**搭建步骤：**

**1、通过maven引入所需要jar包**

<**dependency**>  
<**groupId**>org.springframework</**groupId**>  
<**artifactId**>spring-context</**artifactId**>  
<**version**>5.2.1.RELEASE</**version**>  
</**dependency**>

**2、编写beans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"  
xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"**>  
  
<**bean id="accountService" class="main.java.wufan.springIOC2.service.impl.AccountserviceImpl"**></**bean**>  
<**bean id="accountDao" class="main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl"**></**bean**>  
</**beans**>

**3、 spring提供了两个接口（ApplicationContext BeanFactory） 来创建容器对象**

**ApplicationContext ac1 = new ClassPathXmlApplicationContext("SpringIOC2beans.xml");** *//根据ID获取bean*IAccountService iAccountService ;  
AccountDao accountDao ;  
**for** (**int** i = 0; i < 5; i++) {  
iAccountService = (IAccountService) ac1.getBean(**"accountService"**);  
accountDao= (AccountDao) ac1.getBean(**"accountDao"**);  
  
System.***out***.println(**"IAccountService ======== "** + i +**"===="**+ iAccountService);  
System.***out***.println(**"accountDao ========= "** + i +**"===="**+ accountDao);  
}

IAccountService ======== 0====main.java.wufan.springIOC2.service.impl.AccountserviceImpl@464bee09

accountDao ========= 0====main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl@f6c48ac

IAccountService ======== 1====main.java.wufan.springIOC2.service.impl.AccountserviceImpl@464bee09

accountDao ========= 1====main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl@f6c48ac

IAccountService ======== 2====main.java.wufan.springIOC2.service.impl.AccountserviceImpl@464bee09

accountDao ========= 2====main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl@f6c48ac

IAccountService ======== 3====main.java.wufan.springIOC2.service.impl.AccountserviceImpl@464bee09

accountDao ========= 3====main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl@f6c48ac

IAccountService ======== 4====main.java.wufan.springIOC2.service.impl.AccountserviceImpl@464bee09

accountDao ========= 4====main.java.wufan.springIOC2.Dao.Impl.AccountDaoImpl@f6c48ac

**通过循环打印可以看出容器的对象只创建了一次，是单利模式。可以通过beans配置文件指定多利模式**

ApplicationContext继承接口结构及 它的实现子类 常用的三个红框标记

区别：

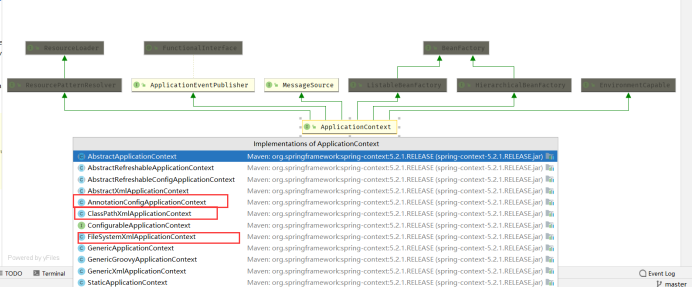
1、ClassPathXmlApplicationContext：**可加载在类路径classpath（classes）下的配置文件,要求文件必须存在**

ApplicationContext ac2 = **new** ClassPathXmlApplicationContext(**"SpringIOC2beans.xml"**);

2、FileSystemXmlApplicationContext：**可加载磁盘任意路径下的配置文件（必须有访问目录的权限）**

ApplicationContext ac1 = **new** FileSystemXmlApplicationContext  
(**"D:/allSoftWare/code/development/SpringTest/target/classes/SpringIOC2beans.xml"**);

3、AnnotationConfigApplicationContext：**它用于读取通过注解来创建容器。**



**BeanFactory 初始化容器**

Resource resource = **new** ClassPathResource(**"SpringIOC2beans.xml"**);  
BeanFactory ac4 = **new** XmlBeanFactory(resource);

**ApplicationContext BeanFactory区别**

**ApplicationContext 读取完配置文件立即创建bean**

**BeanFactory 读取完配置文件不会创建bean，等到使用时候再进行创建**