# **方维P2P短信接口修改**

原创 2016年11月16日 13:53:01

* 标签：
* [方维](http://so.csdn.net/so/search/s.do?q=%E6%96%B9%E7%BB%B4&t=blog" \t "http://blog.csdn.net/mindev/article/details/_blank) /
* [方维p2p](http://so.csdn.net/so/search/s.do?q=%E6%96%B9%E7%BB%B4p2p&t=blog" \t "http://blog.csdn.net/mindev/article/details/_blank) /
* [方维插件](http://so.csdn.net/so/search/s.do?q=%E6%96%B9%E7%BB%B4%E6%8F%92%E4%BB%B6&t=blog" \t "http://blog.csdn.net/mindev/article/details/_blank) /
* [方维短信插件](http://so.csdn.net/so/search/s.do?q=%E6%96%B9%E7%BB%B4%E7%9F%AD%E4%BF%A1%E6%8F%92%E4%BB%B6&t=blog" \t "http://blog.csdn.net/mindev/article/details/_blank) /
* [短信宝](http://so.csdn.net/so/search/s.do?q=%E7%9F%AD%E4%BF%A1%E5%AE%9D&t=blog" \t "http://blog.csdn.net/mindev/article/details/_blank)
* 1684

我的一个同事最近在替换方维p2p系统的短信接口，版本为3.5。短信接口我们公司使用的是短信宝( 网址：[http://www.smsbao.com](http://www.smsbao.com/" \t "http://blog.csdn.net/mindev/article/details/_blank))的短信接口，短信宝的接口稳定，快速，值得向大家推荐。因为方维p2p的用户使用量非常大，而且我研究了一下，方维p2p的其他版本的短信接口替换都大同小异的，所以我在这里和大家分享一下修改全过程，让有这方面业务需求的小伙伴能够快速学会。

为了帮助更多的用户解决方维p2p的短信接口替换问题，我已经把功能代码打包成了插件，提供给了短信宝官网，直接去他们的网站下载即可。下载地址：[http://www.smsbao.com/plugin/80.html](http://www.smsbao.com/plugin/80.html" \t "http://blog.csdn.net/mindev/article/details/_blank)

下面我来阐述一下替换过程：   
1.首先要开启短信的功能，在系统设置->邮件与短信中开启设置。如图所示：



2.创建一个sms\_trans.php文件上传至项目根目录下的system\sms\smsbao这个文件下(smsbao目录需要创建)，代码如下：

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112

3.创建smsbao\_sms.php这个文件上传至项目根目录下的system/sms文件下，代码如下：

<?php

class sms\_trans

{/\*\*

\* 脚本执行时间。－1表示采用PHP的默认值。

\*

\* @access private

\* @var integer $time\_limit

\*/var $time\_limit = -1;

/\*\*

\* 在多少秒之内，如果连接不可用，脚本就停止连接。－1表示采用PHP的默认值。

\*

\* @access private

\* @var integer $connect\_timeout

\*/var $connect\_timeout = -1;

/\*\*

\* 连接后，限定多少秒超时。－1表示采用PHP的默认值。此项仅当采用CURL库时启用。

\*

\* @access private

\* @var integer $stream\_timeout

\*/var $stream\_timeout = -1;

/\*\*

\* 是否使用CURL库来连接。false表示采用fsockopen进行连接。

\*

\* @access private

\* @var boolean $use\_curl

\*/var $use\_curl = false;

/\*\*

\* 构造函数

\*

\* @access public

\* @param integer $time\_limit

\* @param integer $connect\_timeout

\* @param integer $stream\_timeout

\* @param boolean $use\_curl

\* @return void

\*/function \_\_construct($time\_limit = -1, $connect\_timeout = -1, $stream\_timeout = -1, $use\_curl = false)

{//$this->transport($time\_limit, $connect\_timeout, $stream\_timeout, $use\_curl);$this->time\_limit = $time\_limit;$this->connect\_timeout = $connect\_timeout;$this->stream\_timeout = $stream\_timeout;$this->use\_curl = $use\_curl;

}

/\*\*

\* 请求远程服务器

\*

\* @access public

\* @param string $url 远程服务器的URL

\* @param mix $params 查询参数，形如bar=foo&foo=bar；或者是一维关联数组，形如array('a'=>'aa',...)

\* @param string $method 请求方式，是POST还是GET

\* @param array $my\_header 用户要发送的头部信息，为一维关联数组，形如array('a'=>'aa',...)

\* @return array 成功返回一维关联数组，形如array('header'=>'bar', 'body'=>'foo')，

\* 重大错误程序直接停止运行，否则返回false。

\*/function request($url, $params = '', $method = 'POST', $my\_header = '')

{$fsock\_exists = function\_exists('fsockopen');$curl\_exists = function\_exists('curl\_init');

if (!$fsock\_exists && !$curl\_exists)

{die('No method available!');

}

if (!$url)

{die('Invalid url!');

}

if ($this->time\_limit > -1)//如果为0，不限制执行时间

{

set\_time\_limit($this->time\_limit);

}

$method = $method === 'GET' ? $method : 'POST';$response = '';$temp\_str = '';

/\* 格式化将要发要送的参数 \*/if ($params && is\_array($params))

{foreach ($params AS $key => $value)

{$temp\_str .= '&' . $key . '=' . $value;

}$params = preg\_replace('/^&/', '', $temp\_str);

}

/\* 如果fsockopen存在，且用户不指定使用curl，则调用use\_socket函数 \*/if ($fsock\_exists && !$this->use\_curl)

{$response = $this->use\_socket($url, $params, $method, $my\_header);

}/\* 只要上述条件中的任一个不成立，流程就转向这里，这时如果curl模块可用，就调用use\_curl函数 \*/elseif ($curl\_exists)

{$response = $this->use\_curl($url, $params, $method, $my\_header);

}

/\* 空响应或者传输过程中发生错误，程序将返回false \*/if (!$response)

{return false;

}

return $response;

}

/\*\*

\* 使用fsockopen进行连接

\*

\* @access private

\* @param string $url 远程服务器的URL

\* @param string $params 查询参数，形如bar=foo&foo=bar

\* @param string $method 请求方式，是POST还是GET

\* @param array $my\_header 用户要发送的头部信息，为一维关联数组，形如array('a'=>'aa',...)

\* @return array 成功返回一维关联数组，形如array('header'=>'bar', 'body'=>'foo')，

\* 否则返回false。

\*/function use\_socket($url, $params, $method, $my\_header)

{$query = '';$auth = '';$content\_type = '';$content\_length = '';$request\_body = '';$request = '';$http\_response = '';$temp\_str = '';$error = '';$errstr = '';$crlf = $this->generate\_crlf();

if ($method === 'GET')

{$query = $params ? "?$params" : '';

}else

{$request\_body = $params;$content\_type = 'Content-Type: application/x-www-form-urlencoded' . $crlf;$content\_length = 'Content-Length: ' . strlen($request\_body) . $crlf . $crlf;

}

$url\_parts = $this->parse\_raw\_url($url);$path = $url\_parts['path'] . $query;

if (!empty($url\_parts['user']))

{$auth = 'Authorization: Basic '

. base64\_encode($url\_parts['user'] . ':' . $url\_parts['pass']) . $crlf;

}

/\* 格式化自定义头部信息 \*/if ($my\_header && is\_array($my\_header))

{foreach ($my\_header AS $key => $value)

{$temp\_str .= $key . ': ' . $value . $crlf;

}$my\_header = $temp\_str;

}

/\* 构造HTTP请求头部 \*/$request = "$method $path HTTP/1.0$crlf"

. 'Host: ' . $url\_parts['host'] . $crlf

. $auth

. $my\_header

. $content\_type

. $content\_length

. $request\_body;

if ($this->connect\_timeout > -1)

{$fp = @fsockopen($url\_parts['host'], $url\_parts['port'], $error, $errstr, $connect\_timeout);

}else

{$fp = @fsockopen($url\_parts['host'], $url\_parts['port'], $error, $errstr);

}

if (!$fp)

{return false;//打开失败

}

if (!@fwrite($fp, $request))

{return false;//写入失败

}

while (!feof($fp))

{$http\_response .= fgets($fp);

}

if (!$http\_response)

{return false;//空响应

}

$separator = '/\r\n\r\n|\n\n|\r\r/';list($http\_header, $http\_body) = preg\_split($separator, $http\_response, 2);

$http\_response = array('header' => $http\_header,//header肯定有值'body' => $http\_body);//body可能为空

@fclose($fp);

return $http\_response;

}

/\*\*

\* 使用curl进行连接

\*

\* @access private

\* @param string $url 远程服务器的URL

\* @param string $params 查询参数，形如bar=foo&foo=bar

\* @param string $method 请求方式，是POST还是GET

\* @param array $my\_header 用户要发送的头部信息，为一维关联数组，形如array('a'=>'aa',...)

\* @return array 成功返回一维关联数组，形如array('header'=>'bar', 'body'=>'foo')，

\* 失败返回false。

\*/function use\_curl($url, $params, $method, $my\_header)

{/\* 开始一个新会话 \*/$curl\_session = curl\_init();

/\* 基本设置 \*/

curl\_setopt($curl\_session, CURLOPT\_FORBID\_REUSE, true); // 处理完后，关闭连接，释放资源

curl\_setopt($curl\_session, CURLOPT\_HEADER, true);//结果中包含头部信息

curl\_setopt($curl\_session, CURLOPT\_RETURNTRANSFER, true);//把结果返回，而非直接输出

curl\_setopt($curl\_session, CURLOPT\_HTTP\_VERSION, CURL\_HTTP\_VERSION\_1\_0);//采用1.0版的HTTP协议

$url\_parts = $this->parse\_raw\_url($url);

/\* 设置验证策略 \*/if (!empty($url\_parts['user']))

{$auth = $url\_parts['user'] . ':' . $url\_parts['pass'];

curl\_setopt($curl\_session, CURLOPT\_USERPWD, $auth);

curl\_setopt($curl\_session, CURLOPT\_HTTPAUTH, CURLAUTH\_BASIC);

}

$header = array();

/\* 设置主机 \*/$header[] = 'Host: ' . $url\_parts['host'];

/\* 格式化自定义头部信息 \*/if ($my\_header && is\_array($my\_header))

{foreach ($my\_header AS $key => $value)

{$header[] = $key . ': ' . $value;

}

}

if ($method === 'GET')

{

curl\_setopt($curl\_session, CURLOPT\_HTTPGET, true);$url .= $params ? '?' . $params : '';

}else

{

curl\_setopt($curl\_session, CURLOPT\_POST, true);$header[] = 'Content-Type: application/x-www-form-urlencoded';$header[] = 'Content-Length: ' . strlen($params);

curl\_setopt($curl\_session, CURLOPT\_POSTFIELDS, $params);

}

/\* 设置请求地址 \*/

curl\_setopt($curl\_session, CURLOPT\_URL, $url);

/\* 设置头部信息 \*/

curl\_setopt($curl\_session, CURLOPT\_HTTPHEADER, $header);

if ($this->connect\_timeout > -1)

{

curl\_setopt($curl\_session, CURLOPT\_CONNECTTIMEOUT, $this->connect\_timeout);

}

if ($this->stream\_timeout > -1)

{

curl\_setopt($curl\_session, CURLOPT\_TIMEOUT, $this->stream\_timeout);

}

/\* 发送请求 \*/$http\_response = curl\_exec($curl\_session);

if (curl\_errno($curl\_session) != 0)

{return false;

}

$separator = '/\r\n\r\n|\n\n|\r\r/';list($http\_header, $http\_body) = preg\_split($separator, $http\_response, 2);

$http\_response = array('header' => $http\_header,//肯定有值'body' => $http\_body); //可能为空

curl\_close($curl\_session);

return $http\_response;

}

/\*\*

\* Similar to PHP's builtin parse\_url() function, but makes sure what the schema,

\* path and port keys are set to http, /, 80 respectively if they're missing

\*

\* @access private

\* @param string $raw\_url Raw URL to be split into an array

\* @author http://www.cpaint.net/

\* @return array

\*/function parse\_raw\_url($raw\_url)

{$retval = array();$raw\_url = (string) $raw\_url;

// make sure parse\_url() recognizes the URL correctly.if (strpos($raw\_url, '://') === false)

{$raw\_url = 'http://' . $raw\_url;

}

// split request into array$retval = parse\_url($raw\_url);

// make sure a path key existsif (!isset($retval['path']))

{$retval['path'] = '/';

}

// set port to 80 if none existsif (!isset($retval['port']))

{$retval['port'] = '80';

}

return $retval;

}

/\*\*

\* 产生一个换行符，不同的操作系统会有不同的换行符

\*

\* @access private

\* @return string 用双引号引用的换行符

\*/function generate\_crlf()

{$crlf = '';

if (strtoupper(substr(PHP\_OS, 0, 3) === 'WIN'))

{$crlf = "\r\n";

}elseif (strtoupper(substr(PHP\_OS, 0, 3) === 'MAC'))

{$crlf = "\r";

}else

{$crlf = "\n";

}

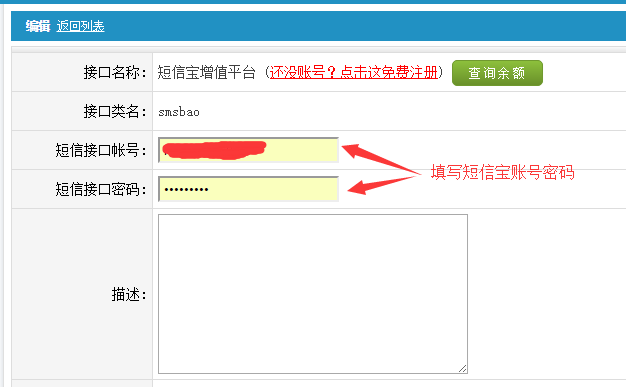
return $crlf;

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
* 158
* 159
* 160
* 161
* 162
* 163
* 164
* 165
* 166
* 167
* 168
* 169
* 170
* 171
* 172
* 173
* 174
* 175
* 176
* 177
* 178
* 179
* 180
* 181
* 182
* 183
* 184
* 185
* 186
* 187
* 188
* 189
* 190
* 191
* 192
* 193
* 194
* 195
* 196
* 197
* 198
* 199
* 200
* 201
* 202
* 203
* 204
* 205
* 206
* 207
* 208
* 209
* 210
* 211
* 212
* 213
* 214
* 215
* 216
* 217
* 218
* 219
* 220
* 221
* 222
* 223
* 224
* 225
* 226
* 227
* 228
* 229
* 230
* 231
* 232
* 233
* 234
* 235
* 236
* 237
* 238
* 239
* 240
* 241
* 242
* 243
* 244
* 245
* 246
* 247
* 248
* 249
* 250
* 251
* 252
* 253
* 254
* 255
* 256
* 257
* 258
* 259
* 260
* 261
* 262
* 263
* 264
* 265
* 266
* 267
* 268
* 269
* 270
* 271
* 272
* 273
* 274
* 275
* 276
* 277
* 278
* 279
* 280
* 281
* 282
* 283
* 284
* 285
* 286
* 287
* 288
* 289
* 290
* 291
* 292
* 293
* 294
* 295
* 296
* 297
* 298
* 299
* 300
* 301
* 302
* 303
* 304
* 305
* 306
* 307
* 308
* 309
* 310
* 311
* 312
* 313
* 314
* 315
* 316
* 317
* 318
* 319
* 320
* 321
* 322
* 323
* 324
* 325
* 326
* 327
* 328
* 329
* 330
* 331
* 332
* 333
* 334
* 335
* 336
* 337
* 338
* 339
* 340
* 341
* 342
* 343
* 344
* 345
* 346
* 347
* 348
* 349
* 350
* 351
* 352
* 353
* 354
* 355
* 356
* 357
* 358
* 359
* 360
* 361
* 362
* 363
* 364
* 365
* 366
* 367
* 368
* 369
* 370
* 371
* 372
* 373
* 374
* 375
* 376
* 377
* 378
* 379
* 380
* 381

4.进行短信宝接口配置：在后台->系统配置->短信接口列表中，启用短信宝的配置，并进行编辑：



5.最后进行短信宝的接口测试，可以使用前台的注册功能测试，如图：



通过以上测试，短信接口修改算是完成了，是不是很方便呀。

到这儿方维p2p的短信接口修改算是完成了，在调试的时候也遇到了一些问题，我也一并分享给大家：   
1. 首先，要确认在php环境配置文件php.ini中开启了curl的扩展，并且服务器或空间商支持curl或者fsocket。   
2. 其次是短信宝接口配置一切都正确，但是短信还是发不出去。这时找到系统设置->队列管理->业务队列列表，全选列表数据，执行彻底删除操作，并执行队列发送。这样就可以了。发生此类问题的原因是队列任务卡死造成的。

如果还有什么不明白的地方，可以去短信宝官网([http://www.smsbao.com](http://www.smsbao.com/" \t "http://blog.csdn.net/mindev/article/details/_blank))，找他们的在线客服解决问题，他们的客服可是很热情的哦。

<http://blog.csdn.net/mindev/article/details/53185827> 原网址