1、Spring创建对象的第一个例子

实现步骤：

```
1、创建一个idea-maven-spring的空工程
2、在该工程中添加新模块，选择maven项目，由于这里是测试Spring功能的，选择的模板为quickstart
模块名：a-spring-hello-01
3、加入maven依赖
    Spring的依赖，选择仓库中已经下载好的
    Junit依赖
    <dependencies>

    <!--单元测试依赖-->
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>

    <!--Spring依赖-->
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context</artifactId>
      <version>5.2.4.RELEASE</version>
    </dependency>

  </dependencies>
4、创建类（一个接口以及该接口的实现类）
    跟普通的业务类一样
5、创建Spring需要使用的配置文件
    目的是声明类的基本信息，由Spring读取这些信息，然后Spring就可以创建这些类的对象和进行管理
6、测试
    如何使用Spring的创建类
```

2、步骤

1、创建一个idea-maven-spring的空工程

2、在该工程中添加新模块，选择maven项目，由于这里是测试Spring功能的，选择的模板为quickstart

模块名：a-spring-hello-01

将该添加的目录添加，resources。将不需要的删掉，如pom文件中的无用配置

3、加入maven依赖，pom文件如下

```xml
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.studymyself</groupId>
  <artifactId>a-spring-hello-01</artifactId>
  <version>1.0-SNAPSHOT</version>


  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>

  <dependencies>

    <!--单元测试依赖-->
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>

    <!--Spring依赖-->
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-context</artifactId>
      <version>5.2.4.RELEASE</version>
    </dependency>

  </dependencies>

  <build>

  </build>
</project>
```

4、创建类（一个接口以及该接口的实现类）

创建一个service包，新建一个SomeService接口，定义一个方法doSome，然后在改包下创建一个子包impl，里面新建接口的实现类SomeServiceImpl，重写doSome方法，里面输出一句话。

```java
package com.studymyself.service;

public interface SomeService {

    public void doSome();

}
```

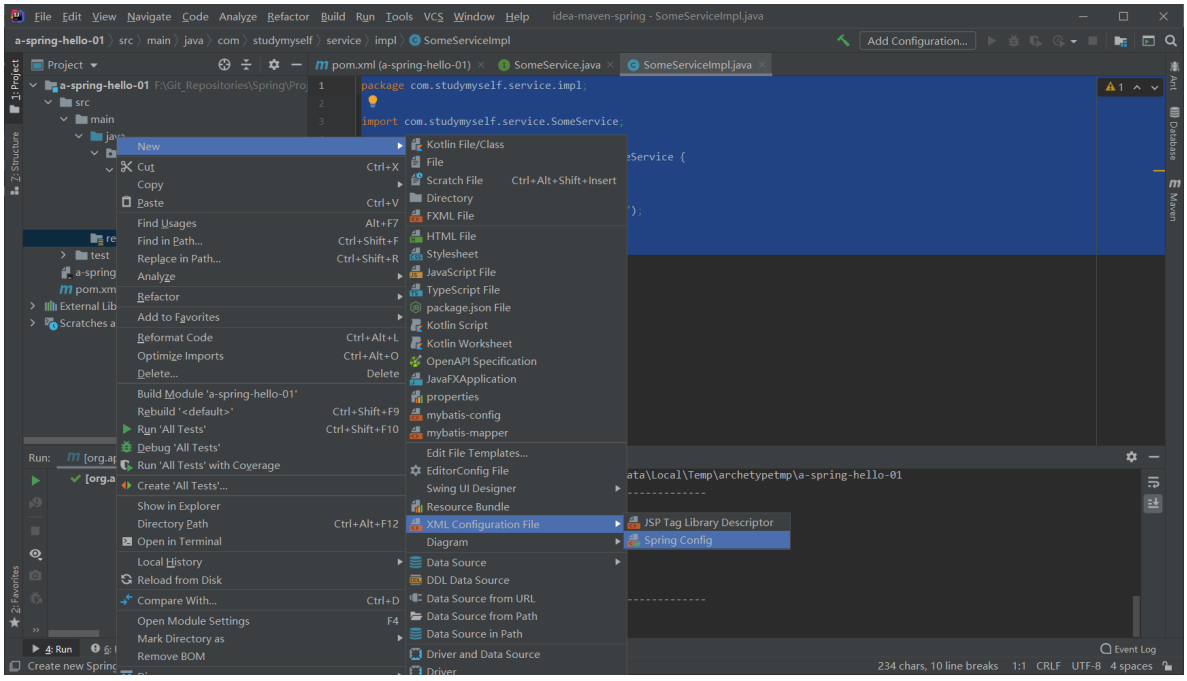
```java
package com.studymyself.service.impl;

import com.studymyself.service.SomeService;

public class SomeServiceImpl implements SomeService {
    @Override
    public void doSome() {
        System.out.println("Hello Spring!!!");
    }
}
```

5、在resources目录下创建Spring需要使用的配置文件，beans.xml

右键resources->new->XML Configuration File->Spring Config




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

    <!--
        告诉Spring创建哪个类的对象，叫做：
        声明bean，就是告诉Spring要创建某个类的的对象，使用<bean>标签
```

```
        id:表示Spring创建的这个对象的名称，自定义的，唯一的。
            当我们要使用该对象时把这个id，Spring就根据该id查找对应的对象交给我们
        class：类的全限定名称，Spring获取类的完整类名就可以获取该类的Class对象，
                然后通过反射机制创建这个类的对象，所以不能填接口的全限定名
    -->
    <bean id="someService"
class="com.studymyself.service.impl.SomeServiceImpl"/>
    <!--
        底层实现：
        Class someServiceImpl =
Class.forName(com.studymyself.service.impl.SomeServiceImpl);
        SomeService someService = someService.newInstance();
        spring中有专门存放bean的Map集合
        springMap.put(id的值，创建的对象),如
        springMap.put("someService",someService.newInstance());
    -->

</beans>

<!--
    Spring配置文件说明：
        1、<beans>：是根标签，复数，所以该标签中有许多<bean>标签，每一个<bean>标签中配置一
个java类信息
            Spring将Java对象定义为bean，一个bean就是一个Java对象
        2、上面中的spring-beans.xsd 是一个约束文件，和mybatis的核心配置文件中的dtd文件一
样。
-->
```

6、在测试类中使用用Spring创建的对象，MyTest.java

```
package com.studymyself;

import com.studymyself.service.SomeService;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MyTest {

    /**
     * 测试使用Spring创建的对象
     */
    @Test
    public void testDoSome(){

        //1、指定Spring配置文件的路径及名称
        String config = "beans.xml";

        //2、创建代表Spring容器的对象，用ApplicationContext
        /*
            ApplicationContext是一个接口，Ctrl+H查看该接口的实现类，主要看的有两个：
        FileSystemXmlApplicationContext实现类：
            用这个类new容器对象获取Spring配置文件是从系统的硬盘中获取的，
            就是说Spring配置文件存储在d盘或其他盘中，构造方法中传的是带有
            硬盘路径的配置文件路径字符串
        ClassPathXmlApplicationContext实现类：
            表示new容器对象需要的Spring配置文件只从类的根路径下搜索，超出这个范围的配置文件
```

```
            会报错（这是最常用的，因为在resources目录中的配置文件编译后存放的就是类的根路径下target\classes）

         */
        ApplicationContext ac = new ClassPathXmlApplicationContext(config);

        //3、使用容器代表对象中获取类的对象的方法来获取需要的对象
        //getBean方法中参数可以传字符串类型的（配置文件中的bean的id），也可是Class类型的等等
        //这里将配置在配置类中的SomeServiceImpl类的id放进该方法中，
        // 由于该方法返回的是Object类型，强转成接口类型,多态
        SomeService someServiceImpl = (SomeService) ac.getBean("someService");

        //4、调用doSome方法
        someServiceImpl.doSome();
    }

}
```

7、分析对象被容器创建的时间

通过在实现类中的无参数构造方法添加输出数据，可知，在我们获取Spring容器的代表对象的时候，期间底层把配置文件传入到Spring内部，通过读取Spring配置文件中的信息，把里面的每一个标签代表的类都创建了对象。

```
//Spring默认创建对象的时间：在创建Spring容器对象时创建配置文件中所有的javabean对象
//即执行下面方法时创建javabean
ApplicationContext ac = new ClassPathXmlApplicationContext(config);
```

8、获取容器存储的Javabean的信息

```
    /*
        测试如何获取容器中Javabean的信息
     */
    @Test
    public void test01(){

        //1、指定Spring配置文件的路径及名称
        String config = "beans.xml";
        //2、创建代表Spring容器的对象
        ApplicationContext ac = new ClassPathXmlApplicationContext(config);

        /*
            通过容器对象中的方法获取容器对象的信息
         */
        //获取容器中创建的javabean数量
        int count = ac.getBeanDefinitionCount();
        System.out.println("spring容器中创建的对象数量："+count+"\n=======================================");

        //获取容器中创建的所有对象的id名称，返回的是一个
        String[] names = ac.getBeanDefinitionNames();
        for (String name:
```

```
            names) {
            System.out.println(name);
        }

    }
```

9、让Spring创建非自定义的对象

Spring创建对象，使用DI的方式，底层使用的是反射，我们只需要将JavaSE类库或者JavaEE类库中的一个类的完整类名作为配置文件中class属性的值，Spring就可以帮助创建该类的对象了，如下

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

    <!--配置非自定义的类如常用类，让Spring创建该类对象-->
    <!--Date类-->
    <bean id="myDate" class="java.util.Date"/>
    <!--String类-->
    <bean id="myString" class="java.lang.String"/>

</beans>
```

10、从中我们可以看到Spring中的每个类的对象都是单例的，

放进容器中的对象是dao层、service层、Controller层的类

不应该放进容器中的对象：实体类，实体类数据是来自数据库的。还有Servlet、listener、filter等，这是让服务器创建的