Atitit usbQb212 oo 面向对象封装的标准化与规范解决方案java c# php js

[1.1. 封装性是面象对象编程中的三大特性之一  三个基本的特性:封装、继承与多态 1](#_Toc15002)

[1.2. 魔术方法 预定义方法 2](#_Toc30075)

[1.3. 魔术常量(Magic constants) 预定义变量 2](#_Toc6876)

[1.4. 适用领域 js php java c# 2](#_Toc25184)

[1.5. 构造函数 \_\_construct( ) vs xxxCls() 3](#_Toc30669)

[1.6. 析构函数名称必须是\_\_destruct( ) finally() 3](#_Toc31437)

[1.7. 私有属性 与工友属性 3](#_Toc21092)

[1.8. $this指针 3](#_Toc27271)

[1.9. \_\_autoload() 类自动加载 3](#_Toc17243)

[1.10. \_\_clone 4](#_Toc7004)

[1.11. serialize() 检查类中是否有魔术名称 \_\_sleep 的函数。 4](#_Toc22867)

[1.12. \_\_invoke 5](#_Toc17711)

[1.13. \_\_callStatic 5](#_Toc26950)

## 封装性是面象对象编程中的三大特性之一  三个基本的特性:封装、继承与多态

，封装性就是把对象的属性和服务结合成一个独立的相同单位，并尽可能隐蔽对象的内部细节，包含两个含义：

1. 把对象的全部属性和全部服务结合在一起，形成一个不可分割的独立单位（即对象）。

2. 信息隐蔽，即尽可能隐蔽对象的内部细节，对外形成一个边界〔或者说形成一道屏障〕，只保留有限的对外接口使之与外部发生联系。

## 魔术方法 预定义方法

实现魔术方法的机制，单根继承。。。

如果已经有了，那么适用多重继承机制实现。。

Super\_magic

## 魔术常量(Magic constants) 预定义变量

PHP中的常量大部分都是不变的，但是有8个常量会随着他们所在代码位置的变化而变化，这8个常量被称为魔术常量。

* \_\_LINE\_\_，文件中的当前行号
* \_\_FILE\_\_，文件的完整路径和文件名
* \_\_DIR\_\_，文件所在的目录
* \_\_FUNCTION\_\_，函数名称
* \_\_CLASS\_\_，类的名称
* \_\_TRAIT\_\_，Trait的名字
* \_\_METHOD\_\_，类的方法名
* \_\_NAMESPACE\_\_，当前命名空间的名称

这些魔术常量常常被用于获得当前环境信息或者记录日志。

## 适用领域 js php java c#

[面向对象  封装的实现原理](http://blog.csdn.net/attilax/article/details/53150034)

Js的面向对象

## 构造函数 \_\_construct( ) vs xxxCls()

\_\_construct( )。这是PHP5中的变化，以前的版本中，构造函数的名称必须与类名相同，这种在PHP5中仍然可以用，但现在以经很少有人用了，这样做的好处是可以使构 造函数独立于类名，当类名发生改变时不需要改相应的构造函数名称了。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。

格式

## 析构函数名称必须是\_\_destruct( ) finally()

## 私有属性 与工友属性

## ****$this指针****

**$this就是对象内部代表这个对象的引用**，在对象内部和调用本对象的成员和对象外部调用对象的成员所使用的方式是一样的。

## \_\_autoload() 类自动加载

很多开发者写面向对象的应用程序时，对每个类的定义建立一个 PHP 源文件。一个很大的烦恼是不得不在每个脚本（每个类一个文件）开头写一个长长的包含文件的列表。

在软件开发的系统中，不可能把所有的类都写在一个PHP文件中，当在一个PHP文件中需要调用另一个文件中声明的类时，就需要通过include把 这个文件引入。不过有的时候，在文件众多的项目中，要一一将所需类的文件都include进来，是一个很让人头疼的事，所以我们能不能在用到什么类的时 候，再把这个类所在的php文件导入呢？这就是我们这里我们要讲的**自动加载类**。

 function \_\_autoload($classname)

{

    require\_once $classname . '.php';

}

//MyClass1类不存在时，自动调用\_\_autoload()函数，传入参数”MyClass1”

$obj = new MyClass1();

## ****\_\_clone****

## serialize() 检查类中是否有魔术名称 \_\_sleep 的函数。

如果这样，该函数将在任何序列化之前运行。它可以清除对象并应该返回一个包含有该对象中应被序列化的所有变量名的数组。

使用 \_\_sleep 的目的是关闭对象可能具有的任何数据库连接，提交等待中的数据或进行类似的清除任务。此外，如果有非常大的对象而并不需要完全储存下来时此函数也很有用。

相反地，unserialize() 检查具有魔术名称 \_\_wakeup 的函数的存在。如果存在，此函数可以重建对象可能具有的任何资源。使用 \_\_wakeup 的目的是重建在序列化中可能丢失的任何数据库连接以及处理其它重新初始化的任务。

## ****\_\_invoke****

当尝试以调用函数的方式调用一个对象时，\_\_invoke 方法会被自动调用。PHP5.3.0以上版本有效

## ****\_\_callStatic****

它的工作方式类似于 \_\_call() 魔术方法，\_\_callStatic() 是为了处理静态方法调用，PHP5.3.0以上版本有效。

PHP 确实加强了对 \_\_callStatic() 方法的定义；它必须是公共的，并且必须被声明为静态的。同样，\_\_call() 魔术方法必须被定义为公共的，所有其他魔术方法都必须如此