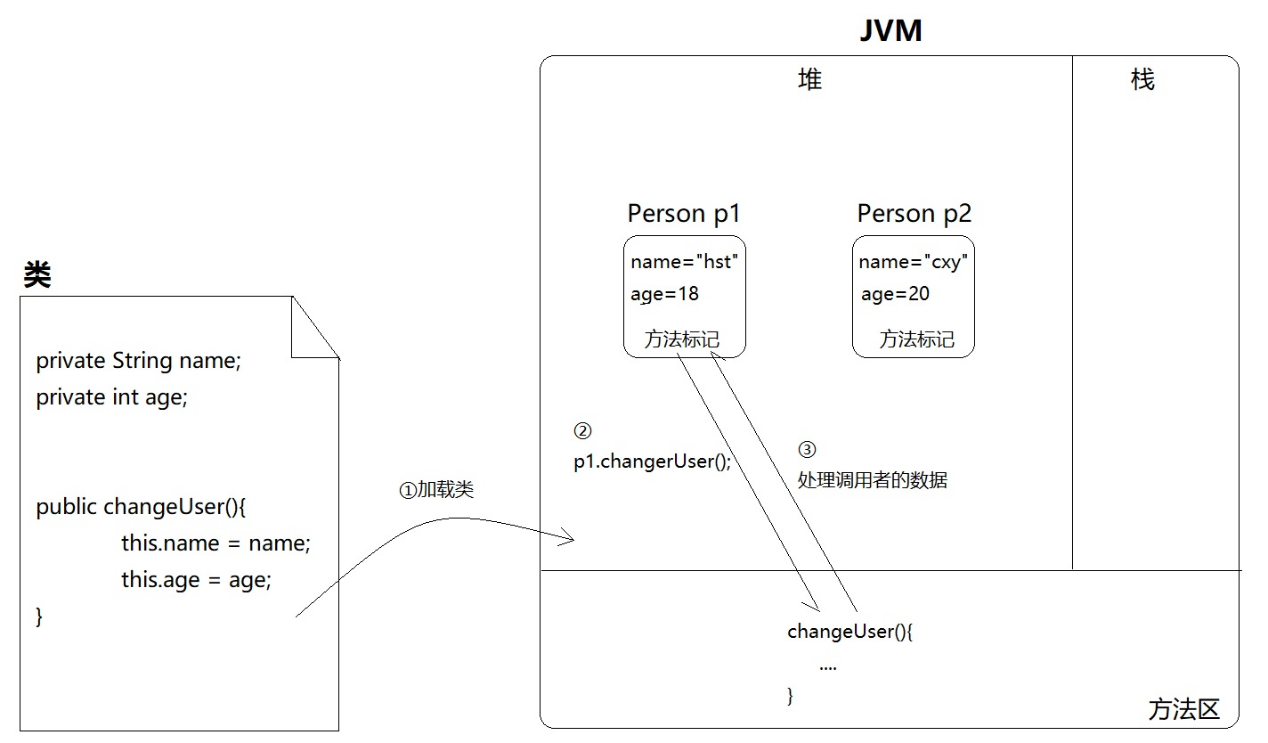
1、对象的本质理解为“多个相关数据的统一载体”。

2、Java中对象是在堆空间中生成的，数据会在堆空间占据一定内存开销。



3、因为多个个体，属性可能不同，比如我身高180，你身高150，我18岁，你30了。但我们都能跑、能跳、能吃饭，这些技能（method）都是共通的，没必要和属性数据一样单独在堆空间各存一份，所以被抽取出来存放。形成了方法。

方法相当于一套指令模板，谁都可以传入数据交给它执行，然后得到执行后的结果返回。

但此时会存在一个问题：张三这个对象调用了eat()方法，你应该把饭送到他嘴里，而不是送到李四嘴里。那么方法如何知道把饭送到哪里呢？

这就涉及到共性的方法如何处理特定的数据。

而Python的self、Java的this其实就是解决这个问题的。你可以理解为对象内部持有一个引用，当你调用某个方法时，必须传递这个对象引用，然后方法根据这个引用就知道当前这套指令是对哪个对象的数据进行操作了。

**4、static与this**

我们都知道，static修饰的属性或方法其实都是属于类的，是所有对象共享的。但接触Python后我多了一层理解：

之所以一个变量或者方法要声明为static，是因为

* static变量：大家共有的，大家都一样，不是特定的差异化数据
* static方法：这个方法不处理差异化数据

也就是说，static注定是与差异化数据无关，也就是与具体对象的数据无关。

以静态方法为例，当你确定一个方法只提供通用的操作流程，而不会在内部引用具体对象的数据时，你就可以把它定为静态方法。

这个其实和我们之前听到的解释不一样。网络上一贯的解释都是上来就告诉你静态方法不能访问实例变量，再解释为什么，是倒着解释的。而上面这段话的出发点是，当你满足什么条件时，你就可以把一个方法定为静态方法。

反正**静态方法是不处理特定对象数据的**。

这或许可以反过来解释，为什么Java中静态方法无法访问非静态数据（实例字段）和非静态方法（实例方法）。**因为Java不会在调用静态方法时传递this，静态方法内没有this当然无法调用实例相关的一切。**

**我们在一个实例方法中调用另一个实例方法或者实例变量时，其实都是通过this调用的，比如**

**public void test(){**

**System.out.println(this.name);**

**this.show();**

**}**

**只不过Java允许我们不显示书写。**

当然，有些培训班视频会说**静态方法随着类加载而加载，此时并没有对象实例化，所以静态方法无法访问实例相关数据**，倒也勉强说得通。看大家自己怎么理解了，能自圆其说即可。