

By Justin James

With the recent changes in the economy, a lot of developers are focused on their short-term job prospects. At the same time, it's important to make sure that you get the most bang for your buck when it comes to taking the time and energy to learn new skills. Here is our list of 10 skills you should be learning right now to make sure that your resume is relevant for the next five years. The list is hardly exhaustive, and there are huge swaths of the industry it won't cover (mainframe developers, for example). Nonetheless, for average mainstream development, you can't go wrong learning at least seven of these skills—not only to the point where you can talk convincingly about them at a job interview, but actually use them on the job.

1 One of the "Big Three" (.NET, Java, PHP)

Unless there is a radical shift in the development world (akin to an asteroid hitting Redmond), most developers will need to know at least one of the Big Three development systems—.NET (VB.NET or C#), Java, or PHP—for the near future. It's not enough to know the core languages, either. As projects encompass more and more disparate functionality, you'll need to know the associated frameworks and libraries more deeply.

2 Rich Internet Applications (RIAs)

Love it or hate it, in the last few years, Flash is suddenly being used for more than just animations of politicians singing goofy songs. Flash has also sprouted additional functionality in the form of Flex and AIR. Flash's competitors, such as JavaFx and Silverlight, are also upping the ante on features and performance. To make things even more complicated, HTML 5 is incorporating all sorts of RIA functionality, including database connectivity, and putting the formal W3C stamp on AJAX. In the near future, being an RIA pro will be a key resume differentiator.

3 Web development

Web development is not going away anytime soon. Many developers have been content to lay back and ignore the Web or to just stick to "the basics" their framework provides them with. But companies have been demanding more and more who really know how to work with the underlying technology at a "hand code" level. So bone up on JavaScript, CSS, and HTML to succeed over the next five years.

4 Web services

REST or SOAP? JSON or XML? While the choices and the answers depend on the project, it's getting increasingly difficult to be a developer (even one not writing Web applications) without consuming or creating a Web service. Even areas that used to be ODBC, COM, or RPC domains are now being transitioned to Web services of some variety. Developers who can't work with Web services will find themselves relegated to legacy and maintenance roles.

5 Soft skills

One trend that has been going for quite some time is the increasing visibility of IT within and outside the enterprise. Developers are being brought into more and more non-development meetings and processes to provide feedback. For example, the CFO can't change the accounting rules without working with IT to update the systems. And an operations manager can't change a call center process without IT updating the CRM workflow. Likewise, customers often need to work directly with the development teams to make sure that their needs are met. Will every developer need to go to Toastmasters or study *How to Win Friends and Influence People*? No. But the developers who do will be much more valuable to their employers—and highly sought after in the job market.

6 One dynamic and/or functional programming language

Languages like Ruby, Python, F#, and Groovy still aren't quite mainstream— but the ideas in them are. For example, the LINQ system in Microsoft's .NET is a direct descendent of functional programming techniques. Both Ruby and Python are becoming hot in some sectors, thanks to the Rails framework and Silverlight, respectively. Learning one of these languages won't just improve your resume, though; it will expand your horizons. Every top-flight developer I've met recommends learning at least one dynamic or functional programming language to learn new ways of thinking, and from personal experience, I can tell you that it works.

7 Agile methodologies

When Agile first hit mainstream awareness, I was a skeptic, along with many other folks I know. It seemed to be some sort of knee-jerk reaction to tradition, throwing away the controls and standards in favor of anarchy. But as time went on, the ideas behind Agile became both better defined and better expressed. Many shops are either adopting Agile or running proof-of-concept experiments with Agile. While Agile is not the ultimate panacea for project failure, it does indeed have a place on many projects. Developers with a proven track record of understanding and succeeding in Agile environments will be in increasingly high demand over the next few years.

8 Domain knowledge

Hand-in-hand with Agile methodologies, development teams are increasingly being viewed as partners in the definition of projects. This means that developers who understand the problem domain are able to contribute to the project in a highly visible, valuable way. With Agile, a developer who can say, "From here, we can also add this functionality fairly easily, and it will get us a lot of value," or "Gee, that requirement really doesn't match the usage patterns our logs show" will excel. As much as many developers resist the idea of having to know anything about the problem domain at all, it is undeniable that increasing numbers of organizations prefer (if not require) developers to at least understand the basics.

9 Development "hygiene"

A few years ago, many (if not most) shops did not have access to bug tracking systems, version control, and other such tools; it was just the developers and their IDE of choice. But thanks to the development of new, integrated stacks, like the Microsoft Visual Studio Team System, and the explosion in availability of high quality, open source environments, organizations without these tools are becoming much less common. Developers must know more than just how to check code in and out of source control or how to use the VM system to build test environments. They need to have a rigorous habit of hygiene in place to make sure that they are properly coordinating with their teams. "Code cowboys" who store everything on a personal USB drive, don't document which changes correspond to which task item, and so on, are unwelcome in more traditional shops and even more unwelcome in Agile environments, which rely on a tight coordination between team members to operate.

10 Mobile development

The late 1990s saw Web development rise to mainstream acceptance and then begin to marginalize traditional desktop applications in many areas. In 2008, mobile development left the launch pad, and over the next five years, it will become increasingly important. There are, of course, different approaches to mobile development: Web applications designed to work on mobile devices, RIAs aimed at that market, and applications that run directly on the devices. Regardless of which of these paths you choose, adding mobile development to your skill set will ensure that you are in demand for the future.

Additional resources

- TechRepublic's [Downloads RSS Feed](#) **XML**
- Sign up for the [Downloads at TechRepublic](#) newsletter
- Sign up for our [10 Things](#) newsletter
- Check out all of TechRepublic's [free newsletters](#)
- [10+ tips for getting your application into production](#)
- [10 signs that you aren't cut out to be a developer](#)
- [10 signs that your project is about to be cut](#)

Version history

Version: 1.0

Published: April 1, 2009

Tell us what you think

TechRepublic downloads are designed to help you get your job done as painlessly and effectively as possible. Because we're continually looking for ways to improve the usefulness of these tools, we need your feedback. Please take a minute to [drop us a line](#) and tell us how well this download worked for you and offer your suggestions for improvement.

Thanks!

—The TechRepublic Content Team