6. Handling Exceptions in Servlets

Contents

[1. Introduction 1](#_Toc24326)

[2. web.xml Configuration 1](#_Toc25993)

[3. Request Attributes - Exceptions 1](#_Toc31432)

[4. Demo: Error Handler Servlet 1](#_Toc6693)

[5. Summary 2](#_Toc21106)

# 1. Introduction

=>slides: Pg. 1

In this module, we shall understand one of the important features to be known in Servlet, that is, handling exceptions in Servlets. Whenever a Servlet throws an exception, the web container searches the configurations in web. xml that use the exception type element for a match with the thrown exception type. We need to use error-page elements in web. xml to specify the invocation of Servlet in response to certain exceptions or HTTP status code. Let us understand how to configure web. xml for handling the exceptions in the next section.

# web.xml Configuration

=>slides: Pg. 2

In the previous section, I informed that whenever a Servlet throws an exception, the web container searches the configurations in web. xml that use the exception type element for a match with the thrown exception type. Let us understand how to configure the web. xml file.

=>slides: Pg. 3

Whenever we need to configure web. xml file for handling exceptions, then we need to use an error-page tag. And if you want to configure based on error code, then with that error-page tag, we need to use a tag with the name error-code. And then we need to specify the error-code, for example, 404. And then close the error-code tag. Once we've provided the error-code, then we need to specify the location of the Servlet to be used for handling the exception using a location tag where we need to use the url-pattern of the Servlet. If we need to configure the error pages based on exception-types, then also we need to use error-page. But instead of using error-code tag, we need to use exception-type tag. I have specified the exception-type, for example, javax. servlet. ServletException. And then close the exception-type tag. And then to handle that exception, we need to specify the location tag with a url-pattern of the matching Servlet to handle the exception.

=>slides: Pg. 4

If we want to have a generic error handler for all the exceptions, then we should define the following error-page instead of defining separate error-page elements for every exception, that is, error-page, exception-type, java. lang, Throwable, close the exception-type, location, url-pattern of the matching Servlet for handling the exception, and close the location tag.

=>slides: Pg. 5

We need to remember two points while configuring the web. xml. The first one, a Servlet which handles the exception will be defined in the same usual way as any other Servlet, and it will be configured within the web. xml file. And the second, a single Servlet error handler page can be used to handle any number of error-code or exception-types. Once we have an idea on the configuration details required for the web. xml file, then next we need to understand what are the details that an error handling Servlet can access. So let us understand about the request attributes related to the exceptions in the next section.

# Request Attributes - Exceptions

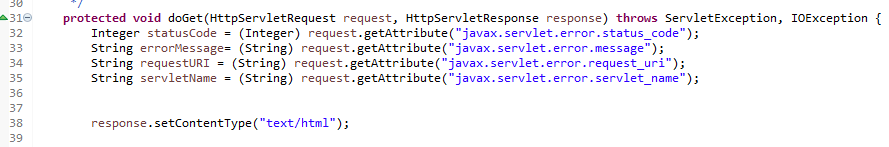
=>slides: Pg. 6

Whenever we create a Servlet page for handling the exceptions, then we require the details about the exceptions that are raised to analyze the nature of an error or exception. We have been provided with a list of request attributes that an error handling Servlet can access to get the details about an exception. Javax. servlet. error. status\_code--this attribute gives us a status code and returns the value in integer data type. Javax. servlet. error. exception\_type--this attribute gives information about exception type and returns java. lang. class data type. Javax. servlet. error. message--this attribute gives information about the exact error message and returns a string value. Javax. servlet. error. request\_uri--this attribute gives information about a URL calling the Servlet, and it returns a string data type. Javax. servlet. error. exception--this attribute gives information about the exception raised. Javax. servlet. error. servlet\_name--this attribute gives a Servlet name and returns the value in a string data type.

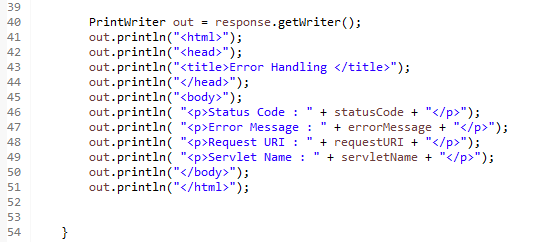
# Demo: Error Handler Servlet

=>slides: Pg. 7

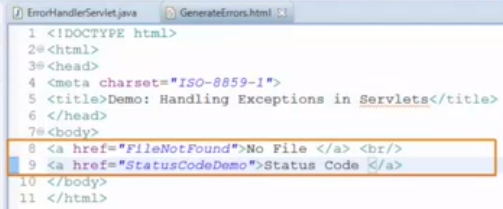
Now   let   us   understand   how   to   handle exception   in   servlets   practically. 

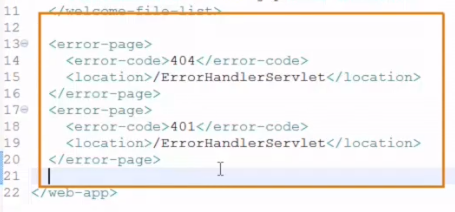
I   already   added   the   servlet   page   where   I provided   the   package   named   com. pluralsight. demos   and   the   class   name   as   ErrorHandlerServlet. 

Now   let   me   add   the   code   to   extract   the values   of   the   statusCode,   errorMessage, requestURI,   and   the servletName, which   is   causing   the   error   using   the request   attributes of   exception.

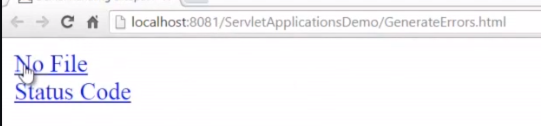


Now   let   me   add   the   code   to   set   the   response content   type   and   add   a   couple   of   out   statements to   generate   the basic   HTML   page. Now   let   me   add   some   more   out   statements   to display   a   heading   as   an   Error   Page   and   also to display   the   Status   Code, Error   Message, Request   URI,   and   the   Servlet   Name   within   the   page. Now   let   me   add   an   HTML   page   to   generate   the   errors. So   let   me   right­click   on   the   WebContent folder   and   click on   New,   HTML,

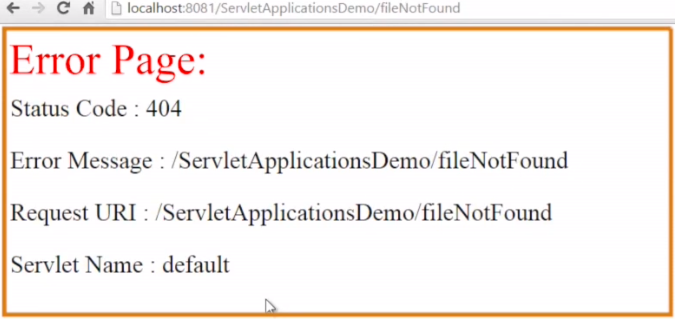


and   let   me   provide   the   name   as   GenerateErrors. html. Now   let   me   update   the   title   of   the   page   as Demo:     Handling   Exceptions   in   Servlets. Within   the   body   tag,   let   me   add   two   hyperlinks. The   first   link   refers   to   a   servlet, which   is   not   existing, and   the   second   hyperlink   refers   to   the   StatusCodeDemo, which   we   have   observed   in   the   previous   code   as   a demonstration   for   understanding   the   HTTP   status   code. Quite   a   simple   HTML   page. Next, the   most   important   step   that   is   we need   to   configure   the   web. xml   file   for   handling   the   exceptions. So   let   me   open   that   web. xml   file   present   at   the   WEB­INF   folder. 

Let   me   add   the   entry   for   error   pages   within   the   file. I   have   configured   the   web. xml   file   to   support   404   and 401   error   codes. Now   let   me   execute   the   GenerateErrors. html.



Let   me   click   on   the   first   link.



We   can   observe   the   custom   error   page, which   we   have   created   as   displayed.

# Summary

=>slides: Pg. 8

In this module, we have understood how to handle exceptions in Servlets by configuring the web. xml file and generating ErrorHandler Servlet page. The next module can be considered as one of the most important modules that we need to know to work with Servlet programming efficiently where I will be discussing about the various methods of state management, which helps in tracking the user data across the pages.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*