Column: From the Editor

Figures:
File for Subscriber Downloads:

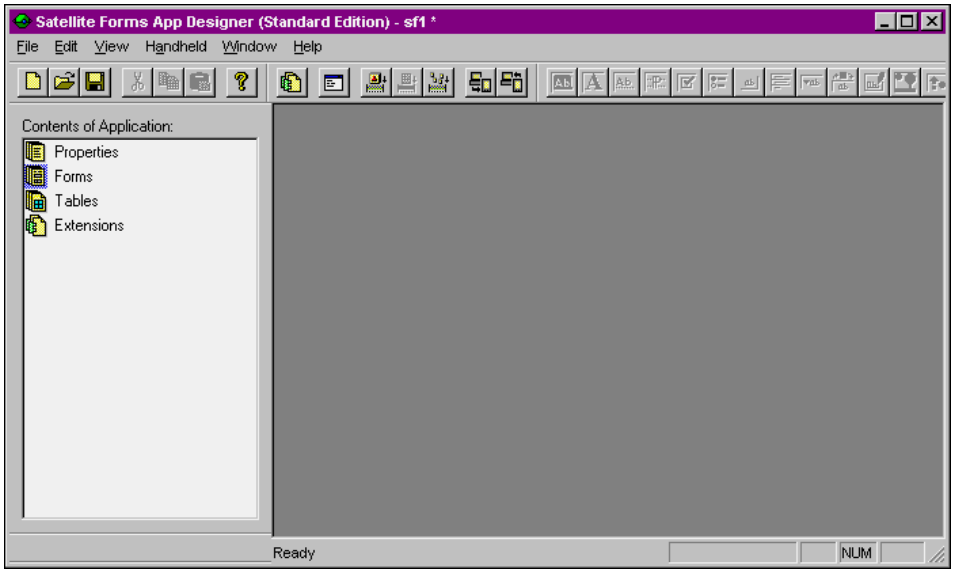
VFP Data on Your Handheld: The SatForms IDE

Whil Hentzen

Two months ago I walked you through the structure of a simple Palm application that you can synchronize with a matching desktop VFP app. This month it's time to get up close and personal with the Satellite Forms IDE – how you build data structures and forms that you deploy on your Palm.

The Satellite Forms development package is used to create applications that run on your Palm handheld. This month, I'm going to show you how to use this rich and flexible environment.

The Satellite Forms IDE, shown in Figure 1, is used for three purposes – to create tables that your Satellite Forms application will access on your PC, to create the actual application (you know, a UI and some business logic) that resides on your handheld, and a mechanism to deploy that application and the related tables to your handheld. There are actually additional functions, like the ability to manage extensions (the Satellite Forms version of ActiveX controls), but we won't get into those here.



PALM-1.BMP
Figure 1. The Satellite Forms IDE before adding tables or forms.

Before you begin, you'll want to set a couple of application-wide properties. Right-click on the Properties node in the Contents of Application tree view and select "Properties" from the resulting context menu, or just double-click on the Properties node. The Application Properties screen will be displayed as shown in Figure 2.

Figure 2. Setting Application-wide properties.

The one property you'll really want to set is the "Name of Application" property – if you don't, your application will be named "Untitled", and that's what will show up on your handheld. You don't have to specify an Initial Form if you only have one form in your app; otherwise, come back to this dialog once you're done with creating the initial form.

The other properties can wait till later.

The SatForms Table Designer

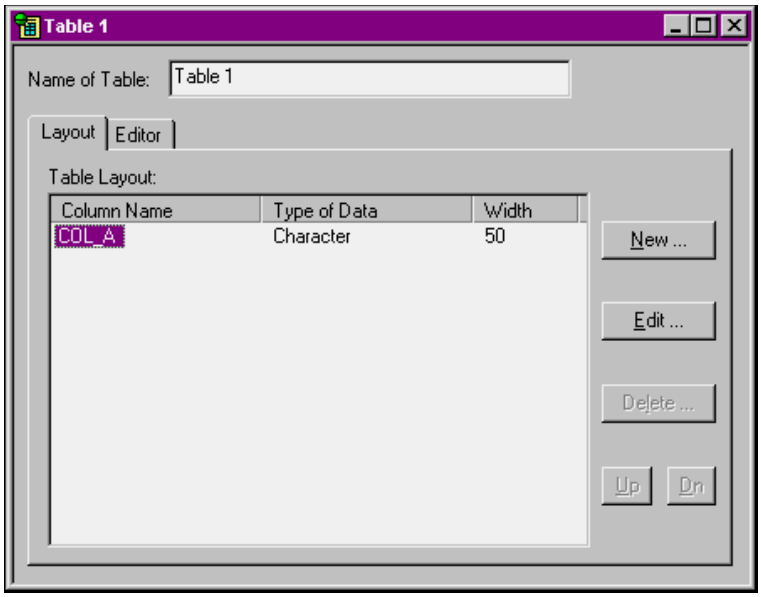
The Table Designer in Satellite Forms is extremely easy to work with, nearly to the point of being trivial. Firing up the tool and playing around for a few minutes is all you need to completely cover its capabilities.

As you may remember, SatForms relies on an intermediate data store on your PC when moving data back and forth between your handheld and your desktop. You use the Table Designer to create this intermediate data store. When you deploy the SatForms application to your handheld, this intermediate data store will be converted to a proprietary ".PDB" field that is loaded onto your handheld.

By the way, this intermediate data store that resides on your PC takes the form of a dBASE V .DBF table – pretty darn convenient, eh?

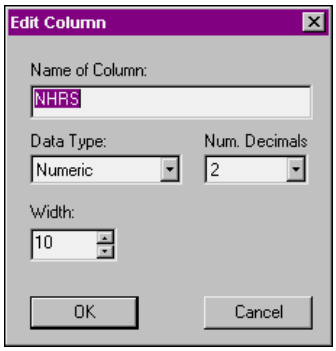
In order to create a table, right click on the Tables node in the Contents of Application tree view in the left side of the SatForms IDE. You'll have three choices in the context menu: Insert Table, Import Table, and then either Show or Hide Tables, depending on whether the Tables node is expanded to show the tables below the node.

If you click on Insert Table, the Table Designer dialog will appear, as shown in Figure 3. The first thing to do, obviously, is name the table. The value you enter in the Name of Table text box will be used both internally by the Satellite Forms engine as well as for the name of the file on disk. Note that you can later change this value – but doing so only changes the value used internally by SatForms – the name of the file on disk stays the same.



PALM-2.BMP
Figure 3. Creating a new table with the SatForms Table Designer.

As you see in Figure 3, there are two tabs in the designer. The Layout tab is used to create and edit the table structure – much like MODIFY STRUCTURE does in Visual FoxPro. You click the New button to create a new field, the Edit button to modify the highlighted field, and the Delete button to remove a row. Up and Dn ("Down") are used to rearrange the order of the fields, which, as all good theorists know, is relationally meaningless.

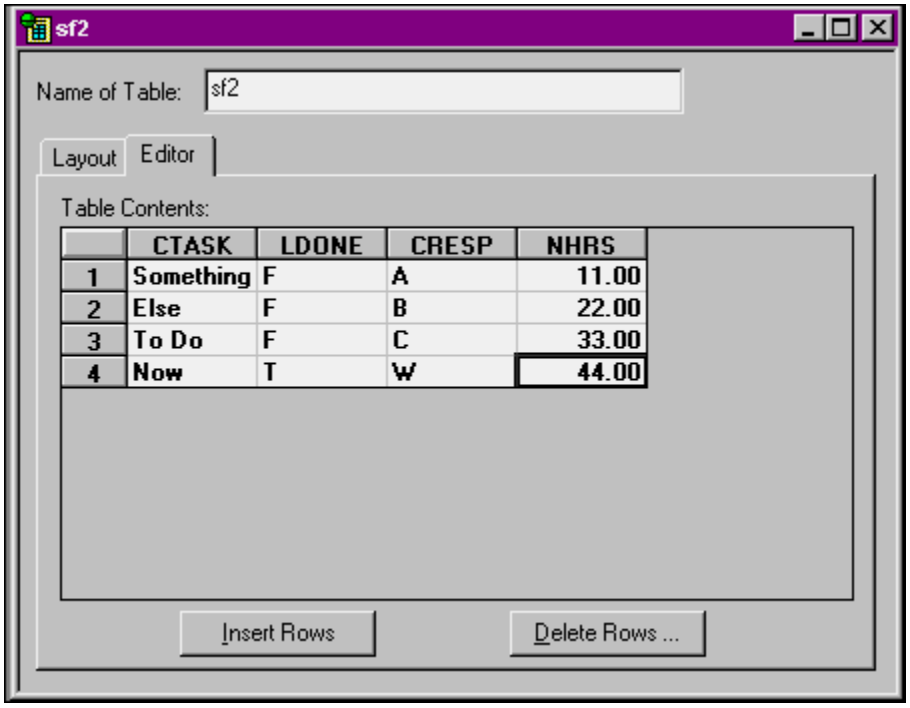


PALM-3.BMP
Figure 4. Use the Edit Column dialog to change attributes of a field definition.

When you click the New or Edit buttons, you get the Edit Column dialog as shown in Figure 4. You can enter the name of the column, select the data type (available choices are Character, Numeric, True/False, Date, Time, Ink, and Time Stamp – and I'll cover "Ink" and "Time Stamp" in future columns), and then choose the width and number of decimals when appropriate. For example, True/False fields are automatically defined as having a width of 1, and you don't get to change it, just like in VFP.

You may be cringing at having to call up a separate dialog to edit a field's definition, but there is a reason. Back in Figure 3, you'll notice that there is no "OK" button in the Table Designer dialog. Yes, closing the dialog saves your changes – thus, instead of confirming your changes at the table level, you do so with each field.

The real excitement starts when you click on the Editor tab of the Table Designer, as shown in Figure 5. Yes, it's a real, honest-to-goodness Browse window for you to enter and edit data!



PALM-4.BMP
Figure 5. You can enter data into a Satellite Forms table via the Editor tab of the Table Designer.

You insert and delete rows via the appropriate command buttons in the bottom of the dialog, and type in data in the appropriate rows and columns as you desire. As soon as you type data in a field, the information is committed to the disk – I've found that you don't even have to leave the field (much less the row) in order to save your changes. This is okay, really, because the Editor tab is supposed to be used for entering sample data for testing – it's not for your users, nor even for you to put large amounts of data into the table.

That's about all there is to creating a table. In this example, I've created a table for a simple To Do list – with fields for the name of the Task, who is Responsible for the task, how long it should take, and a logical flag for whether the task is done or not.

Now let's build a data entry form for this table that we can use on our handheld!

The SatForms Forms Designer

Just like with a table, you can create a new form in the SatForms IDE by right-clicking on the Forms node and selecting the Insert Form menu option in the resulting context menu. Doing so will display an empty form like shown in Figure 6.

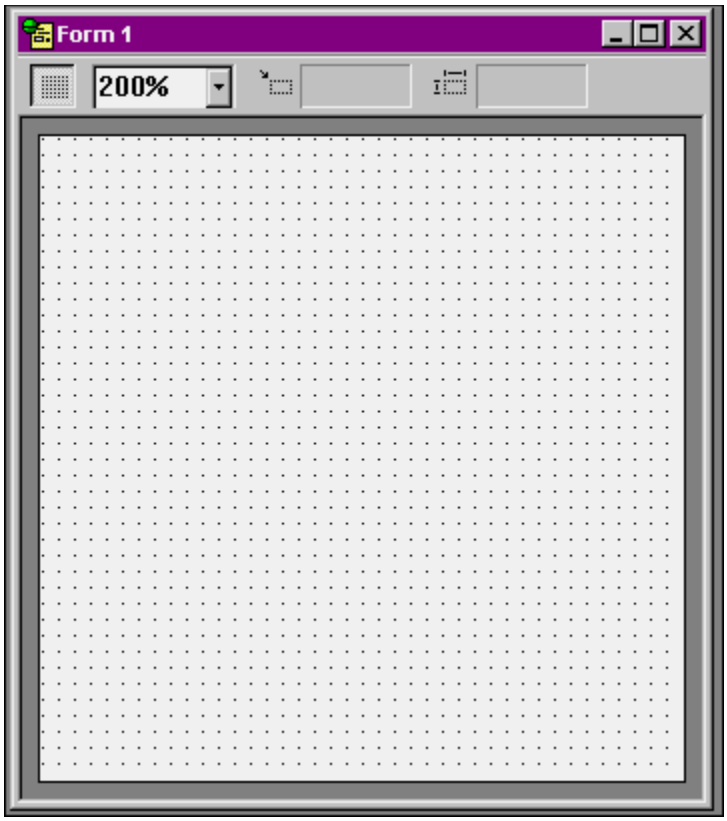


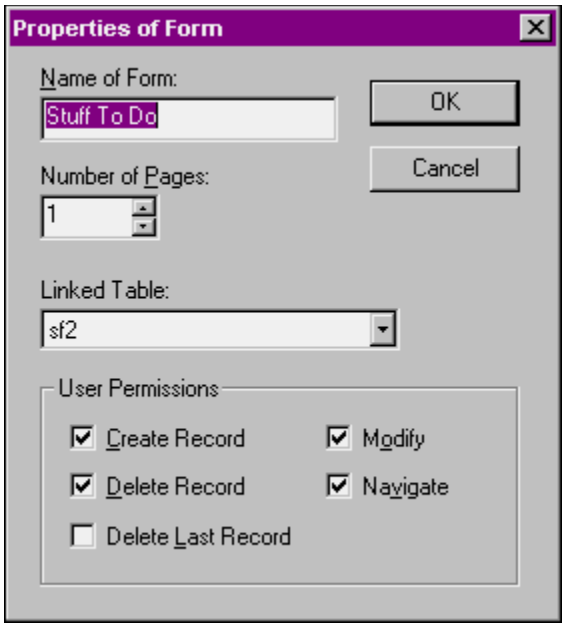
Figure 6. An empty form in the SatForms Form Designer.

The first thing you'll notice is the size of the empty form – it's 160 pixels square. You can't resize this form – by definition, all SatForms forms are this size because it's the size of the Palm "desktop." You don't have to use the entire form, of course, but you can't "tile" windows like you can in GUIs like Windows or the Mac OS.

Manipulating a Form

There's a bunch of stuff you can do with a form even before you start to put controls on it. First of all, there's the toolbar in the Form Designer window. The left most button allows you to turn the 'grid' (the dots on the empty form) on or off. The combo box (that shows 200% in Figure 5) allows you to change the magnification of the form displayed in the IDE. If you have a control selected, as in Figure 7, the next two read-only text boxes show the position of the upper left corner of the selected control, and the width and height, respectively, of the selected control.

Right-clicking in the empty form displays a context menu with two available choices – Form Properties and Scripts. We'll tackle the Scripts option later – it allows you to attach code to various form-level events. Selecting the Form Properties menu option opens the Form Properties dialog as shown in Figure 7.



PALM-6.BMP.
Figure 7. The Satellite Forms Form Properties dialog.

You can name the form – this name is not displayed to the user, but rather is used by the developer – for example, it's what is displayed under the Forms node in the Contents of Application tree view.

SatForms doesn't support a 'tabbed dialog' interface, but their substitute is a multiple page form. If you've used Microsoft Access, you're probably familiar with the concept of "multi-page forms." Think of Leslie Nielsen in The Naked Gun, when he pulls out his police ID and an accordion of credit cards unfolds from his wallet. SatForms multiple page form works sort of like that – you can specify events or write code that moves from page to page, much like you move from tab to tab in a page frame in VFP. You can specify how many pages your form has in this dialog.

The Linked Table combo box performs the same function as the VFP data environment, with the difference that you can only have one primary table linked to a form. (This isn't as big a restriction as it may seem at first. More in a future column.) The Linked Table combo box is populated with all of the tables in the Tables node in the Contents of Application tree view.

The last part of the Form properties dialog allows you to set what they call "User Permissions" – or, in other words, what functions a user is allowed to do. This functionality is extensible – these are defaults for the form. As you can see in Figure 6, you can allow the user to add, delete and modify records, navigate between records, and even delete (or not delete) the last record in the table.

Placing Controls on a Form

Placing controls on a form in SatForms works much like other visual tools you may be familiar with. You have a palette or toolbar of available controls and you drop those controls onto your form. With the SatForms IDE, all you have to do is click on the toolbar button and the corresponding control will be placed on the form.

Note that the control is always placed in the upper left corner of the form – you then need to drag the control to where you want it to reside.

The SatForms Controls Inventory

Many of the native SatForms controls are similar to VFP controls:

Title
Text
Edit

Paragraph
Check box
Radio button
Command button
List box
Droplist (combo box)
Lookup
Ink Control
Bitmap
Graffiti Shift Indicator
Auto Stamp
Custom Control

Most need no explanation but a few don't have direct counterparts to VFP or VB, or aren't immediately obvious.

The Title control is essentially a caption for the form. An example is shown in Figure 8, where its value is set to "Dumb Stuff I Gotta Do."

The Ink Control allows you to capture a signature or other freehand drawing on the Palm – you can think of it as a Palm-specific General control.

The Graffiti Shift Indicator control places an "indicator" graphic on the form that shows the shift status of the Graffiti handwriting recognizer. This graphic will show different symbols depending on whether Graffiti is in lowercase, shifted or caps-lock mode.

The Auto Stamp control automatically enters a date or time stamp in a table column.

The Custom Control allows you to specify a custom control you've built or acquired from another source, much like you'd specify an ActiveX control in Windows.

Controls Properties

You can set properties of every control by right clicking on a control of interest and selecting Control Properties from the context menu, as shown in Figure 8. As with other visual environments, some properties are shared by more than one control, while other properties are unique to a specific control.

All controls have Name of Control, Dimensions and Visible properties. Most controls have Data Source, Font, Read-Only and Don't Modify Table properties. Most controls also allow you to specify an Action When Clicked.

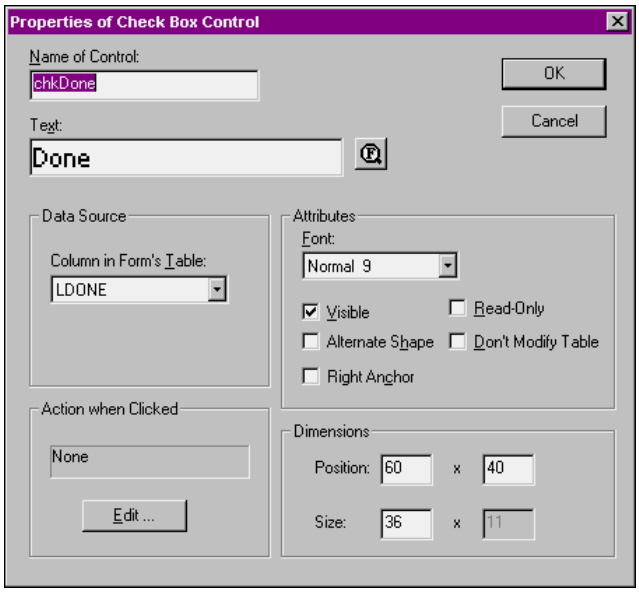


Figure 8. The properties dialog for the Check Box control

The combo box in the Data Source property is populated with the appropriate fields for the table that is linked to this form. For a check box control, only logical fields in the linked table are displayed in the combo box, for text box controls, only character and numeric fields are displayed in the combo.

Command buttons have special properties because they're never attached to data – they're always used to initiate an action. (Well, I suppose you could use one as a fancy label, and just change the caption, but that's not the intended purpose of the control.) See Figure 9 for the properties dialog of the Command Button control.

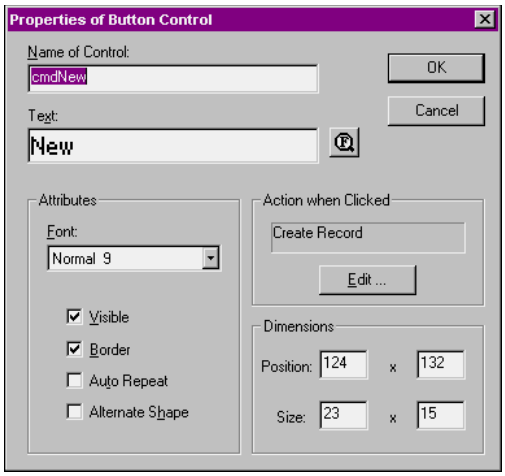


Figure 9. The Button control's property dialog.

As you can see, it has common properties like Name, Text (what you and I think of as "caption"), Font, and Dimensions. There are also some obvious button-specific properties like "Visible" and "Border". "Auto-Repeat" changes the nature of how the control fires its action on the handheld. Normally, a control fires at the end of a tap – when the stylus is lifted from the handheld screen. (That's our "mouse-up" event.) When AutoRepeat is checked, the control's action fires as soon as contact is made – and if you keep the stylus pressed against the screen (lightly, folks, lightly – you're not drilling for oil!), the action will be repeated at a periodic (but undocumented) interval.

The real magic is contained in the Action when Clicked property. Click on the Edit button, and the dialog shown in Figure 10 will appear.

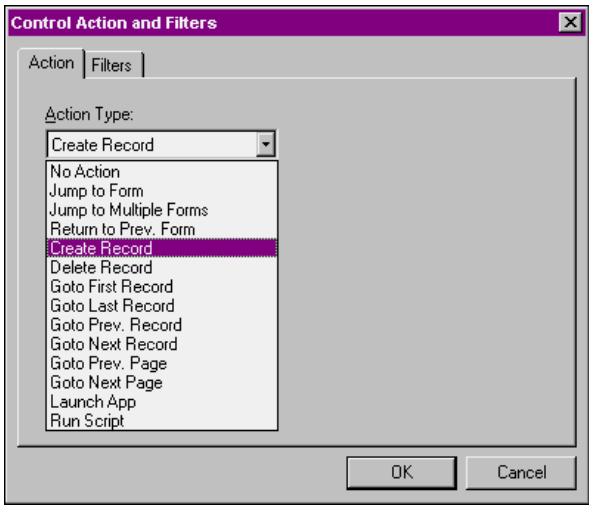


Figure 10. Selecting an Action to attach to a button.

As you can see, there are a wealth – or, as Kelly Bundy would say, a *veritable cornucopia* - of pre-defined actions, including record and form navigation, and record maintenance. Note the last two options in the combo –Run Script allows you to run code that you've written yourself in case one of these options isn't suitable – and Launch App allows you to run another SatForms application.

There's even more power under the hood, though. Selecting the Filters tab, and then clicking the Add button, as shown in Figure 11, allows you to create a specialized type of action that filters a table. This interface is a bit awkward at first – you can think of the Filters tab as being just another type of action – a "Filter Table" action in the Action combo box.

Thus, you could create a button named "Filter" that would set a filter on the table based on criteria you specify in the Create New Filter dialog. Note that you can use either a hard-coded expression or the current value of a control on the form.

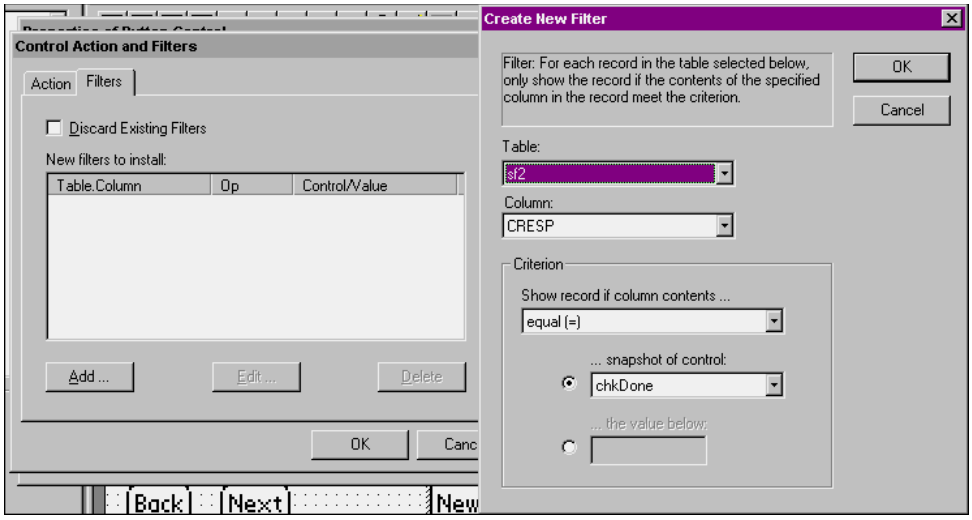


Figure 11. Adding a filter to a button.

The form for the table created above in Figure 4 might look like that shown in Figure 12.

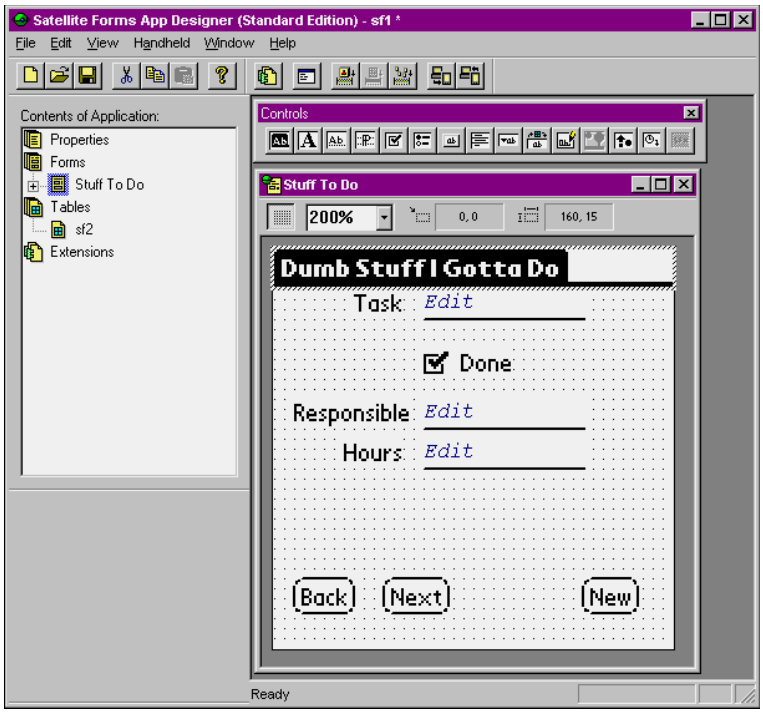


Figure 12. A sample Satellite Forms application form.

Saving your Project

When you create your tables and forms, SatForms will try, like many other brain-dead tools, to save your work in a subdirectory under the directory where you installed Satellite Forms itself – yes, buried deep on drive C. Jeesh. Hasn't anyone at Puma ever had to reformat their C drive?

Obviously, don't accept their defaults – create your own "SatFormsProjects" directory on your development drive, and create separate directories for each SatForms project you create. Once you start saving components for a project there, SatForms will be good about saving subsequent components in the same directory as well.

When you're done with creating your tables and forms and stuff, you'll press Save (if you haven't done it already), and be greeted with a zillion (a zillion is a fraction of a bazillion) dialogs about naming your files. Well, OK, two dialogs.

The first one will ask you to save the SatForms application itself. The resulting file will have an "SFA" extension. You'll next be asked to save your table, and that file will have a "DBF" extension. However, if you look in your directory, you'll see lots more files than just those two. Let's take a look at what each of these are.

First, suppose that you named the project "SFDEMOAPP" and the table "SFDEMO1". You'll see the following files in your project directory:

sfdemo1.dbf
sfdemoapp.pda
sfdemoapp.sfa
sf-se_sfdemoapp.pdb
sft-se_sfdemo1.pdb

The DBF and SFA files are pretty easy to identify, since I just told you what they were. The SFDEMOAPP.PDA file is created by SatForms, and is the file that gets transferred to your handheld. In other words, it's the application that runs on your Palm. The SFT-SE_SFDEMO1.PDB file is the proprietary data file on the handheld where data is saved. I believe that the SF-SE_SFDEMOAPP.PDB file

is associated with the PDA file, in that it contains part of your application – not your data – but the documentation is less than clear about this point.

Deploying your SatForms Application on your Palm

Once you've finished your application, and I recommend you start out really simple, like these articles have, just to get all the steps down, it's time to put it on your handheld and run it there.

First, put your handheld in the cradle, and make sure the connection to your PC is tight. Next, in SatForms, open your project. Select the Handheld|Download App and Tables menu option. The dialog shown in Figure 13 will appear on your PC.

Figure 13. Press the Sync button your handheld cradle when ready to deploy your application and tables from the SatForms IDE to your handheld.

At this point, press the Sync button on the cradle. Both your PC and the handheld will display the usual messages about which files and applications are being synchronized – if you watch closely to the messages on your PC, you'll see Satellite Forms in the list.

Finally, the messages will finish. On your PC, you'll be returned to the Download Application to Handheld dialog in Figure 13. On your handheld, you'll probably see a message about 'Cleaning up' for a few seconds longer. Once that message goes away, the sync is done and you're ready to try out your new handheld application.

Take your handheld out of your cradle, turn it on (if it's not already on), and tap the SatForms icon on the main application screen. You'll see your sample app (along with others that you may have downloaded) in a list. Tap the name of your sample (hopefully it's not called "Untitled'), and the initial form will display. Tap the Next and Back buttons, change some data, add a new record – whatever you like. Open a different application on your Palm, turn your Palm off, and then turn it on and open your sample SatForms app again – the changes you made will still be there. That's all there is to it!

Transferring back to your PC

Finally, let's move those changes back to your PC.

Repeat the steps in the previous section, with the one exception of selecting the Handheld|Upload Tables menu option instead of "Download App and Tables." Once you're done synchronizing, open the table in the SatForms Table Designer, click on the Editor tab, and you'll see the changes you made on your handheld back on your PC. Don't believe me? Take your handheld out of it's cradle, wrap it in tin foil, and put it in a lead-shielded box in the basement – then look at the table on your PC again. Yes, you've really moved your data from the handheld back to your PC!

Next Month

The first time I did this, I experienced the same sort of rush as when I ran a HELLOWLD.EXE of my own making back in the early 80's. However, after the glow wears off, there's one important thing to realize – the synchronization of the DBF on your PC and the data on your handheld isn't intelligent at all. In fact, it's actually just a wholesale replacement of the entire table.

This can cause all sorts of problems. Next month, I'll explain each of problems, and then direct our attention to Visual FoxPro – and show you how to build a VFP application that will handle all of these issues.