今日学习目标

* 能够阐述JDK和JRE之间区别
* 能够使用常见的DOS命令
* 能够独立安装JDK工具
* 能够编写HelloWorld源文件
* 能够编译HelloWorld.java
* 能够运行HelloWorld.class文件
* 能够独立配置Path环境变量
* 能够在源代码中使用注释
* 能够解释关键字的概念
* 能够理解常量的概念和分类
* 能够复述变量的概念
* 能够记住变量的定义格式
* 能够定义八种数据类型变量 byte,short,int,long, float,double, char,boolean
* 能够使用标识符定义变量和类名

# 前言

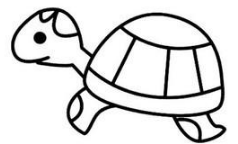
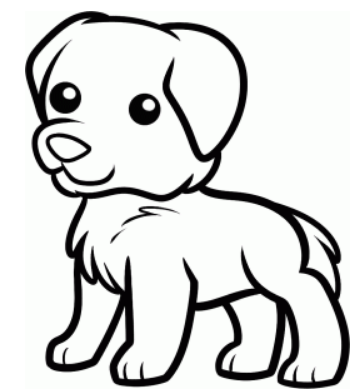
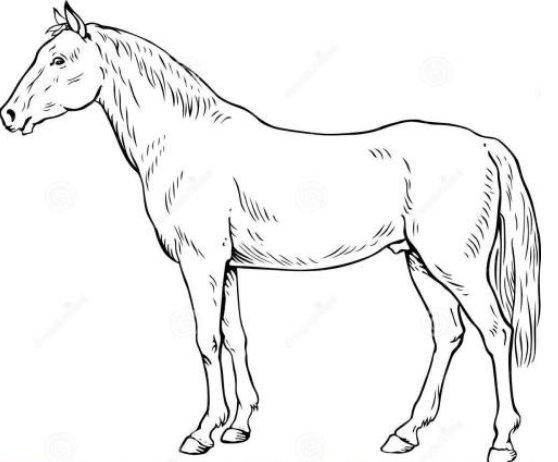
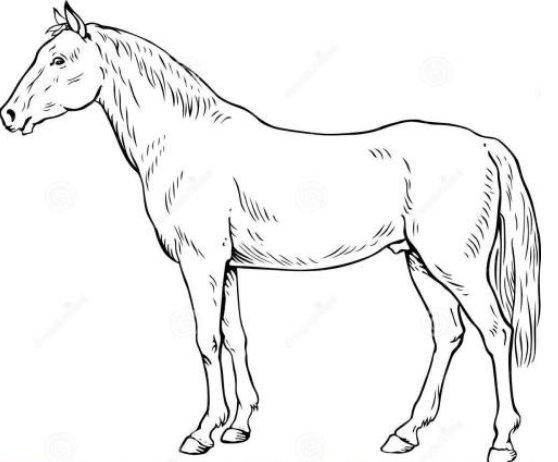
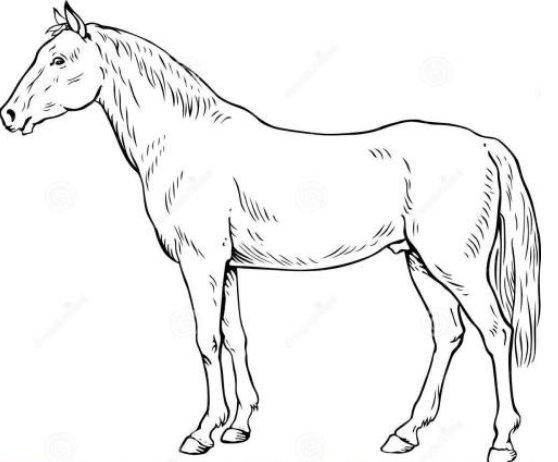
## 课程介绍

* 层次化设计(步步深入)：只讲解这个阶段使用的内容,并没有把全部的知识点都进行讲解。
* JavaSE基础内容11天
  + 基础语法5天
    - 任何编程语法都有相通的地方,学习了Java,再学其他的就比较轻松。
    - 后期的内容学习,都离不开基础语法
  + 基础语法练习强化1天
  + 面向对象1天
    - 面向对象的思想,特点,类和对象,创建对象,调用对象的方法
  + 常用API(字符串,集合,IO流)3天
    - 多个对象进行操作(存储,取出),文本内容进行处理(读取,写入)
    - 完成集合版的学生管理系统
  + 常用API练习强化1天
    - 完成IO版的学生管理系统
* 就业班前期讲解JavaSE的进阶内容

# Java入门

## Java语言概述

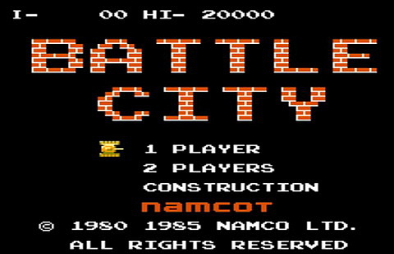
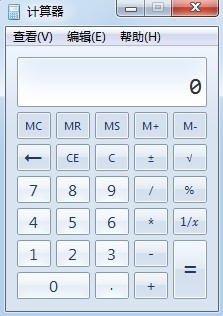
### Java语言发展史(了解)

* 詹姆斯·高斯林（James Gosling）
* SUN (Stanford University Network斯坦福大学网络公司)
* 1995年5月23日 Java语言诞生
* 1996年1月 JDK1.0 
* 1997年2月 JDK1.1 
* 1998年12月 JDK1.2(将Java分成了J2SE,J2EE,J2ME) 
* 2000年5月 J2SE1.3 
* 2002年2月 J2SE1.4(1.4.2版本很多公司在使用) 
* 2004年10月 JDK1.5(改名JavaSE5.0,JavaEE5.0,JavaME5.0) 
* 2006年12月 JavaSE6.0
* 2009年4月20日 甲骨文(Oracle)74亿美元收购Sun。
* 2011年7月 JavaSE7.0(市场主流版本)
* 2014年3月 JavaSE8.0
* 2017年9月 JavaSE9.0

### Java语言平台概述

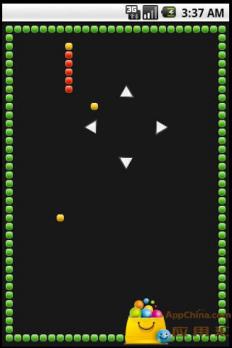
* JavaSE / J2SE(Java 2 Platform Standard Edition)标准版

是为开发普通桌面和商务应用程序提供的解决方案，该技术体系是其他两者的基础，可以完成一些桌面应用程序的开发。

* JavaME / J2ME(Java 2 Platform Micro Edition)小型版

是为开发电子消费产品和嵌入式设备提供的解决方案。

* JavaEE / J2EE(Java 2 Platform Enterprise Edition)企业版

是为开发企业环境下的应用程序提供的一套解决方案,该技术体系中包含的技术如 Servlet、JSP等，主要针对于Web应用程序开发。



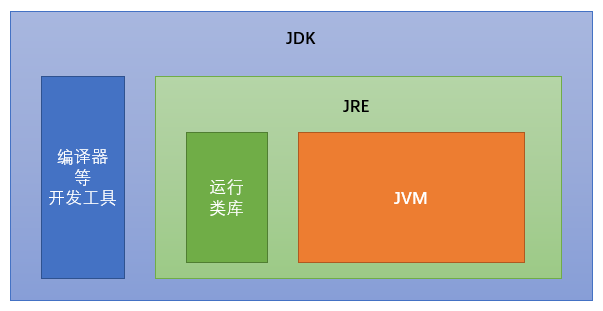
### JRE和JDK



JDK : JRE+开发工具

JRE： JVM+核心类库

JVM：保证语言的跨平台

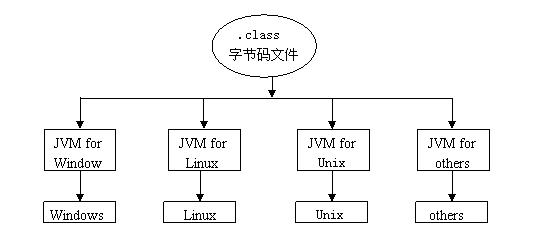


### Java语言跨平台原理

* 平台：指的是操作系统(Windows，Linux，macOS)
* 跨平台：Java程序可以在任意操作系统上运行，“一次编写，到处运行”。

即Java的.class程序可以在不同操作系统上均可以运行。(Java的.class文件相当于Windows中的.exe文件，是可执行的程序。)

* 原理：实现跨平台需要依赖Java的虚拟机 JVM (Java Virtual Machine)



* Java语言是跨平台的，Java虚拟机不是跨平台的。

## 常用DOS命令

* 打开控制台

Win + R，然后cmd回车

* 常用命令

: 回车 盘符切换 d: c: e:

dir: 列出当前目录下的文件以及文件夹

cd: (进入指定目录)

进入 cd 目录；cd 多级目录

回退 cd.. ；cd\

cls : (clear screen)清屏

exit : 退出dos命令行

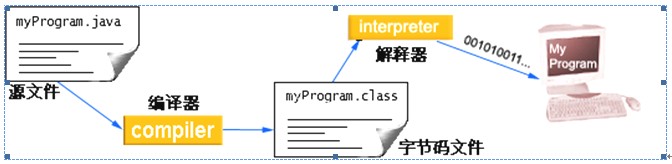
运行应用程序：程序完整名称 参数

一个快捷方式,在目录上直接输入cmd就进入了当前目录的DOS界面

## \*JDK的下载与安装

* JDK的下载(不演示)
  + 通过官方网站获取JDK
  + http://www.oracle.com
  + 针对不同操作系统，下载不同的JDK版本
    - Windows x86
    - Windows x64
  + 请参照(JDK下载安装文档.doc)回去练习
* JDK的安装(演示)
  + 傻瓜式安装，下一步即可。
  + 建议：安装路径不要有中文或者特殊符号如空格等。开发工具最好安装目录统一。

## HelloWorld案例



### \*HelloWorld案例的编写与运行

* 案例的代码编写

Java源代码编写只需要使用最普通的文本编辑器编写即可，例如Windows系统自带的Notepad（记事本）即可。

然后修改文件名称为HelloWorld.java，注意文件名称的大`小写。以及是否已经修改了文件的扩展名。

* + 首先定义一个类

public class 类名

* + 在类定义后加上一对大括号

{}

* + 在大括号中间添加一个主(main)方法/函数

public static void main(String [] args){ }

* + 在主方法的大括号中间添加一行输出语句

System.out.println(“HelloWorld”);

* + 完整代码:

**public** **class** HelloWorld{

**public** **static** **void** main(String[] args){

System.***out***.println("HelloWorld");

}

}

* 案例的代码运行
  + 编译
* 进行源文件所在目录：即进入D:\java目录；
* 我的JDK安装在D:\develop\Java\jdk1.8.0\_1511目录，那么编译器完整名称为：D:\develop\Java\jdk1.8.0\_1511\bin\javac.exe；
* 我的源文件存放在：D:\java\HelloWorld.java；
* 编译命令为：D:\java>D:\develop\Java\jdk1.8.0\_1511\bin\javac.exe HelloWorld.java
  + 运行
    - 进入字节码文件所在目录：即进入D:\java目录；
    - D:\develop\Java\jdk1.8.0\_1511\bin\java.exe HelloWord；
    - 注意，在运行HelloWorld.class文件时不能给出“.class”，切记！！！

### HelloWorld案例常见问题

* 后缀名问题
* 单词拼写问题
  + class Class
  + String string
  + System system
  + main mian
* 中文符号问题
  + 提示非法字符

## Notepad++软件安装和配置

* 为了让我们写的程序错误看起来更直接，我们安装一款高级记事本软件。

Notepad++

* Notepad软件的安装和配置

设置 – 首选项 – 新建 – 默认语言(修改为java)和编码(修改为ANSI)

* 使用Notepad软件编写一个HelloWorld案例

## Path环境变量配置

* 为什么要配置
  + 我们发现每次程序编译与运行都要输入D:\develop\Java\jdk1.8.0\_1511\bin后，才能使用javac.exe与java.exe，这样的方式很麻烦，那么，能否不输入D:\develop\Java\jdk1.8.0\_1511\bin呢？是可以的，通过配置Windows的PATH环境变量来完成。
  + 我们配置path的目的是让Java的bin目录下的.exe命令可以在任意目录下执行，无需进入到javac所在的目录就可以找到javac.exe命令，方便完成程序。
* 如何配置
  + 创建新的变量名称：JAVA\_HOME

计算机-右键属性-高级系统设置-高级-环境变量-系统变量

* + 为JAVA\_HOME添加变量值：JDK安装目录
  + 在path环境变量最前面添加如下内容

%JAVA\_HOME%\bin;

## 注释

* 注释概述
  + 用于解释说明程序的文字
* 注释分类
  + 单行注释

格式： //注释文字

* + 多行注释

格式： /\* 注释文字 \*/

* + 文档注释（了解）

格式：/\*\* 注释文字 \*/

* 注释作用
  + 解释说明程序，提高程序的阅读性

**public** **class** Demo02ZhuShi {

/\*

这是main(主)方法,是程序执行的入口

所有的代码执行都是从main方法开始的

\*/

**public** **static** **void** main(String[] args) {

//这是输出语句,作用是吧()中的内容打印到控制台

System.***out***.println("HelloWorld");

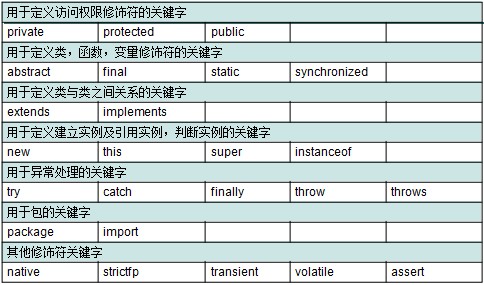
}

}

## 关键字

* 关键字概述
  + 关键字是被Java语言赋予特殊含义，具有专门用途的单词，比如之前接触的class，public，static，void均为Java已经预设好的。你可以先把Java关键字理解为“命令”！
* 关键字特点
  + 组成关键字的字母全部小写
  + 常用的代码编辑器,针对关键字有特殊的颜色标记，非常直观，所以我们不需要去死记硬背，在今后的学习中重要的关键字也会不断的出来。
* Java中的所有关键字展示





# 常量

* 常量概述
  + 在程序执行的过程中，其值不可以发生改变的量
* 常量分类
  + 字符串常量 用双引号括起来的内容(“HelloWorld”)
  + 整数常量 所有整数(12,-23)
  + 小数常量 所有小数(12.34)
  + 字符常量 用单引号括起来的内容(‘a’，’A’，’0’)
  + 布尔常量 较为特有，只有true和false
  + 空常量 null(数组部分讲解)

public class Const {

public static void main(String[] args) {

//字符串常量

System.*out*.println("HelloWorld");

//整数常量

System.*out*.println(12);

System.*out*.println(-23);

//小数常量

System.*out*.println(12.34);

//字符常量

System.*out*.println('a');

System.*out*.println('0');

//布尔常量

System.*out*.println(true);

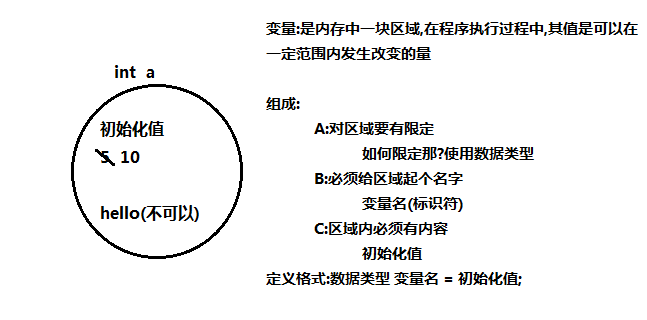
System.*out*.println(false);

}

}

# 变量

* 变量概述
  + 在程序执行的过程中，在某个范围内其值可以发生改变的量
  + 从本质上讲，变量是内存中储存数据的区域，你能用它来存数据和取数据,还可以改变这个数据。
  + 图解:



* 变量定义格式
  + 数据类型 变量名 = 初始化值;
  + 注意：格式是固定的，记住格式，以不变应万变

## 计算机存储单元

* 变量是内存中的小容器，用来存储数据。那么计算机内存是怎么存储数据的呢？无论是内存还是硬盘，计算机存储设备的最小信息单元叫“位（bit）”，我们又称之为“比特位”，通常用小写的字母b表示。而计算机最小的存储单元叫“字节（byte）”，通常用大写字母B表示，字节是由连续的8个位组成。
* 除了字节外还有一些常用的存储单位，大家可能比较熟悉，我们一起来看看：

1byte(B)（字节） = 8bit

1KB = 1024B

1MB = 1024KB

1GB = 1024MB

1TB = 1024GB

1PB = 1024TB

## 数据类型概述

* 数据类型分类

|- 基本数据类型

|- 数值型

|- 整数类型byte,short,int,long

|- 浮点(小数)类型float,double

|- 字符型char

|- 布尔型boolean

|- 引用数据类型

|- 类class

|- 接口interface

|- 数组[]

* 基本数据类型(4类8种)

|  |  |  |  |
| --- | --- | --- | --- |
| **四类** | **八种** | **字节数** | **数据表示范围** |
| **整型**  **(精确)** | **byte** | 1 | -128～127(-27～27-1) |
| **short** | 2 | -32768～32767(-215～215-1) |
| **int(默认)** | 4 | -2147483648～2147483647(-231～231-1) |
| **long** | 8 | -263～263-1 |
| **浮点型**  **(不精确)** | **float** | 4 | -3.403E38～3.403E38  (-3.4\*10的38次方 - 3.4 \*10的38次方) |
| **double(默认)** | 8 | -1.798E308～1.798E308 |
| **字符型** | **char** | 2 | 表示一个字符，如('a'，'A'，'0'，'家') |
| **布尔型** | **boolean** | 1 | 只有两个值true与false |

* 注意事项：

A:整数默认是int类型,浮点数默认是double

B:定义long类型数据的时候，要加L或者l，建议加L

定义float类型数据的时候，要加F或者f，建议加F

C:字符

Java中的字符采用的是Unicode字符集，每一个字符占用两个字节，就可以存储一个汉字

D:boolean类型适用于逻辑运算,一般用于流程控制

## 标识符

* 作用

给包,类,方法,变量等起名字

* 组成规则（**硬性规定**）

由字符，下划线\_，美元符$组成

这里的字符采用的是Unicode字符集，所以包括英文大小写字母，中文字符，数字字符等，但符号只能有两个：下划线\_和美元符$。

* 注意事项（**硬性规定**）

不能以数字开头

不能是Java中的关键字

* 命名规则（**软性建议**）
  + 基本要求

见名知意

* + 常见命名的规则
    - 包 (其实就是文件夹,用于对类进行管理)

全部小写，多级包用点隔开。公司域名的反写。

cn.itcast （相当于两级文件夹：cd\itcast）

com.itheima （相当于两级文件夹：com\itheima）

* + - 类

一个单词首字母大写 Student,Car

多个单词，每个单词首字母大写 HelloWorld

* + - 变量或者方法

一个单词首字母小写。 age,show()

多个单词，从第二个单词开始每个单词首字母大写 maxAge,getAge()

**public** **class** Demo06BiaoZhiFu {

**public** **static** **void** main(String[] args) {

//定义变量

//数据类型 变量名 = 初始化值;

**int** a = 10;

//正确

**int** b2 = 20;

//错误

//int 2b = 30;

//不能是java中的关键字

//错误

//int public = 40;

}

}

## 基本数据类型变量的定义和使用

* 编写案例演示每种不同数据类型的变量定义

**public** **class** Demo07Variable {

**public** **static** **void** main(String[] args) {

//定义byte类型的变量

**byte** b = 99;

System.***out***.println(99);

System.***out***.println(b);

//定义short类型的变量

**short** s = 10000;

System.***out***.println(s);

//定义int类型的变量

**int** i = 1600000000;

System.***out***.println(i);

//定义long类型的变量

**long** l = 6000000000L;

System.***out***.println(l);

//定义float类型的变量

**float** f = 12.34F;

System.***out***.println(f);

//定义double类型的变量

**double** d = 12.34;

System.***out***.println(d);

//定义char类型的变量

**char** c = 'a';

System.***out***.println(c);

//定义boolean类型的变量

**boolean** bb = **false**;

System.***out***.println(bb);

}

}

## 变量定义的注意事项

* 变量未赋值,不能直接使用

引出变量的第二种使用格式

* 变量只在它所属的范围内有效。(作用域)

变量在哪对大括号内，变量就属于哪对大括号

* 一行上可以定义多个变量，但是不建议

**public** **class** Demo08Variable {

**public** **static** **void** main(String[] args) {

//定义变量

**int** a = 10;

System.***out***.println(a);

//变量未赋值,不能使用

**int** b;

b = 20; //变量在使用前赋值都是可以的

System.***out***.println(b);

//变量只在它所属的范围内有效

{

**int** c = 100;

System.***out***.println(c);

}

//System.out.println(c);

//一行上可以同时定义多个变量,但是不建议

/\*

int aa,bb,cc;

aa = 10;

bb = 20;

cc = 30;

\*/

**int** aa=10,bb=20,cc=30;

}

}