**入门**

1. 添加公众号：
2. 普通接入
3. 授权登录接入：（推荐使用）提供普通订阅号自定义菜单等功能（普通接入没有此功能）
4. 关键字回复：
5. 系统关键字回复：包括文字回复、图文回复、音乐回复、图片回复、语音回复、视频回复、卡券回复、自定义接口回复、系统回复；
6. 扩展模块关键字回复
7. 自定义菜单：

提供设置公众号下方导航菜单的功能，最多可设置三个父菜单，5个子菜单。

**编码规范**

1、编辑器文件编码为UTF-8，关闭UTF-8 BOM((Byte Order Mark))的功能。不能使用Windows自带的记事本编辑项目文件。

2、所有临时性、调试性、试验性的代码，必须添加统一的注释标记“//debug”并后跟完整的注释信息。

3、书写规则

（1）大括号{} ：首括号与关键词同行，尾括号与关键词同列；

（2）运算符：每个运算符与两边参与运算符的值或表达式中间要有一个空格；

（3）一般情况下，在程序以及HTML中不出现两个连续的空格；

（4）引号：

1. 在能使用单引号的情况下，禁止使用双引号；
2. 数据库SQL语句一律使用双引号，SQL语句中所有数据必须加单引号，无论数值还是字串；
3. 数据库操作：
4. 所有数据库操作时必须使用PDO操作；
5. 所有SQL查询关键字大写，方便代码审查；



1. 安全性

所有变量在定义使用前都要初始化：防止恶意构造提交的变量覆盖程序中使用的变量。

1. 调试与开发阶段：需修改/data/config.php中的$config[‘setting’][‘development’]=1; 方便调试；

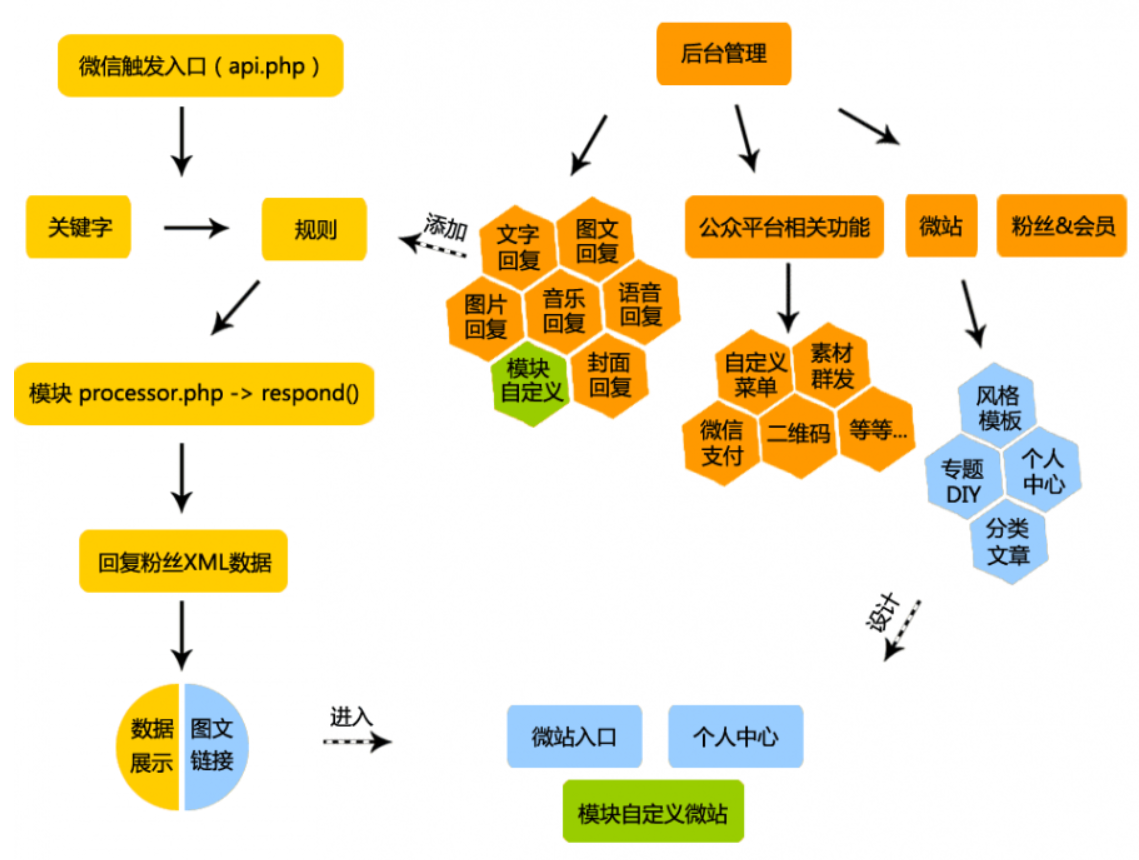
软件发布阶段：修改/data/config.php中的$config[‘setting’][‘development’]=0;

1. 数据表命名规则：若名词为可数名词，则必须以复数命名。例如：members、rules等。

**系统概述**

1. 微擎系统分为4部分：
2. api关键字回复
3. 微站：分为3部分：
4. 微站（首页）
5. 个人中心
6. 扩展模块本身的微站页面
7. 粉丝&会员
8. 扩展模块

2、微擎结构示意图：



1. 微擎系统分3个入口：
2. Web端入口：负责微擎系统的后台管理功能，入口文件为”web/index.php”。此入口主要完成以下工作：
3. 登录用户的身份验证
4. 初始化登录用户的身份信息，分派管理员权限或是操作员权限
5. 权限菜单划分及权限判断
6. 加载系统配置
7. 路由用户的请求
8. App端入口：主要负责微擎系统的前端“微站”、“个人中心”以及其他普通用户访问的功能。入口文件：”/app/index.php”。访问“微站”无需身份验证，访问“个人中心”则要求用户进行登录。此入口主要完成以下工作：
9. 初始化当前公众号的数据
10. 初始化微站站点信息、风格、模板、导航等数据
11. 初始化粉丝&会员的身份信息（如果有权限获取）
12. 加载系统配置
13. 路由用户的请求
14. 微信Api入口：主要负责处理微信公众平台请求微擎系统的关键字、图片、语音等数据。入口文件：”api.php”。此入口只用于公众平台访问。此入口主要完成以下工作：
15. 初始化当前公众号的数据
16. 解析公众平台推送过来的数据结构，转化为微擎消息结构
17. 根据消息结构中的关键字或是事件匹配到相应的模块
18. 调用模块的processor类返回数据给微信公众平台
19. 系统配置文件：

存放着微擎系统重要的启动数据，文件为：”/data/config.php”。

1. 微擎MVC
2. 控制器：

以文件夹、文件的形式组织，位于系统的source目录下，每一个目录代表一个controller，文件夹中的每个文件都为一个action。某些情况下，一个action可能会包含多个操作，系统中提供do参数来用于区分同一个action中的不同操作。在下图的例子中：extension为控制器，module为action，designer为该action下的某一个具体的do。



1. 模型：

系统的模型位于framework/model中，每一个文件代表一个独立的功能模型，模型通常会与控制器对应。

1. 模板：

系统的模板位于[web|app]/themes/default/中，模板文件与controller，action对应。

1. URL路由：

入口脚本程序获取到URL中相关的GET参数，解析后进行权限判断，然后调用相应的控制器处理这个请求。这个过程就被称为URL路由。

1. 加载器：

微擎系统依靠Load类进行类、函数、模型的加载，该类会在系统初始化时引用并且实例化，系统提供load()函数来引用该类的实例（ps：使用load()加载文件时，可以重复加载）

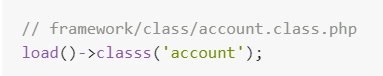
1. 加载系统公告函数

load()->func(‘filename’)加载文件为framework/function/\*.php。例如：



1. 加载系统公共类

load()->class(‘filename’)加载文件为framework/class/\*.php



1. 加载系统model函数

load()->model(‘filename’)加载文件为framework/model/\*.php



1. 加载web端公共函数

load()->web(‘filename’)加载文件为web/common/\*.php



1. 加载app端公共函数

load()->app(‘filename’)加载文件为app/common/\*.php



1. 微擎系统中使用error($errno,$message = ‘ ‘)来定义一个错误结构，使用is\_error($data)函数来判断是否发生错误。
2. 日志记录：

日志文件存放在/data/logs/run\_{YYYYmmdd}.log文件中。

**模板**

1. 调用系统模板文件：

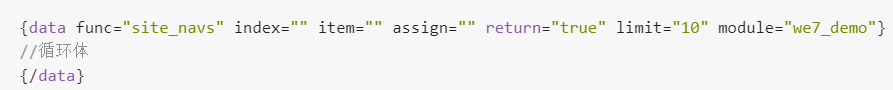
在任何php代码中可以使用template()函数来渲染一个视图文件。

1. 循环语句：

循环一个数组使用loop标签，与PHP中的foreach函数类似，第一个参数为数组的索引，第二个参数为数组第一项的值。

只要标签成对匹配，模板中的标签是可以嵌套使用的。

1. 模板数据标签：

微擎中在app端集成了一些常用数据的调用标签，web端不可调用，数据调用都是通过{data}{/data}标签来实现。一个完整的data标签：

1. func：调用标签的功能，也就是需要获取什么数据
2. item：默认为row，使用：$row。指定循环体中数组值的变量，在循环体中可以通过用此变量输出值
3. index：默认为iteration，使用：$row[‘iteration’]。指定循环体中的索引参数，如果指定过item参数，则{item}[‘{index}’]
4. assign,return默认为false，这两个标签通常共同使用，指定是否直接返回值，还是直接循环输出，如果return为true时，则会直接把内容赋值给assign设置的参数中。
5. limit：默认为10，指定获取数据的条数
6. module：如果模块中有扩展标签func，则通过此参数传递模块名称。

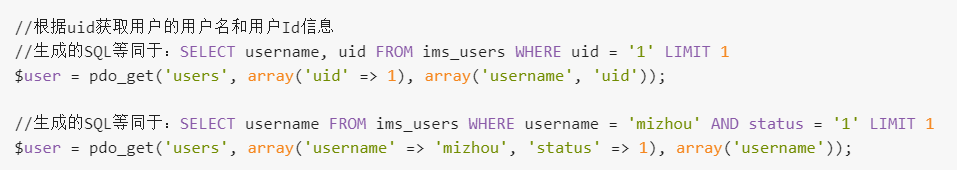
**数据库**

1. 参数绑定：

参数绑定就是绑定一个PHP变量到用作预处理的SQL语句中的对应命名占位符或问号占位符。可以有效的防止SQL注入。（PS：推荐无论何时尽量使用参数绑定的形式来构建SQL语句）

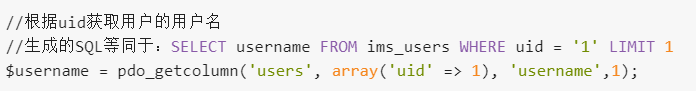
1. 数据库操作
2. 查询
3. pdo\_get

根据条件（AND连接）到指定的表中获取一条记录。



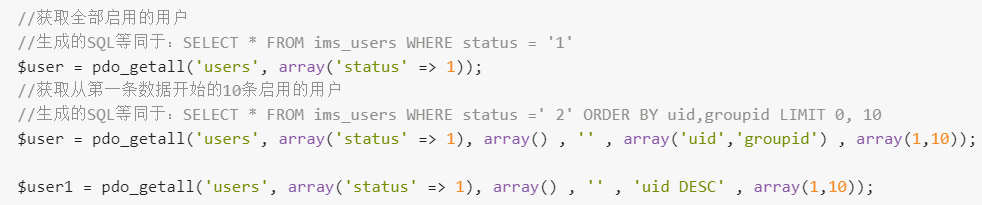
1. pdo\_getcolumn

根据条件（AND连接）到指定的表中获取一条记录的指定字段。



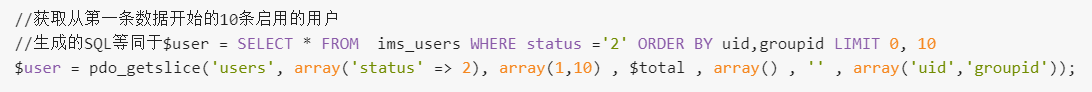
1. pdo\_getall

根据条件（AND连接）到指定的表中获取全部记录。



1. pdo\_getslice

根据条件（AND连接）到指定的表中获取某个区间的记录，此函数和pdo\_getall的区别是可以指定limit值。



1. pdo\_fetch

根据SQL语句，查询一条记录



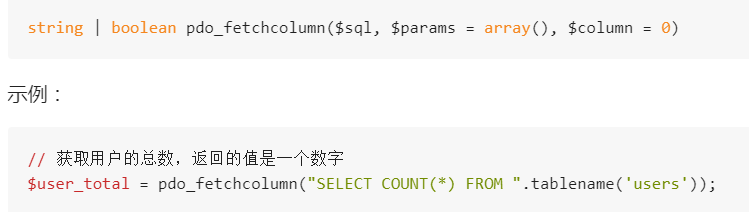
$sql参数指定要返回记录集的SQL语句

$params参数指定为 SQL

1. pdo\_fetchcolumn

根据SQL语句，查询第一条记录的第N列的值，此语句与pdo\_fetch使用相同，只是此函数返回的不是一个数组而是一个字符串

$column参数指定返回记录集的第几列数据



1. pdo\_fetchall

根据SQL语句，查询全部记录，使用方法与pdo\_fetch相同

1. 插入 pdo\_insert

对指定数据表插入一条新纪录



$tablename参数指定要插入记录的数据表名，此处传入的表名不要使用tablename()函数

$data参数指定要插入的记录，格式为与数据表字段对应的关联数组

$replace参数指定插入方式使用INSERT语句或是REPLACE语句（查找到主键相同的数据选择update）

1. 更新 pdo\_update

更新指定的数据表的记录

$glue参数指定前面$condition数组条件的关联字AND或是OR



1. 删除 pdo\_delete

删除指定条件的数据



1. 运行SQL
2. 当更新，插入，删除无法满足时，可以直接构造SQL语句进行操作 pdo\_query

$params指定SQL语句中绑定参数的值，参数占位与pdo\_

fetch一致



例子：



1. 批量执行SQL语句 pdo\_run



$stuff函数将会将此参数指定的值，替换为当前系统的表前缀。

PS：与pdo\_query不同的是，pdo\_run是可以一次执行多条SQL语句，每条SQL必须以；分隔。

1. 辅助函数
2. 检查表中是否存在某个字段 pdo\_fieldexists



$tablename参数指定要检查的表名称

$fieldname参数指定要检查是否存在的字段名

例子：



1. 检查表中是否存在某个索引 pdo\_indexexists



$tablename参数指定要检查的表名称

$indexname参数指定要检查是否存在的索引

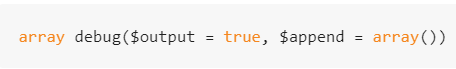
例子：



1. 检查数据库中是否存在某个表 pdo\_tableexists



1. 调试运行SQL语句，显示执行过的SQL的栈情况



$output参数指定是否直接打印出调试信息

1. 主从配置

微擎系统支持数据库的主从配置（一主多从）。

配置主从支持在系统”data/config.php”文件中。

**缓存**

1. 缓存配置：

微擎系统支持三种缓存（mysql,memcache,file），默认是mysql缓存，同一时间只能启用其中一种。

配置缓存在系统”data/config.php”文件中。

1. 数据缓存

数据缓存是指将一些PHP变量存储到缓存中，使用时再从缓存中取回，避免过多的操作直接从数据库中存取，减轻数据库压力。

1. 键名命名规则

缓存的键名必须保证是唯一，如果使用相同的键名，以最后一次存储为准。

命名键名时应该使键名以功能分组，以冒号隔开，例如公众号相关的缓存命名为：account:info，account:auth等等。

1. 按照指定的键名存储缓存数据 cache\_write

cache\_write($key,$data)

$key参数指定要存储缓存数据的键名，键名必须保证是唯一；

$data参数指定要存储数据的内容，可以为字符串，数组等；

1. 读取指定键名的缓存数据 cache\_load

cache\_load($key)

$key参数指定要读取缓存数据的键名

1. 删除指定的缓存 cache\_delete

cache\_delete($key)

$key参数指定要删除缓存数据的键名

1. 清除所有缓存： cache\_clean()

**会员与积分**

1、Openid与uid的关系

这里的Openid可以泛指从微信、易信或是其他平台进入的粉丝，当这些粉丝进入模块或是微站时，在系统中是不产生任何资料数据。

仅当这些粉丝完成绑定、注册操作时在系统中产生一个uid与之对应，才可以存储一些额外资料（包括真实姓名，性别等等）和积分数据，系统也允许多个Openid对应同一个Uid。

1. 验证粉丝身份信息

当模块或是系统需要获取会员资料或是积分时，可以使用checkauth()函数来判断会员是否进行登录（未登录时会跳转至登录页面），以便可以使用其资料和积分数据。

1. OAuth权限

OAuth权限是指：粉丝进入公众号提供的一个网页时，程序可以通过接口调用到该粉丝的Openid、昵称、头像等相关信息，相当于该粉丝进行了登录的操作，可以获取到该粉丝的唯一身份标识信息。

PS：在微信公众号的权限体系中，只有认证服务号才可以拥有OAuth权限。

1. 无OAuth权限获取用户信息的方法：
2. 通过触发关键字
3. 通过借用OAuth权限
4. 更新用户积分数据



$uid：要变更数据会员的uid

$credittype：要变更积分数据的类型，分为credit1，credit2，credit3，credit4，credit5，credit6

$creditval：要增减积分的值，正数为增，负数为减

$log：积分操作日志

**卡券**

1. 营销卡券

营销卡券在微擎中分为：微信卡券和系统卡券。

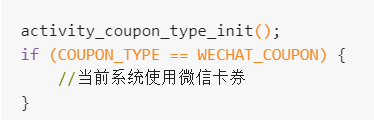
这两个类型在系统中使用常量COUPON\_TYPE来标识，使用前需先调用activity\_coupon\_type\_init()来初始化公众号支持卡券类型。

1. 微信卡券

当公众号为认证服务号时，系统默认切换为微信卡券来使用卡券。

卡券类型分：折扣券、代金券、礼品券、团购券、优惠券。

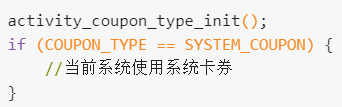
领取、发放及核销全部通过微信接口来操作。



1. 系统卡券

当非认证服务号时，系统模拟公众平台微信卡券的功能实现一套卡券功能。

支持的卡券类型：折扣券、代金券。（此卡券与微信卡券不通用）



1. 发放卡券
2. 通过营销活动发放

发放卡券在“粉丝营销 - 卡券管理 - 卡券营销”中进行卡券发放。

1. 通过模块发放

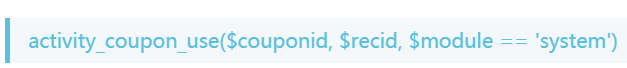
在模块中调用acticity\_coupon\_grant()方法发放卡券



$id ：卡券id

$openid ：指定会员的openid

1. 核销卡券



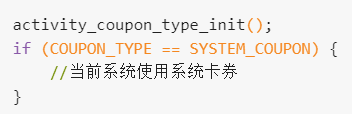
$couponid ：卡券id

$recid ：兑换记录id

$module ：模块名称，默认为系统模块

1. 卡券相关函数
2. 初始化公众号支持卡券类型

activity\_coupon\_type\_init()



1. 获取某张卡券的所有信息

activity\_coupon\_info($id)

$id ：卡券id

1. 获取当前公众号所有有效的卡券

activity\_coupon\_available()

1. 获取当前会员当前已有卡券及使用情况

activity\_coupon\_owned()

1. 指定会员兑换指定真是物品

activity\_goods\_grant($uid,$exid)

$uid ：会员id

$exid ：真是物品id

1. 指定会员兑换指定活动

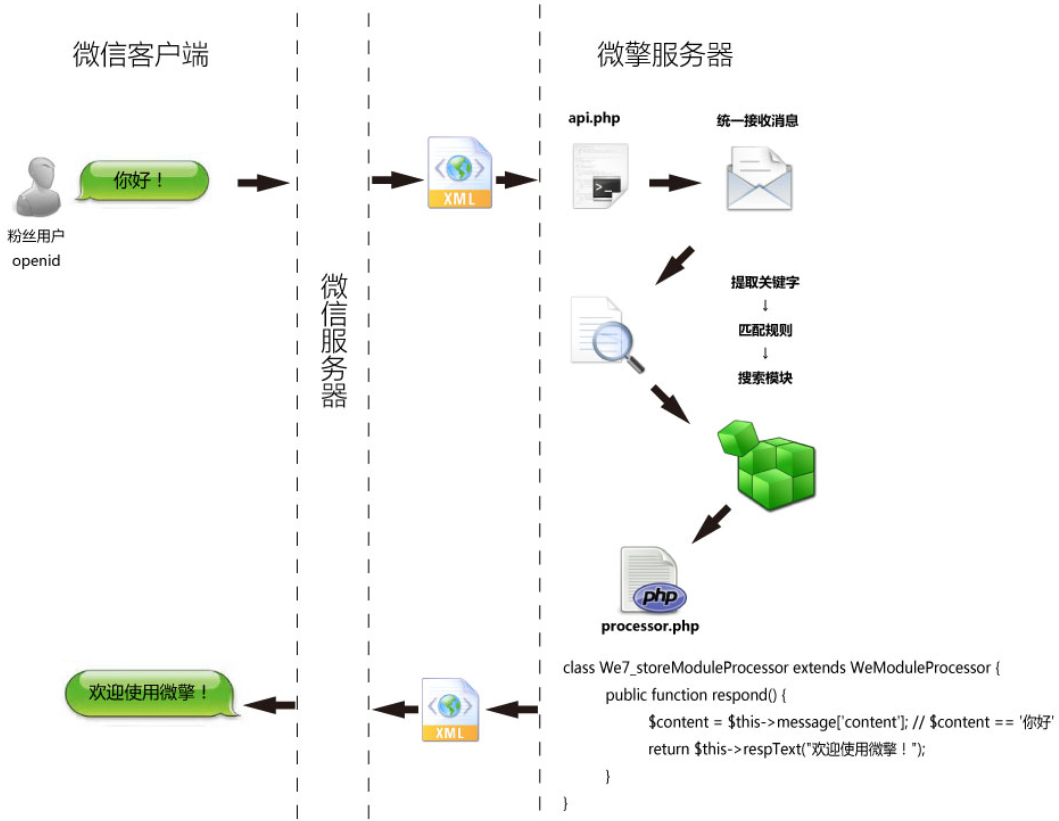
Activity\_module\_grant($uid,$exid)

$uid ：会员id

$exid ：活动id

**消息响应**

1. 消息响应是什么？
2. 粉丝用微信给公众号发送消息
3. 公众平台将粉丝用户的请求消息传递给微擎系统（数据为XML格式）
4. 微擎系统按照消息类型和对应的公众号所设定的规则列表匹配到合适的规则，规则定义包括处理此消息所使用的模块和此模块处理消息时所需要的其他附加数据
5. 模块将按照请求的消息数据和模块附加数据进行相关业务处理并返回处理结果
6. 微擎系统将处理结果返回给公众平台
7. 公众平台将结果再次返回给粉丝用户



微擎的主处理流程实现定义于：api.php中的class WeEngine。

1. 消息的路由

消息路由是指 粉丝用户经公众平台发送消息内容至微擎时，微擎系统查找对应的规则记录，并将消息分配至合适的模块处理的过程。

微擎系统按照不同的消息类型，进行不同的处理。处理方式如下：

1. 上下文消息路由

微擎支持上下文操作。

通过上下文支持微擎可将用户对话锁定至特定的模块，如果当前消息时上下文对话的消息，那么将会自动路由至上下文锁定的模块。

1. 文本消息规则匹配

针对文本消息，微擎使用文本匹配来选择合适的规则和模块，规则是针对特定消息的处理方式。微擎选择规则的方式包括：

1. 关键字包含

只粉丝用户发送的消息内容**含有**指定的关键字就指派到特定规则

1. 内容等价

指粉丝用户发送的消息内容**完全等于**指定的内容才指派到特定规则

1. 正则表达式

指粉丝用户发送的消息类型**符合指定正则**表达式定义的模式时指派到特定规则

1. 系统回复

系统回复是指 系统内置的两种回复类型。

包括：

1. 欢迎回复消息：为用户关注时触发的消息回复；
2. 默认回复：为未匹配到关键字时的消息回复。
3. 特殊路由消息

特殊消息路由是指 除文本消息之外的消息类型，包括图片消息、语音消息、视频消息、小视频消息、位置消息、\*上报地理位置、链接消息、进入聊天窗口、微小店消息。

PS：设置特殊消息接管后，用户发来此类消息优先会路由至该消息的处理模块中。

**微信API**

WeiXinAccount类

所有封装的接口方法均位于framework/class/weixin.account.class.php文件的WeiXinAccount类中，使用此类时，请按下方的方法实例化：

$account\_api = WeAccount::creat();

1. 公众号AccessToken
2. 获取公众号AccessToken：



1. 刷新公众号AccessToken



1. 共享收货地址

获取微信共享地址：



注意事项：

1. 必须更新微擎系统最新版（2015年5月19日更新）
2. 模板页面必须引入公共的头部和尾部文件

调用方式如下：



1. 标签管理

开发者可以使用用户标签管理的相关接口，实现对公众号的标签进行创建、查询、修改、删除等操作，也可以对用户进行打标签、取消标签等操作。

|  |  |  |
| --- | --- | --- |
| 标签名 | | 描述说明 |
| fansTagFetchALL() |  | 获取所有标签 |
| fansTagAdd($tagname) | $tagname：标签名称 | 新建粉丝标签 |
| fansTagEdit($tagid,$tagname) | $tagid：标签id，添加成功后返回的id  $tagname:标签名称 | 编辑标签 |
| fansTagDelete($tagid) | $tagid：标签id，添加成功后返回的id | 删除标签 |
| fansTagFetchOwnTags($openid) | $openid：粉丝openid | 获取粉丝拥有的标签 |
| fansTagTagging($openid,$tagids) | $openid:粉丝openid  $tagids:标签id，最多为三个，结构为array(1,2,3) | 设置粉丝标签 |
| fansTagBatchTagging($openid\_list,$tagid) | $openid\_list：粉丝openid列表，结构为array(‘openid1’,’openid2’)  $tagid：标签id | 批量设置标签 |
| fansTagBatchUnTagging($openid\_list,$tagid) | $openid\_list：粉丝openid列表，结构为array(‘openid1’,’openid2’)  $tagid：标签id | 批量取消标签 |

例子：



1. 客服消息

发送文本客服消息 sendCustomNotice($message)

$message：要发送客服消息结构，消息分为文本，图片，语音，视频，音乐，图文，卡券。

1. 粉丝信息

手动获取粉丝信息：fansQueryInfo($openid)

1. 素材

在回复或是发送给粉丝消息内容时，对多媒体文件、多媒体消息的获取和调用等操作，是通过media\_id来进行的，这个media\_id就是一个素材的id。

素材分为临时素材和永久素材。

1. 临时素材：

对于临时素材，每个素材（media\_id）会在开发者上传或粉丝发送到微信服务器3天后自动删除（所以用户发送给开发者的素材，若开发者需要，应尽快下载到本地），以节省服务器资源。

1. 永久素材：

永久素材会一直存在，除非用户删除，永久素材的存储上限为图片最多5000个，其他类型为1000个。

永久素材可以在公众平台官网素材管理模块中看到永久素材。

1. 上传素材
2. 上传临时素材： uploadMedia($path,$type)

$path：待上传文件物理路径（绝对路径）

$type：上传素材的类型，分别有图片（image），语音（invoice），视频（video）和缩略图（thumb，用于视频与音乐格式的缩略图）

1. 上传永久素材 uploadMediaFixed($path,$type)

$path：待上传文件物理路径（绝对路径）

$type：上传素材的类型，分别有图片（image），语音（voice）、缩略图（thumb，用于视频与音乐格式的缩略图）。

此函数无法上传永久视频文件！！！上传永久视频文件：uploadVideoFixed

1. 上传永久视频素材函数：

uploadVideoFixed($title,$description,$path)

$title：视频素材的标题

$description：视频素材的描述

$path：待上传文件物理路径（绝对路径）

1. 上传永久图文素材： addMetrialNews($data)

$data：图文素材结构

1. 获取素材
2. 获取临时素材 downloadMedia($mediaid,$savefile = true)

$mediaid：要获取素材的id

$savefile：是否保存成文件，或是直接返回资源文件二进制内容

1. 获取永久素材 getMeterial($mediaid,$savefile = true)

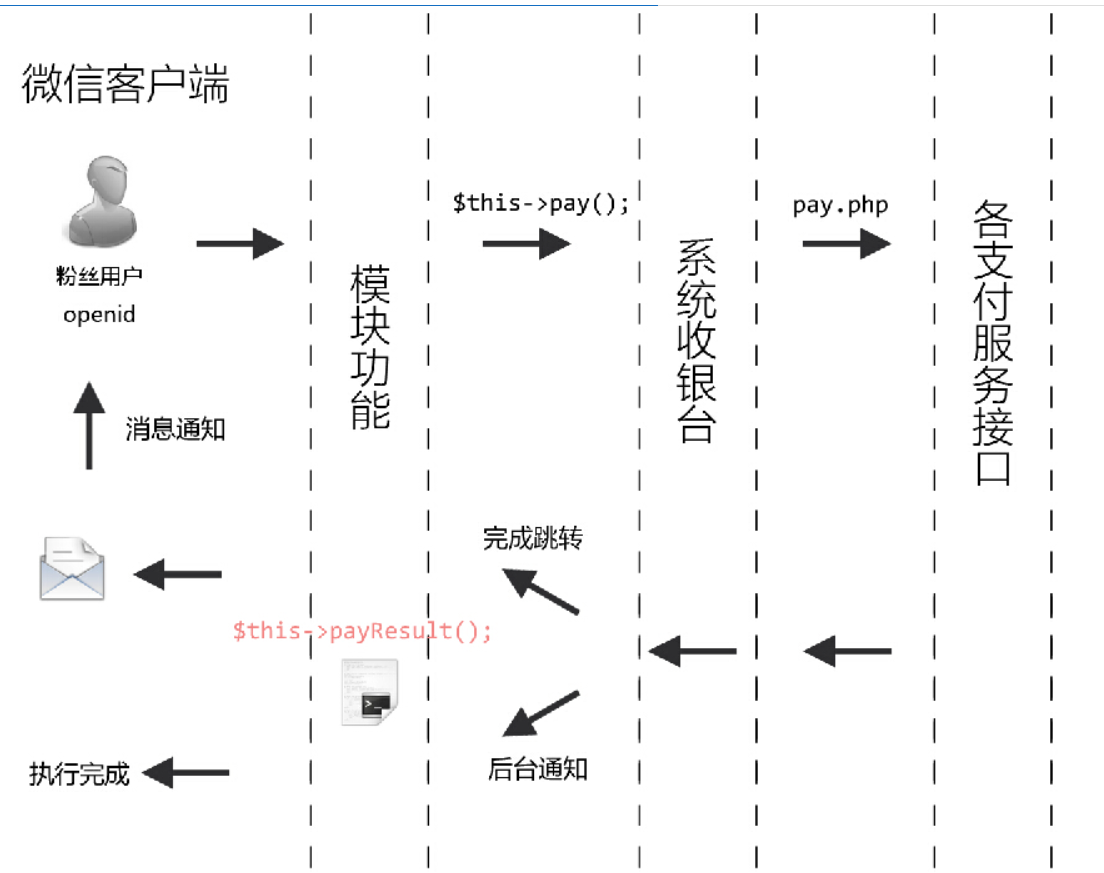
$mediaid：要获取素材的id

$savefile：是否保存成文件，或是直接返回资源文件二进制内容

1. 获取永久素材数量 getMeterialCount()
2. 删除永久素材 delMaterial($mediaid)

**在线支付**

1. 微擎在线支付工作流程：
2. 用户在使用模块功能需要支付时，模块调用父类的$this->pay($params);方法来调用系统支付功能；
3. 系统收银台接到模块调用请求，生成支付界面（金额，商品信息等）；
4. 用户在收银台中选择支付手段，如微信支付、支付宝支付等（需要在支付参数中设置开启）；
5. 收银台根据用户选择的支付手段，请求到相应的支付接口上，用户完成支付；
6. 完成支付后，系统会调用模块中的$this->payResult($params);方法来通知支付状态（成功或是失败）。



1. 发起支付（PHP版）

在微擎系统中，发起支付只需调用父类的$this->pay();方法即可。

参数说明：

tid：充值模块中的订单号，此号码用于业务模块中区分订单，保证在业务中是唯一即可；

title：收银台中显示的标题；

fee：收银台中显示需要支付的金额，单位是元，只能大于0；

ordersn：收银台中显示的订单号，仅供显示用

user：付款用户，付款的用户名（选填项）

1. 发起支付（JS版）

在PHP中发起支付时，粉丝在支付时界面会跳转至系统的收银台界面中。对于希望在当前页面直接处理的开发者，微擎提供了利用JS发起支付的方式，该方式如果是微信支付则会直接在当前页面中支付，如果是百度钱包、支付宝其他支付则会直接跳转至支付方进行支付。

util.pay(option)

option参数列表如下：

orderFee：要支付的金额

payMethod：发起的支付类型，分别为微信支付（wechat）、百度钱包（baifubao）、支付宝（alipay）、银联支付（unionpay）

orderTitle：支付标题

orderTid：充值模块中的订单号，此号码用于业务模块中区分订单，保证在业务中是唯一即可

module：发起支付的模块，支付成功后会通知该模块

success：支付成功回调函数

fail：支付失败时回调函数

complete：支付完成（成功和失败）时回调函数

1. 验证支付

当用户支付完成后，系统会调用模块中的$this->payResult($params);方法

**微擎模板语法**

1. PHP中使用template()函数显示已存在模板

在PHP代码中展示模板页请使用template()函数，在开发模块时模块内部应该使用$this->template()函数，这两个函数是用方式完全相同，只是开发模块时特定模块的模板定义于当前模块文件夹内部。此方法定义如下：

mixed function template(string $filename,int $flag = TEMPLATE\_DISPLAY)

说明：展示特定模板内容

参数：

$filename：模板名称，格式为：’模板文件夹/模板名称无后缀’,如：commom/header

$flag 模板展示方式

$flag含义：TEMPLATE\_DISPLAY导入全局变量，渲染并直接展示模板内容（默认值）

TEMPLATE\_FETCH：导入全局变量，渲染模板内容，但不展示模板内容，而是将其作为返回值获取。可用于静态化页面。

TEMPLATE\_INCLUDEPATH：不导入全局变量，也不渲染模板内容，只是将编译后的模板文件路径返回，返回的模板编译路径可以直接使用include嵌入至当前上下文。

示例：以下三种调用方式效果相同

$list = array();... //其他更多上下文数据

template(‘common/template’); //直接展示模板

$content = template(‘common/template’,TEMPLATE\_FETCH); //获取模板渲染出的内容

echo $content; //输出渲染的内容；

include template(‘common/template’,TEMPLATE\_INCLUDEPATH); //嵌入模板编译路径

1. 变量输出 {$var}

使用花括号包含的变量将直接输出至页面，功能等同于

<?php echo $var;?>

转义：一般情况模板引擎能自动识别javascript中的花括号和变量输入的花括号。如果遇到程序不能自动分辨的情况可以强制使用“{##”符号来代表“{”，使用“##}”符号代表“}”

特殊情况：变量输出语法同时支持直接输出数组元素或嵌套数组元素，例如：

{$row[‘name’]} {$\_W[‘member’][‘username’]}

1. 循环语句 {loop $list $row}

使用循环语法能够遍历某个集合的内容。示例：

{loop $wechats $wechat}

<li>Name：{$wechat[‘name’]}</li>

{/loop}

扩展语法：可以使用扩展的语法{$loop $list $key $wechat}来遍历集合的键名和键值，例如：

{loop $wechats $weid $wechat}

<li>Id：{$key}；Name：{$wechat[‘name’]}</li>

{/loop}

1. 模板嵌套 {template $name}

在模板的当前位置嵌入另一个模板，例如：使用{template common/header}来嵌入标准页头。

1. PHP语法嵌入 {php statement}

如遇到不能满足需要的情况可以使用PHP语法嵌入标记，例如：

{php echo date(‘Y-m-d H:i:s’,$row[‘dateline’]);}

1. 数据获取标签

{data module=”” func=”” return=”” item=”” limit=”” assign=””}{/data}

其中：func：指定获取数据的函数，此函数定义在模块目录下的model.php文件中

module：指定获取数据的模块；

assign：指定该标签得到数据后，存入的变量名称。如果为空则存在于func同名的变量中,方便在下方的代码中使用

item：指定循环体内的迭代时的变量名。相当于foreach($foo as $i=>$row)中$row变量。

limit：指定获取变量时条数。

return：为true时，获取到数据后直接循环输出，为false时，获取到数据后作为变量返回。默认为false。

目前支持的获取数据func

{data func = “site\_slide\_search”} 获取微站幻灯片

获取出4条幻灯片数据，示例：

**模块**

设置新模块配置说明

1. 模块基本信息

“模块名称”用于在系统中给用户显示出模块的名字

“模块标识”只能为英文，数字，下划线，为了减少模块标识的重复和冲突，系统规定模块必须有前缀！

“版本号”用于标识模块版本，主要是用于模块升级更新使用

1. 模块全局配置项

当勾选此项时，模块安装后系统会生成一个“模块配置”的菜单，用于保存一些模块内部使用的配置项。

开发者必须要完善module.php类文件中的public function settingsDisplay($settings){}成员方法。（PS：module.php路径为addons/lonaking\_gift\_shop）

1. 订阅的消息类型

当勾选此项下的事件类型后，系统会在接收到相关类型的事件后，转发消息到模块中。模块接到消息后可以处理数据，但无法返回结果（系统不接收）。

开发者必须要完善receiver.php类中的public function receive(){}成员方法。（PS：reveiver.php位于addons/lonaking\_gift\_shop）

1. 是否要嵌入规则

当勾选此项后，模块安装后系统会自动添加“回复规则列表”菜单，用户可以设置关键字触发到模块中。

开发者必须要完善processor.php类文件中的public function respond(){}方法。（PS：processor.php位于addons/lonaking\_gift\_shop）