CHAPTER 1

Introduction

I can still remember the time I first realized what the Spring Framework was and how it could help me. I was tasked with building a web application that will register new businesses with the local government, and being a Java shop this meant the standard set of frameworks at the time: Struts, JavaServer Pages (JSP), and Hibernate. Having built many applications with these technologies, we dove right into development.

When beginning a new application, I always want to improve a few things from the last product development cycle. This time around, it was time to get serious about two things, unit testing and good object-oriented design. Sure, I had written plenty of unit tests before, but I had never begun a project by writing tests first. And although I've been studying and developing with OOP for many years now, I continue to learn new techniques that help the design of the application retain sustainability in the face of change.

So, off we went developing the application, writing tests for the domain model, creating a service layer (a façade for the web layer to integrate with), and beginning the build-out of the Struts layer. Each layer in the system seemed to progress nicely, but that's exactly when we ran into trouble.

As integration between layers began, we noticed that it became harder and harder to write good tests for the system. The application was using the Service Locator pattern to integrate the service layer and the web layers together. This pattern was implemented using a static lookup, which proved impossible to change for our unit tests. The question soon became, "How do we integrate these components such that both writing tests and running in production is simple and efficient?"

Enter the Spring Framework.

More precisely, enter an introduction article about the Spring Framework, posted to TheServerSide (`http://www.theserverside.com`). The original article has since been updated: `http://www.theserverside.com/articles/article.tss?l=SpringFramework`. I still remember printing it out, stapling it together, and sitting back down to my desk to see what all the fuss was about. Could it really help me create easily testable applications? Could it really bring OOP back to web development? There was only one way to find out.

I passed the article off to the boss, and I still remember his Aha moment after reading it. We decided to go for it and use the framework to integrate the components through the new-fangled Dependency Injection. This led to easily testing the components, which led to better code, which led to happier clients. We then replaced our in-house Data Access Object (DAO) framework, one thing led to another, and we had a highly tested, full-blown Spring MVC application.