Introduction to Spring MVC

Developing a Spring Framework MVC application step-by-step

Version 2.5

Overview

This document is a step-by-step guide on how to develop a web application from scratch using the Spring Framework.

Only a cursory knowledge of Spring itself is assumed, and as such this tutorial is ideal if you are learning or investigating Spring. Hopefully by the time you have worked your way through the tutorial material you will see how the constituent parts of the Spring Framework, namely Inversion of Control (IoC), Aspect-Oriented Programming (AOP), and the various Spring service libraries (such as the JDBC library) all fit together in the context of a Spring MVC web application.

Spring provides several options for configuring your application. The most popular one is using XML files. This is also the traditional way that has been supported from the first release of Spring. With the introduction of Annotations in Java 5, we now have an alternate way of configuring our Spring applications. The new Spring 2.5 release introduces extensive support for using Annotations to configure a web application.

This document uses the traditional XML style for configuration. We are working on an *"Annotation Edition"* of this document and hope to publish it in the near future.

Please note that we are not going to cover any background information or theory in this tutorial; there are plenty of books available that cover the areas in depth; whenever a new class or feature is used in the tutorial, forward pointers to the relevant section(s) of the Spring reference documentation are provided where the class or feature is covered in depth.

1. What's covered

The following list details all of the various parts of the Spring Framework that are covered over the course of the tutorial.

- Inversion of Control (IoC)
- The Spring Web MVC framework
- Data access with JDBC
- Unit and integration testing
- Transaction management

2. Prerequisite software

The following prerequisite software and environment setup is assumed. You should also be reasonably comfortable using the following technologies.

- Java SDK 1.5
- Ant 1.7
- Apache Tomcat 6.0.14

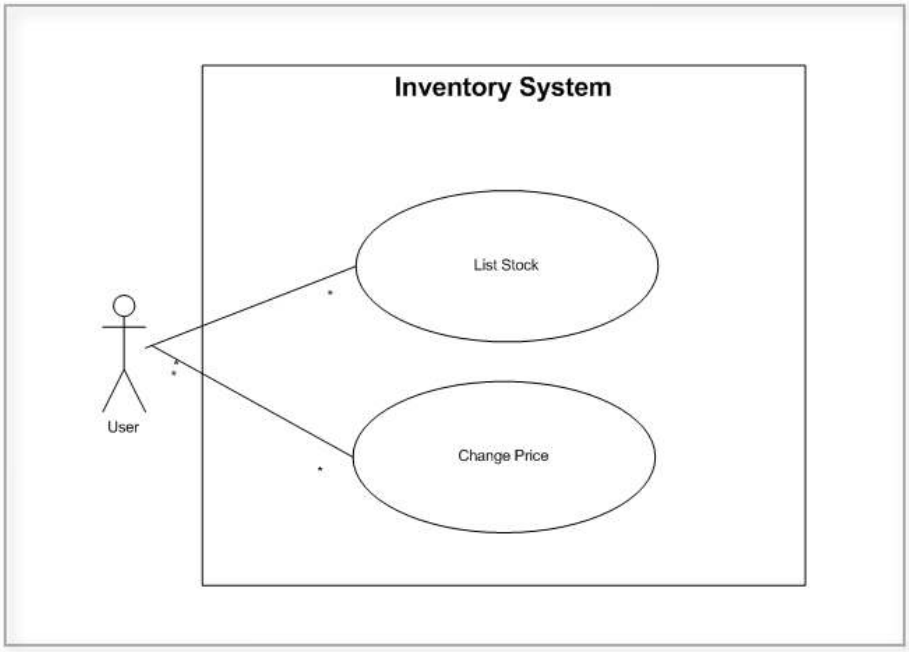
- Eclipse 3.3 (Recommended, but not necessary)

Eclipse 3.3 Europa (http://www.eclipse.org/europa) with the Web Tools Platform (WTP) Project (http://www.eclipse.org/webtools) and the Spring IDE Project (http://www.springide.org) provides an excellent envirnment for web development.

You may of course use pretty much any variation or version of the above software. If you want to use NetBeans or IntelliJ instead of Eclipse or Jetty instead of Tomcat, then many of the tutorial steps will not translate directly to your environment but you should be able to follow along anyway.

3. The application we are building

The application we will be building from scratch over the course of this tutorial is a *very basic* inventory management system. This inventory management system is severely constrained in terms of scope; find below a use case diagram illustrating the simple use cases that we will be implementing. The reason why the application is so constrained is so that you can concentrate on the specifics of Spring Web MVC and Spring, and not the finer details of inventory management.



Use case diagram of the inventory management system

We will start by setting up the basic project directory structure for our application, downloading the required libraries, setting up our Ant build scripts, etc. The first step gives us a solid foundation on which to develop the application proper in parts 2, 3, and 4.

Once the basic setup is out of the way, Spring itself will be introduced, starting with the Spring Web MVC framework. We will use Spring Web MVC to display the inventoried stock, which will involve writing some simple Java classes and some JSPs. We will then move onto introducing persistent data access into our application, using Spring's Simple JDBC support.

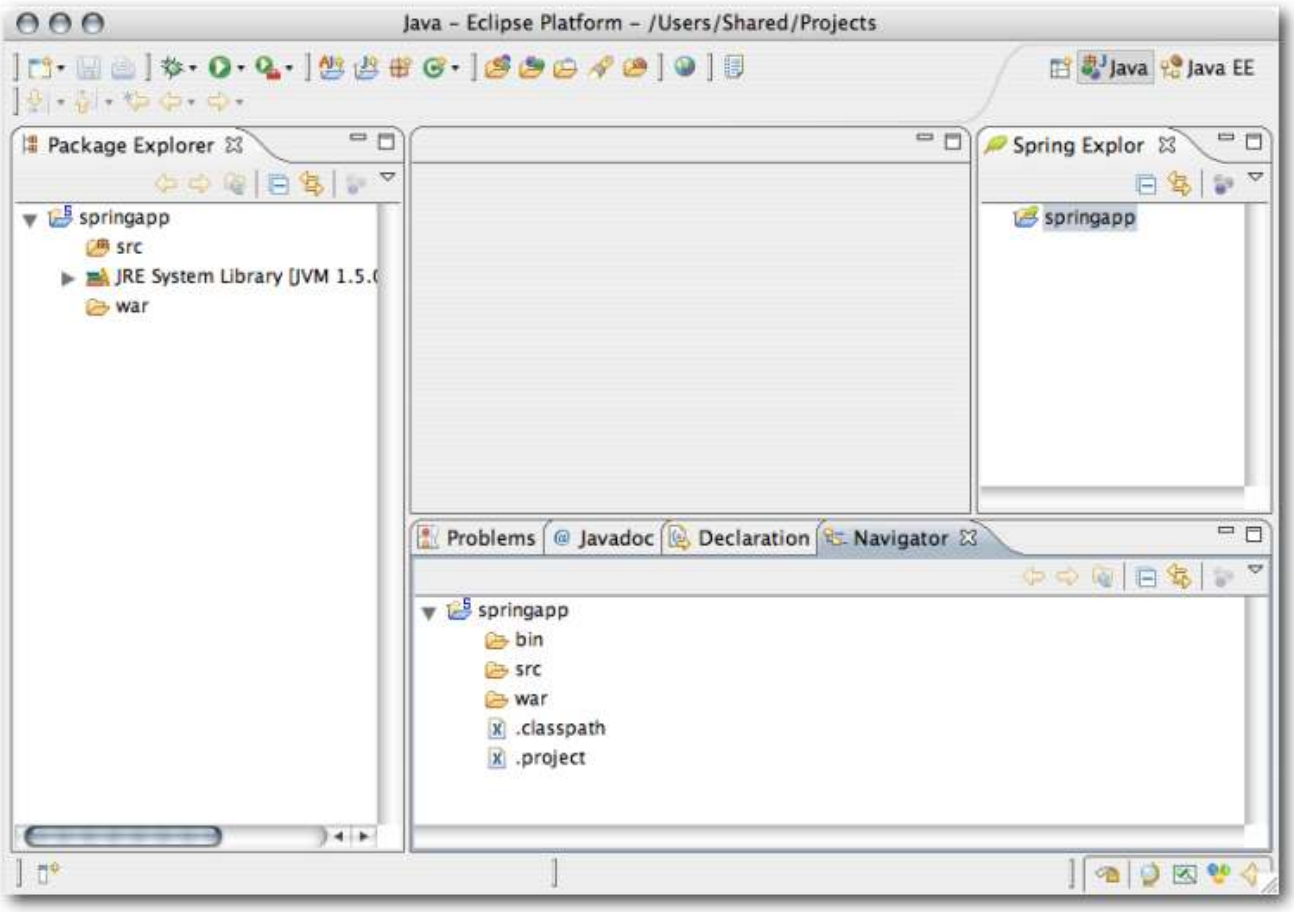
By the time we have finished all of the steps in the tutorial, we will have an application that does basic inventory management, including listing stock and permitting the price increase of such stock.

Chapter 1. Basic Application and Environment Setup

1.1. Create the project directory structure

We are going to need a place to keep all the source and other files we will be creating, so let's create a directory named `'springapp'`. The decision as to where you create this directory is totally up to you; we created ours in a `'Projects'` directory that we already had in our `'home'` directory so the complete path to our project directory is now `'$HOME/Projects/springapp'`. Inside this directory we create a sub-directory named `'src'` to hold all the Java source files that we are going to create. Then we create another sub-directory that we name `'war'`. This directory will hold everything that should go into the WAR file that we will use to package and deploy our application. All source files other than Java source, like JSPs and configuration files, belong in the `'war'` directory.

Find below a screen shot of what your project directory structure must look like after following the above instructions. *(The screen shot shows the project directory structure inside the Eclipse IDE: you do not need to use the Eclipse IDE to complete this tutorial successfully, but using Eclipse will make it much easier to follow along.)*



The project directory structure

1.2. Create `'index.jsp'`

Since we are creating a web application, let's start by creating a very simple JSP page named `'index.jsp'` in