a“...”打印变量的值，’...’代表字符串，等同于\...（<<< 界定符，后接相同标识符包裹的字符串）

unset(...)释放变量 is\_null(...)判断是否为空

is\_numeric(…)是否为数字或数字组成的字符串// echo接Bool不会输出任何东西

类型转换: (integer)... settype(..., ‘integer’)返回转换成功或失败

圆周率pi()

define ("MESSAGE","常量");定义常量，默认大小写敏感，最后加true可使大小写不敏感

echo Message."<br>";//输出Message，并Notice

define("COUNT", "能看到多次",true);

echo COUNT."<br>";

echo Count."<br>";//也输出 ”能看到多次”，因为设定了大小写不敏感

$name = "count";

echo constant ($name)."<br>";//也输出 ”能看到多次”

echo (defined("MESSAGE"))."<BR>";//返回 1

预定义常量

echo "当前文件路径:".\_\_FILE\_\_;

echo "<br>当前行数:".\_\_LINE\_\_;

echo "<br>当前PHP版本信息:".PHP\_VERSION;

echo "<br>当前操作系统:".PHP\_OS;

某些预定义常量：p53

可变变量

$j=&$i是引用赋值，直接=是直接赋值

$change\_name = "trans";

$trans = "You can see me!";

echo "<br>";

echo $$change\_name; //输出 You can see me!

var\_dump输出变量相关信息，包括类型

@屏蔽错误

$err = @(5/0)//返回INF，不会报错

变量函数：跟可变变量很像 函数引用，须在函数定义和赋值前面都加&

取消引用：unset(…)

命名规则：类BigClassObject 类属性mValue function GetLessonOne($firstWord){}

引用变量r前缀 全局变量g前缀 全局常量define(‘DEFAULT\_NUM\_AVE’,90)

静态变量s前缀 函数function this\_good\_idear(){}

if: …endif; : endwhile等等可代替写大括号 break n;跳出几重循环

双引号字符串会检测其中的变量，单引号不会

trim(str1, str2)，去掉两边的可选参数(\0、\t、\n、\x0B、\r、” ”)，str2可选，去掉哪些字符

同理，ltrim()左边，rtrim右边

addslashes($str), 返回转义后的字符串(如给引号加斜线)，stripslashes($str)返回还原字符串

addcslashes($str1,$str2) $str2指定$str1中哪些字符转义，stripcslashes还原

strlen()获取字符串长度 substr(string str, int start, int length)截取指定字符串，第二个参数可为负(倒数第几个开始) 若str为中文且为奇数，就会出现乱码, 可以用mb\_substr

strcmp(str1,str2)比较字符串，返回值与0的关系(依次比较每个字符的ASCII码值,2参减1参)。 strncmp(str1,str2,n)比较前n个字符

将字符串中的数字按自然排序法比较(其他字符比较方式同strcmp)： strnatcmp(str1, str2), 参1减参2 同样有不区分大小写的strnatcasecmp(str1, str2)

strstr(str1,str2)检索截取str2首次在str1中出现的位置直到末尾,返回字符串 strchr()正好相反//?lymark

substr\_count(str1,str2)获取str2在str1中出现的次数

str\_ireplace($str2,str1,str,n)将str中的str2以str1替换，n可选, str\_replace区分大小写

substr\_replace($str,$replace,int start, int length);替换指定范围的字符串

number\_format($number, 2, ‘.’, ‘.’) 1参：取整；2参：小数点后几位；3参:代替小数点 4个：代替千位分隔符的逗号

explode($separator, $str, limit) limit可选(表示分成几段)，如果为负，则返回除了最后limit个元素的所有元素

输出数组：print\_r($str\_arr)

合成字符串：implode($separator, arr)

正则表达式: ^开始 $结尾 \b \B单词定界符 [Tt] (T/t)大小写都可匹配 [a-z]范围 排除字符[^a-z] 限定符：?零次或一次 +一次或多次 \*零次或多次 {n}n次 {n,}最少n次 {n,m}最少n次，最多m次 .换行符外的任意一个字符

\a警报 \b退格 \e Escape \f换页 \n换行 \r回车 \t水平制表符 \xhh十六进制代码 \ddd八进制代码 \cx contrl-x的缩写

预定义字符集:\d十进制数 \D非十进制数 \s空白字符 \S非空白字符 \w单词字符 \W非单词字符

限定符:\b单词分界符 \B非单词分界符序列 \A总是能够匹配待搜索文本的起始位置 \Z在未指定任何模式下匹配的字符 \z只匹配字符串的末尾，而不考虑任何换行符 \G当前匹配的起始位置

反向引用，模式修饰符？p122

//ereg已废弃，用preg代替 验证表达式:ereg(string pattern, string string, $array) $array是返回的匹配成功的子串数 eregi不区分大小写

ereg\_replace(string pattern, string replacement, string str)返回的替换完后的字符串 eregi\_replace不区分大小写

split/spliti(string pattern, string string, int limit)分割字符串 //已废弃

preg\_grep(string pattern, array input)返回由input中所有匹配pattern的元素组成的数组

preg\_match(string pattern, string subject)返回匹配次数0或1，preg\_match\_all(pattern, subject, array)返回匹配次数, 其中array接收匹配。

preg\_quote($str, $str2)将str中的特殊字符和包含$str2的字符转义, 若$str2是字符串，按其第一个字符处理

preg\_replace(pattern, replacement, subject, limit)将匹配替换

preg\_replace\_callback(pattern, callback, subject, limit)回调的方式替换,其中是匿名函数,参数是匹配成功的数组,匿名函数的返回值的是被替换的

preg\_split(pattern , subject, limit)按匹配分割

$ubb=preg\_replace(“/\[url\](.\*)\[\(url)\]/”,”<a href=\\1>\\1</a>”,$ubb)

遍历数组foreach ($url as $link){...}

while (list($name, $value)=each($\_POST)){...}

count(arr, COUNT\_RECURSIVE)返回数组元素的个数，第二个参数可选，可计算二维数组

array\_search(mixed needle, array, bool strict)返回搜索到的键值(数组), 若strict为true，还讲在数组中检查给定值的类型

array\_pop()获取数组中最后一个单元，并将长度减1

array\_push(arr, …)添加元素，返回新数组元素个数

array\_unique()删除数组中重复的元素(按值判定重复(本身数组如果key一样的话就会按后赋值的key只保留一个值),返回值,原数组不变)。

array\_search(str1, strArr)返回strArr中str1的第一个索引

数组排序: 升序:sort($cars); 降序:resort(不会保留原来的key) 按值升序排序: asort($age); 按键升序排序: ksort($age); 按键降序 会改变原数组,不是返回值

form标签：name，method，action指向该表单页面的URL，enctype设置表单内容的编码方式，target设置返回信息的显示方式

表单:<form name= method= action= enctype=”mutipart/form-data” target= >…</form>

输入域:<form><input name=”name” type=”type\_name” value=”value” size=”12” maxlength=”1000”></form> type属性:”text” “password” “file” “image” “radio” “checkbox” “submit” “reset” “button” “hidden”隐藏域

选择域:<select name=”spec” id=”spec” mutiple><option value=”0” selected>选项1</option></select>

文本域<form name = "form1" method="post" action="index.php"><textarea name="a" cols="20" rows="3" wrap="soft">"text"</textarea></form> 其中wrap属性自动换行 get方法是将表单内容附在URL地址后，post方法是将表单的信息作为一个数据块发到服务器的处理程序中，浏览器上不显示。

$\_POST[] $\_GET[] $\_SESSION[]

Include(“top.php”)

url编码：urlencode() 解码:urldecode()

**Java Script:**

<script language="javascript"> … </script>

打印：document.write(…)

javascript: 变量名严格区分大小写

输出：document.write() 拼接字符串: +

事件：鼠标相关：onclick ；双击ondbclick； 按下鼠标onmousedown； 松开鼠标onmouseup； 鼠标移动到某对象上onmouseover onmousemove；鼠标移动onmousemove；鼠标离开某对象:onmouseout；键盘上某个按钮被按下并释放onkeypress；键盘上某个按钮被按下onkeydown；键盘上某个按钮按下后松开onkeyup

页面相关：图片下载时被用户中断 onabort；页面内容完成时onload；当浏览器的窗口大小改变时；当前页面将被改变时onunload

表单相关：当前元素失去焦点onblur；当前元素失去焦点并且元素的内容发生改变onchange；当某个元素获得焦点时onfocus；表单中的reset属性被激活onreset；一个表单被提交时onsubmit；

滚动时间：onbounce当Marquee内的内容移动至Marquee显示范围之外；onfinish当Marquee元素完成需要显示的内容后；onstart当Marquee元素开始显示内容。

调用：onClick=”javascript:alert(‘单击’)” 调用外部js文件:<script src=”script.js”></script> js文件中不能有<script>和html代码，<script>与</script>之间不能有javascript代码

打开浏览器对js的支持：工具-Internet选项-安全-自定义级别-Java小程序脚本，活动脚本-启用

出现“允许阻止的内容”，永久支持：工具-Internet选项-高级-允许活动内容在我的计算机上的文件中运行，允许来自CD的活动在我的计算机上运行（勾选）

取得当前时间戳time()(默认美国时区,为了获取中国时间:date\_default\_timezone\_set('PRC');)；格式化date(“Y-m-d”,time()) date(“H:i:s”,time()) date("Y-m-d");

date(string format,int timestamp) 几个关于format的预定义常量:p203 格式化参数:a(am或pm) A(AM或PM) B(swatch Internet标准时间) d(月份中的第几天,有前导零的两位数字) D(星期中的第几天,3个字母) F(月份,完整的文本格式) h(12小时格式) H(24小时格式) i(有前导零的分钟数) I(是夏令时返回1,否则0) j(月份中的第几天,没有前导零) l(星期数，完整的文本格式) L(闰年为1,否则为0) m(数字表示的月份) M(3个字母表示的月份) n(数字表示的月份，没有前导零) s(秒数,有前导零) S每月天数后面的英文后缀,两个字母,如th，可以和j连用。 t(指定月份所应有的天数) T(本机所在的时区) w(星期中的第几天,0-6) y(两位数字表示的年份) Y(四位数表示的年份) z(年份中的第几天) Z(时差偏移量的秒数)

array getdate(int timestamp) 其中键:seconds minutes hours mday wday mon year yday weekday month

检查日期有效性bool checkdate(int month, int day, int year)

日期输出形式：setlocale(LC\_ALL, “en\_US”)//不起作用可能是未安装相应的语言包 简体中文“chs” 繁体中文”cht”

%a星期的简写；%A星期的全称；%b月份的简写；%B月份的全称；%C世纪值,年份除以100后取整；%d月份中的第几天；%e月份中的第几天，1位数字前是空格；%H 24小时制的十进制小时数；%I12小时制的十进制小时数；%j年份中的第几天；%m十进制月份；%M十进制分钟数；%n换行符；%p根据给定的时间值为am或pm；%r用a.m和p.m的时间；%R 24小时符号的时间；%S十进制秒数；%t制表符；%T当前时间:和%H:%M:%S一样；%D:和%m/%d/%y一样；%u星期几的十进制表达；%w星期中的第几天，星期天为0；%y没有世纪数的十进制年份；%Y包括世纪数的十进制年份

返回时间戳:strtotime(“now”) strtotime(“21 May 2009”) strtotime(“+3 day”) strtotime(“+1 week”) strtotime(“+1 week 2 days 3 hours 4 seconds”) strtotime(“next Thrusday”) strtotime(“last Monday”) strtotime("10:38pm April 15 2015");

float ceil(float value)返回不小于value的最小整数，返回值是float型，不是整型。

microtime()返回当前时间戳和微秒数，list($msec, $sec)=explode(“ ”, microtime())

若要显示</script>本身，可将<改写为&lt，将> 改写为”&gt”;

时间：int mktime(int hour, int minute, int second, int month, int day, int year, int [is\_dst])

date(“Y-m-d”, mktime()) date(“H:i:s”, mktime()) date(“Y”)

$arr=getdate()//返回日期信息的数组 $arr[weekday]

Checkdate($month, $day, $year)检查日期是否正确

Setlocale(LC\_All,”chs”); strftime(“%A”);得到中文格式的星期几

require：会生成致命错误并停止脚本

include：只生成警告，并且脚本会继续

bool setcookie(string name, string value, int expire, string path, string domain, int secure)；可以通过$\_COOKIE[“cookiename”]调用变量名为cookiename的Cookie；可以通过$\_COOKIE[“values”]获取名为values的值；expire：cookie的失效时间；path：cookie在服务器的有效路径；domain：cookie的有效域名；secure：1：只在HTTPS连接上有效，0：在HTTP和HTTPS连接上均有效。

删除cookie：setcookie(“name”,””,time()-1) 手动：Internet选项-常规-删除cookie

创建会话session\_start() session\_register()

会话变量启动后，全部保存在数组$\_SESSION中

删除会话变量：unset($\_SEEEION[‘user’]); 删除多个会话$\_SESSION=array(); 结束当前会话session\_destroy();

Session失效时间: setcookie(session\_name(),session\_id(),time()+$time,"/");

session\_start() session\_register() $\_SESSION 删除一条会话unset($\_SESSION[‘user’]) 删除多个会话$\_SESSION=array() 结束当前会话session\_destroy() 设置session失效时间：session\_set\_cookie\_params($time)必须在session\_start()之前调用在一些浏览器上会出现问题，所以，一般用setcookie(session\_name(),session\_id(),time()+$time,”/”);

session缓存：session\_cache\_limiter() 缓存时间的设置:session\_cache\_expire()

文件fopen、fclose、读readfile、file、file\_get\_contents、读一行：fgets、fgetss(会过滤html和php标记)；读一个字符fgetc 读取指定长度的数:fread 回到文件开头: rewind 指针偏移到那个位置: fseek 判断是否到文件末尾: feof 返回在打开文件中的当前位置: ftell

写fwrite file\_put\_contents

操作文件: copy rename unlink fileatime filemtime filesize pathinfo realpath stat

目录：opendir closedir scandir mkdir rmdir getcwd chdir disk\_free\_space total\_space readdir rewinddir

锁定文件：flock

上传文件：$\_FILES[filename][name] … move\_uploaded\_file

类中的构造方法：void \_\_construct(…) 析构方法void \_\_destruct(void)自动调用

mail(to,subject,message,headers,parameters)邮件

错误处理: die(…)

function customError($errno, $errstr){ echo "<b>Error:</b> [$errno] $errstr";}//写在类外面 set\_error\_handler("customError");

触发(自定义错误描述)错误: if ($test>1){ trigger\_error("Value must be 1 or below"); }

向指定的文件或远程目的地发送错误记录:error\_log("Error: [$errno] $errstr",1, "someone@example.com","From: webmaster@example.com");

异常捕获: function checkNum($number){ if($number>1){ throw new Exception("Value must be 1 or below"); } return true; } try{ checkNum(2); echo 'If you see this, the number is 1 or below'; }catch(Exception $e) { echo 'Message: ' .$e->getMessage(); }

自定义异常处理类: class customException extends Exception{ public function errorMessage() { $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile().': <b>'.$this->getMessage().'</b> is not a valid E-Mail address'; return $errorMsg; } }

多个异常: try { if(…)throw… if(…)throw… }catch(…){…}catch(…){…}

重新抛出异常: try{ try{if (…)catch(…){throw…}} }catch(…){…}

处理所有未被捕获的异常: set\_exception\_handler('myException');

验证邮箱格式(返回Bool): filter\_var($email, FILTER\_VALIDATE\_EMAIL)

判断$email中是否包含” example”(只会验证@后面的): strpos($email, "example")

验证url输入:filter\_has\_var(INPUT\_GET, "email") filter\_input(INPUT\_GET, "email", FILTER\_VALIDATE\_EMAIL)

过滤: filter\_var($int, FILTER\_VALIDATE\_INT)

判断是否在某个数组内: $var=300; $int\_options = array("options"=>array("min\_range"=>0, "max\_range"=>256) ); filter\_var($var, FILTER\_VALIDATE\_INT, $int\_options)

过滤多个输入: $filters = array("name" => array("filter"=>FILTER\_SANITIZE\_STRING), "age" => array("filter"=>FILTER\_VALIDATE\_INT,"options"=>array("min\_range"=>1,"max\_range"=>120 ) ), "email"=> FILTER\_VALIDATE\_EMAIL, ); $result = filter\_input\_array(INPUT\_GET, $filters);

函数回调过滤; filter\_var($string, FILTER\_CALLBACK, array("options"=>"convertSpace"))其中convertSpace是回调函数

继承：class subclass extends superclass{…}

isset()判断对象是否存在 get\_class()得到类名

父类:parent “::”可以不经实例化调用类中的成员和方法,不能访问私有变量

静态变量static可以不经实例化访问类中的成员

final 类不能被继承，方法不能被重写和覆盖

abstract不能被实例化，方法在子类中完成

接口interface InterfaceName{ function interfaceName1(); function interfaceName2();}

PHP5中“=”传递引用，传值则要“clone”。 clone的时候会自动调用\_\_clone方法

Instanceof判断对象是否属于某个类

function \_\_get($name){} function \_\_set() 定义了这两个方法相当于所有变量都定义了

function \_\_call($method, $parameter){…}当程序试图调用不存在或不可见的成员方法时，PHP会先调用\_\_call()方法来存储方法名和参数。

serialize()序列化对象。 如果有­\_\_sleep()方法，则会先执行该方法并且清除对象返回对象中所有变量的数组，目的是关闭对象可能具有的连接数据库等类似的善后工作。unserialize重新还原一个呗serialize()序列化的对象，\_\_wakeup()回复在序列化中可能丢失的数据库连接及相关工作

\_\_toString()：当使用echo或print输出对象时，将其转化为字符串。echo只能直接接对象

\_\_autoload自动实例化需要使用的类

加密函数crypt($str,$salt\_str)//其中如果不加$salt\_str则2参随机,无解密 md5 sha1

Mcrypt: 例子: $key = "miyao"; $string="jiami"; $crypttext = base64\_encode (mcrypt\_encrypt(MCRYPT\_RIJNDAEL\_256, md5($key), $string, MCRYPT\_MODE\_CBC, md5(md5($key))));//加密 $decryptedtb = mcrypt\_decrypt(MCRYPT\_RIJNDAEL\_256, md5($key), base64\_decode($crypttext), MCRYPT\_MODE\_CBC, md5(md5($key)));//解密

Mhash: mhash\_count、mhash\_get\_hash\_name()、mhash\_get\_block\_size()、mhash(int, string, [string])、 mhash\_keygen\_s2k()

bin2hex()

数据库: create table table\_name (列名1 属性, 列名2 属性)

SHOW [FULL]COLUMNS FROM 数据表名 [FROM 数据库名]; SHOW [FULL]COLUMNS FROM 数据表名.数据库名;

DESCRIBE 数据表名; DESCRIBE 数据表名 列名;

Alter table tb\_admin add email varchar(50) not null, modify user varchar(40);

RENAME TABLE 数据表名1 To 数据表名2;

DROP TABLE 数据表名; drop table if exists 数据表名;

Insert Into 数据表名(column\_name1, column\_name2, …) values (value1, value2, …)

select … from … where … group by … order by … having … limit …

多表查询：select 字段名 from 表1, 表2 …… where 表1.字段=表2.字段 and 其他查询条件

更新记录：update 数据表名 set column\_name=new\_value, column\_name2=new\_value2, … where condition;

delete from 数据表名 where condition

phpMyAdmin导出导入数据库脚本.sql

$link = mysql\_connect(‘hostname’, ‘username’, ‘password’) or die(“…”.mysql\_error());

$db\_selected=mysql\_select\_db(“db\_database18”, $link);

或$db\_selected=mysql\_query(“use db\_database18”, $link);

$result = mysql\_query(“insert into tb\_member values(‘tm’, ‘111’, ‘tm@tmsoft.com’)”, $link);

$result=mysql\_query(“update tb\_member set user=’纯净水’,pwd=’1025’ where user=’tm’”, $link)

$result=mysql\_query(“delete from tb\_member where user=’纯净水’”,$link);

$result=mysql\_query(“select \* from tb\_member”, $link);

$result=mysql\_query(“DESC tb\_member”);

%通配符，$info=mysql\_fetch\_array($sql);返回字典数组； mysql\_fetch\_object()返回对象； $row=mysql\_fetch\_row($sql) mysql\_num\_rows($sql)

ADDNewConnection($databaseType) 持久化PConnect($host, [$user],[$password],[$database])

连接MYSQL数据库的方式： 一$conn=ADONewConnection(‘mysql’); $conn->PConnect(‘localhost’,’root’,’root’,’db\_database19’); 二$conn=ADDNewConnection(‘mysq://root:root@localhost/db\_database19’);

连接Microsoft SQL Server(ODBC)：$conn=ADONewConnection(‘odbc\_mssql’); $conn->PConnect(“Driver={SQL Server}”;Server=localhost;Database=mydb;”,’username’,’password’)

连接Microsoft Access(ODBC): $conn=ADONewConnection(‘access’); $conn->PConnect(“Driver={Microsoft Access Driver (\*.mdb)};Dbq=d:\\mydb.mdb;Uid=Admin;Pwd=;”);

$ADODB\_FETCH\_MODE=ADODB\_FETCH\_NUM;设置结果集以字段序号为索引，$rst->fields[0];

$ADODB\_FETCH\_MODE=ADODB\_FETCH\_ASSOC;以字段作为索引

CatheExecute([$sec],$sql,[$inputarr=false])可将查询结果保存到缓存

SelectLimit($sql,$numrows,$offset,$inputarr)得到的记录数

CatheSelectLimit(…)

CachFlush()清除ADODB数据库的缓存

GetUpdateSQL(&$rs,$arrFields,$forceUpdate)更新记录

GetInsertSQL(&$rs,$arrFields)添加新纪录

DBDate()实现不同数据库之间日期格式的转换； qstr()将双引号转单引号

控制结果集函数：fields当前指向的记录，EOF，GetArray[row]当前记录到row-1记录，UserDate($str,$fmt)将日期格式字符串$str转换为$fmt设置的时间格式，UserTimeStamp($str,$fmt)将时间格式字符串转换为$fmt设置的时间格式，MoveNext()将结果集的指针下移一位，Move($to)将结果集的指针移动到指定位置、MoveFirst()指针移动到第一条数据，MoveLast()指针移动到最后一条数据，FetchRow()返回当前指针指向的记录的数组，FetchField()返回一个对象包含字段名称、空间长度等信息，FetchNextObject()返回当前指针所指向的记录的对象形式并且指针自动下移一行，FieldCount()返回结果集中的字段数，RecordCount()返回结果集中的记录数，CurrentRow()返回当前指针所指的记录序号。

生成HTML表格函数: rs2html($rst, [$table\_attributes], [$col\_titles])

分页函数PageExecute($sql, $nrows, $page)//$sql:SQL查询语句;$nrows:每页显示的记录数;$page:保存当前页数

CachePageExecute($sec,$sql,$nrows,$page)缓存；

AbsolutePage($page)返回当前页数；

AtFirstPage()如果当前页是第一页，返回true；AtLastPage()如果当前页是最后一页,返回true

ADODB\_Pager($conn,$sql,$id,$showPageLinks)是ADODB\_Pager类的构造函数，通过类中的render()函数可以实现一个小巧的分页技巧

ADODB调试：将连接的属性设置为true，当有输出操作时，同时也输出调试信息。error返回最后的状态或出错信息。

创建数据库:CREATE TABLE; DROP TABLE 修改一个字段：RENAME TABLE ‘原字段’ TO ‘新字段’; 删除一个字段：ALTER TABLE ‘表’ DROP ; 插入一个字段:ALTER TABLE ‘表’ ADD ‘新字段’ … AFTER ‘原字段’。 CREATE TABLE ‘test’ (‘id’ int(10) NOT NULL auto\_increment, PRIMARY KEY (id))

SELECT 查询字段 FROM 表名 as A WEHRE 条件 条件:= <> in not in like BETWEEN not BETWEEN and or

group by

order by ‘…’ ASC/desc (大小写不敏感)

limit 0,3

语句顺序:group by>order by>limit

计算:COUNT(…) SELECT count(\*) FROM ‘test ’ WHERE 1

MAX() MIN() AVG() SUM()

Insert into 表名 (字段…,…) values(值…,…) now()当前日期

更新: UPDATE 表名 SET 字段 = 值 WHERE 条件 limit

删除: delete from 表名 where …

mysql\_connect(“主机”, “用户名”, “密码”) mysql\_select\_db(“打开数据库”,连接标识符)

mysql\_query(SQL语句, 连接标识符);

$row=mysql\_fetch\_array()值返回下一行的所有域值，没有行返回false。 $row[‘id’]这里是单引号，不是1前面那个。 mysql\_fetch\_row()与前者区别在于不支持域名作索引。

其他函数:mysql\_num\_rows用于计算查询结果中所得行的数目; mysql\_insert\_id传回最后一次使用INSERT指令的ID; mysql\_tablename取得数据库的名称; mysql\_error返回错误的信息; mysql\_close关闭Mysql链接

php里的页面相互调用：require()与require\_once()程序的最前面。Include()与include\_once()任何位置，出错会提示。

$\_SERVER[“REQUEST\_URI”]; $url=parse\_url($url) $url[‘path’]得到路径

传递文件：form的属性：enctype=”mutipart/form-data”传递文件数据 input的type=”file”

$\_FILES[‘myFile’][‘name’] name，type，size，tmp\_name系统默认的临时文件名，error

将上传后的文件移动到目标位置：move\_uploaded\_file(临时文件，目标位置和文件名)

文件操作函数：fopen(路径和文件名，打开方式R/W/A)，fread(打开的文件，结束位置(字节为单位))，filesize(路径和文件名)，fwrite(路径和文件名，写入的内容)，fclose(路径和文件名) 删除文件unlink() 删除目录mkdir() 读取整个文件file\_get\_cotents(‘doc.txt’,r)

\_\_construct初始化 function \_\_set($n,$v){$this->$n=$v} $n可泛指 function \_\_get($n){return $this->$n}

final方法不能被重载；self::类内部成员；static静态、共享内存，静态方法内部禁止出现非静态内容。访问静态成员只能用self，不能用$this，因为$this涉及到实例化。Const定义常量，定义的常量一般大写，没有$。外部方位静态成员：类名::静态成员。

接口：类中全部为抽象方法；方法前不用加abstract；方法为public；成员属性必须为常量。

interface demo{const NAME=”名”; function fun1();}

class mypc extends root implements demo, demo2{} interface demo3 extends demo{}

判断对象是否来自某个类$thing instanceof A

对象的描述配置：class My {function \_\_tostring(){…}} $p=new My(); echo $p;

对象的异常处理:function \_\_call($n,$v){} 克隆对象:$b=clone $a;

function \_\_autoload($ClassName{ include(“$ClassName.php”) }自动载入

数据库中乱码问题: mysql\_query("SET NAMES GBK”);

调用Smarty: include\_once(“Smarty/Smarty.class.php”) 相关配置：$smarty=new Smarty(); $smarty->config\_dir=”Smarty/Config\_File.class.php”;目录变量 $smarty->caching=false; 是否缓存;$smarty->template\_dir=”./template”模板目录,放缓存文件; $smarty->compile\_dir=”/templates\_c”设置编译目录; $smarty->cache\_dir=”.smarty\_cache”缓存文件夹; 左右边界符:$smarty->left\_delimiter=”{”; $smarty->right\_delimiter=”}”

在线编辑器:fckeditor $sBaseOath=$\_SERVER[‘PHP\_SELF’] $sBasePath=dirname ($sBasePath).’/’ $oFCKeditor=new FCKeditor(‘FCKeditor’) $oFCKeditor->BasePath=$sBasePath $oFCKeditor->Create() oFCKeditor->Value=’初始值

setcookie(‘id’,$\_POST[name]); $\_COOKIE(‘id’)

session 初始化session\_start()必须在最前面 $\_SESSION[name]=value; echo $\_SESSION[name] isset（#\_SESSION[name]） unset($\_SESSION[name]) session\_destroy()

$\_SESSION[id]=$\_POST[name];

随机数函数:rand(min, max) 十进制转十六进制:dechex(十进制数)

$im=imagecreatetruecolor(w,h)创建一定大小的图片；$bg=imagecolorallocate($im,0,0,0)设置背景色；imagestring($im,font,int x,int y,内容,$bg)在图片上添加文字

header(Content-Type: image/gif) imagejpeg($im)

画线函数Imageline(resource image,int x1,int y1,int x2,int y2,int color)

画点函数:imagesetpixl(resource image,int x,int y,int color)

带字体的写入函数:imagettftext(resource image,float size,float angle,int x,int y,int color,string font file,string text)

将文字转换为utf-8 $str=iconv(“gbk”,”UTF-8”,”中文”)

Imagecreatefromgif(‘php.gif’) jpeg png wbmp string

图片合并Imagecopy(水印图片,被打水印的图片,int x,int y,int src\_x,int src\_y(水印图片剪切位置),int src\_w, int src\_h)

图片剪切imagecopyresized(resource dst\_image(新的画板), resource src\_image, int dst\_x, int dst\_y, int src\_x, int src\_y, int dst\_w, int dst\_h, int src\_w, int src\_h)

获取图片信息:Array GetImageSize($image); 1:gif, 2:jpeg, 3:png

生成新的图片Imagejpeg($im,”新的图片.jpg”)，那么这里不要header(…)

preg前缀下，正则”//”或”##”作为开始结束符 […]里的内容只要出现一次就成功 [^xx]匹配除xx以外的内容 \d=[0-9] \D=[^0-9] \w= [a-zA-Z\_0-9] \W=[^a-zA-Z\_0-9] \s=[\f\n\r]

\*, .代替任意字符 + ? | ^ $ \b \B {m} {m,} {m,n} ()合并整体匹配，并放入内存，可使用\1\2…依次获取

优先级() > 重复匹配内容\* ? + {} > 边界^ $ \b > 条件|

模式修正符： /正则/i i不区分大小写, m匹配首内容或尾内容采用多行匹配(每一行都可做匹配), S将转义回车取消(\n两边可以连接,结果回车处是空格) x忽略正则中的空格 A强制从头开始匹配(针对多行) D强制$匹配尾部无任何内容(最后的\n不会被忽略) U禁止贪婪匹配，匹配到最近的一个匹配符并结束 ie结合()可对替换值做操作

匹配preg\_match\_all(string pattern,string subject,array matches)

替换preg\_replace(mixed pattern, mixed replacement, mixed subject, int limit)pattern和replacement可以是数组，

分割preg\_split(string pattern,string subject,int limit,int flags)

%通配符: SELECT \* FROM table WHERE title LIKE ‘PHP%’ or/and title LIKE ‘%教程%

发邮件:smtp p第42讲 $smtp=new smtp(“smtp.163.com”,”25”,true,”info263”,$pw); $smtp->sendmail([info263@163.com](mailto:info263@163.com), [info263@163.com](mailto:info263@163.com)(发送方),”标题”,”内容”,”HTML”);

第43讲：ThinkPHP.php

输出表格: header(“Content-type:application/vnd.ms-excel”);

header(“Content-Disposition:filename=php100.xls”)

ajax技术: js: if (window.ActiveXObject){xmlHttp=new ActiveXObjec(‘Microsoft.XMLHTTP’);}else if(window.XMLHttpRequest){xmlHttp=new XMLHttpRequest();}

xmlHttp.open(“GET”,”for.php?id=”+url,true); xmlHttp.onreadystatechange=byphp; XmlHttp.send(null); function byphp(){var byphp100=xmlHttp.responseText; document.getElementById(‘php100’).innerHTML=byphp100; }

index.php: <a href=”#” onclick=”funphp100(‘o’)”>o</a> <div id=”php100”></div>

XHR.readyState==0未初始化,1建立,2发送,3处理中,4响应完成

事件：鼠标相关：onclick ；双击ondbclick； 按下鼠标onmousedown； 松开鼠标onmouseup； 鼠标移动到某对象上onmouseover onmousemove；鼠标移动onmousemove；鼠标离开某对象:onmouseout；键盘上某个按钮被按下并释放onkeypress；键盘上某个按钮被按下onkeydown；键盘上某个按钮按下后松开onkeyup

页面相关：图片下载时被用户中断 onabort；页面内容完成时onload；当浏览器的窗口大小改变时；当前页面将被改变时onunload

表单相关：当前元素失去焦点onblur；当前元素失去焦点并且元素的内容发生改变onchange；当某个元素获得焦点时onfocus；表单中的reset属性被激活onreset；一个表单被提交时onsubmit；

滚动时间：onbounce当Marquee内的内容移动至Marquee显示范围之外；onfinish当Marquee元素完成需要显示的内容后；onstart当Marquee元素开始显示内容。

过滤：用正则check或addslashes

拟静态:在apache的httpd.conf(去掉LoadModule rewrite\_module前的#)或.htaccess中: (第 54 讲) RewriteEngine on RewriteRule ^/t\_(.\*).html$ /test.php?id=$1

PDO连接数据库: new PDO(“mysql:host=localhost;dbname=php100”,”root”,””); 长连接:new PDO(“连接信息”,’root’,’’,array(PDO::ATTR\_PERSISTENT=>true));

PDO::query() PDO::exec()针对无需返回结果,如INSERT,UPDATE PDO::lastInsertId()最后插入的ID PDOStatement::fetch()获取一条记录 PDOStatement::fetchAll()

配置:在php.ini中去掉pdo前的;

Is\_writable(…) is\_readable(…)

$tq=mysql\_list\_table($dbname);等同于qurey查询所有表 mysql\_fetch\_field($query)

BIWEB 第58讲

sqlite\_open(‘xxx.db’,模式,错误信息) sqlite\_query(数据库连接参数,SQL参数) sqlite\_num\_rows(SQL返回句柄) sqlite\_last\_insert\_rowid(数据库连接参数) sqlite\_fetch\_array(SQL返回句柄,返回索引) sqlite\_exec()返回Bool值

高级缓存MemCached第73讲 $mem=new Memcache $mem->connect(‘127.0.0.1’,11211) or die(‘连接失败’); $mem->getVersion; $mem->set(‘Key’,’Value’,’Zip’,’Time’) $val=$memcache->get(‘Key’) cd ..->cd \->cd memcached

Zend framework第75讲 MVC

解压: $dir=getcwd(); $obj=new com(“wscript.shell”) $obj->run(“winrar x $dir\\”.$nname.” “.$dir, 1, true) win x 被解压文件 加压的位置

ADODB PHP万能数据库引擎：include\_once(“adodb5/adodb.inc.php”); $db=NewADOConnection(‘mysql’); $db->Connect(“localhost”,”root”,”pwd”,”dbname”)or die (“错误”) Execute FetchRow $db->AutoExecute(“up\_admin\_user”,$arr,”INSERT”)

地址方式:$db=NewADOConnnection(‘mysql://root:pwd@localhost/upload’); $db->SetFetchMode(ADODB\_FETCH\_ASSOC);

自动分页:include\_once(“adodb5/adodb-pager.inc.php”) session\_start(); $pager=new ADODB\_Pager($db,$sql); $pager->Render(3);

对象方式:$PConnect(“localhost”,”root”,””,”upload”); $row=$rs2->FetchNextObject() $row->NAME;

html方式:include\_once(“adodb5/tohtml.inc.php”); echo rs2html($rs2);

echo “<script>location.href=’vi.php?id=1’</script>”

jpgraph第85讲：require\_once('jpgraph/jpgraph.php'); require\_once('jpgraph/jpgraph\_pie.php');

$data = array(40,60,21,33,12,33); $graph = new PieGraph(400,400); $graph->SetShadow();

$graph->title->Set(":'sand' Theme"); $graph->title->SetFont(FF\_SIMSUN,FS\_BOLD,24); $p1 = new PiePlot($data); $p1->SetTheme("sand"); $p1->SetCenter(0.5,0.55); $p1->value->Show(false); $graph->Add($p1); $graph->Stroke();//CI框架中怎么用???lymark

curl处理需要登录才能抓取的:$curl=curl\_init(); curl\_setopt($curl,CURLOPT\_URL,”http://...”);

curl\_setopt($curl,CURLOPT\_RETURNTRANSFER,1); $data=curl\_exec($curl);

$cookie\_file=tempnam(‘./temp’,’cookie’); curl\_setopt($curl,CURLOPT\_COOKIEJAR,$cookie\_file);

xml解析: xml\_set\_element\_handler($parser,"start","stop"); xml\_set\_character\_data\_handler($parser,"char"); while ($data=fread($fp,4096)){ xml\_parse($parser,$data,feof($fp)) or die (sprintf("XML Error: %s at line %d", xml\_error\_string(xml\_get\_error\_code($parser)), xml\_get\_current\_line\_number($parser))); }

调用xml: $doc=new DOMDocument(); $title=$doc->load(‘books.xml’); $->getElementsByTagName(‘book’); echo $title->item(0)->nodeValue; $>getAttribute(‘id’)

生成xml: $doc=new DOMDocument(‘1.0’,’utf-8’); $doc->formatOutout=true; 创建标签:$root=$doc->createElement(‘root’); 创建属性$id=$doc->creatAttribute(‘id’); $newsid=$doc->createTextNode(“1”);继承:$index->appendChild($id); $doc->save(“php100.xml”)

获取根标签:$root=$doc->documentElement; 移除标签$root->removeChild($book);

Xdebug第101讲

<script type=text/javascript src=”a.js”><!-- … //--></script>

document.write(“输出<br>内容”); alert(n); confirm(n); prompt(n,v);

this.value当前标签的值

JQuery:

$(document).ready(function()){ $(this).hide(); $(“p”).hide(); $(.”test”).hide();//class $(#test).hide();//id?});

Jquery第107讲：选择器$(“p”) 事件器：$(selector).click() Dom节点$(“a”).addClass(“red”); hover执行完一个再执行第二个 toggle点击执行第一个，再点击执行第二个。

Ajax技术: $.ajax({ type:”POST”, [url:”php.php](url:)”, data:{name:”12345”}, success:function(data,st){$(“div”).html(data)} });

$.post(“php.php”,{name:”12345”},function(data){$(“div”).html(data)})

$(“div”).load(“php.php”,{name:”12345”})

下拉框左右移动：$(“#select1 option:selected”).appendTo(“#select2”);

$(“#select1”).dblclick(function(){ $(“#select1 option:selected”).appendTo(“#select2”) });

$(“tbody>tr:has(:checked)”).addClass(“ed”); $(this).hasClass(‘ed’);

$(this).[hashed?removeClass:addClass](‘ed’).find(“:input”).attr(“checked”,!hashed);

$(this).html(“<input id=’edit”+infd+inid+”’ value=’”+value+”’>”);

$(“#edit”+infd+inid).focus().live(“blur”,function(){ $(this).parents(“td”).html($(this).val()) })

CI框架：

URI 分段： example.com/class/function/ID

1.第一段表示要调用的控制器 类 ；

2.第二段表示要调用的类中的 函数 或 方法 ；

3.第三段以及后面的段代表传给控制器的参数，如 ID 或其他任何变量；

移除 URL 中的 index.php

在根目录创建.htaccess，注意不是将application里的移到根目录，是直接新建，添加RewriteEngine on RewriteCond $1 !^(index\.php|robots\.txt) RewriteRule ^(.\*)$ /index.php?/$1 [L]。结果然后加或不加index.php都可以

添加 URL 后缀

config.php中$config['url\_suffix'] = '.html';

启用查询字符串格式的 URL: $config['enable\_query\_strings'] = FALSE;改为TRUE即可使用,如若此做，则不能使用CI带的URL库

控制器:放在controller下的.php文件，继承CI\_Controller，首字母大写

方法:第二段，紧接控制器后面

参数:紧接方法后面，多个参数用/分隔。注意是否使用了URI路由

定义默认控制器，首页: $route['default\_controller'] = 'pages/view';其中Pages.php的view方法中路径要正确并存在

重映射:原有的一些方法都不会执行public function \_remap($method, $params = array()){…}比如<http://localhost/news/2312/adf/23>，$params是Array ( [0] => adf [1] => 23 ) $method是2312

输出类: public function \_output($output){…}其中$output即是原本要输出的页面内容。处理缓存: if ($this->output->cache\_expiration > 0){$this->output->\_write\_cache($output);}

设置输出:$this->output->set\_output($data); set\_content\_type('application/json') 'text/plain' 'jpeg' set\_content\_type('css', 'utf-8') get\_content\_type() get\_header('content-type') get\_output() append\_output($data) set\_header($data) set\_status\_header(401) enable\_profiler(TRUE) 启用分析器 \_display()发送最终输出结果以及服务器的HTTP头到浏览器

私有方法: private 或 protected或方法名前加下划线前缀

控制器放入子目录中:URI应该像example.com/index.php/products/shoes/show，但这时若controller下同时有products.php会报错

加载视图:$this->load->view('name');

向视图加动态数据: $data = array('title' => 'My Title', 'heading' => 'My Heading', 'message' => 'My Message' ); $this->load->view('blogview', $data); blogview.php中调用:<?php echo $title;?>

视图作为数据返回(不会直接加载): $string = $this->load->view('myfile', '', TRUE);其中默认是FALSE

模型public function \_\_construct(){parent::\_\_construct(); $this->load->model('model\_name');//加载}

查$query = $this->db->get('entries', 10); return $query->result();或

增: $this->title = $\_POST['title']; $this->input->post('title') $this->db->insert('entries', $this);

改: $this->db->update('entries', $this, array('id' => $\_POST['id']));

访问: $this->model\_name->method(); 将你的模型对象赋值给一个不同名字的对象: $this->load->model('model\_name', 'foobar');然后调用: $this->foobar->method();

模型的自动加载？

模型连接数据库:$this->load->model('model\_name', '', TRUE);

传数据库连接配置: $config['hostname'] = 'localhost';.. $this->load->model('model\_name', '', $config);

加载多个辅助函数$this->load->helper(array('helper1', 'helper2', 'helper3') );

用辅助函数创建链接echo anchor('http://localhost/upload', 'Click Here');

扩展辅助函数:创建application/helpers/MY\_array\_helper.php $config['subclass\_prefix'] = 'MY\_';

自动加载辅助函数:在config/autoload.php中$autoload['helper'] = array('url', 'file');

系统辅助函数的位置在.system/helpers

辅助函数清单:http://codeigniter.org.cn/user\_guide/helpers/index.html

初始化系统类库$this->load->library('form\_validation');

初始化类时传入参数: $params = array('type' => 'large', 'color' => 'red'); $this->load->library('someclass', $params); 那么定义时class Someclass {public function \_\_construct($params) { } }

访问config: $this->config->item('base\_url');(类的方法中不能这么用)

类中使用CodeIgniter: $CI =& get\_instance(); $CI->load->helper('url'); $CI->load->library('session'); $CI->config->item('base\_url');

替换原生类库：application/libraries/Email.php class CI\_Email { }

继承原生类库：class MY\_Email extends CI\_Email { }

驱动器(父类可以访问子类): $this->load->driver('some\_parent'); $this->some\_parent->some\_method(); $this->some\_parent->child\_one->some\_method();

创建驱动器?

继承核心类class MY\_Input extends CI\_Input{…} application/core/ MY\_Input.php

替换核心类class CI\_Input{…} application/core/ CI \_Input.php

钩子?

自动加载资源: $config['composer\_autoload'] = TRUE;(不同每次手动load?)

公共函数：不需要加载类库。is\_php('5.3'):版本是否大于等于5.3 is\_really\_writable($file)文件只读时返回FALSE config\_item($key)配置信息 set\_status\_header($code,$text)设置状态码对应的描述文本<https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html(状态码清单>) remove\_invisible\_characters('Java\\0script'); 防止在 ASCII 字符串中插入空字符 html\_escape($var) 类似于PHP原生的htmlspecialchars()函数，只是它除了可以接受字符串参数外，还可以接受数组参数 get\_mimes()返回application/config/mimes.php文件中定义的 MIME数组的引用 is\_https()在使用HTTPS安全连接时返回TRUE，没有使用HTTPS（包括非HTTP的请求）则返回FALSE is\_cli()当程序在命令行下运行时返回TRUE，反之返回FALSE。 function\_usable($function\_name)检查一个函数是否可用，可用返回TRUE，否则返回FALSE。

兼容性函数?

路由：$route['product/(:num)'] = 'catalog/product\_lookup\_by\_id/$1';其中(:num)数字或(:any)字符

用正则：$route['products/([a-z]+)/(\d+)'] = '$1/id\_$2';

回调函数：$route['products/([a-zA-Z]+)/edit/(\d+)'] = function ($product\_type, $id){return 'catalog/product\_edit/' . strtolower($product\_type) . '/' . $id;};

HTTP动词：$route['products/(:num)']['DELETE'] = 'product/delete/$1'; 当发送 DELETE 请求到第一段为 "products" ，第二段为数字这个 URL时，将会调用 Product::delete() 方法。HTTP动词可以使用标准的 HTTP 动词（GET、PUT、POST、DELETE、PATCH），也可以使用自定义的动词

错误页面的路由：$route['404\_override'] = ‘…’;当URI不存在时默认的页面，覆盖原来的404

'-'转'\_'： $route['translate\_uri\_dashes'] = FALSE;

生成错误信息:show\_error($message, $status\_code, $heading = 'An Error Was Encountered'),错误模板在application/views/errors/html/error\_general.php或application/views/errors/cli/error\_general.php

show\_404($page = '', $log\_error = TRUE)模板在application/views/errors/html/error\_404.php

向日志文件写入信息：log\_message($level, $message)第一个参数：'error', 'debug' or 'info'

开启缓存:$this->output->cache($n);其中$n是缓存的时间，会在application/cache生成新文件

分析器:页面任意位置加$this->output->enable\_profiler(TRUE); 分析器类会在页面下方显示基准测试结果，运行过的SQL语句，以及$\_POST数据。禁用某个字段$config['queries'] = FALSE;在application/config/profiler.php中。或$this->output->set\_profiler\_sections($sections);其中$sectionsarray('config' => TRUE,'queries' => TRUE);

命令行：php index.php tools message "John Smith"其中Tool.php在controller目录下含message方法

重命名应用程序目录：根目录index.php中:$application\_folder = '/path/application';

处理多环境?

短标记: <?php echo $variable; ?>等同于<?=$variable?> 控制结构<?php foreach ($todo as $item): ?><li><?=$item?></li><?php endforeach; ?> 同理if、for、while

类库：

基准测试类：

标记时间点$this->benchmark->mark('code\_start'); $this->benchmark->mark('code\_end'); echo $this->benchmark->elapsed\_time('code\_start', 'code\_end');最后一条，不传参则是总时间，同{elapsed\_time}

内存占用：<?php echo $this->benchmark->memory\_usage();?>或{memory\_usage}

缓存驱动器：$this->load->driver('cache', array('adapter' => 'apc', 'backup' => 'file')); if ( ! $foo = $this->cache->get('foo')) {echo 'Saving to the cache!<br />'; $foo = 'foobarbaz!'; $this->cache->save('foo', $foo, 300); }echo $foo;方法: get、delete、save、increment(对缓存中的值执行原子自增操作)、decrement(对缓存中的值执行原子自减操作)、clean、cache\_info()、get\_metadata($id)

其他驱动器？

日历类：$this->load->library('calendar'); echo $this->calendar->generate(2006,6);

$data = array(3=>'http://… ',27=>'http://… ') echo $this->calendar->generate(2006,6,$data);3和27会成为链接

$prefs = array('start\_day'=> 'saturday', 'month\_type'=> 'long', 'day\_type'=> 'short' );表示从星期六开始，使用用完整的月份标题和缩写的天数格式。其他:'show\_next\_prev'=>TRUE, 'next\_prev\_url'=>'http://…'显示上一月/下一月链接,'show\_other\_days'=>TRUE显示邻月日期

日历样式$prefs['template'] = array('table\_open'=> '<table class="calendar">','cal\_cell\_start' => '<td class="day">','cal\_cell\_start\_today' => '<td class="today">');

返回月份的名称get\_month\_name($month)，返回星期的名称get\_day\_names($day\_type = '')

日期调整print\_r($this->calendar->adjust\_date(13,2014)); 指定月的天数get\_total\_days($month, $year)

获取配置信息： echo $config->item('language');

设置配置项：$CI ->config->set\_item('item\_name', 'item\_value');

slash\_item($item)在获取的配置项后加/

site\_url()返回网站的URL，包括index。base\_url()返回根URL。system\_url()返回system 目录的 URL

邮件：$this->load->library('email'); $this->email->from('your@example.com', 'Your Name'); 其他详细见<http://codeigniter.org.cn/user_guide/libraries/email.html>的评论

文件上传类：$config['upload\_path']='./uploads/'; $config['allowed\_types'] ='gif|jpg|png'; $config['max\_size'] = 100; $config['max\_width'] = 1024; $config['max\_height'] = 768; $this->load->library('upload', $config);返回能否上传(Bool)$this->upload->do\_upload('userfile')其中userfile是表单名 $this->upload->display\_errors() 上传成功的返回信息$this->upload->data() 表单中:form\_open\_multipart('upload/do\_upload')

表单验证: $this->form\_validation->set\_rules('username', '用户名', 'required', array('required' => 'You must provide a %s.',));

或

$configArr = array(array('field' => 'username', 'label' => 'Username', 'rules' => 'required'), array('field' => 'password', 'label' => 'Password', 'rules' => 'required', 'errors' =>array('required' => 'You must provide a %s.',)) $this->form\_validation->set\_rules($configArr);

或

$this->form\_validation->set\_rules('username', 'Username', 'trim|required|min\_length[5]| max\_length[12]|is\_unique[users.username]', array('required'=> 'You have not provided %s.', 'is\_unique' => 'This %s already exists.') );其中trim为预处理，去掉两端空白

或 最后接验证函数'callback\_username\_check[sec]' 其中public function username\_check($str,$sec) { $this->form\_validation->set\_message('username\_check', 'The {field} field can not be …');}其中sec作为第二个参数

或 最后直接匿名函数$this->form\_validation->set\_rules('username', '用户名', array (function($str){echo $str;}));

自定义错误样式set\_message('rule', 'Error Message'); set\_rules('field\_name', 'Field Label', 'rule1|rule2|rule3',array('rule2' => 'Error Message on rule2 for this field\_name'));

$this->form\_validation->set\_error\_delimiters('<div class="error">', '</div>');或<?php echo validation\_errors('<div class="error">', '</div>'); ?>

单独定义的错误<?php echo form\_error('password', '<div class="error">', '</div>'); ?>

其中错误边界符的给定$config['error\_prefix'] = '<div class="error\_prefix">'; $config ['error\_suffix'] = '</div>';

自动加载验证application/config/form\_validation.php中$config = array(array('field' => 'username','label' => 'Username','rules' => 'required'));

验证某个数组$data = array('username' => 'johndoe'); $this->form\_validation->set\_data($data);

调用规则集:$this->form\_validation->run('signup') 其中$config = array('signup' => array(…))

关联规则集: $config = array('member/signup' =>…)其中Member类含signup方法并这样调用$this->form\_validation->run()

原生验证规则: required；matches；regex\_match；differs；is\_unique；min\_length；max\_length；exact\_length；exact\_length；greater\_than；greater\_than\_equal\_to；less\_than；less\_than\_equal\_to；in\_list[red,blue,green]表单元素值是否在list内；alpha如果表单元素值包含除字母以外的其他字符，返回FALSE；alpha\_numeric字母和数字；alpha\_numeric\_spaces字母、数字和空格、alpha\_dash字母/数字/下划线/破折号；numeric；integer；decimal；is\_natural自然数；is\_natural\_no\_zero非零自然数；valid\_url；valid\_email；valid\_emails地址之间用逗号分割；valid\_ip

FTP类？服务器?

图像处理$config['image\_library'] = 'gd2';$config['source\_image'] = './uploads/icon\_discover.png'; $config['create\_thumb'] = TRUE创建缩略图;$config['maintain\_ratio'] = TRUE;$config['width'] = 75; $config['height'] = 50;$this->load->library('image\_lib', $config);$this->image\_lib->resize();可选/其他:$config['new\_image'] = '/path/to/new\_image.jpg'新图路径; $config['image\_library'] = 'imagemagick';$config['x\_axis'] = 100; $config['y\_axis'] = 40;$this->image\_lib->crop()按起始坐标裁剪; 旋转?: $config['image\_library'] = 'netpbm';$config['rotation\_angle'] = '90'(选项:'90', '180', '270', 'hor', 'vrt'); 水印?

过滤:$this->input->post('something');(选项:post,get,cookie,server,post\_get,get\_post,ip\_address, valid\_ip,user\_agent,request\_headers, get\_request\_header, is\_ajax\_request, is\_cli\_request, method)

读取php://input流中的数据: $this->input->raw\_input\_stream; $this->input->input\_stream('key');

js库: $this->load->library('javascript/jquery', FALSE); $this->jquery->hide(target, optional speed, optional extra information); 其中speed可选，可为slow、normal、fast或毫秒数, extra information可选，可以包含一个回调

hide;show;toggle;animate;click;toggleClass,fadeIn,fadeout,slideUp,slideDown,slideToggle, corner

语言类?

迁移类?

分页类(//不好用): $this->load->library('pagination'); $config['base\_url'] = 'http://…'; $config['total\_rows'] = 200; $config['per\_page'] = 20; $this->pagination->initialize($config); echo $this->pagination->create\_links();

其他自定义: 自动检测你URI的哪一段包含页数$config['uri\_segment'] = 3; 你当前页码“数字”链接的数量$config['num\_links'] = 2; 链接将自动地被重写成查询字符串格式$config['page\_query\_string'] = TRUE; 给路径添加一个自定义前缀$config['prefix'] = ''; $config['attributes'] = array('class' => 'myclass');

模板解析类: $this->load->library('parser');$data = array('blog\_title' => 'My Blog Title', 'blog\_heading' => 'My Blog Heading');$this->parser->parse('blog\_template', $data);其中blog\_template.php中这样调用{blog\_title} {blog\_heading}

数组{blog\_entries}<h5>{title}</h5><p>{body}</p>{/blog\_entries}

$template = 'Hello, {firstname} {lastname}';$data = array('title' => 'Mr','firstname' => 'John',

'lastname' => 'Doe');$this->parser->parse\_string($template, $data);

$template = '<ul>{menuitems}<li><a href="{link}">{title}</a></li>{/menuitems}</ul>';

$data = array('menuitems' => array(array('title' => 'First Link', 'link' => '/first'),array('title' => 'Second Link', 'link' => '/second'),));$this->parser->parse\_string($template, $data);

session类: $\_SESSION['item']或$this->session->item或$this->session->userdata('item'); 写入session: $this->session->set\_userdata($array); 是否含某个session: isset ($\_SESSION['some\_name'])或$this->session->has\_userdata('some\_name'); 删除session: unset($\_SESSION['some\_name'],$\_SESSION['another\_name']);或$this->session-> unset\_userdata($array\_items);

标记flashdata(只对下一次请求有效, 之后将会自动被清除): $this->session->mark\_as\_flash('item');或$this->session->set\_flashdata('item', 'value');读取flashdata: $this->session->flashdata('item');

tempdata(有有效时间): $\_SESSION['item'] = 'value'; $this->session->mark\_as\_temp('item', 300);或$this->session->set\_tempdata('item', 'value', 300);读取: $this->session->tempdata ('item');或$\_SESSION['item'] 删除: unset($\_SESSION['item']);或$this->session-> unset\_tempdata('item'); 清除: $this->session->sess\_destroy();

表格类: $this->load->library('table');$data = array(array('Name', 'Color', 'Size'),array('Fred', 'Blue', 'Small'),); 用数据库数据: $query = $this->db->get('news'); echo $this->table-> generate($query); set\_heading('Name', 'Color', 'Size'); add\_row('Fred', 'Blue', 'Small'); 表格样式: $this->table->set\_template($template);

一维源数据$list = array('one', 'two', 'three', 'four', 'five'); $new\_list = $this->table-> make\_columns($list, 3); echo $this->table->generate($new\_list);

引用通告类?

单元测试类: $this->unit->run('test', 'expected result', 'test name', 'notes');其中test是测试目标，expected result是期望的结果(可以判断类型，如is\_object、is\_string等)，test name是测试名，notes是备注。

生成报告$this->unit->report(); 生成数组$this->unit->result()

启用严格匹配(值和类型)$this->unit->use\_strict(TRUE);

禁用单元测试$this->unit->active(FALSE);

自定义显示测试结果$this->unit->set\_test\_items(array('test\_name', 'result'));

测试结果显示的模板$str = '<table>{rows}<tr><td>{item}</td><td>{result}</td></tr> {/rows} </table>'; $this->unit->set\_template($str);

URI类: $product\_id = $this->uri->segment(2, 'aa'); 从URI中获取指定段, 第二个参数为所请求的段不存在时的返回值

前后加斜线:$this->uri->slash\_segment(3, 'leading'); 'trailing' 'both'

uri转数组: uri\_to\_assoc(3) 数组转uri: assoc\_to\_uri($array)

返回相对的uri 字符串:uri\_string() 返回总段数total\_segments() segment数组: segment\_array()

用户代理类: $this->load->library('user\_agent'); is\_browser browser version is\_robot robot is\_mobile mobile platform agent\_string()

XML-RPC类?

zip编码类: $name = 'mydata1.txt'; $data = 'A Data String!'; $this->load->library('zip'); $this->zip->add\_data($name, $data); $this->zip->archive('./uploads/my\_backup.zip');

compression\_level(0到9) add\_dir('myfolder') add\_data($arr)(其中$arr是数组); read\_file($path)读取文件 read\_dir get\_zip clear\_data

数据库类:

$this->load->database(); $query = $this->db->query(数据库语句) 多结果foreach ($query->result() as $row){…$row->title…} $query->num\_rows() $query->num\_fields()字段数 或foreach ($query->result\_array() as $row){… $row['title']…} 单结果: $row = $query->row();$row->name或$row = $query->row\_array(); $row['name']; $query->row(5)特定行的数据 $query->row(0, 'User');其中User为已加载的某个类的类名

$row = $query->first\_row() last\_row next\_row previous\_row可选参数’array’

$sql = "INSERT INTO news (title) VALUES ('".$str."')";(注意value的写法), $this->db->query($sql);

$this->db->get('table\_name'); $this->db->insert('mytable', $array); $array为数组或对象

$query->free\_result()释放内存

数据库配置:

故障转移: $db['default']['failover'] = array(array(…),array(…))

测试环境: $db['test']=array(…) $active\_group = 'test';

自动连接数据库：在config/autoload.php中：$autoload['libraries'] = array('database');

reconnect close

自动转义(会检测数据类型，仅转义字符串类型的数据，如自动加单引号等)db->escape；

查询绑定$sql = "SELECT \* FROM some\_table WHERE id IN ? AND status = ? AND author = ?"; $this->db->query($sql, array(array(3, 6), 'live', 'Rick'));所有值会被自动转义

错误：$this->db->error()

foreach ($query->result('User') as $user){…}其中User为类名，这个类已经加载

一条条读取,省内存while ($row = $query->unbuffered\_row()){…} 可选参数’array’,’object’

$query->data\_seek($n);指针指向第$n+1行，返回Bool，与unbuffered\_row()一起使用，默认值是0

$query->custom\_row\_object(0, 'User');其中0是row number，User是类名

num\_rows，num\_fields，free\_result

返回新插入行的id：$this->db->insert\_id() 返回受影响的行数$this->db->affected\_rows() 返回上一次的查询语句$this->db->last\_query() 返回数据表的总行数$this->db->count\_all('my\_table'); 返回数据库平台:$this->db->platform(); 返回数据库版本$this->db->version(); 返回SQL插入语句: $this->db->insert\_string('news', $arr);其中$arr = array('title' => '$name'); 返回SQL更新语句: $this->db->update\_string('news', $array, $where);其中$where = "id = 1";

$query = $this->db->get('news',$m,$n);取出从第$n+1个开始的共$m个元素

get\_compiled\_select同get但不执行，返回SQL语句,但会对后面的数据库操作起作用。带limit的查询： $this->db->limit(10,20)->get\_compiled\_select('mytable', FALSE); $query = $this->db->select ('title')->get\_compiled\_select();其中FALSE表示不会重置查询,重置: $this->db->reset\_query()

$query = $this->db->get\_where('news', array('id' => 7));

$this->db->select\_max('id','new\_id'); $query = $this->db->get('news');返回最大值，其中’new\_id’可选，重命名结果字段。select\_min、select\_avg、select\_sum同理

this->db->from

JOIN关联: $this->db->from('news'); $this->db->join('blogs', 'news.id = blogs.tag'); $query = $this->db->get();

返回条件查询语句$this->db->where('title', 'title')->get\_compiled\_select(); where('name !=', $name) where('id <', $id) where($array) $where = "name='Joe' AND status='boss'"; $this->db->where($where); or\_where('id >', $id);

返回WHERE IN字句:$names = array('Frank', 'Todd'); $this->db->where\_in('username', $names)->get\_compiled\_select() or\_where\_not\_in同where\_in

模糊匹配: $this->db->like('title', 'match', 'before');同`title` LIKE '%match%' 第三个参数可选'before'、'after'、'both',默认both $this->db->like($array); or\_like not\_like or\_not\_like

group\_by($arr) 添加DISTINCT 关键字：$this->db->distinct()

生成HAVING字句: $this->db->having('user\_id = 45');或having('user\_id', 45); $this->db->having(array('title =' => 'My Title', 'id <' => $id));

WHERE和HAVING区别: WHERE是约束，先于聚合语句；HAVING是过滤，后于聚合语句

having的第三个参数Bool，默认TRUE(转义)

排序: order\_by('title', 'DESC');第二个参数'ASC'、'DESC'、'RANDOM'；order\_by('title DESC, name ASC'); 当'RANDOM'第一个参数被忽略，但可以传入数字作seed

获取特定查询返回结果的数量$this->db->count\_all\_results();会重置设置的所有值, 可以将 第二个参数设置为FALSE来保留count\_all\_results('my\_table', FALSE);

某个表的总行数: count\_all('my\_table');

查询条件组group\_start()和group\_end(): $this->db->select('\*')->from('my\_table')-> group\_start()->where('a', 'a')->or\_group\_start()->where('b', 'b')->where('c', 'c') -> group\_end()->group\_end()->where ('d', 'd')->get(); not\_group\_start or\_not\_group\_start

生成插入语句: $this->db->set($array)->get\_compiled\_insert('mytable');或set('title', 'My Title') get\_compiled\_insert('mytable',FALSE)其中FALSE表示不会重置 set的第三个参数若为FALSE表示不自动转义

多个插入: $data = array(array(…),array(…)); $this->db->insert\_batch('mytable', $data);

更新: $this->db->replace('table', $array);若$array中没有主键则插入

set可以用于对象db->set($object)

按条件更新$this->db->where('id',4); $this->db->update('news', $data);

$data = array(array('title' => 'My title' ,'name' => 'My Name 2' ,'date' => 'My date 2')); $this->db->update\_batch('mytable', $data, 'title');按'title'更新

get\_compiled\_update

删: $this->db->delete('mytable', array('id' => $id)); 或where('id', $id); delete('mytable');

多个表删: $tables = array('table1', 'table2'); where('id', '5'); delete($tables);

删除表中的数据: db->empty\_table('mytable');或$this->db->truncate('mytable');

生成删除语句: get\_compiled\_delete()

链式方法: $query=$this->db->select('title')->where('id', $id)->limit(10, 20)->get('mytable');

开启缓存$this->db->start\_cache() 停止缓存stop\_cache() 清空缓存flush\_cache()

事务:(通过rollback回滚控制，不满足条件时，整个事务区间都不执行,影响的主要是非查询操作)

事务开始: $this->db->trans\_start(); 事务完成: $this->db->trans\_complete();若有判断并执行rollback或trans\_commit的语句，写在start和comlete之间

禁用严格模式: $this->db->trans\_strict(FALSE);

错误处理: if ($this->db->trans\_status() === FALSE){… trans\_rollback(); …}else{… trans\_commit(); …}

禁用事务: trans\_off(); 测试模式(所有查询都被回滚): trans\_start(TRUE);

手工运行事务: $this->db->trans\_begin();

返回数据库的所有表名称的数组:$this->db->list\_tables(); 表存在table\_exists('table\_name')

返回字表中字段名的数组: $this->db->list\_fields('table\_name'); 字段存在: field\_exists('field\_name', 'table\_name')

返回字段信息组成的数组; $this->db->field\_data('news'); 或$query->field\_data();

$this->db->call\_function('some\_function', $param1, $param2, etc..); CodeIgniter 中没有定义的 PHP 数据库函数，如get\_client\_info

启用缓存: application/config/database.php中，cachedir/ cache\_set\_path设置路径，cache\_on设置TRUE,手动缓存(catch\_on为FALSE):$this->db->cache\_on();要缓存的内容 $this->db->cache\_off();删除缓存: $this->db->cache\_delete('blog', 'comments');其中缓存目录是…cachedir/blog/comments只是删了文件，但文件夹还在,其中默认blog是类名，comments是函数名

清除所有缓存文件(包括文件夹)$this->db->cache\_delete\_all();

数据库工厂类:

加载数据库工厂类: $this->load->dbforge()

创建数据库: $this->dbforge->create\_database('my\_db');返回Bool 删除: drop\_database

创建表: $fields = array('users' => array('type' => 'VARCHAR','constraint' => '100',),); 添加字段$this->dbforge->add\_field($fields); $this->dbforge->create\_table('table\_name');按先添加字段再创建表的顺序。其他键值对: unsigned/true, default/value, null/true, auto\_increment/true, unique/true

create\_table('table\_name', TRUE);第二个参数设置为TRUE,表示添加"IF NOT EXISTS"字句

添加键: $this->dbforge->add\_key('blog\_id', TRUE);第二个参数选TRUE表示主键

删除表drop\_table('table\_name',TRUE);第二个参数TRUE表示IF EXISTS字句，默认FALSE

重命名表: rename\_table('old\_table\_name', 'new\_table\_name');

给表添加列: $fields = array('preferences' => array('type' => 'TEXT', 'first' => TRUE)); $this->dbforge->add\_column('table\_name', $fields);其中'first'表示位置第一个，还可以'after' => 'another\_field'表示添加到哪个字段后面

删除列drop\_column('table\_name', 'column\_to\_drop');

修改列: $fields = array('old\_name' => array('name' => 'new\_name','type' => 'TEXT',),); modify\_column('table\_name', $fields);

数据库工具类

加载$this->load->dbutil();加载其他数据库$this->myutil = $this->load->dbutil($this-> other\_db, TRUE);

返回包含所有数据库名称的列表$this->dbutil->list\_databases();

database\_exists('database\_name')

返回CSV内容(即字段数组?)$this->load->dbutil();$query = $this->db->query("SELECT \* FROM mytable"); echo $this->dbutil->csv\_from\_result($query); 附加可选的三个参数: $delimiter(分隔符默认,), $newline(换行符默认\n), $enclosure(包围字符默认"")//(没用?)

备份数据库:$backup = $this->dbutil->backup();$this->load->helper('file'); write\_file ('./uploads/myPhp.zip', $backup); //有时报错?

设置备份参数:$prefs = array('tables'=>array('table\_name1', 'table\_name2'), 'ignore'=> array(),'format'=>'txt', 'filename'=>'mybackup.sql', 'add\_drop'=>TRUE, 'add\_insert'=> TRUE, 'newline'=> "\n");$backup = $this->dbutil->backup($prefs);其中ignore要忽略备份的表, format导出文件的格式, add\_drop是否在导出的SQL文件里包含DROP TABLE语句, add\_insert是否在导出的SQL文件里包含INSERT语句, newline导出的SQL文件使用的换行符, foreign\_key\_checks导出的SQL文件中是否继续保持外键约束

数据库驱动器类

initialize, db\_connect, db\_pconnect使用持久连接, reconnect, db\_select切换到某个数据库, db\_set\_charset设置客户端字符集, compile\_binds根据绑定的参数值编译SQL查询, is\_write\_type($sql)判断读TRUE/写类型, elapsed\_time返回当前已经执行了多少次查询, last\_query返回上一次执行的查询, escape($str)根据输入数据的类型进行数据转义, primary($table)获取表的主键, escape\_identifiers($item)对SQL标识符进行转义

辅助函数:

数组辅助函数: $this->load->helper('array'); element('size', $array, 'foobar');第三个参数是取不到元素时的值，默认NULL

$my\_shape = elements(array('color', 'shape', 'height'), $array, 'foobar');

random\_element

验证码辅助函数: $this->load->helper('captcha'); //图像显示损坏(原因:可能开了重定向，而需要在RewriteEngine on关闭captcha目录中内容的重定向)

cookie辅助函数:

$this->load->helper('cookie');

set\_cookie($name, $value, $expire, $domain, $path, $prefix, $secure(Bool), $httponly), //详见输入类 get\_cookie delete\_cookie//好像不比原生的方便多少

时间辅助函数: $this->load->helper('date'); //好像不比原生的方便多少

目录辅助函数: $this->load->helper('directory'); directory\_map($source\_dir, 0, FALSE)其中如果第二个参数不是1:根目录中有文件夹的文件夹名做键,相应数组做值；是1:数字作键,值是文件夹名字符串。第三个参数如果设置TRUE则显示隐藏文件，默认FALSE

下载辅助函数$this->load->helper('download'); force\_download($name, $data);//???

文件辅助函数: $this->load->helper('file');

读取用原生的file\_get\_contents()

写write\_file('./path/to/file.php', $data);返回Bool

删除指定路径下的所有文件delete\_files('./path/to/directory/',TRUE); 如果第二个参数设置为TRUE,文件夹也一并删除(//实际上删不掉,用原生的删),默认FALSE

get\_filenames('./captcha/')返回指定目录下的文件名数组,第二个参数为TRUE则显示完整路径，默认FALSE

get\_dir\_file\_info获取指定目录下所有文件信息组成的数组

get\_file\_info获取指定文件的信息, 第二个参数声明只返回想要的信息，可选name、size、date、readable、writeable、 executable 和 fileperms

get\_mime\_by\_extension($filename)获取MIME类型

表单辅助函数: $this->load->helper('form');

echo form\_open('email/send', 'class="email" id="myform"');第二个参数可选,第三个参数可选,作为隐藏域,例如$hidden = array('username' => 'Joe', 'member\_id' => '234');

form\_open\_multipart上传文件时使用

生成隐藏域: form\_hidden($array);

生成标准的文本输入框echo form\_input('username', 'johndoe', 'onClick="some\_function()" ');第三个参数可选，也可以是数组

form\_password类似form\_input，只是输入框类型为passwordwei

form\_upload输入框为file类型 form\_textarea

form\_dropdown($name,$array,$selectedStr)下拉框,第三个参数也可是数组，表示多选。也可以传入第四个参数$js = 'id="shirts" onChange="some\_function();"';其中$js也可以是Array

标准的多选下拉框form\_multiselect域的名称需要使用数组语法

显示legend : echo form\_fieldset('Address Information');echo "<p>fieldset content </p>\n"; echo form\_fieldset\_close();其中form\_fieldset的第二个参数也可添加额外的属性

复选框: echo form\_checkbox('newsletter', 'accept', TRUE, $js);参数一:id和name；参数二:value。也可以form\_checkbox($array)danxuan

单选框:form\_radio

label标签: echo form\_label('What is your Name', 'username', $attributes);

提交按钮: echo form\_submit('mysubmit', 'Submit Post!');

form\_reset

普通按钮: echo form\_button('mybutton', 'Click Me', $js);

表单结束标签form\_close($str)

set\_value设置元素的值 set\_select set\_checkbox set\_radio form\_error validation\_errors

HTML辅助函数

$this->load->helper('html');

标题标签: echo heading('How are you?', 4, array('id' => 'question', 'class' => 'green'));第二个参数是字体大小

img('路径(如果加了重定向,需要重写RewriteEngine on)')//用<http://localhost>，不要用./ RewriteCond $1 !^(index\.php|ext-6.2.0|uploads/.+|robots\.txt)其中图片放在uploads中，这里用了.+的正则

link\_tag 生成(css)link

ul标签: $list = array('red','blue','green','yellow');$attributes = array('class' => 'boldlist','id' => 'mylist');echo ul($list, $attributes);

ol类似ul

echo meta($name, $content, $type, $newline)也可以整个数组作参数//???

doctype//??? br(3)生成3个br标签

Inflector辅助函数 $this->load->helper('inflector');

复数变单数echo singular('dogs'); 单数变复数: plural('dog'); 将一个以空格或下划线分隔的单词转换为驼峰格式camelize; 将以空格分隔的多个单词转换为下划线分隔格式underscore; 将以下划线分隔的多个单词转换为以空格分隔，并且每个单词以大写开头humanize('my\_dog\_spot')如果单词是以连接符分割的，第二个参数传入连接符echo humanize('my-dog-spot', '-'); 判断某个单词是否有复数形式is\_countable('equipment');

数字辅助函数: $this->load->helper('number');

byte\_format(45678, 2);根据数值大小以字节的形式格式化，并添加适合的缩写单位,第二参数是精度

路径辅助函数: $this->load->helper('path');

set\_realpath($file,TRUE);返回服务端的绝对路径，路径不存在时报错

安全辅助函数//???

文本辅助函数: $this->load->helper('text');

$string = word\_limiter($string, 4);按第二个参数(单词个数)裁剪字符串,第三个参数是默认多余字符的替代字符串，默认省略号

$string = character\_limiter($string, 20);按字符个数裁剪字符串,会保持单词的完整性,第三个参数默认省略号

ascii\_to\_entities($str)将ASCII字符转换为字符实体,避免页面显示和数据库存储的问题,如ascii\_to\_entities("\41")输出!

convert\_accented\_characters//??

$string = word\_censor($string, $disallowed, $replace);将$string中含有$disallowed的单词替换为$replace,默认####

$string = highlight\_code($string);对一段代码进行着色

$string = "Here is a nice text string about nothing in particular.";

echo highlight\_phrase($string, "nice text", '<span style="color:#990000;">', '</span>');对字符串内的一个短语进行突出显示

ellipsize($str, 32, .5);根据指定长度(第二个参数)裁剪字符串,第三个参数是插入省略号的位置0-1

排版辅助函数: $this->load->helper('typography');

auto\_typography格式化文本以便纠正语义和印刷错误的 HTML 代码

nl2br\_except\_pre将换行符转换为 <br /> 标签 entity\_decode//??

URL辅助函数

返回站点URL: echo site\_url('news/local/123');或$segments = array('news', 'local', '123'); echo site\_url($segments);

base\_url用法似site\_url只是不会在最后加index.php

current\_url()返回当前的URL

uri\_string()返回包含该函数的页面的URI分段

index\_page()返回你在配置文件中配置的index\_page参数

echo anchor('news/local/123', 'My News', 'title="News title"');根据URL生成标准的HTML 链接,2参是链接的文本,留空则用链接本身作文本,3参是链接的属性,字符串或文本

anchor\_popup似anchor，只是生成的URL会在新窗口打开，可以用3参（数组）指定 JavaScript 的窗口属性

echo mailto('me@my-site.com', 'Click Here to Contact Me'); safe\_mailto似mailto,但可以防止被垃圾邮件，3参(数组)指定其他参数，如标题等

$string = auto\_link($string, 'both', TRUE);将一个字符串中的URL和e-mail地址自动转换为链接,2参选项: 'url', 'email', 'both', 3参TRUE表示在新窗口打开

$url\_title = url\_title($title, 'underscore', TRUE); 将字符串转换为对人类友好的URL字符串格式,2参指定分隔符,默认连字符, 'underscore'是下划线,3参为TRUE则默认转小写

$url = prep\_url('example.com');添加协议前缀

redirect重定向//??

xml辅助函数$this->load->helper('xml');

xml\_convert($string); 将输入字符串中的&,<,>,' ,"-保留字符转换为实体