# 第一章 基础常识

硬件 : 可以看得见, 摸的着的电子器件.

CPU 执行指令 add(80)

外存 程序(指令和数据)

内存(RAM) 进程

软件 : 完成一定的功能的指令和数据的集合.

人机交互 : 人通过使用OS来使用计算机

图形化(GUI) 简单, 方便, 开发难

命令行(CLI) 复杂, 功能强大, 开发容易

启动命令行 : win + R -> cmd 回车

目录 : Directory 就是文件夹(folder)

目录可以包含子目录和子文件, 子目录也可以包含子目录和子文件.

当前工作目录 : 前导符左侧的目录

路径(path) C:\Users\pauliuyou, 是一个字符串, \表示父子关系.

路径 : 一条路, 最终指向一个文件或一个目录.

常见命令 :

dir – directory

作用 : 列出当前目录中的所有子目录和子文件的信息

字节 : 数据存储或传输的基本单位, byte, 一个字节=8比特(bit)

切换盘符

目标盘符: 回车

创建子目录, 在当前目录下创建的.

md (make directory)

命令行参数 : 是一些字符串, 跟在命令后面, 用空格隔开, 作用是告诉命令进一步的详细信息

切换工作目录

cd (change directory) 目标目录(路径)

练习 : 创建目录 D:\MyWork\JavaSE\day01\code

特殊目录 :

. 表示当前目录

.. 表示当前目录的上级目录

./mywork/javase/day01/code 以当前目录为开始的目录的路径, 称为相对路径

D:/mywork/javase/day01/code以根目录为开始的目录的路径 称为绝对路径

命令 :

1. 内部命令 : OS内置的命令, 可以在任意位置执行
2. 外部命令 :保存在硬盘上的一个文件(.exe, .com, .bat)

PATH是一个路径列表, 当OS执行外部命令时, 根据PATH中的目录进行搜索, 如果搜索成功, 则执行外部命令, 搜索失败, 则报错

start 启动一个新命令行窗口, 以当前目录为工作目录

start . 以当前目录为目标打开资源管理器

java主要特性8

1) 简单 : 相比C/C++, 丢弃复杂的语法

2) 面向对象 : 面向过程, 面向对象关注的是(具有功能的)对象

3) 分布式 : 基于网络的多主机协助

4) 健壮 : 强类型, 异常处理, GC, 标记垃圾为可用, 引用 : 安全的指针.

5) 安全 : 所有类的加载执行都必须经过类加载器(ClassLoader)

6) 跨平台 : JVM跨平台, 我们的程序只需要JVM

7) 性能好 : 编译型, 直接把源文件变成可执行的文件.

8) 多线程 : 最大化利用CPU, 提高吞吐量.

JRE = JVM + 核心类库

JDK = JRE + 开发工具

配置PATH环境变量

开发并运行java程序步骤:

1) 在相应的目录中创建源文件, Hello.java, 并编辑这个文件

2) 在命令行中, 以源文件的目录为工作目录执行命令javac Hello.java,进行编译, 产生Hello.class文件

3) 仍然还是在当前工作目录下, 执行外部命令 java Hello

一个源文件可以包含多个类,但是只能有一个公共类, 因为公共类的类名必须和文件名一致, 一个类中包含多个方法,

一个方法中包含多条语句, 方法不可以嵌套.

// 主类 : 包含主方法的类, 只有主类能执行

// 公共类 : 被public修饰的类

NotePad++使用, 快捷键

全选 ctrl + a

复制 ctrl + c

粘贴 ctrl + v

剪切 ctrl + x

撤消 ctrl + z

重做 ctrl + y

保存 ctrl + s

shift + tab 选中的内容整体左移

tab 加入一个缩进

整体右移

快速复制当前行 ctrl + d

快速删除当前行 ctrl + l

# 第二章 基本语法

命名规则 :

1. 组成 : 52个字母, 数字字符0~9, \_和$(禁止使用)
2. 数字不能开头
3. 不可以直接使用关键字, 可以包含关键字
4. 区分大小写, 长度65535个字符
5. 不能包含空格

命名规范

1. 包名 : 全部字母小写
2. 类名 : 首字母要大写, 后面的单词首字母大写
3. 变量名, 方法名 : 首字母要小写, 后面的单词首字母大写.
4. 常量名, 全部字母要大写,使用\_隔开(唯一允许使用\_的地方)

变量 : 变化的量.

内存中的一块被命名的被特定的数据类型约束的区域. 此区域中可以保存数据, 并且数据可以变化.

变量的声明 :

数据类型 变量名;

## 8种基本数据类型 :

整型 :

byte 1byte -128~127

short 2byte -32768~32767

int 4byte -20多亿~20多亿

long 8byte -巨大~巨大

浮点型

float 4byte -10^38~10^38

double 8byte -10^308~10^308

字符型

char 2byte

布尔型

boolean 1byte true, false

## 每日一考\_day02

1. 写出java语言的8个特性

简单：相较于C，C++舍弃了很多复杂的语法；

面向对象：面向的是具有各种功能的对象；

分布式：基于网络的各主机协助；

健壮：强类型,所有数据都必须要有类型，异常处理GC标记可用状态, 指针的安全化；

安全；类的加载必须经过类加载器(ClassLoader)；

跨平台：JVM可以跨平台；

性能好：编译型语言，可以把执行的文件编译成字节码文件；

多线程：最大化使用CPU，提高吞吐量

2. 描述一下语句,类,和方法之间的关系,

类是java程序的基本单位, 方法是java程序的某种独立功能的单位, 语句是java程序的最小的执行单位.

一个类可以包含多个方法，但只能有一个主方法，主方法是执行的入口

一个方法可以包含多个语句

3. 什么是主类, 什么是公共类, 公共类有什么注意点?

主类是包含主方法的类；公共类是被public修饰的类；

公共类的类名必须与源文件名一致；

4. 变量是什么?

变量是被命名的且被数据类型约束使用范围和大小的一段空间，这个空间中可以保存数据, 并且值可以发生变化.

5. 使用变量有哪些注意事项(至少3点)?

1) 使用变量必须声明其数据类型和变量名；

2) 读取变量前必须进行初始化；

3) 变量必须先声明, 后使用

4) 变量有其作用范围, 它的声明语句所隶属的一对{}内

5) 在同一个范围内, 变量不允许重复声明.

6) 变量内存中的数据只能在数据类型约束的范围内变化.

变量声明 :

数据类型 变量名;

变量按照数据类型来分

1. 基本数据类型 : 内存空间中保存的就是数据本身.
   1. 数值型

整数

byte 1字节 -128~127

short 2字节 -32768~32767

int 4字节 -20多亿~20多亿

long 8字节 -巨大~巨大

浮点数

float 4字节 -10^38~10^38

double 8字节 -10^308~10^308

* 1. 字符型
     1. char 2字节
  2. 布尔型

boolean 1字节 true, false

2) 引用数据类型 : 内存空间中保存的是其他数据的地址.

// 整数字面量默认使用4个字节的int型来存储

long l2 = 4240234234L; // L的作用是告诉编译器请使用8字节的long来保存这个字面量.

// .324是字面量, 浮点数的字面量默认是double型

float f2 = 8.22F; // F的作用是告诉编译器8.22这个字面量请使用4个字节的float保存.

// 任意非long整数作运算, 结果总是int型

// 混合数据类型运算时, 结果总是类型范围最大的那种类型

进制

287354 十进制数 : 权值计算以10为底求n次幂, 逢10进1

2 \* 10 ^ 5 + // 权值 = 10的n次幂

8 \* 10 ^ 4 +

7 \* 10 ^ 3 +

3 \* 10 ^ 2 +

5 \* 10 ^ 1 +

4 \* 10 ^ 0

十六进制, 逢16进一, 计算权值以16为底求n次幂

0x34

3 \* 16 ^ 1 +

4 \* 16 ^ 0 = 52

二进制, 逢2进一, 计算权值以2为底求n次幂

0110 1101

0 \* 2 ^ 7 + -> 128 \* 0

1 \* 2 ^ 6 + -> 64 \* 1

1 \* 2 ^ 5 + -> 32 \* 1

0 \* 2 ^ 4 + -> 16 \* 0

1 \* 2 ^ 3 + -> 8 \* 1

1 \* 2 ^ 2 + -> 4 \* 1

0 \* 2 ^ 1 + -> 2 \* 0

1 \* 2 ^ 0 + -> 1 \* 1 = 109

0x6211

6 \* 16 ^ 3 + -> 4096 \* 6 -> 24576

2 \* 16 ^ 2 + -> 512

1 \* 16 ^ 1 + -> 16 + 1

1 \* 16 ^ 0 = 25105

0010 1110 0101 0111

7 + 16 + 64 + 512 + 1024 + 2048 + 8192 = 11863

0x10 -> 16 -> 1 0000

十进制 二进制 十六进制

0 0000 0

1 0001 1

2 0010 2

3 0011 3

4 0100 4

5 0101 5

6 0110 6

7 0111 7

8 1000 8

9 1001 9

10 1010 A

11 1011 B

12 1100 C

13 1101 D

14 1110 E

15 1111 F

0x39 -> 0011 1001

0x9C52 ->

1001 1100 0101 0010

0101 0111 1010 0101 ->

0x57A5

0xE2C98F3D ->

1110 0010 1100 1001 1000 1111 0011 1101

0111 0101 0011 1110 1011 1111 0101 1011 ->

0x753EBF5B

每个二进制的最高位用于表示符号, 如果符号位为0表示这个数是正数, 如果符号位为1则表示这个数是一个负数.

计算机底层所有数据都是使用二进制的补码形式来保存.

正数的补码就是自身

负数的补码是它的相反数全部取反+1得到.

0011 0111 -> 是一个正数

0x37 -> 3\*16+7 = 55

结论就是0011 0111这个二进制数在计算机中表示的就是55.

1101 1011 -> 这是一个负数的补码?

-1 -> 1101 1010

取反 -> 0010 0101 -> 0x25 -> 37

结论: 1101 1011这个数是-37在计算机中的存储.

0101 1111 这是多少?

0x5F -> 95

1010 0011 这是多少? 这是一个负数.是负多少

-1 -> 1010 0010

取反 -> 0101 1101 -> 0x5D -> 93

1010 0011就是-93

1111 1111 这是多少? 这是一个负数, 负多少?

-1 -> 1111 1110

取反 -> 0000 0001 -> 1

所以1111 1111 就是-1

byte型整数的最小值

1000 0000 这是一个负数, 负多少?

-1 -> 0111 1111

取反-> 1000 0000 -> 0x80 -> 128

所以1000 0000是-128

byte型整数的最大值

0111 1111 正数最大

0x7F -> 16 \* 7 + 15 = 127

short型的最大值

0111 1111 1111 1111 ->

0x7FFF -> 32767

short型的最小值

1000 0000 0000 0000 ->

0x8000 -> -32768

int型最大值

0111 1111 1111 1111 1111 1111 1111 1111 ->

0x7FFFFFFF

int型最小值

1000 0000 0000 0000 0000 0000 0000 0000

0x80000000

long型最小值最大值

0x8000000000000000

0x7FFFFFFFFFFFFFFF

char型最小值, 最大值(没有负数)

0000 0000 0000 0000

1111 1111 1111 1111

int n = 09;