### Chapter01-Java基础语法

#### 1、注释

1.1、注释种类

- 单行注释：//...

- 多行注释：/\* ... \*/

- 文档注释（javadoc）：/\*\* ...

@ parameter 1 ...

@ parameter 2 ...

\*/

#### 2、Java代码规范（新手注意）

- 类、方法的注释用javadoc的方式来写

- 非javadoc的注释一般给代码的维护者来看

- tab右移，shift+tab左移（选中后按键可以整段移动）

- 运算符两边有空格。例如：要写1 + 2 = 3，不要1+2=3

- 源文件都用utf-8编码

- 行宽不超过80字符（便于一张屏幕上阅读）

- 次行风格和行尾风格（前括号写在行尾还是次行）

更推荐行尾风格

\*！行尾风格：public Arraylist(int abc){

if(abc>0){

...

} else{

...

}

}

次行风格：public Arraylist(int abc)

{

if(abc>0)

{

...

} else

{

...

}

}

#### 3、JAVA 特性：

1) 面向对象

2) 跨平台性（编译的.class可被windows、linux等各种系统执行

3) 解释性（和编译性相对，编译性编译后直接是0-1，解释性解释后仍不是二进制文件？）

#### 4、JDK、JRE是什么：

1、JDK=Java Development Kit，包含了JRE+JAVA开发工具（javac, javap,...）

2、JRE=Java Runtime Environment，包含了JVM+核心库（类）。作为java软件使用者，只需要安装JRE

3、JVM=Java Virtual Machine，是一次编译后，.class能在多个平台运行的关键

JAVA应用程序执行入口：

固定格式：public static void main(String[] args){...}，不能修改

public类：

1、一个源文件中只能有一个public类，其他类个数不限。每一个class都会生成一个.class文件

2、若一个源文件中有一个public类，则源文件名字必须和public类名相同

3、也可以在非public类中存放main方法，然后指定运行非public类。例如：

public class Hello{

public static void main(String[] args){

System.out.println("Hello, world!");

}

}

class Dog {

public static void main(String[] args){

System.out.println("Hello, little puppy!");

}

}

class Cat {

public static void main(String[] args){

System.out.println("Hello, little kitty!");

}

}

运行java Dog，java Cat，java Hello将会得到不同的反应

#### 5、DOS指令

5.1 cd切换路径

无法直接用cd命令换到其他盘符（D, E, etc.）,若要切换盘符，需要用“**cd /d [路径]**”命令

例子：C:\Users\Kexin>cd /d E:\JavaCode

返回上一层：**cd ..**

### Chapter02-变量

#### 1、字符赋给int变量

可以直接把字符赋给int型变量，例如：int c = 'a'。因为字符的存储就是int整数。注意字符 ≠ 字符串。

#### 2、java基础变量类型

·整数型：byte[1], short[2], int[4], long[8]

·浮点数型：float[4], double[8]

·字符型：char[2]

·逻辑型：boolean[1] （java中不能直接用int 0、1表示bool逻辑）

#### 3、自动转换

不同种类变量相加时，会把所有变量统一转换为精度更高的变量

两条转换路径：

char -> int -> long -> float -> double

byte -> short -> int -> long -> float -> double

但相乘时不会，例如计算华氏转摄氏时的如下代码会输出0:

double huaShi = 236.4

double celsius = 5 / 9 \* (husShi - 100)

3.1 byte、short、char运算

byte、short、char类型可以互相或在类型内进行运算，但是运算结果会被转换成int类型

### Chapter03-运算符

#### 1、i++、++i区别

一个经典问题看post-increment: i++ & pre-increment: ++i 后台处理方法：

i = 1;

i = i++;

此时print(i)，显示多少？

i = 1;

i = ++i;

此时print(i)，显示多少？

解答：对于i++，java的后台步骤是：

(1) temp = i; (2) i = i + 1; (3)i = temp;

所以一个情况，会显示1。

对于++i，java的后台步骤是：

(1) i = i + 1; (2) temp = i; (3) i = temp;

所以第二个情况，会显示2。

#### 2. 自动类型转换

复合赋值运算符会自动进行类型转换。例如直接写byte a = 1; a = a + 1;程序会报错，因为

a + 1是int（强制转换），无法被赋给byte类型。但如果写成byte a = 1; a += 1;则程序正常运行，

背后底层等同于a = (byte)(a + 1)

#### 3. 标识符规范：

- 包名：由多个单词组成时，所有字母小写。e.g. xxx.wkx.com

- 类名、接口名：由多个单词组成时，所有字母首字母大写。e.g. AssignOperator

- 变量名、方法名：由多个单词组成时，从第二个单词起，所有字母首字母大写。第一个单词小写。e.g. tankShotGame

- 常量名：所有单词都大写，用\_分隔单词。e.g. TAX\_RATE

#### 4. 原码、反码、补码：

- 对于有符号的数而言，二进制最高位是符号位。0为正数，1为负数

- 正数的原码、反码、补码都一样

- 反码：符号位不变，其他位取反

- 补码 = 反码 + 1

- 计算机运算时，以补码方式运算

- 运算结果也是补码，看运算结果时要转换成原码来看

#### 5. 位运算符：

介绍：>>：算数右移，>>>：逻辑右移（无符号右移），<<：算数左移

规则：1. >>：低位溢出，符号位不变，用符号位补溢出的高位

2. <<： 符号位不变，低位补0

3. >>>：低位溢出，高位补0

特别说明：没有<<<运算符

#### 6. / 和 %：

/使用的一些例子：

10/4=2 //10和4都是int

10.0/4=2.5

%使用的一些例子：

10%3=1

-10%3=-1

10%-3=1 //重要！Java中，a%b=a-a/b\*b

-10%-3=-1 //-10-(-10)/(-3)\*(-3)=-10-3\*(-3)=-10+9=-1

-10.5%3 = -1.5//重要！我原本以为是0，但-10.5-(-10.5)/3\*3=-10.5-(-3)\*3=-1.5

并且再次提醒：小数参与的运算，得到的都是近似值，不会是完完全全-1.5

#### 7. 运算符优先级

单目运算符(++, --) > 算术运算符(+, -, %) > 位移运算符(>>, <<, >>>) > 比较运算符(>, <, >=, ==)

> 逻辑运算符(&, |, &&) > 三元运算符(a?b:c) > 赋值运算符(=, +=, \*=, >>=)

#### 8. 短路与和逻辑与

&& 短路与和& 逻辑与。

共通点：都是当两个条件同时为真时才为真，任意一个为假就为假；

区别：短路与在发现第一个条件为假时就不会再判断第二个条件，会直接输出结果为假；而逻辑与会全部判断完。

由于效率，一般都使用短路逻辑运算符。

当存在如下语句时将能很清楚地看出短路与的特性：

int a, b = 14, 20;

a == 15 && ++b == 21 //执行后，返回false。且a=14, b=20

a == 15 & ++b == 21 //执行后，返回false。且a=14, b=21

### Chapter04因为编号丢失，内容未遗漏

### Chapter05-顺序控制（if&loop）

#### 1、本章回顾：

- if, if-else单分支、多分支条件控制

- switch。推荐使用switch而非if的场合：case较少&是固定数，变量类型为char、int、enum等，代码看着比较简洁

使用if更好的场合：case是一个范围。变量类型是bool等

- for, while, dowhile的loop语句。有break, continue, return的三种流程控制语句。

多重嵌套循环：不建议超过3层，超过的话可读性会非常差。

其中，break语句默认break当前循环，但可以通过添加label（并非固定名，程序员自己取）break外层循环。

return用于结束当前方法，如果方法是main，则结束当前程序。

### Chapter06-数组、查找、排序

#### 1、该章仅介绍冒泡排序

更多排序会在高级章节的算法部分进行讲解

#### 2、数组赋值机制：

- 数组赋值机制是最重要的特性：

数组的赋值方式是引用传递(pass-by-reference)，基础变量是值传递(pass-by-value)。

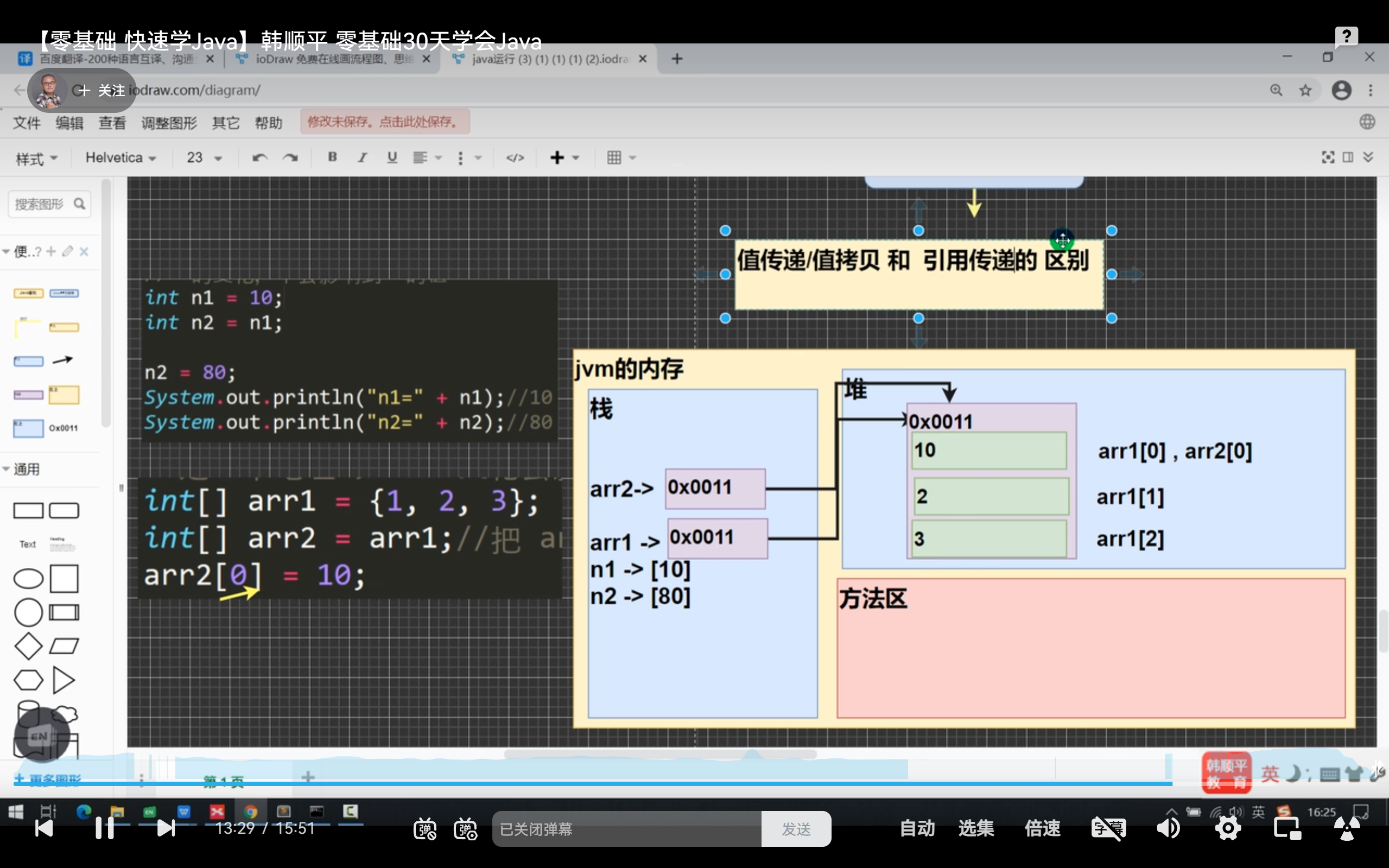
- jvm内存分为三块：

栈(stack)、堆(heap)、方法区(method area)

Primitive variable and its value are both stored in stack.

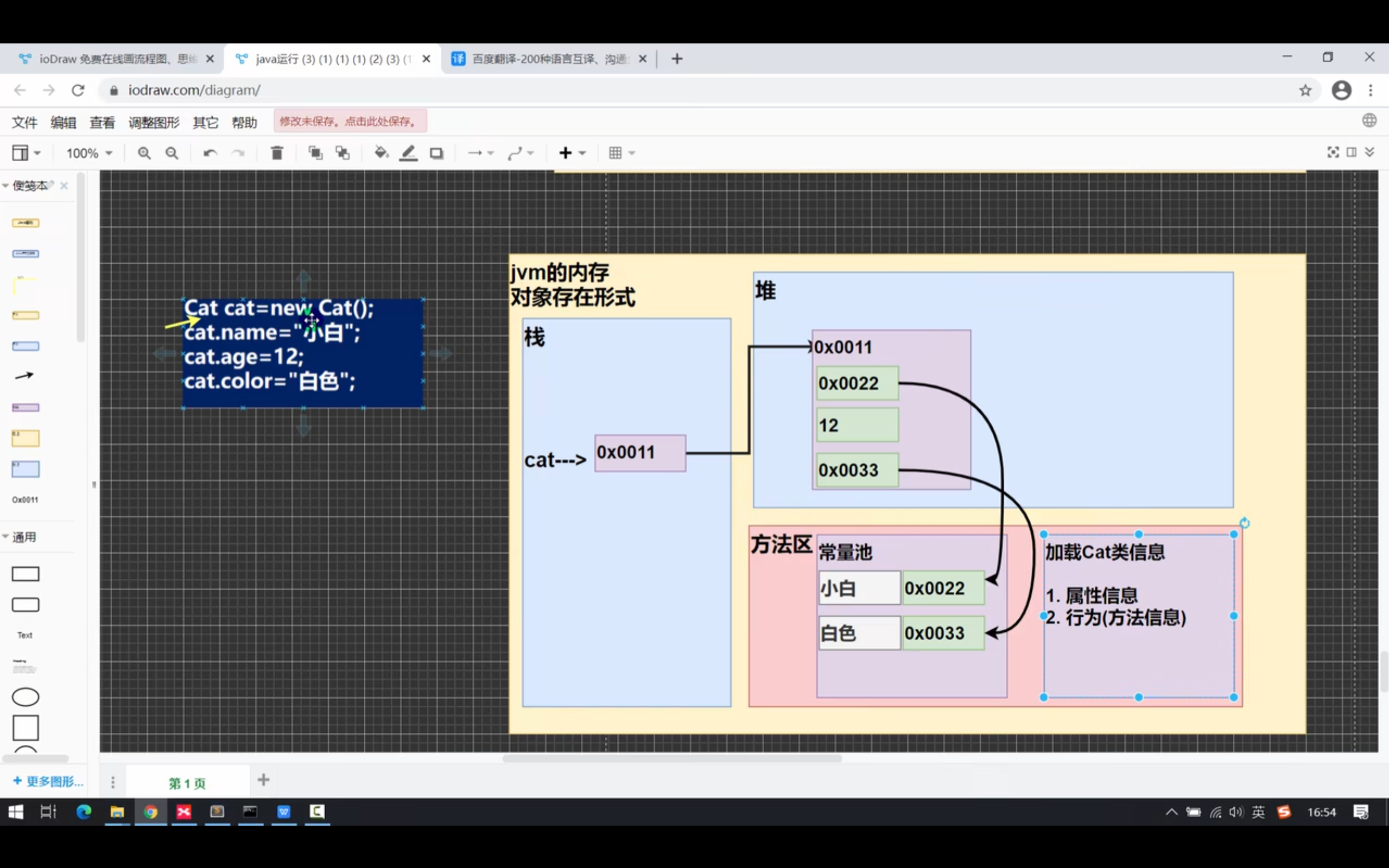
Array variable and the address it points to are both stored in stack.

But the values that array stores are stored in the according address in heap.



### Chapter07-类与对象

#### 1、对象内存布局：



#### 2、方法

一个方法只能返回最多1个值。如果想返回多个值？使用数组。（例子：Method01.java）

把对象a赋给b，是把a指向的地址赋给b，这样a、b会指向同一个堆。

若执行b=null，则是让b不指向原堆，而非清空堆中地址。

在下述代码中，System.out.println(b.age);会抛出异常

