Ruby on Rails: What I Like

Posted At : September 9, 2008 6:18 PM | Posted By : Jon Hartmann
Related Categories: Frameworks

I started a short series of blogs about what I like in a number of modern frameworks/web programming environments. This is part of an effort to find what really *works* in these languages, and to see how best I can port them to ColdFusion. This time around its the king of RAD, Ruby on Rails.

Rails is all the rage these days, so its hard not to compare any framework to the "RoR". Its even harder when lost of frameworks for existing languages are inspired by Ruby on Rails, and try to re-implement things it does in their own language. You know what they say: "Imitation is the sincerest form of flattery." Now its time for me to see what I like.

Ruby on Rails

Code Generation

Obviously, any code that a system can right for you is phenomenal. In the case of Ruby on Rails you can use a combination of meta programming and code generation to "scaffold" whole applications can save a lot of time, especially when dealing with internal systems that don't need any sparkles or flash.

Integrated Form Responses

Rails has an integrated setup for handling "flashes" of information at the top of a page like form validation notifications. This not only helps when working with generated code, but helps to enforce good practices about system responsiveness and element placement.

Convention over Configuration

It can't be stated enough the time that is saved when a system can just do what you want it to do without needing to describe every aspect of the system. With RoR the flexibility is there, but you don't need to worry about it unless you need it.

ActiveRecord

I think that the ORM implmentation in Ruby on Rails has become the gold standard by which other ORMs are measured. To me, thats a good thing; the ActiveRecord system does just about everything that a developer needs to do, and even has a very nice integrated searching ability.

Simple URL Mappings

URLs follow a convention in Ruby on Rails, so its easy to know what goes where. The organization is clean and simple, which a .rhtml file representing the end result of any controller method that requires a view.

Intelligent Layout Support

Ruby on Rails understand layouts that can be applied to views, and can add these on based on convention. It also knows that most apps will use the same layout, so it includes an application layout that is automatically used. Very nice.

Also, just about everything that I said about CakePHPthat I didn't already duplicate here could probably be said about Rails. I think that I'm going to find a number of common themes in what I like about modern frameworks and web programming environments, and unfortunately, I'm not sure how well ColdFusion stacks up on a lot of them.