Content Management System

Contents

[MediaWiki 1](#_Toc460067417)

[PHP 2](#_Toc460067418)

[install LAMP (见MediaWiki) 2](#_Toc460067419)

[脚本方式 2](#_Toc460067420)

[语法 3](#_Toc460067421)

[PHP 变量作用域 3](#_Toc460067422)

[数组和关联数组 4](#_Toc460067423)

[条件语句同c 5](#_Toc460067424)

[文件系统 6](#_Toc460067425)

# 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

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

修改导航栏

搜索栏中输入“mediawiki:sidebar”，进入页面后点击编辑即可

\*导航栏名称一

\*\*链接一地址|链接一名称

\*\*链接二地址|链接二名称

\*导航栏名称二

\*\*链接一地址|链接一名称

\*\*链接二地址|链接二名称

功能：

导览，搜索，跟踪更改，监视列表

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

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

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

method3:

install restbase

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

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

restbase$npm install

restbase$cp config.example.yaml config.yaml 修改配置

restbase$npm 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

# 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