Maven

Maven：软件管理工具，管理项目之间的相互依赖关系。

Maven环境的安装：

1.下载并解压Maven。

2.配置环境变量（M2_HOME，Maven的路径，Path）。

3.设置Maven的配置文件（本地仓库路径，远程仓库路径）。

- POM（Project Object Model）是一个xml文件，维护配置信息，jar依赖在该文件中配置。
- Dependency Managerment 设置依赖关系。
- Corrdinates Maven是一个jar包仓库，软件开发商将自己的产品提交到jar包仓库中，开发者直接从jar包仓库中下载需要的jar，通过坐标确定具体的jar包。
    - groupId
    - artifactId
    - version
    - packaging（可缺省）

Spring

Spring是一个企业级开发框架，Java领域公认的第一框架，为解决企业级项目开发过于复杂而创建的，框架的主要优势之一就是分层架构，允许开发者自主选择组件。MVC（Sturts2，Spring MVC）---Repository（Hibernate、MyBatis、Spring JPA、Sprint Data）

Spring的两大核心机制 IoC（控制反转）和 AOP（面向切面编程），从开发者的角度来讲，我们使用Spring框架就是用它的IoC和AOP。IoC是典型的工厂模式，通过工厂来注入对象，AOP是代理模式。

IoC是Spring框架的基石，IoC也叫控制反转，传统的开发方式中，需要调用对象时，需要手动来创建对象，即对象是由调用者主动创建出来的。

但在Spring框架中创建对象的工作不再由开发者完成，而是交给IoC容器来完成，IoC容器主动创建好对象，推送给开发者来调用，相对于传统的开发方式来讲，对象的创建过程完全反转，所以叫做控制反转。

传统方式

- 创建Student类

```
package com.southwind.entity;
```

```
public class Student {
    private int id;
    private String name;
    private int age;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }
}
```

- 通过Student的构造函数来创建对象

```
public class Test {
    public static void main(String[] args) {
        Student student = new Student();
        System.out.println(student);
    }
}
```

通过IoC容器来创建对象

- 搭建Spring环境

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
```

```
    <version>4.3.7.RELEASE</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aop</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-aspects</artifactId>
    <version>4.3.7.RELEASE</version>
</dependency>
```

- 创建IoC配置文件，可以自定义名称，spring.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.springframework.org/schema/context"
       xmlns:p="http://www.springframework.org/schema/p"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
    http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
">

    <bean id="student" class="com.southwind.entity.Student"></bean>

</beans>
```

- 调用API获取IoC创建好的对象。

```
//加载spring.xml配置文件
ApplicationContext applicationContext = new
ClassPathXmlApplicationContext("spring.xml");
//获取IoC中的对象
Student student = (Student) applicationContext.getBean("student");
System.out.println(student);
```

注意，通过IoC容器来管理对象时，对象对应的实体类必须有无参构造函数，否则IoC无法实例化对象，因为IoC底层是通过反射机制来创建对象的，反射机制默认调用实体类的无参构造函数。