

# 第一个 Java 程序

本书第一个 Java 程序是通过控制台输出 HelloWorld,以这个示例为切入点,系统介绍 Java 程序的编写、Java 源代码结构以及一些基础知识。

在 Java 中,程序都是以类的方式组织的,Java 源文件都保存在 .java 文件中。每个可运行的程序都是一个类文件,或者称为字节码文件,保存为 .class 文件。要实现在控制台中输出 HelloWorld 示例,则需要编写一个 Java 类。

## 3.1 使用 IntelliJ IDEA 实现

HelloWorld 示例可通过多种工具实现,本节首先介绍如何通过 IntelliJ IDEA 实现。

### 3.1.1 创建项目

在 IntelliJ IDEA 中通过项目(Project)管理 Java 类,因此需要先创建一个 Java 项目,然后在项目中创建一个 Java 类。

IntelliJ IDEA 创建项目步骤如下:

(1) 如果 IntelliJ IDEA 第一次启动,则先启动如图 3-1 所示的欢迎页面,在欢迎页面中单击 Create New Project 按钮,进入如图 3-2 所示对话框。



微课视频

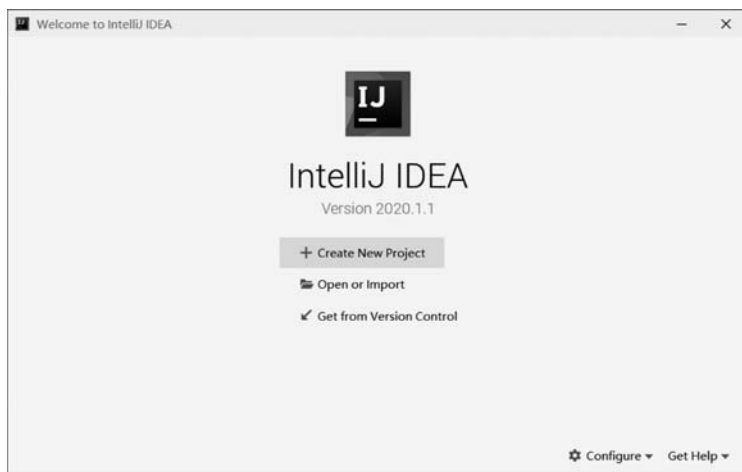


图 3-1 欢迎页面

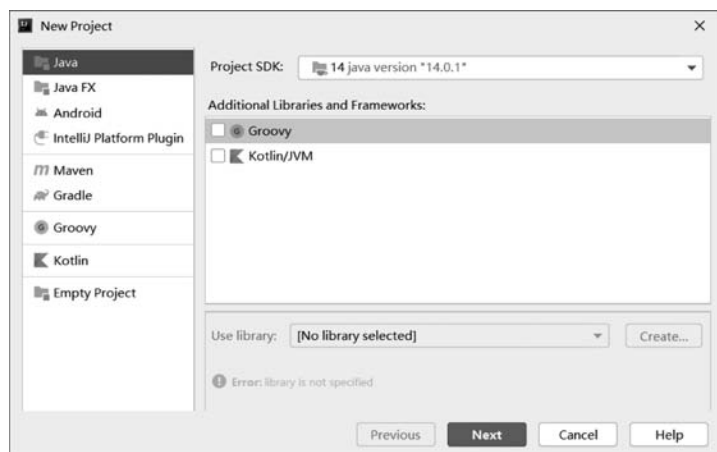


图 3-2 新建 Java 项目

(2) 如果已经进入 IntelliJ IDEA 工具,则通过选择 File→New→Project 菜单,也可以进入如图 3-2 所示对话框。

在图 3-2 中选择 Java 项目类型。如果 JDK 配置没有问题,则在 Project SDK 下拉框中会识别 JDK 版本。其他的选项保持默认值,然后单击 Next 按钮进入如图 3-3 所示对话框进行 Java 项目模板选择。注意不要选择任何选项,直接单击 Next 按钮进入如图 3-4 所示对话框进行项目名称设置。根据自己的情况在 Project name 输入框中输入项目名称,在 Project location 输入框中输入项目保存路径。设置完成后单击 Finish 按钮完成项目创建,然后进入如图 3-5 所示的 IntelliJ IDEA 工具。

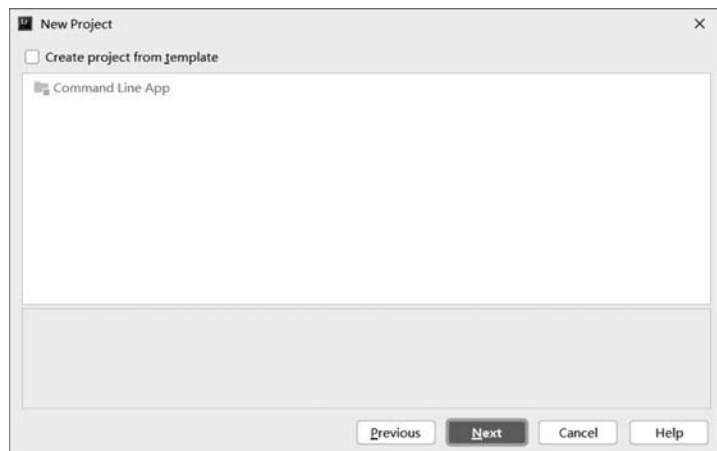


图 3-3 新建 Java 项目模板

### 3.1.2 创建类

项目创建完成后,右击 src 文件夹,选择 New→Java Class 菜单命令,打开如图 3-6(a)所示 New Java Class(新建 Java 类)对话框。然后在输入框中输入类名 HelloWorld,如图 3-6(b)所示,双击 Class 类型创建 HelloWorld 类,如图 3-7 所示。

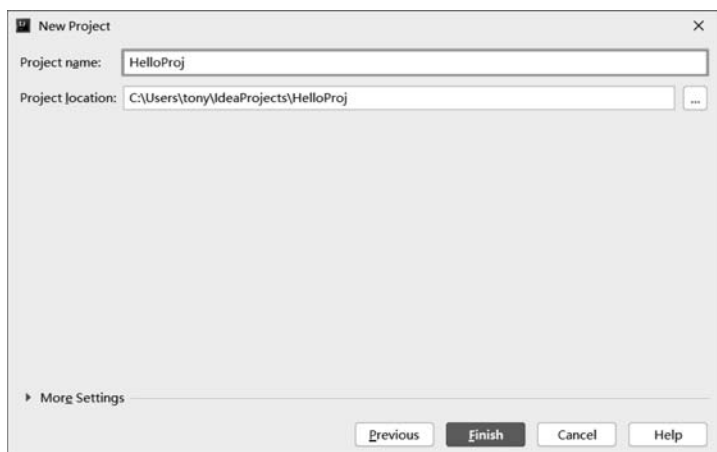


图 3-4 项目名称设置

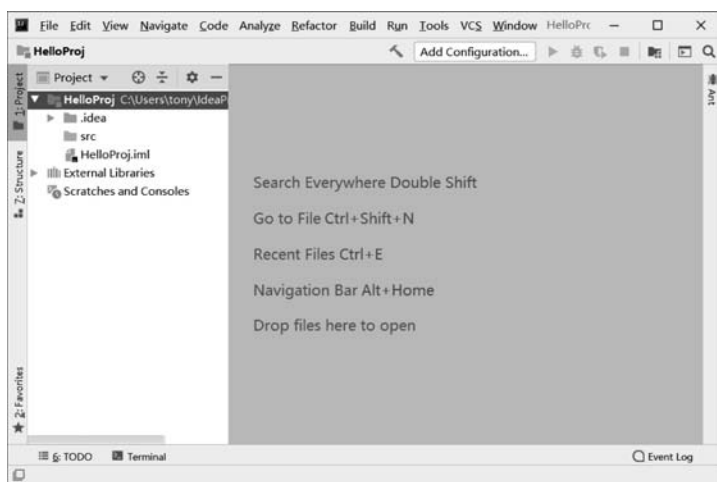


图 3-5 项目创建完成

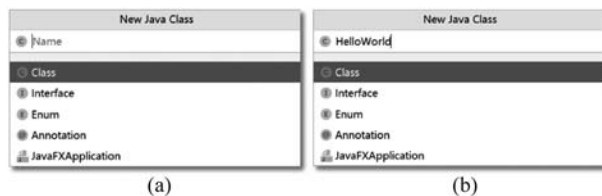


图 3-6 “New Java Class”对话框

### 3.1.3 运行程序

修改刚生成的 HelloWorld.java 源文件,添加 main 方法,并添加输出语句。修改完成后代码如下:

```
public class HelloWorld {  
  
    public static void main(String[] args) { ①  
        System.out.print("Hello World."); ②  
    }  
}
```

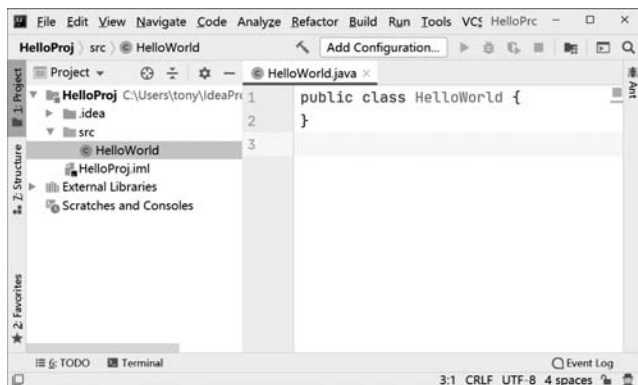


图 3-7 创建类完成

}

}

代码第①行中的 `public static void main(String[] args)` 方法是一个应用程序的入口,也表明了 HelloWorld 是一个 Java 应用程序(Java Application),可以独立运行。代码第②行的 `System.out.print("Hello World.")` 语句是输出 Hello World. 字符串到控制台。

程序编写完毕就可以运行了。如果是第一次运行,则需要选择运行方法,具体步骤:右击 HelloWorld 文件,选择 `Run 'HelloWorld.main()'` 菜单运行 HelloWorld 程序。如果已经运行过一次,就不需要这么麻烦了,直接单击工具栏中的“运行”按钮,或选择菜单 `Run→Run 'HelloWorld'`,或使用快捷键 `Shift+F10`,就可以运行上次的程序。运行结果如图 3-8 所示,则 Hello World. 字符串显示在下面的控制台中。

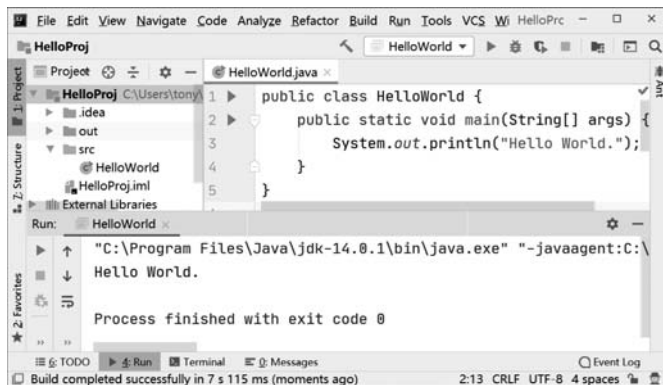


图 3-8 运行结果



微课视频

## 3.2 文本编辑工具+JDK 实现

如果不想使用 IDE 工具(建议初学者通过这种方式学习 Java),那么文本编辑工具+JDK 对于初学者而言是一个不错的选择,这种方式可以使初学者了解到 Java 程序的编译和运行过程,通过自己在编辑器中输入所有代码,可以帮助熟悉常用类和方法。

### 3.2.1 编写源代码文件

首先使用任何文本编辑工具创建一个文件,然后将文件保存为 HelloWorld.java。接着在 HelloWorld.java 文件中编写如下代码。

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.print("Hello World.");
    }

}
```

### 3.2.2 编译程序

编译程序需要在命令行中使用 JDK 的 javac 指令编写,参考 2.1.3 节打开命令提示符工具,如图 3-9 所示,通过 cd 命令进入源文件所在的目录,然后执行 javac 指令。如果没有错误提示,则说明编译成功。编译成功时会在当前目录下生成类文件,如图 3-10 所示生成了三个类文件,这是因为 HelloWorld.java 源文件中定义了三个类。

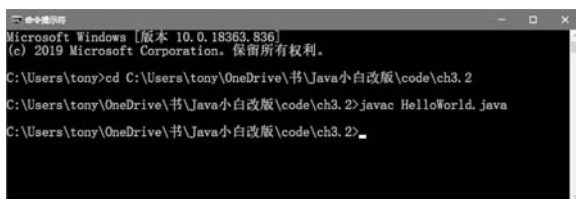


图 3-9 编译源文件



图 3-10 编译成功

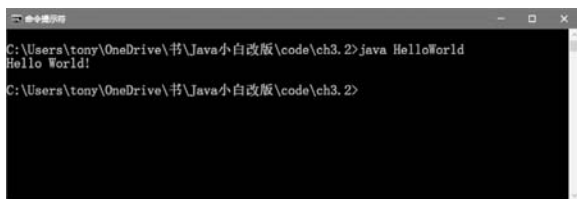


图 3-11 运行类文件

### 3.2.3 运行程序

编译成功之后就可以运行了。执行类文件需要在命令行中使用 JDK 的 java 指令,参考 2.1.3 节打开命令行,如图 3-11 所示,通过 cd 命令进入源文件所在的目录,然后执行 java HelloWorld 指令,执行成功后在命令行窗口输出 Hello World! 字符串。

## 3.3 代码解释

经过前面的介绍,读者应该能够照猫画虎,自己动手做一个 Java 应用程序了。但可能还是对其中的一些代码不甚了解,下面详细解释 HelloWorld 示例中的代码。



微课视频

```
//类定义
public class HelloWorld {                                ①

    //定义静态 main 方法
    public static void main(String[] args) {            ②
        System.out.print("Hello World.");              ③
    }

}
```

代码第①行是定义类,public 修饰符用于声明类是公有的,class 是定义类关键字,HelloWorld 是自定义的类名,后面跟着的“{...}”是类体,类体中会有成员变量和方法,也会有一些静态变量和方法。

代码第②行是定义静态 main 方法,而作为一个 Java 应用程序,类中必须包含静态 main 方法,程序执行是从 main 方法开始的。main 方法中除参数名 args 可以自定义外,其他必须严格遵守如下两种格式:

```
public static void main(String args[])
public static void main(String[] args)
```

这两种格式本质上就是一种,String args[]和 String[] args 都是声明 String 数组。另外,args 参数是程序运行时通过控制台向应用程序传递字符串参数。

代码第③行 System.out.print("Hello World."); 语句是通过 Java 输出流(PrintStream)对象 System.out 打印 Hello World. 字符串,System.out 是标准输出流对象,它默认输出到控制台。输出流(PrintStream)中常用打印方法如下:

- print(String s): 打印字符串不换行,有多个重载方法,可以打印任何类型数据。
- println(String x): 打印字符串换行,有多个重载方法,可以打印任何类型数据。
- printf(String format, Object... args): 使用指定输出格式,打印任何长度的数据,但不换行。

修改 HelloWorld.java 示例代码如下。

```
public class HelloWorld {
    public static void main(String[] args) {

        //通过 print 打印第一个控制台参数
        System.out.print(args[0]);                      ①
        //通过 println 打印第二个控制台参数
        System.out.println(args[1]);                    ②
        //通过 printf 打印第三个控制台参数,%s 表示格式化字符串
        System.out.printf("%s", args[2]);               ③
        System.out.println();

        int i = 123;
        // %d 表示格式化整数
        System.out.printf("%d\n", i);                   ④

        double d = 123.456;
        // %f 表示格式化浮点数
        System.out.printf("%f %n", d);                 ⑤
        System.out.printf("%5.2f", d);                 ⑥

    }
}
```

编译 HelloWorld.java 源代码后,如图 3-12 所示,其中的 java 命令行后面的 HelloWorld 是要运行

的类文件, Tony Hello World. 是参数, 多个参数用空格分隔。

上述代码第①行使用 print 方法打印第一个参数 args[0], 注意该方法是在打印完成后不换行, 从输出结果中可见第一个参数 Tony 和第二个参数 Hello 连在一起了。代码第②行使用 println 方法打印第二个参数 args[1], 从输出结果中可见第二个参数 Hello 后面是有换行的。

代码第③、④、⑤、⑥行都是使用 printf 方法打印, 注意 printf 方法后面是没有换行的, 想在后面换行可以通过 System.out.println() 语句实现, 或在打印的字符串后面添加换行符号(\n 或 %n), 见代码第④行和代码第⑤行。代码第⑥行中 %5.2f 也表示格式化浮点数, 5 表示总输出的长度, 2 表示保留的小数位。

**提示** 在简体中文版本的 Windows 平台中默认编码集是 GBK。所以 javac 指令编译源代码文件时, 默认文件的编码集是 GBK。一般情况下使用记事本和 EditPlus 等文本编辑工具创建的源代码文件默认编码集也是 GBK, 因此在编译这些源代码文件时不会发生错误。但是使用 Sublime Text 工具创建的源代码文件默认编码集也是 UTF-8, 如果源代码文件中有中文, 则发生如图 3-13 所示的编译错误。为了解决这种错误, 可以在编译时指定源代码文件字符集, 如图 3-14 所示使用 javac-encoding UTF-8 HelloWorld.java 指令即可。

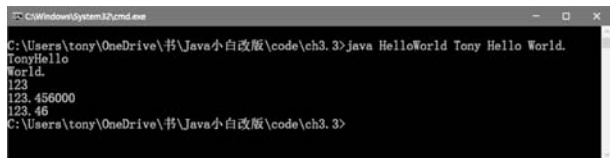


图 3-12 在命令行中运行程序

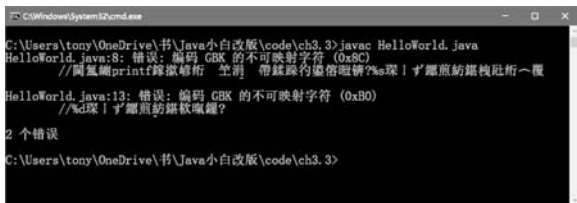


图 3-13 采用 UTF-8 编码的源代码文件在 Windows 平台中的编译错误

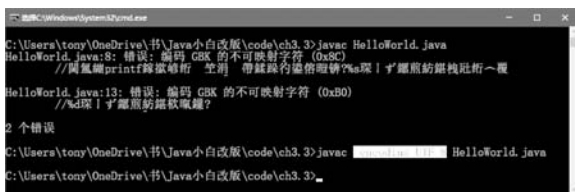


图 3-14 编译时指定源代码文件字符集

### 3.4 本章小结

本章通过一个 HelloWorld 示例, 介绍使用 IntelliJ IDEA 和使用文本编辑工具+JDK 实现该示例的具体过程。掌握 IntelliJ IDEA 的使用非常重要, 但是使用文本编辑工具+JDK 对于初学者也很有帮助。本章最后详细解释了 HelloWorld 示例。

### 3.5 动手实践：世界，你好！

#### 编程题

1. 使用 IntelliJ IDEA 工具编写并运行 Java 应用程序, 使其在控制台输出字符串“世界, 你好!”。
2. 使用文本编辑工具编写 Java 应用程序, 然后使用 JDK 编译并运行该程序, 使其在控制台输出字符串“世界, 你好!”。