第1章　概述

1.1　实践目标

本教材围绕教育部职业教育“1＋X”技能等级证书项目——Java Web 方向中级内容进行设计，采用以知识点为索引，以项目贯穿的形式，综合训练读者 Java Web 开发应用能力。通过学习和实践本教材中提供的知识点和项目，希望实现以下实践目标：

(1)掌握 JDK 的安装。

(2)掌握 IDEA 的安装与使用。

(3)掌握 Maven 搭建 Java 框架项目环境。

(4)理解 Spring 核心容器的原理，掌握 BeanFactory 和 ApplicationContext 接口的基本使用。

(5)理解 Spring 框架核心 IoC 的原理，掌握使用配置文件和注解的方式操作 Bean，包括 Bean 的创建、管理和属性注入等。

(6)理解 Spring 框架 AOP 动态代理机制，掌握配置文件和注解的方式配置 AOP 的切入点和通知的应用。

(7)了解 Spring 框架对数据持久层的支持实现，掌握 Spring 框架对数据库连接池的配置和使用。

(8)理解 Spring MVC 框架工作原理及核心组件组成，包括 DispatcherServlet、HandlerMapping、ViewResolver、Controller 和 ModelAndView 等。

(9)掌握 Spring MVC 框架注解的使用，实现前后台页面跳转和前后台参数传递，以及 JSON 格式数据的传递。

(10)了解 Spring MVC 框架的数据校验、异常处理、拦截器和国际化等。

(11)理解 MyBatis 框架的基本原理、功能架构和工作原理，并能够进行 MyBatis 开发环境的配置。

(12)掌握 MyBatis 框架核心配置文件的使用，包括 properties、settings、typeAliaces、typeHandlers、plugins、environments 和 mappers 等标签元素的应用。

(13)掌握 MyBatis 框架映射文件的使用，包括数据的增、删、改、查标签的应用。

(14)理解 MyBatis 框架的关联映射，掌握使用 resultMap 对于级联关系的使用。

(15)掌握 MyBatis 框架动态 SQL 语句的使用，包括 if、choose-when-otherwise、trim、

where、foreach 等标签元素的应用。

(16)掌握 SSM 框架的整合过程,各配置文件的配合使用,以及项目的布局等。

(17)掌握使用 SSM 框架进行项目编程,以及真实工作中项目的构思、设计、实施和运行过程。

(18)遵循企业 Java Web 标准设计和开发过程,培养良好的工程能力,提高 PC 端动态网站开发的实践能力,达到 Java Web 中级开发工程师水平。

1.2 实践知识图

1. SSM 框架知识图

本教材主要讲解了 SSM 框架的应用,包括 Maven 搭建框架项目环境;Spring 框架的 IoC 和 AOP;Spring MVC 框架的原理(各组件 DispatcherServlet、HandlerMapping、HandlerAdapter、ViewResolver 和 ModelAndView)、注解和前后台数据传递;MyBatis 框架配置文件、映射文件和动态 SQL 的使用。如图 1.1 所示。

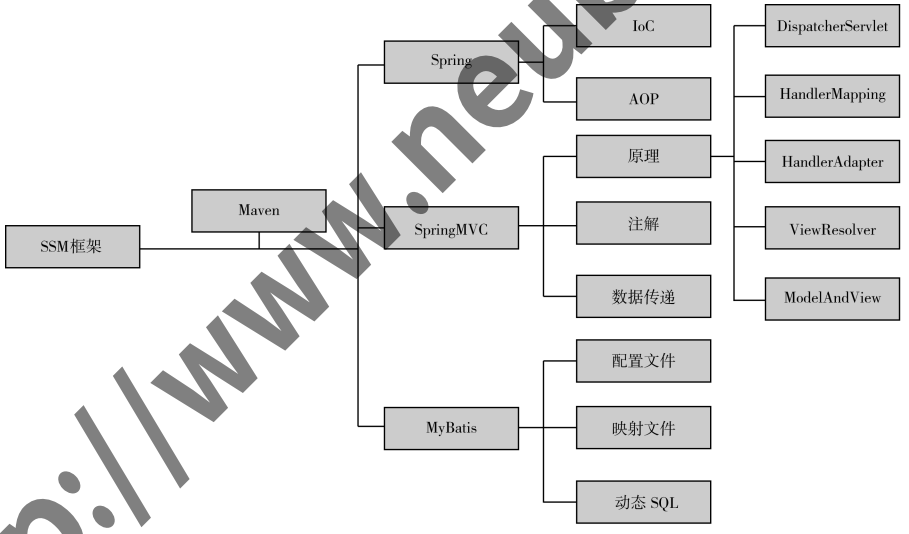

图 1.1 SSM 知识结构图

2. Spring 框架知识图

Spring 框架主要讲解 Spring 框架工作原理、Spring 框架核心 IoC、Spring 框架核心 AOP、Spring 框架程序架构、面向接口编程等,如图 1.2 所示。

Spring:轻量级框架。

IoC:Inversion of Control,即控制反转,把对程序的控制权交给 Spring 容器。

依赖注入:给属性注入值的时候需要依赖其他 Bean。

AOP:Aspect-Oriented Programming,面向切面编程。

Spring 核心容器打开方式有两种:BeanFactory 和 ApplicationContext。

创建 Bean 方式有三种:构造器方式、普通工厂方式和静态工厂方式。

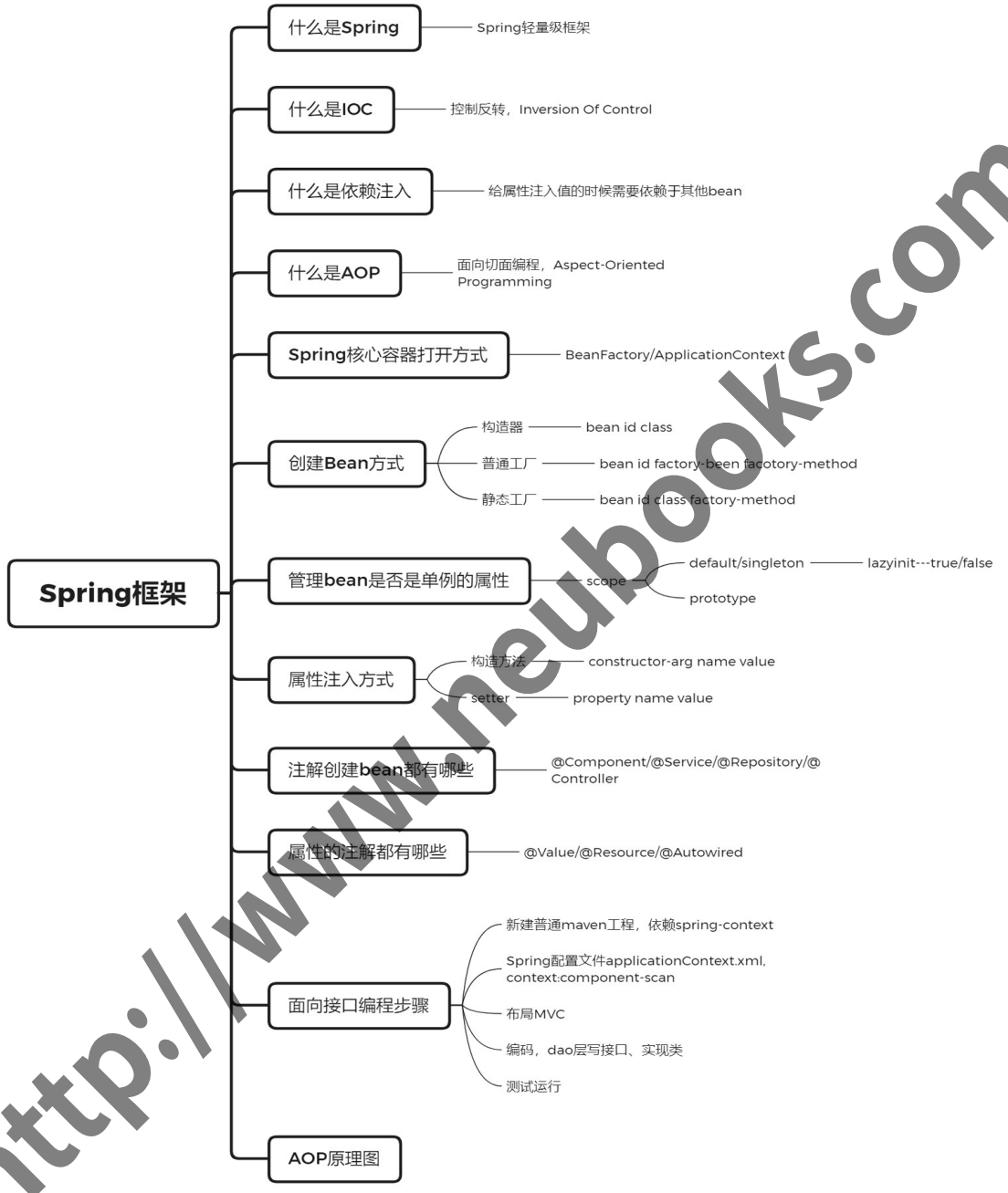

图 1.2　Spring 框架知识图

管理 Bean 是否是单例的属性是 scope，当 scope 值为 default 或者 singleton 时，是单例模式，此时可以通过 lazyinit 属性来设置 Bean 的加载时机；当 scope 值为 prototype 时，是非单例模式。

属性注入方式有：构造器方式注入和 setter 设置注入方式。

可以创建 Bean 的注解有：@Component、@Service、@Repository、@Controller。

给属性注入值的注解有：@Value、@Resource、@Autowired。

面向接口编程的步骤：新建普通 Maven 工程，添加 spring-context 依赖→Spring 配置文件 applicationContext.xml 中通过 context:component-scan 配置支持 Spring 注解→布局 MVC 架构模式→编码，dao 层写接口、实现类（面向接口编程）→测试运行。

AOP 有两套方案：前置通知——最终通知/异常通知——后置通知；环绕通知。

3. Spring MVC 框架知识图

SpringMVC 框架主要讲解 Spring MVC 框架工作原理、各个组件的应用、使用注解完成前后端页面跳转和前后端数据传递等，如图 1.3 所示。

Spring MVC 框架原理：客户端发送请求，后台在 web.xml 配置文件里面注册 DispatcherServlet，首先通过拦截器拦截客户端请求，DispatcherServlet 查找相应的 HandlerMapping 对请求进行数据封装，封装成数据对象；DispatcherServlet 查找 HandlerAdapater，通过它查找到对应的控制器进行业务处理（controller-service-dao），返回 ModelAndView 对象；DispatcherServlet 查找 ViewResolver 对 ModelAndView 进行解析，生成对应的 ModelAndView 对象，Model 渲染 View；最后 DispatcherServlet 把 View 响应返回客户端；

Spring MVC 框架核心组件：

DispatcherServlet：调度中心，在 web.xml 里面注册；

HandlerMapping：映射处理器，把数据请求封装成数据对象；

HandlerAdapter：适配器处理器，查找对应的 controller；

ModelAndView：数据对象，封装 Model 对象和 View 对象；

ViewResolver：视图解析器，解析视图。

Spring 常用注解：

@RequestMapping：匹配 url，放到类、方法上面；

@RequestParam：匹配请求参数，当前台表单 name 和后台参数名不一致时使用；

@ModelAttribute：把后台参数使用 model 绑定回前台；

@ResponseBody：返回 JSON 格式数据。

Spring MVC 注解原理：客户端发送请求，服务器端通过拦截器拦截请求，web.xml 里面注册 DispatcherServlet，DispatcherServlet 查找 HandlerMapping，帮助把拦截到的数据封装成同名的数据对象，DispatcherServlet 通过 HandlerAdapater 查找用户自定义的 controller（@RequestMapping），把数据对象进行业务处理，生成 ModelAndView 对象，DispatcherServlet 查找 ViewResolver，把 ModelAndView 对象解析成 View，DispatcherServlet 响应返回客户端。

前后台参数传递，后台 Controller 的参数数据类型：HttpRequest/HttpResponse、String/Integer、Person（用户自定义类）、Model/Map/ModelMap；后台 Controller 返回值的数据类型：ModelAndView、String、void/其他数据类型（@ResponseBody 注解）。

查看用户案例中，后台参数传递到前台，前台使用 EL+JSTL 解析。

修改用户案例中，前台使用 Ajax 实现异步传输，并在前台解析后台传递回来的数据，渲染到前台。

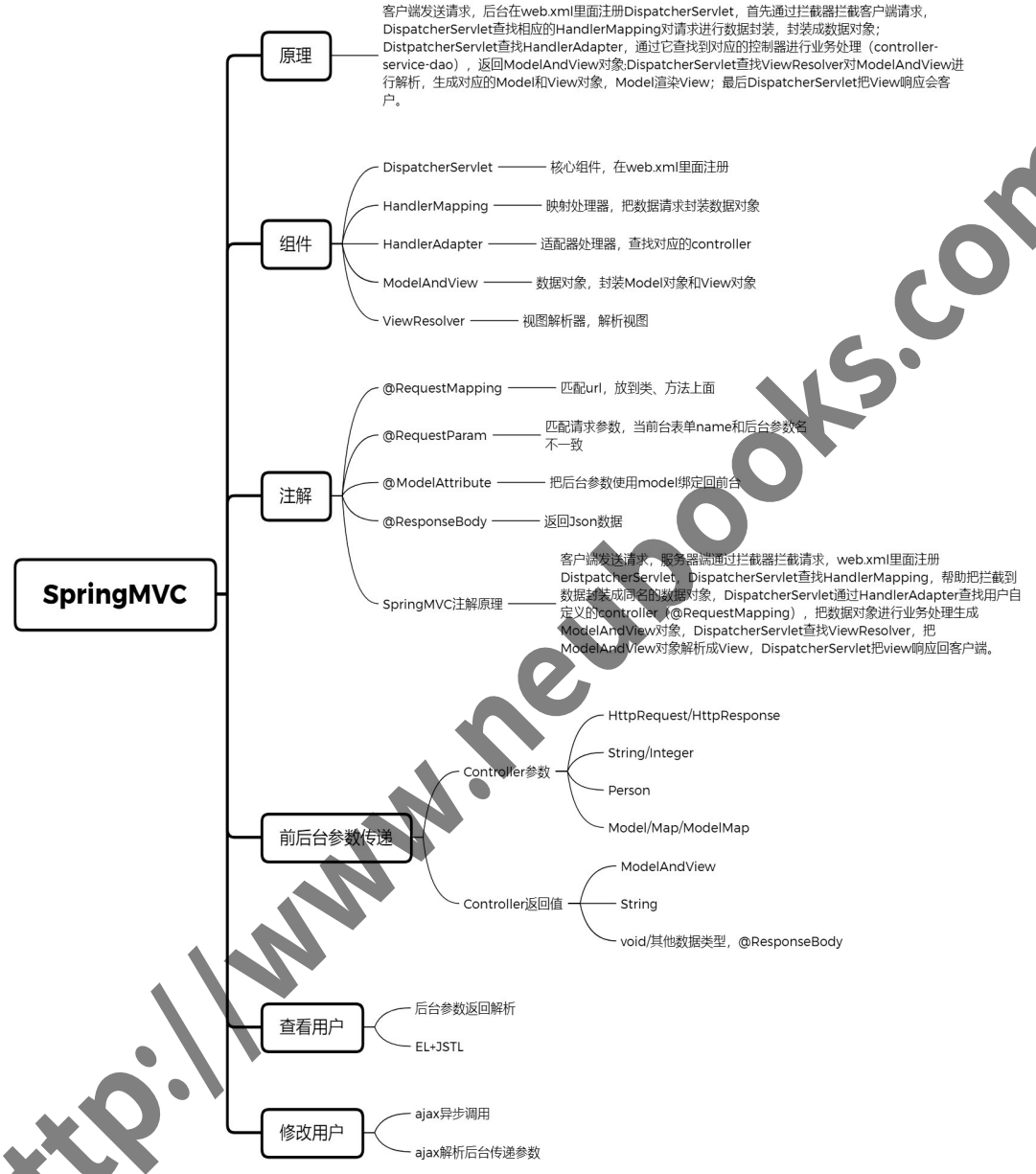

图 1.3　Spring MVC 框架知识图

4. MyBatis 框架知识图

MyBatis 框架主要讲解了 MyBatis 配置文件、映射文件的基本使用，高级部分讲解了 MyBatis 的关联映射和动态 SQL 等。如图 1.4 所示。

MyBatis 框架是数据持久层框架。

MyBatis 框架搭建步骤：新建 Maven 工程→在 pom.xml 配置文件添加 MyBatis 和 MySQL 相关依赖→MySQL 数据库建表→新建并填写 MyBatis 配置文件→项目布局→编码→测试运行。

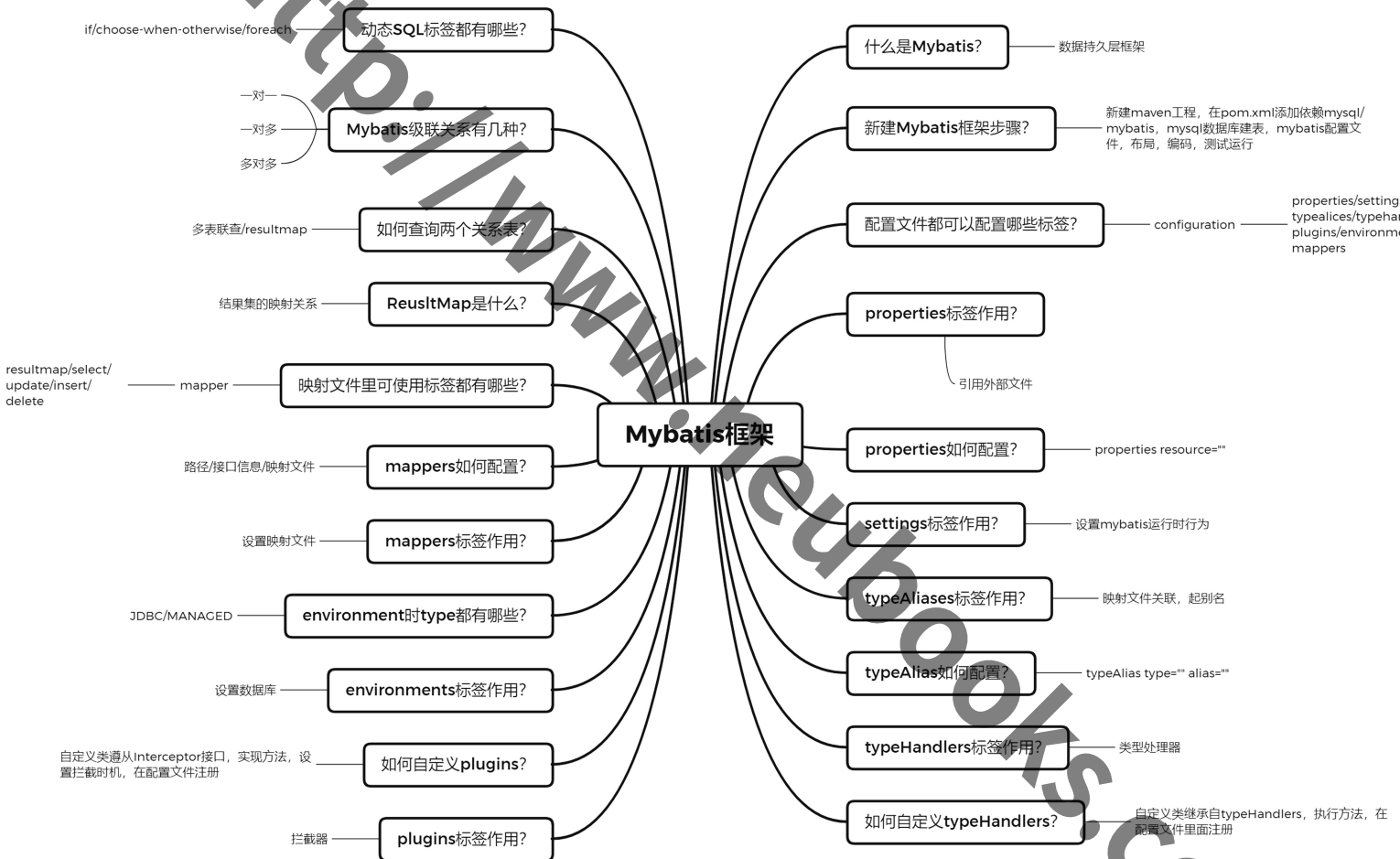

图1.4 Mybatis框架知识图

MyBatis 框架配置文件在 configuration 里可以配置如下标签：properties、settings、typealices、typehanlders、plugins、environments、mappers。

properties 标签的作用：引用外部文件。

settings 标签作用：设置 MyBatis 运行时行为。

typeAliases 标签作用：映射文件关联，起别名。

typeHandlers 标签作用：类型处理器。

如何自定义 typeHandlers：自定义类需要继承自 typeHandlers，执行方法，在配置文件里面注册。

plugins 标签作用：拦截器。

如何自定义 plugins：自定义类需要遵从 Interceptor 接口，实现方法，设置拦截时机，在配置文件注册。

environments 标签作用：设置数据库。

environment 设置时 type 类型值：JDBC、MANAGED。

mappers 标签作用：设置映射文件。

mappers 如何配置：可以设置路径、接口信息、映射文件。

映射文件 mapper 里可使用如下标签：resultMap、select、update、insert、delete。

resultMap 标签作用：结果集的映射关系。

如何查询两个关系表：多表联查、resultMap。

MyBatis 框架级联关系：一对一关系、一对多关系、多对多关系。

动态 SQL 标签都有哪些：if、choose-when-otherwise、foreach 等。

1.3　实践安排

本教材围绕教育部职业教育“1＋X”技能等级证书项目——Java Web 方向中级内容进行设计，中级主要涵盖 Spring、Spring MVC、MyBatis 三大主流框架内容，以及复杂数据库开发与应用内容。

本教材分为三部分：项目准备篇、基础知识篇和综合项目篇。项目准备篇是搭建项目环境，使用 Java 框架开发需要准备 JDK 环境和 IDEA 开发工具，使用 Maven 构建框架环境；基础知识篇是以知识点为索引，以项目贯穿的形式，通过具体应用加深对知识点的理解；综合项目篇是把 Spring、Spring MVC 和 MyBatis 三个框架进行整合，并使用整合后的 SSM 框架进行项目编程。

综合项目篇的项目是人员模块管理系统，主要是使用整合后的 SSM 框架实现人员模块的增、删、改、查，从前台页面到后台进行操作，把数据存储到数据库。人员的增、删、改、查包括用户登录、用户注册、查看用户、修改用户和删除用户。本教材在基础知识篇讲解每个框架的时候，把人员管理项目进行拆分，以拆分后的项目为导引，通过具体案例学习知识点的应用。

比如，用户登录实际上就是对人员模块的查找操作，如果数据库中存在该用户，则允许登录；否则不允许登录。学习 Spring 框架时，在 IoC 部分通过登录案例练习对 Bean 的操作；

学习 Spring MVC 框架时，通过登录案例练习前后台参数传递、注解的使用等；学习 MyBatis 框架时，通过登录案例练习配置文件和映射文件的使用等。具体如图 1.5 所示。

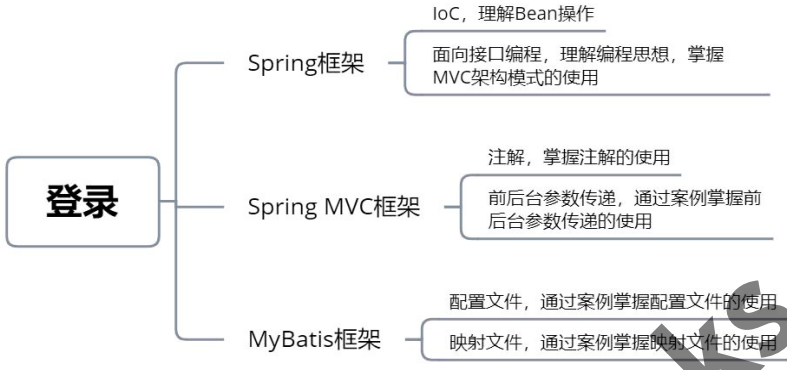

图 1.5　登录案例在每个框架中的应用

通过案例的重复使用，让学生在忽略案例本身的业务需求的同时，更注重技能需求，掌握每个知识点的不同与具体应用，加深对于知识点的理解与运用。

本教材在基础知识篇的案例应用和知识点描绘见表 1.1。

表 1.1　　教材实践案例导引

项目名称	项目内容	应用知识点	对应标准
Maven 部署 Junit 测试环境	使用 Maven 搭建项目，配置 Junit 单元测试环境	2.2.2 Maven 配置文件 pom.xml 2.2.3 Maven 资源库	掌握 Maven 搭建 Java 框架项目环境
给 Person 类注入属性值	定义 Person 类，有各种类型的属性，使用手动注入方式给属性注入值，并测试	4.1 Spring 操作 Bean 4.2.1 构造器方法 4.4.1 手动注入属性值	Spring 操作 Bean
Spring 框架实现用户登录	使用 Spring 框架实现用户登录	3.3 Spring 框架体系结构 3.4 Spring 项目布局 3.5 Spring 核心容器 4.2 创建 Bean 4.3 管理 Bean 4.4 依赖注入 4.5 Spring 注解 5.1 面向接口编程 5.2 Spring 的三层架构模式	Spring 注解
Spring 框架实现 AOP	以配置文件方式通过前置、后置、最终、异常通知完成面向切面编程	6.1 认识 AOP 6.2.1 前置、后置、异常、最终通知 6.2.4 切入点表达式	AOP 配置及应用
Spring MVC 框架实现用户登录	Spring MVC 框架完成用户登录	7.1 Spring MVC 工作原理 7.2 Spring MVC 核心类	Spring MVC 工作原理

（续表）

项目名称	项目内容	应用知识点	对应标准
Spring MVC 框架注解实现用户登录	使用 Spring MVC 框架注解方式实现用户登录案例	8.2 @RequestMapping 注解 8.5 前后台传递参数	处理器映射器 Controller
MyBatis 框架查询数据库	使用 MyBatis 框架查询数据库表 person 中的数据信息	9.2 properties 元素 9.7 environments 元素 9.8 mappers 元素	MyBatis 入门
MyBatis 框架对数据库进行增、删、改、查	使用 MyBatis 框架完成数据表的增、删、改、查	10.2 命名空间 10.3 select 元素 10.4 insert、update、delete 元素	MyBatis 配置文件
MyBatis 框架实现多表联查	MyBatis 框架实现多表联查	11.1 resultMap 11.2 多级联关系	级联关系
MyBatis 框架实现模糊查询	MyBatis 框架使用动态 SQL 实现模式搜索	12.2 if 元素 12.5 where 元素	动态 SQL
SSM 框架实现用户登录	使用 Spring MVC 框架连接前台页面和后端服务器，使用 Spring 框架完成 JavaBean 创建和管理，使用 MyBatis 框架完成数据持久层操作，整合 SSM 框架，完成用户登录操作	2.1.3 使用 IDEA 编程 2.2.2 Maven 配置文件 pom.xml 2.2.3 Maven 资源库 3.4 Spring 项目布局 3.5 Spring 核心容器 4.1 Spring 操作 Bean 4.3 管理 Bean 4.4 依赖注入 4.5 Spring 注解 5.1 面向接口编程 5.2 Spring 三层架构模式 7.1 Spring MVC 框架工作原理 7.2 Spring MVC 核心类 8.1 Sprnig MVC 框架常用注解 8.2 @RequestMapping 注解 8.5 前后台数据传递 8.7 @ResponseBody 注解 8.8 Ajax 操作 JSON 数据 9.1 MyBatis 框架配置文件 9.4 typeAliaces 元素 10.1 MyBatis 框架映射器 10.2 命名空间 10.3 select 元素	Spring 核心容器 Spring 操作 Bean Spring 注解 Sprig MVC 工作原理 处理器映射器 Controller 实现应用操作 MyBatis 入门 MyBatis 配置文件 级联关系

第 2 章　开发环境的搭建

本章知识点

掌握 JDK 和 IDEA 的安装及使用；

掌握 Maven 两要素的原理及使用：配置文件和资源库。

所支撑的职业技能

能够使用 Maven 搭建 Java 框架项目环境，以及能够使用 Spring 整合 Junit 进行单元测试。

2.1　开发环境

2.1.1　安装 JDK

JDK(Java SE Development Kit)需要 1.8 及以上版本，可以从 Java 的官网下载安装包。如果访问官网速度慢的话，也可以通过百度搜索 JDK，然后在百度软件中心下载符合要求的 Windows 版本和配置的 JDK1.8 安装包。

下载后点击下一步进行安装即可。安装完成后，配置环境变量 JAVA_HOME，例如，使用路径 C:\Program Files\Java\jdk1.8.0_65(如果你安装的是这个目录的话)。JAVA_HOME 配置好之后，将%JAVA_HOME%\bin 加入系统的环境变量 path 中。完成后，打开一个命令行窗口，输入命令 java - version，如果能正确输出版本号则说明安装成功了。输出版本的信息如下：

```
C:\Users\NEUSOFT>java-version
java version "1.8.0_65"
Java(TM) SE Runtime Environment (build 1.8.0_65-b17)
Java HotSpot(TM) 64-Bit Server VM (build 25.65-b01, mixed mode)
```

【注意】

(1)可以在网上搜索“JDK 的下载和安装”，会出现很多详细的文档。

(2)事实上，配置环境变量 JAVA_HOME 只是为了让系统找到 JDK 的路径，你也可以

不配置该环境变量，而在系统 path 中直接配置 C:\Program Files\Java\jdk1.8.0_65\bin 目录，如图 2.1 所示。

(3)验证 JDK 是否安装并配置成功，可以在命令行窗口进行验证，当输入 java 提示“‘java’不是内部或外部命令，也不是可运行的程序或批处理文件”时，说明 JDK 没有安装成功，此时需要重新安装 JDK；如果输入 java 好用(显示 java 命令使用方法)而输入 Javac 不好用(显示“‘javac’不是内部或外部命令，也不是可运行的程序或批处理文件”)，说明 JDK 已经安装，但是环境变量配置不成功，此时需要在环境变量 path 里面添加 JDK 的路径。

(4)配置环境变量的时候需要保证两点：JDK 的路径要正确配置到 bin 文件夹下；其次注意更改的是系统环境变量。

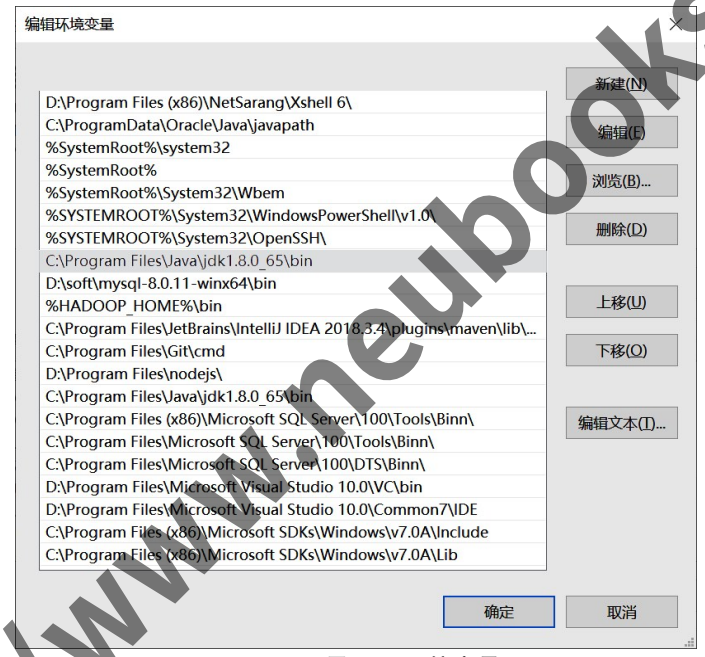

图 2.1　配置 JDK 环境变量

2.1.2　安装 IDEA

IDEA 全称 IntelliJ IDEA，是 JetBrains 公司推出的 Java 编程语言集成开发环境。

IDEA 最突出的功能是调试(Debug)，可以对 Java 代码、JavaScript、jQuery、Ajax 等技术进行调试。

官网下载最新版本 IDEA，如图 2.2 所示，可以使用社区版，如果是专业版本请在试用期内使用或者购买破解版。

本教材使用的具体版本如图 2.3 所示，对于开发人员来说，IDEA 只是一个编辑工具，IDEA 的版本可以不同，企业版本差别不是很大，社区版没有一些框架的图形界面支持，可以手动布局，效果是一样的。

图 2.2 官网下载 IDEA 界面

图 2.3 IDEA 版本界面

下载后，点击下一步进行安装，安装成功后界面如图 2.4 所示。

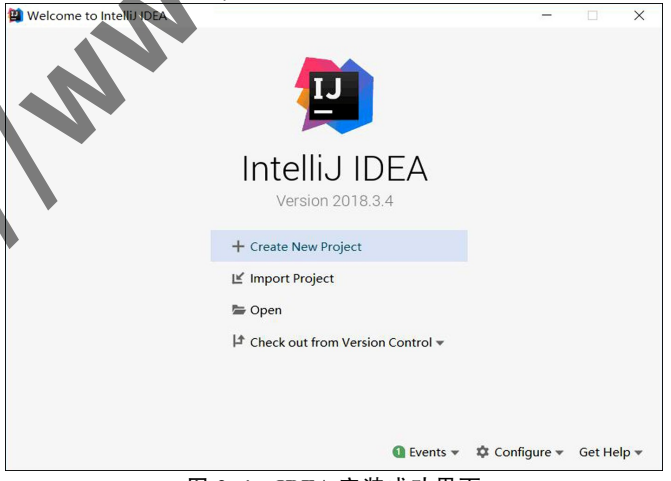

图 2.4 IDEA 安装成功界面

【注意】

(1)使用 IDEA 的前提是本机上面 JDK 已经安装成功。可以在命令行窗口进行验证，当输入 Java 提示“‘java’不是内部或外部命令，也不是可运行的程序或批处理文件”时，说明没有 JDK，此时需要安装 JDK；如果输入 Java 好用而输入 Javac 不好用(显示“‘javac’不是内

部或外部命令，也不是可运行的程序或批处理文件”），说明 JDK 已经安装，但是环境变量配置不成功，此时需要在环境变量 path 里面添加 jdk/bin 的路径。

配置环境变量的时候需要保证两点：JDK 的路径要正确到 bin 文件夹下；更改的是系统环境变量。

(2)安装 IDEA 完成后，可以使用试用版，在试用期内使用；也可以使用学生账号在 Jetbrains 官网注册账号，之后便可以使用教育账号免费使用了；如果没有学生账号，也可以在官网购买使用。

另外，也可以使用社区版(Community)，社区版只是有些模板界面与专业版不相同，使用模板创建完工程后，也可以手动进行创建，最后的结果都是一样的。

2.1.3　使用 IDEA 编程

点击“Create New Project”创建项目，弹出界面如图 2.5 所示。

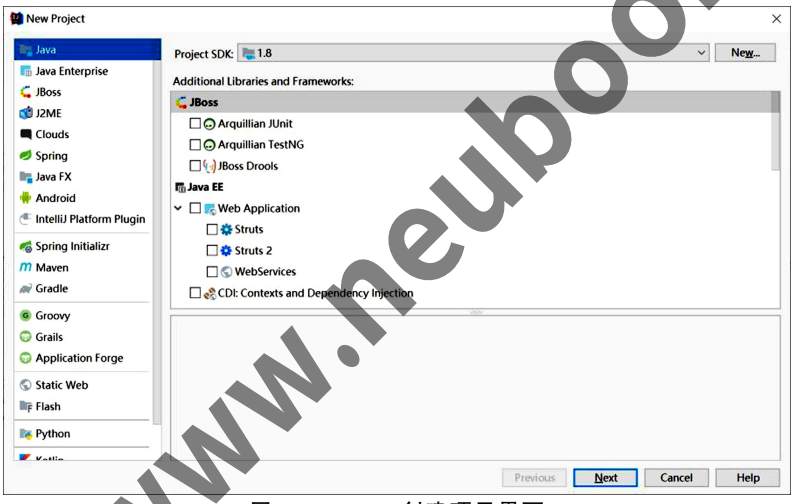

图 2.5　IDEA 创建项目界面

点击“Next”创建 JavaSE 项目，如果想要创建 JavaEE 项目，需要选择 JavaEE 下的“Web Application”，再选择“Next”，此时 IDEA 帮助我们提供 JavaEE 基本模板。基本的 JavaSE 项目和 JavaEE 项目布局如图 2.6 所示。

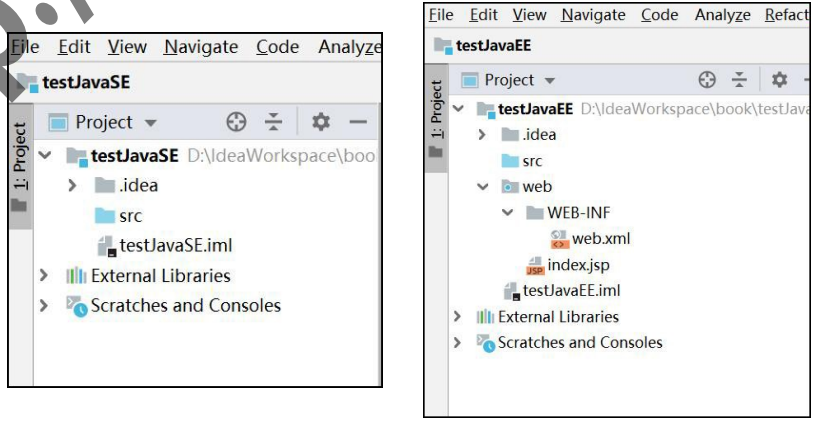

图 2.6　IDEA 项目布局界面

配置 Tomcat 注意点：

编写 JavaEE 项目，运行前需要先配置 Tomcat 服务器。在界面菜单栏右上角点击“Add Configuration”，弹出的界面如图 2.7 所示。

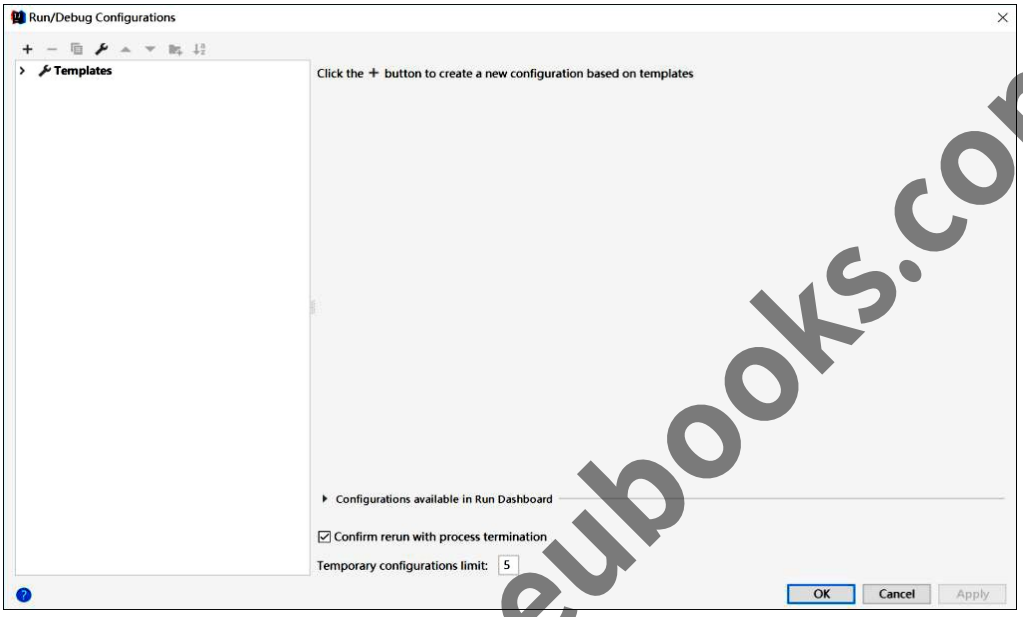

图 2.7　准备配置 Tomcat 界面

首次使用需要点击“Templates”下面“Tomcat Server”，选择“Local”，配置服务器。之后再使用，就可以直接点击左上角的“+”添加 Tomcat 服务器了，同样是选择“Tomcat Server”下面的“Local”，点击后弹出界面如图 2.8 所示。

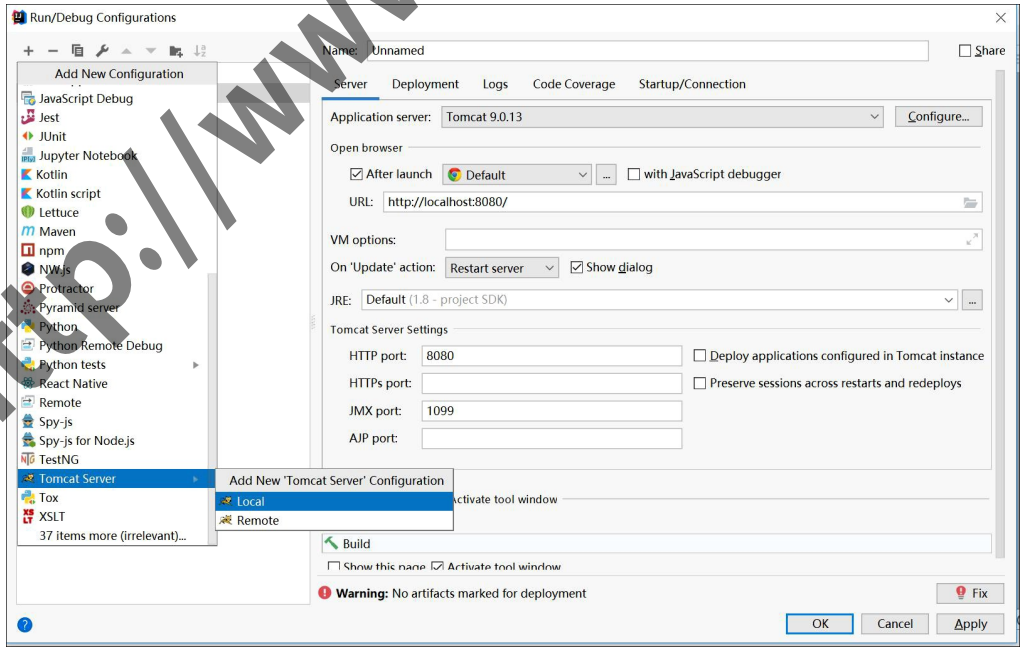

图 2.8　配置 Tomcat 界面

“Name”是 Tomcat 名字，需要用户自定义；“Application Server”是实际添加的 Tomcat，如果你的 Tomcat 匹配不上，则此处报红；Tomcat 配置好后，最下面一行提示红色 Warning，说明你当前的项目没有部署到服务器上面，需要点击“Fix”，把我们的项目部署到服务器上面，如图 2.9 所示，此时点击“OK”后，你的项目就可以运行了。

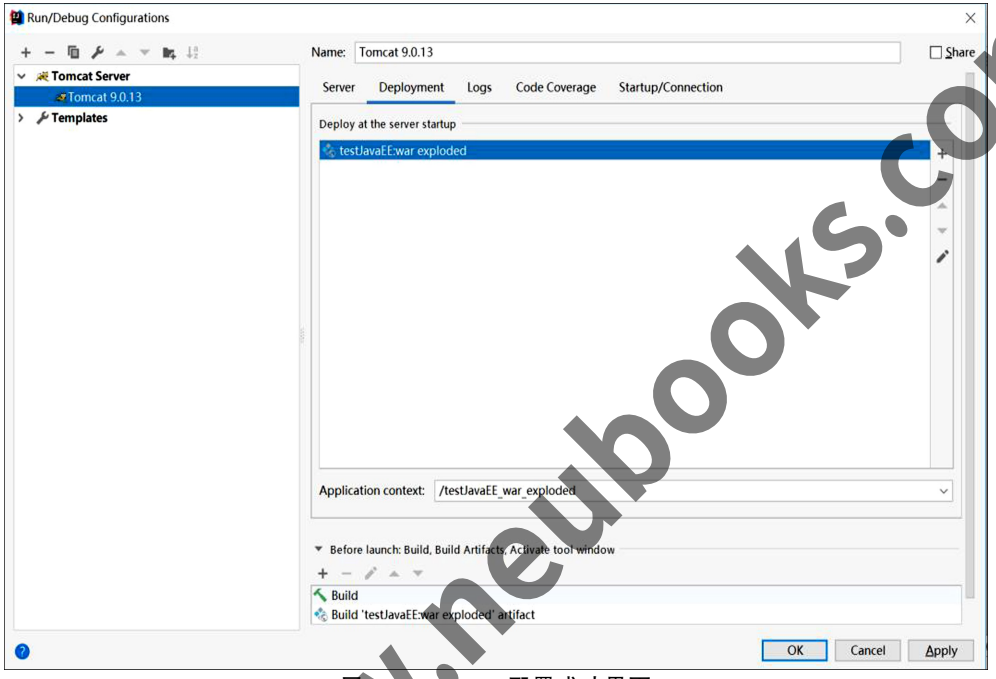

图 2.9　Tomcat 配置成功界面

2.2　Maven 介绍

本教材是使用框架进行编程，框架在布局的时候需要依赖很多 Jar 包。随着每个框架版本的升级，各种 Jar 包与依赖包之间都有差异，导致编写项目过程中遇到很多不必要的麻烦。本教材使用 Maven 搭建项目，避免了因为版本匹配产生的问题。

Maven 是一个项目管理和综合工具（基于项目对象模型 POM）。它帮助开发人员构建一个完整的生命周期框架。Maven 使用标准的目录结构，并且默认构建生命周期。

Maven 是一个工具，也是一个框架。工具的作用是帮助我们查找 Jar 包，搭建项目环境；框架是因为它需要在配置文件中配置才能工作。即 Maven 需要在配置文件中配置依赖，才能帮助填充资源库，安装项目环境。

2.2.1　安装 Maven

Maven 的安装方式有以下两种：

（1）使用 IDEA 自带的 Maven（推荐）。

（2）在官网下载 Maven 二进制压缩包，配置环境变量后直接使用，控制台使用 mvn-version 验

证。如果在 IDEA 里使用本机安装的 Maven，需要在 IDEA 里面配置 Maven 使用路径。

2.2.2 Maven 配置文件 pom.xml

Maven 中最主要的两要素：配置文件 pom.xml 和资源库。

pom.xml 是 Maven 配置文件，默认包括项目的版本、名称等信息。配置文件 pom.xml 里面配置好依赖，系统就会帮助我们到默认镜像地址查找相应的 jar 包及依赖包，并安装到项目中。

可以通过 dependencies 标签配置项目依赖。比如，想要配置 Junit 依赖，具体代码如下：

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

添加依赖只需要在 dependencies 标签下添加 dependency 标签即可。可到网址“https://mvnrepository.com/”查找所需 dependency。项目中我们会用到很多依赖，只需要添加 dependency 就可以，每个 dependencies 下面可以添加多个 dependency。

2.2.3 Maven 资源库

资源库有本地资源库、中央资源库、远程资源库。

本地资源库通常在当前用户的～/.m2 文件夹下，下载后的 jar 包会保存在本地资源库，以供下次使用。当建立 Maven 项目时，系统会根据 pom.xml 配置文件，确定需要的依赖。首先 Maven 从本地资源库中查找，如果没有找到，会到中央资源库（http://repo1.maven.org/maven2/）中查找 jar 包，下载到本地资源库。然后系统会从本地资源库中把需要的 jar 包拷贝到项目中。

远程资源库是个人存放 jar 包的位置，只有在使用一些特殊 jar 包时配置使用。

2.2.4 Maven 镜像

当从中央资源库下载 jar 包时，经常因为网速的问题导致下载失败，此时项目中 pom.xml 的相应依赖就会报红。

从中央资源库下载 jar 包的速度有时很慢，因为默认的下载地址是国外官方网址，也可以改成国内镜像，像阿里云或清华等镜像就比较快。

更改镜像需要更改 maven/conf/settings.xml 文件，在 mirrors 里面添加 mirror 镜像即可。

```
<mirrors>
    <mirror>
      <id>alimaven</id>
      <name>aliyun maven</name>
      <url>http://maven.aliyun.com/nexus/content/groups/public/</url>
      <mirrorOf>central</mirrorOf>
    </mirror>
</mirrors>
```

【项目】Maven 部署 Junit 测试环境

要求：使用 Maven 搭建项目，配置 Junit 单元测试环境。

应用知识点如下：

2.2.2 Maven 配置文件 pom.xml

2.2.3 Maven 资源库

☑ 项目构思

使用 Junit 测试环境编程，需要导入 junit.jar 包，关于 Junit 的 jar 包，随着版本的升级，在版本 3.x 之后还需要有依赖包 hamcrest-core.jar，这是在经过测试后发现的问题，而如果仅仅导入 Junit 包却没有下载相应依赖包，则会出现编译错误。这个仅仅是 Junit 测试环境，如果我们使用框架编程，相应的 jar 包和依赖包更多。所以我们使用 Maven 工具进行环境的搭建，它会帮助开发人员自动查找相应版本的 jar 包及对应的依赖包，并下载到项目中。

使用 Maven 搭建测试环境，在 Maven 的配置文件 pom.xml 中配置 Junit 测试环境需要相应的包名和版本号，Maven 会自动帮助我们查找相应版本的 jar 包，下载并导入到项目中。

☑ 项目设计

使用 Maven 搭建 junit 测试环境，创建普通 Java 工程就行，在 pom.xml 里面添加依赖，系统自动导入所需 jar 包和相应依赖包。编写 Maven 项目的具体步骤是：

(1)创建项目，选择 Maven 模板；

(2)填写 pom.xml 配置文件，填写依赖 dependency；

(3)系统自动查找相应的 jar 包和依赖包到项目中。

也就是说，我们只需要在配置文件里面填写好依赖 dependency，系统就会帮助我们查找相应的 jar 包和依赖包(下图简称相应包)，并导入到项目中。实现原理如图 2.10 所示。

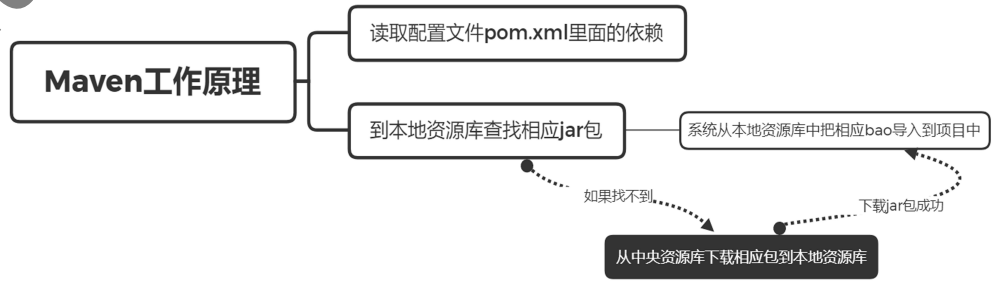

图 2.10　Maven 工作原理图

☑ 项目实施

步骤 1:新建普通 Maven 工程,不使用 Maven 自带模板,如图 2.11 所示。

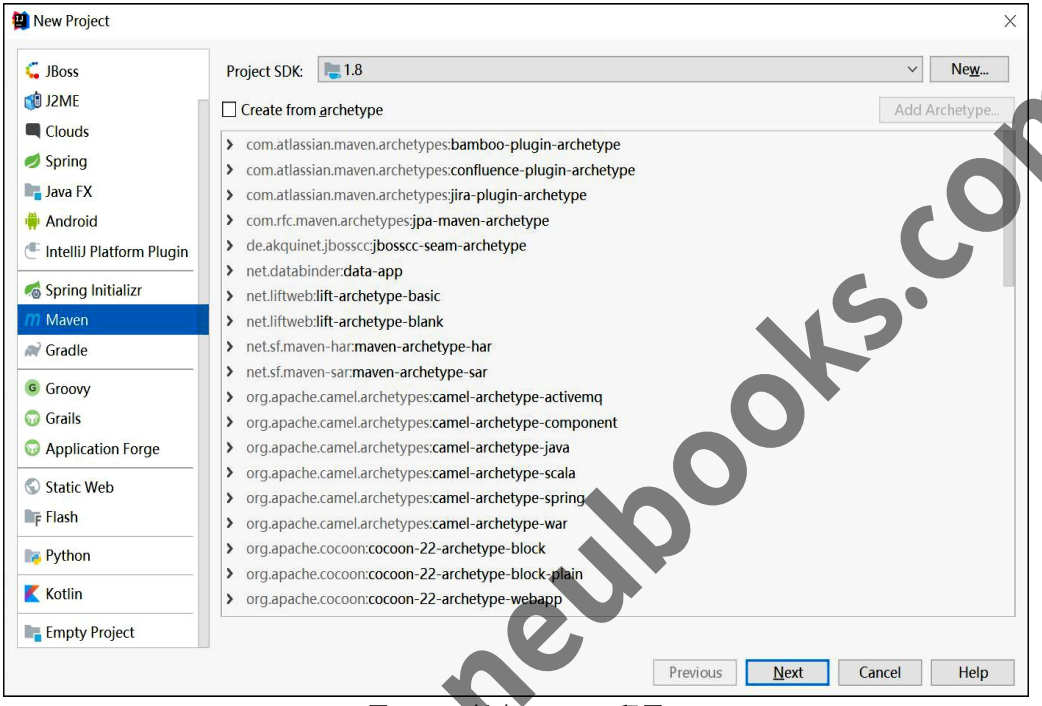

图 2.11 新建 Maven 工程图

步骤 2:填写项目信息,如图 2.12 所示,其中,GroupId 是项目组名,我们通常使用“项目类型. 公司名”,ArtifactId 是项目名,Version 记录项目版本号。

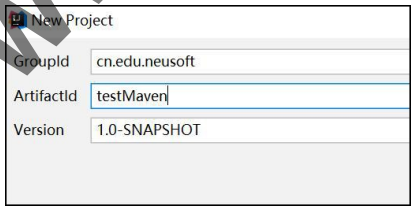

图 2.12 GroupId 和 ArtifactId 配置图

步骤 3:填写 pom.xml 文件,在配置文件里面写依赖,让 Maven 帮助我们查找相应版本的 jar 包及依赖包。

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.11</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

步骤 4：写测试函数。

```
package test1;
import org.junit.Test;

public class TestSecond {
//     使用@Test 注解，表明要使用测试环境，此函数是测试入口，不需要 main。
//     测试函数不能有参数和返回值。
    @Test
    public void testse(){
//         打印测试语句，测试本函数即可。
        System.out.println("hello junit");
    }
}
```

☑ 项目运行

本项目比较简单，运行结果时在控制台看到“hello junit”就表示成功了，如图 2.13 所示。

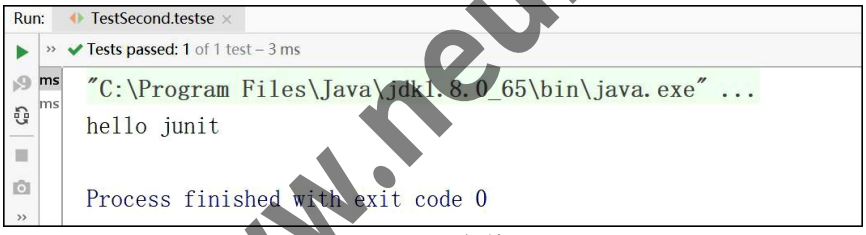

图 2.13　运行结果

☑ 项目总结

问题：当使用 Maven 进行项目布局时，经常因为网速的原因导致下载失败，也就是下载完成后（IDEA 进度条执行完毕），在工程中的 External Libraries 中看不到下载的 jar 包，此时 pom.xml 中依赖的 dependency 报红。

原理是下载失败，此时在本地资源库目录“C:\Users\NEUSOFT\.m2\repository\junit\junit”中，可以看到对应的版本号已经存在，但是进入即可看到下载失败（没有下载完对应的 jar）。

解决方法有两种：

（1）更改下载版本号，比如本项目中，下载版本号是 4.11，我们可以更改成之前版本 4.10，重新下载，如果网络正常，就会下载成功。

（2）在本地资源库目录下面，删除对应下载的文件夹，比如本项目中是“C:\Users\NEUSOFT\.m2\repository\junit\junit\4.11”，我们可以删除文件夹 4.11，重新下载。对应的操作是，先在 pom.xml 配置文件中删除 Junit 的配置项，接着在本地资源库删除对应版本号的文件夹，然后在 pom.xml 配置文件中重新配置 Junit，保存即可重新下载。

本章小结

本章通过搭建 Junit 单元测试环境的案例，讲解 Maven 框架的组成、原理及使用。

Maven 是工具也是框架，说它是工具是因为它帮助我们搭建项目环境；说它是框架是因为它有配置文件 pom. xml。

主要的两要素是：配置文件 pom. xml 和资源管理库。

通过在配置文件里面配置依赖，Maven 帮助我们查找中央资源库，下载相应 jar 包和依赖包到本地资源库，并导入到项目中搭建项目环境。

习　题

(1)创建 JavaSE 项目，使用 main 函数编辑“helloworld”，运行后输出到控制台。

(2)创建 JavaEE 项目，在 index. jsp 页面输出“helloworld”，配置 Tomcat 并启动项目。

(3) * . 尝试使用 IDEA 的调试功能，调试 JavaSE 项目和 JavaEE 项目。

(4) * . 使用 maven 工具导入 spring-context 依赖，搭建 Spring 框架项目环境。