***Summary***

When it comes to Spring, there’s always more: more features, more choices, and more ways to achieve your development goals. Spring MVC has a lot of capabilities and many tricks up its sleeves. Spring MVC setup is certainly one area where you have a lot of choices. In this chapter, we started by looking at various ways to set up Spring MVC’s DispatcherServlet and ContextLoaderListener. You saw how to tweak DispatcherServlet’s registration and how to register additional servlets and filters. And, in case you’re

deploying your application to an older application server, we took a quick look at how to declare DispatcherServlet and ContextLoaderListener in web.xml. Then we took a look at how to handle exceptions thrown from Spring MVC controllers. Although an @RequestMapping method could handle exceptions itself, your controller code is much cleaner when you extract the exception handling into a separate

method. To consistently handle common tasks, including exception handling, across all controllers in your application, Spring 3.2 introduced @ControllerAdvice to create classes that collect common controller behavior in one place. Finally, we looked at how to carry data across redirects, including Spring’s support

for flash attributes: model-like attributes that will survive a redirect. This enables you to properly respond to POST requests with a redirect, but to still carry model data obtained while handling a POST request and use it or display it after the redirect.