1. 初始Java

【本节目标】

1. **Java语言简介、发展概述、语言优势、与C/C++区别**
2. **初识Java程序入口之main方法**
3. **注释、标识符、关键字**

1. Java语言概述

1.1 Java是什么

Java是一种优秀的程序设计语言，它具有令人赏心悦目的语法和易于理解的语义。

不仅如此，**Java还是一个有一系列计算机软件和规范形成的技术体系**，这个技术体系提供了完整的用于软件开发和跨平台部署的支持环境，并广泛应用于嵌入式系统、移动终端、企业服务器、大型机等各种场合。

1.2 Java语言重要性

1. **语言广泛使用程度**

   下图数据来自于**TIOBE编程语言社区**2017年12月和2018年10月最新的排行榜，常年占据语言排行榜榜首，是近些年最火的编程语言之一。

Oct 2018	Oct 2017	Change	Programming Language	Ratings	Change
1	1		Java	17.801%	+5.37%
2	2		C	15.376%	+7.00%
3	3		C++	7.593%	+2.59%
4	5	↑	Python	7.156%	+3.35%
5	8	↑	Visual Basic .NET	5.884%	+3.15%
6	4	↓	C#	3.485%	-0.37%
7	7		PHP	2.794%	+0.00%
8	6	↓	JavaScript	2.280%	-0.73%
9	-	⇈	SQL	2.038%	+2.04%
10	16	⇈	Swift	1.500%	-0.17%
11	13	↑	MATLAB	1.317%	-0.56%
12	20	⇈	Go	1.253%	-0.10%
13	9	⇊	Assembly language	1.245%	-1.13%
14	15	↑	R	1.214%	-0.47%
15	17	↑	Objective-C	1.202%	-0.31%
16	12	⇊	Perl	1.168%	-0.80%
17	11	⇊	Delphi/Object Pascal	1.154%	-1.03%
18	10	⇊	Ruby	1.108%	-1.22%
19	19		PL/SQL	0.779%	-0.63%
20	18	↓	Visual Basic	0.652%	-0.77%

TIOBE 编程语言社区排行榜是编程语言流行趋势的一个指标，每月更新，这份排行榜排名基于互联网上有经验的程序员、 课程和第三方厂商的数量。排名使用著名的搜索引擎（诸如 Google、MSN、Yahoo!、Wikipedia、YouTube 以及 Baidu 等）进行计算。

注意：上述排名不能说明那个语言好，那个语言不好，每门编程语言都有适应自己的应用场景。

2. **工作领域**

Java语言目前在IT领域的应用是非常广泛的，掌握Java语言可以从事不少IT行业的相关开发岗位，具体包括：

- **企业级系统**

  比如大型复杂的企业级软件系统，Java的安全机制以及跨平台性的优势，其在分布式系统领域开发中有广泛应用，涉及到金融、电信、交通、电子商务、ERP系统等。

- **Web开发领域**

  Java语言在设计初期，赶上了互联网发展的风口，当时就瞄准了互联网开发，凭借稳定的性能表现和较好的扩展性，Java语言一直是大型互联网平台的重要解决方案。

- **android平台应用**

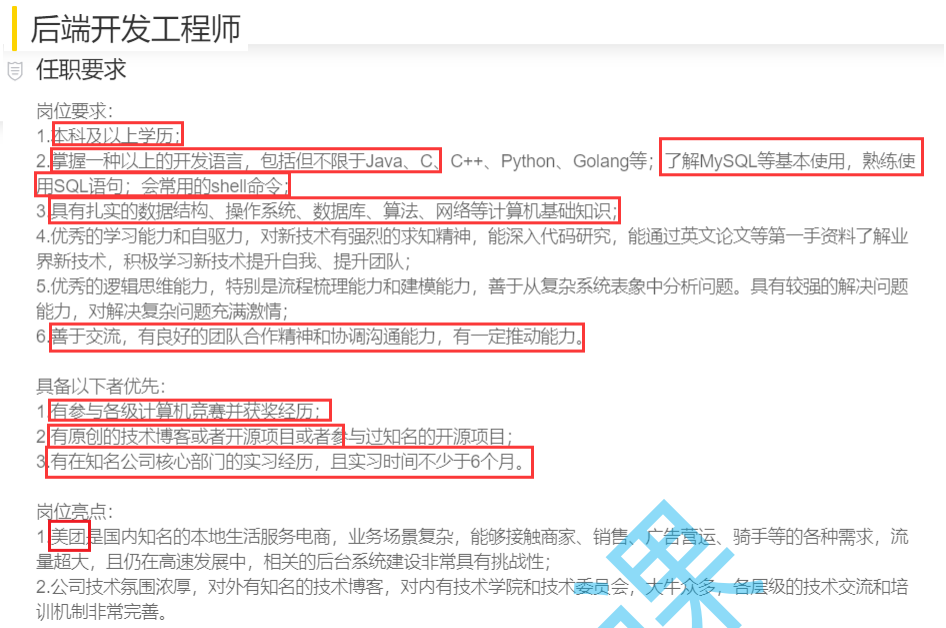
  Android是一种智能手机操作系统，Java是一门非常流行的编程语言。Android上的应用程序就是大多是用Java编写的，Android的SDK大部分就是直接将Java SDK翻译过来的，所以具有Java基础，也可以快速上手Android开发。

- **大数据平台开发**

  大数据相关的各类框架，比如：Hadoop、spark、storm、flink等，以及各种中间件flume、kafka、sqoop等，这些框架以及工具等大多数是用Java语言开发的，随着大数据技术的落地应用，Java在大数据领域的应用前景也是比较广阔的。

除上述开发领域外，Java在游戏领域、人工智能领域、科学计算领域、嵌入式领域也有一定的应用。因此学好Java，将来就业的选择也会非常广泛。

3. **在校招中的岗位需求**



后端开发工程师

任职要求

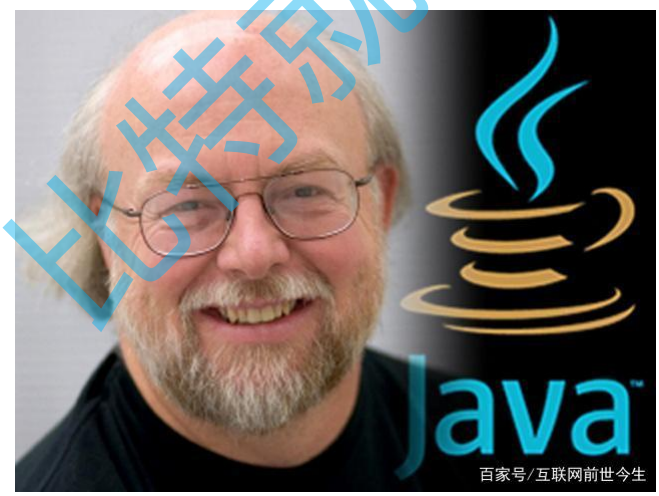
岗位要求：
1.本科及以上学历；
2.掌握一种以上的开发语言，包括但不限于Java、C、C++、Python、Golang等；了解MySQL等基本使用，熟练使用SQL语句；会常用的shell命令；
3.具有扎实的数据结构、操作系统、数据库、算法、网络等计算机基础知识；
4.优秀的学习能力和自驱力，对新技术有强烈的求知精神，能深入代码研究，能通过英文论文等第一手资料了解业界新技术，积极学习新技术提升自我、提升团队；
5.优秀的逻辑思维能力，特别是流程梳理能力和建模能力，善于从复杂系统表象中分析问题。具有较强的解决问题能力，对解决复杂问题充满激情；
6.善于交流，有良好的团队合作精神和协调沟通能力，有一定推动能力。

具备以下者优先：
1.有参与各级计算机竞赛并获奖经历；
2.有原创的技术博客或者开源项目或者参与过知名的开源项目；
3.有在知名公司核心部门的实习经历，且实习时间不少于6个月。

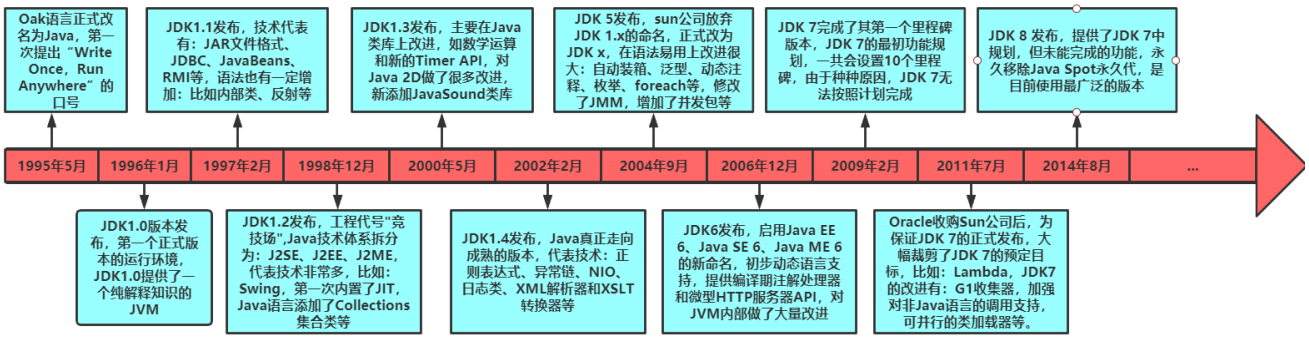
岗位亮点：
1.美团是国内知名的本地生活服务电商，业务场景复杂，能够接触商家、销售、广告营运、骑手等的各种需求，流量超大，且仍在高速发展中，相关的后台系统建设非常具有挑战性；
2.公司技术氛围浓厚，对外有知名的技术博客，对内有技术学院和技术委员会，大牛众多，各层级的技术交流和培训机制非常完善。

从上述位置需求中可以看出，由于应届生缺少实际工作经验，因此校招中企业更看重学生的基础，也就是：语言、数据结构/算法、操作系统、网络、数据库等，其他的属于加分项。

1.3 Java语言发展简史



Java 语言源于 1991 年 4 月，Sun 公司 **James Gosling博士** 领导的绿色计划(Green Project) 开始启动，此计划最初的目标是开发一种能够在各种消费性电子产品(如机顶盒、冰箱、收音机等)上运行的程序架构。这个就是Java的前身： Oak (得名与Java创始人James Gosling办公室外的一棵橡树)，但由于这些智能化家电的市场需求没有预期的高，Sun公司放弃了该项计划。随着**1995年代互联网的发展**，Sun公司看见**Oak在互联网上应用的前景，于是改造了Oak，于1995年5月以Java的名称正式发布，并提出"Write once, Run anywhere" 的口号。**



1. 2002年左右，微软.NET Framework发布，无论技术实现还是目标用户与Java有很多相似之处
2. JDK6发布后，2006年11月13日，在JavaOnce大会上，sun公司宣布Java开源
3. 在JDK7开发期间，sun公司相继在技术急症和商业竞争中陷入泥潭，公司股票跌至尖峰时的3%，无力推进JDK7的研发，Oracle以74亿美元收购sun公司(sun尖峰时期市值超过2000亿美元)
4. 2017年9月JDK 9发布
5. 2018年3月JDK 10发布
6. 2018年9月JDK 11发布，是长期支持的一个版本
7. 2018年3月，Android 侵权案最终判决，Google赔偿Oracle合计88亿美元，让收购Sun公司等同免费，同年，Oracle将无法盈利也没有战略前景的部分“按计划报废”，宣告Java EE称为历史名词，直接捐赠给Eclipse基金会
7. 2019年3月JDK 12发布

1.4 Java语言特性

以下Java语言特性来自于Java白皮书：

1. **简单性**

   Java语法是C++语法的一个“纯净版本”，相当于对C++做了一个减法。这里没有头文件、指针运算（甚至指针语法）、结构、联合、操作符重载、虚基类等等。不仅如此，Java开发环境远远超出大多数其他编程语言的开发环境。

2. **面相对象**

   什么是面向对象？

   在Java的世界里，一切皆对象。比如：人、狗、手机、电脑等都是对象。所谓面相对象，就是依靠对象之间的交互来完成事情，比如：人用手机网上购物，狗吃骨头...

   Java的面向对象特性与C++旗鼓相当，与C++的主要不同点在于多重继承。在Java中，取而代之的是更简单的接口概念。而且与C++相比，Java提供了更丰富的运行时自省功能。

3. **分布式(微服务)**

   Java有丰富的例程库，用于处理像HTTP和FTP之类的TCP/IP协议。Java应用程序能够通过URL打开和访问网络上的对象，其便捷程度就好像访问本地文件一样。

4. **健壮性**

   Java与C++最大的不同在于Java采用的指针模型可以消除重写内存和损坏数据的可能性（对于曾经花费几个小时来检查由于指针bug而引起内存冲突的人来说，一定很喜欢Java的这一特性）。不仅如此，Java编译器能够检测许多在其他语言中仅在运行时才能够检测出来的问题。

5. **安全性**

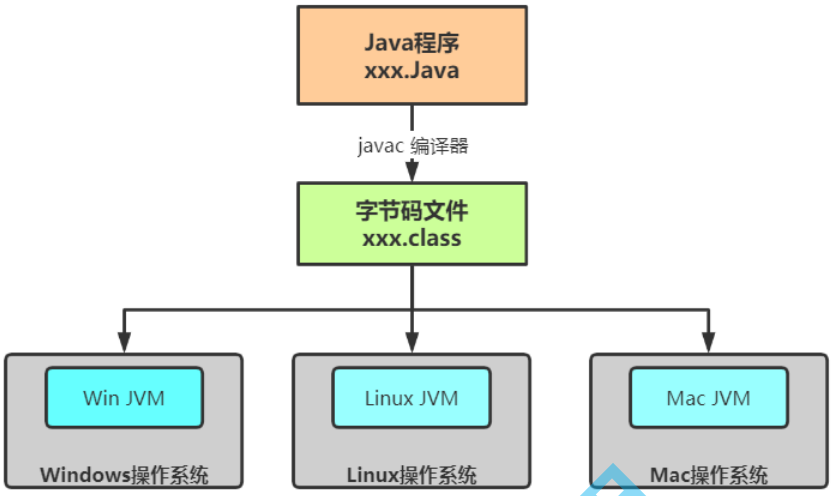
   Java适用于网络/分布式环境。为了达到这个目标，在安全性方面投入了大量的精力。使用Java可以构建防病毒、防篡改的系统

   从一开始，Java就设计成能够防范常见的各种攻击：

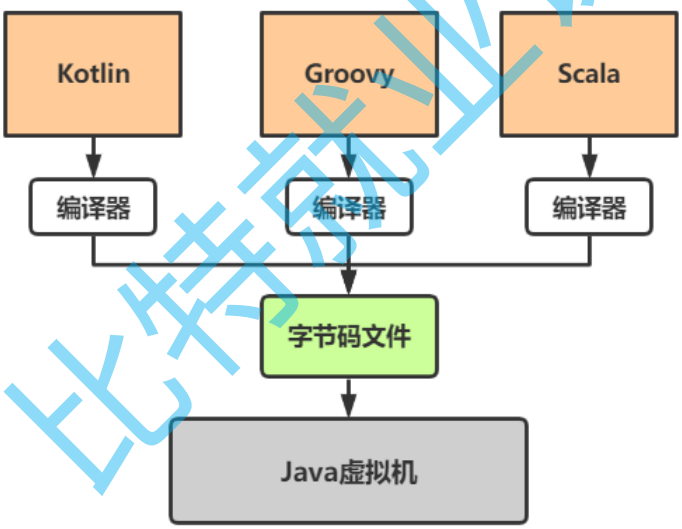
   > - 运行时堆栈溢出。蠕虫和病毒常用的攻击手段。
   > - 破坏自己进程空间之外的内存。
   > - 未经授权读写文件

6. **体系结构中立**

编译器生成一个体系结构中立的目标文件格式，按照该中规范生成的文件，只要有Java运行时系统，这些编译后的代码就可以在许多处理器上运行。**Java编译器通过生成与特定计算机体系结构无关的字节码指令来实现这一特性。**精心设计的字节码不仅可以很容易的在任何机器上解释执行，而且还可以动态地翻译成本地机器代码。这就是为什么可以：**“Wrice once，Run anywhere”。**



而且其他语言编写的程序，在编译后如果能够严格按照字节码文件的规范生成.class文件，也可以在JVM上运行。



7. **可移植性**

   与C/C++不同，Java规范中没有“依赖具体实现的地方”。**基本数据类型的大小以及有关运算都做了明确的说明。**例如，Java中的int永远是32位的整数，而在C/C++中，int可能是16位整数、32位整数，也可能是编译器提供商指定的其他大小。在Java中，数据类型具有固定的大小，这消除了代码移植时令人头疼的主要问题。

8. **解释性**

   Java为了实现与平台无关，自己维护了一套基于栈架构的指令集，Java源代码经过编译之后，字节码文件中的指令就是按照自己的指令集来组织的，但是在具体硬件环境中运行时，系统并不能识别，因为Java程序在执行时，Java解释器会逐条的将字节码文件中的指令翻译成CPU的指令集。

9. **高性能**

   边解释边执行，垃圾会说等导致了Java代码运行效率偏低，近年来JVM也在不断的优化，比如：JIT(即时编译器)，热点代码探测，让Java程序的执行效率大幅提高，在有些场合不亚于C/C++。

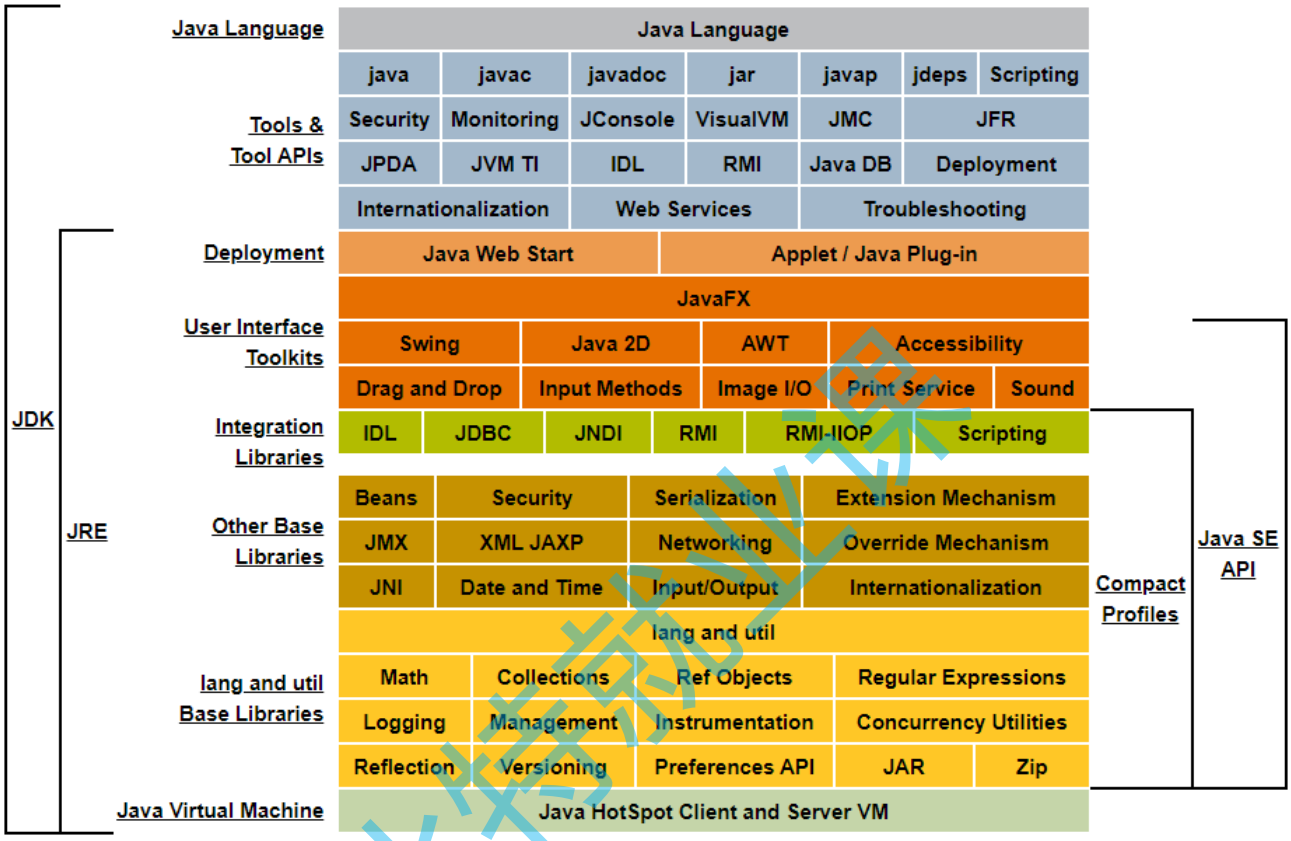
10. **多线程**

Java在当时很超前。它是第一个支持并发程序设计的主流语言。多线程可以带来更好的交互响应和实时行为。并发程序设计绝非易事，但是Java在这方面表现出色，可以很好的管理这个工作。

11. **动态性**

    Java与C/C++相比更加具有动态性。它能够适应不断发展的环境。库中可以自由的添加新方法和实例变量，而对客户端没有任何影响。在Java中找出运行时类型信息十分简单（反射的特性，后续会学到）

因此：**Java不仅仅是一门编程语言，也是一个由一些列计算机软件和规范组成的技术体系。**



1.5 Java开发环境安装

- 可能是Windows下最简单的Java环境安装指南
- Linux下JDK的安装（多种方式）
- Mac下JDK的安装
- Java 软件 https://pan.baidu.com/s/1X7zPb-YT11xR_UDqjN-oJw 提取码:r471

2. 初识Java的main方法

2.1 main方法示例

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

}
```

如上展示的就是最简单的一个Java程序，可能同学们看到后一头雾水，可以说，Java的main方法应该是当前主流编程语言中最“长”的。

通过上述代码，我们可以看到一个完整的Java程序的结构，Java程序的结构由如下三个部分组成：

> 1.源文件（扩展名为*.java）：源文件带有类的定义。类用来表示程序的一个组件，小程序或许只会有一个类。类的内容必须包含在花括号里面。

> 2.类：类中带有一个或多个方法。方法必须在类的内部声明。

> 3.方法：在方法的花括号中编写方法应该执行的语句。

> 总结一下：类存在于源文件里面；方法存在于类中；语句存在于方法中。

注意：在一个源文件中只能有一个public修饰的类，而且源文件名字必须与public修饰的类名字相同。

好了，代码编写完了，如何让它“运行”起来呢？

2.2 运行Java程序

Java是一门半编译型、半解释型语言。先通过javac编译程序把源文件进行编译，编译后生成的.class文件是由字节码组成的平台无关、面向JVM的文件。最后启动java虚拟机来运行.class文件，此时JVM会将字节码转换成平台能够理解的形式来运行。

1. 使用记事本或者IDEA(集成开发环境)编写Java源程序

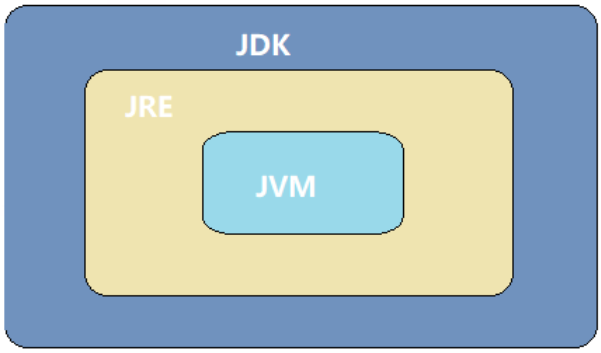
2. 使用javac.exe编译器编译Java源程序，生成xxx.class的字节码文件　语法格式：javac xxx.java

3. 使用java运行xxx.class字节码文件　语法格式：java xxx.java

注意：在运行Java程序前，必须先安装好JDK(Java Development Kit即Java开发工具包)，JDK里面就包含了javac和java工具，Java程序最终是在JVM(Java虚拟机)中运行的。

【面试题】JDK、JRE、JVM之间的关系？

> - JDK(Java Development Kit):Java开发工具包，提供给Java程序员使用，包含了JRE，同时还包含了编译器javac与自带的调试工具Jconsole、jstack等。
> - JRE(Java Runtime Environment):Java运行时环境，包含了JVM，Java基础类库。是使用Java语言编写程序运行的所需环境。
> - JVM：Java虚拟机，运行Java代码



JDK = JRE + 开发工具集(比如：javac)

JRE = JVM + JavaSE标准类库

编写和运行第一个Java程序时，可能会遇到的一些错误：

1. 源文件名后缀不是.java
2. 类名与文件名不一致
3. main方法名字写错：mian
4. 类没有使用public修饰
5. 方法中语句没有以分号结尾
6. 中文格式的分号
7. JDK环境没有配置好，操作系统不能识别javac或者java命令

3. 注释

注释是为了让代码更容易被读懂而附加的描述信息. 不参与编译运行, 但是却非常重要.

时刻牢记! 代码写出来是为了给人看的, 更是为了给三个月后的你自己看的.

3.1 基本规则

Java中的注释主要分为以下三种

- 单行注释：// 注释内容（用的最多）
- 多行注释：/* 注释内容*/（不推荐）
- 文档注释： /** 文档注释 */（常见于方法和类之上描述方法和类的作用），可以被javadoc工具解析，生成一套以网页文件形式体现的程序说明文档

注意：

1. 多行注释不能嵌套使用
2. 不论是单行还是多行注释，都不参与编译，即编译之后生成的.class文件中不包含注释信息。

```
/**
文档注释:
   @version v1.0.0
   @author will

   作用HelloWorld类，入门第一个程序练习
*/
public class HelloWorld{
   /*
```

```
    多行注释：
     1. main方法是Java程序的入口方法
     2. main函数的格式是固定的，必须为public static void main(String[] args)
    */

    /**
    main方法是程序的入口函数
    @param args 命令行参数。
    */
    public static void main(String[] args){
        // 单行注释：System.out.println是Java中标准输出，会将内容输出到控制台
        System.out.println("Hello World");
    }
}

--------------------------------------------------------------------------------
// 在cmd中，使用javadoc工具从Java源码中抽离出注释
// -d 创建目录  myHello为目录名
// -author 显示作者
// -version 显示版本号
// -encoding UTF-8 -charset UTF-8 字符集修改为UTF-8
javadoc -d myHello -author -version -encoding UTF-8 -charset UTF-8 HelloWorld.java
```

3.2 注释规范

1. 内容准确: 注释内容要和代码一致, 匹配, 并在代码修改时及时更新.
2. 篇幅合理: 注释既不应该太精简, 也不应该长篇大论.
3. 使用中文: 一般中国公司都要求使用中文写注释, 外企另当别论.
4. 积极向上: 注释中不要包含负能量(例如 领导 SB 等).

4. 标识符

在上述程序中，Test称为类名，main称为方法名，也可以将其称为标识符，即：**在程序中由用户给类名、方法名或者变量所取的名字。**

【硬性规则】

标识符中可以包含：字母、数字以及 下划线和 $ 符号等等。

注意：标识符不能以数字开头，也不能是关键字，且严格区分大小写。

【软性建议】

- 类名：每个单词的首字母大写(大驼峰)
- 方法名：首字母小写，后面每个单词的首字母大写(小驼峰)
- 变量名：与方法名规则相同

一个大型的工程，是由多名工程师协同开发的，如果每个人都按照自己的方式随意取名，比如：person、PERSON、Person、_person，将会使程序非常混乱。如果大家在取名时能够遵守一定的约束(即规范)，那多人写除的代码仿佛一个人写的。

下面那些标识符是合法的?

A：class B：HelloWorld C：main D：123abc E：ARRAY_SIZE F: $name G: name:jim

5. 关键字

通过观察上述程序可以发现，public、class以及static等颜色会发生变化，将这些具有特殊含义的标识符称为关键字。即：**关键字是由Java语言提前定义好的，有特殊含义的标识符，或者保留字。**

注意：用户不能使用关键字定义标识符。

在Java中关键字有很多，这里给大家列出来一部分，先了解下后序在逐一详细解释。

用于定义访问权限修饰符的关键字				
private	protected	public		
用于定义类，函数，变量修饰符的关键字				
abstract	final	static	synchronized	
用于定义类与类之间关系的关键字				
extends	implements			
用于定义建立实例及引用实例，判断实例的关键字				
new	this	super	instanceof	
用于异常处理的关键字				
try	catch	finally	throw	throws
用于包的关键字				
package	import			
其他修饰符关键字				
native	strictfp	transient	volatile	assert