一，注解

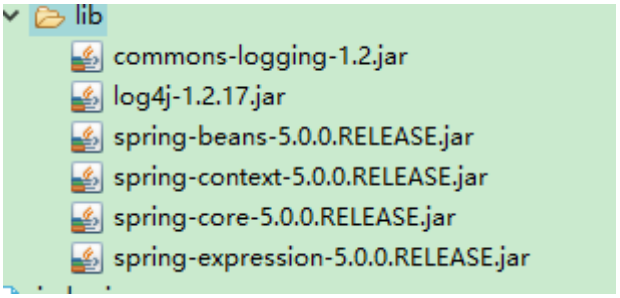
1，注解是代码中的特殊的标记，使用注解可以完成相关的功能。

2，注解写法：@注解名称（属性名称=属性值）

3，注解可以使用在类上、方法上和属性上。

二，spring注解开发的准备工作（使用注解的方式可以替代配置文件，不是完全配置，只是少写配置文件）

1，导入相关jar包



log4j-1.2.17.jar
478.4KB

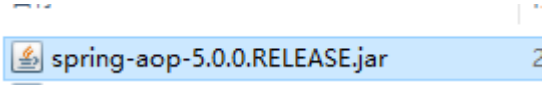
commons-logging-1.2.jar
60.37KB

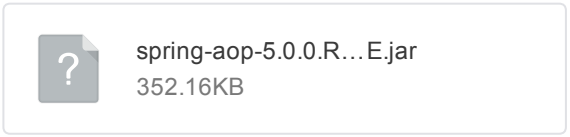
spring-expression-…E.jar
256.53KB

spring-beans-5.0.0…E.jar
639.36KB

spring-context-5.0.… E.jar
1.02MB

2，导入使用注解需要的aopjar包





三，创建对象，创建方法

```
package cn.java.entity;
public class user {
public void add(){
	System.out.println("add.....");
}
}
```

四，创建spring配置文件，引入约束

1，第一天做ioc基本功能，引入beans约束。

2，做spring的ioc的ioc注解开发，需要引入一个context约束。

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
	xsi:schemaLocation="http://www.springframework.org/schema/beans
		http://www.springframework.org/schema/beans/spring-beans.xsd
		http://www.springframework.org/schema/context
		http://www.springframework.org/schema/context/spring-context.xsd
		http://www.springframework.org/schema/util
		http://www.springframework.org/schema/util/spring-util.xsd ">
</beans>
```

五，注解方式创建对象

1，在需要创建的实体类上写上相应的注解

```
package cn.java.entity;
import org.springframework.stereotype.Component;
@Component(value="user")//==<bean id="user" class="">
public class user {
public void add(){
	System.out.println("add.....");
}
}
```

2，applicationcontext.xml的配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
	xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
	xmlns:jdbc="http://www.springframework.org/schema/jdbc"
xmlns:jee="http://www.springframework.org/schema/jee"
	xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:util="http://www.springframework.org/schema/util"
	xmlns:task="http://www.springframework.org/schema/task"
	xsi:schemaLocation="
	http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
	http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd
	http://www.springframework.org/schema/jdbc
http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd
	http://www.springframework.org/schema/jee
http://www.springframework.org/schema/jee/spring-jee-4.0.xsd
	http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd
	http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util-4.0.xsd
	http://www.springframework.org/schema/task
http://www.springframework.org/schema/task/spring-task-4.0.xsd"
	default-lazy-init="false">
	<!-- 开启注解扫描,base-package的值就是需要进行注解开发的类所在的包，若只有俩三个包的话，可以写cn.java或者cn （１）到指定的包中扫描类、方法、属性上是否有注解 -->
	<context:component-scan base-package="cn.java"></context:component-scan>
	<!-- 扫描属性上面的注解 -->
	<context:annotation-config></context:annotation-config>
</beans>
```

3，测试代码

```
package cn.java.test;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import cn.java.entity.user;
public class test1 {
@Test
public void testUser(){
	ApplicationContext context =
			new ClassPathXmlApplicationContext("applicationcontext.xml");
	user user = (user)context.getBean("user");
	System.out.println(user);
```

```
}
}
```

六，spring的bean管理中常用的注解

1，④@component：组件（作用在类上）

spring中提供@component的三个衍生注解（功能目前来说是一致的）

①@Controller：web层。

②@Service：业务层。

③@Repository：持久层。

这三个注解是为了让标注类本身的通途清晰，spring在后续版本会对其进行增强。

在类的上面写上以下的代码，表示该实体对象是一个单实例的对象。

@Scope(value="prototype")//配置这个对象是单实例还是多实例

2，创建对象是单实例还是多实例。

```
package cn.java.entity;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;
@Component(value="user")//<bean id="user" class="">
@Scope(value="prototype")//配置这个对象是单实例还是多实例
public class user {
public void add(){
        System.out.println("add.....");
}
}
```