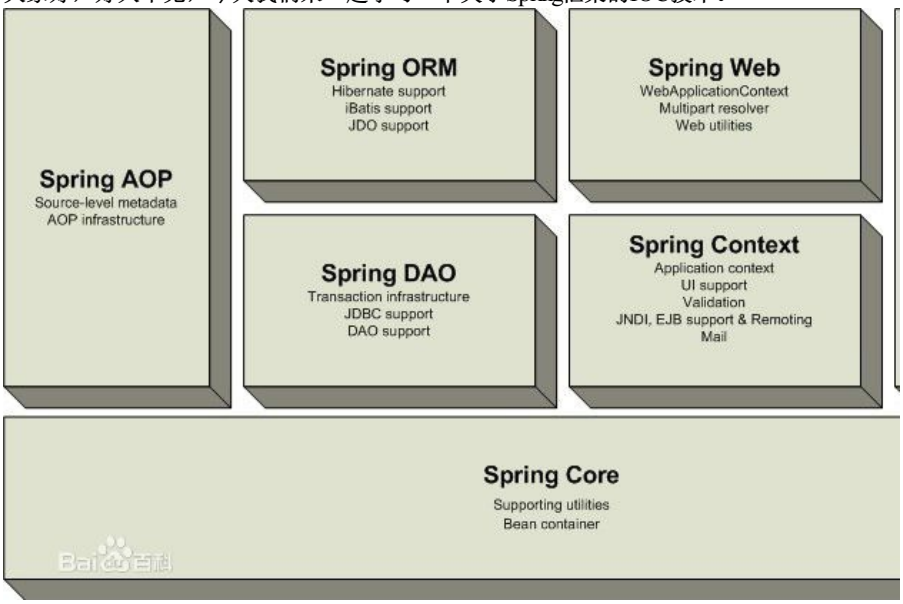
【Java框架型项目从入门到装逼】第一节 - Spring框架 IOC的丧心病狂解说

大家好，好久不见，今天我们来一起学习一下关于Spring框架的IOC技术。



控制反转——Spring通过一种称作控制反转（IoC）的技术促进了松耦合。当应用了IoC，一个对象依赖的其它对象会通过被动的方式传递进来，而不是这个对象自己创建或者查找依赖对象。你可以认为IoC与JNDI相反——不是对象从容器中查找依赖，而是容器在对象初始化时不等对象请求就主动将依赖传递给它。

麻烦，说人话！

好吧，那我们从简单的说起。直接讲概念有点抽象，让我们从一个实际的场景来分析这个事情。大毛想要找个女朋友，他的朋友二毛帮忙介绍了一个女生，叫翠花。然后大毛跟翠花认识了。相处了一段时间，因为翠花要大毛和他一起打王者荣耀，可是大毛只会打斗地主，所以分手了，为此大毛难过了好长一段时间。后来，大毛又去找二毛帮忙，希望重新找一个女朋友。大家看这个过程，其实很麻烦的，大毛如果通过熟人介绍的方式来找女朋友，必然是一个一个的接触，比如第一次他跟翠花相亲，第二次跟桂花相亲，第三次又跟西兰花相亲，每次都耗费大量的精力。

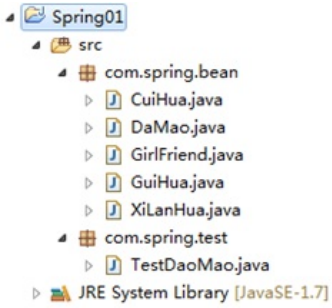
相亲吗？

相亲吃饭得花钱吧，看电影得花钱吧，万一谈不来，分手还麻烦。这就是大毛跟他每一个女朋友的依赖性太强了。

现在，请大家进行一个想象，想象我们自己是天神，以我们自己的意志来给大毛安排一个女朋友。既然我们是天神，就可以给大毛编写一个人生的剧本。

反正我历完劫
就可以飞升上神了

之前大毛是通过二毛介绍，才认识了翠花，桂花，和西兰花。现在，我们通过代码的形式来重现一个这个过程。



剧本：

大毛：

```
package com.spring.bean;

public class DaMao {

    private GirlFriend girlFriend;

    public GirlFriend getGirlFriend() {
        return girlFriend;
    }

    public void setGirlFriend(GirlFriend girlFriend) {
        this.girlFriend = girlFriend;
    }

}
```

女朋友作为一个超类：

```
package com.spring.bean;

/**
 * 女朋友
 * @author Administrator
 */
public class GirlFriend {
    private String hobby;

    public String getHobby() {
        return hobby;
    }

    public void setHobby(String hobby) {
        this.hobby = hobby;
    }

}
```

翠花继承自GirlFriend：

```
package com.spring.bean;

public class CuiHua extends GirlFriend{

}
```

测试类：

```
package com.spring.test;

import com.spring.bean.CuiHua;
import com.spring.bean.DaMao;
import com.spring.bean.GirlFriend;

public class TestDaoMao {

    public static void main(String[] args) {
        DaMao daMao = new DaMao();

        GirlFriend cuihua = new CuiHua();
        cuihua.setHobby("王者荣耀");

        daMao.setGirlFriend(cuihua);

        //看一下大毛的女朋友喜欢做什么？
        System.out.println(daMao.getGirlFriend().getHobby());

    }
}
```

看完了这个例子，我们发现，虽然我们是天神，可是安排这样的剧本就要去不断地去new新的GirlFriend，这样是不是很麻烦？

实际我们天神的职责是掌管这个大毛的命运，在合适的时间，让大毛去谈一场恋爱就可以了，至于让谁充当大毛的女朋友，作为天神，我们

不关心，对不对？

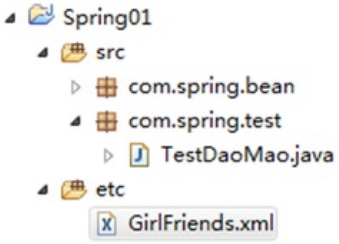
或者说，反正是我们决定，无所谓是谁。

所以，我们只需要安心编写大毛谈恋爱的剧本就行了，我们只需要知道大毛有一个女朋友就OK了，到底是谁，以后再说。因为，我们是在大毛出生之前就给他安排了剧本。那是不是我们永远都不给大毛指定一个女朋友呢？当然也不是，我们天神一般喜欢把那些可能会改变的东西写在“神圣的草稿纸”上，这个草稿纸就是XML文件。

接下来，让我们引入Spring的集成环境，来模拟这个事情！

首先，引入Spring需要的jar包。

第二步，写配置文件。




```
<?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="DaMao" class="com.spring.bean.DaMao">
        <property name="girlFriend" ref="CuiHua"></property>

    </bean>

    <bean id="CuiHua" class="com.spring.bean.CuiHua">
        <property name="hobby" value="王者荣耀"></property>
    </bean>

</beans>
```

然后，剧本就变成了这样：

```
package com.spring.test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
```

```
import com.spring.bean.DaMao;

public class TestDaoMao2 {

    public static void main(String[] args) {

        ApplicationContext context = new ClassPathXmlApplicationContext("GirlFriends.xml");
        DaMao daMao = (DaMao)context.getBean("DaMao");

        //看一下大毛的女朋友喜欢做什么？
        System.out.println(daMao.getGirlFriend().getHobby());

    }
}
```

运行效果：

```
十一月 21, 2017 11:01:22 上午 org.springframework.context.support.AbstractApplicationContext prepareRefresh
INFO: Refreshing org.springframework.context.support.ClassPathXmlApplicationContext@18105e8: startup date [Tue
十一月 21, 2017 11:01:22 上午 org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
INFO: Loading XML bean definitions from class path resource [GirlFriends.xml]
王者荣耀
```

这样的好处显而易见，我们不需要在剧本（Java代码）里面指定谁是大毛的女朋友了，这些事情我们都写在“神圣的草稿纸”上。牛逼了我的哥。IOC的原理就是在不改变剧本的情况下，由作为“天神”的你来指定谁来出演，体现在一个XML文件上。而且这些配置都是可以更改的。剧本就是Java代码，能不改就不改。

作业：
新建一个Java项目SpringTest01，创建包 com.spring.bean。
需要创建的Java类：
Hero .java
属性：
Private String heroName;
Priate String type;
Private String description;
生成对应的get、set方法，再跟据你的喜好重写toString方法。

新建一个源文件夹etc，专门用于存放配置文件。编写英雄池的xml文件——heroPool.xml。
在配置中加入一个英雄的具体信息：鲁班七号、 射手、 嘻嘻，成功检测到对面的智商，看来我无法发挥全部实力了。

创建包com.spring.test，新建测试类TestHero，读取heroPool.xml，然后打印出鲁班七号的具体信息。