第五章 类和对象

1. 类

1.1 类的概述

1. 类：对现实世界客观存在的事物的描述，也叫抽象定义
2. 抽象定义：对客观存在事物的共性特征抽取
3. 举例：猫，狗，牛，空调，鸟等
    a. 猫：
        i. 能干什么（行为）：喵喵，爱吃鱼，抓老鼠，爬树
        ii. 有什么（属性）：有毛，有爪子，有胡须等
    b. 空调：行为：吹风，制冷，制热；属性：型号，功率 给这类电器起了一个名字叫空调

1.2 类的两大成员（属性和方法）

1. 属性：有什么
2. 方法：能干什么
3. 老师为什么叫老师
    a. 属性：姓名，年龄，性别，职称，讲师编号等
    b. 方法：授课，布置作业等
4. 有什么和能干什么都是对现实世界客观存在事物的描述

1.3 类和对象的关系

1. 类是对象的抽象定义
2. 对象是类的实例，创建对象使用new关键字
3. 一个类可以创建多个对象
4. 每个对象都有一块独立的内存
5. 类是对象的模版

1.4 简单认识属性和方法

代码块

```java
package com.powernode.class02;

/**
 * 老师为什么叫老师
 *   1. 属性：姓名，年龄，性别，职称，讲师编号等
 *   2. 方法：授课，布置作业等
 */
class Teacher{
    //1.属性|实例变量|字段|成员变量
    String name = "zs";
    int age = 23;
    char sex = '男';
    //2.方法
    /**
     * 1.static修饰的方法，不需要创建对象可以直接使用
     * 2.非static修饰的方法，需要创建对象后才可以使用，所以这种方法也叫实例方法
     */
    public  void lecture(){
        System.out.println("授课");
    }
    public void assignHomeWork(){
        System.out.println("布置家庭作业");
    }
}


public class Test01 {
    public static void main(String[] args) {

    }
}
```

2. 属性

2.1 对象访问属性

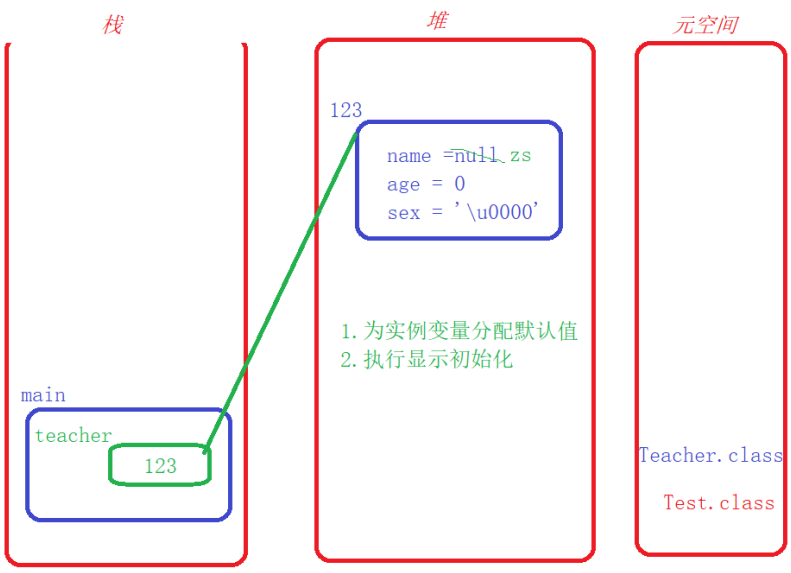
代码块

```java
package com.powernode.class03;

class Teacher{
    //属性：实例变量，属性对象的变量
    String name = "zs";
    int age;
    char sex;
```

```java
}
public class Test {
    public static void main(String[] args) {
        /**
         * 1.创建对象
         *    语法：类名称  对象名称 = new 类名称();
         *          1.类名称:创建哪个类的对象使用哪个类的名称
         *          2.对象名称，可以理解为变量名称，可以自定义
         * 2.对象访问实例变量
         *      对象名称.实例变量名称
         */
        //类名称  对象名称 = new 类名称();
        Teacher teacher = new Teacher();
        //对象名称.实例变量名称
        System.out.println(teacher.name);//zs
        String sname = teacher.name;//拿到name命名空间中的值，赋值给sname
        System.out.println(sname);
        //变量：先声明，后赋值，再使用？age和sex为什么没报错呢
        System.out.println(teacher.age);
        System.out.println(teacher.sex);
    }
}
```

2.2 对象的内存分析

```java
package com.powernode.class03;

class Teacher{
    //属性：实例变量，属性对象的变量
    String name = "zs";
    int age;
    char sex;
}
public class Test {
    public static void main(String[] args) {
        //类名称  对象名称 = new 类名称();
        Teacher teacher = new Teacher();
        //对象名称.实例变量名称
        System.out.println(teacher.name);//zs
        String sname = teacher.name;
        System.out.println(sname);
        //变量：先声明，后赋值，再使用？
        System.out.println(teacher.age);
        System.out.println(teacher.sex);
    }
}
```



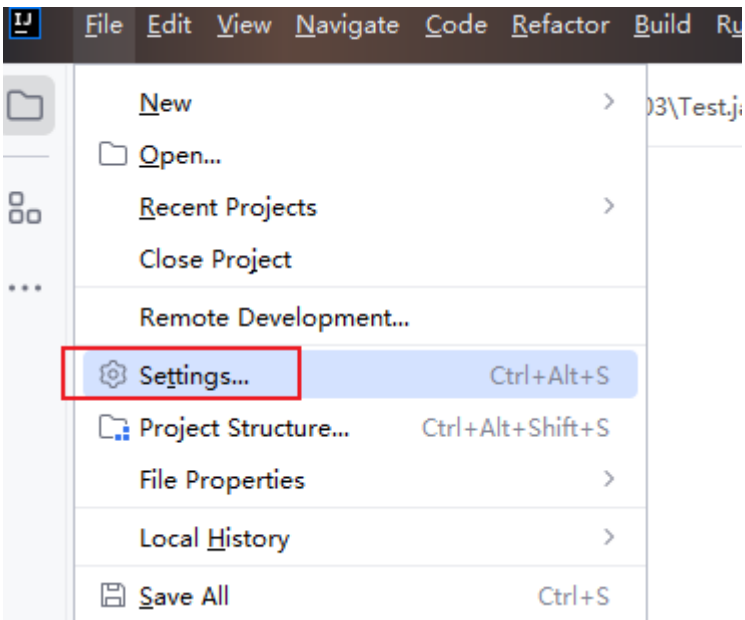
2.3 对象地址（了解）

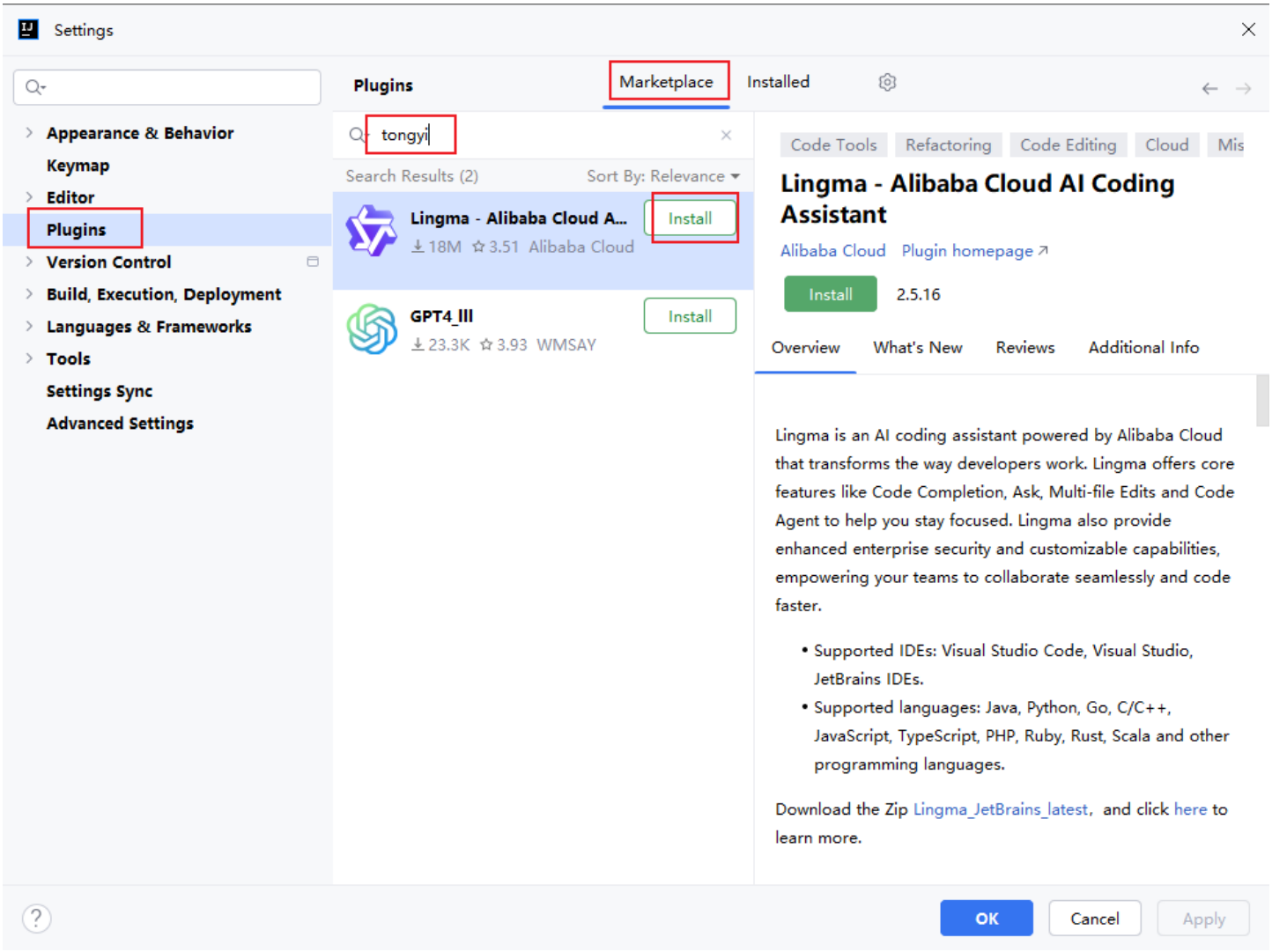
代码块

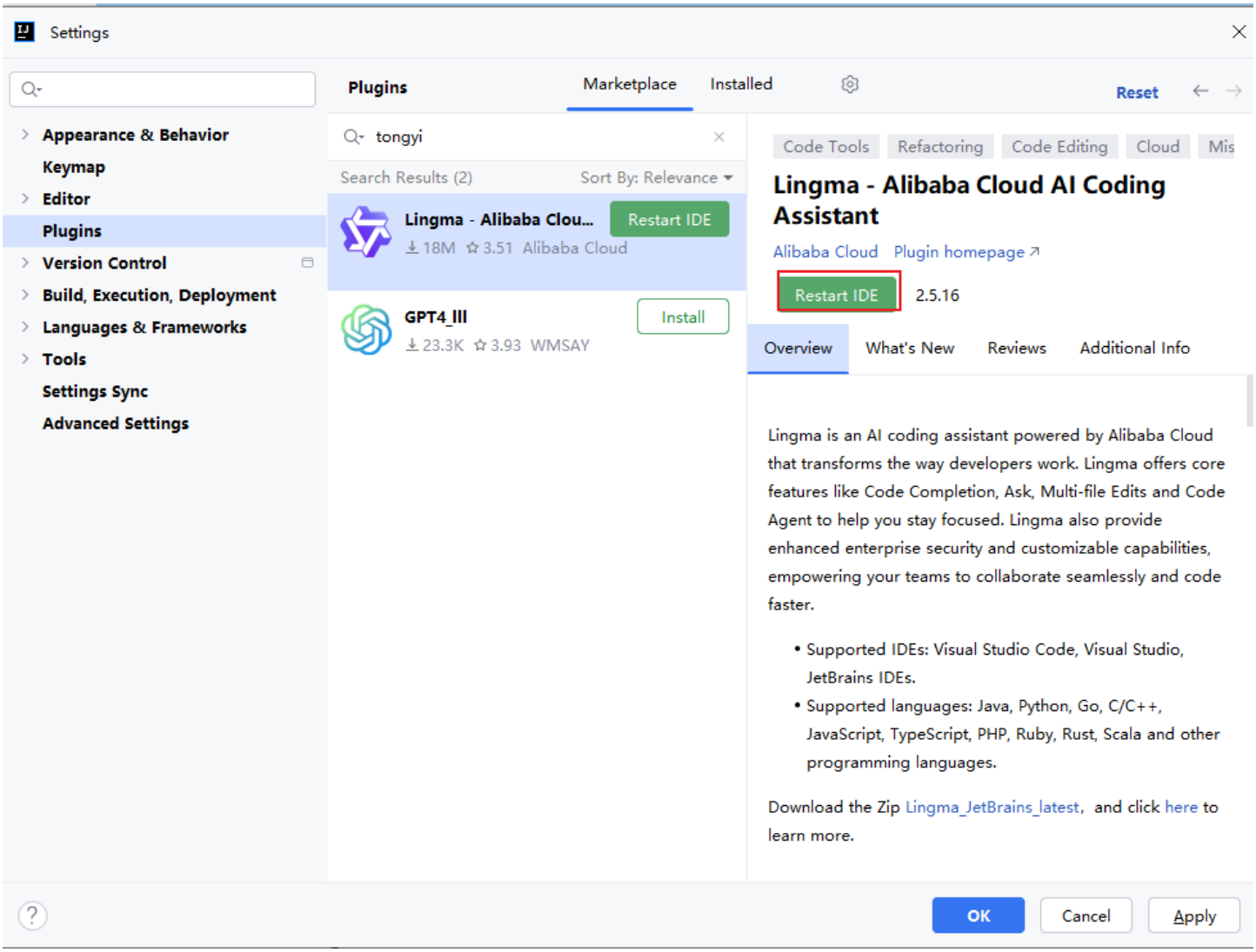
```java
package com.powernode.class04;
```

```java

class Teacher{
    String name;
    int age;
}
public class Test {
    public static void main(String[] args) {
        //类名称 对象名称 = new  类名称();
        Teacher teacher = new Teacher();
        //com.powernode.class04.Teacher@2f4d3709
        System.out.println(teacher);
        //了解，不需要掌握
        System.out.println(Teacher.class);//拿到元空间中的class对象（Teacher的字节码文件对象）
        System.out.println(Teacher.class.getName());//通过字节码文件对象拿到 包名 + 类名称
        //teacher.hashCode() 通过hash算法，计算出的一个十进制数字
        System.out.println(Teacher.class.getName() + "@" + teacher.hashCode());
        //Integer.toHexString(teacher.hashCode()) 把十进制的hashCode转换为十六进制
        System.out.println(Teacher.class.getName() + "@" + Integer.toHexString(teacher.hashCode()));
    }
}
```

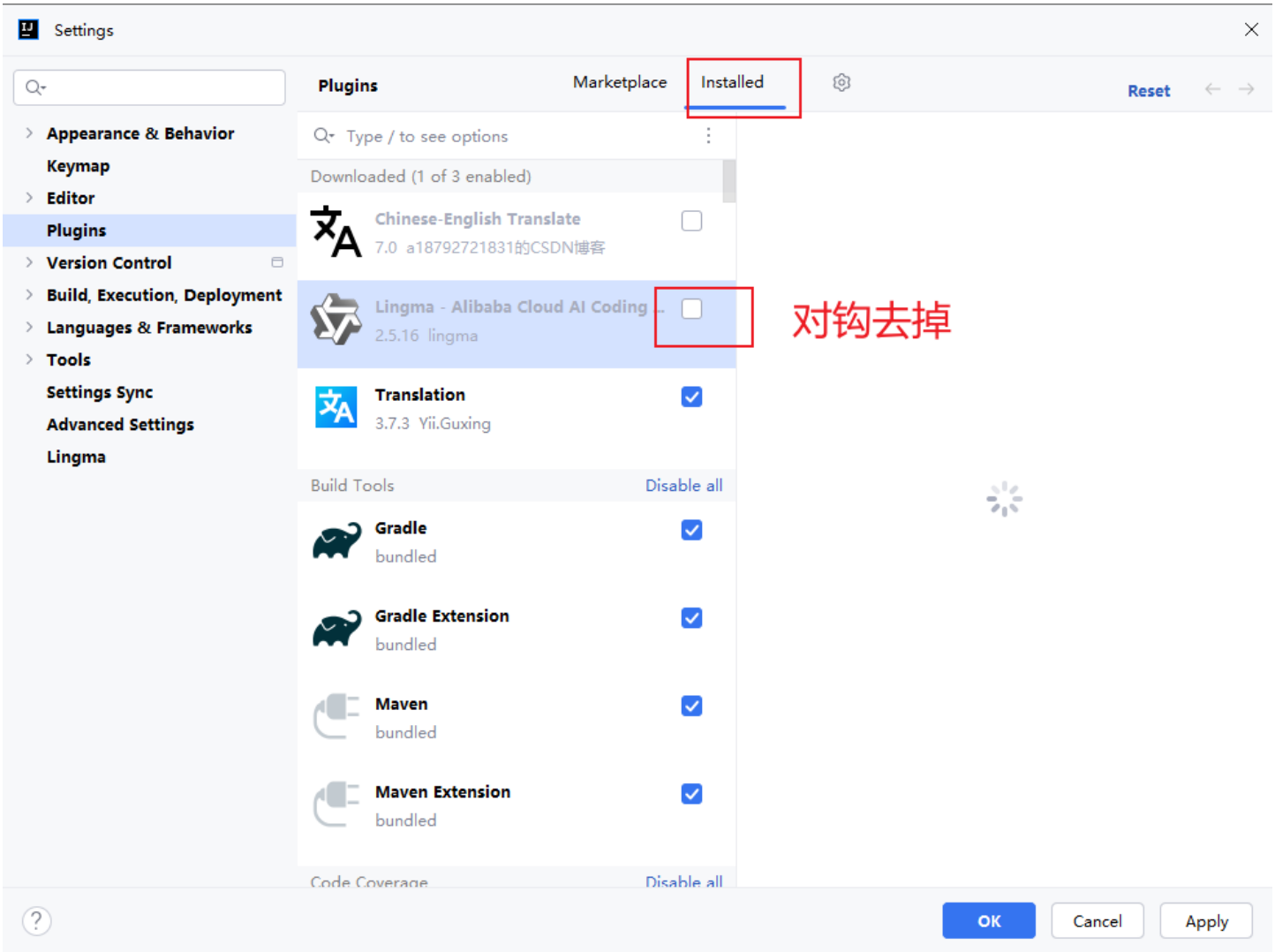
2.4 集成AI插件




Settings
Plugins
Marketplace
Installed
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Build, Execution, Deployment
Languages & Frameworks
Tools
Settings Sync
Advanced Settings
tongyi
Search Results (2)
Sort By: Relevance
Lingma - Alibaba Cloud A...
18M 3.51 Alibaba Cloud
Install
GPT4_III
23.3K 3.93 WMSAY
Install
Code Tools
Refactoring
Code Editing
Cloud
Mis
Lingma - Alibaba Cloud AI Coding Assistant
Alibaba Cloud
Plugin homepage
Install
2.5.16
Overview
What's New
Reviews
Additional Info
Lingma is an AI coding assistant powered by Alibaba Cloud that transforms the way developers work. Lingma offers core features like Code Completion, Ask, Multi-file Edits and Code Agent to help you stay focused. Lingma also provide enhanced enterprise security and customizable capabilities, empowering your teams to collaborate seamlessly and code faster.
Supported IDEs: Visual Studio Code, Visual Studio, JetBrains IDEs.
Supported languages: Java, Python, Go, C/C++, JavaScript, TypeScript, PHP, Ruby, Rust, Scala and other programming languages.
Download the Zip Lingma_JetBrains_latest, and click here to learn more.
OK
Cancel
Apply



- 不使用 AI插件（lingma）



2.5 基本类型和引用类型的默认值

代码块

```
1  package com.powernode.class05;
2
3  public class DefaultValue {
4      //基本类型
5      byte b;
6      short s;
7      int i;
8      long l;
9
10     char c;//\u0000
11
12     float f;
13     double d;
14
```

```
    boolean flag;// false
    //引用类型
    String str;// null

    public static void main(String[] args) {
        DefaultValue dv = new DefaultValue();
        System.out.println("dv.b = " + dv.b);
        System.out.println("dv.s = " + dv.s);
        System.out.println("dv.i = " + dv.i);
        System.out.println("dv.l = " + dv.l);

        System.out.println("dv.c = " + dv.c);

        System.out.println("dv.f = " + dv.f);
        System.out.println("dv.d = " + dv.d);

        System.out.println("dv.flag = " + dv.flag);

        System.out.println("dv.str = " + dv.str);
    }
}
```

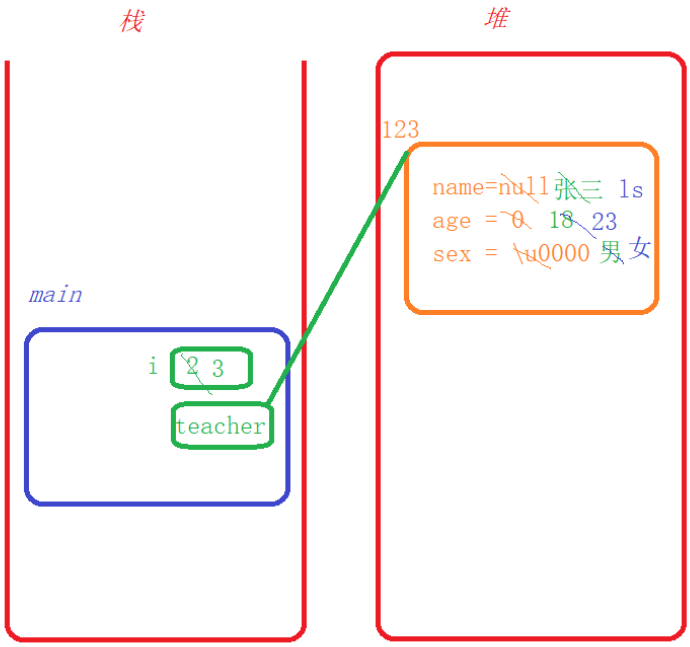
2.6 修改属性值

代码块

```
package com.powernode.class05;

class Teacher{
    String name = "张三";
    int age = 18;
    char sex = '男';
}
public class Test {
    public static void main(String[] args) {
        //1.修改局部变量的值
        int i = 2;
        i = 3;
        //2.实例变量修改值
        //2.1创建对象
        Teacher teacher = new Teacher();
        //2.2通过对象修改变量值
        teacher.name = "ls";
        teacher.age = 23;
        teacher.sex = '女';

```

```
21            System.out.println(teacher.name);
22            System.out.println(teacher.age);
23            System.out.println(teacher.sex);
24        }
25    }
```

```
class Teacher{
    String name = "张三";
    int age = 18;
    char sex = '男';
}
public class Test {
    public static void main(String[] args) {
        //1.修改局部变量的值
        int i = 2;
        i = 3;
        //2.实例变量修改值
        //2.1创建对象
        Teacher teacher = new Teacher();
        //2.2通过对象修改变量值
        teacher.name = "ls";
        teacher.age = 23;
        teacher.sex = '女';

        System.out.println(teacher.name);
        System.out.println(teacher.age);
        System.out.println(teacher.sex);
    }
}
```



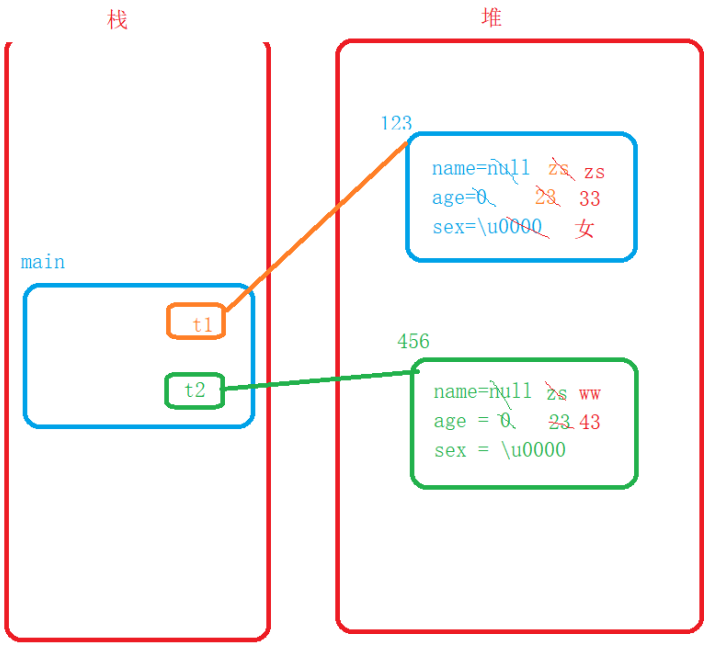
2.7 一个类可以创建多个对象

代码块

```
1   package com.powernode.class06;
2
3   class Teacher{
4       String name = "zs";
5       int age = 23;
6       char sex;
7
8   }
9   public class Test {
10      public static void main(String[] args) {
11          //类名称 对象名称 = new 类名称()
12          Teacher t1  = new Teacher();
13          Teacher t2  = new Teacher();
14
15          t1.name = "zs";
16          t1.age = 33;
17          t1.sex = '女';
```

```
18
19          t2.name = "ww";
20          t2.age = 43;
21
22          System.out.println("t1.name = " + t1.name);
23          System.out.println("t1.age = " + t1.age);
24          System.out.println("t1.sex = " + t1.sex);
25
26          System.out.println("t2.name = " + t2.name);
27          System.out.println("t2.age = " + t2.age);
28          System.out.println("t2.sex = " + t2.sex);
29      }
30  }
```

```
class Teacher{
    String name = "zs";
    int age = 23;
    char sex;
}
public class Test {
    public static void main(String[] args) {
        //类名称 对象名称 = new 类名称()
        Teacher t1  = new Teacher();
        Teacher t2  = new Teacher();

        t1.name = "zs";
        t1.age = 33;
        t1.sex = '女';

        t2.name = "ww";
        t2.age = 43;

        System.out.println("t1.name = " + t1.name);
        System.out.println("t1.age = " + t1.age);
        System.out.println("t1.sex = " + t1.sex);

        System.out.println("t2.name = " + t2.name);
        System.out.println("t2.age = " + t2.age);
        System.out.println("t2.sex = " + t2.sex);
    }
```



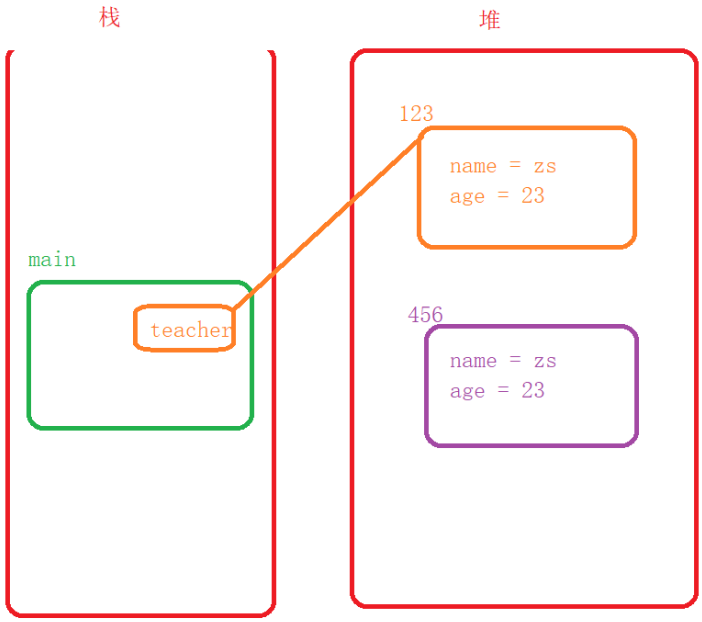
2.8 匿名对象

代码块

```
1   package com.powernode.class07;
2
3   class Teacher{
4       String name = "zs";
5       int age = 23;
6   }
7   public class Test {
```

```java
    public static void main(String[] args) {
        //需求：访问name并输出
        //第一种方案：
        Teacher teacher = new Teacher();
        System.out.println(teacher.name);
        System.out.println(teacher.age);

        //第二种方案
        System.out.println(new Teacher().name);
        //System.out.println(new Teacher().age);
        /**
         * 两种方案的区别：
         *   第一种方案：
         *      1.更加灵活，对象可以多次使用
         *      2.内存利用率：对象占用内存过长
         *   第二中方案：
         *      1.不灵活，只能使用一次
         *      2.内存利用率：对象占用内存较短
         */

    }
}
```

```java
class Teacher{
    String name = "zs";
    int age = 23;
}
public class Test {
    public static void main(String[] args) {
        //需求：访问name并输出
        //第一种方案：
        Teacher teacher = new Teacher();
        System.out.println(teacher.name);
        System.out.println(teacher.age);

        //第二种方案
        System.out.println(new Teacher().name);
        //System.out.println(new Teacher().age);
        /**
         * 两种方案的区别：
         *   第一种方案：
         *      1.更加灵活，对象可以多次使用
         *      2.内存利用率：对象占用内存过长
         *   第二中方案：
         *      1.不灵活，只能使用一次
         *      2.内存利用率：对象占用内存较短
         */
```



- 声明一个Student类
- 属性name和age

- 修改name和age的值
- 输出修改过的值

3. 方法

3.1 静态方法和实例方法

1. **静态方法**
   a. 使用static修饰的方法，称为静态方法，也叫类方法
   b. 静态方法的访问：**类名称.方法名称([实参列表])**
   c. 静态方法访问时机：类加载后即可访问（把.class文件加载到元空间中，该方法就可以使用）
2. **实例方法**
   a. 不使用static修饰的方法，称为实例方法，也叫对象方法
   b. 实例方法的访问：对象名称**.方法名称([实参列表])**
   c. 实例方法访问时机: 创建对象后才可以访问
3. 创建对象的步骤：
   a. 加载.class文件到元空间
   b. 创建对象
4. 静态方法和实例方法的访问：

代码块

```
package com.powernode.class09;
class Teacher{
    //静态方法
    public static void method01(){
        System.out.println("Teacher.method01");
    }
    //实例方法
    public  void method02(){
        System.out.println("Teacher.method02");
    }

}
public class Test {
    public static void main(String[] args) {
        //静态方法访问：类名称.方法名称（[实参列表]）
```

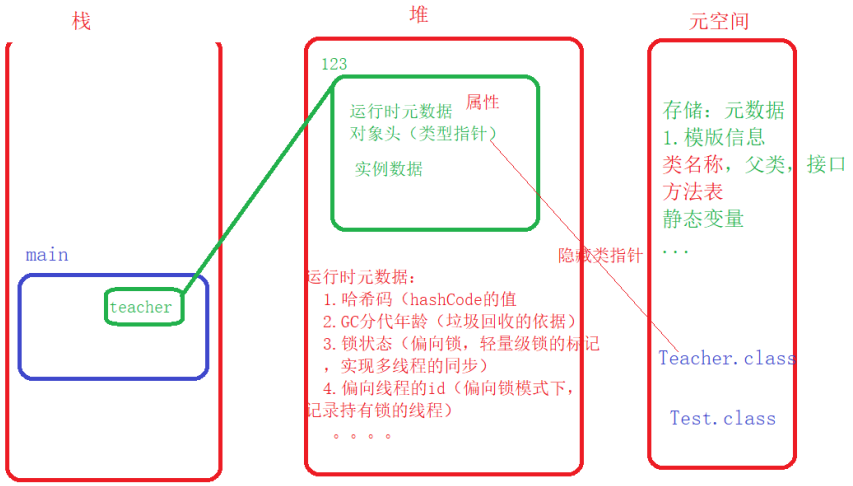
```
        Teacher.method01();
        //实例方法访问：对象名称.方法名称（[实参列表]）
        Teacher teacher = new Teacher();
        teacher.method02();
    }
}
```

3.2 NullPonterException

代码块

```
package com.powernode.class10;

class Teacher{
    String name = "zs";
    public void lecture(){
        System.out.println("Teacher.lecture");
    }
}
public class Test {
    public static void main(String[] args) {
        Teacher teacher = new Teacher();
        System.out.println(teacher.name);
        teacher.lecture();
        teacher = null;
        /**
         * 1.NullPointerException:Cannot read field "name" because "teacher" is null
         * 2.NullPointerException:称为：空指针异常
         *    1.什么原因造成的：
         *      1.使用了null的对象访问了属性
         *      2.使用了null的对象访问方法
         *
         */
        //System.out.println(teacher.name);
        teacher.lecture();
    }
}
```

```java
package com.powernode.class10;

class Teacher{
    String name = "zs";
    public void lecture(){
        System.out.println("Teacher.lecture");
    }
}
public class Test {
    public static void main(String[] args) {
        Teacher teacher = new Teacher();
        System.out.println(teacher.name);
        teacher.lecture();
        teacher = null;
        //System.out.println(teacher.name);
        teacher.lecture();
    }
}
```

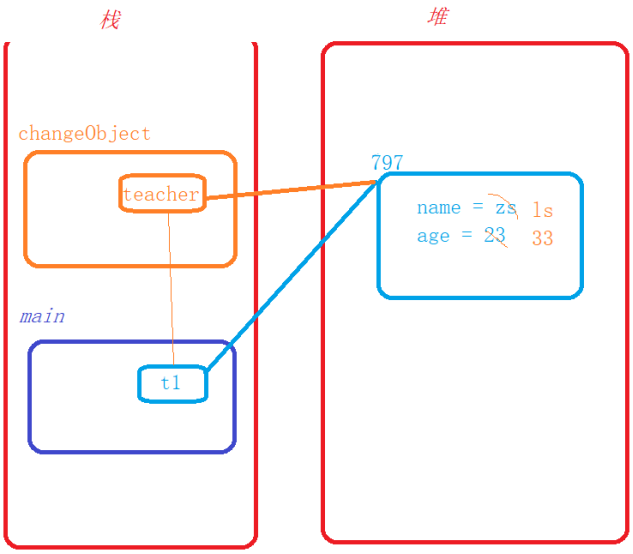


3.3 引用类型的地址传递

代码块

```java
package com.powernode.class11;

class Teacher{
    String name = "zs";
    int age = 23;

}
public class Test {
    public static void main(String[] args) {
        Teacher t1 = new Teacher();
        System.out.println("t1.name = " + t1.name);//zs
        System.out.println("t1.age = " + t1.age);//23
        changeObject(t1);
        System.out.println("t1.name = " + t1.name);//ls
        System.out.println("t1.age = " + t1.age);//33
    }

    public static void changeObject(Teacher teacher) {
        System.out.println("teacher.name = " + teacher.name);//zs
        System.out.println("teacher.age = " + teacher.age);//23
        teacher.name = "ls";
        teacher.age = 33;
    }
}
```

```
package com.powernode.class11;

class Teacher{
    String name = "zs";
    int age = 23;

}
public class Test {
    public static void main(String[] args) {
        Teacher t1 = new Teacher();
        System.out.println("t1.name = " + t1.name);//zs
        System.out.println("t1.age = " + t1.age);//23
        changeObject(t1);
        System.out.println("t1.name = " + t1.name);//ls
        System.out.println("t1.age = " + t1.age);//33
    }

    public static void changeObject(Teacher teacher) {
        System.out.println("teacher.name = " + teacher.name);//zs
        System.out.println("teacher.age = " + teacher.age);//23
        teacher.name = "ls";
        teacher.age = 33;
    }
}
```



4. 实例方法访问实例变量

代码块

```
1   package com.powernode.class12;
2
3   class Teacher{
4       String name = "zs";
5       int age = 23;
6
7       //实例方法：描述实例变量的信息
8       public String getDetails(){
9           /* String info = "姓名：" + name + "\t年龄：" + age;
10           return  info;*/
11          return  "姓名：" + name + "\t年龄：" + age;
12      }
13  }
14  public class Test {
15      public static void main(String[] args) {
16          Teacher teacher = new Teacher();
17          System.out.println(teacher.getDetails());
18      }
19  }
```

5. JVM的垃圾回收机制

5.1 栈：存储

1. 局部变量
2. 方法调用信息

- 方法被调用：JVM会为方法分配一个对应的栈帧，栈帧中存储的是方法的参数和局部变量，当这个方法调用完毕，对应的栈帧会清除，局部变量失效，所就是所谓的弹栈

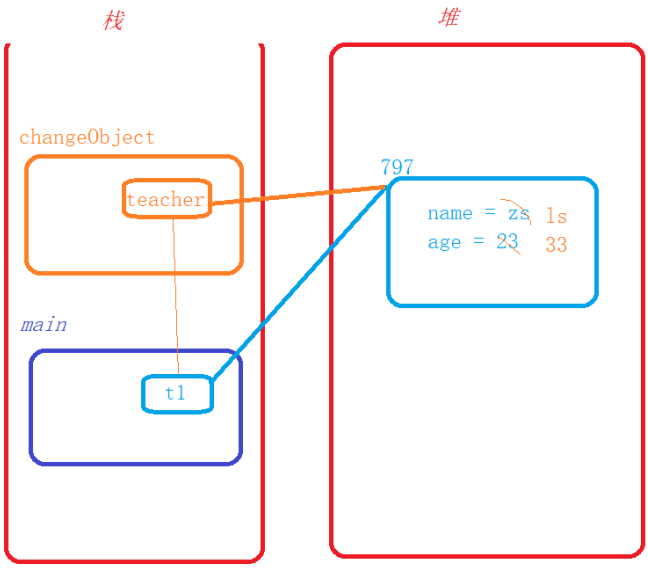
5.2 堆：存储

1. new 关键字创建的对象
2. 当一个对象被创建，堆中存储的是对象的信息，栈中存储的是对象的引用地址
3. 栈的地址指向堆中的空间
4. 当这个对象没有被引用时，视为垃圾对象，该对象会被垃圾回收机制回收
5. 不一定是立即回收，什么时候回收，取决于JVM什么时候调用它
6. 以上几点，称为JVM的垃圾回收机制

5.3 元空间：存储

1. Java在运行之前，会把.class文件加载到元空间中（JDK8之后）
2. JVM实行的是懒加载，是按需加载（后面讲）

```
package com.powernode.class11;

class Teacher{
    String name = "zs";
    int age = 23;

}
public class Test {
    public static void main(String[] args) {
        Teacher t1 = new Teacher();
        System.out.println("t1.name = " + t1.name);//zs
        System.out.println("t1.age = " + t1.age);//23
        changeObject(t1);
        System.out.println("t1.name = " + t1.name);//ls
        System.out.println("t1.age = " + t1.age);//33
    }

    public static void changeObject(Teacher teacher) {
        System.out.println("teacher.name = " + teacher.name);//zs
        System.out.println("teacher.age = " + teacher.age);//23
        teacher.name = "ls";
        teacher.age = 33;
    }
}
```



作业

1. 练习一：
    a. 编写一个Student类，包含name、age、sex、id、score（分数）属性，并为每个属性赋值，

b. 数据类型分别为String、int、char、int、double。

c. 类中声明一个say方法，返回String类型，方法返回属性的描述信息。

d. 在另一个TestStudent类中的main方法中，创建Student对象，并访问say方法和所有属性，并将调用结果打印输出。

2. 练习二：继续在main方法中创建另一个Student对象，将姓名赋予新值，并打印say方法结果。

3. 练习三：

a. 编写一个Cat类，包含name、age、weight（重量）属性 为每个属性赋值 ,类中声明一个say方法，返回类型为String，方法返回属性的描述信息。

b. 在另一个TestCat类的main方法中，创建Cat对象，并访问say方法和所有属性，将调用结果打印输出。

4. 练习题四：

a. 改写Cat类，name属性使用缺省初始值，age和weight属性使用显式初始值1和10。

b. 在TestCat类的main方法中，创建两个Cat对象，分别调用两对象的say方法，将调用结果打印输出。

c. 继续在main方法中，将两个Cat对象的name属性分别设为“喵喵 ”和“咪咪”，第二个对象的体重设为8。再分别调用两对象的say方法，将调用结果打印输出。

d. 根据输出结果，理解由同一类创建的不同对象的属性的独立性。