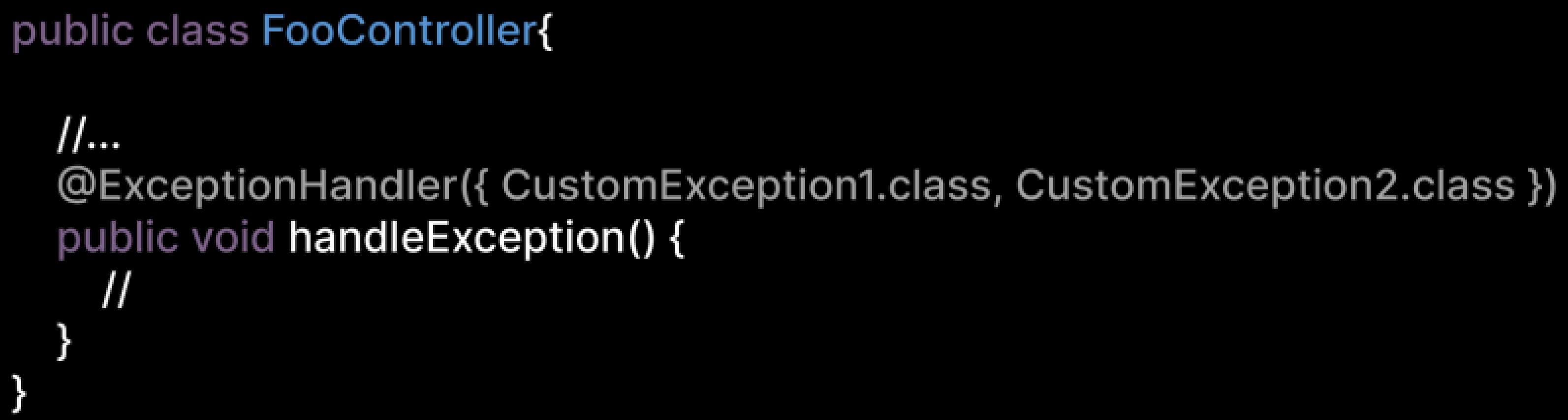
@ExceptionHandler

Spring @ExceptionHandler annotation

@ExceptionHandler annotation is used for handling exceptions in specific handler classes and/or handler methods.

To handle exceptions in String MVC, we can define a method in controller class and use the annotation @ExceptionHandler on it. Spring configuration will detect this annotation and register the method as exception handler for argument exception class and its subclasses.



```
public class FooController{

    //...
    @ExceptionHandler({ CustomException1.class, CustomException2.class })
    public void handleException() {
        //
    }
}
```

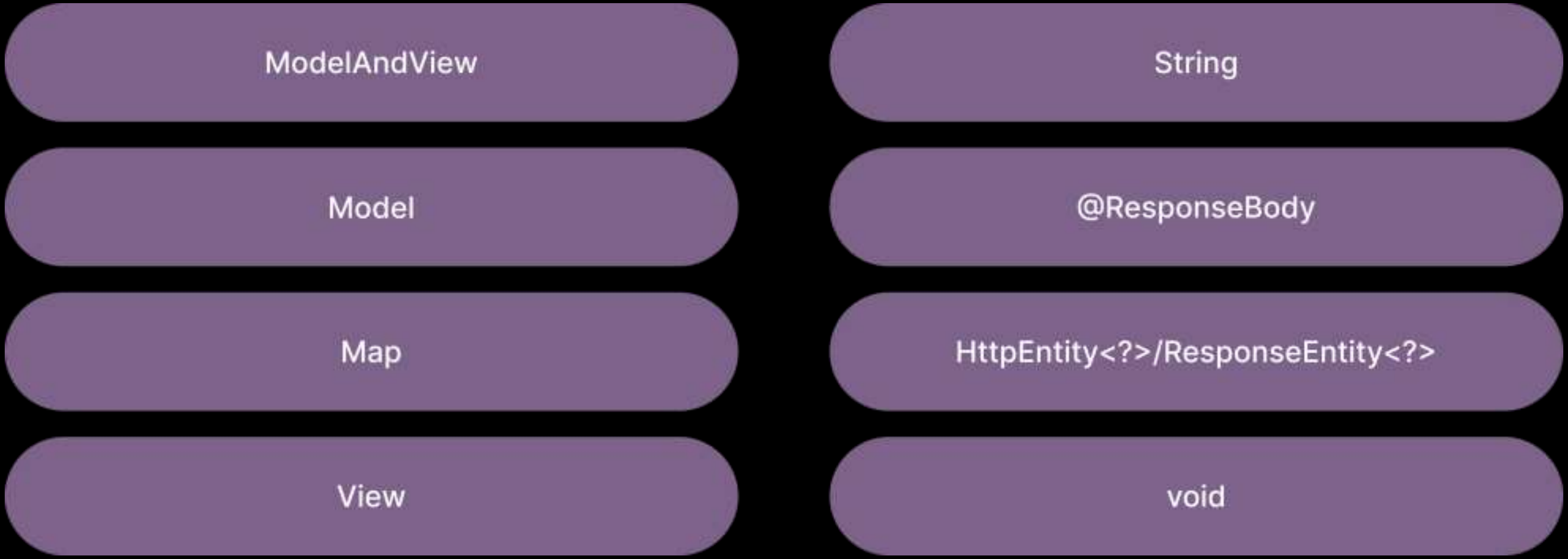
Exception handler method arguments

Handler methods which are annotated with this annotation are allowed to have very flexible signatures. They may have parameters of the following types, in arbitrary order:

- An exception argument: declared as a general Exception or as a more specific exception
- Request and/or response objects (typically from the Servlet API)
- Session object: typically HttpSession
- WebRequest or NativeWebRequest
- Locale for the current request locale
- InputStream / Reader for access to the request's content
- OutputStream / Writer for generating the response's content
- Model as an alternative to returning a model map from the handler method

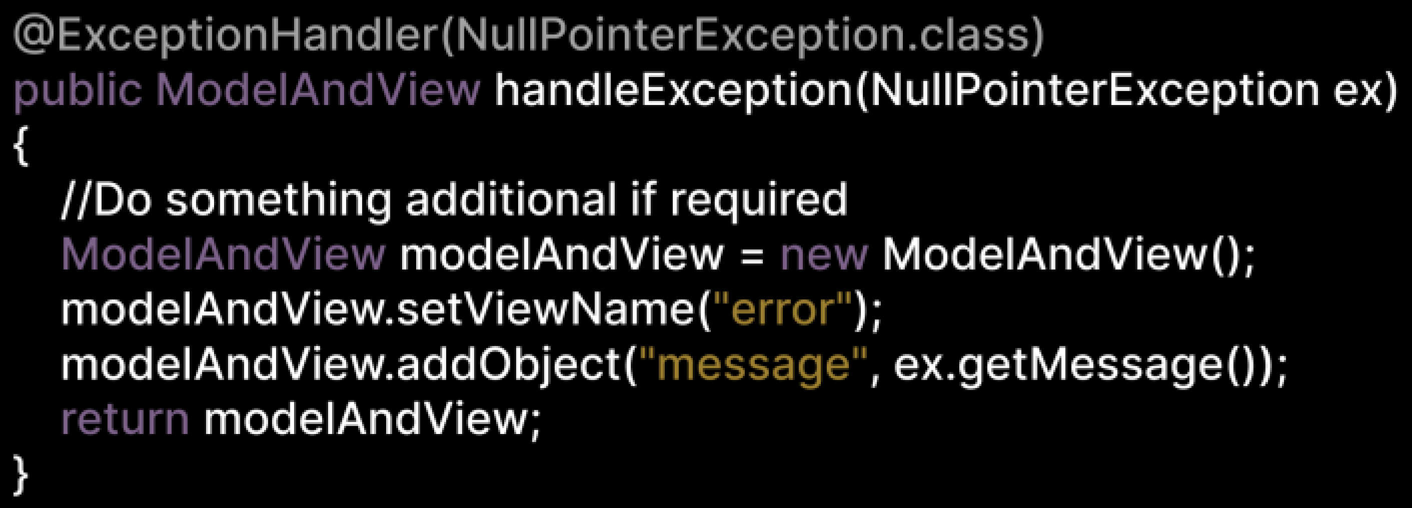
Exception handler method return types

The following return types are supported for handler methods:



- ModelAndView
- Model
- Map
- View
- String
- @ResponseBody
- HttpEntity<?>/ResponseEntity<?>
- void

Exception handler method example

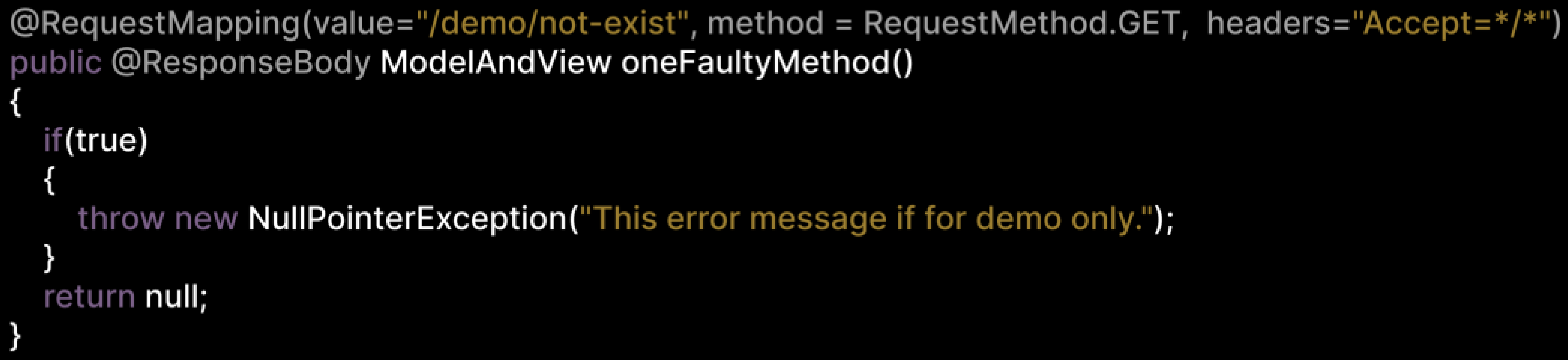


```
@ExceptionHandler(NullPointerException.class)
public ModelAndView handleException(NullPointerException ex)
{
    //Do something additional if required
    ModelAndView modelAndView = new ModelAndView();
    modelAndView.setViewName("error");
    modelAndView.addObject("message", ex.getMessage());
    return modelAndView;
}
```

Now every time, controller encounter NullPointerException in request processing for any web request in this controller, control will automatically come to this handler method.

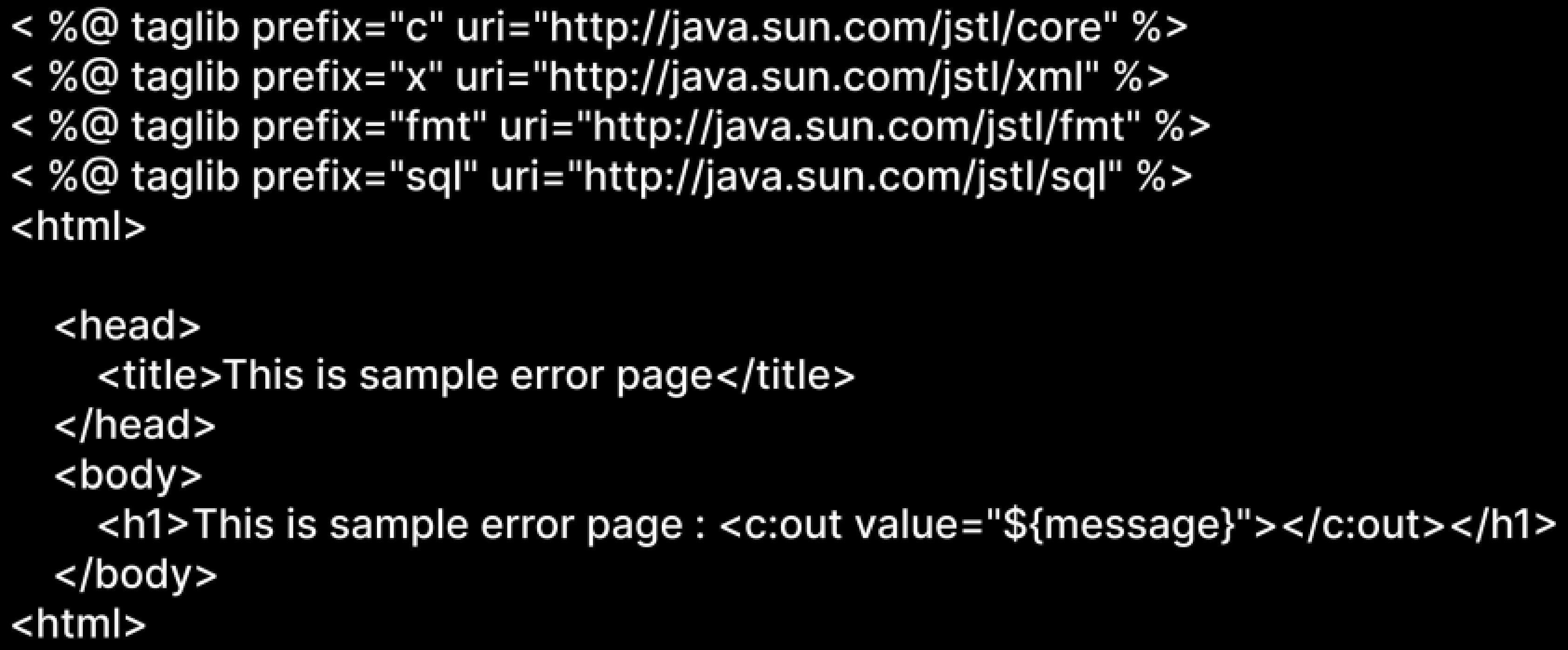
Spring @ExceptionHandler example

For example, below handler method is intentionally returning NullPointerException.



```
@RequestMapping(value="/demo/not-exist", method = RequestMethod.GET,  headers="Accept=*/*")
public @ResponseBody ModelAndView oneFaultyMethod()
{
    if(true)
    {
        throw new NullPointerException("This error message if for demo only.");
    }
    return null;
}
```

If we deploy above application and hit the URL [/SpringApplication/users/demo/not-exist] in browser, it will show the “error” page as configured in first section.



```
< %@ taglib prefix="c" uri="http://java.sun.com/jstl/core" %>
< %@ taglib prefix="x" uri="http://java.sun.com/jstl/xml" %>
< %@ taglib prefix="fmt" uri="http://java.sun.com/jstl/fmt" %>
< %@ taglib prefix="sql" uri="http://java.sun.com/jstl/sql" %>
<html>

  <head>
    <title>This is sample error page</title>
  </head>
  <body>
    <h1>This is sample error page : <c:out value="${message}"></c:out></h1>
  </body>
<html>
```