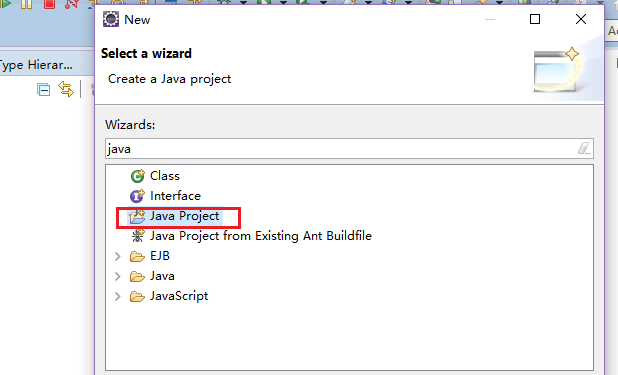
eclipse 开发java的一个工具

日食 SUN

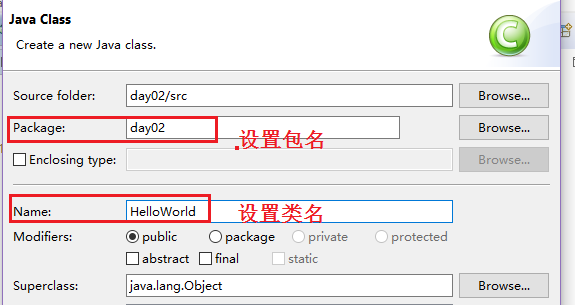
项目为单位组织代码

1. 创建项目

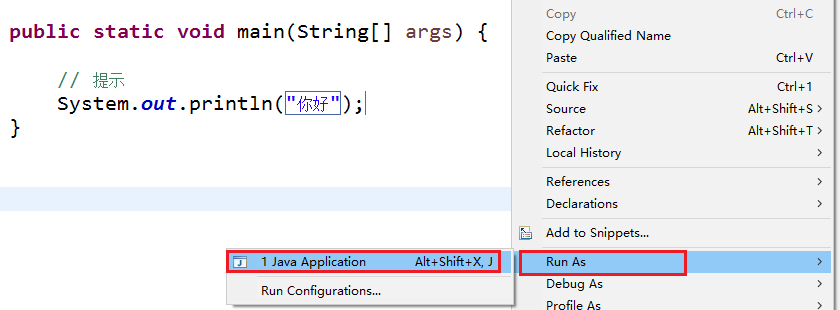


java代码放置的位置要位于项目的 src（source源代码）目录

1. 创建java类



1. 运行java代码



# 变量和数据类型（重点）

## 变量

用来存储数据

2 存入变量 int a = 2; 赋值，存储

3 存入变量

5 存入变量

## 8种基本数据类型

整数类型

byte short int long

浮点数类型 （小数）

float double

字符类

char

布尔类型

boolean （真，假）

语法：

类型 变量名;

或者

类型 变量名 = 值;

注意：

变量的类型要与值得类型一致

变量的名称, 要求必须 字母 数字 或 \_ 或 $， 数字不能开头

整数类型分了4种

byte 1 个字节 字节 -2^7 ~ 2^7 -1

-128 ~ 127

short 2 个字节 短整型 -2^15 ~ 2^15-1

-32768 ~ 32767

int 4 个字节 整型 -2^31 ~ 2^31-1

long 8 个字节 长整型 -2^63 ~ 2^63-1

优盘 8g 16G

1024 字节 是1 k

1024 \* 1024 个字节 是1M

1024 \* 1024 \*1024 是 1G

8 \* 1024 \* 1024 \* \*1024 \* 8 个 bit = 8G

byte

二进制 十进制

0000 0000 0

0000 0001 1

0000 0010 2

0000 0011 3

...

java 中最高位代表符号位 ， 符号位0代表正数，符号位1代表负数

负数的存储方式：

符号位的1理解为 -128

1111 1111 -1 -128 + 127 = -1

0000 0001

1111 1111

10000 0000

### 整数运算:

+

-

\*

/ 除法 :只保留商，会忽略掉所有小数(没有四舍五入)

不能够除0

% 求余数:

例如： 5%3得 2

### 浮点类型：

float 单精度浮点数 4个字节

double 双精度浮点数 8个字节

最高位表示符号位， 0 正数 1 负数

float 8 指数 23 小数

大于此2^24数字的，不能够精确保存

换算为10进制 16 777 216

double 11 指数 52 小数

大于此2^53数字的，不能够精确保存

十进制 二进制 科学计数法

9.5 => 1001.1 1.0011 \* 2^3

### 浮点数运算：

1. 精度问题

float小数点后只能精确保存6 ~ 7 位10进制数

double 小数点后只能精确保存14～ 15位10进制数

1. 赋值时

float的值后必须加f/F结尾， jdk8之后可以不加

double的值后可以加d/D结尾

1. 他们用来表示小数时，对于很多小数是一个近似值

例如： 2.0 - 1.1

### 字符类型 char

character 两个字节

它可以表示一个任意字符（字母，数字，汉字)

十六进制

0 1 2 3 4 5 6 7 8 9 a b c d e f 10

十进制

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

掌握三种赋值方法：

char a = 'B';

char a = 66;

char a = '\u0042'

asc ii 码表 ： 英文字符与数字的对应关系

unicode 码表： 用两个字节表示，全球所有字符与数字的对应关系（汉字，阿拉伯..）

#### 转义字符

char a = 'B';

char b = '''; 错误

\特殊字符 char b = '\'' 正确

/ 斜杠

\ 反斜杠

\\ 代表真正的 \

\' 代表真正的 '

\" 代表真正的 "

\n 换行

\r window 特有， 回车

\t 制表符 （特殊空格 , 相当于Tab键）

### 布尔类型

（没有长度）

true(真) false(假)

boolean b = true;

boolean c = false;

逻辑运算

与 And： 两个条件都为true，最后结果才为真 &&

a b 结果

true true true

true false false

false true false

false false false

或 Or: 两个条件只要有一个为true，结果就为true ||

a b 结果

true true true

true false true

false true true

false false false

非Not: 取反 ， ！

a 结果

true false

false true

### String 类型

char 存储一个字符

String 由多个char组成, 可以存储多个字符

String a = "多个字符";

与基本类型不同，它可以有一些方法、属性

## 运算:

### 算数运算：

+ - \* / %

### 自增，自减运算符 (难点 重点)

++(自增) 在a原有值基础上加1

--(自减) 在a原有值基础上减1

int a = 0;

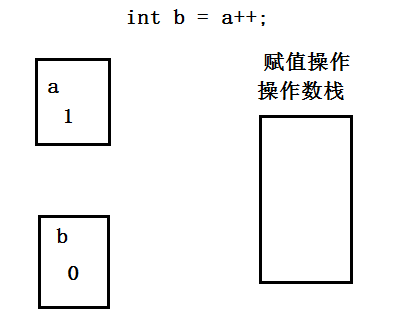
a++; a 1

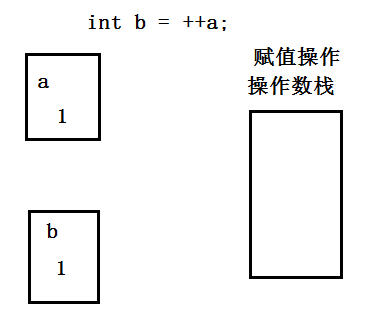
a--; a 0

a--; a -1

int a = 0;

++a; a 1

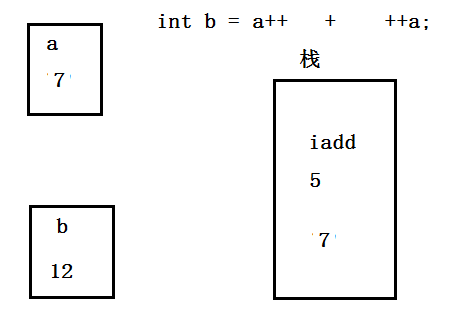




int a = 5;

int b = (a++) + (++a);

System.out.println(b); // 12



结论：

a++ 先做运算，再自增

++a 先自增，再运算

### 类型转换问题

byte short int long float double

char

规则1： 小的类型可以赋值给大的类型（安全）

byte a = 2;

int b = a;

规则2：大的类型不能直接赋值给小的类型，强制转换(不安全)

int a = 2;

byte b = a ; // 错误

byte b = (byte)a; // 强制转换，可能会引起数据的丢失

丢失的例子：

int a = 300;

byte b = (byte)a; // 44

正数变为负数的例子：

int a = 130;

byte b = (byte)a; // -126

### 类型自动提升：

int a = 10;

double b = 10.0;

double c = a + b;

// a被提升为double类型

规则1： 遇到了这三个类型之一: long, float, double, 找到其中最大的类型进行转换，再运算

规则2： 如果没有long, float, double, 你的变量是 byte, short , char, int 参与算数运算，都会转换为int再运算

规则3：字符串 可以有 +

"aaa" + "bbb" = "aaabbb"

字符串可以与任意其他类型做+法，结果把其他类型与此字符串拼接为新的字符串

### 比较运算符

> 大于

>= 大于等于

< 小于

<= 小于等于

== 判断是否相等 ， float 和 double如果超出精度，不能用==判断

!= 判断是否不等

返回一个布尔值 true false

### 逻辑运算符

短路特性： 对于&& 第一个条件为false && 之后的运算就不会执行了

int a = 6;

boolean b = 10 < 5 && a++ > 4;

请问a的值是多少？

对于|| 第一个条件为 true， || 之后的运算就不会执行了

### 增强的赋值运算符(了解)

int a = 10;

a = a + 2;

a 结果是12

增强赋值运算：

int a = 10;

a += 2; // 先把a+2，再赋值给a，等价于 a = a+2

-= \*= /= %=

a -= 2; // a = a - 2

a \*= 2; // a = a \* 2;

....

### 位运算符 (了解)

& | ^ ~

& 按位与 （

0 & 0 = 0

0 & 1 = 0

1 & 0 = 0

1 & 1 = 1

）

2 & 3

0000 0010 二进制的 2

0000 0011 二进制的 3

0000 0010 结果是2

2 | 4

0000 0010 二进制的 2

0000 0100 二进制的 4

0000 0110 结果是6

~ 2

0000 0010 二进制的 2

1111 1101 结果 -3

异或

0 ^ 0 = 0

0 ^ 1 = 1

1 ^ 0 = 1

1 ^ 1 = 0

2 ^ 3

0000 0010 二进制的 2

0000 0011 二进制的 3

0000 0001 结果是1

byte a = 2;

byte b = 3;

int c = a ^ b;

### 移位运算符(了解)

>> n 算数右移 符号位不动，右移n位，空出来的位，正数补0负数补1

右移 1位 相当于/2

右移 2位 相当于/4

...

<< n 算数左移 符号位不动，左移n位，空出来的位，补0

左移1位 相当于\*2

左移2位 相当于\*4

...

>>> 逻辑右移 符号位会一起向右移动

例如: -4 >>> 1 =

## 变量的作用范围

从变量的定义开始，直到它所在的{} 的结束}位置，是这个变量的作用范围

同一个块作用域内，变量名不能重复

同名变量的作用范围不能有交叉

作业

1. 有如下代码：

int a = 5;

int b = (a++) + (--a) + (++a);

分析执行过程，说出b的结果是多少？

2.一个三位数，求其个位+十位+百位的和