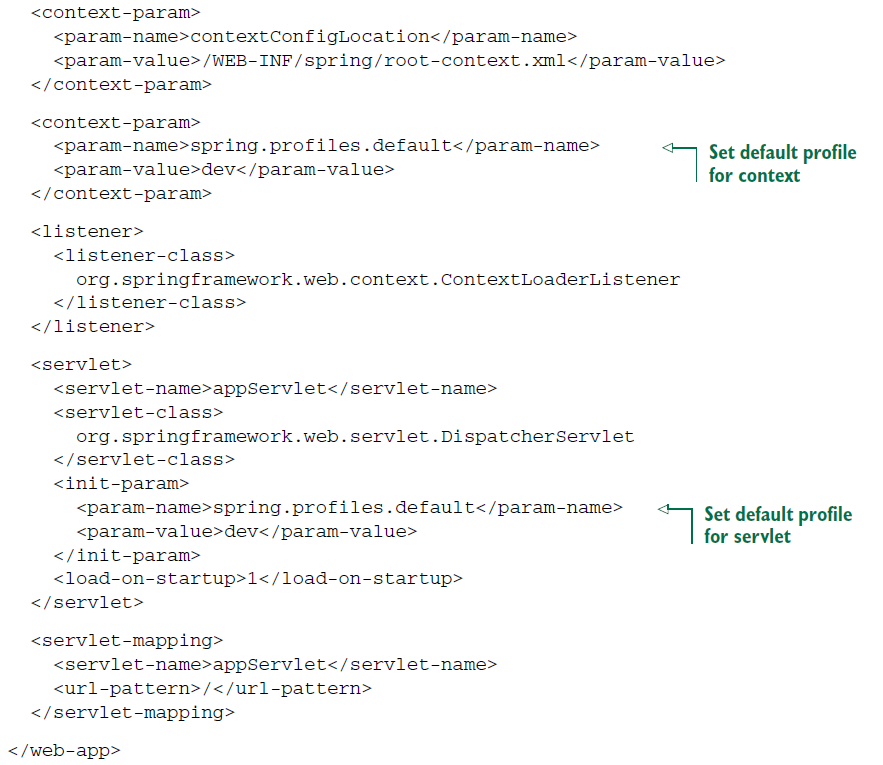
***Activating profiles***

* Spring honors two separate properties when determining which profile are active:
* *spring.profiles.active*
* *spring.profiles.default*
* If *spring.profiles.active* is set, then its value determines which profiles are active. But if *spring.profiles.active* isn’t set, then Spring looks to *spring.profiles.default.*
* If neither *spring.profiles.active* nor *spring.profiles.default* is set, then there are no active profiles, and only those beans that aren’t define as being in profile are created.
* There are several ways to set these properties:
* As initialization parameters on *DispatcherServlet*
* As context parameters of a web application
* As JNDI entries
* As environment variable
* As JVM system properties
* Using @*ActiveProfiles* annotation on an integration test class.

****

* With *spring.profiles.default* set this way, any developer can retrieve the application code from source control and run it using development settings (such as an embedded database) without any additional configuration.
* Then, when the application for deploying in a QA, production, or other environment, the person responsible for deploying it can set *spring.profiles.active* using system properties, environment variables, or JNDI as appropriate. When *spring.profiles.actives* is set, it doesn’t matter what *spring.profiles.default* is set to; the profiles set in *spring.profiles.active* take precedence.
* You’ve probably noticed that the word *profiles* is plural in *spring.profiles.active* and *spring.profiles.default.* This means you can activate multiple profiles at the same time by listing the profile names, separated by commas.