PHP 基础知识

false 的七种情况

1. 整型 0
2. 浮点 0.0
3. 布尔 false
4. 空字符串'',""
5. 字符串'0'
6. 空数组[]
7. NULL

超全局数组

1. $GLOBALS，包含下面 8 个超全局数组的值
2. $_GET
3. $_POST
4. $_REQUERT，包含$_GET,$_POST,$_COOKIE
5. $_SEESION
6. $_COOKIE
7. $_SERVER

```
$_SERVER['SERVER_ADDR'] //服务器地址

$_SERVER['SERVER_NAME'] //服务名称
```

```
$_SERVER['REQUEST_TIME'] //请求时间

$_SERVER['QUERY_STRING'] //请求地址中问号后的内容

$_SERVER['HTTP_REFERER'] //上次请求地址

$_SERVER['HTTP_USER_AGENT'] //浏览器信息

$_SERVER['REMOTE_ARRR'] //客户端请求 ip

$_SERVER['REQUEST_URI'] // 请求中脚本名称

$_SERVER['PATH_INFO'] // 请求中路径
```

1. $_FIELS
2. $_ENV

null 的三种情况

1. 直接赋值 NULL
2. 未定义变量
3. unset 销毁后的变量

常量

一定定义，不可删除和修改

1. const 更快，是语言结构，可定义类常量
2. define 是函数

预定义常量

1. **FILE** 文件所在路径+文件名
2. **LINE** 所在代码行
3. **DIR** 所在文件夹路径
4. **FUNCTION** 方法名
5. **CLASS** 类名
6. **TRAIT** TRAIT 的名称
7. **METHOD** 类名+方法名
8. **NAMESPACE** 命名空间名

引用变量

用不同名字访问同一个变量内容，用『&』符号表示

抽象类和接口

抽象类:

1. 定义为抽象的类不能被实例化.
2. 任何一个类，如果它里面至少有一个方法是被声明为抽象的，那么这个类就必须被声明为抽象的。
3. 被定义为抽象的方法只是声明了其调用方式（参数），不能定义其具体的功能实现。
4. 继承一个抽象类的时候，子类必须定义父类中的所有抽象方法；另外，这些方法的访问控制必须和父类中一样（或者更为宽松）。

5. 例如某个抽象方法被声明为受保护的，那么子类中实现的方法就应该声明为受保护的或者公有的，而不能定义为私有的。此外方法的调用方式必须匹配，即类型和所需参数数量必须一致。例如，子类定义了一个可选参数，而父类抽象方法的声明里没有，则两者的声明并无冲突。
6. 这也适用于 PHP 5.4 起的构造函数。在 PHP 5.4 之前的构造函数声明可以不一样的.

接口:

1. 使用接口（interface），可以指定某个类必须实现哪些方法，但不需要定义这些方法的具体内容。
2. 接口是通过 interface 关键字来定义的，就像定义一个标准的类一样，但其中定义所有的方法都是空的。
3. 接口中定义的所有方法都必须是公有，这是接口的特性。
4. 要实现一个接口，使用 implements 操作符。类中必须实现接口中定义的所有方法，否则会报一个致命错误。类可以实现多个接口，用逗号来分隔多个接口的名称。
5. 实现多个接口时，接口中的方法不能有重名。
6. 接口也可以继承，通过使用 extends 操作符.
7. 类要实现接口，必须使用和接口中所定义的方法完全一致的方式。否则会导致致命错误.

区别:

1. 对接口的继承使用 implements,抽象类使用 extends.
2. 接口中不可以声明变量,但可以声明类常量.抽象类中可以声明各种变量

3. 接口没有构造函数,抽象类可以有
4. 接口中的方法默认为 public,抽象类中的方法可以用 public,protected,private 修饰
5. 一个类可以继承多个接口,但只能继承一个抽象类

运算符优先级

> 优先级由高到低排序

1. ==递增/递减==
2. 非（!）
3. ==算术运算符==
4. ==大小比较==
5. （不）相等比较
6. 引用
7. 位运算符（^）
8. 位运算符（|）
9. ==逻辑与==
10. ==逻辑或==
11. ==三目==
12. ==赋值==
13. and
14. oxr
15. or

浮点数值得精度丢失问题

原因：因为计算机存储是二进制，准换进制时会有精度丢失

解决方案：先将浮点字符串化，再进行整数获取，输出可通过 print

```
$f = 0.57;

$f = $f * 100;// 输入可通过 printprint('%d', $f);

// 用于存储或二次计算，先将浮点字符串化，再进行整数获取

$f = strval($f);

var_dump($f);echo floor($f);echo intval($f);echo (int)($f);
```

switch 只能判断整型、浮点、字符

变量类型

1. 普通变量
2. 全局变量，通过 global 定义，可以在局部域调用全局变量，可通过$_GLOBAL['XXX'] 读取变量的值
3. 静态变量，通过 static 定义，仅在局部域中存在，执行函数离开作用域，其值也不会消失

ip 处理函数

1. ip2long()
2. long2ip()

时间日期处理函数

1. date()
2. strtotime()
3. mktime()
4. time()
5. microtime()
6. date_default_timezone_set()

打印处理

1. print() 仅输出单个变量
2. printf() 按格式输出
3. print_r() 格式化输出
4. echo 输出多个变量
5. sprintf() 按格式返回
6. var_dump() 格式化输出，并输出变量类型
7. var_export() 将格式化输出，加 true 可返回，返回内容可直接做变量使用

序列化

1. serialize()
2. unserialize()

字符串处理

1. implode(),join()
2. explode()
3. strrev() 反转字符
4. trim(),ltrim(),rtrim()
5. strstr() 获取第一次出现指定字符串的字符串
6. number_format() 数字格式化为金钱格式

数组处理

1. array_keys()
2. array_values()
3. array_diff()
4. array_megre()
5. array_shift()
6. array_unshift()
7. array_pop()
8. array_push()
9. sort(), rsort() 对数组升降序排序
10. asort(),arsort() 对数组键值升降序排序
11. ksort()，krsort() 对数组键名升降序排序

文件操作

fopen() 打开文件并指定模式

1. r/r+ 只读打开/读写打开，指针在文件开头
2. w/w+ 只写打开/读写打开，文件存在会清空，不存在会创建
3. a/a+ 写入追加写入/读写的追加写入，指针在文件末尾
4. x/x+ 写入/读写打开，指针开头，文件存在返回 false，不存在就直接创建
5. b 二进制打开

写入

1. fwrite()
2. fputs()

读取

1. fread() 获取指定长度字符
2. fgets() 获取一行字符
3. fgetc() 获取一个字符

关闭 fopen()

文件大小 filesize()

文件复制 copy()

文件删除 unlink()

文件类型 filetype()

重命名或移动 rename()

文件属性

1. file_exist()
2. is_readable()
3. is_writable()
4. is_executable()
5. filectime() 创建时间
6. fileatime() 访问时间
7. filemtime() 更新时间

其他不需要 fopen()打开的函数

1. file_get_contents()
2. file_put_contents()

其他

1. file()整个文件内容按行读取到一个数组里
2. readfile()整个文件读取出来，并输出

远程访问

`php.ini` 中打开 `allow_url_fopen` 配置,http 协议只能使用只读，ftp 协议，只能只读或只写

目录操作

名称相关

1. basename() 文件基础名称
2. dirname() 文件夹名称
3. pathinfo() 文件信息数组

目录读取

1. opendir()
2. readdir()
3. closedir()
4. rewinddir() 重置句柄
5. disk_free_space()
6. disk_total_space()

目录删除 rmdir()

目录必须为空

目录创建 mkdir()

重命名或移动 rename()

设计模式

1. 工厂模式
2. 单例模式

3. 适配器模式
4. 观察者模式
5. 策略模式
6. 注册树模式

魔术方法

1. __construct()
2. __destruct()
3. __call()
4. __callStatic()
5. __get()
6. __set()
7. __isset()
8. __unset()
9. __sleep()
10. __wakeup()
11. __toString()
12. __close()

网络协议

http 状态码

1. 200 请求成功

2. 204 not content
3. 206 reset content
4. 301 永久重定向
5. 302 临时重定向
6. 307 临时重定向
7. 400 错误请求
8. 401 缺少认证信息
9. 403 拒绝
10. 404 不存在
11. 500 服务器异常
12. 502 Bad Gateway
13. 503 服务器超负载或停机维护

OSI 七层协议

1. 物理层 建立、维护、断开物理连接
2. 数据链路层 建立逻辑连接，进行硬件地址寻址，差错校验等功能
3. 网络层 进行逻辑地址寻址，师兄不同网络之间的路径选择
4. 传输层 定义传输数据的协议端口号，一级流控和差错校验。协议有 TCP/UDP，数据包一旦离开网卡即进入网络传输层
5. 会话层 建立、管理、终止会话
6. 表示层 数据的表示、安全、压缩

7. 应用层 网络服务与最终用户的一个借口，协议有：http(80),ftp(21),tftp,smtp(25),snmp,dns(53),telnet(23),https(443),pop3(110),dhcp

HTTP 协议的工作特点和工作原理

工作特点：

1. 基于 B/S 模式
2. 通信开销小，简单快速，传输成本低
3. 使用灵活，可使用超文本传输协议
4. 节省传输时间
5. 无状态

工作原理：

客户端发送请求给服务器，建立一个 TCP 连接，指定端口号，默认 80，连接到服务器，服务器监听到浏览器的请求，一旦监听到客户端的请求，分析请求类型后，服务器会向客户端发送状态信息和数据内容

HTTP 协议常见请求头/响应头

1. Content-Type 指定数据内容类型
2. Accept 指定客户端能接受数据内容类型
3. Origin 最初请求来源（POST）
4. Cookie
5. Cache-Control 指定请求的缓存机制

6. User-Agent 用户浏览器信息
7. Referrer 上级请求路径
8. X-Forwarded-For 请求端真实 ip
9. Access-Control-Allow-Origin 允许其他请求域名，用于跨域
10. Last-Modified 最后响应时间

HTTPS 的工作原理

HTTPS 是一种基于 SSL/TLS 的 HTTP 协议，所有的 HTTP 数据都是 SSL/TLS 协议封装之上传输的

HTTPS 是在 HTTP 协议的基础上，添加了 SSL/TLS 握手一级数据加密传输，也属于应用层协议

微信二维码扫一扫咨询大神进阶课程内容

微信号码 若兰：zqf19907493857

QQ 咨询联系 妮妮：194210485