*## Column: From the Editor*

*## Figures:*
*## File for Subscriber Downloads:*

# VFP Data on Your Handheld (Reprise)

*Whil Hentzen*

**Back in January I wrote about a new column that was going to talk about getting your VFP down to your handheld and back. Just after that issue hit the streets, Doug Hennig alerted me to Satellite Forms - a tool I hadn't had on my list. A brief investigation convinced me that this was the answer – if only one or two small technical problems could be solved. Well, as with so many things software, those "one or two small technical problems" turned out to take a while to work out, but I've finally got the whole thing working. In this article, I'll discuss the basics of this new tool, and then head on into Extended Article territory for those of you who want to follow along with details and advanced topics.**

Since January, I've seen a number of queries about getting VFP data on your handheld – and back to VFP – on the various electronic forums. Indeed, I've received more than a couple of emails about the topic, asking to be kept up to date when the story breaks. Since then, also, Microsoft has introduced WinCE 3.0, but from my point of view, the reasons I switched to the Palm OS haven't changed, and, indeed, with the emergence of new companies supporting the Palm OS, I think the business case for staying with Palm is stronger than ever.

There are basically three routes you can take when developing applications for your Palm, because, when you want to transfer data between your desktop PC and a handheld, that's really what you're doing – developing an application. The first route is to go quick and dirty, and, in some cases, that's plenty good. If you're simply moving personal data from your PC to your handheld and back, a shareware tool that costs $19 or so is probably the best choice, and I covered three of varying sophistication in January.

The third route is to go hard core – writing in C++ (or, believe it or not, Java). But, as Markus Egger says, "Life is too short to write in C++." Seriously, if you're going to become a full time Palm developer, this is the way to go – there are several development environments, including CodeWarrior for C and the KVM for Java (it's a virtual machine for the Palm, and it's the next step up from Java, thus, the "K" moniker), that are good.

But what about the rest of us? Those of us who need a more robust environment than a flat file with a five or 36 field limitation? This middle ground is addressed by several development environments, including Pendragon Forms (originally known as Pilot Forms, produced by Pendragon Software) and Satellite Forms by PUMA Technologies. Pendragon has a number of limitations that, while much more programmable than the three shareware products I discussed in January, just didn't do much for me

Satellite Forms, on the other hand, is a pretty darn nice development tool for the Palm OS, and it may well be the only thing you ever need to deploy robust handheld applications on your Palm and keep them synced with a desktop or server-based application back at the home office. If you noticed how I slid the word "server-based" into the previous sentence, you might be wondering, "Just exactly what did he mean by that? Could he have meant…" and, yes, I did. Not only can you set up your Satellite Forms app to sync back with your desktop, you can even create a Satellite Forms application that will synchronize directly with a server application like Oracle. Yes, "WOW!"

## An Overview of Satellite Forms

Satellite Forms (SF) is a development tool that allows you to create an application that runs on your Palm – with forms, controls, tables, and such. However, it's much more than just that. The SF development package also includes a couple of tools – an ActiveX control and a DLL – that you use in your VFP (or VB, or Access) application to make your application "HotSync-aware."

This is a critical concept, so I'll explain in more detail. The big win with handheld applications is being able to (1) download some stuff from your PC (or some data source – perhaps a company LAN application, a server-based application, like Oracle, or even a web-based application) to your handheld, (2) manipulate that data on your handheld – modifying existing records or adding new and deleting existing records, and

(3) taking your handheld back to the office and updating the PC database with that information from your handheld. And, most of all, you want all this to happen automatically when you put your Palm into its cradle and hit the magic Sync button. You don't want to have to write a program and run it while your Palm is cradled in order to manually fetch and dump data.

Thus, SF consists of four pieces. The first is an IDE that runs on your PC, as shown in Figure 1.

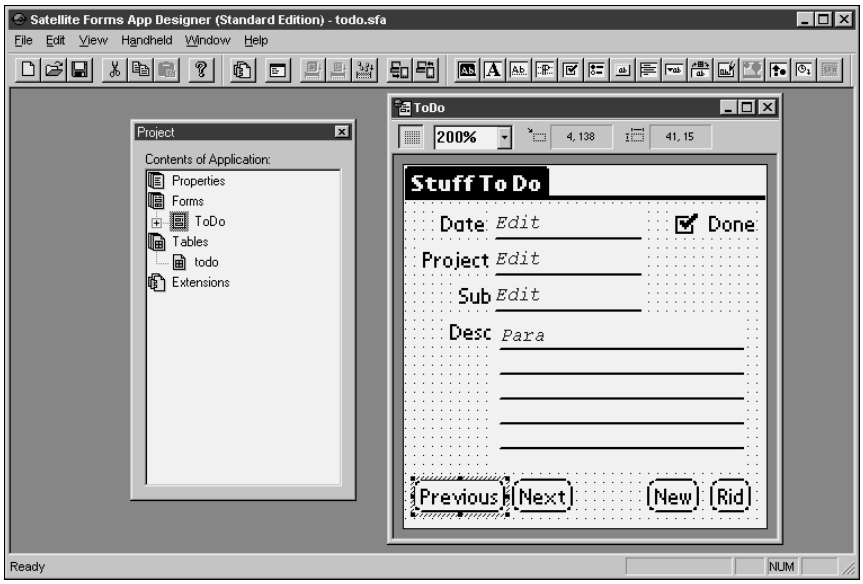


*Figure 1. The Satellite Forms IDE.*

Sure, it looks a bit simplistic, but, hey, think about what you're developing – a monochrome application running on a 33 MHz processor with a meg of RAM and a 160x160 pixel square display. You don't need a lot of bells and whistles.

# The Palm Philosophy

It's probably a good idea to this point. The biggest difficulty I've had (other than deciphering the documentation) is getting used to the mindset of developing for the Palm OS. The whole design concept of the Palm was – and still is – centered around a minimalist philosophy. The Palm is not a replacement for your PC, and they have steadfastly resisted the inclination to keep adding stuff to it.

Thus, when you're developing for the Palm, keep this in mind: What is the absolute minimum you need to have on the Palm? What can you dispense with? Get rid of the "nice to have" mentality – cuz you probably don't have room for it. It's like traveling across country in an MG vs. a Cherokee.

Given this frame of mind, don't think that you can't develop a rich interface and rewarding usability. It's just not going to be on the same scale as the application you're developing for PCs.

The IDE does more than allow you to create forms, though. It's also the mechanism you use to "compile" your application and download the application you developed and the tables that are used by your application to your Palm.

The second piece of SF is an SDK that you load on your Palm in order to "host" SF applications. (You can host multiple SF applications on the same Palm, as you probably expected.) Thus, when you want to run a SF application on your Palm, you tap the SF icon, and then select which SF application you want from a list.

The third piece of SF is an RDK (Redistribution Development Kit) that you can use if you don't want all of your SF apps hosted in the same container. This RDK allows you to create Palm icons for each app and install each of them separately. It's more work than using the SDK, but for some folks, it's more appropriate.

The last piece is an ActiveX control (and a matching DLL for ActiveX-impaired folks). This is the cool part. In order to have VFP exchange data with your handheld automatically, you'll have to have a VFP application running on your desktop before you initialize the HotSync process on your Palm cradle. You

drop this ActiveX control on a form that's part of your application, and it automatically detects the execution of a HotSync process.

You use methods of this ActiveX control to perform actions like copying tables between your handheld and your desktop. You can even automatically update the application on the handheld during a HotSync.

### It's Not Quite That Easy

The documentation that comes with SF is pretty good for what it covers – I'd give it a 9 ½ for most chapters. The one place it falls down is explaining how to use the ActiveX control, which, in my opinion, is the most important piece of the whole thing. It took all of an hour and a half to learn how to write a simple one page form, create a table, connect the form and the table, download the app and the table to my Palm, and run it successfully on the Palm.

On the other hand, it took a couple of weeks to get the ActiveX control to work. It's not that hard – it's just that the doc, the FAQ, the online forums, the archives and the other support mechanisms are, er, bad. The doc is sparse, the relevant files on PUMA website are simplistic to the point of being wrong, and PUMA doesn't seem to have a significant presence on their own message board. There's an OK tutorial that explains how to do this with Access 97, but it misses a couple of key points, and explanation of the Access code is awful.

Once I puzzled it all out, though, it's pretty straightforward, and I'm enjoying working with the product a lot. In future articles, I'll discuss the process of developing an application for your Palm, and then walk you through the use of the ActiveX control with VFP, and show you some advanced techniques for syncing data between your Palm and your VFP desktop application.