java关键字

this关键字：

- 在类的方法定义中使用this关键字代表使用该方法的对象的引用
- 当必须指出当前使用方法的对象是谁时要使用this
- 有时使用this可以处理成员变量和参数重名的情况
- this可以看作是一个变量，它的值是当前对象的引用
- 在构造方法中调用自己的另一个构造方法时需使用

this关键字举例：

一、返回当前对象（对象作为方法返回值）

```
package 面向对象编程;

public class day03_this关键字 {
    public static class Leaf{
        int i = 0;
        Leaf(int i){
            this.i = i;
        }
        public Leaf increment(){
            i++;
            return this;
        }
        public void print(){
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        Leaf leaf = new Leaf(200);
        leaf.increment().increment().print();
    }
}
```

二、解决标识符歧义问题（同名的方法参数和成员变量，方法参数的优先级高，this不能省略）

```
public static class Apple{
        private String color;
        Apple(String color){
            color = color;
        }
        public String getColor(){
            return this.color;
        }
```

```
    public static void main(String[] args) {
        Apple apple = new Apple("red");
        System.out.println(apple.getColor());
    }
}
```

输出：null

```
public static class Apple{
    private String color;
    Apple(String color){
        this.color = color;
    }
    public String getColor(){
        return this.color;
    }

    public static void main(String[] args) {
        Apple apple = new Apple("red");
        System.out.println(apple.getColor());
    }
}
```

输出：red

三、在构造器中调用另一个构成方法

```
public static class Flower{
    int petalConunt = 0;
    String s = "initial value";

    Flower(int petalConunt){
        this.petalConunt = petalConunt;
        System.out.println("调用int型构造方法");
    }
    Flower(String s){
        this.s = s;
        System.out.println("调用string型构造方法");
    }
    Flower(String s,int petalConunt){
        this(petalConunt);
        this.s = s;
        System.out.println("调用string和int型构造方法");
    }
    Flower(){
        this("hi",47);
        System.out.println("调用无参构造方法");
    }
```

```
    public void display(){
        System.out.println(petalConunt);
    }

    public static void main(String[] args) {
        Flower flower = new Flower();
        flower.display();
    }
}

结果:
    调用int型构造方法
    调用string和int型构造方法
    调用无参构造方法
    47
```

Static关键字:

- 在类中，用static声明的成员变量为静态变量，它为该类的共有变量，在第一次使用时被初始化，对应该类的所以对象来说，static成员变量只有一份
- 用static声明的方法为静态方法，在调用该方法时，不会将对象的引用传递给它，所以在static方法中不可以访问非static的成员
- 静态方法不再是针对某个对象的调用，所以不能访问非静态成员，静态方法只能访问静态成员变量
- 可以通过对象引用或类名（不需要实例化）访问静态成员

实例一

```
package 面向对象编程.static关键字;

public class Cat {
    private static int sid;
    private String name;
    int id;
    Cat(String name){
        name = name;
        sid++;
    }
    public void info(){
        System.out.println(name + " " + sid);
    }

    public static void main(String[] args) {
        Cat.sid = 100;
        Cat m = new Cat("m");
        m.info();
        System.out.println(sid);
        Cat p = new Cat("p");
        System.out.println(sid);
        p.info();
```

```
    }
}
```

实例二

```
package 面向对象编程.static关键字;

class Apple{
    public Apple getDeleteApple(Apple apple){
        System.out.println("需要获得一个削好皮的苹果");
        return Peel.deletePeel(apple);
    }
}
class Peel{
    static Apple deletePeel(Apple apple){
        System.out.println("正在削皮");
        return apple;
    }
}
class Person{
    public void eat(Apple apple){
        System.out.println("我要吃苹果");
        apple.getDeleteApple(apple);
        System.out.println("获得一个削好皮的苹果");
    }
}
public class test {
    public static void main(String[] args) {
        Person person = new Person();
        person.eat(new Apple());
    }
}
```