October 11, 2023

FRC 101 Lecture 1: Java 基础特性

Evan Luo

目录

1 前言

1.1 本节课内容

在这节课中，你将会学到：

- Java 基本语法
- 变量
- 基本类型以及运算符
- 方法 (Method)
- `if-else` 控制
- `for` 控制

此外，之后的每一个 Lecture 都会有配套的 Lab。请在看完 Lecture 内容后完成 Lab 以复习并实践内容。有疑问可以先上 Google 进行搜索。若无果可以在程序部群中发送描述清楚的问题。正确地描述问题和使用搜索引擎是学习 CS 相关内容的基本技能。

1.2 如何使用

本文档的前半部分是课程内容的教学。最后一部分是一个 Lab 部分，参考了大学的 Lab 即实验。为了学习效果，在学习完前半部分后，请在一周内完成。

2 开始

Java 是一门有 20 多年历史的语言。其以其简单的语法（与 C、C++ 相近）、面向对象特性和安全性等在业界被广泛运用。在 FRC 当中，Java 是 FRC 官方库 WPILib 的支持语言之一。因此为了编写用于 FRC 机器人的代码，我们需要学习 Java。

2.1 环境配置

对于 Java，推荐使用 `Jetbrains Intellij IDEA`（后称 `IDEA`）。原因是 `Visual Studio Code` 等软件在静态分析以及集成的 Debugger 和提示等都不如 `IDEA`。因此 `IDEA` 是一个非常好的选择。

2.1.1 安装 `IDEA`

前往 Jetbrains 官网：https://jetbrains.com/idea 进行 `IDEA` 安装包的下载。此处需要注意，如果你不想申请 `Jetbrains Student License`，那么你需要下载 Community 版本而不是 Ultimate 版本。

2.1.2 配置 `IDEA`

打开 `IDEA` 后，左菜单栏会有一个 `Plugins` 选项。进入后在 `Marketplace` 搜索 `FRC` 并下载该插件。

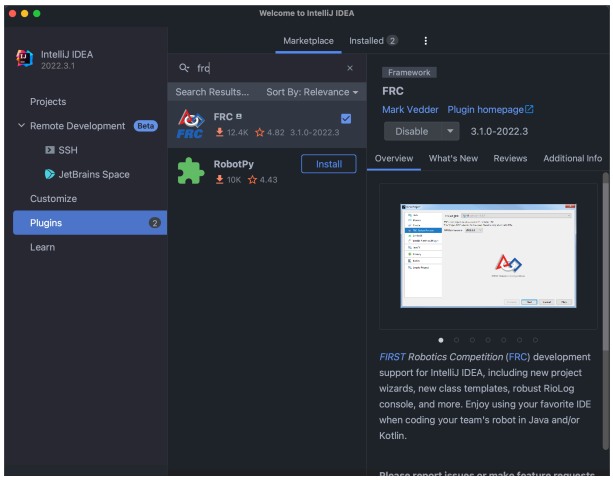

图 1: Plugin 页面

2.1.3 创建项目

进入左菜单栏中的 Projects 页面。点击右上角 New Project。选择左侧 New Project。 Name 与 Location 自行修改（Name 建议写为 lab-1 的形式）。Language 选择 Java。Build System 选择 Intellij。 JDK 若没有下载过 JDK 11 建议选择 Download JDK 并下载 JDK 11。

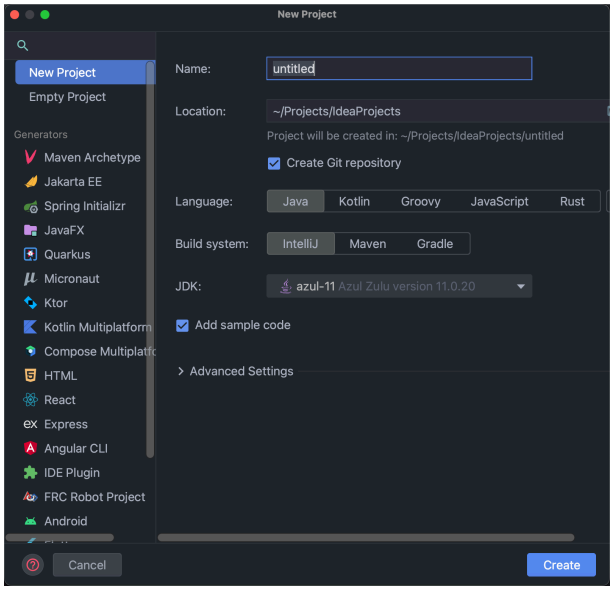

图 2: 新建 Project 页面

2.1.4 界面介绍

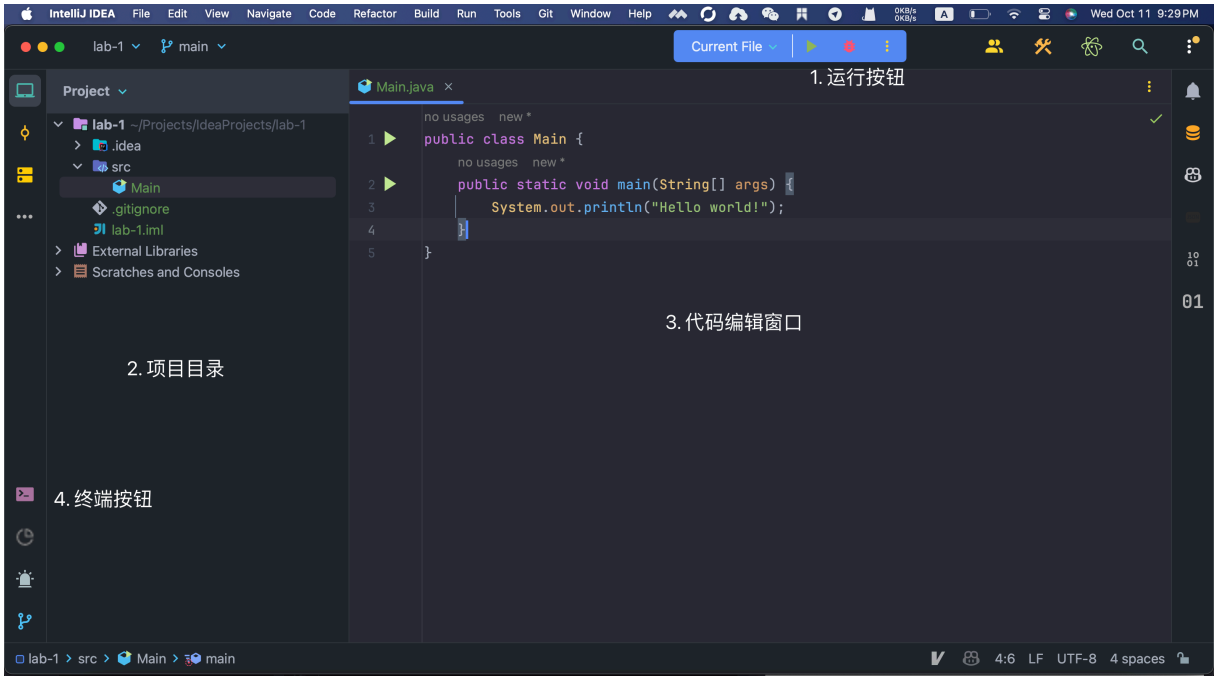

图 3: IDEA 页面

功能介绍：

1. 运行按钮：编写完程序后可以点击该按钮运行。结果将显示在终端
2. 项目目录：项目目录结构在此查看
3. 代码编辑窗口
4. 终端按钮：执行命令可以在此处执行。默认目录在项目根目录

2.2 Java 基本语法

2.2.1 第一个 Java 程序

```
// 这是一个注释
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

这是一个最简单的 Java 程序。我们将会围绕这段程序讲解 Java 的基础语法。

在 Java 当中，有以下书写规则：

- 大小写敏感：Java 是大小写敏感的，这就意味着标识符 Hello 与 hello 是不同的。
- 对于所有的类来说，类名的首字母应该大写。如果类名由若干单词组成，那么每个单词的首字母应该大写，例如 MyFirstJavaClass。
- 所有的方法名都应该以小写字母开头。如果方法名含有若干单词，则后面的每个单词首字母大写。
- 源文件名：源文件名必须和类名相同。当保存文件的时候，你应该使用类名作为文件名保存（切记 Java 是大小写敏感的），文件名的后缀为 .java。（如果文件名和类名不相同则会导致编译错误）。
- 主方法入口：所有的 Java 程序由 `public static void main(String[] args)` 方法（类似于函数）开始执行。
- 每一行完整的代码都需要用分号（;）隔开。

2.2.2 输出

要在 Java 当中输出一串字符，可以使用：`System.out.println("<输出内容>")`。该方法会输出一串字符并在结尾加一个换行符。

2.2.3 注释

如果你想在代码中添加一些评论或者你对代码的理解，可以使用注释。在 Java 当中，你可以用 `// 注释内容` 或 `/* 注释内容 */` 进行注释。后者可以跨行进行注释。

2.3 变量

变量是一种可以装载数据的结构。「变」意味着他的值是时刻改变且是临时的。因此变量一般用来存储临时性的数据。

在 Java 当中，声明变量的语法如下：

```
// 变量类型 变量名 = 变量值;
int a = 1;
```

在 JDK 11 当中，你可以使用 var 关键字自动推断变量的类型：`var a = 1;`

当然，你也可以创建一个空的变量，如：`int a;`

但一般不建议这样做，因为如果后续没有对其进行赋值的话，会产生错误并导致程序崩溃。

2.4 基本类型

Java 当中有以下几种基本数据类型：

- byte
- short
- int：整数类型（1、2...；可有符号）

- long：与整数一样，但可存储上限更高
- float：单精度浮点类型（即小数；存储精确数值时会丢失精度）
- double： 双精度浮点类型（同样不能存储精确数值）
- boolean：布尔类型（即 true 与 false，表示逻辑是与否）
- char

并不需要记忆这些数据类型的大小范围，只需要记得其用处即可。

除此之外，还有一些其他的内置类型，如 String 用于字符串。这些类型不同于以上提到的基本类型，而是引用类型，会在之后提到。

2.5 运算符

2.5.1 算术运算符

算数运算符有以下几种：

- 加法运算符：+
- 减法运算符：-
- 乘法运算符：*
- 除法运算符：/
- 取余运算符：%

其中，可以使用括号 () 进行计算优先级的规定。如 (a + b) * c 中，a + b 会优先计算。

对于加法和减法运算符，可以使用 ++ 和 -- 进行自增的操作。如：a++

等价于 a = a + 1

还有另一种类似于自增运算符的运算符叫赋值运算符。对于以上算术运算符，可以使用例如 a += 2 的操作实现 a = a + 2

例子：

```
var a = 1;
var b = 2;
var c = (a + b) * 4;

assertTrue(c == 12);
```

2.5.2 关系运算符

关系运算符有以下几种：

- 等于：==
- 不等于：!=
- 大于：>
- 小于：<
- 大于等于：>=
- 小于等于：<=

可用于比较两个值的大小，如：a > b

这个表达式将会返回一个 Boolean 类型，即如果 a > b，则返回 true

例子：

```
var a = 3;
var b = 1;
var c = a > b;
```

```
assertFalse(c);
```

2.5.3 逻辑运算符

逻辑运算符有以下几种：

- 与（and）：`&&`
- 或（or）：`||`
- 非（not）：`!`

这些运算符需要作用于 `Boolean` 类型的数据。

例子：

```
var a = true;
var b = false;
var c = a && b;

assertFalse(c);
```

2.6 方法 (Method)

Java 中，函数（function）被叫做方法（method）。方法必须声明于类当中，如：

```
public class Main {
  public static void main(String[] args) {
    // Do something
  }

  // 你声明的方法
  void a() {
    // Do something
  }
}
```

方法的声明如下：

```
// 方法声明格式：返回值类型 方法名(参数类型 参数名) { 方法体 }

// 一个没有返回值没有参数的方法
void a() {
  System.out.println("Hello World!");
}

// 一个有返回值的方法，但没有参数
int b() {
  return 1;
}

// 一个有返回值有参数的方法
int c(int x, int y) {
  return x + y;
}
```

可以看到方法可以有一个返回值和多个参数。类似于数学，你可以往 $f(x)$ 当中提供一个参数，即 x，其返回值就是 $f(x)$。

方法的调用如下：

```
int f(int x) {
  return x + 1;
}

System.out.println(f(1)); // 返回: 2
```

2.7 if-else 控制

程序当中会有很多逻辑的操作，比如需要判断各种条件。此时就需要用到 `if-else` 控制。其用法如下：

```
var a = true;
var b = false;
var c = false;
if (a) {
  // Do something
} else if (b) {
  // Do something
} else if () {
  // Do something
} else {
  // Do something
}
```

`if` 和 `else if` 后括号中需要填写一个 `Boolean` 类型的数据。当括号中的数据为 `true` 时，则会执行其对应大括号当中的代码。

2.8 for 控制

`for` 用于进行代码的循环。其格式如下：

```
for (var a = 0; a < 5; a++) {
  System.out.println(a);
}

/*
 * 输出:
 * 0
 * 1
 * 2
 * 3
 * 4
 */
```

可以将这段代码用求和符号表达 $\sum_{a=0}^{4} \mathrm{print}(a)$

3 作业：Lab

请写一个方法，使其接受三个参数。其中一个参数是用字符串表达的运算符类型，另两个是运算符作用于的两个变量。方法的返回值是使用接收到的运算符对输入的两个变量运算后的结果。

在完成以上方法之后，请枚举 0 到 5 分别作为加法运算符两边变量的所有情况并输出。例如：$0+0$,$0+1$,$0+2$,...,$5+5$

请在下周活动之前将代码私发给我。