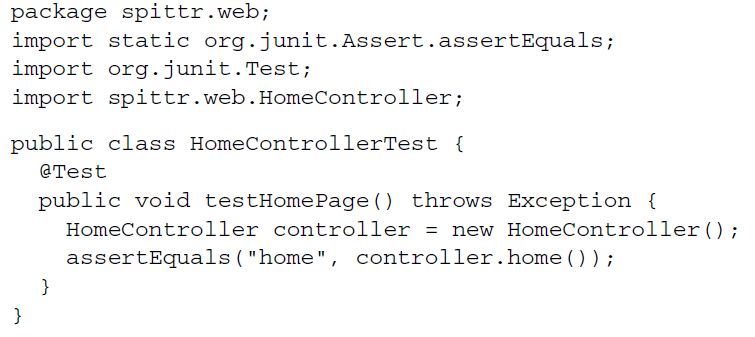
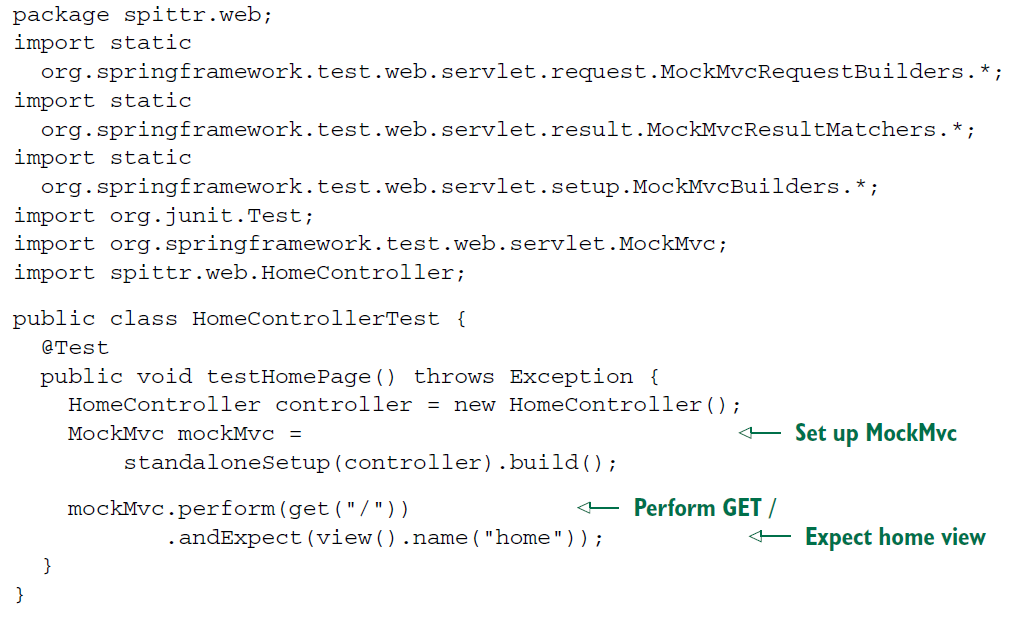
***Testing the controller***

* It’s easy to test POJOs. Therefore, you can test HomeController by writing a simple test like the following.



* Although the test is straightforward, it only tests what happens in the home() method. It calls home() directly and asserts that a String containing the value “home” is returned. It completely fails to test what makes that method a Spring MVC controller method. Nothing about the test asserts that home() will be called when a GET request for / comes in. And just because it returns “home”, there’s nothing to truly test that home is the name of the view.
* Starting with Spring 3.2, however, you have a way to test Spring MVC controllers as controllers, not merely as POJOs. Spring now includes a mechanism for mocking all the mechanics of Spring MVC and executing HTTP requests against controllers. This will enable you to test your controllers without firing up a web server or web browser.
* To demonstrate proper testing of a Spring MVC controller, you can rewrite HomeControllerTest to take advantage of the new Spring MVC testing features. The following listing shows the new HomeControllerTest:



* Even though this new version of the test is a few lines longer than its predecessor, it more completely tests HomeController. Rather than call home() directly and test its return value, this test issues a GET request for / and asserts that the resulting view is named home. It starts by passing an instance of HomeController to MockMvcBuilders.standaloneSetup() and calling build() to set up the MockMvc instance. Then it asks the MockMvc instance to perform a GET request for / and sets an expectation for the view name.