# Spring 入门

首先需要下载俩个包

1.核心包　　spring-framework-3.2.0.RELEASE

2.依赖包　　spring-framework-3.0.2.RELEASE-dependencies

创建web项目

解压核心包进入libs目录导入四个核心jar包

1.spring-beans-3.2.0.RELEASE.jar

2.spring-context-3.2.0.RELEASE.jar

3.spring-core-3.2.0.RELEASE.jar

4.spring-expression-3.2.0.RELEASE.jar

将以上四个jar包复制到eclipse中的WebContent -> lib目录下

解压依赖包找到：

org.apache.commons ->

com.springsource.org.apache.commons.logging ->

1.1.1 ->

com.springsource.org.apache.commons.logging-1.1.1.jar

也添加到lib目录下

注意：

*-3.2.0.RELEASE.jar 所需的jar包

*-3.2.0.RELEASE-javadoc.jar 文档

*-3.2.0.RELEASE-sources.jar 源码

# IOC (控制反转)

原本创建对象是自己的事情，现在反转给spring，让spring来做这件事情

以下就是pring

## 目标类：

对象类如：UserServlet

## 配置文件：

位置：任意　开发中一般在classpath（src目录下）

名称：配置文件名任意..一般名为：applicationContext.xml

内容：添加 schema 约束

约束格式在此文件内：

spring-framework-3.2.0.RELEASE-dist\

spring-framework-3.2.0.RELEASE\

docs\

spring-framework-reference\

html\

xsd-config.html

双击然后下滑复制一段约束

```
<?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 definitions here -->

</beans>
```

创建配置文件 applicationContext.xml

将以上代码复制粘贴到 applicationContext.xml 中

若想让spring给我们创建对象得向spring提供对应的类，与配置信息

## 在.xml中添加标签

```
<bean id="用于后期用户取对象实例的一个标识 “ class="需要被创建对象实例的全限定类名"></bean>
```

这样我们就可以通过指定的id 或 name 获取对象了

```
//在需要创建对象界面通过spring创建对象
ApplicationContext context = new ClassPathXmlApplicationContext("放开始创建xml的全限定地址");
//全限定名指的是从包名开始如："com/znsd/servlet/applicationContext.xml"
//通过 ApplicationContext 获取到xml中配置的id或name
```

```
UserService user = (UserService)context.getBean("放id或name名称");
//使用对应的类型接收，需要强转 这就让spring帮我们创建了一个UserService 对象了!
```

# DI（依赖注入）

1. 依赖：

一个类需要使用另外一个类的数据就称为依赖

2. 注入：

一个类中需要另一个类的数据，然后通过调用set将另一个类的数据添加到类中称为注入

举例：

```
class Dao{
    private int id;
}
class UserSerletImpt{
    //平常
    private Dao dao = new Dao(); //也可以通过构造方法或者set传入这里不写了
    //注意spring的核心 IOC（控制反转） AOP（切面编程） 一切对象皆有spring创建
    //使用spring
    private Dao dao;
    //此处有set 或者 构造（省略不写） 通过spring 创建的对象传入
    public void setDao(Dao dao){
        this.dao = dao;
    }
}
//使用spring 创建 对象 并传入
class main{
    public static void main(String[] args){
        UserServiceImpt userImpt = new UserServiceImpt();
        Dao dao = new Dao();
        userImpt.set(dao);
        /*
            这就是一个平常的举例
        */
    }
    public static void main(String[] args){
        Application cation = ClassPathXmlApplication("xml的全限定地址");
        UserSerletImpt userImpt = cation.getBean("UserSerletImpt");//存放的是xml配置的id或name
        Dao dao = cation.getBean("Dao");//存放的是xml配置的id或name并非类名
        userImpt.set(dao);
```

```
    /*
       这就是一个Ioc + DI举例
          Ioc --> <bean>
          DI --> <poperty> poperty 是 bean 的一个 子标签
          就类似于 bean = 对象  poperty = 属性
    */
  }
}
```

而spring 不需要 用户自己调用set 方法 可以通过配置文件将数据注入到该类中

## 注:

Ioc --> <bean>

DI --> <property> property 是 bean 的一个 子标签

就类似于 bean = 对象  poperty = 属性

举例:

Ioc <bean>创建完对象，对象中有未复制的对象，可以使用DI <property>直接注入

这样用户就不需要调用set了

## 配置文件:

### 在.xml中添加Poperty标签

```
<bean id="UserSerletImpt " class="com.znsd.service.UserSerletImpt">
  //setDao -> Dao -> dao  set 演化为 属性
  /*
     对象中需要添加多少个其他类对象属性就添加多少个poperty 并指名ref 这样可以在使用时不需要调用set
  */
  <poperty name="dao" ref="Dao"></poperty>  //name存放属性名
</bean>
<bean id="Dao " class="com.znsd.service.Dao"></bean>
```

## Ioc + DI 举例:

```
class Dao{
  private int id;
}
class UserSerletImpt{
  private Dao dao;
  public void setDao(Dao dao){
     this.dao = dao;
  }
}
//使用spring 创建 对象 并传入
```

```
class main{
    public static void main(String[] args){
        ApplicationContext cation = ClassPathXmlApplicationContext("xml的全限定地
址");
        UserSerletImpt userImpt = cation.getBean("UserSerletImpt");//存放的是xml配置
的id或name
        /*
            这就是一个Ioc + DI举例
                Ioc --> <bean>
                DI --> <poperty> poperty 是 bean 的一个 子标签
                就类似于 bean = 对象  poperty = 属性

                Application cation = ClassPathXmlApplication("xml的全限定地址");
                执行这一行时会首先初始化xml配置文件
                <bean id="UserSerletImpt “ class="com.znsd.service.UserSerletImpt">
                读取这一行时创建UserSerletImpt对象
                <property name="dao" ref="Dao"></property>
                读取这一行代表UserSerletImpt需要Dao对象
                出去再读取
                <bean id="Dao “ class="com.znsd.service.Dao"></bean>
                读取完之后创建Dao对象
                然后回到
                <property name="dao" ref="Dao"></property>
                将ref 放入 UserSerletImpt 对象中 的name 属性中
                name 必须为类中需要注入的属性名
                底层通过反射
                //dao -> Dao ->setDao set演化为 set方法
                若类中没有set方法则抛出异常
        */
    }
}
```