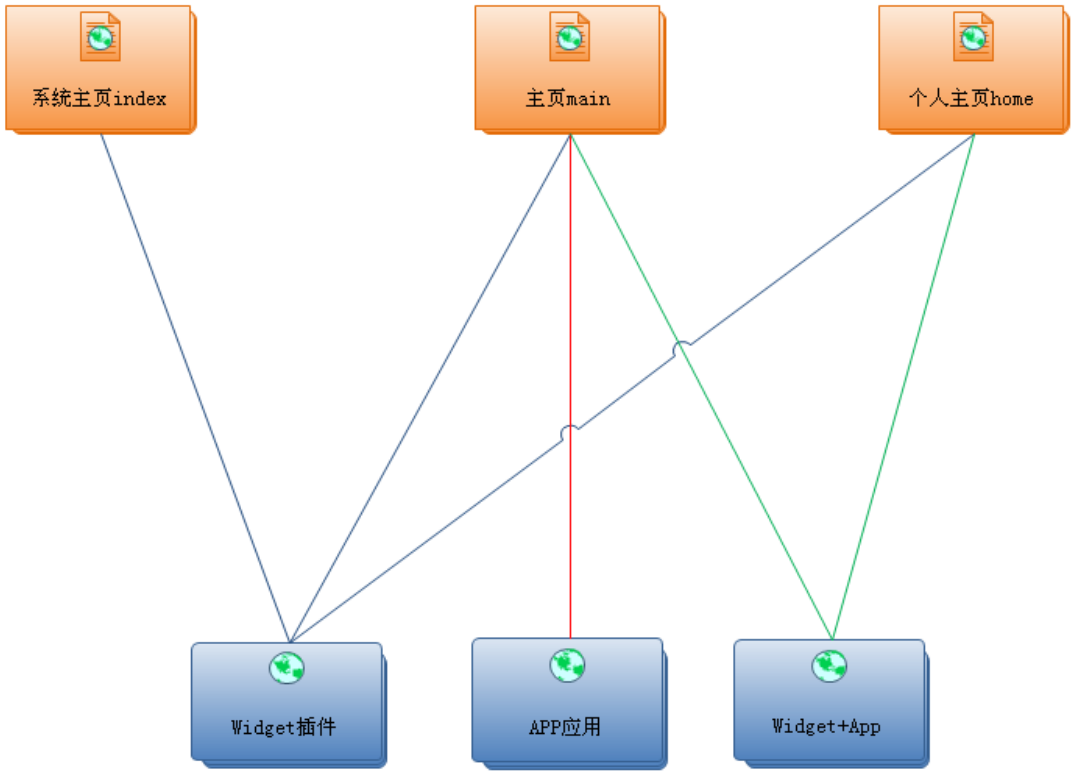
iWebSNS插件规范(beta版)

插件简介

插件就是为了满足个性化需求按照 **iWebSNS** 插件开发规范编写的可插拔程序，虽然可以直接对 **iWebSNS** 进行二次开发实现插件同样的功能，但是这样做势必影响到系统的升级和稳定性。采用插件的方式，可以随时进行停用和卸载，而且对系统毫无影响。

插件的类型

插件显示位置和分布如下。



插件布局分布图

插件规范及注意： 我们这里以插件的标识符 myplugin （建议唯一）为例说明。

1、myplugin 应该放在 plugins 目录下面；

2、myplugin 必需包含 plugin.xml 文件，此文件用来配制插件的基本信息，可以通过工具自动生成；

3、如果插件有复杂的交互功能，还需在 myplugin 目录下创建 actions.xml 文件，并且编写相

应的脚本文件。

插件开发步骤:

1、明确自己需要的功能，先认真看清楚 iWebSNS 是否已经包含；

2、插件命名（中英文皆可）。插件的标识符（必需为英文）建议唯一，这点非常重要，建议在开发插件的时候一定要遵守下面的规范；

3、明确插件要显示的位置（参见上面的图示），以确定插件的类型。

插件的分类有：

1)、以嵌入的方式来显示，交互主要是通过 Ajax 来实现。此类插件为系统的 Widget 插件；

2)、通过页面的主显区域显示，实现较为复杂的交互功能。此类插件为系统的 APP 应用；

3)、在两者之间，部分嵌入页面，部分显示在主显区域，即系统的 APP+Widget 插件。

4、如果开发需要有 API 的支持，请参阅 API 文档。

5、编写开发完成后，生成自己的 Plugin.xml 插件的安装信息文件。此文件的创建由 createplugin.php 文件生成，该文件可从官方下载，或者运行系统 plugins 目录下的 createplugin.php 文件生成，如果插件涉及到交互脚本，还需要创建自己的 Actions.xml 文件，下面会有详细的介绍 Actions.xml 的编写规范。

6、开发的插件如果涉及到数据表的操作，则应该先创建好自己的数据表，然后用 phpmyadmin、SQLyog 导出，如果自己编写一定要遵守 SQL 文件的编写规范,并把些文件配制到 plugin.xml 中。

7、调试通过后，插件即可发布。

关于 actions.xml 文件编写:

格式如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE actions SYSTEM "../Actions.dtd">//此行一定要引入，否则程序不能正常运行
<actions>
    <package name="test">
        <action name="show" file="action/PersonAction.php" class="Person" method="show">
```

```
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
        <action name="add" file="action/PersonAction.php" method="add">
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
        <action name="del" file="action/PersonAction.php" method="del">
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
    </package>
</actions>
```

关于此文件的详细说明如下：

根结点(actions)：指 XML 配制文件的根结点。

包结点(package)：指包含一类功能的包名，方便开发者把同一类功能的 action 放在一起，利于管理，此节点只包含一个 name 必选属性，指包的名字。

动作结点(action)：指每一个功能处理节点，此节点包含 name、file、class、method 等属性，name 是必选属性，功能名是方便开发者调用的；file 是必选属性，指用来处理此功能的文件；class 是可选属性，是开发者的类名，如果开发者没用到类，那么此属性不用填写。Method 是必选属性，处理此动作的功能函数。

返回节点(return)：指动作处理完成后要跳转地址，name 是必选属性是指 method 方法中处理完后的返回的字符串；内容是指此字符串对应的跳转地址，地址的路径是指开发者插件的根目录开发的。

如果开发的插件是包含对于数据库的操作，那么分为两类。一类是有自己的数据库服务器，那么开发者不必提供对应的 SQL 安装文件，也不用编写 actions.xml 文件；另一类是没有自己的数据库服务器，要用此系统的数据库，就要编写自己的 SQL 安装文件（建议此文件是用 PHPMyAdmin 和 SQLyou 工具或者由 MySql 直接导出的文件），系统识别的注释标记有“/**/”“--”“#” ，每句都要以“;”为结束符号，并且要在生成 plugin.xml 文件时填写上此 SQL 安装文件，为了防止冲突，开发者创建的表必需添加表前缀，并且在插件的根目录

下建立一个 config.php 文件，文件中包含$table_prefix='表前缀名';开发者的程序中要引用此变量，是保证能系统发生冲突时自动解决的前提。

关于用户文件有编写：

在系统目录下，有一个 plugins 文件夹，这是存放用户所有插件的地方，此目录下 includes.php 文件，是用户开发插件的基础，也是用户调用 Api 的必需文件。

在引入此文件之前要引入以下变量，

$getpost_power=true;开启统一读入$_GET 和$_POST 的方法，如果开发者不使用此功能不用引入此行代码。

$iweb_power=true;开启对系统平台数据库的支持，如果只是调用 API 则不用开启此功能。

$session_power=true;如果开发者要使用，session 那么需要开启此功能。

如果只引用些文件，前面不引入变量，那么系统自动会开启 API 的支持。

如果开发者开发的功能用不到以上功能，则此文件不必引入。

一些补充说明：

1、 widget 插件的编写。此类型的插件一般为一些小型的程序，如一些广告、FLSAH、日历、天气预报、IP 查询等等的小型插件，编写开发都比较自由，显示的方式是网页嵌入式的方式，如果有内容的更新和切换，是由系统提供的 Ajax 类来实现的，用户直接调用即可，此插件可以安装到系统的首页，个人首页及个人主页的页面上。

2、 Application(App)应用的开发，一般都涉及到交互式的操作，如：开心农场、关注好友日志、关注度排行榜等等。此应用插件是可以由用户自由定制的，此插件有显示位置如下图:

APP 应用开发的要求，如果此插件要在系统内建立表，那么开发者要提供自己的 MySql 安装文件，并且配制到自己的 plugin.xml 文件中，建议用户在自己的插件根目录下建立 config.php 文件，并写入变量$table_prefix='表前缀名';此举主要是防止开发的插件出现同名冲突，以实现产生冲突时系统能自动分配唯一标识符，保证插件的正常运行。其它的文件完全由用户自由的开发。在 App 应用显示的位置（如图所示），如果有内容提交及跳转请求应使用系统提供的 do_action(__file__)函数，使用此函数必需引用 includes.php 文件。

3、 Widget+APP 式的插件，只能显示在个人主页及个人首页上，以嵌入式和交互式的方式显示，如：音乐盒等程序。需要配制的文件和注意的事项同上。

开发样例:

下面我们以一个实例来讲述如何开发一个插件。

插件名为：日志排行榜 英文唯一标识：blogranking

建立一个名为 blogranking（即唯一标识名）文件夹。

开始编写文件，文件名为 index.php,代码如下：

<?php

```
include(dirname(__file__)."/../includes.php");//引入调用 API 的必要条件
?>
<link href="<?php echo self_url(__file__)?>test.css" rel="stylesheet" type="text/css" />
//调用系统方法得到当前路径
<div id="blogranking">//用户开发插件自定义 ID，也是用户定义的工作区
<div id="bar" style="background:url(<?php echo self_url(__file__)?>image/top.gif); height:29px;
align='center'"><a href="<?php echo self_url(__file__)?>index.php?type=0" name="ajax"
target="'blogranking"' >日</a>
//
<a href="<?php echo self_url(__file__)?>index.php?type=1" name="ajax" target="'blogranking"'
> 周 </a>          <a href="<?php echo self_url(__file__)?>index.php?type=2" name="ajax"
target="'blogranking"' >月</a></div>
<table width="100%" height="20" border="0" align="center" cellspacing="0" >
   <tr style=" background-color:#c7ebf9 ;height:20px">
      <td bgcolor="">日志标题</td>
      <td bgcolor="">发表者</td>
      <td bgcolor="">点击数</td>
   </tr>
<?php
function change($type) /根据自己的需要编写自己的函数
{
      date_default_timezone_set("Asia/ShangHai");
      $date=date('Y-m-d');
      $week=604800;
      $month=2592000;
      switch(intval($type))
      {
            case 1: return date('Y-m-d',strtotime($date)-$week); break;
            case 2: return date('Y-m-d',strtotime($date)-$month); break;
            default:
```

```
            {
                return $date; break;
            }
        }
}
$date=change(0);
if(isset($_POST['type'])) $date=change($_POST['type']);
$blogs=Api_Proxy("blog_self_by_date","log_title,user_name,hits,user_id,log_id",$date."~",'hits');
//调用系统的 API 函数，下面会讲述此函数的调用。
$num=0;
foreach($blogs as $blog)
{
$num=(++$num)%2;
if(get_sess_username())$blog_url="href='modules.php?app=blog_show&id=$blog[log_id]'
target='frame_content'";
else $blog_url="href='home.php?h=$blog[user_id]&app=blog_show&id=$blog[log_id]'";
echo<<<EOD
<tr class="tr{$num}">
<td><a $blog_url >$blog[log_title]</a></td>
<td>$blog[user_name]</td>
<td class="num">$blog[hits]</td>
</tr>
EOD;
}
?>
</table>
</div>
```

test.css

```
@charset "utf-8";
/* CSS Document */
```

```
#blogranking table
{
border-bottom:#999999 1px dotted;
}
#blogranking td
{
height:24px;
padding-left:4px;
border-top:#999999 1px dotted;
}
#blogranking #bar a
{
float:right;
color:#FF0000;
padding-top:4px;
padding-right:2px;
}
.tr1
{
background:#fafafa;
}
.tr0
{
background:#f2f2f2;
}
```

关于插件中的 a 连接和 form 表单的作用说明：

1、 a 连接：

如：<a href=” <?php echo self_url(__file__);?>a.php” name=”ajax” target=”targetid”>AJAX 连接

如果一个 A 连接的 name 属性声明为”ajax” 并且 target 属性也已经声明，那么系统会自

动对此标签进行 AJAX 处理，target 属性有两种方式，一种是在“‘targetid’”，此 targetid 是开发者自定义的 ID，也就是开发者定义的工作区 ID，连接请求的内容会展示的工作区内，另一种是“targetid”，此 targetid 是开发者定义的函数，系统会自动回调此函数，此函数的格式为：function functionname (content){……code}，content 变量就是请求连接的回显内容，在函数体内开发者可以对连接返回的信息进行再加工处理，注意这里的引号是半角引号。

2、 form 表单:

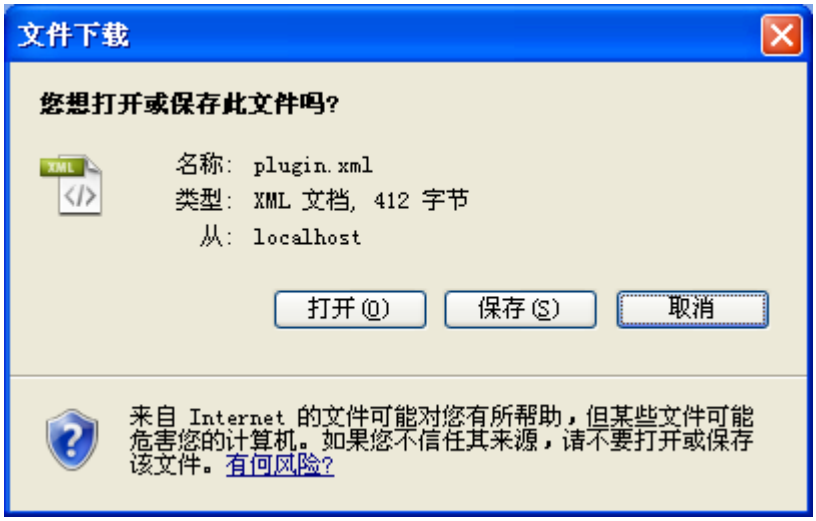
如 <form name="ajax" action="<?php echo self_url(__file__)?>php.php" target="'hello_word"'>表单内容</form>,处理的方式和上面的 A 连接方式是一样的，这里要注意，目前不支持对文件有上传。

然后打开插件信息生成器，填写如下信息，

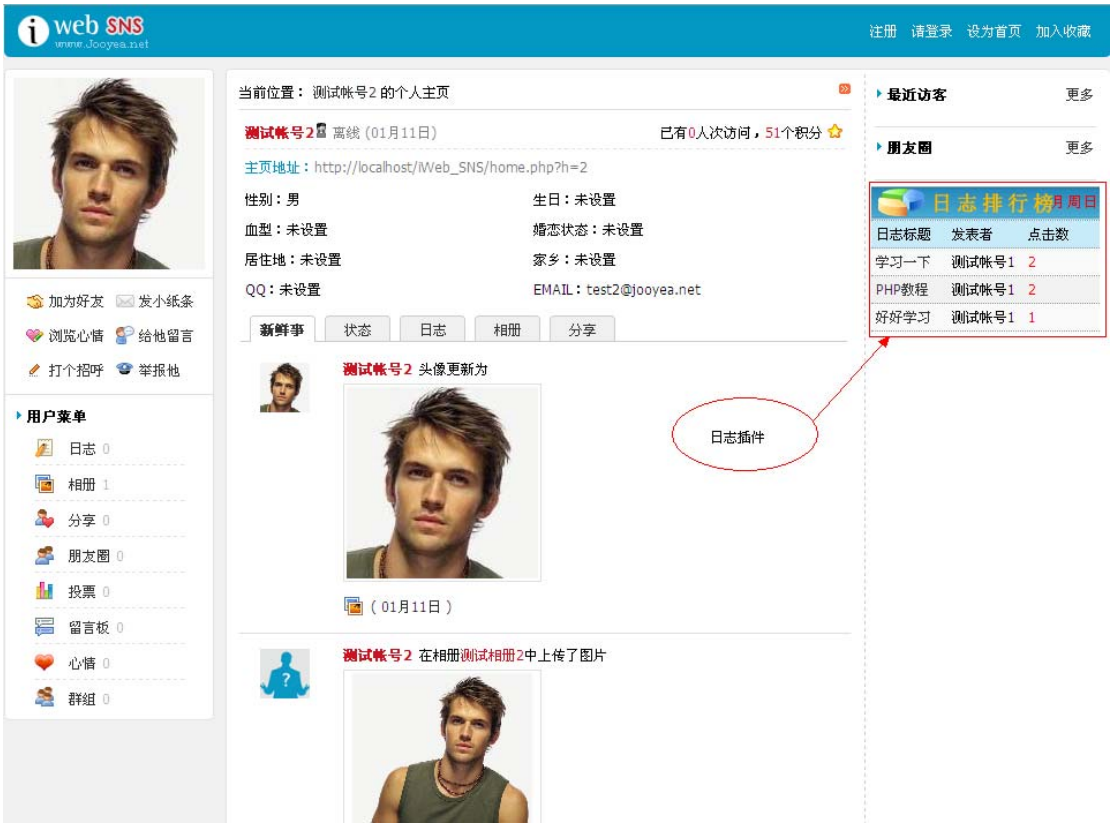
说明：关于权限的格式：

如：add:添加记录,del:删除记录，前面为权限码，后面为权限名，各权限之间用逗号分开，用户如果想实现权限的验证属性实用 plugin_check_right(__file__,”add”)，通过返回 true，未通过返回 false，这里的冒号和逗号者是英文半角,如果无权限功能，则不必添加。

点击提交：



把文件保存到自己的插件目录下，一个插件就完成了，后台安装后前台显示如图：



重要文件及函数说明：

1、 Includes.php 文件，这是开发插件的基础文件，此文件有几个重要开关变量：

$session_power 控制是否开启 session，值如果为 true 为开启，不开启则不用写此变量;

$iweb_power 控制着是否开启对系统数据库的支持，并且兼容系统的平滑 SI 的支持，开启为 true，不开启则不用此变量；

**$getpost_powe**控制是否开启系统对$_GET 和$_POST 进行封装的支持，调用函数为

get_args(name),系统会自动取 GET 和 POST 上来的变量，如果有同名的变量，函数会返回 POST 的上来的变量，如果查询不到返回 NULL，开启为 true，不开启则不用写此变量。

2、 关于 Api_Proxy(参数 1，参数 2,……)的使用：

用户调用的所有的 API 都是通过此方法，参数 1 是 API 方法的名称，参数 2、参数 3……是对应的 API 的参数。

3、 关于路径问题：

用户自己也可以编写自己的 CSS 文件的 JS 文件，也可以引用自己的图片，在引用这些文件的时候，一定用要调用 self_url(__file__)函数，此函数返回到开发的插件的根目录，例如开发者要引用自己目录下的 CSS 目录的 test.cee 文件，应该写成<link href="<?php echo self_url(__file__)?>css/test.css" rel="stylesheet" type="text/css" />说明一下，在 css/test.css 前不要添加“/”符号。引用 JS 文件格式同上。

4、 关于权限验证的问题：

用户自定义的

5、 关于提交信息问题：

如果开发者开发的插件(这里是指 APP 应用和 APP+Widget)有信息的提交和页面的跳转，要先在自己的 actions.xml 文件里配制上相应的信息，然后在开发的文件里应用系统提供的 do_action(__file__)函数,

格式：

<form action="<?php echo do_action(__file__);?>test_add" method="post">

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE actions SYSTEM "../Actions.dtd">
<actions>
    <package name="test">
        <action name="show" file="action/PersonAction.php" class="Person" method="show">
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
        <action name="add" file="action/PersonAction.php" method="add">
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
        <action name="del" file="action/PersonAction.php" method="del">
            <return name="input">error.php</return>
            <return name="success">view/helloword_list.php</return>
        </action>
    </package>
</actions>
```

首先是调用<?php echo do_action(__file__);?>，然后是包名_Action 动作名，如上图开发者 Actions.xml 的配制信息的标注。

对应的 action 节点 file 属性文件 PersonAction.php 的内容

```php
<?php
//$getpost_power=true;
$iweb_power=true;
$session_power=true;
include(dirname(__file__)."/../../includes.php");
function lis()
{
    return "success";
}
function add()
{
    if(!check())return "input";
    $db=new dbex();
    dbplugin('w');
    $db->exeUpdate("insert into helloword(msg) values ('".get_args('msg')."')");
    return "success";
}
function del()
{
    if(!check())return "input";
    $db=new dbex();
    dbplugin('w');
    $db->exeUpdate("delete from helloword where id=".get_args('id'));
    return "success";
}
function check()
```

对应的 return 节点 helloword_list.php 文件内容

```php
<?php
$iweb_power=true;
$session_power=true;
include(dirname(__file__)."/../../includes.php");
$t_hello = "helloword";
$db=new dbex();
dbplugin('r');
$sql="select * from $t_hello";
$result=$db->getRs($sql);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title></title>
<link rel="stylesheet" type="text/css" href="skin/default/css/row_menus.css">
<link rel="stylesheet" type="text/css" href="skin/default/css/right.css">
</head>
<body>
<div id="test_helloword">
<h1 style="text-align:center"><?php if(!is_null(get_sess_username()))echo get_sess_username()."你好!";?></h1>
<div class='main_right_title'>
<table width="100%">
<tr><td>编号</td><td>信息</td><td>操作</td></tr>
<?php foreach( $result as $rs){?>
<tr><td><?php echo $rs['id'];?></td><td><?php echo $rs['msg'];?></td><td><a href="<?php echo do_action(__file__);?>test_del&id=<?php
echo $rs['id'];?>" name="ajax" target="'test_helloword'">删除</a></td></tr>
```

实现了插件的 MVC 架构。

6、 如果需要使自己的页面显示在个人主页 home 和主页 main 页面的主显区域：

1)、通过连接，设定 target 属性为 frame_content。

2)、通过 JS，用 document.getElementById("frame_content").src=url;的方式改变。

以上两的 Url,要用上面的动作处理格式(<?php echo do_action(__file__);?>包名_Action 动作名)来实现，不要自己写地址，否则会找不到你填写的地址的。

相关 API:

说明：api 函数中绿色字表示返回的数据类型。

用户api

$fields 字段名称

user_id	用户 id
user_marry	婚姻状态
user_email	电子邮箱
user_name	名字
user_sex	性别
user_call	电话
birth_province	出生省份
birth_city	出生城市
reside_province	居住省份
reside_city	居住城市
user_ico	头像
user_add_time	注册时间
birth_year	出生年份
birth_month	出生月份
birth_day	出生日
user_blood	血型
user_qq	qq 号
creat_group	创建的群组 id（多个用“,”间隔）
join_group	加入的群组 id（多个用“,”间隔）
guest_num	访问量
integral	积分数
lastlogin_datetime	最后登录时间
hidden_pals_id	已屏蔽的好友
hidden_type_id	已屏蔽的类别

（注：标有“*”红色星号的参数表示必填参数）

```
Array  user_self_by_uid(str $fields="*",str $id)
```

说明：　根据用户 id 值取得用户信息。

$fields　要取得的字段值数据，默认为全部。

$id(*)　用户 id 值，多个用“,”隔开。

Array user_self_by_gnum(str $fields="*",int $num=10)
说明： 取得访问量最多的用户。
$fields 要取得的字段值数据，默认为全部。
$num 取得数据的条数。

Array user_self_by_integral(str $fields="*",int $num=10)
说明： 取得积分最多的用户。
$fields 要取得的字段值数据，默认为全部。
$num 取得数据的条数。

Array user_self_by_logindate(str $fields="*",str $date='',int $num=10)
说明： 根据最后登录时间取得用户信息。
$fields 要取得的字段值数据，默认为全部。
$date 日期时间，例如”2000-01-01~2010-01-01”。
$num 取得数据条数。

Array user_self_by_new(str $fields="*",int $num=4)
说明： 根据注册的时间取得用户信息。
$fields 要取得的字段值数据，默认为全部。
$num 取得数据条数。

Array user_recommend_get(str $field="*")
说明： 获取推荐会员的信息。
$fields 要取得的字段值数据，默认为全部。

日志api

日志$fields 字段名称

log_id	日志 id
user_id	作者 id
user_name	作者名
user_ico	作者头像
log_title	日志标题
log_sort_name	日志分类名称
log_content	日志内容
add_time	发布时间
hits	点击数
comments	回复数

Array blog_self_by_bid(str $fields="*",str $id)
说明： 根据日志 id 取得公开的日志信息。
$fields 要取得的字段值数据，默认为全部。

$id(*)　日志的 id 值字符串，当要取得多个时用 ”，” 分隔开。

Array blog_self_by_uid(str $fields="*", str $id,str $sort_id='',int $num='')
说明：　根据作者 id 取得公开的日志信息；
$fields　要取得的字段值数据，默认为全部。
$id(*)　作者的 id 值组合，当要取得多个时用 ”，” 分隔开。
$sort_id　日志的分类，当要取得多个时用 ”，” 分隔开。
$num　显示数据条数。

Array blog_self_by_hits(str $fields="*", str $order="desc", int $num=10)
说明：　根据日志阅读数量的多少来取得$num 篇公开的日志信息；
$fields　要取得的字段值数据，默认为全部。
$order　排列是顺序，desc 为倒序，asc 为正序。
$num　取得数据条数。

Array blog_self_get_new(str $fields="*",int $num=10);
说明：　取得最新发布$num 篇公开的日志信息。
$fields　要取得的字段值数据，默认为全部。
$num　取得数据条数。

Array blog_self_by_date(str $fields="*",int $date,str $by_col='hits',str $order="desc",int $num=10)
说明：　根据输入的日期，取得$num 条公开的日志信息并按照阅读数或回复数排序。
$fields　要取得的字段值数据，默认为全部。
$date　发布的时间段，开始日期～结束日期。
$ by_col　根据点击量或回复数排序。hits 为点击量；comments 为回复量；
$order　设置 by_col 的排序，desc 为倒序，asc 为正序。
$num　取得数据条数。

Array blog_sort_by_uid(str $id="")
说明：　根据会员 id 获取该会员的日志分类。
$id　会员 id。

相册api

相册$fields 字段信息。

album_id	相册 id 值
album_name	相册名称
user_id	创建者 id
user_name	创建者名字
album_info	相册描述
add_time	建立时间
album_skin	相册封面

photo_num	包含照片数量
comments	评论数

照片$fields 字段信息。

photo_id	照片 id
user_id	创建者 id
photo_name	创建者名字
photo_information	照片描述
add_time	创建时间
photo_src	照片地址
photo_thumb_src	缩略图地址
album_id	所在相册 id
comments	评论数

（注：标有“*”红色星号的参数表示必填参数）

Array album_self_by_aid(str $fields="*",str $id)
说明： 根据相册 id 取得公开相册信息
$ fields 要取得的字段值数据，默认为全部。
$id(*) 日志 id 值组合，多个 id 值用“,”分隔开。

Array album_self_by_uid(str $fields="*", str $id)
说明： 根据相册作者 id 取得公开相册信息。
$ fields 要取得的字段值数据，默认为全部。
$id(*) 用户 id 值组合，多个 id 值用“,”分隔开。

Array album_self_by_coms(str $fields="*",int $num=10)
说明： 取得$num 个回复数最多的相册信息。
$fields 要取得的字段值数据，默认为全部。
$num 取得相册的数据条数。

Array album_self_get_new(str $fields="*",int $num=10);
说明： 取得$num 个最新的相册信息。
$fields 要取得的字段值数据，默认为全部。
$num 取得相册的数据条数。

Array album_photo_by_photoid(str $fields="*",str $id)
说明： 根据照片 id 取得信息。
$fields： 要取得的字段值数据，默认为全部。
$id(*)： 照片的 id 值。

Array album_photo_by_aid(str $fields="*",str $id)
说明： 根据相册 id 取得照片信息。
$fields： 要取得的字段值数据，默认为全部。
$id(*)： 相册的 id 值。

Array album_photo_by_uid(str $fields="*",str $id,int $num=10)
说明： 根据照片用户 id 取得信息。
$fields 要取得的字段值数据，默认为全部。
$id(*) 照片作者的 id 值。
$num 取得相册的数据条数。

Array album_photo_by_coms(str $fields="*",int $num=10)
说明： 取得评论最多的$num 个照片信息。
$fields 要取得的字段值数据，默认为全部。
$num 取得相册的数据条数。

Array album_photo_get_new(str $fields="*",int $num=10)
说明： 取得最新的$num 个照片信息。
$fields 要取得的字段值数据，默认为全部。
$num 取得相册的数据条数。

分享api

分享$field 字段

s _id	分享 id 值
type_id	分享类别 id
user_id	作者 id
user_name	作者名称
user_ico	作者头像地址
content	分享内容
s_title	分享标题
out_link	外部链接地址
add_time	添加时间
for_content_id	站内分享内容的 id 值
comments	评论数
movie_thumb	视频分享缩略图地址
movie_link	视频分享的地址

（注：标有“*”红色星号的参数表示必填参数）

Array share_self_by_sid(str $fields="*", str $id)
说明： 根据分享 id 值取得分享信息。
$fields 要取得的字段值数据，默认为全部。
$id(*) 分享的 id 值组合，多值用“,”分隔开。

Array share_self_by_uid(str $fields="*", str $id)
说明： 根据发起者 id 值取得分享信息。
$fields 要取得的字段值数据，默认为全部。

$id(*)　发起者的 id 值，多值用“,”分隔开。

Array　share_self_by_sort(str $fields="*",int $ sort, int $num=10)
说明：　根据分享类别取得分享信息。
$fields　要取得的字段值数据，默认为全部。
$sort　分享的类别 id。各个参数所代表的类别如下：
0：日志；1：群组；2：相册；3：照片；4：投票；5：链接；6：音乐；7：视频；8：话题；
$num　显示的数据条数。

朋友圈api

user_id	主人 id
pals_id	好友 id
pals_sort_name	好友分类名称
pals_name	好友名字
pals_sex	好友性别
add_time	成为好友时间
pals_ico	好友头像
accepted	接受状态。0：没有得到确认的；1：同意列为好友；2：双方互为好友

（注：标有“*”红色星号的参数表示必填参数）

Array　pals_self_by_paid(str $fields="*",int$id , int $accept=1)
说明：　取得当前用户好友数据。
$fields　要取得的字段值数据，默认为全部。
$id　显示指定好友的 id 值，多个组合用“,”分隔开。
$accept　好友确认状态。

Array　pals_self_by_sort(str $fields="*",str $sort="")
说明：　根据分类取得当前用户的好友数据。
$fields　要取得的字段值数据，默认为全部。
$sort(*)　好友分类的类别名称。

Array　pals_self_by_online(str $fields="*",str $num="")
说明：　取得当前在线的好友信息。
$fields　要取得的字段值数据，默认为全部。
$num　要显示的数据量。默认为全部。

Array　pals_self_by_uid(str $fields="*",str $id,str $num="")
说明：　取得某用户的朋友圈。
$fields　要取得的字段值数据，默认为全部。
$id(*)　用户 id 值。
$num　要显示的数据量。默认为全部。

Array pals_sort(str $uid)
说明： 取得指定会员的朋友圈。
$uid(*) 会员 id 值。

Array pals_sort_def()
说明： 取得默认朋友圈。

群组api

群组$fields 字段值

group_id	群组 id
add_userid	创建者 id
member_count	组员人数
group_name	群组名称
group_resume	群组简介
group_time	创建时间
group_creat_name	创建者名字
group_logo	群组 logo 地址
group_join_type	群组加入方式
group_type	群组类别名称
group_type_id	群组类别 id
affiche	公告
group_tag	群组标签
subjects_num	话题数
comments	评论数
group_manager_name	管理员名（多个用“,”间隔）

话题$fields 字段值

subject_id	话题 id
group_id	所属群组 id
user_id	作者 id
user_name	作者名字
user_ico	作者头像地址
title	话题标题
content	话题内容
add_time	发布时间
hits	点击数
comments	回复数

成员$fields 字段值

id	成员 id

group_id	所属群组 id
user_id	成员会员 id
user_name	成员名字
user_sex	成员性别
user_ico	成员头像地址
state	审核状态
role	角色
add_time	发布时间

（注：标有“*”红色星号的参数表示必填参数）

```
Array  group_self_by_gid(str $fields="*", str $id);
```
说明：　根据群组的 id 值取得公开群组信息。
$fields　要取得的字段值数据，默认为全部
$id(*)　群组 id 值，多个用“,”分割。

```
Array  group_self_get_by_uid(str $fields="*", str $id)
```
说明：　根据创建者取得群组信息。
$fields　要取得的字段值数据，默认为全部。
$id(*)　用户 id 值，多个用“,”分割。

```
Array  group_self_get_new(str $fields="*",int $num=10)
```
说明：　取得最新创建的群组。
$fields　要取得的字段值数据，默认为全部。
$num　数据量。

```
Array  group_self_get_by_subs(str $fields="*",int $num=10)
```
说明：　取得话题数最多的群组。
$fields　要取得的字段值数据，默认为全部。
$num　数据量。

```
Array  group_self_by_pals(str $fields="*")
```
说明：　取得好友所加入的群组。
$fields　要取得的字段值数据，默认为全部。

```
Array  group_self_by_memberCount(str $fields="*",int $num=10)
```
说明：　取得成员数最多的群组。
$fields　要取得的字段值数据，默认为全部。
$num　数据量。

```
Array  group_sub_by_sid(str $fields="*", str $id)
```
说明：　根据话题 id 取得话题数据。
$fields　要取得的字段值数据，默认为全部。
$id　话题 id 值，多个用“,”分隔开。

Array group_sub_by_gid(str $fields="*", str $id,int $num=10)
说明： 根据群组 id 值取得话题数据。
$fields 要取得的字段值数据，默认为全部。
$id 群组 id 值，多个用“,”分隔开。
$num 数据量。

Array group_sub_by_uid(str $fields="*", str $id,int $num=10)
说明： 根据作者取得话题数据。
$fields 要取得的字段值数据，默认为全部。
$id 用户 id 值，多个用“,”分隔开。
$num 数据量。
Array group_sub_by_date(str $fields="*",str $date='',str $col_by='hits',str $order='desc',int $num=10)
说明： 根据发布日期时间取得话题数据结果按照点击量或回复量排列。
$fields 要取得的字段值数据，默认为全部。
$date 发布的日期范围，例：”2000-01-01～2000-02-01”。
$col_by 排列的依据字段。”hits”为点击量；’comments’为回复数。
$order 排序类型。
$num 数据量。

Array group_member_manager(str $fields="*", str $gid)
说明： 根据群组 id 取得群组成员数据。
$fields 要取得的字段值数据，默认为全部。
$gid 群组 id 值，多个用“,”分隔开。

Array group_member_by_role(str $gid, str $uid)
说明： 根据群组 id 和会员 id 取得会员在群组中的角色。
$gid 群组 id 值。
$uid 会员 id 值。

Array group_member_by_gid (str $fields="*", str $gid, str $state)
说明： 根据群组 id 和成员审核状态取得群组成员数据。
$fields 要取得的字段值数据，默认为全部。
$gid 群组 id 值，多个用“,”分隔开。
$state 会员审核状态。

Array group_sort_by_self ()
说明： 获取分组类别。

留言板api

留言板$fields 字段值

from_user_id	留言者 id
from_user_name	留言者名字
from_user_ico	留言者头像地址
message	留言内容
to_user_id	被留言者 id
add_time	留言时间
readed	是否被阅读

（注：标有“*”红色星号的参数表示必填参数）

```
Array  msgboard_self_by_uid(str$fields="*",str $uid="",int$is_read="" ,
int $num,str $date="")
```

说明：　　取得本人留言板数据内容。

$fields　　要取得的字段值数据，默认为全部。

$uid　　会员 id。

$is_read　　是否已读。0 未读；1 已读

$date　　发布日期的时间范围。

```
Array  msgboard_set(str $to_user_id="")
```

说明：　　设置指定会员的留言板为已读。

$to_user_id　　会员 id。

投票api

投票$fields 字段信息

p_id	投票 id 值
user_id	发起者 id
username	发起者名字
user_ico	发起者头像地址
subject	投票主题名称
voternum	投票数
comments	评论数
multiple	是否为多选；0 为单选；1 为多选；
sex	限制投票性别，0 为女；1 为男；2 为不限；
credit	投票的总积分奖励
percredit	每次投票的积分奖励数
expiration	投票截止的日期
lastvote	最后投票的时间
dateline	发起投票的时间
message	投票的描述
summary	投票总结
option	前两项的序列化字符串

（注：标有“*”红色星号的参数表示必填参数）

Array poll_self_by_pollid(str $fields="*",str $id="")
说明： 根据投票 id 取得投票信息。
$fields 要取得的字段值数据，默认为全部。
$id 投票 id 值，多个用”,”分割。

Array poll_self_by_uid(str $fields="*",str $id="")
说明： 根据投票发起者的 id 值取得信息。
$fields 要取得的字段值数据，默认为全部。
$id 用户 id 值，多个用”,”分割。

Array poll_self_by_date (str $fields="*",str $date="",str $by_col='voternum',str $order="desc",int $num=10)
说明： 根据发布日期，按投票数或回复数来排序。
$fields 要取得的字段值数据，默认为全部。
$date 发布日期。
$by_col 排序的字段。voternum:参与人数；comments:回复数；
$order 排列顺序。desc 降序 asc 升序。
$num 数据量。

Array poll_self_by_new(str $fields="*" , str $order="desc", int $num=20)
说明： 根据投票发起投票的时间取得投票信息。
$fields 要取得的字段值数据，默认为全部。
$order 排列顺序。desc 降序 asc 升序。
$num 数据量。

Array poll_self_by_hot(str $fields="*", str $order="desc", int $num=20)
说明： 根据投票数取得投票信息。
$fields 要取得的字段值数据，默认为全部。
$order 排列顺序。desc 降序 asc 升序。
$num 数据量。

Array poll_self_by_reward(str$fields="*",str$order="desc",int $num=20)
说明： 根据投票奖励积分取得投票信息。
$fields 要取得的字段值数据，默认为全部。
$order 排列顺序。desc 降序 asc 升序。
$num 数据量。

小纸条api

inbox 收件箱$fields 字段信息

mess_id	收件箱纸条 id
mess_title	纸条标题
mess_content	纸条内容

from_user_id	发送者 id
from_user	发送者名字
from_user_ico	发送者头像地址
user_id	接收者 id
add_time	发送时间
readed	是否阅读；1 未读；0 已读

outbox 发件箱$fields 字段信息

mess_id	发件箱纸条 id
mess_title	纸条主题
mess_content	纸条内容
to_user_id	接收者的 id
to_user	接收者的名字
to_user_ico	接收者头像地址
user_id	发送者 id
state	接受者是否阅读；0 已读；1 未读；
add_time	发送时间

（注：标有“*”红色星号的参数表示必填参数）

```
Array  scrip_outbox_get_mine(str $fields="*",int $to_id="",int $is_read="" , str $date="")
```

说明：　　取得本人收件箱内的小纸条。

$fields　　要取得的字段值数据，默认为全部。

$to_id　　接收者的 id。

$is_read　　是否阅读。

$date　　发送时间段。例如“2000-01-01~2010-01-01”。

```
Array  scrip_inbox_get_mine(str $fields="*",str $date="",int $is_read="" , int $from_id="")
```

说明：　　取得本人发件箱内的小纸条。

$fields　　要取得的字段值数据，默认为全部。

$date　　取得数据的时间段。

$is_read　　阅读状态。

$from_id　　发件人的 id。

```
Bool  scrip_send(str $sender,str $title,str $content,int $to_id)
```

说明：　　给某个用户发送小纸条。

$sendor(*)　　发送者名称

$ title(*)　　小纸条标题。

$content(*)　　小纸条内容。

$to_id(*)　　接收者 id。

心情api

心情$fields 字段信息

mood_id	心情 id
user_id	会员 id
user_name	会员名称
user_ico	会员头像
mood	发送者名字
comments	发送者头像地址
add_time	发送时间

（注：标有“*”红色星号的参数表示必填参数）

```
Array  mood_self_by_mid (str $fields="*",str $id="")
```

说明：　根据心情 id 取得心情信息。

$fields　要取得的字段值数据，默认为全部。

$id(*)　心情 id。

```
Array  mood_self_by_uid (str $fields="*",str $id="")
```

说明：　根据会员 id 取得心情信息。

$fields　要取得的字段值数据，默认为全部。

$id(*)　会员 id。

访客api

访客$fields 字段信息

guest_id	访客 id
guest_user_id	访客会员 id
guest_user_name	访客会员名称
guest_user_ico	访客会员头像
user_id	会员 id
add_time	发送时间

（注：标有“*”红色星号的参数表示必填参数）

```
Array  guest_self_by_uid (str $fields="*",str $id="" ,int $num=6)
```

说明：　根据会员 id 取得访客信息。

$fields　要取得的字段值数据，默认为全部。

$id(*)　会员 id。

消息api

提醒$fields 字段信息

id	提醒 id
user_id	会员 id
type_id	展示类型（好友申请，小纸条，留言板，打招呼，群组申请）
date	提醒时间
content	提醒内容
is_focus	显示位置（0-首页右上角 1-首页我的空间动态提醒）
from_uid	提醒会员 id
from_uname	提醒会员名称
from_uico	提醒会员头像
link	链接地址
count	提醒次数

新鲜事$fields 字段信息

id	访客 id
type_id	首页展示类型（新鲜事 状态 日志 相册 分享）
content	新鲜事内容
user_id	会员 id
user_name	会员名称
user_ico	会员头像
date_time	触发时间
update_time	更新时间
for_content_id	新鲜事的来源 ID
mod_type	细化的模块类型
title	新鲜事标题

（注：标有“*”红色星号的参数表示必填参数）

```
Array  message_get(str $mod,str $is_focus ,int $num=6)
```

说明：　　根据模块与展示位置获取提醒信息。

$mod　　模块类型。

$is_focus　　展示位置。

$num　　数据量。

```
Array  message_get_remind(str $is_focus ,int $num=6)
```

说明：　　根据展示位置获取提醒信息。

$is_focus　　展示位置。

$num　　数据量。

```
Array  message_get_remind_count(int $uid)
```

说明：　　根据会员 id 获取提醒数。

$uid　　会员 id。

Array message_get_affair_uid(int $id,str $type,int $num=6)
说明： 根据会员 id 和模块类型获取新鲜事儿信息。
$id(*) 会员 id。
$type 模块类型。
$num 数据量。

Bool message_set_affair(int $id ,str $title,str $content, int $show_type_id,str $mod_type)
说明： 添加新鲜事。
$id 新鲜事来源 ID。
$title 新鲜事标题。
$content 新鲜事内容。
$show_type_id 展示模块 ID。
$mod_type 模块类型。

Bool message_set_remind(int $touid,str $content, str $link,str $is_focus , int $type)
说明： 添加新提醒。
$touid(*) 会员 id。
$content 提醒内容。
$link 提醒显示位置。
$is_focus 展示位置。
$type 提醒展示位置。

Array message_set_del(str $type, str $rid, int $uid)
说明： 根据消息类型、会员 id 或消息 id 删除消息。
$type(*) 消息类型(新鲜事儿、提醒)。
$rid(*) 提醒 id。
$uid(*) 会员 id。

Bool message_set_del(str $rid, int $uid)
说明： 根据会员 id 或消息 id 删除提醒。
$rid(*) 提醒 id。
$uid(*) 会员 id。

Array message_set_del(str $rid)
说明： 根据新鲜事儿 id 删除新鲜事儿。
$rid(*) 新鲜事儿 id。

插件api

插件$fields 字段信息

id	插件 id
title	插件标题
name	插件名称
valid	是否激活
autoorder	是否自由定制
reg_date	添加时间
image	图片路径
comment_num	评论个数
use_num	使用人数
info	插件信息

（注：标有“*”红色星号的参数表示必填参数）

```
Bool  plugins_set_mine(int $id, int $is_del=0)
```

说明：　根据插件 id 设置插件。

$id(*)　插件 id。

```
Array  plugins_get_all()
```

说明：　获取所有插件信息。

```
Array  plugins_get_pid(int $id)
```

说明：　根据插件 id 获取插件信息。

$id(*)　插件 id。

```
Array  plugins_get_mine ()
```

说明：　根据 session 中当前的会员获取该会员的插件信息。