming

Features of MultiSelection Lists

To make a selection in a MultiSelection List field, check one or more checkboxes in the field.

If the list contains more items than can be seen, there will be a scroll arrow in the lower right corner of the MultiSelection box. You can tap on the scroll arrow to scroll down the list, or use the handheld PageUp and PageDown buttons to scroll.

A MultiSelection List can only be modified in Layout View or Field View.

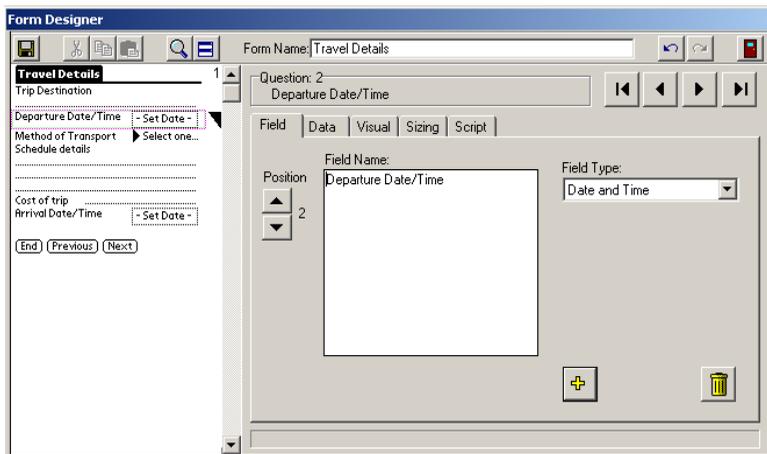
If you are in Record View, tapping in a MultiSelection List field will switch you to Field View to see the list. Tap the Record View button to return to Record View.

On the Review screen of the handheld, when you are reviewing a list of records, the contents of a MultiSelection List will not be visible unless you select the record to view in detail.

Date & Time Field

A Date & Time field allows the handheld user to select a date and a time from a popup calendar and clock.

The default date and time that are displayed in the popup calendar and clock are the current date and time on the handheld.



It is possible to perform calculations on dates. See Chapter 14, *Scripting Examples*, page 354.

A script can also be used to place a default date, such as the current date and time, into a Date & Time field. See page 354.

Features of Date & Time Fields

Travel Details

Trip Destination
Chicago

Departure Date/Time
5/19/03 8:00 am

Method of Transport ▼ Airline
Schedule details
Flight 25

Cost of trip \$299.00
Arrival Date/Time 5/19/03 11:25 am

End Previous Next

On the handheld, tap in the Date & Time field to display a popup calendar and clock.

Forms 5.1

Travel Details

Trip Destination
Chicago

Departure Date/Time
5/19/03 8:00 am

Method of Transport Airline ▼
Schedule details
Flight 25

Cost of trip \$299.00
Arrival Date/Time 5/19/03 11:25 am

End Previous Next

Departure Date/Ti...

◀ 2003 ▶

Jan	Feb	Mar	Apr	May	Jun	
Jul	Aug	Sep	Oct	Nov	Dec	
S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Cancel Today

To change the year, tap the arrows next to the year. 1904 is the earliest year you can enter. Tap on a month to select that month. Tap on a day to select that day.

Forms 5.1

Departure Date/Ti...

◀ 2003 ▶

Jan	Feb	Mar	Apr	May	Jun	
Jul	Aug	Sep	Oct	Nov	Dec	
S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Cancel Today

Travel Details

Trip Destination
Chicago

Departure Date/Time
5/19/03 8:00 am

Method of Transport ▼ Airline
Schedule details
Flight 25

Departure Date/Ti...

08 : 00 AM PM

OK Cancel

After a date has been selected, a clock dialog will be displayed for you to select a time. Tap in each digit position and use the arrows to change that digit. Tap AM or PM.

Forms 5.1

Travel Details

Trip Destination
Chicago

Departure Date/Time
5/19/03 8:00 am

Method of Transport Airline ▼
Schedule details
Flight 25

Departure Date/Ti...

08 : 00 AM PM

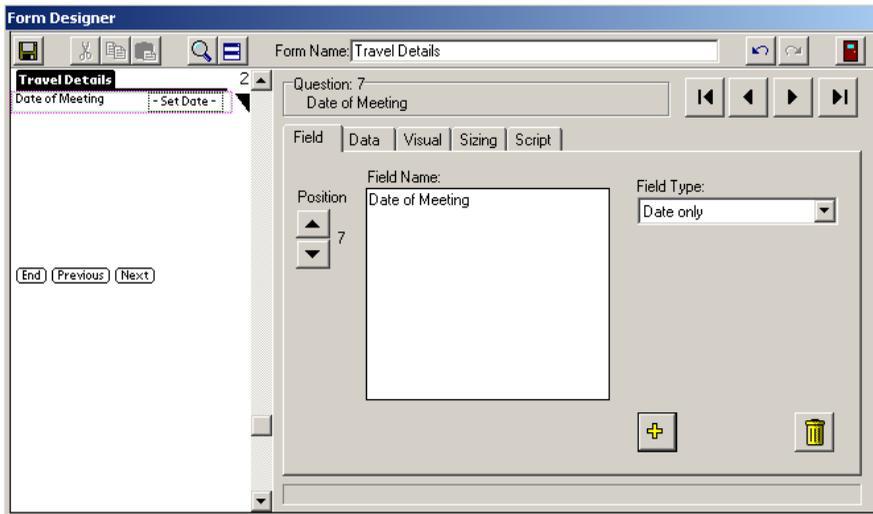
OK Cancel

You can also use Graffiti or the handheld keyboard to set the clock dialog.

Date Field

A Date Only field, also called a Date field, allows the handheld user to select a date from a popup calendar.

The default date that is displayed in the popup calendar is the current date on the handheld.

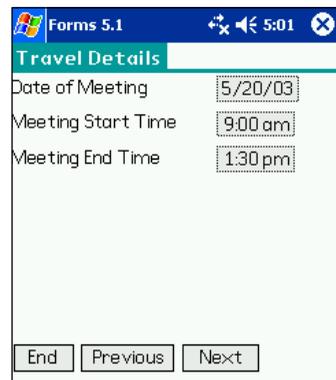


Features of a Date Field



On the handheld, tap in the Date field to display a calendar and choose a date.

If you do not want to select a date, tap the Cancel button on the calendar.

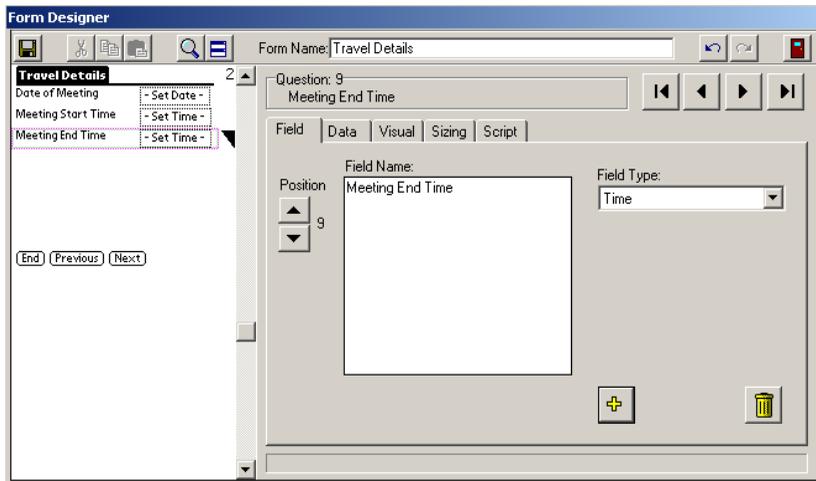


You can perform calculations on dates. See Chapter 14 *Scripting Examples*, page 354.

A script can also be used to place a default date, such as the current date, into a Date field. See page 354.

Time Field

A Time field allows the handheld user to select a time from a popup clock. The default time that is displayed on the popup clock is the current time on the handheld.



Features of Time Fields

Travel Details	
Date of Meeting	5/20/03
Meeting Start Time	9:00 am
Meeting End Time	1:30 pm
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

On the handheld, tap the Time field to display a clock and select a time.

If you do not want to select a time, tap the Cancel button on the clock.

You can use a script to automatically insert the current time into a Time field. See page 354.

Travel Details	
Date of Meeting	5/20/03
Meeting Start Time	9:00 am
Meeting End Time	1:30 pm
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

You can switch to a 24-hour clock by making a global change on the handheld. Note, however, that after synchronization, the data on the PC will be displayed in the format selected in the Windows Control Panel.

PALM OS

On Palm OS devices, tap the Applications button (Home icon), then tap the Preferences icon. In the General section, tap Formats, then set Time to HH:MM.

PPC

On Pocket PC devices, tap Start...Settings. On the System tab, tap Regional Settings. On the Time tab change the Time style to HH:mm:ss.

Yes or No Checkbox Field

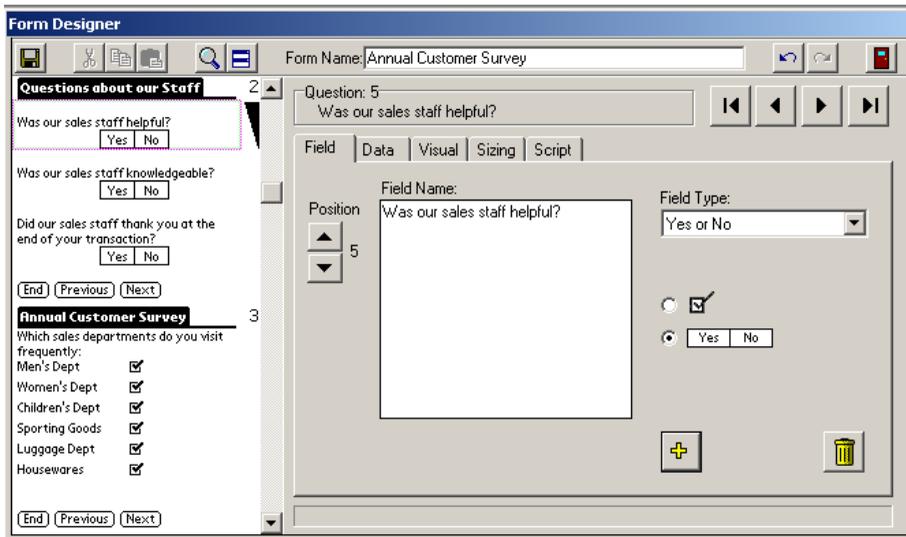
A Yes or No checkbox allows the handheld user to select one of two options: Yes or No.

Data coming back to the PC will contain three possible values:

Y for Yes,

N for No,

or be blank if no option was selected.



Click the appropriate button to choose whether the Yes/No checkbox appears as a single checkbox, or whether the field is displayed as two individual boxes labeled Yes and No. If you choose a single checkbox, then to select No, the handheld user would need to check and then uncheck the checkbox, or you can default the field to N (for No) - see Advanced Field Properties on the next page.

If you choose to view records in Field View, Yes/No checkboxes always show the Yes and No boxes as separate. Record View shows a single checkbox.

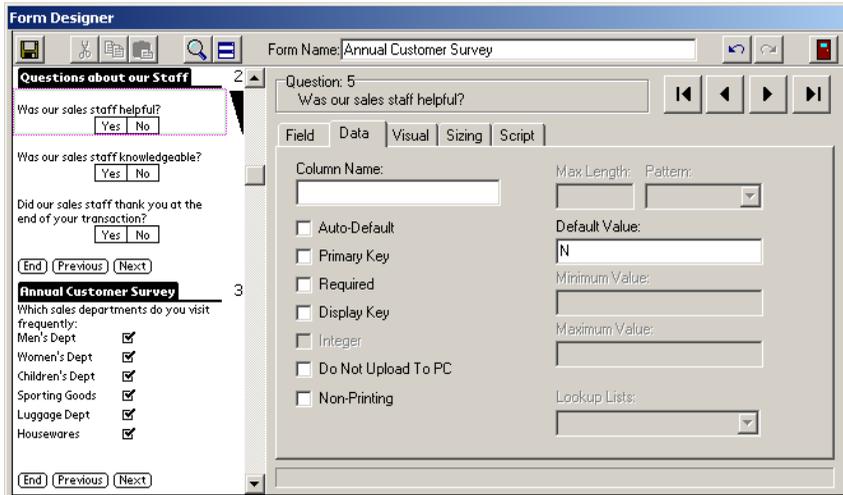
Advanced Field Properties of Yes/ No Checkbox Fields

If the user leaves a Yes/No checkbox blank, the data value will be blank (null).

If you do not want to allow a null value, you can set a default so that if the handheld user does not touch the checkbox, the default value will be the value stored in the field.

To set a default value for a Yes/No checkbox, click the Data tab for the Yes/No field.

In the Default Value field, type N to default to No, or type Y to default to Yes.



Instead of using a default value, another way to avoid a null value is to require the handheld user to fill in the Yes/No checkbox. On the Data tab, check the Required checkbox. For more information see Chapter 9, *Advanced Field Properties*, page 189.

Features of Yes/No Checkboxes

The screenshot shows a form titled 'Questions about our' with three questions, each with a Yes/No checkbox. The 'Yes' boxes are highlighted in black, indicating they are selected. The questions are: 'Was our sales staff helpful?', 'Was our sales staff knowledgeable?', and 'Did our sales staff thank you at the end of your transaction?'. Navigation buttons 'End', 'Previous', and 'Next' are at the bottom.

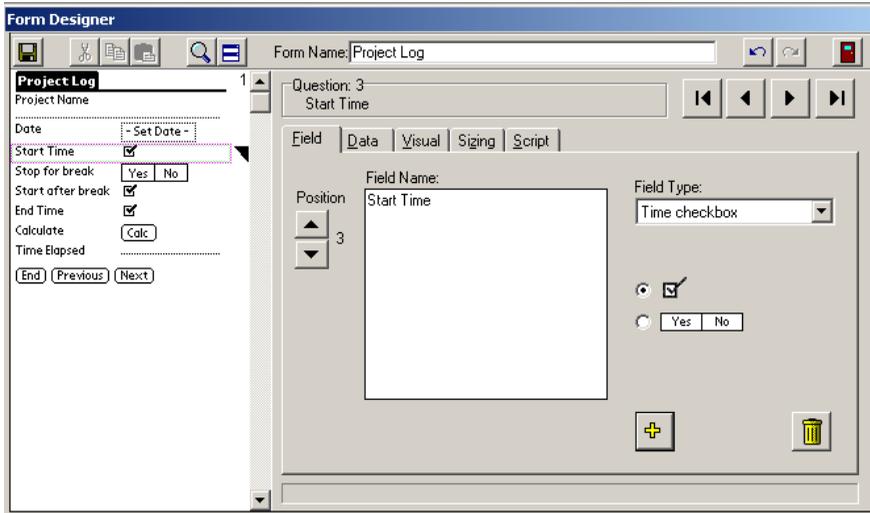
If two separate boxes are available for Yes and No, tap one box to make a selection. If only one checkbox is visible, check the box for Yes, or check and then un-check the box for No.

Yes/No Checkboxes can be used in branching scripts, to jump to another part of the form depending on the response of Yes or No. See Chapter 14, *Scripting Examples*, page 358.

The screenshot shows a form titled 'Questions about our' with three questions, each with a Yes/No checkbox. The 'Yes' boxes are highlighted in green, indicating they are selected. The questions are: 'Was our sales staff helpful?', 'Was our sales staff knowledgeable?', and 'Did our sales staff thank you at the end of your transaction?'. Navigation buttons 'End', 'Previous', and 'Next' are at the bottom.

Time Checkbox Field

A Time Checkbox is a special type of Yes/No checkbox in which, when the handheld user checks the box for Yes, a date and time stamp is recorded. The date and time stamp is only visible when you view data on the PC.



Features of Time Checkboxes

Project Log	
Project Name	Alpha Project
Date	4/11/03
Start Time	<input checked="" type="checkbox"/>
Stop for break	<input checked="" type="checkbox"/>
Start after break	<input checked="" type="checkbox"/>
End Time	<input checked="" type="checkbox"/>
Calculate	<input type="button" value="Calc"/>
Total Time Elapsed	4 hrs and 43 mins...
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

On the handheld, if you tap Yes or check the Time checkbox, a date and time stamp will be added. This date and time stamp is only visible on the PC.

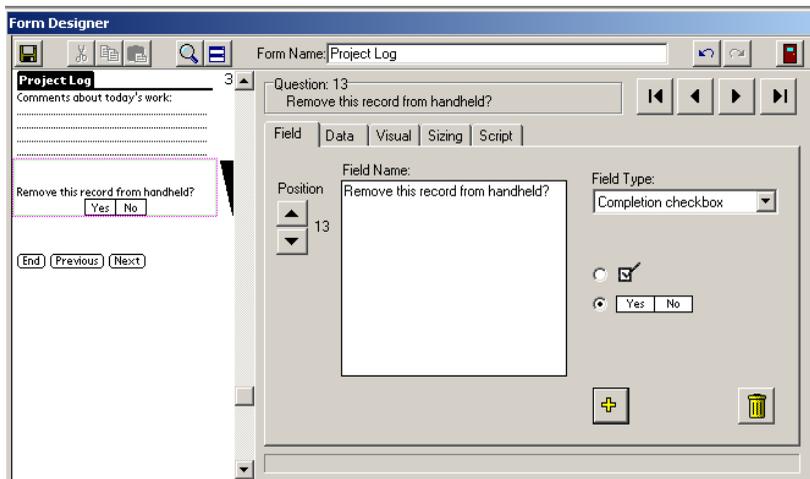
If a Time Checkbox is left blank, or if the option No is checked, a null field is sent back to the PC - no date or time stamp.

Project Log	
Project Name	Alpha Project
Date	4/11/03
Start Time	<input checked="" type="checkbox"/>
Stop for break	<input checked="" type="checkbox"/>
Start after break	<input checked="" type="checkbox"/>
End Time	<input checked="" type="checkbox"/>
Calculate	<input type="button" value="Calc"/>
Total Time Elapsed	4 hrs and 43 mins...
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

Completion Checkbox Field

A Completion Checkbox is a special type of Yes/No checkbox. When the handheld user checks Yes, it flags the record for removal from the handheld on the next synchronization.

NOTE: For a Completion Checkbox to work, you must also go to the Data Persistence section of the **Form Properties** window, and select to **Keep Incomplete records on the Handheld**. (See page 198.) To access the Form Properties window, click on the name of the form in the Forms Manager and then click the Properties button.



Features of Completion Checkboxes

The screenshot shows the 'Project Log' form. The 'Remove this record from handheld?' question has a 'Yes' checkbox that is checked. The 'Yes' checkbox is highlighted with a black background. The form also includes a 'Comments about today's work:' section and 'End', 'Previous', and 'Next' buttons.

To select a record to be removed on the next synchronization, check the Yes checkbox of a Completion Checkbox field.

A Completion Checkbox field can cascade from a parent form to a subform. See page 139.

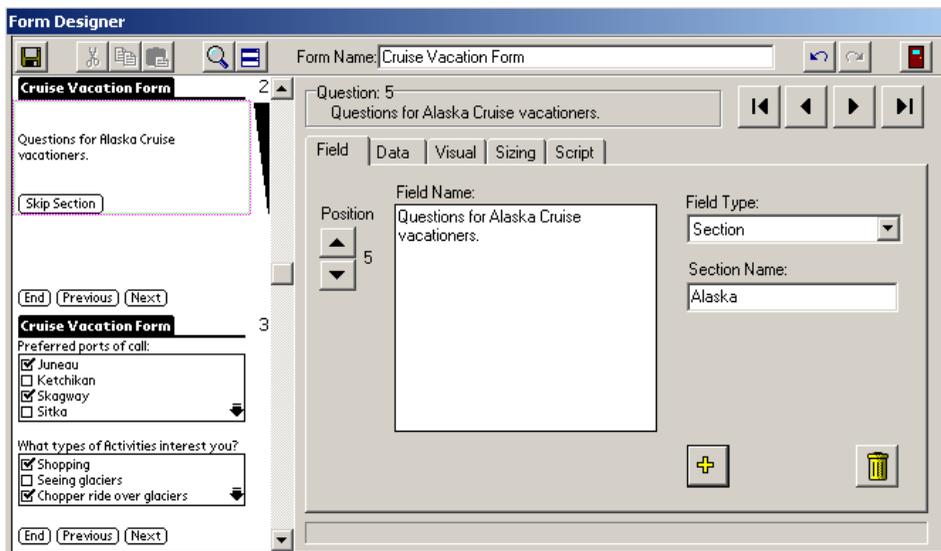
The screenshot shows the 'Project Log' form. The 'Remove this record from handheld?' question has a 'Yes' checkbox that is checked. The 'Yes' checkbox is highlighted with a green background. The form also includes a 'Comments about today's work:' section and 'End', 'Previous', and 'Next' buttons.

Section Field

A Section field is a read-only topic heading.

The handheld user cannot enter data in a Section field. Instead, the name of the Section field provides the user with information for filling in the fields that follow the Section field.

In the Form Designer window, when you select Section field as your field type, you will need to type a name for the section in the Section Name field. The Section Name can be up to 30 characters.

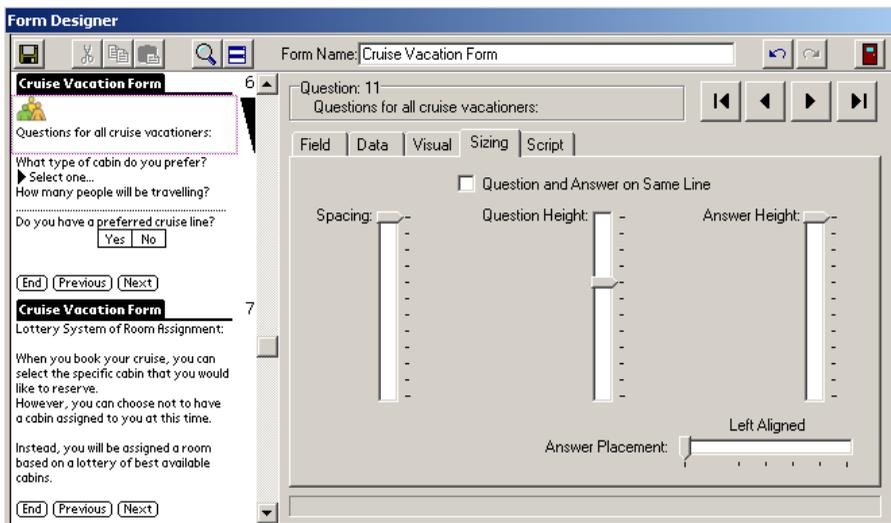


A Section field has three uses:

1. Section headings can form a visual divider between different categories of information.
2. Sections can be used in connection with Jump to Section fields to make it possible to jump to a section. Note however that Scripting may be a more powerful feature to use than Sections and Jump to Sections.
3. A Section field can be used if you need more space for a long question. By putting a Section field ahead of another field, you can double the onscreen space of your question.

Section fields include a Skip Section button. Tapping this button skips the questions in the current section of the form, and advances the user to the next Section field on the form. If there are no more Section fields, the user will jump to the end of the record.

If you do not want the handheld user to be able to skip over sections of a form, you can click the Sizing tab of a Section field and make the Answer Height zero. This will hide the Skip Section button.



You can choose to add a graphic to a Section field name, to help to illustrate the purpose of the section. If you click the Visual tab of a Section field, you can click the Get Picture button to select a graphic to use with the Section field. See Chapter 8, *Advanced Form Designer*, page 171, for more information on adding pictures to fields.

You can also use Section fields containing pictures to act as a 'main menu' screen, from which the user can jump to other forms to fill in. See Chapter 21, *Custom Main Menu*, page 471.

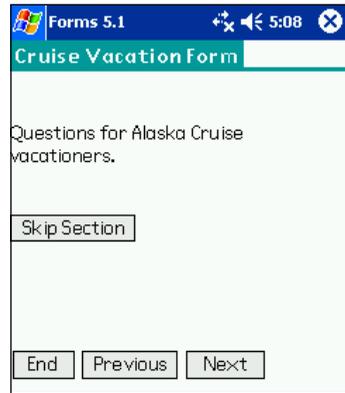
Features of Sections



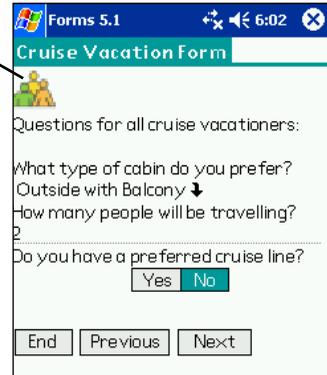
A Section field provides information about the questions in this section of the form. If a Section field occupies a whole screen, the questions in this section of the form begin after the Section field.

To skip a Section and jump to the next Section field on the form, tap the Skip Section button. If there are no more Section fields on the form, you will end the record.

To start answering the questions in this section of the form, tap the Next button.



Here a Section field includes a graphic, and has the Section Skip button hidden.

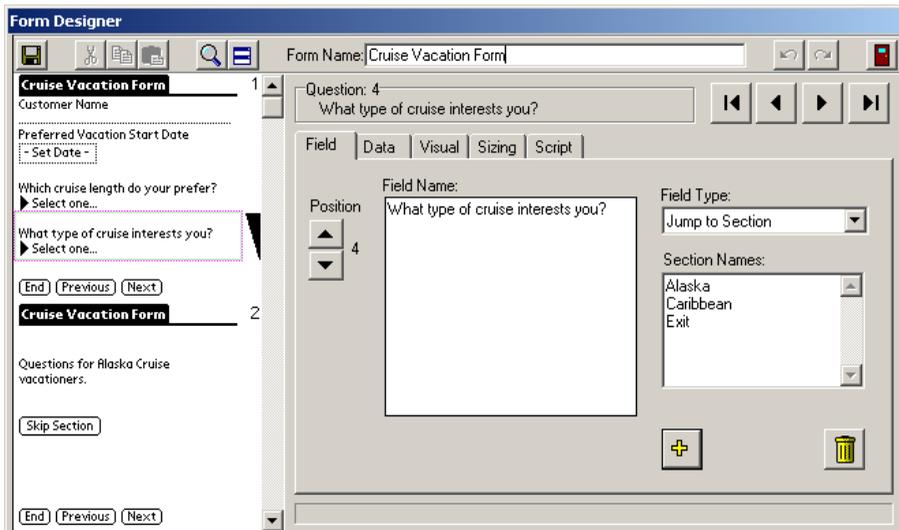


Jump to Section Field

A Jump to Section field is a special type of Popup List that allows the handheld user to jump to a Section field on a form.

When you select the Jump to Section field type in the Form Designer window, a Section Names field appears. Each item in the list must correspond to the Section Name of a Section field. See page 108.

The maximum length of a Jump to Section list is 512 characters.
A carriage return (pressing Enter) counts as 2 characters.



Features of Jump to Sections

On the handheld, selecting an item from a Jump to Section list...

...jumps the user to that Section field on the form.

The user can jump forward or backward in the form, depending on where the Section fields are located on the form.

After filling out the questions in a given section of the form, you may need a branching script to return the user to the Jump to Section field so that they can select another section to fill in. See Chapter 14, *Scripting Examples*, page 363.

Remember to have a way for the user to end the form. In the Jump to Section field shown in the top pictures, there is an Exit section that does not return the user to the Jump to Section field.

Lookup List Field

Lookup Lists allow the handheld user to select an item from a list.

With a Lookup List, the handheld user has the option to edit the field after a selection has been made, if nothing in the list exactly matches what they want to enter in the field.

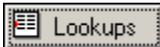
The question and answer components of the field must be on separate lines to allow editing of the field. The edited field is not added to the list, it is just stored in the current record.

An Exclusive Lookup option can be set if you do not want the user to edit the selection.

Lookup Lists are created separately from the form design, and can be modified even after a form has been frozen. This makes a Lookup List field more flexible than a Popup List or a MultiSelection List. Modified Lookup Lists are sent to the handheld during synchronization.

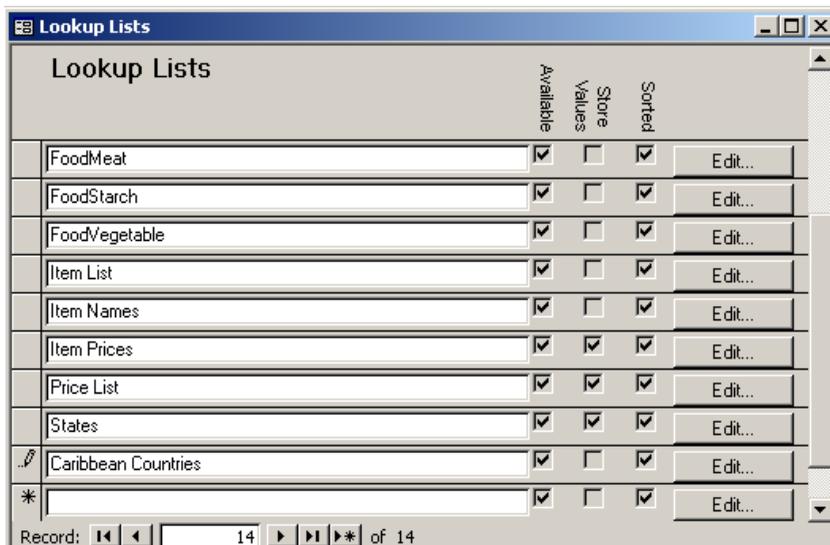
Step 1: Creating a Lookup List

In the Pendragon Forms Manager, click the Lookups button to display the Lookup Editor.



To create a new Lookup List, type a name for the list in the blank row. Click the Edit button next to that Lookup List name.

To edit an existing Lookup List, click the Edit button next to the name of the list.



In the LookupDisplay column, type each item that you want in the Lookup List. To save an entry, click in the next row, or press TAB until the cursor moves to the next row.

The maximum length for a single item is 50 characters. However on the handheld screen you can only view a width of 36 characters.

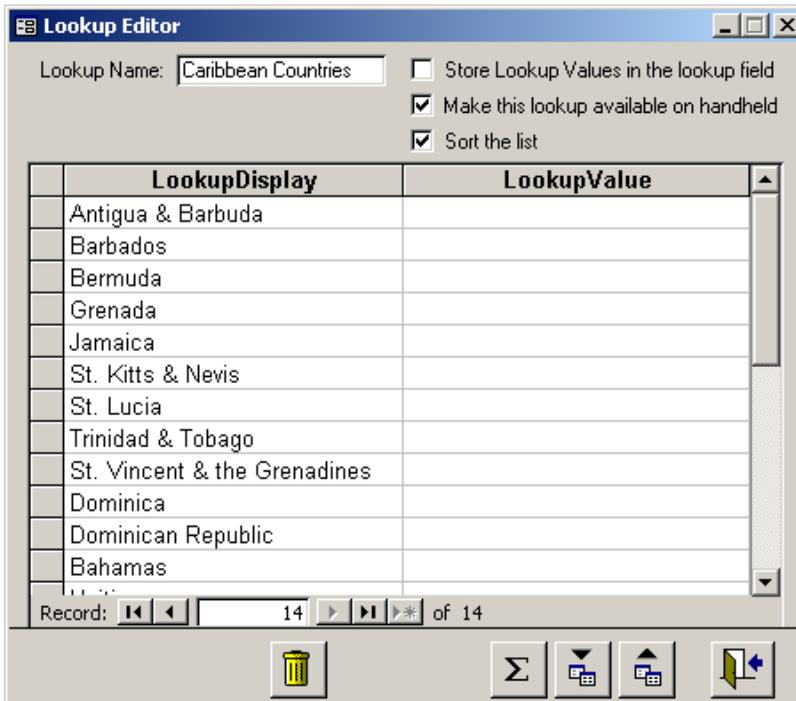
The maximum length for the entire Lookup List is 1000 items or 32KB.

To store the item name that the user selects, leave the LookupValue column blank.

Check the checkboxes:

Make this lookup available on handheld

Sort the list (Optional: sorts the list A..Z.)



To save and close the Lookup Editor, click the Close button (Exit Door icon) or click the X button in the upper right of the Lookup Editor screen. Also close the Lookup List window.

The LookupValue column can be used if you want the handheld user to see the item names in the list, but when they make a selection, store an abbreviation or alternate value in the field.

For example, you may want the user to see the names of US states in the list, but store the two-letter postal abbreviation for each state.

In this case, type the full names of the items in the LookupDisplay column, and type the corresponding abbreviations in the LookupValue column. Do not leave any blanks in the LookupValue column.

Check the checkboxes:

Store LookupValues in Lookup List

Make this lookup available on handheld

Sort the list (Optional: sorts the list A..Z.)

NOTE: On the PC, if you click the Edit/View button to view the records for the form, you will still see the LookupDisplay items, not the Lookup Value items. The actual database table, which can be viewed by clicking ALT + Edit/View, contains the Lookup Value items.

Lookup Editor

Lookup Name:

Store Lookup Values in the lookup field

Make this lookup available on handheld

Sort the list

	LookupDisplay	LookupValue
▶	Alabama	AL
	Arkansas	AR
	Illinois	IL
	Indiana	IN
	Wisconsin	WI
	Iowa	IA
	Hawaii	HI
	Alaska	AK
	Minnesota	MN
	Maine	ME
	New York	NY
	New Jersey	NJ

Record: of 51

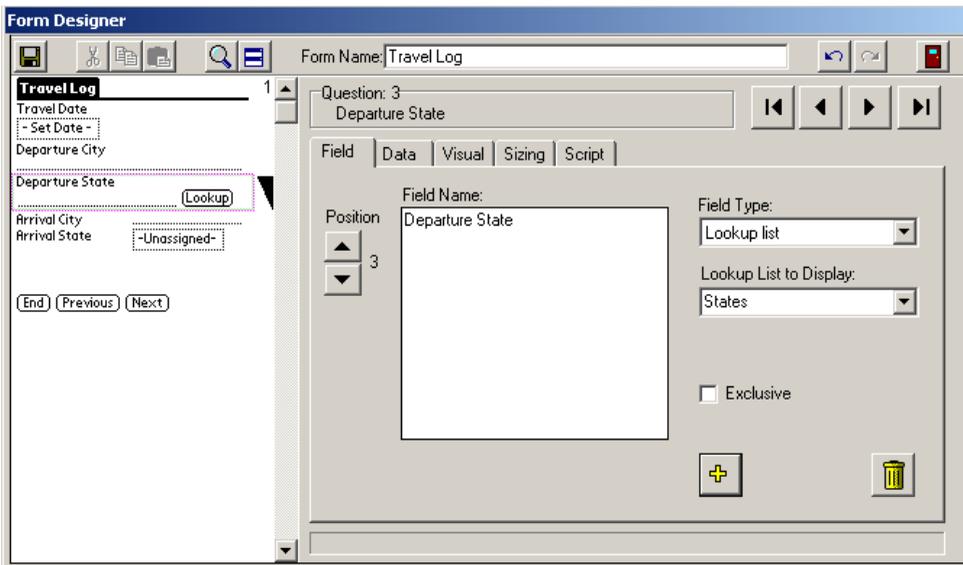
Step 2: Creating a Lookup List Field

Once you have created your Lookup List, you can create a Lookup List field on your form.

In the Form Designer window, select Lookup List as the field type.

In the Lookup List to Display section of the screen, click the arrow and select the name of a Lookup List.

If you do not want handheld users to have the ability to edit a selection to create an item that is not in the list, check the Exclusive checkbox.



Lookup List fields appear in two different ways depending on whether the question and answer components of the field are on the same line or not.

In the picture above, the Departure State field takes up two lines on the handheld: the question on one line and the answer on the next line. In this case the Lookup List field appears as a line for storing text, with a Lookup button next to the line. The handheld user can tap the button to make a selection from the list, and can optionally edit the selection if the list is not exclusive.

In the Arrival State field in the picture, the question and answer components of the field have been put on the same line via the the Sizing tab of the Form Designer. In this case the Lookup List appears as a box marked Unassigned. The user can tap the box to display the Lookup List. No manual editing of the field is possible, whether exclusive is set or not.

Features of Lookup Lists

On the handheld, tap the Lookup button or box to display the list. Tap an item to select it.

If the list is too long to fit on one screen, there will be arrows to allow you to scroll up and down the list. You can also use the handheld PageUp and PageDown buttons to scroll through the list.

If the list is sorted in alphabetic order, you can also jump to the first item beginning with the letter that you write in the Graffiti area or on the handheld keyboard.

Troubleshooting Lookup Lists

On the handheld, if you get an error message *Lookup Failed. List not found.*, it means that the Lookup List that is mentioned in the error message is not on the handheld.

On the PC, click the Lookups button in the Forms Manager on the PC. If the *Available* checkbox is not checked, edit the Lookup List and check the *Make this lookup available on handheld* checkbox. Synchronize the handheld and see if the Lookup List appears.

If the problem persists, click on the form in the Forms Manager, then click the Edit button to open the Form Designer. Display the Lookup List field and re-select the Lookup List that you want to use. Re-distribute the form design, synchronize, and check the handheld again.

Deleting Lookup Lists

Deleting a Lookup List requires care to ensure that the list is removed from the handheld.

1. In the Forms Manager, click the Lookups button. Click the Edit button next to the Lookup List that you want to delete.
2. Un-check the checkbox *Make this lookup available on handheld*, so that this checkbox is blank. Close the Lookup Editor window.
3. Synchronize all the handheld devices to remove the Lookup List from each handheld.
4. To delete the Lookup List from the PC, edit the Lookup List and click the Trash can button.

Importing/Exporting Lookup Lists

If you want to create a Lookup List based on data in an ASCII file, you can create the Lookup List and then import the data into the list. See page 241 for more information on importing Lookup Lists.

If you are giving someone a form design that you have created, and the form contains Lookup Lists, you will need to export the Lookup Lists separately from the form design. See page 241 for more information on exporting Lookup Lists.

Other Uses for Lookup Lists

It is possible to allow the handheld user to perform a lookup from a Text field, and paste the selected item into the Text field. This allows the user to select more than one item from the Lookup List. See page 88.

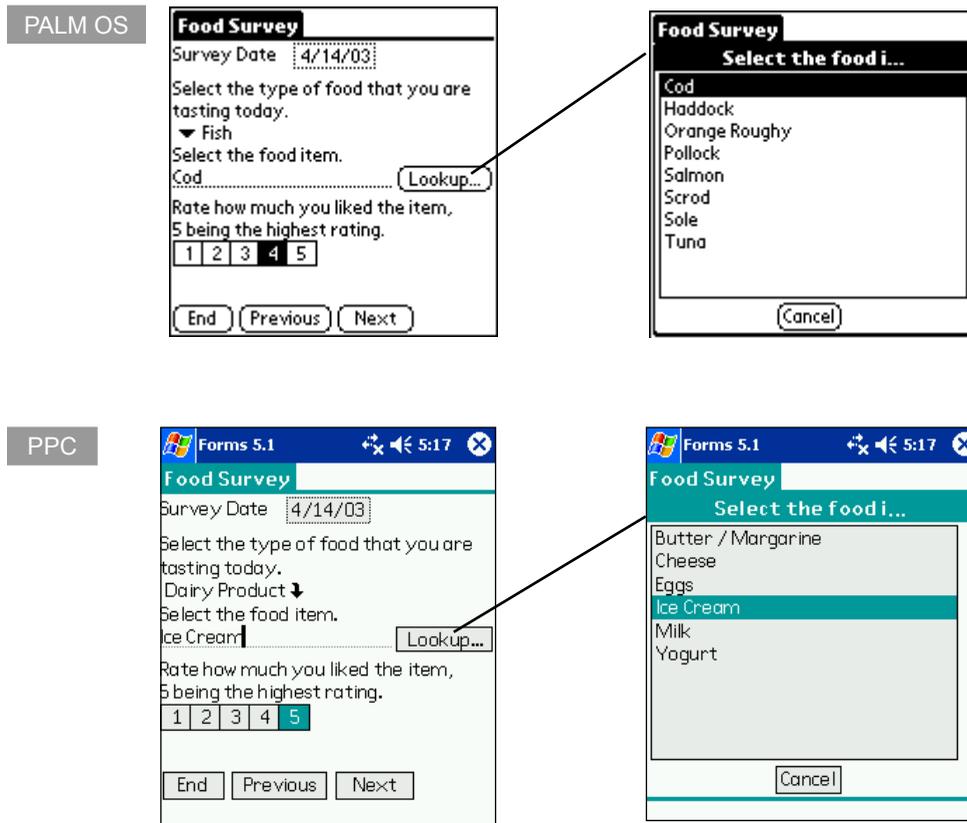
It is also possible to use two fields to create a Cascading Popup/Lookup relationship, in which a selection in a Popup List field (or a Lookup List) determines which Lookup List the user sees in the next field. See *Cascading Lookup*, page 119.

A Lookup List field can also be used to perform a lookup to another form, also called a lookup to a reference form. See page 123.

Cascading Lookup

You can use two fields - either a Popup List and a Lookup List, or two Lookup Lists - to create what is called a cascading lookup. This means that making a selection in one field determines which Lookup List the handheld user sees in the next field on the form.

In the example shown below, selecting a food type in field 2 will display a list of foods within that category in field 3.

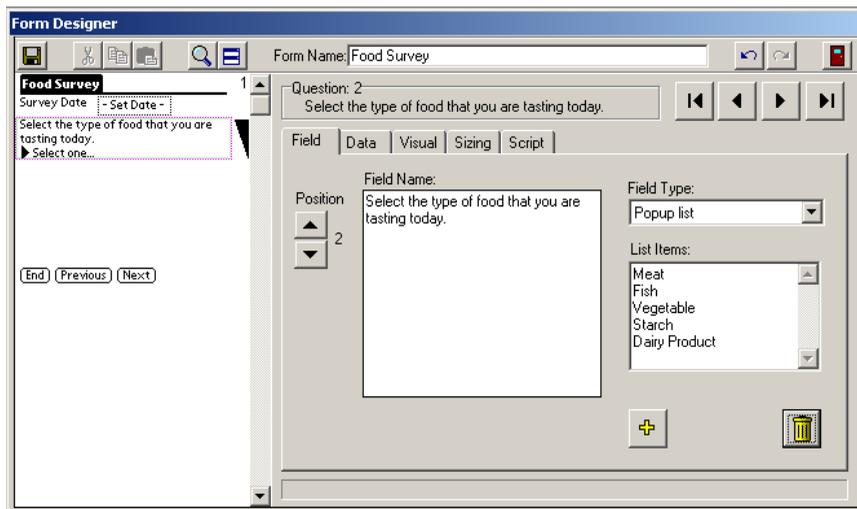


Three steps are involved in creating a cascading relationship between two fields.

Cascading Lookup Step 1: Create the first Popup or Lookup List field

To create a cascading relationship between two fields, make the first field either a Popup List or a Lookup List. (A Lookup List is preferred if you think you will need to add or delete items from the list in the future.)

The list of items in the Popup List or Lookup List are the categories from which the user will initially select.



In the example shown in the picture above, the categories are food types: Meat, Fish, Vegetable, Starch, and Dairy Product.

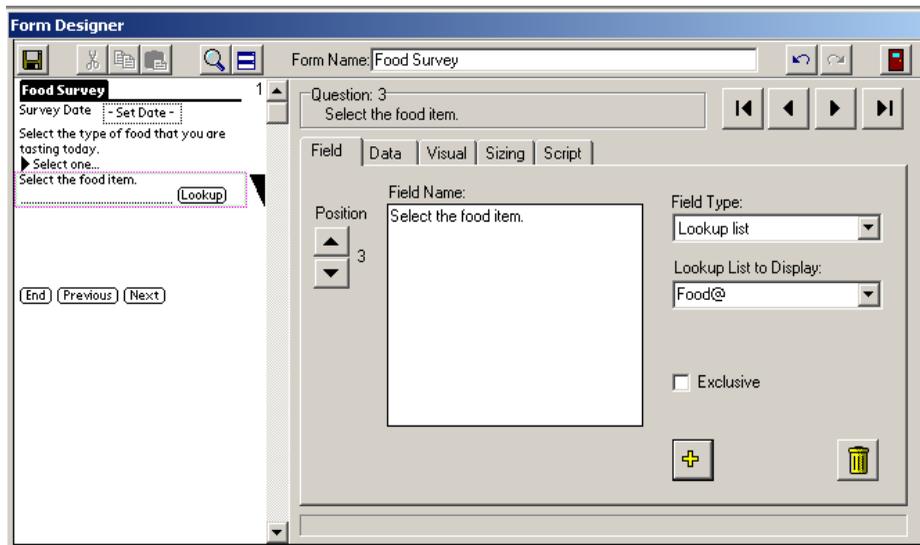
If a Lookup List was being used instead of a Popup List, you would create a Lookup List, for example Food Categories, and the items in the Lookup List would be: Meat, Fish, Vegetable, Starch, and Dairy Product.

Cascading Lookup Step 2: Create the second Lookup List field

Create a second field that immediately follows the first Popup or Lookup field. Make this second field a Lookup List field.

Do not select a Lookup List to display. Instead, in the Lookup List to Display section of the field, type a keyword followed by the @ symbol. The keyword can be a short word.

In the example shown below, the keyword is Food, so the Lookup List to Display is written as Food@



When the handheld user selects an item in the first Popup or Lookup List field, their selection will determine which Lookup List will be displayed in the next field. To figure out which Lookup List to display, the program on the handheld will look for a Lookup List that starts with the keyword and instead of the @ symbol, will insert the name of the item selected in the previous field.

For example, if the items in the previous field are Meat, Fish, Vegetable, Starch, and Dairy Product, and the keyword is Food, then the possible Lookup Lists will be FoodMeat, FoodFish, FoodVegetable, FoodStarch and FoodDairy Product.

The third step in making a Cascading Lookup is to make the specially named Lookup Lists, as shown on the next page.

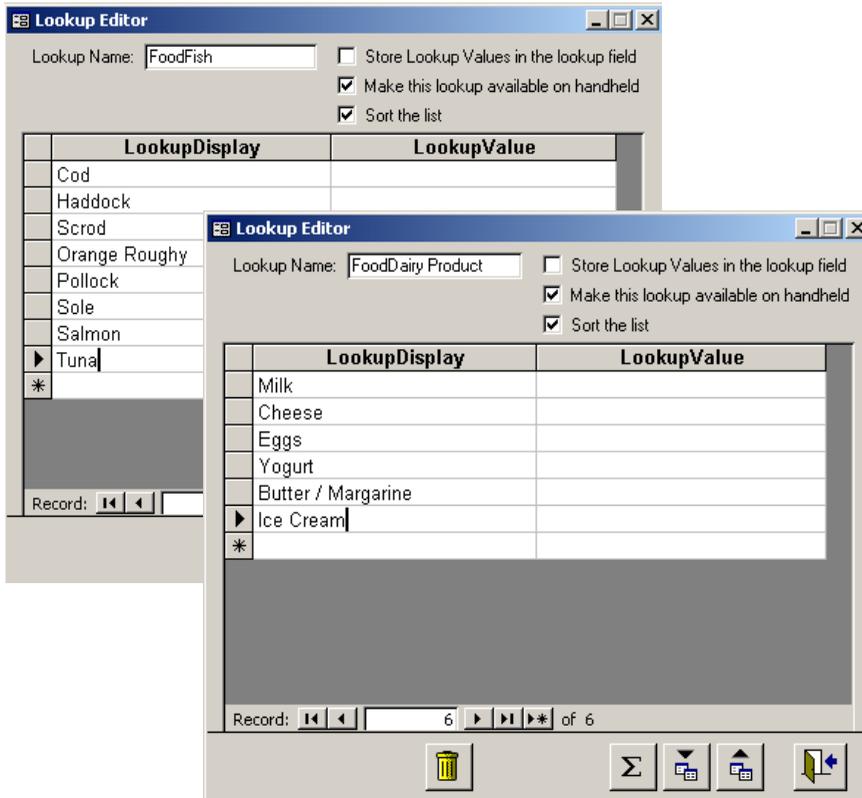
Cascading Lookup Step 3: Create a Lookup List for each category

You will need to create a separate Lookup List for each item in the Popup/Lookup List of the first field. For example, if you have four choices in the first field, you have to make four separate Lookup Lists.

The names of the Lookup Lists are very important in making the cascading relationship work. On the handheld, when the user makes a selection in the first field, the second field is going to display a Lookup List whose name begins with the keyword followed by the item name selected in the first field.

If the keyword is Food and the item selected is Fish, the name of the corresponding Lookup List is FoodFish.

The Lookup List name must match the same spelling and case sensitivity as the keyword and the item name. Do not put a space between the keyword and the item name.

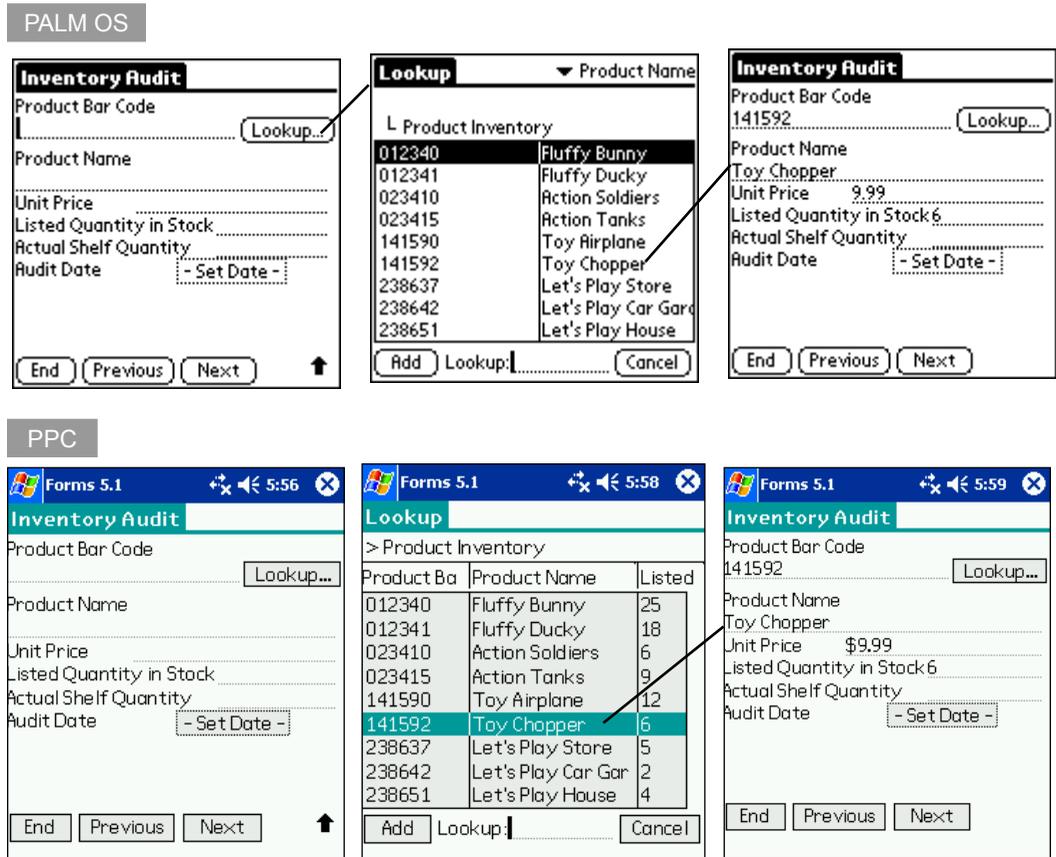


Lookup to Another Form (Lookup to a Reference Form)

A Lookup to Another Form, also called a Lookup to a Reference Form, allows you to jump from one form to a second form, select a record, and copy fields from the second form into the first form.

You might want to use a Lookup to Another Form if you have reference information that you want to maintain in one form, that you want to access from another form.

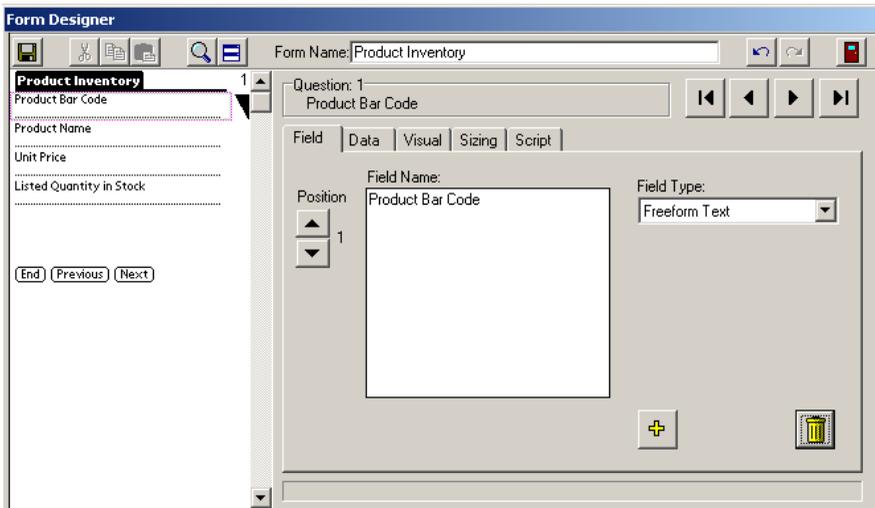
In the example below, the handheld user starts out in one form. Tapping the Lookup button jumps the user to a second (reference) form, where a record is selected. The user is returned to the first form, and several fields on the first form are automatically filled in from the reference form.



Lookup to Another Form Step 1: Create the reference form

To create a Lookup to Another Form, first create the form that will contain the reference information.

NOTE: When typing field names, be mindful about the case sensitivity (that is, the use of upper and lower case letters), and the use of spaces and punctuation. Whatever field names and field types you use on the reference form, you will have to use the exact same field names and field types in the form that is doing the lookup. Signature, Sketch and Image fields cannot copy from one form to another, and should therefore not be used in the reference form.



Freeze the reference form, and populate it with data, either by typing the data into the Edit/View screen (see page 227-228), or by importing the data from another source (see page 242).

Viewer for FORM_ID_235602664 : Form						
	UserName	TimeStamp	ProductBarCo	ProductName	UnitPrice	ListedQuantity
	No one	003 9:48:12 PM	012340	Fluffy Bunny	\$14.99	25
	No one	003 9:48:32 PM	012341	Fluffy Ducky	\$14.99	18
	No one	003 9:49:31 PM	023410	Action Soldiers	\$4.49	6
	No one	003 9:50:11 PM	023415	Action Tanks	\$4.49	9
	No one	003 9:51:38 PM	141590	Toy Airplane	\$9.99	12
	No one	003 9:52:33 PM	141592	Toy Chopper	\$9.99	6
	No one	003 9:53:07 PM	238637	Let's Play Store	\$19.95	5
	No one	003 9:53:41 PM	238642	Let's Play Car C	\$19.95	2
	No one	003 9:54:53 PM	238651	Let's Play Hous	\$19.95	4
*	No one	003 9:56:36 PM				

Lookup to Another Form Step 2: Create the form that does the lookup

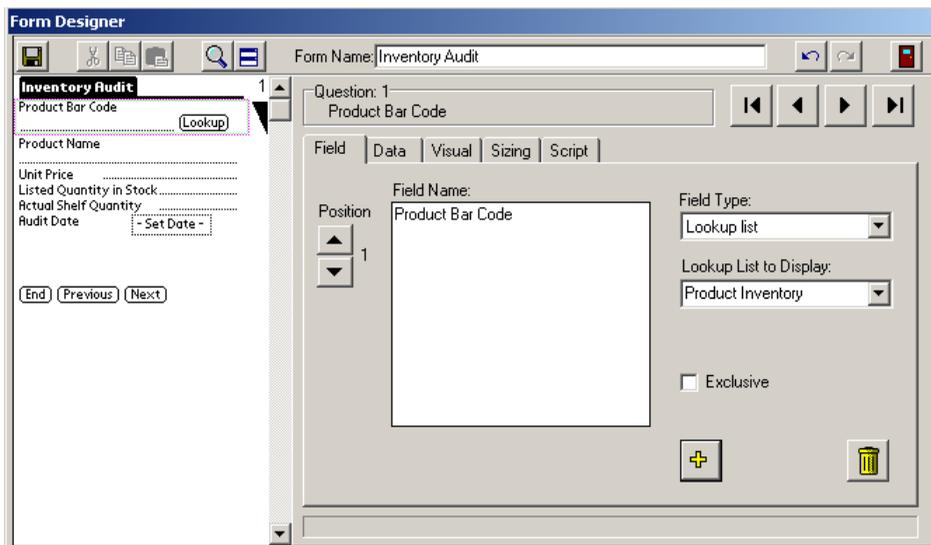
Next, create the form that will be doing the lookup.

In order to do the lookup, one field on this form must be a **Lookup List** field type. In the Lookup List to Display field, instead of selecting a Lookup List, type the exact name of the reference form, using the same upper and lower case letters and spaces.

If you make the Lookup List field match the field name of a Text field on the reference form, up to 50 characters of the Text field will be copied into the Lookup List field, even though the field types are different. This is the only case where field types do not have to match between the reference form and the form doing the lookup.

For all the other fields that you want to copy from the reference form into this form, create a field on this form that has the exact same **field name** and **field type** as the reference form. For instance, if a field is Numeric on the reference form, it will only copy into an identically named Numeric field on this form.

In the picture below, the fields that match the reference form on the previous page are: Product Bar Code, Product Name, Unit Price and Listed Quantity in Stock. When the user performs the lookup, data from the reference form will copy into these four fields.



Special Features



One feature of a Lookup to Another Form is that if you do a lookup and you do not find the record for which you are looking, you can add a new record to the reference form.

This is useful, for example, if you are taking inventory, and you find an item on the shelf that is not listed in the reference form. In this scenario you may want to add the item to the reference form.



To add an item to the form, tap the Add button on the lookup screen. This jumps you to the reference form so that you can add the new record. When you end the new record you will return to the lookup screen and the new record will be in the list.

Note: If each record in the reference form is supposed to be unique, you may want to use a primary key on that form to prevent users from entering the same record twice. For information on primary keys see page 188.



For examples of doing a Lookup to Another Form, see page 457.

Lookup to a Read-Only Reference Form

Reference forms with up to about 5000 records can work reasonably well on the handheld. If you have more than 5000 records, a read-only reference form is a more efficient mechanism to use than a regular reference form.

Advantages:

- A read-only reference form provides for a fast lookup on a handheld by maintaining fixed-width columns for each field.

Limitations:

- Only Text and Numeric fields can be stored in a read-only reference form.
- The individual records of the read-only reference form cannot be viewed or modified on the handheld. The read-only reference form can only be accessed by doing a lookup from another form.

PPC

Note: Pendragon Forms version 5.1 differs from the previous version in that read-only reference forms were not needed in version 5.0, but they can be used on Pocket PC devices in version 5.1.

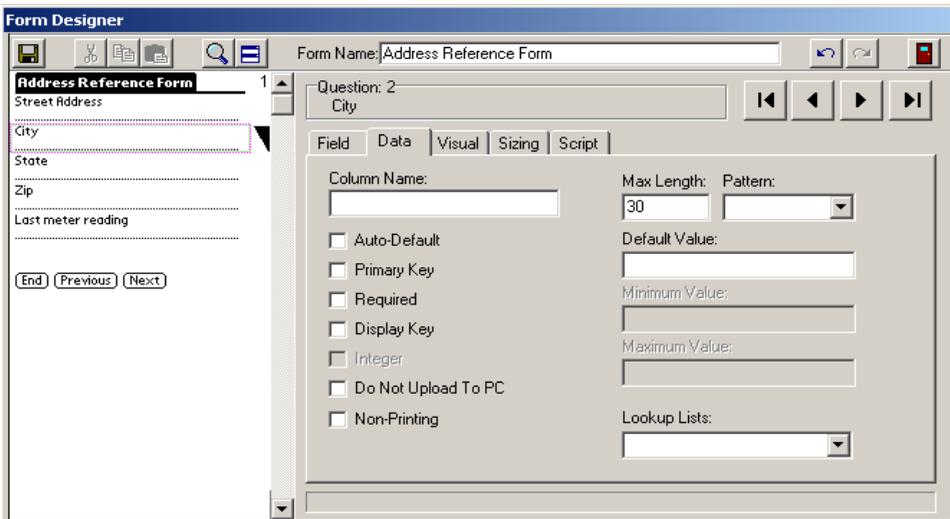
If read-only reference forms are used on a Pocket PC device, the limitations listed above will apply.

Step 1: Creating the Read-Only Reference Form

To create a read-only reference form, create a form with only Text and Numeric fields.

For each Text field, click the Data tab and set the Maximum Length to the smallest maximum that you need. For instance, if a field called Last Name never contains more than 50 characters, set the Max Length to 50.

If you do not set the Maximum Length of each Text field yourself, the default length of 255 characters will be reserved for each Text field. In a fixed-width format, this means that 255 bytes of space will be reserved even if the field only contains one character.



The Display Key field of the reference form will be used to view the records in the reference form when the handheld user performs the lookup from another form.

By default, the Display Key field is the first field on the form.

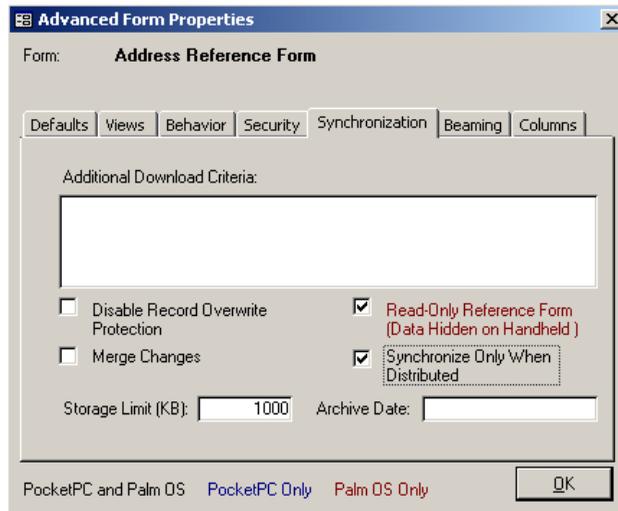
To select a different field on the form to be the Display Key field, click on the field in the Preview Area of the Form Designer window, to select that field. Then click the Data tab and check the Display Key checkbox to make the field the Display Key field.

Step 2: Setting the Advanced Form Properties of the Read-Only Reference form

After you have designed the form and closed the Form Designer window, click the name of the form in the Forms Manager and then click the Properties button. In the Properties window, check the Data Persistence checkbox of Keep a Copy of Records on Handheld.

Click the Advanced Form Properties button. In the Advanced Form Properties window, click the Synchronization tab and do the following:

1. Check the Read-Only Reference Form checkbox to turn this form into a read-only reference form.
2. Optional: If the records in the read-only reference do not change frequently, you can speed up synchronization time by checking the Synchronize Only When Distributed checkbox. That way the form will only be synchronized when you re-distribute the form, for example after adding new records on the PC.



3. If several handheld users are going to share the same read-only reference form, delete the default Additional Download Criteria of `UserName = ##USERNAME##` so that the Additional Download Criteria field is blank. The default criteria causes records to be sent to a handheld device only if the record is specifically assigned to that device.
4. If you have a lot of records, you may want to increase the Storage Limit field to more than 500KB. (Try 1000 or 2000.)

Freeze the reference form and then either click the Edit/View button in the Forms Manager and type in the records for that form, or you can import your data into the database table from an ASCII file - see page 242.

863 : Form						
UserName	TimeStamp	StreetAddress	City	State	Zip	LastMeterR
No one	003 1:42:06 PM	Cherry Lane, #1	Chicago	IL	60606	159264
No one	003 1:42:12 PM	Cherry Lane, #2	Chicago	IL	60606	162971
No one	003 1:42:49 PM	Cherry Lane, #3	Chicago	IL	60606	471523
No one	003 1:43:22 PM	Cherry Lane, #4	Chicago	IL	60606	369888
No one	003 1:43:38 PM	Cherry Lane, #5	Chicago	IL	60606	233521
No one	003 1:43:58 PM	Cherry Lane, #6	Chicago	IL	60606	187532
No one	003 1:44:28 PM	Apple Blossom Trail, #2	Chicago	IL	60607	244122
No one	003 1:44:59 PM	Apple Blossom Trail, #4	Chicago	IL	60607	215093
No one	003 1:47:01 PM	Apple Blossom Trail, #6	Chicago	IL	60607	208974
No one	003 1:47:34 PM	Apple Blossom Trail, #8	Chicago	IL	60607	245687
No one	003 1:48:01 PM	Charles Street, Apt 1A	Chicago	IL	60606	2134
No one	003 1:48:56 PM	Charles Street, Apt 1B	Chicago	IL	60606	2155
No one	003 1:49:41 PM	Charles Street, Apt 1C	Chicago	IL	60606	2987
No one	003 1:50:10 PM	Charles Street, Apt 2A	Chicago	IL	60606	1955
No one	003 1:50:36 PM	Charles Street, Apt 2B	Chicago	IL	60606	2051
No one	003 1:51:05 PM	Charles Street, Apt 2C	Chicago	IL	60606	2113
No one	003 1:51:25 PM	Arthur Ave, Suite A	Chicago	IL	60700	631522

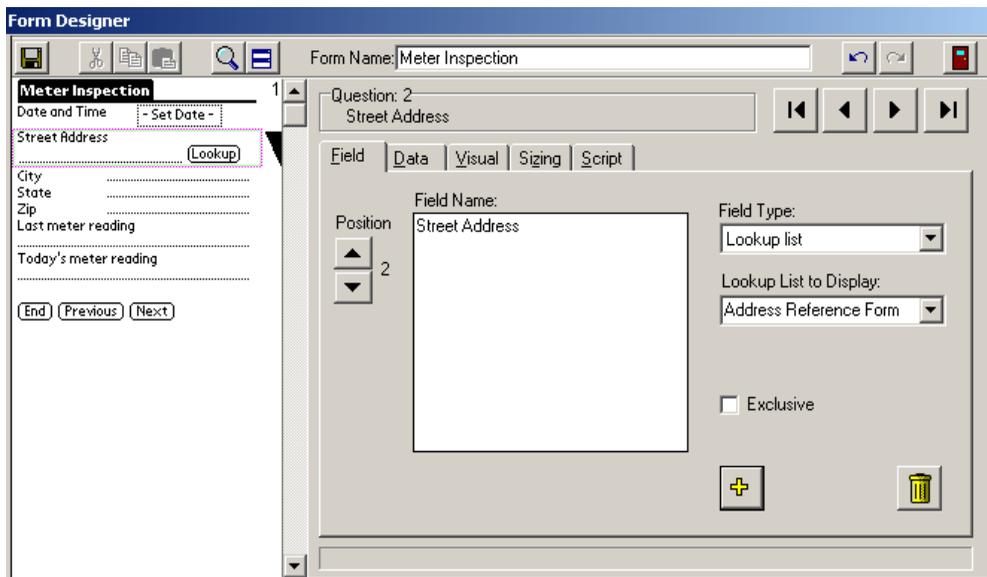
Step 3: Create the form that does the lookup to the Read-Only Reference form

The next step is to create the form that will do the lookup to the read-only reference form.

On this form, create a **Lookup List** field. This field will be used to perform the lookup to the read-only reference form. For the Lookup List to Display, instead of selecting a Lookup List, type the exact name of the read-only reference form, using the same upper and lower case letters and spaces as in the name of the read-only reference form.

If you make the Lookup List field match the field name of a Text field on the read-only reference form, up to 50 characters of the Text field will be copied into the Lookup List field, even though the field types are different. This is the only case where field types do not have to match between the read-only reference form and the form doing the lookup. In the picture below, data from a Text field called Street Address in the read-only reference form will copy into the Lookup List field called Street Address in the form doing the lookup.

For all other fields that you want to copy from the read-only reference form, create fields on the current form with the exact **same field names and field types (Text or Numeric)** as the fields on the read-only reference form.



Freeze the form. Distribute both this form and the read-only reference form to the handheld, and synchronize.

Features of a Lookup to a Read-Only Reference Form

On the handheld, tap the Lookup button in the Lookup List field.

A blank lookup screen is first displayed.

Enter a character in the lookup field to display a maximum of 100 records that match that character.

The Display Key field of the reference form is shown. By tapping the arrow in the upper right corner of the screen, you can select to view a second field.

To refine the search, enter more characters in the search string until you see the record you need.

Select a record from the list to copy fields from the reference form into the current form.

If your handheld device has a built-in bar code scanner, you can scan a bar code in the Lookup List field and use a script to lookup to the read-only reference form. See Chapter 15, *Using Bar Codes*, page 373.

Subform Field

If you need to collect information that is related, for example: customer contact information and individual customer visits, you may want to create two forms: a 'parent' form and a 'child' form. The 'child' form is also called a subform.

Parent and subforms are used when you have a one-to-many relationship: for every one record in the parent form, there will be many records in the subform.

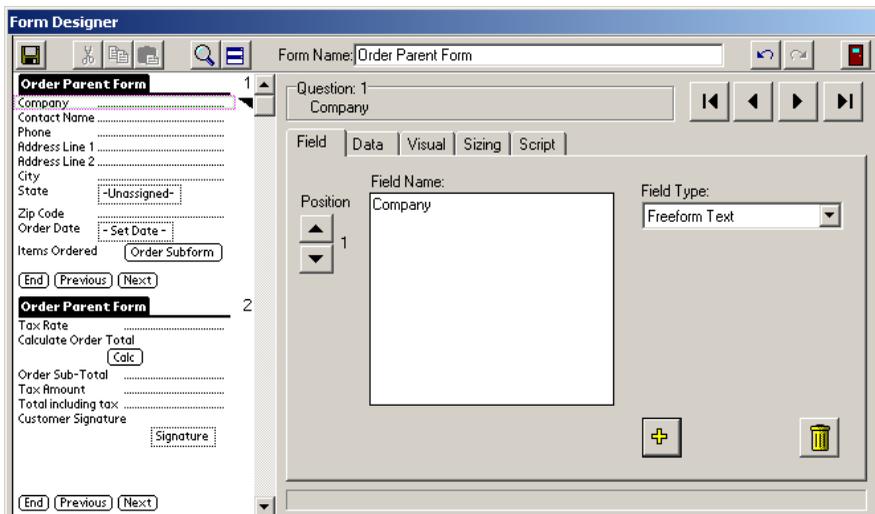
Pendragon Forms is not a relational database on the handheld, but a Subform field on a parent form makes it appear to the user that the parent form and subform are linked, by allowing the handheld user to jump from the parent form to the subform to fill in the subform record. Ending the subform record returns the user to the parent form. On the PC, data for the parent form and subform are stored in separate database tables.

Subform Step 1: Create a Parent Form

The parent form should be created first.

To connect the parent form to the subform, make at least one field on both the parent form and the subform identical in field name and field type. (Typically the first field.)

If you need to copy data from the parent form to the subform every time you create a new subform record, you can make up to the first 10 fields on the parent form and the subform identical in field name and field type.

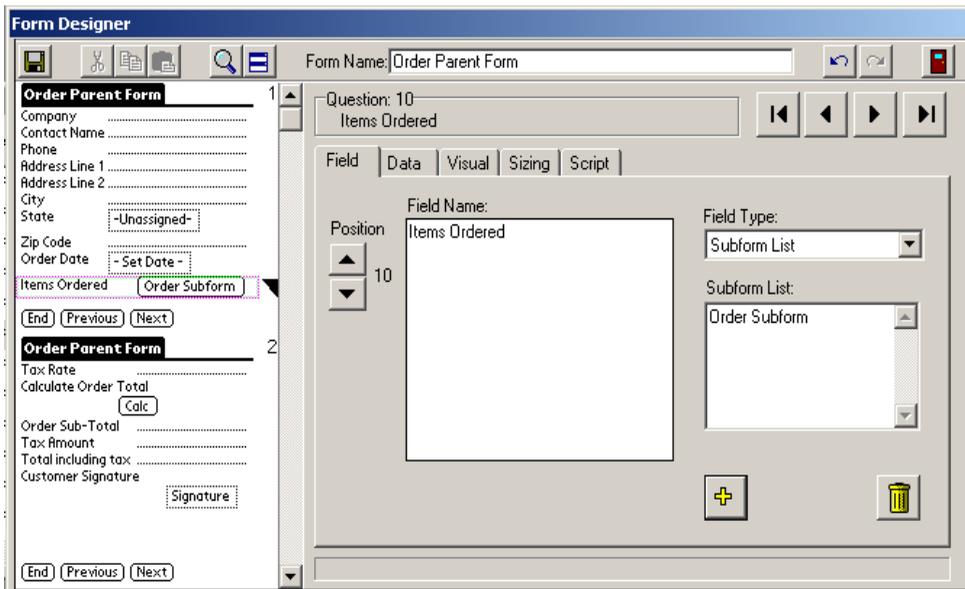


To allow the handheld user to jump from the parent form to the subform, add a Subform List field to the parent form.

The Subform field should occur after all the fields that you want to copy from the parent form to the subform record. That way the user will have a chance to fill in all the necessary fields before jumping to the subform.

In the Form Designer window, when you select Subform List as the field type, a Subform List section appears. Type the name of the form that will be used as the subform.

In the picture below, the name of the subform is going to be Order Subform.



Subform Step 2: Create a Subform

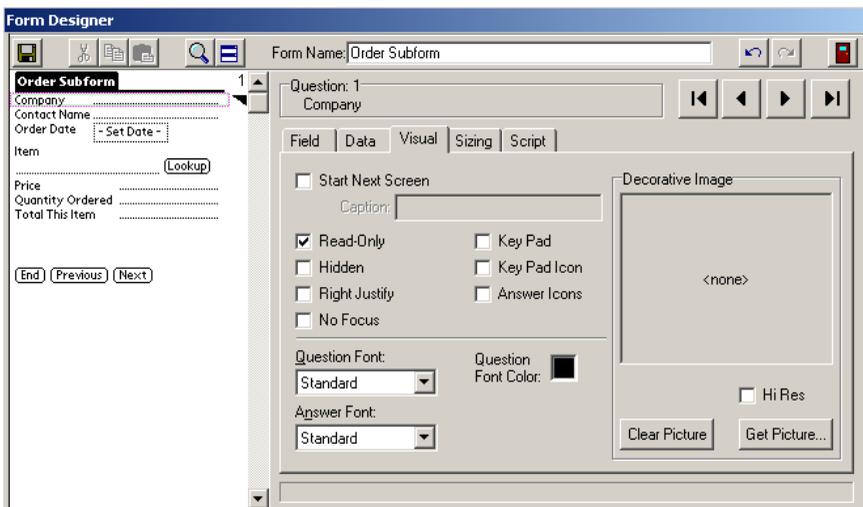
Create a new form to be the child form or subform. When you create the subform, make sure that the name of the form is identical to the name that you referenced in the Subform List field on the parent form (including upper and lower case letters and spaces).

At least one field on the subform must be identical in field name and field type to one of the first ten fields on the parent form. The matching field or fields provide the link between parent form and subform.

As a precaution, it is recommended that all the fields on the subform that match fields on the parent form should be made Read-Only on the subform. If you change one of these fields on the subform you will lose the link to the parent form, since Pendragon Forms is looking for an exact match between parent and subform in these fields. For each of the matching fields on the subform, click the Data tab in the Form Designer window and check the Read-Only checkbox.

If you want information from more than one field to copy from parent to subform, you can make up to 10 fields on the subform match the first 10 fields on the parent form. The field names and field types must be identical from parent to subform.

In the picture below, three fields match the parent form: Company, Contact Name and Order Date. The remaining fields are unique to the subform and are not on the parent form.

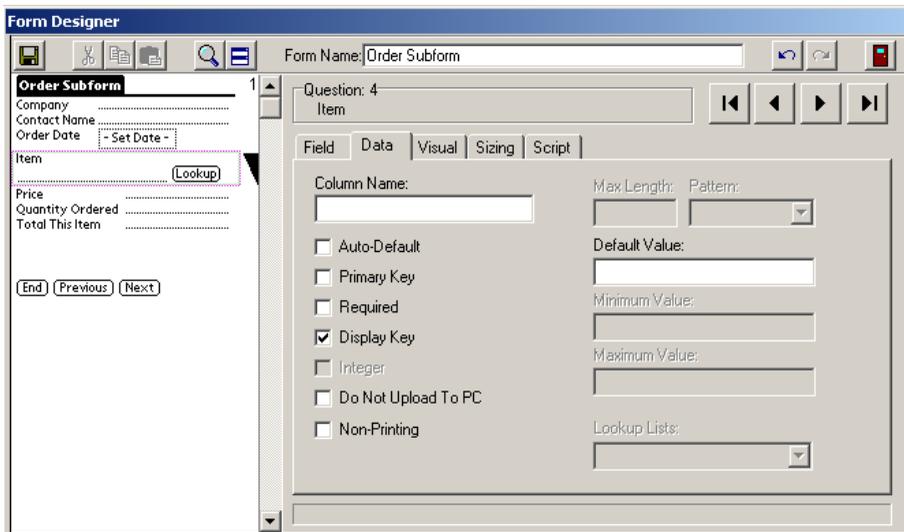


Select a field on the subform that will be the Display Key field, that is, the field that will be used to display subform records when the handheld user jumps from the parent to the subform.

Note that when you jump from the parent form to the subform, the first matching field between parent and subform is displayed at the top of the list of subform records. Therefore, you may want to choose as the Display Key field, a field that will help you tell the difference between subform records.

In the Form Designer window, click in the Preview Area on the field that you want to use as the Display Key field. On the Data tab, check the Display Key checkbox to make that field the Display Key field.

In the picture below, the Item field on the subform is used as the Display Key field, because each subform record will contain a different item.



Features of Subforms

Order Parent Form

Company Polar Explorations
 Contact Name B. Bear
 Phone 847-555-1505
 Address Line 1 3 Snowy Crescent
 Address Line 2
 City Barrow
 State AK
 Zip Code 99723
 Order Date 4/16/03

Items Ordered

Subform ▼ Quantity Ord

Polar Explorations
 L Order Subform

Toy polar bear	5
Toy sea otter	1

Order Subform

Company Polar Explorations
 Contact Name B. Bear
 Order Date 4/16/03

Item

Price
 Quantity Ordered
 Total This Item

When using a parent form and a subform on the handheld, the user must always enter data starting from the parent form.

This means that if the parent record already exists, the user will need to review the parent record in order to add a new subform record.

To access the subform, tap the subform field on the parent form.

Existing subform records will be displayed according to the Display Key field of the subform.

Tap an existing record to display it, or tap the Add button to add a new subform record.

When you create a new subform record, fields that match from the parent record are copied into the subform record.

After filling in the remaining subform fields, tap the End button to return to the list of subform records. Tap the Done button to return to the parent form.

If you have any problems with getting subforms to work, see Appendix B, *Troubleshooting*, page 536.

Forms 5.1 2:42

Order Parent Form

Company Polar Explorations
 Contact Name B. Bear
 Phone 847-555-1505
 Address Line 1 3 Snowy Crescent
 Address Line 2
 City Barrow
 State AK
 Zip Code 99723
 Order Date 4/16/03

Items Ordered

Forms 5.1 2:43

Subform

Polar Explorations > Order Subform

Item	Price	Quan	Total TH
Toy polar bear	\$15.00	5	\$75.0
Toy sea otter	\$20.00	1	\$20.0

Forms 5.1 2:44

Order Subform

Company Polar Explorations
 Contact Name B. Bear
 Order Date 4/16/03

Item

Price
 Quantity Ordered
 Total This Item

Special Features

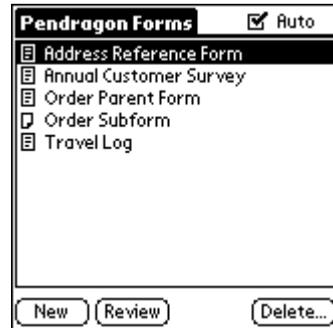
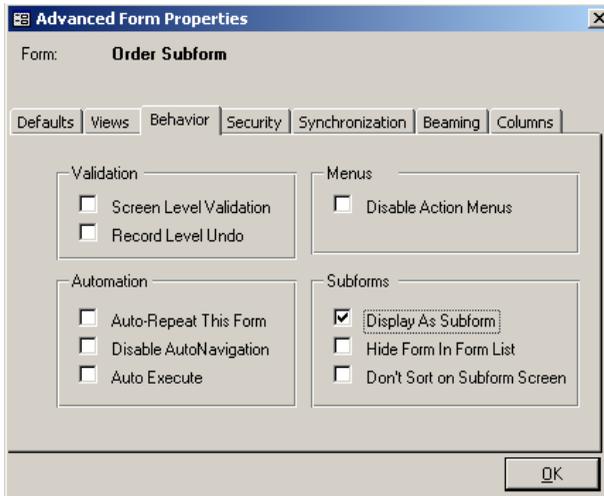
On the handheld, it is necessary to enter all data starting from the parent form. By default, however, both the names of the parent form and the subform will be visible on the handheld.

To make it easier for the handheld user to use only the parent form, you can choose to mark a subform so that the user cannot enter new records starting directly from the subform. Or you can hide the subform name from the list of forms entirely.

In the Pendragon Forms Manager, click the name of the subform, then click the Properties button. In the Properties window, click the Advanced Form Properties button.

In the Advanced Form Properties window, click the Behavior tab.

- Either: Check the Display as Subform checkbox to mark the form as being a subform.
- Or: Check the Hide Form in Forms List checkbox so that the subform does not appear in the list of forms on the handheld.



Here a form called Order Subform is marked as a subform on the handheld. Users cannot create new records by tapping on this form. Instead, the users have to tap on the parent form.

TIP: If you have any problems getting subforms to work, see Appendix B, *Troubleshooting*, page 536.

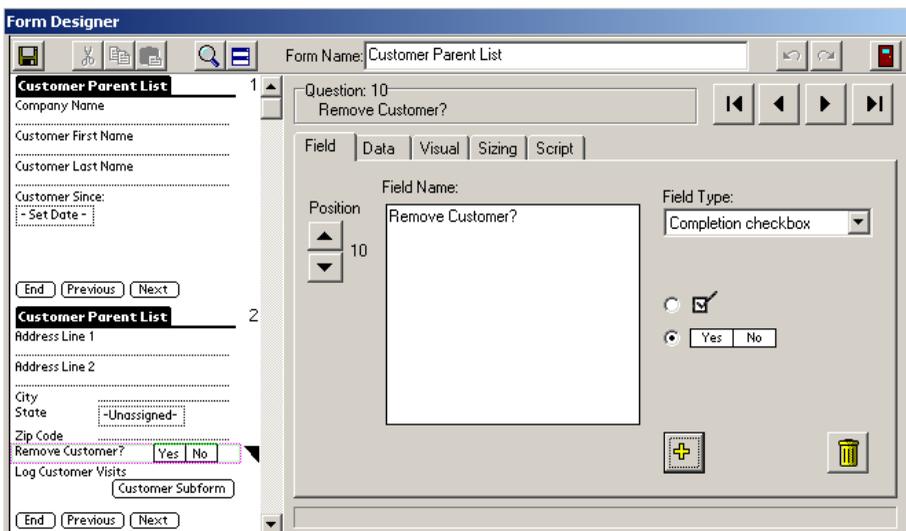
Cascading Completion Checkbox from Parent to Subform

If you have a parent form with several subform records, you might want to have the ability to remove the parent record from the handheld, and automatically remove all the associated subform records from the handheld as well. To achieve this, a Completion Checkbox field can be used on both the parent form and the subform.

Other fields only copy from the parent to the subform when a new subform record is created. A Completion Checkbox can cascade from the parent form to already existing subform records. This means that the parent form and subform records can be kept on the handheld indefinitely, and then at a later date the Completion Checkbox can be checked on the parent record. This value will then be copied to all the related subform records. All records marked as completed are removed from the handheld during synchronization.

Cascading Completion Checkbox Step 1: Create the Parent form

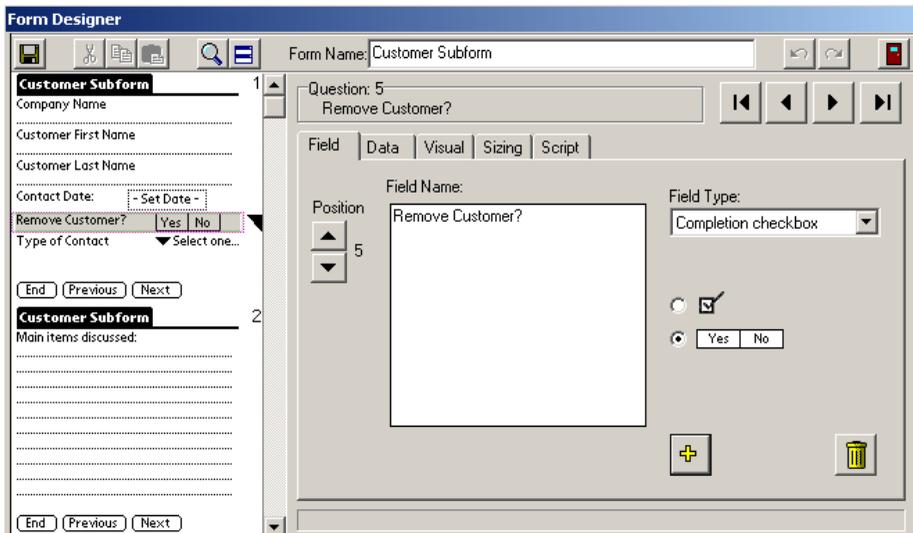
First create the parent form. Add a Completion Checkbox field within the first ten fields of the parent form. In the picture below, Field 10 is the Completion Checkbox field. Also create a Subform List field on the parent form, and type the name of the form that will be used as the subform.



Cascading Completion Checkbox Step 2: Create the Subform

Next, create the subform. For the form name, type the exact name referenced by the parent form. As with all subforms, one or more fields in the first ten fields must exactly match the field names and field types that you want to copy from the parent form. The Completion Checkbox field must be one of these first ten fields on the subform.

In the picture below, Field 5 is the Completion Checkbox field on the subform. You can choose to hide the Completion Checkbox field to prevent the handheld user from accidentally checking the box. To hide the field, click on the name of the field in the Preview Area, then click the Data tab and check the Hidden checkbox.



Cascading Completion Checkbox Step 3: Set Data Persistence on both forms

For a Completion Checkbox to work, you must also go to the Data Persistence section of the **Form Properties** window, and select **Keep Incomplete records on the Handheld**. (See page 198.) To access the Form Properties window, click on the name of the form in the Forms Manager and then click the Properties button.

Set Data Persistence to **Keep Incomplete records on the Handheld** for **both** the parent form and the subform. Then freeze and distribute both parent form and subform.

Features of Cascading Completion Checkbox from Parent to Subform

Customer Parent List

Address Line 1
3440 Hibiscus Drive
Address Line 2

City Deerfield
State FL
Zip Code 51235

Remove Customer? Yes No

Log Customer Visits

Customer Subform

End Previous Next

On the handheld, leave the Completion Checkbox blank or set to No for as long as the parent record needs to stay on the handheld.

Subform Type of Conto

Teeny Tiny Toys
L Customer Subform

5/10/04	Phone Call to custo
8/23/04	Phone Call from cust
9/6/04	Meeting at custom

Add Lookup: Done

Tap the Subform button to jump to the subform review screen.

Tap an existing record to view and edit that record, or tap the Add button to create a new subform record.

Customer Subform

Company Name
Teeny Tiny Toys
Customer First Name
Jane
Customer Last Name
Cambert
Contact Date: 1/18/05

Type of Contact Meeting at

End Previous Next

In a subform record, the user does not see that there is a Completion Checkbox field, as this field has been hidden.

Customer Parent List

Address Line 1
3440 Hibiscus Drive
Address Line 2

City Deerfield
State FL
Zip Code 51235

Remove Customer? Yes No

Log Customer Visits

Customer Subform

End Previous Next

At a later date, if the user reviews the parent record and checks Yes in the Completion Checkbox field, then the parent record, and all related subform records, will be removed from the handheld during the next synchronization.

Forms 5.1 2:51

Customer Parent List

Address Line 1
3440 Hibiscus Drive
Address Line 2

City Deerfield
State FL
Zip Code 51235

Remove Customer? Yes No

Log Customer Visits

Customer Subform

Forms 5.1 2:46

Subform

Teeny Tiny Toys > Customer

Contact Date: Type of Contact

5/10/04	Phone Call to custom
8/23/04	Phone Call from cust
9/6/04	Meeting at customer

Add Lookup: Done

Forms 5.1 2:48

Customer Subform

Company Name
Teeny Tiny Toys
Customer First Name
Jane
Customer Last Name
Cambert
Contact Date: 1/18/05

Type of Contact Meeting at cu

Forms 5.1 2:52

Customer Parent List

Address Line 1
3440 Hibiscus Drive
Address Line 2

City Deerfield
State FL
Zip Code 51235

Remove Customer? Yes No

Single Subform Field

A Single Subform field is a variation on the regular subform. Whereas you can enter many records in a subform, with a Single Subform you can only enter one record.

A Single Subform offers a way to lengthen a form. The maximum number of fields that you can have on a form is 250 fields. If you need to more than 250 fields, you can split the form into a parent form and one or more single subforms.

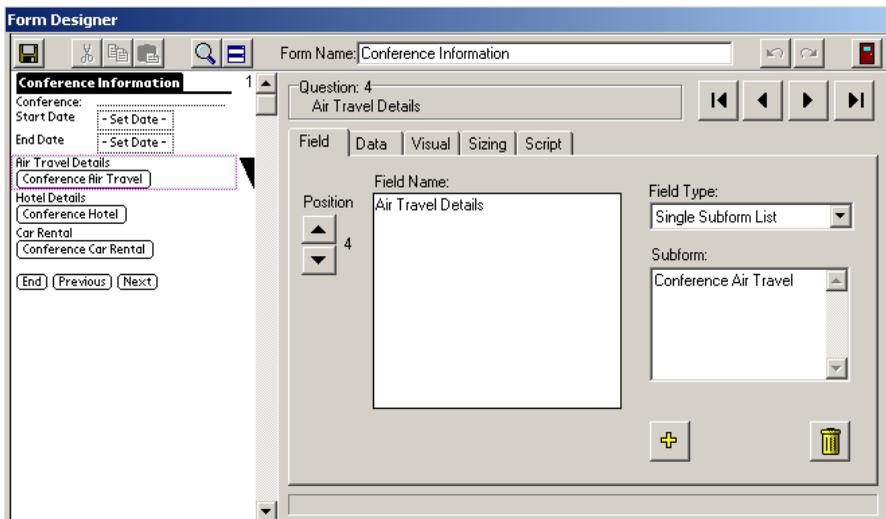
All data entry is done from the parent form. A Single Subform field on the parent form allows you to jump to a subform to enter data once, and then return to the parent form.

Step 1: Create a Parent Form

The same rules that apply to subforms apply to single subforms:

- At least one field in the first ten fields on the parent form must be identical in field name and field type on the parent form and on each subform.
- To copy data from the parent form to the single subform, you can make up to the first 10 fields match on both parent and subform.
- A separate database table is created for the parent and each subform.

On the parent form, create a Single Subform field. In the Subform List section, type the name of the form that will be used as the single subform.



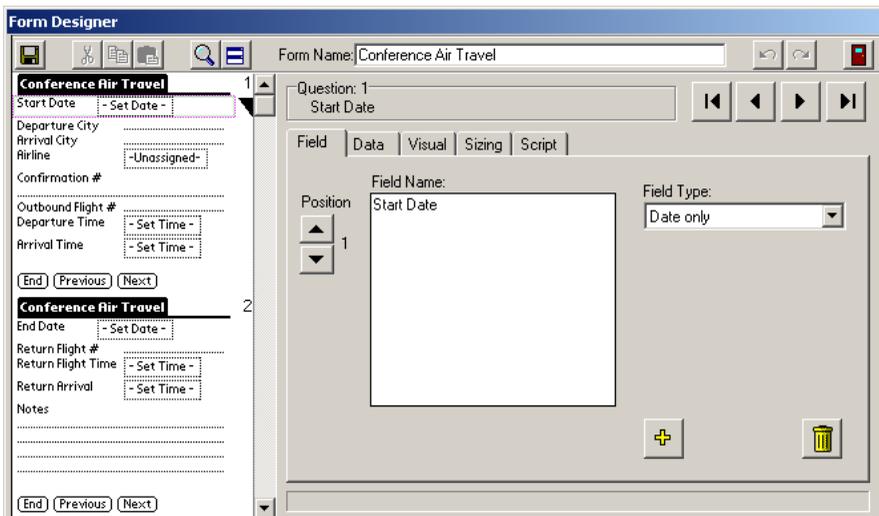
Step 2: Create a Single Subform

Create the form that will be used as the single subform. Take care so that the name of the form exactly matches the name that you typed in the Single Subform field of the parent form. Use the same upper and lower case letters and spaces.

At least one field on the single subform must exactly match the field name and field type of one of the first 10 fields of the parent form. If you need to copy data from the parent to the single subform, you can make up to the first ten fields match in both parent form and single subform.

Freeze the parent form and each single subform, then distribute to the handheld.

In the picture below, the Start Date and End Date fields are copied from the parent form into the single subform.



Features of Single Subforms

On the parent form on the handheld, when you tap in a Single Subform field, you will jump to the single subform.

Any of the first ten fields that match in both parent and single subform will be copied from the parent form into the single subform.

The handheld user can finish filling in the single subform, and then press the End button to jump back to the parent form.

PALM OS

The image shows two screenshots from a PALM OS handheld device. The left screenshot is titled "Conference Information" and contains the following fields: Conference: Handheld Seminar, Start Date: 5/5/03, End Date: 5/8/03, Air Travel Details (with a subform button "Conference Air Travel"), Hotel Details (with a subform button "Conference Hotel"), and Car Rental (with a subform button "Conference Car Rental"). At the bottom are buttons for "End", "Previous", and "Next". The right screenshot is titled "Conference Air Travel" and contains the following fields: Start Date: 5/5/03, Departure City: ORD, Arrival City: SFO, Airline: UA, Confirmation #: 87654A, Outbound Flight #: 12, Departure Time: 10:00 am, and Arrival Time: 12:31 pm. It also has "End", "Previous", and "Next" buttons at the bottom. A line connects the "Conference Air Travel" button in the parent form to the subform.

PPC

The image shows two screenshots from a PPC (Personal PC) window titled "Forms 5.1". The left screenshot is titled "Conference Information" and contains the following fields: Conference: Handheld Seminar, Start Date: 5/5/03, End Date: 5/8/03, Air Travel Details (with a subform button "Conference Air Travel"), Hotel Details (with a subform button "Conference Hotel"), and Car Rental (with a subform button "Conference Car Rental"). At the bottom are buttons for "End", "Previous", and "Next". The right screenshot is titled "Conference Hotel" and contains the following fields: Start Date: 5/5/03, End Date: 5/8/03, Name of Hotel: Marriott, Address: 4 Apricot Drive, Phone: 415-555-9822, Confirmation #: JMB52, and "End", "Previous", and "Next" buttons at the bottom. A line connects the "Conference Hotel" button in the parent form to the subform.

The special features that apply to subforms also apply to single subforms. See page 108.

TIP: If you have any problems with single subforms, refer to Appendix B, *Troubleshooting*, page 536.

Signature Field

A Signature field allows the handheld user to capture a person's signature.

In the Form Designer window, when you select a Signature as the field type, a Legal terms field appears.

- If you want to indicate to the customer that signing implies agreement to certain terms, type the terms in the Legal Terms field.
- If there are no terms to agree to, leave the Legal Terms field blank.

You may need to consult with a legal professional to determine if it is sufficient for you to display legal terms on the handheld.

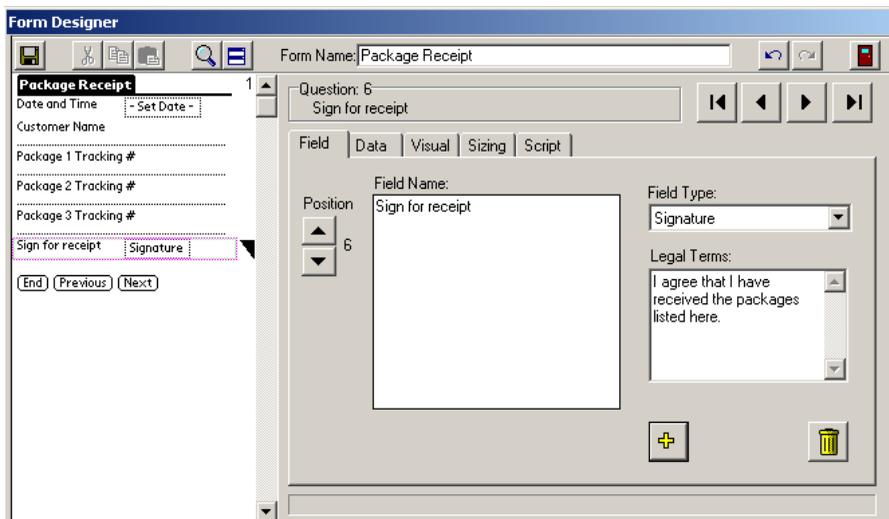
IMPORTANT: Signature fields and legal terms do not print directly from the handheld. Also, signatures created on one platform (Palm OS or Pocket PC) cannot be sent to another platform.

PALM OS

Signature fields that are created on Palm OS devices are 160 pixels wide x 50 pixels high. Signatures created on a Palm OS device cannot be viewed or edited on a Pocket PC device.

PPC

Signature fields that are created on Pocket PC devices are 240 pixels wide x 75 pixels high. Signature fields that are created on a Pocket PC device cannot be viewed or edited on a Palm OS device. Also, Signature fields created in Pendragon Forms version 5.0 on a Pocket PC device cannot be viewed or edited in Pendragon Forms 5.1 on a Pocket PC device.



Features of Signatures

Package Receipt

Date and Time 4/21/03

Customer Name Paulette Bunting

Package 1 Tracking # 032563342

Package 2 Tracking # 032598716

Package 3 Tracking #

Sign for receipt Unsigned.

End Previous Next

On the handheld, tap the Sign button or tap in the Unsigned box in a Signature field in order to sign.

Terms

Sign for receipt

I agree that I have received the packages listed here.

Agree Cancel

If the customer has to read a statement of the terms that are being agreed to, these terms will be displayed. Tap the Agree button to advance to the screen that the customer can sign.

Signature

Sign for receipt

PLEASE PRESS LIGHTLY AND SIGN SLOWLY Clear...

P. Bunting

OK Cancel

If the signature is satisfactory, tap the OK button. To re-capture the signature, tap the Clear button.

To leave this screen without saving any changes, tap Cancel.

Package Receipt

Date and Time 4/21/03

Customer Name Paulette Bunting

Package 1 Tracking # 032563342

Package 2 Tracking # 032598716

Package 3 Tracking #

Sign for receipt Signed.

End Previous Next

A Signature field displays the message "Signed" if a signature has been entered, and "Unsigned" if the field is blank.

Forms 5.1 3:00

Package Receipt

Date and Time 4/21/03

Customer Name Paulette Bunting

Package 1 Tracking # 032563342

Package 2 Tracking # 032598716

Package 3 Tracking #

Sign for receipt Unsigned.

End Previous Next

Forms 5.1 3:01

Terms

Sign for receipt

I agree that I have received the packages listed here.

Agree Cancel

Forms 5.1 3:02

Signature

Sign for receipt

PLEASE PRESS LIGHTLY AND SIGN SLOWLY Clear...

P. Bunting

OK Cancel

Viewing a Signature on the PC

After synchronization, each signature is stored as a Long Binary Object in the Pendragon Forms Manager database.

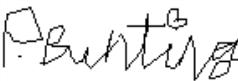
There are two ways to view a signature in the database:

1. Click the name of the form in the Forms Manager database, and then click the Edit/View button to view the records. Signature fields will say Long Binary Object. Double-click in a Long Binary Object cell to open Windows Paint to view the image. The image can then be saved as a bitmap (.BMP) file if necessary.

OR

2. Click the name of the form in the Forms Manager database. Hold down the CTRL key and click the Edit/View button. This displays an Access form in Form View, showing one record at a time. The signature will be visible. If you click the Open button next to the signature, you can open Windows Paint to save the image as a bitmap (.BMP) file if needed.

The screenshot shows a window titled "Viewer for FORM_ID_235845970 : Form". The window contains a list of fields and their values:

RecordID	0
UnitID	0
UserName	Sarah Jane
TimeStamp	4/21/2003 4:57:47 PM
Date and Time	4/21/2003
Customer Name	Paulette Bunting
Package 1 Tracking #	032563342
Package 2 Tracking #	032598716
Package 3 Tracking #	
Sign for receipt	

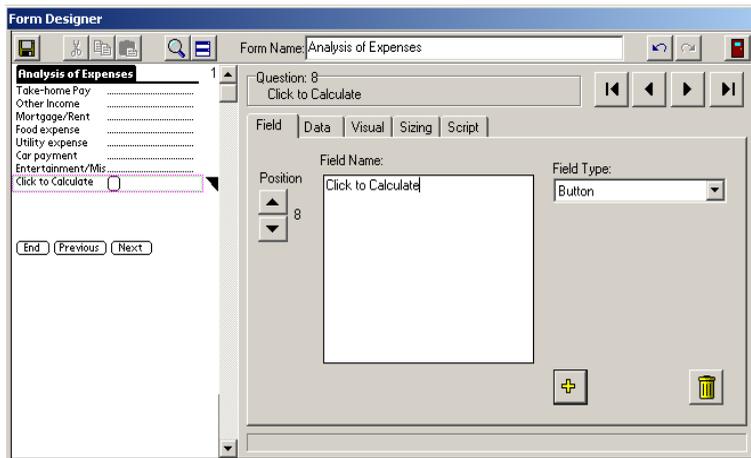
Below the signature field is an "Open..." button. At the bottom of the window, there is a "Record:" label followed by navigation buttons and the text "1 of 2".

Button Field

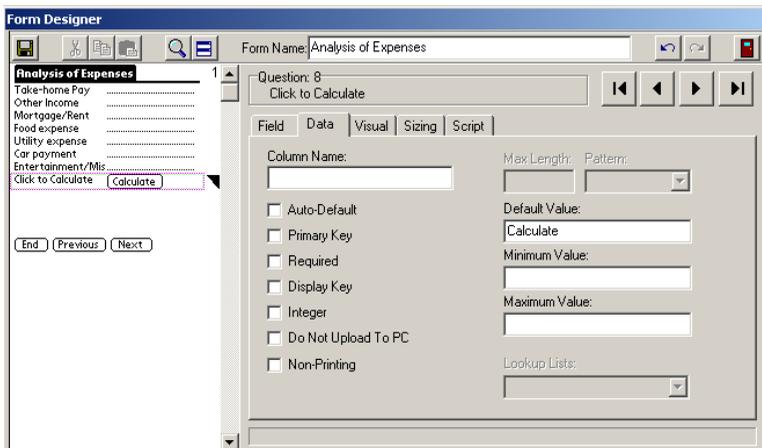
A Button field allows the handheld user to click a button in order to perform an action.

Button fields are always used in conjunction with **click:** event scripts. The **click:** event determines what happens when the handheld user taps the button. See Chapter 14, *Scripting Examples*, page 364.

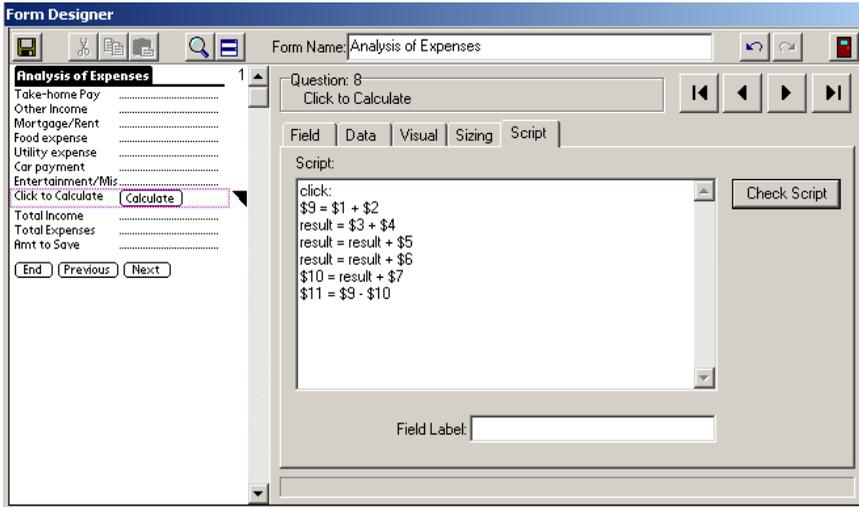
Actions that can be performed using buttons include doing a lookup to the handheld Address Book (Palm OS only), performing calculations, jumping to subforms, and printing.



To add a label to a button, click the Data tab and type a name for the button in the Default Value field. The limit is 11 characters.



The Scripting tab of the Button field is where you write the **click:** event script that will run when the handheld user taps the button. See Chapter 13, *Scripting Reference*, page 252.



Features of Button Fields

Analysis of Expenses	
Take-home Pay	\$3,450.00
Other Income	\$100.00
Mortgage/Rent	\$1,800.00
Food expense	\$400.00
Utility expense	\$350.00
Car payment	\$460.00
Entertainment/Mi	\$300.00
Click to Calculate	<input type="button" value="Calculate"/>
Total Income	
Total Expenses	
Amt to Save	

On the handheld, tapping a button performs the action associated with the button.

In the example here, the user enters values in all the fields that appear before the button.

Analysis of Expenses	
Take-home Pay	\$3,450.00
Other Income	\$100.00
Mortgage/Rent	\$1,800.00
Food expense	\$400.00
Utility expense	\$350.00
Car payment	\$460.00
Entertainment/Mi	\$300.00
Click to Calculate	<input type="button" value="Calculate"/>
Total Income	\$3,550.00
Total Expenses	\$3,310.00
Amt to Save	\$240.00

Tapping the button labeled 'Calculate' performs calculations based on the data that the user has entered.

See page 365 for details on implementing this example.

Analysis of Expenses	
Take-home Pay	\$3,450.00
Other Income	\$100.00
Mortgage/Rent	\$1,800.00
Food expense	\$400.00
Utility expense	\$350.00
Car payment	\$460.00
Entertainment/Mi	\$300.00
Click to Calculate	<input type="button" value="Calculate"/>
Total Income	
Total Expenses	
Amt to Save	

Analysis of Expenses	
Take-home Pay	\$3,450.00
Other Income	\$100.00
Mortgage/Rent	\$1,800.00
Food expense	\$400.00
Utility expense	\$350.00
Car payment	\$460.00
Entertainment/Mi	\$300.00
Click to Calculate	<input type="button" value="Calculate"/>
Total Income	\$3,550.00
Total Expenses	\$3,310.00
Amt to Save	\$240.00

Slider Field

A Slider field allows the handheld user to select a number by tapping or sliding a marker along a horizontal line.

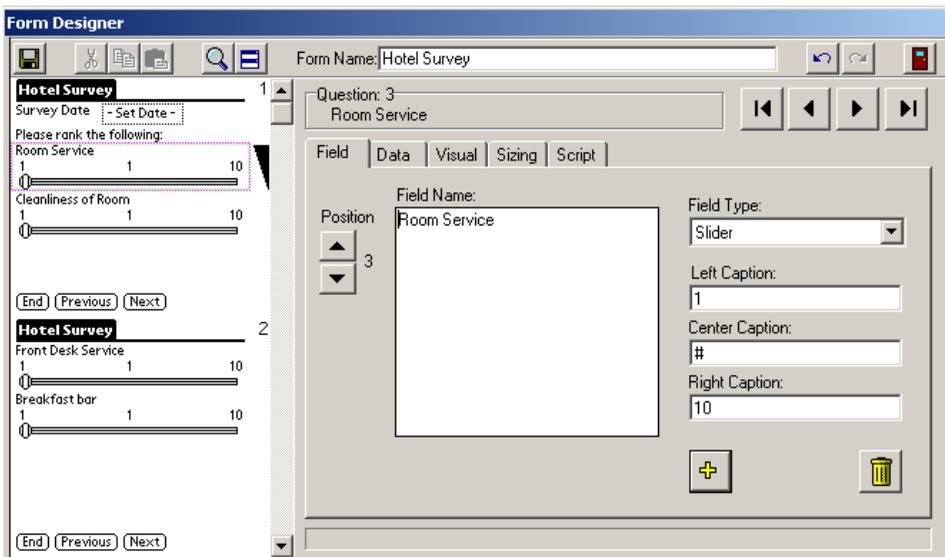
The Slider is only visible in Layout View and Field View. In Record View, the field looks like a Numeric field.

Internally, a Slider field is a Numeric field that accepts integers only, and has a Minimum and Maximum numeric range set.

In the Form Designer window, create a field with the Slider field type.

You can label the scale by entering captions for the left, center and right of the scale. A # symbol as the center caption will display the numeric value of the marker position. To display numeric values for ends of the scale, type the minimum value that you want to use in the Left Caption field, and type the maximum value that you want to use in the Right caption field.

If you enter the # symbol in each of the left, center and right caption fields, you will see labels across the slider for every number in the range. This is only recommended if you have a small range, such as 0-5 or 1-10.



Click the Data tab to set the range of the Slider scale.

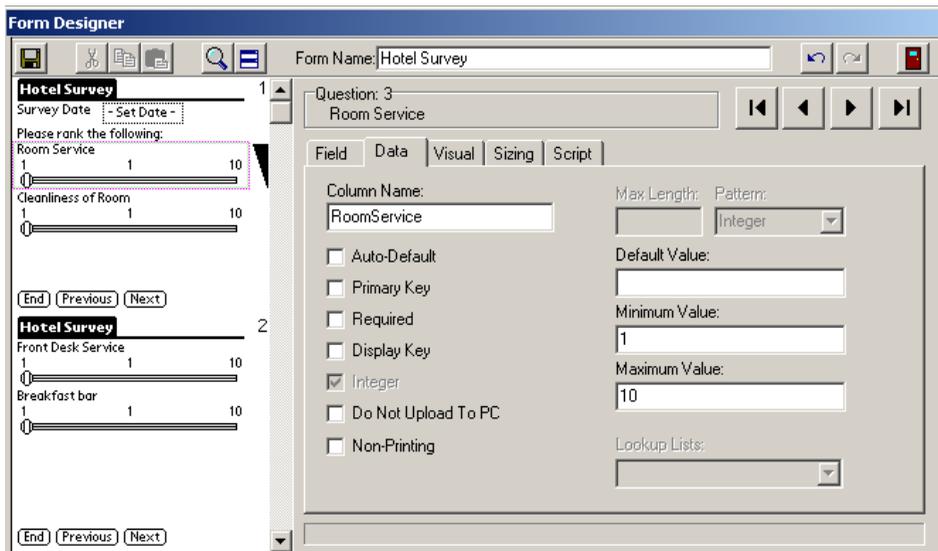
The Integer checkbox is pre-selected because a Slider field can only be an integer.

The default range is a minimum of 1 and a maximum of 10. To use a different range, enter a minimum value in the Minimum field and enter a maximum value in the Maximum field. Use the same minimum and maximum that you specified in the Left Caption and Right Caption fields on the Field tab of the Slider field.

If the Default Value field is left blank, then the value of the Slider field will be null until the handheld user taps the scale. The marker for the slider scale will not be displayed until the user taps the Slider bar to select a value.

If you enter a value in the Default Value field, then the value of the Slider scale will be pre-set to that number, and the marker on the scale will be at that position along the scale.

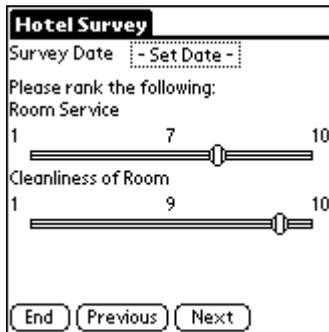
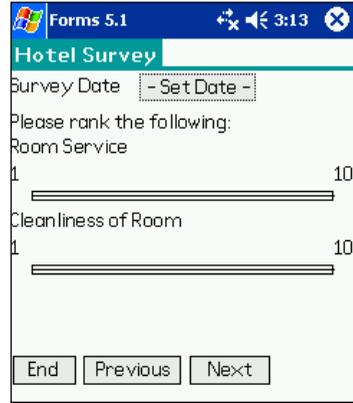
The slider bar is only 150 pixels wide. This means that if you make your slider range from 1 to 300, the smallest movement along the scale is 2 units at a time.



Features of Slider Fields



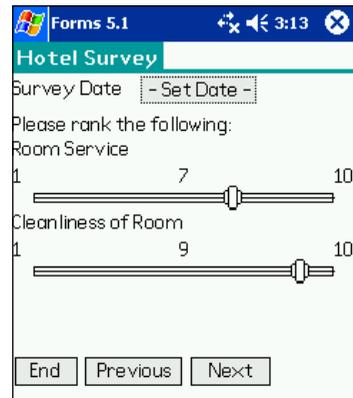
On the handheld, the Slider bar initially does not have a marker when the field is null.



Tap the horizontal line to position the marker along the scale. Or you can drag the marker along the scale. The marker will only stop at integer points along the scale.

After an initial tap, any subsequent tap on the slider bar will move the marker to that position. You can also drag the marker to a new position.

In Record View, the Slider field is a Numeric field without a scale, but with the same numeric range.



Sketch Field

A Sketch field allows the handheld user to draw a picture.

Sketch fields are similar to Signatures, but they allow a larger area on the handheld for drawing.

IMPORTANT: Sketch fields do not print directly from the handheld. Also, Sketch fields created on one platform (Palm OS or Pocket PC) cannot be sent to another platform.

In the Form Designer window, create a field of field type Sketch.

In the Title field, type a name for the sketch that the handheld user has to draw.

PALM OS

On Palm OS devices, the area of a sketch can be 160 pixels wide x 130 pixels high.

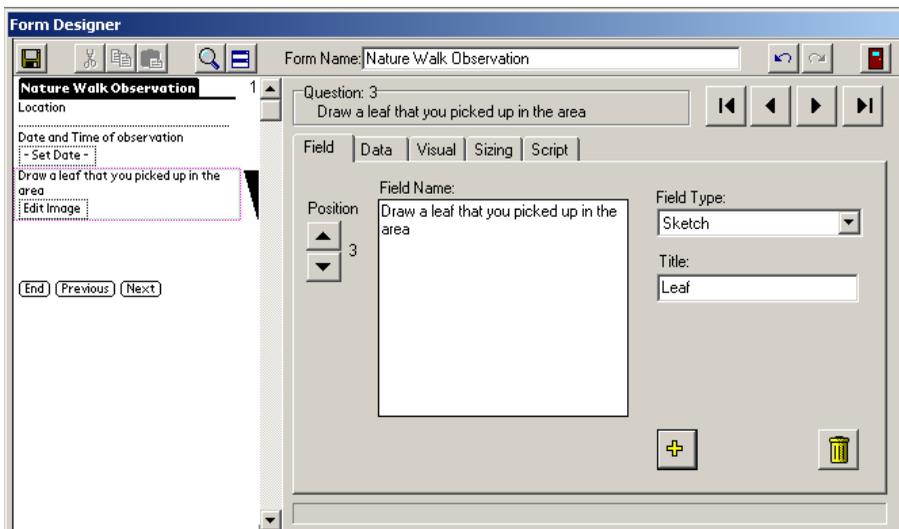
If a Sketch is created on a Palm OS device, the field will not be viewable or editable if the record is then put on a Pocket PC device.

PPC

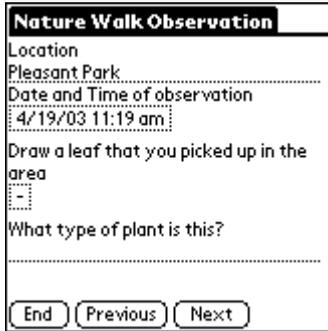
On Pocket PC devices, the area of a sketch can be 240 pixels wide x 195 pixels high.

If a Sketch is created on a Pocket PC device, the field will not be viewable or editable if the record is then put on a Palm OS device.

Also, sketches created in Pendragon Forms version 5.0 on a Pocket PC cannot be viewed or edited in Pendragon Forms 5.1 on a Pocket PC.

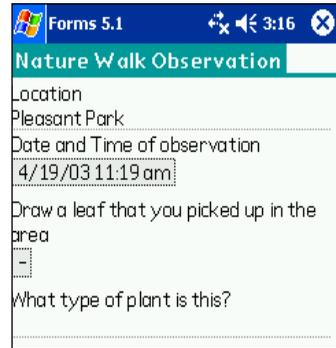


Features of Sketch Fields

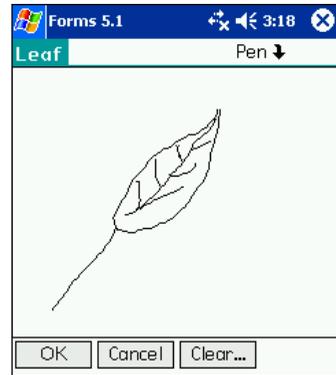


On the handheld, a dash (-) indicates that no sketch has been drawn in the field.

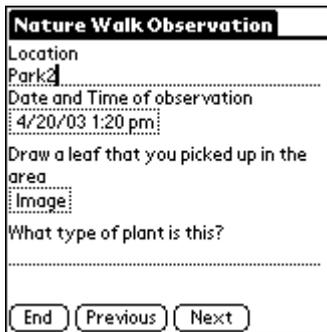
Tap the Sketch field to display the sketch area.



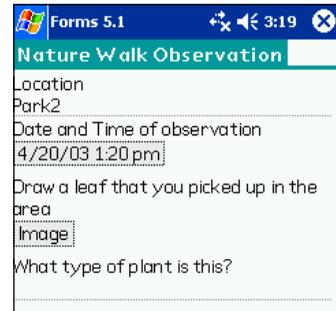
In the Sketch area, the arrow button in the upper right corner of the screen allows the handheld user to select Pen (a thin line), Marker (a thick line) or Eraser to erase. The user can draw in the sketch area and then tap OK to save the image.



Tapping the Clear button erases the entire sketch area. Tapping the Cancel button cancels all changes and reverts to the original state (no image, or previously saved image.)



The Sketch field will contain the word Image if a sketch has been saved.



Viewing a Sketch on the PC

After synchronization, each sketch is stored as a Long Binary Object in the Pendragon Forms Manager database, just like Signature fields. (See page 147.) You can double-click in a Long Binary Object cell to open Windows Paint to view the sketch. The sketch can then be saved as a bitmap (.BMP) file if necessary.

Image Field

An Image field is used to attach a picture to a record.

You can attach images to records in several ways:

- Attach images to records on the PC, then send the records to the handheld.
- Attach images to records on the handheld.
- Use a handheld with a built-in camera to take pictures that are then attached to records on the handheld. Supported handhelds include the Palm Treo 650, Treo 700p and Treo 700w. Refer to www.pendragonsoftware.com to check if any other handheld devices with cameras are supported.

An Image field can also be used to:

- Display an image that is stored on the handheld (or an external storage card), without attaching the image to the Pendragon Forms record.

Here are the requirements for using attached images on Pocket PC and Palm OS devices:

PPC

- Pocket PC devices have built-in image display capability, and can display .BMP, .JPG and .GIF files. The file size is up to 1MB.
- Attached image files are stored in built-in memory on the Pocket PC device, or an external storage card.

PALM OS

- Palm-branded Palm OS devices running Palm OS version 5.4 can display .JPG files
- On non-Palm-branded devices running Palm OS 5.4, and on Palm-branded devices running Palm OS versions prior to 5.4, Pendragon Forms by itself can only display small .BMP files with specific limitations (60KB file size, plus other limitations). Additional software that ships with Pendragon Forms can be installed to view .JPG files up to 300KB in size. Third-party software is required to view .GIF files or large .BMP and .JPG files.
- Attached image files can be stored on an external storage card. If the Palm OS device has a file system in memory (e.g. Palm TX and Palm LifeDrive), image files can also be stored in memory.

Refer to Chapter 17, Working with Images, page 422 for more details on the requirements for using attached images on Palm OS devices.

To create an Image field, select Image as the field type in the Form Designer window.

The following options are available for Image fields:

Hi Res

PPC

The Hi Res option does not affect Pocket PC devices, which will automatically scale the image appropriately.

PALM OS

Check the Hi Res checkbox if you are designing an application for a high resolution device (320x320 pixel screen such as Palm Treo series). The image will be displayed in high resolution. Hi Res images appear 1/4 the size of low resolution images, but are sharper. If you check the Hi Res option, but put your form on a low resolution device (160x160 pixel screen), the image will be displayed in low resolution.

Inline

Check the Inline checkbox if you want the image to be automatically displayed onscreen within the Pendragon Forms record. (An image must conform to the Pendragon Forms limits described on page 422 to be visible inline.)

For performance reasons, if you do not want to display an image every time the user views the screen, you can choose not to check the Inline checkbox. Then the

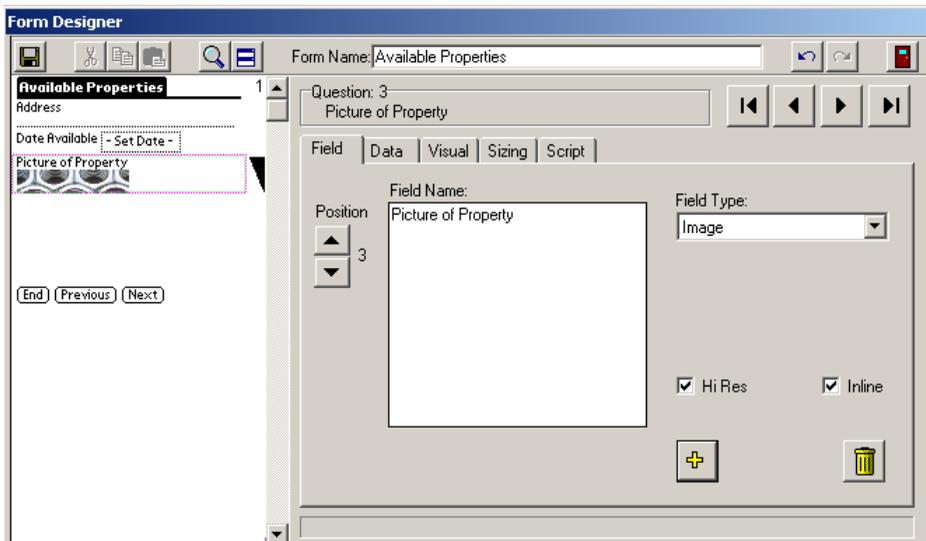


Image field will contain an icon that the user can tap when they want to see the image. On Palm OS devices, if you are using the third-party AcidImage program to display large files, you should **not** check the Inline checkbox.

If you are viewing images Inline, then to reserve a space on the handheld screen that is large enough to view the image, click the Sizing tab and make the Answer Height bigger. You can also make the Question Height zero.

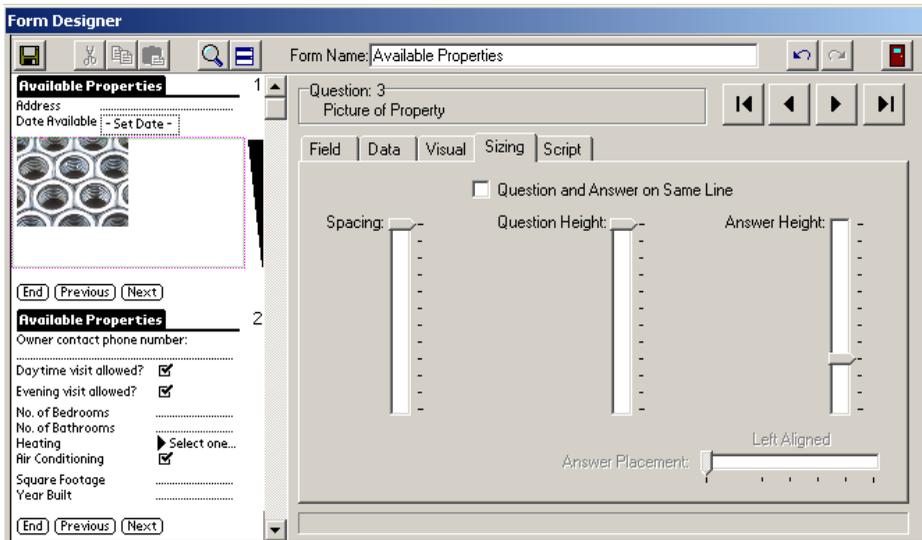
PPC

Pocket PC devices will scale the image appropriately for whatever Answer Height you set. Make the Answer Height big enough so that the image does not appear too small.

PALM OS

If you are using .JPG files on a Palm OS device, the image files will scale appropriately for the Answer Height. Make the Answer Height big enough so that the image is not too small.

If you are using .BMP files on a Palm OS device, the sample graphic that displays in the Preview Area of the Form Designer is 160x128 pixels in size, so if your image file is that size (e.g. on Low Res devices), make the Answer Height big enough to see the entire sample graphic. If your image file is 320x192 pixels (e.g. on Hi Res devices), you will need to have the Image field on a screen by itself, and make the Answer Height as big as the Form Designer will allow. Note that low resolution devices (160x160 pixel screens) will only be able to display a portion of a 320x192 pixel graphic.

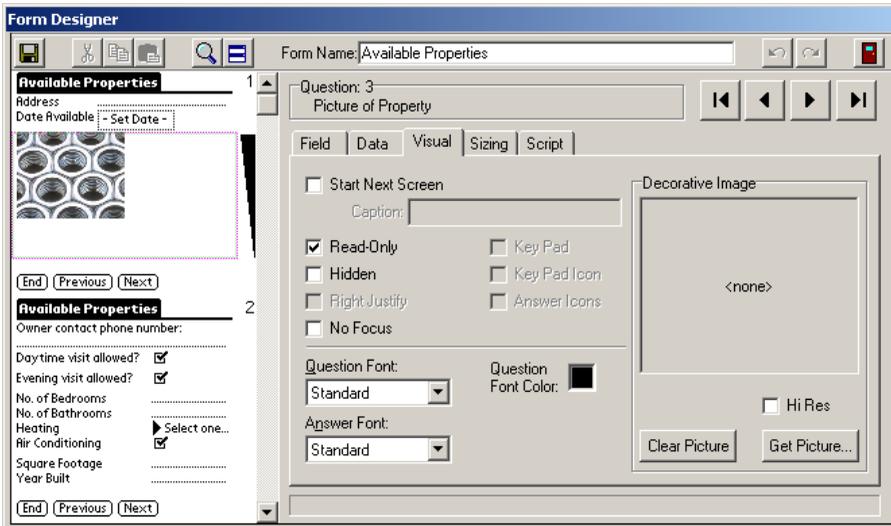


PALM OS

On Palm OS devices, if images are **not** being displayed Inline, users have the option to clear an image when they tap to view the image. This mechanism gives the user the chance to delete an image, in the event that they attached the wrong image. However, if you are attaching the images on the PC and sending them to the handheld, you might not want the handheld user to be able to delete the images. To prevent the handheld user from deleting the images, click the Visual tab for the Image field and check the Read-Only checkbox.

NOTE:

Making an Image field Read-Only does not protect the image file if a script is used to attach an image file to a record.



NOTE:

If you want several handhelds to share records, you will need to select one or more fields to be the primary key field(s) before you freeze the form. See pages 31-34 for information on creating forms for users to share records.

Attaching Image Files to Records on the PC

You might want to attach image files to records on the PC if you already have the images on your PC. For example, you might have pictures of real estate properties that you want to attach to records with information about each property.

For details on how to attach image files to records on the PC, see Chapter 17, Working with Images, page 423.

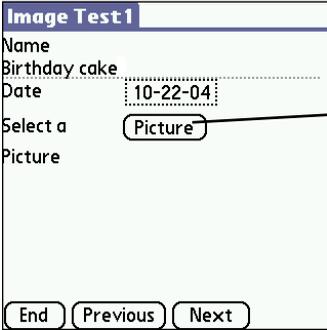
PALM OS		PPC
<div style="background-color: #4a7ebb; color: white; padding: 2px;">Available Properties</div> Address 1 Country Lane Date Available 4-21-03  <div style="display: flex; justify-content: space-between; margin-top: 5px;"> End Previous Next </div>	<p>If an Image field is set to display Inline, and the image conforms to the limits for being viewable in Pendragon Forms, it will be displayed whenever you are on the screen containing that field.</p>	<div style="background-color: #00a68a; color: white; padding: 2px;">Available Properties</div> Address 1 Country Lane Date Available 4/21/03  <div style="display: flex; justify-content: space-between; margin-top: 5px;"> End Previous Next </div>
<div style="background-color: #4a7ebb; color: white; padding: 2px;">Available Properties</div> Address 1 Country Lane Date Available 4-21-03 Image/Object: <div style="display: flex; justify-content: space-between; margin-top: 5px;"> End Previous Next </div>	<p>If an Image field is not set to display Inline, a selector box will be displayed.</p> <p>Tapping the selector box will display an Image Detail screen. If the image is viewable in Pendragon Forms, the image will be displayed on the Image Detail screen.</p>	<div style="background-color: #00a68a; color: white; padding: 2px;">Available Properties</div> Address 1 Country Lane Date Available 4/21/03 Image/Object: <div style="display: flex; justify-content: space-between; margin-top: 5px;"> End Previous Next </div>
<div style="background-color: #4a7ebb; color: white; padding: 2px;">Image Detail</div>  <div style="display: flex; justify-content: space-between; margin-top: 5px;"> OK Viewer </div>		<div style="background-color: #00a68a; color: white; padding: 2px;">Image Detail</div>  <div style="display: flex; justify-content: space-between; margin-top: 5px;"> OK Viewer </div>

Attaching Image Files to Records on the Handheld

You might want to attach image files to records on the handheld if the images are on the handheld. On Palm OS devices, the images must be on an external storage card. On Pocket PC devices, the images can be stored anywhere in the file system of the device.

For details on how to attach image files to records on the handheld, see Chapter 17, Working with Images, page 426.

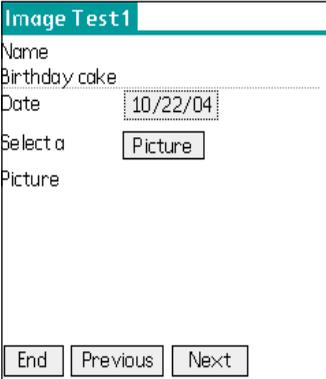
PALM OS



To give the handheld user the ability to select an image on the handheld, a Button field must be present on the form, as well as an Image field for storing the image.

The script needed in the Button field is described on page 427-428.

PPC



PALM OS



On Palm OS devices with Palm OS 5.4, tap the storage card icon to view images on the card. Images are typically in the DCIM folder on the card. Tap a file to select it.

PPC



On Pocket PC devices, images can be stored on a storage card, or anywhere in the file system of the device. Select the folder or storage card where the image is located, then tap the image to attach the image.

PALM OS



If an Image field is set to display Inline, it will be displayed within Pendragon Forms.

PPC



Using a Zire Camera Custom Control to Attach Images on the Handheld

PALM OS

On older Palm OS devices with a built-in camera, such as the Palm Zire 71 and Zire 72, you can use a Custom Control field to allow the handheld user to take a picture with the built-in camera, and automatically attach the picture to the current Pendragon Forms record.

Refer to the Support page at www.pendragonsoftware.com for information on using the Zire 72 Custom Control.

Image Files

On the PC, when you import an image file, a copy of the image is stored in a binary field in the Forms Manager database.

On the handheld, image files are stored in a FORMS5 folder on the external media card (SD card, MMC card or Memory Stick), or on the FORMS5 folder in the memory of the device. The name of each file is composed of the FormID number (8 hex (hexadecimal) digits), the RecordID (8 hex digits) and the Question ID (4 hex digits), plus a file extension that is assigned by the conduit when the image is downloaded to the handheld. Pendragon Forms on the handheld uses the file extension to determine whether or not a file can be displayed internally.

Every time you synchronize the handheld, all records are removed from the handheld, and then the records that are to remain on the handheld are re-sent to the handheld. This is true for image files also. If you have a form with many records that contain images, synchronization may be slow as the images are removed from the handheld and then re-sent to the handheld. If the form is for reference purposes and the records are not updated by the handheld user, you can set the Advanced Form Property to Synchronize Only When Distributed. See page 222.

For more information, see Chapter 17, *Working with Images*, starting on page 419.

Custom Control Field

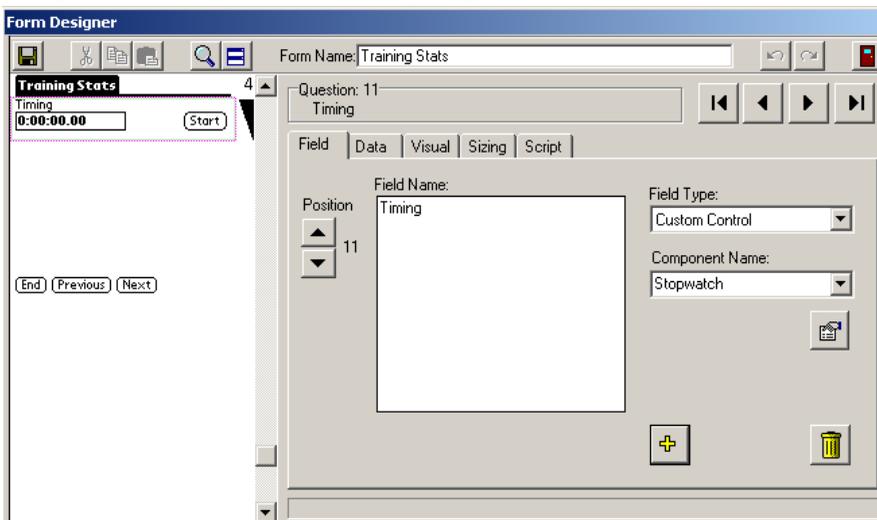
Pendragon Forms supports Custom Controls - separate program modules that can be installed on the handheld in order to extend Pendragon Forms' capabilities.

Pendragon Forms ships with three Custom Controls: a GPS (Global Positioning System) control (FormsGPS.prc); a stopwatch control (FormsStp.prc); and a Zire 72 Camera control (FormsZire71JPG.prc). New in Forms 5.1, the GPS Custom Control had been built into the regular installation of Forms 5.1 on the handheld. The other Custom Controls are in the C:\Program Files\Forms3 folder and can be installed on the handheld if you are creating a form that requires them.

There is also a Software Development Kit for developers to create their own Custom Controls using the C (Palm OS) or C# (Pocket PC) programming language. The kit can be downloaded from www.pendragonsoftware.com.

A Custom Control field type is a placeholder for one of these external program modules. A Custom Control field is like a Text field with a customized appearance. The Custom Control field stores a value generated by the external program module. In the case of the stopwatch control, the value placed in the Custom Control field is the elapsed time with 0.1 second precision. In the case of the GPS control, the value placed in the Custom Control field is a set of GPS coordinates.

Once you install a Custom Control .PRC file onto the handheld, that program module can be accessed by several fields on the same form, or by several forms.

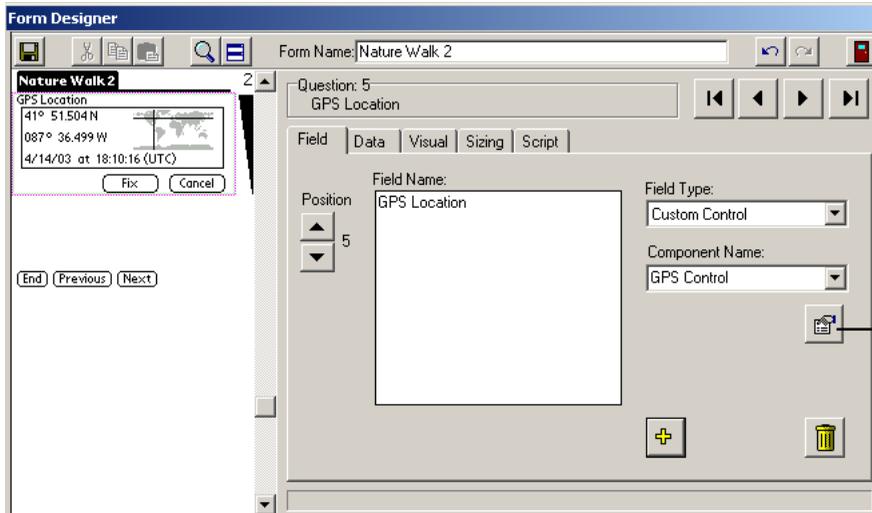


Using a Custom Control Field

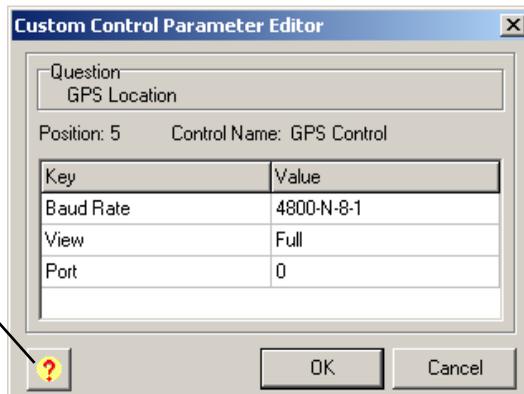
In the Form Designer window, type a name for the Custom Control field and select Custom Control as the field type.

The Component Name field will display a list of the available custom controls. Select the custom control that you want to use. The built-in custom controls are Stopwatch, GPS Control and Zire JPEG Camera. (Note: The 'Other' option is reserved for developers who are building their own custom controls.)

Next, click the Parameters button to enter the parameters that are required for the custom control that you are using. For information on the required parameters, click the Help button.



Click the Help button in the Custom Control Parameter Editor window to display information on the parameters required for the Custom Control you are using. The Help is usually in .PDF file format, readable with Adobe Reader software.



Stopwatch Custom Control



For the Palm OS platform, the Stopwatch custom control is called FormsStp.prc. This file is typically located in the C:\Program Files\Forms3 folder, and will need to be installed on the handheld. For Pocket PC devices the stopwatch custom control installs automatically when you install Pendragon Forms 5.1.

In the Custom Control Parameter Editor window in the Form Designer, click the Help button to read more about using the Stopwatch control.

Tapping the Start button starts the stopwatch. Tapping Stop stops the timer. The time elapsed is recorded to 0.1 second precision and stored in the Custom Control field. Note: the Stopwatch control is not a lap timer. If the user goes back to the Custom Control field and taps the Start button again, the Stopwatch will start counting from zero again.

GPS Custom Control



In order to use the GPS (Global Positioning System) Custom Control module, the handheld user will need an external GPS device that can attach to the handheld, or a Bluetooth-enabled GPS device or a handheld with a built-in GPS receiver.

In the Form Designer window, if you are using the GPS custom control, click the Parameters button to enter the parameters required to make your external GPS device work with Pendragon Forms. See page 574-576 or click the Help button in the Custom Control Parameter Editor window to read more about the required parameters.

New in Forms 5.1, the GPS custom control is automatically installed when you install Pendragon Forms on a Palm OS device or a Pocket PC device.

Zire 72 Camera Custom Control

For information on the Zire Camera Custom Control, see the Support page at www.pendragonsoftware.com.

8. Advanced Form Designer

The Form Designer is where you design forms to send to the handheld. This chapter assumes that you have read the basics of using the Form Designer in Chapter 2 (see page 13).

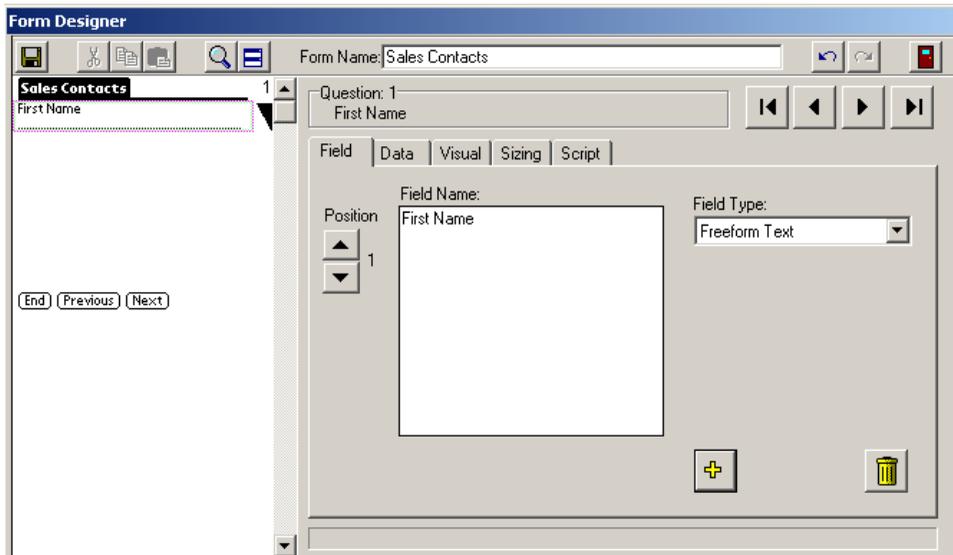
To open the Form Designer window, either click the New button in the Pendragon Forms Manager, or click on the name of an existing form and click the Edit button.

The Form Designer Window

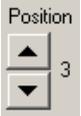
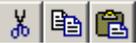
Every form must have a Form Name, which is typed into the Form Name field at the top of the Form Designer window.

The Preview Area on the left side of the Form Designer shows which field is currently selected, by displaying a pink dotted-line box around the field. There is also a black triangle to the right of the current field, so if the pink dotted-line box is difficult to see, use the black triangle as your marker for which field is selected.

The five tabs: Field, Data, Visual, Sizing and Script all apply to the selected field.



The following table shows what the buttons in the Form Designer are used for:

Button	Function
	Click the Plus button to add a field after the selected field.
	Click the Up and Down arrow buttons to re-position the selected field, either moving the field up or down the form.
	<p>Click the Left and Right arrow buttons to make the next or previous field the selected field.</p> <p>The arrow buttons with lines jump you to the first and last fields on the form.</p> <p>If you are on the last field on the form, and you click the right arrow button, a new field will be created at the end of the form.</p>
	Click the Save button to save the form design. If you are working on a very long form, save often.
	Click the Exit button to close the Form Designer window. If you have not yet saved changes to the form design, you will be prompted to do so. Choose Yes to save changes.
	Click the Delete button to delete the selected field.
	Click the Field View Preview button to change the Preview Area to display what Field View looks like (one field displayed at a time on the handheld). Only use this if you are going to use Field View on a Palm OS handheld.
	Click the Find button to search for a phrase in any part of a field: the field name, Popup List items, scripts. This is useful if you have a large form and you want to find a specific field.
	Hold down the Shift button and click on a field in order to use the Cut Copy and Paste buttons to cut or copy a field and paste the field into a different location on the form. You can also move a field with the Up and Down arrow buttons instead of using Cut and Paste.

Field Tab

The Field Tab of the Form Designer is where you specify the field name and field type of each field on the form.

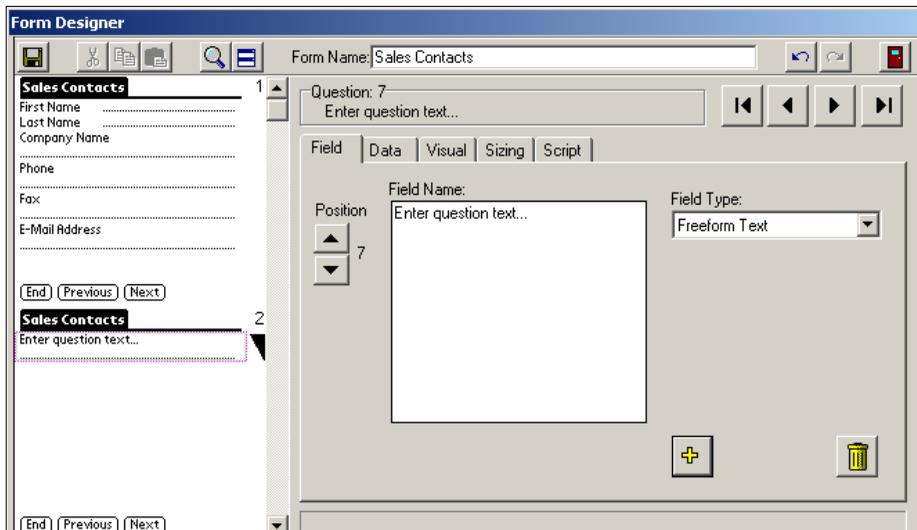
When you create a new field, the Field Name area contains the text *Enter question text...*. Highlight this text if it is not already highlighted, and type the name that you want for this field.

In the Field Type field, click the arrow button to select a field type for this field. The field type determines the kind of data that the handheld user will be allowed to enter in this field. Chapter 7, *Field Types*, page 74, details the 23 different field types that are available.

The field number, that is, the position of the field on the form, is shown in two places: the Position area of the Field tab and in the Question area that is visible above the Form Designer tabs. You can use the Up and Down arrow buttons to move a field up or down from its current position.

To add another field to the form, click the + button. The new field will be added after the current position. Alternatively, if you are on the last field of the form, you can click the right arrow button to create a new field at the end of the form.

To make a different field the selected field, either click on the field in the Preview Area, or use the left and right arrow buttons until the field is displayed in the Question Area.

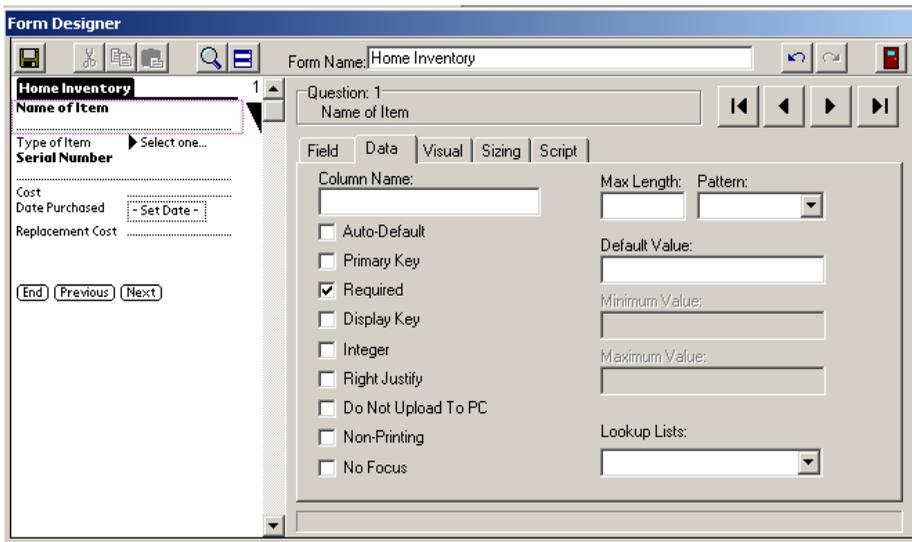


Data Tab

The Data Tab in the Form Designer window displays the Advanced Field Properties of the selected field.

The Advanced Field Properties that are available depend on the field type of the selected field.

For a complete description see Chapter 9, *Advanced Field Properties*, starting on page 185.



Visual Tab

The Visual Tab in the Form Designer window displays various attributes that affect how the field appears visually on the handheld.

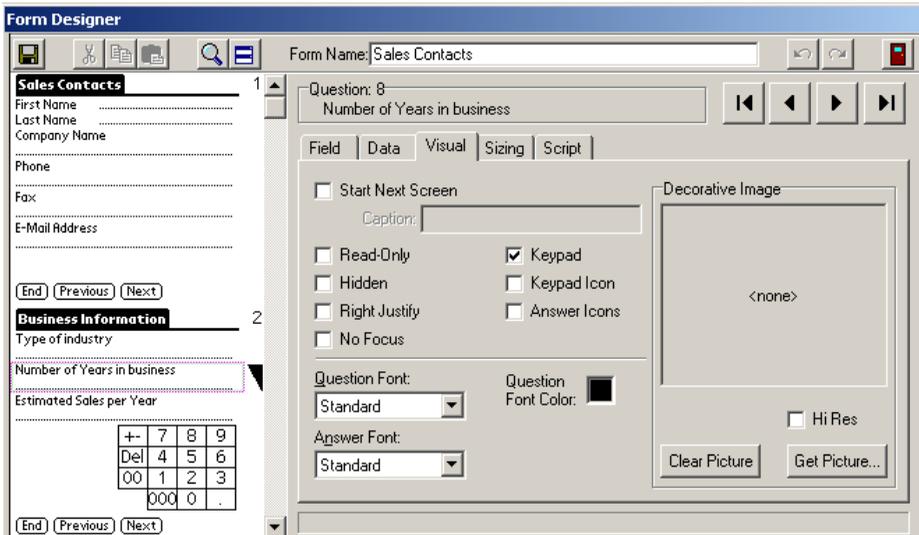
The visual attributes are described here.

Adding a Numeric Keypad

In Text, Numeric and Currency fields, you can choose to add a numeric keypad to the handheld screen, to make it easier for the user to enter data.

To add a keypad, click the Visual tab of a Text, Numeric or Currency field, and check the Keypad checkbox. Only one keypad will be displayed on the screen, so you only need to set the Keypad option for one of the fields per screen.

If you add a keypad to a screen, you will, of course, be able to fit fewer fields on that screen.



Making a field start on a new screen

When you add a field to a form, if the field can fit on the current screen, the Form Designer will just add the field to the current screen.

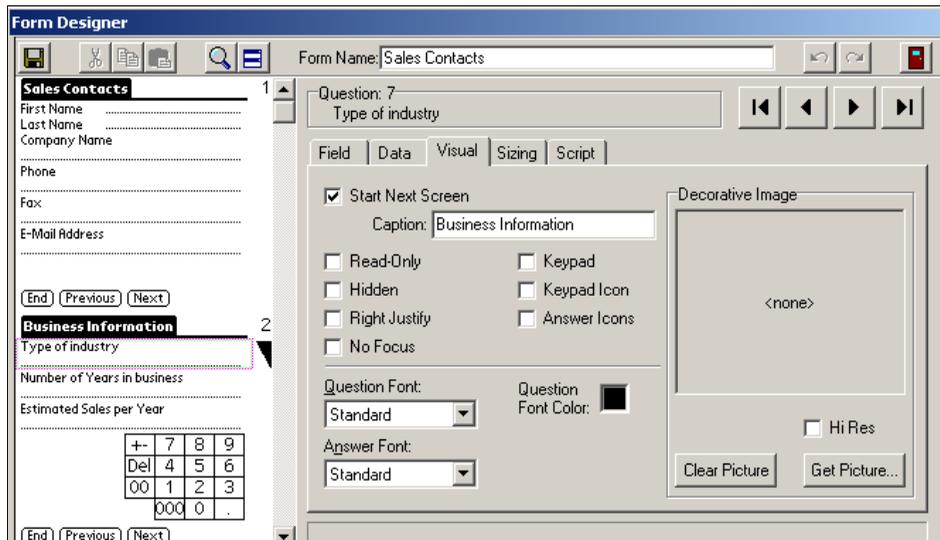
In some instances, it is helpful for a group of similar questions to start on a new screen.

To make a field start on a new screen, click the Visual tab for that field and check the Start Next Screen checkbox.

In the Caption field, you have the option of typing a new header, so that the new screen has a different header on the handheld screen. If you leave the Caption field blank, the form name will be used as the default header.

PPC

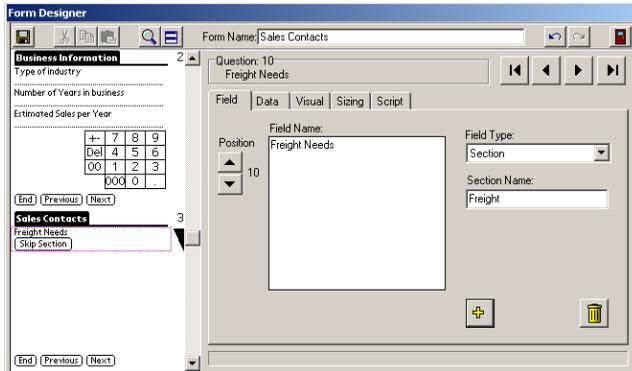
NOTE: In Forms 5.1, on Pocket PC devices, screen captions will not be visible if the onscreen keyboard is up. If you minimize the keyboard, the title bar showing the screen caption will be visible.



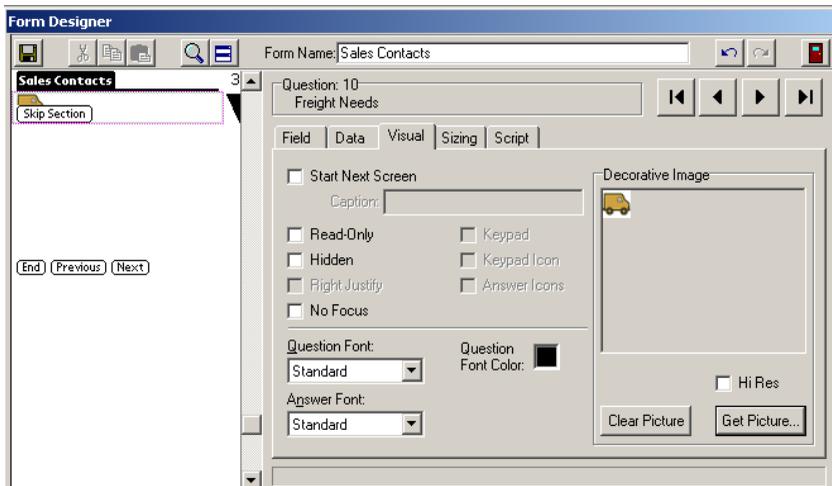
Adding an image to a screen

You can add a decorative (static) image to a question to add style, color or emphasis. Unlike images attached to Image fields, decorative images are part of the form design, not part of the data, so they do not increase the size of each record.

Section fields are frequently used to display a static image, because Section fields do not require the user to enter any data. However, you can add a static image to any type of field.



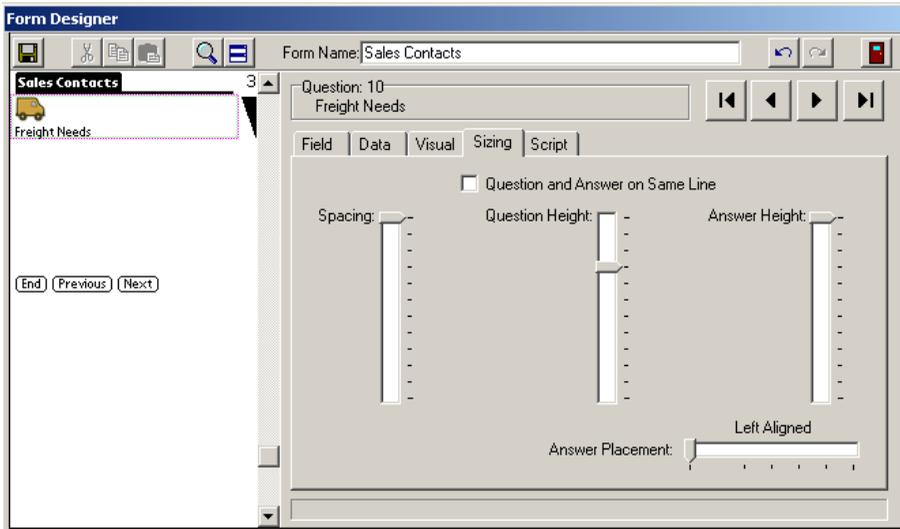
To add a decorative image, select the Visual tab and click the Get Picture button. You can select a picture from the C:\Program Files\Forms4\ClipArt folder that ships with Pendragon Forms. Or you can select a picture of your own, as long as the picture is a Windows bitmap file (.BMP file) or a JPEG file (.JPG file) not more than 320x192 pixels in size. If the selected image has more than 256 colors, it will be reduced to 256 colors before being placed on the form.



On the Sizing tab of the field, make the Question Height bigger so that you can see all of the picture. The field name of the field appears below the picture.

If you want to hide the field name, and just have the picture visible, adjust the Question Height accordingly.

For Section fields, you can adjust the Answer Height to zero to hide the Section Skip button.



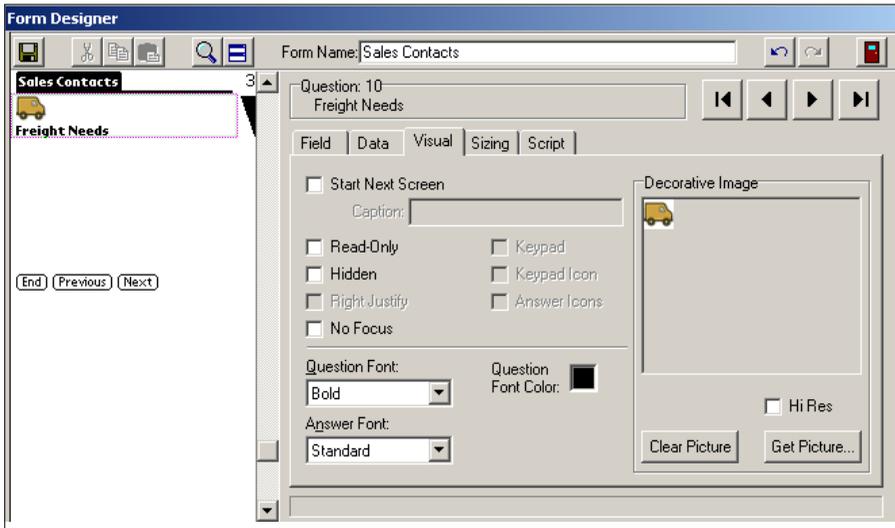
Changing font size and color

In Layout View, you can change the font of the question or answer component of a field on the Visual tab of the Form Designer. Your choices are: Standard font, Bold, Large or Large and Bold.

The Question Font field controls the font for the field name or question. The Answer Font field controls the font for the answer, that is, the handheld user's response.

When you select font options, it is assumed that you want to continue with these options, and so new fields that are created will have to same font options set. If you only intended one field to have different fonts, change the font options back to Standard on the next field.

In the example shown below, the field name (question) is set to Bold.



On color handheld devices in Layout View, question fonts can also be a color other than black. Answer fonts can only be black.

To change the question font, click in the Question Font Color field to display a color palette.



To choose a basic color, click on the color in the Basic colors section, then click OK.

The current field, plus any subsequent new fields, will have the question font in the selected color. If you only want one field to be in a different color, choose black as the question font color of the next field.

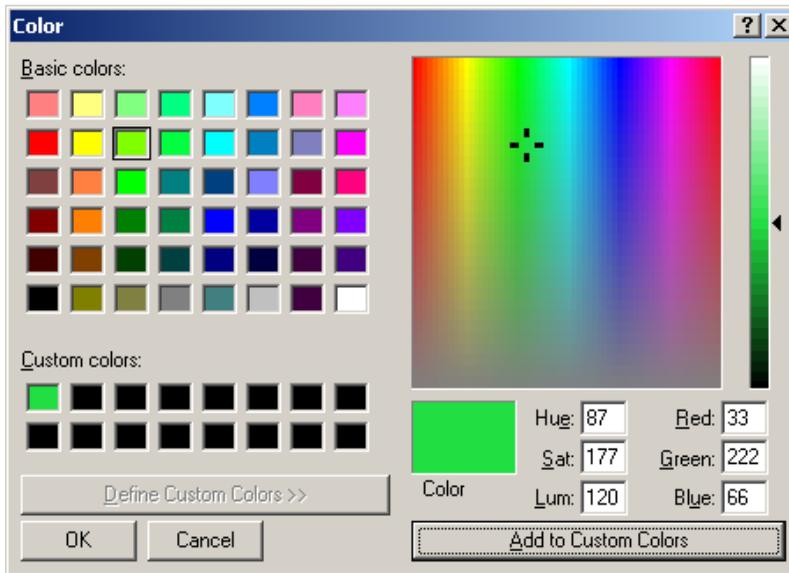
Avoid using light colors for the question font color, as they may be hard to read on the handheld.

If you do not like your font size or color, you can go back and change fonts even after a form is frozen.

It is also possible to define your own custom colors. You must be using Layout View on a color handheld device. From the Color palette, click the Define Custom Colors button to expand the Color window.

To start, first click on a basic color (anything except black or white) as your starting point, then click in the graduated palette area on the right to choose your custom color. When you are satisfied, click the Add to Custom Colors button. The new color will be added to the Custom colors palette.

To make a question font a custom color, click on the color in the Custom colors palette and click OK.

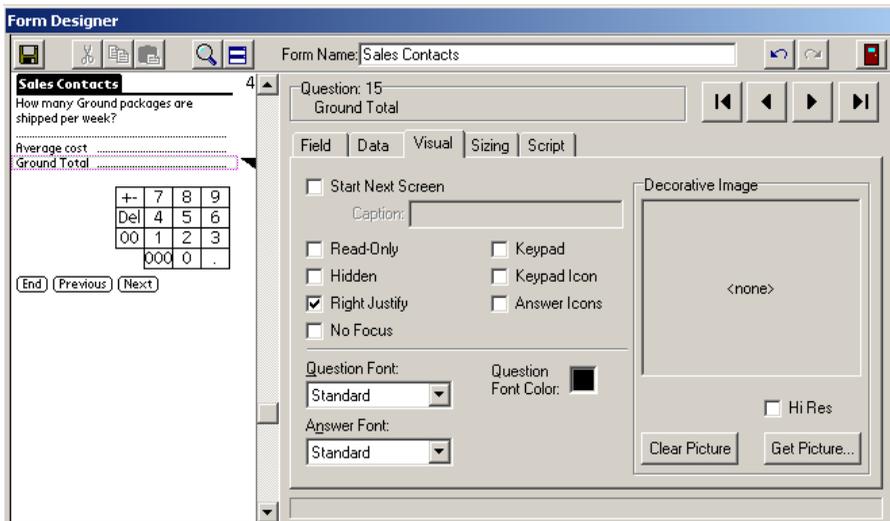


If you change font sizes for questions and answers, you will be able to fit fewer fields on a screen.

Right-Justifying Numeric and Currency fields

If you have Numeric and Currency fields on your form, you can right-justify the answer field. This is especially useful in Currency fields where the decimal points will line up vertically.

For each field that you want to right-justify, click the Visual tab and check the Right Justify checkbox.



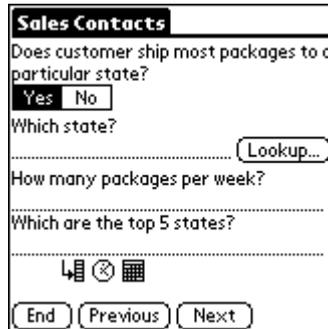
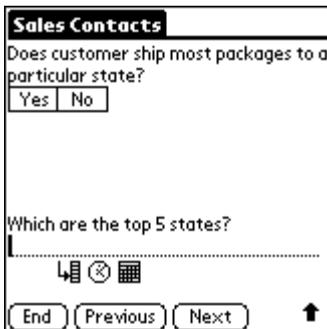
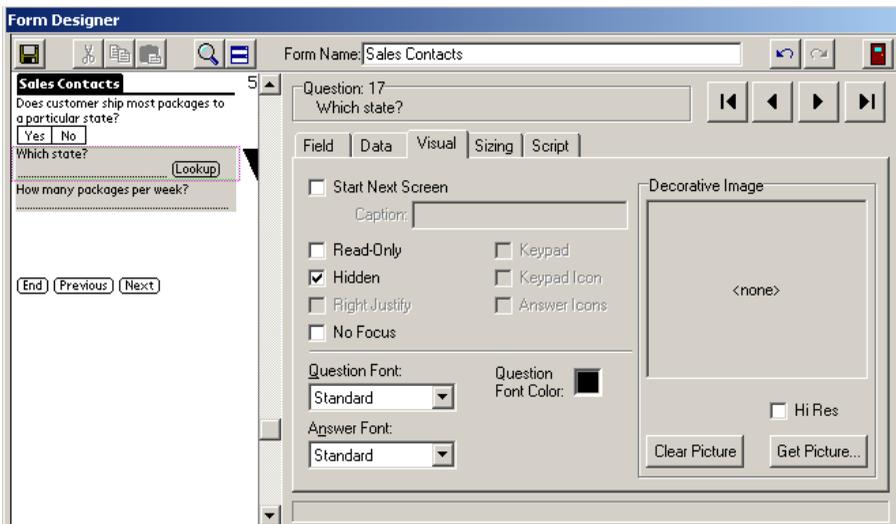
Sales Contacts																			
How many Ground packages are shipped per week?																			
			15																
Average cost		6.95																
Ground Total		104.25																
<table border="1" style="width: 100%; text-align: center;"> <tr> <td>+ -</td> <td>7</td> <td>8</td> <td>9</td> </tr> <tr> <td>Del</td> <td>4</td> <td>5</td> <td>6</td> </tr> <tr> <td>00</td> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td></td> <td>000</td> <td>0</td> <td>.</td> </tr> </table>				+ -	7	8	9	Del	4	5	6	00	1	2	3		000	0	.
+ -	7	8	9																
Del	4	5	6																
00	1	2	3																
	000	0	.																
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>																			

Hidden fields

If some questions on your form do not apply to everyone filling in the form, you can choose to hide these fields, and then use a script to show the fields under certain conditions.

To make a field hidden, click the Visual tab for that field and check the Hidden checkbox. Refer to Chapter 13, *Scripting Reference*, pages 297 and 328, and Chapter 14, *Scripting Examples*, page 357, for information on how to write a script to display a hidden field.

Pendragon Forms reserves the on-screen space of a Hidden field, even though the field is not visible. The Preview Area of the Form Designer reflects this by showing the Hidden field as grayed out. If the Hidden field is made visible, it will occupy the same space as seen in the Preview Area.



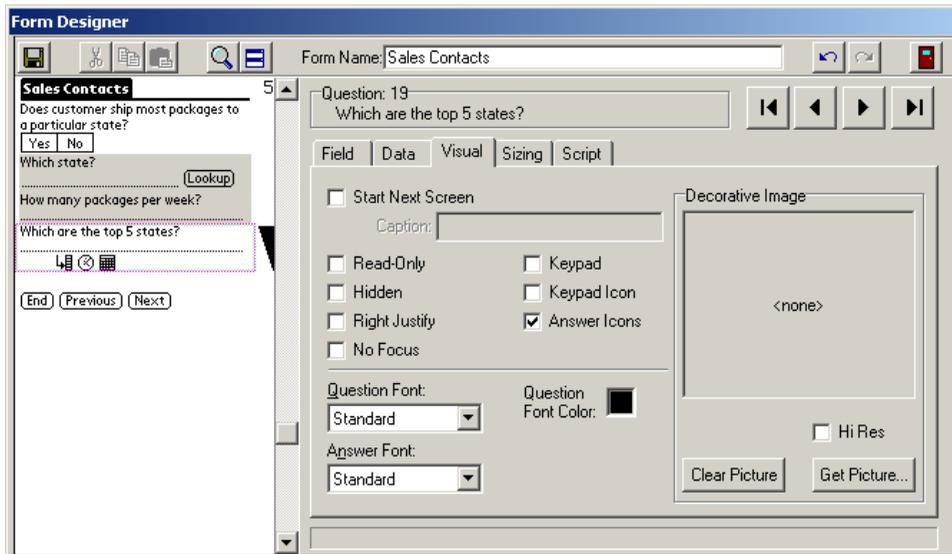
Answer Icons

It is possible to add a Clock and a Calendar icon to a Text field, to allow the handheld user to insert the current time or current date and time to the Text field. If the Text field also references a Lookup List, a Lookup List icon allows the handheld user to make selections from the Lookup List repeatedly, and copy each selection into the Text field.

On the Visual tab, check the Answer icons checkbox. This will display icons for a Clock and a Calendar. Tapping on the Clock icon will insert the current time into the field, and tapping on the Calendar will insert the current data and time.

On the Sizing tab, make the Answer Height large enough so that you can see the icons in the Preview Area of the Form Designer.

If you want to reference a Lookup List in addition to the Clock and Calendar icons, then on the Data tab of the Text field, select a Lookup List in the Lookup List field.



No Focus (No Cursor Blinking in a Text field)

The screenshot shows a form titled "Sales Contacts". It contains the following elements:

- A question: "Is customer close to end of current freight contract?" with "Yes" and "No" buttons.
- A question: "How many years was current contract?" with a numeric keypad showing digits 1, 2, 3, 4, and 5.
- A question: "Customer willing to switch?" with an unchecked checkbox.
- A text field labeled "Send bid to:" with a dotted line below it.
- Navigation buttons: "End", "Previous", and "Next".

By default, a cursor blinks in the first available Text, Numeric, Currency or Lookup List field on the handheld screen. If the user were to immediately begin writing, their response would be placed in the field where the cursor is blinking.

In some instances, users may be confused at the cursor position, especially if there are other fields such as Popup Lists or Yes/No Checkboxes that the user is supposed to fill in before getting to the field where the cursor is blinking.

You can switch off the blinking cursor for a field by clicking the Visual tab in the Form Designer and checking the No Focus checkbox. The cursor will not blink in the field unless the user taps in that field.

Keypad Icon

The screenshot shows a form titled "Returned Items". It contains the following elements:

- Product Name: "Super Snow-Maker"
- Product Serial Number: (empty field)
- Date Returned: A field with a keypad icon and the text "- Set Date -"
- Reason for return: (empty field)
- Navigation buttons: "End", "Previous", "Next", and an upward arrow icon.

PALM OS

On Palm OS devices, you can choose to add a Keypad icon to a Text field, to give the handheld user another way to access the built-in keyboard. (Pocket PC devices always have an icon at the bottom of the screen to pop up the onscreen keyboard.)

The cursor has to be blinking in the Text field for the Keypad icon to be functional. If the cursor is not blinking in the field, the user must tap in the field before tapping the Keypad icon.

To add a Keypad icon, click the Visual tab in the Form Designer, and check the Keypad icon checkbox. Then click the Sizing tab and make the answer height large enough so that the Keypad icon is visible in the preview area.

Making a field Read-Only

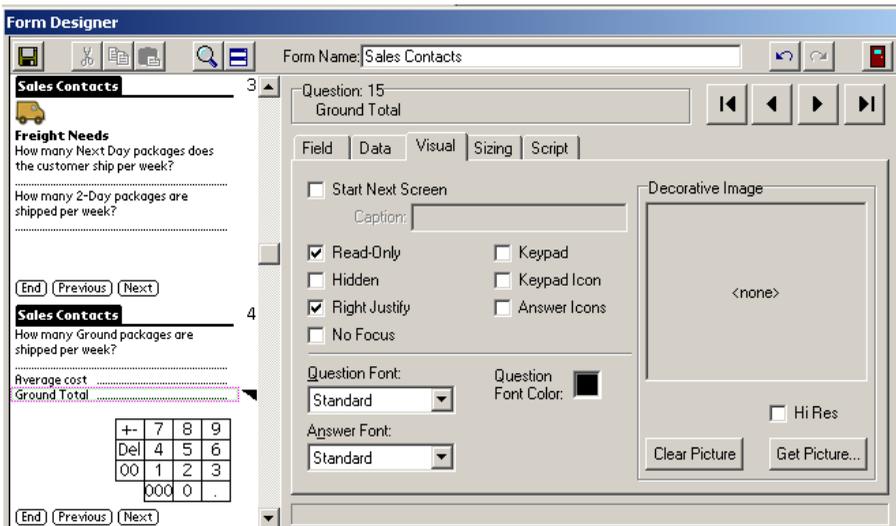
In some circumstances, you may want to make a field Read-Only on the handheld.

For example, you might want the handheld user to input the variables used to perform the calculation, but not be able to change the result of the calculation directly. In this case you would make the field that is storing the calculated result read-only.

To make a field read-only, click the Visual tab and check the Read-Only checkbox.

PPC

In the prior version, Forms 5.0, on Pocket PC devices, making a field read-only caused the answer font color to be gray. In Forms 5.1 this does not occur and the answer font remains black.



Sizing Tab

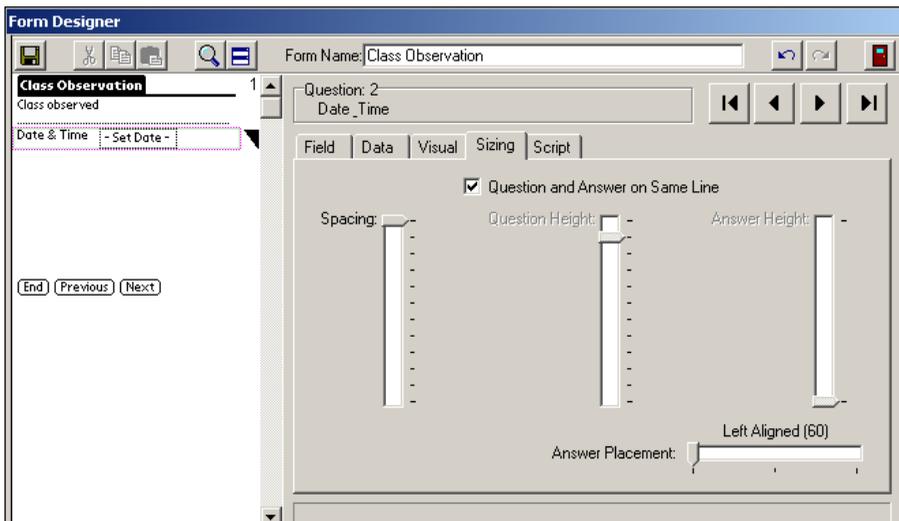
The Sizing Tab in the Form Designer window lets you adjust the amount of screen space that a field occupies on the handheld screen.

You can change sizing settings even after a form has been frozen.

Fitting a Question and Answer on one line

By default, every field takes up at least 2 lines on the handheld screen: one line for the question (the field name), and another line for the answer (the handheld user's response).

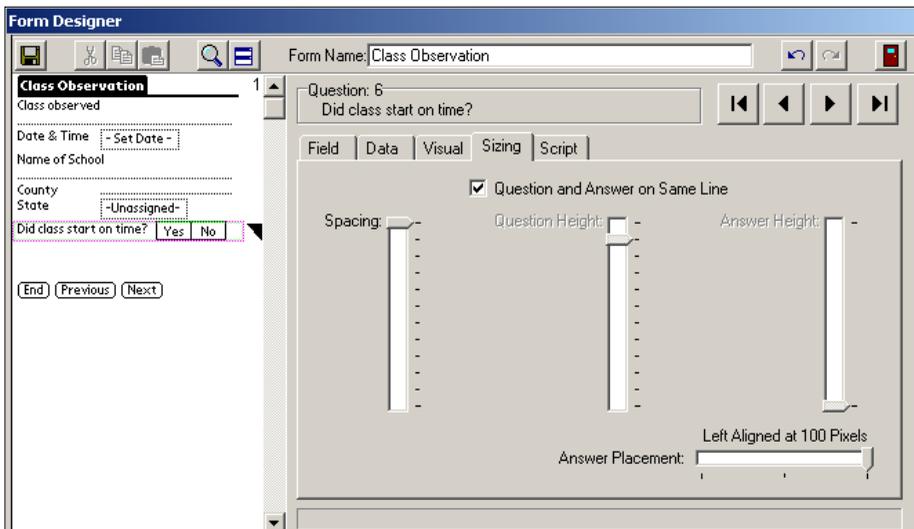
To fit more fields per screen, click on the Sizing tab for a field and check the checkbox labeled Question and Answer on Same Line. This places both the question and answer of that field on one line of the handheld screen. This option works well if the question is not too long.



Adjusting Answer Placement

If you fit a question and answer on one line, and the answer partially blocks the question, you can try adjusting the slider in the Answer Placement field on the Sizing tab.

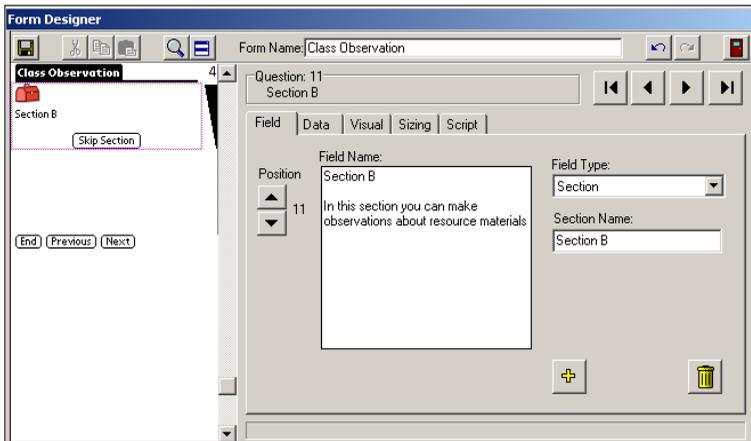
In many cases, moving the placement of the answer to the right will allow the whole field name to be seen. If this does not work, the field name may be too long to fit both question and answer on one line. Either shorten the field name or return to a 2-line format by un-checking the Question and Answer on Same Line checkbox.



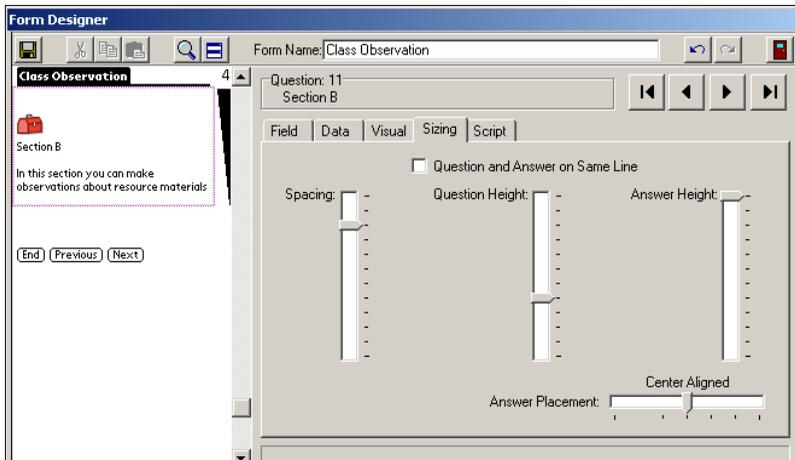
Adding Spacing before or after a Question

The Spacing, Question Height and Answer Height sliders are used to adjust the amount of space before the question, and the amount of space used by the question and the answer, respectively. The Question and Answer Height sliders are available if a field takes up more than one line.

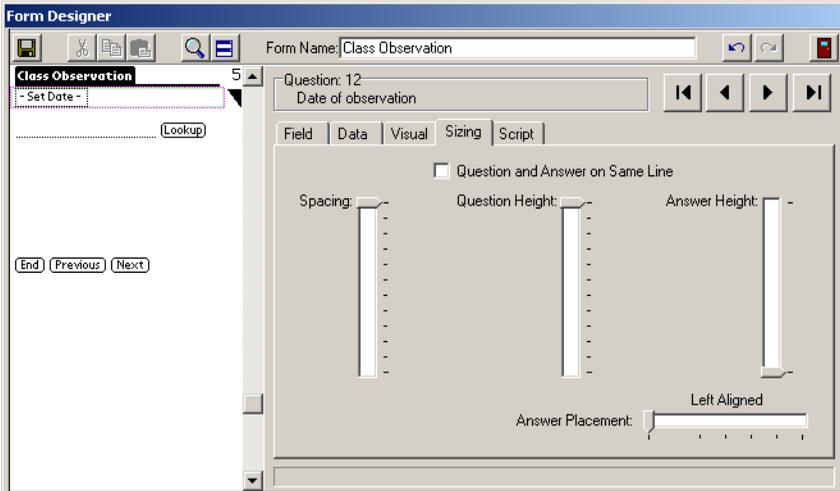
In the example shown here, a Section field contains a picture, and the question height is initially too small to display both the picture and the whole field name.



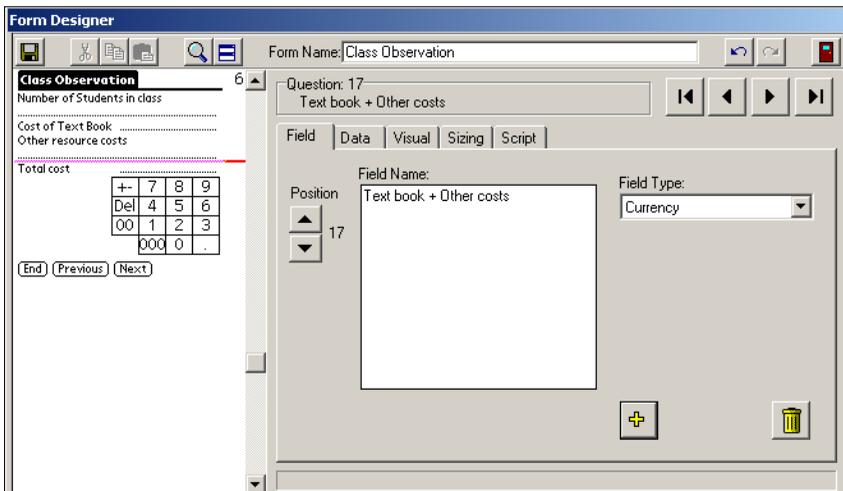
To make the field look better on screen, the Spacing slider is adjusted to add spacing before the question, and the Question Height slider is adjusted to display the picture and the whole question. Then, to prevent the user from skipping the section, the Answer Height is made zero to hide the Section Skip button.



With some fields, such as Date fields and Date & Time fields, it may not be necessary to give the handheld user instructions via the field name. In this case, you can make the Question Height zero.



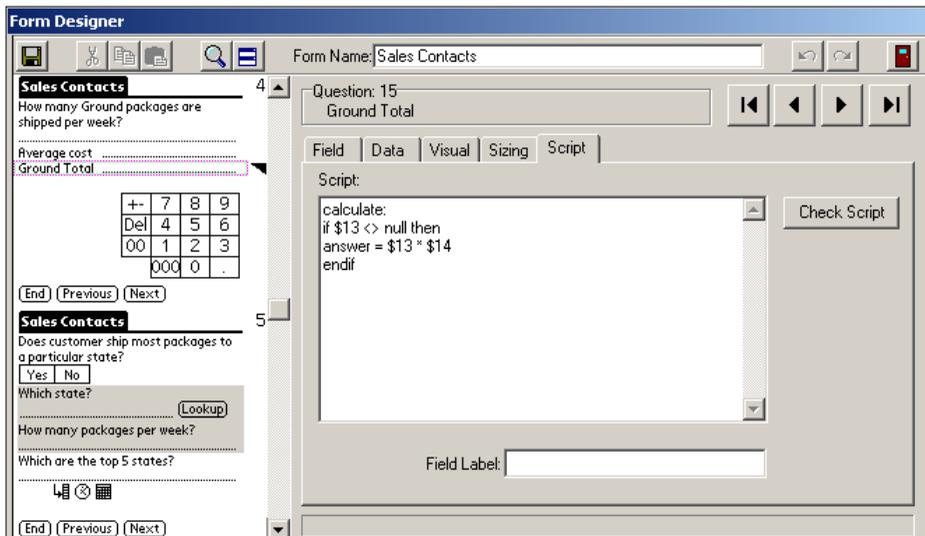
In rare cases, you may want to make both the question height and the answer height zero, so that the field does not appear on the handheld screen at all. This is different from making a field Hidden (see page 176), because hidden fields still use up space on screen. You may want to have a field with zero height to store a partial calculation, or for use in linking to another form. The Preview Area in the Form Designer shows the zero-height field as a red line. To display a zero-height field, use the Left and Right arrow buttons until the field is displayed.



Script Tab

The Script tab of the Form Designer is where you can write a script in a given field. A script performs an action, such as jumping the user to a particular field, or performs a calculation based on data the user has entered in various fields.

Chapter 13, *Scripting Reference*, page 245, provides information on the types of scripts you can write.

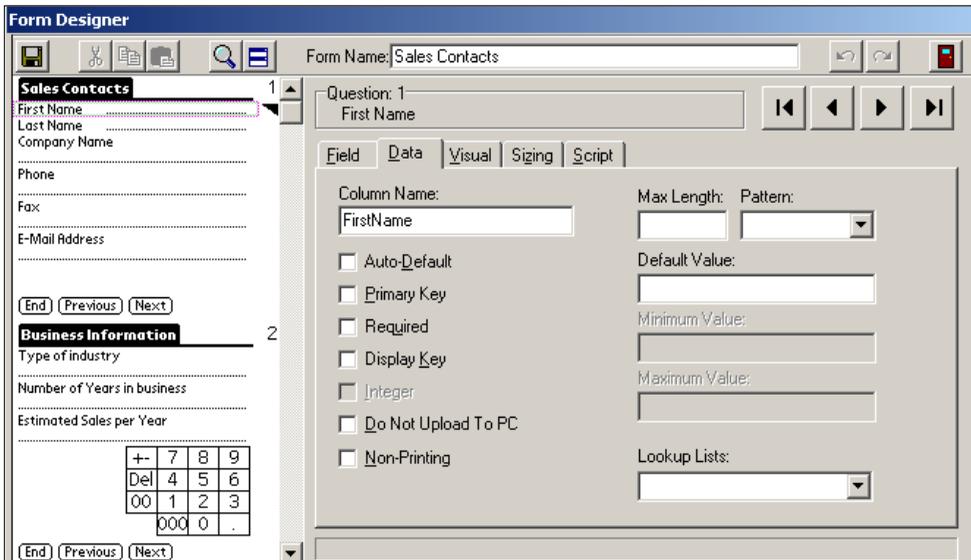


9. Advanced Field Properties

The Data tab of the Form Designer window displays the Advanced Field Properties available for the selected field. Advanced Field Properties give you extra control over the field.

If an Advanced Field Property does not apply to a particular field type, it will be grayed out.

Column Name, Primary Key and Max Length are Advanced Field Properties that must be set before a form is frozen, and cannot be changed after the form has been frozen. All other Advanced Field Properties can be set or changed whether or not the form is frozen.



Column Name

The Column Name on the Data tab of the Form Designer window is the name that the current field will be called in the database table that is created for storing the records for the form design. The database table is created when you freeze the form. You cannot change the column names after the form is frozen, or you will break the ability of the handheld to synchronize data to the PC.

- The default is that the Column Name is blank. Then, when you freeze the form design, the Column Name will be based on the name of the field, up to the first 64 characters of the field name. Only alphanumeric characters are used in the Column Name, so spaces and punctuation are ignored. If two fields are identical up to the first 64 characters of the field name, the Column Name will be modified upon freezing the form, so that each database column name is unique. (This is a Microsoft Access requirement.) To view the column names of a frozen form, click on the name of the form and then click the Edit/View button.
- If you prefer to use your own names for the database columns, type a name in the Column Name field. When you freeze the form design, you will be notified if there are any duplicate column names, and if so, you will need to edit the form so that each database column name is unique.
- If you copy a frozen form, the column names from the original form will be retained to make it easier to import data from the original form. If you re-name fields on the form, delete the old column name for each field before freezing the form, so that the new column names will be based on the new names of the fields..
- If you are linking to an external Access database (see page 486), the purpose of field-mapping is to ensure that the Column Name used in Pendragon Forms matches a column name in your external Access database. Without this match, data from the handheld cannot synchronize with your external Access database.
- Applies to: All fields.

5602664 : Form					
UserName	TimeStamp	ProductBarCode	ProductName	UnitPrice	ListedQuantity
No one	03 9:48:12 PM	012340	Fluffy Bunny	\$14.99	25
No one	03 9:48:32 PM	012341	Fluffy Ducky	\$14.99	18
No one	03 9:49:31 PM	023410	Action Soldiers	\$4.49	6
No one	03 9:50:11 PM	023415	Action Tanks	\$4.49	9
No one	03 9:51:38 PM	141590	Toy Airplane	\$9.99	12
No one	03 9:52:33 PM	141592	Toy Chopper	\$9.99	6
No one	03 9:53:07 PM	238637	Let's Play Store	\$19.95	5
No one	03 9:53:41 PM	238642	Let's Play Car C	\$19.95	2
No one	03 9:54:53 PM	238651	Let's Play Hous	\$19.95	4
No one	03 7:32:07 PM				

Autodefault

Turning autodefault on means that when the handheld user creates a new record, any field with autodefault on will be filled in with the same value from the previous record entered.

- The autodefault option is primarily a time-saving device. You can override the value entered in the field at any time. If you change the value for one record, that value becomes the new autodefault the next time the form is filled in.
- Autodefault fields default to the last value typed into the field or to the last value set by a script.
- Autodefault values are not retained if the form is re-distributed from the PC, or if the Forms application is re-installed on the handheld. In these cases, the handheld user will have to fill in a new record in order to re-set the autodefault values.
- Autodefault does not work within the fields of a subform.
- Applies to: Text, Numeric, Currency, Yes/No Checkbox, Popup List, Multi-Selection List, Lookup List, Exclusive Lookup List, Option 1-5, Date, Date&Time, Time.

Bar Code Inventory	Bar Code Inventory	Bar Code Inventory
Item Count: 1	Item Count: 1	Item Count: 2
Date - Set Date -	Date 4/24/03	Date 4/24/03
Building Location	Building Location North Building	Building Location North Building
Floor	Floor 2	Floor 2
Room Number	Room Number 201	Room Number 201
Item Bar Code	Item Bar Code 01357901	Item Bar Code
Item Name	Item Name P.d	Item Name
End Previous Next	End Previous Next	End Previous Next

The pictures above show a simple inventory form in which the handheld user is recording inventory a room at a time. The Date, Building Location, Floor and Room Number all have Autodefault switched on, so that after the user enters the first record, all subsequent records will keep the same values for these fields. The user can keep these values until the entire room has been inventoried, and then change the values when starting on a new room.

Item Count is a Numeric field which also has the Autodefault option switched on. A script in the field increments the value by one for each new record, so that the count goes up with each new item. See page 353 for the script required to make a counter.

Primary Key

A Primary Key is a field or combination of fields that uniquely identify a record. Pendragon Forms checks that when the handheld user tries to exit out of a new record, the Primary Key field(s) are filled in and are unique. If more than one field is used to define a Primary Key, the combination of primary key fields must be unique for each record.

- If the Primary Key box is left blank, then by default Pendragon Forms will create a Primary Key consisting of the UnitID, UserName and TimeStamp fields which are automatically created when a record is created on the handheld. You can use the default primary key if users do not have to share records.
- If you want several handhelds to share the same records, you must select a different Primary Key than the default. This is because each handheld user who modifies a record will have his/her handheld user name assigned to the UserName field, and this will cause a conflict with the existing records with the original UserName. If users are sharing records, you will also need to remove the default Additional Download Criteria setting which sends to the handheld only records whose UserName matches the handheld user name. See *Additional Download Criteria*, page 214. For general information on allowing users to share records, see pages 31-34.
- If you want to select your own Primary Key, check the Primary Key checkbox for the field or fields that you want to make part of the Primary Key.
On a form that is used for recording customer information, an example of a single field primary key might be a Customer Account Number that is different for each customer. If a form was being used to record visits to customers, you would want to allow seeing the same customer more than once, and so a multi-field Primary Key such as Customer Account Number and Date of Visit could be used.
- If you are linking to an external database, then you will need to select a Primary Key that is identical to the primary key in your database table.
- Applies to: Text, Numeric, Currency, Yes/No Checkbox, Popup List, Lookup List, Option 1-5, Date, Date&Time, Time.

Required

A Required field is a field into which the handheld user must enter data and cannot leave the field blank. You may want to make certain fields on your form required fields to ensure that the handheld user enters a response.

Note that requiring a user to fill in a field does not mean that the user will fill in correct data, it just means that the field will not be left blank.

Important: If a user does not fill in required fields, and then tries to synchronize, the records with the missing required fields will not be sent to the PC, as it is considered to be an unfinished record. If you want to allow a user to leave certain fields blank, do not make them required fields.

- On the Data tab of the Form Designer window, check the Required checkbox to make the selected field a required field.
- The default is that when the user exits a new record on the handheld, the Forms program will alert the user if any required fields have not yet been filled in. The record cannot be exited until all required fields are filled in. The problem with this default is that the user does not receive any warnings until the very end of the form.
- You can change the default so that when the handheld user moves off a screen containing a required field, the warning message is displayed immediately if the required field is empty. This may be more convenient for the user than waiting until the end of the form to perform the check. To change this default, set the Advanced Form Property of *Screen Level Validation* - see page 207.
- A script can be used to make a required field optional or an optional field required. See Chapter 13, *Scripting Reference*, pages 305 and 312.
- Applies to: Text, Numeric, Yes/No Checkbox, Popup List, Lookup List, Exclusive Lookup List, Option 1-5, Date, Date&Time, Time, Signature, Time Checkbox, Slider.

Display Key

The Display Key is used to display records on the handheld. When the handheld user chooses to review records on the handheld, a list of existing records is displayed on the handheld. The contents of the Display Key field of each record on the handheld is shown on the handheld.

- If the Display Key checkbox is left blank, the default is that the first field on the form is used as the Display Key field.
- Only one field can be the Display Key field. The first field on the form with the Display Key checkbox checked will be the Display Key.
- The Display Key field should be a field which is unique for each record. For example, a Text field or a Date field are good fields to use as Display Key fields. A Popup List or a Yes/No checkbox are less appropriate as Display Key fields, because several records can have the same value.
- When using subforms, you can make the Display Key on the subform different from the Display Key on the parent form. For example, the Display Key on the parent form might be 'Customer Name', and the Display Key on the subform might be 'Date of Visit'. Selecting a given customer, and jumping to the subform, displays a list of all visits.
- Applies to: Text, Numeric, Currency, Yes/No Checkbox, Popup, Lookup List, Option 1-5, Date, Date&Time, Time, Slider.



In the first picture above, the Display Key field is the first field on the form, which happens to be a customer's First Name. This field does not provide the user with much information, and so in the second picture, the Display Key field is changed to Company Name, a field which helps the user select which record he or she needs.

Picture three above shows that no matter what the Display Key field is, it is always possible on the handheld to select two more fields to view in addition to the Display Key field. See page 61.

Default Value

A default value is a value that will be filled in if the handheld user does not fill in a field.

For example, in a Yes/No checkbox field, if the handheld user leaves the field blank, then the data that goes back to the PC is a null value - blank. If you want to change things so that leaving a Yes/No checkbox blank is the same as responding No, you can enter a default value of N.

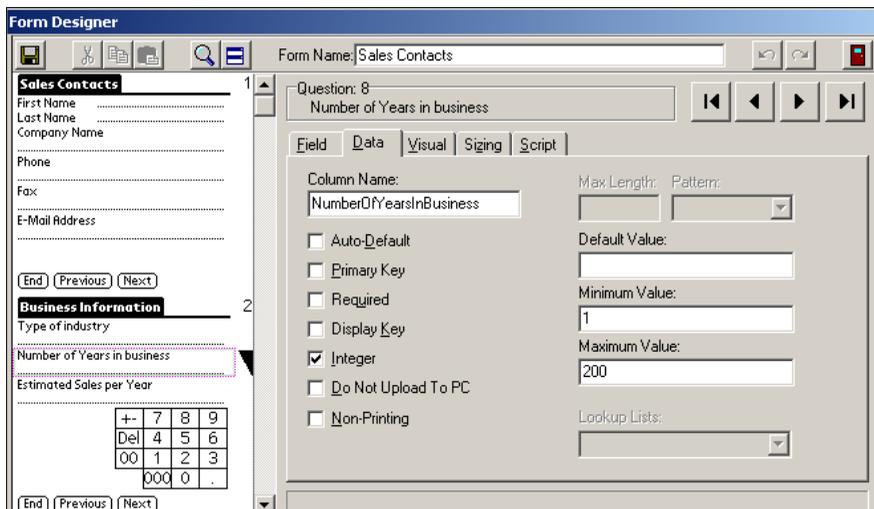
- Type a value in the Default Value field if you want a default to apply to the selected field.
- For a Yes/No field, type Y to default to Yes, or type N to default to No.
In a Popup or Lookup List field, type one of the options that appears in the list.

The screenshot shows a form titled "Cruise Vacation Form" with a purple header. Below the title is a small icon of three people. The form contains the following text and controls:

- Questions for all cruise vacationers:
- What type of cabin do you prefer?
▼ Select one...
- How many people will be travelling?
- Do you have a preferred cruise line?
- Navigation buttons:

- Date Only and Date & Time fields can only be made to default to a specific date or date and time, not the current date and time. Enter the date in the Short Date format setting that you are using in Windows. For example, to default to September 15th 1999, you can enter 09/15/1999 into the Default Value field. To default to the current date or current date&time, see Chapter 14, *Scripting Examples*, page 354.
- To make a Time field default to a specific time, you need to enter a date as well as a time. For instance, to default to 9:00AM, you will need to enter something like 7/31/1999 9:00 AM. To default to the current time, see *Scripting Examples*, page 354.
- On a Slider field, the Default Value sets the default and positions the marker of the scale at that value on the scale.
- The Default Value field has special meaning on a Button field. Typing a word into the Default Value field creates the label on the button. You can enter up to 11 characters for the name of a button.
- When you exit the Form Designer, a check will be performed to ensure that the default value that you entered is valid. For example, you cannot enter a default of Other in a Yes/No field, or a default of Purple in a popup field whose options are Red, Green, Blue.
- Applies to: Text, Numeric, Currency, Yes/No Checkbox, Popup List, Lookup List, Exclusive Lookup List, Option 1-5, Date, Date&Time, Time, Slider.

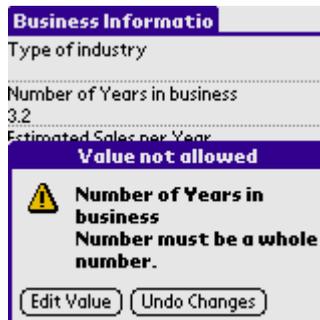
Three Advanced Field Properties apply to Numeric fields only: Integer, Minimum Value and Maximum Value.



Integer

The Integer option applies to Numeric fields and Slider fields. If the Integer checkbox is checked, the handheld user can only enter a whole number in a Numeric field.

- On the Data tab of the Form Designer window, check the Integer checkbox if you want to require the handheld user to enter a whole number (no decimal places) in a Numeric field.
- On the handheld, as the user exits the screen, the program checks whether the number entered is an integer or not. A message is displayed if the number is not an integer, and the program allows the user to edit the field to make it a whole number.
- Slider fields are a type of numeric field, and have the Integer checkbox checked by default.
- Applies to: Numeric fields and Slider fields only.

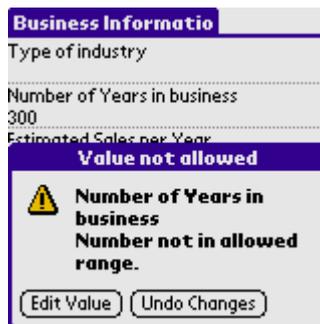


Minimum Value & Maximum Value

Minimum Value and Maximum Value apply to Numeric fields and Slider fields only.

By specifying a maximum value and a minimum value, the handheld user must enter a number within the range specified.

- Both a minimum value and a maximum value have to be entered on the Data tab of the Form Designer in order to specify the range on the handheld. You cannot enter one value and leave the other one blank.
- If the user enters a value outside of the valid range, then when the user tries to leave the screen, an error message is displayed to alert the user that the value entered is not in the allowed range. The user can then edit the value.
- On a Slider field, the Maximum Value and Minimum Value specify the ends of the slider scale. The default range on a Slider field is 1-10. You can change this default.
- Applies to: Numeric fields and Slider fields.



Three Advanced Field Properties apply to Text fields only: Max Length, Pattern and Lookup List.

Max Length

The maximum length only applies to Text fields.

- By default, the maximum length of a response in a text field is 255 characters. This is the maximum length allowable in Microsoft Access for a text field which can be sorted.
- You can change the maximum length to a number of characters from 1 to 2000, if you need to allow a shorter or longer response in the field. (Access is very efficient at storing data, so if you do not need long responses, leave the default at 255.) If you change the length of a Text field to more than 255 characters, internally Microsoft Access will store the response as a Memo field which cannot be sorted on the PC.
- When you freeze a form design, a database table is created in the Forms Manager for storing the records associated with your form. Each field on your form corresponds to a column in the database table. If you set a Max Length on a Text field, then when you freeze the form, that column will freeze with the specified maximum number of characters. This means that once a form is frozen, you cannot make the Max Length any larger. If you make the Max Length smaller, the database column will not get any smaller, but the handheld will allow only the smaller number of characters to be entered.
- If a form is going to be used as a Read-Only Reference form (see pages 127 and 221), the Max Length of all Text fields on the form should be set to the smallest value possible. This is because Read-Only Reference forms are stored as fixed-width columns on the handheld, and the default size of 255 characters in a Text field will waste space on the handheld.
- Applies to: Text fields and Lookup List fields (since the handheld user can enter text in a Lookup List field).

Pattern

Pattern applies to Text fields. You can specify that the handheld user must enter characters of a certain type.

- Possible character patterns include Alpha characters only, Alphanumeric, Digits only, Printable characters (i.e.: no Tabs or carriage returns), Uppercase or Lowercase.
- If you choose a pattern of Alpha, Alphanumeric or Digits, and the handheld user enters characters which do not conform to the pattern, a dialog box will warn the user that the text pattern is not valid. Ideally, in your field name you should inform the user of the text pattern that you expect to be entered.

- If you choose Uppercase or Lowercase as the pattern, data entered into the Text field will be converted to upper or lower case as necessary, after the user leaves the field..
- Note that if you specify a character pattern, no characters outside that pattern will be allowed. For example, if you specify Digits only, the handheld user can enter 1234567 but not 123-4567 and not 123 4567.
- Applies to: Text fields only.

Lookup List

With a Lookup List field, it is only possible to select one item from the list. The Lookup List item on the Advanced Field Properties window allows you to use a Lookup List with a Text field, enabling the handheld user to select more than one item from the list and to paste each selection in turn into the Text field.

- On the Text field, go to the Advanced Field Properties window and select a Lookup List to be used with the Text field.
- The handheld user will see a Lookup List icon in the Text Field. The cursor has to be blinking in the field for the Lookup List icon to be available. Tapping on the Lookup List icon allows the user to select an item from the list and paste it into the Text field. The user can tap on the Lookup List icon more than once.
- Applies to: Text fields only.

Sales Contacts

Does customer ship most packages to a particular state?

Yes No

Which are the top 5 states?

FL: IL: OH: NY: CA

Do Not Upload to PC

The Do Not Upload to PC option allows you to choose not to upload data from the handheld to the PC for the selected field.

- To mark a field as non-uploadable to the PC, check the Do Not Upload to PC checkbox on the Data tab of the Form Designer window.
- You may choose not to upload data for a field if, for example, you are linking to an external Access database that has an AutoNumber field which will be generated on the desktop. A number that is generated on the handheld in this field does not need to be sent to the PC.
- If there are fields that you want to only be able to update on the PC, you can make these fields non-uploadable. For example, imagine that you have an Appointment Time field into which you enter 9:00am on the PC, and then send the record to the handheld. Back at the office, the customer calls to change the time to 9:30am. If the handheld modifies the record and synchronizes, the old appointment time of 9:00am will be uploaded to the PC. To prevent the handheld from overwriting the PC in this field, the Appointment Time field can be made non-uploadable.
- You may not want to upload fields that contain calculated results which the desktop PC will calculate.
- Applies to: All fields.

Non-Printing

The Non-Printing option allows you to select which fields should not be printed if a record is printed directly from the handheld using the Print scripting statement, or if a record is e-mailed using the Transmit MultiMail scripting statement.

- To mark a field as non-printing, check the Non-Printing checkbox.
- Any fields marked as Non-Printing will not be selectable for display on the Review screen, and cannot be searched or sorted. You may want to make fields such as Section fields and Button fields Non-Printing, as these fields do not contain any data.
- Applies to: All fields.

10. Form Properties

Before you send a form to the handheld, there are some Form Properties that you must set. The Form Properties window is where you select the Data Persistence rules that determine how long records are kept on the handheld. The Form Properties window is also where you freeze a form design in order to create a database table for storing the records associated with that form.

From the Form Properties window you can also access Advanced Form Properties that give you some control over what you want the handheld user to be allowed to do in the form.

Form Properties and Advanced Form Properties can be changed after freezing a form. You will need to redistribute a form design and synchronize for any changes to take effect on the handheld.

Form Properties

To access the Form Properties window, click the name of your form in the Forms Manager, then click the Properties button.

The screenshot shows the 'Form Properties' dialog box with the following fields and options:

- Identification:**
 - Form: Customer Order
 - ID: 234712307
 - Table Name: (empty)
 - ASCII File: DFD6CF3.OUT
 - Query Name: (empty)
 - Category: Unfiled (dropdown menu)
 - Field Map
- Data Persistence:**
 - Keep a copy of records on handheld
 - Keep new records on handheld for 0 days.
 - Keep incomplete records on handheld
- Access Rights:**
 - No additions on handheld
 - No updates on handheld
 - No deletions on handheld
- Freeze Form Design:**
 - Freeze Form design for distribution to handheld and create database. (button)

At the bottom of the dialog are four buttons: Help (with a question mark icon), Field Mappings..., Advanced Properties..., and OK.

Data Persistence

The Data Persistence section of the Form Properties screen allows you to choose how long records remain on the handheld.

You can change Data Persistence options if needed. Re-distribute the form and synchronize for the changes to take effect on the handheld.

The Data Persistence options are described here:

Data Persistence Option	Meaning
Default option - No checkboxes checked	Records are removed from the handheld during synchronization.
Keep a Copy of Records on Handheld	Records remain on the handheld after synchronization. Checking this box overrides the other data persistence options. If you are linking to an external Access database, you must check this option, and use the Advanced Form Property of Additional Download Criteria to control which records remain on the handheld. See pages 214-218.
Keep New Records on Handheld for X Days	If you leave the Keep a Copy of Records on Handheld checkbox blank, and instead enter a number of days, then after synchronization, records will continue to be stored on the handheld for that number of days. The number X can be from 0 to 999.
Keep Incomplete Records on Handheld	This option can only be used if you have a Completion Checkbox field on your form. See page 107 for information on Completion Checkbox fields. If this option is selected, then only records that have the Completion Checkbox field checked as Yes, will be removed from the handheld during synchronization.

Access Rights

In the Form Properties window, the Access Rights section allows you to choose if the handheld user can add, modify or delete records. You can choose any combination of the Access Rights checkboxes.

You can change the Access Rights even after a form has been frozen. Re-distribute the form and synchronize for the changes to take effect on the handheld.

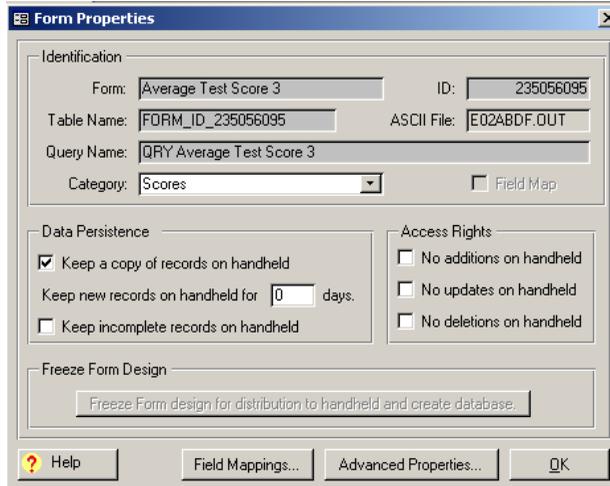
The Access Rights options are shown here:

Access Right	Meaning
Default option - No checkboxes checked	The handheld user is allowed to add new records, modify existing records, and delete records.
No additions on handheld	<p>Check this checkbox if you do not want the handheld user to be able to create new records.</p> <p>This option can be used if you are creating all the records on the PC and then sending them to the handheld, and users are not allowed to create their own new records.</p>
No updates on handheld	<p>Check this checkbox if you do not want the handheld user to be able to modify existing records.</p> <p>Use this option with care. If you want to allow users to start creating a record and then be able to go to other handheld applications and then come back to the record, you cannot use this option. Once a user leaves the record, and comes back to it, the user will be modifying an existing record which is not allowed when No updates is selected.</p>
No deletions on handheld	<p>Check this checkbox if you do not want users to delete any records from the handheld.</p> <p>Use this option with care. If a user starts a new, blank record, and then wants to delete the blank record, they will not be able to.</p>

Category

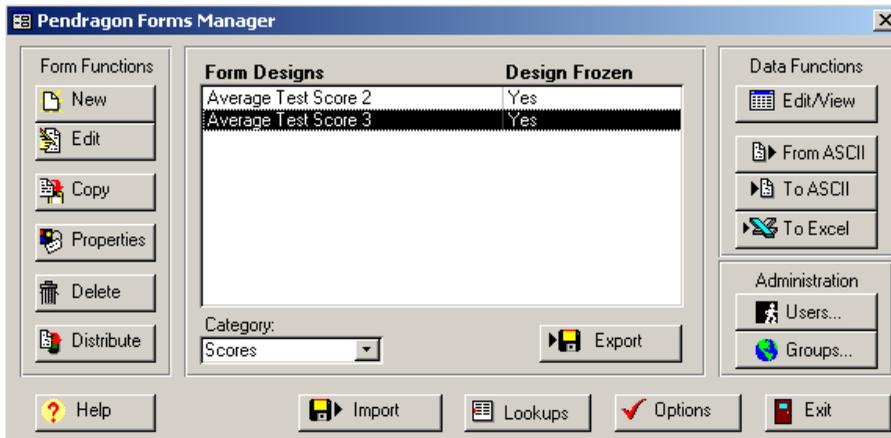
The Category field in the Form Properties window allows you group form designs into categories.

To create a new category, type a name for the category in the Category field. To assign a form to an existing category, click the arrow in the Category field and select a category.



To view the forms in a Category, select the category name in the Forms Manager window. The Recycle Bin is a unique category that stores deleted form designs.

To take a form out of a category, including the Recycle Bin, first view that category in the Forms Manager, then click on the form, and click the Properties button. In the Category field of the Form Properties window, choose Unfiled as the category.



Freezing a Form

Freezing a form design creates a database table in the Forms Manager database for storing records for a particular form design. You must freeze a form design before you can send it to the handheld.

Once you freeze a form design, you cannot add fields to the form, delete fields from the form or change the field types. To do any of these things, make a copy of the form and modify the copied form, which will not be frozen. See page 39 for information on copying form designs.

To freeze a form design, click the name of the form in the Forms Manager, then click the Properties button. In the Form Properties window, click the button labeled Freeze Form Design for Distribution to Handheld and Create Database. Then click the OK button to close the Form Properties window. Typically, the next activity is to distribute the form by clicking the name of the form in the Forms Manager, then clicking the Distribute button. Synchronize the handheld to send the form to the handheld.

If you click the Freeze Form Design button and receive an error message that prevents the form from being frozen, see Appendix B, *Troubleshooting*, page 531.

Once a form design has been frozen, the Table Name and Query Name fields in the Form Properties window are filled in. The Query Name is the name of the query that makes it possible to view the data for the form when you click the Edit/View button in the Forms Manager. The Table Name is the name of the Access database table that was created to store the records for the form in the Forms Manager database. You must NEVER delete the Access database table for a form if the form is still in use on the handheld, as deleting the table will break the ability to synchronize the form.

The screenshot shows the 'Form Properties' dialog box with the following details:

- Identification:**
 - Form: Customer Order
 - ID: 234712307
 - Table Name: FORM_ID_234712307
 - ASCII File: DFD6CF3.OUT
 - Query Name: QRY Customer Order
 - Category: Unfiled
 - Field Map:
- Data Persistence:**
 - Keep a copy of records on handheld
 - Keep new records on handheld for 0 days.
 - Keep incomplete records on handheld
- Access Rights:**
 - No additions on handheld
 - No updates on handheld
 - No deletions on handheld
- Freeze Form Design:**
 - Freeze Form design for distribution to handheld and create database.

Buttons at the bottom include Help, Field Mappings..., Advanced Properties..., and OK.

Advanced Form Properties

Advanced Form Properties give you additional control over how a form works on the handheld.

To access the Advanced Form Properties window, click on the name of a form in the Forms Manager, then click the Properties button. In the Form Properties window, click the Advanced Properties button.

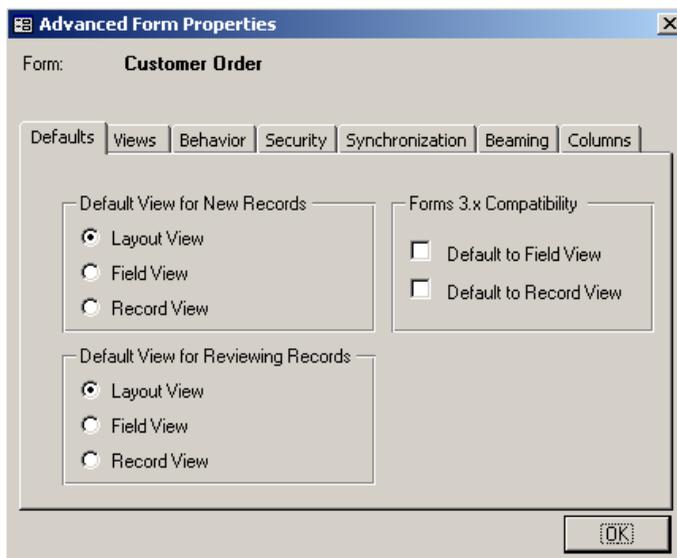
You can change the Advanced Form Properties of a form at any time. In most cases, you will need to re-distribute the form and synchronize for the changes to take effect on the handheld.

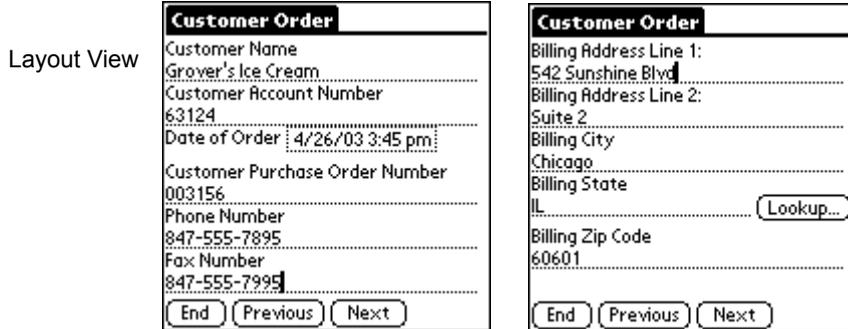
Each tab of the Advanced Form Properties window covers a different aspect of the form.

Defaults Tab

On the Advanced Form Properties screen, the Defaults tab lets you choose how the form appears when the handheld user is creating a new record, and how the form appears when the handheld user is reviewing an existing record.

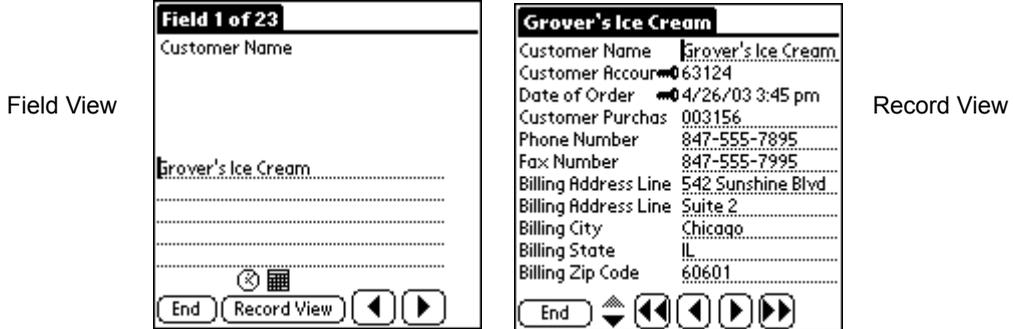
There are three possible views of a form on the handheld: Layout View, Field View and Record View. These three views are described on the next page.





Layout View, shown above, is the default view for both new and reviewed records on both Palm OS and Pocket PC devices. When you are designing a form in the Form Designer, Layout View gives you the most flexibility for choosing how many fields appear on a screen, and how many lines on the screen the question and answer components of a field occupy.

Field View and Record View, shown below, were available in earlier versions of Pendragon Forms. Field View displays one field per screen. Record View displays up to 11 fields per screen in a two-column format, questions on the left and answers on the right. Field View and Record View were not available on Pocket PC devices in Forms version 5.0, but are available in Forms 5.1.



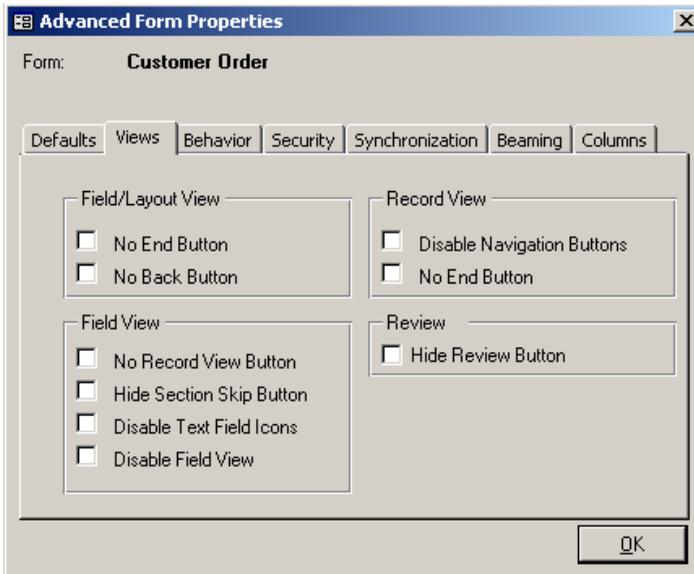
Since Layout View is the default, if you want to use Layout View for both new and reviewed records, you do not need to make any changes on the Defaults tab of the Advanced Form Properties window.

If you want to change which view is used for new or reviewed records, select the appropriate button in the Default View for New Records and the Default View for Reviewing Records sections of the Defaults tab.

For backward compatibility with earlier versions of Forms, the Default to Field View and Default to Record View buttons are present, as these options were available previously. These options do not apply to new forms that you create in Forms version 5.0/5.1.

Views Tab

In the Advanced Form Properties window, the Views tab gives you options for controlling whether or not certain buttons appear on the handheld screen. The buttons that appear by default on the handheld screen depend on whether you are in Layout View, Field View or Record View. (See page 203 for descriptions of each of these views.) The Views tab is therefore divided into sections that apply to each view. You can change any of these options at any time, and then re-distribute the form and synchronize for the changes to take effect on the handheld.



No End Button (Field/Layout View)

The End button allows the handheld user to exit a record at any time. If you want to force the handheld user to go through every field on the form before exiting a record, check the No End Button checkbox in the Field/Layout View section of the Views tab. This will hide the End button in Layout View or Field View. The handheld user can only end the record by tapping the the Next button in Layout View or the right arrow button in Field View on the last screen of the form.

No Back Button

In Field View or Layout View on the handheld, there is a Previous button which allows the handheld user to go back to the previous screen of the form. If you want the handheld user to enter a response once and then move onto the next field, and you want to prevent the user from going back and changing answers, you can remove the Back button from the handheld screen. (See also Hide Review Button, page 206.)

No Record View Button

Field View contains a Record View button that lets the user switch to Record View at any time. If you want the handheld user to enter records in Field View only, you can check the No Record View Button checkbox to remove the Record View button from the Field View screen. The handheld user will not be able to toggle between Field View and Record View. Hiding the Record View button may be useful if you are using Field View and you have branching scripts on your form.

Hide Section Skip Button

Section fields, when viewed in Field View, have a Skip button that allows the handheld user to skip a section and jump to the next section. If you check the Hide Section Skip Button checkbox, the handheld user will not be able to skip over a section. This may be useful if you want the handheld user to go through every section on the form, or if you want to use a section field just to provide instructions to the user. The Hide Section Skip Button option for Field View will apply to all Section fields on the form.

Do not use this Advanced Form Property if you are using Layout View. Instead, you can open the Form Designer window, display the Section field, click the Sizing tab and make the Answer Height of the field zero. Layout View allows you to choose whether or not to hide the Section Skip button for each individual Section field individually.

Disable Text Field Icons

When a Text field is viewed in Field View (one field at a time), two icons appear at the bottom of the Text field - the calendar icon allows the handheld user to insert the current date and time, the clock icon allows the user to insert the current time. If you do not want to allow the handheld user access to these icons, check the Disable Text Field Icons checkbox.

Do not use this Advanced Form Property if you are using Layout View. In Layout View, the Visual tab of the Form Designer window allows you to select whether Text Field Icons are present or not. The default is that Text Field Icons are not visible.

Disable Field View

In Record View (viewing eleven fields at a time), if the handheld user taps on the name of a field, that is, taps in the left-hand column of the Record View screen, then the user will switch to Field View, viewing one field at a time. If you do not want the handheld user to use Field View at all, check the Disable Field View checkbox.

Disable Navigation Buttons

This applies to Record View only.

In Record View (viewing eleven fields at a time), a series of navigation buttons (left and right arrow buttons) at the bottom of the Record View screen allow the handheld user to jump from one record to the next, and also to jump to the first and last records. If you want the handheld user to stay within a record, and not have the ability to jump across records, you can check the Disable Navigation Buttons checkbox.

No End Button (Record View)

This applies to Record View only.

The End button in Record View allows the handheld user to end the record at any time. If you prefer the user to go through every field on the form, you can check the No End Button (Record View) to hide the End button in Record View.

Warning: If you hide the End button in Record View, the handheld user will not have a way to exit the record. You should typically add a Button field at the end of the form, with a click: event script that allows the user to end the form.

Hide Review Button

This applies to any form.

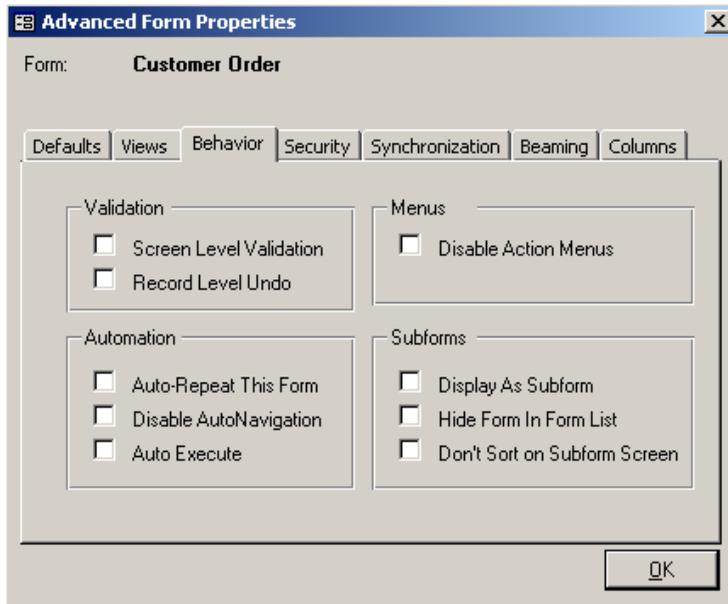
On the main Forms screen that displays the list of forms on the handheld, there are two buttons: the New button allows the handheld user to create new records for the form, and the Review button allows the user to review existing records.

If you do not want users to be able to go back and review records, check the Hide Review Button checkbox. Users will still be able to create new records, but not to review the records.

If the reverse is true, i.e., you want users to be able to review records but not create new records, leave the Hide Review Button unchecked, and instead change the Access Rights on the Form Properties screen to not allow additions on the handheld (see page 199).

Behavior Tab

The options on the Behavior tab of the Advanced Form Properties window allow you to change the behavior of a form on the handheld. You can change Advanced Form Properties on the Behavior tab at any time. Simply re-distribute the form and synchronize for the change to take effect on the handheld.



Screen Level Validation

If you have Required fields on your form (see page 189), you may want to check the Screen Level Validation checkbox. This causes the program on the handheld to display a warning message if the user leaves a Required field blank and tries to move to the next screen of the form. If Screen Level Validation is not selected, then Required Fields will only be checked when the user exits the record, not on a screen by screen basis.

Record Level Undo

If Record Level Undo is checked, then when the handheld user is modifying an existing record on the handheld, and taps the End button to exit a record, a dialog box will ask user whether changes are to be kept. If the user chooses Yes, then the record is modified. If the user chooses No, then changes are thrown away, and the record reverts to the way it was before the changes were made.

AutoRepeat This Form

If you want the handheld user to repeatedly enter records on the form, you can check the AutoRepeat This Form button. When the user finishes filling in the last field on the form for one record, the form cycles back to the first field to begin data entry for the next record.

- To break the AutoRepeat cycle, the handheld user can tap the End button on the form to end the current record. When the form autorepeats to create a new record, tap End again and choose to delete the empty new record.
- Alternatively, the handheld user can tap the handheld Menu button, then tap the Record menu and select Delete/Cancel.

Disable AutoNavigation

If using Field View on the handheld, the user can check the AutoNavigate button to speed up data entry (see page 72). When AutoNavigate is switched on in Field View, Popup Lists, Lookup Lists, Date and Time fields automatically pop up, and as soon as a selection is made, the handheld user is advanced to the next field on the form. The user can switch off AutoNavigate on the handheld if desired.

If you want to ensure that AutoNavigation is never used for a particular form, you can check the Disable AutoNavigation checkbox. The handheld user will be able to switch AutoNavigate on and off for other forms but not the selected form.

Auto-Execute

Automatically creates a new record when Forms is launched on handheld. Can only be used for one form on the handheld.

Used with Custom Main Menu forms - see page 471.

Disable Action Menus

On the Review screen of the handheld, if the user taps the Menu button (the drop-down menu icon on Palm OS devices; tapping the title bar on Pocket PC devices), he/she can access the following menu options: New, Clone (copies a record, including all filled in fields), Mark All Changed (manually flags every record on the handheld as new to force an update to the PC), Print All.

If you do not want the handheld user to have access to these menu options, check the Disable Action Menus checkbox.

Display as Subform

On the handheld, parent forms and subforms are displayed in the list of forms. To enable data from the parent record to be copied into the subform, it is necessary to always enter data starting from the parent form.

If you check the Display as Subform checkbox, this will ensure that the handheld user cannot enter records in the subform by starting from the subform. The subform will still be visible in the Forms List on the handheld, and the user can review records but not create new subform records except by starting from the parent form. The icon next to the subform will be different from that of regular forms.

- Another solution is hiding the subform from the Forms List entirely. See *Hide Form in Forms List*, below.

Hide Form in Forms List

When using a parent form and one or more subforms or single subforms, the handheld user has to always enter data from the parent form, in order for data to be copied from the parent to the subform. On the Forms List on the handheld, the user will see both the parent form and the subforms listed. If you check the Hide Form in Forms List checkbox for a subform, the subform will not appear in the Forms List on the handheld, and this gives the user the appearance that there is just one form in which to enter data, namely the parent form.

Don't Sort on Subform Screen

By default, when subform records are viewed from the parent form, they appear sorted by the Display Key (see page 190) of the subform. This also applies to records in a reference form which are being accessed by a Lookup to the reference form (see page 127-128).

The sorting of records on the subform screen (or reference form screen) is done on the handheld, which may slow down performance on the handheld. If you have a large number of records in the subform or reference form, it will take extra time to sort the records before displaying them.

In some instances you might not need to sort the records, for example, if you are using bar codes, and are scanning a bar code to find a record in the reference form.

In other instances, you can set Additional Download Criteria to sort the records on the PC before sending the records to the handheld, and switch off sorting on the subform screen.

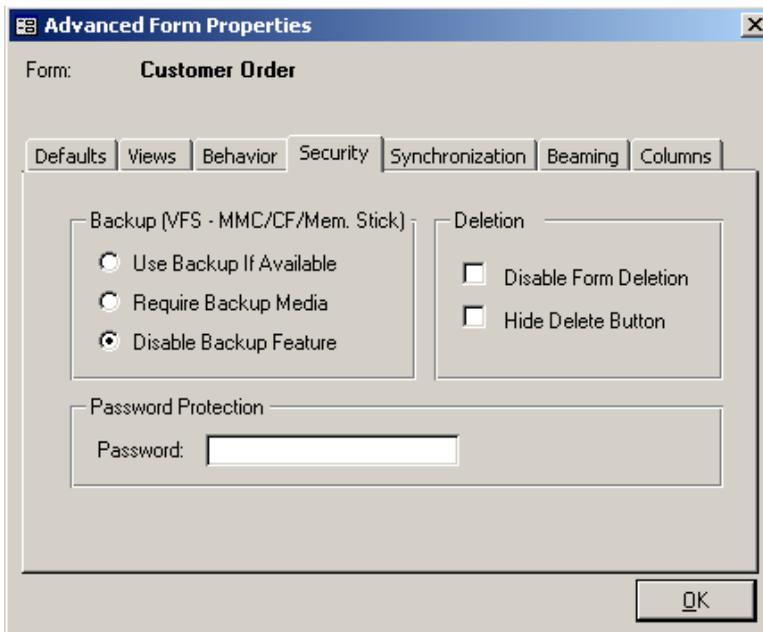
Check the Don't Sort on Subform Screen to switch off sorting records.

Security Tab

The options on the Security tab of the Advanced Form Properties window allow you to:

- Limit access to the form design on the handheld by not allowing deletion.
- Password protect a given form design.
- Choose to back up data to an external storage card.
For this option to apply on Palm OS devices, the device must support VFS, Virtual File System.

You can change Advanced Form Properties on the Security tab at any time. Simply re-distribute the form and synchronize for the changes to take effect on the handheld.



VFS Backup (Backup to an External Storage Card)

See page 512 for instructions on performing a VFS backup and restore on the handheld, and for information on the limitations of the VFS Backup mechanism.

Virtual File System, or VFS, is a feature of Palm OS 4.0 (and higher) that allows the handheld to store data on external media such as a compact flash card, an SD (Secure Digital) card, an MMC (Multi-Media Card) or a memory stick. If your handheld device has Palm OS 4.0 or higher, and supports VFS, then the VFS Backup options in Pendragon Forms can be used. Examples of handhelds that support VFS are the Palm Treo series, the Palm E2, Palm TX and Palm LifeDrive.

New in Pendragon Forms 5.1 is backup support for Pocket PC devices running Windows Mobile 5 for Pocket PC. Pocket PC devices do not use VFS, but have their own file system that supports backing up data to an external media card (typically a Secure Digital card - an SD card). **Note to Pocket PC users:** Instead of using VFS Backup, you can also consider installing Pendragon Forms 5.1 directly to an external SD card (see page 8). If the handheld device fails, you can install Pendragon Forms to an external card on a replacement handheld device and then swap in the original card with your existing form designs and data.

The VFS Backup options are:

- **Use Backup If Available**

This option allows the handheld user to use the VFS Backup if a media card is plugged into the handheld, and to ignore the backup if the media card is not present or is out of memory.

If this option is selected, then if the external media card is plugged into the handheld, every new record and every modified record for this form design will be written to the external media card in addition to being written to the Forms application in RAM on the handheld.

If the card is NOT plugged into the device, no backup will occur.

- **Require Backup Media**

This option means that an external memory card must be plugged into the handheld device for the handheld user to be able to create new records, or review or modify existing records. If the memory card is not plugged into the handheld, the user will be prevented from creating, viewing or modifying records on the handheld, and will receive an error message to this effect.

- **Disable Backup Feature**

This option is the default setting. It means that no VFS backup is performed even if there is a media card plugged into the handheld.

If you change the VFS Backup option, you will need to re-distribute the form and synchronize for the change to take effect.

Disable Form Deletion

Check this box if you do not want the handheld user to be able to delete the form from within the Forms application on the handheld.

- Note that the handheld user can still delete the entire Forms application and all forms and data via the Delete menu on the Applications screen on Palm OS devices, or via File Explorer on Pocket PC devices.
- The proper way to remove a form from the handheld is to remove the form from the Default User Group or other User Group in the Forms Manager on the PC, and then synchronize the handheld. (See *How to Delete Forms*, page 50). This applies whether or not Disable Form Deletion is switched on.
- If Disable Form Deletion is switched on and you change your mind and decide to allow the deletion of the form design from the handheld, you will have to un-check Disable Form Deletion and then re-distribute the form and synchronize for the change to take effect.
- If you lose the form design on the PC, and Disable Form Deletion is switched on, there is no way to switch Disable Form Deletion off. In this scenario, the only way to remove the form from the handheld is to delete the entire Forms application, including all form designs and data. Before you delete the Forms application, synchronize to back up your data for the form designs that you do have in the Forms Manager on the PC.

Hide Delete Button

If you check the Hide Delete Button checkbox for a form, then when the user taps on the form in the Forms List on the handheld, the Delete button will disappear. This means that the user cannot go into the form and delete records.

- It is still possible for the user to delete the form and thus delete all associated records. To prevent deletion of the form from within Pendragon Forms, see *Disable Form Deletion*, above.
- If you need to allow the user to delete records, simply un-check the Hide Delete Button, then re-distribute the form to the handheld and synchronize the handheld. Data on the handheld will not be affected by re-distributing the form.

Password Protection

If you want to password protect a form design, so that the handheld user cannot access the form design without putting in a password, type the password in the Password field. The password is not case sensitive.

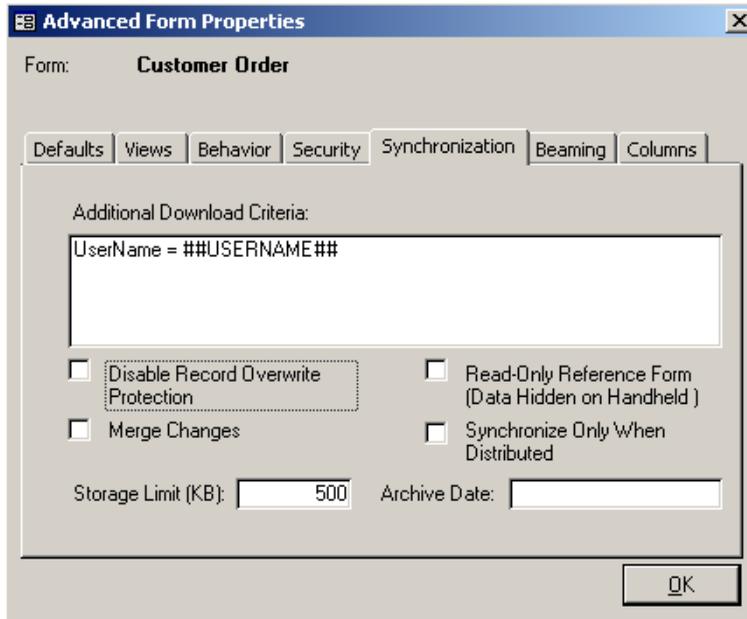
If you forget the password on the handheld, type a new password in the Advanced Forms Properties screen, then re-distribute the form and synchronize the handheld. To stop requiring a password on a form design, erase the password, then re-distribute the form and synchronize.

Warning: Password protecting a form design only prevents the data from being accessed via Pendragon Forms. However, since records on the handheld are not encrypted, a person with technical knowledge of the Palm OS or Pocket PC operating system would be able to access the records even without knowing the password.

Some handhelds with Palm OS 5.2 or higher have built-in strong encryption. On other devices, it is possible to purchase security software that encrypts handheld data. In both cases, encryption may be selective, so you should verify that the security mechanism is encrypting Pendragon Forms and its databases in particular.

Synchronization Tab

The options on the Synchronization tab of the Advanced Form Properties window allow you to specify which records are sent from the PC to the handheld, and also provide some control over data going from the handheld to the PC.



Additional Download Criteria

Additional download criteria allow you to select which records from the PC are sent to the handheld.

The default setting is:

UserName = ##USERNAME##

This means that by default, only records where the UserName field matches the handheld user name, will be sent to a particular handheld. ##USERNAME## is a wildcard used by Pendragon Forms to refer to the handheld user name.

When the default setting is on, users cannot share records. If you need several handhelds to share the same records, you will need to select a different primary key (see Primary Key, page 188) and then delete the UserName = ##USERNAME## statement from the Additional Download Criteria field. Then records from all handhelds will be sent to all handhelds.

In general, the selections made in Data Persistence section of the Form Properties window (e.g.: *Keep a copy of records on Handheld*, or *Keep new records on Handheld for X days* or *Keep incomplete records on Handheld*) are sufficient for determining which records are sent to the handheld. If this is the case, you do not need to specify any Additional Download Criteria.

However, if you are linking to an external database and are not storing the UserName and TimeStamp fields that Pendragon Forms generates for each record, then the Data Persistence options will not apply. In order to select which records go to the handheld, you must specify Additional Download Criteria before field-mapping to the external database. (See page 486 for information on linking to an external Access database.)

The Additional Download Criteria is an SQL WHERE clause (without the word 'Where') which is used to specify the selection criteria for sending records to the handheld.

The WHERE clause has the format:

[Database-Column-Name] = Criteria

When a form is frozen, each field is assigned a database column name. The database column name for a field can be viewed on the Advanced Field Properties screen, that is, the Data tab in the Form Designer window (see pages 185-186). You have to use the database column names of your form when specifying Additional Download Criteria.

Date Criteria

Additional Download Criteria can be used to select only the records that meet a date-specific criteria. For example:

[DateofVisit] > now - 30

To send only records within the last 30 days to the handheld, you will need a Date field or a Date&Time field on your form. In this example, there is a field on the form called Date of Visit. The word **now** represents the current date and time. This download criteria will send only those records whose Date of Visit field is in the last 30 days.

[AppointmentDate] > #05/15/2003#

In this example, there is a field on the form called Appointment Date. This download criteria will only send records to the handheld if the value in the Appointment Date field is after May 15th 2003. Enter the date in the Windows short date format used on your PC (see the Control Panel in Windows). The # symbol is used by Access as a quotation symbol when specifying dates in SQL.

Selection List Criteria

Your Additional Download Criteria can depend on the value in a Yes/No Checkbox, Popup List, or Lookup List. It is not recommended for download criteria to depend on the value in a Text field, because users may not all write the exact same text in the Text field. With a selection list field, however, you can guarantee what the possible values in the field are.

Here is an example of using selection list criteria:

```
([WorkOrderStatus] = "New") OR ([WorkOrderStatus] = "In Progress")
```

In this example, there is a Popup List or Lookup List field called Work Order Status on the form. Two of the options in the Popup or Lookup List are the phrases New and In Progress.

The download criteria sends records the handheld if the Work Order Status field is either equal to New or In Progress.

The OR statement allows for either the first criteria (New) or the second criteria (In Progress) to apply.

When selecting download criteria based on a Popup List, Lookup List or Yes/No Checkbox field, it is important to include a download criteria if a field is blank. For example, the above example could be changed to:

```
[WorkOrderStatus] = "New"  
OR  
[WorkOrderStatus] = "In Progress"  
OR  
[WorkOrderStatus] is null
```

The addition of the last statement allows for records which have not been assigned any Work Order Status to be sent to the handheld.

Completion Checkbox Criteria

If you are using a Completion Checkbox field and your data is being stored in the Pendragon Forms Manager database, you can just check the form property to *Keep incomplete records on Handheld* in the Form Properties window. You do not need to enter any additional download criteria.

However, if you are linking to an external database (see page 486) then you may need to specify download criteria to send incomplete records to the handheld.

You need to know to which external database field type the Pendragon Forms Completion Checkbox field is mapping. If you are mapping a Pendragon Forms Completion Checkbox to a Text field in an external Access database or an ODBC database, you can actually just use the Data Persistence option of *Keep incomplete records on Handheld*, and you do not need to specify any additional download criteria.

If you are using an Access Yes/No (Boolean) field in your external database, then you will need to specify additional download criteria in order to remove completed records from the handheld. The values used by Access are 0 for No and -1 for Yes. SQL Server uses 0 for No and 1 for Yes. Since both databases use 0 for No, that is what is used to test a Completion Checkbox field.

```
[RemoveRecord] = 0 OR [RemoveRecord] is null
```

In this example, there is a Completion Checkbox field called Remove Record on the form. Records are only sent to the handheld if the value in the field is either null (blank) or 0 (zero). This means that incomplete records are sent back to the handheld.

```
([RepairTechnician] = ##USERNAME##) AND  
([Completed] is null OR [Completed] = 0)
```

In this example, the external database has a column called RepairTechnician that maps to the UserName field in Pendragon Forms. The form on the handheld and a column in the external database also has a field called Completed.

The AND operator means that both the first criteria and the second criteria have to be met for the record to be sent to the handheld. So the handheld user name has to match the value in the RepairTechnician field, and the Completed field has to be null or 0 (zero) for a record to be sent to the handheld. This means that a user will receive the incomplete records that have been assigned to him or her.

Criteria for Sorting Records as they download to the handheld

Pendragon Forms creates an internal query to download the appropriate records to the handheld. By default the query returns records sorted on primary key columns, but this may vary depending on the version of the database you are using to store records.

To guarantee that records are sent to the handheld in a specific order, you can use an **order by** clause in your additional download criteria.

Not every field type can be sorted, however. Text fields longer than 255 characters, Signature, Sketch, and Image fields cannot be sorted. Section, Jump to Section, Subform and Single Subform fields do not usually contain any data that can be sorted.

UserName = ##USERNAME## order by [Company]

The **order by** clause sorts records in ascending order (A..Z) of the specified column.

In this example, records assigned to a given user are sent to the handheld sorted in alphabetic order by a field on the form called Company.

UserName = ##USERNAME## order by [DateofVisit] desc

The word **desc** added to the end of an order by clause sorts the records in descending order.

Since Date fields are stored internally as a number (the number of seconds since 01/01/1904), sorting a date field in descending order will put the most recent dates at the top of the list of records.

In this example records assigned to a given user are sorted by a field on the form called Date of Visit, with the most recent dates at the top of the list.

The order by clause must follow a criteria. If you have your own primary key and users are sharing records, you cannot use **UserName = ##USERNAME##** as the criteria, because users would not be able to share records. Instead you can use a criteria that will always be true, such as:

1 = 1 order by [LastName]

In this example, the criteria **1 = 1** is always true, and the order by clause sends records to the handheld sorted in ascending order (A..Z) by a field on the form called Last Name.

If you have the full version of Microsoft Access, you can refer to the Access Help on WHERE clauses for additional information on the types of selection criteria that you can specify.

Disable Record Overwrite Protection

In Pendragon Forms, the default is that records which are newly created on the handheld cannot have a primary key which is already in the database on the PC. If a new record is created on the handheld, and the record has a primary key which is already in the database, the record will not be added to the database, and after a HotSync data transfer, the record will be marked with an arrow to indicate that it was not sent to the PC.

The reason why new primary keys on the handheld are not allowed to match primary keys of existing records is as follows: Imagine that Customer ID# is the primary key, and a record exists in the database on the PC for XYZ Corp., Customer ID# 345. If the handheld user creates a record for a new customer, Alpha Corp, and accidentally gives Alpha Corp Customer ID# 345, then the new record on the handheld for Alpha Corp. will overwrite the existing record for XYZ Corp. To prevent this accidental overwriting of existing records on the desktop, new records on the handheld have to have primary keys which are also new in the database.

NOTE: This restriction only applies to NEW records. Existing records from the database can be modified on the handheld and changes will update the existing records.

In very rare circumstances, you may want new records on the handheld to be allowed to overwrite existing records in the database.

Example 1:

You want to maintain a database for a library in order to keep track of which books have been checked in or out. Each book in the database has a unique bar code number. If a bar code is scanned on the handheld, the record will be new, but you want to be able to update the existing record to mark the book as in or out. In this type of circumstance you can check the checkbox to Disable Record Overwrite Protection.

Example 2:

During the synchronization process, if new records on the handheld are uploaded to the PC, but then something happens to interrupt the synchronization before the records on the handheld are marked for deletion, the records may remain marked as new records on the handheld. This causes a conflict on the next synchronization, because the records that already exist in the database on the PC will have the same primary keys as supposedly new records on the handheld.

In this scenario, first compare the records on the handheld with the records on the database, to verify that the same records are indeed in both places. Then switch on Disable Record Overwrite Protection and synchronize again. The records that were marked as new on the handheld will overwrite the existing records on the PC. If there are no more HotSync Error messages, switch off Disable Record Overwrite Protection again, especially if you are using your own primary keys on the form.

Merge Changes

Warning: This option should only be used if you are sharing records on more than one handheld, and you want to allow a record to be updated with a change to one field on one handheld, merged with a change to a different field on another handheld.

To share records across handheld devices, you need to select your own primary key field or fields that are unique for every record (See page 188). The synchronization rule is that if two people modify the same record, then the last person to synchronize will overwrite the record of the first person.

If the two handheld users never change the same field of the record, and you want to merge changes from both handhelds into the same record, you can switch on Merge Changes. For example, in a Customer database application, if one user modifies a field called Phone Number, and a different user modifies a field called E-Mail Address, the modified record on the PC will contain both the changed Phone Number and the changed E-Mail Address.

Note that even if Merge Changes is switched on, if two handheld users modify the same field of the same record, then the record on the PC will still be overwritten by the last person to synchronize, even if the first person to synchronize has more up-to-date changes.

Read-Only Reference Form (Data Hidden on Handheld)

If you need to store a large number of records (more than 4000) on a handheld device for lookup purposes only, you can choose to make a form a Read-Only Reference form. A second form has to be used in conjunction with the reference form, so that the handheld user can perform a lookup to the reference form, and then copy data from the reference form into the second form. Records in the reference form will not be visible except via the lookup mechanism. See page 127 for instructions on creating a Read-Only Reference form and another form to do the lookup.

Pendragon Forms version 5.1 now supports read-only reference forms on Pocket PC devices. In the prior version of Forms, version 5.0, read-only reference forms were not used on Pocket PC devices.

Before you freeze a form that you want to use as a reference form, you must set the Maximum Length (see page 194) of each Text field on the form to the smallest possible number of characters. This is because Pendragon Forms stores the read-only reference form on the handheld in a fixed-width format for each field, to facilitate a fast lookup over thousands of records. A Text field with a default value of 255 characters will reserve 255 characters for every record, even if only 10 characters are actually in the field.

On the reference form, check the Read-Only Reference Form checkbox on the Synchronization tab of the Advanced Form Properties window. Checking this checkbox also causes the Advanced Form Property *Hide Form in Forms List* to be checked. This is because records in a Read-Only Reference form are not visible by reviewing a form, and cannot be modified on the handheld, and so in most cases it is best for the handheld user not to see the name of the form on the handheld.

The records in the reference form are all stored as Text or Numeric on the handheld. This means that in order to copy data from the reference form into a second form, the fields on the second form must be Text or Numeric fields.

A Read-Only Reference form will synchronize all the records every time the user performs a HotSync data transfer. This will increase the synchronization time. However, if the data in the form does not change often, you can switch on the Advanced Form Property of *Synchronize Only When Distributed* - see next page.

Synchronize Only When Distributed

If you have a form that is used for reference purposes only and contains either a lot of records (such as a Read-Only Reference form) or a lot of images, and the records or images originate on the PC and do not change frequently, you can switch on the Advanced Form Property of Synchronize Only When Distributed. This will speed up synchronization time, because the form will not be synchronized unless you click on the name of the form in the Forms Manager and then click the Distribute button. If data in the form only changes once a week, for example, then you can re-distribute the form when the data is updated, so that the form synchronizes once a week.

Warning: Do not switch on this option if users are allowed to create or modify records, as the records will not be backed up during everyday synchronization.

Storage Limit

The storage limit is the absolute size limit allowed for data on the handheld for each form. In cases where you want to set the form property to *Keep a copy of records on Handheld* (see page 198), if you have thousands of records on the PC, the storage limit prevents the entire handheld memory from being filled with records.

- The default storage limit is 500KB per form. You can change the storage limit for a given form by making this number larger, for example 700KB or 1000KB.
- If there are more records on the PC than can fit on the handheld, records from the PC will be placed on the handheld in ascending order of primary keys. By default Pendragon Forms uses the creation TimeStamp of a record as part of the primary key, which will mean that records will be placed on the handheld starting from the earliest created record, up to a maximum 500KB size. If you are storing months of data on the PC and do not want to put the earliest records onto the handheld, you can use the *Archive Date* (see next topic) to determine the starting date for placing records on the handheld. You can also use an Additional Download Criteria (see page 214) to select which records are sent to the handheld and in what order the records are sorted.

Archive Date

The Archive Date is a cut-off date for placing records on the handheld. Any records created before the archive date will not be placed on the handheld.

- To enter an Archive Date, type in a date in the same format as the Windows short date format on your PC. (The Windows short date format is set in the Windows Control Panel. Click on Start...Settings...Control Panel, and double-click the Regional Settings (or Regional Options) icon in the Control Panel.)
- The Archive Date feature relies on the creation TimeStamp which Pendragon Forms automatically generates. If you are linking to an external database and are not storing the TimeStamp field, then the Archive Date feature will not work. You can use Additional Download Criteria (see page 214) to determine which records are put on the handheld.

Beaming Tab

PALM OS

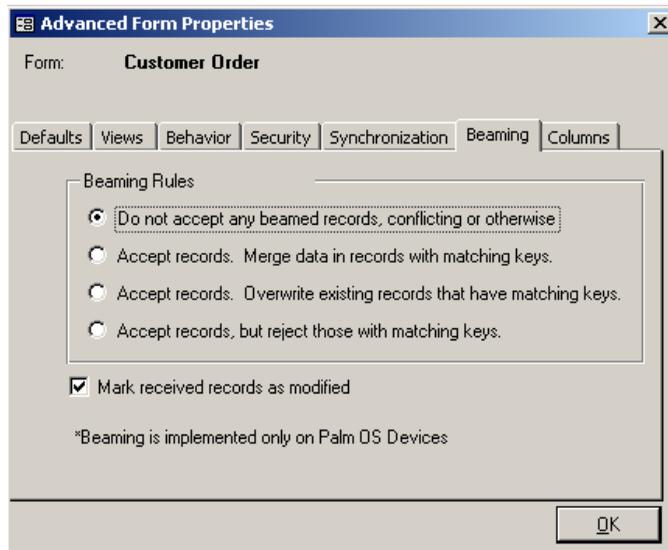
Beaming is only available on Palm OS devices.

The Beaming tab in the Advanced Form Properties window allows you to set the rules for allowing infrared beaming of Pendragon Forms records from one handheld to another handheld.

For beaming to take place, the following conditions are necessary:

- 1) Both handhelds must already have both the Pendragon Forms application and the identical form design (same Form ID# - see pages 556-557) present on the handheld. You cannot beam the Forms application or a form design. Only records can be beamed.
- 2) Both handhelds must be configured to allow beaming.
- 3) Typically, you will need to select a primary key field or fields when you design the form.
- 4) The form design must have a Button field with a script that selects records to be beamed. The script also initiates the beaming.
- 5) An appropriate Advanced Form Property Beaming Rule must be selected. See page 224.
- 6) The Advanced Form Property to Mark Received Records as Modified must be set appropriately. See page 225.
- 7) The Advanced Form Property to Disable Record Overwrite Protection must be switched on. See page 219.

These conditions are discussed in Chapter 19, *Beaming Records*, page 445. Items 5) and 6) are discussed here.



Beaming Rules

The Beaming Rules are:

- **Do not accept any beamed records, conflicting or otherwise**
This is the default option. This option means that handheld users are not allowed to beam Pendragon Forms records to each other.
- **Accept records. Merge data in records with matching keys**
This option means that beaming of records is allowed.
Merge data in records with matching keys means that if the same record already exists on both handhelds, and both users have modified different fields of that record, the handheld that is receiving the beamed record will have the most up-to-date version of the record, containing both users' changes.

If both users have modified the same field within the same record, then the handheld that is doing the beaming will overwrite the receiving handheld in that field.

This option must be selected with care. If both handhelds have modified the same record, the handheld that is doing the beaming will not have the other handheld's changes.

- **Accept Records. Overwrite existing records that have matching keys**
This option means that beaming of records is allowed.
Overwrite existing records that have matching keys means that if the same record already exists on both handhelds, the handheld that is doing the beaming will overwrite that record on the receiving handheld, even if the receiving handheld had modified the record or has more up-to-date information.

If the handheld users can work out who has the more up-to-date information and have that person initiate the beaming, then this option is simpler than the merge changes option, because at the end of the beaming, both handhelds have the same data in the same record.

- **Accept Records, but reject those with matching keys**
This option means that beaming of records is allowed.
Reject records with matching keys means that if the same record exists on both handhelds, the beamed record will be discarded by the receiving handheld.

This option is useful if you only want handheld users to beam new records to each other. That way there is less confusion over what to do with records that are already on both handhelds.

Mark Received Records as Modified

PALM OS

This option only applies to Palm OS devices.

The Advanced Form Property of Mark Received Records as Modified is located on the Beaming tab, and is switched on by default.

If a record that is new or changed on a handheld, is beamed to another handheld, then the record will automatically be marked as changed on the second handheld as well.

When the Mark Received Records as Modified option is switched on, then even existing unchanged records that are beamed from the first handheld will be marked as changed on the receiving handheld. This will force the records to be uploaded to the PC when the second handheld synchronizes.

If handheld users are using beaming as a way to ensure that a record is on more than one handheld, so that if one handheld fails the other handheld has the record, then it is very important to keep the Mark Received Records as Modified checkbox checked. That way the receiving handheld is guaranteed to send the records to the PC on the next synchronization. (Note: the script that beams the records should select and queue all the records in a form in order to guarantee that all records are beamed.)

Changing Advanced Form Properties on a Frozen Form

Once a form design has been frozen and distributed to the handheld, if you change any Advanced Form Properties on the Defaults tab, the Views tab, the Behavior tab, the Security tab or the Beaming tab, you will need to a) re-distribute the form design, and b) synchronize the handheld, for the changes to take effect. Data on the handheld is not affected by re-distributing the form.

Changes made on the Synchronization tab of the Advanced Form Properties screen will take place the next time the handheld synchronizes. The form design does not have to be re-distributed in this case.

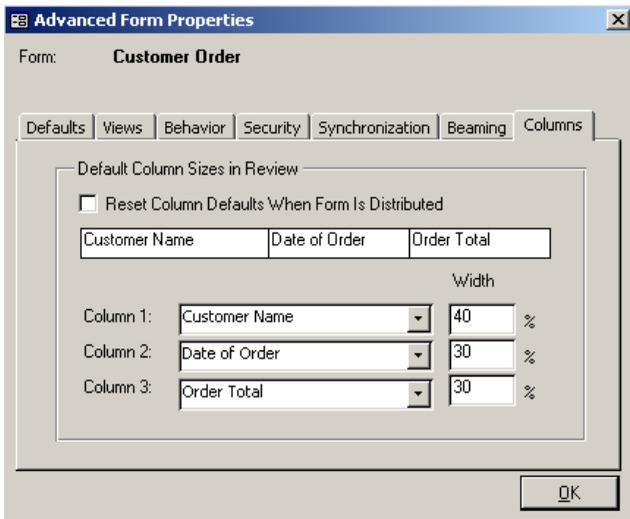
Columns Tab

The Columns tab in the Advanced Form Properties window allows you to select up to three columns that will be displayed by default when a user reviews records on the handheld.

For each column, select the field that will be displayed in that column, and specify the percentage of the width of the screen that you want the column to occupy.

The total percentage for all three columns cannot exceed 100%.

NOTE: The default settings are only used if the handheld user does not change any columns on the Review screen. If the user selects different columns to view, or changes the width of columns, then the user's preferences will override the default settings on the Columns Tab. If you need to override the user settings and revert to the columns and widths specified on the Columns tab, check the Reset Column Defaults When Form is Distributed checkbox and re-distribute the form.



Customer Order		
Customer Nam	Date of	Order Total
Grover's Ice Cr	4/26/03	\$15.49
Marina's	5/18/03	\$23.80
Zebra Mix	5/18/03	\$67.55
Alpha Garden	5/11/03	\$71.15
Ocean Fresh	5/18/03	\$35.40
Megan Vegan	11/28/06	\$69.70

Below the table are controls: 'All Fields' with a dropdown arrow, a search icon, and a close icon. At the bottom are 'Sort...' and 'Done' buttons.

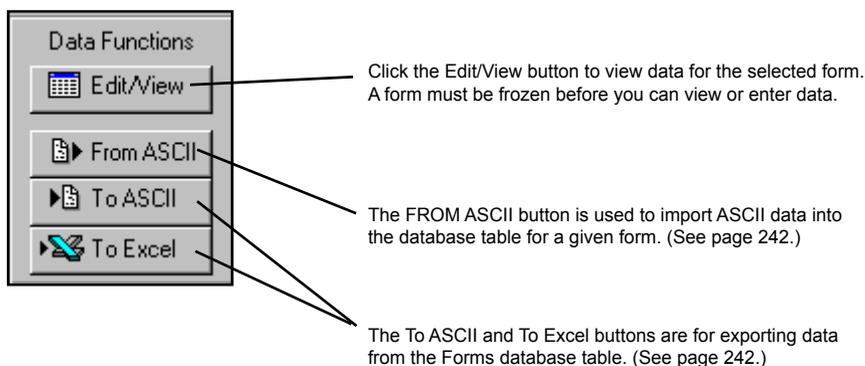
11. Managing Data on the PC

Pendragon Forms supports bi-directional synchronization. When you synchronize the handheld, new or changed records on the handheld are uploaded to the PC, and then records (including new records on the PC) are written back to the handheld.

- The records on the handheld overwrite the records on the PC. This means that if the same record is simultaneously modified on both the handheld and the PC, the changes from the handheld will take precedence. (If necessary, it is possible to protect individual fields from being overwritten by the handheld by setting the Advanced Field Property of Do Not Upload to PC - see page 196. You can also set the Advanced Form Property of Merge Changes - see page 220.)
- During the synchronization process, data is uploaded directly into the database. The default Pendragon Forms database is Form32k.mdb, which is stored in the C:\Program Files\Forms3 folder; alternatively you can link to your own external database (see page 486).

Viewing and Editing Data in the Database

1. In the Pendragon Forms Manager on the PC, click on a form to select that form.
2. Click the Edit/View button to display the data associated with that form. An Access form is displayed in datasheet view, to allow you to view and modify your data.



3. The fields in the form appear as columns, and each record is a separate row.
 - To edit a record, click in a cell and type your corrections. To save changes in a row, position the cursor outside of that row.

156482 : Form

UnitID	UserName	TimeStamp	ItemDescription	DatePurchase	TypeOfItem	MakeAndModel
0	Debra Sanch	399 3:57:30 PM	Washing machi	04/07/1995	Large appliar	Maytag
0	Debra Sanch	399 4:00:13 PM	Dryer	04/07/1995	Large appliar	Maytag
0	Debra Sanch	399 4:01:27 PM	TV	10/16/1998	TV / Electron	General
0	No one	399 4:17:14 PM			TV / Electronic	

The first four fields that you see: Record ID, UnitID, UserName and TimeStamp, are used to uniquely identify each record coming back from the handheld.

If you export data to Excel or to ASCII, the first four columns of data will also be exported. You can delete these columns from your Excel file, but you should NEVER delete these columns from the Pendragon Forms Manager database.

In Popup fields and Lookup List fields, when you click in the cell you can select an item from the list. Note that this feature does not work for cascading lookup lists. Lookup Lists will show the LookupDisplay column.

4. To add a record, click in the blank row (the last row in the list of records.)
 - In the UserName field, select the handheld user who is to receive the record.
 - Then enter information in the various cells, pressing TAB to move from one cell to the next. Position the cursor outside of the row to save the record.
 - When you close the Edit/View window and perform a HotSync data transfer, the new record will be sent to the specified handheld.

Note: Not all of the features of Pendragon Forms that are available on the handheld are available when editing a record on the PC. For example, cascading lookups do not work when editing on the PC. Similarly, scripts, branching, hidden fields, button fields, jumping to subforms, and checks for required fields or numeric ranges are all features that are present on the handheld but are not present when editing records on the PC.

Alternative Methods for Viewing Data

Instead of clicking the Edit/View button to view the data as a datasheet, in which each field is a column and each row is a record, you can use these alternatives:

- Press the CTRL + Edit/View button.
This displays an Access form in Form View, which shows one record at a time. Signatures are visible in this format.

Click the right and left arrow buttons to move from one record to the next.

The arrow button with a vertical line moves to the last record.

The arrow button with a star (*) is used to create a New record.

- Pressing SHIFT + Edit/View displays a the data in the form of a query.
- Pressing ALT + Edit/View displays the actual Access database table containing the data. This method is useful if you need to quickly find out the name of the Access database table where the data for your form is being stored.

Deleting a Record

1. Click in the gray cell to the left of the row that you want to delete. This will highlight the entire row.
2. Press the Delete key on the keyboard. You will be asked to confirm the deletion. Click Yes to delete or No to cancel.

Note: Deleting a record from the PC will remove the record from the handheld only if the record has not been modified on the handheld. If the record has been updated on the handheld, it will be uploaded to the PC on the next HotSync data transfer, and can then be deleted from the PC.

WARNING: Deletion of records from the PC is permanent. Make sure that you have a backup of the Forms Manager database before you delete large numbers of records. (See page 510.)

Exporting Data to Microsoft Excel

To create a report, you can export the data to Microsoft Excel.

1. In the Pendragon Forms Manager, click on a form to select that form.
2. Click on the **To Excel** button to export the data to an Excel spreadsheet file. You will be prompted for a file name for the Excel Spreadsheet.
3. After exporting the data to Excel, close the Pendragon Forms Manager. You can then open the Excel file and use Excel to create your report.

Troubleshooting tip:

If you attempt to export to Excel and get an error message about Installable ISAMs, close the Pendragon Forms Manager. Then click on Start...Programs...Pendragon Forms...Configuration Tool. In the Configuration Tool window, click on the Diagnostics tab and click the button to Register Installable ISAMs. Then re-try exporting data to Excel.

Exporting Data to ASCII (CSV File)

If you need to send the data to another database, you can export the data to ASCII, and then import the ASCII data into your own database.

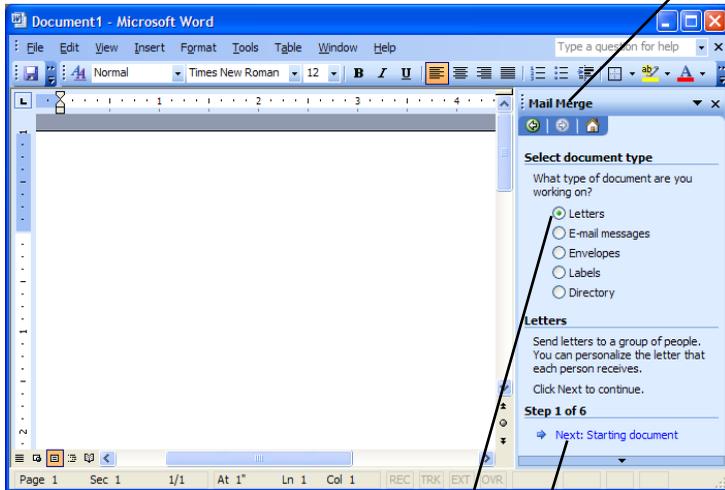
1. In the Pendragon Forms Manager, click on a form to select that form.
2. Click on the To ASCII button to export the data to a comma delimited ASCII file (a .TXT file or a .CSV file - Comma Separated Variable file).
You will be prompted for a file name for the ASCII file.

Creating a Report in Microsoft Word

Once you receive data from the handheld back on the PC, you can use Microsoft Word to create a report to print out your data.

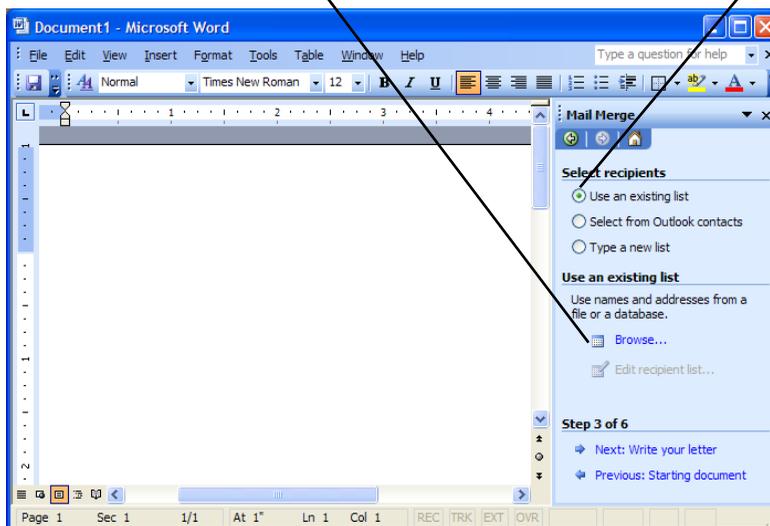
The following instructions can be used with Microsoft Word 2003.

1. In the Pendragon Forms Manager on the PC, click on a form and then click the To ASCII button.
2. A **Save to ASCII** window will appear. Select a directory folder and type a name for the Text file to which the data will be exported. Click Save. If prompted that a CSV file has been created, click OK.
3. Close the Pendragon Forms Manager.
4. Open Microsoft Word 2003, and choose to open a New, blank document.
5. In Microsoft Word 2003, click the **Tools** menu, then click **Letters and Mailings**, then click **Mail Merge**. A Mail Merge Wizard screen appears on the right-hand side of the screen.

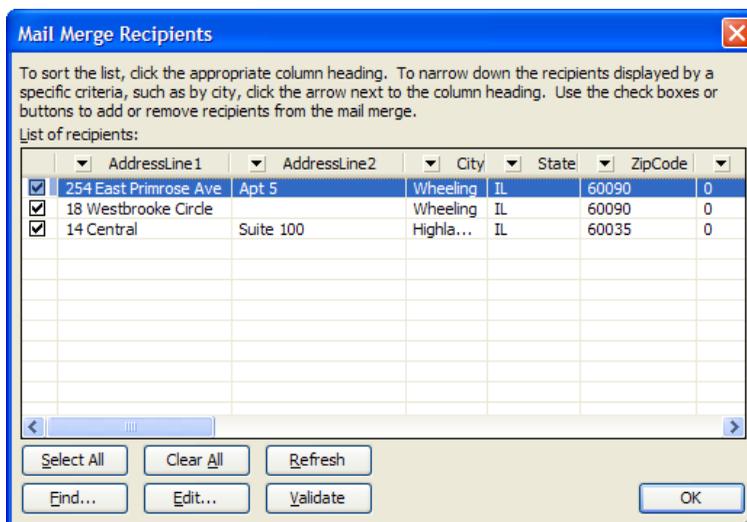


6. In the Mail Merge Wizard screen:
In the Select document type section, choose **Letters**.
At the bottom of the screen, click **Next: Starting document**.

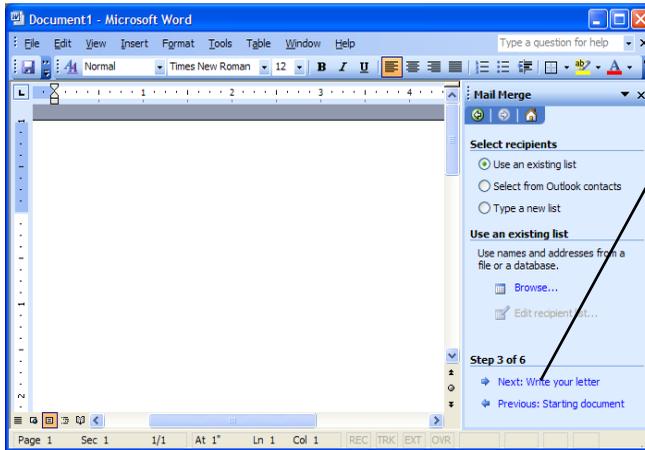
7. In the Mail Merge Wizard screen, in the Select recipients section, choose **Use an existing list**, and then click the link to **Browse**.



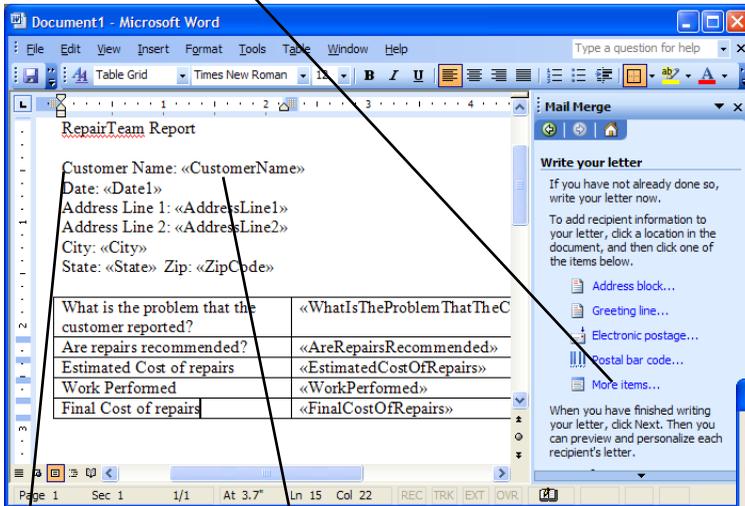
8. A Select Data Source window appears. Select the directory folder where you stored the Text file (in step 2) and select to view files of type: All Data Sources. Click on your selected file and then click OK.
9. A Mail Merge Recipients window appears. To choose all the records, click the Select All button, then click OK.



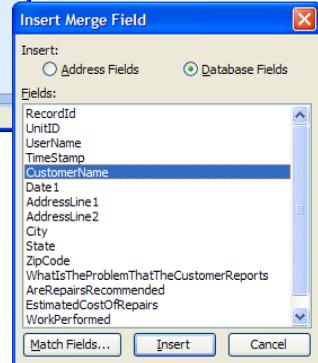
10. At the bottom of the Mail Merge Wizard screen, click **Next: Write Your Letter**.



11. Type the text of your report. To position a Pendragon Forms field on your report, click **More Items** in the Write your letter section of the Mail Merge Wizard.



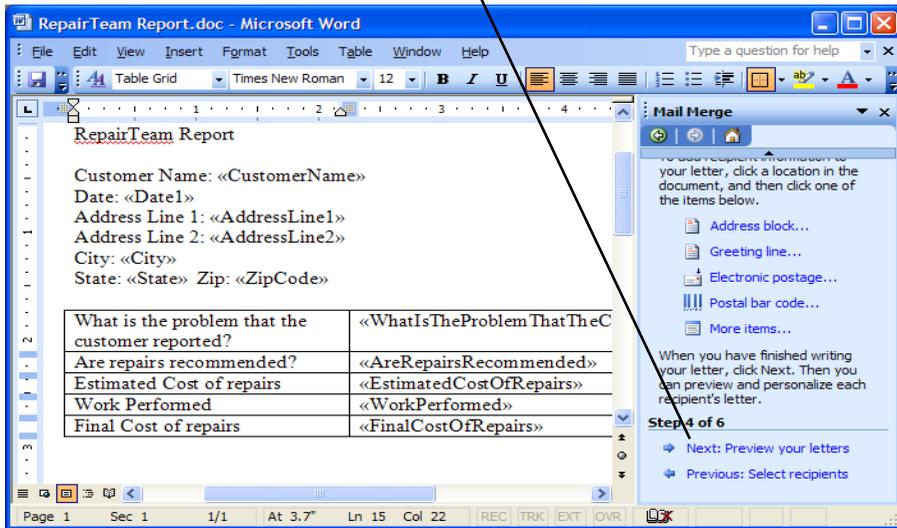
When you click More Items, an Insert Merge Field window appears. Select the Pendragon Forms field to insert, then click the Insert button, then click the Close button. The field will appear in special angled brackets (Example: <<CustomerName>>).



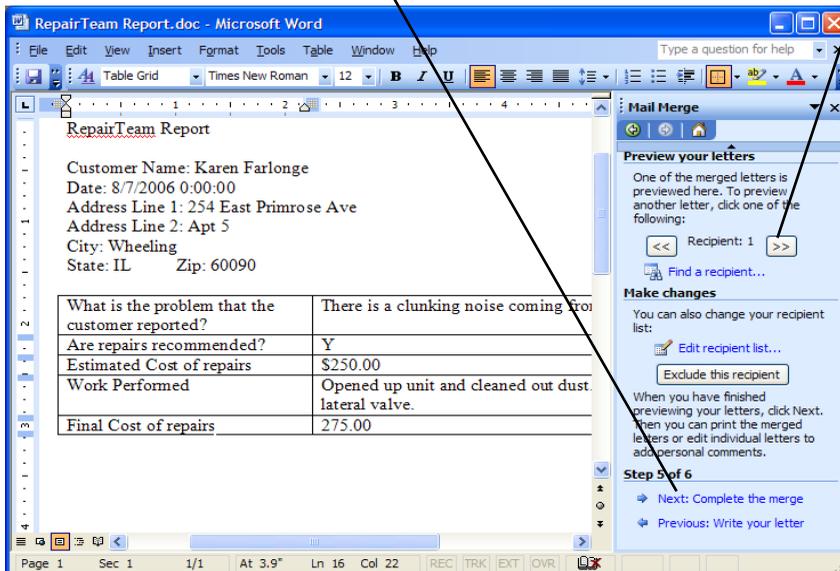
This text was typed in Word to create the report.

This is an inserted Merge field.

12. Save the Word document. This document will serve as your report template, and can be used over and over.
13. In the Mail Merge Wizard screen, click **Next: Preview your letters**.



14. In the Preview section of the Mail Merge Wizard, you can choose to view each report (letter) in turn, or you can click **Next: Complete the merge** to print the reports.



Creating Queries and Reports in Microsoft Access

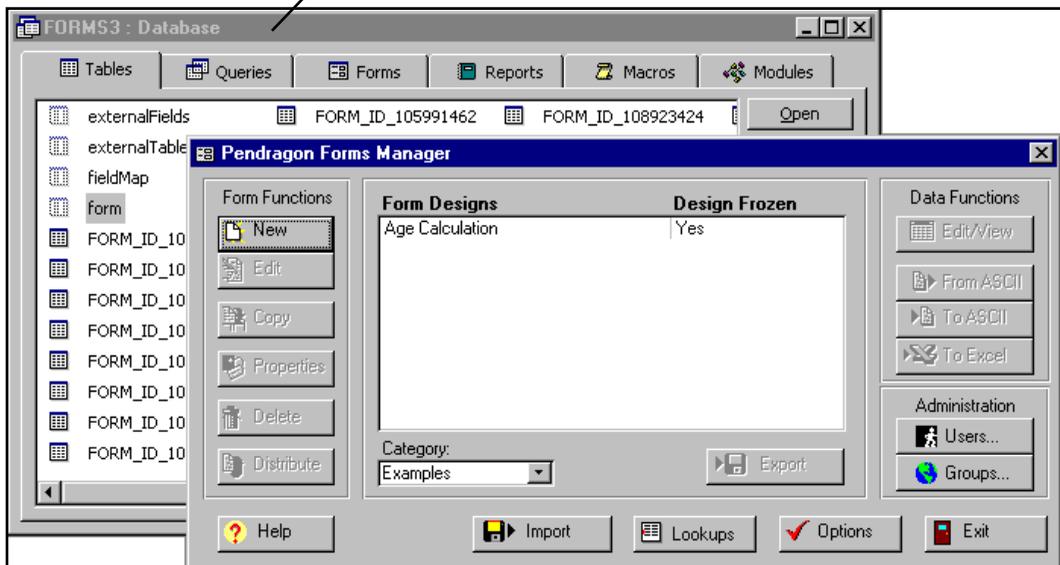
If you have the full version of Microsoft Access 2000, Access 2002 or Access 2003, you can tap into the full features of Access to create queries and reports on your data.

Refer to your Microsoft Access online Help or printed documentation for instructions on creating reports and queries.

The Pendragon Forms database is typically located in the C:\Program Files\Forms3 folder. The file name is Forms32k.mdb.

Within the Pendragon Forms Manager, you can bring the **FORMS32K: Database** window to the foreground in order to create queries and reports in Access. The **FORMS32K: Database** window may be minimized, or you may have to click on Windows...Cascade to view this window.

Bring the **FORMS32K: Database** window to the foreground to use the full features of Microsoft Access.



To see which database table corresponds to a form, in the Forms Manager click on the form name and then click on the Properties button. The Form ID is the same as the database table name for that form.

12. Importing & Exporting

Import Button

In the Pendragon Forms Manager, you can click the Import button to access the following screen:

This button allows you to import form designs, data and Lookup Lists from a previous version of Pendragon Forms.

The Import Data button allows you to import data from one form into another. Useful if you have had to copy a form to make changes, but you want to keep the data from the previous form design.

Import form designs, data and lookups from a previous version of Pendragon Forms

Import a form design from a .PFF or .CSV file.

Import data from a different form design

Build Forms Based On External Tables

Cancel

The Import button allows you to import form designs which are in .PFF file format.

You can also create a form design by importing an ASCII Comma Separated Variable file.

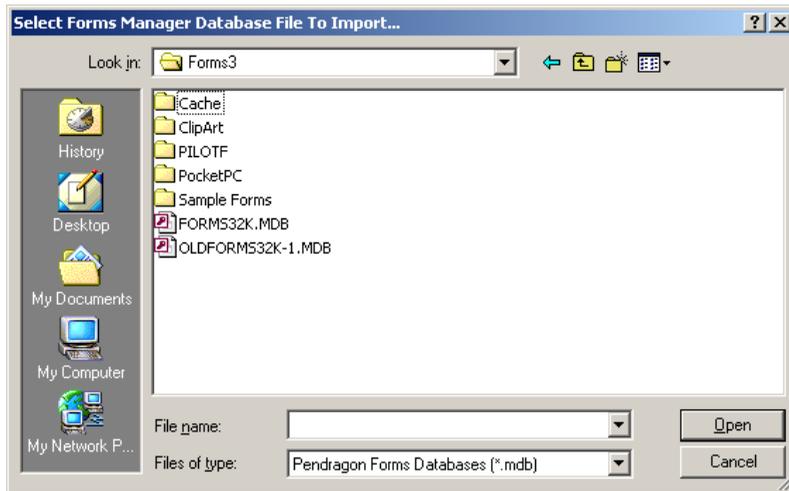
These buttons allow you to create form designs from external database tables. See Linking to external databases, page 486 and 506.

Importing from a previous Pendragon Forms database

If you have been using an earlier version of Pendragon Forms, you can import your form designs, data and lookup lists into the current Forms database.

To import from a previous version of Pendragon Forms:

1. In the Pendragon Forms Manager, click the Import button, then click the Import Forms/Data/Lookups button.
2. An Import window will prompt you to select the Forms database to open.



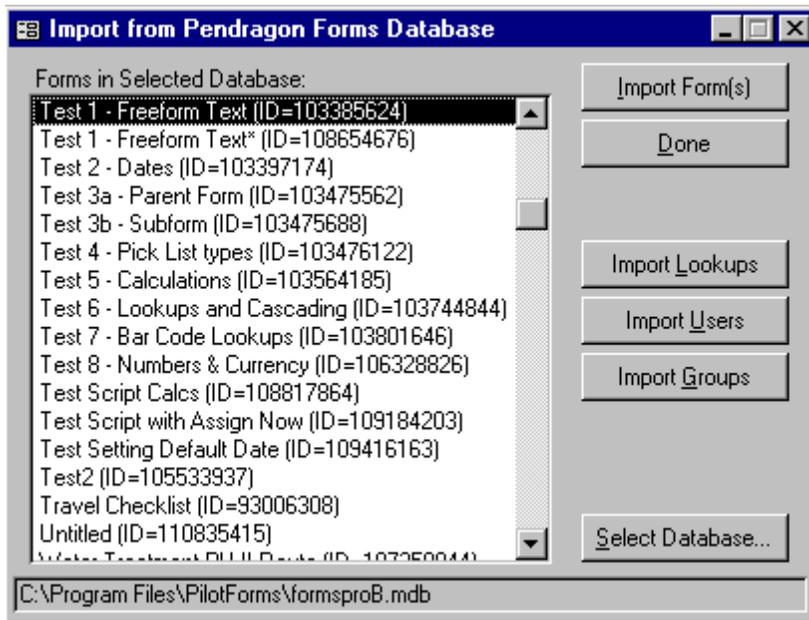
Valid database names to select are shown here:

Note: If you re-install Forms, the database gets renamed, e.g FORMS32K.MDB is re-named to OLDFORMS32K-1.MDB on the first re-install, OLDFORMS32K-2.MDB on the second re-install, etc.

Version of Pendragon Forms	Name of Database
Pendragon Forms version 5.0	FORMS32K.MDB (Access 2000/2002/XP/2003)
Pendragon Forms version 4.0	Forms32k.mdb (Access 2000
Pendragon Forms version 3.2	Access 2002/AccessXP
Pendragon Forms version 3.1	Access 2003)
Pendragon Forms version 3.0	Forms3.mdb (Access 97)
Pendragon Forms version 2.0	FormsProB.mdb
	FormsProA.mdb
	FormsPro.mdb

Select a database and click Open.

3. A list of forms in the selected database will be displayed.
 - To import a form design and the data in that form, click on the name of the form and then click the Import Form button. Repeat for each form that you want to import; or you can use CTRL + click on each form design that you want to import.
 - To import all of your Lookup Lists, click the Import Lookups button. (Do this only once.)
 - For Forms version 3 and higher only: Use the Import Users and Import Groups buttons to import the User List and User Groups into the current database.



After the import process is complete, click on the name of each form in the Pendragon Forms Manager and click the Edit/View button to verify that your data has been imported correctly.

Importing Data from a Different Form Design

When you copy a form in order to make changes to the form design, the data associated with the original form is not automatically copied into the new form. If you need the data from the original form, you can import the data as follows:

1. After you have made the necessary modifications to the copied form, freeze the copied form.
2. In the Pendragon Forms Manager, click on the name of the copied form, then click the Import button, and then the Import Data button.
3. An Import Existing Form Data window appears. Select the form whose data you want to import.

Import Existing Form Data

Importing Into: **Test 13 - General test2**

Import From Form: **Test 13 - General test**

Field Mappings

ID: 16	Assign initialize	Column: AssignInitialize	Source: AssignInitialize	<input type="checkbox"/> Primary
ID: 3	Common colour of flower	Column: CommonColourOfFlower	Source: CommonColourOfFlower	<input type="checkbox"/> Primary
ID: 1	Country	Column: Country	Source: Country	<input type="checkbox"/> Primary
ID: 8	Do you make homemade cakes?	Column: DoYouMakeHomemadeCakes	Source: DoYouMakeHomemadeCakes	<input type="checkbox"/> Primary

Record: 1 of 19

Import **Cancel**

4. Once the original form has been selected, a list of fields will be displayed.
 - If the column names of the copied form were kept the same as the original column names, then click the Import button to map data from the original form to the copy.
 - If database column names were changed in the copy, then select the old column name for each field, by clicking on the arrow in the Source field for each field on the form. Click the Import button when you have finished mapping the old column names to the new column names. NOTE: If you accidentally map a Text field to a Yes/No field or Popup List, data for these fields will not be sent to the handheld.

Importing & Exporting Form Designs

If you want to design a form on one PC, and then use the form on another PC, you will need to export the form design.

- The Export button in the Pendragon Forms Manager window allows you to export your form design to a file. Form designs are stored in a .PFF file format when exported.
- The Import button in the Pendragon Forms Manager window allows you to import a form design from a .PFF file format.

Limits of Import and Export functions

The Import and Export buttons only convert form designs into .PFF files. Lookup Lists are not automatically converted. If your form uses Lookup Lists, you will need to import/export the Lookup Lists in addition to the import/export of the form design.

To Import a Form Design

1. Click the Import button in the Pendragon Forms Manager window, then click the Import...button. A Select Pendragon Forms Design File window appears. Pendragon Forms Design files (*.PFF) will be displayed by default.
2. Select the directory where the form to be imported is stored.
3. Select the name of the file to be imported.
4. Click the Open button. The selected form will be imported into your Form Designs List in the Pendragon Forms Manager.
5. If the form that you are importing contains Lookup Lists, you will need to import the Lookup Lists separately. See *Importing & Exporting Lookup Lists*, page 241.

To Export a Form Design

1. In the Pendragon Forms Manager, click on the form to be exported.
2. Click the Export button. An Export Data window appears.
3. Select the directory where the form to be exported will be stored.
4. Type a file name for the form to be exported. The file will be saved as a Pendragon Forms Design file (.PFF).
5. Click the Save button.
6. If the form that you are exporting contains Lookup Lists, you will need to export the Lookup Lists separately. See *Importing & Exporting Lookup Lists*, page 241.

Importing & Exporting Lookup Lists

You may want to import a Lookup List if you already have ASCII data that you want to use in a Lookup List. You may also need to import a Lookup List if you are importing a form design, and the form contains Lookup Lists. The Lookup Lists have to be imported separately from the form design. You may need to export a Lookup List if you design a form and want to export your form design. Any Lookup Lists used by the form will have to be exported separately.

To Import Data into a Lookup List

Data must be in a CSV file format (see page 243) to be imported into a Lookup List. The CSV file can have one or two columns: one column of data for the Lookup Display field, and a second, optional column of data for the Lookup Value field.

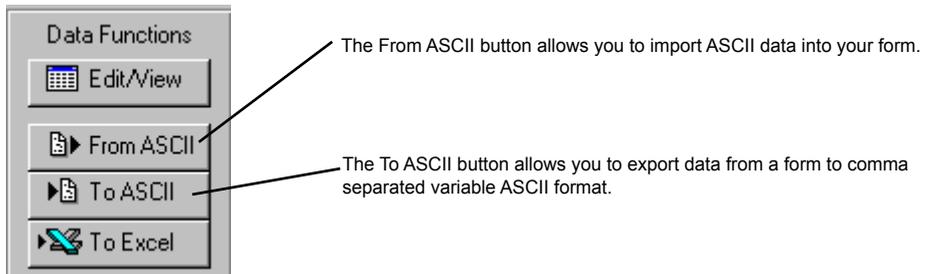
1. In the Pendragon Forms Manager window, click the Lookups button. The Lookup Lists window will appear.
2. In the Lookup Lists window, click in the blank row below all the existing Lookup Lists, and type a name for a new Lookup List. If you are importing a Lookup List for use with a form which is also being imported, the name that you type here for the Lookup List must exactly match the name of the Lookup List in the Lookup List field on the form.
3. Click the Edit button to the right of the Lookup List name.
4. In the Lookup Editor window, click the Import Lookup button.
5. Select the directory where the Lookup List CSV (ASCII) file is stored.
6. Select the Lookup List CSV file to be imported.
7. Click Open.

To Export a Lookup List to an ASCII (CSV) file

1. In the Pendragon Forms Manager window, click the Lookups button. The Lookup Lists window will appear.
2. In the Lookup Lists window, click the Edit button next to the Lookup List that you want to export.
3. Click the Export Lookup button.
4. Select the directory where the Lookup List to be exported will be stored.
5. Type a file name for the exported Lookup List. The file will be saved as a CSV (ASCII) file.
6. Click the Save button.

Importing & Exporting Data

In the Pendragon Forms Manager, the Data Functions buttons allow you to import and export data from your forms.



Importing Data

You can import data which is in comma delimited (or comma separated variable .CSV) ASCII format into a form. (See next page for information on CSV files.)

- In order to import data correctly from a CSV file into the database table for a form, the column names in the CSV file must exactly match the column names in the Forms database table.
- After you freeze your form, click on the name of the form, then hold the Shift key down and click the Edit/View button. You will be able to see the column names. You may need to widen the columns to view the entire column names. Make a note of these column names, and use the identical names in your CSV file.

To import ASCII data into a form:

1. Click on the name of a form, then click the From ASCII button.
2. You will be prompted to enter a handheld user name to whom the records in the ASCII file will be assigned. If you do not want to assign the records at this time, leave the default of No One. Click OK.
3. You will be prompted to select the ASCII .CSV file to import. Select the appropriate .CSV file and then click OK.
4. Click the name of the form and then click the Edit/View button to verify that your data has imported correctly.

Exporting Data

Once data comes back to the PC from the handheld, you can choose to export the data. You can *Export Data to Excel* or *Export Data to ASCII*, page 230.

Creating an ASCII (CSV) File

If you have data in Excel or in a database, you can use this data to create a form, or to import into the fields on the form. Data has to be in an ASCII format called comma delimited ASCII or comma separated variable ASCII (.CSV file extension) in order to work with Pendragon Forms.

Format of a CSV File

A CSV File is a text file in which the first row identifies the field names, and all subsequent rows are individual records. Data within each record is separated by a comma (hence the name comma separated variable or CSV). Text is surrounded by double quotes.

Sample CSV File:

```
Name,Address,IDNumber,FavoriteColor
"John Smith","22 Cherry Lane",45,"Blue"
"Peter Panera","418 N. Filmont Ave.",78,"Green"
"Sarah Jane Smyth","71 Barley Lane",5180,"Purple"
```

IMPORTANT:

The column names in the first row of the CSV file must exactly match the column names in the database table of the form. To see the column names used by the form:

1. In the Pendragon Forms Manager, create the form and freeze the form.
2. Click on the name of the form to select it, then hold down the SHIFT key and press the Edit/View button. This will display the columns in the database table for the form.
3. You may need to widen the columns to view the entire column names. Make a note of the exact column names, and use the same column names in your CSV file.

To create a CSV File from Microsoft Excel

Refer to the picture on the next page.

1. In your data file, the first row should contain the name of each column of data. These names must correspond to the column names in your form.
2. Each row should contain a separate data record.
3. Keep your file in its original format (e.g. an Excel worksheet). But also make a copy and save the copy as file type .CSV - comma delimited ASCII.

Column names must match the column names in the Forms database table.

Each row of data is a record in your form.

	A	B	C	D
1	CustomerLastName	CustomerFirstName	Company	Dateoflastc
2	Anderson	Joseph	ABC Technologies	09/08
3	Bacchus	David	XYZ Communications Inc.	09/01
4	Jurvis	Sally	Jurvis, Parker & Pilgrim	09/29
5	Smith	Martin	Zippydyne	10/02
6	Smyth	Sarah	Alpha Beta Epsilon	10/10
7	Potter	George	Chopper Rescue Corp.	10/15
8	Rampersad	Douglas	Checkmate Players	09/30
9				

Creating a Form from a CSV File

Once you have data in a comma delimited ASCII format (or comma separated variable .CSV file), you can use the CSV file as the basis for creating a form. Once the form is created, you can import the data from the file into the form, and then send both form and data to the handheld.

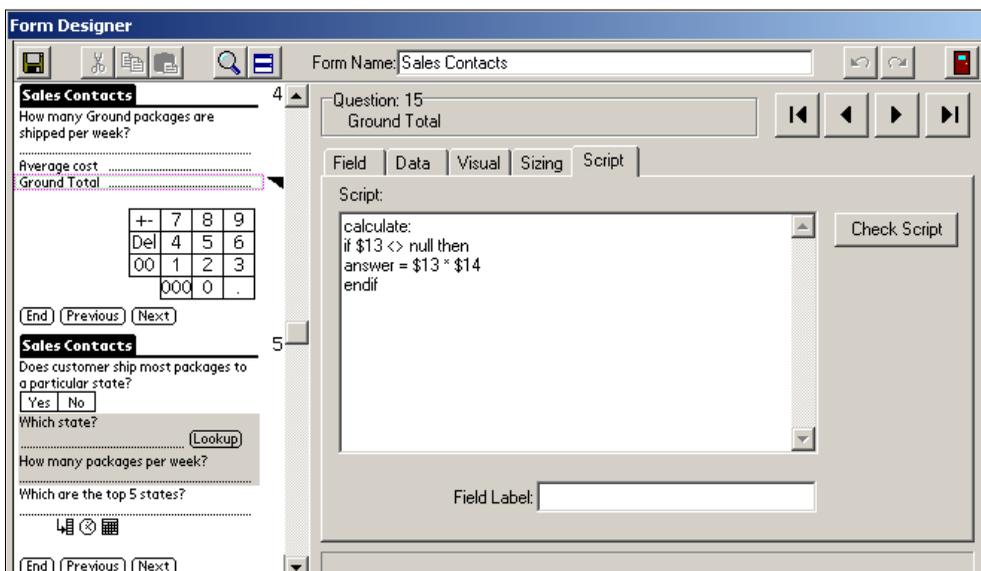
1. In the Pendragon Forms Manager window, click the Import button, then click the Import...button.
2. A Select Pendragon Forms Design File window will appear. Instead of displaying files of type .PFF, select to display files of type .CSV. Select the directory where the CSV file is stored. Double-click on the CSV file to select it.
3. An Import Data dialog box will appear, prompting you for a name for the form whose design will be based on the CSV file. Type a name for the form.
4. The new form will be displayed in the Forms List in the Pendragon Forms Manager. Click on the form and then click the Edit button.
5. Review the form. You may want to make minor changes, such as:
 - Change text fields to Yes/No fields, Popup Lists or Lookup Lists as appropriate.
Or change Numeric fields to Currency fields, as appropriate.
6. When you are satisfied with the design of the new form, you can freeze the form design.
7. To import the data in the CSV file into the form, see page 242.

13. Scripting Reference

The Scripting tab in the Form Designer window allows you to write scripts to control what happens when the handheld user makes a selection in a field, or enters or exits a field or a screen.

Scripting allows you to:

- Perform calculations in fields.
- Create branching, so that the user's response in one field determines which field(s) are displayed next.
- Minimize handheld data-entry by pre-filling certain fields.



Format of a Script

A script consists of at least one event procedure. An event procedure contains:

- An event procedure label + statement(s)

The event procedure label determines when the script will be run.

The statements are the actions which are performed when the script is run.

The pages which follow describe the possible event procedures and statements which you can put in a script.

To create a script for a field:

1. In the Form Designer window, click on a field in the Preview Area to select the field.
2. Click the Script tab.
3. Type the script for the field. A script consists of an event procedure and one or more statements. (See following pages.)
4. Click the Check Script button to check the syntax of the script.
5. You can change a script even after you freeze a form design.
This means that if you need to modify a script, you can do so, and then just re-distribute the form to the handheld. You do not need to copy the form design in order to modify a script.

How Scripting Works & Limits of Scripts

Scripts compile when you distribute a form. If an error occurs during compilation, you will receive an error message when you distribute a form. See Appendix B, *Troubleshooting*, page 534.

Limits of scripts include:

- The text in each script cannot exceed 2000 characters.
- Some scripting events only run in Layout View, others may only run in Field View. Other scripting events run in all views: Layout View, Field View and Record View. On Palm OS handhelds, you must decide which view the handheld user will be using, and only use scripting events that work in that view. Pocket PC handhelds only support Layout View.
- If you have a lot of scripts, or very long scripts, you may notice a delay on the handheld as scripts run. The solution is to limit the use of scripts and to minimize the length of scripts.

Field Labels

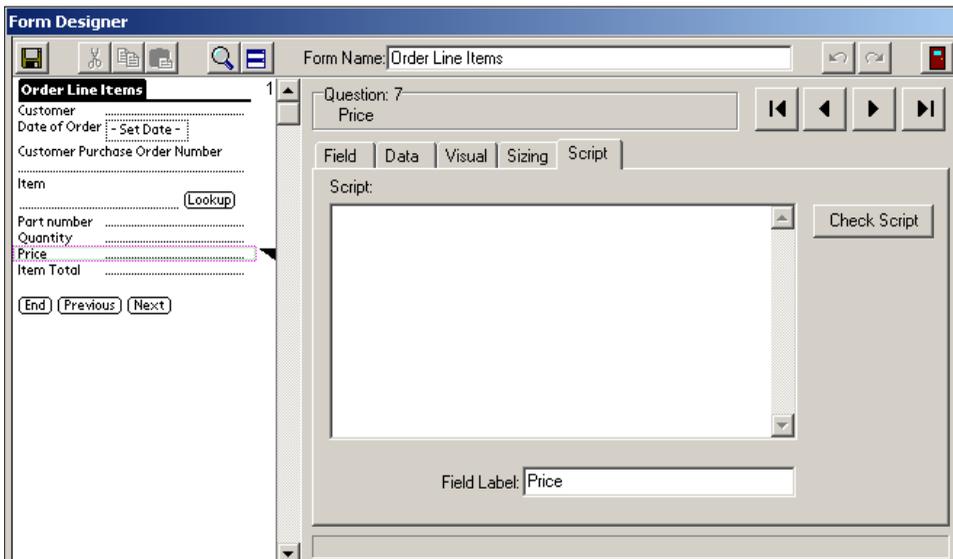
In versions of Pendragon Forms prior to version 3.2, fields had to be referred to by their field number in a script. For example, the value in Field 3 of a form is referred to as \$3 in a script. Or, to jump to Field 17, a script would have a statement such as goto 17.

One limitation of referring to fields by their field numbers is that if you copy a frozen form and then insert fields into the copied form, the field numbers may all change. For instance, what was Field 4 on the original form may become Field 5 on the copied form after an extra field is inserted. If a field number changes, then any references to that field number in any script also has to change.

Forms version 3.2 and higher have the ability to assign a field label to a field, and then reference that field in a script by referencing the field label. In this way, if the field number changes, the script does not have to change.

To assign a field label to a field, click the Script tab in the Form Designer window. In the Field Label field, type a word or phrase that will be the label for that field. The field label must not exceed 50 characters, and must contain only alphanumeric characters and no spaces.

In the picture below, field 7 is given a field label of Price. (The field name is also Price in this instance.)



Field labels are referenced in scripts by placing the label in square brackets. For example, if Field 8 of a form is given a label of ItemTotal, the value in that field can be referred to in a script as

```
[$[ItemTotal]]
```

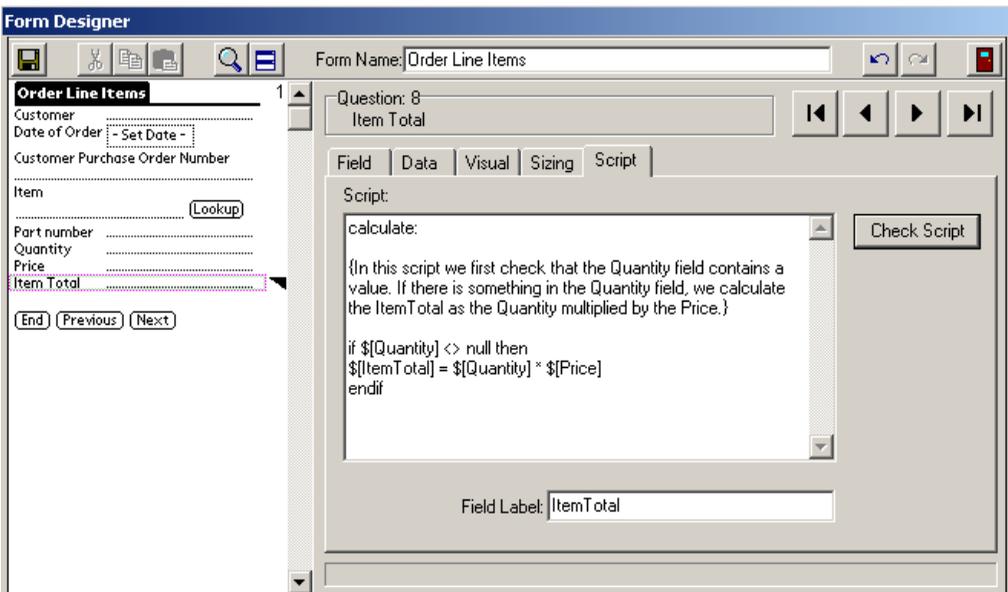
The \$ symbol means the value in the field.

Similarly, scripting statements that might reference a field number, such as goto 8, can now reference the field label, such as:

```
goto [$[ItemTotal]]
```

You can add field labels at any time, even after a form is frozen.

Note, however, that if you assign a field label to a field and then you want to change that field label, any scripts that reference the old field label will have to be manually changed to contain the new field label.



Adding Comments to Scripts

You can add comments to scripts. Comments help to document the script, but are not sent to the handheld and are not part of the script instructions.

To add a comment to a script, put the comment in curly brackets { }.

For example: {Here is a comment}

The picture above shows an example of a comment in a script.

Platform-Specific Scripts

One change in Forms 5.1 is that the program runs the same on both Palm OS devices, and on Pocket PC devices. Previously, the program ran differently on the two platforms, causing a need for different scripts to run on the different platforms under certain circumstances. This is no longer the case.

Platform-specific scripts should therefore not be used in Forms 5.1.

For users who are maintaining Pocket PC devices which are only running the prior version of Forms, version 5.0, platform-specific scripts can still be used.

This only applies to Forms 5.0:

In Forms version 5.0, there may be times when you want a script to run only on a Palm OS device or only on a Pocket PC device. For example, if Pocket PC devices have a feature that Palm OS devices do not, you may want a script that runs on Pocket PC devices only. Or, for example, if certain scripting commands have only been implemented for Palm OS devices, scripts that you write with those scripting commands should be for the Palm OS platform only.

Pendragon Forms 5.0 has flags that you can write into your scripts to indicate that a part of a script is intended for Palm OS devices only or for Pocket PC devices only.

The flags for Palm OS platform are:

```
{+PALMONLY}  
{-PALMONLY}
```

The flags for the Pocket PC platform are:

```
{+PPCONLY}  
{-PPCONLY}
```

Any scripting statements contained between a set of flags will only be sent to devices on that platform.

Event Procedures

An event procedure's label determines when a script will run. Every script must contain at least one event procedure. There are fourteen event procedures. The table below and continuing on the next few pages describes each event procedure, and identifies whether that event procedure will run in Layout View, Field View or Record View on the handheld.

When selecting event procedure labels, remember that Pocket PC devices only use Layout View. Palm OS devices use Layout View as the default, but can use Field View or Record View also.

Event Procedure	Purpose	Layout View	Field View	Record View
enterscreen:	The enterscreen: event runs as soon as the user arrives on a screen in Layout View. The enterscreen: event can be used to pre-fill certain fields on a screen as the user gets to that screen.	√		
exitscreen:	The exitscreen: event runs as soon as the user leaves a screen to go to another screen in Layout View. The user must tap the Next button or End button on a PalmOS device, or the right arrow button on a Pocket PC device. The exitscreen: event can be used to pre-fill certain fields on the next screen based on values entered in the current screen. A goto statement (see page 295) in an exitscreen: event lets the user branch to a specified screen.	√		
enter:	The enter: event occurs when the field is about to be drawn. This means that the script is run when the user displays the field on the handheld. In Field View an enter: event can affect any field as the handheld user views the field. *In Layout View and Record View, an enter: event can only be triggered by Text, Numeric and Currency fields, when the handheld user taps in these fields.	√ (See * note)	√	√ (See * note)

Event Procedure	Purpose	Layout View	Field View	Record View
exit:	<p>In Field View, the exit: event occurs when the user exits a field, that is, the user taps the Next arrow button to move to the next field. The exit: script is only run when the user is moving ‘forward’, for example from field 8 to field 9, and not ‘backward’ from field 9 to field 8.</p> <p>*In Layout View and Record View, an exit: event is only triggered by leaving a Text, Numeric, or Currency field.</p>	√ (See * note)	√	√ (See * note)
select:	<p>The select: event occurs when the user selects an option or accepts a dialog associated with a field. The select: event is typically used in branching, so that when the user selects an option, he/she branches to a field depending on the selection made. A select: event is only triggered in fields that allow a selection, such as a Popup List, Lookup List or a checkbox. Leaving a selection field blank will cause the select: script not to run.</p> <p>*Text, Numeric and Currency fields do not trigger a select: event.</p>	√ (See * note)	√ (See * note)	√ (See * note)
initialize:	<p>The initialize: event runs only once when a new record is created. The initialize: event is often used to set values in date and date&time fields, so that when the user creates a new record, these fields are already filled in.</p>	√	√	√
open:	<p>The open: event runs only once whenever a record is entered. The open: event is used instead of an initialize: event if you need a script to run every time a record is accessed.</p>	√	√	√

Event Procedure	Purpose	Layout View	Field View	Record View
calculate:	<p>A calculate: event is used to place the result of a calculation into a field. Typically, the calculated result depends on the values in other fields. The calculate: event is triggered when any field is updated manually or updated via a script. As you exit the updated field, any calculate: script in Field 1 is run, followed by a calculate: script in Field 2, then Field 3, etc. There is a performance penalty if there are a lot of fields on the form - for example, if you have a form with 150 fields and the form contains one calculate: script, then that script will be run 150 times as you enter data and then exit each field.</p>	√	√	√
click:	<p>A click: event is primarily used in Button fields. The click: event allows a script to run when the handheld user taps on the button in the Button field. See page 364.</p> <p>A click: event can also run when you tap in a Lookup List field, before the list is displayed. This is useful if you are performing a Lookup to Another Form, and you want to filter the records that are displayed in the list. See pages 461-462.</p> <p>A click: event can be run in a Section field, if you tap the question area or graphic in a Section field. See page 475.</p> <p>New since Forms 5.0, a click: event can run in a Subform field.</p> <p>A click: event can run in a Custom Control field, provided that the creator of the Custom Control application allows it.</p>	√	√	√

Event Procedure	Purpose	Layout View	Field View	Record View
scan:	A scan: event runs when a bar code is scanned. All scan: events run in sequence, so that scan: events in Field 1 run first, followed by scan: events in Field 2, etc. A scan: event can only be used with certain bar code scanning hardware - see page 373.	√	√	√
validate:	A validate: event runs when the handheld user leaves a record. The script in a validate: event can be used to perform additional validation on various fields.	√	√	√
cascade:	<p>A cascade: event is used on a subform to allow a change on a parent form to populate a change on the subform records associated with that parent record. A cascade statement in a script on the parent form is used to trigger the cascade: events on the subform.</p> <p>Cascade: events only work with one level of subforms, that is, from a parent to a subform, and not with nested subforms.</p> <p>Cascade: events only work on Palm OS devices.</p>	√	√	√

Event Procedure	Purpose	Layout View	Field View	Record View
report:	<p>The report: event procedure only applies to Palm OS devices.</p> <p>The report: event is triggered by a printreport statement in a script.</p> <p>The report: event specifies the layout of printer output from the handheld. report: events run in sequence, so that a report: event in Field 1 will run first, followed by a report: event in Field 2, Field 3, etc. the report: events on a parent form run first, followed by report: events on a subform. See page 400.</p>	√	√	√
report2:	<p>The report2: event procedure only applies to Palm OS devices.</p> <p>The report2: event runs after all report: events on a parent and subform have been run. Report2: is used if you need to print information from a parent form after all the subform records have been printed. See page 400.</p>	√	√	√

Variables

Variables are used in statements to represent values in calculations, and to refer to the values of fields.

The following variables are used in Pendragon Forms scripts:

- answer** Represents the value of the current field.
Example: `answer = 10` puts the number 10 in the current field.
Example2: `answer = $7` puts the value in field 7 into the current field.
Example3: `answer = result` puts the value that is currently in the **result** variable into the current field.
- result** Used as a temporary variable for storing intermediate calculations. Several scripting statements also place values into the **result** variable.
Example: `result = 6 + 19` stores the number 25 in the result variable.

Example 2: `result = $5 + $6` adds up the values in fields 5 and 6, and places the sum of the two fields in the result variable

Example 3: `result = result + $15` adds the existing value in **result** to the value in field 15, and stores the sum of the two values back in the **result** variable.
- \$number*
\${label} **\$** is a field reference, used to refer to the value in the specified field.
Example: `$5` means the value in field 5.
Example2: `${OrderTotal}` means the value in a field with the field label of OrderTotal.
- temp** **temp** is a 'free' variable that can be used to store data. Whereas scripting statements place values into the **result** variable, the value in the temp variable can be set in a script and will be preserved until you change this value in a script yourself, or until Pendragon Forms is no longer the active application on the handheld.

null

Null is a constant which is equivalent to an empty string.
Example: `$10 = null` sets the value of field 10 to null.

Example 2:

```
calculate:
  if $4 <> null then
    answer = $3 / $4
  endif
```

In this example, if field 4 is not null, then a calculation is performed.

buffer

buffer is another 'free' variable that can be used to store data, similar to the temp variable. The buffer variable can be set in a script and will be preserved until you change this value in a script yourself, or until Pendragon Forms is no longer the active application on the handheld.

lookupname

lookupname is a variable that can be used to determine the name of the Lookup List that will be displayed when a user taps in a Lookup List field. Lookupname is set with a setlookupname statement - see page 370.

lookuplocale

lookuplocale is another variable that can be used to determine the name of the Lookup List that will be displayed when a user taps in a Lookup List field. Lookuplocale is set with a setlookuplocale statement - see page 372.

Functions

Functions are like read-only variables that are used to retrieve recently acquired data or information about the state of the system.

The following functions work on both Pocket PC and Palm OS devices:

column <i>number</i>	If a record is selected with the select statement (see page 320), the column function returns the value of the specified column <i>number</i> in the record.
flashid	Stores the serial number of the handheld device, provided that the device has flash ROM and has a unique flash ID. Not all devices with flash ROM have flash IDs.
now	Now is a function, used to store the current date and time. For an example, see page 354.
recordtimestamp	Returns the creation date and time of the record.
recordmodified	Returns 0 if the record is not modified, or returns 1 if the record has been modified.
scandata	Scandata is a function that stores the most recently scanned bar code. Maximum size can be up to 2000 characters. Text fields are only 255 characters by default, so if your bar code is more than 255 characters, increase the Maximum Length of the Text field before the form is frozen, in order to store the full value of scandata.
scanfield	Scanfield records the field that the user was in when the last bar code scan took place.
scantype	Scantype records the bar code symbology of the last bar code scanned.

selectedbutton	Stores the name of the button that was tapped to generate the current exit event. Valid values are the following case-sensitive text strings (upper-case only): END NEXT OTHER meaning that the End button has been tapped, or the Next button (right arrow button) or another button.
tapx	Stores an X-coordinate (horizontal coordinate) when a user taps in a Section field in Layout View . On a handheld device (even a high resolution device), the screen has X-coordinates ranging from 0 - 159. The tapx function can be used to determine where on a picture in a Section field a user has tapped horizontally.
tapy	Stores a Y-coordinate (vertical coordinate) when a user taps in a Section field in Layout View . On a handheld device (even a high resolution device), the screen has Y-coordinates ranging from 0 - 159. Pendragon Forms reserves Y-coordinates 0-15 for the title bar, and Y-coordinates 145-159 for the buttons at the bottom of the Layout View screen. Valid values for tapy are therefore in the range 16-144. The tapy function can be used to determine where on a picture in a Section field a user has tapped vertically.
username	Username stores the handheld user name of the handheld device.
webdata	Stores the HTML data returned by a server after the Transmit Web statement in a script has sent an HTTP request to the server. The Web server has to return the HTML data between <PFDATA> and </PFDATA> tags, and the Webdata function will store the data that occurs between these tags. The Maximum size of Webdata is 2KB.

cascadename	Cascadename stores the value of the parameter (up to 15 characters) that was given to the last cascade statement. The value of Cascadename can then be used to determine what action is to be performed in a Cascade event on a subform.
reportname	Reportname stores the value of the parameter (up to 15 characters) that was given to the last printreport statement. The value of Reportname can then be used to determine what type of report is printed.
pagenum	Pagenum stores the page number of the page that is currently being printed from the handheld. Before you print anything, the value of Pagenum is 1.
linenum	Linenum stores the line number of the line currently being printed. Before you print anything, the value of linenum is 1.

Scientific Functions

PALM OS

On Palm OS devices, scientific math functions are available in scripts if an additional .PRC file, called MATHLIB.PRC, is installed on the handheld. Forms 5.1 installs MathLib automatically. MathLib is not a part of Pendragon Forms, it is a free shared library that can be used by Palm OS handheld devices.

MATHLIB.PRC is also available in the MATHLIB.ZIP archive which is installed into the C:\Program Files\Forms3 folder.

The scientific functions that are supported via scripts start on page 344.

PPC

On Pocket PC devices MATHLIB is built-in to the Forms 5.1 application that is installed on the Pocket PC. The scientific functions that are supported via scripts are start on page 344.

Operators

The Pendragon Forms scripting language contains unary and binary operators which make it possible to perform calculations.

UNARY OPERATORS

Unary operators act on one variable. Unary operators include:

integer *value*

Converts a number to its integer part.

For instance, 10.6 is converted to 10.

In this example, the answer in the current field is given the value of the integer part of field 5.

```
answer = integer $5
```

- (Minus Sign)

Converts a number to a negative number.

Example: `answer = -$8`

length *value*

Returns the number of characters in a value.

This example returns the number of characters in field 5.

```
calculate:  
answer = length $5
```

BINARY OPERATORS

Binary operators act on two variables. Binary operators include:

- +** (Addition) Adds two values. Example:
`answer = $4 + $5` adds the value in field 4 to the value in field 5.
- (Subtraction) Subtracts one value from another.
Example: `result = $18 - 20` subtracts the number 20 from the value in field 18.
- *** (Multiplication) Multiplies two values.
Example: `$6 = $5 * $4` multiplies the values in fields 5 and 4, and places the result in field 6.
- /** (Division) Divides one value by another.
Example: `answer = $12 / $13` divides the value in field 12 by the value in field 13.
- #** (Contains) Looks for a character string.
Example: `if answer # Red then goto 4 endif` looks for the string Red in the answer. Quotation marks are used if the string has spaces, example:
`if answer # "Critical condition" then require 5 endif`
- %** (Modulo division) Returns the remainder of a division.
Example: `result = $4 % 12` returns the remainder after dividing field 4 by the number 12.
- &** (Concatenate) Concatenates two strings. Example: If field 2 contained the word Red, then `answer = $2 & " apples"` would put the string "Red apples" into the current field.
- AND** Performs a bitwise AND operation on a binary number. Used to extract information from Multi-Selection fields. See page 366.
- OR** Performs a bitwise OR operation. Used to assign a value to a Multi-Selection field.

Statements

There are three types of statements which can be used in scripts: Assignment statements, Conditional statements and Action statements.

Assignment Statements

Assignment statements place a value into a field or into a temporary variable. A dollar sign before a field number means the value in that field. For example, \$7 means the value in field 7, whereas 7 means the number seven.

You can also use field labels to reference a field by a label instead of by field number. If field labels are being used, they must appear in square brackets. A dollar sign in front of a field label means the value in that field, e.g. \$[SubTotal] means the value in a field whose field label is SubTotal.

The assignment statements are:

answer = <expression> Places the result of a calculation into the current field. Note that only simple expressions, involving not more than two values and one operator (+ - * /) can be used. In the following example, the value in field 4 is multiplied by the value in field 5, and the calculated result is placed in the current field.

```
calculate:  
answer = $4 * $5
```

In this example, the value in field 8 is divided by the number 10, and the result is placed in the current field.

```
calculate:  
answer = $8 / 10
```

The Answer statement can be used with field labels. In this example, one field has a field label called ItemTotal and another field has a label called TaxRate. Clicking a button calculates the tax and places the calculated result into the current field.

```
click:  
answer = $[ItemTotal] * $[TaxRate]
```

result = <expression>

Result is a temporary variable. Since not more than two values can be calculated at a time, result is used as extra 'storage space' for intermediate calculations. Several scripting statements also use the Result variable to store the result of a calculation.

In the following example, the values in fields 4, 5, 6 and 7 are being added, and the final calculated result is placed in the current field.

```
calculate:
  result = $4 + $5
  result = result + $6
  answer = result + $7
```

To calculate an expression like $8*(9+7)$, the script would be

```
calculate:
  result = 9 + 7
  answer = 8 * result
```

The Result statement can be used with field labels.

In this example, fields are given labels of Discount, Retail Price and Discounted Price. If the handheld user selects to give a discount, the discount amount is calculated by multiplying the discount by the price. The discounted price is then calculated by subtracting the discount from the retail price.

```
select:
  if answer == Y then
    result = $[Discount] * $[Retail Price]
    $[Discounted Price] = $[Retail Price]
                        - result
  endif
```

\$N = <expression>

This assignment statement is used to assign a value to a field from another field.

In the following example if the handheld user selects Yes, then field 15 is set to the number 10; if the handheld user selects No or leaves the field blank, then field 15 is set to the number 5.

```
select:
  if answer == Y then
    $15 = 10
  else
    $15 = 5
  endif
```

Field labels can be used when assigning a value to a field. In this example a field with the label Daily Total is assigned the value of the sum of a field with the label Total and a field with the label Surcharge.

```
calculate:  
    $[Daily Total] = $[Total] + $[Surcharge]
```

temp = <expression> Temp is a temporary variable, similar to Result. However, whereas the Result variable is used by various scripting statements to store a value, the Temp variable is reserved for use by you, the form designer. This means that if you assign a value to the Temp variable, that value will remain until you change that value or until the handheld user exits the Forms application on the handheld.

buffer = <expression> Buffer is another temporary variable. Similar to **temp**, if you assign a value to buffer, that value is retained until you change the value or until the handheld user exits the Forms application on the handheld.

assign Assigns a value to the current field. Functions in the same way as answer =. However, answer = is preferred because an answer statement is less ambiguous and easier to read.

The following script creates a default value of 100 for a numeric field

```
calculate  
    if answer = null then  
        assign 100  
    endif
```

Conditional Statements

Conditional statements are used to make a decision and take action depending on a response that the handheld user has entered in a field.

There are two types of conditional statements:
IF...THEN statements and SWITCH/CASE statements.

if condition then statements endif

if condition then statements else statements endif

Tests for a condition and allows you to take action if the condition is true. The conditions are made with conditional operators:

= equals (numeric equality)

== string equality (equality between two text strings)

> greater than

< less than

>= greater than or equal to

<= less than or equal to

<> not equal to

contains (string)

This example checks to see if field 27 has been left blank, and if yes, places the value N into field 27.

```
exitscreen:
  if $27 = null then
    $27 = N
  endif
```

In this example, if field 2 contains the word Basement, the script jumps to field 6, and if field 2 contains the phrase Level 1, the script jumps to field 10. Quotes are used around the phrase "Level 1" because there is a space in the phrase.

```
exitscreen:
  if $2 # Basement then
    goto 6
  endif

  if $2 # "Level 1" then
    goto 10
  endif
```

In this example, if the answer selected in the current field is High Priority Case, the script will make fields 12 - 18 visible, and jump to field 12. Any other selection in the field causes fields 19 - 21 to be made visible, and causes a jump to field 19. Quotes are used around the phrase “High Priority Case” because there are spaces in the phrase.

```
select:
  if answer == "High Priority Case" then
    show from 12 to 18
    goto 12
  else
    show from 19 to 21
    goto 19
  endif
```

The use of the == (string equality operator) means that the answer selected must exactly match the text string given. String equality is recommended for use in Popup List, Lookup List or Yes/No checkbox fields, where you can exactly predict the possible answers that the user can select.

IF...THEN...ENDIF statements and IF...THEN...ELSE...ENDIF statements can be used with field labels.

In this example, one field has a label TaxExempt, and another field has a label ItemSum. If the TaxExempt field is No, the current field is calculated as $(1 + \text{TaxRate}) * \text{ItemSum}$. If the TaxExempt field is Yes, the current field is calculated as just the ItemSum.

```
calculate:
  if $[TaxExempt] == N then
    result = 1 + $[TaxRate]
    answer = $[ItemSum] * result
  else
    answer = $[ItemSum]
  endif
```

Pendragon Forms supports the use of nested `if...then...endif` or `if...then...else...endif` statements. This means that you can have one or more `if...then...endif` statements inside another `if...then...endif`.

For example, imagine a field with a field label of Score is a Numeric field for storing a test score. In another field with a field label of grade, a student gets:

Grade A if their score is in the range 90 -100,

Grade B if their score is in the range 80-89,

Grade C if their score is in the range 70-79,

Grade D if their score is in the range 60-69 ,

A failed grade if their score is less than 60.

A nested **`if...then...else...endif`** can be used to detect the range.

```
exitsscreen:
  if ${Score} >= 80 then
    if ${Score} < 90 then
      ${Grade} = B
    else
      ${Grade} = A
    endif
  else
    if ${Score} >= 70 then
      ${Grade} = C
    endif

    if ${Score} < 70 then
      if ${Score} >= 60 then
        ${Grade} = D
      else
        ${Grade} = Failed
      endif
    endif
  endif
endif
```

```
switch value
case testvalue1
    statements executed if value == testvalue1
case testvalue2
    statements executed if value == testvalue2
case testvalue3
    statements executed if value == testvalue3
...
case testvalueN
    statements executed if value == testvalueN
case anyvalue
    statements executed if value was not equal to any of the testvalues
endswitch
```

The switch/case statement is a compact way to test a variable (*value*) against several different constant values (*testvalue1..testvalueN*).

For example, you might have a Popup List or a Lookup List, and may want to take a different action depending on which item is selected in the list. The case components within the switch statement allow you to specify the scripting *statements* that will be run for each item in the list. Note: Switch/case statements do not work with MultiSelection fields.

In this example, a Popup List lets the user select a discount level, and depending on the discount selected, an OrderTotal field is calculated with that discount.

```
select:
switch answer
  case "10% Discount"
    ${OrderTotal} = ${SubTotal} * 0.9

  case "20% Discount"
    ${OrderTotal} = ${SubTotal} * 0.8

  case "25% Discount"
    ${OrderTotal} = ${SubTotal} * 0.75

  case "No Discount"
    ${OrderTotal} = ${SubTotal}

  case anyvalue
    msgbox "No other discounts are available."

endswitch
```

The statements within a switch/case statement can contain if..then statements.

In this example, the user enters an age in a field that has the scripting label Age, and then in field 5, the user selects "Male" or "Female". The next field that the user has to fill in could be one of four options, depending on whether the user selected male aged 50 or more, male less than 50, female aged 50 or more, or female less than 50.

```
exitscreen:
switch $5
  case "Male"
    if ${Age} >= 50 then
      goto 6
    else
      goto 8
    endif

  case "Female"
    if ${Age} >= 50 then
      goto 10
    else
      goto 12
    endif

  case anyvalue
    msgbox "Please select a sex."
    goto 5

endswitch
```

Note that it is optional to use the **case anyvalue** option in a switch/case statement. However, it is good to have an action that occurs if none of the options in the list is selected.

See page 360-361 for another example of a switch/case statement.

Action Statements

Action statements allow a script to perform an action, such as branching to a different field, showing or hiding a field, or displaying a message. The action statements are listed here.

PALM OS means that a scripting statement only applies to the Palm OS platform.

PPC means that a scripting statement only applies to the Pocket PC platform.

abortform Aborts the form, deleting the current record from the handheld device. Can be used in Click, Select, Exit and Scan events only. In the following example, the record is deleted and the form is aborted if the value in field 5 is less than zero.

```
exit:
  if $5 < 0 then
    abortform
  endif
```

acquire "*settings*" "*startpattern*" "*stoppattern*"

Opens the handheld serial port to acquire data from an attached device, and places the collected data in the Result variable.

The *settings* parameter specifies the serial communications, and consists of the baud rate, parity, data bits, stop bit and optionally the handshaking method. (See Transmit Serial statement for typical values.)

The *startpattern* identifies the character(s) in the data string that mark the start of the data. The serial port will read data but not collect the data until the startpattern is encountered.

The *stoppattern* identifies the character(s) in the data string that mark the end of the data. When the stoppattern is encountered, the serial port of the handheld is closed. If the stoppattern is not found, then data will be collected up to a maximum of 2KB or until communications time out (15 seconds of inactivity).

The collected data that is placed into the Result variable does not contain the startpattern, but does include the stoppattern.

In this example, clicking a button on a form opens the handheld serial port at 9600 baud, no parity, 8 data bits, 1 stop bit, using hardware handshaking. Data is immediately collected since there is no start pattern. Data stops being collected when the ASCII 13 (carriage return) character is encountered. The collected data is placed in Field 4.

```
click:
  acquire "9600-N-8-1" "" "\\013"
  $4 = result
```

also

Used to combine `select` statements for the same form, to narrow the selection of records that are displayed when performing a lookup to another form.

Typically, `select` statements only work one at a time, meaning that if one `select` statement is followed by another, the second `select` statement will cause the first `select` statement to be discarded. The use of the `also` statement between two `select` statements causes the first selection to be kept, and the second selection to be performed on the subset of records from the first selection.

Note: The `also` statement only works if the `select` statements are referencing the same form.

As with any lookup to another form, the current form must have a Lookup List field that references the name of the reference form. Also, fields that you want to copy from the reference form to the current form must have the same field name and field type in both the reference form and the current form.

In this example, imagine that Field 2 of the current form contains a State and Field 3 contains a City. The “Employee List” reference form also contains a State and City. The `click:` event script is placed in the Lookup List of the current form that is doing the lookup to the reference form. The first `select` statement selects all employees in the specified State. The `also` statement retains that selection, and the second `select` statement then selects all employees within the selected State who are in the selected City.

```
click:
  select "Employee List" where field 1 is $2
  also
  select "Employee List" where field 2 is $3
```

beam "*caption-name*" {**IR** | **BT** }

PALM OS

Used in conjunction with the `select` and `queue` statements, to beam queued records from one handheld to another.

Beaming should be performed in a `click:` event in a button field, to allow the handheld user to control when the beaming starts.

A `select` statement selects the records from a form, based on the criteria in the `select` statement. `Select` statements include `select all`, `select where`, `select current` and `select matching`. A `queue` statement queues the selected records to be beamed. A `beam` statement actually performs the beaming.

You can have several `select-queue` sequences to queue records from multiple forms. The `beam` statement will beam all queued records from all forms.

The *caption-name* parameter is word or phrase that the handheld user will see when their handheld prompts if they want to accept the beamed data. You can put the name of the form as the caption, or some other phrase. In addition to specifying the *caption-name*, you must specify a beaming method. There are two options:

IR - stands for Infrared. If you choose this option, records will be beamed via the handheld's infrared port.

BT - stands for Bluetooth. If you have handheld devices that support Bluetooth, and you choose this option, records will be beamed via Bluetooth.

In this example, all the records of a form named `Patients` are selected with a `select` statement, but then only the changed records (i.e. new records or modified records) are queued to be beamed. The `beam` statement then beams the queued records via infrared.

```
click:
  select all "Patients"
  queue changed
  beam "Patients" IR
```

In this beaming example, a select where statement is used to select only the records in a form named Work Orders that have a value on N in field 18. These might be incomplete work order records that one user is beaming to another to finish the work. The queue all statement queues all the selected records to be beamed. The beam statement beams the queued records via Bluetooth.

```
click:
  select "Work Orders" where field 18 is "N"
  queue all
  beam " Incomplete Work Orders" BT
```

In this example, changed records from a parent form and a subform are beamed. The parent form is called Patient Main Form, and the subform is called Patient Visit. Each form is selected in turn, and then all the changed records (i.e. new or modified records) for that form are queued to be beamed. The beam statement beams all queued records, so queued records from both parent and subform are beamed.

```
click:
  select all "Patient Main Form"
  queue changed

  select all "Patient Visit"
  queue changed

  beam "Patient Record Information" IR
```

In this example only the current record is selected. The queue changed statement means that the record is only queued for beaming if the record is new or has been modified. To beam the record whether or not it has been changed, queue all would have been used instead of queue changed.

```
click:
  select current
  queue changed
  beam "My new record" IR
```

beep

Sounds a standard warning beep.

In the following example, if the handheld user exits the current field without entering any data, the program will beep and then display a message.

```
exit:
  if answer = null then
    beep
    msgbox
      "Please do not leave this field blank."
  endif
```

call {*creator* | *DLL* | *path*}

The **call** statement allows you to call a custom Palm OS or Pocket PC program. The custom program can utilize and modify the value of the **result** variable. The script continues after call statement with the updated value of the **result** variable. Typically, the **call** statement is used to call programs that perform computations or access peripheral devices. For information on how to write custom programs that can be called in this way, visit the Pendragon Software Web site at <http://www.pendragonsoftware.com>.

PALM OS

On Palm OS, the **call** *creator* statement locates the first application with type 'acry' and the specified creator code, and sends it the custom launch code 33333. The data parameter for the call points to a buffer containing the **result** variable. The called application may modify this buffer, and after the call, the **result** variable (2KB max) will be set to the value in the buffer.

PPC

In Forms 5.1, on Pocket PC, the **call** statement refers to an “Accessory” DLL. The named DLL file should be in the \Program Files\Forms51 folder on the Pocket PC. The Accessory DLL file should have an entry point called “Accessory” which takes a single parameter, a pointer to a null-terminated, wide character array (WCHAR *). The array will contain the current value of the **result** variable, and is modifiable. Changes to the array are reflected in the **result** variable after the **call** statement. The array is 2048 characters long, including the null termination character. Do not overflow the array or the software will crash. Sample code for an accessory can be found on the CD-ROM in the BONUS folder.

callmethod *field-number* "*methodname*"

For use with custom controls that provide methods for you to call from a script.

A method is a term for a callable function that is built-in to a custom control. The callmethod statement calls the named method in the custom control that is located in the specified field.

A custom control is a separate program from Pendragon Forms. If the custom control is not loaded or visible on the current screen of your form design on the handheld, then you cannot use callmethod (it will have no effect).

The result variable is passed to the custom control, and the result variable may be modified by the call.

Each custom control will have its own documentation that specifies what methods are available to be called. See page 162 for information on Custom Controls and how to access the documentation for a Custom Control.

In this example, the GPS Custom Control in field 5 has a method called GPSINFO. The GPS Custom Control runs the click: event script written in field 5. The GPSINFO method puts GPS data into the result variable. This data is copied to the temp variable, and then extract statements are used to extract the GPS Latitude and put the value in field 6, the GPS Longitude is put in field 7, etc. (Refer to the GPS Custom Control documentation starting on page 574 for more information.)

```
click:
  result = ""
  callmethod 5 "GPSINFO"
  if result <> null then
    temp = result
    extract "GPS.LAT" from temp
    $6 = result
    extract "GPS.LONG" from temp
    $7 = result
    extract "GPS.KNOTS" from temp
    $8 = result
    extract "GPS.ALTITUDE" from temp
    $9 = result
  endif
```

cascade label

The cascade statement on a parent form triggers cascade: events on each subform record associated with the parent record. For example, if a Completion Checkbox is marked on the parent, a cascade statement on the parent can be used to trigger cascade: events on the subform records that mark the subform records as complete also.

(Note: You do not currently need to use cascade in a script in order to cascade Completion checkbox fields from parent to subform. The Forms program will automatically do this for you.)

The *label* parameter that is given to the cascade statement can be retrieved in a cascade: event using the Cascadename function. The label can be up to 15 characters. The value of Cascadename can be used on the subform to determine what action to take on the subform.

If you want a Completion Checkbox on a parent form to cascade to a Completion Checkbox on the subform, the following script can be used in the Completion Checkbox field on the parent form:

```
select:
  cascade answer
```

If the handheld user selects Yes in the parent form's Completion Checkbox, the `cascade answer` statement places the value Y (for Yes) in the Cascadename function. The following script in the subform then causes all the subform records associated with that parent record to be marked as Y also. (The subform script is in the Completion Checkbox field of the subform in this case.)

```
cascade:
  answer = cascadename
```

Important: A cascade statement and a cascade: event cannot be used to make a change to any of the fields that are common to both the parent and the subform. This is because Pendragon Forms uses the values that are in fields with identical names on both the parent and the subform in order to determine which subform records are associated with a given parent record. If a field that exists on both parent and subform is changed on the parent, you will 'orphan' the subform records, because the subform records will no longer match the parent.

checkflag statements clearflag

Checkflag is used in conjunction with **clearflag**, to enable you to modify a record with a script without marking the record as modified. This is useful when you want to have your form update some fields for cosmetic purposes without causing the changes to be synchronized back to the server.

For instance, the following script:

```
open:
  lookup $5 within "Expense Category"
```

might be used to lookup a value in a reference table every time the form is opened. Such a script is typically used when some of the fields on the form are not synchronized from the desktop to the handheld, but must be calculated or looked up instead. (Database developers call this de-normalization.)

While this technique enables the handheld user to see related values on the screen, it has the side effect of modifying the record. This would mean that the simple act of viewing a record marks that record as modified, and causes it to be sent back to the desktop, overwriting the original record.

To prevent this side effect from occurring, use the **checkflag** statement to "memorize" the modification state of the record before any unintended modifications are made. After performing the script statements that you do not want changing the record, use the **clearflag** statement. Clearflag sets the modified flag back to the memorized value that it had when checkflag was called.

Using checkflag and clearflag, the script above would now be:

```
open:
  checkflag
  lookup $5 within "Expense Category"
  clearflag
```

NOTE: Any scripting actions that are supposed to modify the record should be made before checkflag or after clearflag. Otherwise the record will not properly be marked as changed, and will not synchronize.

clone

The clone statement saves the current record, then creates a new record with identical values in all the fields as the original record. Scripting statements that occur after the clone statement are applied to the new record.

Clone statements work well when using barcode scanners, as the clone statement reduces the need for manual data entry. See page 386 for an example of using a clone statement in a scan: event script.

When you use the clone statement, no validation is done on the record that you are leaving nor on the new record created by the clone statement. If you are using your own primary key, the combination of the primary key fields has to be unique for every record. You can use the `keyunique` statement before the clone statement, to check if the record that you are leaving has a unique primary key. If the record that you are leaving does not have a unique primary key, you can display a message to alert the user, and abandon cloning the record.

In this example, if you have a form in which fields 1, 2 and 3 basically remain the same, but fields 4 and 5 vary every time you record data, you might make Field 6 a Yes/No checkbox that asks the handheld user “Do you need to make another reading?” If the user selects Yes in the following script, the record will be cloned and the values in fields 4 - 6 will be made null so that the user can enter the new values for these fields while retaining the existing values in fields 1 - 3.

```
select:
  if answer == Y then
    clone
    $4 = null
    $5 = null
    $6 = null
  endif
```

column *number*

Column is a function that is used with a `select` statement to retrieve the value in a specified field of a selected record in a reference form.

The *number* is the column number (i.e. the field number) of the reference form.

In this example, a bar code is placed into field 1 of a form, and then a select statement is used to lookup that barcode in a reference form called Inventory. If a match is found, field 2 of the current form is populated with data from column 3 of the reference form. Field 3 of the current form is populated with data from column 5 of the reference form.

```
scan:
  $1 = scandata
  select "Inventory" where field 2 is $1
  $2 = column 3
  $3 = column 5
```

count *field-number*

Checks the current record against all the other records in the database, to see how many records have the same value in the specified field. The number of records that match is placed into the Result variable.

Note: For performance reasons, the count statement should not be used in calculate: events.

If you need to check for uniqueness in a primary key, use the `keyunique` statement instead of `count`, because `keyunique` can check for uniqueness even if the primary key is comprised of more than one field.

This example checks to see if field 6 is unique.

```
validate:
  count 6
  if result > 0 then
    msgbox "This is a duplicate barcode"
  endif
```

delete

Used in conjunction with select statements to delete all the selected records in a reference form.

WARNING: Should only be used with extreme caution to avoid accidentally deleting records from a reference form.

Records are selected with a `select`, `select matching` or `select where` statement.

Not recommended for use with `select all` statement, due to the possibility of deleting all records in a reference form.

deletemode "formname"

Used in a **click:** event in a Section field or a Button field of a Custom Main Menu form to allow the user to be able to delete records from the specified form.

With a Custom Main Menu form, users cannot access the Delete screen to delete records unless you provide this option via the `deletemode` statement. The Delete screen allows users to delete form designs as well as records, so typically you may want set the Advanced Form Property to Disable Form Deletion.

See page 480 for an example of using the `deletemode` statement.

disable barcode

Disables the barcode scanner of a handheld device. To switch on the barcode scanner, see the `enable barcode` scripting command. For a list of compatible devices, see Chapter 15, *Using Bar Codes*, page 373.

This scripting command is equivalent to switching off the laser in the Diagnostics application of an SPT 1550/1800 - (Scanner Enabled checkbox unchecked). This can be used to prevent the handheld user from scanning a bar code into a Text field which is not supposed to be a bar code field.

Important: Setting the scanner is a global setting, meaning that if you switch off the scanner or enable just one bar code symbology in a script, then these settings will apply across all fields and forms, and possibly across other applications on the handheld (unless the other applications also make global settings). Since scanner settings are global across all fields, if you switch the scanner off in one script and forget to switch the scanner on again, you will not be able to scan into any field on the form.

To prevent the handheld user from scanning randomly into fields on a form, the following script can be used to switch off the laser scanner:

```
open:
  disable barcode
```

The laser scanner can then be switched on when needed. In this example, the scanner is switched on once the current field is filled.

```
exit:
  if answer <> null then
    enable barcode
  endif
```

disable { endbutton | nextbutton | backbutton | navbuttons | icons | menus }

Note: It may be easier to switch off the use of these options via Advanced Form Properties - see page 204.

In some form designs, you may want to hide certain buttons and menu options that are usually available to the handheld user. The `disable` statement will hide the specified button, and the `enable` statement will display the specified button. Disable be used in an **open:** event if you want the button or menu to be hidden in both new and existing records.

The `endbutton` specifier will hide the End button. You may want to hide the End button to prevent a handheld user from exiting a form before viewing all the fields on the form. Warning: if you hide the End button on a form, you will have to give the user a way to exit the form, such as a Button field that when tapped, runs a script to end the form.

The `nextbutton` specifier will hide the Next button (in Layout View) or the right arrow button (in Field View). You may want to hide the Next button on a form that is being used as a 'main menu' from which the user jumps to other forms.

The `backbutton` specifier will hide the Previous button (in Layout View) or the leftarrow button (in Field View). You may want to hide the Previous button on a form that is being used as a 'main menu' from which the user jumps to other forms. Or you may want to hide this button if your form requires users to answer a question and not go back to view or change answers.

The `navbuttons` specifier will hide the navigation buttons in Record View. The Navigation buttons in Record View allow the handheld user to jump from record to record, and can be disabled if you want the handheld user to stay within a given record until the form is filled out.

The `icons` specifier will hide/show the timestamp/datestamp icons that are usually visible in Text fields in Field View. Normally, the handheld user can tap these icons to add the current date or date & time to a Text field.

The `menus` specifier will hide certain options that normally appear if the handheld user taps the handheld Menu button (the drop-down menu icon below the house icon on the handheld). The menus that are disabled include: Clone, Mark All Changed, Print, Print All, Acquire Barcode, Acquire GPS. You may want to hide these menu options if your form design does not use them and you do not want to allow the handheld user access to these menu options.

In this example, the End button, Navigation buttons and menu options are all disabled in an `open:` script in Field 1 of a form:

```
open:
  disable endbutton
  disable navbuttons
  disable menus
```

Since the End button is disabled in the above script, a Button field is added to the end of the form to give the handheld user a way to exit the form. The script in the Button field is:

```
click:
  endform
```

disable grabba

PALM OS

Closes the port of a Grabba International bar code scanner attached to a PalmOne handheld. Pendragon forms will close the Grabba port at certain intervals, but it is recommended that you disable the Grabba whenever you can safely do so, such as when exiting a record.

See **enable grabba** for information on activating the Grabba port.

See page 373 for more information on using Pendragon Forms with Grabba devices.

In this example, the grabba port is closed in a `validate:` script when a record closes:

```
validate:
  disable grabba
```

drawimage *top height filename*

The **drawimage** statement is used to draw an image on the handheld screen without using a Pendragon Forms Image field. This is useful if you want to keep a collection of image files on a VFS volume (a storage card, usually an SD card or Memory Stick), and display the images within Pendragon Forms without having to synchronize the images to the Forms Manager database on the PC. Synchronizing images to a storage card is time-consuming if the images are not changing, but other fields on the form are changing.

Using the **drawimage** statement, there is an alternative to using Image fields. You can copy images to a directory on the card using some external mechanism (e.g. a flash memory drive) and maintain only the image filename in your Pendragon Forms record. When you need your form to display an image, you can use the **drawimage** statement to draw the named file on the card onto the handheld screen.

The **top** parameter is the vertical pixel at which the image will be painted. The **height** parameter is the maximum amount of vertical space used to display the image.

Pendragon Forms always refers to the handheld screen as though it has a 160x160 pixel resolution, with coordinates ranging from 0 - 159. Coordinate 0 is the upper left corner of the screen.

The title bar of the application takes up 16 pixels, and the buttons at the bottom of the application take up 16 pixels as well. Therefore, although the coordinates range from 0 - 159, the largest available on-screen space has **top** pixel 16 (to allow for space for the title bar) and **height** pixel 128 (to allow space for title bar and the buttons at the bottom of the screen).

Note that the **drawimage** statement will overwrite controls on the screen if you allow it to. This is not recommended because you may see unusual effects. For example, if the **top** parameter is set to zero, the image will overwrite the title bar.

For high resolution devices, such as 320x320 pixels or 320x480 pixels, Pendragon Forms still treats the screen as a 160x160 pixel screen, and scales up automatically. Therefore, always write your script as though it is intended for a 160x160 screen.

The filename parameter in a drawimage statement is the name of the file on the VFS volume, including the path to that file.

In this example, an image called 1234.jpg, located in a folder called employees, is displayed on handheld screen:

```
enterscreen:
  temp = "/employees/1234.jpg"
  drawimage 16 128 temp
```

The image starts at top position 16, and has height to pixel 128, which is the largest available area on the screen.

Note that the forward slash character (/) is used to separate folder names. New in Forms 5.1 is that the forward slash character is used to separate folder names for both the Palm OS and the Pocket PC platforms.

In this example, imagine that records are created on the PC and downloaded to the appropriate handheld. Field 3 of each record is a Text field containing the path and file name of the image file to be drawn for that record. This script draws the image:

```
enterscreen:
  drawimage 16 128 $3
```

Refer to page 433 to see an example using drawimage.

enable barcode

Enables the barcode scanner of a handheld device. To switch off the barcode scanner, see the `disable barcode` scripting command. For a list of compatible devices, see Chapter 15, *Using Bar Codes*, page 373.

This scripting command is equivalent to switching on the laser in the Diagnostics application of an SPT 1550/1800 - (Scanner Enabled checkbox checked).

Important: Setting the scanner is a global setting, meaning that if you switch on the scanner in a script, then these settings will apply across all fields and forms, and possibly across other applications on the handheld (unless the other applications also make global settings). Since scanner settings are global across all fields, if you switch the scanner on in a script you will be able to scan into any field on the form.

To prevent the handheld user from scanning randomly into fields on a form, the following script can be used to switch off the laser scanner:

```
open:  
  disable barcode
```

The laser scanner can then be switched on when needed. In this example, the scanner is switched on when the handheld user reaches the current screen of the form (in Layout View).

```
enterscreen:  
  enable barcode
```

enable grabba

PALM OS

Grabba International manufactures a line of bar code scanners that clip onto PalmOne handheld devices. Pendragon Forms supports Grabba scanners, so that scan: events and text and numeric fields respond to scans with the Grabba hardware.

To activate Grabba compatibility, you need to place a grabba enable command in an open: or an enterscreen: event on your form.

For example:

```
open:  
  enable grabba
```

For more information on Grabba hardware, see page 373.

enable { endbutton | nextbutton | backbutton | navbuttons | icons | menus }

Note: It may be easier to switch off the use of these options via Advanced Form Properties - see page 204.

In some form designs, you may want to hide certain buttons and menu options that are usually available to the handheld user. The `disable` statement will hide the specified button, and the `enable` statement will display the specified button. Disable should be used in an **open:** event to ensure that the button or menu is hidden in both new and existing record. Enable can be used in a scripting event where it makes sense to do so. For example, if you use the `disable` statement or Advanced Form Properties to hide an End button, you can have a button field with a **click:** event that allows the user to tap the button to end the form. Or you can have a Yes/No Checkbox field and use a **select:** event script that ends the form if the user selects Yes.

The `endbutton` specifier refers to the End button that allows the user to end a form at any time.

The `nextbutton` specifier refers to the Next button (in Layout View) or the right arrow button (in Field View).

The `backbutton` specifier refers to the Previous button (in Layout View) or the left arrow button (in Field View).

The `navbuttons` specifier refers to the navigation buttons in Record View. The Navigation buttons in Record View allow the handheld user to jump from record to record.

The `icons` specifier refers to timestamp/datestamp icons visible at the bottom of Text fields in Field View. (In Layout View you can choose on a field by field basis whether to display the Text icons in a text field.)

The `menus` specifier refers to certain options that normally appear if the handheld user taps the handheld Menu button (the drop-down menu icon below the house icon on the handheld). The menus include:

Clone, Mark All Changed, Print, Print All, Acquire Barcode, Acquire GPS. If you disable the menus via a disable script or Advanced Form Properties, and you want to use a script to enable the menus, you will turn on all the menus. You cannot select a particular menu to enable.

In this example, the End button, Navigation buttons and menu options are all disabled in an open: script in Field 1 of a form:

```
open:
  disable endbutton
  disable navbuttons
  disable menus
```

Since the End button is disabled in the above script, a Yes/No Checkboxfield is added to the end of the form to give the handheld user a way to exit the form. If the user selects yes to end the form, the end button appears. The script in the Yes/No field is:

```
select:
  if answer == Y then
    enable endbutton
  endif
```

Also, see the use of endform as a means of exiting a form that does not have an End button.

endform

Equivalent to pressing the End button. Saves relevant data, then exits the form. This statement is only available in click:, select:, exitscreen:, exit: and scan: events.

In the following example, if the answer in the current field contains the word Finished, then the record is saved and the form is exited.

```
exitscreen:
  if answer # Finished then
    endform
  endif
```

In this example, a Button field is used to exit a form.

```
click:
  endform
```

extract "*entity*" from *value*

Extracts XML formatted data from the specified value. The XML formatted data can originate from a transmit web statement, a call to an external CodeWarrior application or a Custom Control.

Extracted data is placed into the Result variable.

The *entity* parameter defines the XML tags that delimit the data. The *value* parameter depends on where the XML data originates from.

When the Transmit web statement is used, data returned from a Web server is stored in the Webdata function. The Extract...From statement can be used to extract XML from Webdata. In this case the *value* parameter in the extract statement is the Webdata function. See page 443 for an example.

In this example, clicking a button on a form sends a part number stored in Field 6 to a Web URL. The data returned by the Web server is stored in the Webdata function, and the Extract statement is used to place a sale price into Field 8 and the quantity in stock in Field 9.

```
click:
  transmit web "http://www.site.com/cgi-bin/
inventory.cgi?partnum=$6$"
  extract "saleprice" from webdata
  $8 = result
  extract "qtyinstock" from webdata
  $9 = result
```

exportfile *questionID* "*filename*"

Exports the image or object attached at the specified field position (*questionID*) to the named path and file (*filename*). This provides a way to export a binary object from the Forms database on the handheld to a storage card in the case of Palm OS devices, or to any folder on a Pocket PC (including a storage card).

The field in position QuestionID must be an Image field. Most of the time, the field will contain graphics (.BMP, .JPG, .GIF files), but an Image field can, in principle, contain any type of data file.

In this example, tapping a button exports the binary object in field 5 to a path and file called /soundbite.wav. The developer must know in advance that the object in field 5 is a .WAV file.

```
click:
  exportfile 5 "/soundbite.wav"
```

In this example, tapping a button exports the binary object in field 8 to a path and file contained in field 3.

```
click:
  exportfile 8 $3
```

fieldview

Used on Palm OS devices to switch from Record View to Field View. Note that Layout View lets you can control how many fields are visible per screen, so you can design a form layout that works for your users without having to resort to using scripts to manipulate the layout of a form.

The `fieldview` command only takes effect after an entire script is run, so a `fieldview` statement in the middle of a script does not mean that the form will switch to Field View and then the remainder of the script will run.

If you want the handheld user to default to Field View, set the Advanced Form Property for the default view for new records and the default view for reviewing records, instead of using scripts. See page 202.

The `fieldview` statement can be used to automate “zooming in” to Field View from Record View for certain types of fields. For example, it may be easier for the handheld user to enter data in a Text field in Field View, but then return to Record View for the rest of the form. The `recordview` statement is the counterpart to the `fieldview` statement.

In this example, entering a text field switches to Field View, and then exiting the field returns to Record View:

```
enter:
  fieldview

exit:
  recordview
```

filtercount *field-number*

Used to check uniqueness within a subform. In a subform, records are filtered to display only those which match the parent record. A filtercount statement should be placed in a script in the subform, to check the current record against currently filtered records, to determine how many filtered records match in the specified field. The number of records that match is placed into the Result variable.

Note: For performance reasons, filtercount should not be used in a calculate: event.

This example checks to see if field 13 in a subform is unique:

```
validate:
  filtercount 13
  if result > 0 then
    msgbox "A record already has this value."
  endif
```

font {**answer** | **question**} {**bold** | **normal** | **large**}

Sets the font for the question or answer area of the current field. This scripting command only applies to Field View and is only available in enter:, select: and exit: events.

Layout View does not require this scripting command because you can set the question and answer font of each field separately when designing the form (see page 172).

The following example causes the field name to be displayed in bold font, and the handheld user's response to be displayed in large font.

```
enter:
  font question bold
  font answer large
```

format *value* {**date** | **time** | **datetime** | **currency** | **fixed**}

Internally, Pendragon Forms stores dates and times as a number - the number of seconds since 01/01/1904. Currency fields are stored internally as a number of cents. No special formatting is required if these types of fields are stored in the appropriate field type - for instance, if a currency amount is stored and displayed in a Currency field, no formatting is required.

However, if a Date, Time, Date&Time or Currency field needs to be printed directly from the handheld, or displayed in a Text field, the format statement can be used to display the field in a way that is readable to the handheld user. The formatted *value* is placed in the Result variable.

The following example causes the date in Field 1 to be formatted, and the formatted date to be placed in Field 8.

```
calculate:
  format $1 date
  $8 = result
```

The `fixed` specifier displays a currency without a currency symbol, and the `currency` specifier displays a currency with a currency symbol. In this example, if Field 7 contains the value of \$15.95, then Field 8 will contain 15.95 and Field 9 will contain \$15.95.

```
calculate:
  format $7 fixed
  $8 = result
  format $7 currency
  $9 = result
```

formsum "*name-of-form*" *fieldnumber*

Adds up the values across all records for the specified form and the specified field, and places the calculated result in the Result variable.

In the following example, clicking a button adds up all the values of field 17 across all records in the form "Daily Sales". The result is then put in field 8 of the current form.

```
click:  
formsum "Daily Sales" 17  
$8 = result
```

getaddress

PALM OS

Prompts the user to select an address from the AddressBook application and then pastes the contact information for the selected record into corresponding fields on the form. Use in a Button field with a **click:** event. Not available in calculate: events.

The fields on the form must have one or more of the following field names (including uppercase characters and colon (:) character) to receive the corresponding AddressBook data.

First name:

Last name:

Title:

Company:

Address:

City:

State:

Zip:

Country:

Tel:

Other:

Email:

Custom:

Tel refers to the Work phone number. Custom is the Custom 4 field.

In the following example, a click: event is used in a button field to allow the handheld user to lookup an AddressBook contact by clicking a button.

```
click:  
  getaddress
```

See page 364 for an example.

goto *field-number*

Moves to the specified field on the form and then exits the current script. Can be used in the following scripting events:

select: event**click:** event**scan:** event**exitscreen:** event (in Layout View only)**exit:** event (in Field View only)

In the following example, when the handheld user exits the current screen in Layout View, the script checks the response in field 6. If the response contains the word Red, the script jumps to field 7; if the response contains the word Blue, the script jumps to field 12; if the response contains neither Red nor Blue, the script jumps to field 20.

```
exitscreen:
  if $6 # Red then
    goto 7
  endif

  if $6 # Blue then
    goto 12
  else
    goto 20
  endif
```

With a goto statement, a field must be visible for the script to be able to execute. If a field is hidden, the goto statement will go to the next visible field. For instance, if a script says goto 5, but field 5 is hidden, the goto statement will go to field 6. If there are no more visible fields after the specified field, the goto statement will do nothing.

Generally, if you use a goto statement to go to a specific field, you should also include a goto statement for the case in which you do not want to go to that field. Normally, Forms will move you to the next field automatically. However, if you use a script Forms may not move to the next field in the normal way, unless you add another goto statement. In this example, selecting Yes in a Yes/No checkbox jumps you to Field 5. Selecting No or not making a selection at all, jumps you to Field 4.

```
select:
  if answer == Y then
    goto 5
  else
    goto 4
  endif
```

gotosubform *formname* { **new** | **review** | **normal** | **oneway** }

The Gotosubform statement allows programmatic jumps to subform records without the use of a Subform List field on the parent form. The Gotosubform statement can be used to make the transition from a parent form to a subform transparent to the handheld user.

The `normal` specifier causes the Gotosubform statement to behave just like a regular Subform List field, meaning that a review screen of existing subform records is displayed and the handheld user can review an existing subform record or to create a new record. In this example, a button field is used to allow the handheld user to access a subform:

```
click:
  gotosubform "Customer Contacts" normal
```

The `new` specifier causes a new subform record to be created, without the handheld user seeing a list of existing subform records. In this example, clicking a button creates a new subform record.

```
click:
  gotosubform "Item Ordered" new
```

The `review` specifier is similar to using a Single Subform field on the parent form, and provides a way to create a form with more than 250 fields by linking together more than one form design. Using the review specifier displays the most recently created subform record, or if none exists, a new subform record will be created.

In this example, clicking a button jumps to the next form.

```
click:
  gotosubform "Survey2" review
```

With `gotosubform normal`, `new` and `review`, when the user ends the subform, he/she returns to the parent form. The `oneway` specifier, however, allows you to return to any form: a parent or a grandparent form. Normally if you go from form A --> B --> C --> D, when you end form D you have to return from D --> C --> B --> A.

With `gotosubform oneway`, you can choose, for example, to go from form A --> B --> C --> D --> A or from form A --> B --> C --> D --> B.

The destination form retains any parent-child association (if any) that it had previously, so if you jump from Form D --> B, and B is a subform of A, then when you end form B you would return to form A.

In this example, the script is placed in a Button field in Form D.

```
click:
  gotosubform "Form B" oneway
```

hide *field-number*

hide from *field-number to field-number*

The Hide statement is used to hide a single field on a form. Hide From...To is used to hide multiple fields. Works in all views on the handheld: Layout View, Field View and Record View.

In this example, as the user exits the current screen, field 51 is hidden if the answer in the current field is Y:

```
exitscreen:
  if answer == Y then
    hide 51
  endif
```

This example hides fields 4 - 20 if the value in the current field is 1.

```
exitscreen:
  if answer = 1 then
    hide from 4 to 20
  endif
```

imagefilename *fieldnumber*

Fieldnumber is the field number of an Image field.

Given the specified *fieldnumber*, the **imagefilename** statement places into the **result** variable the name that an image in that field must be called in order to be attached to the image field in Pendragon Forms.

For example, if the value placed into the result variable is: "0ABCDEF80000000A0003", this means that 0ABCDEF8 is the hexadecimal value of the Form ID number, 0000000A is the hexadecimal of the Record ID, 0003 is the hexadecimal of the field number. The image file also needs a file extension in addition to a file name.

The imagefilename statement just tells you what an image file should be called in order to be attached to the Image field of that *fieldnumber*. Note that the imagefilename statement does not actually assign the file name to an image, nor does it attach the image file to the field. Other statements in the script will be needed to perform these function.

The imagefilename statement was designed for use with the Zire Camera Custom Control that ships with Pendragon Forms. For instructions on this custom control, visit the Support page at www.pendragonsoftware.com.

importfile *questionID* "*filename*"

Imports the named file (*filename*) into the specified Image field at field position *questionID*.

Most of the time, the files to be imported into Image fields will contain graphics (e.g. .BMP, .JPG or .GIF files), but they can, in principle, contain any kind of data file. For example, a developer could write an accessory program that brings binary data into Pendragon Forms. The accessory program could create a binary file on the device file system (a storage card on Palm OS devices, or anywhere in the file system on Pocket PC), and then pass that filename to Pendragon Forms. After a CALL script, Pendragon Forms would use importfile to attach the binary file to the record.

In this example, tapping a button imports a file named image1.jpg from the DCIM folder of a storage card, and places the image into field 15.

```
click:
importfile $15 "/DCIM/image1.jpg"
```

insert into "*formname*" Creates a new record in the specified form and selects the new record. The handheld user cannot see the record, and the only means for filling in the fields of this record is via the use of the **update field** statement .

In this example, when the handheld user taps a button, the **select** statement checks if field 2 on the current form matches field 1 on the "Inventory" reference form. If not, a new record is created in the "Inventory" form, and fields 2 and 3 of the current form are copied into fields 1 and 2 of the new record.

```
click:
select "Inventory" where field 1 is $2
if result = 0 then
insert into "Inventory"
update field 1 = $2
update field 2 = $3
endif
```

invalidate “*message*”

Only used with the **validate:** event. The invalidate statement flags the record as having an invalid value, and the handheld user cannot exit the record until the invalid value is corrected. A message is displayed on the handheld screen, with buttons for the user to Edit or Delete the record.

Note that a validate: script only runs when the user exits the form. If the form has a lot of fields, it may be easier to use an exitscreen: script (in Layout View) to check that fields contain valid data before the user moves to the next screen.

In this example, when the user tries to exit a record, if field 25 has a value greater than 100, an error message is displayed and the record is invalidated. The user has to edit field 25 or delete the record.

```
validate:
  if $25 > 100 then
    invalidate "Grade cannot exceed 100 points"
  endif
```

keycolumn *number*

Normally, the **lookup value within formname** statement (page 302) finds a record in the form *formname* by matching the specified *value* to contents of the Display Key field of the form *formname*. (If no Display Key is specified, field 1 of the form is the Display Key field by default.)

The keycolumn statement allows you to perform a lookup to another form using a field other than the Display Key field of the reference form. The specified *number* is the alternate column of the reference form that the lookup...within statement will use to search for a matching record. After the lookup...within statement is run, the keycolumn is reset to the Display Key field once more.

In this example, the lookup...within script searches field 5 of a form called “Inventory Form”.

```
click:
  keycolumn 5
  lookup $3 within "Inventory Form"
```

keyunique

Determines whether the primary key of the current record is unique on the handheld.

Keyunique places the value 1 in the Result variable if the primary key of the current record is unique on the handheld, or places the value 0 in the Result variable if the primary key is not unique.

Keyunique is typically used before a **clone** statement. For instance, if you have a barcode as a primary key field, and you are using the clone statement to create new records automatically, before you clone a record you can use keyunique to check that the record has a unique primary key. If the record does not have a unique primary key, the user can be prompted to correct the error before making a new record.

In the example below, field 1 is used to store a barcode. When the user scans a barcode, if there is already a value in field 1, the keyunique statement is used to check that the existing barcode (primary key) is unique. If it is not unique, a message is displayed to the user and field 1 is erased. If the primary key is unique then the record is cloned and the new barcode is placed in field 1 of the new record. Keyunique is used again to check that the new barcode is unique, and if it is not, then the barcode is erased from field 1.

```
scan:
  if $1 <> null then
    keyunique
    if result = 1 then
      clone
    else
      msgbox "Barcode has already been scanned."
      $1 = null
      return
    endif
  endif

$1 = scandata
keyunique
if result = 0 then
  msgbox "Barcode has already been scanned."
  $1 = null
endif
```

launch *creator*

PALM OS

Launches a CodeWarrior application that contains the specified creator code. The Launch statement ends the Pendragon Forms application and launches the specified application.

PPC

On a Pocket PC device, the *creator* specifier should be a path to a Pocket PC .EXE application.

left *value length*

Extracts the left-most characters and places the extracted characters into the Result variable.

The **value** specifier is the value from which the left-most characters are to be extracted.

The **length** specifier is the number of characters to be extracted.

The following example puts the value MXP in the current field.

```
enter:
  left "MXP53802" 3
  answer = result
```

This example puts the first 4 characters from field 16 into the Result variable, and then into the current field.

```
enter:
  left $16 4
  answer = result
```

lookup *value* **within** *lookup-list-name*

lookup *value* **within** *form-name*

Used to lookup a value in a Lookup List or in another form.

The **value** specifier is the display entry to search for.

If a *lookup-list-name* is specified, this is the name of the Lookup List to search. The Lookup List that you use should be set to Store Lookup Values in the Lookup field (see page 115). The value that is found (if any) is put into the Result variable. Your script can then place the value in the **result** variable into a field on the form.

Imagine that you want to create a form with a field called Item Name and a field called Price. When you select the Item Name in field 1, you want to see the correspond Price appear in field 2. To create this requires two Lookup Lists on your form. One Lookup List contains the item names. The second Lookup List, called 'Prices', contains the item names and prices, and is set to Store Lookup Values in the Lookup field. The script in field 1 should be as follows:

```
select:
  lookup $1 within "Prices"
  $2 = result
```

If a *form-name* is specified instead of a Lookup List name, the lookup...within script will perform a lookup to the specified form. The field that is being looked up must be the Display Key on the reference form. See page 391 for an example. If a match is found in the reference form, all fields that are named the same on both forms will be copied from the reference form into the current form.

You can use the **keycolumn** statement to look up a field other than the Display Key field when doing a lookup...within another form.

Note: You do not need to use a lookup...within script to perform a manual lookup to another form - see page 123. Where a lookup...within script is useful is if you want to perform the lookup to another form in a **scan:** event when a bar code is scanned into a field - see page 391.

mid *value start length*

This function copies *length* characters from *value*, starting at position *start*, and places the characters into the Result variable. The following example places “123” into the Result variable, and then into the current field.

```
calculate:
mid "012345678A" 2 3
answer = result
```

In this example, 10 characters are extracted from field 5, starting from the 7th character position in field 5. The extracted characters are placed in field 6.

```
calculate:
mid $5 7 10
$6 = result
```

msgbox “*string*”**msgbox** “*prefix+string*”

Displays a dialog box on the handheld. An individual string cannot exceed 64 characters. If you need longer strings, **concatenate** two strings and place into the Result variable, then use **msgbox result**. This example displays a message if the user enters a value greater than 100 in the current field.

```
exitsscreen:
if answer > 100 then
msgbox "The temperature is very high"
endif
```

If you just use a string, the message box dialog will contain an OK button that the user can tap to clear the dialog window. Alternatively, you can use a special prefix at the start of the text of the string, to add a series of buttons to the message box dialog. The **result** variable stores a character depending on the button that the user taps.

Prefix included in msgbox string	Labels on Buttons	Response stored in Result Variable
?YNC?	Yes, No, Cancel	Y, N, C
?YN?	Yes, No	Y, N
?OC?	OK, Cancel	O, C
No prefix used	OK	O

You can write a script to take action depending on which button the

handheld user taps in the message box dialog.

In this example, if the answer in field 8 is less than zero, a dialog will appear asking if the user wants to continue. The dialog will have two buttons, Yes and No. If the user taps the Yes button, the user will jump to field 12. If the user taps the No button, he/she will jump to field 20.

```
exitscreen:
  if $8 < 0 then
    msgbox "?YN?Would you like to continue?"

    if result == Y then
      goto 12
    else
      goto 20
    endif
  endif

endif
```

nullfrom *field-number1* to *field-number2*

Allows a range of fields to be set to NULL.

Field-number1 is the first field in the range and *field-number2* is the last field in the range. A field name can be an actual field number, such as 5, or a field label such as [AccountName].

In this example, if the user selects Y in the current field, the record is cloned and fields 5 to 20 are set to null.

```
select:
  if answer == Y then
    clone
    nullfrom 5 to 20
  endif
```

The words NEXTFIELD and LASTFIELD can be used instead of field numbers. NEXTFIELD means the field immediately after the field containing the script. LASTFIELD means the last field on the form. See example on next page.

In this example, if the user selects to erase information, then fields will be erased starting from the field with the field label [CustomerName] and ending with the last field on the form.

```
exitscreen:
if $12 == "Erase" then
  msgbox "?YN?Are you sure you want to erase
  this info?"

  if result == Y then
    nullfrom [CustomerName] to LASTFIELD
  endif
endif
```

optional *field-number*

optional from *field-number to field-number*

Makes the specified field optional. Used to override setting the field as Required in the Advanced Field Properties screen. (Note that all fields are optional by default, unless made Required either by setting the Advanced Field Property of Required (page 189), or by making the field Required in a script - page 312.)

In this example, if the answer in the current field is Y, then field 7 is optional.

```
exitscreen:
if answer == Y then
  optional 7
endif
```

In this example, if the response in field 8 is “Never”, fields 9 to 15 are optional.

```
exitscreen:
if $8 == "Never" then
  optional from 9 to 15
endif
```

prinhtml *value*

New in Forms 5.1.

Applies only if using PrintBoy 7 software from Bachmann Software.

Used in report: and report2: events to specify the HTML layout of the report to be printed by the PrintBoy HTML engine.

See page 408 for more information, and also refer to the documentation for Bachmann PrintBoy to see what HTML tags are allowed.

The *value* parameter can be:

a) A text string in double quotes containing HTML tags. For example:

```
prinhtml "<P>SummaryReport</P>"
```

If an HTML text string requires double quotes, the double quotation marks within the string have to be preceded by a backslash (\).

For example:

```
prinhtml "<P ALIGN=\\\"CENTER\\\">SummaryReport</P>"
```

b) The value in a field. For example, to print the value in Field 5:

```
prinhtml $5
```

A **prinhtml** statement cannot contain both a text string and a variable, so several prinhtml statements need to be used if you are placing a variable in the middle of text. For example, if field 5 is being placed in a table cell, the prinhtml statements might look like:

```
prinhtml "<TD>"  
prinhtml $5  
prinhtml "</TD>"
```

c) A variable. For example:

```
prinhtml result
```

See page 408 for an example that shows printing using prinhtml statements. Refer to the PrintBoy documentation for details on the HTML statements supported by PrintBoy.

printleft *value column*

Used in **report:** and **report2:** events.

Positions the text contained in the *value* parameter as left-justified, starting from the specified *column*. Note that a **printline** statement is required after the **printleft** statement to send the text to the handheld's memory.

The number of columns per page depends on your printer specifications. A printer printing to 8.5" x 11" paper typically prints 80 characters (also called 80 columns) per line, and 54 lines per page.

This example prints the phrase "Chopper Rescue Corp." on the current line, starting in column 20:

```
report:
  printleft "Chopper Rescue Corp." 20
  printline
```

In this example, the phrase "Customer Name:" is printed starting in column 1, followed by the contents of Field 7.

Since "Customer Name:" takes up 14 characters, in order to leave a space after the colon symbol, Field 7 is printed starting in column 16.

```
report:
  printleft "Customer Name:" 1
  printleft $7 16
  printline
```

printline

Used in **report:** and **report2:** events.

Commits the current line to the end of the current report in the handheld's memory. A **printline** statement is needed to actually print a line of text. Typically a **printline** statement follows a **printleft** or **printright** statement.

To obtain a blank line, use two **printline** statements in a row. For example:

```
report:
  printleft "Invoice Number:" 10
  printleft $1 16
  printline
  printline
```

printmemo value “*startcolumn:endcolumn:maxlines*”

Used in **report:** and **report2:** events.

Allows a Text field to word wrap in a block of text. If a printmemo statement is not used on a Text field, the first line of text will be printed, and then the remainder of the field will be truncated.

The **startcolumn** and **endcolumn** specifiers indicate the column in which to start printing the text, and the column in which to end the block of text. The number of columns per page depends on your printer specifications. A printer printing to 8.5" x 11" paper typically prints 80 characters (also called 80 columns) per line, and 54 lines per page.

The **maxlines** specifier indicates the maximum number of lines to print in the block of text.

A **printline** statement after the printmemo statement is required to actually print the block of text.

This script prints Field 8 in a text block that starts in column 5, ends in column 75, and prints up to 6 lines of text.

```
report:
  printmemo $8 "5:75:6"
  printline
```

printnewpage

Used in **report:** and **report2:** events.

Sends the current report to the printer, then begins a new page.

To know when to begin a new page, you can use the **pagenum** and **linenum** functions. Pagenum retrieves the current page number, and linenum retrieves the current line being printed on the current page.

The number of lines per page depends on your printer specifications. A printer printing to 8.5" x 11" paper typically prints 80 characters (also called 80 columns) per line, and 54 lines per page.

In this example a new page is started if the report has reached line 50 of the current page.

```
report:
  if linenum = 50 then
    printnewpage
  endif
```

printreport *name* {**serial** | **IR** }

Used in a **click:** event.

Printreport is an alternative to using the Print statement, and allows you to customize print output from the handheld. If you are using PalmPrint software, you will need to use the printreport statement.

A Printreport statement triggers the **report:** and **report2:** events on a form. The report: and report2: events specify the printer output.

The *name* specified is stored in the **reportname** function, and gives you the capability of printing different types of reports from the same form. If you are just printing one report, you can use the word VIA as the report name.

The serial or IR specifier tells the handheld whether to print via the serial port or via the infrared port.

In this example, clicking a button prints the current record via infrared. The Printreport statement causes the report: events on the form to be run in sequence, meaning that a report event in Field 1 runs first, then a report: event in Field 2, etc.

```
click:
  printreport via IR
```

printreport via {*\$field-number* | *[\$field-label]* | "*e-mail address*"}**PALM OS**

On Palm OS devices running VersaMail software, the printreport statement can be used in a **click:** event to send a report to an e-mail. The recipient's e-mail address is referenced in the printreport statement by the field number or field label of a field containing the e-mail address, or by directly typing the e-mail address into the script.

The layout of the report has to be specified in a **report:** event.

The Result variable can be used to add a subject line to the e-mail and optionally to include cc: and bcc: addresses. If the Result variable begins with "subject:" then the text in the Result variable will be added

to the subject line of the VersaMail e-mail. The format for the Result variable is:

subject:*subjectline*;**cc:***cc_address*;**bcc:***bcc_address*

In this example, an e-mail report is given a subject line of *Here is my report*, and the e-mail is copied to JohnSmith@mycompany.com. The address of the main e-mail recipient is stored in a field with the field label Customer.

```
click:
  result = "subject:Here is my report;"
  result = result & "cc:JohnSmith@mycompany.com"
  printreport via ${Customer}
```

Tapping a Button field with this script will run the VersaMail application on the Palm OS device, and create a new e-mail containing the report that was specified in a report: script elsewhere on the form. The handheld user can then use VersaMail to send the e-mail.

For another example, see page 441.

printright *value column*

Used in **report:** and **report2:** events.

Positions the text contained in the *value* parameter as right-justified, ending at the specified *column*. Note that a **printline** statement is required to send the text to the handheld's memory.

The number of columns per page depends on your printer specifications. A printer printing to 8.5" x 11" paper typically prints 80 characters (also called 80 columns) per line, and 54 lines per page.

Printright is useful if you need print lines of currency values, and you need to line up the decimal points.

In this example, fields 9 and 10 are right-justified against column 70 on two lines of a report:

```
report:
  printright $9 70
  printline
  printright $10 70
  printline
```

queue { all | changed }

PALM OS

Typically used in a **click:** event.

Used with **select** statements to put the selected records in a queue, ready to be beamed to another handheld. A **beam** statement is then used to beam the queued records.

The **all** specifier means that all of the selected records will be queued to be beamed.

The **changed** specifier means that of the selected records, only those which are new or modified records will be queued to be beamed.

Several sequences of select statements and queue statements can be used with one beam statement.

In this example, all records of a form called Customers are selected, but only the new or modified records are queued to be beamed. The beaming is performed via infrared (IR).

```
click:
  select all "Customers"
  queue changed
  beam "Customer data" IR
```

readonly from *field-number* to *field-number*

Used to make a field or a range of fields read-only.

Instead of using a script, you can also make a field read-only in the Form Designer window (see page 179).

In this example, if the value in field 15 is greater than 100, field 16 and fields 20 - 25 are made read-only.

```
exit:
  if $15 > 100 then
    readonly 16
    readonly from 20 to 25
  endif
```

readwrite *field-number***readwrite from *field-number* to *field-number***

Used to make a read-only field updatable.

In this example, field 27 is made writable:

```
enter:
  readwrite 27
```

require *field-number*

require from *field-number to field-number*

Makes the specified field(s) required, meaning that the field(s) cannot be left blank.

Instead of using a script, you can also make a field required in the Form Designer window (see page 189).

In this example, if the response in the current field is Y, then field 45 is a required field:

```
select:
  if answer == Y then
    require 45
  endif
```

In this example, if the response in the current field is “Critical”, then fields 18 to 22 are required.

```
exitsscreen:
  if answer == "Critical" then
    require from 18 to 22
  endif
```

right *value length*

Extracts the right-most characters and places the extracted characters into the Result variable.

The **value** specifier is the value from which the right-most characters are to be extracted.

The **length** specifier is the number of characters to be extracted.

In this example, the value “129” is placed into the current field.

```
enterscreen:
  right "Procedure Code 129" 3
  answer = result
```

Here the last five characters from field 35 are placed into the current field.

```
exitsscreen:
  right $35 5
  answer = result
```

return

Returns from the current script without executing any more instructions. Useful if you want to prevent default instructions from being executed after an IF statement.

In this example, a field with the field label Quantity is to be multiplied by a field labeled Price, and stored in a field labeled Total. However, if the Quantity field is left blank, a message is displayed to the user and the return statement stops the script before the multiplication takes place.

```
calculate:
  if ${Quantity} is null then
    msgbox "Please fill in the quantity"
    return
  endif
  ${Total} = ${Price} * ${Quantity}
```

reverselookup *value within lookup-list-name*

Reverselookup...Within is the opposite of Lookup...Within. Reverselookup takes the lookup value of a Lookup List and returns the corresponding lookup display item, and places the result in the Result variable.

review *"formname"*

Used primarily in a **click:** event or in a **scan:** event. Used to go to the Review screen for the specified form, so that the user can select and edit a record.

The review statement can be used in conjunction with making your own custom 'Main Menu' that users see when they launch the Forms application. See Chapter 21, *Creating a Custom Main Menu*, page 471.

The review statement can be used with select statements. A **select where** statement can select records based on a certain criteria, and then the review statement allows the user to see and edit the filtered records. See page 484.

In this example, tapping a picture in a Section field that is part of a custom 'Main Menu' displays a list of records in a form called Customer Log.

```
click:
  review "Customer Log"
```

scanner *conversion-code* { **enable** | **disable** }

Enables or disables some barcode conversions. Conversions are used, for example, to convert a shortened 7-character UPCE code to a standard 10-character UPCA code.

Sample script to allow conversion from UPCE to UPCA:

```
enterscreen:  
  scanner UPCETOUPCA enable
```

Valid conversion codes are:

UPCETOUPCA	EAN8TOEAN13	CODE39TOCODE32
UPCE1TOUPCA	I2OF5TOEAN13	

scanner { **enable** | **disable** } **all**

Enables or disables all barcode symbologies. Enabling all bar code symbologies means that any bar code can be scanned into a Text field. Disabling all symbologies means that no bar codes can be scanned into any Text fields.

Important: Since scanner settings are global across all fields, if you disable all symbologies in one script and forget to enable any, you will not be able to scan into any field on the form.

A sample script to enable all symbologies is:

```
open:  
  scanner enable all
```

An **open**: event is used in the example above so that the scanner is enabled on both new records and existing records.

scanner { **enable** | **disable** } *symbology*

Switches on or off specific types of bar codes which can be scanned.

Sample script to switch ON the ability to scan the UPCA symbology:

```
open:
  scanner enable UPCA
```

Sample script to switch OFF the ability to scan the EAN13 symbology:

```
open:
  scanner disable EAN13
```

Valid symbologies are:

CODE39	I2OF5	MSI_PLESSEY
UPCA	CODABAR	UPCE1
UPCE	CODE128	
EAN13	CODE93	BOOKLAND_EAN
EAN8	TRIOPTIC39	ISBT128
D2OF5	UCC_EAN128	COUPON

scanner {**system_character** | **no_preamble** | **system_character_country_code**} *symbology*

scanner {**checkdigit** | **no_checkdigit**} *symbology*

Switches on or off whether leading/prefix or trailing/suffix barcode digits are recorded.

A **system_character** or **preamble** is the leading digit of the bar code, and a **checkdigit** is the trailing character of the bar code.

Sample script to switch ON the leading character of UPCA bar codes:

```
open:
scanner system_character UPCA
```

Sample script to switch OFF the leading character of UPCA bar codes:

```
open:
scanner no_preamble UPCA
```

Symbologies which use leading characters (**system_character** or **preamble**) are:

UPCA UPCE UPCE1

Sample script to switch ON the trailing character of UPCA bar codes:

```
open:
scanner checkdigit UPCA
```

Sample script to switch OFF the trailing character of UPCE bar codes:

```
open:
scanner no_checkdigit UPCE
```

Symbologies which use **checkdigits** are:

UPCA	CODE39
UPCE	I2OF5
UPCE1	MSI_PLESSEY

scanner {*verify* | *no_verify*} *symbology*

scanner {*opcc_checkdigit* | *uss_checkdigit*} **I2OF5**

CODE39 and I2OF5 bar code symbologies may or may not use the last character of the bar code as a check digit. If verification of the check digit is switched on, then the scanner will assume that the final character of the barcode is a checkdigit, which means that if you scan a barcode without a checkdigit, the scan will probably fail. If you switch verification off, then the scanner will accept barcodes both with and without a checkdigit. Note that when verification is off, barcodes with a checkdigit are not checked, just accepted.

I2OF5 barcodes also have two types of checkdigits, OPCC or USS checkdigits, and you can choose which type of checkdigit the scanner will allow.

Sample script to switch OFF verification of CODE39 bar codes:

```
open:
  scanner no_verify CODE39
```

Symbologies with which scanner verification is valid:

```
CODE39          I2OF5
```

select all "*formname*"

Selects all records in the specified form. Might be used to undo a **select matching** statement or a **select where** statement, or to show all records in a reference form if a matching criteria was not specified. Can also be used with **queue** and **beam** statements to select records to be beamed.

The *formname* is the name of a reference form.

If you are performing a lookup to another form, the current form must have a Lookup List field that references the name of the reference form. If you want to copy records from the reference form into the current form automatically, the field names and field types have to match in both the reference form and the current form.

In this example, the script is written in a Lookup List field that references a reference form. If field 5 of the current form is null (blank), then the **select all** statement selects all records of the Parts List reference form. If field 5 of the current form is filled in,

then the **select where** statement selects only those records that match field 2 of the Parts List form to field 5 of the current form.

```
click:
  if $5 = null then
    select all "Parts List"
  else
    select "Parts List" where field 2 is $5
endif
```

In this example, a select all statement is used to select all the records from a form called Customer Info. The queue changed statement queues only the new or modified records to be beamed. The beam statement does the beaming.

```
click:
  select all "Customer Info"
  queue changed
  beam "Customer Info" IR
```

select current

Selects the current record in the current form.

Used primarily in a **click:** event, and in conjunction with **queue** and **beam** statements, to beam the current record to another handheld. The receiving handheld must already have Pendragon Forms and the particular form design installed on the handheld.

In this example, the current record is selected, queued for beaming, and then beamed.

```
click:
  select current
  queue all
  beam "My record" IR
```

In the above example, using a **queue all** statement means that the record will be beamed even if there are no changes to the record. The rules that determine whether the receiving handheld will accept a record that has not been changed are set in Advanced Form Properties. See page 223-224 for more information.

To beam a record only if it is new or has been modified, the **queue changed** statement can be used instead.

select matching *formname*

Used to perform cascading updates or deletes from a parent form to a subform.

Can also be used with **queue** and **beam** statements to beam subform records. If you beam subform records, you should also beam the parent record.

Selects all records in the subform called *formname* that match the current record of the current (parent) form. The first 10 fields of the current form are used, and the field names and values in those fields must match in the current form and the subform for a record to be selected. The Result variable stores the number of matching records.

Once selected, the subform records can be updated with the **update field** statement. Alternatively, the selected subform records can be deleted with the **delete** statement.

WARNING: If you change the values in any fields that are used to match the parent record to the subform records, you will lose the link from parent form to subform and the parent form will appear not to have any subform records.

In this example, a parent form has a Yes/No checkbox field with a field name such as “Do you want to delete this equipment log and all related equipment readings?”. Choosing Yes runs a select: script to delete all the corresponding records on the “Equipment Readings” subform.

```
select:
  if answer == Y then
    select matching "Equipment Readings"
    delete
  endif
```

NOTE: This type of subform record deletion will only work in the case where you are not storing records on the handheld after synchronization. If you are choosing to keep records on the handheld, deleting subform records only works if the record is a newly created record that has not been backed up to the PC. If records have been backed up to the PC, then those records will be re-sent to the handheld during the next synchronization. The solution in this case is to either use a Completion Checkbox field within the first 10 fields of the parent and subform, or to have a field for storing the status of the field (e.g.: Active or Deleted), and then use Additional Download Criteria (see page 214) to send only Active records to the handheld.

select *formname* where field *field-number* is *expression*

The **select where** statement is used to select records from a reference form. This has the effect of performing a filter on the reference form, and is useful if you want to limit the list of records that the handheld user sees when performing a lookup to another form. Only one form can be selected at a time.

A select where statement can also be used with **queue** and **beam** statements to select the records that you want to beam to another handheld.

The *formname* is the name of the reference form.
The *field-number* is a field number (or column number) in the reference form.

The *expression* is the criteria for finding a matching record in the reference form. The expression can be:

A text string (e.g.: "Cherry Street"). For example:
select "Addresses" where field 4 is "Cherry Street"

A numeric value (e.g. 15). For example:
select "Inventory" where field 3 is 15

A value in a field (e.g. \$8 means the value in field 8). For example:
select "Parts List" where field 1 is \$8

A function that takes no arguments (e.g. scandata). For example:
select "Bar Code Inventory" where field 1 is scandata

The *expression* in a select where statement can also contain a range, such as:

Greater than (> numeric comparison). For example:
select "Student Scores" where field 6 > 90

Less than (< numeric comparison). For example:
select "Sale Items" where field 2 < 995
(Note that if field 2 is a Currency field, 995 means \$9.95)

Equal (= numeric comparison). For example:
select "Items to Re-Stock" where field 5 = 0

Contains a character sequence (e.g.: contains "Ave"). For example:
`select "Addresses" where field 2 contains "Ave"`

Starts with a character sequence (e.g.: startswith "Tr"). For example:
`select "Countries" where field 1 startswith "Tr"`

The number of records that match the select where criteria is stored in the Result variable.

The **also** statement can be used to combine selection criteria for the same form.

If a lookup to another form is being performed, then one field on the current form has to be a Lookup List field type that references the name of the reference form. A **click**: event script in the Lookup List field causes the select where statement to be run when the handheld user taps in the Lookup List field, so that the user sees the filtered list of records. Fields that are named the same on both the reference form and the current form will automatically copy from the reference form into the current form when the user selects a record from the list.

In this example, the user enters a value in field 2 of the current form. Tapping in a Lookup List displays all the records in the Customers reference form, where field 4 of the reference form matches field 2 of the current form. When the user taps on a record, all the fields named the same in both forms are copied from the reference form into the current form.

```
click:
  select "Customers" where field 4 is $2
```

If you are looking up records automatically via a script, fields can be copied from the reference form to the current form using the **column** function, or fields on the reference form can be updated or deleted.

In this example, the barcode on a student badge is scanned into field 2 of the current form. The select statement selects the record where field 1 in a form called StudentID matches field 2 in the current form. After the match is found, the value in field 3 of StudentID is copied into field 4 of the current form.

```
scan:
  $2 = scandata
  select "StudentID" where field 1 is $2
  $4 = column 3
```

selectfile *fieldnumber* { **copy** | **move** | **capture**}

Used in a **click**: event to allow the user to attach an image from an external media card (e.g.: SD card or Memory Stick card) or from the memory of the device to the current record. This enables the user to attach images from cameras (or other binary files) and have them uploaded to the PC.

The **selectfile** statement opens a file browsing screen that enables the user to view files on the storage card. After the user selects a file on the card, the file is either copied or moved to a folder on the card that is reserved for Pendragon Forms attachments.

The form to which you are attaching the image must have a Button field for running the **selectfile** statement in a **click**: event. The form must also have an Image field.

The *fieldnumber* parameter is the field number of the Image field that will hold the attached image.

The **copy** directive copies the selected file to the FORMS5 folder on the storage card, and renames the copy. The **copy** directive is the safest method to use, because in the event that the file attachment is deleted from within Pendragon Forms, the original file still exists in the original folder on the storage card. See pages 426-427 for an example.

The **move** directive only applies to Palm OS devices. The **move** directive moves the selected file from its original location on the storage card to the FORMS4 folder, and renames the file.

For example, if the user selects the file /CAMERA/IMG0001.JPG, the file is first copied to /FORMS5/0D1234560000234B0004.JPG. The filename 0D1234560000234B0004 represents the form ID number, the record ID number and the field number. If the **copy** directive was used, the original /CAMERA/IMG0001.JPG file still exists. If the **move** directive was used, the original /CAMERA/IMG0001.JPG file is deleted from the card.

New in Forms 5.1, the **capture** directive allows the handheld user to take a picture with a handheld camera and attach it to a record. This is available on Palm OS devices running Palm OS 5.2.8 or higher (Zire 72 or Treo devices), or Windows Mobile 5 Pocket PC devices. See pages 430-432 for an example.

In this example, clicking a button allows the user to select an image from the storage card to attach to field 4 of the current record. (Field 4 has to be an Image field on the form.)

```
click:
  selectfile 4 copy
```

setlookuplocale *value*

Setlookuplocale is used to set the **lookuplocale** variable to the specified *value*. This is used to create a cascading lookup list, that is, a lookup list in which the list that is displayed depends on a selection in another field on the form. (Note: You can also use **setlookupname** with the variable **lookupname** to do a cascading lookup.)

The *value* can be the value in a field, such as \$3, or a constant value such as "Car" or "Plane".

setlookuplocale is usually run in a click: event script in a Lookup List field. The value that the **lookuplocale** variable is set to, is then used to help determine which Lookup List to display.

In the form design, the name of the Lookup List is a keyword followed by the caret (^) character. For example, if the keyword is Transport, then the Lookup List to display is written as Transport^. When the user taps in the Lookup List field, the ^ character is replaced by the value in the **lookuplocale** variable. So if **lookuplocale** has been set to "Car", the Lookup List that will be displayed is called TransportCar.

In this example, the user makes a selection in field 8 of the form, and that value is assigned to the **lookuplocale** variable in the following script in a Lookup List field:

```
click:
  setlookuplocale $8
```

See page 370 for an example of using **setlookuplocale**.

setlookupname *value* Setlookupname is used to set the **lookupname** variable to the specified *value*. This is used to create a cascading lookup list, that is, a lookup list in which the list that is displayed depends on a selection in another field on the form. (Note: You can also use **setlookuplocale** with the variable **lookuplocale** to do a cascading lookup.)

The *value* can be the value in a field, such as \$4, or a constant value such as "Male" or "Female".

setlookupname is usually run in a click: event script in a Lookup List field. The value that the **lookupname** variable is set to, is then used to help determine which Lookup List to display.

In the form design, the name of the Lookup List is a keyword followed by the tilde (~) character. For example, if the keyword is Adult, then the Lookup List to display is written as Adult~. When the user taps in the Lookup List field, the ~ character is replaced by the value in the **lookupname** variable. So if **lookupname** has been set to "Female", the Lookup List that will be displayed is called AdultFemale.

In this example, the user makes a selection in field 12 of the form, and that value is assigned to the **lookupname** variable in the following script in a Lookup List field:

```
click:
  setlookupname $12
```

See page 370 for an example of using **setlookupname**.

setparam allowhtml {0 | 1}

New in Forms 5.1. This command is only available if you are using PrintBoy printer driver software (version 7) from Bachmann Software.

The **setparam allowhtml** statement determines whether the layout of a report specified in a report: or report 2: event is interpreted as HTML or not. The default setting is **setparam allowhtml 0** which means that statements in a report: event are not considered to be HTML.

If you are using PrintBoy software for printing, and you want to use HTML tags in report: and report2: events instead of formatting your report based on the rows and columns of the report, then you will need to switch on **setparam allowhtml 1**.

A click: event script is typically used to switch on the use of HTML, and the **setparam allowhtml 1** statement must come **before** the printreport statement in the script.

```
click:
  setparam allowhtml 1
  printreport via IR
```

setparam modified {0 | 1}

New in Forms 5.1. This command sets the change flag for the current record. Setting this parameter to 0 marks the record as unmodified, while setting it to 1 marks the record as modified or changed.

Important: An unmodified record does not save any changes back to the desktop/server during synchronization, and so extreme care should be taken when using this command.

You might want to set records as unmodified on a form that contains a calculator feature in which the user fills in some fields to perform a calculation, but does not need to keep any of the data entered or generated. By setting such a record as unmodified, the user does not keep generating new records every time they use the calculator.

A validate: event script is typically used to set a record as unmodified. You can also use exitscreen: and select: events, basically setting the record as unmodified each time the user changes a field.

```
validate:
  setparam modified 0
```

setparam printhtmlheader "*header*"

setparam printhtmlfooter "*footer*"

New in Forms 5.1. These commands are only available if you are using PrintBoy printer driver software (version 7) from Bachmann Software.

When printing via PrintBoy, Pendragon Forms generates HTML source code from the statements in a report: event. The HTML document that is generated is then printed by the PrintBoy HTML engine. By default, the header of this HTML document just sets the HTML to preformatted mode, which is:

```
<HTML><BODY><PRE>
```

By default, the matching footer closes these tags:

```
</PRE></BODY></HTML>
```

The preformatted mode is a fixed-width mode (e.g. *Courier* font). If you had report: events that you used in Forms 5.0, keep the default preformatted mode to avoid having to modify your report: event scripts in Forms 5.1.

You can choose to override the default HTML header and footer settings with the **setparam printhtmlheader** *header* and **setparam printhtmlfooter** *footer* statements. The *header* and *footer* parameters are HTML headers and footers that are supported by the PrintBoy software. For example, to print in bold, you can do this:

```
setparam printhtmlheader "<HTML><BODY><B>"
setparam printhtmlfooter "</B></BODY></HTML>"
```

Any double quotation marks within the HTML text have to be preceded by an escape character - a backslash \ character. For example, to set an HTML header that sets the font to "Arial", the commands would be:

```
setparam printhtmlheader "<HTML><BODY><FONT FACE='Arial'>"
setparam printhtmlfooter "</FONT></BODY></HTML>"
```

A click: event script is typically used to trigger printing. The HTML headers and footers have to be specified before the printreport statement runs, as does a **setparam allowhtml 1** statement. This script causes the entire report to be printed in bold.

```
click:
setparam printhtmlheader "<HTML><BODY><B>"
setparam printhtmlfooter "</B></BODY></HTML>"
setparam allowhtml 1
printreport via IR
```

See page 408 for more information on printing with PrintBoy.

setparam serialport "*portnumber*"

New in Forms 5.1. This command sets the serial port for future calls to **acquire** and **transmit serial** statements. This enables you to acquire data via a different port such as a Bluetooth port or an infrared port.

The *portnumber* can be numeric or a special four-character code:

32771 - this is the default port number which is the serial port.

rftm - this is a Bluetooth serial port.

ircm - this is an infrared (IrComm) serial port.

The letters rftm or ircm must be lower case.

An open: event is typically used to set the port that will be used:

```
open:  
  setparam serialport "rftm"
```

setview "*formname*|*field-number:percent*|*field-number:percent*|*field-number:percent*"

Setview allows you to customize the Review screen for forms that are accessed via another form. For example, if the handheld user reviews records via a Custom Main Menu, or goes to a subform and views subform records, or does a lookup to a regular reference form (**not** a read-only reference), the setview statement can be used to display the specified fields at the specified percentage of the screen width on the Review screen that the user sees.

For an example, see page 368.

NOTE: In order to set a default view of the Review screen for forms that the user accesses from the Forms Main Menu (Forms List), set an Advanced Form Property - see page 226.

In the script shown here, a Section field of a Custom Main Menu form allows the user to review the records of the form "List of Customers", displaying field 1 occupying 50 percent of the screen width, field 2 occupying 30 percent of the screen width and field 3 occupying 20 percent of the screen width.

```
click:  
  setview "List of Customers|1:50|2:30|3:20"  
  review
```

You can specify up to 4 fields in the setview statement, and the sum of the percentages cannot exceed 100%.

show *field-number*

show from *field-number to field-number*

Makes a hidden field or range of hidden fields visible.

In this example, if the answer in the current field is N, then field 4 is made visible.

```
select:
  if answer == N then
    show 4
  endif
```

In this example, several hidden fields are displayed.

```
exit:
  if $3 # "More" then
    show from 8 to 30
  endif
```

socketscan defaults on

PALM OS

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E).

Resets the scanner to the default settings.

socketscan *lengthparameter length*

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E)

This statement controls the minimum or maximum length for various barcode types.

The *lengthparameter* must be one of the following:

CODE39_MIN	CODABAR_MIN
CODE39_MAX	CODABAR_MAX
CODE93_MIN	MSI_MIN
CODE93_MAX	MSI_MAX
I2OF5_MIN	CODE11_MIN
I2OF5_MAX	CODE11_MAX
D25_MIN	WEBC_MIN
D25_MAX	WEBC_MAX

The *length* value sets the number of characters that will be accepted for the maximum or minimum of the barcode type you specify.

In this example, acceptable CODE39 barcodes are set to a minimum length of 10 characters. This means that a CODE39 barcode with less than 10 characters will not be scanned.

```
open:
  socketscan code39_min 10
```

socketscan set_trigger *keycode*

PALM OS

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E) bar code readers.

This statement sets the hardware button keycode that will trigger the scanner.

It is suggested that you use one of the following keycodes:

- 516 for the DateBook/Calendar button.
- 517 for the AddressBook/Contacts button.
- 518 for the ToDo/Tasks button.
- 519 for the Memo/Notepad button.

The socketscan set_trigger command would typically be used in an open: event script, such as:

```
open:
  socketscan set_trigger 519
```

In this example, the handheld user would then be able to press the Memo/Notepad button on the handheld to scan a barcode.

socketscan *setting value*

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E) bar code readers.

This statement allows you to set parameters such as the transmission and verification of check digits. The *setting* and *value* parameters can be set according to the following table. Refer to the SocketScan SDK for Palm OS if you need more detailed information.

Setting	Value
DECODE_UPCEAN_SUPPLEMENTALS	0 = UPC/EAN without Supplemental Characters 1 = UPC/EAN with Supplemental Characters 2 = Autodiscriminate
UPCEAN_SUPPLEMENTAL_REDUNDANCY	This value ranges from 2 to 20. 5 or above is recommended when decoding a mix of UPC/EAN symbols with and without supplementals.
TRANSMIT_UPCA_CHECKDIGIT TRANSMIT_UPCE_CHECKDIGIT TRANSMIT_UPCE1_CHECKDIGIT CONVERTUPC_ETOA CONVERTUPC_E1TOA EAN8_ZEROEXTEND CONVERT_CODE39TOCODE32 CODE32_PREFIX CODE39_CHECKDIGITVERIFY TRANSMIT_CODE39_CHECKDIGIT TRANSMIT_I2OF5_CHECKDIGIT CONVERT_I2OF5TOEAN13 TRANSMIT_MSIPL_CHECKDIGIT	0 = Disable 1 = Enable

Setting	Value
UPCA_PREAMBLE UPCE_PREAMBLE UPCE1_PREAMBLE	0 = No preamble transmitted 1 = Transmit system character only 2 = Transmit system character and country code ("0" for USA)
CONVERTEAN_8TO13	0 = Type is EAN-8 1 = Type is EAN-13
UPCEAN_SECURITYLEVEL	See the SocketScan SDK
I2OF5_CHECKDIGITVERIFY	0 = Disable 1 = USS Check Digit 2 = OPCC Check Digit
MSIPL_CHECKDIGIT	0 = Two Check Digits 1 = 1 Check Digit
MSIPL_CHECKDIGIT_ALGORITHM	0 = MOD10/MOD 11 1 = MOD 10/MOD 10
CODE_ID_CHAR A code ID character identifies the bar code type of a scanned bar code. This may be useful when the scanner is decoding more than one code type. In addition to any single character prefix already selected, the code ID character is inserted between the prefix and the decoded symbol.	0 = No character 1 = Symbol ID code, one of the following is inserted: A = UPC-A, UPC-E, UPC-E1, EAN-8, EAN-13 B = Code 39, Code 32 C = Codabar D = Code 128, ISBT 128 E = Code 93 F = Interleaved 2 of 5 G = Discrete 2 of 5 or Discrete 2 of 5 IATA H = Code 11 J = MSI Plessey K = UCC/EAN-128 L = Bookland EAN M = Trioptic Code 39 N = Coupon Code

socketscan *symbology value*

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E) bar code readers.

This statement allows you to enable or disable the scanning of certain bar code symbologies, e.g. ISBN, CODE39, UPCA, Etc.

The *value* parameter is set to **1** to enable the specified symbology, or set to **0** to disable it.

Valid symbology values are:

UPCA	TRIOPTIC39
UPCE	CODE93
EAN8	I2OF5
EAN13	D25
BOOKLAND_EAN	CODABAR
USS128	MSI_PLESSEY
UCC_EAN128	CODE11
CODE39	WEBC

You would typically enable a symbology in an open: event script, for example:

```
open:  
socketscan UPCA 1
```

socketscan trigger now

This scripting statement only applies to Socket Communications Secure Digital Scan Card 3E (SDSC 3E) bar code readers.

This statement turns on the scanner to search for a bar code. This can be used to trigger a scan using an on-screen Button field.

The script in the Button field might be:

```
click:  
socketscan trigger now
```

NOTE: You will also need a scan: event script in your form to determine what happens after the scanner scans the bar code.

subformsum “*name-of-subform*” *field-number*

Used to add all values in a specific field across all subform records associated with a given parent record. The Subformsum statement is placed in a script on the parent form, typically a **click:** event. The calculated sum is placed in the Result variable.

The *name-of-subform* is the name of the subform.
The *field-number* is the field on the subform which is to be added across all subform records for that parent.

In this example, clicking a button on the parent form adds up the values in Field 6 of the Items Ordered subform. The calculated result is placed in Field 12 of the parent record.

```
click:
  subformsum "Item Ordered" 6
  $12 = result
```

synchronize

The synchronize statement is typically placed in a **click:** event script in a Section field or a Button field on a Custom Main Menu form, to provide users with a menu option for initiating a synchronization.

PALM OS

Runs the Pendragon S-Link program and starts synchronizing Pendragon Forms data. Should only be used on wireless Palm OS devices that are running the S-Link application.

PPC

On Pocket PC devices, this command starts synchronization of Pendragon Forms.

For example:

```
click:
  synchronize
```

transmit multimap "*E-mail address*"

PALM OS

Used in conjunction with additional VersaMail e-mail software from www.palm.com. Creates a text version of the current record and adds it to the e-mail Outbox to be sent to the specified recipient when next the handheld user sends e-mail. Use in **click:** events only.

In this example, the current record is e-mailed to `sales@xyz.com`.

```
click:  
  transmit multimap "sales@xyz.com"
```

In this example, the current record is e-mailed to the address that was entered in Field 7.

```
click:  
  transmit multimap $7
```

transmit palmnet “*address*”

PALM OS

Warning: This option can only transfer small amounts of data, typically less than 500 bytes of data.

Used in conjunction with Palm i705.

The transmit palmnet statement asks the Palm i705 Web clipping application to display the Web page at the specified URL.

NOTE: Requires familiarity with Web and Internet concepts.

By choosing the address parameter appropriately, it is possible to send the contents of the current record to a Web site. Depending on the way the Web site software is written, the returned Web page could contain the results of a search, or simply save the record on the Web server.

Note that Pendragon Forms does not interact with the data that is returned by the server. For example, if the handheld user sent a request for information on a particular item, and received a Web page containing the response “Item out of stock”, then it would be the handheld user’s responsibility to use the information received when filling in the Pendragon form. Information from the server would not be placed into the form automatically.

Option 1: If the address parameter is a simple address, such as “<http://www.ourcompany.com>” the Palm VII or Palm i705 will display the contents of the Web page at <http://www.ourcompany.com>.

Option 2: If the address parameter contains a question mark and ends with an ampersand, Pendragon Forms will append the data in all fields in the current record to the URL before sending. The data is written using the standard “Quoted Printable” format. The data is appended in the following format:

FID=<formID>&TS=<timestamp>&Q1=<field value>&Q2=<field value>...

FID stands for “Form ID” and is followed by the form’s ID number. TS stands for TimeStamp, and is the creation date of the record provided in seconds since 01/01/1904. See next page for an example.

Example: Imagine that your form contains two fields: Field 1 contains a Customer Name which you fill in as John Smith, and Field 2 contains the customer's zip code, which you fill in as 90125.

If your address parameter in the Transmit Palmnet statement is:
"http://www.site.com/cgi-bin/getinfo.cgi?check=1&"

then the actual URL that would be sent to the Web server is:

```
"http://www.site.com/cgi-bin/  
getinfo.cgi?check=1&FID=12345&TS=32415236&Q1=John+Smith&  
Q2=90125"
```

The page that would be displayed on the Palm VII depends on the way that the getinfo.cgi script was written. For example, the getinfo.cgi program could be programmed to expect Q2 to be a zip code, and return a listing of your company outlets in or near the zip code.

Option 3: If you do not want to send data from all the fields on your form to the Web server, you can specify the value in a given field by using the field number between two \$ signs in the URL. For example, \$\$8\$ means the value in Field 8.

In this example, only Fields 1 and 4 are sent to the Web server:

```
click:  
transmit palmnet "http://www.site.com/cgi-  
bin/testdate.cgi?preferdate=$1$&altdate=$4$"
```

transmit printer *value*

PALM OS

Similar to the **transmit serial** statement, but requires printer driver software to be installed on the handheld. No communications parameters need to be specified. Instead, the printer driver software handles the communications settings. Works with serial and infrared printing. The value specified is a Text string up to 2KB.

In a text string, the value `\\013` represents a carriage return (ASCII 13), and `\\010` represents a line feed (ASCII 10). These values must be in double quotes in order to be interpreted properly by the script compiler on the handheld. You can use the concatenate operator (`&`) to connect two text strings together to transmit to the serial port.

In this example, the Text string “Invoice #” is printed, followed by the value in Field 1. On a new line, the value in Field 2 is printed.

```
click:
temp = "Invoice #" & $1
temp = temp & "\\013\\010"
temp = temp & $2
transmit printer temp
```

transmit serial *value* Uses the **result** variable to specify communications parameters for communicating via the serial port of the handheld device. The specified *value* is the data string that is sent to the serial device.

The serial communications parameters that the result variable needs to contain are the baud rate, the parity, the number of data bits, the number of stop bits and the handshaking method (optional). Hyphens separate each of these parameters.

For example, the following statement sets up the communications parameters as 9600 baud, No parity, 8 data bits, 1 stop bit, hardware handshaking method:

```
result = "9600-N-8-1"
```

Valid values depend on the serial device to which you are connecting the handheld. Typical values are:

Baud rate: Any standard speed up to 115200 baud.

Parity: N (No parity), E (Even parity), O (Odd parity)

Data bits: 7 or 8

Stop bits: 1 or 2

Handshaking (optional parameter; default is hardware handshake):

N (No handshaking), B (hardware handshaking)

In a text string, the value `\013` represents a carriage return (ASCII 13), and `\010` represents a line feed (ASCII 10). These values must be in double quotes in order to be interpreted properly by the script compiler on the handheld. You can use the concatenate operator (`&`) to connect two text strings together to transmit to the serial port. A maximum of 2KB can be transmitted.

In this example, the text string "My Company", followed by a carriage return and line feed, followed by Field 1, another carriage return and line feed and then Field 2 are sent to the serial port:

(**temp** is a scripting variable used for temporary storage.)

```
click:
result = "9600-N-8-1"
temp = "My Company\013\010"
temp = temp & $1
temp = temp & "\013\010"
temp = temp & $2
transmit serial temp
```

transmit web “URL”

Used in conjunction with Palm devices that have a TCP/IP connection, such as a Palm Tungsten C with WiFi connection, or a Palm Treo 650 with high speed data service.

The Transmit Web statement transmits the URL to a Web server and returns all the data contained between <PFDATA> and </PFDATA> tags. The data that is returned by the Web server is accessible via the Webdata function, which may return up to 2KB of data. See page 353.

To send the value of a field to the Web server, you can reference the field number in the URL by using the field number between \$ signs. For instance, \$16\$ means the value in Field 16.

In this example, clicking a button sends the value in Fields 2 and 7 to the Web server.

```
click:
  transmit web "http://www.mycorp.com/cgi-bin/
  premium.cgi?age=$2$&smoke=$7$"
```

update field *number* = *expression*

Updates selected records in a reference form by setting the value of the field *number* to the specified *expression*.

The reference form cannot be a read-only reference form.

Records in the reference form are selected using one of the following statements: **select where**, **select matching**, **select all** or **insert into**.

The *number* is the field number of the reference form.

The *expression* is the value that will be assigned to the specified field. The expression can be:

A text string (e.g.: "Active Customer"). For example:

```
update field 2 = "Active Customer"
```

A numeric value (e.g. 42). For example:

```
update field 4 = 42
```

A value in a field (e.g. \$16 means the value in field 16) on the current form. For example:

```
update field 7 = $16
```

A function that takes no arguments (e.g. scandata). For example:

```
update field 1 = scandata
```

In this example, the current form is used to scan a product barcode in field 1, and the handheld user counts the number of an item in the warehouse, and enters the quantity in stock in field 2. The user then taps a button to update the "Inventory Items" form. A select statement matches the barcode in field 1 with the corresponding record in the Inventory Items form. Field 3 of the Inventory Items form is updated with the quantity that the handheld user entered in field 2 of the current form. If the quantity is less than 3, field 4 of the Inventory Items form is updated to read "Time to re-order".

```
click:
  select "Inventory Items" where field 1 is $1
  update field 3 = $2
  if $2 <= 3 then
    update field 4 = "Time to re-order"
  endif
```

Scripting Errors

Compilation Scripting Errors Can Prevent Distribution of a Form Design

Scripts are compiled when you distribute a form.

This means that if there are compilation errors in your scripts, then when you attempt to distribute a form, you may receive error messages on the PC, and your form will not be queued for distribution to the handheld. The error message that you receive will indicate the field number of the field containing the problem script. You can edit the form and correct the script, and then try re-distributing the form.

Compilation errors mean that a script is invalid in some way. For example, you may have an if-then statement without an endif at the end of the statement.

To reduce scripting compilation errors, you can check each script that you write by clicking the Check Script button on the Script tab of the Form Designer window. That way you can catch any errors field by field, instead of catching the errors after you have designed the whole form.

Runtime Scripting Errors

A Runtime error is a scripting error that occurs on the handheld. If a runtime error occurs, an error message is displayed on the handheld.

An example of a runtime error is a division by zero error. If, for example, you have a script in Field 8 that divides Field 2 by Field 4, and the handheld user enters zero as the value of Field 4, a number divided by zero gives an answer of infinity, which is not a number that can be stored.

If a runtime error occurs, an error message is displayed. Since a script with a calculate: event runs every time a field is updated, it will take a little guesswork to determine the problem. For instance, the script in Field 8 is the problem, but the error occurs after Field 4 is filled in. By looking at the scripts which use Field 4, you can find the script with the problem. (In the case of division by zero, you may want to add a line in your script which tests if the answer is zero, and takes some action to prevent division by zero - see page 351.)

Logical Flow Scripting Errors

Another type of error which can occur in scripting is an error in the logical flow of the script. This type of error occurs if, for example, your script says goto 5 but you meant it to say goto 6, or if the script says \$5 + \$8 when it should be \$5 * \$8. The script will compile and run flawlessly, but it will not do what you had originally intended. The only way to catch this type of error is to go through the form field by field, selecting every possible option in turn, and seeing if the script branches correctly and if all calculated results are correct. You can correct any scripts and re-distribute the form as necessary.

Sometimes it is helpful to switch on field numbers on the handheld when you are debugging scripts - that way you can verify that the field numbers that you are referencing in your script are the correct field numbers on the handheld. To switch on viewing field numbers on the handheld, see page 73.

Using Scientific Functions in Scripts

PALM OS

On Palm OS devices, scientific math functions are available in scripts only if an additional .PRC file, called MATHLIB.PRC, is installed on the handheld. MathLib is not a part of Pendragon Forms, it is a free shared library that can be used by Palm OS handheld devices.

Forms 5.1 automatically installs MathLib on Palm OS devices. MATHLIB.PRC is also available in the MATHLIB.ZIP archive which is installed into the C:\Program Files\FORMS3 folder.

PPC

Pocket PC devices have MATHLIB built-in to the Forms 5.1 application. No additional programs need to be installed.

Scientific functions that are available in scripts if MathLib is installed on the handheld are:

sin *value* Returns the sine of the angle *value*, expressed in radians.

To convert degrees to radians, multiply by $\pi / 180$.

Pi is a constant representing the number π

In this example, an angle in radians is entered into field 5.
The sine of the angle is then placed into field 6.

```
calculate:
$6 = sin $5
```

In this example, an angle in degrees is entered into field 5. The angle is first converted to radians for use with the sin function, and the sine of the angle in radians is placed into field 6.

```
calculate:
result = $5 * pi
result = result / 180
$6 = sin result
```

cos *value* Returns the cosine of the angle *value*, expressed in radians.

In this example, an angle in radians is entered into field 7.
The cosine of the angle is placed in the current field.

```
calculate:
answer = cos $7
```

tan *value* Returns the tangent of the angle *value*, expressed in radians.

asin *value* Returns the arcsine of the number *value*. The resulting angle is expressed in radians.

To convert radians to degrees, multiply by 180 / pi.

Pi is a constant representing the number π .

In this example, clicking a button calculates the arcsine of a number in field 24, and places the result in field 25. The angle in field 25 is expressed in radians, and so the script then converts the radians to degrees and places the angle in degrees in field 26.

```
click:  
$25 = asin $24  
result = $25 * 180  
$26 = result / pi
```

acos *value* Returns the arccosine of the number *value*. The resulting angle is expressed in radians.

To convert radians to degrees, multiply by pi / 180.

Pi is a constant representing the number π .

atan *value* Returns the arctangent of the number *value*. The resulting angle is expressed in radians.

To convert radians to degrees, multiply by pi / 180.

Pi is a constant representing the number π .

sinh *value* Returns the hyperbolic sine of the number *value*.

cosh *value* Returns the hyperbolic cosine of the number *value*.

tanh *value* Returns the hyperbolic tangent of the number *value*.

sqr *value*

Returns the square root of the number *value*.

In this example, a number is entered in field 12. The square root of the number is placed in the current field.

```
calculate:  
answer = sqr $12
```

value1 pow value2

Pow is a binary operator that calculates the number *value1* raised to the power of *value2*.

For example, the answer in this script would be $10 * 10 * 10 = 1000$.

```
calculate:  
answer = 10 pow 3
```

In this example, a number in field 4 is squared.

```
calculate:  
answer = $4 pow 2
```

In this example, a number in field 6 is raised to the power specified in field 7, and the calculated result is placed in field 8.

```
click:  
$8 = $6 pow $7
```

exp *value*

Exp calculates the number *e* raised to the power *value*.

e is an irrational number that begins with: $e = 2.7182818284590...$

In this example, *e* is squared.

```
calculate:  
answer = exp 2
```

In this example, *e* is calculated to the power specified in field 3. The calculated answer is placed in field 4.

```
calculate:  
$4 = exp $3
```

log *value*

Returns the log base *e* of the number *value*.

In this example, the current field calculates the log base *e* of the number in field 5.

```
calculate:  
answer = log $5
```

log10 *value*

Returns the log base 10 of the number *value*.

In this example, the log base 10 of the number in field 3 is placed into field 4.

```
calculate:  
$4 = log10 $3
```

round *value*

Rounds the number *value* to the nearest whole number.

For example, 20.5 rounds up to the nearest integer, namely 21.

20.49 rounds down to 20.

In this example, field 7 is rounded to the nearest whole number, and the answer is placed in field 9.

```
calculate:  
$9 = round $7
```

In this example, fields 10 and 11 are multiplied together, and then rounded to the nearest whole number.

```
calculate:  
result = $10 * $11  
answer = round result
```

trunc *value*

Truncates the number *value*, discarding any decimal places and leaving the integer part of the number. No rounding occurs, so truncating the number 29.9 returns the number 29.

In this example, the number in field 23 is truncated and the integer part of the number is stored in the current field.

```
calculate:  
answer = trunc $23
```

14. Scripting Examples

This chapter shows some examples of using scripts in forms. Before you read this chapter, it is important to read Chapter 13, *Scripting Reference*, starting on page 245, to learn about the scripting language used in Pendragon Forms.

Other chapters that follow will show examples of using scripts in:

- Forms designed for use with bar codes - see Chapter 15, *Using Bar Codes*, page 373.
- Printing from the handheld - see Chapter 16, *Printing from the Handheld*, page 399.
- Forms that attach images - see Chapter 17, *Working with Images*, page 419.
- Forms that allow beaming - see Chapter 19, *Beaming Records*, page 445.
- Performing lookups from one form to another - see Chapter 20 - page 455.

Using Scripts to Perform Calculations

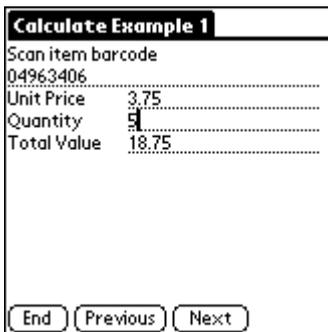
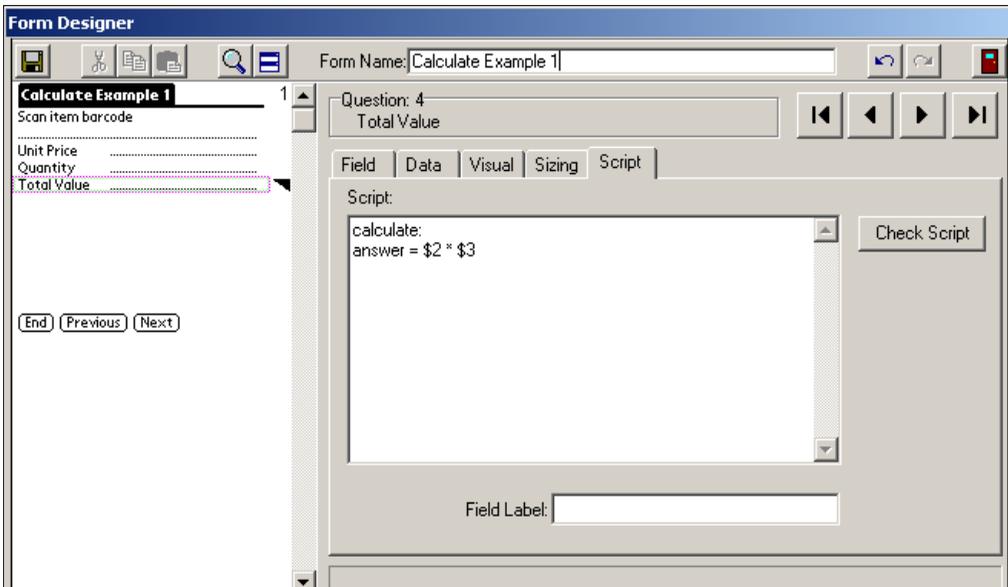
If you need to perform a calculation, you can add a script to the field in which you want to display the result of the calculation.

- The **calculate**: event is most commonly used in calculation scripts. Since the calculate event will run the script whenever any field on the form is updated, fields can be updated in Layout View, Field View or Record View on the handheld.
- If you do not want the handheld user to be able to overwrite the result of a calculation, you can make the field which displays the result a read-only field (see page 179).
- The **answer** statement assigns a value to the current field of the form. (See example 1.) For example, `answer = $1 + $2` adds the values in fields 1 and 2 and places the result in the current field.
The `$N = expression` statement assigns the value in the calculated expression to field number N. For example, `$5 = $3 * $4` multiplies the values in fields 3 and 4 and puts the result in field 5.
- Only binary math expressions are supported on any line of a script. This means that a math expression such as `A + B + C` takes two lines of a script. (See examples 2 and 3.)
- If a calculation involving a currency is performed, and the result is placed into a currency field, the result will display properly as a currency. If the result is placed in a Numeric or Text field, the number displayed will be in cents. A line can be added to the script to convert the cents to dollars and cents.

Calculation Example 1: Performing a simple calculation

In the following form, the user can enter the price of an item and the quantity being purchased, and then calculate the total value of the items being purchased. The script to perform the calculation is in Field 4.

answer = \$2 * \$3 means that the answer to be placed in Field 4 is the multiplication of the value in Field 2 and the value in Field 3.



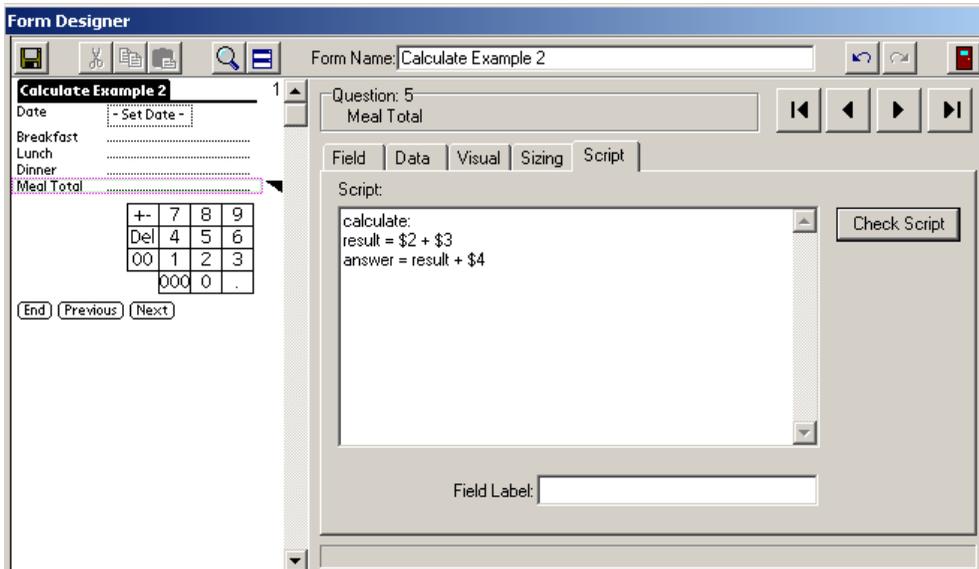
On the handheld, the **calculate:** script triggers when any field on the form changes. The user has to leave a field for the change in that field to be registered.

When the user fills in the Unit Price and Quantity, he/she will need to tap in any field for the calculation to occur.

Calculation Example 2: Adding more than two values together

The scripting language in Pendragon Forms only supports binary math functions, that is, math functions with two numeric expressions. In order to add more than two values, you can break a calculation into smaller parts, using the **result** variable as temporary extra storage space for storing intermediate results of a larger calculation.

In the form below, in order to calculate total meal expenses for the day in field 5, the statement **result = \$2 + \$3** stores the sum of Breakfast (field 2) and Lunch (field 3) in the **result** variable. The statement **answer = result + \$4** stores the sum of the **result** variable + Dinner (field 4) in the current field (field 5, the Meal Total).



Calculate Example 2				
Date	2/11/05			
Break fast	\$7.95			
Lunch	\$10.00			
Dinner	\$19.50			
Meal Total	\$37.45			
	+-	7	8	9
	Del	4	5	6
	00	1	2	3
		000	0	.
End	Previous	Next		

Calculation Example 3: Adding more than three values together

In this example, in order to calculate an average test score, the sum of test scores 1 through 5 must first be calculated.

Adding Several Fields	
Student Name	Mandy
Test score 1	72.5
Test score 2	81.2
Test score 3	90.3
Test score 4	74.5
Test score 5	95.0
Total points	413.5
# of Tests Taken	5
Average score	82.7

End Previous Next

```
calculate:
  result = $2 + $3
  result = result + $4
  result = result + $5
  answer = result + $6
```

Script in field 7 to calculate Total points

The "# of Tests Taken" field records how many tests have been taken, by adding a 1 for each non-zero score.

```
calculate:
  if $8 = 0 then
    return
  endif

  answer = $7 / $8
```

The script which calculates the average has a check to see if the number of tests taken (Field 8) is zero. If it is zero, the return statement ends the script to prevent a divide by zero error.

If Field 8 is not zero, then the average of Total points / # of tests is calculated.

```
calculate:
  result = 0

  if $2 <> null then
    result = result + 1
  endif

  if $3 <> null then
    result = result + 1
  endif

  if $4 <> null then
    result = result + 1
  endif

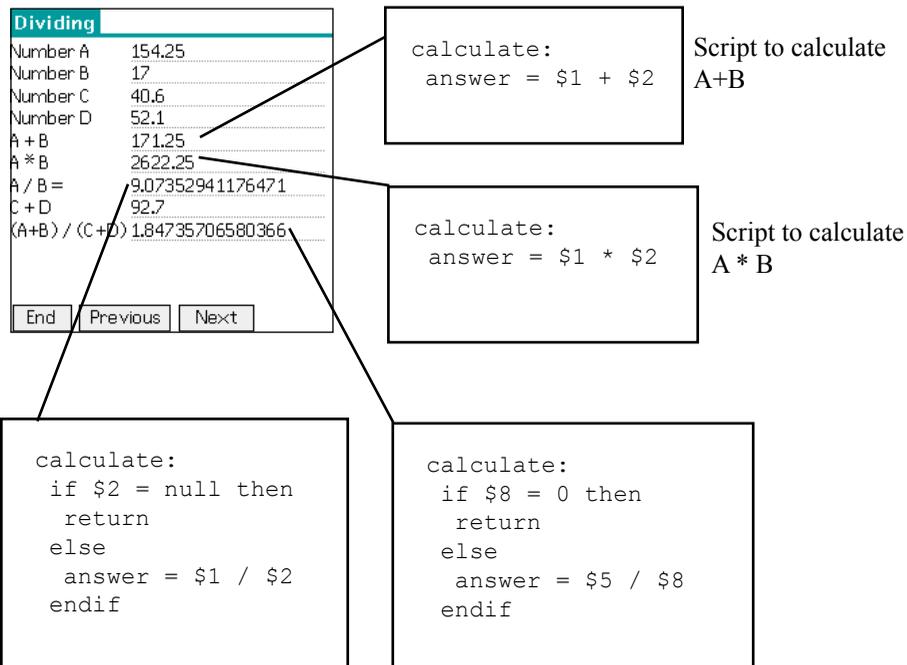
  if $5 <> null then
    result = result + 1
  endif

  if $6 <> null then
    result = result + 1
  endif

  answer = result
```

Calculation Example 4: Dividing

Whenever you perform a division in a form, you have to be careful to avoid division by zero errors. These errors can occur because **calculate:** scripts are triggered whenever an entry has been made in a field. If you have a form in which a number X is entered, followed by a number Y, and you want to calculate X/Y, the value of Y will be zero until the handheld user enters a value. As soon as a value is entered for X, the calculate script will run, and X/0 will cause an error on the handheld. There are simple scripting statements which can be used to prevent the calculation from occurring until field Y has a non-zero value.

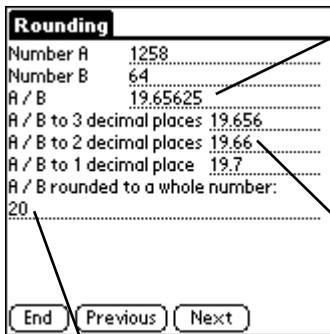


To calculate A/B in Field 7, a check is first done to see whether the divisor (number B in Field 2) is null (blank). If Field 2 is null, the return statement breaks out of the script so that the division is not performed. If Field 2 is not null, then the division of Field 1 by Field 2 proceeds.

Similarly, to calculate (A+B) / (C+D) in Field 9, a check is first carried out to see if the divisor (C+D) in Field 8 is zero.

Calculation Example 5: Rounding

When you divide one number by another, the handheld displays as many decimal places as possible. In some instances you may prefer to round your answer to a specific number of decimal places.



```
calculate:
  if $2 = null then
    return
  endif

  answer = $1 / $2
```

Script to calculate A/B, protecting against divide by zero errors. The answer has many decimal places.

```
calculate:
  if $2 = null then
    return
  endif

  result = $1 / $2
  result = result * 100
  result = result + 0.5
  result = integer result
  answer = result / 100
```

```
calculate:

  result = $3 + 0.5
  answer = integer result
```

Field 7 rounds the calculation of A/B in Field 3 to a whole number.

Adding 0.5 to the value in Field 3 causes Field 3 to be rounded up if the decimal part is greater than 0.5.

The integer statement takes the integer part of Field 3 and discards the decimal part.

The script shown above is in Field 5, and rounds the calculation of A/B to 2 decimal places.

Multiplying by 100 preserves two decimal places' worth of precision. Adding 0.5 rounds up if the decimal part is greater than 0.5. The integer statement discards the unwanted decimal places. Finally, dividing by 100 reverts to the correct level of precision.

To round to 3 decimal places, multiply and divide by 1000 instead of 100.

To round to 1 decimal place, multiply and divide by 10 instead of 100.

Calculation Example 6: Counter field

If you need to count the number of records in a form, there is a simple initialization script which can be used.

- In addition to the script, it is necessary to make the counter field an Autodefault field on the Advanced Field Properties screen for that field. (See page 187.)

The diagram illustrates the Counter field initialization script. It shows two screenshots of a form titled "Counter 1".

The first screenshot shows the form with the following data:

Tag #	1
Name of item	Chair
Price	\$199.95

The second screenshot shows the form with the following data:

Tag #	2
Name of item	Vase
Price	\$75.00

A callout box contains the initialize script:

```
initialize:
  answer = answer + 1
```

A line connects the callout box to the second screenshot, indicating that the script is applied to the counter field.

Setting Field 1 to autodefaut means that when a new record is created, the value of the previous record is retained. The initialize script then adds 1 to increment the counter.

If you need to start the counter from 300, for example, simply enter the number 300 in the first record created. Subsequent records will be 301, 302, 303, etc.

Note that if the form is re-distributed to the handheld, or if the handheld is reset, the counter will be reset to 1.

Working with Dates

Time fields, Date fields and Date & Time Fields can be used in calculations. Some notes:

- Dates are converted to numbers for calculations - the number of seconds since 01/01/1904.
- Results of calculations with dates must be stored in Text fields. It is not possible to subtract two times, for instance, and display the result in a Time field.
- The result of a calculation of two dates is expressed in seconds. The seconds can then be converted back to minutes, hours or days, as appropriate. (See Example 1.)
- An **initialize:** script can be used to set a default of the current date in a Date field, or the current date and time in a Date&Time field (see Example 1). Note however that the earliest date that you can set as a default is 01/01/1970.

Date Example 1: Today's date, and Time Elapsed

In this form, the first field defaults to today's date, and other fields calculate the time elapsed between a starting and ending time.

Time Elapsed	
Today's Date	2/11/05
Start Time	8:30 am
End Time	11:51 am
Seconds Elapsed	12060
Minutes Elapsed	201
Hours Elapsed	3.35
Hours part of time	3
Minutes part of	21
Total Time Elapsed	3 hrs and 21 mins
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

```
initialize:
  answer = now
```

The script in field 1 only runs once, when the record is created. The **answer = now** statement places the current date (& time if a Date&Time field) into Field 1.

```
calculate:
  answer = $3 - $2
```

A calculation of dates results in an answer in seconds.

```
calculate:
  result = $3 - $2
  answer = result / 60
```

To calculate minutes, divide the seconds by 60.

```
calculate:
  result = $3 - $2
  answer = result / 3600
```

To calculate hours, divide the seconds by 3600. (There are 3600 seconds in an hour.)

Date Example 2: Displaying Time Elapsed in Hours and Minutes

Since most people are used to expressed elapsed time in hours and minutes, you may want to display the results of date calculations in this way.

Time Elapsed	
Today's Date	2/11/05
Start Time	8:30 am
End Time	11:51 am
Seconds Elapsed	12060
Minutes Elapsed	201
Hours Elapsed	3.35
Hours part of time	3
Minutes part of	21
Total Time Elapsed	3 hrs and 21 mins
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

```
calculate:
answer = $3 - $2
```

In Field 4, the time elapsed is calculated in seconds.

```
calculate:
result = $4 / 3600
answer = integer result
```

The seconds are converted to hours by dividing by 3600, since there are 3600 seconds in an hour.

```
calculate:
result = $4 % 3600
result = result / 60
answer = integer result
```

The integer statement takes just the whole number of hours, and this is placed into a field for storage.

The modulo (%) statement records a remainder, in this case the remainder of seconds after the whole number of hours is discarded.

The seconds are converted to minutes, and the whole number of minutes is stored in a field.

```
calculate:
result = $7 & " hrs and "
result = result & $8
answer = result & " mins"
```

Finally, to display the time elapsed as the phrase "X hrs and Y mins", the concatenate operator (&) is used to combine the whole number of hours, + the phrase " hrs and " + the whole number of minutes + the phrase " mins".

Date Example 3: Calculating Ages

A useful date calculation is to calculate a person's present age.

Calculating Ages	
Last Name	Kingston
First Name	Jenni
Date of Birth	3/9/73
Present Age	31.9307818085024
Age in years to 2 d.p.	31.93
Age this year	32

End Previous Next

```
calculate:  
  result = now - $3  
  result = result / 86400  
  answer = result / 365.25
```

The statement **result = now - \$3** calculates the number of seconds from the date of birth to the present time, and put the answer in the result variable.

The result is divided by 86400 to convert seconds to days.

The number of days is divided by 365.25 to convert to years. (There are approx. 365.25 days in a year, accounting for leap years.)

```
calculate:  
  result = $4 * 100  
  result = result + 0.5  
  result = integer result  
  answer = result / 100
```

The result can be truncated to two decimal places by multiplying by 100 to preserve two decimal places' of precision, then adding 0.5 to round up. The integer statement takes a whole number and discards the excess decimal places, and then dividing by 100 reverts to the correct level of precision.

```
calculate:  
  result = $4 + 0.5  
  answer = integer result
```

To round the present age to the age that the person will be this year, add 0.5 to round up and then the integer statement takes the whole number of years.

Branching

Scripts can be used in fields to allow branching on a form. With branching, the handheld user's response in one field determines which additional fields are displayed.

- **Layout View** displays several fields on a screen at a time. If you are branching from a Layout View screen, put the extra questions on a separate screen, and as the user exits the current screen, branch (jump) to that screen if needed, or jump past that screen if it is not needed.
The **exitscreen:** event is used to branch in Layout View.
A **select:** event can also be used for branching in Layout View if you want the user to jump to another part of the form immediately as a selection is made in a Popup List, Yes/No Checkbox or Lookup List field.
- A **select:** event will not run if the handheld user does not make a selection.
In Layout View an **exitscreen:** event is more failsafe than a **select:** event. In Record View, there is no way to branch from a selection field if the handheld user does not make a selection.
- Field View displays one field per screen. Record View displays 11 fields per screen in a two-column format.
In Field View, an **exit:** event runs the script when the user exits the field (**exitscreen:** does not work in Field View). A **select:** event can also be used in Field View.
In Record View, **exit:** events only run in Text, Numeric and Currency fields, and **select:** events only run in selection fields such as Yes/No checkboxes, Popup Lists and Lookup Lists. There is no scripting event that is guaranteed to run to allow branching in Record View. Therefore, Layout View or Field View are preferred when using branching scripts.
- The **goto** statement is used to branch from one field to another. The **goto** statement can only be used in **select:** , **exitscreen:** (in Layout View only) , **exit:** (in Field View only), **click:** and **scan:** events.
- The **hide** and **show** statements are used to hide fields which the user does not need to fill in, and to display fields which the user does need to fill in. A field can be Hidden (see page 176) during the form design process in the Form Designer, and then the **show** or **show from...to** statements can be used to display fields as needed.
- Conditional statements, such as IF...THEN...ELSE...ENDIF and SWITCH/CASE are used to determine whether to branch to a question or not.

Branching Example 1: Branching from a Yes/No Checkbox in Layout View

In a Yes/No checkbox, there are three possible answers: Y for Yes, N for No, or null if the field is left blank. A branching script has to contain instructions for all possible answers. A blank response can be treated the same as answering No, if that is appropriate.

A **select:** event will run the script as the user makes a selection in the field. However, if the user leaves the field blank, a select: script will not run. An **exitsscreen:** event runs when the user taps the Next button to move to the next screen, so an exitsscreen: script will run even if the Yes/No field is left blank.

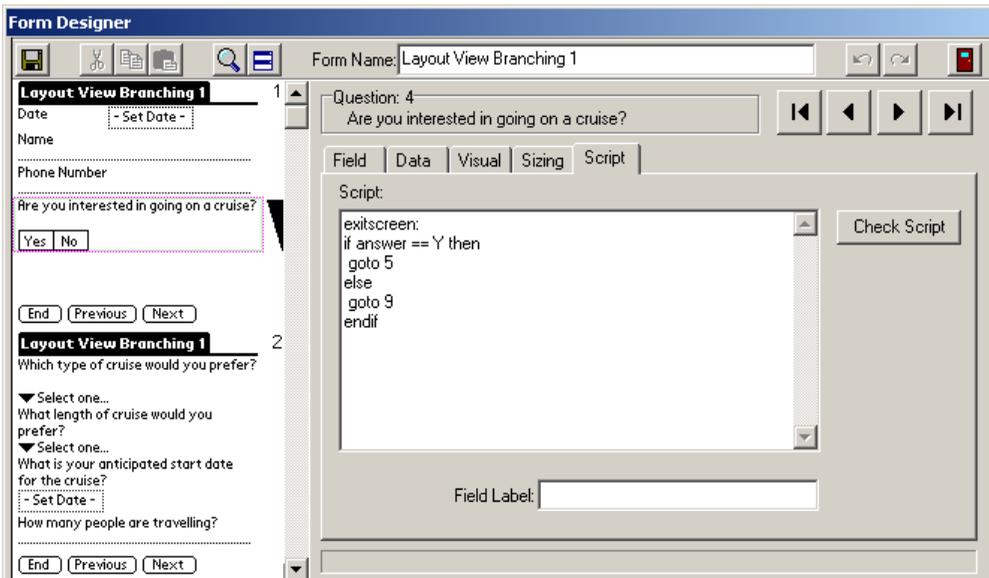
On the form below, field 4 asks a Yes/No question. If the user answers Yes, then upon exiting the screen, the user will branch to field 5, which starts on the second screen of the form. If the user answers No, or leaves the Yes/No field blank, then he/she will skip all the questions on the second screen of the form, and will instead branch to field 9.

The **exitsscreen:** event in field 4 runs when the user taps the Next button to move to the next screen. The if...then...else...endif statement sets up the branching conditions.

The statement

```
if answer == Y then  
  goto 5
```

means that if the answer in field 4 is Yes, the form will advance to display field 5.



The **else** component of the `if...then...else...endif` statement covers what happens if the answer is not Yes, that is, if the answer is No or null.

```
else
  goto 9
endif
```

means that if the first condition (`answer == Y`) is not met, the form will branch to field 9.

Layout View Branching 1

Date: 5/15/03

Name: Julianna Martinez

Phone Number: 708-555-6323

Are you interested in going on a cruise?

Yes No

End Previous Next

Layout View Branching 1

Which type of cruise would you prefer?

▼ Select one...

What length of cruise would you prefer?

▼ Select one...

What is your anticipated start date for the cruise?

- Set Date -

How many people are travelling?

End Previous Next

Layout View Branching 1

Do you prefer to travel by air?

Yes No

Do you have a destination in mind?

▼ Select one...

End Previous Next

On the handheld, selecting Yes in field 4 jumps the user to the next screen (field 5) when the Next button is tapped and the `exitScreen: event` runs.

Selecting No in field 4, or leaving field 4 blank causes the user to skip all the questions on the second screen, and jump ahead to field 9 on the third screen.

If the user jumps ahead to field 9, but then taps the Previous button, the screen with the skipped questions will be visible.

Note: If you were using Field View instead of Layout View, the script would need to be changed from an `exitScreen: script` to an `exit: script`.

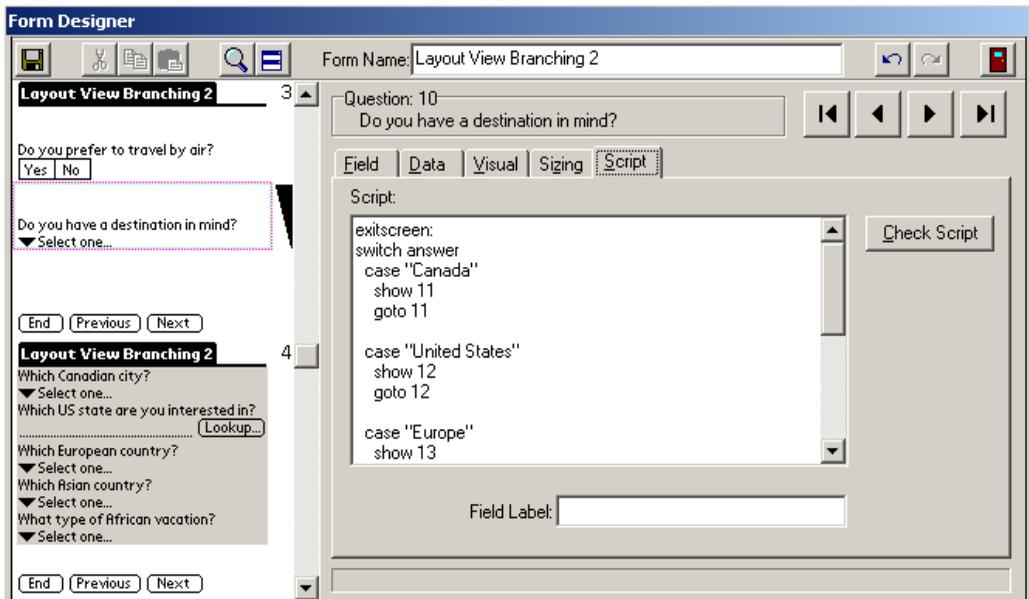
Branching Example 2: Branching from a Popup or Lookup List in Layout View

If you want to branch based on a selection in a Popup or Lookup List, your script must account for every possible selection that the user can make from the list.

A **select:** event can be used, but the user has to make a selection for the script to run. If the user leaves the field blank, the script will never run. You can use an **exitscreen:** event if you want the script to run when the user leaves the current screen, whether or not a selection has been made.

A switch/case statement is used to specify what branching will occur in the for each of the possible options in the Popup or Lookup List.

In the form shown below, field 10 is a Popup List with a list of travel destinations. Depending on the user's selection, a question relating to that destination will be displayed, and the form will advance to that question. The extra questions are hidden in advance by checking the Hidden checkbox on the Visual tab of the Form Designer window, and the script will unhide the relevant question.



Here is the script in Field 10 in full:

```

exitscreen:
switch answer
  case "Canada"
    show 11
    goto 11

  case "United States"
    show 12
    goto 12

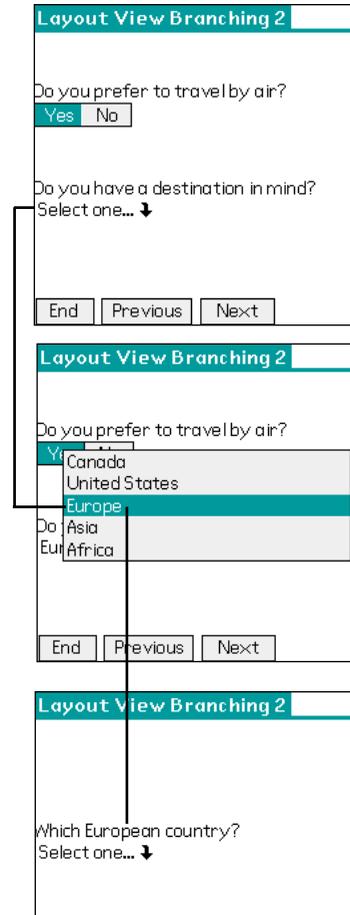
  case "Europe"
    show 13
    goto 13

  case "Asia"
    show 14
    goto 14

  case "Africa"
    show 15
    goto 15

  case anyvalue
    msgbox "Please pick a destination"
endswitch

```



The **exitscreen:** event script has to contain instructions for each possible selection that the user can make in the Popup List.

```

exitscreen:
switch answer

```

means that the value in the `answer` variable (that is, the value selected in the current field, will be compared to each of the options in the case statements.

```

  case "Canada"
    show 11
    goto 11

```

means that if the value in `answer` is Canada, field 11 will be displayed (`show 11`) and the form will branch to field 11 (`goto 11`). Similarly, each case statement details the actions that will be performed for each option in the Popup List.

At end of the script is this statement:

```
case anyvalue
    msgbox "Please pick a destination"
```

Adding `case anyvalue` to the script will cover the actions to take if none of the case statement values match the value in the answer variable. For instance, if the user does make any selection from the list, the `case anyvalue` actions will be performed.

Adding `case anyvalue` to a switch/case statement is optional.

The last line of the script is:

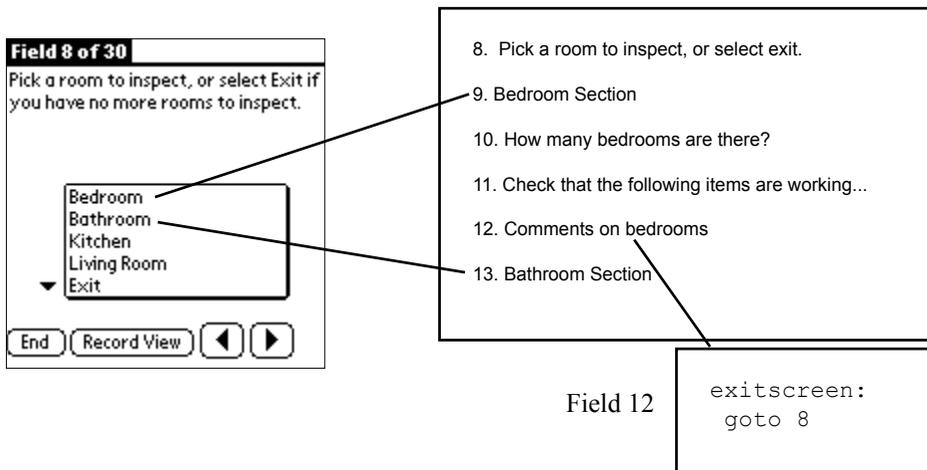
```
endswitch
```

Every `switch` statement must end with the word `endswitch`.

Branching Example 3: Using a Goto statement with a Jump to Section field

A Jump to Section field contains some built-in branching capability that automatically allows you to jump to Section fields on your form. (See page 108 on Section fields and page 111 on Jump to Section fields.) If you want the handheld user to branch back to the Jump Popup field after completing a section of a form, you can use a **goto** statement in the last field of that section.

In the following form, Field 8 is a Jump to Section field that allows the handheld user to jump to a section on the form. After the user has gone through the fields in a section, you may want the user to branch back to the Jump to Section field to be able to select another section.



When the handheld user exits the last field in a section, the form branches back to Field 8, the Jump Popup field. The Exit section is the only section which does not branch back to Field 8, so if the user selects the Exit option, he/she jumps to the final field(s) on the form and can end the record.

Note:

In the example above, an **exitscreen:** event is used because the form is displayed in Layout View (several fields per screen). If Field View was being used (one field per screen), an **exit:** event would have to be used instead, as **exit:** events do not run in Layout View.

Using Scripts in Button Fields

Scripts are used to carry out actions when the handheld user taps on a button in a Button field.

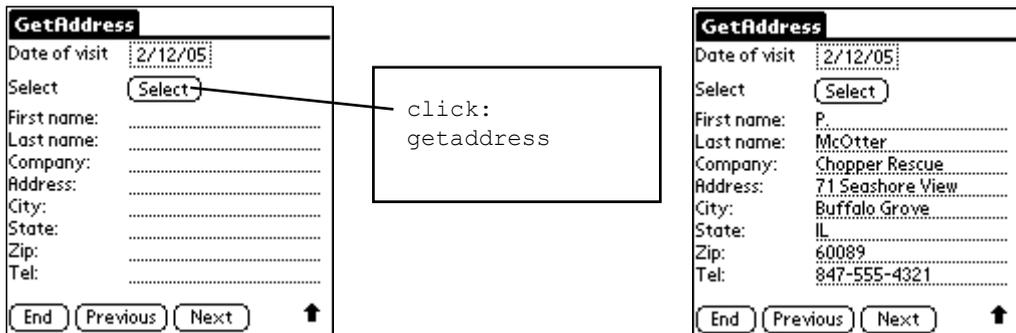
- The label on a button is set by typing in a Default value (up to 11 characters) on the Data tab of the Button field in the Form Designer window (see page 191).
- The **click:** event is used to activate what happens when a button is tapped. A click: script runs whenever the button is tapped, in Layout View, Field View or Record View.
- The label on a button can be changed in a click: script with the assignment statement `answer = "Name-of-label"`

Button script Example 1: GetAddress button

PALM OS

A **click:** event with the `getaddress` statement allows the user to perform a Lookup to the handheld AddressBook, and select a name to paste into the Pendragon form.

The `getaddress` statement only works on Palm OS devices.



The `getaddress` statement only copies fields from the AddressBook if there are fields with corresponding names on the Pendragon Form. The fields which can be copied are:

First name:	Address:	Country:
Last name:	City:	Tel:
Title:	State:	Email:
Company:	Zip:	Custom:

The colon character (:) has to be included in the field name on the form.

Tel: is the Work phone number, and Custom is the Custom 4 field in the AddressBook.

Button script Example 2: Calculate button

Calculate: scripts run whenever a field is updated. This is processor intensive, and may also cause calculations to be performed before the handheld user is ready. An alternative is to use a button to give the handheld user control of when the calculation is performed.

Calculate Button		
Take-home Pay	\$3,450.00	
Other Income	\$100.00	
Mortgage/Rent	\$1,500.00	
Food expense	\$400.00	
Utility / Car	\$600.00	
Credit card	\$400.00	
Entertainment	\$300.00	
Click to calculate	<input type="button" value="Calculate"/>	
Total Income		
Total Expenses		
Amt to Save		
<input type="button" value="End"/>	<input type="button" value="Previous"/>	<input type="button" value="Next"/>

```

click:
  $9 = $1 + $2
  result = $3 + $4
  result = result + $5
  result = result + $6
  $10 = result + $7
  $11 = $9 - $10
  
```

Calculate Button		
Take-home Pay	\$3,450.00	
Other Income	\$100.00	
Mortgage/Rent	\$1,500.00	
Food expense	\$400.00	
Utility / Car	\$600.00	
Credit card	\$400.00	
Entertainment	\$300.00	
Click to calculate	<input type="button" value="Calculate"/>	
Total Income	\$3,550.00	
Total Expenses	\$3,200.00	
Amt to Save	\$350.00	
<input type="button" value="End"/>	<input type="button" value="Previous"/>	<input type="button" value="Next"/>

In the form above, the last three fields on the form (Fields 9, 10 and 11) are not calculated until the user taps the Calculate button.

The advantage of using a button for calculations is that the user does not see intermediate results on screen before all the necessary data is input.

IMPORTANT: The disadvantage to using a button for calculations is that if the user changes a field, the calculated results do not change until the button is tapped again.

Other Button scripts

Button fields with **click:** event scripts can be used for:

- Calculating a total across all subform records for a given parent record, and placing the result in the parent record. See page 367 for an example.
- Printing directly from the handheld - see page 405.
- Beaming records from one handheld to another - see page 447.
- Attaching image files to a record - see page 426.

Using a Script in a MultiSelection field

MultiSelection fields are stored internally as a binary number, with each option in the MultiSelection List being represented by a bit position in the binary number.



For example, the Multi-Selection List shown here uses the following bit positions:

64	32	16	8	4	2	1
Dessert	Vegetarian	Fish	Meat	Salad	Soup	Appetizer

Each checked box in the list sets the corresponding bit position as 1. Internally, the choice of Appetizer, Salad, Vegetarian and Dessert is stored as the binary number 1100101.

For a script to react to the selections in a Multi-Selection list, the binary **AND** operator can be used to check whether a given bit position has been set. In the script shown at right, the answer in the Multi-Selection field is compared to each bit position in turn. If the bit is set, the result of the AND operation is equal to the value of the bit position. If the bit is not set, the result is 0.

For example, the statement `result = answer and 4` compares the answer in the Multi-Selection field with the bit position 4, to test if the “Salad” checkbox has been checked. Depending on the options checked in the field, various hidden fields are displayed using the Show statement.

A **goto** statement is needed at the end of an **exitscreen:** script to decide which field is displayed next. If a field is hidden, the goto statement will go to the next visible field on the form.

```

exit:

result = answer and 1
if result = 1 then
show 4
endif

result = answer and 2
if result = 2 then
show 5
endif

result = answer and 4
if result = 4 then
show 6
endif

result = answer and 8
if result = 8 then
show from 7 to 8
endif

result = answer and 16
if result = 16 then
show 9
endif

result = answer and 32
if result = 32 then
show 10
endif

result = answer and 64
if result = 64 then
show 11
endif

goto 4

```

Using Scripts with Parent forms and Subforms

A parent form and a subform are not really linked on the handheld - they only appear to be connected. Some scripts can be used to further enhance the appearance of a connection between the parent form and subform.

Example 1: Adding a field across subform records, and placing the result on the parent form

Customer Order (Parent2)	
Company	Chopper Rescue
Order Date	2/12/05
Contact Name	P. McOtter
Contact Phone	847-555-4321
Items Ordered	
Customer Order (Subform2)	
Calculate Order	
Total	
Order Total	\$39.95
End	Previous Next

The **subformsum** statement can be used to add up the values in a given field on a subform, and place the calculated result in the Result variable.

In the parent form shown at left, Field 5 is a Subform field that jumps to a subform used to enter items being ordered.

Field 6 is a Button field, that the user taps to run the following script:

```
click:
subformsum "Customer Order (Subform2)" 8
$7 = result
```

The statement `subformsum "Customer Order (Subform2)" 8` adds up the values in Field 8 of all the subform records that match the parent record. (In this example, field 8 on the subform contains the total for one item being ordered.) "Customer Order (Subform2)" is the name of the subform. The result of the calculation is placed in the **result** variable.

The statement `$7 = result` then places the calculated result in Field 7 of the parent form.

Note that if the handheld user enters more subform records after the Order Total button has been tapped, he/she will need to tap the button again to update the parent form with the new total.

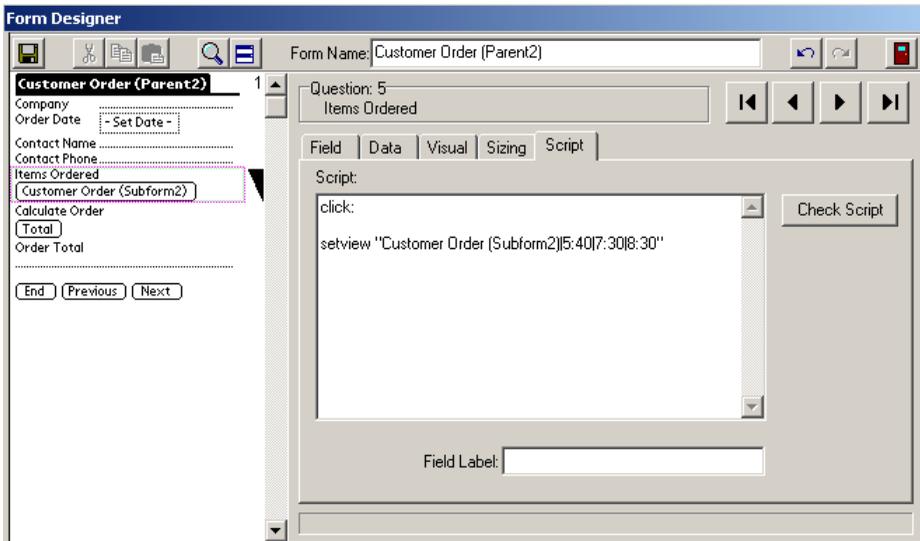
Example 2: Setting the columns to view when the Subform Review screen is displayed

You can select which columns the handheld user will see on a subform review screen.

Note: you can also set the default columns that will be viewed on the review screen via the Columns tab in Advanced Form Properties - see page 226.

The setview scripting statement allows you to set the columns that will be visible on the subform review screen, and the percentage of the screen width that the column will occupy. (See page 327) The setview statement can be run in a click: event in the subform field on the parent form. Alternatively, setview can be put in an open: event on the parent form. Setview must be on the parent form, not the subform.

Setview can only be used to specify up to 3 columns, and the total screen percentage cannot exceed 100%.



The script in the Subform field of the parent form is:

```
click:
```

```
setview "Customer Order (Subform) |5:40|7:30|8:30"
```

The setview statement

```
setview "Customer Order (Subform) |5:40|7:30|8:30"
```

means that the subform review screen for the form Customer Order (Subform) should display column 5 at 40% of the screen width, column 7 at 30% of the screen width and column 8 at 30% of the screen width. (40% + 30% + 30% = 100%)

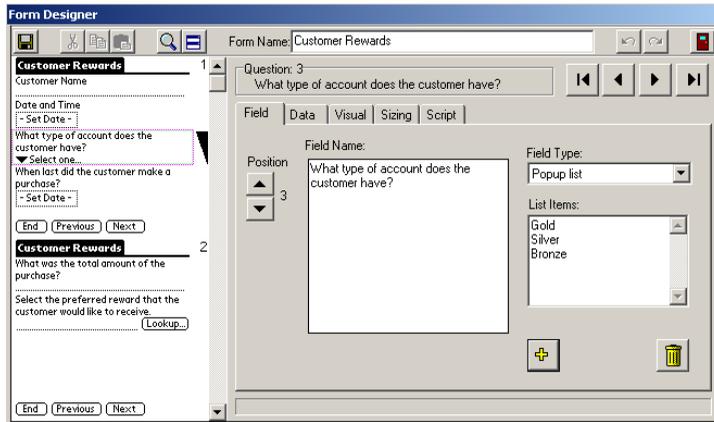
The image shows three screenshots related to a subform in a parent form:

- Customer Order (Parent):** A form with fields for Company (Chopper Rescue), Order Date (2/12/05), Contact Name (P. McOtter), and Contact Phone (847-555-4321). It includes a subform field labeled "Customer Order (Subform)", a "Calculate Order" button with a "Total" sub-button, and an "Order Total" of \$39.95. Navigation buttons "End", "Previous", and "Next" are at the bottom.
- PALM OS:** A screenshot of the subform on a PALM OS device. The subform title is "Subform" and the content is "Chopper Rescue > Customer Order". It displays a table with three columns: "Item", "Quantity", and "Total". The table contains two rows: "Toy Polar Bear" with a quantity of 1 and a total of \$19.95, and "Toy Sea Otter" with a quantity of 1 and a total of \$20.00. Below the table are "Add", "Lookup:", and "Done" buttons.
- PPC:** A screenshot of the subform on a PPC device. The subform title is "Subform" and the content is "Chopper Rescue > Customer Order". It displays the same table as the PALM OS version, but with different column widths. Below the table are "Add", "Lookup:", and "Done" buttons.

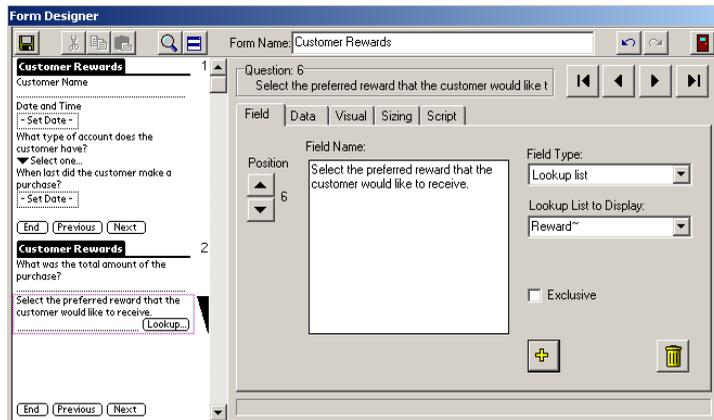
Using Scripts to do a Cascading Lookup

A Cascading Lookup enables you to display a different Lookup List depending on a selection in another Popup List or Lookup List field. If the two selection fields are adjacent to each other (e.g. field 5 and 6) , no scripting is required to do a cascading lookup - see page 119.

With the **lookupname** variable and the **setlookupname** statement, it is possible to do a cascading lookup without the two fields being adjacent to each other.



As with a regular cascading lookup, one field on the form is a Popup List or a Lookup List that allows the user to select some type of category. In the form shown at left, field 3 is a Popup List that lets the user choose from Gold, Silver and Bronze categories.



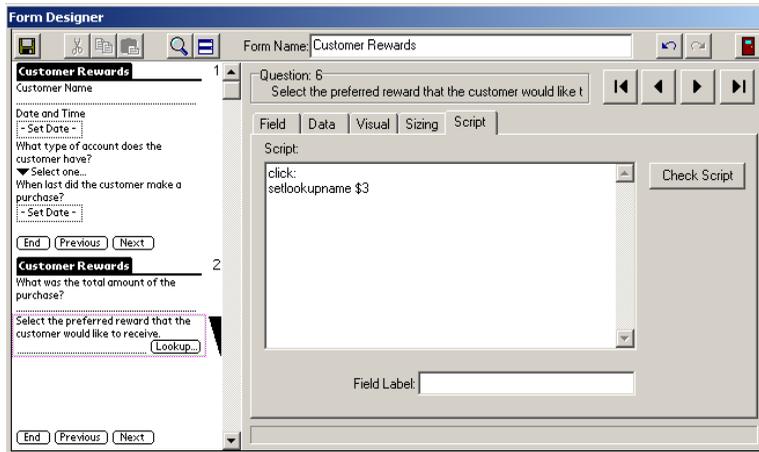
Field 6 is the Lookup List field that will display a Lookup List that depends on what the user selects in field 3. Instead of referencing the name of an actual Lookup List, the Lookup to Display field references a keyword followed by the tilde symbol (~).

In the example below, the keyword is the word Reward, so the phrase **Reward~** is entered as the Lookup List to display.

The following script is written in field 6, the Lookup List field:

```
click:
  setlookupname $3
```

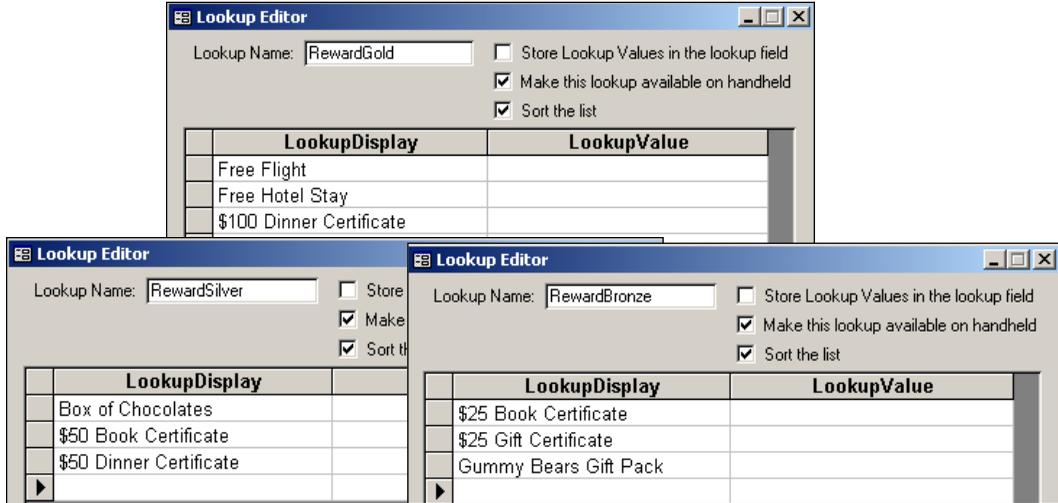
The **click**: event runs when the user taps in the Lookup List field. The statement `setlookupname $3` assigns the value in field 3 to the variable **lookupname**. To determine which Lookup List to display, Pendragon Forms will replace the tilde symbol (~) with the value in **lookupname**.



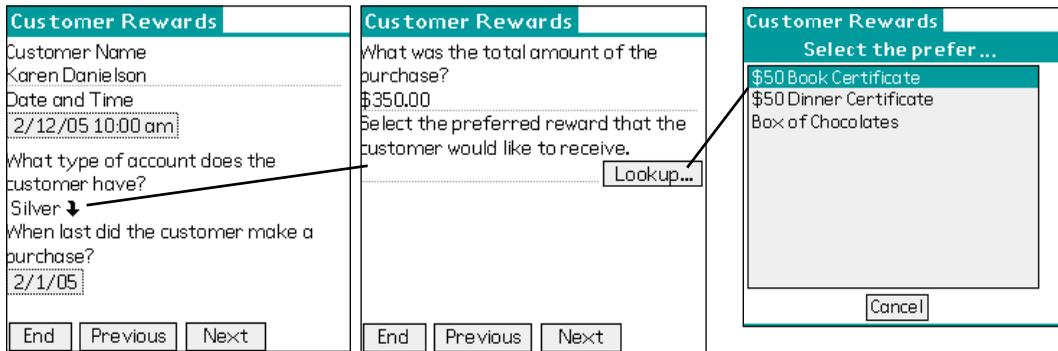
To make the Cascading Lookup work, you will also need to create a Lookup List for each of the possible options that the user can select. The name of each Lookup List must be the keyword followed by the category name from the first Popup or Lookup List.

For example, if the keyword was Reward, and the categories were Gold, Silver and Bronze, it would be necessary to create three Lookup Lists, called RewardGold, RewardSilver and RewardBronze. Do not put a space between the keyword and the category name.

If a category was called "Copper and Zinc", then with a keyword of Reward, the Lookup List would be called "RewardCopper and Zinc".



On the handheld, selecting an option in field 3 determines which Lookup List will be displayed in field 6.



NOTES:

Instead of using the lookupname variable, setlookup statement and the tilde (~) symbol, you can use the **lookuplocale** variable, **setlookuplocale** statement and the caret (^) symbol. So, for example, if you are using setlookuplocale, and your keyword is Reward, the LookupList to display would be Reward^ .

For an even more selective cascading lookup, you can actually have two selection fields followed by the cascading Lookup List field, and for the Lookup List to display, enter the keyword followed by both the tilde and the caret symbols. For example, with a keyword like Reward, the LookupList to display would be Reward~^ . The click: event in the Lookup List field would use both setlookupname and setlookuplocale statements.

15. Using Bar Codes

Bar Code Scanners

Pendragon Forms supports the following types of bar code scanners for handheld devices:

Manufacturer	Model	Platform	Description
Microvision www.flicscanner.com	Flic Cordless	Pocket PC	Bluetooth-enabled laser bar code scanner. Note: Requires a Bluetooth-enabled handheld.
Microvision www.flicscanner.com	Flic Cordless	Palm OS	Bluetooth-enabled laser bar code scanner. Note 1: Requires a Bluetooth-enabled handheld. Note 2: Also requires third-party software (see page 374).
Janam Technologies www.janam.com	XP20 Series XP30 Series	Palm OS	Handheld with built-in imaging bar code scanner
Symbol Technologies www.symbol.com	SPT 1550 SPT 1800	Palm OS	Handheld with built-in laser bar code scanner.
Symbol Technologies www.symbol.com	PPT 8800 MC 50	Pocket PC	Handheld with built-in laser bar code scanner.
Socket Communications www.socketcom.com	Secure Digital Scan Card 3E (SDSC) 3E	Palm OS or Pocket PC	Imaging bar code scanner that fits into an SD/MMC slot of a handheld device. The SD slot must support I/O.

Using a Microvision Flic Cordless Bar Code Scanner

The Microvision Flic Cordless bar code scanner works with Pocket PC devices and Palm OS devices. On Palm OS devices, additional third party software is also required:

Type of Handheld Device	Additional Third Party Software Required?
Pocket PC device with Bluetooth	No.
Palm OS device with Bluetooth	Yes. SerialMagic software from www.serialio.com may be required. See www.pendragonsoftware.com for updates.

Pairing a Flic Cordless device with a Bluetooth-enabled Pocket PC Device

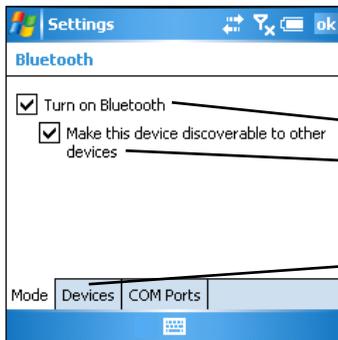
PPC

On Pocket PC devices, before you can use a Flic Cordless device to scan bar codes into Pendragon Forms, the Flic Cordless device must be paired with the Bluetooth-enabled Pocket PC device.

Note: These pictures show a Palm Treo 700w device. Your Pocket PC may display the Bluetooth setup screens differently, but the same functionality will apply.



1. Tap the Bluetooth icon on the Pocket PC device.
(The Bluetooth icon may be in a different location on your Pocket PC device.)



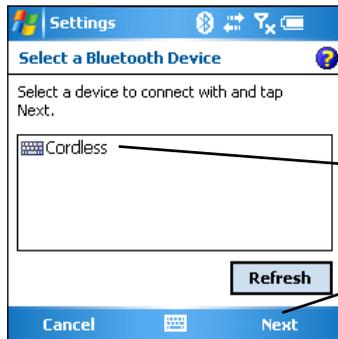
2. Tap the checkboxes or buttons to:
 - a) Turn on Bluetooth
 - b) Make this device discoverable to other devices.

Then tap the Devices button.

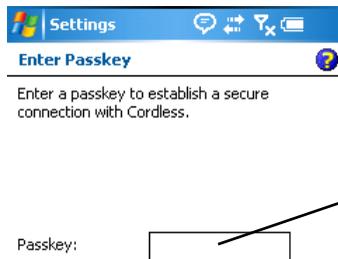


3. **Important:** On the Flic device, press the button to activate the device.

Then on the Pocket PC device, tap New Partnership for the handheld to detect and pair with the Flic device.



4. After scanning for devices, the Pocket PC will detect the Flic as a device with the name Cordless. Tap on the Cordless name, and then tap Next.



5. You will be prompted to enter a Passkey for a secure connection with the Flic device. Enter the Flic passkey that came with your device. (See the manufacturer documentation.) Then tap Next.

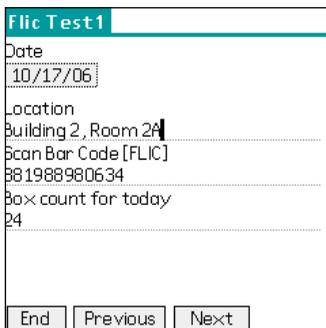


6. On the Partnership Settings screen, tap Serial Port, and then tap Finish.

Using a Flic Cordless Scanner with Pendragon Forms on a Pocket PC

PPC

On Pocket PC devices, Pendragon Forms provides two ways of connecting with a Flic scanner that has been paired with the handheld. (See page 374 for information on pairing a Flic scanner with a Pocket PC.)



Option 1:

The first option is to add the word FLIC in square brackets and uppercase letters [FLIC] to a field name on each screen where you will be scanning a bar code.

Pendragon Forms will try to connect with the Flic device on any screen containing a field with the word [FLIC].

If you do not want your users to see the word [FLIC], you can hide it by adding blank lines to the field name and adjusting the visible question height.

When the handheld user gets to the screen on the handheld containing the word [FLIC], he/she will be prompted to connect with the Flic scanner.



To activate the Flic scanner, press the button on the scanner. Then tap OK on the *Connect with Flic Scanner* dialog window in Pendragon Forms. The connection may take a few seconds to be established, and the Flic scanner will beep when a successful connection has been made.

The connection should stay open if the scanner remains in use. This means that after an initial delay to connect the first time, subsequent scans will be recorded in Pendragon Forms without the user being prompted to connect with the scanner.

If the scanner is not used for a while and switches off to conserve battery power, then on the next screen containing [FLIC] in a field name, you may be prompted once again to connect to the Flic scanner. Remember to press the scanner button to activate the scanner.

Forms Preferences

Color Record Lists Set Color...

Show Record ID Numbers (#00)

Indicate Modified Records (*)

SocketScan Trigger: Button 4 ↓

Use Treo Keyboard Navigation

Use Microvision Flic

Show Sync Button

Option 2:

If you do not want to add the word [FLIC] to a field on your form design, an alternative is to go to the Forms Preferences screen on the handheld and check the *Use Microvision Flic* checkbox.

For information on how to access the Form Preferences screen on the handheld, see page 70.

Connect with Flic Scanner

i Press the button on the Flic scanner, then tap OK to connect.

If *Use Microvision Flic* is switched on in Form Preferences, Pendragon Forms will attempt to connect to the Flic device whenever you run the Pendragon Forms application on the handheld.

The handheld user can press the Flic scanner button to activate the Flic device, and then tap OK on the *Connect with Flic Scanner* dialog in Pendragon Forms to proceed with connecting to the Flic scanner.

Or, the handheld user can tap Cancel if he/she does not want to connect with the Flic scanner during this Pendragon Forms session. If the user decides to use the Flic scanner after all, he/she will need to exit Pendragon Forms and then go back into the application to be prompted to connect with the Flic scanner again.

Pairing a FLIC Cordless device with a Bluetooth-enabled Palm OS Device

PALM OS

On Palm OS devices, before you can use a Flic Cordless device to scan bar codes into Pendragon Forms, the Flic Cordless device must be paired with the Bluetooth-enabled Palm OS device.

Important: You may need to use a third-party software application called SerialMagic from www.serialio.com in order to pair the Flic device with the Palm OS handheld. Refer to www.pendragonsoftware.com for updated information.

Pendragon Forms will work with both SerialMagic LC and SerialMagic Pro.

To pair the Flic device with the Bluetooth-enabled Palm OS device do the following:

1. Install the SerialMagic .PRC program on the Palm OS device.
2. Tap the SerialMagic icon.
3. On the SerialMagic screen:
 - a) Tap the Bluetooth button.
 - b) Check the LaserChamp BT checkbox.
 - c) Tap the SerialMagic Actions button.
4. On the SerialMagic Actions screen, if <<CR>> is not displayed on the screen, tap the Actions selection list at the bottom of the screen, and then select CR. This selects a Carriage Return as the character that occurs at the end of a bar code. Tap OK to exit the SerialMagic Actions screen.
5. On the Flic Cordless device, press the button to activate the device.
6. On the SerialMagic screen, tap the Start button to cause the handheld to search for nearby Bluetooth devices.
7. A Discovery Results screen will be displayed. The Flic Cordless device will appear with the name Cordless. Tap the name Cordless and then tap OK.
8. A Bluetooth Security screen will be displayed. Enter the Passkey for the Flic Cordless device. (See manufacturer's documentation.) Check the checkbox to Add to Trusted Device List.
9. You will be returned to the SerialMagic screen. If a Stop button is on the screen, it means that SerialMagic has made a connection to the Flic Cordless device, and the connection will run until the Stop button is pressed. You can now go to Pendragon Forms on the handheld to begin using the Flic device to scan bar codes in a form.

Using a Flic Cordless Scanner with Pendragon Forms on a Palm OS Handheld

PALM OS

On Palm OS devices, if the SerialMagic application is being used to connect the handheld to the Flic Cordless scanner, then before going into Pendragon Forms, the user will need to tap the SerialMagic icon on the handheld and select the Cordless scanner to connect with.

If the Flic Cordless scanner has already been paired with the handheld (see previous page), then:

1. Press the scanner button on the Flic Cordless scanner to activate the device.
2. Tap the SerialMagic icon. On the SerialMagic screen, tap Start.
3. A Discovery Results screen will display a list of Bluetooth-enabled devices that the handheld has found. Tap Cordless and then tap Connect.
4. The SerialMagic screen will be displayed with a message saying Connecting... If a Stop button is then displayed, it means that the handheld has connected with the Flic scanner. The user can now go to the Pendragon Forms application.

Using the Socket Secure Digital Scan Card 3E

The Socket Communications Secure Digital Scan Card 3E device (SCDSC 3E) fits into the SD/MMC slot of a handheld device. The SD slot must support I/O.

Secure Digital Scan Cards do not beep when a scan is done, so to give audible feedback you can add a BEEP statement to your scan: event scripts.

PALM OS

On Palm OS devices, the button used to trigger SDIO In-Hand Scan Card bar code scanner can be selected in a script in the form, or selected via the Forms Preferences screen.

PPC

On Pocket PC devices, you must install and run the SocketScan application that ships with the Secure Digital Scan Card to configure which handheld button is used to trigger the bar code.

Creating a field for receiving bar code input

Text fields are capable of receiving bar code input. There are two ways to implement bar code scanning on a form:

Bar Code Basic 1

Scan Bar Code
04963406

Enter Quantity on Shelf 24

Date and Time 2/17/05 10:14 am

+-	7	8	9
Del	4	5	6
00	1	2	3
	000	0	.

End Previous Next

1. The default behavior is that if the cursor is in a Text field and the bar code scanner is activated, the bar code will be placed into the current Text field.

Office Inventory

Date 2/17/05

Type of Item
▼ Monitor

Location
Building 1, Floor 3, Room 3C

Scan bar code
0051340374310

End Previous Next

2. Alternatively, a **scan:** event script can be added to a form to control bar code input into any Text field on the form. This means that the cursor can be in any field on the form when the bar code is scanned.

Field 1 of 7

Scan UPC/A barcode
[BARCODE]

073852096507

End Record View ◀ ▶ ↕

If bar codes are scanned in Field View (see page 59), then adding the code [BARCODE] to the end of the field name makes it possible to automatically jump to the next field if AutoNavigate is switched on. Having [BARCODE] in the field name also causes subsequent scans in the same field to replace the contents of the field.

Using Scripts to Control Bar Code Scans

Bar code scripts serve two main functions:

1. To control what happens when a bar code is scanned, such as placing the bar code into a particular field on the form, displaying a message if the same bar code is scanned twice, or cloning a record so that every scan creates a new record.
2. To control which types of bar codes (bar code symbologies) are allowed to be scanned.

Scripting commands are described in detail in Chapter 13, *Scripting Reference*, which starts on page 245. The table here is a quick summary of the scripting commands that apply primarily to bar code forms:

Scripting Command	Purpose	Flic Cordless	Symbol SPT 1550 SPT 1800 PPT 8800 Janam XP	Socket Secure Digital Scan Card 3E
open: event	An open: event is good for setting up which bar codes can be scanned, as an open: event runs whenever a record is opened.	√	√	√
enterscreen: event	Can also be used to set up which bar codes can be scanned.	√	√	√
scan: event	A scan: event runs when a bar code is scanned, and is used to control what happens after a scan.	√	√	√
scandata function	The scandata function stores the value of the most recently scanned bar code.	√	√	√
scantype function	The scantype function stores the bar code symbology of the most recently scanned bar code.		√	√ PPC
clone statement	The clone statement is used to make a copy of a record.	√	√	√
{enable disable} barcode	Switches on or off the the bar code laser scanner.		√	

Scripting Command	Purpose	Flic	Symbol SPT 1550 SPT 1800 PPT 8800 Janam XP	Socket SD Scan Card 3E
scanner <i>conversion code</i> {enable disable}	Enables or disables some bar code conversions, such as UPCE to UPCA. Valid conversion codes: UPCETOUPCA UPCE1TOUPCA EAN8TOEAN13 I2OF5TOEAN13 CODE39TOCODE32		√	
scanner {enable disable} all	Enables or disables all bar code symbologies.		√	
scanner {enable disable} <i>symbology</i>	Switches on or off a specific type of bar code. Valid symbologies: UPCA EAN8 UPCE EAN13 UPCE1 I2OF5 CODABAR ISBT128 CODE39 CODE128 CODE93 COUPON D2OF5 BOOKLAND_EAN MSI_PLESSEY TRIOPTIC39 UCC_EAN128		√	
scanner {system_character no_preamble system_character_country_ code checkdigit no_checkdigit} <i>symbology</i>	Switches on or off whether leading/prefix or trailing/suffix bar code digits are recorded. Valid symbologies: CODE39 UPCA I2OF5 UPCE MSI_PLESSEY UPCE1		√	
scanner {verify no_verify} <i>symbology</i>	Switches on or off verification of check digits on certain bar codes: CODE39 I2OF5		√	
scanner {opcc_checkdigit uss_checkdigit} I2OF5	Sets which type of I2OF5 checkdigit is allowed.		√	

Scripting Command	Purpose	Flic	Symbol SPT 1550 SPT 1800 PPT 8800 Janam XP	Grabba	Socket SD Scan Card 3E
enable grabba	Used in an open: event to enable Grabba compatibility and opens a connection to the Grabba device.			√	
disable grabba	Used to disable the Grabba port, in a validate: event when exiting a record.			√	
socketscan defaults on	Resets the Socket SDIO In-Hand scanner to its default settings.				√ PALM OS
socketscan <i>lengthparameter length</i>	Controls the min and max length for various barcode types. <i>lengthparameter</i> can be: CODE39_MIN CODE39_MAX CODE93_MIN CODE93_MAX I2OF5_MIN I2OF5_MAX D25_MIN D25_MAX CODABAR_MIN CODABAR_MAX MSI_MIN MSI_MAX CODE11_MIN CODE11_MAX WEBC_MIN WEBC_MAX <i>length</i> is the number of characters for the min or max.				√

Scripting Command	Purpose	Flic	Symbol SPT 1550 SPT 1800 PPT 8800 Janam XP	Socket SD Scan Card 3E
socketscan set_trigger <i>keycode</i>	Sets the hardware button that will trigger the scanner. Valid keycode values: 516 for DateBook/Calendar button 517 for AddressBook/Contacts button 518 for ToDo/Tasks button 519 for Memo/Notepad button			√ PALM OS
socketscan <i>setting value</i>	Sets various parameters such as the transmission and verification of checkdigits. (See page 330)			√
socketscan <i>symbology value</i>	Enables or disables the scanning of certain bar code symbologies. <i>value</i> is set to 1 to enable or 0 to disable the bar code symbology. Valid <i>symbology</i> values are: UPCA CODE39 UPCE CODE93 EAN8 I2OF5 EAN13 D25 CODABAR USS128 UCC_EAN128 CODE11 TRIOPTIC39 WEBC BOOKLAND_EAN MSI_PLESSEY			√
socketscan trigger now	Triggers a bar code scan immediately. Can be used to trigger a scan using a Button field.			√

Scanning from any field on a form

The default behavior of Pendragon Forms is that you can scan a bar code into a Text field only if the cursor is present in the Text field.

The screenshot shows a form titled "Bar Code Test 1". The form has the following fields: "Item bar code" (containing "04963406"), "Quantity" (containing "24"), "Cost" (containing "\$0.25"), and "Total Value" (containing "\$6.00"). Below these fields is a numeric keypad with buttons for "+-", "7", "8", "9", "Del", "4", "5", "6", "00", "1", "2", "3", "000", "0", and ".". At the bottom are buttons for "End", "Previous", and "Next".

When a new record is created in Layout View or Record View, the cursor is automatically placed in the first Text, Numeric or Currency field on the form.

If the bar code field is the first field on the form, as shown at left, then when the handheld user creates a new record, a bar code can be scanned without the user having to position the cursor.

An alternative to making a bar code field the first field on the form is to use a **scan:** event script. When a scan: event script is used, the scanned barcode is placed in the **scandata** function, not in any particular field. The value of **scandata** can then be assigned to any Text field on the form by the **scan:** event script.

A scanned bar code can be up to a maximum of 255 characters by default, or up to 2000 characters if you set the Max Length (see page 194) of the Text field to 2000 before you freeze the form.

The screenshot shows a form titled "Bar Code Example 1". The form has the following fields: "Item Name" (containing "Royal Blue Sweater"), "Barcode" (containing "0314254214"), "Quantity on shelf", and "Need to re-order?" (with a checkbox). Below these fields is a numeric keypad with buttons for "+-", "7", "8", "9", "Del", "4", "5", "6", "00", "1", "2", "3", "000", "0", and ".". At the bottom are buttons for "End", "Previous", and "Next".

In the example shown at left, the bar code field is in Field 2. The following script is used in Field 2 to control what happens when the handheld user scans a bar code:

```
scan:
  $2 = scandata
```

The **scan:** event overrides the default behavior of Pendragon Forms, which would have been to place the scanned bar code in Field 1, where the cursor is.

Instead, the scan: event stores the bar code in the scandata function, and the statement `$2 = scandata` then places the bar code into Field 2. With this scan: event script, the handheld user can perform a scan from any field on the form, and the bar code will be placed in Field 2.

Repeating Forms

Bar code applications are useful for capturing large quantities of data quickly. Scanning a bar code is ergonomically more efficient than writing a text string. In the case of inventory applications, the goal is to minimize the number of pen taps that the handheld user has to enter. This includes copying data from one record to another if appropriate, and eliminating the need to tap the End button between records.

If most of the data on a form is the same for many records at a time, the **clone** statement can be used in a **scan:** event script to create a new record based on the existing record. Clone ends the current record, then creates a new record with the same values as the previous record. Fields that are not the same from one record to the next can be set to null on the new record.

Bar Code Clone Example 1: Creating a New Record for Each Bar Code Scanned

Bar Code Clone 1	
Date	10/21/06
Building Location	Bldg 1, Level 1
Room Number 1A	
Scan Bar Code	027700090018
Box Count for today:	1
[End] [Previous] [Next]	

Here is a form for taking inventory. Once the handheld user enters the date, building location and room number, these values do not need to change until the user goes to a new room. The script in Field 4 is:

```
scan:
if answer = null then
    answer = scandata
    $[BoxCount] = $[BoxCount] + 1
else
    clone
    beep
    answer = scandata
    $[BoxCount] = $[BoxCount] + 1
endif
```

On the very first record, the handheld user fills in fields 1, 2 and 3 and then scans the bar code. Since Field 4 is initially blank (null), the first part of the If...Then...Else statement is true, and the scanned bar code is placed in Field 4. The Box Count field is incremented by 1. (Important: To retain the box count from record to record, this field is set to Autodefault - see page 187.)

If the handheld user scans another bar code, Field 4 of the current record is not null. Thus the 'Else' part of the If...Then...Else...Endif statement runs. The **clone** statement ends the current record and makes an exact copy of the record, so that Fields 1, 2, and 3 are retained on the new record. The remainder of the script after the Clone statement runs on the new record, so that Field 4 of the new record is assigned the value of the second bar code. The beep statement is added to alert the handheld user to the fact that a new record has been created, and the Box Count field is incremented by one. The handheld user can continue creating new records each time a bar code is scanned. Tapping the End button after a scan will allow the user to exit the form.

Bar Code Clone Example 2: Re-Scanning into the Same Record if any Blank Fields

Bar Code Clone 2	
Date	5/20/03
Building Location	East Building B1
Room Number	43C
Scan Bar Code	0256983050
Enter Quantity	24
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

On this form, when the user first scans a bar code, a check is made to see if Fields 1, 2 and 3 have been filled in. If any of these fields is empty, a message is displayed and the **return** statement ends the script.

If Fields 1, 2 and 3 have been filled, a check is made to see if Field 4 already contains a bar code. If Field 4 is empty, the bar code is placed into Field 4, and the handheld user can then enter a Quantity in Field 5.

If the handheld user scans again, a check is made to see if a quantity has been entered into Field 5. If Field 5 is empty, it is assumed that the handheld user wants to stay on the same

record, and have the current bar code replace the previously scanned bar code. The user might want to scan into the same record if an incorrect bar code was scanned.

If Field 5 contains a Quantity, the next scan will clone the current record, and place the newly scanned bar code into Field 4 of the new record. Field 5 of the new record is cleared to receive the new quantity. The user can keep scanning to create new records, or tap End to end the form.

```

scan:
  if $1 = null then
    msgbox "Missing Date"
    return
  endif

  if $2 = null then
    msgbox "Missing Building location"
    return
  endif

  if $3 = null then
    msgbox "Missing Room number"
    return
  endif

  if $4 = null then
    $4 = scandata
    goto 5
  endif

  if $5 = null then
    $4 = scandata
    msgbox "You re-scanned into same
      record. Enter a quantity before
      you scan again."
    return
  else
    clone
    beep
    $4 = scandata
    $5 = null
  endif

```

Performing Reference Lookups

In some inventory applications, you may already have a list of inventory items on the PC, and you need to place these records on the handheld for the user to verify whether the listed inventories match the actual quantities in the warehouse.

There are two approaches that can be used with Pendragon Forms:

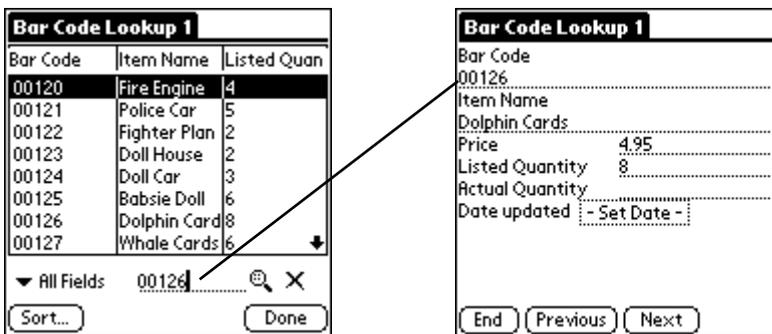
- a) You can put the records in a form on the handheld, and the user can select a record to update it.

- b) You can put the records in a reference form on the handheld, and the user can use another form to perform a lookup to the reference form and copy data from the reference form into the current form. The reference form can be just another form, or it can be a read-only reference form if you have more than 2000 records.

Example 1: Looking up Records within a Single Form

If you have a small enough number of items, you can create and freeze a Pendragon form, then import the inventory data into the Forms Manager database table that has been created to store records associated with that form. (See page 242 for importing data.) When the records are sent to the handheld, the user can tap the name of the form and then tap the Review button to see the list of records.

Tapping on the name of the form and then tapping the Review button will display the records on the Review screen. Scanning a bar code on the Review screen will display the corresponding record, assuming that all the bar codes are unique. The user can update the record.



If more than one record matches the bar code (that is, if the bar codes are not unique), then the user will see a list of matching records and can select one.

If the bar code scanner is not working, the user can write in the field at the bottom of the Review screen, and tap the magnifying glass icon to perform the filter to find the matching record(s).

When the user exits a record, he/she will have to tap the X icon to return to viewing all records.

Example 2: Looking up Records from a Reference form

Another approach to sending records to the handheld is to use two forms: one form acts as a reference form and contains, for example, a record for each inventory item. The second form is the form that the user fills in. When the user scans a bar code, a matching record in the reference form is copied into the current form. Since two forms are being used to perform the lookup, there are two steps involved: Step 1 is to create the reference form, and Step 2 is to create the form that is doing the lookup.

PALM OS

On Palm OS devices, there are two types of reference form that you can create:

- A regular form that is acting as a reference form.
- A read-only reference form. Typically you would use a read-only reference form if you have a lot of records (for instance, more than 2000 records). A read-only reference has more restrictions than a regular form (e.g: only Text and Numeric fields are allowed, and Text fields have to be set to the smallest possible Maximum Length), but can be fast for doing lookups. See page 127 for details on how to create a read-only reference form.

PPC

In Pendragon Forms 5.1, Pocket PC devices behave like Palm OS devices, with both reference forms and read-only reference forms.

In Pendragon Forms version 5.0, Pocket PC devices only support reference forms, not read-only reference forms.

Step 1: Create the reference form

First create the reference form that will store all the inventory records. As mentioned above, the form can be a regular form or it can be a read-only reference form.

Bar Code Lookup 2 (Ref)	
Item Bar Code	073852096507
Item Name	Hand Sanitizer
Quantity in Stock	20
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

Here the reference form contains three fields:

- Item Bar Code (the primary key field)
- Item Name
- Quantity in Stock.

The reference form can be frozen, and then populated with records. You can import data into the database table from an ASCII file (see page 242), or you can click on the name of the form in the Forms Manager, then click the Edit/View button, and manually enter the records. You can also put the form on a handheld and manually scan each item and fill in the remaining fields.

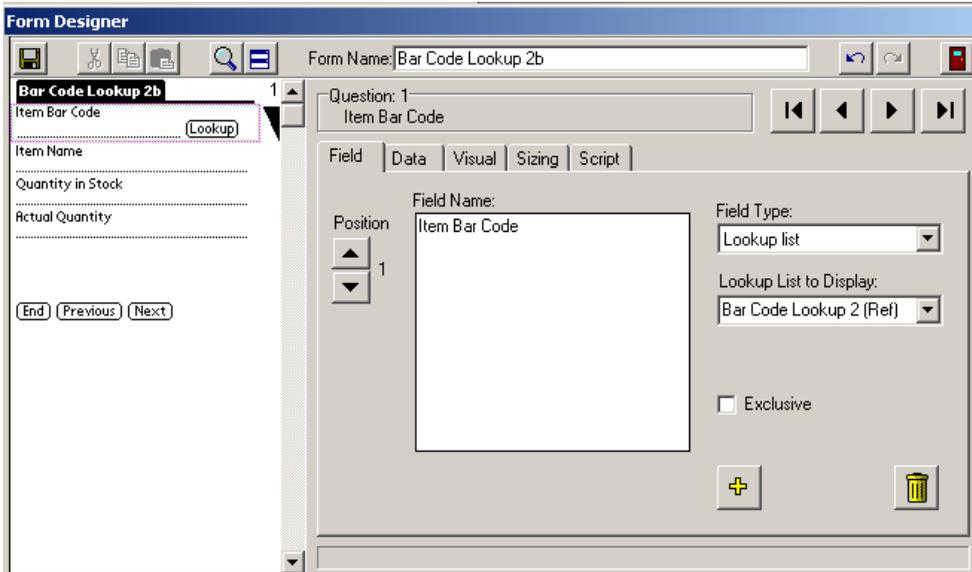
Step 2: Create the form to do the lookup to the reference form

The form that is going to perform the lookup to the reference form can be set up to do the lookup in two ways: programatically, when the user scans a bar code, and manually if the bar code scanner is not working.

For fields to copy from the reference form into the current form, the field names and field types have to be identical. The one exception is that a Text field on the reference form can copy into a Lookup List field on the current form, and this is how the manual lookup is made possible.

In the form shown below, the Item Bar Code field is a Lookup List field. In the Lookup List to Display field, instead of picking a Lookup List, type the name of the reference form.

If the bar code scanner is not working, the user can tap the Lookup button and see a list of records in the reference form. Selecting a record copies all the matching fields from the reference form to the current form. In this example, the matching fields are Item Bar Code, Item Name and Quantity in Stock, which match the reference form created on the previous page.



To make it possible to automatically perform the lookup to the reference form when the user scans a bar code, a **scan:** event script is added to field 1, the Item Bar Code field.

A **lookup...within** statement performs the lookup to the named reference form. The default is that a lookup...within statement does the lookup based on the **Display Key** field of the reference form. The default Display Key field is the first field on the form. If you do not want to perform the lookup based on the Display key field, use the **keycolumn** statement to select which column of the reference form should be used for the lookup.

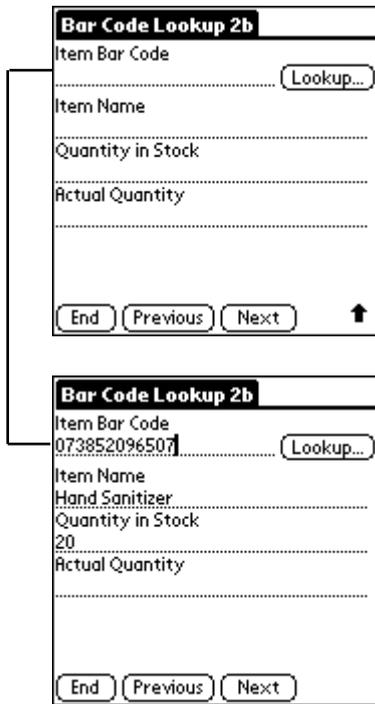
```
scan:
  if $1 = null then
    $1 = scandata
    lookup $1 within "Bar Code Lookup 2 (Ref)"
    goto 4
  endif

  if $4 = null then
    $1 = scandata
    $2 = null
    $3 = null
    lookup $1 within "Bar Code Lookup 2 (Ref)"
    msgbox "You re-scanned into same record.
      Enter a quantity before you scan again."
    return
  else
    clone
    $1 = scandata
    $2 = null
    $3 = null
    $4 = null
    lookup $1 within "Bar Code Lookup 2 (Ref)"
  endif
```

The above script is in field 1 of the form that is doing the lookup to the reference form. When the handheld user creates the first record, field 1 (the Bar Code field) is null. The script therefore places the bar code into field 1. The statement `lookup $1 within "Bar Code Lookup 2 (Ref)"` performs the lookup to the reference form named Bar Code Lookup 2 (Ref), and copies all fields that match from the reference form into the current form. This fills in the Item Name (field 2) and Quantity in Stock (field 3) for the corresponding Item Bar Code. The Goto statement ends the script and the handheld user can then fill in field 4, the Actual Quantity in Stock.

If the user does not fill in field 4, and instead scans another bar code, it is assumed that the user wants to overwrite the same record. If the user does fill in field 4 and then scans another bar code, it is assumed that the user wants to create a new record. The **clone** statement creates the new record, and another lookup...within statement looks up the new bar code against the reference form.

If the handheld user scans a bar code that is not in the reference form, the lookup...within statement will not find a match, and fields 2 and 3 will remain blank. The user can then enter this information manually.



With the bar code scanner, the user can repeatedly scan a bar code, find a match and update the actual quantity. Scanning again automatically creates a new record.

If the bar code scanner is not working, the user has to do a manual lookup. After finding a match and updating the actual quantity, the user has to manually tap the End button to end the record, and then tap the New button to create a new record to select again.

NOTE:

In this example, the user never updates the Quantity in Stock on the reference form. All the updates are being entered in the form that is doing the lookup. In this scenario, it is the responsibility of the administrator for the Forms program to update the reference form before the next inventory audit.

It is possible to programatically update the data in a reference form (but NOT a read-only reference form). See page 466 for an example.

Scanning Multiple Bar Codes into the Same Record

Here is a simple example of a form that places three different bar codes into three different fields. The first scanned bar code goes into field 3, the next into field 4, etc.

This example has no error checking to ensure that the correct bar code is scanned first. If different bar code symbologies were used, the **scantype** function could be used to verify that the correct bar code is placed in the correct field (see page 397).

Bar Code Multiples	
Date	2/17/05
Location	
Level 1	
Scan Barcode1	079298000511
Scan Barcode2	037000624059
Scan Barcode3	041163254220
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

```

scan:
  if $3 = null then
    answer = scandata
    return
  endif

  if $4 = null then
    $4 = scandata
    return
  endif

  if $5 = null then
    $5 = scandata
    return
  endif

```

Controlling which Bar Code Symbologies can be scanned

This section applies to:

Symbol SPT 1550, SPT 1800, PPT 8800, MC 50 Janam XP20 and XP30 Series	Socket Secure Digital Scan Card 3E	Grabba	Microvision Flic
Yes	Yes	No	No

There are various bar code standards, also called bar code symbologies. For instance, bar codes on grocery products often use a 10-character UPCA symbology.

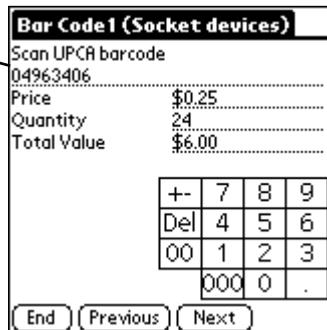
If you are using a specific bar code symbology, you can allow only that symbology to be scanned.

A script is used to set up which bar codes you want to allow. Typically an **open:** event script is used, as an open: event script will run whether a new record is being created or an existing record is being reviewed. You can also use an **enterscreen:** event if you want to allow different bar code symbologies on different screens of the form. (See examples starting on the next page.)

Example 1: Enabling only one bar code symbology (UPCA in this example)

```
open:
  scanner disable all
  scanner enable UPCA
```

This script would be used on a Symbol Technologies device (Palm OS or Pocket PC). All bar code symbologies are disabled, and then just UPCA bar codes are enabled.



```
open:
  socketscan UPCA 1
  socketscan UPCE 0
  socketscan CODE39 0
  socketscan I2OF5 0
```

This script would be used on a Palm OS device with a Socket Secure Digital Scan Card 3E. UPCA bar codes are enabled, and various other bar codes are disabled.

PALM OS

On Symbol Technologies SPT 1550 / SPT 1800 series Palm OS devices, you can also manually control which bar code symbologies are accepted via the bar code scanner hardware. Tap the Applications button and then tap the Diag icon (for Diagnostics). On the Diagnostics screen, tap the Bar Code Scanner option, and page down to page 2 of the Bar Code Scanner program. On the second screen of the Bar Code Scanner you can check a box for each bar code symbology that you want to allow. If you are not sure that the handheld users will remember to manually set the bar code scanner to allow specific symbologies, use a script instead.

Example 2: Disabling Preamble (Leading) and Checkdigit (Trailing) Characters on UPCA Bar Codes

The scripts here sets the scanner to omit the prefix and suffix characters of the bar code. Once set, these settings apply to all fields and to all records, unless another script changes these settings.

Symbol Technologies devices:

```
open:
  scanner disable all
  scanner enable UPCA
  scanner no_preamble UPCA
  scanner no_checkdigit UPCA
```

Bar Code2 (SPT/PPT devices)

Scan a UPCA bar code.
Note: the prefix (leading) and suffix (trailing) characters of the bar code will not be included in the scan.

3700062405

Name of Item:
Box of tissues

Quantity on Shelf
3

End Previous Next

The script above applies to Symbol Technologies SPT and PPT devices, and to Janam XP20 and XP30 Series devices. In this example, only UPCA bar codes are being allowed to be scanned, and the prefix (preamble, or leading character) and suffix (checkdigit or trailing character) of UPCA bar codes are not being recorded in the scan.

Socket Secure Digital Scan Card 3E:

```
open:
  socketscan UPCA 1
  socketscan UPCA_PREAMBLE 0
  socketscan TRANSMIT_UPCA_CHECKDIGIT 0
```

On handheld devices with Socket Secure Digital Scan Cards, the script above enables UPCA bar codes and then disables the prefix (preamble, or leading character) and suffix (checkdigit or trailing character) of UPCA bar codes.

Example 3: Converting UPCE to UPCA Bar Codes

These sample scripts allow UPCE and UPCA bar codes to be scanned. Any UPCE bar codes are converted to UPCA format.

Symbol Technologies devices:

```
open:
  scanner disable all
  scanner enable UPCA
  scanner enable UPCE
  scanner UPCETOUPCA enable
```



As with all bar code settings, once a conversion is switched on, it applies globally to all fields, all records, all applications. To switch the conversion from UPCE to UPCA off, and to re-enable the scanning of all bar code symbologies, you can add a validate: script that will run when the record is closed.

```
validate:
  scanner UPCETOUPCA disable
  scanner enable all
```

Socket Secure Digital Scan Card 3E:

On handheld devices with Socket Secure Digital Scan Cards, the script to convert UPCE to UPCA codes might look like this:

```
open:
  socketscan UPCA 1
  socketscan UPCE 1
  socketscan CONVERTUPC_ETOA 1
```

Checking the Type of Bar Code Scanned

The **scantype** function stores the bar code symbology of the most recently scanned bar code. This can be used to check the type of bar code that you are scanning.

This section applies to:

Symbol SPT 1550, SPT 1800, PPT 8800, MC 50 Janam XP20 and XP30 Series	Socket Secure Digital Scan Card 3E	Grabba	Microvision Flic
Yes	Yes (Pocket PC only)	Yes	No

Barcode Scan Type
Scan Barcode 04963406
Type of Bar Code UPCE

In the example shown at left, scanning a bar code in any field places the bar code into Field 1, and displays the bar code symbology in Field 2.

The script in Field 1 is:

```
scan:
  answer = scandata
  $2 = scantype
```

Valid bar code symbologies that the Scantype function can receive are:

CODE39	I2OF5	MSI_PLESSEY
UPCA	CODABAR	UPCE1
UPCE	CODE128	
EAN13	CODE93	BOOKLAND_EAN
EAN8	TRIOPTIC39	ISBT128
D25	UCC_EAN128	COUPON

Bar Codes in Different Fields 1				
Item UPCA code	7855502254			
Item ISBN Number	9780764546518			
Quantity to Order	250			
	+ -	7	8	9
	Del	4	5	6
	00	1	2	3
		000	0	.
End	Previous	Next		

```
open:
  scanner disable all
  scanner enable UPCA
  scanner enable EAN13

scan:
  result = scantype
  if result = UPCA then
    answer = scandata
  else
    $2 = scandata
  endif
```

In the form shown above, an **open:** script in Field 1 configures the bar code scanner to accept only UPCA and EAN13 symbologies. The handheld user can scan into any field, and if the bar code is a UPCA code, the scan: event script places the bar code in Field 1. If the bar code is EAN13, the script places the bar code in Field 2.

Troubleshooting Bar Codes

Check the following if you are not having success with scanning a barcode:

Troubleshooting Tips	Symbol SPT 1550 SPT 1800 Janam XP	Symbol PPT 8800 MC 50	Socket Secure Digital Scan Card 3E	Grabba	Microvision Flic
Verify that the scanner is enabled and the bar code that you are scanning is enabled in the Diagnostics app (on Symbol devices) or the Janam app (on Janam devices).	√				
Check that the SocketScan application is running on the device and that the correct bar code symbologies have been enabled.			√ (Pocket PC devices)		
Check that the handheld has sufficient battery power. If the handheld battery is too low, the scanner will not work.	√	√	√		
If the external scanner seems slow, replace/recharge the batteries in the scanner.				√	√
Check that the Flic Cordless has been paired with the Bluetooth-enabled handheld.					√
After a period of inactivity, a Flic scanner may take up to 15 seconds to place a bar code in a field.					√
Has a script on your form altered how the scanned bar code is displayed? Bar code scanner settings are global, and if set in a script, will apply to all fields unless changed by another script.	√	√	√		
Is the scanner being held too close to the bar code? Sometimes moving the scanner further away from the bar code widens the scanner beam.	√	√	√	√	√
If the surface material of the bar code is shiny, hold the scanner at an angle to the bar code for the scan to register.	√	√	√	√	√

16. Printing from the Handheld

You can print via the handheld serial port to a serial printer, print via the infrared port to a printer with infrared capability, or print via BlueTooth to a BlueTooth-enabled printer.

Serial Port Printing

The capability exists to print to a serial RS-232 printer, including the Monarch 6015 printer, which is designed for use with the Palm III and Symbol Technologies SPT family of handheld devices.

Pendragon Forms defaults to serial communications parameters of 9600-N-8-1, meaning 9600 baud, No parity, 8 data bits and 1 stop bit.

If your serial printer uses different serial communications settings - (for example, the Monarch 6015 uses 19200-N-8-1) - then you can either use additional printer driver software on the handheld, or you can send data to the printer via a Transmit Serial scripting statement.

Infrared or BlueTooth Printing

Printing via the handheld's infrared port or BlueTooth port requires additional printer driver software on the handheld (see below). You can print to most infrared-capable or BlueTooth-enabled printers.

Printer Driver Software

Pendragon Forms supports these printer drivers:

PALM OS

- PalmPrint™ software from www.stevenscreek.com
- PrintBoy software (version 7) from www.bachmannsoftware.com

PPC

- PrintBoy software (version 7) from www.bachmannsoftware.com

Stevens Creek Software and Bachmann Software offer demo versions of their printer driver software, that you can download from their respective Web sites.

Refer to the printer driver documentation for installation instructions.

Using Scripts to Control Printer Output

Pendragon Forms uses scripts to allow you to control how a form prints directly from the handheld.

All the scripting event procedures, statements and functions that are mentioned in this chapter are described in detail in Chapter 13, *Scripting Reference*, starting on page 245.

Creating scripts for printing is a two-stage process:

1. Create **report:** event scripts to specify what the printer output will look like. If you are printing from a parent form and a subform, you may also need **report2:** event scripts.
2. Create a Button field with a **click:** event script with a **printreport** statement to trigger when the printing occurs.

It is best to add a button field to your form design, to allow the handheld user to control printing by tapping a button. That way the handheld user can check that the printer is switched on and has sufficient paper before attempting to print. (There is alternatively a Menu option for printing if you are in Record View, but this option does not work with the PalmPrint printer driver.)

Step 1: Create report: and report2: event scripts

A **report:** event script specifies the layout of the printer output from the handheld.

Pendragon Forms controls printer output by appending lines of characters to a file in the handheld's memory. Every line is composed of characters with fixed width. Most letter-sized printers - (printing on 8.5" x 11" paper) - will use a Courier font that provides 80 characters per line (also called an 80-column format), and 54 lines per page. Other printers, such as receipt printers, may provide fewer characters per line but have no specific number of lines if the paper is dispensed from a roll. You will need to refer to your printer specifications to know exactly how many characters you can use per line, and whether there is a specific number of lines per page.

If you imagine the paper of your report as a grid that is 80 columns wide and 54 rows (lines) long, the purpose of **report:** events is to start from the first column of the first line, and to specify what characters to print on each line, starting at what column location.

New in Pendragon Forms 5.1: If you are using PrintBoy software (version 7) from Bachmann Software, you can use HTML tags in **report:** events to define the layout of your report, instead of specifying the report layout in terms of rows and columns. See page 408.

report: events run in sequence, meaning that a **report:** event in field 1 of your form will run first, followed by a **report:** event in field 2, etc. Since each script cannot exceed more than 2000 characters, it is a good idea to break up the layout of the report into several **report:** events.

If you are printing a report for a parent form and a subform, all the **report:** events on the parent form will run, followed by all the **report:** events on each of the subform records that is associated with the parent record. If there is additional information on the parent form that you want to print after printing the subform records, use **report2:** events instead of **report:** events on the remainder of the parent record.

Several scripting statements are used in **report:** events and **report2:** events to enable you to specify the report layout. These statements are described in detail in Chapter 13, *Scripting Reference*, which starts on page 245.

These commands are used when printing with PalmPrint software. If you are using PrintBoy software and you want to print forms designed for a fixed-width font, these commands can also be used.

- printleft** *value column* Prints left-justified text in the value specified, starting in the column specified.
- printright** *value column* Prints right-justified text in the value specified, ending at the specified column.
- printline** Sends the current line to the end of the current report in the handheld's memory. A **printline** statement is needed to actually print a line of text. To create a blank line, use two printline statements in a row.
- printmemo** *value "startcolumn:endcolumn:maxlines"*
Used to word wrap a block of text in a Text field, in the case where a Text field contains more than one line.
- printnewpage** Sends the current page to the printer, then begins a new page. To know when to begin a new page, use the **pagenum** and **linenum** functions (see page 259).
- format** *value {date | time | datetime | currency | fixed}*
Date, Time and Date & Time fields are all stored internally as the number of seconds since 1/1/1904. Currency fields are stored as a number of cents. In order to print these types of fields, they need to be formatted first. The format command stores the formatted value in the **result** variable.

The examples that follow in this chapter will illustrate the use of these scripting statements.

If you are using PrintBoy software and you want to use HTML tags in **report:** events to specify the layout of your report, these are the scripting statements that can be used in the **report:** and **report2:** events.

printhtml *value* Sends the text string within the double quotation marks to be printed by the PrintBoy HTML engine.
The contents of *value* can be

- a) Text and any HTML tags supported by PrintBoy. Text must be enclosed in double quotes (").
- b) A value in a field, such as \$12 for the value in field 12.
- c) A variable, such as the **result** variable.

See page 306 for examples of *value*, and see page 408 for a sample script. Refer to the PrintBoy documentation for details on the HTML statements supported by PrintBoy.

printnewpage Sends the current page to the printer, then begins a new page.
To know when to begin a new page, use the **pagenum** and **linenum** functions (see page 259).

format *value* {**date** | **time** | **datetime** | **currency** | **fixed**}

Date, Time and Date & Time fields are all stored internally as the number of seconds since 1/1/1904. Currency fields are stored as a number of cents. In order to print these types of fields, they need to be formatted first. The format command stores the formatted value in the **result** variable.

Step 2: Create a Button field with a printreport statement

To run the **report:** and **report2:** scripting events, your form will need a Button field with a **click:** event that contains a **printreport** statement.

When printing via PalmPrint, or if you are choosing to print to a fixed-width font in PrintBoy to maintain compatibility with prior versions of Pendragon Forms, only the **printreport** statement is needed in the **click:** event.

printreport *name* {**serial** | **IR**}

Triggers **report:** and **report2:** events.

The *name* specified is placed in the `reportname` function, and can be used in an `if...then...else...endif` statement in the **report:** script, to print different types of reports. If you only have one report to print, you can use the word **VIA** as the *name*.

serial means printing via the handheld serial port, and **IR** means printing via the handheld infrared port. **IR** is also used for printing via BlueTooth.

The script in the Button field for a report printed with a fixed-width font might look like:

```
click:
  printreport via IR
```

If you are using PrintBoy software and are choosing to use HTML tags in your **report:** event, then there are some additional statements that may be needed in the **click:** event script.

setparam allowhtml {**0** | **1**}

The **setparam allowhtml** statement determines whether the layout of a report specified in a **report:** or **report 2:** event is interpreted as HTML or not. The default setting is **setparam allowhtml 0** which means that statements in a **report:** event are not considered to be HTML.

If you are using PrintBoy software and want to use HTML tags to specify the layout of your report instead of specifying rows and columns, you will need to switch on the use of HTML by setting **setparam allowhtml 1**. This must be done in the **click:** event before the **printreport** statement runs. For example:

```
click:
  setparam allowhtml 1
  printreport via IR
```

```
setparam printhtmlheader "header"  
setparam printhtmlheader "footer"
```

When printing via PrintBoy, Pendragon Forms generates HTML source code from the statements in report: and report2: events. The HTML document that is generated is then printed by the PrintBoy HTML engine. By default, the header of this HTML document just sets the HTML to preformatted mode, which is:

```
<HTML><BODY><PRE>
```

By default, the matching footer closes these tags:

```
</PRE></BODY></HTML>
```

The preformatted mode is a fixed-width mode (e.g. *Courier* font). If you had report: events that you used in Forms 5.0, keep the default preformatted mode to avoid having to modify your report: event scripts in Forms 5.1.

You can choose to override the default HTML header and footer settings with the **setparam printhtmlheader** *header* and **setparam printhtmlfooter** *footer* statements. The *header* and *footer* parameters are HTML headers and footers that are supported by the PrintBoy software.

Since double quotation marks are used to define the header and footer, any double quotation marks within the HTML text have to be preceded by an escape character - a backslash \ character. For example, to set an HTML header that sets the font to "Arial", the commands would be:

```
setparam printhtmlheader "<HTML><BODY><FONT FACE=\"Arial\">"  
setparam printhtmlfooter "</FONT></BODY></HTML>"
```

The HTML headers and footers have to be specified before the report: event runs. A click: event script is typically used to trigger printing, so the setparam printhtmlheader and setparamprinthtmlfooter statements are placed in the click: event before the printreport statement. For example:

click:

```
setparam printhtmlheader "<HTML><BODY><FONT FACE=\"Arial\">"  
setparam printhtmlfooter "</FONT></BODY></HTML>"
```

```
setparam allowhtml 1  
printreport via IR
```

Printing a Single Form

Here is an example of an invoice form that can be printed directly from the handheld. Three **report:** events in Fields 1, 2 and 3 are used to control the printer output. The script that actually initiates the printing is in a **click:** event in the last field on the form - a Print button. See page 407 for a sample of the printer output.

If you are using PrintBoy software, then instead of laying out your report in terms of columns on a page, you may prefer to use HTML tags - see page 408.

2/11/00	
1	Date that call wa 2/11/00
2	Date and Time o 2/14/00
3	Customer Name Lara Rodriguez
4	Customer Addre 24 Sunset Circle
5	Customer Addre
6	City Highland Park
7	State IL
8	Zip Code 60035
9	Phone 847-555-0184
10	Problem Report Heating Compon
11	Start Time 10:30 am

End [Navigation icons]

This is the script in Field 1:

```
report:
printleft "SuperQuick Maintenance" 30
printline
printleft "22 Cherry Street, Mytown, IL 60048" 20
printline
printline
printline
printleft "Date of Customer Work Request:" 1
format $1 date
printleft result 35
printline
printleft "Customer Name:" 1
printleft $3 20
printline
printleft "Customer Address:" 1
printleft $4 20
printline
if $5 <> null then
printleft $5 20
printline
endif
printleft $6 20
printline
printleft $7 20
printleft $8 24
printline
printleft $9 20
printline
printline
printleft "Problem Reported:" 1
printline
printleft $10 10
printline
printline
```

2/11/00	
12	Air Intake Valve <input checked="" type="checkbox"/>
13	Work performe ↓Air Intake Val
14	Replacement Co \$0.00
15	Filter Checked? <input checked="" type="checkbox"/>
16	Work Performe ↓Filter was Repl
17	Replacement Co \$25.00
18	Ignition Switch C <input checked="" type="checkbox"/>
19	Work Performe ↓Ignition Switch
20	Replacement Co \$36.50
21	Heater Coil Chec <input checked="" type="checkbox"/>
22	Work Performe ↓No Work Done

End [Navigation icons]

2/11/00	
25	End Time 11:52 am
26	Work Performe Sarah Jane
27	Time elapsed (a 1.5
29	Labor Rate per h \$50.00
30	Total Replaceme \$61.50
31	Total Labor Cost \$75.00
32	Tax Rate on Par 6.5
33	Tax on Parts \$4.00
34	Total Bill Amoun \$140.50
35	Sign Signed
36	Print (Print)

End [Navigation icons]

The script in the Button field to trigger the report events is:

```
click:
printreport via IR
```

This is the script in Field 2:

```
report:
printleft "Date of Technician's Visit:" 1
format $2 datetime
printleft result 30
printline
printline
printline
printleft "Item" 1
printleft "Checked?" 20
printleft "Work Performed" 30
printright "Replacement Cost" 80
printline
printline
printleft "Air Intake Valve" 1
printleft $12 24
printleft $13 30
format $14 fixed
printright result 80
printline
printline
printleft "Filter" 1
printleft $15 24
printleft $16 30
format $17 fixed
printright result 80
printline
printline
printleft "Ignition Switch" 1
printleft $18 24
printleft $19 30
format $20 fixed
printright result 80
printline
printline
printleft "Heater Coil" 1
printleft $21 24
printleft $22 30
format $23 fixed
printright result 80
printline
printline
printleft "Technician's Comments:" 1
printline
printleft $24 10
printline
printline
```

This is the script in Field 3:

```
report:
printline
printline
printleft "Technician Name:" 1
printleft $26 18
printleft "Hours Worked:" 38
printleft $27 53
printleft "Labor Cost:" 57
format $31 currency
printright result 80
printline
printline
printleft "Total Replacement Parts Cost" 39
format $30 currency
printright result 80
printline
printleft "Tax rate " 39
printleft $32 48
printleft "% on Parts" 53
format $33 currency
printright result 80
printline
printline
printleft "Total Amount to Pay:" 39
format $34 currency
printright result 80
printline
printline
printleft "Customer Signature: _____"
20
printline
```

Note: Signature fields do not print out the signature that the customer enters on the handheld. Instead, you can print text on a report to add a place on the report where the customer can sign on paper in addition to signing on the handheld.

Sketch fields and Image fields also do not print.

Here is a sample printout:

SuperQuick Maintenance 22 Cherry Street, Mytown, IL 60048			
Date of Customer Work Request:	2/11/00		
Customer Name:	Lara Rodriguez		
Customer Address:	24 Sunset Circle		
	Highland Park		
	IL 60035		
	847-555-0184		
Problem Reported:	Heating Component seems broken. Hearing a loud noise and no heat.		
Date of Technician's Visit:	2/14/00		
Item	Checked?	Work Performed	Replacement Cost
Air Intake Valve	Y	Air Intake Valve Cleaned	0.00
Filter	Y	Filter was Replaced	25.00
Ignition Switch	Y	Ignition Switch Replaced	36.50
Heater Coil	Y	No Work Done - Heater Coil OK	0.00
Technician's Comments:	Left unit in working condition. All replaced parts under warranty.		
Technician Name: Sarah Jane	Hours Worked: 1.5	Labor Cost:	\$75.00
	Total Replacement Parts Cost:		\$61.50
	Tax rate 6.5 % on Parts		\$4.00
	Total Amount to Pay:		\$140.50
Customer Signature:	_____		

Printing a Single Form using PrintBoy and HTML

New in Forms 5.1.

Bachmann Software's PrintBoy 7.0 printer driver software can be used to print Pendragon Forms records from a Palm OS 5.4 device or a Windows Mobile 5 for Pocket PC device.

PrintBoy 7.0 supports HTML printing, meaning that the layout of the printed report is specified using HTML. Pendragon Forms 5.1 contains a new scripting statement, the PRINTHTML statement, that allows you to use HTML tags in a report: or report2: event instead of having to specify column numbers using printleft and printright commands.

Here is the same form that was shown on page 405, but this time using HTML tags to create the report layout. Three **report:** events in Fields 1, 2 and 3 are used to control the printer output. The script that actually initiates the printing is in a **click:** event in the last field on the form - a Print button. See page 411 for a sample of the printer output.

Also refer to the PrintBoy documentation for information on which HTML tags are supported by PrintBoy 7.0.

1/3/07	
1	Date of call 1/3/07
2	Date of Field Tec 1/3/07
3	Customer Name Francine Franajo
4	Customer Address 5 Oak Ridge Drive
5	Customer Address
6	City Northbrook
7	State IL
8	Zip Code 60036
9	Phone 847-555-0001
10	Problem Report The heating doe
11	Start Time 1:22 pm
End	⬆ ⬅ ➡ ⬇

1/3/07	
12	Air Intake Valve <input checked="" type="checkbox"/>
13	Work performe ↕ Air Intake Val
14	Replacement Co \$0.00
15	Filter Checked? <input checked="" type="checkbox"/>
16	Work Performe ↕ Filter was Repl
17	Replacement Co \$25.00
18	Ignition Switch C <input checked="" type="checkbox"/>
19	Work Performe ↕ Ignition Switch
20	Replacement Co \$250.00
21	Heater Coil Chec <input checked="" type="checkbox"/>
22	Work Performe ↕ Heater Coil Re
End	⬆ ⬅ ➡ ⬇

1/3/07	
25	End Time 3:28 pm
26	Work Performe 2FSW4AN28Y
27	Time elapsed (a 2.25
29	Labor Rate per h \$50.00
30	Total Replaceme \$374.00
31	Total Labor Cost \$112.50
32	Tax Rate on Par 6.5
33	Tax on Parts \$24.31
34	Total Bill Amount \$510.81
35	Sign Signed.
36	Print (Print)
End	⬆ ⬅ ➡ ⬇

This is the script in Field 1:

```
report:
PRINTHTML "<P ALIGN='CENTER'\><H1>SuperQuick Maintenance</
H1>"
PRINTHTML "22 Cherry Street, Mytown, IL 60048</P>"
PRINTHTML "<BR>"
PRINTHTML "<BR>"

PRINTHTML "<TABLE WIDTH='80%\> BORDER='0\> ALIGN='LEFT'\>"
PRINTHTML "<TR>"
PRINTHTML "<TD WIDTH='30%\>Date of Customer Work Request:
</TD>"
format $1 date
PRINTHTML "<TD WIDTH='70%\>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Customer Name: </TD>"
PRINTHTML "<TD>"
PRINTHTML $3
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD VALIGN='TOP'\>Customer Address: </TD>"
PRINTHTML "<TD ALIGN='LEFT'\>"
PRINTHTML $4
PRINTHTML "<BR>"
if $5 <> null then
PRINTHTML $5
PRINTHTML "<BR>"
endif
PRINTHTML $6
PRINTHTML "<BR>"
PRINTHTML $7
PRINTHTML " "
PRINTHTML $8
PRINTHTML "<BR>"
PRINTHTML $9
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Problem Reported: </TD>"
PRINTHTML "<TD>"
PRINTHTML $10
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Date of Technician's Visit: </TD>"
format $2 datetime
PRINTHTML "<TD>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"
PRINTHTML "</TABLE>"
```

This is the script in Field 2:

```
report:
PRINTHTML "<BR>"

PRINTHTML "<TABLE WIDTH='100%\>
BORDER='0'\>"
PRINTHTML "<TR>"
PRINTHTML "<TD WIDTH='30%\><B>Item</
B></TD>"
PRINTHTML "<TD WIDTH='15%\>
"><B>Checked?</B></TD>"
PRINTHTML "<TD WIDTH='40%\><B>Work
Performed</B></TD>"
PRINTHTML "<TD WIDTH='15%\> ALIGN='
RIGHT'\><B>Replacement Cost</B></TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Air Intake Valve</TD>"
PRINTHTML "<TD>"
PRINTHTML $12
PRINTHTML "</TD>"
PRINTHTML "<TD>"
PRINTHTML $13
PRINTHTML "</TD>"
format $14 fixed
PRINTHTML "<TD ALIGN='RIGHT'\>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Filter</TD>"
PRINTHTML "<TD>"
PRINTHTML $15
PRINTHTML "</TD>"
PRINTHTML "<TD>"
PRINTHTML $16
PRINTHTML "</TD>"
format $17 fixed
PRINTHTML "<TD ALIGN='RIGHT'\>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Ignition Switch</TD>"
PRINTHTML "<TD>"
PRINTHTML $18
PRINTHTML "</TD>"
PRINTHTML "<TD>"
PRINTHTML $19
PRINTHTML "</TD>"
format $20 fixed
PRINTHTML "<TD ALIGN='RIGHT'\>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"
```

This is the script in Field 2, con't:

```
PRINTHTML "<TR>"
PRINTHTML "<TD>Heater Coil</TD>"
PRINTHTML "<TD>"
PRINTHTML "$21"
PRINTHTML "</TD>"
PRINTHTML "<TD>"
PRINTHTML "$22"
PRINTHTML "</TD>"
format $23 fixed
PRINTHTML "<TD ALIGN='RIGHT'>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"
PRINTHTML "</TABLE>"

PRINTHTML "<P>Technician's Comments:
<BR>"
PRINTHTML "$24"
PRINTHTML "</P><BR>"
```

This is the script in Field 3:

```
report:
PRINTHTML "<P>Technician Name: "
PRINTHTML "$26"
PRINTHTML "<BR>"
PRINTHTML "Hours Worked: "
PRINTHTML "$27"
PRINTHTML "</P>"

PRINTHTML "<TABLE WIDTH='50%' BORDER='0' ALIGN='RIGHT'>"
PRINTHTML "<TR>"
PRINTHTML "<TD WIDTH='50%' ALIGN='LEFT'>Labor Cost:</TD>"
format $31 currency
PRINTHTML "<TD WIDTH='50%' ALIGN='RIGHT'>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Total Replacement Parts Cost:</TD>"
format $30 currency
PRINTHTML "<TD ALIGN='RIGHT'>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Tax rate "
PRINTHTML "$32"
PRINTHTML "% on Parts</TD>"
format $33 currency
PRINTHTML "<TD ALIGN='RIGHT'>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"

PRINTHTML "<TR>"
PRINTHTML "<TD>Total Amount to Pay:</TD>"
format $34 currency
PRINTHTML "<TD ALIGN='RIGHT'>"
PRINTHTML result
PRINTHTML "</TD>"
PRINTHTML "</TR>"
PRINTHTML "</TABLE>"

PRINTHTML "<BR>"
PRINTHTML "<P>Customer Signature: _____
_</P>"
```

1/3/07		
25	End Time	3:28 pm
26	Work Performe	2FSW4AN28Y
27	Time elapsed (a	2.25
29	Labor Rate per h	\$50.00
30	Total Replaceme	\$374.00
31	Total Labor Cost	\$112.50
32	Tax Rate on Par	6.5
33	Tax on Parts	\$24.31
34	Total Bill Amoun	\$510.81
35	Sign	Signed.
36	Print	(Print)

End ⬆ ⬅ ➡ ➤

This is the script in the Button field, Field 36:

```
click:
setparam printhtmlheader "<HTML><BODY><FONT
FACE='Arial'>"

setparam printhtmlfooter "</FONT></BODY></HTML>"

setparam allowhtml 1

printreport via IR
```

Here is a sample printout:

SuperQuick Maintenance

22 Cherry Street, Mytown, IL 60048

Date of Customer Work Request: 1/3/07

Customer Name: Francine Frangipani

Customer Address: 5 Oak Ridge Drive
Northbrook
IL 60036
847-555-0001

Problem Reported: The heating does not appear to be working. No air is blowing and no heat is emanating from the vents.

Date of Technician's Visit: 1/3/07

Item	Checked?	Work Performed	Replacement Cost
Air Intake Valve	Y	Air Intake Valve Cleaned	0.00
Filter	Y	Filter was Replaced	25.00
Ignition Switch	Y	Ignition Switch Replaced	250.00
Heater Coil	Y	Heater Coil Replaced	99.00

Technician's Comments:
Ignition switch and heater coil were not working. Replaced both and air filter also, and hot air is now blowing normally.

Technician Name: 2FSW4AN28Y
Hours Worked: 2.25

Labor Cost:	\$112.50
Total Replacement Parts Cost:	\$374.00
Tax rate 6.5 % on Parts	\$24.31
Total Amount to Pay:	\$510.81

Customer Signature: _____

Printing a Parent and Subform

If you are using a parent form and a subform, a **printreport** statement on the parent will trigger all the **report:** events on the parent to run, followed by all the **report:** events on the subform. (The parent form must have a Subform field for this to work.) If you have additional information that you want to print from the parent after the subform records have printed, you can use a **report2:** event on the parent.

2/15/00		
1	Date of order	2/15/00
2	Select Customer	(Select)
3	First name:	P.
4	Last name:	McOtter
5	Company:	Chopper Rescue
6	Address:	71 Seashore Vie
7	City:	Buffalo Grove
8	State:	IL
9	Zip:	60089
10	Tel:	847-555-4321
11	Items Ordered	<input type="checkbox"/> <input type="checkbox"/>
End		

In this example, a parent form is used for customer information, and a subform is used to record individual items being ordered by a customer.

2/15/00		
1	Date of order	2/15/00
2	First name:	P.
3	Last name:	McOtter
4	Company:	Chopper Rescue
5	Item	Toy sea otter
6	Price	\$20.00
7	Quantity	1
8	Item Total	\$20.00
End		

2/15/00		
5	Company:	Chopper Rescue
6	Address:	71 Seashore Vie
7	City:	Buffalo Grove
8	State:	IL
9	Zip:	60089
10	Tel:	847-555-4321
11	Items Ordered	<input type="checkbox"/> <input type="checkbox"/>
12	Calculate Order	(Total)
13	Order Total	\$95.00
14	Customer Signat	Signed
15	Print Order	(Print)
End		

2/15/00		
1	Date of order	2/15/00
2	First name:	P.
3	Last name:	McOtter
4	Company:	Chopper Rescue
5	Item	Toy polar bear
6	Price	\$15.00
7	Quantity	1
8	Item Total	\$15.00
End		

When the handheld user finishes entering all the items ordered, he/she returns to the parent form to add up a total and print an invoice.

2/15/00		
1	Date of order	2/15/00
2	First name:	P.
3	Last name:	McOtter
4	Company:	Chopper Rescue
5	Item	Dinosaurs
6	Price	\$5.00
7	Quantity	12
8	Item Total	\$60.00
End		

```
report:
printline
printleft "Amazing Stuff" 35
printline
printleft "9400 W. Deerfield Rd., Suite 15, Libertyville IL 60048" 20
printline
printleft "847-555-4567" 35
printline
printline
printline
printleft "Order Date:" 1
format $1 date
printleft result 12
printline
printleft "Customer Name:" 1
result = $3 & " "
result = result & $4
printleft result 16
printline
printleft "Company:" 1
printleft $5 16
printline
printleft "Address:" 1
printleft $6 16
printline
printleft $7 16
printline
printleft $8 16
printleft $9 28
printline
printleft "Phone:" 1
printleft $10 8
printline
printline
printline
printleft "Item Ordered" 1
printleft "Qty" 20
printleft "Price" 25
printright "Total" 50
printline
printline
```

A **report:** event script in field 1 of the form design is used to set up the 'header' of the printed report.

After the 'header' is printed, all of the subform records are printed using a **report:** event on the subform, as shown on the next page.

```

report:
if linenum >= 48 then
printline
printline
result = "Page " & pagenum
printleft result 1
printline
printnewpage
format $1 date
printleft result 1
printline
printleft "Customer Name:" 1
result = $2 & " "
result = result & $3
printleft result 16
printline
printleft "Company:" 1
printleft $4 16
printline
printline
printleft "Item Ordered" 1
printleft "Qty" 20
printleft "Price" 25
printright "Total" 50
printline
printline
printline
endif

printleft $5 1
printleft $7 20
format $6 fixed
printright result 30
format $8 fixed
printright result 50
printright pagenum 59
printright linenum 69
printline
printline

```

This is the **report:** event in field 1 of the subform.

The `if linenum >= 48 then` statement checks whether the printed report has passed line 48. If yes, then the page number is printed at the bottom of the page.

The `printnewpage` statement then starts a new page, and some header information such as the Customer Name and Company name are repeated at the top of the new page. The headers for the columns Item Ordered, Qty, Price and Total are also repeated on the new page.

This section is the main part of of the printed report for the subform. It prints the actual Item, Quantity, Price and Total from individual subform records.

To print the 'footer' of the report after all the subform records have been printed, a **report2:** event is used in field 2 of the parent form. The **report2:** event only runs after all the **report:** events on the subform records have been run.

```

report2:
printline
printleft "Order Total:" 30
format $13 currency
printright result 50
printline
printline
printline
printline
printleft "Customer Signature: _____"
8
printline

```

Here is what the printed report looks like:

Amazing Stuff				1
9400 W. Deerfield Rd., Suite 15, Libertyville IL 60048				
847-555-4567				
Order Date: 2/15/00				2
Customer Name: P. McOtter				
Company: Chopper Rescue				
Address: 71 Seashore View				
Buffalo Grove				
IL 60089				3
Phone: 847-555-4321				
Item Ordered	Qty	Price	Total	
Dinosaurs	12	5.00	60.00	
Toy polar bear	1	15.00	15.00	
Toy sea otter	1	20.00	20.00	
Order Total:			\$95.00	
Customer Signature: _____				

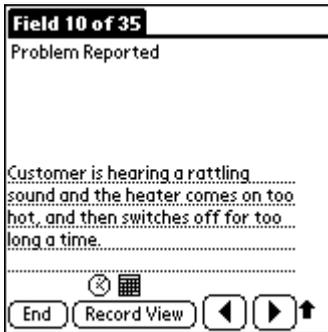
Section 1 was printed with the **report:** event in Field 1 of the parent form.

Section 2 was printed with the **report:** event in Field 1 of the subform.

Section 3 was printed with the **report2:** event in Field 2 of the parent form.

Printing a Block of Text

A Text field can contain up to 255 characters, and can be set to contain up to 2000 characters. By default, printing a Text field will only fill the current line, and will then truncate the remainder of the field. In order to accommodate printing the full contents of a Text field, you can use the **printmemo** statement in a **report:** event.



Here is what the printmemo statement might look like:

```
report:
printline
printleft "Problem Reported:" 1
printline
printmemo $10 "10:70:3"
printline
printline
```

The format of the **printmemo** statement is:

```
printmemo value "startcolumn:endcolumn:maxlines"
```

In the sample script shown above, the statement

```
printmemo $10 "10:70:3"
```

prints Field 10, starting in column 10 and ending in column 70, for a maximum of 3 lines.

Sample report output is shown below.

```
Problem Reported:
  Customer is hearing a rattling sound and the heater comes on
  too hot, and then switches off for too long a time.
```

Printing a Multi-Selection List Field

Multi-Selection fields are a special case in Pendragon Forms, because internally these fields are not stored as text, they are stored as a binary number, with each bit representing one of the possible selections in the list.

In this field, there are 6 options in a Multi-Selection list, and so six bits are used to store the value of each checkbox in the list:

Bit position: 32 16 8 4 2 1
 Option: Orange Purple Green Yellow Blue Red

If Blue, Green and Purple are selected, the internal number is stored in binary as: 011010.

The AND operator can be used in a script to identify if a bit in a Multi-Selection field is set. For example, in the field shown above (Field 2), the option Purple is in bit position 16. If the AND operator is used to combine the value in Field 2 is combined with the number 16, the result will be 16 if the bit is set. For example:

```
result = $2 and 16
```

will contain the value 16 if bit position 16 is set, and will contain the value 0 if bit position 16 is not set.

If you need to print output similar to this, a sample script is shown on the next page:

```
Pick the colors that you like:  

Red [ ] Blue [X] Yellow [ ] Green [X] Purple [X] Orange [ ]
```

```
report:
printleft "Pick the colors that you like:" 1
printline

printleft "Red [" 1
result = $2 and 1
if result = 1 then
printleft "X]" 6
else
printleft "]" 6
endif

printleft "Blue [" 10
result = $2 and 2
if result = 2 then
printleft "X]" 16
else
printleft "]" 16
endif

printleft "Yellow [" 22
result = $2 and 4
if result = 4 then
printleft "X]" 30
else
printleft "]" 30
endif

printleft "Green [" 35
result = $2 and 8
if result = 8 then
printleft "X]" 42
else
printleft "]" 42
endif

printleft "Purple [" 48
result = $2 and 16
if result = 16 then
printleft "X]" 56
else
printleft "]" 56
endif
printleft "Orange [" 62
result = $2 and 32
if result = 32 then
printleft "X]" 70
else
printleft "]" 70
endif

printline
```

Here is the script in field 1, which is triggered by a **printreport** statement in a Button field elsewhere on the form.

In the report, the text Red [is printed first.

The statement `result = $2 and 1` does a bitwise comparison with bit position 1 to see if the checkbox for Red has been selected.

If the result = 1, then Red is selected, and the text X] is printed to place an X in the box for Red.

If the result is not equal to 1, then a blank space and the closing bracket] is printed.

Red [X] will be printed if Red has been selected.
Red [] will be printed if Red has not been selected.

Similarly, tests are done to see if:
Blue in bit position 2,
Yellow in bit position 4,
Green in bit position 8,
Purple in bit position 16 and
Orange in bit position 32
have been selected.

The **printline** statement at the end sends the entire line to be printed.

The script in the Button field that triggers the report: event to run looks like:

```
click:
  printreport via IR
```

17. Working with Images

Pendragon Forms supports the use of images in three ways:

- **Decorative (Static) Images**



Decorative images are part of a form design, and are used to illustrate types of questions on a screen, or used to create a custom main menu. For example, an image of a person can be used to show that the current screen contains information about a customer.

Decorative images are also called static images, because the image is the same no matter what record the user is on.

- **Attached (Dynamic) Images**



Attached images are called dynamic images, because typically no two records have the same image. These images are not part of the form design, they are part of each record. For example, a form with a list of employees might have a picture of each employee. When you attach an image to a record and synchronize the handheld, a copy of the image is stored as a long binary object inside the Forms Manager database.

- **Non-Attached (Dynamic) Images**

Non-attached images are image files that are placed on the handheld in an external storage card (or on the device itself, in the case of Pocket PC devices). The images are not attached to Pendragon Forms records, but a **drawimage** script in the form enables the image to be displayed in the form.

Non-attached images are also dynamic, as a different image can be displayed for each record. After synchronization, non-attached images are **not** stored in the Forms Manager database. This can speed up synchronization times, but it means that installing or removing the images from the device must be done manually, separately from Pendragon Forms.

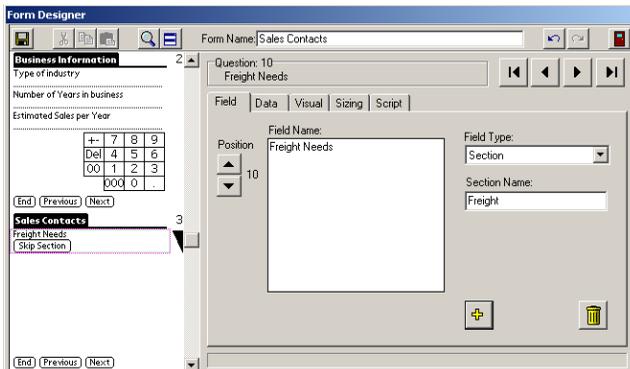
Decorative (Static) Images

Since a decorative or static image is part of a form design, you select the image in the Form Designer window as you design the form.

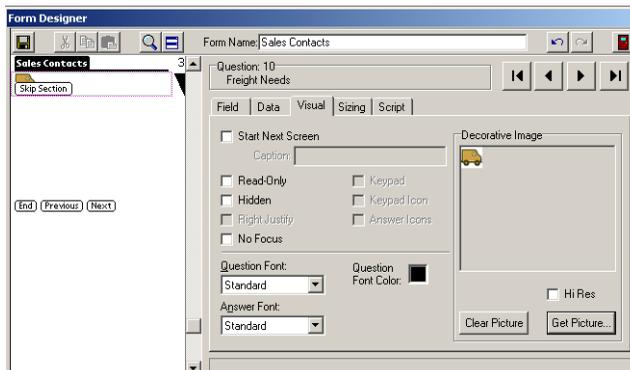
Section fields are typically used for placing static images on a form design. Section fields do not store data, so the handheld user will not need to write anything into the field. You do not have to use Section fields, however: a decorative image can be added to any type of field.

Images that you use as static images in a form must not exceed 320 x 192 pixels in size, and must be .BMP (bitmap) or .JPG (JPEG) file types. The image file should not be more than 256 colors; image files that have more colors will be shrunk to 256 colors to fit on the handheld.

To add a decorative (static) image to a field such as a Section field, first create the field in the Forms Manager.



Click on the Visual tab of the Form Designer window, and then click the Get Picture button. Select a picture from the C:\Program Files\Forms4\ClipArt folder that ships with Pendragon Forms. Or you can select a picture of your own, as long as the picture is either a Windows bitmap

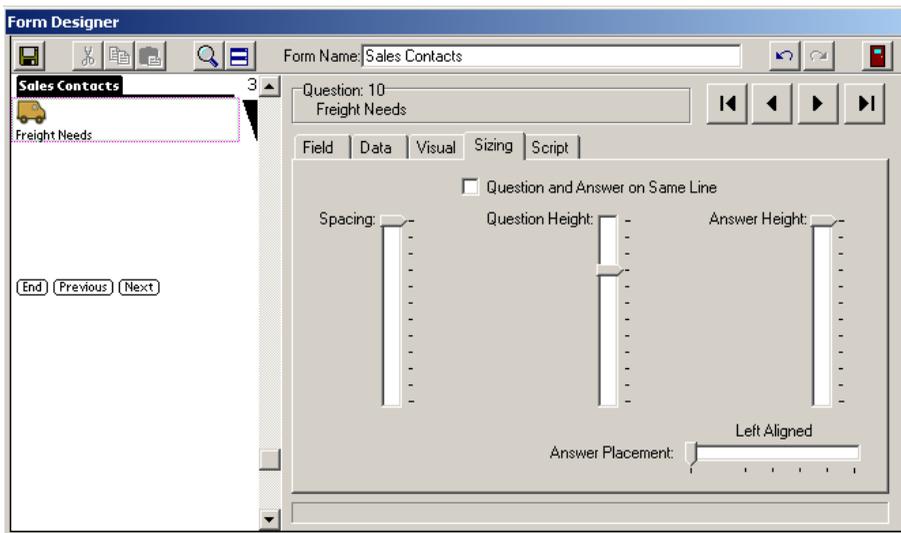


file (.BMP file) or a JPEG file (.JPG file) not more than 320 x 192 pixels in size. If the selected image has more than 256 colors, it will be reduced to 256 colors before being placed on the form.

To see the whole image, click the Sizing tab in the Form Designer window, and make the Question Height bigger.

The field name appears below the picture. If you want to hide the field name, and just have the picture visible, adjust the Question Height accordingly.

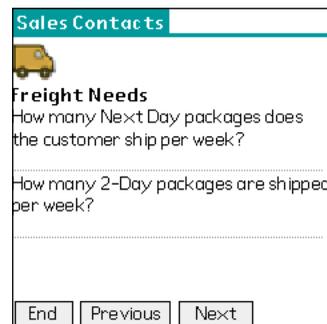
For Section fields, you can adjust the Answer Height to zero to hide the Section Skip button.



PALM OS



PPC



Attached (Dynamic) Images

Attached (or dynamic) images are image files that are attached to records within Pendragon Forms.

You can attach images to records in three ways:

1. Attach images to records on the PC, then send the records to the handheld.
2. Attach images to records on the handheld.
3. Use a handheld with a built-in camera to take a picture from within Pendragon Forms, and have the picture automatically be attached to the current Forms record. Palm-branded Treo devices running Palm OS 5.4 or Windows Mobile 5 for Pocket PC are supported.

These are the requirements for using attached images on each platform:

Type of Device:	Pocket PC	Palm-Branded Palm OS Device	Non-Palm-Branded Palm OS Device
Operating System:	Windows Mobile 5	Palm OS 5.4	Palm OS 5.4
Types of images supported:	.JPG files (JPEG files) .BMP files, .GIF files	.JPG files (JPEG files) .BMP files, .GIF files	.JPG files .BMP files
Need additional software to view .JPG?	No	No	Yes JPEGLib.prc
Is an external storage card required?	Optional. If no storage card is available, attached images are stored in memory.	Yes - for Palm OS Treo and Palm E2 devices. An external storage card must be used to store attached images. Optional - for Palm OS devices with an internal VFS volume, such as the Palm TX and the Palm LifeDrive. If no storage card is available, images will be stored in memory.	Yes. An external storage card must be used to store attached images.
.BMP Limitations	Max size: 60 KB 256 colors (8-bit). Lo-Res devices: 160x128 pixels Hi-Res devices: 320x192 pixels	Max size: 60 KB 256 colors (8-bit). Lo-Res devices: 160x128 pixels Hi-Res devices: 320x192 pixels	Max size: 60 KB 256 colors (8-bit). Lo-Res devices: 160x128 pixels Hi-Res devices: 320x192 pixels
.JPG Limitations	1MB	1MB	Max size: 300KB 24-bit color

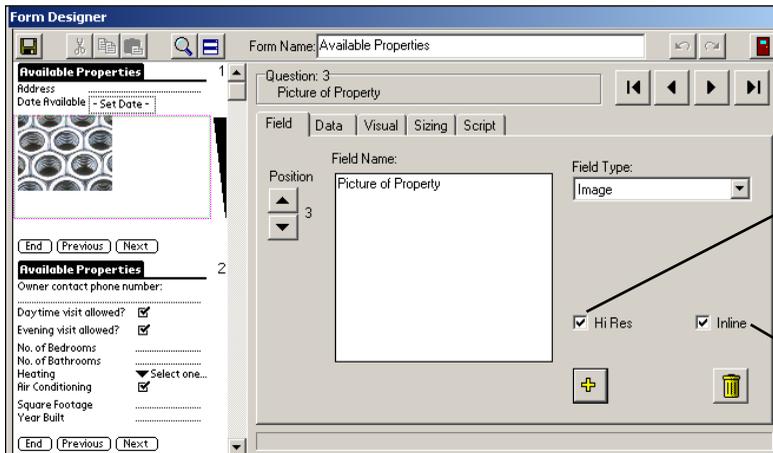
Attaching Images to Records on the PC

In many instances, you may have all the images that you need on the PC. For instance, if you are creating a form that contains pictures of employees, you may have all the employee pictures on your PC already.

If this is the case, you can create records in the Forms Manager on the PC, and import your image files into each record. Refer to the table on the previous page for information on the requirements of image files (.BMP or .JPG files) that you can view from within Pendragon Forms on Palm OS handhelds. Pocket PC devices have fewer limitations on image sizes.

Special fields needed on your form:

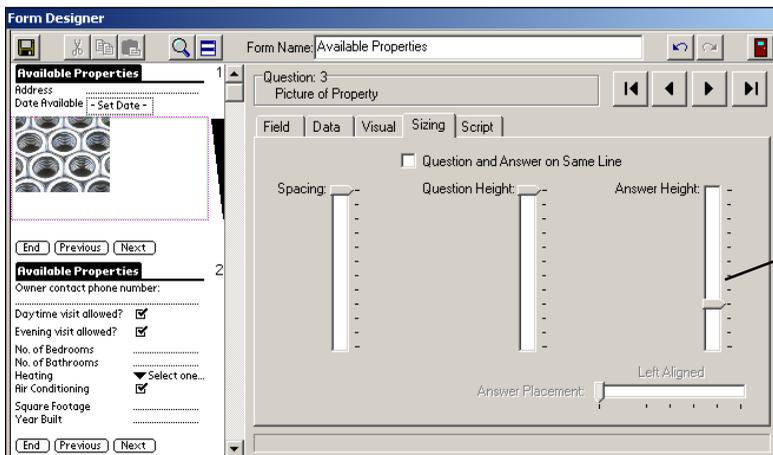
- Image field.



In the Form Designer, create an Image field on your form.

Select Hi Res for high resolution Palm OS devices. (Pocket PC devices are not affected.)

Select Inline if you want to display the attached image on the same screen as other fields on the form.



If you are viewing images Inline, click the Sizing tab and make the Answer Height as large as possible to accommodate the size of the images.

NOTE: If you want to put the same records on several handheld devices, you will need to create a primary key field for your form - see page 188.

When you have completed your form design, freeze the form design. Once the form is frozen, you can create records for the form and attach image files to the records.

To create records, click on the name of the form in the Forms Manager, then hold down the CTRL key and click the Edit/View button. This displays an Access form that shows one record at a time.

RecordID	0
UnitID	0
UserName	Sarah Jane
TimeStamp	4/12/2003 5:00:30 PM
Address	1 Country Lane
Date Available	4/21/2003
Picture of Property	
BMP File	
Owner contact phone num	
Daytime visit allowed?	
Evening visit allowed?	
No. of Bedrooms	
No. of Bathrooms	
Heating	
Air Conditioning	
Square Footage	
Year Built	

Before you can import a graphic into an Image field, you will need to fill in at least one other field on the form to create a new record. Then click the Import button and select the file that you want to attach to the record.

The preview only displays .BMP graphics as 160x128 pixels, even if they are 320x192 pixels. On the handheld the graphic will display appropriately. If you are using .JPG files on the handheld, you will not see a preview of the image here, but the words 'JPG File' will appear in the gray area to the left of the image so that you will know that the image has been imported successfully. (If you are using .GIF files, the words 'GIF File' will appear in the gray area to the left of the image.)

To move to the next record, click the right arrow button at the bottom of the screen, so that the record count advances by one (e.g. from 1 to 2). Remember to fill in a field before importing a graphic.

If you have chosen to display images Inline, this is how they will appear on the handheld:



If you have not chosen to display images Inline, this is how they will appear on the handheld:



If an Image field is **not** set to display Inline, a selector box will be displayed.

Tapping the selector box will display an Image Detail screen. If the image is viewable in Pendragon Forms, the image will be displayed on the Image Detail screen.



Attaching Images to Records on the Handheld

In some circumstances, the images that you want to attach to Pendragon Forms records are originally on the handheld device or an external storage card, not the PC. An example might be a handheld that allows you to take pictures and store the pictures on an external storage card or on the device itself. For something like a home inspection form, you could take a picture of a house and then attach that picture to the Forms record you are filling in on the handheld.

Any images that you attach to Pendragon Forms records on the handheld must conform to the requirements for Pocket PC devices and/or for PalmOS devices as described on page 422. Remember that for non-Palm-branded Palm OS devices, additional software must be installed to view .JPG files on the handheld.

Special fields needed on your form:

- Button field - with a script that lets the user view the images files and select a file to attach to the record.
- Image field - for storing the filename of the file to attach.

Storage requirements for images:

PALM OS

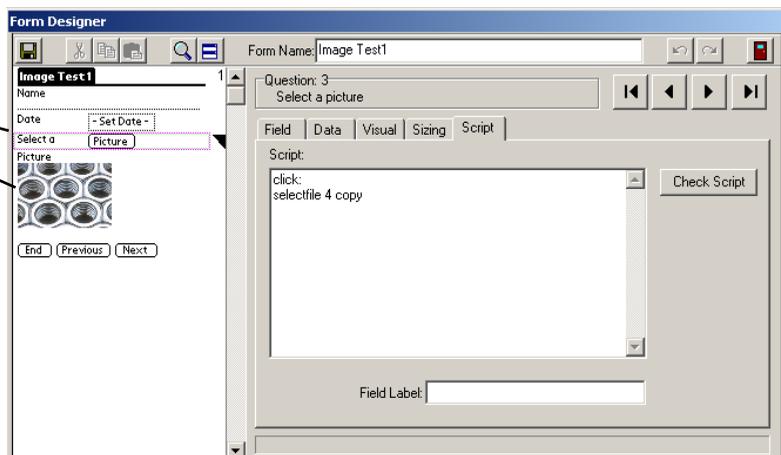
On Palm OS devices without an internal VFS volume, such as Palm OS Treo devices, images can only be stored on an external storage card.

PPC

On Pocket PC devices, images will first be stored on an external storage card, and if no card is available, images will be stored in RAM memory. Do not alternate between having a card and not having a card. Choose one storage option (storage card is preferred).

In the Form Designer, create a Button field and an Image field on your form.

In the Button field, you will need to write a script to allow the user to select an Image file to attach. See next page for a description of this type of script.



Every Button field script begins with a **click:** event. A **click:** event runs when the button is tapped.

The **selectfile** scripting statement (see page 322) allows the user to view files and attach the files to a Forms record.

The format of the selectfile scripting statement is:

selectfile *fieldnumber* **copy**

where *fieldnumber* is the field number of the Image field on the form.

The **copy** parameter means that a copy of the image file is placed into a FORMS5 folder and the original image file remains in its original folder. This is the safest option to use, because it means that Pendragon Forms is not removing your original image file. On Palm OS Treo devices the FORMS5 folder must be on a storage card. On Palm OS TX and LifeDrive devices, and on Pocket PC devices, the FORMS5 folder should be on a storage card but can also be in the RAM memory of the device.

Other possible options are:

selectfile *fieldnumber* **move** which moves the original image file to the FORMS5 folder on the storage card, renames the file to the convention used by Forms, and deletes the original image file. This option uses less space on the card, but is not as redundant as **selectfile** *fieldnumber* **copy** because the original image file is deleted.

New in Pendragon Forms 5.1 is a **selectfile** *fieldnumber* **capture** option that allows you to take a picture with the handheld camera and attach it to a record. This option is available on Palm OS devices with Palm OS version 5.2.8 or higher (Zire 72 or Treo devices), or Windows Mobile 5 Pocket PC devices. See pages 430-432 for an example.

In the script shown in the picture on the previous page, field 3 is a Button field with a script, and field 4 is the Image field for storing the name of the attached file. The script in field 3 (the Button field) is:

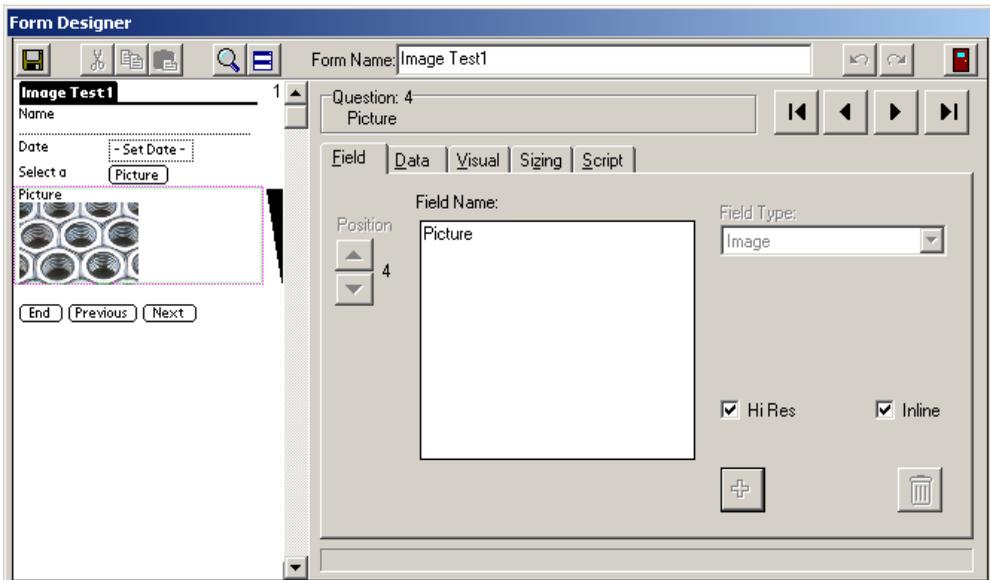
```
click:
  selectfile 4 copy
```

The number after the word selectfile is the field number of the Image field, in this case, field 4 is the Image field.

If the Image field was field 28 on the form, the script in the Button field would be:

```
click:
  selectfile 28 copy
```

To view the attached image from within Pendragon Forms, check the Hi Res and Inline checkboxes of the Image field.

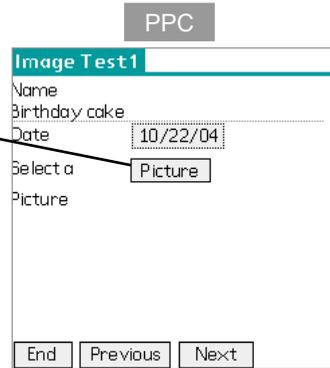


Attaching an Image on a Handheld



On the handheld, the user can tap the button (labeled Picture in this example) to display available images.

These screen shots were taken on a Palm OS 5.4 device and a Windows Mobile 5 Pocket PC.



If the original image is on a storage card, tap to select the storage card. Typically, the DCIM folder on a storage card contains camera images.

Or tap a different folder if the image files are stored elsewhere. Pocket PC devices and Palm OS devices with an internal VFS volume (e.g. Palm LifeDrive and Palm TX) can store images in memory.



Tap an image to select it.



After an image file has been selected, the user returns to the Pendragon Forms record.

If the image has been set to view Inline, then the image will be visible within Pendragon Forms, containing the image.



Taking a Photo on a Treo Smartphone

Pendragon Forms 5.1 enables you to take a picture with a PalmTreo smartphone, including Palm OS Treo models such as the Treo 650 or the Treo 700p, and also Windows Mobile 5 for Pocket PC models such as the Treo 700w and Treo 700wx.

Palm OS Treo smartphones must have a storage card.

Windows Mobile 5 for Pocket PC Treo smartphones can store photos on the device or on a storage card. A storage card is preferred.

Special fields needed on your form:

- Button field - to allow the handheld user
- Image field - for storing the filename of the file to attach.

In the Button field a selectfile statement is used to initiate taking a picture on the handheld.

The format of the selectfile scripting statement is:

selectfile *fieldnumber* capture

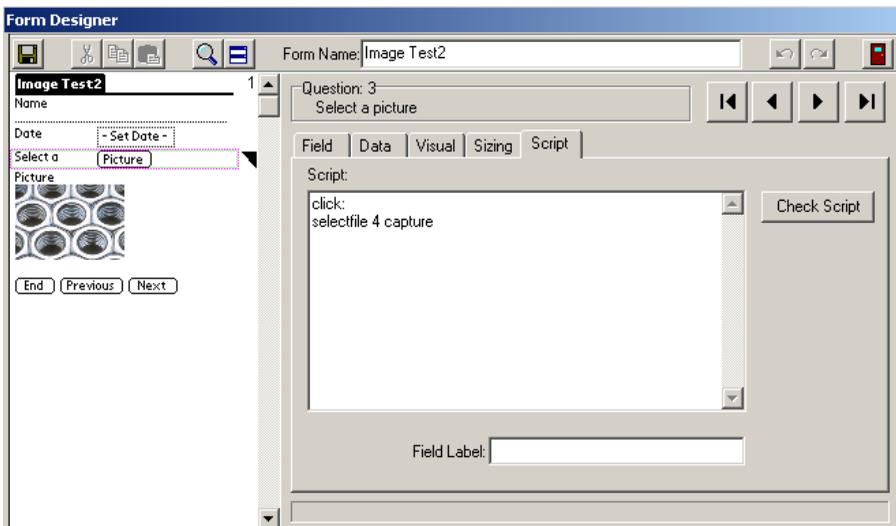
where *fieldnumber* is the field number of the Image field on the form.

In the picture below, the Image field is field 4 on the form, and so the script in the Button field is:

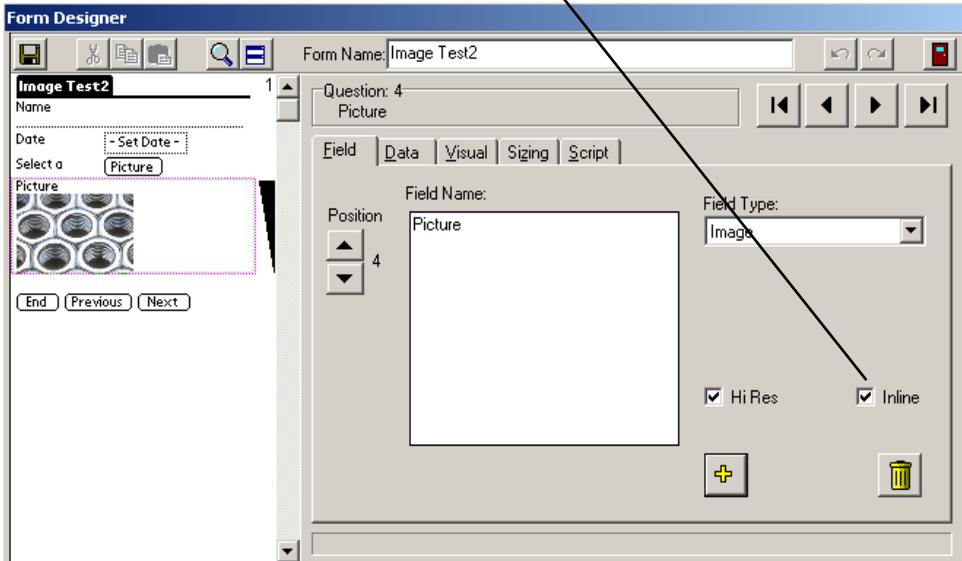
```
click:  
selectfile 4 capture
```

PPC

Note: On Windows Mobile 5 Pocket PC devices, you can also use **selectfile *fieldnumber* copy** for taking pictures.



The Image field can be set to display images Inline, so that the picture will be displayed within Pendragon Forms.



PALM OS

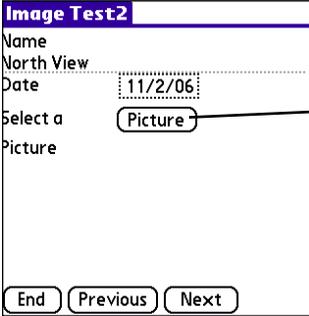


Image Test2
Name
North View
Date 11/2/06
Select a Picture
Picture
End Previous Next

Tap on the button to activate the handheld camera.



Cancel 2x 1x 640



7K



Image Test2
Name
North View
Date 11/2/06
Select a Picture
Picture
End Previous Next

PPC

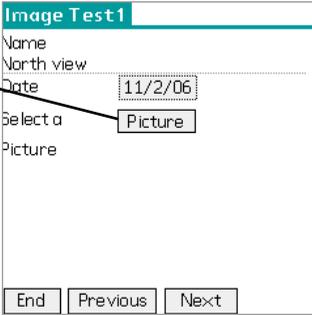


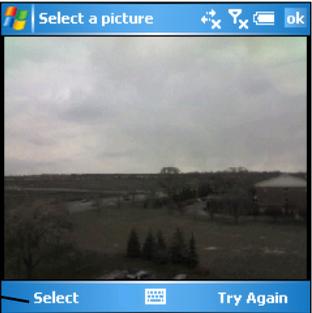
Image Test1
Name
North view
Date 11/2/06
Select a Picture
Picture
End Previous Next

On Windows Mobile 5 Pocket PC devices, tap the Camera option.



Select a picture Select Exhibit Photo or Video My Pictures Date Camera

On Palm OS devices, the camera will be activated. Press the button to take a picture. Then tap the Save icon to save the picture.



Select a picture Try Again

On Pocket PC devices, once the camera is active, press the appropriate button to take a picture. When the image freezes to indicate that the photo has been taken, tap Select to select that photo.



Image Test1
Name
North view
Date 11/2/06
Select a Picture
Picture
End Previous Next

Once the photo has been selected, you will be returned to the Pendragon Forms application and the picture will be displayed if the Image field has been set to display Inline.

Drawing Non-Attached Image Files

A non-attached image file is an image file that is present on the handheld - either on an external storage card in the case of Palm OS devices, or anywhere on the file system in the case of Pocket PC devices - but the file is not attached to a Pendragon Forms record and therefore does not get synchronized and stored in the Forms Manager database.

An example might be if you have pictures of employees or inventory items that do not change frequently, and you want to put all the images on a storage card, and store only the image file names in your Pendragon Forms records. When you need your form to display an image, a **drawimage** scripting statement is used to draw the named file on the handheld screen.

This method saves on synchronization time, since the images are not stored in the Pendragon Forms database on the PC, and do not get synchronized. However, in this scenario, you are responsible for managing the installation or removal of image files from the handheld or storage card, and this task is done manually, outside of the Pendragon Forms environment.

The **drawimage** statement draws a non-attached image on the handheld screen without using an Image field. The format of the statement is:

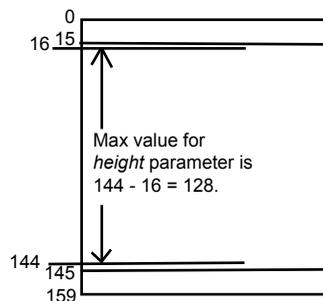
drawimage *top height filename*

where *top* is the vertical pixel at which the image will be painted, *height* is the maximum amount of vertical space used to display the image, and *filename* contains the directory path and file name of the image file.

For drawing images, Pendragon Forms always refers to the screen as if it has 160x160 pixel resolution, with pixels being numbered from 0 to 159. For devices with a higher screen resolution, such as 320x320 pixels or 320x480 pixels, still use a 160x160 screen space, and Pendragon Forms will scale the image accordingly.

On a Palm OS device, the title bar takes up 16 pixels at the top of the screen (from pixel 0 to pixel 15), and the buttons at the bottom of the screen take up 14 pixels (from pixel 145 to pixel 159). Therefore, the smallest value for the *top* parameter is 16, and the largest value for *height* is 128.

Smallest value
for *top*
parameter is 16.



NOTES:

- a) The drawimage statement will overwrite controls on the screen if you allow it to. This is not recommended because you may see unusual effects. For example, if you set the *top* parameter to zero, the image will overwrite the title bar.

- b) A Text field occupying one line for the question and one line for the answer takes up 11 pixels for the question and 11 pixels for the answer. Therefore, if you wanted to have a Text field precede the drawn image, you would need to shift the *top* parameter down by 22 pixels.

Special fields needed on your form:

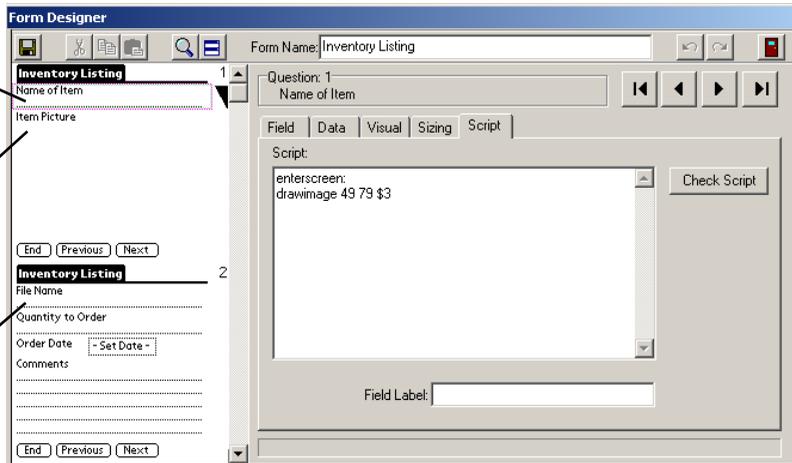
- Text field - for storing the path to and file name of the image file.
- A field such as a Section field for reserving the on-screen space where the image will be displayed.

In the example below, an inventory form displays a picture of a different item for each record.

In this form:
Field 1 stores the name of an item. A script in Field 1 draws the image.

Field 2 is a Section field used to reserve space where the image will be drawn.

Field 3 stores the path and file name of the image file to be drawn.



The script in field 1 is an **enterscreen:** script, so the script will run as soon as the user gets to this screen. The script in this example is:

```
enterscreen:
drawimage 49 79 $3
```

In this example, we want to display field 1 and the field name of field 2 and then draw the image. The value for the *top* parameter is therefore 16 pixels (for title bar) + 22 pixels (for the question and answer lines of field 1) + 11 (for the question of field 2) = 49.

The maximum value that the height parameter can be is 128, but we are already starting our image at pixel position 49, so the maximum remaining height is 128 - 49 = 79.

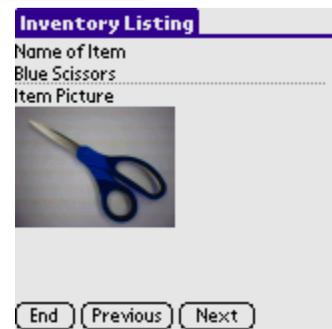
The final parameter in the drawimage statement, is the file name of the image to be drawn. In this example, \$3 means the value in field 3, which will store the path and file name of the image.

To make this form work, after the form is frozen, the records are created on the PC and sent to the handheld. The FileName field (field 3) contains the exact directory path and file name of each image that will be drawn for each record. Note that Palm OS devices use a forward slash (/) symbol in path names, while Pocket PC devices used a back slash (\) symbol in path names. If you are designing for both platforms you can send specific records to specific devices, or use additional fields to store the path name separately from the file name, and use a script to concatenate the appropriate path for a given platform with the file name.

The image files must be placed in the specified directory on the external storage card in the case of Palm OS devices. On Pocket PC devices, the images can be stored on the device or on an external card.

Form			
TimeStamp	NameOfItem	ItemPicture	FileName
05 11:40:10 AM	Blue Scissors		/DCIM/Set56_01.jpg
005 3:03:55 PM	Printer		/DCIM/Set52_01.jpg
005 3:04:52 PM	Paper Clips		/DCIM/Set31_01.jpg
005 3:05:29 PM			

When the handheld user selects a record to review, an image is drawn on the screen. The image is not contained in the Pendragon Forms database, it is maintained separately on an external storage card (in the case of Palm OS devices), or in the case of Pocket PC devices, the images can be stored anywhere in the file system, including on an external storage card.



Location of Image Files on the Handheld

On the PC, image files are stored in a binary field in the Forms Manager database.

On handheld devices, image files are stored separately from Forms records. Forms records are stored in RAM on the handheld, and image files are typically stored in a FORMS5 folder on an external storage card (SD card, MMC card or Memory Stick).

Pocket PC devices, and Palm OS devices with an internal VFS volume (e.g. Palm TX and Palm LifeDrive) are also capable of storing images in a FORMS5 folder in RAM if no external storage card is available. If a device is capable of storing images either on a card or in memory, it is not recommended that you alternate between the two. Decide whether or not to use a storage card and stick with that option. A storage card is preferred.

The name of each image file on the handheld is composed of the FormID number (8 hex (hexadecimal) digits), the RecordID (8 hex digits) and the Question ID (4 hex digits), plus a file extension that is assigned by the conduit when the image is downloaded to the handheld. Pendragon Forms on the handheld uses the file extension to determine whether or not a file can be displayed internally.

Synchronization of Image Files

A Forms record with 20 fields might take up 10-20 KB. In contrast, a single attached image file might be on the order of 70-200KB for built-in cameras. Image files are therefore much bigger than Forms records, and during synchronization, this means that a much larger amount of data has to be downloaded to the handheld if you are using image files. This causes synchronization to be slower.

Important: Due to the large size of images, it is not recommended that you use images on devices that are synchronizing via a serial port, or on devices that synchronize via modems with speeds of less than 100Kbps (kilobits per second).

Pendragon Forms 5.1 uses Pendragon SyncServer technology to cache records, so that only new or changed records on the handheld are synchronized. If the records that have image files are for reference purposes, and do not change frequently, then the SyncServer caching mechanism will prevent these records from having to be synchronized every time the handheld synchronizes. This will help to speed up synchronization time.

Another way to reduce synchronization time is to minimize the number of records with images that you keep on the handheld. Set a Data Persistence option (see page 197-198) that removes unnecessary records from the handheld during synchronization.

Saving Images out of the Forms Manager Database

After synchronization, if you are storing data in the Pendragon Forms Manager database, there are two ways to save the image files as separate files outside of the Forms Manager database.

Option 1: Saving from the Edit/View window

When you click on the name of a form in the Forms Manager and then click the Edit/View button to view the data, Image fields will contain the phrase Long Binary Data.

If the original image was a .BMP file, then double-clicking the Long Binary Data field will open the Windows Paint program (MSPaint.exe). The file will be displayed in Windows Paint with the file name PFTEMP2.BMP. Choose to save the file under a different name. Also, if the file was originally a .JPG file, then when you save the file, make sure to save as file type .JPG and not .BMP.

If the original file was a .JPG file, then double-clicking the Long Binary Data field may open Internet Explorer to display the picture, if Internet Explorer is the program that is associated with opening .JPG files. You can right-click on the image and choose Save As to save the file separately from the Forms Manager.

Note: Versions of Windows prior to Windows 2000 may not be able to open .JPG files in Windows Paint. If you experience this problem, use option 2 on the next page.

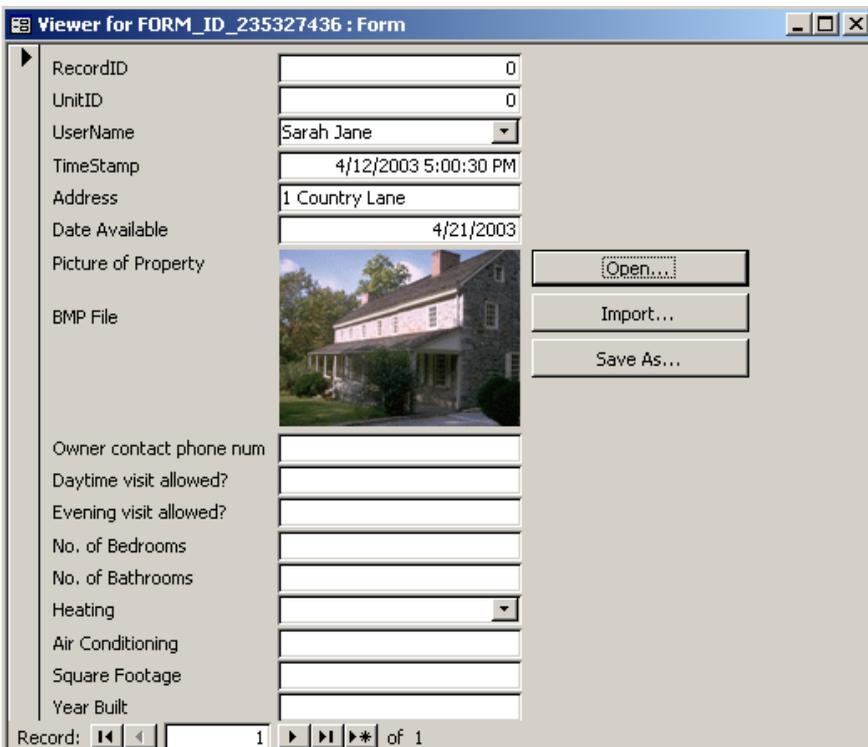
Option 2: Saving from the Ctrl + Edit/View window

Another option for saving images is to click on the name of a form in the Forms Manager and then hold down the CTRL button and click the Edit/View button. This displays records one at a time.

Bitmap files (.BMP) files will be visible in the record. .JPG files will not be visible in the record.

Clicking the Open button behaves like Option 1 on the previous page. The file will open the Windows Paint program (MSPaint.exe). The file will be displayed in Windows Paint with the file name PFTEMP2.BMP. Choose to save the file under a different name. Also, if the file was originally a .JPG file, then when you save the file, make sure to save as file type .JPG and not .BMP.

If you have a version of Windows that cannot open .JPG files in Windows Paint, click the Save As button instead of the Open button. Save the file with any file name and the appropriate file extension. For instance, if the original file was a .JPG file, save as a .JPG file. Once the file has been saved, you can use your own image-viewing software that you use for viewing .JPG images. (If you do not have image-viewing software, double-clicking on a .JPG file typically runs Internet Explorer, and you can view the .JPG image in your Web browser.)



18. Internet and Serial Communications

Various handheld devices offer ways to connect to the Internet wirelessly. If you want to synchronize multiple records at a time across the Internet, Pendragon SyncServer, which is built into Pendragon Forms 5.1, allows you to do this. Refer to Chapter 1 for information on network and client installations.

If you need to submit one record at a time to do live querying of a Web database, Pendragon Forms contains several scripting commands that allow you to send data wirelessly to a Web URL, or via e-mail. Pendragon Forms also allows you to send and receive data from devices that are attached to the serial port of the handheld device. A Button field with a **click:** event script gives the handheld user control over wireless or serial transmission of records.

Sending individual records wirelessly is not a substitute for synchronizing the handheld device. Pendragon Forms is designed to use the synchronization process to place records from the handheld into a database table in the Forms database on the PC. Sending a record wirelessly via e-mail or to a Web URL will not place the data in the Forms Manager database. However, if you have your own Web database application, you can use Pendragon Forms to look up or store data in your Web database.

Wireless / Serial Scripting Statements

The following scripting statements can be used in a **click:** event to control the transmission or receipt of wireless or serial data. These scripting statements are described in detail in Chapter 13 *Scripting Reference*, which starts on page 245.

printreport via `{$Fieldnumber | $[FieldLabel] | "E-mail address"}`

PALM OS

Used on Palm OS devices in conjunction with VersaMail e-mail software from www.palm.com. (Some Palm devices ship with VersaMail.) A report specified in a **report:** event is used to specify the layout of an e-mail, and the **printreport** statement is added to a **click:** event in a Button field to create an e-mail to the specified recipient. The result variable is used to specify a subject line and to add cc: and bcc: e-mail recipients. See page 441 for an example.

transmit web "*URL*"

Used in conjunction with handheld devices that have a TCP/IP connection, such as a Palm Tungsten C with WiFi connection, or a Palm Treo 650 with high speed data service.

extract "*entity*" **from** *value*

Extracts XML formatted data from the specified value. When the **transmit web** statement is used, data returned from a Web server is stored in the **webdata** function. The **extract...from** statement can be used to extract XML from Webdata. Extracted data is placed into the **result** variable. See page 443 for an example.

transmit serial *value*

Uses the **result** variable to specify communications parameters for communicating via the serial port of the handheld device. The specified *value* is the data string that is sent to the serial device. See page 338 for details.

acquire "*settings*" "*startpattern*" "*stoppattern*"

Opens the handheld serial port to acquire data from an attached device, and places the collected data in the **result** variable. The attached device could be a GPS (Global Positioning System) receiver with a serial data port. Note: Forms 5.1 has a GPS Custom Control that should be used before trying a script (see page 574).

E-Mailing a Record

PALM OS

Pendragon Forms offers a way to e-mail a record directly from a Palm OS handheld. If you have a Palm OS device such as a Treo 650 or Treo 700p with high speed data service, you can use the **printreport via *E-Mail address*** scripting statement in conjunction with **VersaMail** e-mail software from www.palm.com. (Check if your Palm OS device has VersaMail included or if it needs to be purchased separately.)

Sending an e-mail is useful for allowing the handheld user a quick way to send data back to the office. However, e-mailing a record is not a substitute for synchronization. A text record received at the office would might need to be re-typed into a database, which is not as efficient as the handheld user synchronizing to send data to the database.

To e-mail a record, first a **report:** event script is used to specify the layout of the e-mail. All fields on the form can be included in the e-mail, except Signature, Sketch and Image fields.

Then, a **printreport via *E-Mail address*** statement is typically used in a **click:** event in a Button field, so that the handheld user can choose when a record is ready to be e-mailed.

The *E-Mail address* can be a direct e-mail address such as:

```
click:
printreport via "SarahJ@mycompany.com"
```

Alternatively, if an e-mail has been entered into a field, the field can be referenced by its field number, such as:

```
click:
printreport via $5
```

Or a field containing an e-mail can be referenced by a field label, such as:

```
click:
printreport via $[Customer]
```

The **result** variable can be used to specify a subject line, and optionally to specify cc: and bcc: recipients of the e-mail. If the result variable begins with "subject:" then the text in the **result** variable will be added to the Subject line of the e-mail. For example:

```
click:
result = "subject:New Customer Request"
printreport via $3
```

The format for the result variable including subject line and cc: and bcc: e-mail recipients is:

subject:*subject_line*;**cc:***cc_address*;**bcc:***bcc_address*

The example on the next page shows an e-mail that includes a subject line and a person copied in on the cc: line of the e-mail being sent.

In this example, the handheld user fills in the form, then taps the SEND button on the second screen to e-mail the record to the e-mail address in Field 5.

In Field 1, a **report:** event script specifies the layout of the e-mail.

```
report:
printline
printleft "Here is a new sales contact:" 1
printline
printline
printleft "First Name:" 1
printleft ${FirstName} 13
printline
printleft "Last Name:" 1
printleft ${LastName} 12
printline
printleft "Company Name:" 1
printleft ${Company} 15
printline
printleft "Contact Phone Number:" 1
printleft ${Phone} 23
printline
printleft "Contact E-Mail:" 1
printleft ${EMail} 17
printline
printleft "Date contacted:" 1
format ${Date} date
printleft result 17
printline
printline
printline
```

In the Button field, a click: event script creates an e-mail in VersaMail with the subject line of *Catalog Request*. The e-mail is copied to the e-mail address in the field with field label SalesDept (the sales@mycompany.com e-mail in the picture above). The primary e-mail recipient is the e-mail address in the field with field label EMail (the contact e-mail address of the customer).

```
click:
result = "subject:Catalog Request;cc:"
result = result & ${SalesDept}
printreport via ${EMail}
```

Communicating with a Web Server

Handheld devices using TCP/IP can send data from a Pendragon form to a Web server, and receive XML formatted data in response.

Devices that support TCP/IP include a Palm Tungsten C with WiFi connection, a Palm Treo 650 with high speed data service, or a Dell Axim Pocket PC device with WiFi.

The Transmit Web scripting statement is used to send data to a Web server, and the response from the server is accessed via the Webdata function. Webdata captures data returned from the server between <PFDATA> and </PFDATA> tags. The Extract statement can be used to extract XML data from the Webdata function.

In this example, entering or scanning a part number and then tapping a button sends a request to a Web server. The response from the server is used to populate fields on the form.

The script in the Button field might look like:

```
click:
if $1 <> null then
transmit web "http://www.mysite.com/cgi-bin/inventory.cgi?part=$1$"

extract "DESCRIPTION" from webdata
$3 = result

extract "QTY" from webdata
$4 = result

extract "REORDERDATE" from webdata
$5 = result
endif
```

In the Transmit Web statement, the parameter \$1\$ refers to the value in Field 1. Sample HTML that the Web server might return is shown on the next page.

If the Web server returns the following HTML, the Extract statement in the script can extract data to be placed in various fields on the form.

```
<HTML>
<PFDATA>
  <DESCRIPTION>Polarizing Sunglasses</DESCRIPTION>
  <QTY>18</QTY>
  <REORDERDATE>March 2, 2000</REORDERDATE>
</PFDATA>
```

Note that if the **extract** statement is to return a date, the easiest thing is to put that date in a Text field on the form. If the Reorder date on the sample form had been a Date field, the Web server would need to return a number of seconds since 01/01/1904. Similarly, if the Web server needs to return a Currency value, it should be placed in a Text or Numeric field on the form, since Currency fields on the handheld are stored internally as a number of cents. Alternatively, a currency amount can be extracted to a hidden field on the form, and then a script can be used to format the currency and place it into a Currency field.

19. Beaming Records

PALM OS

This chapter applies to Palm OS devices only. Pendragon Forms supports the ability to beam records from one Palm OS handheld to another Palm OS handheld.

Beaming can be allowed on a form by form basis. Before you allow a form to be beamed, you need to consider whether it is appropriate for records to be shared.

If two users have the same record on their handhelds, and they both modify that record, then when the users synchronize, the last handheld to synchronize will overwrite the first handheld, even if the first handheld has the more recent data.

Beaming is only appropriate if:

- The second handheld does not modify the beamed record, OR
- The second handheld fills out different fields from the first handheld, and the Advanced Form Property to Merge Changes is switched on, so that during synchronization, data from both handhelds is merged into the same record in the database on the PC.

Beaming records is not a substitute for backing up data to the PC during synchronization.

For beaming to take place, the following conditions are necessary:

- 1) Both handhelds must already have both the Pendragon Forms application and the identical form design (same Form ID#) present on the handheld. Only records can be beamed, not the Forms application nor the form design.
- 2) The receiving handheld must be configured to allow receiving of beamed data.
- 3) Typically, you will need to select a primary key field or fields when you design the form.
- 4) The form design must have a Button field with a script that selects records to be beamed. The script also initiates the beaming.
- 5) An appropriate Advanced Form Property Beaming Rule must be selected.
- 6) The Advanced Form Property to Mark Received Records as modified should be set appropriately.
- 7) Switch on the Advanced Form Property to Disable Record Overwrite Protection.
- 8) Recommended: the Advanced Form Property of Merge Changes should be switched on.

These issues are discussed in this chapter.

Issue 1: Software required to beam records

The Pendragon Forms application itself and the form design cannot be beamed. This means that before you can beam records from one handheld to another, the Forms application must be installed on both handhelds, and the identical form design (same Form ID #) must be on both handhelds.

When you create a form in the Forms Manager, the form design is assigned a unique Form ID number. The Form ID number must be the same on both handhelds to allow beaming.



To check the Form ID number of a form design on the handheld, tap the Forms icon, then tap the name of the form to highlight it.

Tap the handheld Menu button (the drop-down menu icon below the House icon on the handheld), and tap the Help menu. Choose the Form Info menu option.

The Form ID number is the first item displayed in the Form Information dialog. This ID number must be identical on both handhelds for records for this form to be beamed.

Issue 2: Receiving handheld must allow beam receiving

Switching on and off the ability to receive beamed records is done within the Preferences application on the handheld. This is separate from Pendragon Forms.

On the handheld, tap the Applications button (the House icon) and then tap the Prefs icon (for Preferences). On some handhelds, the Preferences application has an arrow button in the upper right corner of the screen. Select the General option, and check that the Beam Receive option is switched ON.

On devices running PalmOS 5 and higher, the Prefs application contains sections on the screen. In the General section, tap the Power option. On the Power screen, check that Beam Receive is switched ON.

Issue 3: Select primary key field(s) in your form design

You will generally need to select a primary key field or fields if allowing beaming.

The default primary key in Pendragon Forms includes three internally generated fields: Username, creation TimeStamp and UnitID. If a record is created on a handheld, the Username field is assigned the username of that handheld device. If that record is then placed on another handheld, and the second user modifies the record, the Username field is updated with the second user's handheld username. This causes duplicate records to be sent to the database on the PC - one record for each user, since there are two different usernames.

If you want users to share records, then to avoid the problem of duplicate records, you will need to choose your own primary key - one or more fields that in combination are unique for every record. See page 188 for information on creating a primary key in Pendragon Forms. The field or fields that make up the primary key have to be selected before the form is frozen, because the database table that is created when a form is frozen must contain information on which fields are the primary key fields.

To allow users to share records, you will need to delete the default Advanced Form Property of **Additional Download Criteria** that causes only those records that match the handheld username to be sent to a given handheld. See page 214 for information on Additional Download Criteria.

Issue 4: Create a Button field containing a beaming script

In order to beam records, a form needs a Button field with a script that controls the beaming.

All scripting events and scripting statements related to beaming are listed in Chapter 13, *Scripting Reference*, which starts on page 245.

A **click**: event script in a Button field runs when the handheld user taps the button.

A **select** statement lets you control which records are selected. Select statements include **select all**, **select current**, **select where**, and **select matching**.

A **queue** statement queues all selected records to be beamed. You can choose to queue all selected records or queue only the changed records from among the selected records. A changed record is a new or a modified record.

You can have several combinations of select and queue statements in one script.

A **beam** statement actually beams the queued records. You can beam via infrared or Bluetooth.

Examples of beaming scripts are shown on the pages that follow.

Beaming Example 1: Beaming the Current Record

In the form shown below, the handheld user can fill in the form, and then on the last field tap a button to beam the current record to another handheld that has this form design.

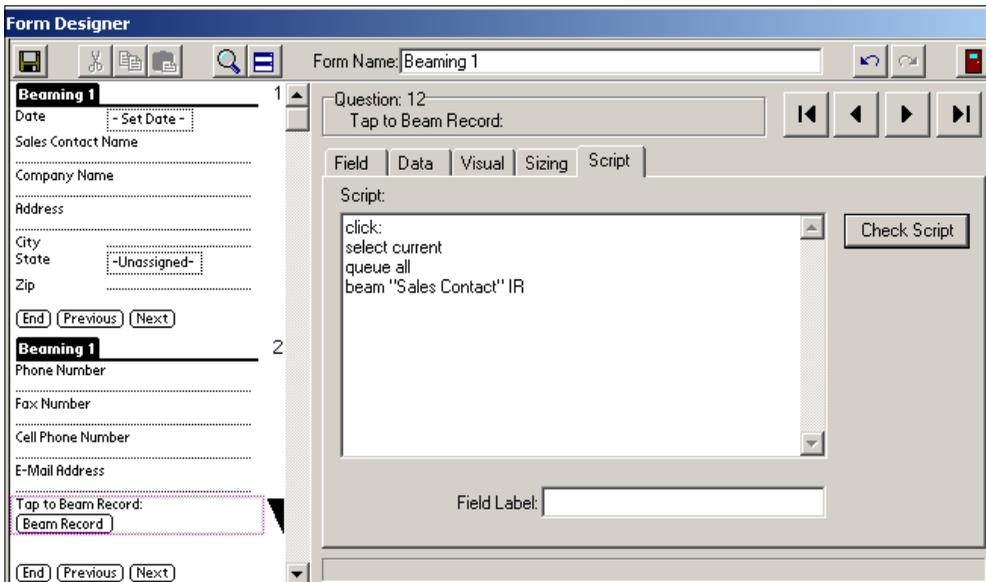
Field 12 in this example is a Button field with a **click:** event script that runs when the button is tapped:

```
click:  
  select current  
  queue all  
  beam "Sales Contact" IR
```

The **select current** statement selects the current record.

The **queue all** statement means that the record will be queued for beaming, whether the record is new, changed (modified), or just an existing unchanged record. If the statement **queue changed** had been used, the record would only be queued for beaming if it was a new or changed record.

The **beam "Sales Contact" IR** statement will beam via the infrared port on the handheld. The phrase Sales Contacts is the phrase that the receiving handheld will display when it prompts the user if the records should be accepted.



Beaming Example 2: Using criteria to select which records to beam

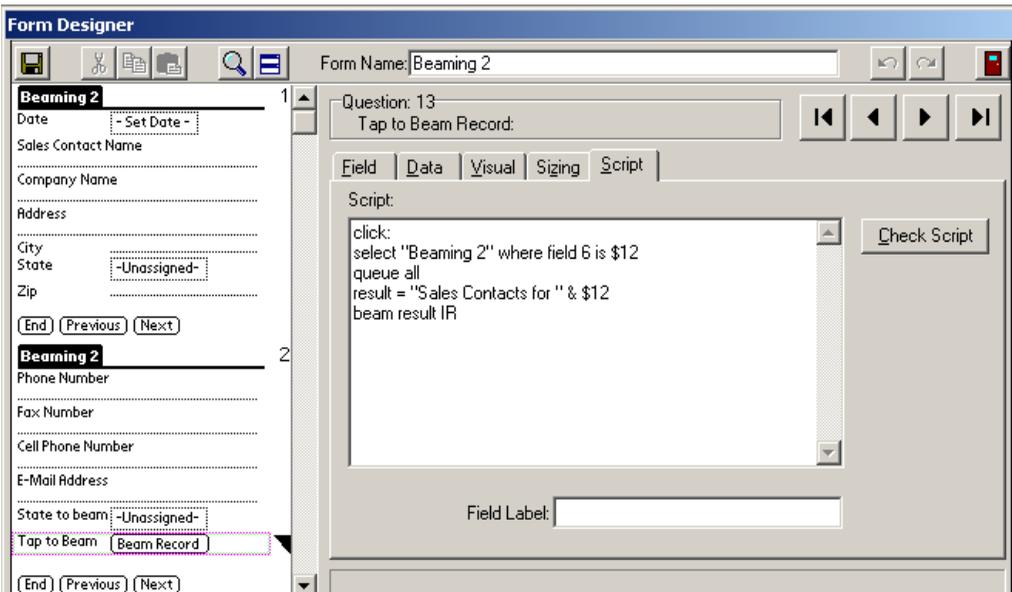
In the form shown below, a **select where** statement is used choose which records are selected to be beamed. See page 320 for more information on select where statements.

Field 6 on the form shown below is the State part of an address. Field 12 also lets the user pick a state. When the user picks a state in field 12 and then taps the button to beam, all the records that match the selected state will be beamed.

`select "Beaming 2" where field 6 is $12`
means that in the form called "Beaming 2", all records in which the State field (field 6) matches the current field 12, will be selected.

The **queue all** statement means that all the selected records will be queued for beaming. If **queue changed** was used instead, only new and changed records would be queued.

For an added touch, the statement
`result = "Sales Contacts for " & $12`
concatenates the phrase "Sales Contacts for " with the value in field 12, which is the selected state. If the selected state is NY, the value in the result variable will be "Sales Contacts for NY". The statement **beam result IR** beams the selected records. The receiving handheld will see a message *Do you want to accept "Sales Contacts for NY" into Forms 5.1?*



Beaming Example 3: Beaming records from both a parent form and a subform

The form shown in this example is a parent form. Field 7 is a subform field that allows the user to jump to a subform to fill out subform records. Field 8 on the parent form is a Button field with a **click:** event that allows the user to beam records from both the parent form and the subform.

```
select all "BeamingTest3 Parent"  
queue changed
```

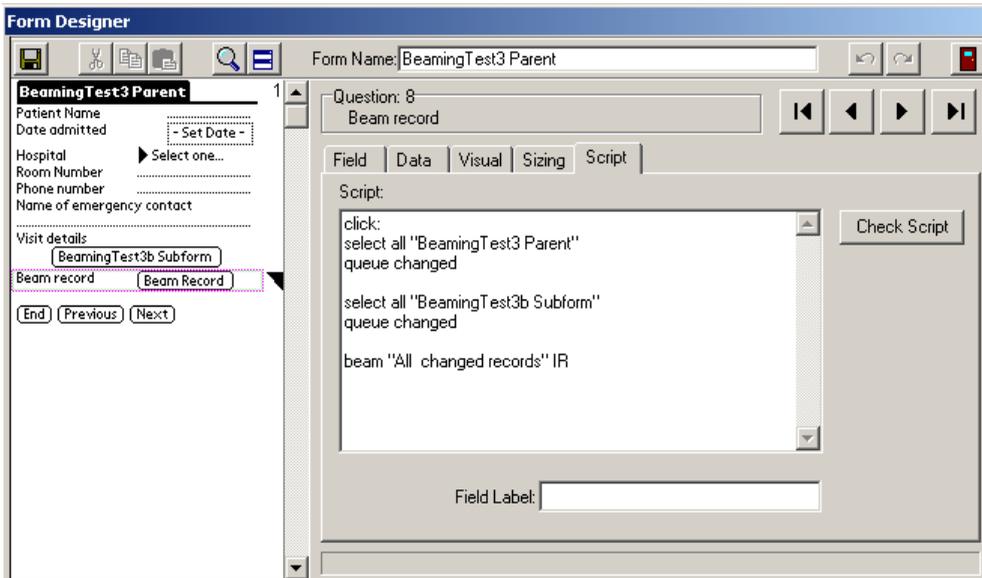
selects all records from the parent form whose name is "BeamingTest3 Parent", and queues only the changed records to be beamed. A changed record is a new record or a modified record.

```
select all "BeamingTest3b Subform"  
queue changed
```

then selects all the records from the subform whose name is "BeamingTest3b Subform", and queues only the changed subform records to be beamed.

```
beam "All changed records" IR
```

beams the selected records from both parent form and subform. The receiving handheld sees the message: *Do you want to accept "All changed records" into Forms 5.1?*



Issue 5: Beaming Rules

Beaming Rules are an Advanced Form Property that allow you to decide under what circumstances the receiving handheld will accept beamed records.

See pages 223-224 for information on how to set a beaming rule. The available options are:

Beaming Rule	Meaning
Do not accept any beamed records, conflicting or otherwise	Beaming is not allowed.
Accept records. Merge data in records with matching keys	<p>Beaming is allowed.</p> <p>If both handheld users have updated different fields of the same record, the changes from the handheld that is doing the beaming will be merged with the changes that are on the receiving handheld.</p> <p>If both handheld users have updated the same field of the same record, the handheld that is doing the beaming will overwrite the record on the receiving handheld. This applies if a record is marked as new on both handhelds (perhaps due to prior beaming). In this case, the beaming handheld will also overwrite the receiving handheld's record with no merging of data.</p> <p>NOTE: With this option, the receiving handheld ends up with a complete record of the changes, but the beaming handheld does not have all the changes.</p>
Accept Records. Overwrite existing records that have matching keys	<p>Beaming is allowed.</p> <p>The handheld that is doing the beaming will overwrite any changes that the receiving handheld has made on the same records.</p> <p>NOTE: With this option, both handhelds end up with the same version of the record, namely the version that was on the beaming handheld.</p>
Accept Records, but reject those with matching keys	<p>Beaming is allowed.</p> <p>If the receiving handheld already has a record, that record will not be beamed.</p>

Issue 6: Choose whether to mark received records as modified

When the handheld user beams records, any new or changed records on the beaming handheld will also be marked as changed on the receiving handheld. This guarantees that the records will be backed up to the PC if the receiving handheld synchronizes first.

The Mark Received Records as Modified option is an Advanced Form Property that lets you control whether existing unchanged records that are beamed should also be marked as changed records on the receiving handheld. If the unchanged records are marked as changed on the receiving handheld, the records will be uploaded to the PC when the receiving handheld synchronizes.

By default, Mark Received Records as Modified is switched on. See page 225 for information on how to switch this option off if necessary.

Issue 7: Disable Record Overwrite Protection

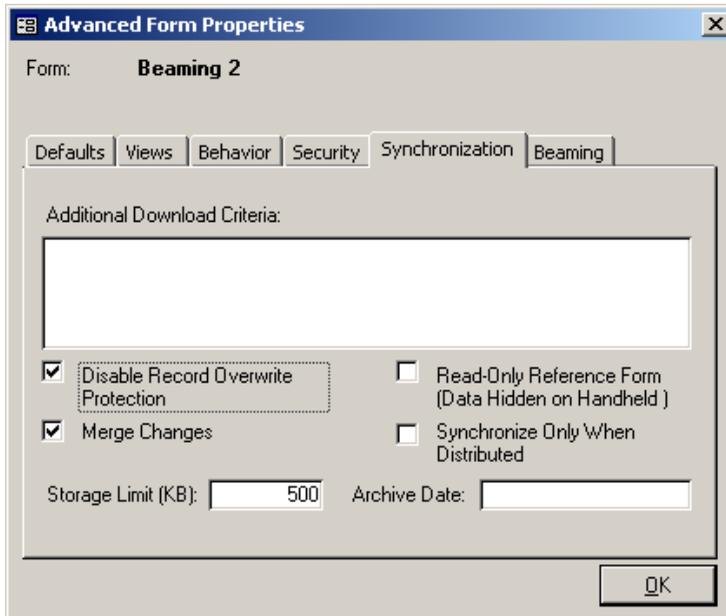
You will need to switch on the Advanced Form Property of Disable Record Overwrite Protection if you are going to allow users to beam records.

Disable Record Overwrite Protection is switched off by default. This option prevents a record that already exists in the database from being overwritten by a new record on the handheld with the same primary key. Primary keys are supposed to be unique, and generally, if an existing record already has a particular primary key, then no other record should have that primary key. If a new record on a handheld has that primary key, Pendragon Forms flags the record as having a conflict with an existing record.

Beaming records is an exception to the rule. If a record is created on a handheld, the record is marked as new on that handheld. If the record is then beamed to another handheld, the record is also marked as new on the second handheld. The first of the two devices to synchronize will send the record to the database on the PC. When the second device synchronizes, there will be a conflict of primary keys, because a seemingly new record on the handheld will match an existing record in the database. Since the record is actually the same record, it may be acceptable to allow the record on the handheld to overwrite the record in the database. To avoid a Pendragon Forms conduit error message warning about the conflict, switch on Disable Record Overwrite Protection.

See the next page and also page 219 for information on switching on Disable Record Overwrite Protection.

Disable Record Overwrite Protection is on the Synchronization tab of the Advanced Form Properties window. To access Advanced Form Properties, click the name of the form in the Forms Manager, then click the Properties button to view the Properties window, then click the Advanced Properties button. Merge Changes, the next item discussed, is on the Synchronization tab also.



Issue 8: Switch on Merge Changes

If you are allowing records to be beamed in order to permit one user to start filling in a form and another user to finish filling in different fields on the form, you will need to switch on the Advanced Form Property of Merge Changes.

If Merge Changes is switched off, which is the default setting, then the second handheld to synchronize will overwrite the entire record of the first handheld to synchronize. The overwriting would occur even if the first handheld to synchronize has more up-to-date information.

If users are filling in different fields of the form, then switching on Merge Changes will allow fields changed by the second handheld to synchronize, to merge into the existing record that the first handheld synchronized. If one user is filling in fields 1-5 on a form, and another user is filling in fields 6-10, it would not matter which user synchronizes first as long as Merge Changes is switched on.

The above picture shows the Merge Changes checkbox. See also page 220.

Memory Requirements when Beaming Records

If you are allowing records to be beamed, the handheld that is doing the beaming needs to have sufficient free memory in order to perform the beaming, and the receiving handheld has to have enough free memory to receive the records.

When the queue statement is used in a beaming script, the queued records are copied into memory, waiting to be beamed. This means that if, for example, you queue 500 KB of records to be beamed, your handheld has to have at least 500KB of free memory, in addition to the space that Pendragon Forms and the records are already using.

20. Working with Multiple Forms

Depending on the application that you are trying to build, you may need more than one form on the handheld to implement your solution. Pendragon Forms supports the following:

- *Parent and Subform*
A parent form and a subform (or child form) are used if for every parent record, you need to be able to create many child records. Visiting the same customer or patient more than once, or taking several readings from the same piece of equipment may require you to create a parent form for the information that stays constant (e.g. the customer's name or the equipment serial number), and create a subform for the information that changes (e.g. the details of a customer visit, or the equipment reading on a particular day). Details on creating a parent form and subform start on page 133.
- *Lookup to Another Form*
If you want to maintain a reference list, such as inventory item numbers and prices, or employee names and addresses, and you want to select from the list and copy the data into a form, then you need to maintain two forms and do a Lookup to Another Form. Details on creating a form that can do a lookup to another form are on page 123.
- *Lookup to a Read-Only Reference Form*
If you have more than 5000 records in a reference form, you can create a Read-Only Reference form instead of a regular form. The advantage is that on the handheld, a Read-Only Reference form has a fast lookup mechanism for finding a record. The disadvantages are that the fields in the Read-Only Reference form can only be Text or Numeric fields, and the Text fields must have the smallest possible fixed size. Another disadvantage is that the handheld user cannot see the individual records in the Read-Only Reference Form - he or she can only access records by performing a lookup to the Read-Only Reference form. Details on creating a Read-Only Reference Form start on page 127.

This chapter illustrates some examples of working with multiple forms. All scripting events and scripting statements mentioned in this chapter are explained in detail in Chapter 13, *Scripting Reference*, starting on page 245.

Example: Taking Orders on the Handheld

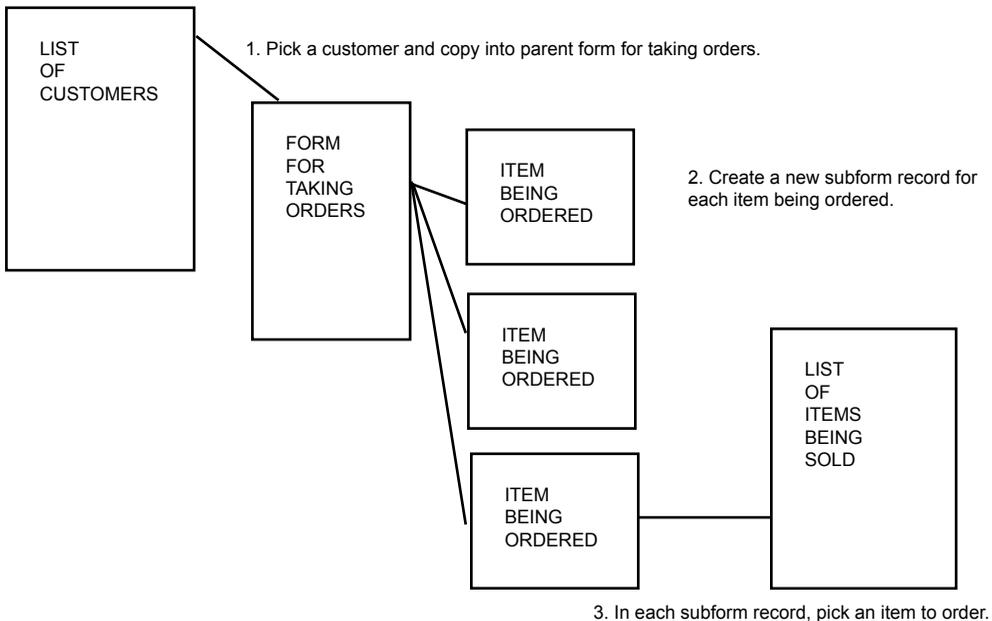
Creating an application to allow handheld users to take customer orders is an example that requires multiple forms.

To start with, you will need a form for taking orders. However, since you cannot predict whether a given order will be for one item or for several items, you may want to split the order-taking form into two: a parent form for the customer information, and a subform for the line items being ordered. A new subform record is created for each item being ordered.

If you have a list of customers, you may want the user to pick from a list to select a customer. To achieve this, you can do a lookup to a customer form from the order-taking parent form.

If you have a list of items for sale, you may want the handheld user to pick from the list to select an item being ordered. To achieve this, you can do a lookup from the subform to the list of items form.

The forms needed for this application and their relationship to each other is shown in the diagram below. Further details are on the pages that follow.



To perform a lookup from the parent form to the reference form:

- Fields that you want to copy from the reference form have to have the same field name and same field type as on the parent form. The only exception is that a Text field on the reference form can copy into a Lookup List field on the parent form - but not more than 50 characters.
- To display the list of records in the reference form, one field on the parent form must be a **Lookup List field**. Instead of referencing a Lookup List, reference the name of the reference form.

This is the Reference Form.

This is the Parent Form.

To allow the handheld user to perform a lookup to the reference form based on Company name, the Company field on the parent form is a **Lookup List field** that references the name of the reference form.

In Field View, the Company field on the Parent form looks like a Lookup List.

Tapping in the Company field in Record View, or tapping the Lookup button in Field View or Layout View displays the list of records from the reference form.

When a record is selected, all fields that are named the same and have the same field type on both the reference form and the parent form will copy into the parent form.

Once a customer has been selected from the reference form and copied into the order-taking parent form, a subform is used to select each item being ordered.

To create a parent and subform relationship between two form designs:

- One or more fields within the first 10 fields have to match in field name and field type on both the parent form and the subform. These fields will copy from parent to subform when a new subform record is created. Never change the values of the matching fields on the subform, or you will lose the link to the parent form.
- To jump from the parent form to the subform, the parent form must have a Subform List field that references the name of the subform.

This is the Parent Form.

1	Date of Order	8/16/01
2	Company	Trim & Zimm
3	First Name	Millicent
4	Last Name	Zimmerman
5	Address	42 Michiqan Ave
6	City	Chicago
7	State	IL
8	Zip Code	60101
9	Phone	312-555-8678
10	Remove order fr	<input type="checkbox"/>
11	Items ordered	<input type="checkbox"/>

Here the Items Ordered field on the parent is a Subform List field that references the name of the subform.

This is the Child Form or Subform.

Subform	
▼ Sub-Total	
8/16/01	
L Order Line Items (Subform)	
[Empty List Box]	
Add	Lookup:
Done	

Tapping the subform icons in this field displays a list of existing subform records. If there are no matching subform records, the list is blank.

Tap the Add button to create a new subform record. Tap an existing subform record to review it.

8/16/01		
1	Date of Order	8/16/01
2	First Name	Millicent
3	Last Name	Zimmerman
4	Company	Trim & Zimm
5	Item
6	Item Number
7	Price	\$0.00
8	Quantity to ord
9	Sub-Total	\$0.00
11	Special Instructi

When a new subform record is created, all matching fields within the first 10 fields of the parent copy into the subform record.

In this example, the first four fields of the parent form are copied into the subform.

To perform a lookup from the subform to a reference form of items in stock:

- Since this is a lookup from one form to another, fields that have to be copied must have the same field name and field type on the reference form and on the subform.
- One field on the Subform has to be a **Lookup List** field that specifies the name of the reference form.

This is the subform.

8/16/01	
1	Date of Order 8/16/01
2	First Name Millicent
3	Last Name Zimmerman
4	Company Trim & Zimm
5	Item
6	Item Number
7	Price \$0.00
8	Quantity to ord
9	Sub-Total \$0.00
11	Special Instructi

This is the reference form of items in stock.

Pretty White Toothpaste	
1	Item Pretty White
2	Item Number 1233
3	Price \$3.99

To perform the lookup, the Item field on the subform is a Lookup List field that references the Items in Stock form.

Tapping in the Item field displays the list of items in the reference form.

Selecting an Item name on the subform is copies the Item Number and Price from the reference form to the subform.

8/16/01	
1	Date of Order 8/16/01
2	First Name Millicent
3	Last Name Zimmerman
4	Company Trim & Zimm
5	Item
6	Item Number
7	Price \$0.00
8	Quantity to ord
9	Sub-Total \$0.00
11	Special Instructi

Lookup		Price
8/16/01		
L Items in Stock		
Blue Toothbrush		\$2.81
Extra White Tooth		\$2.99
Green Toothbrush		\$2.25
Mighty Light Toot		\$1.95
Pretty White Toot		\$3.99
Purple Toothbrush		\$3.00
Red Toothbrush		\$2.75
Super White Tooth		\$2.50

8/16/01	
1	Date of Order 8/16/01
2	First Name Millicent
3	Last Name Zimmerman
4	Company Trim & Zimm
5	Item Pretty White To
6	Item Number 1233
7	Price \$3.99
8	Quantity to ord 2
9	Sub-Total \$7.98
11	Special Instructi

Once the item details have been copied into the subform, the handheld user can enter the quantity of items being ordered. To calculate the sub-total for this line item of the order, a script is added in Field 9 of the subform:

```
calculate:
  if $8 > 0 then
    answer = $7 * $8
  endif
```

Advanced Scripting

There are some advanced scripting statements that are designed for working with multiple forms.

Some of the advanced scripting statements include:

also

column *number*

delete

gotosubform *formname* { **new** | **review** | **normal** }

insert into *formname*

keycolumn *number*

lookup *value* **within** *formname*

select all *formname*

select matching *formname*

select *formname* **where field** *field-number* **is** *expression*

update field *number* = *expression*

For details about these scripting statements, refer to Chapter 13, *Scripting Reference*, which starts on page 245.

Examples of forms that use some of these advanced scripting features are shown on the pages that follow.

Performing a Cascading Lookup to Another Form

In a regular lookup to another form, when the handheld user taps in the Lookup List field that specifies the reference form, all the records from the reference form are displayed.

Here the Company field is a Lookup List field that references another form. Tapping in the Company field shows a list of all companies.

Tapping on a company to select it copies all the commonly named fields from the reference form into the current form.

The first screenshot shows a form titled '8/20/01' with fields: 1 Date of Sales Cal 8/20/01, 2 Company, 3 First Name, 4 Last Name, 5 Address, 6 City, 7 State, 8 Zip Code, 9 Phone, 10 Items Discussed, 11 Date of next call - No Date -.

The second screenshot shows a 'Lookup' dialog titled 'State' with a list of companies: All Good Things (IL), Chopper Rescue (IL), CoolStuff (IL), Fine & Fancy (IL), KBM (OH), Minnow, Carly & As (IL), Techno Consultant (OH), Trim & Zimm (IL).

The third screenshot shows the same form as the first, but with the 'Company' field populated with 'Chopper Rescue', 'First Name' with 'P.', 'Last Name' with 'McOtter', 'Address' with '71 Seashore Vie', 'City' with 'Buffalo Grove', 'State' with 'IL', 'Zip Code' with '60089', and 'Phone' with '847-555-4321'.

If you do not want the handheld user to see a list of all the records in the reference form, you can use advanced scripting and re-design your form to allow for a cascading lookup to another form. A cascading lookup uses selection criteria to select only certain records from the reference form.

Important: A read-only reference form cannot be used in a cascading lookup.

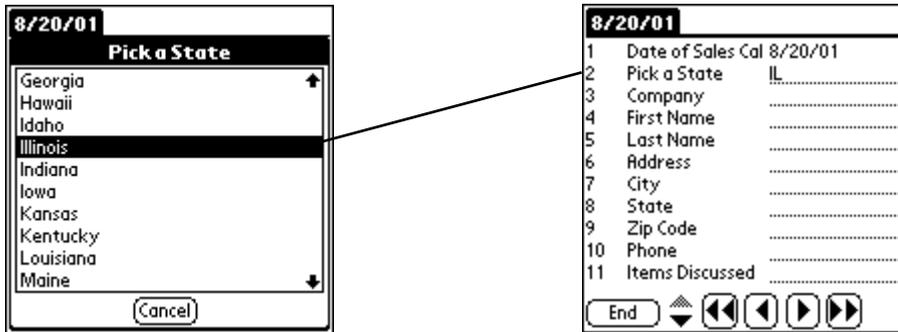
The first screenshot shows a form titled '8/20/01' with fields: 1 Date of Sales Cal 8/20/01, 2 Pick a State, 3 Company, 4 First Name, 5 Last Name, 6 Address, 7 City, 8 State, 9 Zip Code, 10 Phone, 11 Items Discussed.

The second screenshot shows a 'Pick a State' dialog with a list of states: Georgia, Hawaii, Idaho, Illinois (selected), Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine.

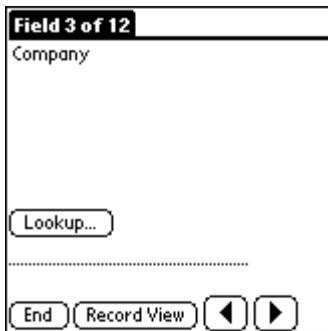
The third screenshot shows a 'Lookup' dialog titled 'State' with a list of companies filtered by the selected state (Illinois): All Good Things (IL), Chopper Rescue (IL), CoolStuff (IL), Fine & Fancy (IL), Minnow, Carly & As (IL), Trim & Zimm (IL).

In this form, selecting a State in field 2 determines what companies will be displayed in field 3. Advanced scripting is required to do this.

To achieve the cascading lookup to another form, an extra field has to be added to allow the user to specify the selection criteria. In this example, field 2 is a regular Lookup List containing a list of all states. The state that the user selects will be the selection criteria.



To perform the selection of records from the reference form, field 3, the Company field, is a **Lookup List field** that references the name of a reference form, "Customer Database". In addition, a click: event script is added to field 3. The click: event runs when the handheld user taps in the Company field, and the **select** statement in the script does the record selection.

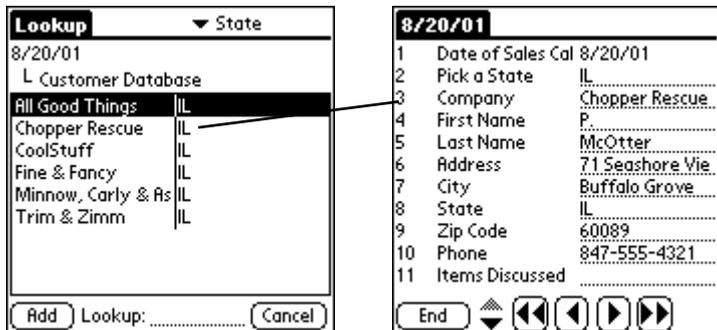


```
click:
select "Customer Database" where field 7 is $2
```

The script selects all the records in the Customer Database form, where field 7 of the Customer Database form matches field 2 of the current form.

In the Customer Database form, field 7 is State, so the select script is choosing all records where the State field is IL in this example.

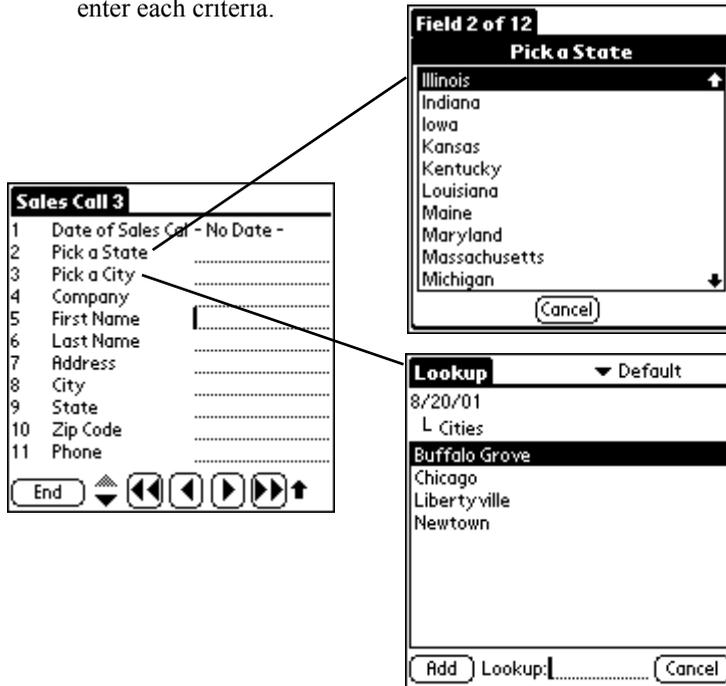
Once a record is selected, all the fields in the reference form that are named the same as fields on the current form are copied from the reference form to the current form.



Performing a Double Cascading Lookup

A double cascading lookup enables you to use two selection criteria to choose which records from a reference form are displayed when doing a lookup to another form.

- Two reference forms are needed for a double cascading lookup.
Important: The reference forms cannot be read-only reference forms.
- Since two criteria are being used, two fields are reserved to allow the handheld user to enter each criteria.



In this example, selecting a State in field 2 determines which cities are displayed in field 3. One reference form is used to lookup the City.

Selecting a city in field 3 determines which companies are displayed in field 4. A second reference form is used to lookup the Company. Only records that match on State and City are displayed.

Selecting a Company copies all the commonly named fields from the company reference form into the current form.



Here is how the double cascading lookup from the previous page is achieved:

Step 1:

To allow the handheld user to select a State, field 2, which is called “Pick a State” is a regular Lookup List field that contains a Lookup List of all the states. If necessary, this could have been made a lookup to another form, but in this case a regular Lookup List is sufficient.

Step 2:

To perform a lookup of cities, a reference form is created with two fields, one field for the city and one field for the corresponding state. There must be a record in this form for every city that the user might need to select.

Since the main form has a field called “Pick a City”, this reference form also has a field called “Pick a City”. This will allow the selected city to be copied into field 3 of the main form.

If you do not want to copy the state from this reference form into the main form (because the user has already selected a state in field 2), then name the state field something different from the main form. In this example, the state field is called “State for City” instead of being called “Pick a State” or “State” as in fields 2 and 9 of the main form.

Step 3:

To display only the cities within the state that the user selects, field 3, “Pick a City” on the main form is a **Lookup List field** that references the name of the “Cities” reference form. The following script is added to field 3 of the main form:

```
click:
  select "Cities" where field 2 is $2
```

Continued on next page...

Chopper Rescue	
1	Customer ID Number 4
2	Company Chopper Rescue
3	First Name P
4	Last Name McOtter
5	Address 71 Seashore Vie
6	City Buffalo Grove
7	State IL
8	Zip Code 60089
9	Phone 847-555-4321

End ⏪ ⏩ ⏴ ⏵

Step 4:

To perform a lookup of companies, a reference form is created with fields for all the data that is to be copied into the main form.

To copy the company name into the main form, field 2 on the reference form is called “Company”, to match the field name of field 4 on the main form. Other fields to be copied such as Address, City, State, Zip Code and Phone must be named the same and have the same field type on the reference form as on the main form.

8/20/01	
1	Date of Sales Cal 8/20/01
2	Pick a State IL
3	Pick a City Buffalo Grove
4	Company
5	First Name
6	Last Name
7	Address
8	City
9	State
10	Zip Code
11	Phone

End ⏪ ⏩ ⏴ ⏵ ⏶

Step 5:

To perform the lookup of companies that match both the City and State, field 4 on the main form is called “Company” and is a **Lookup List field** that references the name of the reference form, which is called “Customer Database” in this example.

To select only those records that match the City and State, the following script is added to field 4 of the main form:

click:

```
select "Customer Database" where field 7 is $2
also
select "Customer Database" where field 6 is $3
```

Field 7 of the “Customer Database” reference form is State, and field 2 of the current form is “Pick a State”, so the first **select** statement selects only the records that match on state.

The **also** statement means that the selected records are to be used when the next select statement in the script is run.

Lookup		▼ City
8/20/01		
L Customer Database		
Chopper Rescue	Buffalo Grove	
Fine & Fancy	Buffalo Grove	

Add Lookup: Cancel

The second **select** statement picks only the records where field 6 (“City”) in the “Customer Database” reference form matches field 3 of the current form, which is “Pick a City”. Since this statement only runs on the records that have already been selected on state, the effect is that only records matching the selected city and state are displayed.

Updating Records in a Reference Form

You may want to maintain records in a reference form, and use another form as the means by which the handheld user can update the reference form. An example might be a list of inventory items that you want the handheld user to update only if the quantity in stock needs to be modified, or if the item is not in the list.

Note: A Read-Only Reference form cannot be updated since it is read-only.

04963406
Barcode number 04963406
Item Name Coca-Cola 12 oz can
Quantity in Stock 30
Date of inventory 9/6/01

End ◀ ▶ ⬆ ⬇

For example, the picture to the left shows an inventory reference form. Field 1, the Barcode number, is made a primary key field to guarantee that each item has a unique barcode number.

A second form, called Barcode Inventory Checker and shown below, is used to scan inventory bar codes and compare the listed quantity in stock with the actual quantity on the shelf. If the quantity on the shelf is different, the handheld user can update the inventory reference form.

Barcode Inventory Checker
Scan item barcode 04963406
Item Description
Quantity in stock
Shelf count differe
Date of Inventory - No Date -

End ◀ ▶ ⬆ ⬇

Step 1: Scanning a bar code into field 1 runs a scan: event script that selects the record from the reference form that matches the barcode number.

Barcode Inventory Checker
Scan item barcode 04963406
Item Description Coca-Cola 12oz can
Quantity in stock 30
Shelf count differe
Date of Inventory 9/8/01

End ◀ ▶ ⬆ ⬇

Step 2: The item description and quantity in stock are copied from the reference form into the current form. If the quantity is correct, the handheld user can simply scan another item bar code.

04963406
Scan item barcode 04963406
Item Description Coca-Cola 12oz can
Quantity in stock 30
Shelf count differe
Enter correct count 36
Tap to update inve (Update)
Date of Inventory 9/8/01

End ◀ ▶ ⬆ ⬇

Step 3: If the handheld user sees that the quantity on the shelf is different from the listed quantity, he/she can check the checkbox in field 4 to flag the difference and fill in the correct shelf quantity. Tapping the Update button updates the reference form.

Continued on next page...

Here is how updating the reference form is achieved:

Step 1:

The Barcode Inventory Checker form has seven fields. Fields 5 and 6 are Hidden (see page 176), so that the handheld user does not see these fields unless the user selects Yes in field 4 to indicate that the shelf quantity is different.

Step 2:

A **scan**: event script in field 1 runs when a barcode is scanned. This is the script in field 1:

```
scan:

answer = scandata
$7 = now

select "Barcode Inventory List" where field 1 is $1
if result = 0 then
    $2 = null
    $3 = null
    msgbox
        "Item not found. Enter item description and shelf quantity."
    hide from 3 to 4
    show from 5 to 6
    goto 2
else
    show from 3 to 4
    $2 = column 2
    $3 = column 3
endif
```

The statement

answer = scandata

puts the barcode into the current field, namely field 1.

The statement

\$7 = now

puts the current date into field 7.

The statement

select "Barcode Inventory List" where field 1 is \$1

selects from the reference form called "Barcode Inventory List" the record whose bar code number matches that in field 1 of the current form.

When a select statement is performed, the **result** variable will contain the number of records that were matched. If the Result variable is zero, it means that a matching record has not been found in the reference form. The remainder of the script is an **if...then...else...endif** statement that will allow the user to create a new record if a match is not found, or will copy a matching record into the current form if a match is found.

Barcode Inventory Checker

Scan item barcode
 Item Description
 Quantity in stock
 Shelf count differe
 Date of Inventory - No Date -

Form Message

⚠ **Item not found. Enter item description and shelf quantity.**

OK

if result = 0 then

If a matching record in the reference form is not found, fields 2 and 3 are set to null to wipe out any existing values from previous scans.

A message box is displayed to alert the user to the fact the the item has not been found in the inventory reference form, and to notify the user that he/she will have to enter the item description and shelf quantity manually.

Fields 3 and 4 are hidden since they are not needed, and fields 5 and 6 are displayed.

else

On the other hand, if the **result** variable is not zero, then a matching record has been found in the reference form. (Note that since the bar code number field in the reference form is a primary key, each record has a unique bar code number and so we do not need to be concerned about multiple records matching on the same bar code.)

If a matching record is found, fields 3 and 4 are made visible, and columns 2 and 3 of the reference form record are copied into fields 2 and 3 of the current form. This copies the Item Description and Quantity in Stock from the reference form into the current form.

Barcode Inventory Checker

Scan item barcode 04963406
 Item Description Coca-Cola 12oz can
 Quantity in stock 30
 Shelf count differe
 Date of Inventory 9/8/01

End ⏪ ⏩ ⏴ ⏵

04963406	
Scan item barcode	04963406
Item Description	Coca-Cola 12oz can
Quantity in stock	30
Shelf count differe	<input checked="" type="checkbox"/>
Enter correct coun	
Tap to update inve	(Update)
Date of Inventory	9/8/01

End ⏪ ⏩ ⏴ ⏵ ⏶ ⏷

Step 3:

If a matching record from the reference form is copied into the current form, the handheld user can compare the listed Quantity in Stock (field 3) with the actual quantity of the item on the shelf. If the quantity on the shelf is the same, the user can simply scan a different item.

However, if the quantity on the shelf is different from the listed Quantity in Stock, the handheld user can check the checkbox in field 4, Shelf Count Different. Field 4 has a Default Value set to N for No. Selecting Yes in this field runs a script to display fields 5 and 6, which are needed to update the reference form.

The script in field 4 is:

```
select:
  if answer == Y then
    show from 5 to 6
    goto 5
  endif
```

04963406	
Scan item barcode	04963406
Item Description	Coca-Cola 12oz can
Quantity in stock	30
Shelf count differe	<input checked="" type="checkbox"/>
Enter correct coun	36
Tap to update inve	(Update)
Date of Inventory	9/8/01

End ⏪ ⏩ ⏴ ⏵ ⏶ ⏷

Step 4:

After the handheld user enters the correct shelf count in field 5, he/she must tap the Update button in field 6 for the changes to be copied to the reference form.

If there was no matching record in the reference form, a **click:** event script in field 6 will insert a new record into the reference form.

This is the click: event script in field 6 that runs when the user taps the Update button:

```
click:
  select "Barcode Inventory List" where field 1 is $1
  if result = 0 then
    insert into "Barcode Inventory List"
    update field 1 = $1
    update field 2 = $2
    update field 3 = $5
    update field 4 = $7
    show from 3 to 4
  else
    update field 3 = $5
    update field 4 = $7
  endif
  $1 = null
  $2 = null
  $3 = null
  $4 = N
  $5 = null
  hide from 5 to 6
```

The **select** statement selects the record in the “Barcode Inventory List” form that matches the bar code number in field 1. After the select statement is run, the **result** variable contains the number of matching records.

The **if...then...else...endif** statement performs different actions depending on two possibilities: if there is no matching record (if result = 0 then), then a new record is inserted into the reference form and fields 1, 2, 3 and 4 of the reference form record are updated with information from fields 1, 2, 5 and 7 of the current form. On the other hand, if a matching record is found, then only fields 3 and 4 of the reference form are updated from the current form.

The remainder of the script after the if...then...else...endif statement nulls out the values in the current field to give the handheld user the impression that tapping the Update button performs the update and then re-starts a blank form to scan the next inventory item. In reality a new record is not being created, the same record on the Barcode Inventory Checker form is being re-used.

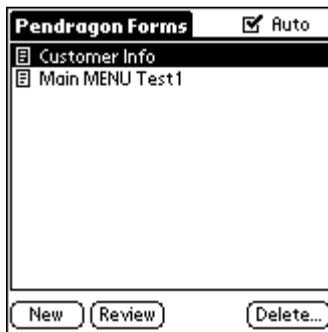
Troubleshooting Problems when working with Multiple Forms

Refer to Appendix B, *Troubleshooting*, page 539, for troubleshooting information related to working with multiple forms.

21. Creating a Custom Main Menu

A Custom Main Menu is a form that acts as a Main Menu and replaces the standard Pendragon Forms main menu screen that shows the list of form designs.

Standard Forms Main Menu



Custom Main Menu Form



Features of a Custom Main Menu Form

A Custom Main Menu form is a Pendragon form with two special attributes:

1. The form is launched as soon as the user runs Pendragon Forms, by setting the Advanced Form Property of Auto-Execute.
2. The name of the form must include the word MENU in uppercase. No more than one record will be created for the form.

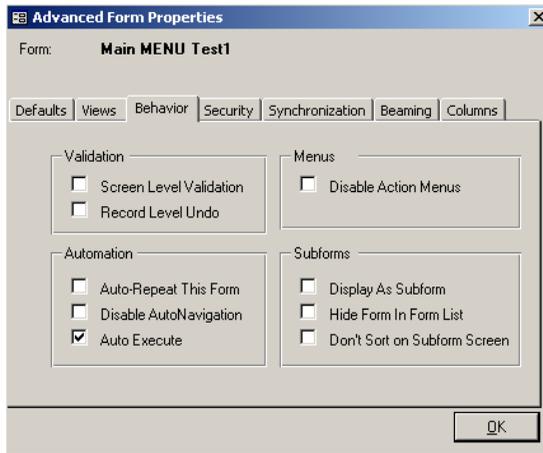
Typically, Custom Main Menu Forms feature Button fields or Section fields (typically with pictures) that are used to direct the user to different forms.

These items are discussed in this chapter.

Item 1: Set the Custom Main Menu form to Auto-Execute

In order to make a Custom Main Menu form run when the user taps the Forms icon on the handheld, the Advanced Form Property of Auto-Execute must be switched on.

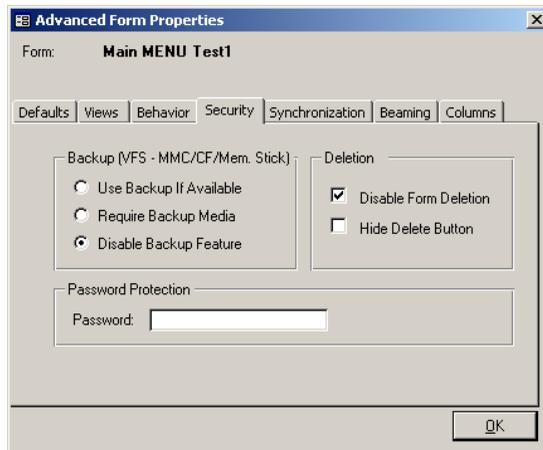
To access the Advanced Form Properties screen, click the name of the form in the Forms Manager, then click the Properties button, then click the Advanced Properties button.



On the Behavior tab of the Advanced Form Properties window, check the Auto-Execute checkbox.

Only one form can have the Auto-Execute form property switched on.

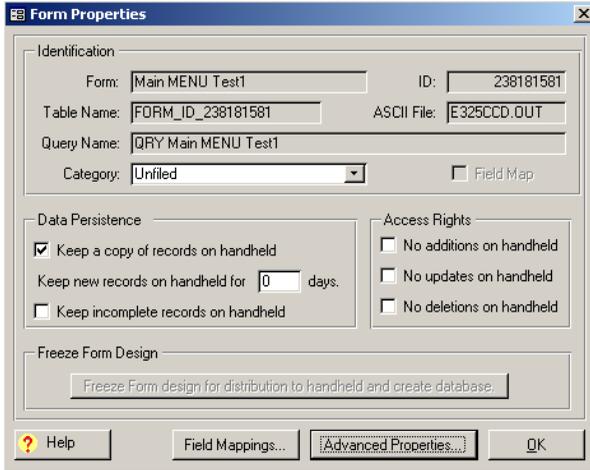
Other Advanced Form Properties to set



If you are allowing users to delete records from the handheld, it is a good idea to disable deletion of the Custom Main Menu form and all the other forms that you are using.

On the Security tab of the Advanced Form Properties window, check the Disable Form Deletion checkbox. Repeat for each form design that is being sent to the handheld.

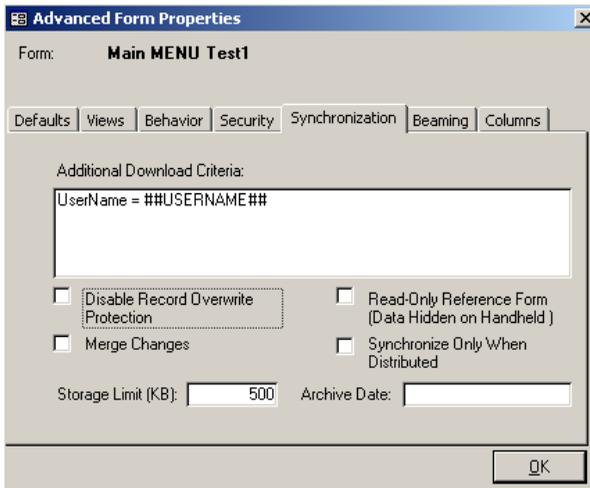
Another Advanced Form Property that you can set is on the Views tab: you can hide the End button of the Custom Main Menu form.



On the Form Properties screen of the Custom Main Menu form, check the Data Persistence option to:
Keep a copy of records on handheld.

When this option is set, there will only be one record per user created for the Custom Main Menu form itself.

If you choose to remove records from the handheld (no checkboxes checked), then a new record will be created each time each handheld user enters the Custom Main Menu. As the Custom Main Menu form does not usually contain any data, it does not make sense to keep adding records to this form.



On the Synchronization tab of the Advanced Form Properties screen of the Custom Main Menu form, leave the default Additional Download Criteria of:

UserName = ##USERNAME##

When this option is set, each handheld will only receive the one record created in the Custom Main Menu by that user.

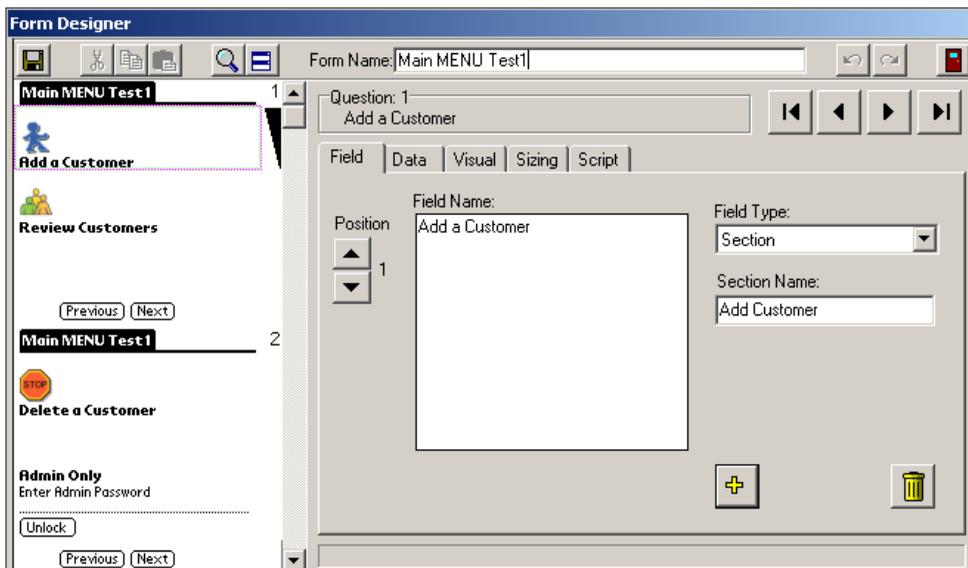
If this default additional download criteria were to be removed, each handheld user would receive the Custom Main Menu record for every other use, which would be a waste of storage memory on the handheld.

Item 2: The form name must include the word MENU

Only one form can be a Custom Main Menu form. To identify this form, the form name must include the word MENU in uppercase letters.

The word MENU in the name of an auto-executing form causes two things to occur:

- Only one record will be created for the form. This allows the user to 'enter' the Custom Main Menu form without generating a new record every time the menu is accessed.
- The Custom Main Menu form is displayed before the standard Pendragon Forms main menu, so the user never sees the list of forms, just the Custom Main Menu form.



Creating Custom Menu Options

For each option on the Custom Main Menu, create a Section field or a Button field. You can add a picture to the field - see page 171.

The activities that users will typically need to access from a custom menu are:

- Add a new record to a form.
- Review existing records for the form.
- Optionally, the ability to delete records from the handheld.
- Synchronize the handheld (applies to Pocket PC devices and wireless Palm OS devices).
- In an emergency, have a way to access the standard Pendragon Forms main menu screen.

You will need to create a field to serve as the menu option for each of these activities.

Click: event scripts in each Section field allow the activities to be performed.

Creating Separate Menu Options for Adding Records and Reviewing Records

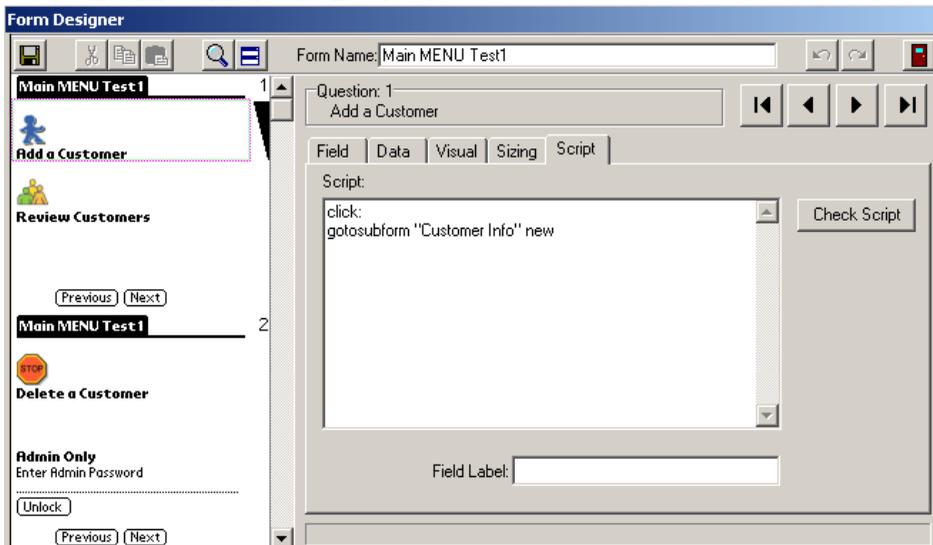
The **gotosubform "formname" new** statement jumps the user to the specified form, and creates a new record for that form.

The following script is in field 1 of the form shown below (the 'Add a Customer' menu option):

```
click:
```

```
gotosubform "Customer Info" new
```

means that when the user taps in the 'Add a Customer' Section field, the script will go to the form named "Customer Info" and create a new record.



To create a menu option for reviewing existing records of a form:

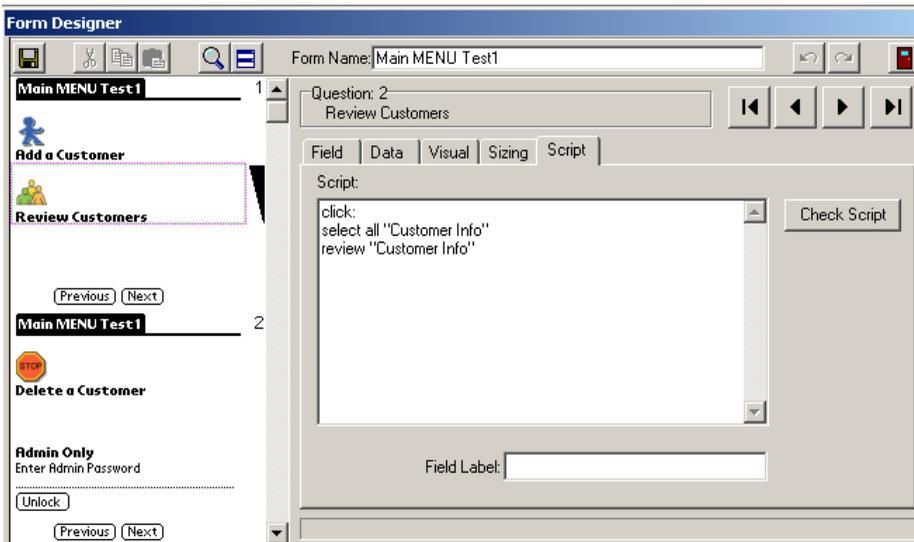
The **select all** "*formname*" statement selects all the records of the specified form.

The **review** "*formname*" statement lets the user review the selected records for the specified form.

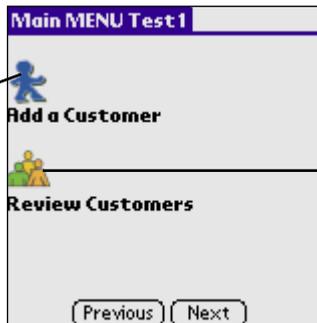
The following script is in field 2 of the form shown below (the 'Review Customers' menu option):

```
click:
  select all "Customer Info"
  review "Customer Info"
```

means that when the user taps in the 'Review Customers' Section field, the script will select all the records of the form named "Customer Info" and display a Review screen. The user can tap on a record on the Review screen to go to that record.



On the handheld, tap the option to Add in order to create a new record.



Or tap the Review option to review existing records.

Customer Info	
Account #	100
First Name	Julie
Last Name	Robbins
Company Name	Seaside Treasures
Contact Phone Number	847-555-8929
Fax Number	847-555-8930
E-Mail Address	julie@seaside-treasures.net

Customer Info	
Account	Company Name
102	Lookin' Good Fellas
101	Eddie's Tavern & Bar
100	Seaside Treasures

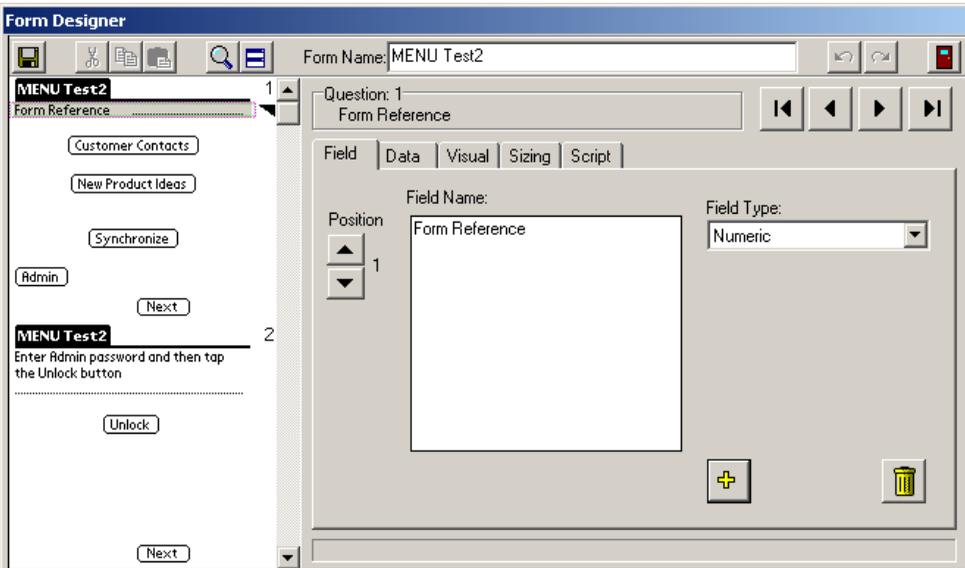
Creating One Menu Option for Both Adding Records and Reviewing Records

Button fields and subforms can be used to create a custom menu option that allows the handheld user to jump to a form and then once in that form, have the choice of reviewing existing records or creating new records.

If a subform is being used, then the Custom Main Menu form will act as the parent form, and the rules governing parent forms and subforms will apply (see page 133). In particular:

- In order to link the parent form to the subform, the two forms must have at least one field in common - (that is, the same field name and field type) - within the first 10 fields of the form. The common field can be hidden on both parent and subform.
- The linking field must be filled in on the parent form in order to automatically copy down to the subform. One way to achieve this is to set a Default Value in the field on the parent form. (See page 105 for information on the Default Value field property.)

In the picture below, field 1, called Form Reference, is going to be used as the linking field between parent and subform. The field is a Numeric field that is made hidden, and a value is pre-populated by setting a Default Value of 1.

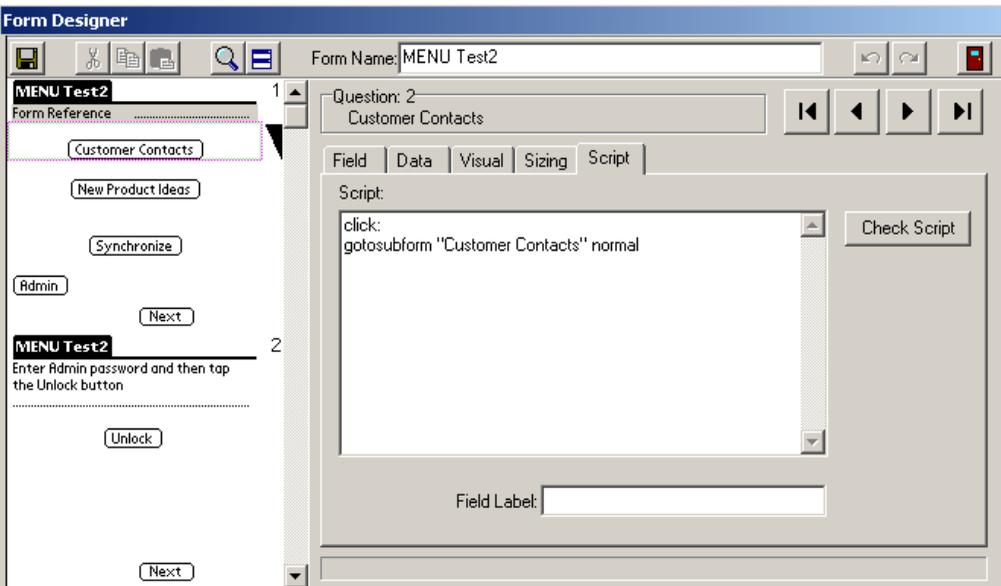


A Button field is added to the custom menu form to allow the handheld user to jump to each subform.

The scripting statement `gotosubform "formname" normal` jumps the user to the specified subform.

In the picture below, the script in field 2 is:

```
click
gotosubform "Customer Contacts" normal
takes the user to a subform called Customer Contacts.
```



Notes on the Custom Main Menu form:

- The End button and the Previous button can be hidden by setting the appropriate Advanced Form Properties for the form. (See page 204).
- If the handheld user only needs to see the first screen of the Custom Main Menu form, you can use a script to hide the Next button as well. The script in field 1 is:

```
open:
disable nextbutton
```

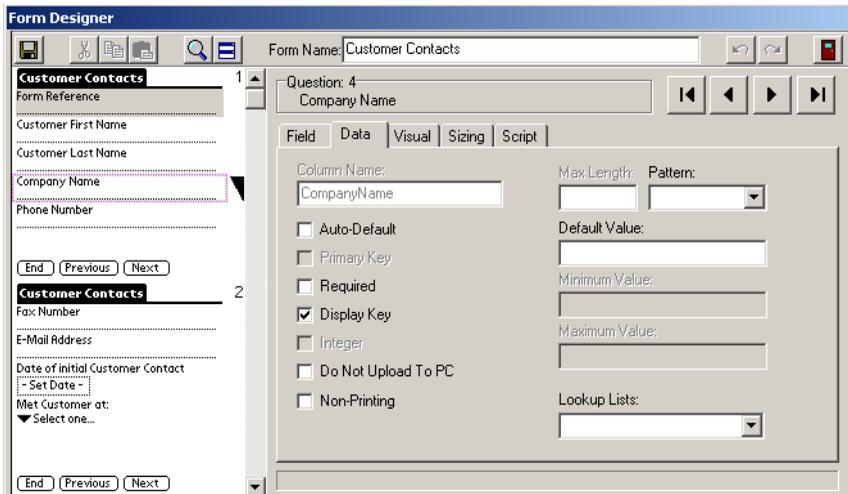
If you needed to show the Next button on some screens and not on others, you could instead use **enterscreen:** events to **disable nextbutton** or to **enable nextbutton**.

On the subforms that you create for the custom menu form, you must add at least one field that the subform has in common with the parent form. The linking field must have the same field name and field type as on the parent form, and must occur within the first 10 fields of the subform.

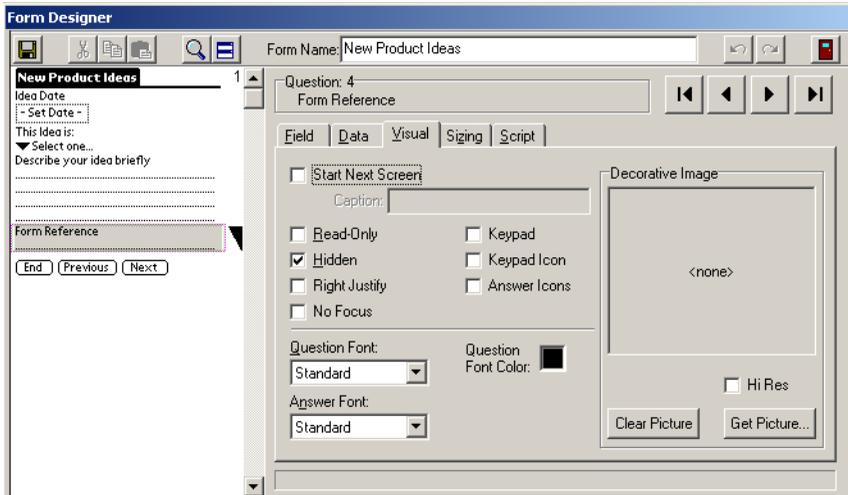
In the Subform A picture below, the field that is used to link with the parent form is field 1, called Form Reference, which is made hidden. Field 1 of a form is typically used for displaying records when reviewing a form, but since field 1 is hidden in this case, a different field on the form is set as the Display key field. In this instance, field 4, Company Name is set as the Display key field.

In the Subform B picture below, the linking field between parent and subform is not the first field on the subform.

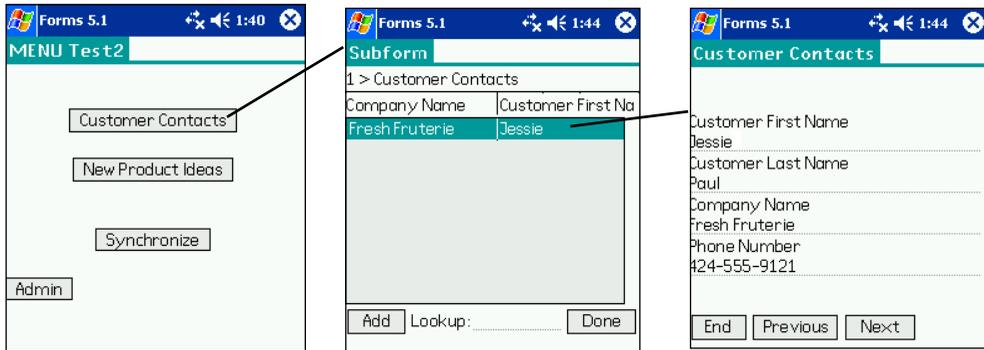
Subform A:



Subform B:



On the handheld, tapping a button jumps the user to a subform. The user can tap on an existing record to review that record, or tap the Add button to add a new record. Tapping the End button on the subform returns the user to the subform review screen, and tapping Done on the review screen returns the user to the custom menu form.



Creating a Menu Option to Delete records from a form

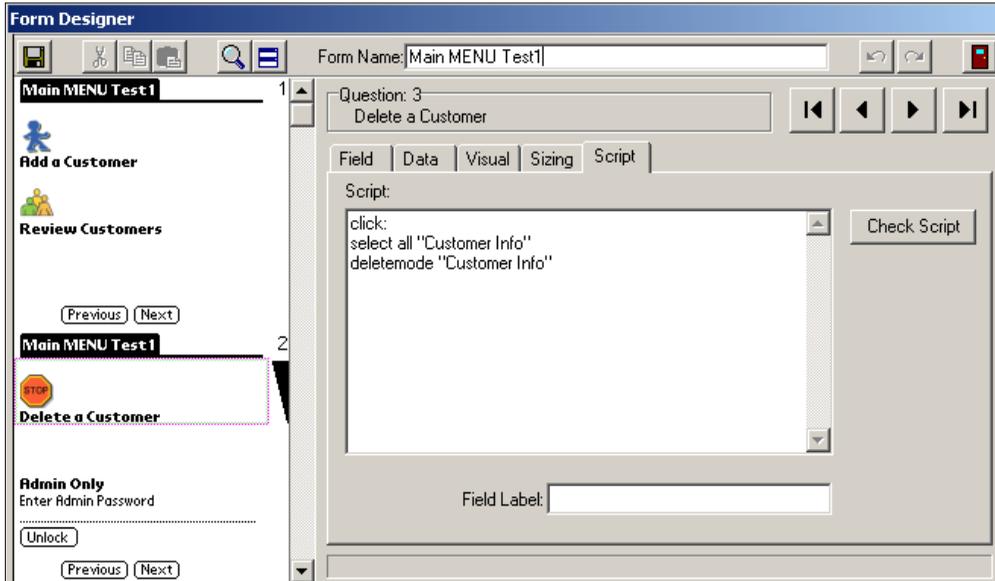
The **select all** "*formname*" statement selects all the records of the specified form. The **deletemode** "*formname*" statement lets the user access a Delete screen to delete records.

The following script is in field 3 of the form shown on the next page (the 'Delete a Customer' menu option):

```
click:  
  select all "Customer Info"  
  deletemode "Customer Info"
```

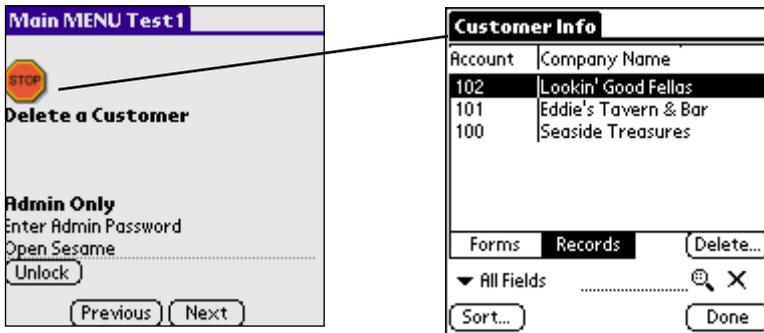
means that when the user taps in the 'Delete a Customer' Section field, the script will select all the records of the form named "Customer Info" and display a Delete screen. The user can tap on a record on the Review screen to go to that record.

WARNING: The Delete screen allows users to delete form designs as well as records. To prevent users from accidentally deleting form designs, set the Advanced Form Property of **Disable Form Deletion** for each form that the handheld user is using, including the Custom Main Menu form itself. See page 212 for information on setting the Disable Form Deletion option.



If deletion is allowed, tapping an option to delete a record will jump to a Delete screen to select a record to delete.

Note that the Delete screen also permits the deletion of form designs, which is why it is recommended that you set the Advanced Form Property of Disable Form Deletion on the Custom Main Menu form, as well as on all other forms that you are installing on the handheld. (See page 212.)



Creating a way to access the standard Pendragon Forms main menu

With a Custom Main Menu form, the handheld user has no way to access the standard Pendragon Forms main menu screen that shows the list of forms.

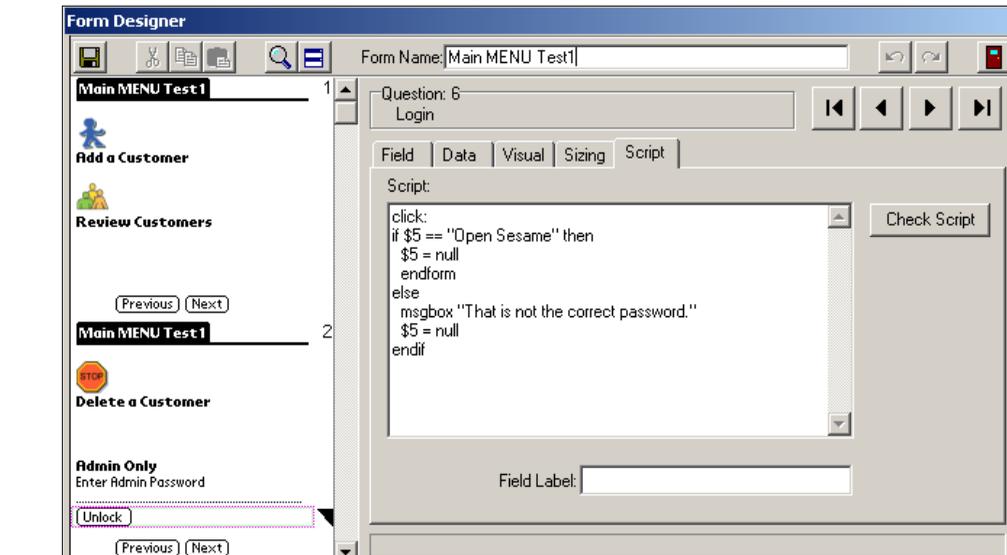
It is strongly recommended that you add to your Custom Main Menu form a way to access the standard Pendragon Forms main menu. That way, if there is a problem on the handheld, you can direct the user to revert to the standard Pendragon Forms main menu to troubleshoot things like verifying how many form designs are on the handheld, verifying what the form ID numbers are, and accessing VFS Backup features to perform a restore of data from an external media card.

A simple way to allow limited access to the standard Pendragon Forms main menu is to use two fields: a Text field to store a password, and a Button field to validate the password.

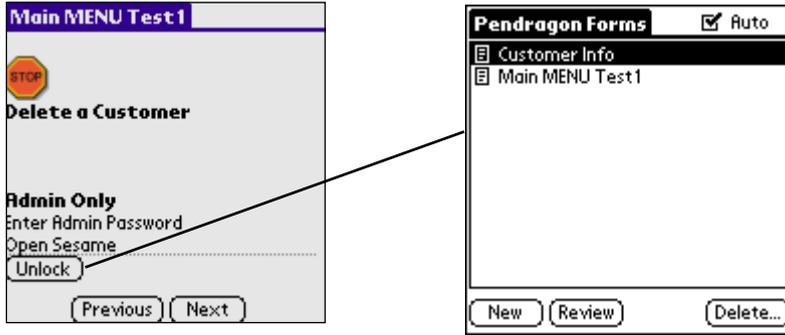
In the form shown below, the script in field 6, the Button field, is as follows:

```
click:
  if $5 == "Open Sesame" then
    $5 = null
  endform
else
  msgbox "That is not the correct password."
  $5 = null
endif
```

If field 5 (the password field) is equal to the phrase "Open Sesame", the field is set to null to erase the password, and the **endform** statement ends the current form, which is the Custom Main Menu form. The user reverts to the standard Pendragon Forms main menu. If the password is not correct, a message is displayed. **Note:** Please use a more complex password than "Open Sesame"!



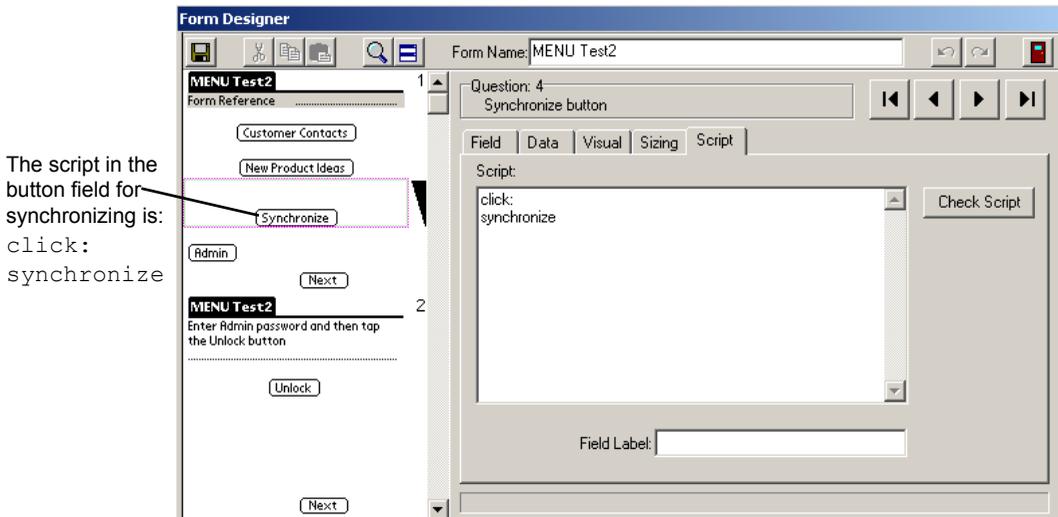
If you have set up an administrative password to access the standard Pendragon Forms Main Menu, then entering the password and tapping the button to unlock the Custom Main Menu form will end the Custom Main Menu form and revert to the standard Pendragon Forms Main Menu that shows the list of forms on the handheld.



Creating a Custom Menu Option to Synchronize Pendragon Forms

If your handheld users are using Pocket PC devices, or if they are using Palm OS devices that will synchronize wirelessly, you can add a Button field to your Custom Main Menu form to initiate synchronization. Palm OS devices that connect to the PC via synchronization cable or cradle do not need a custom synchronization button, as those users can just press the HotSync button on the cable or cradle to synchronize.

If you do not create a synchronization button on your Custom Main Menu, Pocket PC users and wireless Palm OS users will need to tap the S-Link application to synchronize.



Using a Custom Main Menu form to Filter Records

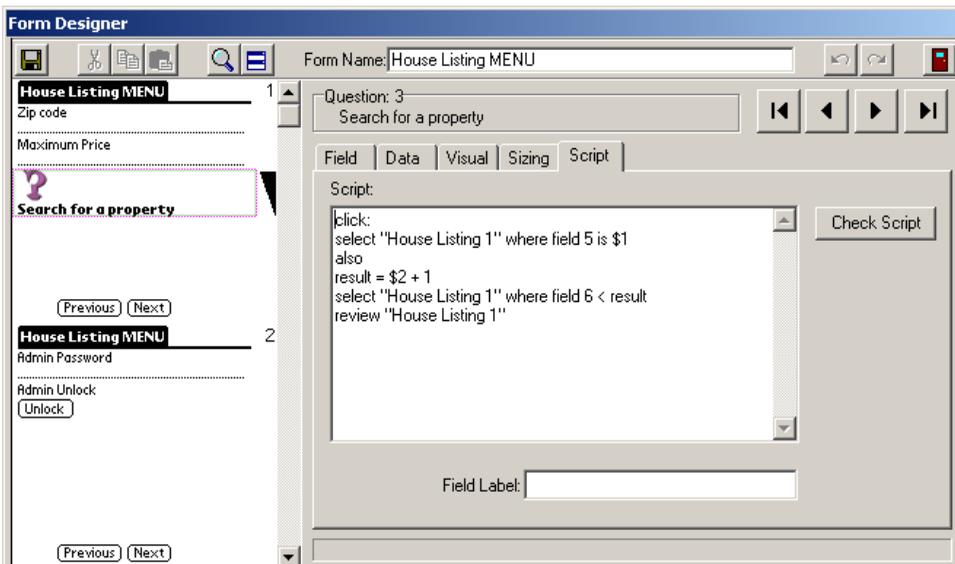
If records of a form are pre-loaded onto the handheld, a Custom Main Menu form can be set up as a way to filter the records. That is, the Custom Main Menu form acts as a place to enter selection criteria.

The **review** statement preserves any **select** statement filters that have been set by a script. The **also** statement is used to combine select statements.

In the form below, field 1 stores a Zip code, and field 2 stores a Maximum Price that a potential customer wants to pay for a house. Field 3 is a Section field with a **click:** event that lets the user view records based on the criteria set up in fields 1 and 2. The script in field 3 is:

```
click:
  select "House Listing 1" where field 5 is $1
  also
  result = $2 + 1
  select "House Listing 1" where field 6 < result
  review "House Listing 1"
```

In this script, "House Listing 1" is the name of a form that contains one record per house on sale. Field 5 of "House Listing 1" is a Zip code, and field 6 is the Asking Price of the seller. The first **select** statement in the script selects all records in "House Listing 1" that match the zip code entered on the Custom Main Menu form. The **also** statement saves the currently selected records, and applies the next select statement on top of the existing selected records.

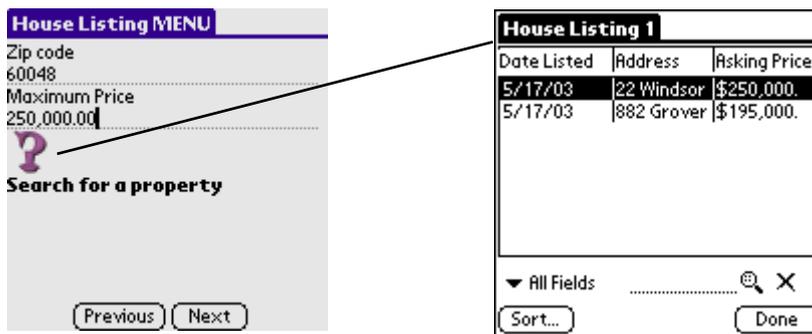


The Custom Main Menu form refers to a Maximum Price of the buyer, and so to include the maximum price in the search, the result = \$2 + 1 statement sets the **result** variable to \$1 more than the maximum.

The second **select** statement then applies a filter in which records are selected if the seller's Asking Price is less then the result **variable**.

The **review** statement then displays the selected records, that is, all records within a certain Zip code and where the seller's asking price is less than the buyer's maximum price.

On the handheld, when the user enters a Zip code and a maximum buyer's price and then taps the Section field to perform a search, a list of available houses that meet the criteria are displayed. The user can tap on a record to select it and view details.



22. Linking to an External Access Database

With Pendragon Forms it is possible to link to an external Microsoft Access 2000, Access 2002 (Access XP) or Access 2003 database instead of the Pendragon Forms database. This feature is particularly useful if you have your own existing Access database and you want to send data from your database to the handheld.

Here are some important issues to consider when linking to your own Access database:

Does my database table have a primary key?

A primary key is a field or combination of fields that uniquely identify a record. No two records can have the same value for the primary key.

Pendragon Forms uses a primary key to tell if a record on the handheld is new or if the record already exists in the external Access database. If the primary key field(s) of a record are not in the database, then the record on the handheld is new and should be added to the database during synchronization. If the primary key field(s) of a record are in the Access database already, then the record on the handheld is not new, and during synchronization the existing record in the database should be updated.

Warning: If the database table in your existing Access database does not have a primary key, you will not be able to update any records on the handheld. Refer to your Microsoft Access documentation on how to select a primary key for a database table.

Whatever primary key you use in your database table, the same primary key must be used in the corresponding Pendragon Form. The primary key can be one field such as a Customer ID#, or a combination of fields such as a Customer Account Number and a Date of Visit.

- To prevent primary keys from accidentally being corrupted by the handheld user, any existing records which are sent from the PC to the handheld will have the primary key field(s) set to being read-only on the handheld.

Will the handheld users be allowed to create new records?

If the user can create new records, the uniqueness of primary keys has to be protected, so that new records on the handheld do not overwrite existing records on the PC.

- The Pendragon Forms conduit compares new records on the handheld to existing records in the database during synchronization. If a new record has a primary key which is the same as an existing primary key, the new record will not be sent to the database, and will be flagged on the handheld. The user can then change the record on the handheld to have a unique primary key.

How will you limit the number of records which get sent to the handheld?

Since the handheld has limited memory compared to a PC, there must be a mechanism for removing records from the handheld, or for determining which handheld receives which record.

- The Advanced Form Properties screen allows you to specify the Additional Download criteria that will determine which records from the database are sent to the handheld. You will need to specify the Additional Download Criteria before you link the Pendragon form to your external database table.

When linking to a parent form and a subform, the parent form must be created first.

If you have referential integrity rules in your Access database, whereby a child record cannot be created before a parent record, you will need to take care with the order in which you create a parent form and subform in Pendragon Forms.

When you create a form design, the Form ID# that is associated with the form is based on the creation date and time of the form. During synchronization, the Pendragon Forms conduit synchronizes forms in the order in which they were created, that is, starting with the smallest Form ID#. If you create a subform before a parent form, the Forms conduit will attempt to synchronize the subform first. If you are linking to an external Access database with referential integrity rules, this is equivalent to creating a child record before the parent record, and Access will give an error during synchronization.

The solution in this case is simply to create your parent form in Pendragon Forms before you create the subform. If you have already created the subform first, make a copy of the subform. The copied subform will have a later creation date than the parent form, and will therefore synchronize after the parent form. In the Subform List field of the parent form, you will need to make sure that you are referencing the copied subform and not the original.

Which Pendragon Forms field types can link to which Microsoft Access field types?

Although the Pendragon Forms Manager is an Access database itself, there are more field types in Pendragon Forms than there are in Microsoft Access. This gives you more flexibility when entering data on the handheld. However, if you want to link your Pendragon form to an external Access database, you will need to make sure that the fields on your Pendragon form are linking to an appropriate Access field type in your database table.

Here are the Access field types and their compatible Pendragon Forms field types:

Access Field Type	Compatible Pendragon Forms Field Type
Text (up to 255 characters)	Text (up to Max Length 255 characters) Popup List Lookup List Exclusive Lookup List Option 1 to 5 Yes/No Checkbox Completion Checkbox Custom Control
Memo	Text (with Max Length 2000 characters) Custom Control
Number: Byte	Numeric (with Min = 0, Max = 255, Integer = checked) Slider (within the range above)
Number: Integer	Numeric (with Min = -16384, Max = 16383, Integer = checked) Slider (within the range above)
Number: Long Integer	Numeric (with Min = -2147483648, Max = 2147483647, Integer = checked) Slider (within the range above)

Access Field Type

Compatible Pendragon Forms Field Type

Number: Single Precision

Numeric or Slider with a specific range.
Slider with a specific range. (must have Integer = checked)
Not recommended if you are creating an Access table from scratch, because the handheld has more precision than a single-precision field in Access. Double precision is preferred.

Number: Double Precision

Numeric
Slider (must have Integer = checked)

Yes/No

Yes/No Checkbox (with a default value of Y or N, according to the convention used in your external database table)

A default value is required because Access does not accept a null value in a Yes/No field, whereas in Pendragon Forms on the handheld the user can leave a Yes/No checkbox blank.

Currency

Currency

Date/Time

Date Only
Date & Time
Time
Time Checkbox

AutoNumber

Numeric

OLE Object

Signature (150x50 monochrome pixels)
Sketch
Image

Sample existing Access database

In order to illustrate linking Pendragon Forms to an existing Access database, we will consider putting the following simple work order database on the handheld:

Work Order : Table	
Field Name	Data Type
Ticket number	AutoNumber
Date of call	Date/Time
Customer Last Name	Text
Customer First Name	Text
Address	Text
City	Text
State	Text
Zip code	Text
Problem	Text
Technician	Text
Date of technician visit	Date/Time
Resolution	Text
Billed hours	Number
Rate per hour	Currency
Total Billed amount	Currency
Work Order Status	Text

In this database, the Ticket Number field is the primary key and is an Autonumber field. This allows each work order to have a unique number.

In Microsoft Access, a form has been created for someone to enter new work orders into the system. Customer information is entered, and the work order is assigned to a technician.

This information now needs to be sent to the handheld so that the technician can complete the work order in the field.

The screenshot shows a 'Work Order' form with the following fields and values:

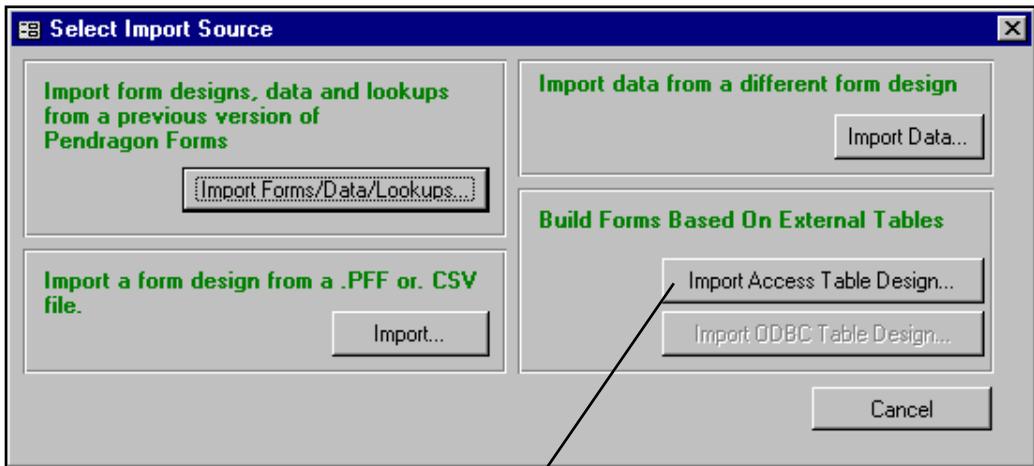
- Ticket number: 19
- Technician: Sarah Jane
- Date of call: 6/14/2003
- Date of technician visit: (empty)
- Customer Last Name: Melvin
- Resolution: (empty)
- Customer First Name: Jerome
- Billed hours: 0
- Address: 15 Larkspur Lane
- Rate per hour: \$0.00
- City: Libertyville
- Total Billed amount: \$0.00
- State: IL
- Work Order Status: New
- Zip code: 60048
- Problem: Clanking noise heard when unit is switched on.

Record: 19 of 19

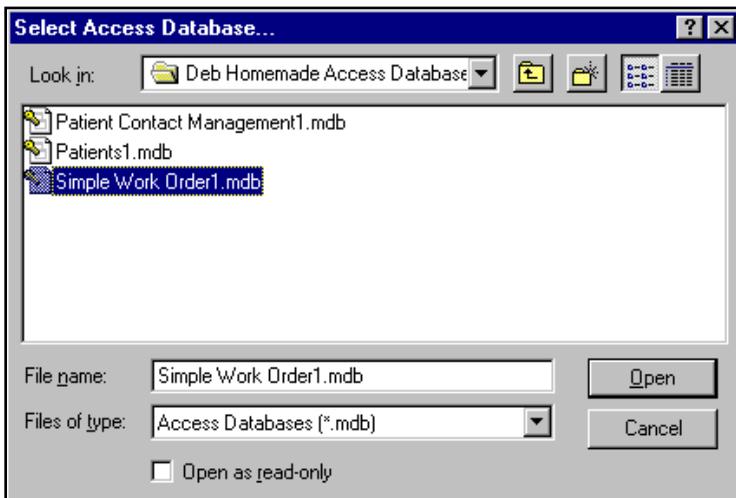
Step 1: Creating a form from an existing database table

The first step in the process of linking to an existing Access database is to create a Pendragon form which is based on the specific Access database table to which you want to link.

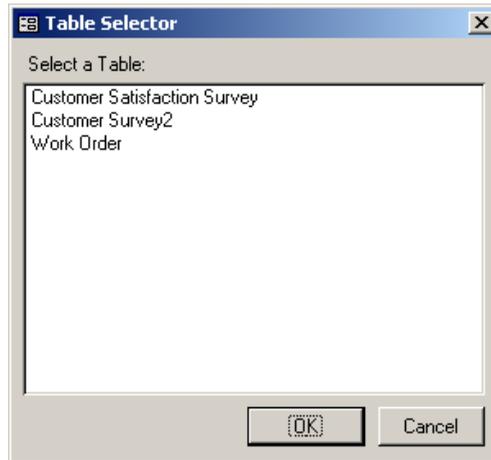
- 1a. In the Pendragon Forms Manager, click the Import button.



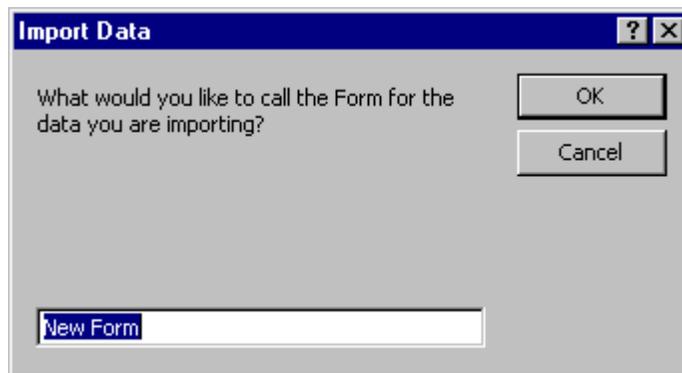
- 1b. Click the Import Access Table Design button. In the Select Access database window, choose the Access .mdb file to which you want to link.



- 1c. A Table Selector window appears, displaying a list of the database tables and queries within the selected Access database. Click on a table name and click OK.



- 1d. An Import Data window appears, prompting you for the name that you want to give the corresponding Pendragon form. Enter a name for the form and click OK.



- 1e. In the Forms List within the Pendragon Forms Manager, click on the name of the newly created form, then click the **Edit** button. In Step 2 on the next page, the form design will be edited for displaying on the handheld.

Step 2: Editing the form design for use on the Handheld

The Pendragon form that was created from an external Access database table may require editing.

2a. In the Form Designer window, you can edit and delete fields as follows:

- Fields that you do not want to put on the handheld can be deleted from the form design.
- Adjust the layout of fields in the Form Designer, for example, by making the question and answer components of a field fit on the same line, or occupy several lines.
- The names of the fields are based on the database column names from the Access database table. These can be modified to be more readable to the handheld user, for example a column name such as FName can be expanded to First Name.

WARNING: Although you can change the field names, you should not change the database column names in the Advanced Field Properties window. The database column names are used to map to the existing Access database.

2b. Make sure that the fields in your form design that are marked as primary key fields match the primary key fields in your external Access database table.

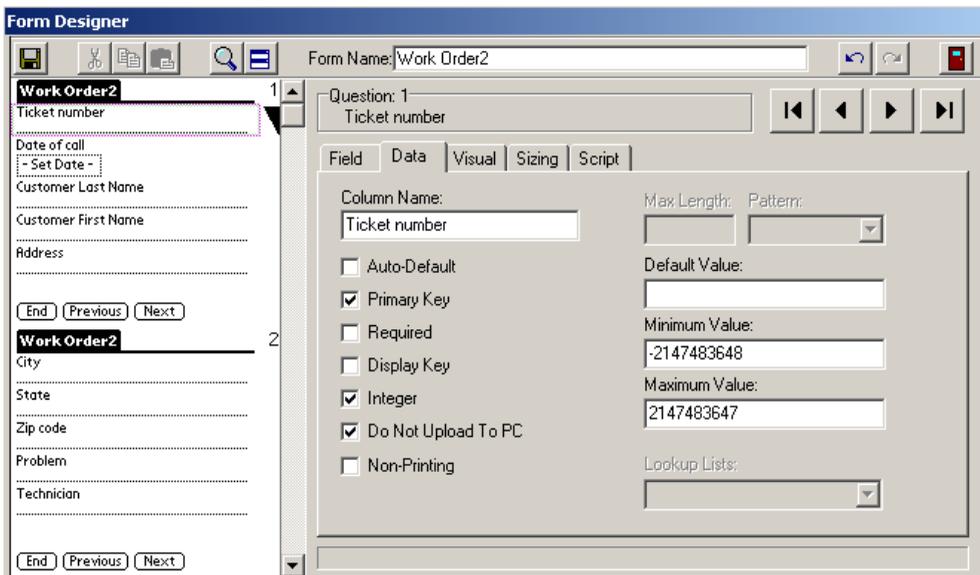
If you imported the form design from an external Access database table that already had primary keys, then the primary key fields in your form design should match the primary key fields in your external Access database.

If you are creating your Pendragon form before creating your external Access database table, you will need to manually check the Primary Key checkbox for each field that is part of your primary key. If you are allowing the handheld user to create new records on the handheld, then you should also set the Advanced Field Property of Required, to ensure that the handheld user fills in the primary key field(s) for every new record.

- To check whether a field is marked as a primary key field in Pendragon Forms, display the field in the Form Designer window and then click the Data tab. Make sure that the Primary Key checkbox is checked.
- In your external Access database, if you view your database table in design view, all the primary key fields will be marked with the icon of a key to the left of the column name.

If the primary key in the existing database table is an Autonumber field, Pendragon Forms treats the field as a special case on the form. The Autonumber field will be populated by Microsoft Access when records are sent to the existing database. However, because the Autonumber field is the primary key, records which are created on the handheld will need to have a unique value in this field at least temporarily until a number is assigned from the PC.

Pendragon Forms automatically makes some special settings in the Advanced Field Properties window to make it possible to store a unique number on the handheld which will be discarded when a new record is sent to the PC.

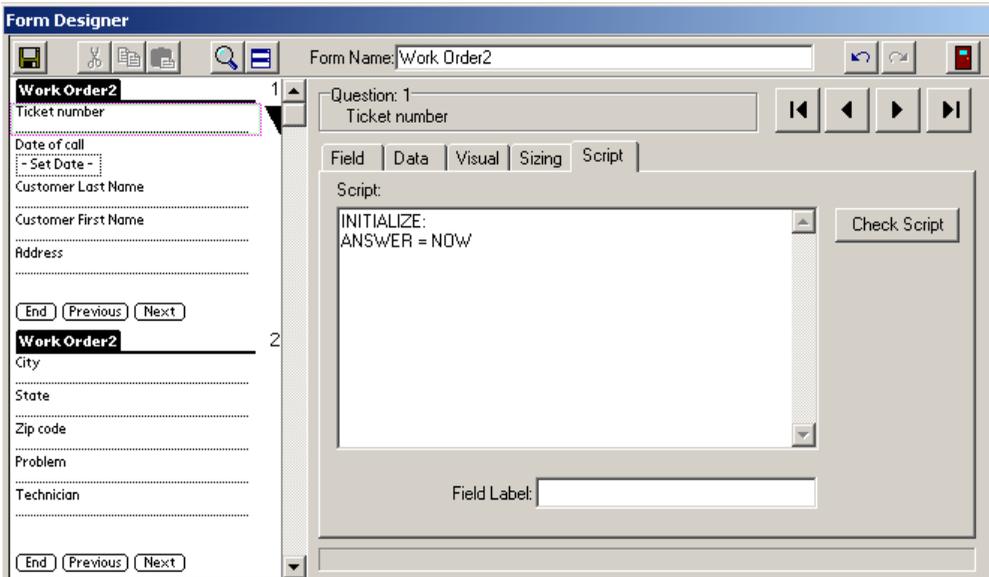


Primary key - this option is checked to indicate that the Autonumber field is the primary key on the form.

- More than one field on a form can be primary. For example, a customer database may use Customer Account Number and Date of Visit as the combined primary key. If a primary key field is not an Autonumber field, and you are going to allow the user to create new records on the handheld, then make the primary key fields Required fields, but do **not** make them read-only.
- If an Autonumber field is the primary key, and this field is going to be filled in via a script (see next page), you can set the primary key to be **Read-Only** to prevent accidental corruption of the primary key on the handheld.

Do Not Upload to PC - this is a special option, used only with AutoNumber fields. When new records are created on the handheld, a unique value will be assigned in the Autonumber field, but this value will not be sent to the PC. This will allow the database on the PC to assign a number according to the sequence in the existing database.

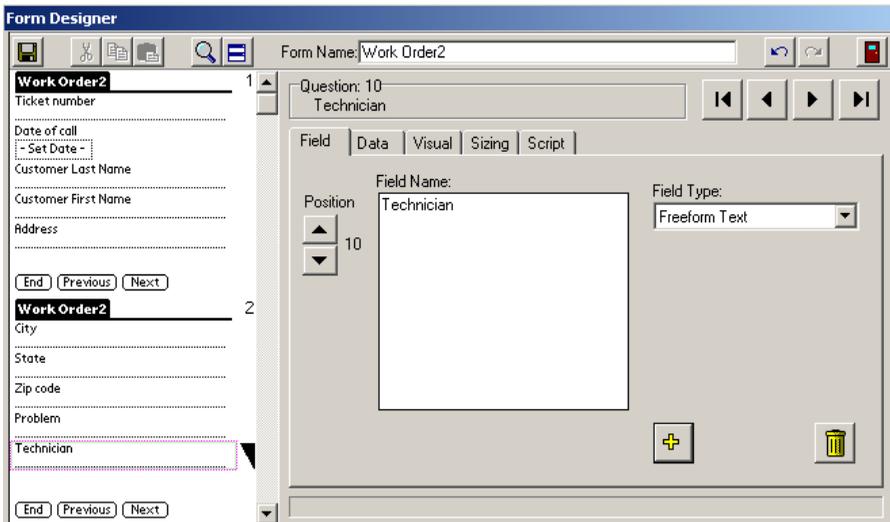
- To assign a unique value on the handheld, Pendragon Forms automatically adds the following script to the Autonumber field:



- An **initialize:** script only runs when a new record is created. By setting the value in the Autonumber field to **now**, a unique number is stored in the field. (The number is equal to the number of seconds since 01/01/1904). This number is just used in order to have a unique value on the handheld until the record is sent to the PC and is assigned a unique number by the Autonumber field in the database.

- 2c. If you need to control which handheld user receives which database record, then there should be a column in your existing database for storing the handheld user name, that is, the name associated with a given handheld unit.

In our database example, the Technician field serves this purpose and allows the person entering data on the PC to assign which record will go to which handheld.



In this example the Technician field is displayed on the form so that the handheld user can see that his/her name has been assigned. (The field can be made Read-Only to prevent the handheld user from assigning someone else.)

It is not actually necessary have a field on the Pendragon form for this information, as long as there is a field in the database which can be used to determine the criteria for downloading records. (See Step 3.)

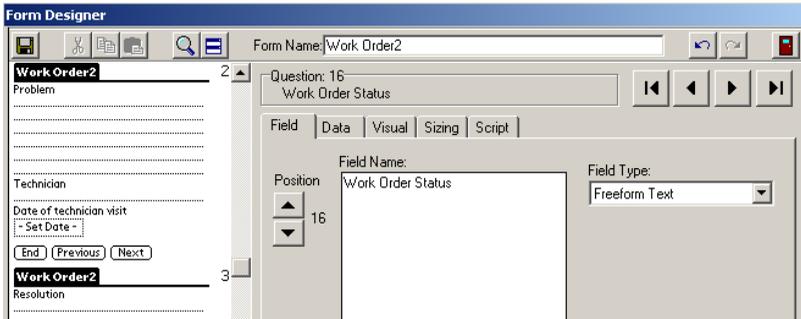
Every Pendragon form has three fields which are generated automatically: UnitID, UserName and TimeStamp. The UserName field is the handheld user name, and the TimeStamp field is the creation date and time of each new record on the handheld.

If your existing database does not have a column for storing the handheld user name, and you want to create one, you can create a column called UserName. If you also want to store the creation date and time of each handheld record, create a column in your database called TimeStamp.

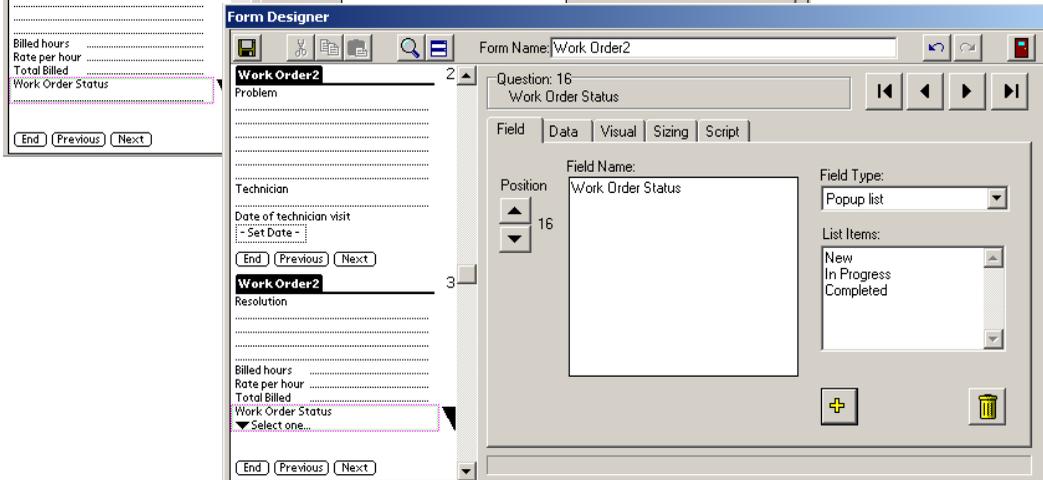
- 2d. Some field types can be changed in the form design, to promote accurate and efficient data entry on the handheld unit.

In this example, a field called Work Order Status was initially assigned to be a Text field type when the form design was created from the database table. However, if for example there are only three possible Work Order Status values, then the field should be changed to a Popup List or Lookup List to ensure that the handheld user always enters an appropriate value in the field.

Before:



After:



Check the fields on your form to see where data entry on the handheld can be optimized by using Popup Lists, Lookup Lists or Yes/No checkboxes instead of Text fields. (Your external database table must have compatible data types. For example, you would not want to change a field type to a Date field if the external database column is a Yes/No field.)

When you are satisfied with your form design, close the Form Designer window.

Step 3: Controlling which records go to the Handheld

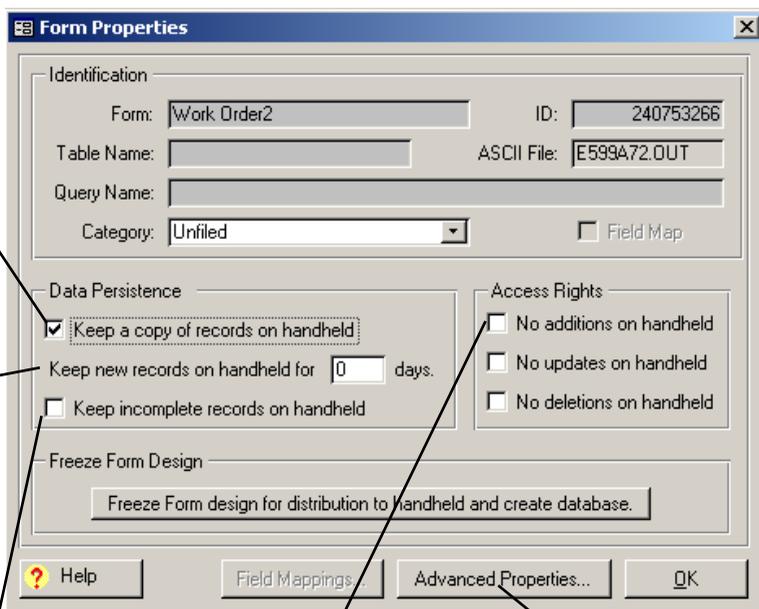
Since the storage space on the handheld is limited, it is very important to specify the rules which will govern which records are sent from the PC to the handheld, and when records will be removed from the handheld.

- 3a. In the Pendragon Forms Manager, click on the name of your form, and then click the Properties button.

In the Data Persistence section, check the box *Keep a copy of records on handheld*. This allows records to be sent to the handheld.

The option *Keep new records on handheld for X days* can only be used if the external database has a TimeStamp column for storing record creation dates.

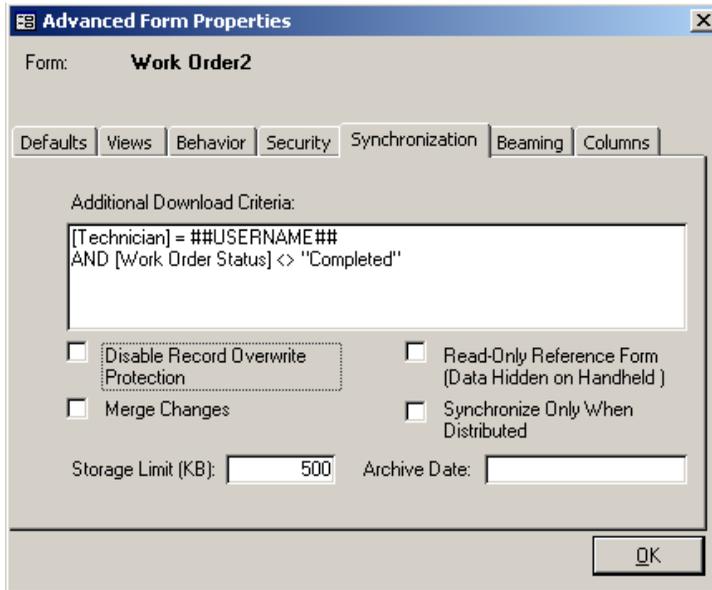
The option *Keep incomplete records on handheld* can only be used if the Completion checkbox field on the Pendragon form is mapping to a Text field in the external database.



If you do not want the handheld user to create new records on the handheld, check the box *No additions on handheld*. This would be the case if new records are always generated on the PC.

To specify which records are sent to the handheld, click the Advanced Properties button.

- 3b. The Advanced Form Properties window is where you specify the download criteria for which records get sent to the handheld, and which records are removed from the handheld.



The Additional Download Criteria field stores an SQL WHERE statement which will restrict the records that get sent to the handheld.

##USERNAME## is a special wildcard which Pendragon Forms uses to represent the handheld user name. [Technician] = ##USERNAME## means that the Technician field in the database has to match the handheld user name for the record to be sent.

[Work Order Status] <> "Completed" means that if a field called Work Order Status is not set to Completed, then the record is sent to the handheld. If the handheld user changes the Work Order Status to Completed, the record will be sent to the PC during the next HotSync data transfer, and then be removed from the handheld.

The general format of the WHERE clause is:

[Database-Column-Name] = "Criteria"

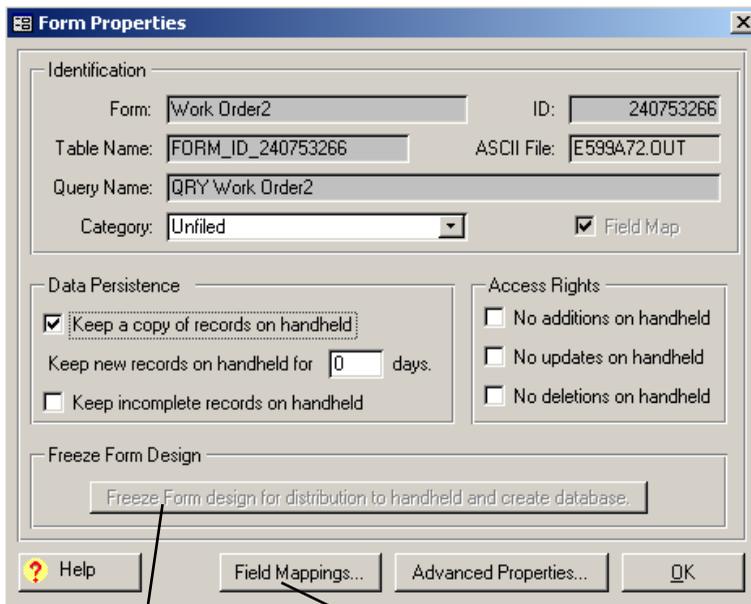
See *Additional Download Criteria* on page 214 for more information on WHERE clauses.

Step 4: Linking the form to the external Access database

Up to this point, the form that has been created based on the existing Access database table is not yet linked to the database table. Once the form properties and Advanced form properties have been selected, the form can be frozen and then linked to the external Access database table.

- 4a. Before you freeze your form design, make sure that the field(s) that you have selected as primary key fields in your form design are also the primary key field(s) in your external Access database.

- 4b. In the Properties window, freeze the form design.



Click on the Freeze form design button to freeze the form design.

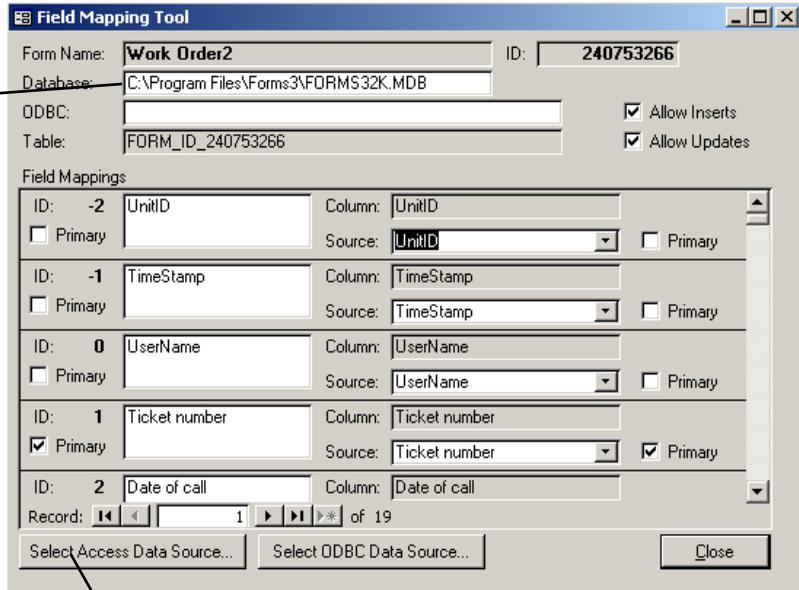
Once the form is frozen, the Field Mappings button becomes accessible.

IMPORTANT!
If you change Advanced Form Properties after doing Field Mappings, you will need to re-do Field Mappings.

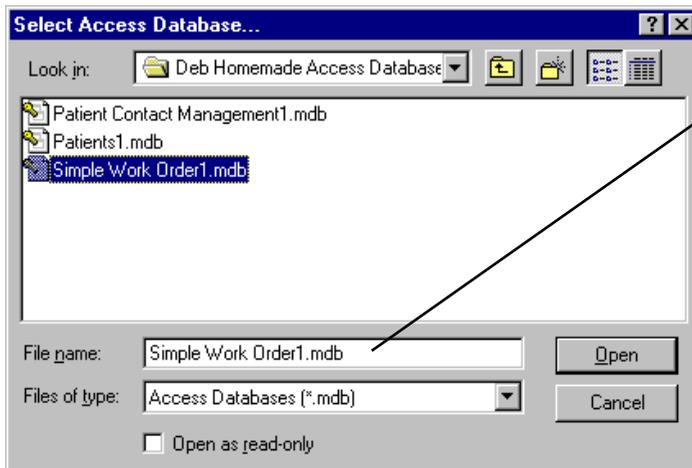
Click the Field Mappings button to link the Pendragon form to your Access database table.

- 4c. The Field Mappings window allows you to determine which field on your Pendragon Form will be stored in which column of your external database.

The default database is the Pendragon Forms database. At this point, if you do not make any changes, then data from the handheld would be sent back to the Pendragon Forms database like a regular form which is not linked to an external Access database.



To link the form to an external Access database table, click the Select Access Data Source button.



Select the external Access database name, and click the Open button.

Then you will be prompted to select the specific database table or query to link to.

- 4d. Once an external database table has been selected, you can verify that the fields on your Pendragon Form are mapping to the correct columns in your database.

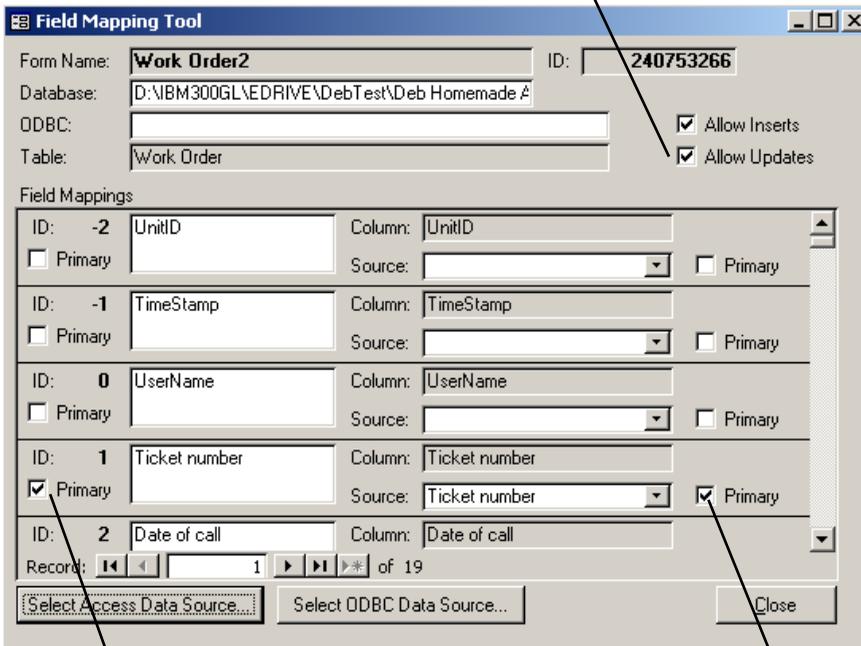
The Column name is the Pendragon form column name.

The Source field is the column from your external database.

If you created your form from the Access database table initially, then the Columns should be mapped to the correct Source. Scroll down the list of fields to verify this.

The UnitID, UserName and TimeStamp fields are implicit on the handheld. Normally, when linking to an existing Access database table with your own primary key, these fields will be unbound, meaning that data in these fields will not be sent to the external database table. However, in the example below, the UserName column in the Pendragon form is being mapped to a column called Technician in the external Access database.

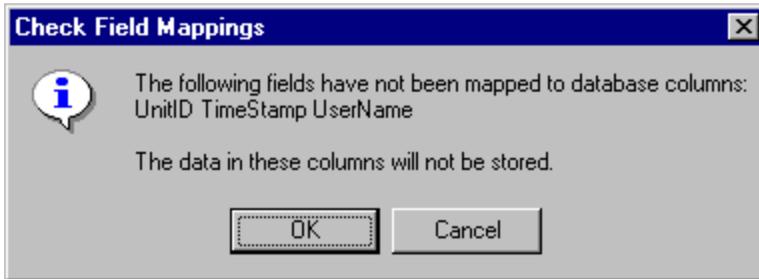
Checking the Allow Inserts checkbox means that you want to allow the handheld user to create new records on the handheld. The Allow Updates checkbox means that database records are allowed to be updated on the handheld.



The Primary key(s) on the form must match the primary key(s) in the external Access database table. If they do not, you will need to modify either the form design or the external Access database table so that the primary keys match. Then re-do the field mapping.

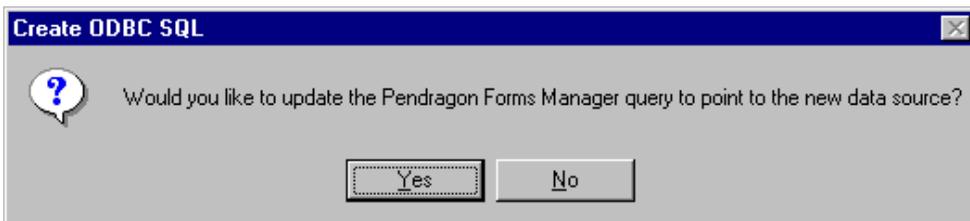
When you click the Close button on the Field Mapping window, you will be prompted to save changes to the field mapping. Choose Yes.

If your database has its own primary key and you do not have columns in your database for storing the UnitID, UserName and Timestamp fields that Pendragon Forms creates with every record, you will also receive the following dialog:



This dialog is informing you that the UnitID, UserName and TimeStamp fields will be unbound, meaning that data in these fields will not be stored in your external Access database. Choose OK. As long as you have your own primary key in the external Access database, and that primary key matches the primary key fields in your Pendragon form design, you do not need to keep the UnitID, UserName and TimeStamp fields. If you want to retain this information, then your external database has to have columns for storing the UnitID, UserName and TimeStamp and you will need to re-do the field mapping.

Another dialog window that is displayed is:



Choose Yes. Pendragon Forms will attempt to link the use of the Edit/View button to your external database, so that if you click Shift + Edit/View you can view the data in your external database from within the Pendragon Forms Manager. There are some limitations - see Viewing Data on the PC, page 505.

Step 5: Sending the form to the Handheld

Once field mappings have been set, the form is ready to be sent to the handheld.

- 5a. Close the Field Mappings window, then close the Properties window.
- 5b. Click on the name of the form, then click the Distribute button.
 - The form will be placed in the Default group. If you are using user groups, click on the Groups button and assign the form to a user group.
- 5c. On the next HotSync data transfer, the form will be sent to the handheld, as well as any records from the database which have been selected for that handheld unit.

These fields were filled in from the Access database.

These fields are filled in on the handheld.

Work Order2	
Ticket	19
Date of call	6/14/03
Last Name	Melvin
First Name	Jerome
Address	15 Larkspur Lane
City	Libertyville
State	IL
Zip code	60048
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

Work Order2	
Resolution	Cleaned unit. Replaced ball bearings and re-balanced rotor.
Billed hours	2
Rate per hour	\$45.00
Total Billed	\$90.00
Work Order Status	▼ Completed
<input type="button" value="End"/> <input type="button" value="Previous"/> <input type="button" value="Next"/>	

Work Order			
Ticket number	19	Technician	Sarah Jane
Date of call	6/14/2003	Date of technician visit	6/14/2003 12:37:00 PM
Customer Last Name	Melvin	Resolution	Cleaned unit. Replaced ball bearings and re-balanced rotor.
Customer First Name	Jerome	Billed hours	2
Address	15 Larkspur Lane	Rate per hour	\$45.00
City	Libertyville	Total Billed amount	\$90.00
State	IL	Work Order Status	Completed
Zip code	60048		
Problem	Clanking noise heard when unit is switched on.		
Record: 19 of 19			

When the handheld user fills out the Pendragon Form and synchronizes, the record updates the database record in the external Access database.

Step 6: Viewing data on the PC

If you are linking to an external database, and during the field-mapping process (see page 503) you selected to update the Pendragon Forms query to point to the new data source (i.e. your external database), then if you click on the name of the form in the Forms Manager and use the Shift + Edit/View button, you may be able to view the data in your external database table from within Pendragon Forms.

If you just use the Edit/View button and not Shift + Edit/View, the UnitID, UserName and TimeStamp fields will be visible but will not contain any valid data if your external database does not store the data in these fields. Another limitation of using Edit/View is that if a column name in Pendragon Forms does not match the column name to which you are field mapping, then the data for that column will not be visible from within Pendragon Forms.

Shift + Edit/View hides the UnitID, UserName and TimeStamp fields from view. However, a limitation of using Shift + Edit/View is that if you try to edit a record from within Pendragon Forms, fields that are Popup Lists or Lookup Lists will not display any lists.

- The best way to view the data in the external database is to open the external database directly, instead of trying to view the data from within Pendragon Forms.

Troubleshooting Tips

Refer to Appendix B, *Troubleshooting*, page 541, for tips on problems that can occur when linking to an external Access database.

23. Linking to an ODBC Database

In addition to being able to link to external Microsoft Access databases, Pendragon Forms supports linking to external ODBC databases such as SQL Server, Oracle, and Informix. The process of linking to an external ODBC database is similar to linking to an Access database, with a few extra steps.

There are two possible ways to link to ODBC databases:

- Option 1: Creating a linked table in Access
- Option 2: Mapping directly to an ODBC table

Option 1: Creating a Linked Table in Access

Before a link can be created, a machine Data Source Name has to be created.

To create a machine Data Source Name:

- i. In Windows, click on Start...Settings...Control Panel. In the Control Panel, double-click on the 32bit ODBC icon.
- ii. An ODBC Data Source Administrator window appears. Click on the System DSN tab. Check if a Data Source Name already exists for the database to which you want to link. If a Data Source Name already exists, you do not need to create one.
- iii. If you need to create a Data Source Name, click the Add button.
- iv. You will need to select a driver (e.g. Oracle driver), enter a name for the Data Source Name, and select the data source (i.e. the path and specific database name to which you want to link.) Different drivers may require different information. ODBC drivers usually ship with a Help file. Refer to the manufacturer of the driver for further information.

For performance reasons, Microsoft recommends creating a linked table in the Access database (i.e. the Pendragon Forms Manager). Once you have created a machine Data Source Name, you can go into the Pendragon Forms program and link to the Data Source Name, as described on the next page.

Do the following to create a linked table in the Pendragon Forms database that links to your Data Source Name.

- a. Open the Pendragon Forms Manager. Bring the Forms3: Database window to the foreground. (If you cannot see this window, click on Window...Cascade.)
- b. Click on the File menu, then choose Get External Data...Link Tables.
- c. A Link window appears. Select to view files of type ODBC databases. A Select Data Source window appears. Click on the Machine Data Source tab, then double-click on the Data Source Name that you previously created.
- d. If you have to enter a password to access the external database, you will be prompted for the password.
- e. A Link Tables window appears, displaying a list of database tables within the selected database. Click on a database table and click OK.
- f. If your database table does not have a primary key, you will be prompted to select one. The primary key is used to uniquely identify each record, and will be used on the handheld.
- g. In the Forms3: Database window, click on the Tables tab. Look for the name of the database table to which you just linked. An arrow next to the name indicates that this is a link.
- h. Click on the database table name, then click the Open button to view the database table. Verify that you are linking to the correct table and are able to view the records that you want to send to the handheld. Then close the database table window.
- i. Bring the Pendragon Forms Manager window to the foreground. Click on the Import button, then click Import Access Table Design.
- j. Choose to open C:\Program Files\FORMS3\FORMS32k.mdb, and select the name of the linked database table.
- k. Follow instructions on pages 492-505, Step 1c to Step 6. When doing Field Mappings in Step 4, click the Select Access database button, and again select C:\Program Files\FORMS3\FORMS32k.mdb and the linked table.

Option 2: Mapping directly to an ODBC table

Mapping directly to an ODBC table is slightly faster to create than a linked table, but may be slower during the HotSync process than using a linked table.

To setup a direct link to ODBC:

- a. In the Pendragon Forms Manager, click the Import button, then click the Import ODBC button.
- b. Select the machine data source and log on to the external database if prompted.
- c. Select a database table within the external database to link to.
- d. Follow instructions on pages 492-505, Step 1c to Step 6. When doing Field Mappings in Step 4, click on the Select ODBC Database button, and select the machine data source, enter the password for that database if one is required, and select the database table to which you want to link.
- e. As you leave the Field Mappings screen, you will be prompted *Would you like to update the query to point to the new data source?* If you select Yes, you will be able to use the Pendragon Forms Edit/View button to view data in the external database table. If you select No, then the Edit/View button will not work, and you will need to open the external database to view your data.

Troubleshooting Tips

See Appendix B, *Troubleshooting*, page 541 for general troubleshooting tips when linking to an external database.

One error that can occur when linking to an ODBC database is related to the way that database columns are named. When database column and table names contain spaces or special symbols, Microsoft Access SQL uses square brackets [and] as quotes to mark the beginning and end of the name. However, other dialects of SQL, including ODBC SQL, do not recognize square brackets. Pendragon Forms uses its best judgement when adding square brackets, but it does not always generate SQL that is appropriate for querying ODBC data sources. For this reason, an option is provided to remove the square brackets when generating SQL.

To change the bracket mode, click the Options button in the Forms Manager window. The default option in the Bracket Mode field is Smart, meaning that Pendragon Forms will try to determine when to use square brackets or not. You can change the default to always use brackets or to never use brackets.

Note: If you change the Bracket Mode, you will need to click the Field Mapping button in the Properties window, and re-save the field mapping.

Security concerns when using ODBC databases

All handheld users will need to have access to the Forms32k.mdb database in order to synchronize. There is therefore a risk that the Forms32k.mdb file can be accidentally deleted by an individual.

- It is extremely important to back up the Forms32k.mdb database on a regular basis. See page 510.

If you are using a linked table in Access to link to an external ODBC database, your data will not be lost if the Forms32k.mdb file is lost, but you will still need to recover your form designs from a backup copy of the Forms Manager. To enable handheld users to synchronize, you will need to restore a backup of the Forms database.

Another security concern involves the use of passwords to access the back-end database. If you are using linked tables in Access, the password to access the external ODBC database will be stored in the Forms database. If you are mapping directly to ODBC, the password to access the external database will be visible in the Field mappings screen in the Forms Manager.

If it is not an acceptable risk for handheld users to have access to passwords to the back-end database, see *Security concerns in a Multi-User environment*, on page 521.

24. Backing Up the Forms Database

Pendragon Forms is designed to be centrally controlled from the PC. In order to synchronize a form on your handheld, each Form ID# on the handheld must exactly match the Form ID# of a form in the database on your PC.

If you lose the program on the handheld, but you still have the database on your PC, you can recover all of your form designs and all data up to your last synchronization.

Warning: If you lose the database on the PC, even if you still have the program on the handheld you will not be able to synchronize Pendragon Forms unless you can recover the form designs on the PC. It is therefore very important to backup the files on the PC, so that in the event of a hard drive failure you can recover your form designs and data.

Backing up the Database

On a regular basis, you should back up the entire C:\Program Files\Forms3 directory folder. In particular, you should back up the following files within the Forms3 folder:

- Forms32k.mdb
(Access 2000 / Access XP (2002) / Access 2003)

This file contains the Pendragon Forms source code, as well as your form designs and all data up to your last successful synchronization.

If you are linking to an external Access or ODBC database, you must back up your external database in order to protect your records.

Backing up Form Designs

As a precaution, whenever you distribute a form to the handheld, Pendragon Forms makes a backup of the form design in a .PFF file, which is placed in the C:\Program Files\Forms3\PilotF folder.

You should also backup these .PFF files to a ZIP disk or a CD-ROM, so that in the event of a hard drive failure, you will be able to re-install Pendragon Forms from the CD-ROM and then import your form designs into the database. To manually make a backup of a single form design, you can export the form to a Pendragon Forms Design (.PFF) file - see pages 240-241.

Recovering Forms Designs

If you accidentally delete a form design from the PC but the form is still on the handheld, and you need to synchronize the form, first check the Recycle Bin (see page 41) to try to recover the form.

If the form design is not in the Recycle Bin, go into Windows Explorer and check whether the form design .PFF file is still in the C:\Program Files\Forms3\PilotF folder. To figure out which .PFF file corresponds to your form design on a Palm OS device, tap the Forms icon on the handheld, then tap the Menu button and choose Help...Form Info. A dialog box will show an ASCII file name. The .PFF file corresponding to the form design will have the same file name as the ASCII file name, but the file extension will be .PFF instead of .OUT.



On a Pocket PC device, tap on the name of a form in the Forms Main Menu on the handheld, and then tap the Pendragon Forms title bar. Tap the help menu and then tap Form Info to see the .OUT file name, which will be the same as the .PFF file name.

If you find the correct .PFF file in the PilotF folder, go into Pendragon Forms and import the .PFF file back into the database. See page 240 for instructions on importing a .PFF file. You will need to re-create lookup lists and freeze the imported form design and re-distribute the form.

IMPORTANT:

On Palm OS devices: Before synchronizing, tap the name of the form on the handheld, then tap the Review button. Tap the handheld Menu button (the drop-down menu icon below or next to the House icon) and choose **Mark All Changed**. This will force all records on the handheld to be sent to the PC on the next synchronization.

On Pocket PC devices: Before synchronizing, tap the name of the form on the handheld, then tap the Review button. Tap the title bar on the review screen, then choose **Mark All Changed**. (If you do not see the title bar, minimize the onscreen keyboard.) Marking all records as changed will force all records on the handheld to be sent to the PC on the next synchronization.

Backing up Data within a Form

To make a backup of the data within an individual form, you can export the data to ASCII (see page 230) and save the ASCII file to ZIP disk or floppy disk.

VFS Backup on the Handheld

Previously, VFS Backup was only available on Palm OS devices, but in Forms 5.1 VFS Backup is also supported on Pocket PC devices.

Palm OS version 4.0 (and higher) supports VFS - Virtual File System - to allow the use of external storage media such as a compact flash card, an SD (Secure Digital) card, an MMC (Multi-Media Card) or a memory stick. Examples of handhelds that support VFS are the Palm Treo series, the Palm E2, Palm TX and Palm LifeDrive.

New in Pendragon Forms 5.1 is backup support for Pocket PC devices running Windows Mobile 5 for Pocket PC. Pocket PC devices do not use VFS, but have their own file system that supports backing up data to an external media card (typically a Secure Digital card - an SD card).

The VFS Backup options in Pendragon Forms can be used to perform a backup of new or changed records from the handheld to the external storage media. You may want to use VFS with Pendragon Forms if the data that you are collecting on the handheld is extremely important to you, and you do not want to risk losing the data if the handheld loses battery power or breaks. By performing a backup to external media, you will have a second copy of your data.

Note to Pocket PC users: Instead of using VFS Backup, you can also consider installing Pendragon Forms 5.1 directly to an external SD card (see page 8). If the handheld device fails, you can install Pendragon Forms to an external card on a replacement handheld device and then swap in the original card with your existing form designs and data.

Switching on VFS Backup for a Form Design

The VFS Backup options are an Advanced Form Property, described in detail on pages 210-211. There are three VFS Backup options in Pendragon Forms:

- **Use Backup If Available** - causes records for this form to be backed up if the media card is present. If the media card is not present, no backup occurs.
- **Require Backup Media** - the media card must be present in order to use this form design, and all new and changed records for this form are backed up.
- **Disable Backup Feature** - no backup occurs even if the media card is present. This is the default setting.

You can select a VFS Backup option before you distribute a form design. If you change the VFS Backup option after a form has been sent to the handheld, you will need to re-distribute the form and synchronize for the change to take effect.

How VFS Backup Works on the Handheld

If VFS Backup is enabled for a form design, then each new record and each modified record is automatically written to the external media storage device. The backup file on the storage media contains a history of activity for that form design. For example, if the handheld user creates a new record, and then two days later makes a change to that record, the backup file will contain one entry with the contents of the record when it was created, and a second entry with the contents of the record when the record was modified. If the data in the backup file is restored, the original record will be written to the handheld, and then will be replaced by the changed record.

Warnings about using VFS Backup

If you manually delete a record on the handheld, performing a restore from the external media will not bring the record back.

Pendragon Forms can only restore from the external storage media to a handheld device that has the appropriate form design. This means that if the handheld device is lost but the external media is safe, a replacement handheld device will have to be obtained, and the replacement handheld must be assigned the same user name as the original handheld. Pendragon Forms must be installed on the replacement handheld, and the correct form design has to be synchronized to the handheld before the records on the external media card can be restored to the replacement handheld.

If the Use Backup if Available option is set, there is a risk that the user might create a record with the external media present, but then make changes to the record when the external media is not present in the handheld device. In this example, the original record is backed up but the changes are not. If the user then restores from the card to the same handheld, the more recent changes will be lost, overwritten by the original copy of the record that was present on the external media card.

One way to ensure that all changes to records are backed up is to set the Require Backup Media option. In this case, the handheld user cannot use the form - even to review records - unless the external media is present in the handheld. The risk in this case is that if the handheld user loses the media card, then he/she cannot use the form until another media card is inserted. This may be a problem if the user is on a field expedition and no replacement cards are available.

Restoring records from the external media card to the handheld requires caution, and should only be attempted if all the records are lost from the handheld. There is a risk that the card may contain older records that have since been modified on the PC, or modified on the handheld without being backed up on the media card. Restoring from the external media card would cause changes on the PC or handheld to be lost. Always synchronize **before** performing a restore, and also backup the Forms database on the PC before performing a restore as well. To backup the Forms database, see page 510.

Restoring from an external media card restores data for all the forms that have been backed up on the external media.

Checking if VFS Backup is Enabled on the Handheld

Since VFS Backup is an Advanced Form Property, each individual form on the handheld may or may not have VFS Backup enabled.



To see if VFS Backup is enabled for a form design:

- Tap the Forms icon on the handheld, then tap the name of the form.
- On Palm OS devices, tap the handheld Menu button (the drop-down menu icon below or next to the House icon). On Pocket PC devices, tap the Pendragon Forms title bar.
- Tap the Help menu and tap Form Info. A Form Info message is displayed, showing whether VFS Backup is enabled or disabled.

Restoring Records from External Media

This is the procedure for restoring records from an external media card to the handheld:

1. Synchronize the handheld. If the handheld is brand new or was hard reset, then when prompted for the handheld user name, select the original user name of the handheld device.
2. Check if the Forms icon is present on the handheld. To check on a Palm OS device, tap the Applications button (House icon) on the handheld. To check on a Pocket PC device, tap Start...Programs and look for the Forms 5.1 icon.
If the Forms icon is not present you will need to install Pendragon Forms to the handheld. To install Pendragon Forms, on the PC click Start...Programs...Pendragon Forms 5.1... Install Forms 5.1 on Palm OS device (or Install Forms 5.1 on Pocket PC device). Then synchronize twice - once to install the Forms 5.1 program on the handheld, the second time to receive your form designs.
3. If the Forms icon is on the handheld, tap the Forms icon to see if your form designs are present on the handheld. If the form designs are not present, open the Forms Manager on the PC and re-distribute the form designs. Then synchronize the handheld.
4. At this point, it is strongly recommended that you backup the Forms database - see page 510. This provides you with a copy of the database before the restore is performed.

Continued on next page...



5. Insert the external media card in the handheld.

On Palm OS devices, tap the Forms icon, then tap the handheld Menu button (the drop-down menu icon below or next to the House icon). Tap the Help menu, then tap Backup Info.

On Pocket PC devices, tap the Forms icon, then tap the Pendragon Forms title bar. Tap the Help menu, then tap Backup Info.



6. The Backup Info dialog screen displays the size of the backup file for all forms that have VFS Backup enabled.

The dates From and To indicate the date of the earliest entry and the latest entry in the backup file. Since the backup file also records the date of each synchronization, the To date might be today's date if you synchronized today.

To restore from the external media, tap the Restore button.



7. A Restore Backup File screen warns you that performing a restore may delete or modify existing records. This might occur if records were changed without being backed up, or if records were changed on the PC. This is why it is recommended to backup the Forms database on the PC before performing a restore.

Tap Yes to continue with the restore procedure, or tap No if you still need to backup the Forms database on the PC.



8. The Restore Backup File gives you one last chance to cancel the restore procedure. Tap OK to continue, or tap Cancel to cancel the restore procedure. If you tap OK, you will be prompted to select the date on which you lost your data.



9. Select the date on which you lost your data. As soon as you tap on a date to select it, the restore procedure will begin. Pendragon Forms will search the backup file on the external media card for the most recent synchronization date before the date you selected, and will restore all new and changed records for all backup-enabled forms since that synchronization date.



10. A Restore Complete dialog will be displayed to confirm that the restore procedure has been completed. Review the records that have been restored to verify that you have all the records. If some records are missing, you may need to re-do the Restore procedure and select an earlier date of data loss in step 9. Finally, synchronize the handheld to send the restored records to the PC.

Purging Old Records from External Media

When VFS Backup is enabled, Pendragon Forms writes to the external media a copy of every change to every record for every form that has the VFS Backup option enabled. In addition, the backup file logs every synchronization date.

Although the external media card can store megabytes of data, you should consider purging old records from the external media regularly. If the backup file grows to more than the amount of free space on the card, a purge may no longer be possible.

Pendragon Forms allows you to delete records that are more than 30 days old from the external media card. In order to perform a purge, there must be as much free space on the external media as the size of the backup file itself. For example, if the backup file is 2MB, you need an additional 2MB of free space on the media card in order to perform the purge.

Warning: Always synchronize and ensure that all of the records on the handheld are successfully backed up to the PC before purging records from the external media.



1. To purge old records from the external media card, make sure that the media card is inserted in the handheld.

On Palm OS devices, tap the Forms icon, then tap the handheld Menu button. Tap the Help menu and then tap Backup Info.

On Pocket PC devices, tap the Forms icon, then tap the Pendragon Forms title bar. Tap the Help menu and then tap Backup Info.

On the Backup Info dialog screen, tap Purge.



2. The Purge Backup File dialog gives you the option to cancel. If you have not yet synchronized the handheld to backup records to the PC, you should cancel the purge procedure and perform a HotSync first.

Tap No if you want to cancel erasing records from the backup file.

Tap Yes if you want to erase records that are more than 30 days old from the backup file on the external media card.

25. Using Forms in a Multi-User Environment

This chapter covers issues involved when you have a multi-user license for Pendragon Forms.

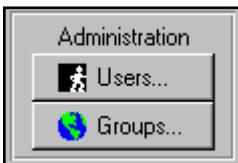
Entering a Multi-User Code

Pendragon Forms initially ships as a single-user license, that is, for use with one handheld.

To activate a multi-user license:

1. In the Pendragon Forms Manager, click on the Options button.
2. Enter your multi-user code.

Setting up Users and User Groups



A User List and User Groups are used to centrally manage the distribution of forms to multiple handhelds.

Refer to the *Users and User Groups*, starting on page 45.

Every handheld user name must be listed in the User List as an active user in order to synchronize with the Forms database.

- **PALM OS**
The first time a Palm OS device is synchronized, you will be prompted for the user name for that handheld. If you change the handheld user name in the Palm Desktop program or by doing a hard reset, you will need to update the User List in Pendragon Forms. To see what the user name is, tap the Applications (House icon) button, then tap the HotSync icon. The user name will be displayed in the upper right corner of the handheld screen.
- **PPC**
The first time a Pocket PC device is synchronized, you will be prompted for the user name for that handheld. This user name should be entered on the Pendragon Forms Server Options screen on the handheld (see page 9) as well as in the User List in the Pendragon Forms Manager. If you do not know the handheld user name of a Pocket PC device, tap Start...Settings on the handheld. In the Settings window, tap the System tab. Tap the About icon, and in the About screen, tap the Device ID tab. The Device name field contains the user name that Pendragon Forms will use.

- If the number of handheld users in the User List exceeds the multi-user license count, the handheld user with the oldest synchronization date will be made inactive. The inactive handheld will not be able to synchronize and will receive a synchronization error message (see page 551) to alert the user about the inactive status.

You can use User Groups to determine which handheld receives which form.

- Handheld user names must be in the User List before they can be added to User Groups.
- When you distribute a form, the form automatically goes into the Default User Group if the form is not assigned to another group. If you are designing forms, you may want to add yourself to the Default User Group so that you can receive all forms that are distributed.
- You can create a User Group for the other handheld users. Adding a form to this User Group will have the effect of distributing the form to all users in the group.
- A handheld user can be a member of more than one User Group, and a form can be assigned to more than one User Group.
- If you modify a frozen form and click the Distribute button to re-distribute the form, all handheld users in a User Group containing that form will receive the updated form during their next synchronization.
- To remove a form from all of the handhelds in a group, go into the User Group and delete the form from the list of forms for that group. Do not delete the actual form from the Pendragon Forms Manager until all users have had an opportunity to synchronize, otherwise the form will remain on the handhelds indefinitely. (See page 50.)
- Removing Lookup Lists from handhelds also requires care. (See page 118.)

Sharing Records Across Handhelds

By default, Pendragon Forms is set up to allow users to share form designs, but not to share records. See pages 31-34 for information on the form design issues when sharing records across multiple handhelds.

Choosing the right Multi-User Installation

Depending on how your organization is set up, you can choose to:

- Install to a stand-alone PC, so that all handhelds synchronize via the same cradle.
- Perform a Network Server installation of Pendragon Forms on a network server, and do Network Client installations on workstation PCs. This allows each handheld to synchronize to an individual workstation that is connected to the server via a local area network (LAN) or via the Internet.

Note: The server must have a fixed IP address and must allow communications on server port 201 or other designated port.

Typical/Stand-Alone Install

If you are synchronizing locally to a stand-alone PC, then after installing Pendragon Forms on the PC (see page 1), enter your multi-user code and set up your User List and User Groups.

Install the Forms application on the handheld. See page 6 for Palm OS handhelds and page 8 for Pocket PC handhelds. Once the application is installed on the handheld, you are all ready to deploy forms to multiple handhelds.

Network Install

If you want users to synchronize from multiple workstations (or wireless handhelds) to the Forms Manager database on a server, you need to do a Network Server installation of Pendragon Forms on the server (see page 1), and do a Network Client installation of Pendragon Forms at each workstation (see page 11).

Important: The server must have a fixed IP address, and communications must be allowed on server port 201 or another server port. If the server has a firewall, the firewall must be configured to allow communications on port 201 or other port.

Pendragon SyncServer is the technology used to synchronize Pendragon Forms data between handheld devices and the Forms Manager database. When you do a Network Server installation, the server component of SyncServer is installed on the server along with the Forms Manager database. When you do a Network Client installation, the client component of SyncServer is installed on the client workstation. The Forms Manager database is not installed on the client workstation as the client will synchronize to the Forms Manager database on the server.

Pendragon Forms 5.1 ships with a free version of Pendragon SyncServer that allows one user to synchronize at a time. If you have a license for more than 10 handheld devices, you may want to consider purchasing a copy of SyncServer that allows more users to synchronize simultaneously. Visit www.pendragonsoftware.com for more information on licensing Pendragon SyncServer.

Security concerns in a Multi-User environment

Security of Data Transmission

Pendragon SyncServer, the Pendragon SyncServer Client Proxy Conduit and Pendragon SyncServer Link (S-Link) applications on Palm OS devices, and Pendragon Forms 5 on Pocket PC devices, all use built-in 56-bit Data Encryption Standard (DES) encryption and MD5 authentication to protect the data as it passes across the TCP/IP network. The encryption permits secure synchronization, even over the Internet. For customers within the United States, a version of Pendragon SyncServer is available that provides 168-bit triple DES (3DES) encryption.

Running SyncServer as a Service or as a Process

If Pendragon Forms 5.1 is installed as a server installation, Pendragon SyncServer will run as two system services: Pendragon SyncServer Service and Pendragon WatchDog Service. The Pendragon SyncServer Service is the Pendragon SyncServer, and the Pendragon WatchDog Service is a utility for restarting the Pendragon SyncServer. (See page 567 for information on the Watchdog Service.)

The benefit of Pendragon SyncServer running as a system service is that users do not need to log onto the server before they synchronize. Another benefit is that only users with Administrator access can switch off the system service, and so it is less prone to accidental shutdown by non-Administrator level users.

Upgrades from Forms 5.0 to Forms 5.1 will also install SyncServer as a service if SyncServer was previously installed as a service.

If Pendragon Forms is installed as a typical / Standalone installation, Pendragon SyncServer will run as a process. Running as a process means that Pendragon SyncServer will appear as an icon in the Windows Task Tray. Since the process is visible, it is therefore susceptible to accidental shutdown by anyone using the PC, and is not as secure for use as a server.

Security Issues to Consider

In many cases, administrators may be comfortable with little or no security, especially with small, close-knit user groups. For example, a small office of professionals who share data on a peer-to-peer network already accept the risks associated with sharing their files, and trust their co-workers implicitly. In this scenario, you would still need to make frequent backups of the Forms Manager database (Forms32k.mdb) and any other databases to which Pendragon Forms synchronizes, as a precaution against data corruption or a hardware failure.

In an enterprise where the data is proprietary or extremely valuable, it makes sense to impose security measures.

There are several security issues to consider when creating your data collection application.

- 1) Access to sensitive data: You may want to limit access to the central database by users with handhelds, or users on the network.
- 2) Deletion/corruption of data: Though there may be a minimal risk of malicious damage, it may be possible for users to delete or corrupt data on the server. Make frequent backups of the Forms Manager database (Forms32k.mdb) and any other databases to which Pendragon Forms synchronizes.
- 3) Deletion of database files: Microsoft Access is a file-based database. This means all users who are given access to design forms in the Forms Manager must have read-write access to an .MDB file. This means that users may be able to delete the entire database file via file management tools. Again, the solution is to make frequent backups of the Forms Manager database (Forms32k.mdb) and any other databases to which Pendragon Forms data synchronizes.
- 4) Access to external databases via the Pendragon Forms database: In order to synchronize with password-protected external database systems, the Pendragon Forms database generally needs to include the database username and password in the Forms32k.mdb file. This is true whether using the Access linking feature, or using a direct ODBC field map. Unfortunately, this means that users who gain access to the Forms32k.mdb file will generally gain some access to the external databases too.
- 5) Impersonation ("spoofing"): It is relatively easy to change the name assigned to a handheld device to impersonate another user. By performing a hard reset of the device, and performing a synchronization, users can rename their handheld device quite easily. If your Pendragon Forms applications send data from the desktop to the handheld (e.g., sales reports), you may need to defend against this type of attack. Since the handheld username is the mechanism used to control which users see which datasets, spoofing may allow users to see data that was only meant for another user. You may want to consider upgrading to Pendragon Forms Enterprise Edition, which allows each user to have a separate password. Changing one's username in this case would not gain access to another person's data if the other password is not known.

- 6) Data on the handheld is not encrypted. This means that if the handheld hardware is compromised, so is the data it holds.

A number of third-party products allow the handheld device to automatically become password protected when powered off. These tools make it more difficult, though not impossible for hackers to access the memory of the device. Some versions of Palm OS 5.2 and higher have built-in encryption features. Consult your handheld documentation for information on switching on encryption features.

Limiting access to the Pendragon SyncServer Password

Pendragon SyncServer is password protected, so that only authorized handheld users can synchronize with the Pendragon Forms database on the server.

All Pocket PC users who synchronize (via cradle/cable or wirelessly) must store the Pendragon SyncServer password in the Server Settings screen of the Forms 5.1 application on the handheld device.

All Palm OS users who synchronize wirelessly must store the Pendragon SyncServer password in the Pendragon SyncServer Link (S-Link) application on the handheld device.

For Palm OS users who synchronize locally at their workstation, there are two ways to store the Pendragon SyncServer password:

- i) Store the password on the workstation. This is the default.
- ii) Install the Pendragon SyncServer Link (S-Link) application on the Palm device in order to store the password on the handheld.

If the workstation is secure, meaning that everyone who uses the workstation has authorization to access the Pendragon Forms database, then storing the Pendragon SyncServer password on the workstation is acceptable, and is easy to administer.

However, if the workstation is not secure, meaning that some people who have access to the workstation should be denied access to the Pendragon Forms database, then storing the Pendragon SyncServer password on the workstation poses a security risk. The risk is that if an unauthorized person renames a handheld device to the name of an authorized user, the unauthorized person will gain access to the Pendragon Forms database during synchronization. To prevent this security problem, it is recommended that on non-secure workstations, the Pendragon SyncServer password should be stored in the Pendragon SyncServer Link (S-Link) application on the handheld devices of the authorized users.

Appendix A: Sample Forms

Several sample forms are stored as files with .PFF file extensions in the C:\Program Files\Forms3\Sample Forms folder.

You can follow the instructions on page 240 to import any of the sample forms into the Forms Manager database. After a form has been imported, you will need to freeze the form and distribute the form before synchronizing the handheld.

Here are a list of the sample form designs:

Form Name	What This Sample Form Demonstrates
HandheldSurvey.pff	This form shows some of various field types that you can use when designing forms. The form has some branching scripts.
AddressBookLookupButton.pff	A form that has a Button field to do a lookup to the handheld Address Book, and copy the selected Address Book data into the form.
AverageofTestScores.pff	A form that illustrates how to add more than one field together, and take an average.
CustomerVisitParent.pff CustomerVisitSubform.pff	Import both of these form designs to see an example of a parent form and a subform.
BarCodeExample1.pff	A simple form that uses a scan: event script to allow the user to scan a bar code into a specific field at any time.
BarCodeClone1.pff	A form that allows the user to scan a bar code to generate a new record via a clone statement in a scan: event script.

Continued on next page...

Form Name	What This Sample Form Demonstrates
BarCodeLookup2Ref.pff BarCodeLookup2.pff	This pair of forms show how you can create a reference form (BarCodeLookup2Ref.pff) with inventory data, and then scan a bar code in another form (BarCodeLookup2.pff) to do a lookup to the reference form and copy data from the reference form into the other form.
Beaming1.pff Beaming2.pff	These two forms can be used independently of each other. They illustrate the scripts necessary for allowing beaming of records from one handheld to another.
PrintReportTest1.pff	A form that illustrates the type of scripts required for printing a record from the handheld.
PrintReport HTML Test.pff	The same form as PrintreportTest1, but with PRINTHTML statements for use with Bachmann PrintBoy 7 printer driver software.
PrintReportTest2Parent.pff PrintReportTest2Subform.pff	Import both of these form designs to see an example of printing from a parent form and a subform.
CustomerMainMenu.pff CustomerInfo1.pff	The Customer Main Menu form is an example of a custom main menu form that replaces the Pendragon Forms main menu. The Customer Info1 form is the form that the Customer Main Menu allows access to.
ReferenceForm1Customers.pff OrderForm1Parent.pff OrderForm1Subform.pff ReferenceForm1Items.pff	These four forms combined, illustrate a simple order entry application that uses a parent form for the customer information on the order, a subform for the order line items, and two reference forms - one for looking up customers and one for looking up items. After you import and freeze these four forms, enter some data records on the PC for the two reference forms before distributing all four forms to the handheld.
GPS Test2.pff	A form for testing what data is being received from a GPS receiver.

Appendix B: Troubleshooting

This Troubleshooting Appendix covers the following topics:

Problems with Sending Form Designs or Data to the Handheld	Page 527
Form Designer Error Messages	Page 530
Problems with Freezing a Form Design	Page 531
Problems with Distributing a Form	Page 534
Problems with Lookup Lists	Page 535
Problems with Subforms or Single Subforms	Page 536
Problems with Working with Multiple Forms	Page 539
Problems with Linking to an External Database	Page 541
Synchronization Error Messages	Page 545
Problems with Sending Data from the Handheld to the PC	Page 556

Problems with Sending Form Designs or Data to the Handheld

If you do not see your form on the handheld after you distribute a form and synchronize, here are some things that you can check:

Is there a Forms icon on the handheld?

Check that there is a Forms 5.1 icon on the handheld.

PALM OS

If you have a Palm OS device, refer to page 6 for instructions on how to look for and how to install the Forms 5.1 program on the handheld.

PPC

If you have a Pocket PC device, refer to pages 8-9 for instructions on how to look for and how to install and set up the Forms 5.1 program on the handheld.

Is the Pendragon Forms conduit listed as an application in the HotSync Manager?

PALM OS

On Palm OS devices, third party applications such as Pendragon Forms must be registered in order to synchronize during the HotSync data transfer. To check if Pendragon Forms is registered:

- a. Click the red & blue arrows icon of the HotSync Manager in the Windows task tray, and choose Custom.
- b. In the Custom window, select your handheld user name. Pendragon Forms 5.1 should be in the list of applications, with the action *Synchronize the Files*.

Network Client PC:

If Pendragon Forms is not listed in the Custom window and your PC is a Network Client PC that is supposed to synchronize Pendragon Forms to a server, then re-run the Pendragon Forms Network Client installation on your PC. See page 11.

Standalone PC or Network Server:

If Pendragon Forms is not listed in the Custom window, and your PC is a standalone PC or a Network Server, here is a quick way to re-register the Pendragon Forms conduit:

1. Close the HotSync Manager Custom window if it is still open.
2. Click the red & blue arrows Hotsync icon, and choose Exit.
3. Click Start...Programs...Pendragon Forms 5.1...Configuration Tool. con't...

4. In the Configuration Tool window, click the Diagnostics tab. Then click the *Setup Forms 5.x HotSync* button.
5. Close the Configuration Tool window. Re-start the HotSync Manager by clicking Start...Programs...PalmOne...HotSync Manager. (On older systems, it may be Start...Programs...Palm Desktop...HotSync Manager.) The red&blue arrows icon of the HotSync Manager should re-appear in the lower right corner of your screen. You can verify that the Pendragon Forms conduit is now registered by following steps a and b on the previous page.
6. Go into the Pendragon Forms Manager, select a form and click the Distribute button distribute to the handheld. Synchronize the handheld.

Is there a message in the Synchronization Log?

If Pendragon Forms has a problem synchronizing, an error message will be recorded in the synchronization log.

Standalone PC or Network Server:

To view the Pendragon SyncServer synchronization log, see page 545. See from page 545 for a list of common Synchronization Error Messages.

Standalone PC or Network Client:

PALM OS

On a Palm OS device that is synchronizing via a cable/cradle, the synchronization error message will appear in the HotSync Log.

To check the HotSync Log, click the HotSync Manager icon on the PC (the red & blue arrows icon in the Windows task tray), and choose View Log. You must view the HotSync Log on the PC and not on the handheld, as the handheld does not provide full details of the log messages.

If Pendragon Forms has a problem synchronizing, the log will contain the following lines:

Pendragon Forms synchronization started at...

Error message here

OK Pendragon Forms with 1 message(s)

See from page 545 for a list of common Synchronization Error Messages.

PPC

On a Pocket PC device, if there is an error during synchronization, the error will be displayed on the Pendragon Forms synchronization screen at the end of the synchronization process.

See from page 545 for a list of common Synchronization Error Messages.

There are no synchronization error messages, but still no form designs on the handheld.

Check the following:

- Have you distributed the form? Click the name of the form in the Forms Manager, then click the Distribute button. Then try synchronizing.
Note: You can only distribute a **frozen** form design (see page 19).
- Is the handheld user assigned to a user group in the Forms Manager? In the Forms Manager, click the Groups button and then click the Edit Members button next to the Default Group. If the handheld user name is not in the list of users, click the arrow button in the empty row in the list of users, and select the handheld user name.
- Is the Advanced Form Property of *Hide Form in Forms List* set? If yes, you will not see the form in the Forms main menu, but the form may be there. See page 209 to unhide the form design. You will need to re-distribute the form design and synchronize.

My form design goes to the handheld, but data that I entered on the PC does not.

Check that the correct Data Persistence option has been set to allow records to be sent to the handheld. In the Forms Manager on the PC, click the name of the form, then click the Properties button. Look in the Data Persistence section of the Properties window. If no Data Persistence options have been selected, then records will not stay on your handheld. See page 198 for the Data Persistence options that you can choose.

Another cause of data not being sent to the handheld is due to a mismatch between the handheld user name and the user name to which the records have been assigned. By default, only records that have been assigned to a given user will be sent to the handheld of the same name. If you freeze a form design and then click Edit/View, you can enter records on the PC. In the Edit/View window, you will need to select your handheld user name in the Username column, in order to assign the records to your handheld device.

If you have a multi-user license for Pendragon Forms and you need to share data across several handhelds, you will need to create your own primary key on a form (before the form is frozen), and then you can choose to delete the default Additional Download Criteria that sends records to the handheld based on the user name. See pages 31-34 for information on sharing records; page 188 for information on primary keys, and page 214 for information on Additional Download Criteria.

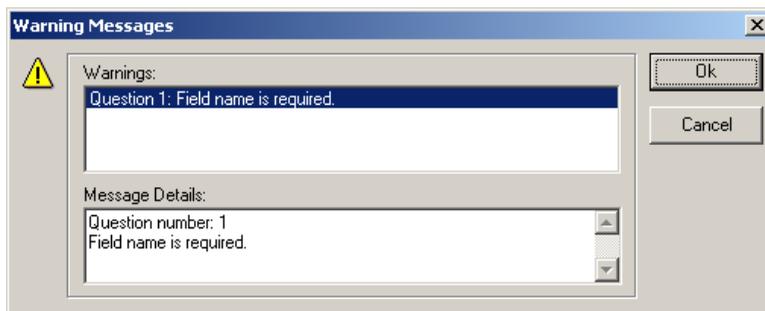
Form Designer Error Messages

If you get an error message when exiting the Form Designer window, the error is usually a warning that there is a problem with your form design. If the problem is not corrected, you may not be able to freeze the form design later.

If a Warning Messages dialog window comes up, it will specify which question (i.e. field number) on your form has the problem, so that you can correct it.

If you click the Cancel button you will cancel closing the Form Designer window. You will return to the Form Designer window and correct the error before saving and exiting the Form Designer window.

If you click OK you will continue with closing the Form Designer window. This means that you will have to correct the error later, before you freeze the form design.



Field Name is Required

This Form Designer error occurs if you added a new field to the form but did not type a field name. The field name area will contain the phrase *Enter question text...*. You will need to highlight that text and type a name for your field.

The Warning Messages dialog box will specify which question (i.e. field number) on your form has the problem, so that you can correct it.

List Item Entries are Required

The means that a Popup List field does not have any items in the Popup List.

A Section Name Entry is Required

This Form Designer error occurs if you have a Section field on your form, but you have not entered a name in the Section Name area. A Section field has to have a field name (question name), and a Section Name.

A Lookup List Entry is Required

The means that a Look List field is not referencing the name of a Lookup List or the name of a form.

Problems with Freezing a Form Design

If there is a problem with your form design, when you attempt to freeze the form an error message may be displayed and the form design may not freeze. The error message that is displayed will indicate the problem that is preventing the form from being frozen, and will also indicate which field on your form has the problem.

The solution is to close the Form Properties window, edit the form design to correct the error, and then try to freeze the form design again.

Common error messages that can occur when freezing a form are described below.



Duplicate field name found - field names must be unique.

When you freeze a form design, a database table is created for storing records associated with the form. Each column in the database table is based on a field in your form, and Microsoft Access requires each database column name to be unique.

If two fields have the same name, then when you try to freeze the form, the Duplicate field name found error will be displayed. You have the option to click Cancel to stop freezing the form, and then go back to the Form Designer to change the field name of one of the specified fields. Or, if you click OK and freeze the form design, Pendragon Forms will internally add the number 1 to the database column for the duplicate field, so that although the field names will be the same, the database column names will not be the same.

This error can also occur if you make a copy of a form. When you copy a form, the column names from the original form design are retained on the copy, even if you rename a field. If you rename one field and then add another field with the same name as the original name of the first field, then when you try to freeze the form design the column names of the two fields will be identical and will cause an error. The solution in this case is to go to the Data tab in the Form Designer window (see page 186) for one of the duplicate fields, and erase the Column Name, then try to freeze the form again.

Popup List (or forms / sections list) is Empty

If this error occurs when you are trying to freeze a form, it means that a Popup List on your form does not contain any items in the list. The error message will tell you which field has the problem, so that you can go to the Form Designer window and edit that field.

Every Section Requires a Name

This means that your form has a Section field that has a blank Section Name. Edit the form and type in a name for the Section Name. (Note that the Section Name is required in addition to a field name.)

Lookup List Not Specified

This means that the field specified in the error message is a Lookup List field that does not reference the name of a Lookup List to display in the field. (If you are doing a lookup to another form, the name of a form is missing.) Edit the form and either select a Lookup List or type the name of the form to which you want to do a lookup.

The Field Name You have Chosen is Reserved by Pendragon Forms

Pendragon Forms always adds four fields to your form design when you freeze a form. The fields are called: RecordID, UnitID, UserName and TimeStamp. Since Microsoft Access requires database column names to be unique, you cannot name any of your fields these reserved field names. If you get this error message, re-name your field so that it does not conflict with any of the the reserved field names.

No Caption (default value) has been specified for your Button field

A Button field needs a name on the button. This is entered by typing in the Default Value field on the Data tab of the Frm Designer window. If you forget to enter a name for the button, you will get the No Caption error message when freezing the form. Your form will still freeze and you can add the Default Value to the Button field before distributing the form.

The Default Value for a Checkbox must be either Y or N

A Yes/No Checkbox can only store the values Y for Yes or N for No. If you type anything other than Y or N in the Default Value field, you will get this error. Edit the form and correct the Default Value (see page 191).

The Default Value for an Option 1-5 field must be in the range 1-5

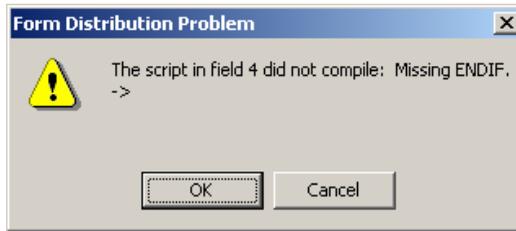
An Option 1 to 5 field can only store the values 1, 2, 3, 4 or 5. If you type anything else as the Default Value of an Option 1-5 field, you will generate this error. Edit the form and correct the Default Value.

Problems with Distributing a Form

In Pendragon Forms 5.1, scripts are compiled when a form design is distributed to the handheld. If there is a problem with the syntax of a script, an error message will be displayed when you try to distribute the form.

The error message will indicate the field number of the field containing the problem script.

In the sample screen shown below, the scripting error is in field 4, and the error is that the script is missing an ENDIF statement at the end of an IF...THEN statement.



If you click Cancel, the form will not be distributed, and you can edit the form, correct the script and then re-distribute the form.

Problems with Lookup Lists

Lookup Failed Error



On the handheld, if you get an error message *Lookup Failed. List not found.* then either:

1. The Lookup List is not on the handheld.
- OR
2. The form is not referencing the correct name of the Lookup List.

The error message tells you the name of the Lookup List that the program cannot find. To check if this Lookup List has been sent to the handheld, click the Lookups button in the Forms Manager on the PC. If the *Available* checkbox is not checked, edit the Lookup List and check the *Make this lookup available on handheld* checkbox. Synchronize the handheld and see if the Lookup List appears.

If the problem persists, check if the form design is referencing the correct Lookup List. Click on the form in the Forms Manager, then click the Edit button to open the Form Designer. Display the Lookup List field and re-select the Lookup List that you want to use. Re-distribute the form design, synchronize, and check the handheld again.

Problems with Subforms or Single Subforms

Form Not Found Error



On the handheld, if you tap the subform field on the parent form and receive the error message: *Form not Found*, then there are two possibilities:

1. The subform is not present on the handheld.
- OR
2. The parent form is referencing the wrong subform name.

If the subform is not on the handheld, distribute the subform and synchronize.

Check the exact spelling of the name of the subform in the list of forms in the Pendragon Forms Manager on the PC.

Then click the name of the parent form and click the Edit button to open the Form Designer.

Display the Subform field on the parent form, and check the Subform List section of the screen to see if the correct subform name is listed. If the subform name is not that same as the name of the actual subform, correct the name here.

A common error that might occur if you have copied the subform to make changes, is that the parent form may reference the original subform name, such as:

Patient Visit

but the actual subform being used is a copy with the name:

Patient Visit*

In this scenario, either change the subform name to match the name referenced by the parent form, or else change the subform field on the parent form to reference the new subform name that includes the asterisk * symbol.

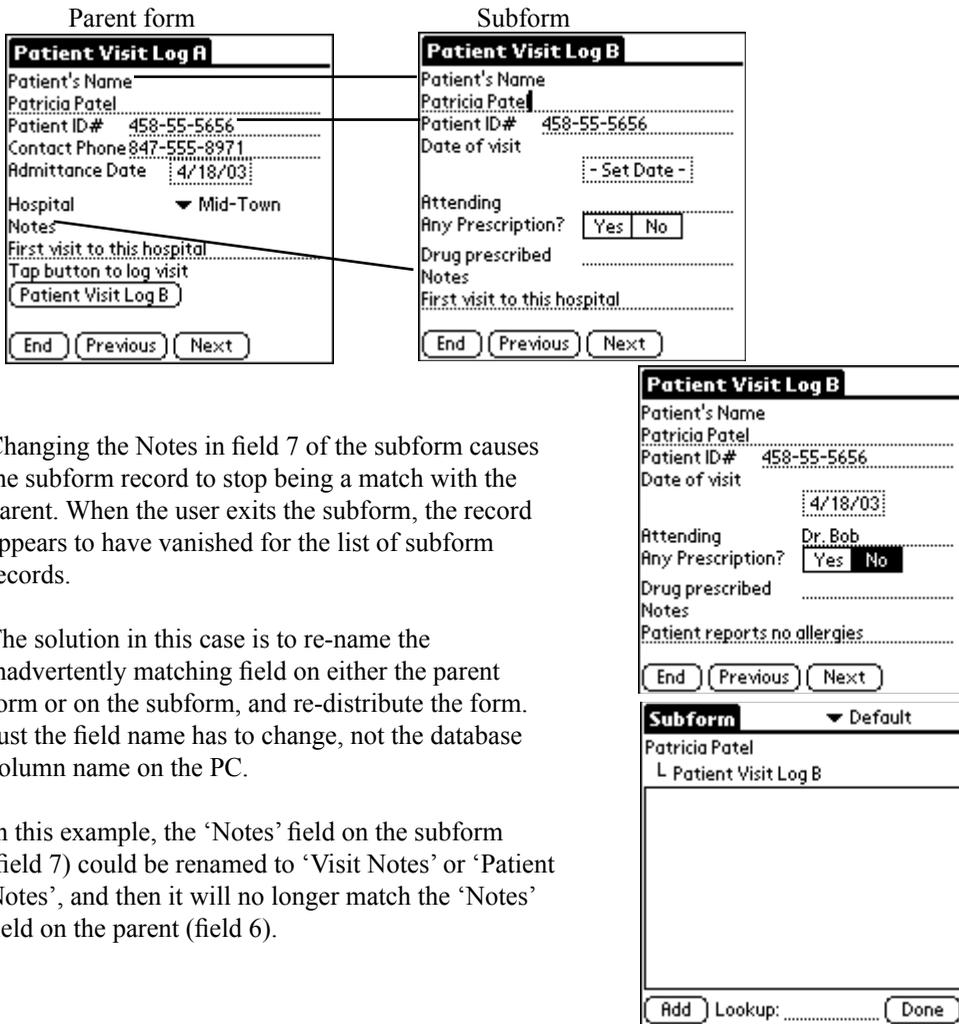
Close the Form Designer window, re-distribute both the parent form and the subform, and then synchronize the handheld. To re-distribute a form, click on the name of the form and click the Distribute button.

Note: If you are using scripts, the *Form not Found* error may be due to a script referencing the wrong subform name. Check your scripts.

Subform records appear to vanish when reviewed in the parent form

If you add a subform record and then tap the End button, and you do not see the newly created subform record in the list of existing subform records, the most common cause is that you have changed a field on the subform that is used to match the parent to the subform.

In the example below, a parent form and subform are supposed to match on fields 1 and 2 only (Patient's Name and Patient ID#), but inadvertently also matches field 6 (Notes field) of the parent with field 7 of the subform.



Changing the Notes in field 7 of the subform causes the subform record to stop being a match with the parent. When the user exits the subform, the record appears to have vanished for the list of subform records.

The solution in this case is to re-name the inadvertently matching field on either the parent form or on the subform, and re-distribute the form. Just the field name has to change, not the database column name on the PC.

In this example, the 'Notes' field on the subform (field 7) could be renamed to 'Visit Notes' or 'Patient Notes', and then it will no longer match the 'Notes' field on the parent (field 6).

You can check which fields match and change the names of any fields that are not supposed to match. This can be done before or after freezing the form designs of the parent form and the subform.

The Pendragon Forms Manager can generate a report to show you which fields within the first 10 fields of the parent form match fields on the subform. If no fields match, or if fields match that you did not want to match, you can change the field names before distributing the form.

To generate the Matching Fields Report:

1. In the Form Designer, display the Subform List field of the parent form.
(The Subform List field must reference the name of a subform.)
2. Click the Matching Fields button on the Field tab of the Form Designer window.

A Matching Fields Report will be displayed. To print the report, click the File menu and choose Print.

Data in a parent form is not copying to the subform

Only data in the first 10 fields from the parent form will copy into the subform.

If data from the first 10 fields on the parent is not copying to the subform, then either the fields are not named identically in both parent form and subform, or else the field types are not identical. Check the spelling of the field names, including upper and lower case letters and spaces. You can change the field names in the Form Designer on the PC to correct them, and re-distribute the parent form or the subform.

If the field types do not match, you will need to copy the subform and change the field types to match those of the parent form. Make sure that the parent form references the new subform.

Problems with Working with Multiple Forms

Form Not Found

See page 536 in this Troubleshooting Appendix.

Data from my parent form is not copying into my subform

See page 538 in this Troubleshooting Appendix.

Subform records seem to vanish

See page 537 in this Troubleshooting Appendix.

In a Lookup to Another Form, I get the error message Lookup List Failed. List Not Found

Check the following:

- Is the Lookup List referencing the correct reference form name? In the Forms Manager, click the name of the form and then click the Edit button to open the Form Designer window. View the Lookup List field and make sure that the name of the form being referenced is spelled correctly, with the same upper and lower case letters and spaces as the reference form name. If you make any corrections, re-distribute the form and synchronize to send the changes to the handheld.
- Has the reference form been sent to the handheld? In the Forms Manager, click the name of the reference form and then click the Distribute button. Synchronize to send the reference form to the handheld.

In a Lookup to Another Form, not all of the fields copy over from reference form to current form

Make sure that the field names and field types are the same in both the reference form and the current form.

For example, if the current form has a field called Part #, but the reference form has a field called Part Number, then data from the reference form will not copy into the current form.

Similarly, if the reference form has a Numeric field called Quantity, and the current form has a Text field called Quantity, data still will not copy because the field types are different.

The only exception is that a Text field in the reference form can copy into a Lookup List field on the current form (up to 50 characters).

I am using scripting with a **select** statement and no data is copying from my reference form to the current form

Check the following:

- Does the script with the **select** statement reference the correct form name? For example, if you made a copy of the reference form, the copied form may have a different name and the **select** statement script may need to be changed to reference the new form name.
- Does a matching record exist in the reference form? Perhaps the select statement is working, but it is not finding a matching record in the reference form to copy into the current form. For testing purposes, you can add to your script a message box to display the number of matching records that the select statement finds in the reference form. Since the **result** variable stores the number of matching records after a select statement has been run, you can add the following line to your script **after** the select statement:

```
msgbox result
```

If the select statement cannot find the reference form, the Result variable will be -1.

If the select statement can find the reference form, but cannot find a matching record, the Result variable will be 0 (zero).

If there is exactly one matching record in the reference form, the Result variable will contain the number 1.

If there is more than one matching record in the reference form, the Result variable will contain the number of matching records.

The message box that appears on the handheld will display the number corresponding to the value in the Result variable.

Problems with Linking to an External Database

When linking to an external Access database or an external ODBC database, it is very important to test that data synchronizes from the external database to the handheld, and the reverse: from the handheld to the external database.

Pay particular attention to synchronization error messages, as they indicate problems with sending form designs or data to or from the handheld. See from page 545 for a list of common synchronization error messages.

In addition to synchronization error messages, here are some common problems when linking to an external database:

No data from the external database appears to be going to the handheld

If the form design is on the handheld, but data from your external Access or ODBC database table does not populate down to the handheld during synchronization, check the Data Persistence settings. In the Forms Manager, click the name of the form, then click the Properties button. In the Properties window, the Data Persistence option *Keep a Copy of Records on handheld* (see page 198) should be checked in order to allow records from the external database to be sent to the handheld. If this checkbox is not checked, check the box and then click the Field Mappings button. In the Field Mappings window, click OK to save your changes. Then try synchronizing again.

New records from the handheld go to the database, but changes to existing records do not

Pendragon Forms uses the primary key to uniquely identify records, and to match a changed record on the handheld with an existing record in the database. If your external database table does not have a primary key, Pendragon Forms will be able to add new records (perform inserts) to the database, but will not be able to update existing records.

If you are linking to a query instead of linking to a database table in the external database, you will also find that you can add records but not update existing records. This is because in Microsoft Access, a query cannot have a primary key.

Unable to Open Database; File Already in Use

When opening a Microsoft Access database, there is an Exclusive option which allows the database to be used only by the person opening the database. If someone on the PC is using the external database exclusively, and a handheld user attempts to synchronize, the database will be locked. Error messages will be generated during synchronization will alert you if this occurs. The solution is to close the external database and re-try the synchronization. The error message that may be generated is shown below.

```
Unable to open database: Couldn't use 'D:\DebTest\Deb
Databases\Simple Work Order1.mdb'; file already in use
Skipping record deletion for form.
Skipping new record download for form.
OK Pendragon Forms with 3 message(s)
OK Backup
```

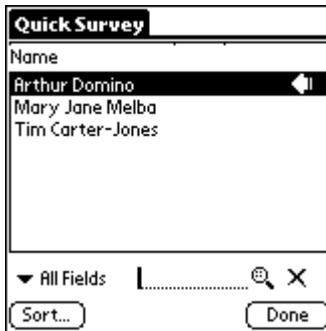
Unable to Append Record - the changes you requested to the table were not successful because they would create duplicate values in the index, primary key or relationship

Since the primary key is used to uniquely define each record, the field (or combination of fields) which make up the primary key has to be unique on the handheld and in the external database.

A synchronization error message that warns about duplicate primary keys can occur if a record that is marked as new on the handheld has a primary key that matches an existing record on the PC.

To eliminate the risk of accidentally overwriting an existing record, the new record will not be allowed to be uploaded to the PC. To alert the handheld user to the fact that a duplicate primary key has been entered, a synchronization error message will be generated, and records that did not get sent to the handheld will be flagged with an arrow. (See pictures below and on the next page.)

```
Unable to append record 'Customer Survey2' (111777267), field 8 -- The changes you
requested to the table were not successful because they would create duplicate
values in the index, primary key, or relationship. Change the data in the field or
fields that contain duplicate data, remove the index, or redefine the index to
permit duplicate entries and try again.
OK Pendragon Forms with 1 message(s)
OK Backup
```



Records marked with an arrow have not been uploaded to the PC.

If the handheld user has entered a duplicate primary key by accident, the record has to be edited on the handheld to make the primary key unique, before the record can be uploaded during the next synchronization.

Important Note:

There is another scenario in which the duplicate primary key error might occur. If synchronization is interrupted, it may be possible for new records to be sent to the PC, but not be flagged as changed on the handheld. This can also cause the duplicate primary key error to occur. The solution in this case is to check whether the records that are marked on the handheld as not uploaded are in fact already successfully in the database on the PC. If yes, there are two possible solutions:

- a) Delete the records on the handheld and synchronize again. (You must be certain that the records are on the PC before you delete them from the handheld.)

OR

- b) You can switch on the Advanced Form Property of Disable Record Overwrite Protection. This allows a new record to overwrite an existing record in the database - see page 219 for more information.

What happens if a field in a column of the external database is too large to fit on the handheld?

A Memo field in Microsoft Access can store up to 2 Gigabytes (2GB) of data. In a Pendragon form on the handheld, the maximum that a Text field can be is 2000 characters or 2 Kilobytes (2KB). If a Memo field for a given database record has more than 2KB in the external database, only the first 2KB will be sent to the handheld. To prevent the 2KB on the handheld from overwriting the 2GB already in the database during the next HotSync data transfer, Pendragon Forms makes this field of the specific record on the handheld read-only and non-updating on the server. Other records which do not exceed 2KB in this field will not be set as read-only or non-updating.

Similarly, if a Currency field in the database is greater than, \$10,000,000.00, (or less than -\$10M) the field is too large to store on the handheld. This field will display as null on the handheld and will be read-only and non-updating on the server for the specific records which are affected.

Number fields in Access which are integer fields will also be set as integers on the handheld, so that the handheld user cannot enter decimal places which will be thrown away by the server.

Synchronization Error Messages

Pendragon SyncServer is the technology that manages the synchronization of Pendragon Forms data. If Pendragon SyncServer experiences a problem during synchronization, an error message will be generated in the synchronization log. Always resolve any synchronization error messages, because the synchronization problem may cause your data not to be backed up to the database.

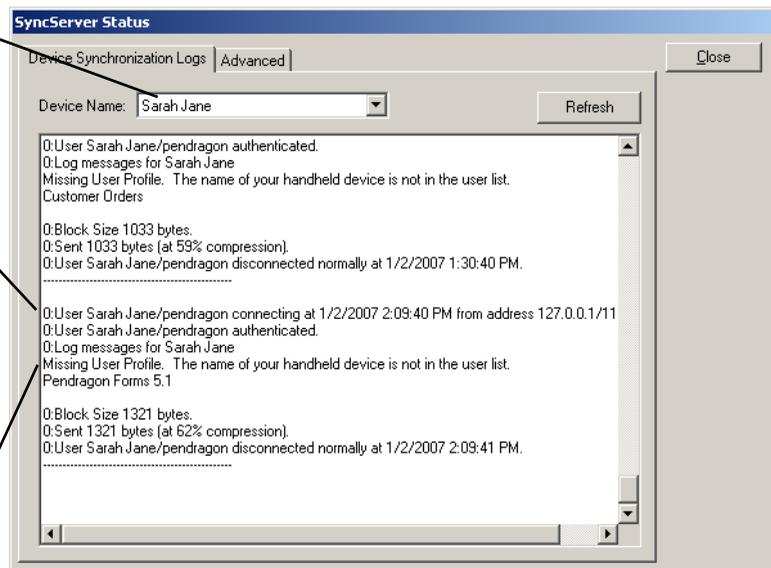
Viewing the Synchronization Log (Standalone PC or Network Server)

On a Standalone PC or a Network Server, you can view the synchronization log by clicking Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools. Select a User Name (Device Name) to view the synchronization log for that user. You can click the Refresh button to update the log.

Pick a User Name.

The most recent log is at the bottom of the screen.

Here is the synchronization error message, which appears after the line that says *Log messages for...*



Viewing the HotSync Log (Standalone PC or Network Client)

PALM OS

If you have a Palm OS device that is synchronizing via a cradle/cable to a PC (whether it is a standalone PC or a network client PC), you can also view synchronization error messages in the HotSync Log on the PC. This is an alternative to viewing the SyncServer synchronization log.

Important: Always view the HotSync Log on the PC, not on the handheld. This is because the log on the handheld does not display the error message in its entirety.

If a synchronization error message has occurred, the HotSync Manager will display a HotSync Problem dialog box on the PC screen for 60 seconds. Click the View Log button to view the log.



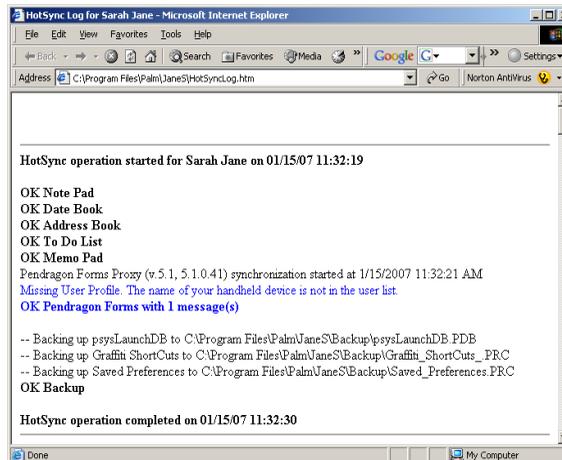
If you miss the HotSync Problem dialog box, then to view the HotSync Log on the PC, click the HotSync Manager icon (the red and blue arrows icon) in the Windows Task Tray, then click View Log. The most recent HotSync Log is at the top.

If Pendragon Forms has a problem synchronizing, the log will contain the following lines:

Pendragon Forms Proxy synchronization started at...

Error message here

OK Pendragon Forms with 1 message(s)



Viewing Synchronization Error Messages on a Pocket PC device

PPC

If you are synchronizing a Pocket PC device and an error occurs during synchronization, the error will be displayed on the synchronization screen of the Pocket PC device.

If you return to the Pendragon Forms application, then in order to see the most recent log message again you will need to tap the Sync button, then tap the Log button. This will display the most recent log message. You can also view the synchronization log on the PC or server if you need to see the error message again - see page 545.



Viewing Synchronization Error Messages on a Wireless Palm OS device

PALM OS

If you are synchronizing a wireless Palm OS device using the S-Link application, and an error occurs during synchronization, the error will be displayed on the S-Link screen.

If you tap the Log button you can see the synchronization log message from the most recent synchronization.

You can also view the synchronization log on the PC or server if you need to see the error message again - see page 545.



Authentication Failed. Incorrect username or password.

This synchronization error message occurs if one or both of these things happens:

- a) The User Name of the handheld device does not match the User Name in the Pendragon Forms Manager User List.
- b) The Password on the client device or network client PC does not match the SyncServer server password.

To check your handheld user name:

1. **PALM OS** Tap the Applications (House icon) button on the handheld, then tap the HotSync icon. The handheld user name will be displayed in the upper right corner of the handheld screen.
PPC On the Pendragon Forms Main Menu on the handheld, tap the Sync button, then tap the Options button. The user name is displayed in the Username field.
2. In the Pendragon Forms Manager, click the Users button to display the User List. Make sure that your handheld user name is in the User List, with the identical spelling and use of upper and lower case letters and spaces as on the handheld. Also make sure that the Active checkbox is checked. Close the User List window.
3. Click the Groups button in the Forms Manager. Click the Edit Members button next to the Default Group. In the User Group Editor window, make sure that your handheld user name is listed in the left-hand column. If it is not listed, click in the Username column and select your user name.

To check your password:

The SyncServer Server Password was created when you installed Pendragon Forms 5.1 on a PC or server - see page 3. If you have forgotten the server password, you can change it on the server, but then you will also have to change on all client PC's, all Pocket PC devices, and all Palm OS devices using S-Link. See page 570 for information on how to change the server password.

PPC

On the Pendragon Forms Main Menu on the handheld, tap the Sync button, then tap the Options button. On the Server Settings screen, re-enter the server password.

PALM OS

For Palm OS devices synchronizing via a cable/cradle, see page 570.
For Palm OS devices synchronizing wirelessly using S-Link, see page 7.

Missing User Profile

This synchronization error typically occurs if:

- a) Your handheld user name is not in User List in the Pendragon Forms Manager, or
- b) There is a mismatch between the user name on the handheld and the user name in the User List of the Pendragon Forms Manager, or
- c) You actually have more than one copy of the Forms Manager database, and while your handheld user name is present in the copy of the Forms Manager that you are checking, the user name is not present in the copy of the Forms Manager database to which you are synchronizing. The second copy of the Forms Manager may be in another folder on your PC, or may be on a shared network drive.

To check your handheld user name:

1.  Tap the Applications (House icon) button on the handheld, then tap the HotSync icon. The handheld user name will be displayed in the upper right corner of the handheld screen.
 On the Pendragon Forms Main Menu on the handheld, tap the Sync button, then tap the Options button. The user name is displayed in the Username field.
2. In the Pendragon Forms Manager, click the Users button to display the User List. Make sure that your handheld user name is in the User List, with the identical spelling and use of upper and lower case letters and spaces as on the handheld. Also make sure that the Active checkbox is checked. Close the User List window.
3. Click the Groups button in the Forms Manager. Click the Edit Members button next to the Default Group. In the User Group Editor window, make sure that your handheld user name is listed in the left-hand column. If it is not listed, click in the Username column and select your user name.

If your user name appears correct in the Forms Manager, you will need to verify that you are actually checking the same copy of the Forms Manager database with which the handheld synchronizes.

To find the location of the Forms Manager database to which you are synchronizing:

1. Click Start...Programs...Pendragon Forms...Configuration Tool.
2. On the Configuration tab of the Configuration Tool window, make a note of the directory folder of the Configuration database.
(This is typically C:\Progra~1\Forms3, meaning C:\Program Files\Forms3). con't...

3. Close the Configuration Tool window and use Windows Explorer to go to the directory folder named in the Configuration database field. Double-click the file Forms32k.mdb. This opens the copy of the Forms Manager to which you are synchronizing. Once again, check that the user name in the User List exactly matches the handheld user name, and check that the user belongs to the Default User Group.

License Count Exceeded

This synchronization error message occur if more handheld devices are synchronizing Forms than there are licenses for.

If this error occurs, Forms will de-activate the user who least recently synchronized. Users will continue to be de-activated until the licensed number of handhelds are synchronizing.

The License Count Exceeded error will occur under these circumstances:

- a) You you have added more users to the User List in the Forms Manager than your current Forms license allows.
The solution in this case is to purchase additional Forms licenses - one per handheld user. You can order additional Forms licenses at www.pendragonsoftware.com.

OR

- b) You have a valid multi-user license, but either the license code has not been entered correctly, or at all.
The solution in this case is to get your multi-user license code(s), open the Forms Manager and click the Options button. If any license codes are not entered correctly, correct the error. If there are no license codes, click the Add button and enter a license code.

OR

- c) You actually have more than one copy of the Forms Manager database, and while your multi-user code is present in the copy of the Forms Manager that you are checking, the multi-user code is not present in the copy of the Forms Manager database to which you are synchronizing. The second copy of the Forms Manager may be in another folder on your PC, or may be on a shared network drive.

To find the location of the Forms Manager database to which you are synchronizing:

1. Click Start...Programs...Pendragon Forms...Configuration Tool.
2. On the Configuration tab of the Configuration Tool window, make a note of of the directory folder of the Configuration database.
(This is typically C:\Progra~1\Forms3, meaning C:\Program Files\Forms3). con't...

3. Close the Configuration Tool window and use Windows Explorer to go to the directory folder specified in the Configuration Tool window. Double-click the file Forms32k.mdb. This opens the copy of the Forms Manager to which you are synchronizing. Once again, check that the multi-user code is entered correctly.

Inactive User

This synchronization error message occurs if a user who has been de-activated in the User List in the Forms Manager, tries to synchronize.

Important: If users have been deactivated due to a License Count Exceeded error message (see page 550), you will need to correct that problem first to avoid continued deactivation of users.

To re-activate a user, open the Forms Manager and click the Users button. Check the Active checkbox on the row with the user's name. Close the Users window and synchronize the user's handheld.

Evaluation Period Expired

This synchronization error message occurs if you have been using the evaluation version of Pendragon Forms and you have passed the expiration date of the evaluation version. Pendragon Forms will no longer synchronize the handheld in this case.

The solution is to purchase a copy of Pendragon Forms and enter your serial number /unlock code.

If you have already purchased a copy of Pendragon Forms and you still get this error message, the problem may be that either the unlock code was not entered correctly or at all. Click Start...Programs...Pendragon Forms...Configuration Tool. Enter the product unlock code or serial number in the Evaluation Unlock Code field. Click Exit to close the Configuration Tool window.

Note: You must have access rights to the Windows Registry in order to save the Forms unlock code or serial number. If you do not have access rights to the Windows registry, contact your LAN administrator or IT support person in your organization for assistance.

Connection Error Messages

If Pendragon SyncServer cannot connect to the network, you will receive a connection error message. For Palm OS devices synchronizing via a cradle/cable, see pages 572 and 573 for a list of network connection error messages.

On Pocket PC devices, a message such as *Unable to synchronize with server at IP address:201. Error reading server information* means that the handheld cannot connect. Verify that the server IP address and port number are correct, and that if there is a firewall, it is allowing communications on port 201 or other designated port. See pages 9 and 570 to check the Pocket PC and the server.

Other Synchronization Error Messages

Synchronization error messages that are generated by Pendragon SyncServer take the following format:

Unable to *<perform action>*
<message>

The *action* tells you what SyncServer was trying to do when it encountered the problem. The *message* tells you what the problem is. Read the message to figure out what the problem is.

There are 4 possible actions that Pendragon SyncServer might be unable to do:

Unable to sync via ADO or DAO

This means that SyncServer has not even started synchronizing any form designs and there is already a problem with the database.

Unable to execute downlink SQL

This means that SyncServer is trying to send records from the database on the PC to the handheld, and is encountering a problem.

Unable to append record

This means that SyncServer is trying to write a new record from the handheld to the database on the PC.

Unable to update record

This means that SyncServer is trying to update an existing record in the database with changes from the handheld.

The *message* part of the synchronization error message contains some or all of:

- The name and ID number of the form design that was synchronizing when the error occurred.
- The SQL statement that the conduit was attempting to execute. This is the command that the conduit was sending to the database when the error occurred.
- The field number of the field on the form that was being updated when the error occurred. If the field number is one more than the number of fields on your form (e.g.: your form has 10 fields, but the error message is referencing field 11), then the error is usually a constraint error such as a referential integrity problem (e.g.: child record without a parent record) or a conflicting primary key.
- An Access or ODBC database error. This is typically the cause of the synchronization problem.

There are hundreds of Access and ODBC errors that could possibly occur. Different database systems may generate different errors. Here is a list of the most common Access database errors you may encounter:

Too few parameters. Expected <number>

One or more column names (possibly specified in your Additional Download Criteria) do not exist in the table. Make certain that you are using column names in your Additional Download Criteria and not field names. If you are field mapping and you changed the name of any columns in your external Access database table, you will need to field map your form again.

Database or object is read-only

Most likely you are attempting to update or append a record in a table or query for which you have no permission for updates or inserts. Determine the database user account that you are using to connect to your database server. Check with your database administrator to confirm that the account you are connecting with has SELECT, INSERT and UPDATE privileges for the database table you are mapping to.

Out of present range

A value entered on the handheld will not fit into the corresponding column in the database table. This could be because the numeric value entered is out of range, or has too great a precision. It may also be that a text value entered on the handheld contains too many characters to fit into the database column.

Database <name> is exclusively locked

The database you are connecting to is in use by another user or program. Exit all of your applications and try synchronizing again. If the database is shared by other users, you may need to wait until the other user has relinquished control of the database.

Database <name> isn't a valid database name

The database to which you are mapping cannot be found. It could be on a drive letter that is no longer mapped to a network resource. Try remapping your form design, or recreating the shared network drive that was used when you originally mapped your form design.

The Microsoft Jet database engine could not find the object <name>.

Make sure the object exists and that you spell its name and the path name correctly.

or

The Microsoft Jet database engine could not find the input table or query <name>.

Make sure it exists and that its name is spelled correctly.

A column or table name cannot be identified by the database. Try changing the Bracket Mode to *Always* and remapping your form design. To change the Bracket Mode, open the Forms Manager and click the Options button. After changing the bracket Mode, click the Properties button, then click the Field Mapping button and re-save your field mapping.

Record is too large

All databases have limits on the number of fields and the total amount of data that can be stored in each record. For example, you can create a Microsoft Access table that has 50 text fields, each able to hold 50 characters. However, you will not be able to fill all the text fields of a single record to capacity because Microsoft Access has a limit of 2000 characters per record. The solution is to abbreviate responses in your record on the handheld, then resynchronize. If necessary, manually transcribe information from your record on the handheld to a document on your PC, delete the transcribed data from the handheld and resynchronize.

The changes you requested to the table were not successful because they would create duplicate values in the index, primary key, or relationship. Change the data in the field or fields that contain duplicate data, remove the index, or redefine the index to permit duplicate values and try again.

A new or modified record on the handheld cannot create a corresponding record on the PC because it would have a duplicate primary key value. By default, Pendragon Forms automatically adds to the database, records that were added on the handheld. It is possible that the record on the handheld was already inserted into the database table, but the handheld is unaware of this because the synchronization operation was interrupted. To confirm this, compare the field values on the handheld with the field values on the database table. If they match exactly, you can delete the offending record from the handheld.

If your Primary Key is unlikely to be a duplicate (e.g., if it contains a timestamp), you can set the Disable Record Overwrite Protection option in the Advanced Form Properties (see page 219) to enable newly created records on the handheld to overwrite existing records on the PC. This will prevent future occurrences of this problem due to synchronization interruptions.

You can't add or change a record because a related record is required in table <name>

A relationship exists between two tables in your external database. Pendragon Forms is not permitted to add a record to a table because a related record is first needed in the named table.

If you are also field mapping to the related table mentioned in the error message, it is possible that this error is caused by synchronizing the child table before its parent. This can usually be confirmed by performing a second synchronization. If the synchronization proceeds without errors, your child form is synchronizing before your parent form. To solve this problem, copy, freeze and field map your child form. Your child form will have a higher Form ID number and will synchronize after the parent form.

The field is too small to accept the amount of data you attempted to add. Try inserting or pasting less data.

A user has entered more text data on the handheld than will fit into the corresponding database column. Abbreviate the text on the handheld, or manually transcribe the data to the PC before deleting the text from the handheld. To prevent this from occurring in the future, either 1) enlarge the capacity of the database column, or 2) copy the form and impose a text length restriction on the field in question, before freezing and mapping the new form.

Can't start your application. The workgroup information file is missing or opened exclusively by another user.

Most frequently, this error occurs because you are field mapping to a password protected Access database. See the Knowledgebase article at www.pendragonsoftware.com for information about how to field map to password protected Access databases.

Data type mismatch in criteria expression

The column types in your database do not match the data types in the Additional Download Criteria or in the SQL generated by Pendragon Forms. Confirm that your Additional Download Criteria is appropriate for the column types involved. If you have chosen **Keep incomplete records on handheld** as your synchronization rule, try instead choosing **Keep a copy of records on handheld** and set equivalent Additional Download Criteria to implement the synchronization rule you need.

Problems with Sending Data from the Handheld to the PC

If you create records on the handheld and synchronize, but the records do not get sent to the Forms Manager database on the PC or server, here are some things to check.

Did Pendragon Forms synchronize?

On the PC or server, view the synchronization log for the user who is unable to send their data to the Forms database. See page 545 for information on how to view the synchronization log.

If there is a synchronization error message in the synchronization log, see pages 545 - 555 for help in resolving the problem.

What if there are No Synchronization Error Messages?

If the Pendragon Forms conduit does not generate any synchronization error messages, but your data on the handheld is still not uploading to the Forms Manager Database on the PC, there are some additional items to check, as described here and on the next few pages.

Check that the Form ID on the Handheld Matches the Form ID on the PC

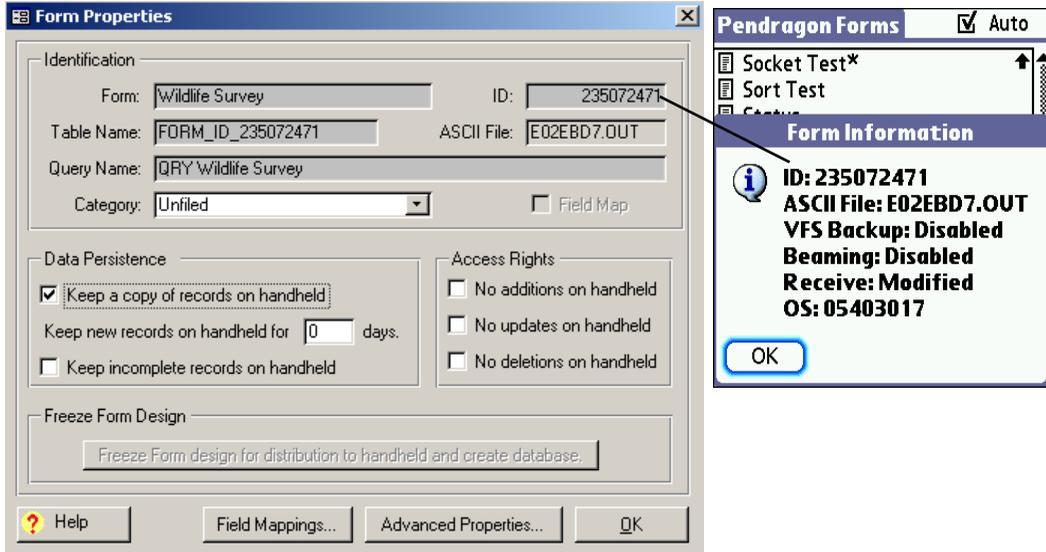
Pendragon Forms can only synchronize a form if the Form ID number on the handheld exactly matches a Form ID of a form design in the Pendragon Forms Manager database on the PC.

If there are no synchronization error messages, but a form on the handheld is still not sending its data to the PC, it is possible that the handheld has a different version of the form design from the Forms Manager database. This can happen if the form on the PC was accidentally deleted, and then a new form was created with the same name as the original form. The Form ID number on the PC will not be the same as on the handheld, even though the forms have the same name.

To check the Form ID number of the form on the PC, click the name of the form in the Forms Manager, and then click the Properties button. Look at the value in the ID field in the upper right of the Form Properties window.

PALM OS

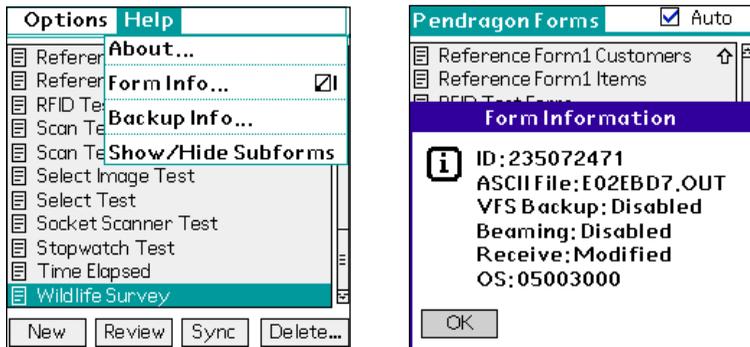
To check the Form ID number on a Palm OS handheld, tap the Forms icon on the handheld, then tap the name of the form. Tap the handheld Menu button (the drop down menu icon below or next to the House icon). On the Help menu, choose Form Info. A Form Information dialog will display the Form ID number.



Compare the Form ID numbers on the PC and on the handheld. If they do not match, you need to find the form design on the PC that does match. Check the Recycle Bin category (see page 41) in case the form design was accidentally deleted. If the form design is not in the Recycle Bin, try recovering the form design from the PilotF folder - see page 511. Or check any backup copies of the Forms Manager that you have.

PPC

To check the Form ID number on a Pocket PC handheld, tap on the name of a form, then tap the Pendragon Forms title bar. (If you do not see the title bar, minimize the onscreen keyboard). On the Help menu, tap Form Info.



Check if the Edit/View window in the Forms Manager is open

If you open the Edit/View window in the Forms Manager to view your data, and then synchronize, you will not actually see any new records in the Edit/View window until the window is refreshed.

Close the Edit/View window and then re-open it and check if your data is updated on the PC.

Are you Linking to an External Access or ODBC Database?

If you are linking to an external Access database or an external ODBC database, you may not be able to view your data from the Edit/View window within Pendragon Forms. Instead, go to your external database and check if the data from the handheld has been uploaded.

Check if you have more than one copy of the Forms Manager

If you have more than one copy of the Forms Manager on your PC, it is possible that the handheld may be synchronizing to one copy of the Forms Manager database, but you are looking in a different copy of the database and the data is not there.

You would not get any synchronization error messages if the handheld was synchronizing successfully with the other copy of the database.

To check the location of the database with which Pendragon Forms is synchronizing, click Start...Programs...Pendragon Forms...Configuration Tool.

On the Configuration tab of the Configuration Tool window, look in the Configuration Database field at the path to the database with which you are synchronizing. The default is:

C:\PROGRA~1\FORMS3\FORMS32K.MDB

where PROGRA~1 means the Program Files folder, so the default is

C:\Program Files\Forms3\Forms32K.MDB

Go into Windows Explorer and open the exactly named .MDB file in this location, to verify if your data synchronized successfully.

Technical Support

Technical support is available via the following:

- Visit the Support page at our Web site, <http://www.pendragonsoftware.com>
- E-mail: support@pendragonsoftware.com

Appendix C:

What's New in Forms 5.1

If you have been using Pendragon Forms 5.0 and you upgrade to Pendragon Forms 5.1, there are some changes to note.

New in Forms 5.1	Notes
<p>PPC</p> <p>The biggest change in Forms 5.1 is that on Pocket PC devices, the software now runs significantly faster, and the handheld user interface is closer to that of Palm OS devices than in Forms version 5.0.</p>	<p>As a result of a closer emulation of the Palm OS handheld interface on Pocket PC devices, users who were using the prior version of Forms (v. 5.0) on Pocket PC devices will experience the following changes:</p> <ul style="list-style-type: none"> - Currency fields only support 9 significant digits (the same as on Palm OS devices). - Signature and Sketch fields that were created on Pocket PC devices in Forms version 5.0 cannot be sent to Pocket PC devices running Forms version 5.1. - The Review screen now supports a maximum of 4 columns.
<p>On Typical installations of Pendragon Forms 5.1, the built-in Pendragon SyncServer will install as a Windows process and not a Windows service. Server installations will still install as a Windows service.</p>	<p>In Forms version 5.0, SyncServer always installed as a Windows service. For users who copied the Forms Manager database to a network drive, this caused a problem as the Windows service was not able to see the network drive. By installing SyncServer as a Windows process, the process can access the network drive if the logged-on user can access that drive. As a Windows process, SyncServer will have an icon in the Windows Task Tray.</p>
<p>Attaching images (including images taken with a built-in camera) is easier.</p>	<p>Forms 5.1 running on Palm OS 5.4 or Windows Mobile 5 for Pocket PC has a seamless way to attach a camera image to a Forms record. No custom controls are required.</p>
<p>Support for Microvision Flic bar code scanner.</p>	<p>BlueTooth-enabled Windows Mobile 5 for Pocket PC devices can receive bar code scans from a BlueTooth-enabled Flic scanner. Palm OS devices required additional software (pg 374)</p>
<p>Printer support via Bachmann Software PrintBoy v 7.0.</p>	<p>With PrintBoy 7.0, Palm OS devices and Pocket PC devices can print Forms records. HTML can be used to define print layouts.</p>
<p>New GPS Custom Control.</p>	<p>Supports UTM coordinates and horizontal dilution of precision.</p>

Appendix D: Pendragon SyncServer

What is Pendragon SyncServer?

Pendragon SyncServer is the component of Pendragon Forms 5.1 that allows handheld devices to synchronize. SyncServer is an application that allows multiple users to synchronize Pendragon Forms over a TCP/IP network. The TCP/IP network can be a Local Area Network (LAN), a Wide Area Network (WAN), or the Internet.

A free version of Pendragon SyncServer is built into Pendragon Forms 5.1. The free version of Pendragon SyncServer allows one user to synchronize to the Pendragon Forms Manager database at a time. If you have multiple Forms licenses, you can still use the free version of SyncServer as long as only one user synchronizes at a time. This is the case when multiple handhelds are all synchronizing at one PC.

If multiple users need to synchronize simultaneously to the Forms Manager database from different workstations, and the users attempt to synchronize at the same time, they may get 'Server is Busy' messages. The users can re-try synchronizing at different times, or you can consider purchasing a SyncServer license to allow multiple workstations to synchronize simultaneously to the Forms Manager database. Visit www.pendragonsoftware.com for SyncServer pricing information.

During installation of Pendragon Forms 5.1, if you choose to do a Server Installation, or if you are upgrading from Pendragon Forms 5.0, then Pendragon SyncServer will run as two system services: Pendragon SyncServer Service and Pendragon WatchDog Service. The Pendragon SyncServer Service is the Pendragon SyncServer, and the Pendragon WatchDog Service is a utility for restarting the Pendragon SyncServer. (See page 567 for information on the Watchdog Service.)

The benefit of Pendragon SyncServer running as a system service is that users do not need to log onto the server before they synchronize. Another benefit is that only users with Administrator access can switch off the system service, and so it is less prone to accidental shutdown by non-Administrator level users.

If Pendragon Forms 5.1 is installed as a Typical/Standalone installation, then Pendragon SyncServer will run as a process. Running as a process means that Pendragon SyncServer will appear as an icon in the Windows Task Tray.

How Pendragon SyncServer Works with Pendragon Forms

Pendragon SyncServer consists of server software that is installed when Pendragon Forms 5.1 is installed on a standalone PC or on a network server. Pendragon SyncServer also consists of client components that enable handhelds to synchronize either locally via a cable/cradle to a PC, or to synchronize wirelessly.

The main Pendragon Forms and Pendragon SyncServer components that are installed on the PC, server, workstation and handheld are shown in this table:

Programs that are installed:	Standalone PC or Network Server	Workstation synchronizing to a Network Server	Palm OS device synchronizing via cradle/cable to a Standalone PC or to a Workstation	Palm OS device synchronizing wirelessly	Pocket PC device (cradle/cable/wireless sync)
Forms Manager database	√				
SyncServer server software	√				
SyncServer client proxy conduit	√	√			
FORMS5.PRC program			√	√	
S-LINK program				√	
FORMS5 Pocket PC program & components					√

PALM OS

Palm OS users have two options, one for synchronizing locally via a cable or cradle, and one for synchronizing wirelessly.

When a Palm OS user performs a local synchronization, the HotSync Manager synchronizes each application in turn. Built-in Palm applications synchronize with the standalone PC or workstation. During its turn to synchronize, the Pendragon SyncServer Proxy Conduit sends Pendragon Forms data across the TCP/IP network to the Pendragon SyncServer application. Pendragon SyncServer then updates the Pendragon Forms database on the server, and sends back to the SyncServer Proxy Conduit any new or changed records and form designs. The SyncServer Proxy Conduit then updates the Forms application on the Palm OS device.

For users who have wireless Palm OS devices with TCP/IP connectivity, it is possible to synchronize Pendragon Forms without performing a HotSync data transfer.

To support this wireless synchronization of Pendragon Forms, the Pendragon SyncServer Link (S-Link) application must be installed on the Palm OS device, in addition to the Pendragon Forms (FORMS5.PRC) application. (See page 7.)

When the wireless Palm OS device user taps the Pendragon S-Link application icon on the handheld, the S-Link application will send new and changed Pendragon Forms records across the TCP/IP network to the Pendragon SyncServer application on the server. Pendragon SyncServer will update the Pendragon Forms database on the server, and send changes and updates back to the S-Link application on the Palm OS device. The S-Link application will then update the Pendragon Forms application on the Palm OS device.

Important: For synchronizing wirelessly, the use of the Pendragon S-Link application doubles the amount of memory required on the handheld, meaning that the handheld device needs to have as much free memory as the Forms database occupies on the handheld. For example, to synchronize 2MB of data, the handheld will need a total of 4MB available memory: 2MB for storage and 2MB during synchronization. Wireless synchronization also requires TCP/IP connectivity on the handheld. (Example: A Palm Tungsten C with WiFi or a Palm Treo 650 with wireless service have TCP/IP connectivity.)

PPC

On Pocket PC devices, the FORMS5 Pocket PC program is equivalent to having the FORMS5.PRC program and the S-Link program combined into one application for the Pocket PC platform. The Pocket PC user can set up the Pendragon Forms program on the handheld to automatically synchronize if a) the Forms program is running on the handheld, and b) the handheld is then placed in a cradle. Alternatively, the user can set up the Pendragon Forms program to synchronize only when the user initiates a manual synchronization. Manual synchronization is required for wireless synchronization.

During synchronization the FORMS5 Pocket PC application will send new and changed Pendragon Forms records across the TCP/IP network to the Pendragon SyncServer application on the server. Pendragon SyncServer will update the Pendragon Forms database on the server, and send changes and updates back to the FORMS5 Pocket PC application on the Pocket PC device.

Important: On all Pocket PC devices, whether synchronizing via local cradle or wirelessly, the synchronization process doubles the amount of memory required on the handheld, meaning that the handheld device needs to have as much free memory as the Forms database occupies on the handheld. For example, to synchronize 2MB of data, the handheld will need a total of 4MB available memory: 2MB for storage and 2MB during synchronization.

SyncServer Cache

Pendragon SyncServer maintains a synchronization cache on the PC to keep track of the data stored on each handheld device. This enables the software to accelerate synchronization by sending only changed records to the handheld. While this makes the server more processing intensive on the server PC, this generally results in much faster synchronization when large amounts of data are maintained on the handheld. Note that in cases where little or no data is maintained on the handheld device, or in cases where the synchronized data has no primary key, there will be no speed advantage gained with this caching scheme.

The cached copy of the handheld data is stored on the PC or server in the C:\Program Files\Forms3\Cache folder.

A separate file is stored in this folder for each handheld device. The size of the entire cache is approximately the size of the data that is stored on all handhelds. If you have 100 users with an average of 500KB of data on each handheld, the cache folder will have about 100 files with total size 50 MB.

When a form is re-distributed, the cached data for that form form design is re-initialized, meaning that it is no longer used. This means that synchronization will be slower when a form is re-distributed, as all the records designated for each handheld are re-sent to the handheld. If a form is not re-distributed, then the caching mechanism speeds up the synchronization since records that are already on the device do not have to be re-sent from the PC.

The cache does not store the images, signatures or sketches that are attached to records. This means that changing only an image, signature or sketch on the desktop will not cause the change to show up on the handheld unless another field that is not an image, signature or sketch is also changed.

The cache does not affect new or changed records on the handheld. New or changed records are always sent from the handheld to the PC during synchronization.

SyncServer Compression

Pendragon Syncserver compresses data before it is encrypted and transmitted across the network.

The compression uses the Zlib compression library. For typical data the compression rate is very good (generally > 50%). For some types of data the compression rate can be 90%.

Using compression when synchronizing via desktop PC is recommended; it is especially helpful when synchronizing via a PC that is connected to the network via a dial-up connection.

PPC

Pocket PC devices have synchronization compression built into the FORMS5 Pocket PC application on the handheld.

PALM OS

Palm OS devices that synchronize locally via a cable/cradle to a PC (standalone PC, server or workstation) have synchronization compression built into the SyncServer Client Proxy conduit.

Palm OS devices that synchronize wirelessly via the S-Link program on the handheld also receive an additional program, SYSZLIB.PRC, to enable synchronization compression. SYSZLIB.PRC is automatically installed on the handheld when you select to install S-Link on the handheld.

Pendragon SyncServer Maintenance

This section describes various Pendragon SyncServer maintenance tasks that the server administrator may have to perform.

Viewing and Maintaining the SyncServer Log

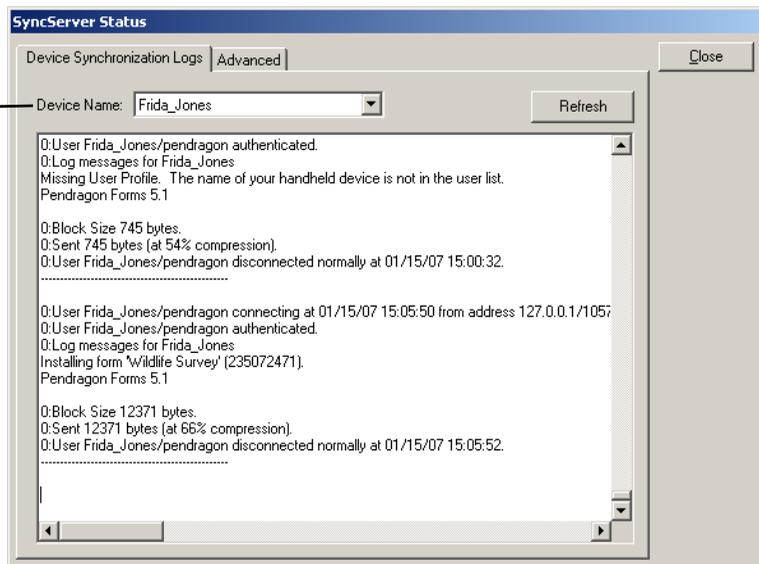
Pendragon SyncServer will create a log (PFSERVER.LOG) containing the times when users connected to the server and the databases that were opened.

The log file is typically in the C:\Program Files\Forms3 folder on the server.

To view the log, click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools. The SyncServer Status window displays the log.

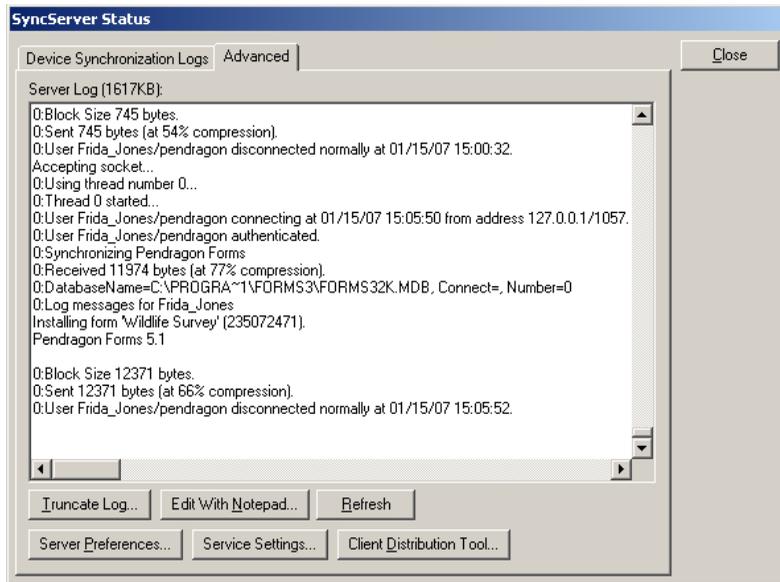
Select a Device Name (User Name) to view the synchronization Log for a particular user.

Select a User Name to view the synchronization Log for that handheld user.



The log is updated every minute in the SyncServer Status window. If you want to force an immediate update, click the Refresh button.

To view the log with all users displayed in the order in which the handhelds connected to the server, click the Advanced tab of the SyncServer Status window. (See next page.)



You can do the following on the Advanced tab of the SyncServer Status window:

To edit the log with Windows Notepad, click the Edit With Notepad button. Viewing the log in Windows Notepad allows you to search for a character string (such as a user's name or a particular error message), or to save the current log as a text file.

To reduce the size of the log file (PFSERVER.LOG), click the Truncate Log button. The log file will be truncated except for the most recent 1000 lines of the log. You will be prompted to confirm that you want to truncate the log.

Performing a Manual Re-Start of Pendragon SyncServer

If an unexpected error causes the Pendragon SyncServer to stop operating normally, you can manually restart SyncServer.

If Pendragon SyncServer is installed as a Windows process (the default for Typical/standalone installations of Pendragon Forms 5.1), then to re-start SyncServer:

Click Start...Programs...Startup...Pendragon SyncServer.

The SyncServer icon will re-appear in the Windows Task tray.

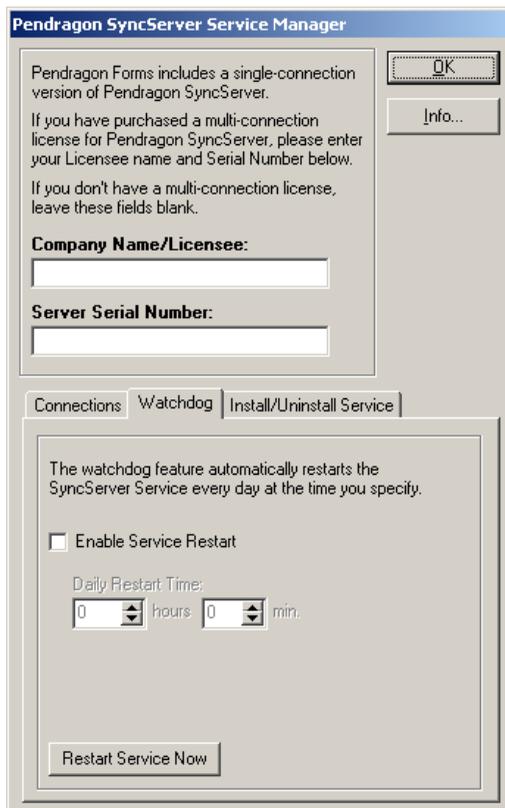
If Pendragon SyncServer is installed as a Windows Service on a server (or on an upgrade from Forms 5.0 that was also previously installed as a service), then to manually re-start Pendragon SyncServer on the server:

Click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools.

Click the Advanced tab of the SyncServer Status window.

Click the Service Settings button to view the SyncServer Service Manager window.

On the Watchdog tab, click the Restart Service Now button.



Enabling the Watchdog Service for Automatic SyncServer Restarts

If you are running Pendragon SyncServer as a Windows service on a server, the Pendragon Watchdog service is provided to give you the capability of restarting the Pendragon SyncServer service on a daily basis.

The Pendragon SyncServer service should not normally need to be restarted, but unexpected synchronization or communications errors may abnormally terminate the service or result in temporary loss of resources on the server.

If you find that the Pendragon SyncServer service needs to be restarted regularly, you can enable the Pendragon Watchdog service to restart the Pendragon SyncServer once daily at a time of your choice.

To enable the Pendragon Watchdog service:

Click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools.

Click the Advanced tab of the SyncServer Status window.

Click the Service Settings button to view the SyncServer Service Manager window.

Click the Watchdog tab. (See picture on previous page.)

To enable the Pendragon Watchdog Service, check the Enable Service Restart checkbox. Also specify a time in 24-hour clock format. 0 hours 0 minutes is midnight. 2 hours 0 minutes is 2am.

Click the Close button.

Note: It is recommended that you leave the Pendragon Watchdog service disabled unless Pendragon SyncServer regularly becomes unresponsive.

SyncServer Service Manager - Connections tab

To access the Connections tab of the SyncServer Service Manager:

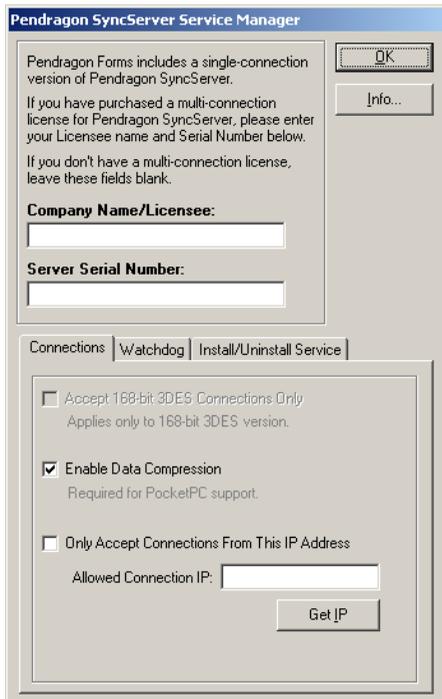
Click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools.

Click the Advanced tab of the SyncServer Status window.

Click the Service Settings button to view the SyncServer Service Manager window.

Click the Connections tab.

Connection options are as follows:



- The Accept 168-bit 3DES Connections Only checkbox tells the server to refuse connections from DES (56-bit) version of the SyncServer client software. This feature only works if you have the 168-bit 3DES version of the SyncServer software installed on the server. The 168-bit 3DES version of SyncServer is only available to users within the United States who purchase a SyncServer license.

Changes to this setting will take effect the next time the server is restarted.

- Enable Data Compression can be turned on and off at the server using the checkbox on this screen. Changes to this setting will take effect the next time the server is restarted. Enable Data Compression is switched on by default, and it is recommended that you leave this option switched on.

- Only Accept Connections From This IP Address provides a way to ensure that only local synchronizations via cradle/cable are allowed on a PC with a fixed IP address. To enable this option, check the checkbox and then enter the IP address of the PC. This option would prevent a person on the same network but on another PC from synchronizing Pendragon Forms over the network to your PC. Wireless synchronization to your PC would also not be allowed. This option provides some additional security for a Standalone (Typical) installation performed on a PC that is on a network.

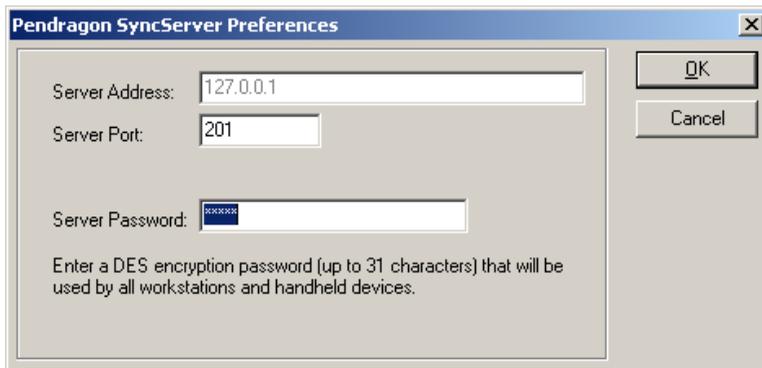
Changing the SyncServer IP Address, Port Number or Password

If you need to change the Pendragon SyncServer IP address, port number or password, you will need to make the changes on all of the following: the server, the client workstations, in the Pendragon S- Link application on Palm OS devices (see page 7), and in the Server Options settings on each Pocket PC device (see page 8).

Typical/Stand-alone PC or Network Server:

To change the port number or password on the server, click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools. Click the Advanced tab of the SyncServer Status window. Click the Server Preferences button. The SyncServer Preferences window will be displayed.

Change the Server Port number and/or the Server Password as needed. Note that if the server IP address changes, the new server IP address has to be entered on the client workstations and the handhelds, not the server.



Network Client Workstation:

PALM OS

If your Palm OS device synchronizes via a cable or cradle to a workstation PC, do this on the PC: Click Start...Programs...Pendragon Forms 5.1...SyncServer Client Preferences. In the Pendragon SyncServer Client Preferences window, change the Server IP Address, Server Port number and/or password as needed to match the new settings on the server.

If Palm OS devices are synchronizing wirelessly using the S-Link program, you will need to make any changes to the Server IP address, the Server Port and/or the Server Password in S-Link. See page 7.

PPC

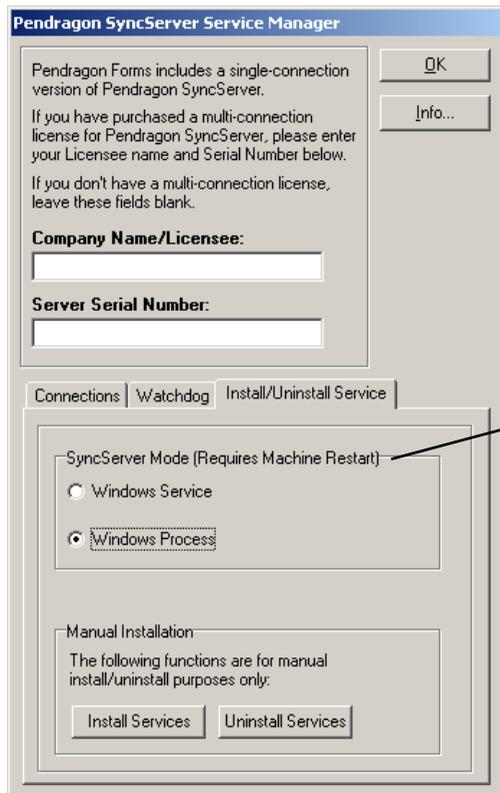
All Pocket PC devices will need to have the Server options screen on the handheld updated with any changes to the Server IP address, Server Port and/or Server Password. See page 8-9.

Changing Syncserver to run as a Windows Process or a Windows Service

By default, SyncServer 5.1 installs as a Windows process on Typical / Standalone installations of Pendragon Forms 5.1, and installs as a Windows service on servers and on upgrades from Forms 5.0 that were previously installed as a Windows service.

Sometimes, it may be necessary to change from running as a Windows process to running as a Windows service, or vice versa. For instance, if you install Pendragon Forms 5.1 as a standalone installation, and later decide to use that PC as your server, you will need to switch from running SyncServer as a Windows process to running SyncServer as a Windows service.

To access the Install/Uninstall Service tab of the SyncServer Service Manager:
Click Start...Programs...Pendragon Forms 5.1...Pendragon SyncServer Tools.
Click the Advanced tab of the SyncServer Status window.
Click the Service Settings button to view the SyncServer Service Manager window.
Click the Install/Uninstall Service tab.



In the SyncServer Mode field, select whether you want SyncServer to run as a Windows Service or a Windows Process, then click OK to close the SyncServer Service Manager window.

You will need to re-boot your PC for the change to take effect.

SyncServer Client Connection Error Messages

PALM OS

This type of error applies to a Palm OS device synchronizing via a cable/cradle to a client PC and then to a server. If there is a network connection error and the Pendragon SyncServer Client Proxy software fails to connect to the server, it will place the failure code into the log as follows:

Server Connection Failed: ABCD

Pendragon Forms with 1 message.

where **ABCD** is the failure code.

Here is a list of failure codes and their possible explanations. Some codes are listed for completeness, even if no explanation is listed:

Failure Code	Explanation
2745 - Software caused connection abort	An established connection was aborted by the software in your host machine, possibly due to a data transmission timeout or protocol error.
274D - Connection Refused	The IP address or Internet address entered for the server is pointing to a PC that is not running the Pendragon SyncServer software. Alternatively, there may be a firewall between you and the specified server that is blocking the connection.
274C - Connection timed out	There is a firewall blocking packets between the workstation and the server, or else there is no machine currently running at the remote server address.
2746 - Connection reset by peer	The Pendragon SyncServer closed the connection, possibly because the server software has abnormally terminated at the server.
2737 - Destination address required	No address was provided for the server. Check your Pendragon SyncServer Proxy Preferences applet to see if the server address was entered.
2754 - Host is down	The server PC or its network subsystem is unavailable or has timed out.
2755 - No route to host	The software could not find a route across the network to your server.
2714 - Interrupted function call	The connection was cancelled, possibly because the server connection timed out.
2726 - Invalid argument.	This error may occur if you use AOL as your Internet provider, but you have not signed on to AOL before attempting your HotSync operation.

Failure Code	Explanation
2728 - Too many open files	Your PC has reached its limit of Internet connections. Try closing unused Internet browser windows or other Internet applications that you are no longer using.
2738 - Message too long	An application other than Pendragon SyncServer may be running on the remote server port.
2742 - Network is down	If possible, restart your PC and try your synchronization again. If this does not work, contact your network administrator.
2744 - Network dropped connection on reset	The server you were connected to crashed and rebooted.
2743 - Network is unreachable	The software could not find a route across the network to your server. It is possible that a firewall is blocking packets between you and the server.
2736 - Socket operation on non-socket	This error may occur if you use AOL as your Internet provider, but you have not signed on to AOL before attempting your HotSync operation.
2753 - Too many processes	Your PC has reached its limit of Internet applications. Try closing unused Internet browser windows or other Internet applications that you are no longer using.
2AF9 - Host not found	The address you entered for the server location could not be found on your DNS server. Check the address. If it is correct, try entering the IP address of the server instead.
276B - Network subsystem is unavailable 276C - WINSOCK.DLL version out of range	TCP/IP networking is not working correctly on your PC. Contact your network administrator.
2AFA - Non-authoritative host not found	Try your synchronization again later.
276E - Graceful shutdown in progress	The remote server has disconnected.

The following codes have not occurred on test or user systems:

271D - Access Denied	273A - Bad protocol option
271E - Bad address	2749 - Socket is not connected
273F - Address family not supported by protocol family	273D - Operation not supported
2740 - Address already in use	273F - Protocol family not supported
273B - Protocol not supported	2741 - Cannot assign requested address
2735 - Operation already in progress	2739 - Protocol wrong type for socket
2734 - Operation now in progress	274A - Cannot send after socket shutdown
2746 - No buffer space available	273C - Socket type not supported
	2733 - Resource temporarily unavailable

Appendix E: GPS Custom Control

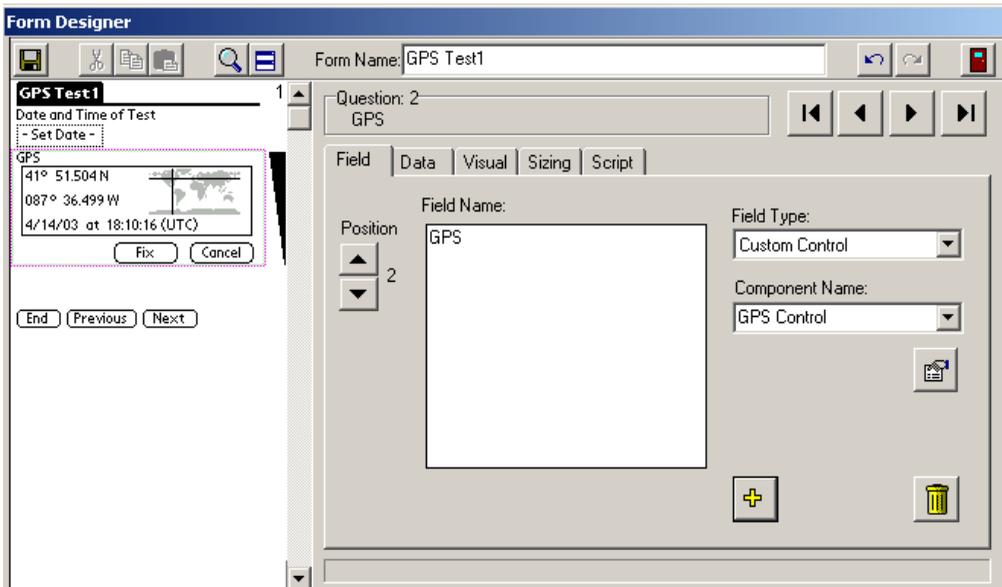
The GPS Custom Control is a program that extends Pendragon Forms to use GPS receivers. The custom control can communicate with an external GPS receiver via a serial port or a virtual serial port. Virtual serial ports simulate serial ports but get their data from Bluetooth-connected devices or internal GPS receivers.

The control expects a stream of data conforming to the NMEA 0183 protocol. In particular, the control makes use of the GPRMC and GPGGA sentences in the NMEA stream.

Adding the GPS Control to Your Form

To add the GPS Control to your form, create a new field on your form and select the Custom Control field type. For the Component Name, select "GPS Control".

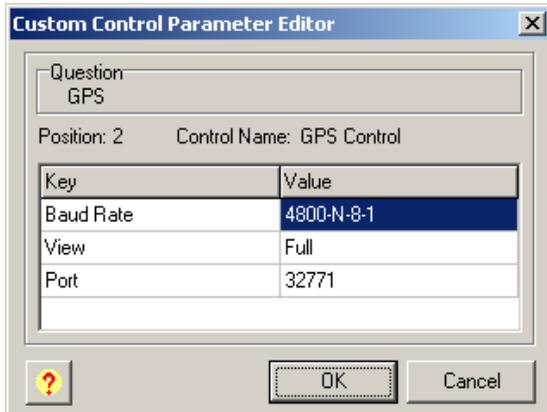
You will need to click on the sizing tab in the Form Designer window to make the Answer Height large enough to view the GPS map and buttons.



The GPS Control has three adjustable parameters. To change their values to suit the type of GPS device that you are using, click the Parameters button:



The Custom Control Parameter Editor will be displayed:



Enter the values for Baud Rate, View and Port that are appropriate for your GPS device. See below and on the next page for descriptions of these parameters.

Baud Rate Parameter

The Baud Rate parameter determines the serial communications speed and protocol. The Baud Rate setting has the following format:

bbbb-p-d-s

where bbbb is the baud rate, p is the parity setting, d is the number of data bits and s is the number of stop bits. The default for this parameter is **4800-N-8-1**.

bbbb can be one of the following: 300, 600, 1200, 2400, **4800**, 9600, 19200, 38400, 57600.

p can be one of the following: **N** (no parity), **E** (even parity) **O** (odd parity).

d can be 7 or **8**.

s can be **1** or 2.

View Parameter

The View Parameter determines whether the GPS Control displays a map or not. The default value for this parameter is **Full**, which displays a map as shown on page 574. Any other value for this parameter removes the map from the display.

Port Parameter

The Port parameter selects the serial port to use to communicate with the external GPS device. The following table provides suggested values for the Port parameter, depending on the type of handheld device you are using.

Device	Recommended Port Value
Palm OS 3.5 - Palm OS 4.1	-1
Palm OS 5.x (Tungsten T/C, SONY NX/NZ series)	32771
Palm OS 5.x (Infrared IrComm Connection)	ircm
Palm OS 5.x (Bluetooth RfComm Connection)	rpcm
Garmin iQue 3600 (Built-in GPS)	ique
Pocket PC using serial port 1	32771
Pocket PC using serial ports 2 - 8	2 - 8 (e.g: 2, 3, 4, 5, 6, 7 or 8)
Pocket PC using Bluetooth	rpcm

Notes

The GPS Control was developed and tested with a Garmin eTrex GPS receiver and with the Socket Communications Bluetooth GPS receiver. The port and baud rate parameters should enable the GPS Control to work with most GPS receivers using NMEA serial data streams, but there will be some devices that are not compatible with the GPS control. For those devices that do work, you may need to experiment with the settings initially in order to find settings that work with your hardware.

Installing the GPS Control

New in Forms 5.1, the GPS Custom Control is automatically installed when you install Pendragon Forms 5.1 on either a Palm OS device or a Pocket PC device. No additional files need to be installed.

Pairing a Handheld with a Bluetooth-enabled GPS Receiver

If you are using a Bluetooth-enabled GPS device, you will first need to pair the handheld with the GPS device.

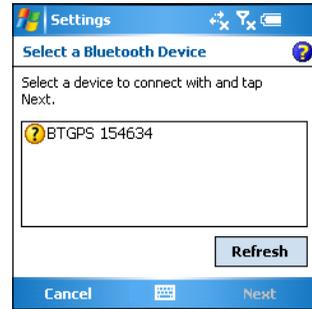
The pictures here show a Windows Mobile 5 device. Palm OS devices will require similar selections. To access the Bluetooth Setting screen, tap the Bluetooth icon on the handheld.



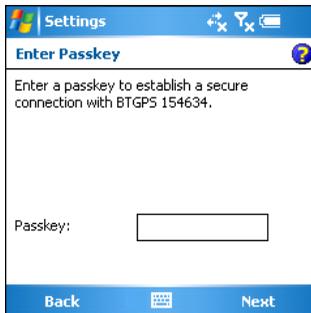
Step 1: On the Mode tab, make sure that Bluetooth is switched on.



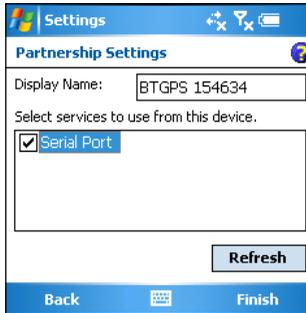
Step 2: On the Devices tab, tap New Partnership... Switch on the Bluetooth GPS receiver.



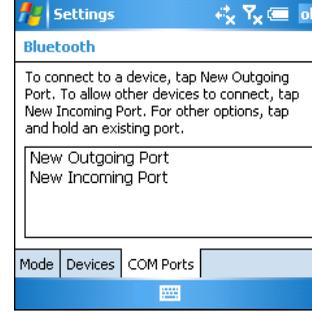
Step 3: When the handheld detects the GPS device, tap on the GPS device name and tap Next.



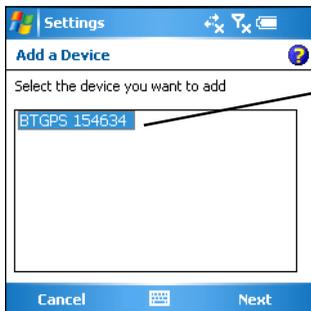
Step 4: Enter the passkey for the GPS device and tap Next.



Step 5: Select to use a Serial Port and then tap Finish.

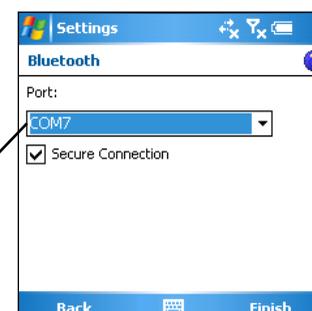


Step 6: On the COM Ports tab, tap New Outgoing Port.



Step 7: Tap on the name of the GPS device and tap Next.

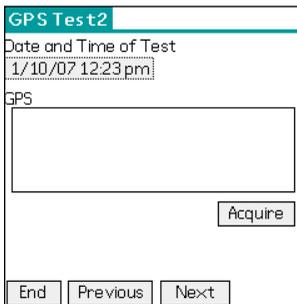
Step 8: Select COM7 for the COM port and tap Finish.



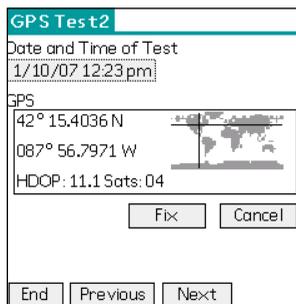
Using the GPS Control on the Handheld

On the handheld, the GPS Control initially appears as an empty box with an Acquire button. Tap the Acquire button to open the serial port and connect to the GPS unit.

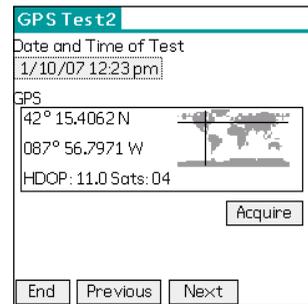
The GPS Control will attempt to connect to your GPS device for up to three minutes. Initially, the control will display a countdown to deactivation. Most GPS receivers will begin transmitting data within 30 seconds. At this point the GPS Control will display the most recent Latitude, Longitude and Time received from the GPS device.



Step 1: Tap the Acquire button to initiate contact with the GPS receiver.



Step 2: When the GPS receiver locates data from at least 3 satellites (optionally more), tap the Fix button.



Step 3: The GPS coordinates for Latitude, Longitude and Time will be transferred into the GPS field on your form.

While the GPS receiver is receiving data from the satellites, if you tap the **Cancel** button you will halt communications with the GPS device and close the serial port.

The **Fix** button will be displayed once the data from the GPS device is determined to be valid. Tap the **Fix** button to transfer the GPS coordinates into the GPS field in Pendragon Forms. The **Fix** button also triggers a **click:** event script if such a script is present in the GPS field of the form.

The Pendragon Forms GPS field will store the captured coordinates in the following format:
42 15.3995 N 087 56.7968 W:: 1/8/2007 22:05:46 (UTC)

If you need to store Latitude, Longitude or other GPS data in separate fields, see the Advanced GPS Features on the next page.

Advanced GPS Features

The GPS Control can be queried for additional information using a "GPSINFO" method. The method can be called from a Pendragon Forms script using the CALLMETHOD statement. Typically, the method would be called from a **click**: event script in the GPS Custom Control field on the form. The data can be extracted from the XML using the EXTRACT statement.

The GPSINFO method returns a block of XML-formatted data in the RESULT variable. The XML data block looks like the following (note that not every GPS receiver provides all these items of data):

```
<GPS>
<LAT>deg mm.ssss X</LAT> (latitude, X = N or S)
<LONG>deg mm.ssss X</LONG> (longitude, X = E or W)
<DECLAT>deg.nnnn X</DECLAT> (latitude in decimal degrees, X = N or S)
<DECLONG>deg.nnnn X</DECLONG> (longitude in decimal degrees, X = E or W)
<TIME>hh:mm:ss</TIME> (UTC Time i.e. Greenwich Mean Time)
<DATE>dd/mm/yy</DATE> (UTC Date)
<KNOTS>nnnn.n</KNOTS>
<COURSE>nnn.n</COURSE>
<MAGVAR>nn.n</MAGVAR> (Magnetic variation i.e. variation from true north)
<SATELLITES>nn</SATELLITES> (Number of satellites receiving data from)
<HZDIL>nn.n</HZDIL> (Horizontal dilution of precision (i.e. uncertainty), in meters)
<ALTITUDE>nnnnn.n</ALTITUDE> (Meters above Mean Sea Level)
<HTGEOID>+-nn.n</HTGEOID> (Meters)
<ZONE>nn</ZONE> (UTM Zone Number)
<EASTING>nnnnnnnnn.n</EASTING> (UTM Easting)
<NORTHING>nnnnnnnnn.n</NORTHING> (UTM Northing)
</GPS>
```

For example, if your GPS Control field is field 5 on your form, and fields 6, 7 and 8 are reserved for Latitude, Longitude and Altitude respectively, then placing the following script in field 5 will extract fields 6, 7 and 8 when the **Fix** button is tapped:

```
click:
  callmethod 5 "GPSINFO"
  temp = result
  extract "GPS.LAT" from temp
  $6 = result
  extract "GPS.LONG" from temp
  $7 = result
  extract "GPS.ALTITUDE" from temp
  $8 = result
```

Index

A

- Access database
 - Linking to external 486,488
 - Pendragon Forms Manager 510
- Access Rights 199
- Additional Download Criteria 214
 - Completion Checkbox Criteria 217
 - Criteria for Sorting Records 218
- Advanced Field Properties 185
 - Autodefault 187
 - Column Name 186
 - Data Tab 168
 - Default Value 105,148,191
 - Display Key 190
 - Do Not Upload to PC 196,495
 - Integer 91,192
 - Lookup List 195
 - Max & Min 91,151,193
 - Max Length 88,194
 - Non-Printing 196
 - Pattern 194
 - Primary Key 31,188,494
 - Required 189
- Advanced Form Properties 197,202
 - Additional Download Criteria 214
 - Archive Date 222
 - Auto-Execute 208
 - AutoRepeat This Form 208
 - Beaming Rules 224
 - Beaming Tab 223
 - Behavior Tab 207
 - Changing Advanced Form Properties 225
 - Columns Tab 226
 - Defaults Tab 202
 - Disable Action Menus 208
 - Disable AutoNavigation 208
 - Disable Field View 206
 - Disable Form Deletion 212
 - Disable Navigation Buttons 206
 - Disable Record Overwrite Protection 219
 - When beaming records 452
 - Disable Text Field Icons 205
 - Display as Subform 138,209
 - Don't Sort on Subform Screen 209
 - Hide Delete Button 212
 - Hide Form in Forms List 138,209
 - Hide Review Button 206
 - Hide Section Skip Button 205
 - Mark Received Records as Modified 225,452
 - Merge Changes 220
 - When beaming records 453
 - No Back Button 204
 - No End Button (Field/Layout View) 204
 - No End Button (Record View) 206
 - No Record View Button 205
 - Password Protecting a form design 213
 - Read-Only Reference form 129,221
 - Record Level Undo 207
 - Screen Level Validation 207
 - Security Tab 210
 - Storage Limit 222
 - Synchronization Tab 214
 - Synchronize Only When Distributed 161,222
 - VFS Backup 211
 - Views Tab 204
- All Users Group 47
- Answer Height 16,182,183
- Answer Icons 177
- Answer Placement 181
- Archive Date 222
- ASCII Backups 511
- Auto-Execute a form 208
- Auto-Sync in Cradle (Pocket PC) 9,10
- Autodefault 187
- AutoNavigate 72,208
- AutoRepeating a form 208
- Average
 - Value across records 63

B

- Backing up the Forms database 510
- Backup
 - To external media card 211,512,514
- Backups, ASCII 511
- Bar Code Scanners 373
 - Controlling which bar codes can be scanned 394
 - Microvision Flic Cordless 374,376,378
 - Socket Secure Digital Scan Card 379
 - Symbol Technologies devices 373
- Beaming Records 445
 - Allowing beam receiving on handheld 446
 - Memory Requirements for 454
 - Scripts to allow beaming 447,448
 - Software required 446
- Beaming Rules 224,451
- Beaming Tab 223
- Behavior Tab 207
- Bluetooth
 - GPS Receiver 577
- Branching 357
- Button field 148,364,365,447

C

- Calculations 347
- Cascading Completion Checkbox (Parent to Subform) 139
- Cascading Lookup 119,370
 - Double cascading lookup to form 463
 - To another form 461
- Categories for organizing form designs 40,200
- Columns Tab 226
- Column Name 186
- Completion Checkbox field 107
- Copying a form 39
- CSV file 243
 - Creating a form from 244
- Currency field 92
- Cursor
 - Switching off in a Text field 178

- Custom Control Field 162
 - GPS Custom Control 164,574
 - Parameters 163,575
 - Stopwatch Custom Control 164
- Custom Main Menu 471
 - Accessing Pendragon Forms main menu 482,483
 - Auto-Executing 472
 - Deleting records 480
 - Filtering Records 484

D

- Data
 - Editing Data on the PC 227
 - Exporting data to ASCII (CSV File) 230
 - Exporting data to Microsoft Excel 230
 - Importing & Exporting data 242
 - Viewing data on the PC 227,229
- Data Persistence 19,25,51,198,498
- Data Tab 168
- Date & Time field 100
- Date field 102
- Defaults Tab 202
- Default User Group 47
- Default Value 105,148,191
- Default View 202
- Deleting
 - Data records on the PC 230
 - Forms 50
 - From the handheld 67,68,69
 - Lookup Lists 118
 - Users 46
 - Users and forms from a User Group 48
- Disable Action Menus 208
- Disable AutoNavigation 208
- Disable Form Deletion 212
- Disable Navigation Buttons 206
- Disable Record Overwrite Protection 219
- Disable Text Field Icons 205
- Display as Subform 209

Display Key 60,136,137,190,391
 Of a Read-Only Reference form 128
 Of a Subform 136
Distributing a form 20
Don't Sort on Subform Screen 209

E

E-Mailing a Record 441
Error Messages
 Freezing a Form Design 531
 HotSync Log 546
 in the Form Designer 530
 Synchronization Log 545
 Synchronizing a Pocket PC device 547
 Synchronizing a Wireless Palm OS device
 547
Event Procedures
 Definitions of 250,251,252,253,254
Exclusive Lookup List field 113
Exporting
 Data to ASCII (.CSV file) 230
 Data to Excel 230
 Form designs 240
 Lookup Lists 241
External Media
 Backing up to 512
 Purging old records 517
 Restoring records from 514
 Warnings about 513

F

Field
 Starting on a new screen 170
Field Compatibility with External Databases
 488,489
Field Height 16
Field Labels 247
Field Level Validation 207
Field Mappings 500
Field Name 74
Field Numbers 73
Field Tab 167

Field Types 74,75
 Button field 148
 Cascading Lookup 119
 Completion Checkbox 107
 Currency 92
 Custom Control 162
 Date 102
 Date & Time 100
 Freeform Text 75,87
 Image Field 155
 Jump to Section 111
 Lookup List 113
 Lookup to Another Form 123,457
 Lookup to a Read-Only Reference form 127
 Multi-Selection List 97
 Numeric Field 90
 Option 1 of 5 94
 Popup List 95
 Section 108,111
 Adding a Picture 171
 Signature 145
 Single Subform 142
 Sketch 153
 Slider 150
 Subform 133
 Text 75,87
 No Cursor Blinking 178
 Time 103
 Time Checkbox 106
 Yes or No Checkbox 104
Field View 59
Filtering Records 64
 From a Custom Main Menu 484
Font 16
 Font size and color 172,173
Form
 Cascading Lookup to another form 461,463
 Copying a form 39
 Creating a form 13
 Creating from a CSV File 244
 Deleting a form from the PC 42
 Form ID 20,235,556,557
 Form Name 14

- Limits of Form Designs 30
- Lookup to another form 123,457,459
- Properties 51,197,498
- Read-Only Reference form 128
- Recovering a form design 511
- Reference form 124
- Forms Main Menu on handheld 57
- Forms Preferences 70
- Form Designer 14,15,74,165
 - Data Tab 168
 - Error Messages 530
 - Field Tab 167
 - Script Tab 184
 - Sizing Tab 172,180
 - Visual Tab 169
- Form Properties 197
 - Access Rights 199
 - Advanced Form Properties 202
 - Category 200
 - Data Persistence 198
 - Freezing a Form 201
- Freeform Text field 87
- Freezing a form design 19,201
 - Troubleshooting 531
- Functions used in scripts 257

G

- GPS Custom Control 162,163,164,574
 - Extracting GPS Data into separate fields 579
 - Parameters 575,576
 - Using on the Handheld 578

H

- Handheld
 - Access Rights 199
 - Backup to external media 512,514
 - Deleting records and forms 67,68
 - Entering New Records 57
 - Field Numbers 73
 - Field View 59
 - Filtering records 64
 - Forms List 57

- Forms Preferences 70
- Keeping records on the handheld 198
- Layout View 58
- Limiting the number of records stored 51,498
- Memory Requirements 26
- Quick Sum and Average 63
- Record View 59
- Restoring records from external media 514
- Reviewing records 60
- Sorting records 65
- Height of a field 16
- Hidden fields 176
- Hide Delete Button 212
- Hide Form in Forms List 209
- Hide Review Button 206
- Hide Section Skip Button 205
- Hi Res Image option 156
- HotSync Error Messages 545

I

- Images
 - Attached (Dynamic) Image 419,422
 - Attaching on the Handheld 424,426
 - Attaching on the PC 159,423
 - Decorative (Static) Image 419,420
 - Adding to a form design 171
 - File Naming Convention 161
 - Image field type 155
 - Location of image files on handheld 436
 - Non-Attached Image 419,433
 - Saving image files outside the Forms database 437,438
 - Synchronization of images 436
- Importing
 - Data from ASCII (.CSV file) 242
 - Data from a different form design 239
 - Form designs 240
 - From a previous version of Pendragon
 - Forms 237
 - Lookup Lists 241
 - Inline Image option 156

Installing Pendragon Forms

- On a Network Client PC 11
 - On a Network Server 1,520
 - On a Palm OS device 6,55
 - On a Pocket PC device 8,56
 - On a Standalone PC 1,520
 - On a Wireless Palm OS device 7
- Integer 192

J

Jump to Section field 111

K

Keypad

- Displaying a Numeric Keypad 169
- Keypad Icon 178

L

Layout View 58,203

Limiting the number of records on the handheld 51

Limits

- Of Form Designs and Records 30
- Linking to an External Access database 486
- Linking to an ODBC database 506
- Lookup List 113
- Cascading Lookup 119,461
 - Deleting 118
 - Exclusive Lookup 116
 - Field type 113
 - Importing/Exporting 118,241
 - In a Text field 195
 - Lookup Display column 114
 - Lookup Value column 115
 - To another form 123,457
 - To a Read-Only Reference form 127
 - Using with a Text field 195
- Lookup to Another Form 123,457

M

Managing Data on the PC 227

Mark Received Records as Modified 225

Max & Min (Setting a range in a numeric field) 193

Max Length (of a Text field) 194

Memory Requirements on Handheld 26

- Wireless handheld 29

Merge Changes 220

Multi-Selection List field 97

Multi-User environment

- Multi-user installation 520
- Multi-User License Code 518
- Security concerns 521,522
- Sharing records across handhelds 31,32,33,34,519

N

No Back Button 204

No End Button (Field/Layout View) 204

No End Button (Record View) 206

No Focus 178

No Record View Button 205

Numeric Field 90

Numeric Keypad 169

O

Option 1 of 5 field 94

Organizing forms into categories 40

P

Parent Form 133,142,367,458

- Beaming records 450
- Cascading Completion Checkbox 139

Password

- Entered in S-Link on Palm OS device 7
- Server Password 4

Password-Protecting a Form Design 69,213

Popup List field 95

Preventing a field from printing 196

Preventing a field from uploading to the PC
196
Preview Area 15,18,165,176,183
Primary Key 31,188,447,486,494
Printer Driver Software 399
Printing
From the handheld 399
Purging old records from external media 517

Q

Question and Answer on one line 180,181
Question Height 16,182,183
Question Spacing 182

R

Re-positioning a field on a form 18
Read-Only field 179
Read-Only Reference form 127,221,389
RecordID 23,70
Records
Backing up to external media 512
Deleting records and forms from the handheld 67,68,69
Displaying in Field View 59
Displaying in Layout View 58
Displaying in Record View 59
Filtering records 64
Keep incomplete records on handheld 107
Restoring from external media 514
Reviewing records on the handheld 60
Sorting records 65
Record Level Undo 207
Record View 59
Recovering a Form Design 511
Recycle Bin 41
Reference form 123,124,389,461,463,466
Reports
Creating a report in Microsoft Access 235
Creating a report in Microsoft Excel 230
Creating a report in Microsoft Word 231
Required field 189

Reviewing Records
Color of Record Lists 70
On the Handheld 60,61
Selecting Columns to view 62
Right-Justifying Numeric and Currency fields
175

S

S-Link application 6,7
Saving
Form Designs 17,18
Scripting 245
Adding Comments to Scripts 248,249
Branching 357
Creating a script in a field 246
Errors on the handheld 341,342
Event Procedures 250
calculate: 252,348,349,350,351,352,354,355,356
cascade: 253
click: 252,357,364,365,367,368,371,448,462
enter: 250
enterscreen: 250
exit: 251,357,363
exitscreen: 250,357,358,360
initialize: 251,353,354
open: 251
report: 254,400,405,412,414
report2: 254,400,412,414
scan: 253,357,385,387,391,393
select: 251,357,358,360
validate: 253
Field Labels 247,248
Format of a script 246
Functions 257
cascadename 259
column 279,321
flashid 257
getaddress 364
linenum 259,414
now 257
pagenum 259,414

- recordtimestamp 257
- reportname 259
- scandata 257,385,387,391,393
- scanfield 257
- scantype 257,397
- selectedbutton 258
- tapx 258
- tapy 258
- username 258
- webdata 258,443
- Limits 246
- Operators 260
 - Binary 261
 - Unary 260
- Platform-Specific Scripts 249
- Scientific Functions 260,343
 - acos 344
 - asin 344
 - atan 344
 - cos 343
 - cosh 344
 - exp 345
 - log 346
 - log10 346
 - pow 345
 - round 346
 - sin 343
 - sinh 344
 - sqr 345
 - tan 344
 - tanh 344
 - trunc 346
- Statements 262
 - $\$N = \langle \text{expression} \rangle$ 263
 - abortform 270
 - acquire 270
 - also 271,460,465,484
 - answer = $\langle \text{expression} \rangle$ 262,349,350
 - assign 264
 - beam 272,448,449
 - beep 274
 - buffer = $\langle \text{expression} \rangle$ 264
 - call 274
 - callmethod 275
 - cascade 276
 - checkflag/clearflag 277
 - clone 278,391
 - column 279,460
 - count 279
 - delete 280,460
 - deletemode 280,480
 - disable
 - barcode 281,381
 - endbutton, nextbutton, backbutton, navbuttons, icons, menus 282
 - grabba 283,383
 - drawimage 284,433
 - enable
 - barcode 286,381
 - endbutton, nextbutton, backbutton, navbuttons, icons, menus 287
 - grabba 286,383
 - endform 288,482
 - exportfile 289
 - extract...from 289,443
 - fieldview 290
 - filtercount 291
 - font
 - answer, question, bold, normal, large 291
 - format
 - date, time, datetime, currency, fixed 292,405
 - formsum 293
 - getaddress 294,364
 - goto 295,358,360,363
 - gotosubform
 - new, review, normal 460,475
 - gotosubform { new | review | normal } 296
 - hide 297,357
 - hide from... to 297
 - if...then...else...endif 265,267,351,387
 - if..then...endif 265,350,393
 - imagefilename 297
 - importfile 298
 - insert into 298,460,470
 - invalidate 299

-
- keycolumn 299,302,391,460
 - keyunique 300
 - launch 301
 - left 301
 - lookup...within 302,391,460
 - mid 303
 - msgbox 303,387
 - nullfrom...to 304
 - optional 305
 - optional from...to 305
 - PALMONLY 249
 - PPCONLY 249
 - print
 - serial, IR 306
 - printhtml 306,408
 - printleft 307,405
 - printline 307,405,418
 - printmemo 308,416
 - printnewpage 308,414
 - printreport 309,400,403,405
 - printright 310,406
 - queue 272,311,448,449
 - readonly from...to 311
 - readwrite 311
 - readwrite from...to 311
 - require from...to 312
 - result = <expression>
 - 263,349,350,352,365
 - return 351,387,393
 - reverselookup...within 313
 - review 313,476,484
 - right 312
 - scanner {checkdigit | no_checkdigit}
 - 316,395
 - scanner {opcc_checkdigit | uss_checkdigit}
 - 317,382
 - scanner {system_character | no_preamble}
 - } 316,395
 - scanner {verify | no_verify} 317,382
 - scanner {enable | disable} symbology
 - 315,382
 - scanner {enable | disable} all 314,382
 - scanner conversion-code { enable | disable
 - } 314,382,396
 - scanner system_character_country_code
 - 316,382
 - selectfile 322,427
 - select all 317,450,460,476
 - select current 318,448
 - select matching 319,460
 - select where 320,449,460,462,464,465
 - setlookuplocale 323,372
 - setlookupname 324,325,326,370
 - setparam allowhtml 325,411
 - setparam modified 325
 - setparam printhtmlfooter 326,411
 - setparam printhtmlheader 326,411
 - setview 327,369
 - show 328,357
 - show from...to 328,357
 - socketscan
 - lengthparameter...length 328,383
 - setting...value 330,384
 - symbology...value 332,384
 - socketscan defaults on 328,383
 - socketscan set_trigger 329,384
 - socketscan trigger now 332,384
 - subformsum 333,367
 - switch/case 268,269,361
 - synchronize 333
 - temp = <expression> 264
 - transmit multimail 334,440,441
 - transmit palmnet 335
 - transmit printer 337
 - transmit serial 338
 - transmit web 339,440,443
 - update field 340,460,470
 - Using Scripts to control Printer Output 400
 - Using scripts to perform Calculations 348
 - Using scripts with Button fields 364
 - Using scripts with Date fields 354
-

- Variables 255
 - \$ 255
 - answer 255
 - buffer 256
 - lookuplocale 256,372
 - lookupname 256,370
 - null 256
 - result 255
 - temp 255
- Scripting Functions 257
- Script Tab 184
- Section field 108,111
- Security concerns in a Multi-User Environment 521
- Security Tab 210
- Sharing Records on Multiple Handhelds 31,32,33,34
- Shortcut Icon on Pocket PC 56
- Signature field 145
- Sizing Tab 172,180
- Sketch field 153
- Slider field 150
- Sorting records 65
- Spacing of questions 182
- Starting a field on a new screen 170
- Stopwatch Custom Control 164
- Storage Limit 28,222
- Subform
 - Regular subform 133,209,458
 - Single subform 142,209
- Sum
 - of records on handheld 63
- Switching On/Off
 - Cursor Blinking in a Text field 178
 - Field Numbers on the handheld 73
 - Synchronizing One Form Design 54
 - Synchronizing Pendragon Forms 53
- Synchronization Log 545
- Synchronize Only When Distributed 221
- Synchronizing a Pocket PC device 10

- SyncServer 560
 - Compression 564
 - Maintenance 565
 - Manual Re-start 567
 - Running as a Service or a Process 521,571
 - Server IP Address 12,570
 - Server Password 4,7,12,523,570
 - Server Port 3,7,9,12,570
 - Synchronization Cache 44,563
 - Synchronization Log 545,565
 - Watchdog Service 567,568

T

- Text field 87
 - Using with a Lookup List 195
- Timestamp 23
- Time Checkbox field 106
- Time field 103
- Troubleshooting
 - Deleting forms 50
 - Exporting to Excel 230
 - Form Designer Error Messages 530
 - Linking to an External Access Database 541
 - Problems Sending Data from Handheld to the PC 556
 - Problems Sending Form Designs or Data to the Handheld 527
 - Problems with Distributing a form 534
 - Problems with Freezing a Form Design 531
 - Problems with Lookup Lists 535
 - Problems with Subforms or Single Subforms 536
 - Recovering a Form Design 511
 - Synchronization Error Messages 545
 - Working with Multiple Forms 539

U

UnitID 23

UserName 5,23

 In Forms Manager User List 46

 In S-Link on Palm OS device 7,9

Users and User Groups 45

 Adding users 45,46

 Adding users and forms to a User Group 48

 Creating User Groups 47

 Default User Group 47

 Deleting users 46

V

Variables used in scripts 255

VersaMail 441

VFS Backup 211,512,513

Viewing

 Data on the PC 23,227,229

 Images on the PC 437

 Signatures on the PC 147

 Sketches on the PC 154

Views Tab 204

W

Wireless Palm device 7

Y

Yes or No field 104