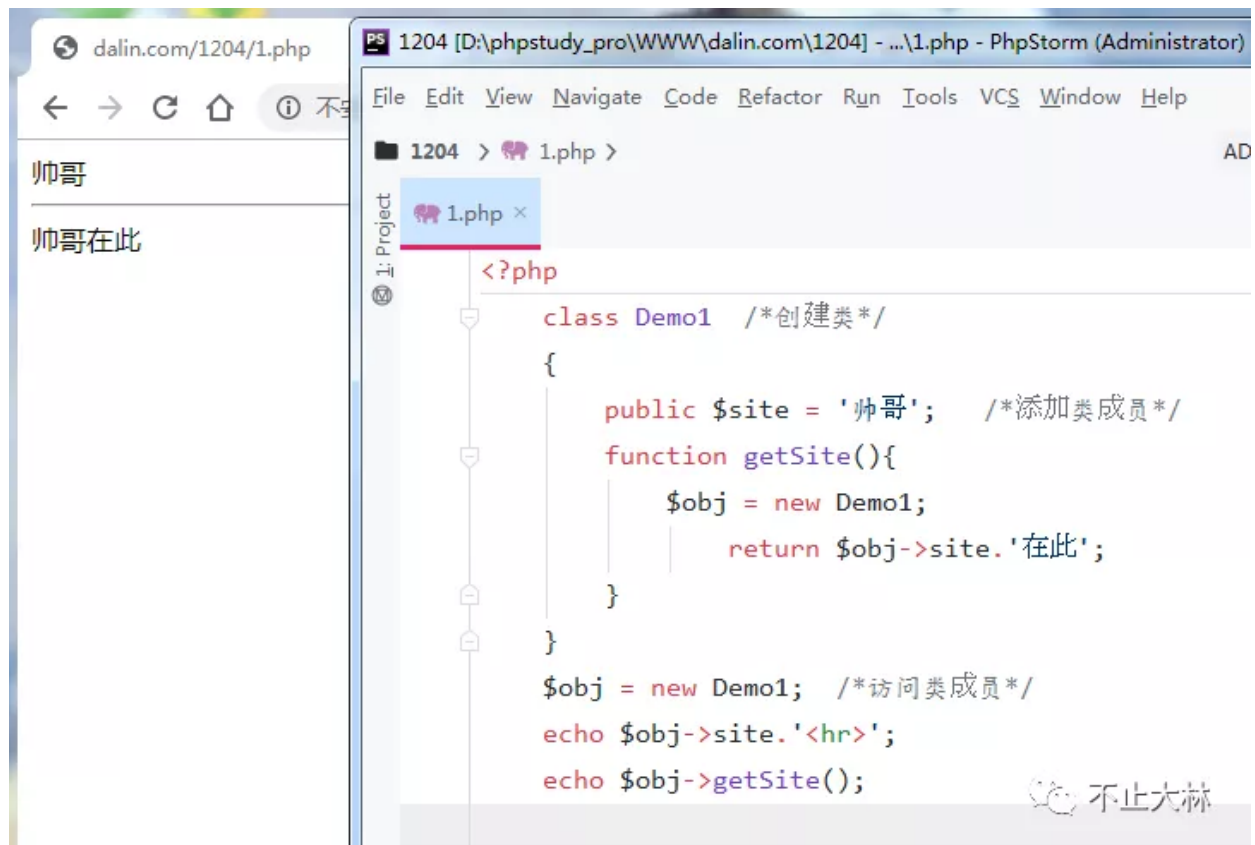
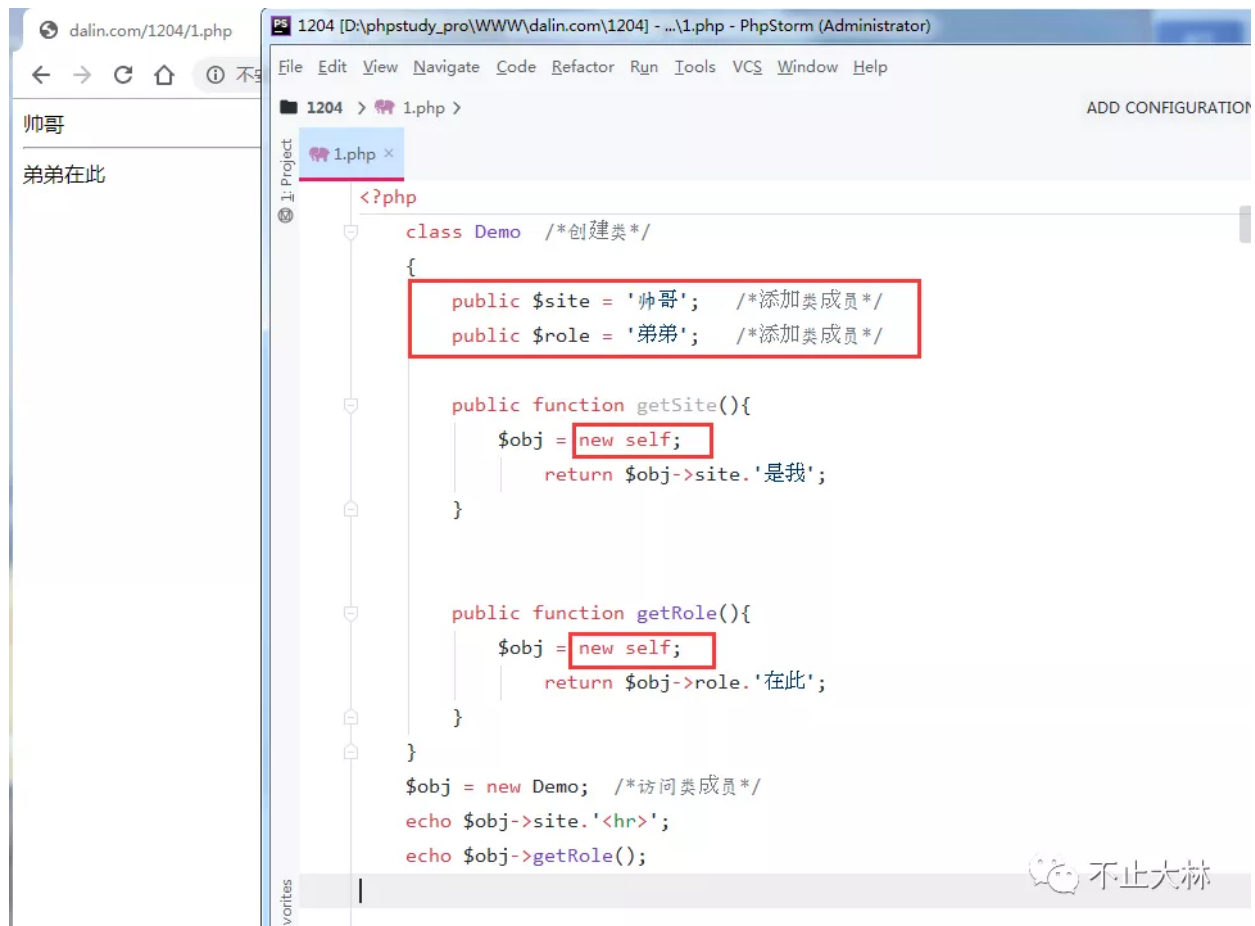


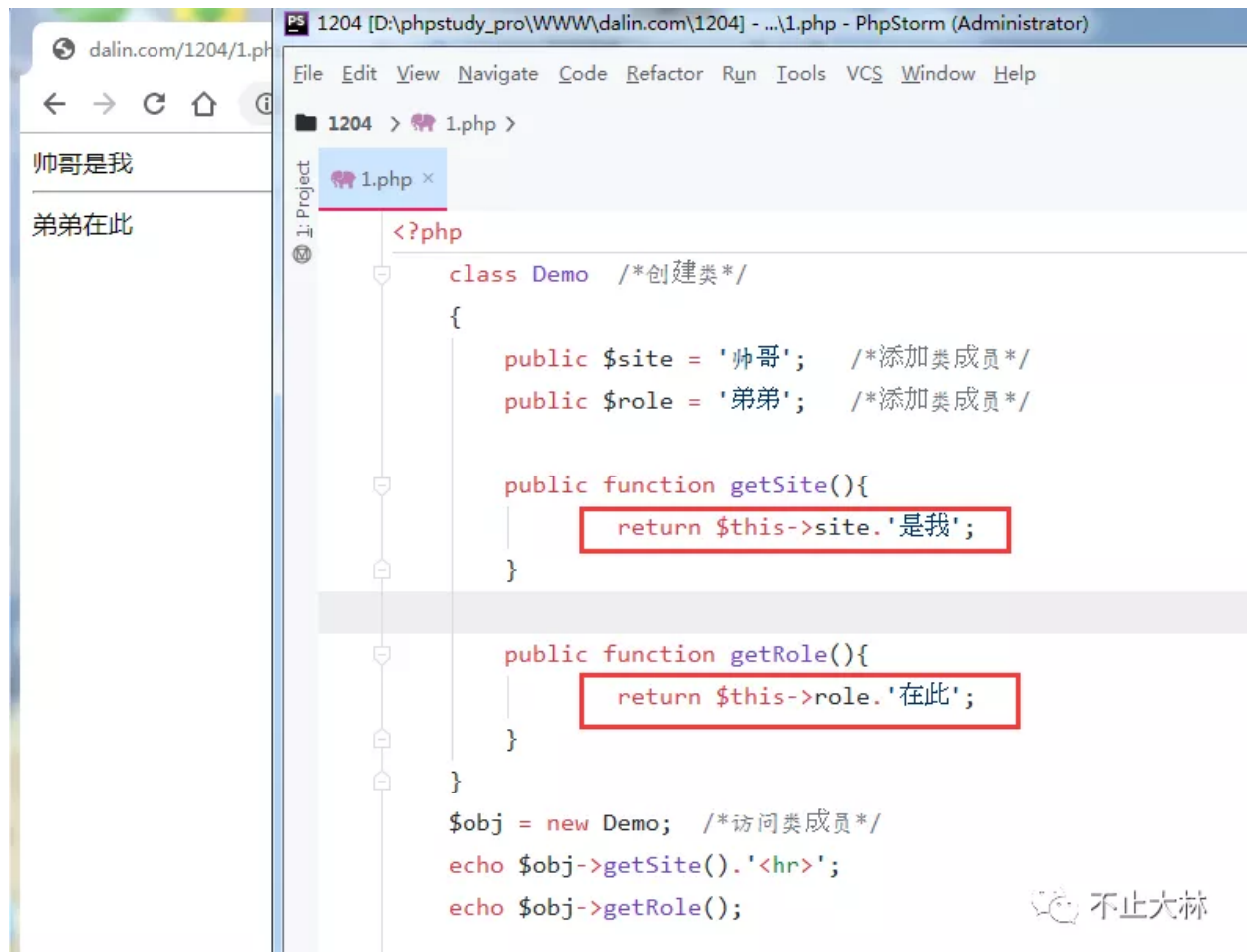
面向对象 创建类->添加类成员->访问类成员



self是类名的引用，始终与类class的当前类名绑定

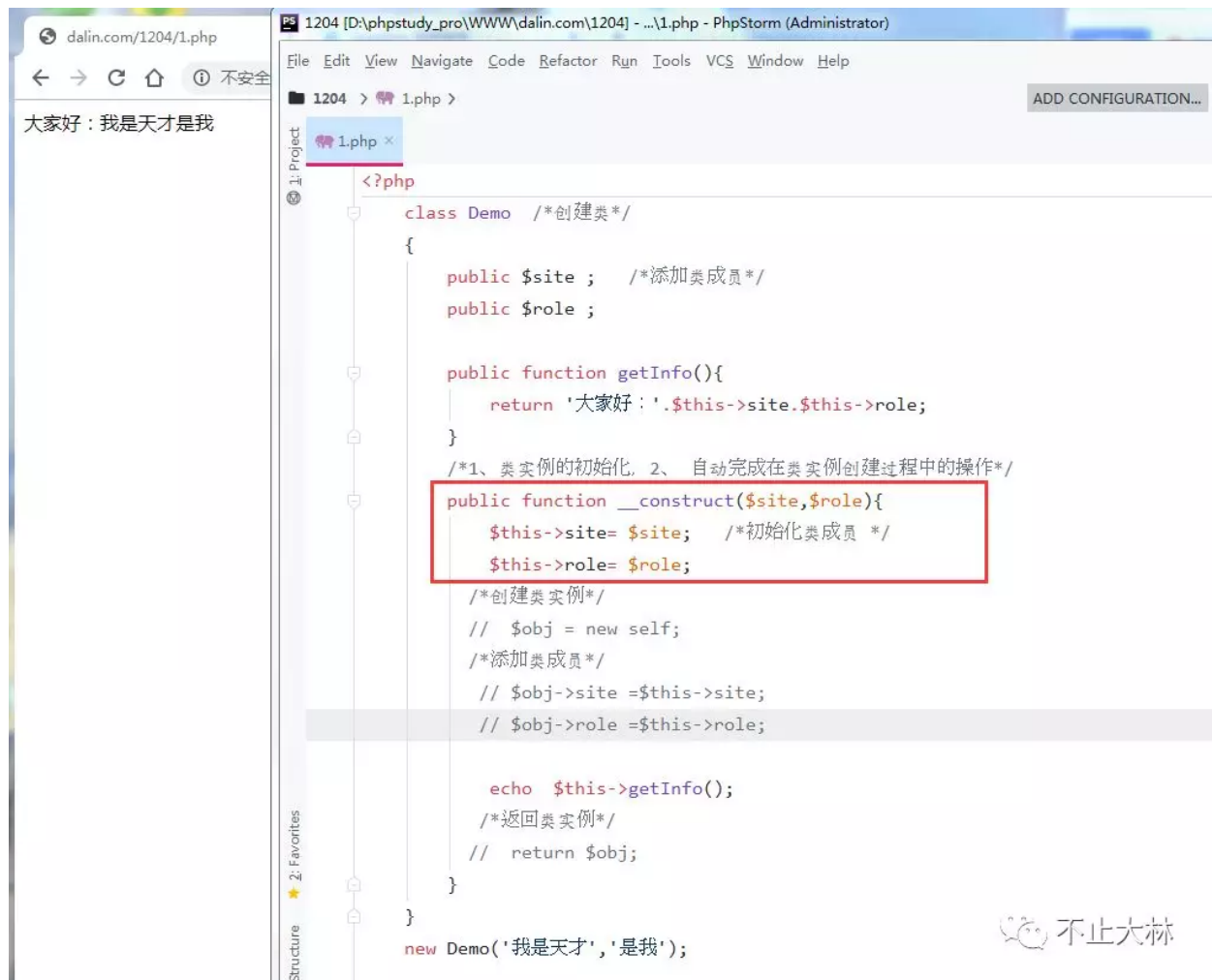


`$this` 是当前类的实例的引用，他始终与当前类的实例绑定，用了`$this`就不用再实例化`$obj=new XXX`了。



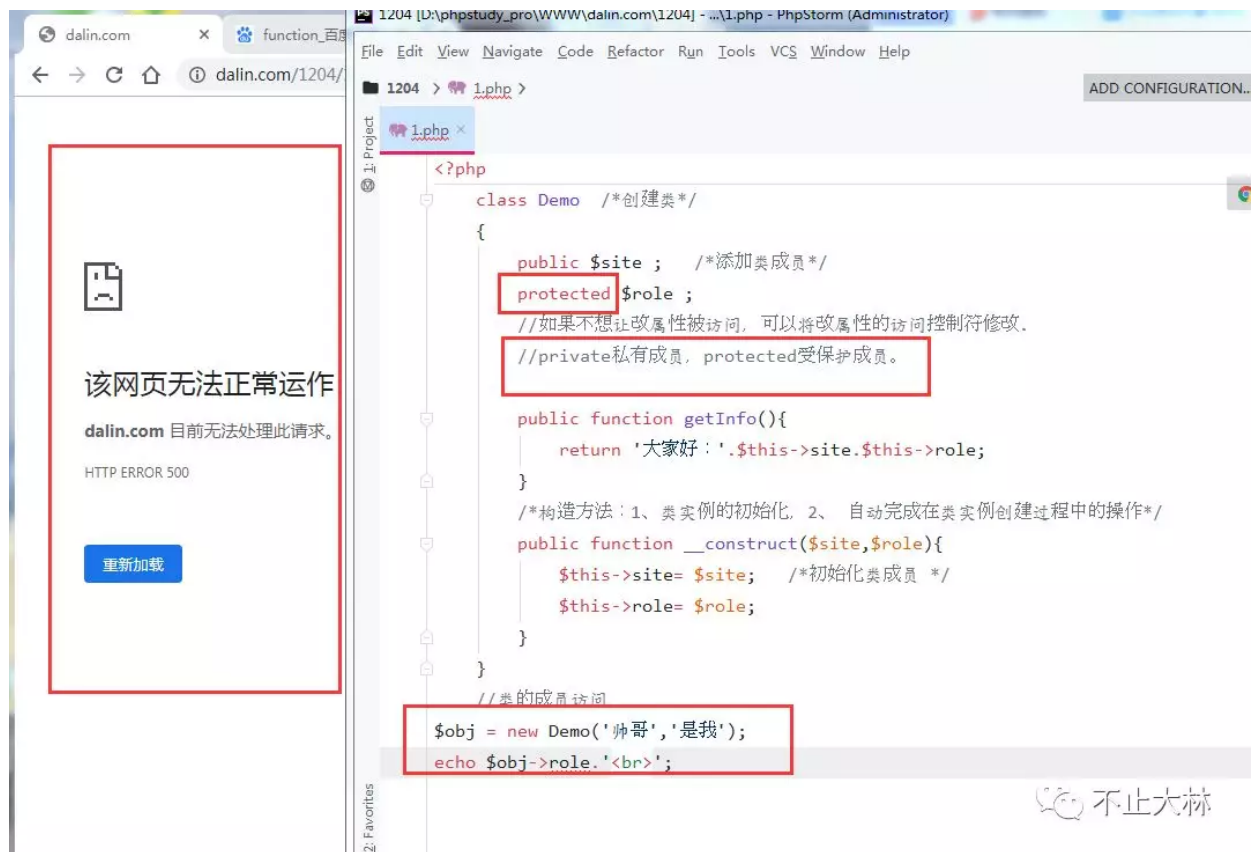
构造方法（魔术方法）预定义方法。一个方法一旦前面加了__下划线，这个方法就由系统根据某种方法，某种构造自动调用。

下图`echo -> $this->getInfo()`类实例化时自动会被执行。

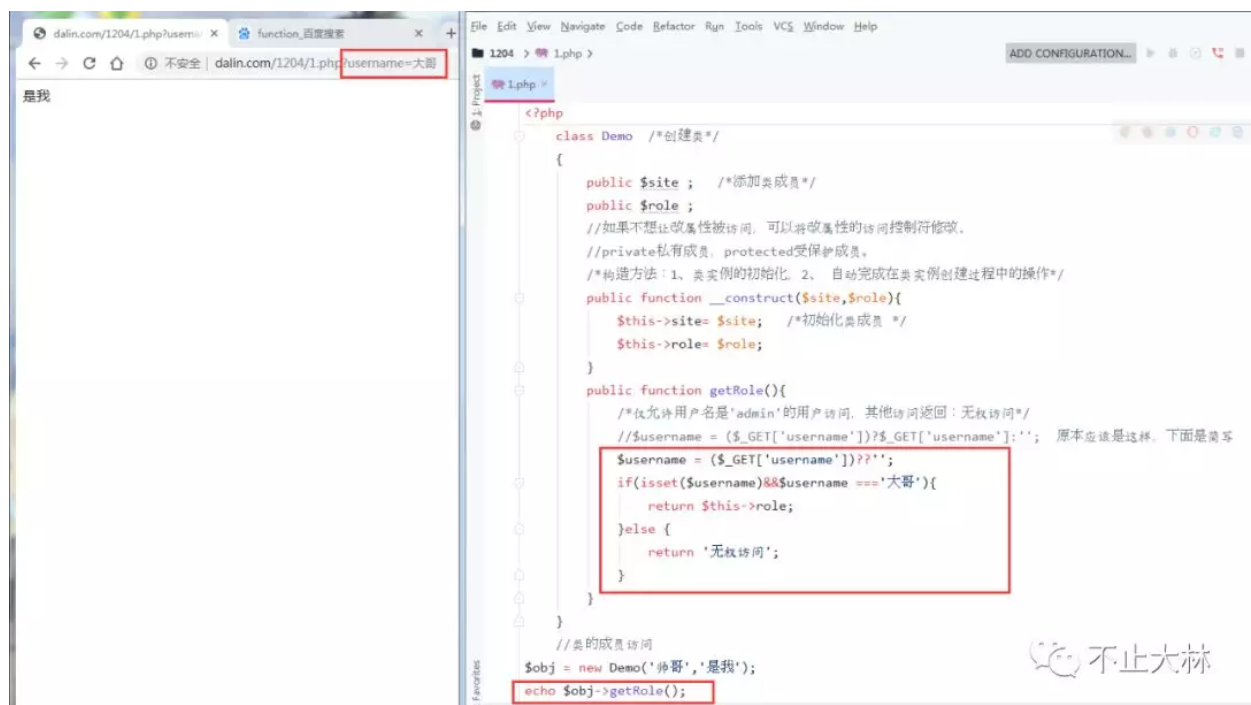


不止大林

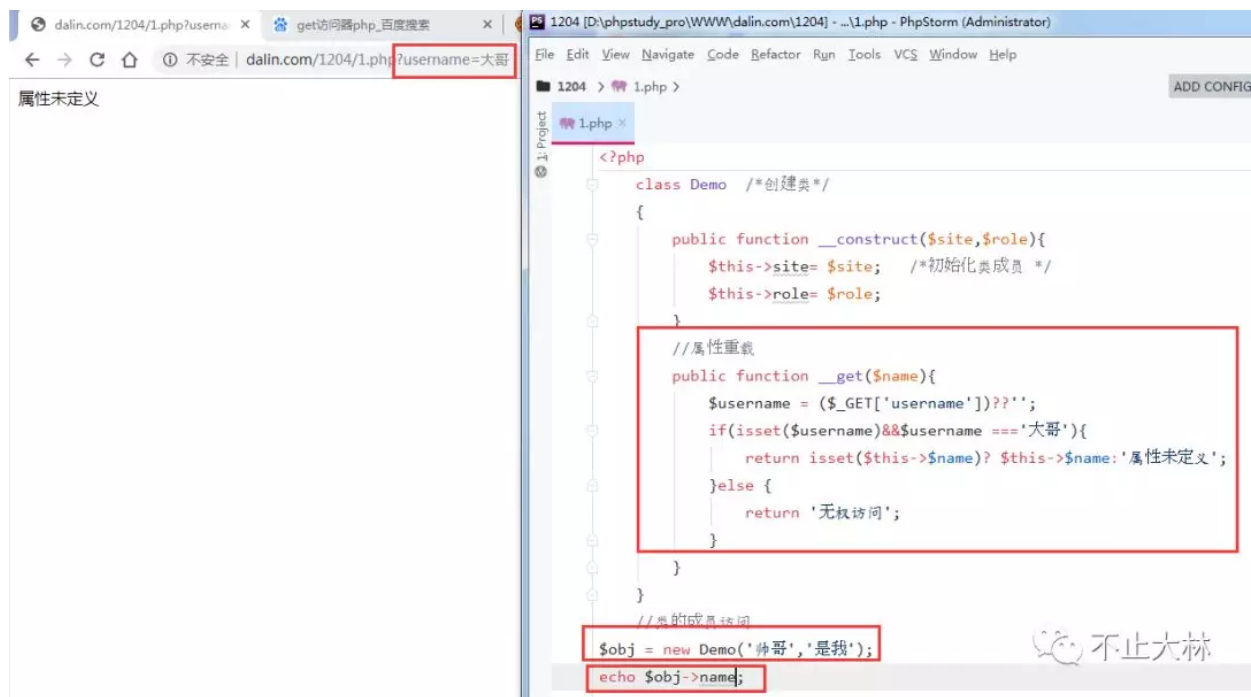
限制类成员访问 实现类的封装，访问控制
`private`私有成员，`protected`受保护成员。



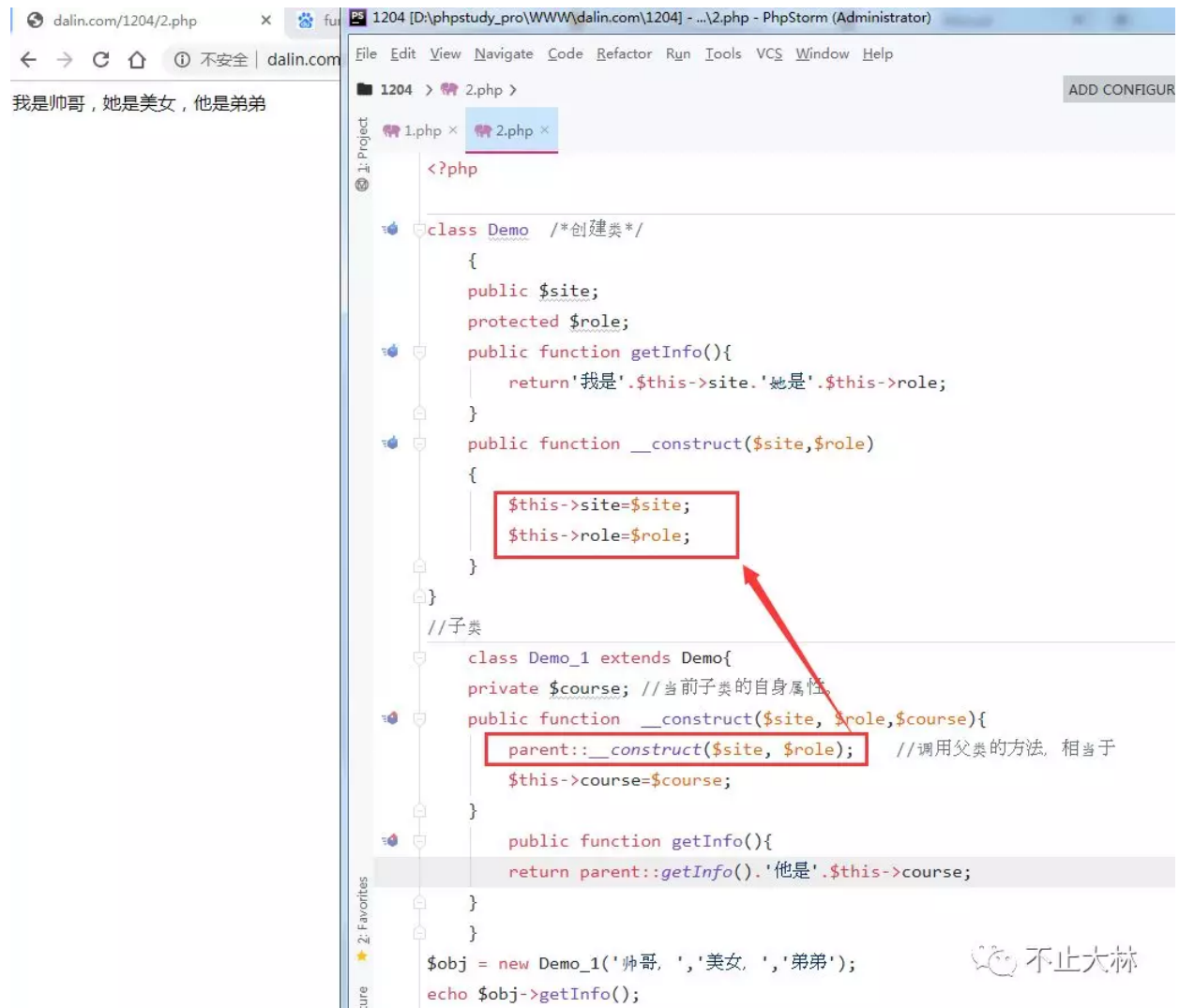
外部无权访问, 是为了防止非法访问, 并不代表禁止访问。



__get(变量) 访问一个无权访问的属性都会被调用, 属性重载



类的继承 子类继承父类，代码复用。

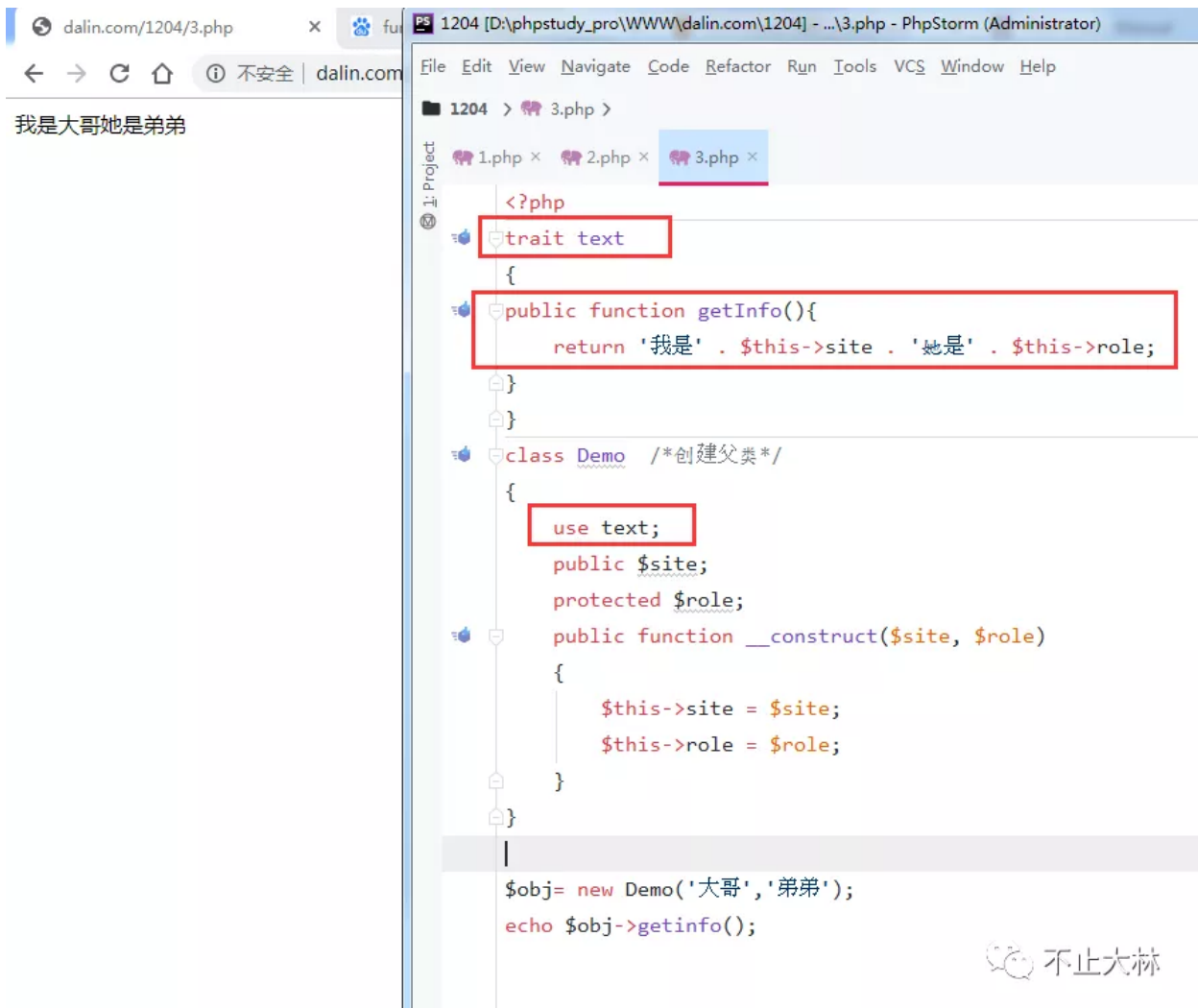


Trait:1、代码复用的方式，用来扩展当前类的功能

2、当成一个公共方法库，

3、使用了类的定义的语法，但不是类，所以不能实例化。

在父类中用use 导入父类，即可使用



优先级，当前类中的同名方法>trait类中的同名方法>父类中的同名方法

The screenshot shows a web browser on the left and a code editor on the right. The browser displays the output of the PHP code: "我是大哥她是弟弟" and "我是帅哥她是美女". The code editor shows the PHP code for a trait and two classes. The code is as follows:

```
<?php
trait text
{
    public function getInfo(){
        return '我是' . $this->site . '她是' . $this->role;
    }
}

class Demo /*创建父类*/{...}

class Demo_1 extends Demo //子类
{
    private $course; //当前子类的自身属性。

    public function __construct($site, $role, $course)
    {
        parent::__construct($site, $role); //调用父类的方法。相当于
        $this->course = $course;
    }

    /* public function getInfo()
    {
        return parent::getInfo() . '他是' . $this->course;
    } */
}

$obj= new Demo('大哥','弟弟');
echo $obj->getinfo();
echo '<hr>';
$sub= new Demo_1(帅哥,美女,弟弟);
echo $sub->getInfo();
```

接口：对象的模板是类，类的模板就是接口

面向接口编程是最重要的思想之一，很多高级应用都严重依附于它。

接口是一种约定，定义了实现他们的类中必须实现的方法。

接口中没有方法的具体实现，所以不能实例化。

用interface创建类接口，

我是帅哥她是美女

大家好

```
1204 > 4.php >
1.php x 2.php x 4.php x 3.php x
<?php
interface iDemo{
    public function getInfo();
    public function hello();
}

class Demo implements iDemo
{
    public $site;
    protected $role;
    public function getInfo()
    {
        return '我是' . $this->site . '她是' . $this->role;
    }
    public function __construct($site, $role)
    {
        $this->site = $site;
        $this->role = $role;
    }
    public function hello()
    {
        return '大家好';
    }
}

$obj=new Demo(帅哥,美女,);
echo $obj->getInfo().'\n';
echo $obj->hello().'\n';
```

不止大赫

抽象类 给其他类当父类

接口：全部都是抽象方法。

抽象类：抽象类中有有抽象方法，也有已实现的方法。

共同之处：统统不能实例化，原因就是内部有抽象方法。

dalin.com/1204/4.php

← → ↺ ⌂ ① 不安全 dalin.com

我是帅哥她是美女

大家好

File Edit View Navigate Code Refactor Run Tools VCS Window Help

1204 > 4.php > ADD CON

1.php × 2.php × 4.php × 3.php ×

```
<?php
abstract class chouxianlei{
    abstract public function getInfo(); //未实现的, 抽象方法
    public function hello()//已经实现的方法
    {
        return '大家好';
    }
}

class Demo extends chouxianlei
{
    public $site;
    protected $role;
    public function getInfo()
    {
        return '我是' . $this->site . '她是' . $this->role;
    }
    public function __construct($site, $role)
    {
        $this->site = $site;
        $this->role = $role;
    }
}

$obj=new Demo(帅哥,美女,);
echo $obj->getInfo().'\n';
echo $obj->hello().'\n';
```

2: Favorites

不止大林