Rasesh Mori
Share what you know…..

Spring Core Reference – Cheatsheet

Posted on **January 6, 2015**

Factory
=======

BeanFactory factory = new XmlBeanFactory(new FileSystemResource(fileName));
factory.getBean(beanName);

ApplicationContext context = new ClassPathXmlApplicationContext(fileName);
context.getBean(beanName);

Property
========

By Value
<bean id=”beanName” class=”org.abcd.efg.AClass”>
<property name=”name” value=”Rasesh”/>
</bean>

By Reference
<bean id=”beanName” class=”org.abcd.efg.AClass”>
<property name=”name” ref=”anotherBeanName”/>
</bean>

Constructor
==========

<bean id=”beanName” class=”org.abcd.efg.AClass”>
<constructor-arg index=”0″ value=”Rasesh”/>
<constructor-arg index=”1″ ref=”anotherBeanName”/>
</bean>

Collections
=========

```
<bean id="beanName" class="org.abcd.efg.AClass">
<property name="names" >
<list>
<ref bean="beanName"/>
<ref bean="anotherBeanName"/>
</list>
</property>
</bean>
```

Autowiring by tags
==========

ByName

```
<bean id="beanName" class="org.abcd.efg.AClass" autowire="byName">
</bean>
```

Here, it will search for the beans with the same name as member variables in AClass and autowire them.

ByType

```
<bean id="beanName" class="org.abcd.efg.AClass" autowire="byType">
</bean>
```

Here, it will try and match the data type of beans and member variables in AClass. If there are multiple beans of same data type, this won't work.

Constructor

```
<bean id="beanName" class="org.abcd.efg.AClass" autowire="constructor">
</bean>
```

This is autowire by Type, but this will set the member variables through constructor instead of setters.

Bean Scopes
==========

Singleton, Prototype

Annotations
==========

@Required

Adding Required annotation to setter will fail when it initializes the bean if the value is null instead of a later NPE when it is referenced.

```
<bean
class="org.springframework.beans.factory.annotation.RequiredAnnotationBeanPostProcessor" />
```

This bean will take care of the process mentioned above.

@Autowired
This works same as auto-wiring by type falling back to auto-wiring by name. If still not found, ***qualifier*** can be used.

<bean id="……>
<qualifier value="qualifierString"/>
…
</bean>

<context:annotation-config/> can be used to avoid adding bean post processor for each annotation.

This along with ***@Qualifier("qualifierString")*** will be able to auto wire the bean.
<bean
class="org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor"
/>

@Resource
This can be used for dependency injection by name. ***@Resource(name="beanName")*** will wire the bean with specified name.

If name is not specified, it will do auto-wiring by name.

@PostConstruct and @PreDestroy can be used for init method and destroy method with context.registerShutDownHook().

@Component

This can be used to avoid xml configuration and spring will create a bean named as the class's name. This can only handle singleton bean.

<context:component-scan base-package="org.abcd.efg"/>

@Service, @Controller, @Repository

component-scan tag will pick this up as well. These map to MVC data model.

MessageSource to get text from property files
==========

<bean id="messageSource"
class="org.springframework.context.support.ResourceBundleMessageSource">
<property name="basenames">
<list>
<value>mymessages</value>
</list>
</property>
</bean>

context.getMessage("greeting", null, //message parameter
"Default greeting", null //Locale
);

OR

MessageSource messageSource = context.getBean("messageSource");
messageSource.getMessage("greeting", null, //message parameter
"Default greeting", null //Locale
);

parameter ex:

drawing.point=Circle: Point is: ({0}, {1})
context.getMessage("drawing.point", new Object[] {"val1", "val2"}, "Default point msg", null)

RELATED

Scala: Part 4: Classes & Objects
In "Programming Language"

Mockito and Power Mockito - Cheatsheet
In "Java"

Design Pattern: Builder Pattern
In "Design Pattern"

This entry was posted in **Java** and tagged **@Component**, **@Qualifier**, **@Required**, **autowire**, **bean**, **beans**, **byName**, **byType**, **cheatsheet**, **constructor**, **constructor-arg**, **controller**, **core**, **property**, **repository**, **resource**, **service**, **spring** by **Rasesh Mori**. Bookmark the **permalink [https://raseshmori.wordpress.com/2015/01/06/spring-core-reference-cheatsheet/]** .