第01天 java基础知识

今日内容介绍

* Java概述、helloworld案例
* 工具安装 、配置环境变量、注释、关键字
* 常量、变量、数据类型、标识符

# Java概述

## Java语言发展史和平台概述

 A:Java语言发展史

詹姆斯·高斯林（James Gosling）1977年获得了加拿大卡尔加里大学计算机科学学士学位，1983年获得了美国卡内基梅隆大学计算机科学博士学位，毕业后到IBM工作，设计IBM第一代工作站NeWS系统，但不受重视。后来转至Sun公司，1990年，与Patrick，Naughton和Mike Sheridan等人合作“绿色计划”，后来发展一套语言叫做“Oak”，后改名为Java。

SUN(Stanford University Network，斯坦福大学网络公司)

B:Java语言版本

\* JDK 1.1.4 Sparkler 宝石 1997-09-12

\* JDK 1.1.5 Pumpkin 南瓜 1997-12-13

\* JDK 1.1.6 Abigail 阿比盖尔--女子名 1998-04-24

\* JDK 1.1.7 Brutus 布鲁图--古罗马政治家和将军 1998-09-28

\* JDK 1.1.8 Chelsea 切尔西--城市名 1999-04-08

\* J2SE 1.2 Playground 运动场 1998-12-04

\* J2SE 1.2.1 none 无 1999-03-30

\* J2SE 1.2.2 Cricket 蟋蟀 1999-07-08

\* J2SE 1.3 Kestrel 美洲红隼(sǔn) 2000-05-08

\* J2SE 1.3.1 Ladybird 瓢虫 2001-05-17

\* J2SE 1.4.0 Merlin 灰背隼 2002-02-13

\* J2SE 1.4.1 grasshopper 蚱蜢 2002-09-16

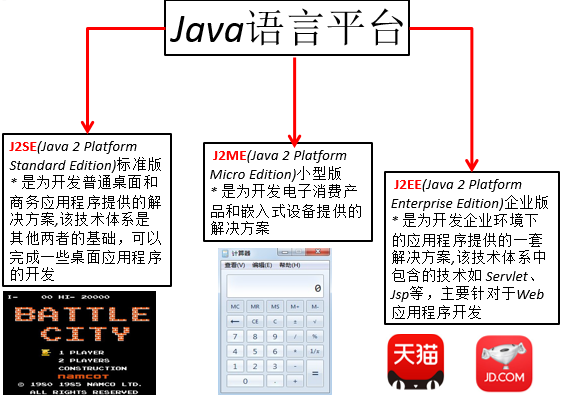
\* J2SE 1.4.2 Mantis 螳螂 2003-06-26

\* JAVASE 5.0 (1.5.0) Tiger 老虎

\* JAVASE 5.1 (1.5.1) Dragonfly 蜻蜓

\* JAVASE 6.0 (1.6.0) Mustang 野马

\* JAVASE 7.0 (1.7.0) Dolphin 海豚



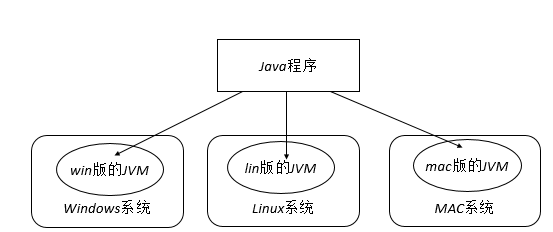
## JVM,JRE,JDK的概述

### 什么是跨平台?（依耐虚拟机实现）

平台：指的是操作系统(Windows，Linux，Mac)

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

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



### JVM JRE JDK说明

A:什么是JVM

JVM是java虚拟机(JVM Java Virtual Machine)，java程序需要运行在虚拟机上，不同平台有自己的虚拟机，因此java语言可以跨平台

B:什么是JRE

包括Java虚拟机(JVM Java Virtual Machine)和Java程序所需的核心类库等如果想要运行一个开发好的Java程序，计算机中只需要安装JRE即可。

JRE:JVM+类库。

C:什么是JDK

JDK是提供给Java开发人员使用的，其中包含了java的开发工具，也包括了JRE。所以安装了JDK，就不用在单独安装JRE了。

其中的开发工具：编译工具(javac.exe)  打包工具(jar.exe)等

  JDK:JRE+JAVA的开发工具。

D:为什么JDK中包含一个JRE

为什么JDK中包含一个JRE呢？

开发完的程序，需要运行一下看看效果。

E:JDK,JRE,JVM的作用和关系

JDK包含JRE 和开发工具包

JRE 包含 核心类库和JVM

## 常用dos命令

### 打开控制台

* + win + R，然后cmd回车

### 常用命令

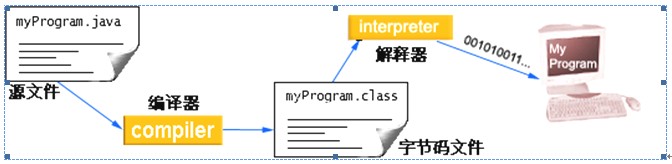
* + d: 回车 盘符切换
  + dir(directory):列出当前目录下的文件以及文件夹
  + cd (change directory)改变指定目录(进入指定目录)
    - 进入 cd 目录；cd 多级目录
    - 回退 cd.. ；cd\
  + cls : (clear screen)清屏
  + exit : 退出dos命令行

## 下载安装JDK

请参考《JDK下载安装文档.doc》安装步骤

## helloworld案例

### 执行流程



### 编写代码步骤

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”);

### 案例代码一

public class HelloWorld {

public static void main(String[] args) {

System.out.println("HelloWorld");

}

}

运行代码步骤：

* 在命令行模式中，输入javac命令对源代码进行编译，生成字节码文件
  + javac 源文件名.java
* 编译完成后，如果没有报错信息，输入java命令对class字节码文件进行解释运行,执行时不需要添加.class扩展名
  + java HelloWorld

### HelloWorld案例常见问题

A:找不到文件(都演示一下，让学生看看出现的都是什么问题)

a:文件扩展名隐藏导致编译失败

b:文件名写错了

B:单词拼写问题(都演示一下，让学生看看出现的都是什么问题)

a:class写成Class

b:String写成string

c:System写成system

d:main写成mian

C:括号匹配问题(都演示一下，让学生看看出现的都是什么问题)

a:把类体的那对大括号弄掉一个

b:把方法体的那对大括号弄掉一个

c:把输出语句的那对小括号弄掉一个

D:中英文问题(都演示一下，让学生看看出现的都是什么问题)

a:提示信息：错误: 非法字符: \????的格式

注意：java编程中需要的基本上都是英文字符

# 环境配置

## 工具安装

### Notepad软件的安装和配置

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软件的安装和配置

设置 – 首选项 – 新建 – 默认语言和编码

## 环境变量配置

### 案例说明

为什么要配置

* + 程序的编译和执行需要使用到javac和java命令，所以只能在bin目录下写程序
  + 实际开发中，不可能把程序写到bin目录下，所以我们必须让javac和java命令在任意目录下能够访问

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

A: 什么是注释

* 用于解释说明程序的文字

B: Java中注释分类

单行注释

* + 格式： //注释文字

多行注释

* + 格式： /\* 注释文字 \*/

文档注释

* + 格式：/\*\* 注释文字 \*/

C: 注释的作用

a:解释说明程序

b:帮助我们调试错误

### 案例代码二

/\*

注释：用于解释说明程序的文字

分类：

单行

多行

作用：解释说明程序，提高程序的阅读性

\*/

//这是我的HelloWorld案例

public class HelloWorld {

/\*

这是main方法

main是程序的入口方法

所有代码的执行都是从main方法开始的

\*/

public static void main(String[] args) {

//这是输出语句

System.out.println("HelloWorld");

}

}

## 关键字

### 关键字概述

* + 被Java语言赋予特定含义的单词

### 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。

### 案例代码三:

/\*

关键字：被Java语言赋予特定含义的单词

特点：

A:组成关键字的字母全部小写

B:常见的代码编辑器,针对关键字有特殊的颜色标记

\*/

public class HelloWorld {

public static void main(String[] args) {

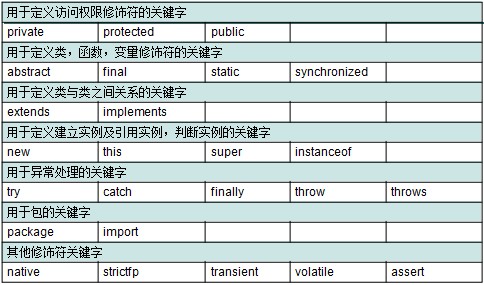
System.out.println("HelloWorld");

}

}

关键字举例:





# 语法格式

## 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)

### 案例代码四:

/\*

常量：在程序执行的过程中，其值不可以发生改变的量

常量分类：

A:字符串常量 "HelloWorld"

B:整数常量 12,-23

C:小数常量 12.34

D:字符常量 'a','0'

E:布尔常量 true,false

F:空常量 null(后面讲解)

\*/

public class ChangLiang {

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

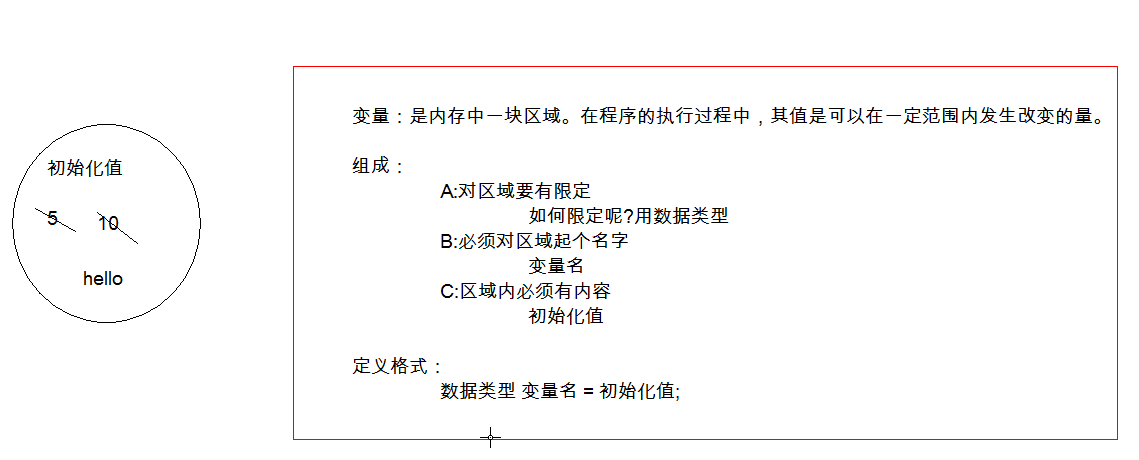
### 变量概述

* + 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，变量其实是内存中的一小块区域

### 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

### 变量图解



## 数据类型

### 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：

* + 1B（字节） = 8bit
  + 1KB = 1024B
  + 1MB = 1024KB
  + 1GB = 1024MB
  + 1TB = 1024GB

### 数据类型概述和分类

A:为什么有数据类型

Java语言是强类型语言，对于每一种数据都定义了明确的具体数据类型，在内存中分配了不同大小的内存空间

B:Java中数据类型的分类

基本数据类型

引用数据类型 （类、接口、数组）

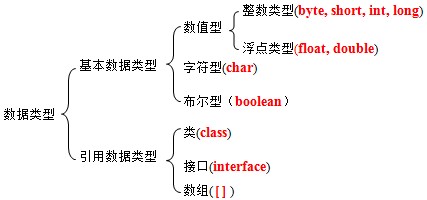
面向对象部分讲解

  注意：

A：整数默认是int类型，浮点数默认是double类型

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；



## 标识符

### 标识符概述

A 作用

* + 给包,类,方法,变量等起名字

B 组成规则

* + 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。不建议使用中文字符；
  + 注意事项
  + 不能以数字开头
  + 不能是Java中的关键字

C : 命名原则:见名知意

a包（其实就是文件夹，用于对类进行管理）

最好是域名倒过来,要求所有的字母小写

b类或者接口

如果是一个单词首字母大写

如果是多个单词每个单词首字母大写(驼峰标识)

c方法或者变量

如果是一个单词全部小写

如果是多个单词,从第二个单词首字母大写

d常量

如果是一个单词,所有字母大写

如果是多个单词,所有的单词大写,用下划线区分每个单词

### 案例代码五

/\*

标识符：就是给包,类,方法,变量起名字的符号。

组成规则：

A:unicode字符

数字字符,英文大小写,汉字(不建议使用汉字)

B:下划线\_

C:美元符$

注意事项

A:不能以数字开头

B:不能是java中的关键字

常见命名规则：

A:基本要求

见名知意

B:常见的命名

a:包(其实就是文件夹,用于对类进行管理)

全部小写,多级包用.隔开

举例：com，com.itheima

b:类

一个单词首字母大写

举例：Student,Car

多个单词每个单词的首字母大写

举例：HelloWorld

c:方法和变量

一个单词首字母小写

举例：age,show()

多个单词从第二个单词开始每个单词的首字母大写

举例：maxAge,getAge()

\*/

public class BiaoZhiFu {

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;

}

}

## 定义变量

### 基本数据类型变量的定义和使用

变量的定义格式：

数据类型 变量名 = 初始化值;

基本数据类型：

byte,short,int,long,float,double,char,boolean

注意：

整数默认是int类型，定义long类型的数据时，要在数据后面加L。

浮点数默认是double类型，定义float类型的数据时，要在数据后面加F。

### 案例代码六

public class VariableDemo {

public static void main(String[] args) {

//定义byte类型的变量

byte b = 10;

System.out.println(10);

System.out.println(b);

//定义short类型的变量

short s = 100;

System.out.println(s);

//定义int类型的变量

int i = 10000;

System.out.println(i);

//定义long类型的变量

long l = 1000000000000000L;

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。
  + 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

### 案例代码七

/\*

变量定义注意事项：

1:变量未赋值,不能直接使用

2:变量只在它所属的范围内有效

变量属于它所在的那对大括号

3:一行上可以定义多个变量,但是不建议

\*/

public class VariableDemo2 {

public static void main(String[] args) {

//定义变量

int a = 10;

System.out.println(a);

int b;

b = 20; //变量在使用前赋值都是可以的

System.out.println(b);

{

int c = 100;

System.out.println(c);

}

//System.out.println(c);

/\*

int aa,bb,cc;

aa = 10;

bb = 20;

cc = 30;

\*/

/\*

int aa = 10;

int bb = 20;

int cc = 30;

\*/

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

}

}

## 数据类型转换

### 隐式数据类型转换

取值范围小的数据类型与取值范围大的数据类型进行运算,会先将小的数据类型提升为大的,再运算

### 案例代码八

/\*

+:是一个运算符，做加法运算的。

我们在做运算的时候，一般要求参与运算的数据类型必须一致。

类型转换：

隐式转换

强制转换

隐式转换

byte,short,char -- int -- long -- float -- double

\*/

public class TypeCastDemo {

public static void main(String[] args) {

//直接输出了运算的结果

System.out.println(3 + 4);

//定义两个int类型的变量

int a = 3;

int b = 4;

int c = a + b;

System.out.println(c);

//定义一个byte类型,定义一个int类型

byte bb = 2;

int cc = 5;

System.out.println(bb + cc);

//我能不能不直接输出，用一个变量接受呢?

//用变量接受，这个变量应该有类型

//可能损失精度

//byte dd = bb + cc;

int dd = bb + cc;

System.out.println(dd);

}

}

### 强制类型数据转换

强制转换的格式

\* b = (byte)(a + b);

强制转换的注意事项

\* 如果超出了被赋值的数据类型的取值范围得到的结果会与你期望的结果不同

### 案例代码九

/\*

强制转换：

目标类型 变量名 = (目标类型) (被转换的数据);

不建议强制转换，因为会有精度的损失。

\*/

public class TypeCastDemo2 {

public static void main(String[] args) {

int a = 3;

byte b = 4;

int c = a + b;

//byte d = a + b;

byte d = (byte) (a + b);

}

}