Get more productive!

July 2008
Number 3

Dear Reader,

Welcome to the fourth issue of FoxRockX! It is the third regular issue, and two special issues have been published so far. If you wonder why you missed a special issue - it was a German edition and was not mailed to regular subscribers. You can download it for free from our portal with the credentials sent to you via e-mail. We'd like to recommend you to visit our document portal anyway, and have a look at our large archive of articles. Anyhow, here we are again with another issue packed with interesting articles to help you to get more productive!

It is often said that a good programmer can work in any language, although of course they will need a little time (and a good help file) to get familiar with the commands and functions. The basis for this assertion is that the techniques for writing good code are not language-specific, even though the syntax employed most definitely is. Is there really any truth to this?

When I was at school I was told that the reason for learning Latin was (even though, as a spoken language, it was no longer used) that it is the basis for many other languages and that a knowledge of Latin would make learning those other languages much easier. In fact I can still quote the first paragraphs of Commentarii de Bello Gallico ('Commentaries about the Gallic War') by Julius Cesar, but somehow I never found it easy to learn French, Spanish or Italian and my English is, at best, moderate. So my personal experience tends to make me doubt that ability in one language leads, automatically, to ability in others.

Of course spoken languages are very different from programming languages. Or are they? Both have a vocabulary (the set of words that comprise the language), grammar (that defines how words interact) and syntax (that defines the structure of sentences). The problem with programming languages is exactly the same as with spoken languages; while the underlying design principles may be the same, the implementation details are very different. Contrast the traditional C version of the 'hello world' program:

```
main() {
        printf("hello, world");
 }
```

With the equivalent VFP version: `? "hello, world"`

Sure, you may be able figure out, with the help of a C manual how to get a C program to display two words on the screen, but does that mean that you are capable of writing a Sales Order Processing system in C ? I don't think so.

When it comes to writing software there are some elements that are common. The application of logic, and the ability to extract the basic rules of a process into multiple lines of code are examples of things that are not language specific but which a good developer must have. But in order to be a productive developer you need to really know, and understand, the development language in which you are working. One of the best examples-

of the importance of this point comes from Visual FoxPro itself. If you look in the help file for Visual FoxPro (yes, even for Version 9.0) you will find, under the TableUpdate() function an example that includes the following line of code:

```
= TABLEUPDATE(.T.)  && Commits changes.
```

Now as any experienced VFP developer knows, it is imperative to test the return value of TableUpdate() because an update that fails does not generate a VFP Error, the function simply returns „false". This one line of code in the help file has caused more grief to more inexperienced developers than anything else because the help file, is after all, the first place you look when you need to find out how do something in a new language.

Now put yourself in the position of tackling a development project in some language in which you are not experienced. How would you avoid falling into this sort of trap? The short answer is that you can't. The only thing that can help you is experience, and that takes time. Lots of it. My best guess is that for an experienced professional developer to get really competent in a new language takes about two years.

Tools and frameworks can, and do, ease the pain of learning a new language. However there is a risk associated with them. The risk is that the greater the reliance you place on such things the further removed you are from the language itself. The result is that it can actually take longer to reach a given level of expertise and productivity in a new language and it is, at the end of the day, productivity that matters.

The ability of a person, or organization, to be productive in a given environment is consistently cited as one of the key factors in success.

I undertake projects for a subsidiary of one of the larger IT companies here in Germany and my main competitor for work is their own outsourced IT department. From an Executive and Financial perspective it makes sense to outsource IT but at the local, functional, levels it can, and often does, cause problems and delays. This is where the smaller, highly productive development company comes into play.

Smaller companies do not have such high communication and management overheads as larger ones and so can bring specific expertise, timely responses and localized business knowledge to bear on problems that cannot be easily conveyed to developers sitting thousands of miles away in a different time zone. However, the crucial word here is „productive". Productivity in this sense includes the provision of rapid turn-around and accurate, reliable code that solves problems.

So what can you do to improve your productivity? There are two things that are important.

First improve your skill set in your day to day work as a Visual FoxPro developer. This will improve your productivity and enable to add more value to your work. How? The fact that you are reading this shows that you have at least some interest in doing so there are other ways too, including participation in the on-line communities, and going to conferences.

Second, start thinking now about where you want to be in five, or ten years time. This is hard because none of us really knows what the future holds but while VFP development may be your day-to-day work now it is imperative that you continue to learn, and expand your ability to remain productive as new things come along, even though it is tough to do that on top of an already full workload.

And, do me a favor! Please do not forget that actual productivity is the most important issue and should not be mistaken for a kind of professionality that you see more and more frequently, even in small companies or single-person shops. That would-be professionalism shows up when people start over-managing the process of software development with logfiles of what was done by whom in tiny details and creating far too long documents with analysis results. But they neither write proper comments into the code nor create prototypes to show the first design results to the customer.

Quite often something sounds professional though it is not, especially if it decreases the turn-around time of your business but does not provide additional value to your customer. Let's not forget, the more time you spend on other tasks than development, and the more frequently you do this, the more likely the customer will change his mind...