简单来说spring实现的功能就是将java自己管理对象转交由spring容器管理

简单的一个接口

```
public interface Person {

    public void show();
}
```

简单的接口实现类

```
public class User implements Person {

    @Override
    public void show() {
        System.out.println("hello kugou");
    }

}
```

主体，application.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" xmlns:context="http://www.springframewor
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org
                http://www.springframework.org/schema/context http://www.springframework.org/schem
                http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/

    <bean id="userBean" class="com.jlb.implBean.User" />
</beans>
```

可以看到，该xml配置了一个bean，指向就是我们User类，这就完成了类的注册，这样User类就可以通过spring来管理

```
 @Test
 public void testSpring() throws Exception {
     // 读取配置文件
     ApplicationContext ctx = new ClassPathXmlApplicationContext("applicationContext.xml");
     // 获取UserBean的实例
     Person person = (Person) ctx.getBean("userBean");
     // 调用方法
     person.show();
     Person p=new User();
     p.show();
 }
```

如图，我们采用了两种方式创建了Person对象，当然可以看到，都是实例化为User对象，第一种方式就是我们新的方式，通过spring获取实例，第二种依旧是传统的new，我们看看结果

```
信息: Pre-instantiati
hello kugou
hello kugou
```

显然，都正确调用了show方法（好处嘛，等我体会到了再来讲（手动允悲））

实例化bean有三种方式

使用类构造器直接实例化

使用静态工厂的方法实例化

使用实例工厂方法实例化

```
<!-- 使用类构造器直接实例化 -->
<bean id="userBean1" class="com.jlb.implBean.User"/>

<!-- 使用静态工厂的方法实例化 -->
<bean id="userBean2" class="com.jlb.factory.BeanFactory" factory-method="UserService"/>

<!-- 使用实例工厂方法实例化 -->
<bean id="factory" class="com.jlb.factory.BeanFactory" />
<bean id="userBean3" factory-bean="factory" factory-method="getUserService" />
```

```
public class BeanFactory {

    //静态工厂使用
    public static User UserService(){
        return new User();
    }

    public User getUserService(){
        return new User();
    }

}
```

我们最常用的还是第一种方式

Bean交给spring管理后，默认情况下是单例的

```
@Test
public void testSpring2() throws Exception {
    // 读取配置文件
    ApplicationContext ctx = new ClassPathXmlApplicationContext("applicationContext.xml");
    // 获取UserBean的实例
    Person person1 = (Person) ctx.getBean("userBean");
    Person person2 = (Person) ctx.getBean("userBean");
    System.out.println(person1 == person2);
}
```

```
true
```

可以看到这两个实例其实是一个实例

如何让每次获取实例不一样呢

在配置bean的时候加一个scope属性

```
<bean id="userBean" class="com.jlb.implBean.User" scope="prototype" />
```

这样每次获取的就是不一样的实例了

```
false
```

结果就是false了