

MODULE 10

COMPONENT AND SERVICE

TABLE OF CONTENT

- JavaBean
- Component
- Bean Injection
- Service
- Session Scope Component

JavaBean

- ▶ JavaBean is a Plain Old Java Object (POJO) that
 - is serializable
 - has a no-argument constructor
 - allows access to properties using getter and setter methods

Component

- ▶ Java class annotated by **@Component**
 - Name of the bean can be specified by the value of the annotation (optional)
- ▶ Definition of bean for the Spring bean container
 - ↳ Spring can pick up an object of the class and pull it in the application context
 - ↳ This bean can be injected in the application when needed
- ▶ By default, application scope (singleton)
 - ⇒ The same object can be reused
 - ⇒ ***Pay attention in use of instance variable !!!***
- ▶ For session scope
 - See further

Component

- ▶ **Types of component for bean injection**
 - Controller (Spring Web MVC controller)
 - Service (see below)
 - Repository (see DAO)

Bean Injection

- ▶ Spring BeanFactory able to autowire relationships between collaborating beans
 - Able to inject a bean into another one
- ▶ Through **@Autowired**

Service

- ▶ For bean injection of service-layer class
 - For classes containing business rules and processes
- ▶ using **@Service** annotation
 - Is a specialization of **@Component**
- ▶ Application scope (singleton)
- ▶ E.g,

```
import org.springframework.stereotype.Service;

@Service
public class OrderService {
    public Double refund(){
    }
```

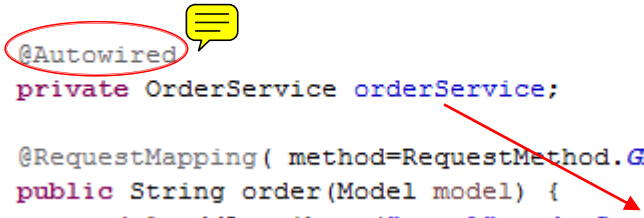
Service

- ▶ Injection of service bean
 - Through **@Autowired**
- ▶ E.g,

```
@Controller
@RequestMapping (value="/order")
public class OrderController {

    @Autowired
    private OrderService orderService;

    @RequestMapping( method=RequestMethod.GET)
    public String order(Model model) {
        model.addAttribute("total",orderService.refund());
    }
}
```

A diagram illustrating the @Autowired annotation. The annotation is circled in red, and a yellow speech bubble points to it. A red arrow points from the orderService field to the refund() method call in the order() method.

Session Scope Component

- ▶ Session scope
 - Persists across multiple HTTP requests belonging to the user
 - Objects with this scope
 - Are visible for any request/response cycle that belongs to a **session**
 - Have their state persisted between requests until the end of the session
- ▶ The common way to define Session attribute
 - ⇒ See Session Attribute Module

Session Scope Component

► E.g,

```
@Controller
@RequestMapping(value="/color")
@SessionAttributes({ColorController.COLOR})
public class ColorController {

    protected static final String COLOR = "mainColor";

    @ModelAttribute(COLOR)
    public Color chosenColor() {
        return new Color();
    }
}
```

Session Scope Component

- ▶ Other way (less used)
 - Java class annotated by **@Component**
 - Name of the bean can be specified by the value of the annotation
 - Session Scope through **@Scope** with **value="session"**

▶ E.g,

```
import org.springframework.context.annotation.Scope;  
import org.springframework.stereotype.Component;  
import org.springframework.context.annotation.ScopedProxyMode;  
  
@Component("currentBooking")  
@Scope(value="session", proxyMode = ScopedProxyMode.TARGET_CLASS)  
public class Booking {
```

Session Scope Component

- ▶ Injection through *@Autowired*
- ▶ E.g,

```
@Controller
@RequestMapping(value="/booking")
public class BookingController {

    @Autowired
    private Booking currentBooking;
```

Session Scope Component

- ▶ In jsp page

`${currentBooking.name}`

- ▶ N.B. Modify tilesConfig

```
@Bean
public ViewResolver tilesViewResolver ()
{
    final TilesViewResolver resolver = new TilesViewResolver();
    resolver.setViewClass(TilesView.class);
    resolver.setExposeContextBeansAsAttributes(true);
    return resolver;
}
```