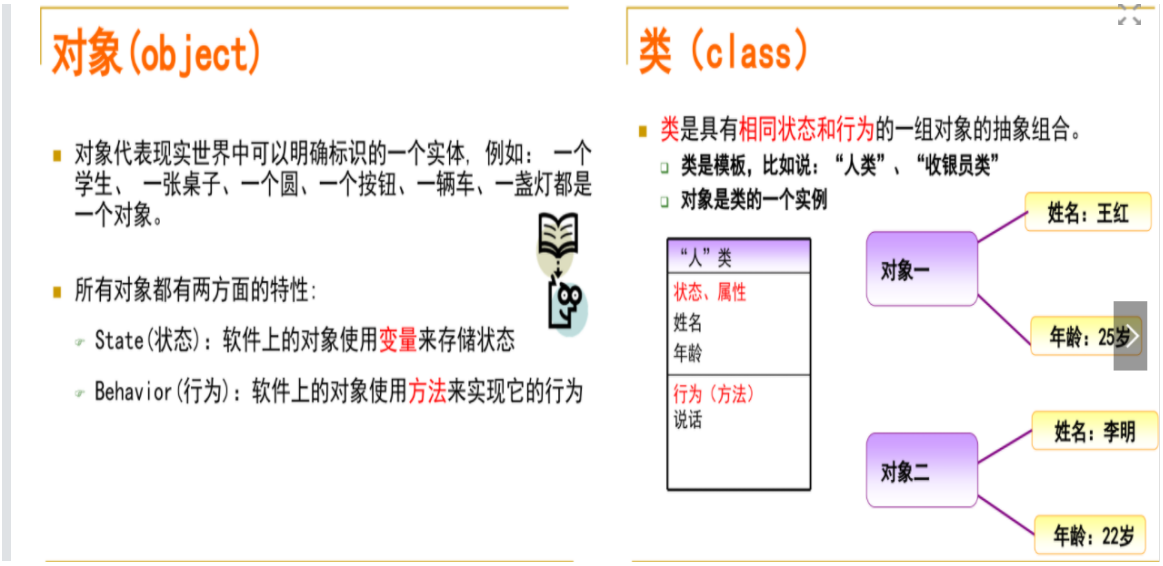
Java create classes and objects

understand classes and objects

A class is a collection of objects with the same state and behavior. Variables store state and methods implement behavior.



Learn how to define classes and create objects of classes

```
[Modifier[^fn1]] + class + class name{//define class
    [modifier[^fn2]] + variable type + objects variable name;//create member state
    [modifier[^fn3]] + method return type + method name(parameter list){//create member behavior

    }
}
```

Particular attention

```java
public class Person {
    public String name;
    public int age;//成员变量，定义在类里，不在任何方法里的变量

    void speak(){
        int age = 60;//方法内部定义的变量称为局部变量
        System.out.println("大家好，我今年" + age + "岁");
    }
}
```

1、Java允许成员变量和局部变量重名
2、在方法中直接访问变量名时，指的是本方法中的局部变量
3、此例中，speak方法中的输出语句，输出的是局部变量age的值
4、想在speak方法中访问成员变量age，必须使用this.age

this引用当前对象的成员变量

```java
public class Person {
    int age;
    String name;
    public Person(String name,int age){
        this.name = name;
        this.age = age;
    }

    void speak(){
        int age = 60;//方法内部定义的变量称为局部变量
        System.out.println("大家好，我叫" + name + "今年" +
        this.age + "岁");
    }
}
```

成员变量和局部变量发生重名时，没有加this的变量是局部变量，加了this的变量是成员变量。此时this是不能省略的。

1、在方法中直接访问本方法中的局部变量
2、speak方法内，没有局部变量和age，name重名，this可以省略，事实上，所有实例成员变量前都隐含this，写不写都存在
3、在speak方法中访问成员变量age，必须使用this.age

learn to access members of object (variable and method)

Understand the role of constructors and use constructors to initialize objects

role

Build and initialize object

how

```
[modifier]+construct method name/*^same with calss*/(parameter list){//no return value
    //Constructor actuator(executable statement)
}
```

Attention

- 构造方法的名称必须和类名相同
- 构造方法没有返回类型，也不用将返回值类型设为void
- 没有参数的构造方法称为无参构造方法
- 一个类中可以有多个构造方法
- 构造方法在创建对象时，使用new关键字调用

none construction method = have none parameter constructor.

have parameter constructor must have none parameter constructor.

创建构造方法

```
public class Person {
   public String name;
   public int age;

   //创建带参的构造方法
   Person(String userName, int userAge){
      name=userName;
      age=userAge;
   }
   void speak( ){ //定义成员方法
     System.out.println("大家好，我叫" + name + "今年" + age + "岁");    }
}
```

调用构造方法创建对象

- Java使用new关键字调用构造方法来创建对象和初始化新对象。
  语法：

```
new 构造方法名称（[实际参数]）；
Person p1 = new Person( );
Person p2 = new Person("Mike",20 );
```

- 对象创建过程如下：
  (1) new关键字创建对象，为对象分配空间，为成员变量赋初值（0值）
  (2) 调用构造方法为新对象初始化
  (3) 构造方法会返回新对象的引用，该引用可以赋值给“同类型”的变量

调用构造方法创建对象

- 创建测试类TestPerson ，要求如下：
  - 使用自己的信息创建一个Person对象，输出对象的属性，并调用对象的speak方法

```
public class TestPerson {
    public static void main(String[] args) {
        Person p1=new Person("Mike",20);
        System.out.println( p1.name +","+ p1.age);
        p1.speak();
    }
}
```

构造方法的作用

```
//定义一个Cat类
class Cat{
  String name = "小白";
  String color = "white";
  char sex = 'f';
  String type = "￥￥￥";
  //没有定义构造方法
 //此时系统提供默认的构造方法,而且没有参数
}
 Cat c1 = new Cat();
 Cat c2 = new Cat();
```

构造方法的作用

```
class Cat{
  String name;
  String color;
  char sex;
  String type;
  Cat(String n, String c, char s,  String t){
    name = n;
    color = c;
    sex = s;
    type = t;
  }
}
Cat c1 = new Cat("花花","brown",'f',"$%$");
Cat c2 = new Cat("小黑","black",'f',"#@@");
```

调用构造方法创建对象

- Java使用new关键字调用构造方法来创建对象和初始化新对象。
  语法：

  new 构造方法名称（[实际参数]）；

  ```
  Person p1 = new Person( );
  Person p2 = new Person("Mike",20 );
  ```

- 对象创建过程如下：
  (1) new关键字创建对象，为对象分配空间，为成员变量赋初值（0值）
  (2) 调用构造方法为新对象初始化
  (3) 构造方法会返回新对象的引用，该引用可以赋值给“同类型”的变量

创建对象

- 变量分类
  - **基本类型**变量：int, byte, float, double, char等。
  - **引用类型**变量：数组变量、String类型、对象变量
- 声明对象变量语法： **类名 对象变量;**

例如：

```
Person p1; //声明了一个Person类型的引用变量p1，里面是空值。
```

创建对象变量语法：

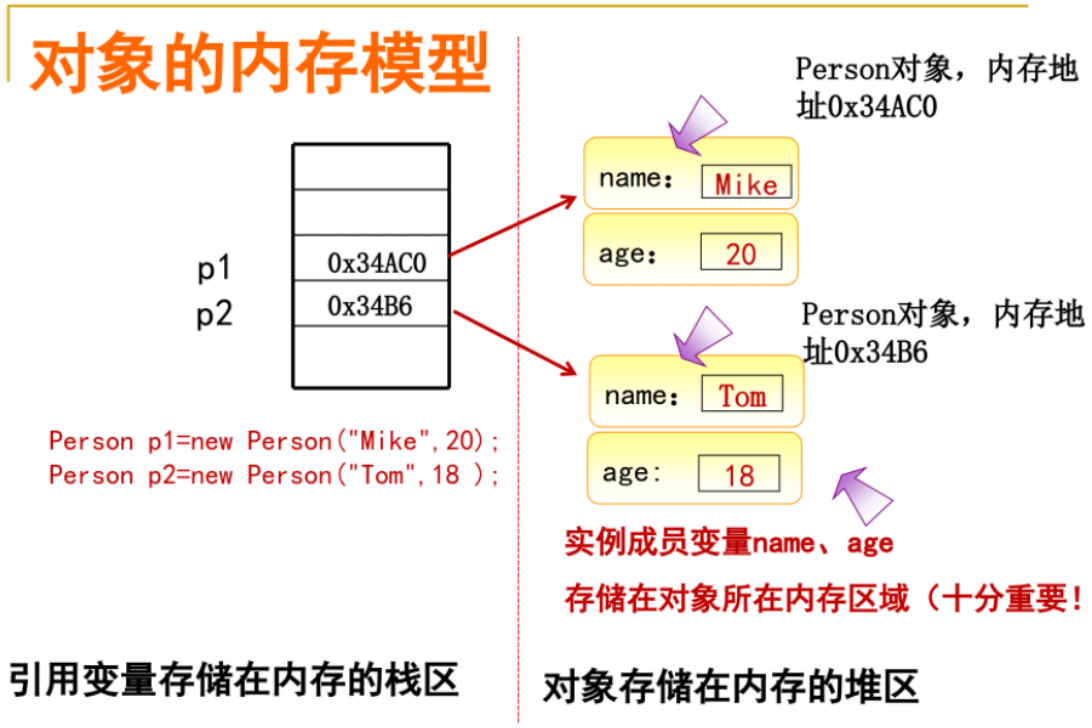
类名 对象变量= new 构造方法名([参数]);

understand the store model of objects and the relationship between objects and reference variable

Object are stored in heap memory and are not accessible to java. Access by reference,manipulate objects.

relationship

Reference variable are used to store the address of the object in the heap



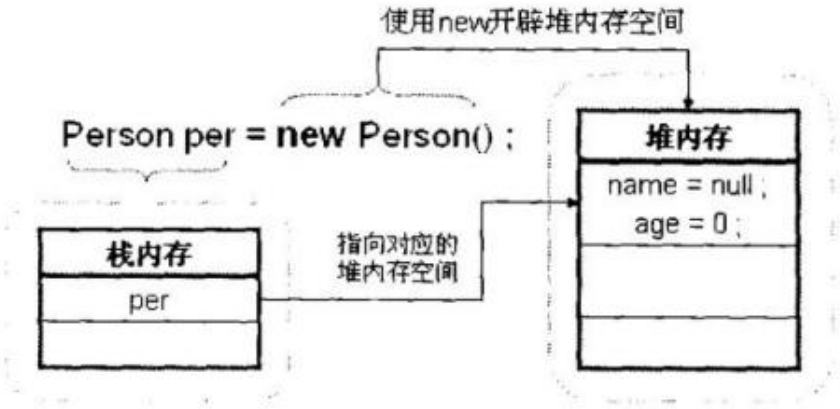


图 5-4　对象的实例化过程

使用对象

- 当一个对象创建成功后，对象被保存在堆内存中，Java程序不允许直接访问堆内存中的对象，Java通过该对象的引用访问、操作对象。对象的引用存放在栈内存中。
- 访问对象的成员变量：

  对象的引用.成员变量名　　例如：p1.name
- 调用对象的方法：

  对象的引用.方法名(实参)　例如：　p1.speak();