# 面向对象核心思想

编程方式:命令式编程和函数式编程。

面向对象和面向过程都是属于命令式编程。

PHP对象是由数组模拟的，对象序列化后，存储的只是对象的属性。类是由属性和方法组成的，而对象则是属性的集合，一个类生成的对象拥有各自不同的属性，但共享了类的代码空间中方法区域的代码。

对象携带了对象的属性和类的入口。就是对象中存在一个指针指向一个类.

序列化后的对象会附带所属的类名，这个类名保证此对象能够在执行类的方法时，可以正确找到方法所在的的代码空间。

由标量进行强制类型转换的object，没有一个类和它对应，此时stdClass类就会收留这个对象。

语法糖:计算机语言中添加的某种语法，这种语法对语言的功能并没有影响，但是更方便程序员使用。如:在C语言里用a[i]表示\*(a+i),用a[i][j]表示\*(\*(a+i\*数组a第二维的长度)+j)，看来语法糖不是“现代语言”才有啊，连我们的老大哥C也有，而且这种写法简洁明了，也更好懂了。

继承和多态都是对类进行复用。

耦合是一个软件结构内不同模块之间互连程度的度量，也就是不同模块之间的依赖关系。

PHP中的接口存在两个不足，一是没有契约限制，二是缺少足够多的内部接口。

函数是编程的特征如:闭包，回调。

命令模式也是SRP的体现，命令模式分离‘命令请求者’和‘命令的实现者’命令请求和实现完成解耦。

对象在PHP中是使用一种zend\_object\_value 结构体来存储的

**魔术方法**:\_\_construct \_\_destruct() \_\_set \_\_get \_\_call(‘调用的方法名’,’含有参数的数组’)

\_\_call当要调用的方法不存在或权限不足时，会自动调用\_\_call 方法。

\_\_callStatic 当调用的静态方法不存在或权限不足时，会自动调用\_\_callStatic方法。

get\_called\_class 获取调用该静态方法的类名\_\_toString() 打印一个对象时调用,也是一种序列化

解耦:解除模块与模块之间的依赖.

低耦合:模块与模块之间尽可能的使模块间独立存在.

继承破坏了封装性.继承是紧耦合的.继承扩展复杂

子类扩展不是覆盖或者使父类的功能失效

Php5.4 trait

权限和角色的问题

Php5 iterator 迭代器

**反射: hook 代理扩展模块异常**

get\_object\_vars

get\_class\_vars

get\_class

new reflectionClass new reflectionObject

PHP只有手动抛出异常后才能不活

# 面向对象的设计原则

**面向对象设计的五大原则**：

1. 单一职责原则(SRP)：

含义:

(1).避免相同的职责分散到不同的类中。

(2).避免一个类承担太多职责。

优点：

1. 可以减少类之间的耦合。
2. 提高类的复用性。

工厂模式:

命令模式:分离命令的请求者和命令的执行者.

代理模式.

1. 接口隔离原则(ISP)(避免胖接口和污染接口)：在基本接口高内聚的前提下细分接口减少耦合

客户端需要什么接口就提供什么接口

含义:

1. 定制化服务设计原则。
2. 接口里的方法应该是通用的，基本的，不易变化的。

思考对于接口的污染处理方式：

1. 利用委托分离接口。
2. 利用多继承分离接口。PHP中类只能单继承接口可以多继承interface c extends a,b {}

委托模式策略模式

1. 开放-封闭原则（OCP）：

含义：

1. Open: 模块行为必须是开放的，支持扩展的，而不是僵化的。
2. Closed: 在对模块的功能进行扩展时，不应该影响或大规模地影响已有的程序模块。

优点:

1. 提高系统的可扩展性和可维护性。

一个模块在扩展性上是开放的修改性是封闭的

1. 替换原则（LSP）:

含义:

1. 子类型必须能够替换掉它们的父类型，并出现在父类能够出现的任何地方。
2. 父类的方法都要在子类中实现或者重写，并且派生类只实现其抽象类中声明的方法，而不应当给出多余的方法定义或实现。
3. **在客户端程序中只应该使用父类对象而不应当直接使用子类对象，这样可以实现运行期绑定(动态多态)**
4. 依赖倒置原则(DIP):

含义:

1. 上层模块不应该依赖于下层模块，它们共同依赖于一个抽象(父类不能依赖子类,它们都要依赖抽象类)。
2. 抽象不能依赖于具体，具体应该要依赖于抽象。
3. 核心原则是解耦。

实现：

1. **每个较高层次类都为它所需要的服务提出一个接口声明，较低层次类实现这个接口。**
2. **每个高层类都通过该抽象接口使用服务。**

# 正则表达式基础与应用

正则表达式: 分隔符表达式修饰符.

分隔符:除了数字字母下滑线反斜线空白符以外的任何字符.

火狐插件 regular expression

元字符:在正则表达式中具有特殊意义的专用字符.

起止元字符: ^ $ .

点号 .

量词: \* + ? {n} {n,} {n,m}

字符组 [auo] 匹配单个字符.字符中的元字符不用转义自动忽略

转义 \ \Q 忽略这里的元字符\E

反义 [^a]

分支因为

分组(?:) 不捕获不计入模式分组计数 (?<name>) 以name 保存匹配的变量

环视(断言): 顺序肯定环视逆序肯定环视

贪婪懒惰匹配模式贪婪模式匹配最长

优先级: 替换`|` 优先级别是最低的所以 ooo|aaa 匹配的是 ooo 或者 aaa 因为定点和序列优先级高于替换符

常用模式 i 忽略大小写 m 多行模式s 点号通配符匹配换行符. U懒惰模式

URL重写分流?

# 第四章PHP网络技术及应用

Session cookie 网络协议分析 socket

HTTP协议:是一个基于应用层的通信规范,是一种无状态协议,通常承载于TCP协议之上,有时也承载于TLS或SSL协议之上,这时候就成了常说的HTTPS.HTTP默认端口号为80 HTTPS 默认端口号为443.典型的问答式交互,服务器端不会主动向客户端PUSH,一问一答的轮询也会使TCP链接频繁建立和断开,导致其交互效率不高.

SPDY协议:是google推出的,优化了浏览器和服务器之间的通信,支持流复用,具备优先级的请求,主动发起请求,强制SSL安全传输等先进的特性.

腾讯QQ和迅雷都使用HTTP协议.

格式:状态行 MIME信息

HTTP工作流程建立连接->客户端发送请求->服务器给予相应信息->客户端接收断开

格式:统一资源标示符协议版本号 MIME信息

**1.连接:**是一个传输层环流

**2.请求**:HTTP1.1协议中 Client和Server 默认对方支持长连接

Connection:false 关闭长连接.若client 和 server 有一方是false tcp请求处理完毕后都会断掉

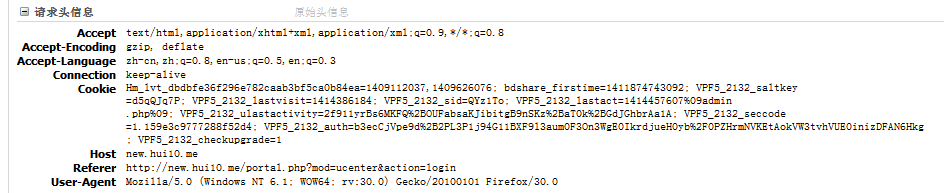
Connection:Keep-Alive长连接.

请求行:Method Request-URI HTTP-Version CRLF(回车和换行)

Method 全为大写:

GET POST HEAD PUT DELETE TRACE CONNECT OPTIONS 108页

消息报头:



请求头和请求正文之间是一个空行,这个空行表示请求头的结束.

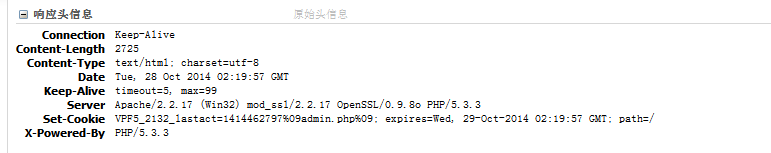
请求正文:客户提交的查询字串信息action=setting&operation=basic

**3.响应:**

状态行: HTTP-VERSION STATUS–Code Reason–Phrase(状态码的文本描述) CRLF

状态码 1xx 请求已接收继续处理 2xx 成功接收理解 3xx 重定向 4xx 客户端错误 5xx 服务器端错误

消息头:



响应正文:

消息报头: 普通报头,请求报头,响应报头,实体报头

名字: 值名字不区分大小写

http1.1 必须包含主机头域否则返回400状态码

Host: User-Agent: Accept:可以接受的文本格式 Cookie: Cache-Control: 指定请求和相应遵循的缓存机制.content-Length:内容长度.content-Range:相应的资源范围.迅雷的断点续传.Accept-Encoding:接受的编码格式

PHP中HTTP有关的函数:

get\_headers 取得服务器相应一个请求所发送的所有标头

file 系列函数 header 函数 cURL函数 socket 函数 stream\_ 函数

用file\_get\_contents 打开文件会创建一个$http\_response\_header 变量保存响应的报头

http\_build\_query

垃圾信息防御措施:IP限制验证码 Token和表单欺骗审核机制

IP限制:

因为HTTP协议是透明的所以判断Referer手段是于事无补的,不少代码会判断HTTP头中的HTTP\_X\_FORWARDED\_FOR是否代理过来若是则把其记为IP 但是HTTP\_X\_FORWARDED\_FOR 来自于HTTP请求中的x-forwarded-fo报头这个可以修改造成潜在攻击合理的判断是完全不考虑代理,而使用SERVER变量中的HTTP\_CLIENT\_IP或REMOTE\_ADDR凡是通过代理来的拒绝方案.

TOKEN 法:

define('TOKEN\_KEY','123abc');

function m\_token(){

$str = mt\_rand(1000,9999);

$str2 = dechex($\_SERVER['REQUEST\_TIME']-$str); //十进制数转换成十六进制REQUEST\_TIME 格林尼治时间戳

$token = $str.substr(md5($str.TOKEN\_KEY),0,10).$str2;

return $token;

}

function v\_token($token,$delay=300){

$rs = substr($token,0,4);

$middle = substr($token,0,14);

$rs2 = substr($str,14,8);

return ($middle == $rs.substr(md5($rs.TOKEN\_KEY),0,10) && ($\_SERVER['REQUEST\_TIME']-hexdec($rs2)-$rs<$delay));

}

**Socket 进程通信机制及应用**

Socket 通常称为“套接字”用于描述IP地址和端口,是一个通信链的句柄.应用程序通过套接字向网络发出请求或者应答网络请求,既不是一个程序也不是协议,这是操作系统提供的通信层的一个API

为解决网间进程标识问题,TCP/IP引入了如下概念

**端口:**网络中可以被命名和寻址的通信端口是操作系统可分配的一种资源.

传输层与网络层在功能上最大的区别是传输层提供进程通信能力.网络通信的最终地址就不仅仅是主机地址了,还包括可以描述进程的某种标识.

应用程序(即进程)通过系统调用与某端口建立连接,传输层传给端口的数据都被相应进程所接受,操作端口类似于一般的io操作进程获取一个端口,相当于获取本地唯一I/O文件,端口号类似于文件的描述符.65536个端口号

端口号小于256的定义为常用端口.任何TCP/IP实现所提供的服务都用1-1023之间的端口号由IANA管理.大多数TCP/IP实现给临时端口号分配1024-5000之间的端口,大于5000的端口号为其他服务器预留

**地址:**

某台主机可与多个网络链接必须指定一个特定网络地址.

网络上每台主机应有其唯一的地址.

每台主机上的每个进程应有在该主机上的唯一标识符

**链接:**两个进程间的通信链路

**Curl 应用**

建立cURL请求的基本步骤

1. 初始化。
2. 设置选项。
3. 执行并获取HTML文档内容。
4. 释放cUrl句柄。

$ch = curl\_init();

curl\_setopt($ch,CURLOPT\_URL,’http:/blog.fatephp.com’);

curl\_setopt($ch,CURLOPT\_RETURNTRANSFER,1);

curl\_setopt($ch,CURLOPT\_HEADER,1);

$output = curl\_exec($ch);

curl\_close($ch);

curl\_getinfo(); 返回执行后这一请求后相关的信息。

curl\_error();

抓取数据时，由于网络原因，时常出现抓取数据不完整的情况。需要一个校验。通过对所获取的数据计算filesize 然后和curl\_getinfo获取的数据进行对比。

一般手机网站会判断手机UA，所以可以使用curl 修改header里的UA头访问手机站。用这种方法“欺骗“移动 联通 电信的网站是不行的。使用真实的手机访问网站时 所有数据都是被移动网管代理的，会带上一些特殊的头，如手机号丶手机特征码丶手机所在基站等。

**Curl上传文件**

$url = <http://localhost/upload_output.php>;

$post\_data = array(“foo”=>’bar’,’upload’=>”@test.zip”);

$ch = curl\_init();

Curl\_setopt($ch,CURLOPT\_URL,$url);

Curl\_setopt($ch,CURLOPT\_RETURNTRANSFER,1);

Curl\_setopt($ch,CURLOPT\_POST,1);

Curl\_setopt($ch,CURLOPT\_POSTFIELDS,$post\_data);

$output = curl\_exec($ch);

Curl\_close($ch);

**Curl 批处理**

define(URL1,'http://blog.fatephp.me');

define(URL2,'http://blog.fatephp.me');

//curl 批处理

$ch1 = curl\_init();

$ch2 = curl\_init();

curl\_setopt($ch1,CURLOPT\_URL,URL1);

curl\_setopt($ch1, CURLOPT\_TIMEOUT, 300000);

curl\_setopt($ch1,CURLOPT\_HEADER,0);

curl\_setopt($ch2, CURLOPT\_URL, URL2);

curl\_setopt($ch2, CURLOPT\_TIMEOUT, 300000);

curl\_setopt($ch2,CURLOPT\_HEADER,0);

$mh = curl\_multi\_init();

curl\_multi\_add\_handle($mh, $ch1);

curl\_multi\_add\_handle($mh, $ch2);

$active = null;

do{

$mrc = curl\_multi\_exec($mh,$active);

var\_dump($active);

}while($mrc==CURLM\_CALL\_MULTI\_PERFORM);

while($active && CURL\_OK == $mrc){

echo 123;

if(curl\_multi\_select($mh) != -1){

do{

$mrc = curl\_multi\_exec($mh,$active);

}while($mrc==CURLM\_CALL\_MULTI\_PERFORM);

}

}

curl\_multi\_remove\_handle($mh, $ch1);

curl\_multi\_remove\_handle($mh, $ch2);

curl\_multi\_close($mh);

# 第五章PHP与数据库基础

# 第六章PHP模板引擎的原理与实践

# 第七章PHP扩展开发

# 第八章缓存详解

# 第九章Memcached 使用和实践

# 第十章Redis 使用与实践

# 第十一章高性能网站架构方案

# 第十二章代码调试和测试

# 第十三章Hash算法与数据库实现

# 第十四章PHP编码规范

IDE:集成环境。

SOAP:简单对象访问协议。

OOPL:面向对象语言。

SPL:标准PHP库。

IOC(Inversion of Control):控制反转。

GIT():分布式版本控制系统。

依赖注入(DI)和依赖查找(DS)是IOC的两种实现。

JVM:java virtual machine。

Std: standard。标准

**HTTP协议：**

HTTP是一个应用层协议，由请求和响应构成，是一个标准客户端服务器模型。

**Hash算法和数据库实现：**