Content Management System

Contents

[MediaWiki 1](#_Toc468044595)

[Install LAMP 1](#_Toc468044596)

[配置wiki 2](#_Toc468044597)

[创建页面 3](#_Toc468044598)

[API调用 5](#_Toc468044599)

[Knowledge Server 7](#_Toc468044600)

[表单自动填写 8](#_Toc468044601)

[WordPress 9](#_Toc468044602)

[Install LAMP 9](#_Toc468044603)

[PHP 10](#_Toc468044604)

[install LAMP (见MediaWiki) 10](#_Toc468044605)

[脚本方式 10](#_Toc468044606)

[语法 11](#_Toc468044607)

[PHP 变量作用域 11](#_Toc468044608)

[字符串 12](#_Toc468044609)

[数组和关联数组 13](#_Toc468044610)

[条件语句同c 14](#_Toc468044611)

[文件系统 18](#_Toc468044612)

# MediaWiki

## Install LAMP

A web server: is required to serve the requested pages to the client browser.

PHP: is required to run the software.

A database server: is required to store the pages and site data.

<https://www.mediawiki.org/wiki/Manual:Running_MediaWiki_on_Ubuntu>

sudo apt-get update

//sudo apt-get upgrade

sudo apt-get install python-software-properties

sudo -E LC\_ALL=C.UTF-8 add-apt-repository ppa:ondrej/php

sudo apt-get update

sudo apt-get install apache2 mysql-server php7.0 php7.0-mysql libapache2-mod-php7.0 php7.0-xml

安装期间需要设置mysql，用户名:root, 密码：q

若apache2无法启动，$nmap localhost查看http 80端口是否打开

$sudo apt-get remove apache2

$sudo apt-get install apache2

Useful Packages

sudo apt-get install php-apc php5-intl imagemagick phpmyadmin vsftpd php5-cli

cd Downloads

wget <https://releases.wikimedia.org/mediawiki/1.27/mediawiki-1.27.0.tar.gz>

解压放于apache2网站根目录下

tar -xvzf /pathtofile/mediawiki-\*.tar.gz

sudo mv mediawiki /var/www/html

浏览器访问：localhost/mediawiki

若出现缺少扩展包，请安装

PHP extensions required

$sudo apt-get install php7.0-mbstring

扩展包安装包之后，要重启apache2

$sudo service apache2 restart

## 配置wiki

<https://www.howtoforge.com/how-to-install-mediawiki-on-ubuntu-14.04>

创建wiki的数据库帐户mediawiki:mediawiki@ip

$mysql -u root -p

mysql> create database mediawiki;

mysql> grant index, create, select, insert, update, delete, alter, lock tables on mediawiki.\* to 'mediawiki'@'%' identified by 'mediawiki';

mediawiki:mediawiki@任意ip,若限制mediawiki只能本机登录，将'%'改为'localhost'

mysql> flush privileges;

mysql> exit;

$sudo service apache2 restart;

$sudo service mysql restart;

浏览器访问： localhost/mediawiki

mediawiki:mediawiki@localhost/mediawiki 连接数据库

wiki名称：knowledge-mediawiki wiki管理员

admin:admin123

download LocalSettings.php and mv LocalSettings.php /var/www/html/mediawiki

Note: 通过客户端手动配置生成的LocalSettings.php，IP地址是静态的，建议改为动态参数如下：

$wgServer = "http://" . $\_SERVER["SERVER\_NAME"];

$wgEmergencyContact = "apache@127.0.0.1";

$wgPasswordSender = "apache@127.0.0.1";

查看mediawiki安装包

特殊页面 -> 数据与工具 -> 版本

浏览器访问：[http://192.168.1.12/mediawiki/](http://192.168.1.12/mediawiki/index.php/%E9%A6%96%E9%A1%B5)

修改默认logo

mediawiki/resources/assets/wiki.png 替换,注意大小是135x135

去除底部powered by图标

unset($wgFooterIcons['poweredby']); 在配置文件LocalSettings.php中加入

访问数据库

$mysql -u root -p

mysql> show databases;

mysql> use mediawiki;

mysql> show tables;

mysql> describe page;

## 创建页面

https://www.mediawiki.org/wiki/Help:Starting\_a\_new\_page/zh

搜索：HelloWorld or index.php/HelloWorld

常规方法：在已有页面编辑，添加[[HelloWorld]]，显示为红色，点击链接则创建页面HelloWorld

页面格式化

https://www.mediawiki.org/wiki/Help:Formatting/zh

''斜体'' 两对单引号

'''粗体''' 三对单引号

<strike>删除线</strike>

<pre style="color:red">

code here

</pre>

标题

== Level 2 ==

=== Level 3 ===

==== Level 4 ====

===== Level 5 =====

====== Level 6 ======

---- 添加水平线

<br /> 换行

空一行 分段

使用部分html标签，如<u>, <s>

[[HelloWorld|到HelloWorld页面]] 内部链接

[http://www.baidu.com 百度] 外部链接

[[Category:分类名]] 页面底部生成分类链接，点击链接后可看到该分类下的所有页面

无序列表 #表示有序列表

\*1

\*\*11

\*\*12

\*2

\*\*21

\*\*22

上传文件

LocalSetting.php

$wgEnableUploads = true;

若出现LocalFileLockError, 执行如下

$sudo chown -R www-data images www-data作为owner

[[File:File.png|200px|thumb|left|替代文字]] 引入图片

自动创建category树

<https://www.mediawiki.org/wiki/Extension:CategoryTree>

install <https://extdist.wmflabs.org/dist/extensions/CategoryTree-REL1_27-b454f2c.tar.gz>

tar -xzf CategoryTree-REL1\_27-b454f2c.tar.gz -C /var/www/mediawiki/extensions

LocalSettings.php

require\_once "$IP/extensions/CategoryTree/CategoryTree.php";

修改导航栏

搜索栏中输入“mediawiki:sidebar”，进入页面后点击编辑即可

\* navigation

\*\* mainpage|mainpage-description

\*\* recentchanges-url|recentchanges

\*\* randompage-url|randompage

\*\* special:allpages|allpages

\*\* helppage|help

\* SEARCH

\* TOOLBOX

\* LANGUAGES

\* Development

\*\* Special:Categories | Categories

\*\* MediaWiki:Sidebar|Navibar

<categorytree mode=pages>Categories</categorytree>

创建Category

Search: Category:Categories

正文为空

Search: Category:疾病

正文：[[Category:Categories]]

Search：Category:冠心病

正文：[[Category:疾病]]

创建卡片

Search: 放了支架不代表“万事大吉”

[[File:Smile.JPG | pic]]

经皮冠动脉介入治疗(PCI)在世界上已经有30多年的应用历史，专治冠心病。手术方法是，伸一条小管，沿着动脉通往心脏，在冠状动脉内植入支架，开通闭塞的血管，改善其狭窄或者阻塞的情况。

经皮冠状动脉介入治疗后，您以前狭窄或者阻塞的冠状动脉会得到改善。您的症状可能得到了缓解，或者您可以做更多的运动了。但是，放了支架并不代表万事大吉.

冠心病风险因互是指会增加你患冠心病和心脏病发作可能性的状况或者习惯，这些因素还会使得已有的病情更加恶化。已经明确的你可以控制的因素包括：

<ul>

<li>高血压</li>

<li>高胆固醇</li>

</ul>

[[Category:冠心病]]

## API调用

通过title获取页面

http://192.168.1.12/mediawiki/api.php?action=query&prop=revisions&rvprop=content&titles=HelloWorld

通过id获取页面

http://192.168.1.12/mediawiki/api.php?action=query&prop=revisions&rvprop=content&pageids=5

http://192.168.1.12/mediawiki/api.php?action=query&prop=revisions&rvprop=content&pageids=5|2

获取图像链接

result = http://192.168.1.12/mediawiki/api.php?action=query&list=allimages&pageids=2

href = result["query"]["allimages"][0]["url"]

通过Category获取页面集

<http://192.168.1.12/mediawiki/api.php?action=query&list=categorymembers&cmtitle=Category:Medical&cmtitle=Category:3D>

查询sub-category

<http://161.92.142.36/mediawiki/api.php?action=query&list=categorymembers&cmtype=subcat&cmtitle=Category:冠心病>

查看mediawiki已安装的库

search: special:version

wikitext -> html

method1: 假定wikitext code在foo.txt

/var/www/html/mediawiki$php maintenance/parse.php foo.txt > foo.html

method2: pandoc

http://pandoc.org/getting-started.html

online try:

http://pandoc.org/try/

html, Markdown, Latex, MediaWiki, docx, pdf...格式转换

$wget pandoc-1.17.2-1-amd64.deb

$dpkg -i pandoc…

$pandoc –help

pandoc -f mediawiki -t html5 -s haskell.wiki -o example32.html //MediaWiki -> html5

pandoc MANUAL.txt -o example1.html //markdown -> html

pandoc -s MANUAL.txt -o example4.tex //markdown -> LateX

pandoc -s -r html http://www.gnu.org/software/make/ -o example12.text //html -> markdown

pandoc MANUAL.txt --latex-engine=xelatex -o example13.pdf //markdown -> PDF

method3:

install restbase

https://github.com/wikimedia/restbase#restbase--

$git clone https://github.com/wikimedia/restbase.git

restbase$ npm install

note:若无法下载，设置代理

<http://stackoverflow.com/questions/7559648/is-there-a-way-to-make-npm-install-the-command-to-work-behind-proxy>

npm config set strict-ssl false

npm config set registry "http://registry.npmjs.org/"

npm config set proxy ...

npm config set https-proxy ...

restbase$ cp config.example.yaml config.yaml 修改配置 （比如url）

restbase$ node server 启动restbase服务器

install mediawiki extension

download RestbaseUpdateJobs

method4:

$ pip install -i http://pypi.pediapress.com/simple/ mwlib

$ pip install -i http://pypi.pediapress.com/simple/ mwlib.rl

$ mw-render -c http://192.168.1.12/mediawiki/ -o output.pdf HelloWorld -w rl

Method5:

<https://github.com/dcramer/py-wikimarkup/>

$ sudo pip -E install -i https://pypi.anaconda.org/pypi/simple py-wikimarkup

from wikimarkup import parse

html = parse(text[, show\_toc=True])

from wikimarkup import parselite

parselite(text)

客户端访问wiki

$pip install requests

$ sudo pip install -i https://pypi.anaconda.org/pypi/simple py-wikimarkup

import requests

import json

from wikimarkup import parselite

r = requests.get("http://localhost/mediawiki/api.php?action=query&prop=revisions& rvprop=content&titles=HelloWorld&format=json")

decode\_json = json.loads(r.text)

wikitext = decode\_json["query"]["pages"]["2"]["revisions"][0]["\*"]

print "======format: wikitext====="

print wikitext

print "======format: html========="

print parselite(wikitext)

## Knowledge Server

脚本创建数据库？

run-db.sh

USER="root"

PASSWORD="q"

mysql -u $USER -p$PASSWORD -e "DROP DATABASE if EXISTS knowledge;"

mysql -u $USER -p$PASSWORD -e "CREATE DATABASE knowledge CHARACTER SET utf8 COLLATE utf8\_general\_ci;"

mysql -u $USER -p$PASSWORD knowledge < knowledge\_DDL.sql

knowledge\_DDL.sql

DROP TABLE IF EXISTS coach\_plan\_package;

CREATE TABLE coach\_plan\_package

(

packageid int unsigned NOT NULL AUTO\_INCREMENT PRIMARY KEY,

name varchar(255) NOT NULL,

description varchar(1024),

diagnosis varchar(255)

);

DESCRIBE coach\_plan\_package;

DROP TABLE IF EXISTS coach\_plan;

CREATE TABLE coach\_plan

(

planid int unsigned NOT NULL AUTO\_INCREMENT PRIMARY KEY,

packageid int unsigned,

start\_day int unsigned NOT NULL,

end\_day int unsigned NOT NULL,

card\_title varchar(255) NOT NULL,

FOREIGN KEY (packageid) REFERENCES coach\_plan\_package(packageid)

);

DESCRIBE coach\_plan;

数据库帐户knowledge:knowledge@ip

mysql> show databases;

mysql> use knowledge;

mysql> show tables;

$mysql -u root -p knowledge < knowledge.sql //create table

$sudo cp data.tsv /var/lib/mysql/knowledge

mysql>load data infile “data.tsv” into table nameid ignore 1 lines; //import data

mysql>select \* from nameid;

mysql>select \* into outfile “data.csv’ fields terminated by ‘,’ from nameid; //output data

创建文章

//upload pic

login

Search: special:upload

Search:高血压是冠心病的高危因素之一 -> 点击红色， 创建页面

[[File:device.JPG | device]]

高血压是冠心病主要的危险因素，它常与其他危险因素（糖尿病、吸烟、肥胖等）同时存在并相互作用。

高血压会对血管壁产生额外的压力，持续的高血压会带来持续的压力。日子久了，这些额外的压力可能会损伤血管内皮，血脂沉积在内皮下，逐渐开成斑块，引起冠状动脉管腔狭窄和阻塞。

2010年中国高血压防治指南提出，我国人群心脑血管死亡占总死亡人数的…

[[Category:高血压]]

## 表单自动填写

<https://selenium-python.readthedocs.io/installation.html>

$apt-get -y install python-pip

$ pip install selenium (若不成功)

or

download selenium\*.tar.gz from <https://pypi.python.org/pypi/selenium#downloads>

tar –xvzf selenium\*.tar.gz

cd selenium\*

selenium\*$sudo python setup.py install

若浏览器用ubuntu自带的firefox，selenium 3版本以上需要geckodriver

download geckodriver <https://github.com/mozilla/geckodriver/releases> and mv geckordriver to /usr/bin

issue (post新页面)

<http://www.obeythetestinggoat.com/how-to-get-selenium-to-wait-for-page-load-after-a-click.html>

for example:

LocalSettings.php

# !/usr/bin/python

# coding:utf-8

import sys

reload(sys)

sys.setdefaultencoding('utf8')

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from contextlib import contextmanager

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support.expected\_conditions import \

staleness\_of

class MediawikiInitializer:

def \_\_init\_\_(self, url):

self.url = url

profile = webdriver.FirefoxProfile()

profile.set\_preference("network.proxy.type", 1)

profile.set\_preference("network.proxy.no\_proxies\_on", " localhost, 127.0.0.1, 161.92.142.\*;161.92.141.\*")

profile.update\_preferences()

self.browser = webdriver.Firefox(firefox\_profile=profile)

self.browser.implicitly\_wait(60\*5) # seconds

@contextmanager

def wait\_for\_page\_load(self, timeout=30):

old\_page = self.browser.find\_element\_by\_tag\_name('html')

yield

WebDriverWait(self.browser, timeout).until(

staleness\_of(old\_page)

)

def send\_keys(self, id, value):

element = self.browser.find\_element\_by\_id(id)

element.clear()

element.send\_keys(value)

def submit(self):

with self.wait\_for\_page\_load(timeout=10):

self.browser.find\_element\_by\_name("submit-continue").submit()

def run(self):

# http://161.92.142.80:8080

self.browser.get(self.url)

with self.wait\_for\_page\_load(timeout=10):

self.browser.find\_element\_by\_link\_text('set up the wiki').click()

# /mw-config/index.php?page=Language

self.submit()

# /mw-config/index.php?page=Welcome

self.submit()

# /mw-config/index.php?page=DBConnect

self.send\_keys('mysql\_wgDBserver', 'mysql')

self.send\_keys('mysql\_wgDBname', 'mediawiki')

self.send\_keys('mysql\_\_InstallUser', 'root')

self.send\_keys('mysql\_\_InstallPassword', 'q')

self.submit()

# /mw-config/index.php?page=DBSettings

self.submit()

# /mw-config/index.php?page=Name

self.send\_keys('config\_wgSitename', 'knowledge-wiki')

self.send\_keys('config\_\_AdminName', 'admin')

self.send\_keys('config\_\_AdminPassword', 'admin123')

self.send\_keys('config\_\_AdminPasswordConfirm', 'admin123')

self.submit()

# /mw-config/index.php?page=Options

self.submit()

# /mw-config/index.php?page=Install

self.submit()

# /mw-config/index.php?page=Install

self.submit()

# /mw-config/index.php?page=Complete

#download\_url = self.url + '/mw-config/index.php?localsettings=1'

if \_\_name\_\_ == '\_\_main\_\_':

url = "http://161.92.142.80:8080"

mediawiki = MediawikiInitializer(url)

mediawiki.run()

# WordPress

个人发布系统,也可以把 WordPress 当作一个内容管理系统（CMS），用来搭建一个小门户网站

WordPress后台+Wordpress主题，就构成了一个完整的网站

目前WordPress已不在是一个简单的Blog程序，你不仅可以使用它来搭建个人博客，还可以搭建其他常见类型的网站，比如门户、下载站、淘宝客、论坛、多博客等等

## Install LAMP

$wget <https://cn.wordpress.org/wordpress-4.5.3-zh_CN.zip>

创建wordpress的数据库帐户wordpress: wordpress@ip

$mysql -u root -p

mysql> create database wordpress;

mysql> grant index, create, select, insert, update, delete, alter, lock tables on wordpress.\* to ‘wordpress’@'%' identified by 'wordpress';

wordpress: wordpress@任意ip,若限制wordpress只能本机登录，将'%'改为'localhost'

mysql> flush privileges;

mysql> exit;

$sudo service apache2 restart;

$sudo service mysql restart;

浏览器访问

<http://192.168.1.12/wordpress/>

wordpress: wordpress @localhost/ wordpress 连接数据库

生成wd-config.php放入wordpress/

Wordpress站点标题：wordpress\_qzlin

admin:admin123 wordpress管理员

管理页面

<http://192.168.1.12/wordpress/wp-admin/>

站点

<http://192.168.1.12/wordpress/>

WordPress除了可以通过分类来关联文章，还可以通过标签来关联。有些WordPress主题也会自动调用这里填写的标签作为这篇文章的关键词（Keyword）。标签的长度一般为2-6个字为宜，一般填写2-4个标签即可

一般来讲 WordPress 提供了博客日志的作者，分类，标签，时间等

你可以设置评论和引用通告的开关。若该篇文章有评论，你可以在这里浏览、审核评论。如果你不允许别人评论这篇文章，不勾选即可。

注：你可以在后台 - 设置 - 讨论，设置全局的讨论设置（比如是否开启整站评论、垃圾评论过滤、评论审核等）

LimeSurvey (Web问卷调查系统)

Install LAMP

//database

$mysql -u root -p

mysql> create database limesurvey;

mysql> grant SELECT, CREATE, INSERT, UPDATE, DELETE, ALTER, DROP, INDEX on limesurvey.\* to ‘limesurvey’@'%' identified by ‘limesurvey’;

//install web application

/var/www/html$ sudo wget http://download.limesurvey.org/latest-stable-release/limesurvey2.53+161004.zip

/var/www/html$sudo unzip \*.zip

/var/www/html$chmod -R 777 limesurvey #set directory permissions

// install via access cli 运行安装脚本

localhost/limesurvey

admin:admin123 管理员和密码

# PHP

## install LAMP (见MediaWiki)

## 脚本方式

PHP 是一种创建动态交互性站点的强有力的服务器端脚本语言。

<!DOCTYPE html>

<html>

<body>

<?php

echo "hello world!"

?>

</body>

</html>

$php helloworld.php 执行<?php …?>里的命令

启动apache2服务

$sudo service apache2 restart

将helloworld.php文件放到apache2网站根目录下，即/var/www/html/

$curl localhost/helloworld.php

若php文件没有解析，说明apache2与php解释器没有关联，需要安装libapache2-mod-php7.0

或在连网机器浏览器

ip/helloworld.php

$php --ini //see CLI ini location

## 语法

### PHP 变量作用域

local（局部）函数内部声明的变量

global（全局）函数之外声明的变量

<?php

$x=5;

$y=10;

function myTest() {

$GLOBALS['y']=$GLOBALS['x']+$GLOBALS['y'];

}

myTest();

echo $y; // 输出 15

?>

static（静态） 同C语言

每当函数被调用时，这个变量所存储的信息都是函数最后一次被调用时所包含的信息。

注释：该变量仍然是函数的局部变量

常量

define('PI', 3.14)

echo PI;

超级全局变量

$GLOBALS 所有全局变量数组

$\_SERVER 服务器环境变量数组

$\_GET 通过GET方法传递给该脚本的变量数组

$\_POST 通过POST方法传给给该脚本的变量数组

$\_COOKIE cookie变量数组

$\_FILES 与文件上传相关的变量数组

$\_ENV 环境变量数组

$\_SESSION 会话变量数组

错误抑制操作符

@ $a = 57/0;

执行操作符 同shell反向单引号，将当中的命令当作服务器端的命令行来执行

$out = `ls -l`

测试函数

gettype(var)

is\_array(var)

is\_double(var), is\_float(var), is\_real(var)

is\_long(var), is\_int(var), is\_integer(var)

is\_string(var)

is\_bool(var)

is\_object(var)

is\_resource(var)

is\_null(var)

is\_scalar(var)

is\_numeric(var)

is\_callable(var) 检查是否是有效的函数名称

is\_set(var) 检查变量是否已设置

unset(var) 销毁变量

empty(var) 检查变量是否存在，值为非空或非0

### 字符串

trim() 除去字符串开始和结束位置的空格，结果返回

nl2br() 将字符串作为输入参数，用html的<br />替代字符串的换行行

str2upper()

str2lower()

ucfirst() 字符串的第一个字母转大写

ucwords() 字符串每个单词的第一个字母转大写

addslashes() 转义字符串

stripslashes() 解转义字符串

$email\_array = explode('@', $email) 分割字符串

implode() or join() 连接字符串

substr()

strcmp()

strlen()

strpos()

str\_replace()

正则表达式

在PHP中，必须将正则表达式模式包括在一个单括号字符串中

int ereg(string pattern, string search, array [matches])

该函数搜索字符串search, 在pattern中寻找与正则表达式相匹配的字符串。将匹配的字符串存储在数组matches中

if (!ergi( '^[a-zA-Z0-9\_\-\.]+@[a-zA-Z0-9\-]+\.[a-zA-Z0-9\-\.]+$', $email) {

echo "<p>That is not a valid email address.</p>";

exit;

}

$arr = split("\.|@ ", "qizhong.lin@gmail.com"); 分割字符串

["qizhong", ".", "lin", "@", "gmail", ".", "com"]

### 数组和关联数组

$products = array( 'Tires', 'Oil', 'Spark Plugs');

$numbers = range(1, 10, 2);

$letters = range('a', 'z');

array\_push($products, 'Hi')

array\_pop($products)

$products[0]

count($products)

array\_count\_values($products) 直方图

foreach ($products as $current) {

echo $current . " "

}

$prices = array('Tires'=100， 'Oil'=>10, 'Spark Plugs'=>4);

$prices['Spark Plugs']

foreach ($prices as $key => $value) {

echo $key. "-" . $value. "<br />";

}

reset($prices);

while ( list($product, $price) = each($prices) ) {

echo "$product - $price<br />";

}

sort($products);

asort($prices); 基于数组的元素值排序

ksort($prices); 基于数组的元素key排序

array\_reverse($products); 反向数组

shuffle($products); 随机排序

$word\_array = explode("\t", $line); 分割行

array\_walk(&$array, 'my\_multiply', 3); 面向函数式编程

function my\_multiply(&$value, $key, $factor) {

$value \*= $factor;

}

### 条件语句同c

if (condition) {

} elseif (condition) {

} else {

}

switch (condition) {

case match1:

break;

default:

break;

}

while (condition) {

}

do {

} while (condition);

for ($i=1; $i <= $len; ++ $i) {

}

代码重用

require() or include() 引入页眉或脚注，等效于c/c++中的#include

require\_once() or include\_once() 引入函数库

<?php require('header.php'); ?>

<div>Main Content</div>

<?php require('footer.php'); ?>

函数

函数调用不区分大小写，变量名区分大小写

如果声明了自己的函数，它们只是在声明它们的脚本中可以使用。将经常用到的函数包含在一个文件中是一个很好的主意。然后可以在所有脚本中调用require()语句，这样这些函数就可以使用了

php不支持函数重载

<?php

function fun($my\_array, $border=1) {

return ...;

}

?>

参数的引用传递和值传递

通常，函数获取参数的方式是值传递。当传递一个参数的时候，一个新的并且包含该传入值的变量被创建。它是原来那个变量的副本，可以以任意的方式修改它，但函数外部原来变量的值是不会改变的

引用传递：在参数被传递给函数的时候，函数不会再创建一个新变量，而是函数获得一个原来变量的引用。

function increment(&$value, $amount=1) {

$value += $amount;

}

递归函数，适用于浏览动态数据结构如列表或树

类

class A

{

public $attribute = "default value";

function \_\_get($name) {

return $this->$name;

}

function \_\_set($name, $value) {

if ( ($name=="attribute") && ($value>=0) && ($value<=100) ) {

$this->$name = $value;

}

}

权限有public, protected, private

function operation() { 默认是public

}

等同于Java toString()

public function \_\_toString() {

return (var\_export($this, TRUE); 打印类中的所有属性值

}

}

class B extends A

{

public $attribute = "different value";

function operation() 重载父类操作

{

parent::operation();

...

}

}

final 关键字

接口

interface Displayable {

function display();

}

class WebPage implements Displayable {

function display() {

}

}

$b = new B();

$c = clone $b;

遍历对象属性

foreach ($b as $attribute) {

echo $attribute. "<br />";

}

反射是通过访问已有类和对象来找到类和对象的结构和内容的能力

$class = new ReflectionClass("Page");

echo "<pre>" . $class . "</pre>";

Web开发常见的网页类

page.inc

<?php

class Page

{

public $content;

public $title = "Hello";

public $keywords = "php, programming";

public $buttons = array("Home" => "home.php",

"Contact" => "contact.php",

"Site Map" => "map.php");

public function \_\_set($name, $value) { $this->$name = $value; }

public function Display() {

echo "<html>\n<head>\n";

$this->DisplayTitle();

$this->DisplayKeywords();

$this->DisplayStyles();

echo "</head>\n<body>\n";

$this->DisplayHeader();

$this->DisplayMenu($this->buttons);

echo $this->content;

$this->DisplayFooter();

echo "</body>\n</html>\n";

}

public function DisplayTitle() {}

...

}

services.php

<?php

require("page.inc")

class ServicesPage extends Page

{

private $row2buttons = array( "Re-engineering" => "reengineering.php", ...);

public function Display()

{

echo "<html>\n<head>\n";

$this->DisplayTitle();

$this->DisplayKeywords();

$this->DisplayStyles();

echo "</head>\n<body>\n";

$this->DisplayHeader();

$this->DisplayMenu($this->buttons);

$this->DisplayMenu($this->row2buttons);

echo $this->content;

$this->DisplayFooter();

echo "</body>\n</html>\n";

}

}

$services = new ServicesPage();

$services->content = "...";

$services->Display();

## 文件系统

fopen也可以通过ftp, http, or 其他协议来打开文件

写文件

@ $fp = fopen( "$\_SERVER['DOCUMENT\_ROOT' ]/../orders/orders.txt", 'ab' )

flock($fp, LOCK\_EX) ; 写文件锁定

if (!$fp) {

echo "...";

exit;

}

fwrite($fp, $output, strlen($output));

flock($fp, LOCK\_UN);

fclose($fp)

读文件

@ $fp = fopen( "$\_SERVER['DOCUMENT\_ROOT' ]/../orders/orders.txt", 'rb' )

flock($fp, LOCK\_SH) ; 读文件锁定

if (!$fp) {

echo "...";

exit;

}

while ( !feof($fp) ) {

$order = fgets($fp, 999); 行读入，直至换行符或文件结束符EOF,或超过大小998B

}

flock($fp, LOCK\_UN);

fclose($fp)

$filearray = readfile(filename) 读取整个文件,返回以行为元素的数组

fgetcsv() 读取csv文件，行方式解析，返回数组

file\_exists() 查看文件是否存在

filesize() 确定文件大小

unlink() 删除一个文件

访问数据库

@ $db = new mysqli('localhost', 'user', 'password', 'books');

$result = $db->query("select \* from books");

for ($i=0; $i<$result->num\_rows; ++ $i) {

$row = $result->fetch\_assoc();

$row['title'], ...

}

$result->free();

$db->close();

包管理

Java从maven->gradle，Python有pip，Ruby 有 gem，Nodejs 有 npm。PHP 从PEAR->Composer

Composer 中的很多理念都借鉴自 npm

安装composer

$curl -sS https://getcomposer.org/installer | php

$sudo mv composer.phar /usr/local/bin/composer

$composer -V

Laravel

$composer global require "laravel/installer"

$gedit ~/.bashrc

export PATH=$PATH:~/.config/composer/vendor/bin

$source ~/.bashrc

创建项目

$laravel new blog

若缺少zip，则安装$sudo apt-get install php7.0-zip