Developer Stew - Being More Productive

Posted At : October 24, 2005 9:34 PM | Posted By : Mark Kruger
Related Categories: Project Management

If you are like me you only have about 6 or 7 hours of really decent programming in you per day. Actually that is probably optimistic. If you are like me you probably have about 4 good hours and 2 or 3 mediocre hours. In fact, for a programmer, productivity might be best defined as a curve with diminishing returns as the day wears on. My best hours for programming are between 7:30 am and around 2:00. After that I'm usually playing catch up with email and other tasks.

The secret to being a *productive* programmer may well be learning how to procrastinate effectively. You have to learn to *put off* email, phone calls, writing and research until your powers of concentration have been weakened by the exerted effort of churning out code. Still, if you have trouble being productive during the day, here are a couple of tips that work for me.

Use Uninterrupted Blocks of Time

Ok - you can't work 4 or 6 hours straight. Can you shut down your email, PDA and Phone for say - 45 minutes? Giving your full attention to something for longer intervals is one of the secrets to getting a project or component to reach the *tipping point*. That's the point where you feel like the project is headed down hill to completion.

Why is using a longer "block" of hours important? Because the first 10 minutes of coding any discreet application usually involves "settling in". That's the process of opening the files you need, connecting to Source management software, databases, ftp sites and the like. It's sort of like starting your car on a cold day. It takes a few miles to get it really warm. Doing short stints that are constantly interrupted is kind of like moving your car a mile or so every hour. It never really warms up.

Along with just the "setup" time of opening programs and connecting to resources, there is also the "mental setup" of remembering where you where last time, which variables hold what values, what various objects do etc. Sometimes this can take quite a while and it's important to get it right.

Use Helpful Tools

Use tools that *speed* your development tasks. Get acquainted with snippets and code reuse. Learn how to map drives, setup projects, search files, and create shortcuts. Learn to type fast and use the keyboard in lieu of the mouse as much as possible (the mouse is a speed killer). Learn the ins and outs of FTP and other low level internet protocols. Know *all the applicatoins* that are running on your workstation and how many resources they consume. Tune your workstation to run like a well-oiled machine. It's surprising how many people bill themselves as "designers" but it seems like they know nothing about their computer or any of its resources.

Take Time Out

Ok - so you've mastered the art of programming for 3 or 4 hours straight. How productive are you in hour 3? If you find yourself spinning your wheels, learn *when to take a break*. Write a blog entry. Get a Starbucks. If you are like my good friend Michael Klostermeyer (who used to be a pro athlete) pull out the gym mat and do some stretching. Give your dog a massage (Ok - so maybe that's going to far). You get the

idea.

Pre - Planning Your Work

My favorite tool is my big low-tech white board hanging on the wall in my office. One of the best things you can do when you are readying for that 2 hour programming stint is take 5 minutes to jot down what you are doing and order it. For example, if you are building a new component, outline the methods and properties that you believe you need before you begin. In fact I always like to add the functions as boiler plate in advance. Order your list by dependency and size (biggest tasks first). Don't fall into the trap of doing all the "easy" stuff first. That's a procrastinator's trick. You will end up with a lot of peripheral tasks done but you will still have the "big stuff" hanging over your head. Instead, do the big item. You will sense the project or component is "mostly done" and it will motivate you to finish.

Use Methodical Troubleshooting

If you are a programmer you are *also a debugger*. Debugging is a learned skill. It requires that you set your ego aside and recognize that you *do introduce bugs* when you program. Don't react like so many programmers do when they find a bug they introduced - scratching their head in exasperation ("..what the...."). Instead, expect bug. Embrace them. Welcome them and learn to take joy in exterminating them. Learn how to do one thing at a time and test it for all possibilities. Don't save things like commenting and validation until later. Make them a part of your development process.

That's my take. If you are a developer and have some tips on productivity please share them with us!!