第 1 章　建立 Java Web 开发环境

Java 技术体系从技术层次上包括两个部分：第一部分是 Java SE，也就是 Java Standard Edition，称为 Java 标准版，是 Java 技术的基础内容。使用 Java SE 可以基于 Java 面向对象技术、方法和基础类库开发桌面应用程序；第二部分是 Java EE，也就是 Java Enterprise Edition，称为 Java 企业版，是 Java 技术的扩展内容。使用 Java EE 可以开发各种类型的网络应用程序，如电子商务系统、电子政务系统等。Java Web 是 Java EE 的核心部分，使用 Java Web 可以开发基于 Web 技术的网络应用系统。本书介绍的内容就是基于 Java Web 技术开发 Web 应用程序。

1.1　Java Web 概述

Java Web 是 Java EE 的重要核心内容，是基于 Java 技术开发动态 Web 应用程序的一系列技术总称。一个 Java Web 应用系统从结构上包括两个部分：其一，前端服务程序，也称为前端程序、前端或客户端程序。用户通过前端服务程序与系统进行交互。例如，在电子商务系统中，用户通过前端浏览器浏览并选择商品等；其二，后端服务程序，也称为后端程序或后端或服务端。后端服务程序接收从前端服务程序发来的请求信息，对请求信息进行处理，然后将处理结果通过前端服务程序展示给用户。例如，在电子商务系统中，将用户已经选择的商品加入商品购物车，然后对购物车中的商品进行支付结算等。典型的基于 Java Web 技术的网络应用的结构如图 1-1 所示。

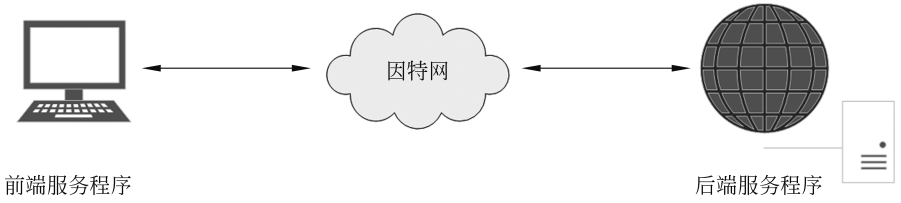

图 1-1　Java Web 应用系统结构

1.1.1　前端服务程序

前端服务程序是完成用户与系统进行交互的程序的总称，通过前端服务程序，用户与系统进行信息交互，录入输入数据和显示处理结果等。典型的进行 Web 前端开发技术包括 HTML 页面设计、CSS 布局技术、JavaScript 前端动态页面技术等，除了这些技术外，移动 APP、微信小程序也是前端服务程序开发技术。有关 HTML、CSS、JavaScript、移动 APP 及

微信小程序开发相关技术不属于本书的讲解范围，可自行搜索相关资料学习。

1.1.2 后端服务程序

在用户与系统通过前端程序进行信息交互的过程中，前端服务程序需要将用户输入的信息发送给后端服务程序进行处理，再将处理结果返回给前端程序并以适当的形式展示给用户。因此，后端服务程序需要具备如下基本能力：其一，接收从前端服务程序发送的信息；其二，对接收到的用户信息进行处理；其三，将信息处理结果返回给前端程序。为了达成以上三个目标，后端服务程序需要以下技术的支撑。

（1）应用服务器：也称为 Web 容器（Web container），是执行后端服务程序的机构。后端服务程序都运行在应用服务器中。典型的应用服务器包括 Tomcat 应用服务器、WebLogic 应用服务器、JBoss 应用服务器等。本书使用目前在国内应用较多的 Tomcat 应用服务器。

（2）Servlet 技术：完整的表述应该是 server applet，也就是服务端小程序。它是重要的运行在应用服务器中的程序。Servlet 程序接收前端服务程序发来的请求数据，对数据进行处理，例如，将数据保存到数据库中等，然后将处理结果返回给前端服务程序，进而通过前端服务程序将处理结果展示给用户。

（3）JSP 技术：也就是 Java Server Page（Java 服务端页面）。通过 JSP，可以将经过 Servlet 程序处理的结果以 HTML 页面的形式展示给用户。

（4）Thymeleaf 技术：一种动态页面生成技术。通过 Thymeleaf，可以将经过 Servlet 处理的结果数据以 HTML 页面的形式展示给用户。Thymeleaf 与 JSP 类似，是一种动态页面生成技术。

（5）Session 技术：也就是会话管理技术。鉴于 HTTP 缺乏会话管理技术，Java Web 提供了自己的称为 Session 的会话管理技术来管理多个 HTTP 请求的关联关系。

（6）Java Web 其他相关技术：包括数据库操作技术、文件上传下载技术、异步请求技术、模板技术等。

本书将对以上技术进行详细讲解，以期通过学习本教材，能够开发中等规模的基于 Java Web 的网络应用程序。下面从建立 Java Web 开发环境开始介绍。

1.2 建立 Java Web 开发环境

Java Web 技术体系构建在 Java SE 技术体系基础上。建立 Java Web 开发环境包括三方面的工作：其一，安装 Java SE，也就是 Java JDK；其二，安装 Java 开发环境，本书以目前企业使用较多的 IntelliJ IDEA 为工具；其三，安装 Tomcat 应用服务器。

建立 Java Web 开发环境

建立 Java Web 开发环境说明

1.3　开发第一个 Java Web 程序

现在已经安装了必要的 Java Web 程序开发环境，可以开发 Java Web 应用程序了。作为第一个 Java Web 应用程序例子，现在设计一个简单的登录页面。设计完成的登录页面如图 1-2 所示。

图 1-2　登录页面效果

1.3.1　新建 Java Web 项目

为了设计如图 1-2 所示的登录页面，需要新建一个 Java Web 程序工程。首先启动 IDEA 开发工具，如图 1-3 所示。

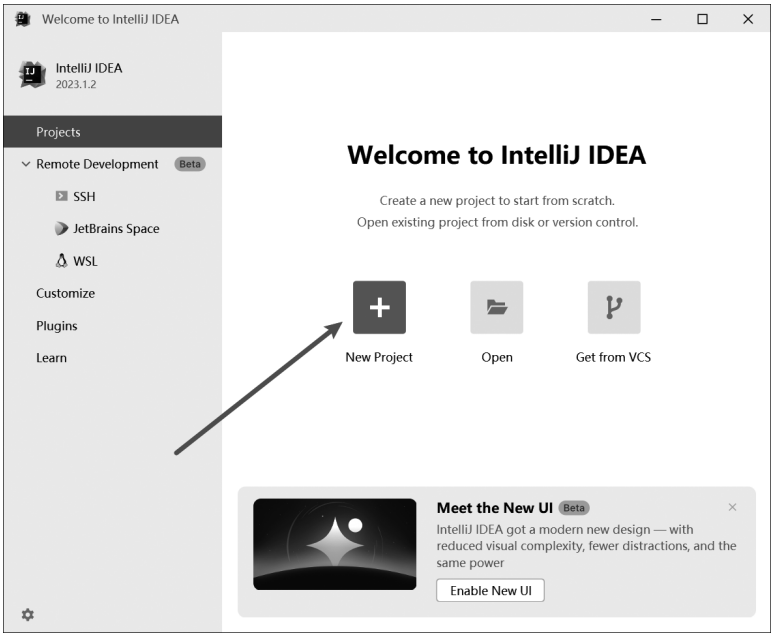

图 1-3　IDEA 启动界面

单击 New Project 按钮，创建一个新的名称为 ch01 的 Java Web 工程，如图 1-4 所示。

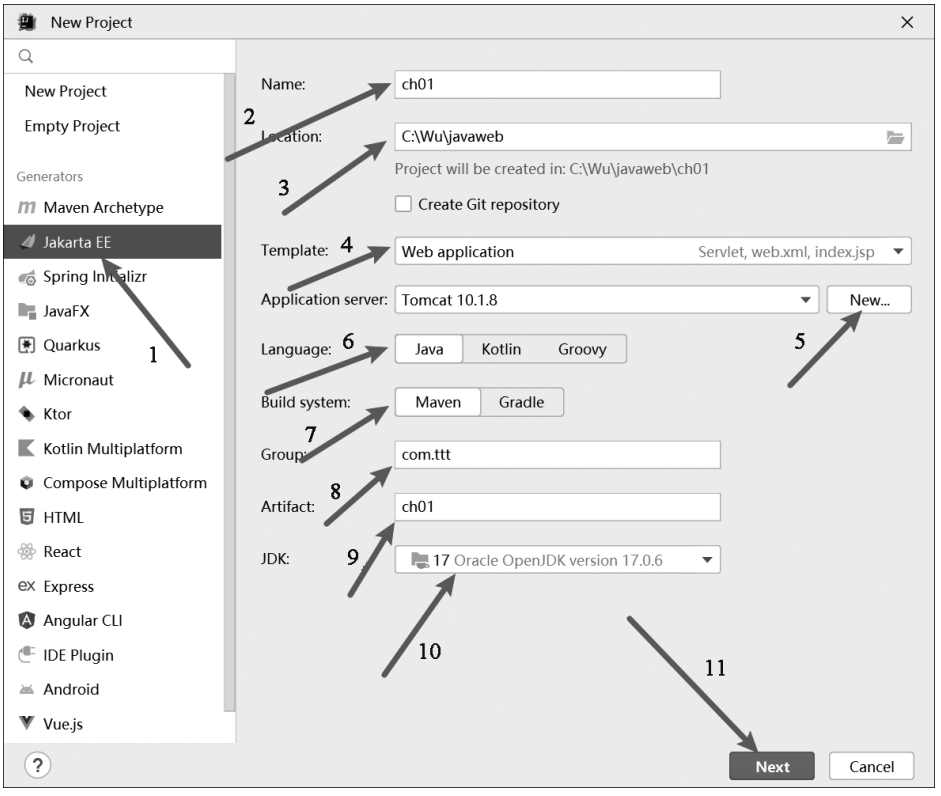

图 1-4　新建名为 ch01 的 Java Web 工程

在图 1-4 的界面中，按箭头所示的顺序选择或者输入相关信息。注意：在箭头 4 指向的下拉列表中选择 Web application；单击箭头 5 指向的 New 按钮，选择在 1.2 节安装的 Tomcat 应用服务器位置。之后，单击 Next 按钮，显示如图 1-5 所示的界面。

在图 1-5 的界面中，单击箭头所指向的下拉列表，其中包含如下几个选择：Java EE 8、Jakarta EE 9.1 和 Jakarta EE 10。为了明确这几个选项的区别，需要对 Java EE 的发展历史有一个初步了解。

如前所述，Java 技术体系包括两个部分：Java SE 和 Java EE，在 2008 年之前，所有的 Java 技术都是由 Orcale 管理和维护的。2008 年，Oracle 将 Java EE 技术体系捐献给 Eclipse 基金会，Oracle 自己只维护 Java SE 技术体系，也就是说，2008 年之后，Java EE 技术体系将由 Eclipse 基金会管理和维护。为了区别于 2008 之前的 Java EE 技术体系，Eclipse 基金会将新的 Java EE 技术体系更名为 Jakarta EE。

截止到 2008 年，Java EE 的最新技术体系的版本是 Java EE 8，这也是 Java EE 的最后一个版本，之后将不再更新，新的版本将以 Jakarta EE 名称出现。Jakarta EE 9.1 是 Jakarta EE 的第一个版本，之后将以 Jakarta 命名并不断推出新的版本。

当然，Java EE 技术体系更名为 Jakarta EE 技术体系，不只是简单的名称变更，还包括以下两个重要方面内容的变更：一方面原 Java EE 的 javax 包名称空间仍归 Oracle，Jakarta EE 不得使用，因此，Jakarta 将原 javax 包名称空间变更为 jakarta；另一方面对于常用的

Tomcat 应用服务器，Tomcat 9.x 是支持 Java EE 技术体系的最后一个版本，从 Tomcat 10.x 开始，Tomcat 应用服务器只支持 Jakarta EE。Java EE、Jakarta EE 与 Tomcat 的版本之间的关系如图 1-6 所示。

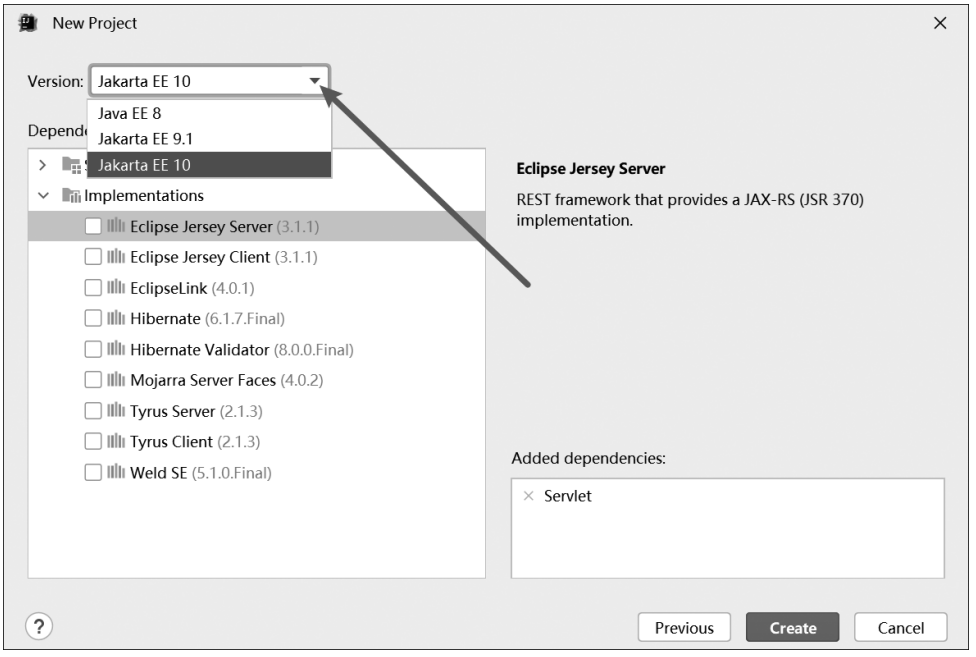

图 1-5　选择 Java Web 的版本

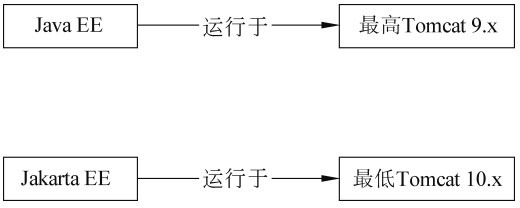

图 1-6　Java EE、Jakarta EE 与 Tomcat 的版本之间的关系

由于 Java EE 不再更新，而 Jakarta EE 才是未来，因此，本书将主要介绍 Jakarta EE。当然，由于本书所介绍的内容是 EE 的核心部分，因此，本书的内容对 Java EE 也是适用的。明确了 Java EE 和 Jakarta EE，现在在图 1-5 的界面箭头所指向的下拉列表中选择 Jakarta EE 10，然后单击 Create 按钮，显示如图 1-7 所示的界面。

现在已经完成了名称为 ch01 的 Java Web 程序工程的创建，下面在 ch01 模块中编写登录界面程序。

1.3.2　编写登录页面代码

编写程序的一个良好习惯是将代码分门别类地放置在工程合适的子目录之中。IDEA 就已经这样做了，IDEA 要求将所有的 Java 代码及其资源放置在“src/main/java”工程目录下；将所有的前端代码，包括 HTML 文件、CSS 文件、JavaScript 文件及用到的其他页面资

源文件放置在 webapp 工程目录下。由于在即将编写的登录页面代码中只有前端页面代码，目前还没有涉及 Java 代码，因此，在 ch01 下的 webapp 工程目录下新建如下几个子目录 html、css、js、images，分别放置前端的 HTML 页面文件、页面布局样式文件、JavaScript 代码文件、页面中用到的图片文件。为此，右击 webapp 子目录，在弹出的上下文菜单中选择 New→Directory，然后在弹出的界面中输入子目录名即可。完成后的 ch01 工程结构如图 1-8 所示。

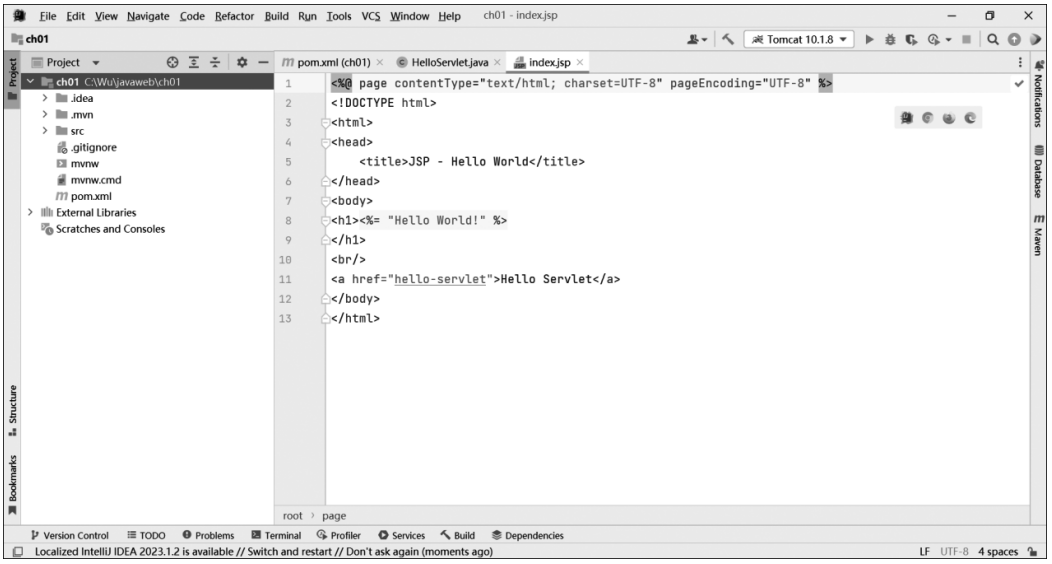

图 1-7　新建的 ch01 Java Web 工程

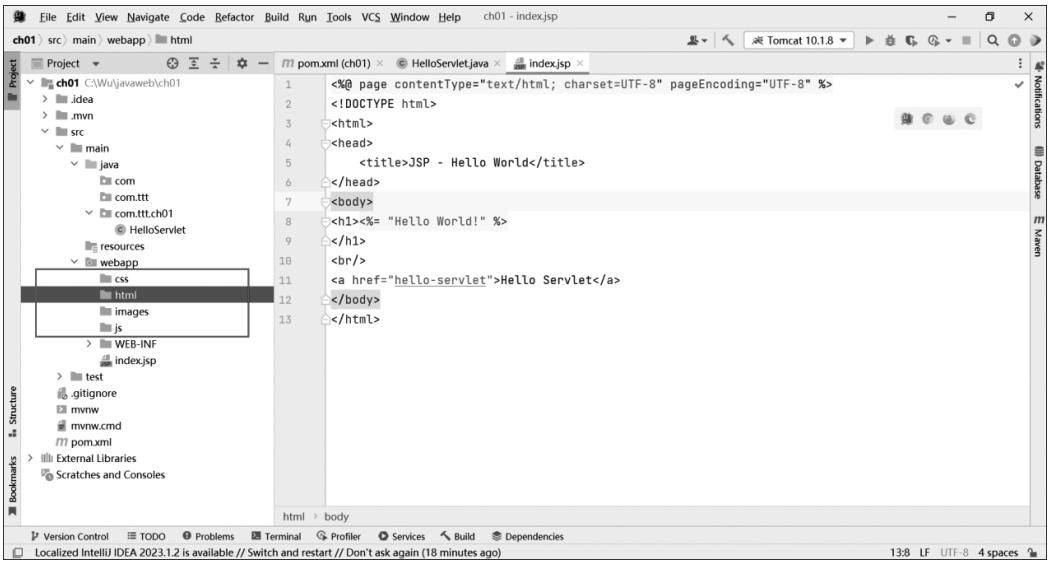

图 1-8　在 webapp 目录下新建子目录

为了使登录页面美观，需要设计 CSS 页面布局文件，在 ch01 工程的 webapp/css 目录下新建一个 style.css 文件。为此，右击 css 子目录，选择 New→Stylesheet，在弹出的界面中

输入文件名 style.css，然后在 style.css 文件中录入如下内容：

```
input[type=text], input[type=password], select {
    width: 100%;
    padding: 12px 20px;
    margin: 8px 0;
    display: inline-block;
    border: 1px solid #ccc;
    border-radius: 4px;
    box-sizing: border-box;
}
input[type=submit] {
    width: 100%;
    background-color: #4CAF50;
    color: white;
    padding: 14px 20px;
    margin: 8px 0;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}
input[type=submit]:hover {
    background-color: #45a049;
}
div {
    border-radius: 5px;
    background-color: #f2f2f2;
    padding: 20px;
}
```

再创建 HTML 页面代码。将页面代码文件 login.html 放置在 ch01 工程的 webapp/html 目录下。为此，右击 html 子目录，在弹出的菜单中选择 New→HTML File，并在弹出的界面中输入文件名 login.html，然后在 login.html 文件中录入如下代码：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>Java Web 程序设计</title>
    <link href="../css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>
<h3>请登录</h3>
<div>
    <form action="#">
        <label for="username">用户名</label>
        <input type="text" id="username" name="username" placeholder="用户
名...">
        <label for="password">密码</label>
        <input type="password" id="password" name="password" placeholder="密
码...">
```

```
        <label for="usertype">用户类型</label>
        <select id="usertype" name="usertype">
            <option value="admin">管理员</option>
            <option value="user">普通用户</option>
        </select>
        <input type="submit" value="提交">
    </form>
</div>
</body>
</html>
```

完成代码编写后的 ch01 工程的结构如图 1-9 所示。

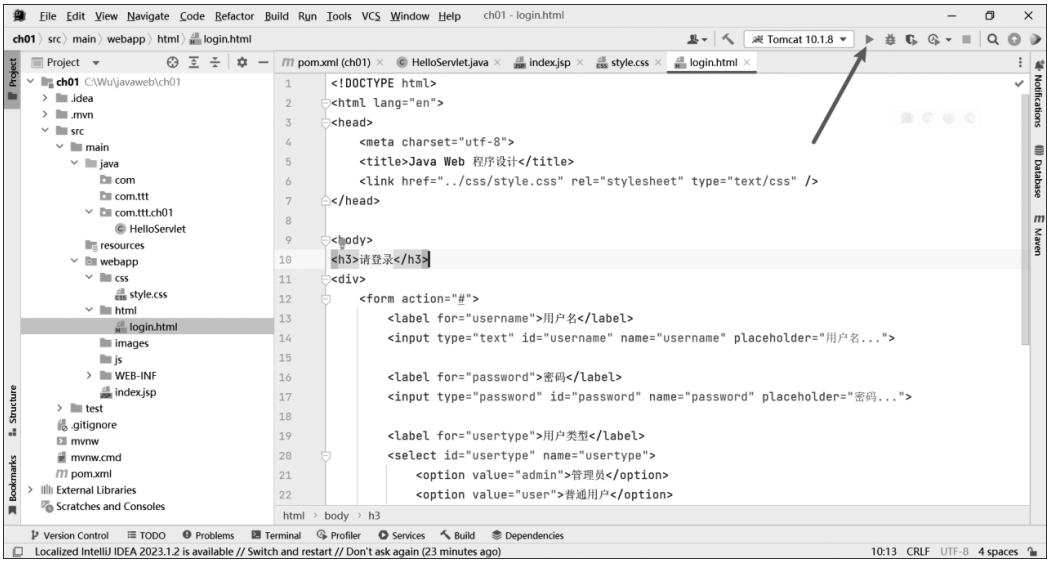

图 1-9　完成代码编写后的 ch01 工程的结构

1.3.3　运行和访问登录页面

为了运行 ch01 程序，在图 1-9 所示的界面中单击箭头所指向的代表“运行”功能的按钮，即可启动程序的运行，如图 1-10 所示。

待 ch01 成功运行后，IDEA 会自动打开浏览器，并显示如图 1-11 所示的界面。

ch01 程序运行后，为什么会自动在浏览器中显示如图 1-11 所示的界面呢？这是 Java Web 程序技术规范的规定：启动一个 Java Web 程序，一旦成功启动，会自动打开 webapp 目录下的名称为 index.html 或者 index.jsp 的称为“主页”的页面。观察图 1-8 或图 1-9 所示的 ch01 工程目录结果，会发现在 webapp 目录下存在名称为 index.jsp 的文件，因此，按照 Java Web 的技术规范要求，将显示 index.jsp 的页面内容，而图 1-11 所显示的内容正是 index.jsp 页面的内容。关于 jsp 页面程序，将在后续内容中介绍。为了显示 login.html 页面内容，在图 1-11 的浏览器地址栏输入 http://localhost:8080/ch01_war_exploded/html/login.html，将显示如图 1-12 所示的页面。

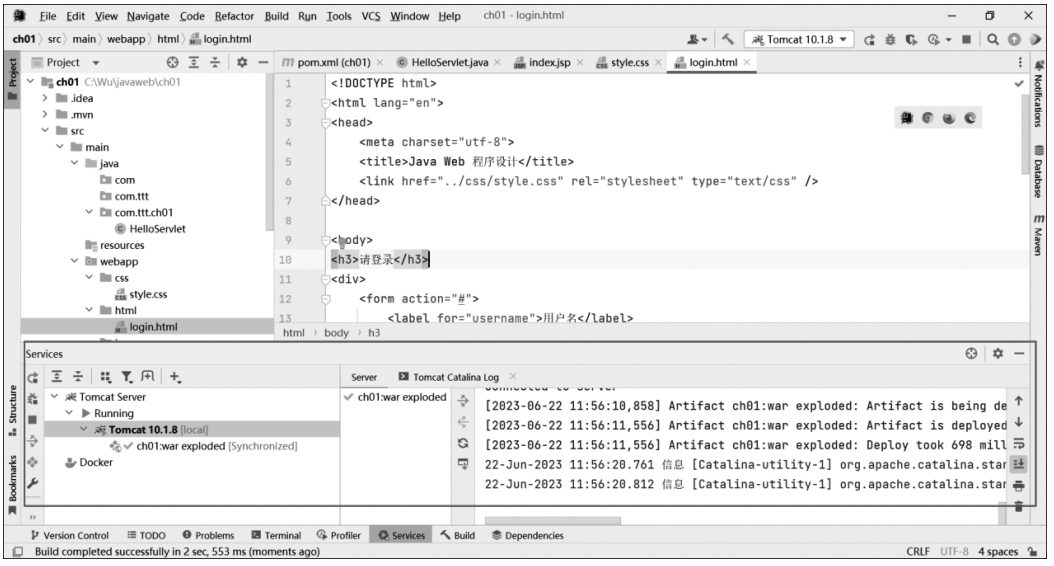

图 1-10　ch01 成功运行后的界面

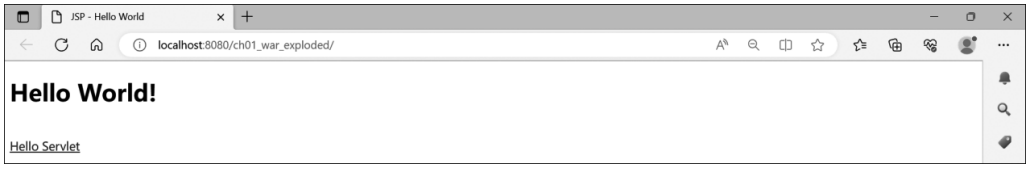

图 1-11　IDEA 自动打开浏览器并显示程序入口页面

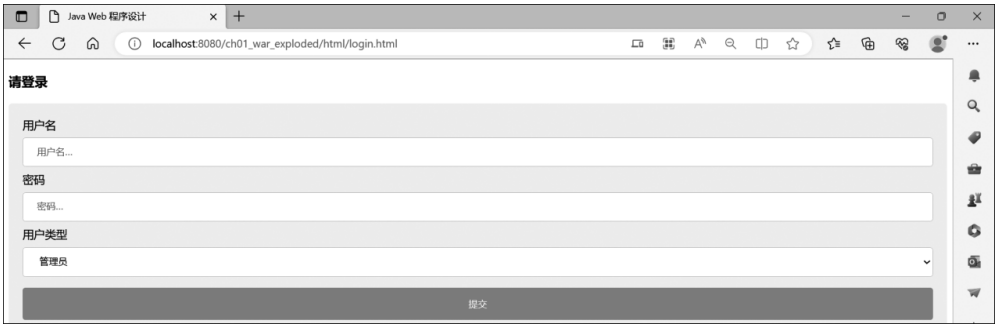

图 1-12　login.html 页面效果

在图 1-12 的页面中，可以在相应的输入框中输入用户名、密码及选择用户类型，然后单击“提交”按钮，此时可以发现浏览器并没有将所输入的用户名、密码及用户类型数据发送到服务器。究其原因是目前这个页面并没有对用户信息进行处理并提交给服务端程序。在后续的章节中会详细介绍如何将信息提交给后端服务程序。

当然，完全可以在另一台计算机的浏览器的地址栏输入 login.html 的页面地址来访问这个登录页面，但是需要将地址 http://localhost:8080/ch01_war_exploded/html/login.html 中的 localhost(因为 localhost 表示运行浏览器的计算机与运行 ch01 程序的计算机是同一台计算机)修改为 ch01 程序运行的计算机的 IP 地址，这时可以发现，完全可以在另一

台计算机上访问 ch01 程序的登录页面。

创建和运行第一个 Java Web 程序

1.3.4 IDEA 中或页面中出现乱码的解决方法

开发一个 Java Web 应用程序会涉及许多相互关联的环节，任何一个环节处理不当就会在浏览器或 IDEA 的信息显示窗口出现乱码，特别是在显示包含中文文字时出现乱码。本质上，出现显示乱码问题是由于在不同环节使用了不同的字符编码导致的，因此，解决显示乱码的根本方法就要保证在各个环节使用统一的字符编码，一般建议使用 UTF-8 编码。因此，建议在 HTML 页面文件中，在其 head 标签下的都加上这一句：

```
<meta charset="utf-8">
```

以确保界面使用了 UTF-8 编码。在 IDEA 中，选择菜单 Help→“Edit Custom VM Options...”，在打开的文件中也加上“-Dfile.encoding=UTF-8”选项，如图 1-13 所示。

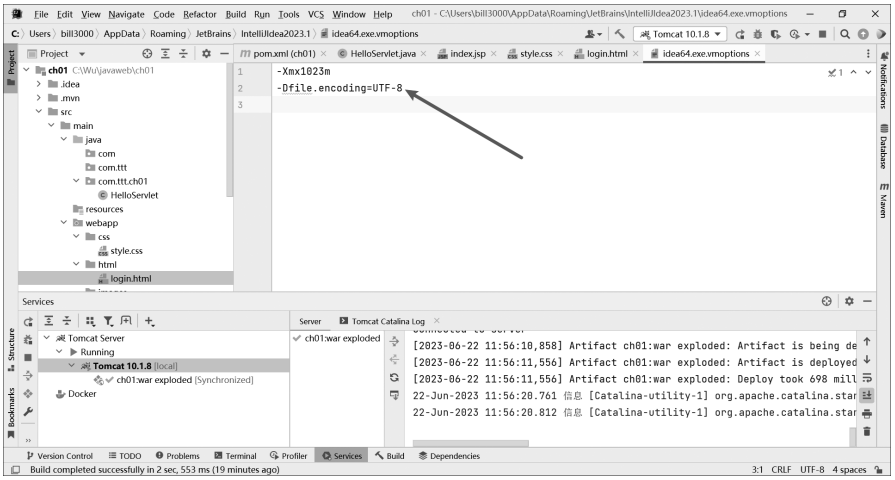

图 1-13　在 IDEA 的配置文件解决控制台乱码问题

提示：在各个环节使用统一的字符编码可以解决信息显示出现时的乱码问题。

1.4　C/S 架构和 B/S 架构

在基于网络的应用中，典型的工作过程是：用户通过计算机向服务器发送请求信息，服务器接收并处理用户发来的信息，并将处理结果返回给用户，然后通过运行在用户计算机上