SESSION -- ANNOTATIONS

Java Annotations

Java annotations are used to provide the extra or supplement information about the program. Annotations in Java are utilized to give supplement data about a program.

Java Annotations begin with '@'.

It is an alternative option for XML and Java marker interfaces.

Used as :

1. **Comments**
2. **Passing Data**
3. **Mapping Web Requests**

Your Understanding of Annotations should be based on :

1. **Where to use @annotations in the class ??**
2. **How many values should be passed to the @annotations ??**

Locations where Annotations are used in a Class:

1. **At the Class Level i.e. above the Class Name eg:** @WebServlet
2. **At the Method Level i.e. above the Method Name eg:** @Override
3. **At the Method Parameter level eg:** @RequestParam
4. **At the Field Level i.e. for Class level Variables eg:** @AuotWired

Built-In Java/Servlet Annotations used in Java code

@Override (for overridding method in child class)
@Deprecated (for methods)
@WebServlet (for Servlets)

Annotations Code in the Background

@Target(value=method) i.e. above the method Name
@Override

```
@interface Override () { }   --- No method or zero value
```

@Target(value=TYPE) i.e. above the Class Name Type means class
@WebServlet

```
@interface WebServlet{
```

```
String name();
String[] value();                    ---Multiple methods or multi value
WebInitParam[] initParams();
.....................
}
```

```
@MyAnnotation                        ---Your Own Annotation

@interface MyAnnotation{ }
```

Types of Annotations

There are three types of annotations:

1. **Marker Annotation**
2. **Single-Value Annotation**
3. **Multi-Value Annotation**

1) Marker Annotation
An annotation that has no method, is called marker annotation. For example:

```
@interface MyAnnotation{}
```

The @Override and @Deprecated are marker annotations.

2) Single-Value Annotation
An annotation that has one method, is called single-value annotation. For example:

```
@interface MyAnnotation{
int value();
}
```

3) Multi-Value Annotation
An annotation that has more than one method, is called Multi-Value annotation. For example:

```
@interface MyAnnotation{
int value1();
String value2();
String value3();
}
}
```

Spring Annotations

@RequestParam : Request Objects Parameter. Use at method parameter level

@SessionAttributes : Session Objects Attributes . Use above Class

@GetMapping : Form action url Mapping method=get . Use above corresponding method called

@PostMapping Form action url Mapping method=post . Use above corresponding method called

@ModelAttribute : Auto transfer forms data to a Java bean object.Use at method parameter level.

@Autowired: Create object of another class without new keyword. Use inside class. Also called as Dependency Injection .Handled by Spring Container .

Spring StereoTypes(Use above Class Name):

@Component : Represents Spring Class

@Controller : Represents Spring Controller Class

@Repository : Represents Data Class with database access methods.

Program1 customerlogin.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title> Customer Login Form</title>
</head>
<body>

<form action="customerview" method="post">
<table>
<tr><td>Customer Login Form </td></tr>
<tr><td>Customer Id : </td><td><input type="text" name="cid"></td></tr>
<tr><td>Customer Name : </td><td><input type="text" name="cname"></td></tr>
<tr><td>Customer Email : </td><td><input type="text" name="cemail"></td></tr>
<tr><td></td><td><input type="submit" value="Submit"></td></tr>
</table>
</form>
</body>
</html>
```

Program2:CustomerController

```
import java.util.Iterator;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.SessionAttributes;
import org.springframework.web.servlet.ModelAndView;

@Controller
@SessionAttributes({"cid","cname","cemail"})
 public class CustomerController
 {
	@Autowired
	CustomerInt impl;

	@GetMapping("customerlogin")
	public String custLogin()
	{
		return "customerlogin";
	}

	@PostMapping("customerview")
	public ModelAndView custDetails(@RequestParam("cid") String cid,
@RequestParam("cname") String cname, @RequestParam("cemail") String cemail ,
@ModelAttribute Customer cust)
	{
		ModelAndView mv = new ModelAndView();
		mv.addObject("cid", cid);
		mv.addObject("cname", cname);
		mv.addObject("cemail", cemail);
		mv.setViewName("CustomerView");

		int cid1 = cust.getCid();
		System.out.println("The id is :" + cid1);
		String cname1 = cust.getCname();
		System.out.println("The name is :" + cname1);
		String cemail1 = cust.getCemail();
		System.out.println("The email is :" + cemail1);
```

```
            return mv;

        }

        @GetMapping("customersessionview")
        public String custSession()
        {

            return "CustomerSessionview";
        }
}
```