PHP面向对象架构

目录

[一、阐述 1](#_Toc484677156)

[1.概念 1](#_Toc484677157)

[2.结构 2](#_Toc484677158)

[3.面向对象三大重要特性之封装 4](#_Toc484677159)

[4.魔术方法 4](#_Toc484677160)

[5.面向对象三大重要特性之继承 5](#_Toc484677161)

[6.重载方法 5](#_Toc484677162)

[7.访问类型 6](#_Toc484677163)

[8.final关键字 6](#_Toc484677164)

[9.static和const关键字 7](#_Toc484677165)

[10.\_\_toString() 7](#_Toc484677166)

[11.\_\_clone() 8](#_Toc484677167)

[12.\_\_call() 8](#_Toc484677168)

[13.抽象方法和抽象类 9](#_Toc484677169)

[14.interface 10](#_Toc484677170)

[15.面向对象三大特性之多态 11](#_Toc484677171)

一、阐述

1.概念

面向对象（oop）：

面向对象编程是一种计算机编程架构

类（class）：

具有相同属性和方法的一组对象的集合

对象（object）：

类的实例化，具有类的所有属性和方法

属性（property）：

类的特性，变量

方法（function）：

类的行为，函数

2.结构

实例化（new）

使用对象成员（->）

特殊引用（$this）：

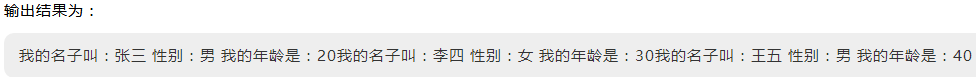
对象对本身的引用

构造方法（\_\_construct（））：

当创建一个对象时自动调用构造函数

为了向下兼容，如果一个类中没有名为\_\_construct()的方法，PHP将搜索一个php4中的写法，与类名相同名的构造方法





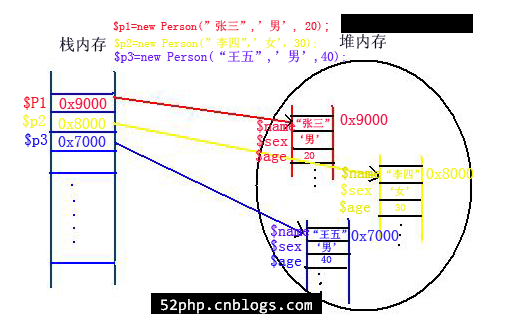
析构方法（\_\_destruct）：

析构函数会在到某个对象的所有引用都被删除或者当对象被显式销毁时执行，也就是对象在内存中被销毁前调用析构函数

由于类实例是以堆栈的形式放在内存中，所以最后调用 析构函数 的时候，输出顺序是按 后进先出 的原则！

类和对象的存储方式：

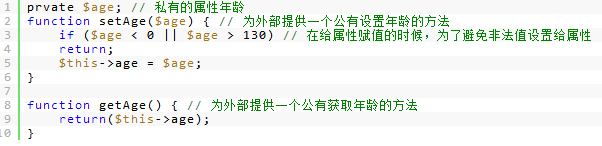
对象是放在堆里面的，但对象名称是放在栈里面



3.面向对象三大重要特性之封装

关键字：private

对象的private属性和方法不能被外部直接访问，需要通过给定的接口来实现调用



4.魔术方法

[\_\_set()，\_\_get()，\_\_isset()，\_\_unset()四个方法的应用](http://www.cnblogs.com/phpgo/p/5658069.html)

对象的公开属性可以直接get、set、isset、unset，私有属性需要设置\_\_get()、\_\_set()、\_\_isset()、\_\_unset()方法

5.面向对象三大重要特性之继承

extends

子类可以继承父类的属性和方法

C++可以单继承也可以多继承

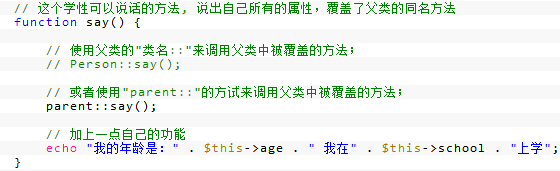
PHP和Java只能单继承

6.重载方法

由于PHP是弱类型语言，同名方法会被覆盖

子类继承父类时可以重载方法

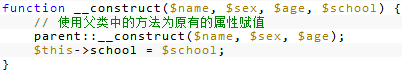
还可以扩展父类方法：



同样，构造方法在子类中如果没有声明的话，也可以使用父类中的构造方法

如果子类中重新定义了一个构造方法也会覆盖掉父类中的构造方法

如果想使用新的构造方法为所有属性赋值也可以用同样的方式



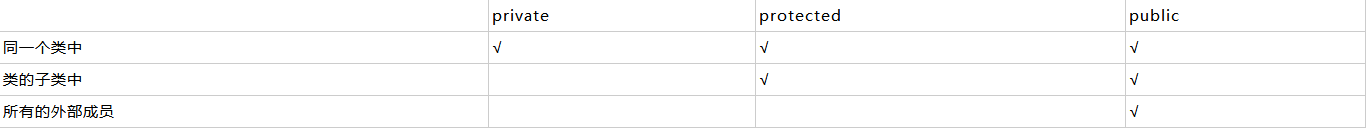
7.访问类型

public（公有的）、protected（受保护的）、private（私有的）

public 公有修饰符，类中的成员将没有访问限制，所有的外部成员都可以访问这个类成员

protected保护成员修饰符，被修饰为protected的成员不能被该类的外部代码访问。但是对于该类的子类有访问权限，可以进行属性、方法的读及写操作，该子类的外部代码不具有访问其属性和方法的权限

private 私有修改符，被定义为private的成员，对于同一个类里的所有成员是可见的，即没有访问限制；但对于该类的外部代码是不允许改变甚至读操作，对于该类的子类，也不能访问private修饰的成员



另外在子类覆盖父类的方法时也要注意一点，**子类中方法的访问权限一定不能低于父类被覆盖方法的访问权限，也就是一定要高于或等于父类方法的访问权限**

8.final关键字

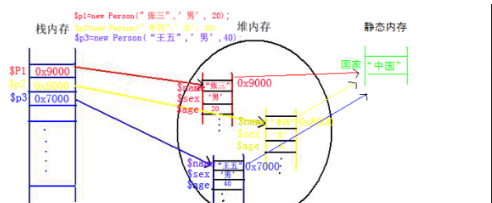
final只能用来定义类和方法，final定义的类不能被子类继承，final定义的方法不能被子类的方法覆盖

9.static和const关键字

static关键字在类中描述属性和方法是静态的

static成员属于类，不属于任何对象实例

**在类第一次被加载的时候分配的空间，其他类是无法访问的，只对类的实例共享**



**使用对象不能访问到静态成员**

类的静态方法只能访问类的静态成员，不能访问非静态成员，使用self，即“self::成员属性”的方式

非静态方法可以访问到静态成员

const类似于static，定义常量

**用“const”修饰的成员属性的访问方式和“static”修饰的成员访问的方式差不多，也是使用“类名”，在方法里面使用“self”关键字。但是不用使用“$”符号，也不能使用对象来访问**

10.\_\_toString()

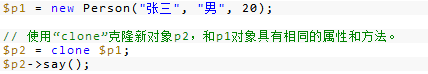
echo 对象会报错

在类中添加\_\_toString()方法，当echo对象时会输出该方法返回值

11.\_\_clone()

有的时候我们需要在一个项目里面，使用两个或多个一样的对象

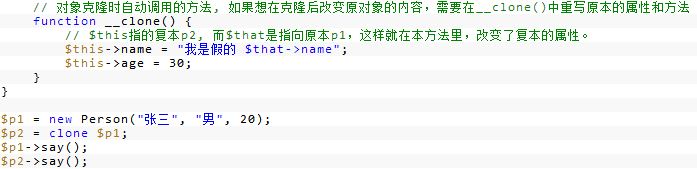
我们使用“clone”这个关键字克隆对象



\_\_clone()是在对象克隆时自动调用的方法

如果想在克隆后改变原对象的内容，需要在\_\_clone()中重写原本的属性和方法

\_\_clone()方法可以没有参数，它自动包含$this和$that两个指针，**$this指向复本，而$that指向原本**

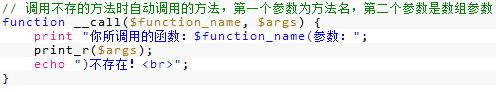


12.\_\_call()

在程序开发中，如果在使用对象调用对象内部方法时候，调用的这个方法不存在那么程序就会出错，然后程序退出不能继续执行

这个时候我们就要使用在调用不存在的方法时自动调用的方法\_\_call()

自动调用\_\_call()方法时，把这个不存在的方法的方法名传给第一个参数，**第二个参数则是把这个方法的多个参数以数组的形式传进来**



13.抽象方法和抽象类

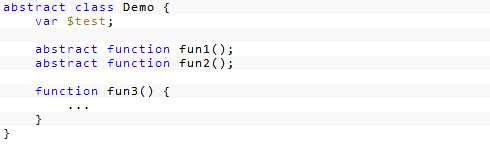
我们在类里面定义的没有方法体的方法就是**抽象方法**

所谓的没有方法体指的是，在方法声明的时候没有大括号以及其中的内容，而是直接在声明时在方法名后加上分号结束，另外在声明抽象方法时还要加一个关键字“**abstract**”来修饰



**只要一个类里面有一个方法是抽象方法，那么这个类就要定义为抽象类**，抽象类也要使用“abstract”关键字来修饰

在抽象类里面可以有不是抽象的方法和成员属性，但只要有一个方法是抽象的方法，这个类就必须声明为抽象类



**抽象类不能产生实例对象**， 所以也不能直接使用

**子类必须把父类中的抽象方法全部都实现，否则子类中还存在抽象方法，那么子类还是抽象类，还是不能实例化类**

14.interface

PHP不支持多重继承，也就是说每个类只能继承一个父类，为了解决这个问题，PHP引入了接口

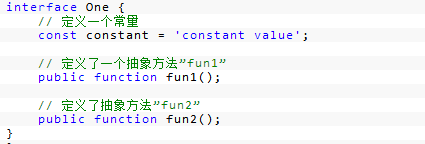
**接口是一种特殊的抽象类**

接口里面所有的方法必须都是声明为抽象方法

接口里面不能声明变量(但可声明常量constant)

接口里面所有的成员都是public权限的

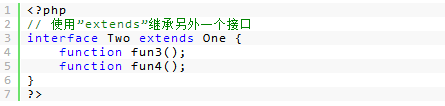
用interface声明一个接口



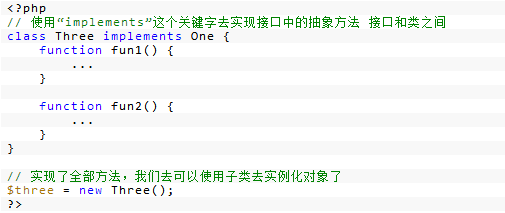
因为接口里面所有的方法都是抽象方法，所以在声明抽象方法的时候就不用像抽象类那样使用abstract这个关键字了，默认的已经加上这个关键字

另外在接口里边的public这个访问权限也可以去掉，因为默认就是public的

我们可以使用**extends**关键字让一个接口去继承另一个接口

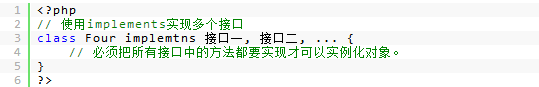


而我们定义一接口的子类去实现接口中全部抽象方法使用的关键字是**implements**

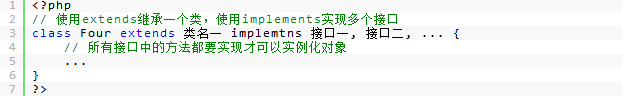


我们也可以使用抽象类，去实现接口中的部分抽象方法，但要想实例化对象，这个抽象类还要有子类把它所有的抽象方法都实现才行

PHP是单继承的，一个类只能有一父类，但是一个类可以实现多个接口，就相当于一个类要遵守多个规范



**PHP中不仅一个类可以实现多个接口，也可以在继承一个类的同时实现多个接口, 一定要先继承类再去实现接口**



15.面向对象三大特性之多态

**多态性**是指一段程序能够处理多种类型对象的能力

对于面向对象的程序来说，**多态就是把子类对象赋值给父类引用，然后调用父类的方法，去执行子类覆盖父类的那个方法**



同一个父类，子类实例调用父类的方法，执行的是各自覆盖的同名方法，返回的值不同

16.串行化serialize(),\_\_sleep(),\_\_wakeup()方法

有两种情况我们必须把对象串行化

**第一种情况**就是把一个对象在网络中传输的时候要将对象串行化

**第二种情况**就是把对象写入文件或是数据库的时候用到串行化

串行化有两个过程，一个是**串行化**，就是把对象转化为二进制的字符串，我们使用**serialize()**函数来串行化一个对象

另一个是**反串行化**，就是把对象转化的二进制字符串再转化为对象， 我们使用**unserialize()**函数来反串行化一个对象

在对象串行化的时候，会调用一个\_\_sleep()方法来完成一些睡前的事情；而在重新醒来，即由二进制串重新组成一个对象的时候，则会自动调用PHP的另一个函数\_\_wakeup()，做一些对象醒来就要做的动作

\_\_sleep()函数不接受任何参数， 但返回一个数组，其中包含需要串行化的属性

末被包含的属性将在串行化时被忽略，如果没有\_\_sleep()方法，PHP将保存所有属性



17.自动加载类\_\_autoload()函数

当类不存在时\_\_autoload()自动传入类名，以便加载文件

