Ruby on Rails: Start your Engines

Luís Ferreira

September 5, 2012

Contents

Chapter 1

A little bit of History

Rails applications have evolved a whole lot throughout the years so, before even talking about engines let's take a step back and look at that nasty beast that were plugins.

Plugins and Engines are very much alike in terms of the problem then intend to solve, that is when you want to experiment on new stuff without hurting the stable code base or just modularize your code so that it is more easily tested and maintained. They do, however, have some differences in how the solve this.

Prior to rails 3 plugins would live in the `vendor/plugin` directory and they would work fine if what you wanted was to extend `ActiveRecord` or `String` or any other class. On the other if you would like to add an actual model, controller, asset or even migration, brace yourself, your in for a boilerplate nightmare.

Adding a controller would involve creating a generator that would copy it to the correct folder in the app, and the same goes for assets and migrations. This not only is a tedious job as it defeats one of the reasons we are using plugins in the first place, since it merges the plugin code with the application code. What a mess!

As you might imagine these are just some of the more obvious problems with rails 2 plugins and quite frankly I think this is why nobody really used them.

When rails 3 came out the concept of engines came with it, these were plugins but were bundled in gem. They were not quite as spectacular as they are today and I will not give them too much attention, I just want to mention a gem that was written by José Valim, a core Rails contributor, called Enginex. What this gem does is scaffold an engine with all the structure you need, as well as some very helpful boilerplate code.

At the time rails 3.1 was released, Enginex was bundled with together with some other nice tweaks. This was the dawn of the age of the Engines!

Chapter 2

Engines

Here is were we get into the good stuff, creating an engine.

2.1 How to create an engine

2.2 Common patterns and best practices

2.3 Extending the app with metaprogramming

Chapter 3

Testing

How to test an engine

3.1 Setting it up

3.2 RSpec

3.3 Cucumber

Chapter 4

Deploying

How to merge with the app, routes, load seeds, migrations