

初步Java开发

一，第一个程序

几乎所有的语言第一个程序都是“Hello ,world”，因为最初C语言出现的第一编程序写的就是“你好，世界!”所以以后的第一个程序都将是“Hello,world”。

如果要编写Java程序，那就需要一个记事本类的工具进行编写，所有Java的编程序后缀都是Java程序。

建立一个hello, world的源程序，后缀为Java，本次使用编写工具是EditPlus，使用EditPlus打开你的Hello.java程序，然后进行编写。

Java程序是需要经过两次编写后此案可以正常执行的：

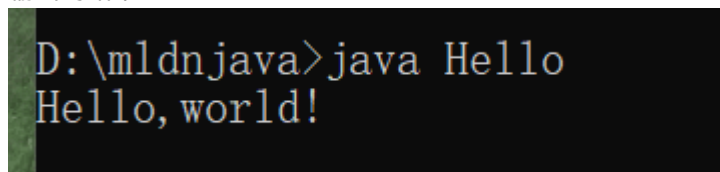
对源代码程序进行编译：javac Hello.java（你可以使用cmd输入 javac），会出现一个有Hello.class的字节码文件，就属于编译后的文件，

可以利用JVM 进行编译，编译出的一套于平台无关的字节码文件（.class）；

在JVM上对程序进行解释执行：Java helo

解释的就是字节码文件，字节码文件的后缀是不需要编写的；

输出的结果：



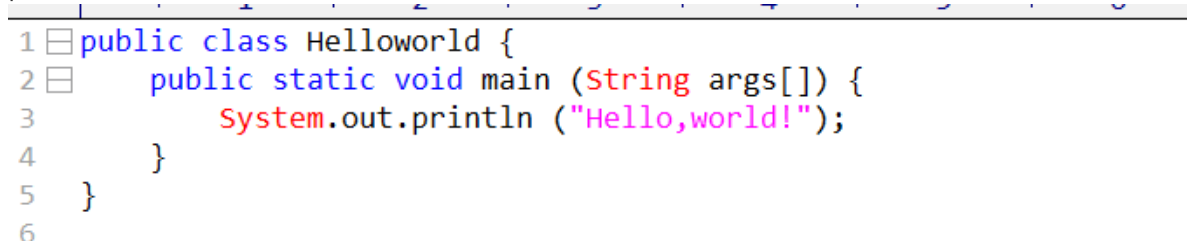
```
D:\mldnjava>java Hello
Hello,world!
```

在Java程序开发之中，最基础的单元是类，所有的程序都必要封装在类中执行，而类的基本定于语法如下：

```
public class Hello {
    public static void main (String args[]) {
        System.out.println ("Hello,world!");
    }
}
```

定义的名称为 hello，而类的定义有两种形式：

public class 类的名称 { }；类名称必须于文件名称保持一致。否则如下：



```
1 public class Helloworld {
2     public static void main (String args[]) {
3         System.out.println ("Hello,world!");
4     }
5 }
6
```

如果不一致则使用public 编译不了文件。

class 类名称 { }：类名称可以与文件名称不一致（如图前二），但是编译后的 .class 名称是 class 定义的名称，解析的要求解析的是生成的 .class名称。

```
D:\mldnjava>java Helloworld
Hello, world!
```

```
class Helloworld {
    public static void main (String args[]) {
        System.out.println ("Hello,world!");
    }
}
```

在一个 .java 文件里面可以有多个class文件，而且编译之后会形成不同的.class文件。

public 和class 总结：

在一个public定义中，只可以一个定义

而在class定义中，可以定义多个文件。

1.关于以后类源代码定义问题：

在以后进行项目开发的时候，很少会出现一个.java源代码而言，很少有class定义很多的一个文件，其实定义一个 public class 文件一个就够了，因为定义多个class文件会产生混乱，定义多个class类只是方便学习。

Java语言有着明确的命名要求，定义类的名称首字母必须要大写

正确的示范：

```
public class Hello {
    public static void main (String args[]) {
        System.out.println ("Hello,world!");
    }
}
```

错误的示范：

```
public class hello {
    public static void main (String args[]) {
        System.out.println ("Hello,world!");
    }
}
```

2.主方法：主方法是所有程序的执行起点，并且一定要定义在类当中，

Java的主方法定义：

```
public class 类名称 {
    public static void main (String [] args ) {
        程序代码由此来执行
    }
}
```

Java的主方法名称定义非常长，主方法所在的类就叫做 主类

3.屏幕打印（系统输出）

可以直接在命令行方式下进行内容显示，有两类语法形式

输出之后追加换行：system.out.println(输出内容)

```
public class Hello {
    public static void main (String args[]) {
        System.out.print("Hello,");
        System.out.println("world!");
        System.out.println("Hello.world!");
    }
}
```

```
}  
}
```

输出之后不追加换行：system.out.print(输出内容)，ln（line，换行）

```
public class Hello {  
    public static void main (String args[]) {  
        System.out.println("Hello,world!");  
        System.out.println("Hello,world!");  
        System.out.println("Hello.world!");  
    }  
}
```

二，JShell工具

shell是脚本程序的含义，在很多的编程语言当中为了方便使用，代码开发，都会提供shell交互式的编程环境。为了解决一些很麻烦的问题，所以提供了 jshell 指令，直接在cmd运行即可。

除了可以在jshell进行程序的编写，也可以将内容直接交给一些文件进行保存。

比如说你在D目录新建一个txt文件，其内容为：

```
System.out.println("Hello,world");  
System.out.println("Hello,world");
```

之后在jshell里面输入 ./open d:/mldn.txt 命令，你会发现直接将hello,world打印在屏幕上。你会发信整个只用编写核心代码即可。如果退出jshell你可以输入 /exit 命令，她会跟你说再见的。

三，CLASSPATH环境属性

比如说你在C盘执行java Hello，会出现如下

错误: 找不到或无法加载主类 Hello

原因: java.lang.ClassNotFoundException: Hello

出现这个的原因就是在C盘没有字节码。那么这样就是可以在不同的目录中执行字节码文件，那么就需要依靠CLASSPATH环境属性来完成。

那么如何定义CLASSPATH环境属性？，在cmd页面中输入 SET CLASSPATH=d:\mldnjava 先定义CLASSPATH，之后在输入 java Hello，最终将Hello.world打印在屏幕上。

最后可以得出一个结论，JVM解释程序的时候需要得到CLASSPATH的支持。

但是在默认的情况在，所有类都是在当前目录下所执行的，CLASSPATH应该采用默认的设置方式，否则将对你产生混乱。

1.1从当前所在路径加载类 SET CLASSPATH=.

但是一点是，Java在你下载其他程序的情况下，他会自动设置修改CLASSPATH，在这种情况下就必须用自己用命令设置回来，但是我们要注意的是，现在CLASSPATH只是在一个命令行下的配置，如果我们把命令行关闭了，那么你所做的相关配置就会消失掉了。

那么最好的做法就是将其定义为全局属性。则可以在系统中追加一个属性信息。变量为CLASSPATH，其值是 .

面试题：PATH和CLASSPATH的区别

PATH：是操作系统提供的路径配置，定义所有可执行的路径

CLASSPATH：是JRE提供的，用于在JAVA程序解释时类加载路径，其默认设置的方式为当前所在的目录加载，默认设置为当前所在的加载目录，可以通过“SET CLASSPATH=路径”的命令形式来进行定义。

|- 关系：JVM ——>CLASSPATH的定义路径 ——>加载字节码文件