

## Rapid Web Development

RAPID WEB DEVELOPMENT CAN HELP DEVELOPERS PRODUCE PROOF OF CONCEPT AND DEMO

APPLICATIONS IN A MATTER OF HOURS AND PERMIT RAPID APPLICATION CHANGES BASED ON CUSTOMER FEEDBACK

BY JAMES HOLLINGSHEAD

Once upon a time, having a web page meant slapping together some HTML and uploading it to a server where it could be accessed. It didn't really do much. In fact, most web pages tended to look like electronic versions of information booklets.

These pages were informative and they were easy to make. However, they were boring and not very useful past imparting simple information.

Then people started coming up with the idea to start doing things other than just imparting information on the Internet. They thought it would be great if we could buy things online without having to go to the nearest store or that we might want Internet-based services (like ways to find other pages).

Our quick, simple HTML world got a lot more complicated really quickly. After a lot of trial-and-error, several security fiascoes, and much gnashing of teeth at pages that really didn't do what people thought they should, things started coming together. This has led to the rise of rapid web development.

### WHY SHOULD I USE RAPID WEB DEVELOPMENT?

First and foremost, rapid web development saves you time. This does one of two things for you – if you run a business which creates web sites and web apps for clients, it allows you to take on more jobs since the overall time for every job is reduced. If you have developers working on internal applications like the non-profit I worked at, it lets them take care of their projects more quickly so they can do other useful things, thereby saving you money that you might otherwise have to spend on additional development staff.

Second, rapid web development technologies allow you to easily create maintainable code. This is a wonderful thing if you have a new developer taking over a project since it will take less time for them to understand what the code is doing. It's also nice because it is often quite a long time between when the application is developed and when it is updated.

This can save hours that your developers would otherwise spend in the pursuit of pulling out their hair because they suddenly find themselves looking at something they haven't even had to think about for months. I speak from experience in this. Maintenance is a developer's worst nightmare and anything that can make that part of their life easier will lower their blood pressure a few points.

Third, flawed, buggy software costs your business money. It costs you the time of your developers to fix security flaws and keeps them from working on things that make money for your company. Releasing flawed software can also damage your reputation. Let's face it – your greatest means of getting new business is from current clients. Advertising is a great way to get eyes looking at your company, but if they start looking around and find that a lot of people are unhappy with your services, they will look somewhere else. Rapid web development helps here as well since most of the technologies are designed to be secure from the ground up instead of leaving security as an afterthought or leaving it up to the developers to do all of the security heavy lifting.

Finally, it's really pretty easy to pick up. Ruby, one of the languages discussed below, can be learned in a couple of days and the Rails framework that uses it can be picked up easily as well. You can even port your existing applications pretty easily if you so desire.

Now that all of the reasons to use rapid web development technologies have gotten your attention, let's take a look at the concept that the technologies are based on.

### MODEL VIEW CONTROLLER

At the core of the rapid web development technologies that we'll be discussing in this article is the Model-View-Controller (MVC) concept. While not a new idea (it was first described in 1979), it has found its way into many peoples' vocabulary for the first time.

Basically, what it boils down to is that the software's architecture is broken down into three parts – the Model (data model), the View (user interface), and the Controller (control logic). This allows modifications to be made to one area of the software without impacting the other two a great deal. That means that you can change the way that the end result looks to your customer (user interface) without having to change either the data model or the control logic behind the application. The same is true for changing the business logic (data model) or what happens when you click a button (control logic).

This makes maintenance easier because your developers don't have to look for things that may break in the other two areas if they need to make a change to any part of the program. It also means that the application is more secure because none of the business or control logic is being presented to your customers or anyone who might want to exploit your application (which was a big problem with previous approaches to web development).

While MVC is a nice, and rather nebulous concept, what we're really interested in is the actual implementation of rapid web development and how it can help us get our business done efficiently and help save us money by saving us time. That said, let's take a look at some of the more popular frameworks that let us quickly and safely get our programs out of the concept phase and onto the server where they can be used.

## RUBY ON RAILS

Rails is a rapid development framework based on the Ruby language. Ruby is a language that started out being roughly modeled on Perl, so it's also used to rapidly develop programs that aren't used online.

At the heart of Ruby is the idea that you shouldn't have to configure everything because the normal, everyday things behave exactly the way any sane person would expect them to. This makes development go much faster because your programmers don't have to worry about every tiny detail in order to create the application.

Rails builds on this by giving you the most common pieces of basically any web application that you could care to create. It includes pre-written libraries for things like communicating with databases, data validation from forms, sending and receiving email, formatting date and time information, and

interactive client-side functionality with AJAX. In fact, all of this is set up for your new application with one command. After that, you just start filling in the blanks.

It has a fairly large amount of documentation online and a helpful and supportive community. In fact, the first edition of one book on Ruby, *Programming Ruby* (currently in its second edition), has been made available online by its authors at (<http://www.rubycentral.com/book/>)

While all of the things stated above are great reasons to use Rails, one of the best reasons is the fact that you don't have to recompile the program in order to cause any updates to your web application to take effect. That's right – you can sit there with your client and make changes on the fly while they give input without having to wait for the program to recompile after every change. Anyone who has ever had to go get a cup of coffee and then still had time left over while their program compiles can tell you how much this can decrease the development time.

## PHP SMARTY

Smarty (<http://smarty.php.net/>) is based on PHP, which has been one of the most used languages to create web applications. Like PHP, it has a large number of plugins and has a built-in debugging console. It provides caching for all or just parts of a page and also supports the use of configuration files to keep common values in one location, allowing a change in one place to effect the entire program.

For the presentation portion of the framework, Smarty uses special tags that have a syntax fairly close to normal HTML. While these templates, which closely resemble what the resulting page will look like, contain no PHP themselves, they are compiled into PHP code in order to reduce the time that the server spends parsing the code.

Since PHP has been around for a while, there is quite a lot of documentation on the base language and the documentation for Smarty is available online under the Documents section of the Smarty homepage.

## JAVA STRUTS

Struts (<http://struts.apache.org>) has been around the longest of any of the frameworks that we're covering here. Having been created by the Apache project as a way to tie together things like Java Server Pages, and

servlets, custom tags, and other resources into a unified framework, it supports industry standards so it's compatible with other Java technologies.

Struts is also very mature and has a huge amount of documentation available online (not to mention more paper based books than I care to count). The major downside of Struts that most people point out is the fact that, being based on Java, it tends to be rather verbose, so unless you have a good editor which allows you to autocomplete things like variable names, it takes a while to develop in just because of the amount of typing. On the upside, it is very powerful and has vast amounts of libraries which provide a lot of the functionality that most online programs use.

#### TURBO GEARS

Turbo Gears (<http://www.turbogears.org>) is a relatively new framework based on the Python language that provides a four-tier approach to rapid web development. Being new, there are still a lot of features being added and new documentation being made while they reach a stable release.

Having said that, it looks like an interesting project. SQLAlchemy makes database queries look more like an object oriented programming language (think C++ or Java). CherryPy is used to quickly create dynamic content. Kid provides an XML templating system. Finally, MochiKit is a JavaScript library that allows programmers to work with AJAX capabilities.

#### CONCLUSION

Whether it's based on Ruby or Java, PHP or Python, the use of rapid web development technologies can help decrease the time that you're waiting for your web-based applications to make it to the production server while keeping the problems with maintenance and security to a minimum.

The next time you need to start a new web application, give some thought to how rapid web development technologies can help you go live sooner and be sure to check the other articles in this issue of O3 which deal with Ruby on Rails in a more in-depth fashion.

**James Hollingshead is the Executive Editor for O3 Magazine. James can be reached via email ([james@o3magazine.com](mailto:james@o3magazine.com)).**



**Spliced Networks**

<http://www.splicednetworks.com>