--004

1 -- Bean 种类

举例1(普通bean):

目标类:

```
class UserName{


}
```

普通bean :

```
<bean name="userName" class="com.znsd.UserName"></bean>
```

Spring 直接创建UserName 的对象 并返回

底层幻想:

```
UserName userName = new UserName();
return userName;
```

举例2 (特殊bean) :

目标类:

```
class agencyFactoryBean implement FactoryBean {
    @Overide
    public Object getObject(){
        return new UserName();
    }
}
```

特殊bean:

FactoryBean : 具有生产对象的能力，只能生产特定的对象

bean必须 实现FactoryBean 接口 ，此方法提供getObject()用于返回特定bean对象

```
<bean name = "userName" class = "com.znsd.agencyFactoryBean" factory-bean="getObject"></bean>
```

不清楚要不要添加：factory-bean="getObject" 可能底层会自动调用

底层幻想:

```
FB fb = new FB();
```

return fb.getObject();//获取到指定bean

factoryBean 与 Beanfactory的区别

1.BeanFactory ：是一个工厂，用于生产任意bean

2.Factorybean：具有工厂生产对象的能力，但只能生产指定bean

2 -- 作用域

类别	说明
singleton	在Spring IoC容器中仅存在一个Bean实例，Bean以单例方式存在，默认值
prototype	每次从容器中调用Bean时，都返回一个新的实例，即每次调用getBean()时，相当于执行new XxxBean()
request	每次HTTP请求都会创建一个新的Bean，该作用域仅适用于WebApplicationContext环境
session	同一个HTTP Session 共享一个Bean，不同Session使用不同Bean，仅适用于WebApplicationContext 环境
globalSession	一般用于Portlet应用环境，该作用域仅适用于WebApplicationContext 环境

当通过spring获取对象时：

singleton(默认)：在spring ioc容器中 仅存一个实例，bean以单例模式存在，默认值.

prototype(非单例)：每次从spring 中调用getbean 时 返回的都是新的对象 相当于：return new bean();

request(一次请求)

session(一次会话)

配置信息:

scope = "singleton || prototype" 看用户需求，若想非单例则填入prototype 不写则单例

<bean name="userName" class="com.znsd.UseName" scope="prototype"></bean>