

PHP 学习笔记

郑华

2019 年 1 月 9 日

目录

第一章 PHP 简介	5
1.1 用途	5
1.2 配置	6
1.2.1 Apache2.4	6
1.2.2 php.ini	7
1.2.3 FastCGI	7
第二章 基本语法	9
2.1 变量	9
2.2 作用域	9
2.3 数组	10
2.3.1 数组函数	11
2.4 字符串函数	11
2.5 面向对象	11
2.6 表单原理	12
2.6.1 form 表单	12
2.6.2 input 标记	12
2.6.3 数据获取方式	13
2.7 JSON 读写	13

2.7.1	JSON 基本格式	13
2.7.2	encode	13
2.7.3	decode	13
2.8	文件上传	14
2.9	Cookie	14
2.10	Session	15
2.11	Token 验证	16
2.11.1	基于服务器的验证	16
2.11.2	基于 Token 的验证原理	16
2.12	PHP 传值	18
2.13	闭包	19
第三章 开发事宜		21
第四章 工具		23
4.1	PhpStorm	23
4.2	Xdebug	23

第一章 PHP 简介

<http://php.net/manual/zh/>

1.1 用途

PHP 脚本主要用于以下三个领域：

- **服务端脚本**。这是 PHP 最传统，也是最主要的目标领域。开展这项工作需要具备以下三点：

1. PHP 解析器（CGI 或者 服务器模块）
2. Web 服务器
3. Web 浏览器

需要在运行 web 服务器时，安装并配置 PHP，然后，可以用 web 浏览器来访问 PHP 程序的输出，即浏览服务端的 PHP 页面。如果只是实验 PHP 编程，所有的这些都可以运行在自己家里的电脑中。

- **命令行脚本**。可以编写一段 PHP 脚本，并且不需要任何服务器或者浏览器来运行它。通过这种方式，仅仅只需要 PHP 解析器来执行。这种用法对于依赖 cron（Unix 或者 Linux 环境）或者 Task Scheduler（Windows 环境）的日常运行的脚本来说是理想的选择。这些脚本也可以用来处理简单的文本。
- **编写桌面应用程序**。对于有着图形界面的桌面应用程序来说，PHP 或许不是一种最好的语言，但是如果用户非常精通 PHP，并且希望在客户端应用程序中使用 PHP 的一些高级特性，可以利用 PHP-GTK 来编写这些程序。用这种方法，还可以编写跨平台的应用程序。PHP-GTK 是 PHP 的一个扩展，在通常发布的 PHP 包中并不包含它。

使用 PHP，并不局限于输出 **HTML**。PHP 还能被用来动态输出图像、PDF 文件甚至 Flash 动画（使用 libswf 和 Ming）。还能够非常简便的输出文本，例如 XHTML 以及任何其它形式的 XML 文件。PHP 能够自动生成这些文件，在服务端开辟出一块动态内容的缓存，可以直接把它

们打印出来，或者将它们存储到文件系统中。

PHP 最强大最显著的特性之一，是它支持很大范围的**数据库**。使用任何针对某数据库的扩展（例如 mysql）编写数据库支持的网页非常简单，或者使用抽象层如 PDO，或者通过 ODBC 扩展连接到任何支持 ODBC 标准的数据库。其它一些数据库也可能会用 cURL 或者 sockets，例如 CouchDB。

PHP 还支持利用诸如 LDAP、IMAP、SNMP、NNTP、POP3、HTTP、COM（Windows 环境）等不计其数的**协议**的服务。还可以开放原始网络端口，使得任何其它的协议能够协同工作。PHP 支持和所有 web 开发语言之间的 WDDX 复杂数据交换。关于相互连接，PHP 已经支持了对 Java 对象的即时连接，并且可以透明地将其用作 PHP 对象。

1.2 配置

主要参考<https://www.cnblogs.com/cyrfr/p/6483529.html>

开发参考<https://www.douban.com/group/topic/111428204/>

1.2.1 Apache2.4

简介

- HTTP Server
- 支持最新的 HTTP/1.1 通信协议
- 支持 FastCGI
- 支持多种方式的 HTTP 认证等

httpd.conf 主要用于设置 Apache2 服务器的跟目录、PHP 模块关联。

- 服务器的根目录

```
Define SRVROOT "D:/LAMP-AMP/APACHE_2.4/Apache24"
ServerRoot "${SRVROOT}" // 指定守护进程httpd的运行目录
DocumentRoot "D:/Develop/Apache2.2/htdocs" // 指定站点目录
<Directory "D:/Develop/Apache2.2/htdocs"> // 指定目录执行规则
```

- PHP 模块关联，包括模块库、访问类型、PHP 主目录

```
LoadModule php7_module D:/LAMP-AMP/PHP_7.1/php7apache2_4.dll
```

```
AddType application/x-httpd-php .php .html .htm
PHPIniDir D:/LAMP-AMP/PHP_7.1
```

ServerRoot ServerRoot 用于指定守护进程 httpd 的运行目录，httpd 在启动之后将自动将进程的当前目录改变为这个目录，因此如果设置文件中指定的文件或目录是相对路径，那么真实路径就位于这个 ServerRoot 定义的路径之下。

启动 Apache 服务

- Windows
 - 开启服务 `net start mysql`
 - 停止服务 `net stop mysql`
 - 移除服务 `mysqld -remove`
- Linux `\etc\init.d\apache2 start`

1.2.2 php.ini

设置 php 的时间 `date.timezone = Asia/Shanghai`

1.2.3 FastCGI

To Be Continue...

第二章 基本语法

2.1 变量

变量 随身携带\$

使用其他文件变量时，使用include 'fileName'

传值 就是单纯的将值赋值给形参，和 copy 是一样的

传引用 类似于 C++ 语言的引用

```
<?php
    $param2=1; //定义变量2
    $param1 = &$param2; //将变量2的引用传给变量1
    echo $param2; //显示为1
    $param1 = 2; //把2赋值给变量1
    echo $param2; //显示为2
?>
```

2.2 作用域

- 局部变量
- 全局变量：将所有全局变量存储在一个名为 \$GLOBALS[index] 的数组中。index 保存变量的名称。这个数组可以在函数内部访问，也可以直接用来更新全局变量。
- 静态变量：当一个函数完成时，它的所有变量通常都会被删除。然而，有时候您希望某个局部变量不要被删除。要做到这一点，请在您第一次声明变量时使用 static 关键字

超级全局变量

- `$GLOBALS` 是一个包含了全部变量的全局组合数组。变量的名字就是数组的键。
- `$_SERVER` 是一个包含了诸如头信息 (header)、路径 (path)、以及脚本位置 (script locations) 等等信息的数组。这个数组中的项目由 Web 服务器创建。不能保证每个服务器都提供全部项目；具体含义查看<http://www.runoob.com/php/php-superglobals.html>
- PHP `$_REQUEST` 用于收集 HTML 表单提交的数据。
- PHP `$_POST` 变量是一个数组，内容是由 HTTP POST 方法发送的变量名称和值。可以在网址的栏目上是看不到传送的内容的
- PHP `$_GET` 变量是一个数组，内容是由 HTTP GET 方法发送的变量名称和值。可以在网址的栏目是看到内容的

花括号的区别 很多语言都以花括号作为作用域界限，PHP 中只有函数的花括号才构成新的作用域。

<https://www.cnblogs.com/52php/p/5670067.html>

2.3 数组

形式

- 普通数组: `$cars = array("Volvo","BMW","Toyota");`
- 键值数组: `$cars = array("Volvo"=>"30","BMW"=>"40","Toyota"=>"50");`
- 二维数组:

```
$cars = array(
    array("volvo",100,96),
    array("bmw",99,34),
    array("toyota",33,24)
);
```

- 新写法:`$cars= ["2","3","5"];`
- 初始化新方法:

```
$arr = array(5 => 1, 12 => 2);

$arr[] = 56; // This is the same as $arr[13] = 56;
           // at this point of the script
```

2.3.1 数组函数

- `count($arrVar)` :统计数组变量个数
- `foreach($arrVar as $varElement)`
- `foreach($arrKeyVar as $keyElement => $valueElement)`
- `array_keys()` 返回数组中所有的键名
- `array_pop()` 出栈
- `array_push()` 入栈
- `array_rand()` 随机选出一个或多个元素
- `array_shift()` 删除数组中的第一个元素，并返回删除元素的值，出队
- `in_array()` 检查数组中是否存在指定的值

2.4 字符串函数

- `explode()` 将字符串转化成数组
- `implode()` 将数组合并成一个字符串，等价于`join()`
- `trim()` 去掉字符串两边的字符
- `md5()` 计算字符串的 md5 散列
- `str_replace()` 替换字符串中的一些字符。

2.5 面向对象

与 c++ 大致一致，需要注意一下特性。

- 对象变量默认为指针，需要使用`->` 调用成员函数。
- `$this` `this` 变量也不例外需要使用 `$`, 与之对应的有`self::`成员变量
- 构造函数 `function __construct($args)`
- 析构函数 `function __destruct()`
- 继承使用关键字`extends`

- 接口使用关键字 `interface`、`implements` 类似于 Java
- `final` 标识的方法，子类不能再对其覆盖重写
- PHP 不会在子类的构造方法中自动的调用父类的构造方法。要执行父类的构造方法，需要在子类的构造方法中调用 `parent::__construct()`，同理，使用父类方法 `parent::xx()`

2.6 表单原理

主要参考<https://www.cnblogs.com/qiujun/p/6801896.html>

2.6.1 form 表单

- GET 将表单内容附加到 URL 地址后面，提交的信息长度有限制，不可以超过 8192 个字节，同时不具有保密性，而且只能传送 ASCII 字符（一般传送的不保密性数据）
- POST 将用户填写的数据包含在表单数据中，不会在地址栏中显示，同时没有数据长度的限制

默认 GET 方法，地址传值使用的 GET 方法

2.6.2 input 标记

- type 属性
 - text 文本域
 - password 密码域
 - radio 单选框
 - file 文件等
- name 表单名称
- action 目标地址, 绝对或相对 URL, 默认为当前页面
- enctype 表单编码方式
- \$_POST 数据存储于此

2.6.3 数据获取方式

- `$_GET['key'] || $_POST['key']`
- `isset($variable)` : 判断一个变量是否设置,isset 判断变量是否已存在 (配置)
- `empty($variable)` :empty 判断变量是否为空

常用判断: <https://blog.csdn.net/qiangzaiying123/article/details/62068438>

2.7 JSON 读写

2.7.1 JSON 基本格式

- 数组: `[1,2,4,"hello",[4,5,6],{"w":"World"}]`
- 对象: `{"h":"Hello", "w":"World", [1,2,3]}`

PHP 打印方式

- 打印对象: `print_r($obj)`
- 打印字符串: `echo ""`

2.7.2 encode

```
json_encode($obj)

|| $arr = array(1,2,3,'Hello','World',array('h'=>'Hello','w'=>'World'));
|| echo json_encode($arr);
```

2.7.3 decode

```
json_decode($obj)

|| $json_str = '{"h":"Hello","w":"World"}';
|| $obj = json_decode($json_str);
|| echo $obj->h;
```

2.8 文件上传

上传的文件全部存在 `$_FILES` 数组下

```
Array
{
    [file] => Array
        {
            [name] => xx
            [type] => xx
            [tmp_name] => xx
            [error] => xx
            [size] => xx
        }
}

$file = $_FILES['file'];
$fileName = $file['name'];
move_uploaded_file($file['tmp_name'], $fileName);
```

2.9 Cookie

什么是 cookie

服务器在客户端保存用户的信息，比如登录名，密码等

这些数据就像小甜饼一样，数据量并不大，服务器端在需要的时候可以从客户端读取，保存在客户端的浏览器缓存目录下

PHP 中 Cookie 的使用—添加/更新/删除/获取 Cookie 及自动填写该用户的用户名和密码和判断是否第一次登陆

- 数据存储在浏览器端
- 特点：
 - 方便与 JavaScript 交换数据
 - 方便获取用户信息
- 风险：浏览器可能会禁用 Cookie
- 替代方案：URL 参数

```

setcookie('user','zhenghua'); //设置一个cookie, 存储cookie 信息到浏览器

setcookie('sex', 'male', time() +30); // 设置过期时间为30s
setcookie('sex', 'male', time() -3600); // 删除cookie

/*
 * 1. 名字
 * 2. 键值
 * 3. 过期时间
 * 4. path cookie路径, 针对url的路径
 * 5. domain 域名, 针对哪个域名生效
 */
var_dump($_COOKIE); //取出cookie

```

2.10 Session

Session 对象存储特定用户会话所需的属性及配置信息。这样，当用户在应用程序的 Web 页之间跳转时，存储在 Session 对象中的变量将不会丢失，而是在整个用户会话中一直存在下去。当用户请求来自应用程序的 Web 页时，如果该用户还没有会话，则 Web 服务器将自动创建一个 Session 对象。当会话过期或被放弃后，服务器将终止该会话。**Session 对象最常见的一个用法就是存储用户的首选项。**例如，如果用户指明不喜欢查看图形，就可以将该信息存储在 *Session* 对象中。

```
session_start() $_SESSION
```

- 数据存储在服务器
- 特点：
 - 高效
 - 安全
 - 不依赖浏览器环境
 - 服务器端会为每一个用户用一个 ID 来标识
 - 短链接

客户端请求后，服务端返回一个 session，客户端然后存储在 cookie 中

2.11 Token 验证

在 Web 领域基于 Token 的身份验证随处可见。在大多数使用 Web API 的互联网公司中，tokens 是多用户下处理认证的最佳方式。特点如下：

- 无状态、可扩展
- 支持移动设备
- 跨程序调用
- 安全

2.11.1 基于服务器的验证

我们都是知道 **HTTP** 协议是无状态的，这种无状态意味着程序需要验证每一次请求，从而辨别客户端的身份。

在这之前，程序都是通过在服务端存储的登录信息来辨别请求的。这种方式一般都是通过存储 Session 来完成。随着 Web，应用程序，已经移动端的兴起，这种验证的方式逐渐暴露出了问题。尤其是在可扩展性方面。

- **Seesion**：每次认证用户发起请求时，服务器需要去创建一个记录来存储信息。当越来越多的用户发请求时，内存的开销也会不断增加。
- **可扩展性**：在服务端的内存中使用 Seesion 存储登录信息，伴随而来的是可扩展性问题。
- **CORS(跨域资源共享)**：当我们需要让数据跨多台移动设备上使用时，跨域资源的共享会是一个让人头疼的问题。在使用 Ajax 抓取另一个域的资源，就可以会出现禁止请求的情况。
- **CSRF(跨站请求伪造)**：用户在访问银行网站时，他们很容易受到跨站请求伪造的攻击，并且能够被利用其访问其他的网站。

2.11.2 基于 Token 的验证原理

基于 Token 的身份验证是无状态的，我们不将用户信息存在服务器或 Session 中。这种概念解决了在服务端存储信息时的许多问题。

NoSession 意味着你的程序可以根据需要去增减机器，而不用去担心用户是否登录。

基于 Token 的身份验证的过程如下：

1. 用户通过用户名和密码发送请求
2. 程序验证
3. 程序返回一个签名的 token 给客户端
4. 客户端储存 token, 并且每次用于每次发送请求
5. 服务端验证 token 并返回数据

每一次请求都需要 token。token 应该在 HTTP 的头部发送从而保证了 Http 请求无状态。我们同样通过设置服务器属性 *Access-Control-Allow-Origin:**，让服务器能接受到来自所有域的请求。需要主要的是，在 ACAO 头部标明 (designating)* 时，不得带有像 HTTP 认证，客户端 SSL 证书和 cookies 的证书。

示例 ->

```
//用户第一次登录
username pwd client_type
//接口判断
if(token&uid)
{
    查询token表
    $token=where uid =uid
    if($token==token)
    {
        登录成功!!
        返回token 和 uid
    }
    else
    {
        登录失败!!
    }
}

if(username pwd client_type)
{
    检验用户名和密码
    if(正确)
    {
        得到uid 并 生成token( md5(uid.pwd.time() 自己定义规则) )
        if(uid不存在)
```

```

        {
            into token 表    id uid token
        }
        else
        {
            where uid=$uid 修改token
        }
        返回token 和 uid
    }
    else
    {
        返回错误信息;
    }
}

```

客户端c进行文件存储uid 和token

下次再次登录时使用uid和token

当我们在程序中认证了信息并取得 token 之后，我们便能通过这个 Token 做许多的事情。

我们甚至能基于创建一个基于权限的 token 传给第三方应用程序，这些第三方可获取到我们的数据（当然只有在我们允许的特定的 token）

2.12 PHP 传值

&\$variable 传引用

\$variable 传值

面向对象的内存分布如下，变量指向该对象的编号，然后该编号 (系统控制) 又指向内存，因此，对于对象的传值和引用相当于改变对象变量和对象编号的赋值条件。如果是对象变量赋值，则是对该变量复制一份对象编号，而该编号还是指向该对象内存；如果是对象引用赋值，则是将对象编号也绑定到该变量身上。

- 对象变量
- 对象编号
- 对象内存

//对象的值传递示例:

```

$o1 = new C1();
$o2 = $o1;

```

```

$o1->p1 = 2;
echo "o1->p1={$o1->p1},o2->p1=␣{$o2->p1}"; //o1->p1=2,o2->p1 = 2

```

对象赋值内存效果如下：



图 2.1: 对象赋值

```

//对象的引用类型传递:
$o3 = new C1();
$o4 = &$o3;
$o3->p1 = 2;
echo "o3->p1={$o3->p1},o4->p1=␣{$o4->p1}"; //o3->p1=2,o4->p1 = 2

```

对象引用拷贝内存效果如下：

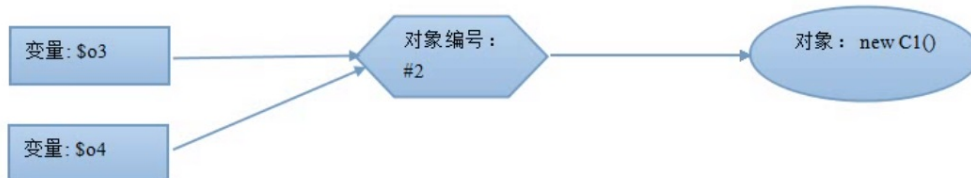


图 2.2: 对象引用

对象编号、克隆、对象模型:<https://blog.csdn.net/mizhenxiao/article/details/51909398>

2.13 闭包

闭包的语法很简单，需要注意的关键字就只有 use，use 意思是连接闭包和外界变量。

```

$a =function()use($b) {}

```


第三章 开发事宜

第四章 工具

4.1 PhpStorm

https://blog.csdn.net/gu_wen_jie/article/details/79136475

4.2 Xdebug

配置 php.ini <https://www.cnblogs.com/LWMLWM/p/8251905.html>

配置 phpstorm And webbrowser :<https://www.cnblogs.com/yxhblogs/p/6598387.html>