Spring

中文意思：春天

但是：Spring框架与春天没有半毛钱的关系。

Spring出现之前，用的框架是EJB。

EJB这东西，真的不是一个东西。

难用至极，用的你头皮发麻。

EJB是冬天，Spring出现了，春天来了。

至于什么是EJB，如果面试问到你，就说：不知道。

一、什么是Spring

Spring是一个轻量级的控制反转(IoC)和面向切面(AOP)的容器框架。

轻量

从大小与开销两方面而言Spring都是轻量的。完整的Spring框架可以在一个大小只有1MB多的JAR文件里发布。并且Spring所需的处理开销也是微不足道的。此外，Spring是非侵入式的：典型地，Spring应用中的对象不依赖于Spring的特定类。

非侵入式

低耦合

什么叫侵入?

1937年，日本狗咬人。所谓的侵略，不是你的东西你要拿，不是你的地方你要占。

对应代码中，侵入式就是：高耦合度。

非侵入式：低耦合。

控制反转IOC

未完待续

面向切面AOP

未完待续

容器

Spring是整个项目的Bean工厂和Bean容器。

所谓的容器，就是一个集合。

Spring是一个Bean的工厂和容器，意思就是：在java项目中，Spring管理了所有的对象的创建和销毁。

管理了所有的Bean的生命周期。所有的bean的对象，都是有Spring来创建。

在Spring框架中，你看不到“new”关键字。

二、控制反转IOC

Spring是一个容器，这个容器指的就是：ioc容器。

IOC的全称：Inversion Of Control 控制反转

关键字：反转。

问题：

1、反转了什么?

控制权的反转。
什么控制权？创建对象的控制权。

举例：
有两个类：A、B
A类中有方法：test，B类中有方法fun，都是非静态的。
现在要在A类的test方法中，调用B类的fun方法。

山顶洞人的做法：

```
class A{
    test(){
        B b = new B();
        b.fun();
    }
}
```

分析这个代码：
B对象的创建，主动权掌握在A手上。
此时：A依赖于B。当B发生改变的时候,A极有可能也会跟着改变。
这就是高耦合。
这是一种不健康的设计

封建社会：
为了降低A和B的耦合度，引入了：工厂模式

```
class BFactory{
    public static B getBInstance(){
        return new B();
    }
}
```

现在要在A类的test方法中，调用B类的fun方法。

```
class A{
    test(){
        B b = BFactory.getBInstance();
        b.fun();
    }
}
```

分析代码：

此时一定程度上降低了A和B的耦合度。

但是，还是存在很大的依赖性。

B对象的创建，此时还是掌握在A手上，A握有主动权。

现代社会：

信息化时代的代码：

```
class A{
    B b;
    test(){
        b.fun();
    }
}
```

如果上述代码可行，此时，A和B的耦合度，降到了最低。

看了上面的代码，有的同学会问：

这肯定会报错，NullPointerException，空指针。

没错，如果只看上面的代码，一定是空指针。

但是：Spring闪亮登场。

Spring的IOC容器，就是解决了上面的问题。让我们的代码来到了信息化时代。

大大的降低了耦合度。

2、从哪里转到了哪里?

三、Spring入门案例

1、导包

导入相关的jar包

2、在web.xml中配置Spring的核心监听器

在web.xml中配置Spring的核心监听器，用来初始化Spring的容器

```
    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
```

这样配置，默认会找/WEB-INF/applicationContext.xml配置文件。

当然，我们也可以不默认，自己指定：在web.xml中加入如下配置

```
    <context-param>
        <!--contextConfigLocation是固定的，不能随意改-->
        <param-name>contextConfigLocation</param-name>
        <!--指定配置文件的路径-->
        <param-value>classpath:spring.xml</param-value>
    </context-param>
```

3、提供Spring的配置文件

默认在/WEB-INF/applicationContext.xml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"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.3.xsd
        http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
        http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
        http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">


    <!--配置Servie的Bean，由于controller是通过注解来配置的，所以Controller不需要在xml中配置--->
    <bean id="stuService" class="com.psfd.spring.service.impl.StuServiceImpl">
    </bean>
</beans>
```

4、使用

```
package com.psfd.spring.controller;
```

```java
import javax.annotation.Resource;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

import com.psfd.spring.domain.Student;
import com.psfd.spring.service.IStuService;

@Controller
public class StuController {

    //@Resource注解，就是用来注入对象。
    @Resource
    private IStuService stuService;

    @RequestMapping("/addStu.do")
    public ModelAndView addStu(Student stu) {
        System.out.println(stu);

        stuService.addStu(stu);

        ModelAndView modelAndView = new ModelAndView();
        modelAndView.addObject("userName", "adminmazi");
        modelAndView.setViewName("welcome");

        return modelAndView;
    }
}
```