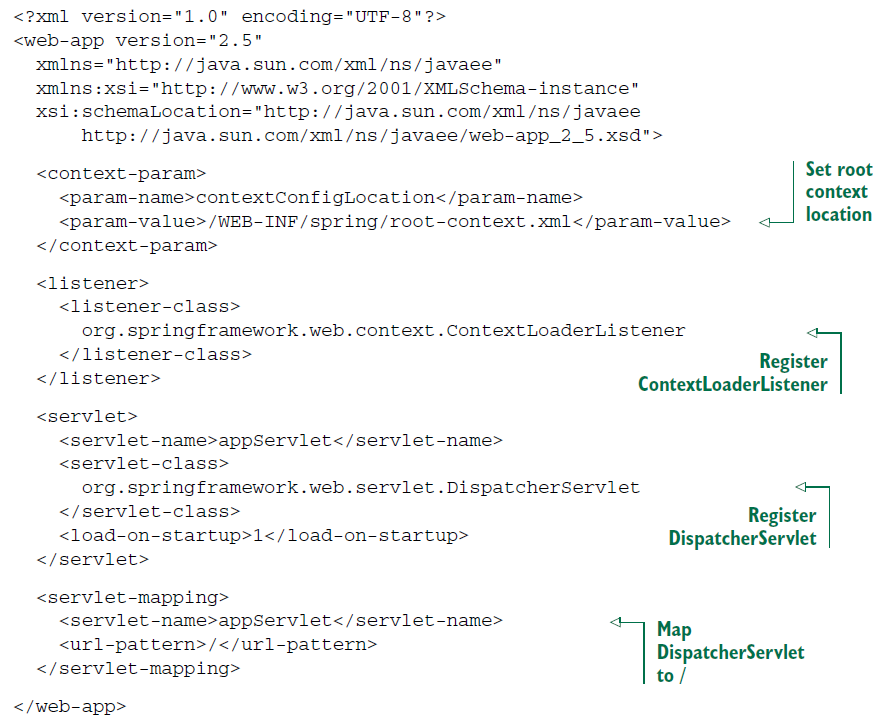
***Declaring DispatcherServlet in web.xml***

* Here’s a basic web.xml file with a typical setup for DispatcherServlet and ContextLoaderListener.



* ContextLoaderListener and DispatcherServlet each load a Spring application context. The contextConfigLocation context parameter specifies the location of the XML file that defines the root application context loaded by ContextLoaderListener. The root context is loaded with bean definitions in /WEB-INF/spring/root-context.xml.
* DispatcherServlet loads its application context with beans defined in a file whose name is based on the servlet name. The servlet is named appServlet. Therefore, DispatcherServlet loads its application context from an XML file at /WEB-INF/**appServlet-context.xml**.
* If you’d rather specify the location of the DispatcherServlet configuration file, you can set a contextConfigLocation initialization parameter on the servlet. For example, the following DispatcherServlet configuration has DispatcherServlet loading its beans from /WEB-INF/spring/appServlet/servlet-context.xml:



* We’ll favor Java configuration over XML configuration. Therefore you’ll need to set up Spring MVC to load the configuration from @Configuration-annotated classes.
* To use Java-based configuration in Spring MVC, you need to tell DispatcherServlet and ContextLoaderListener to use AnnotationConfigWebApplicationContext, an implementation of WebApplicationContext that loads Java configuration classes instead of XML. You can do that by setting the contextClass context parameter and initialization parameter for DispatcherServlet. The next listing shows a new web.xml file that sets up Spring MVC for Java-based Spring configuration.



