

# 半小时速成PHP基础

## 大纲

- 数据类型
- 变量，常量
- 函数
- 数组
- 面向对象

## 1. PHP数据类型

PHP 支持 8 种原始数据类型

四种标量类型：

- boolean（布尔型）
- integer（整型）
- float（浮点型，也称作 double）
- string（字符串）

两种复合类型：

- array（数组）
- object（对象）

最后是两种特殊类型：

- resource（资源）
- NULL（空类型）

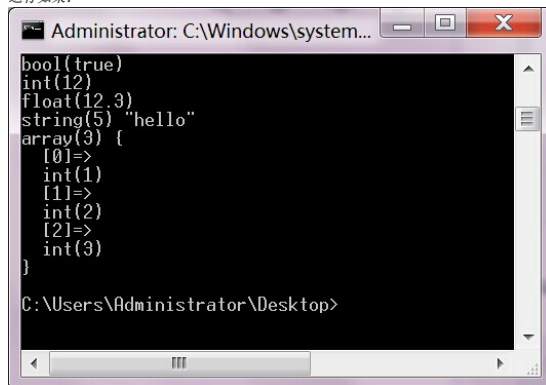
example:

```
<?php
$a = true;
$b = 12;
$c = 12.3;
$d = "hello";
$e = array(1, 2, 3);

//记住，不是这样初始化的哟，亲
//$e = [1, 2, 3];

var_dump($a);
var_dump($b);
var_dump($c);
var_dump($d);
var_dump($e);
?>
```

运行如果：



## 2. 变量

- 所有变量用\$开头，字母，数字，下划线
- 弱类型，不是由程序员决定，由运行时上下文决定
- 变量无需声明，直接使用，随便赋值
- 打开文件、数据库连接属于resource类型
- gettype函数获取变量类型，类型断言使用is\_type函数

example:

```
<?php
$a = "hello";
$b = array("hello", "world");

echo gettype($a) . "\n";
echo gettype($b) . "\n";

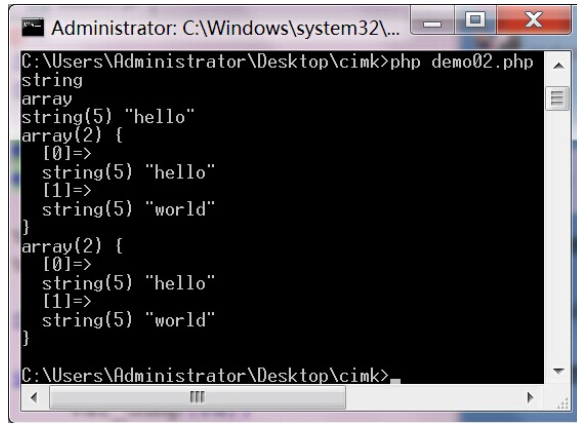
var_dump($a);
var_dump($b);
```

```
$a = $b;

if (is_array($a)) {
    var_dump($a);
}

?>
```

运行如果:



```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\Administrator\Desktop\cimk>php demo02.php
string
array
string(5) "hello"
array(2) {
    [0]=>
        string(5) "hello"
    [1]=>
        string(5) "world"
}
array(2) {
    [0]=>
        string(5) "hello"
    [1]=>
        string(5) "world"
}
C:\Users\Administrator\Desktop\cimk>
```

例子中几个常用函数，死记住

- `gettype`
- `is_type`
- `var_dump`

php 使用 ' ' 号拼接字符串

`echo`, `print` 用于输出字符串，它不是函数，是语言结构

### 3. 常量

example:

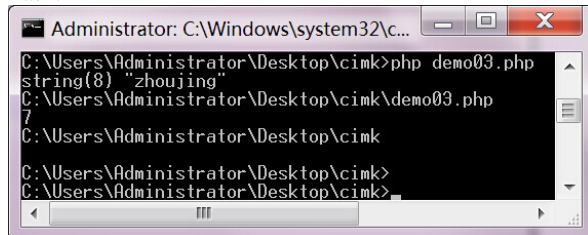
```
<?php
define("F00", "zhoujing");

var_dump(F00);

echo __FILE__ . "\n";
echo __LINE__ . "\n";
echo __DIR__ . "\n";

?>
```

运行如果:



```
Administrator: C:\Windows\system32\cmd.exe
C:\Users\Administrator\Desktop\cimk>php demo03.php
string(8) "zhoujing"
C:\Users\Administrator\Desktop\cimk\demo03.php
7
C:\Users\Administrator\Desktop\cimk
C:\Users\Administrator\Desktop\cimk>
C:\Users\Administrator\Desktop\cimk>
```

常量的特性:

- 常量的范围是全局的
- 自定义常量，不要以 '\_\_' 开头
- 魔术常量，PHP 向脚本注入的预定义常量

### 4. 函数

example:

```
<?php

function say($word) {
    echo $word . "\n";
}

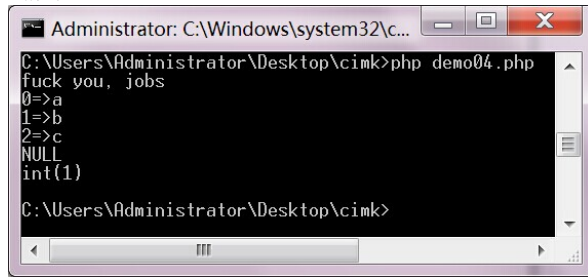
function foo($arr) {
    foreach($arr as $key => $item) {
        echo $key . "=>" . $item . "\n";
    }
    return 1;
}

$name = "jobs";
$return1 = say("fuck you, $name");
$return2 = foo(array("a", "b", "c"));

var_dump($return1);
var_dump($return2);
```

```
?>
```

运行如果:



```
Administrator: C:\Windows\system32\c...
C:\Users\Administrator\Desktop\cimk>php demo04.php
fuck you, jobs
0=>a
1=>b
2=>c
NULL
int(1)
C:\Users\Administrator\Desktop\cimk>
```

1.php不支持函数重载

2.php提供了非常多的内置函数, 可参考手册[PHP函数参考](#)

3.常用的如下:

- 字符串函数
- 数组函数
- 日期函数
- JSON函数
- 文件系统函数
- MySQL函数

对于这些函数, 熟练常用的, 其他要用的时候, 知道在哪查文档就OK

- empty
- count
- isset
- implode
- explode
- in\_array
- array\_push
- json\_encode
- json\_decode
- date

## 5. 数组

PHP中数组的功能非常强大, 几乎是现实了Java中的Array, List, Map的所有功能。因此可以把它当成真正的数组, 或列表(向量), 字典, 集合, 栈, 队列以及更多可能性。

example:

```
<?php
    $map = array(
        "name" => "zhoujing",
        "age" => 26
    );

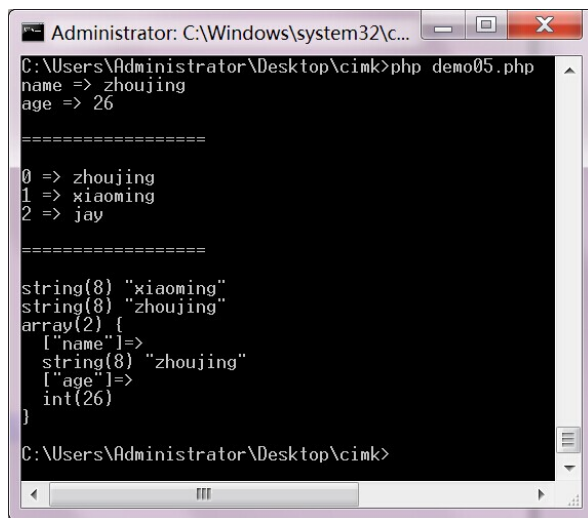
    $list = array("zhoujing", "xiaoming", "jay");

    function iteratorArray($arr) {
        foreach($arr as $key => $val) {
            echo $key . " => " . $val . "\n";
        }
    }

    iteratorArray($map);
    print "\n=====\\n\\n";
    iteratorArray($list);
    print "\n=====\\n\\n";

    var_dump($list[1]);
    var_dump($map["name"]);
    var_dump($map);
?>
```

运行如果:



```
Administrator: C:\Windows\system32\c...
C:\Users\Administrator\Desktop\cimk>php demo05.php
name => zhoujing
age => 26

=====
0 => zhoujing
1 => xiaoming
2 => jay

=====
string(8) "xiaoming"
string(8) "zhoujing"
array(2) {
    ["name"]=>
    string(8) "zhoujing"
    ["age"]=>
    int(26)
}

C:\Users\Administrator\Desktop\cimk>
```

## 6. 面向对象

PHP面向对象是PHP5完善的特性，基本上是参照Java现实的。

example:

```
<?php
class Person {
    private $name = "zhoujing";
    private $age = 26;

    function __construct() {
        print "In " . __CLASS__ . " constructor\n";
    }

    public function getName() {
        return $this->name;
    }

    public function getAge() {
        return $this->age;
    }

    public function setName($name) {
        $this->name = $name;
    }

    public function setAge($age) {
        $this->age = $age;
    }
}

class Studnet extends Person {
    private $school;

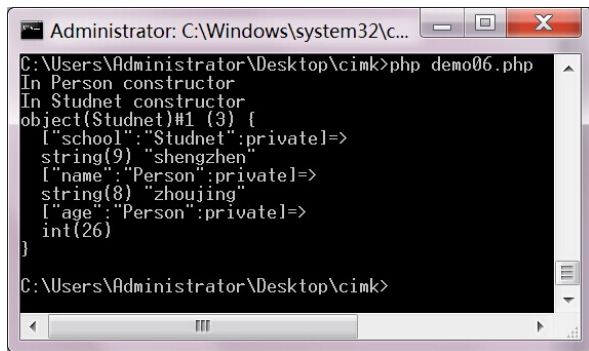
    function __construct() {
        parent::__construct();
        print "In " . __CLASS__ . " constructor\n";
    }

    public function getSchool() {
        return $this->school;
    }

    public function setSchool($school) {
        $this->school = $school;
    }
}

$tom = new Studnet();
$tom->setSchool("shengzhen");
var_dump($tom);
?>
```

运行如果:



## Web服务器搭建

- 理解Web服务器
- PHP程序运行方式
- 了解CGI
- PHP如何获取客户端数据
- Nginx配置

### 1. Web服务器

- 监听端口，接受用户请求，返回指定URL对应的文档资源
- 基于请求响应的被动交互方式
- 运行外部脚本或程序
- 实现Http协议
- apache, nginx, tomcat

### 2. php程序运行方式

- php有三个模块：内核，Zend引擎，扩展层
- Zend将源文件转成机器语言，在虚拟机上运行，将结果返回给内核
- 扩展层是一组函数，类库，流
- PHP内核用来处理请求、文件流、错误处理等相关操作
- apache通过php模块运行php，nginx通过cgi进程运行php

### 3. CGI

nginx不具有php模块，所以要通过cgi进程的方式运行php脚本。所以要在Linux开发应用层接口，必须要有web服务器程序，cgi(FastCGI)程序和PHP解析程序。CGI在物理上就是一个程序，在逻辑上是Web服务器与脚本程序的通信接口，规范。当然，还有其他一些网关程序，比如swgi,uwsgi。

### 4. PHP如何获取客户端请求数据

答案是：[超全局变量](#)

- \$GLOBALS
- \$\_SERVER
- \$\_GET
- \$\_POST
- \$\_FILES
- \$\_COOKIE
- \$\_SESSION
- \$\_REQUEST
- \$\_ENV

用CodeIgniter开发时，不太需要关注这些变量，有封装好的函数获取。

假如已经配置好了一个www.a.com服务，网站的根目C:\data\www\www.a.com。nginx配置如下：

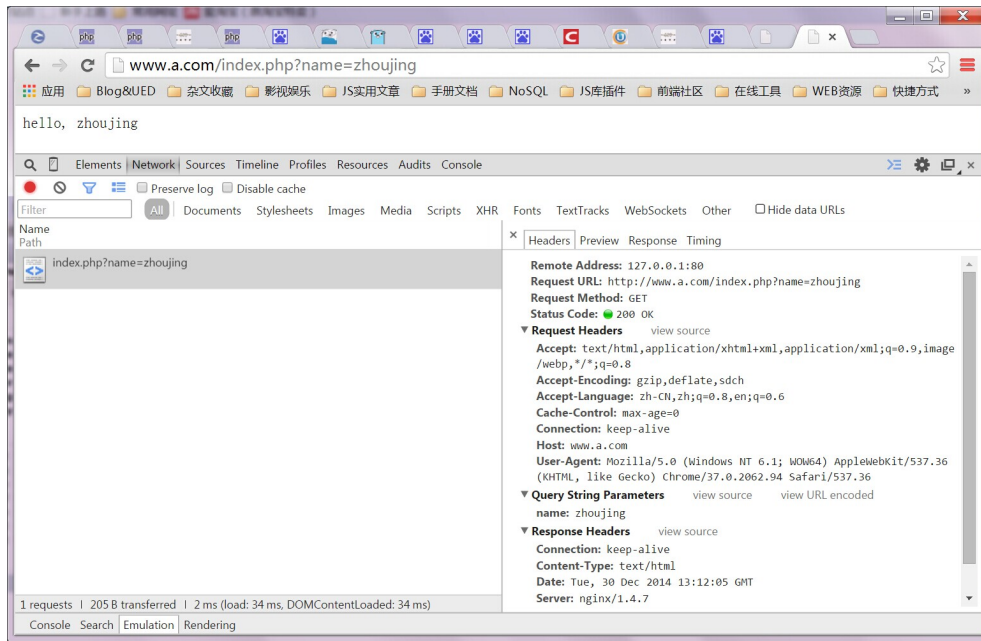
```
server {
    listen      80;
    server_name www.a.com;
    root        C:/data/www/a.com;
    index       index.html index.htm index.php;

    location ~ .*\.php$ {
        fastcgi_connect_timeout 300;
        fastcgi_send_timeout 300;
        fastcgi_read_timeout 300;
        fastcgi_pass 127.0.0.1:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

index.php源码:

```
<?php
if (isset($_GET['name']))
    echo "<h1>hello, " . $_GET['name'] . "</h1>";
else
    echo "you muse input your name.";
?>
```

运行如果:



## 5. Nginx配置CodeIgniter

nginx配置php:

```
server {
    listen      80;
    server_name api.impingo.me;
    root        /data/vhosts/api.impingo.me;
    index       index.html index.htm index.php;

    access_log  /data/logs/nginx_log/api.impingo.me/access.log main;

    location ~ \.(jpg|png|gif|js|css|swf|flv|ico)$ {
        expires 12h;
    }

    location ~* ^/(doc|logs|application|system)/ {
        return 403;
    }

    location / {
        if (!-e $request_filename) {
            rewrite ^(.*)$ /index.php?$1 last;
            break;
        }
    }

    location ~ .*\.php$ {
        fastcgi_connect_timeout 300;
        fastcgi_send_timeout 300;
        fastcgi_read_timeout 300;
        fastcgi_pass 127.0.0.1:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

Nginx服务器监听80端口，接受用户请求，通过调用CGI程序，与PHP脚本解释程序通信。也就意味着，对于每一个客户端请求，服务器要开启一个新的CGI进程进行处理。

Nginx,MySQL的安装配置，Linux系统基本操作不在本文介绍范围之内。请参考其他资料。

至此，PHP开发的最基础的内容已经介绍得差不多了。还有一点重要的内 PHP查询MySQL数据库。这里不介绍如何使用原始API操作MySQL，直接讲CI的MVC即可。