

# 群发终端客户接口 V23

版本号: V23

更新时间: 2019 年 12 月 12 日

## 【使用备注】

1、老接口地址/api/可继续使用,但是不建议对接使用,对接建议使用/apinew/接口地址。接口调用前。请先验证用户名和密码,验证方式请参照第九章【用户认证方式示例代码以及接口调用示例代码】说明。使用新功能建议使用新接口/apinew/。

2、所有接口采用 utf-8 编码格式;如果需要传入中文、韩文、西班牙语时,请务必将接口 http 的 header 头的编码格式设为 utf-8,并将参数的字符编码 gbk, ASCII、BIG5 等原编码转为 utf-8 编码传入,方可避免编码不一致乱码情况。Php 转码请用 rawurlencode 函数。

3、smtp 对接发送示例代码请参照第十章节,无需进行 api 接口的对接

## 【API 接口对接发送的应用场景和特点要求】

1、适合批量同类型邮件对接发送,如单批次需要发送的邮箱数量超过 5000 个及以上数量的电子期刊、广告营销等邮件;

2、发送速度快,可跟踪,使用流程:用 API 接口上传地址池,上传邮件模板,建立发送任务,然后平台后台开始发送邮件;

3、整合较复杂,需要用户有二次开发能力,调用群发平台 API 和用户自有系统进行业务整合;

## 【SMTP 协议对接发送的应用场景和特点要求】

1、适合和客户业务系统对接发送触发类邮件,例如账单,注册信,验证码等非营销类邮件;

2、因为是标准协议,很多系统都已经支持 SMTP 邮件发送,所以无需二次开发,或开发工作量较小,即可和广泛的业务系统整合;

3、用户只需在用户登录后台的“发送域名管理---账号管理”添加 SMTP 发送账号和密码即可使用;

4、smtp 对接暂时无法获取无效地址列表详情,请悉知

# 目 录

|   |    |
|---|----|
| 一、帐号信息相关接口.....                             | 4  |
| 1.1 查询用户信息.....                             | 4  |
| 1.2 查询群发服务信息.....                           | 4  |
| 二、群发统计相关接口.....                             | 6  |
| 2.1 取得指定的群发任务或者指定发送日期的群发任务列表（Web 群发专用）..... | 6  |
| 2.2 获取指定日期的群发统计概览.....                      | 7  |
| 2.3 导出发送失败地址.....                           | 8  |
| 2.4 指定日期邮件批次发送扣点返点统计信息.....                 | 11 |
| 2.5 指定日期扣点信息.....                           | 12 |
| 2.6 发送日志信息查询.....                           | 12 |
| 三、跟踪统计相关接口（Web 群发专用）.....                   | 14 |
| 3.1 获取指定群发任务的跟踪统计概览.....                    | 14 |
| 3.2 获取指定任务邮件打开统计详情.....                     | 15 |
| 3.3 获取指定任务链接打开统计详情.....                     | 16 |
| 3.4 导出指定群发任务跟踪邮箱信息.....                     | 16 |
| 3.5 指定批次的邮件各链接的点击统计.....                    | 18 |
| 四、联系人分类相关接口.....                            | 19 |
| 4.1 取得联系人分类列表.....                          | 19 |
| 4.2 取得联系人分类详情.....                          | 19 |
| 4.3 添加联系人分类.....                            | 20 |
| 4.4 修改联系人分类.....                            | 20 |
| 4.5 删除联系人分类.....                            | 21 |
| 4.6 导出退订地址.....                             | 22 |
| 4.7 分类地址列表.....                             | 22 |
| 4.8 新增分类地址.....                             | 23 |
| 4.9 修改分类地址.....                             | 29 |
| 4.10 删除分类地址.....                            | 29 |
| 4.11 清空联系人分类.....                           | 29 |
| 4.12 邮件订阅.....                              | 30 |
| 4.13 邮件退订.....                              | 30 |
| 4.14 <a href="#">导入分类地址记录查询</a> .....       | 30 |
| 五、邮件模板相关接口.....                             | 32 |
| 5.1 创建群发任务.....                             | 32 |
| 5.2 上传邮件模板.....                             | 33 |
| 5.3 获取邮件模板检测状态和模板信息.....                    | 36 |
| 5.4 模板列表.....                               | 37 |
| 5.5 删除模板.....                               | 38 |
| 5.6 修改模板.....                               | 38 |
| 5.7 上传模板附件.....                             | 39 |

|   |                    |
|---|--------------------|
| 5.8 删除模板附件 .....                                  | 40                 |
| <b>六、群发任务相关接口 .....</b>                           | <b>40</b>          |
| 6.1 群发任务列表 .....                                  | 40                 |
| 6.2 创建群发任务 .....                                  | 42                 |
| 6.3 修改群发任务 .....                                  | 44                 |
| 6.4 删除群发任务 .....                                  | 45                 |
| <b>七、域名及域名下的邮箱账号 .....</b>                        | <b>45</b>          |
| 7.1 域名列表 .....                                    | 45                 |
| 7.2 添加域名 .....                                    | 46                 |
| 7.3 验证域名 .....                                    | 46                 |
| 7.4 为指定子账号用户添加域名(管理员账号才可调用) .....                 | 48                 |
| 7.5 删除指定子帐号的域名(管理员账号才可调用) .....                   | 48                 |
| 7.6 删除域名 .....                                    | 49                 |
| 7.7 指定域名下的账号列表 .....                              | 49                 |
| 7.8 添加账号到指定域名 .....                               | 50                 |
| 7.9 删除指定域名下的账号 .....                              | 50                 |
| 7.10 修改指定域名的账号 .....                              | 51                 |
| <b>八、其他 .....</b>                                 | <b>51</b>          |
| 8.1 客户自定义群发登录页面说明 .....                           | 51                 |
| <b>九、用户认证方式示例代码以及接口调用示例代码 .....</b>               | <b>52</b>          |
| 9.1 Python 用户认证示例代码 .....                         | 52                 |
| 9.2 C#用户认证示例代码: .....                             | 53                 |
| 9.3 java 用户认证示例代码: .....                          | 53                 |
| 9.4 php 用户认证示例代码 .....                            | 55                 |
| 9.5 Ruby On Rails 示例代码: .....                     | 56                 |
| 9.6 JS 用户认证示例代码: .....                            | 57                 |
| 9.7 php 验证用户名密码并调用接口的示例代码 .....                   | 58                 |
| 9.8 java 验证用户密码并调用上传附件接口示例代码 .....                | 59                 |
| <b>十、群发 smtp 对接方式发送邮件代码样例 .....</b>               | <b>63</b>          |
| 1、php 版本 smtp 对接发送 demo 代码 .....                  | 63                 |
| 2、python 版本 smtp 对接发送 demo 代码 .....               | 69                 |
| <a href="#">3、java 版本 smtp 对接发送 demo 代码 .....</a> | <a href="#">70</a> |

## 一、帐号信息相关接口

### 1.1 查询用户信息

接口功能：查询用户信息

接口地址：

```
http://www.bestedm.org/mm-ms/apinew/account.php?do=customer-info
```

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<customer>
  <id>用户 ID</id>
  <username>用户名称</username>
  <company>公司名称</company>
  <contact>联系人</contact>
  <phone>联系电话号码</phone>
  <mobile>移动电话号码</mobile>
  <email>联系邮件地址</email>
  <im>即时聊天工具地址</im>
  <address>通信地址</address>
</customer>
```

### 1.2 查询群发服务信息

接口功能：查询群发服务信息

接口地址：

```
http://www.bestedm.org/mm-ms/apinew/account.php?do=service-info
```

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<service>
  <id>用户 ID</id>
  <service_status>群发服务状态</service_status>
  <service_type>群发服务类型</service_type>
  <send_type>发送方式</send_type>
  <beginning_date>服务开始日期</beginning_date>
  <expiration_date>服务截止日期</expiration_date>
```

```
<qty_available>剩余有效群发量</qty_available>  
<qty_buytotal>群发量购买总计</qty_buytotal>  
</service>
```

**说明:**

1. **"service\_status"**: 用户群发服务状态, "enabled"为允许用户使用群发服务; "disabled"为禁止用户使用群发服务。
2. **"service\_type"**: 群发服务类型, "all"为按群发总量进行发送, "day"为按每日发送量进行发送。
3. **"send\_type"**: 群发发送方式, "remote"为客户使用自己的服务器与平台对接发送, "local"为使用平台服务器来进行发送。

U-Mail 邮件群发

## 二、群发统计相关接口

### 2.1 取得指定的群发任务或者指定发送日期的群发任务列表（Web 群发专用，不适用 smtp）

接口功能：取得指定日期的群发任务列表

接口地址：

<http://www.bestedm.org/mm-ms/apinew/task.php?do=list-task&date=2012-03-02>

（备注：该接口只适用 api 和平台操作方式获取，不适用 smtp 对接方式）

GET 参数（必要参数）：

1. "date"：所要获取的群发任务的日期，可不传，不传则获取指定任务统计详情
2. 'ident'：具体的任务批次号（批次号通过 3.1 章节接口获取），可不传
3. 'id'：具体的任务 id，可不传
4. 'state'：任务发送状态，取值-5 到 4 之间，可不传

备注：date, ident, id 必须传一个

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<task_list>
  <task>
    <id>发送任务 send ID</id>
    <sn>发送任务批次号</sn>
    <sender>发件人地址</sender>
    <sender_name>发件人名称</sender_name>
    <replyto>回复地址</replyto>
    <tpl_id>使用的模板 ID</tpl_id>
    <tpl_name>使用的模板名称</tpl_name>
    <addr_type>发送地址类型</addr_type>
    <list_id>联系人分类 ID</list_id>
    <list_name>联系人分类名称</list_name>
    <send_qty>预定发送数量(用户指定)</send_qty>
    <send_qty_remak>预定发送数量(实际发送)</send_qty_remak>
    <send_time>预定发送时间</send_time>
    <send_satus>任务发送状态（-5 至 4 之间）</send_satus>
    <time_start>实际发送开始时间</time_start>
    <time_end>实际发送结束时间</time_end>
    <send_count>实际发送数量</send_count>
    <error_count>没有发送的失败数（不等于于发送失败数）</error_count>
```

```

    <track_status>跟踪统计状态</track_status>
  </task>
</task_list>

```

**说明:**

1. "addr\_type" 字段为"0"表示为使用用户联系人分类中的邮件地址来进行发送，为"1"表示使用邮件订阅列表中的邮件地址来进行发送。
2. "send\_qty" 字段为用户设置的发送数量，如为"0"则表示使用指定联系人分类的所有地址来发送；此时"send\_qty\_remark"则显示添加此任务时指定联系人分类中的实际地址数量。  
"send\_status" 字段各值含义如下：'0' 为等待发送  
'1' 为等待启动发送  
'2' 为正在发送  
'3' 为发送完成  
'4' 任务停止  
'-1' 为暂不发送  
'-2' 为暂停发送  
'-3' 为取消发送  
'-4' 为发送出错  
'-5' 为等待入库启动
3. "track\_status" 字段各值含义如下："0"：不踊跃；"1"：跟踪邮件打开情况；"2"：跟踪邮件打开与链接点击情况。

**2.2 获取指定日期的群发统计概览**

**接口功能:** 获取指定群发任务的跟踪统计概览

**接口地址:** <http://www.bestedm.org/mm-ms/apinew/statistic.php?date=2012-03-02>

**GET 参数(必要参数):**

1. "date": 所要获取的统计情况的日期

**输出格式:**

```

<?xml version="1.0" encoding="UTF-8"?>
<statistic>
  <date>日期</date>
  <ws_qty>web发送统计任务量</ws_qty>
  <ws_error>web发送统计失败量</ws_error>
  <ws_count>web发送统计发送量</ws_count>
  <invalid_addr>无效地址量</invalid_addr>
  <format_error>格式错误量</format_error>
  <send_count>预统计发送量</send_count>
  <deduc_count>预扣点</deduc_count>

```

```

<actual_send>实际发送量</actual_send>
<send_error>投递失败量</send_error>
<actual_deduc>实际扣点量</actual_deduc>
<blacklist_count>收件人黑名单数量</blacklist_count>
  <refuse_count>投诉地址量</refuse_count>
  <email_not_exist>不存在邮箱地址数量</email_not_exist>
  <jujue_count>拒绝投递量</jujue_count>

</statistic>

```

## 2.3 导出发送失败地址

### 2.3.1 只导出发送失败地址

**接口功能：**只导出发送失败地址

**接口地址：**<http://www.bestedm.org/mm-ms/apinew/failexport.php>

（备注：该接口 api 和平台操作、smtp 对接方式均适用，但是 smtp 对接方式无法准确获取指定批次无效地址，请悉知，若要获取无效地址，请务必不传 task\_id 和 task\_ident，按照日期获取，或者使用 2.2 接口获取）

**GET 参数：**

1. **"date"** (非必要参数, date, task\_id, task\_ident 必传其一即可)：所要导出失败地址批次任务的日期

例：2012-03-02，不要传入 2012-3-2，必须格式是 YYYY-MM-dd

2. **"type"** (必要参数)：导出错误地址类型

"format\_error"或 9：格式错误；  
 "invalid\_addr"或 8：无效地址；  
 "not\_exist"或 1：邮箱不存在；  
 "over\_quota"或 2：邮箱空间满；  
 "user\_reject"或 3：用户拒收  
 5：拒绝投递  
 6：地址是黑名单，收件人黑名单  
 7：投诉地址  
 4：垃圾内容邮件  
 10：连接异常  
 "all"或-1：全部错误失败无效拒收等地址

3. **"task\_id"** (非必要参数, date, task\_id, task\_ident 必传其一即可)：指定统计任务的 task\_id, 非发送任务的任务 id(send\_id), 而是根据 send\_id 调用【3.1 获取指定群发任务的跟踪统计概览】获取返回的 task\_id(请特别注意)



4. **“task\_id”** (非必要参数, date, task\_id, task\_id 必传其一即可): 指定的任务批次号, 就是获取该次任务批次相关的任务的错误地址记录。type 参数为 1, 2, 3 时, task\_id 或者 task\_id 其一是必要参数 (批次号通过 3.1 接口获取)
5. **“out\_type”**: out\_type 参数省略时, 直接输出文本文件, 每行一个邮件地址。可取 file, json, xml 其中一个。

**输出格式:** 默认直接输出文本文件, 每行一个邮件地址。当 out\_type 为空取 file, json, xml 分别表示导出 json, xml 格式数据。

file 导出 csv; xml 结果返回示例:

```
<?xml version="1.0" encoding="UTF-8"?>
<fail_recipient_list>
  <recipient>
    <email>violatte@coolreall.com</email>
  </recipient>
  ...
</fail_recipient_list>
```

Json 结果返回示例:

```
[
  "1595936793@qq.com",
  "1595936793@qq.com",
  ...
]
```

### 2.3.2 导出发送失败地址以及错误类型 (type) 以及任务 task\_id

**接口功能:** 导出发送失败地址以及错误类型以及 task\_id

**接口地址:** <http://www.bestedm.org/mm-ms/apinew/failexportall.php>

(备注: 该接口 api 和平台操作、smtp 对接方式均适用, 但是 smtp 对接方式无法准确获取指定批次无效地址, 请悉知, 若要获取无效地址, 请务必不传 task\_id 和 task\_id, 按照日期获取, 或者使用 2.2 接口获取)

**GET 参数:**

1. **“date”** (非必要参数, date, task\_id, task\_id 必传其一即可): 所要导出失败地址批次任务的日期  
例: 2012-03-02, 不要传入 2012-3-2, 必须格式是 YYYY-MM-dd
2. **“type”** (必要参数): 导出错误地址类型  
“format\_error”或 9: 格式错误;  
“invalid\_addr”或 8: 无效地址;  
“not\_exist”或 1: 邮箱不存在;  
“over\_quota”或 2: 邮箱空间满;  
“user\_reject”或 3: 用户拒收

- 5: 拒绝投递
- 6: 地址是黑名单, 收件人黑名单
- 7: 投诉地址
- 4: 垃圾内容邮件
- 10: 连接异常
- "all"或-1:全部错误失败无效拒收等地址

3. “task\_id” (非必要参数, date, task\_id, task\_id 必传其一即可): 指定统计任务的 task\_id, 非发送任务的 id(send\_id), 而是根据 send\_id 调用【3.1 获取指定群发任务的跟踪统计概览】获取返回的 task\_id(请特别注意)

4. “task\_id” (非必要参数, date, task\_id, task\_id 必传其一即可): 指定的任务批次号, 就是获取该次任务批次相关的任务的错误地址记录。type 参数为 1, 2, 3 时, task\_id 或者 task\_id 其一是必要参数(批次号通过 3.1 接口获取)

5. “out\_type”: out\_type 参数省略时, 直接输出文本文件, 每行一个邮件地址。可取 file, json, xml 其中一个。

**输出格式:** 默认直接输出文本文件, 每行一个邮件地址。当 out\_type 为空取 file, json, xml 分别表示导出 json, xml 格式数据。返回的 task\_id 指的是统计任务的 task\_id, 非发送任务的 id(send\_id), 而是根据 send\_id 调用【3.1 获取指定群发任务的跟踪统计概览】获取返回的 task\_id(请特别注意), 返回的 type 参数代表意义如下:

- 9: 格式错误;
- 8: 无效地址;
- 1: 邮箱不存在;
- 2: 邮箱空间满;
- 3: 用户拒收
- 5: 拒绝投递
- 6: 地址是黑名单, 收件人黑名单
- 7: 投诉地址
- 4: 垃圾内容邮件
- 10: 连接异常

file 导出 csv 格式:

| email             | type | task_id |
|-------------------|------|---------|
| 1595936793@qq.com | 3    | 1036770 |
| 1595936793@qq.com | 3    | 1036885 |
| 550317164@qq.com  | 8    | 1037645 |
| 3085392966@qq.com | 8    | 1037645 |

xml 结果返回示例:

```
<?xml version="1.0" encoding="UTF-8"?>
<fail_recipient_list>
  <recipient>
```

```

    <email>violatte@coolreall.com</email>
    <type>3</type>
    <task_id>1025694</task_id>
  </recipient>
</fail_recipient_list>

```

Json 格式结果样例:

```

[
  {
    "email": "1595936793@qq.com",
    "type": "3",
    "task_id": "1036770"
  },
  {
    "email": "1595936793@qq.com",
    "type": "3",
    "task_id": "1036885"
  },
  ...
]

```

## 2.4 指定日期邮件批次发送扣点返点统计信息

**接口功能:** 指定日期或者指定邮件批次发送统计表信息

**接口地址:**

```

http://www.bestedm.org/mm-ms/apinew/stask.php?date=2012-03-02&ident=20120302-362-F9B1E5

```

(备注: 该接口 api 和平台操作、smtp 对接方式均适用, 但是 smtp 对接方式无法准确获取指定批次无效地址, 请悉知, 若要获取无效地址数目, 请务必不传 **ident**, 按照日期获取, 或者使用 2.2 接口获取)

**GET 参数:**

1. "date": 所要获取的统计情况的日期
2. 'ident': 具体的任务批次号(批次号通过 3.1 章节接口获取)
3. id: 任务 id, 即 (6.2 章节接口) 创建任务时返回的 id

参数传值说明: 只传 date 表示获取改发送日期的任务列表统计信息; 如果传 ident 和 id 其中之一代表获取具体任务的统计信息

**输出格式:**

```

<?xml version="1.0" encoding="UTF-8"?>
<data>

```

```

<task>
  <task_id>任务id</task_id>
  <task_date>日期</task_date>
  <task_ident>任务批次</task_ident>
  <totalsum>预计总发送量（api对接和平台包含格式错误的、投诉的等，smtp对接则不包含包含格式错误的、投诉的等）</totalsum>
  <total>发送量</total>
  <invalid>无效地址和投诉、格式等未发送错误量（smtp方式传批次号时此数目会统计为0，smtp方式对接请2.2接口获取此数目）</invalid>
  <failed>总投递失败量包括发送失败和拒绝发送的数量</failed>
  <send_failed>总发送失败的数量</send_failed>
  <success>投递发送成功数</success>
  <email_not_exist>邮箱不存在数</email_not_exist>
  <spam_content>垃圾内容数</spam_content>
  <connect_error>连接错误数</connect_error>
  <actual>实际扣点数 </actual>
  <over_quota>空间不足</over_quota>
  <user_reject>用户拒收</user_reject>
  <rubbish>判为垃圾邮件拒绝发送的数量</rubbish>
  <point>任务当天扣点数</point>
  <rebate>返点数</rebtas>
</task>
...
</data>

```

## 2.5 指定日期扣点信息

**接口功能：**指定日期扣点信息

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/deduction.php?date=2012-03-02>

此接口只能统计日扣点数，需要统计某个任务实际扣点和返点请调用 2.4 接口

**GET 参数(必要参数)：**

1. "date"：所要获取的统计情况的日期

**输出格式：**

```

<?xml version="1.0" encoding="UTF-8"?>
<data>
  <list>
    <date>日期</date>

```

```
<company>公司名</company>
<deduction>扣点类型</deduction>
<send_num>发送数量</send_num>
<total>扣量总计</total>
</list>
...
</data>
```

## 2.6 发送日志信息查询

**接口功能：**指定日期的发送日志查询

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/maillog.php>

此接口只能统计固定日期的发送日志

**GET 参数 (必要参数)：**

1. "date"：所要获取的统计情况的日期
2. mail\_to: 邮件发送给某人的

输出格式：json ， 空日志格式返回：

```
{"lists": []}
```

## 三、跟踪统计相关接口（Web 群发专用）

### 3.1 获取指定群发任务的跟踪统计概览

**接口功能：**获取指定群发任务的跟踪统计概览（此接口可以通过任务 id 获取批次号 ident）

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/track.php?do=overview>

**GET 参数：**

1. “id”：统计任务 id，创建任务时返回的任务 id，也就是 send\_id
2. “ident”：任务批次号

参数 id, ident 三者必须有一个，任意传一个即可

**输出格式：**

```
<track>
  <track_id>跟踪统计 id</track_id> ----- 20121207 新增
  <task_id>统计任务 id</task_id>
  <task_sn>任务批次号（即 ident）</task_sn>
  <send_id>任务 id</task_id>
  <send_count>发送量</send_count>
  <real_send_count>成功发送量</real_send_count>
  <error_send_count>失败量</error_send_count>
  <open_unique>唯一打开数</open_unique>
  <open_total>总打开数</open_total>
  <open_first>首次打开时间</open_first>
  <open_last>最后打开时间</open_last>
  <click_unique>唯一点击数</click_unique>
  <click_total>总点击数</click_total>
  <click_first>首次点击时间</click_first>
  <click_last>最后点击时间</click_last>
  <open_ratio>打开率</open_ratio>
  <click_ratio>点击率</click_ratio>
  <link_statistic>
    <link>
      <link_id>链接 id</link_id>
      <link>链接</link>
      <click_unique>唯一点击数</click_unique>
      <click_total>总点击数</click_total>
      <click_first>首次点击时间</click_first>
```

```

        <click_last>最后点击时间</click_last>
    </link>
    <link>
    ...
    </link>
    ...
</link_statistic>
</track>

```

### 3.2 获取指定任务邮件打开统计详情

**接口功能：**获取指定群发任务邮件打开统计详情

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/track.php?do=open-detail>

**GET 参数：**

1. "id"：群发任务 send\_id，也就是创建任务成功后返回的 task\_id
2. "ident"：任务批次号（此批次号从接口 20180402101814-7020-53）

参数 id, ident 两者必须有一个，如果两个都存在，取 ident

**输出格式：**

```

<?xml version="1.0" encoding="UTF-8"?>
<data>
    <user>
        <email_id>邮箱 id</email_id>
        <email>邮箱</email>
        <browser>浏览器</browser>
        <os>操作系统</os>
        <addr_first>首次打开地址</addr_first>
        <ip_first>首次打开 ip</ip_first>
        <ip_last>最后打开 ip</ip_last>
        <addr_last>最后打开地址</addr_last>
        <open_total>总打开数</open_total>
        <click_total>总点击数</click_total>
        <open_first>首次打开时间</open_first>
        <open_last>最后打开时间</open_last>
        <click_first>首次点击时间</click_first>
        <click_last>最后点击时间</click_last>
    </user>
    ...
</data>

```

### 3.3 获取指定任务链接打开统计详情

**接口功能：**获取指定群发任务邮件打开统计详情

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/track.php?do=click-detail>

**GET 参数：**

1. “id”：群发任务 send\_id，也就是创建任务成功后返回的 task\_id
2. “ident”：任务批次号
3. “link\_id”：链接 id，指定任务指定链接点击统计详情

参数 id, ident 两者必须有一个，如果两个都存在，取 ident

**输出格式：**

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <click>
    <click_id>点击 id</click_id>
    <email>邮箱地址</email>
    <link>链接地址</link>
    <click_unique>唯一点击数</click_unique>
    <click_total>总点击数</click_total>
    <click_first>首次点击时间</click_first>
    <click_last>最后点击时间</click_last>
  </click>
  ...
</data>
```

### 3.4 导出指定群发任务跟踪邮箱信息

**接口功能：**导出指定群发任务跟踪统计邮箱信息

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/trackexport.php>

**GET 参数：**



1. `track_id` (必须): 群发任务跟踪 id, 多模板任务发送此 `track_id` 有多个, 可以传入多个, 以英文逗号隔开, 例如: 38282, 38283
2. `is_click`: 0 或 1, 0 为导出所有邮箱, 1 为只导出有点击的邮箱
3. `link_id`: 链接地址 id, 导出点击过该链接的邮箱
4. `email_id`: 邮箱 id, 导出指定邮箱

U-Mail 邮件群发

### 3.5 指定批次的邮件各链接的点击统计

接口功能：指定批次的邮件各链接的点击统计

接口地址：

<http://www.bestedm.org/mm-ms/apinew/track.php?do=link-stat>

GET 参数：

1. "id"：群发任务 send\_id;即创建任务时返回的 task\_id
  2. "ident"：任务批次号
  3. "link\_id"：链接 id, 指定任务, 指定链接点击统计
- 参数 id, ident 两者必须有一个, 如果两个都存在, 取 ident

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <link>
    <link_id>链接 id</link_id>
    <link>链接地址</link>
    <click_unique>唯一点击数</click_unique>
    <click_total>总点击数</click_total>
    <click_first>首次点击数</click_first>
    <click_last>最后点击数</click_last>
  </link>
  ...
</data>
```

## 四、联系人分类相关接口

### 4.1 取得联系人分类列表

接口功能：取得联系人分类列表

接口地址：

```
http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=maillist-list
```

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<maillist>
  <item>
    <id>联系人分类 ID</id>
    <subject>联系人分类名称</subject>
    <description>分类说明</description>
    <count>地址数量统计</count>
    <status>联系人分类状态</status>
  </item>
  ...
</maillist>
```

说明：

1. "status"：当前联系人分类的状态，"enabled"为正常使用，"disabled"为禁止使用。

### 4.2 取得联系人分类详情

接口功能：取得联系人分类详情

接口地址：

```
http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=maillist-detail
```

GET 参数(必要参数)：

1. "id"：联系人分类 ID

输出格式：

```
<?xml version="1.0" encoding="UTF-8"?>
<maillist>
```

```

<id>联系人分类 ID</id>
<subject>联系人分类名称</subject>
<description>联系人分类说明</description>
<count>地址数量统计</count>
<status>联系人分类状态</status>
</aillist>

```

说明：“status”字段含义同上。

### 4.3 添加联系人分类

接口功能：添加联系人分类

接口地址：

```

http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=maillist-add

```

POST 参数：

1. “subject”：联系人分类名称 **(必要参数)**
2. “status”：联系人分类状态，“enabled”为正常使用，“disabled”为禁止使用 **(必要参数)**
3. “description”：联系人分类说明

输出格式：

```

<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>执行结果标识符</status>
  <data>78</data>
</result>

```

执行结果标识符说明：

1. “add\_maillist\_error”：添加联系人分类时发生错误
2. “add\_describe\_error”：添加联系人分类描述信息时发生错误
3. “success”：添加联系人分类成功
4. “data”：添加联系人分类成功时 data 直接返回分类 id，失败时则返回失败描述

### 4.4 修改联系人分类

接口功能：修改联系人分类

接口地址:

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=maillist-edit>

GET 参数(必要参数):

1. "id": 联系人分类 ID

POST 参数:

1. "subject": 联系人分类主题
2. "status": 联系人分类状态, "enabled"为正常使用, "disabled"为禁止使用
3. "description": 联系人分类说明

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>执行结果标识符</status>
  <data>执行结果说明</data>
</result>
```

执行结果标识符说明:

1. "update\_maillist\_error": 修改联系人分类时发生错误
2. "update\_describe\_error": 修改联系人分类描述信息时发生错误
3. "success": 修改联系人分类成功

#### 4.5 删除联系人分类

接口功能: 删除联系人分类

接口地址:

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=maillist-del>

GET 参数(必要参数):

1. "id": 联系人分类 ID

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>执行结果标识符</status>
  <data>执行结果说明</data>
</result>
```

**执行结果标识符说明:**

1. "delete\_address\_error": 删除联系人分类出错
2. "delete\_describe\_error": 删除联系人分类描述信息时发生错误
3. "delete\_maillist\_error": 删除联系人分类出错
4. "success": 删除联系人分类成功

**4.6 导出退订地址**

**接口功能:** 导出退订地址

**接口地址:**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=unsubscribe-export>

**GET 参数 (必要参数):**

1. "id": 联系人分类 ID

**输出格式:** 直接输出文本文件, 每行格式为“退订时间\t 邮件地址” 邮件地址。

**4.7 分类地址列表**

**接口功能:** 获取联系人分类下的地址信息或者单个地址信息

**接口地址:**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-list>

**GET 参数说明: (红色必须)**

1. list\_id: 分类 id 必须以 get 方式传递, 如存在, 则查找该分类下的地址。否则查找全部。为 0 时查找未分类地址。
2. address\_id: 不传此参数, 代表获取当前分类, 当 address\_id>0, 表示只获取此单个地址的信息返回
3. keyword: 在 address, fullname 两项中模糊查找
4. page: 分页数, 如果存在该参数, 则为分页查找, 否则为查找全部
5. limit: 分页查找时每页显示的数据记录数目, 默认为 50

**输出格式:**

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <total_num>地址总数目</total_num >
  <addr>
    <id>地址 id</id>
```

```

<list_id>邮件订阅 id</list_id>
<address>![CDATA[邮箱地址]]</address>
<brithday>1986-08-15</brithday>
<phone>13528775736</phone>
<area>![CDATA[深圳市福田区科技园中路 3 号]]</area>
<fullname>![CDATA[名称]]</fullname >
<time>添加时间</time>
<var1>变量 1</var1>
<var2>变量 2</var2>
<var3>变量 3</var3>
<var4>变量 4</var4>
<var5>变量 5</var5>
<var6>变量 6</var6>
<var7>变量 7</var7>
<var8>变量 8</var8>
<var9>变量 9</var9>
<var10>变量 10</var10>
</addr>
<addr>
<id>地址 id</id>
<list_id>邮件订阅 id</list_id>
<address>![CDATA[邮箱地址]]</address>
<brithday>1986-08-15</brithday>
<phone>13528775736</phone>
<area>![CDATA[深圳市福田区科技园中路 3 号]]</area>
<fullname>![CDATA[名称]]</fullname >
<time>添加时间</time>
<var1>变量 1</var1>
<var2>变量 2</var2>
<var3>变量 3</var3>
<var4>变量 4</var4>
<var5>变量 5</var5>
<var6>变量 6</var6>
<var7>变量 7</var7>
<var8>变量 8</var8>
<var9>变量 9</var9>
<var10>变量 10</var10>
</addr>
...
</data>

```

#### 4.8 新增分类地址

## 4.8.1、添加一个

**接口功能：**增加联系人分类下的地址

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add>

**GET 参数说明：（红色必须）**

1. list\_id: 分类 id
2. address: 邮箱地址
3. fullname: 用户名
4. var1, var2, ..., Var10: 变量 1~10
5. return\_id: 成功返回信息包含新添加的地址 id, 1 为包含, 0 为不包含。默认为 0
6. sex 格式: M (M 男 F 女)
7. birthday 格式: 0000-00-00
8. phone 格式: 13528775748
9. area 格式: 北京市丰台
10. return\_id: 成功是否返回地址主键 id, 0 不返回, 1 返回

**说明：**除 username, password, do 三个参数外, 其他的参数还可以通过 post 发送

## 4.8.2、批量添加

**接口功能：**增加联系人分类下的地址

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add>

**GET 参数说明：（红色必须）**

1. list\_id: 分类 id
2. addr\_type: 地址信息数据类型, 默认为 'string'
3. separate: 每个地址信息之间的分隔符, 默认为 "\n" 换行符, 可以自定义任意字符。**特别注意只在 addr\_type 为 string 时生效**

**POST 参数说明：（红色必须）**

1. ml\_addr: 地址信息

**说明：**ml\_addr 为地址信息, 格式为:

文件格式说明

|                  |    |    |    |    |    |     |     |     |     |     |     |  |
|------------------|----|----|----|----|----|-----|-----|-----|-----|-----|-----|--|
| user1@domain.com | 姓名 | 性别 | 生日 | 手机 | 地区 | 变量1 | 变量2 |     |     |     |     |  |
| user2@domain.com | 姓名 | 性别 | 生日 | 手机 | 地区 | 变量1 | 变量2 | 变量3 | 变量4 | 变量5 | ... |  |
| user3@domain.com | 姓名 | 性别 | 生日 | 手机 | 地区 | 变量3 |     |     |     |     |     |  |



说明:

邮箱地址;名称;性别;生日;手机;地区;变量 1; 变量 2; 变量 3; 变量 4; 变量 5; 变量 6;  
变量 7; 变量 8;变量 9;变量 10 (换行)

格式 (一行一个地址信息):

```
Coming1@bestdem.org;coming1;M;1986-08-10;13528775968;北京;var1;var2;....var10;
test@bestdem.org; test;F;1986-08-10;13528775968;area;var1;var2;....var10;
```

行行之间用换行符\n 隔开,同一行不同变量值之间用分号;分割。适合批量添加,但数目不建议超过 500 以上,如果超过 500 以上;建议用【4.9.3】文件导入方式效率高。**特别注意**  
 \n 如果是代码拼接记得写成%0A,也就是 urlencode 后的值

#### 4.8.3、从文件导入

**接口功能:** 从文件中导入到联系人分类

**接口地址:**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add-file>

**GET 参数说明:** (红色必须)

1. list id: 分类 id, 也可通过 post 方式传送

**POST 参数说明:** (红色必须)

1. file: 文件信息

**示例:** 建立一个 html 表单, 上传文件

```
<form
method="post"
action="http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add-file&l
ist_id=418804"
enctype="multipart/form-data">
<input type="file" name="file" />
<input type="submit" value="submit" />
</form>
```

**说明:** 系统定时检测导入地址任务, 自动过滤无效地址和重复地址。

从文件导入 Java 调用代码示例 (验证用户名密码并请参考第九章 java 上传代码):

```
package cn.edu.ustc.file;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
```

```

import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.UUID;
public class UploadFileToPhp {

    private static final int TIME_OUT = 10 * 1000000; // 超时时间
    private static final String CHARSET = "utf-8"; // 设置编码
    public static boolean uploadFile(File file) {
        String BOUNDARY = UUID.randomUUID().toString(); // 边界标识 随机生成
        String PREFIX = "--", LINE_END = "\r\n";
        String CONTENT_TYPE = "multipart/form-data"; // 内容类型
        String
            RequestURL
            =
"http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add-file&list_id=
1";

        System.out.println(RequestURL);
        try {
            URL url = new URL(RequestURL);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setReadTimeout(TIME_OUT);
            conn.setConnectTimeout(TIME_OUT);
            conn.setDoInput(true); // 允许输入流
            conn.setDoOutput(true); // 允许输出流
            conn.setUseCaches(false); //不允许使用缓存
            conn.setRequestMethod("POST"); // 请求方式
            conn.setRequestProperty("Charset", CHARSET); // 设置编码
            conn.setRequestProperty("connection", "keep-alive");
            conn.setRequestProperty("Content-Type", CONTENT_TYPE
            +
";boundary="+BOUNDARY);
            if (file != null) {
                /**
                 * 当文件不为空，把文件包装并且上传
                 */
                OutputStream outputStream = conn.getOutputStream();

                DataOutputStream dos = new DataOutputStream(outputStream);
                StringBuffer sb = new StringBuffer();
                sb.append(PREFIX);
                sb.append(BOUNDARY);
                sb.append(LINE_END);
                /**
                 * 这里重点注意： name 里面的值为服务器端需要 key (php 里面需要是
                 file), 只有这个 key 才可以得到对应的文件

```

\* filename 是文件的名称，包含后缀名的 比如:abc.png, 可以看到下面  
 在开始与结束都写入了一些分隔符等标示。

```

*/
sb.append("Content-Disposition:    form-data;    name=\"file\"";
filename=\"\"
        + file.getName() + "\"" + LINE_END);
sb.append("Content-Type: application/octet-stream; charset="
        + CHARSET + LINE_END);
sb.append(LINE_END);
dos.write(sb.toString().getBytes());
InputStream is = new FileInputStream(file);
byte[] bytes = new byte[1024];
int len = 0;
while ((len = is.read(bytes)) != -1) {
    dos.write(bytes, 0, len);
}
is.close();
dos.write(LINE_END.getBytes());
byte[] end_data = (PREFIX + BOUNDARY + PREFIX + LINE_END)
    .getBytes();
dos.write(end_data);
dos.flush();
/**
 * 获取响应码 200=成功 当响应成功，获取响应的流
 */
int res = conn.getResponseCode();

System.out.println(res);
if (res == 200) {
    /**
     * 获取内容在此写代码
     */
    return true;
}
} catch (MalformedURLException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
return false;
}

```

```

public static void main(String[] args) {
    File file = new File("/dir/address.txt");
    boolean res = uploadFile(file);
    System.out.println(res);
}
}

```

address.txt 格式说明:

邮箱地址;名称;性别;生日;手机;地区;变量 1; 变量 2; 变量 3; 变量 4; 变量 5; 变量 6;  
变量 7; 变量 8;变量 9;变量 10 (换行)

格式:

Coming1@bestdem.org;coming1;M;1986-08-10;13528775968;北京;var1;var2;...var10;  
test@bestdem.org; test;F;1986-08-10;13528775968;area;var1;var2;...var10;  
一行一个地址信息;

每行各个参数之间用“;”隔开。适合批量添加

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<result>
```

```
<status>success</status>
```

```
<data><![CDATA[10]]></data>;//10 为查询是否解析成功的标识 id, 需要记录
```

```
</result>
```

Eg:

```

<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>success</status>
  <data>
    <![CDATA[210586]]>
  </data>
</result>

```

#### 4.8.4、从文件导入查询地址是否解析成功

**接口功能:** 文件导入到联系人分类后查询地址是否解析成功

**接口地址:**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add-file-query&id=1>

**GET 参数说明:** (红色必须)

1. id: 文件导入后返回的 id

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<result>
<status>success</status>
<data><![CDATA[解析完成]]></data>;
</result>
```

Status 返回值说明:

success/not\_exist/error/no\_complete 解析完成/不存在 id/参数错误/正在解析

## 4.9 修改分类地址

接口功能: 修改联系人分类下的地址

接口地址:

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-edit>

GET 参数说明: (红色必须)

1. id: 地址 id
2. address: 邮箱地址
3. fullname: 用户名
4. var1, var2...Var10: 变量 1~10
6. sex 格式: M (M男 F女)
7. birthday 格式: 1988-04-08
8. phone 格式: 13528775748
9. area 格式: 北京市丰台

说明: 除 username, password, do 三个参数外, 其他的参数还可以通过 post 发送

## 4.10 删除分类地址

接口功能: 删除联系人分类下的地址

接口地址:

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-del>

GET 参数说明: (红色必须)

1. id: 地址 id

## 4.11 清空联系人分类

接口功能: 清空联系人分类下的地址

接口地址:

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-empty>

GET 参数说明: (红色必须)

1. **list\_id**: 联系人分类 id

#### 4.12 邮件订阅

接口功能: 添加地址到订阅列表

接口地址:

[http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-sub&list\\_id=邮件订阅id&address=xxx@xxx.com&name=邮箱用户名称](http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-sub&list_id=邮件订阅id&address=xxx@xxx.com&name=邮箱用户名称)

GET 参数 (必要参数):

- 1, "**list\_id**": 邮件订阅 ID
- 2, "**address**": 订阅邮箱
- 3, "**name**": 邮箱用户名称
- 4, "**maillist**": 批量订阅邮箱地址

说明:

name 参数如果没有提供, 默认取邮箱名。如 test@bestedm.com, 名称就取 test  
address, name 两参数, 添加一个地址时使用

maillist 批量添加时使用, 可通过 get 或者 post 传递, 格式为:

格式 1: 邮箱 1, 名称 1; 邮箱 2, 名称 2; .....

test1@bestedm.com, test1; test2@bestedm.com, test2; ...

格式 2: 一行为一个地址信息, 邮箱与名称之间用 tab 键或者 “,” 分隔。

邮箱 1    名称

邮箱 2    名称

...

test1@bestedm.com    test1

test2@bestedm.com    test2

#### 4.13 邮件退订

接口功能: 删除邮件订阅中的地址

接口地址:

[http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-unsub&list\\_id=邮件订阅id&address=xxx@xxx.com](http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-unsub&list_id=邮件订阅id&address=xxx@xxx.com)

GET 参数 (必要参数):

- 1, "list\_id": 邮件订阅 ID
- 2, "address": 退订邮箱

**说明:**

address 可通过 get 或者 post 传递, 格式为:

格式 1: 邮箱 1, 邮箱 2, .....

test1@bestedm.com, test2@bestedm.com, ...

格式 2: 一行为一个地址信息

邮箱 1

邮箱 2

...

test1@bestedm.com

test2@bestedm.com

#### 4.14 导入分类地址记录查询

**接口功能:** 导入分类地址记录查询

**接口地址:**

<http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-add-log>

**说明:**

**返回数据说明**

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>success</status>
  <data>
    <address_log>
      <filename>emaillist2017.csv</filename> //文件名称
      <filepath>/usr/local/umail/data/web/mm-ms/cache/1493962939_71607.csv</filepath>
      //文件服务器路径
      <time_import>2017-05-05 13:42:20</time_import>//上传时间
      <count_all>0</count_all>//导入地址总量
      <count_err_1>0</count_err_1>//无效地址数
      <count_err_2>0</count_err_2>//重复的地址数
      <status>-2</status>//导入的状态 1 成功、-1/-2/-3 失败
      <subject>ddd</subject>//导入列表
    </address_log>
    .....
  </data>
</result>
```

## 五、邮件模板相关接口

### 5.1 创建群发任务

接口功能：创建群发任务

接口地址：

[http://www.bestedm.org/mm-ms/apinew/task.php?do=add-task&tpl\\_id=16&send\\_domain=comingchina.com&send\\_account=linbihuan@comingchina.com&send\\_fullname=linbihuan&maillist\\_id=5&time=2012-07-31\\_12:00](http://www.bestedm.org/mm-ms/apinew/task.php?do=add-task&tpl_id=16&send_domain=comingchina.com&send_account=linbihuan@comingchina.com&send_fullname=linbihuan&maillist_id=5&time=2012-07-31_12:00)

GET 参数说明：（红色必须）

1. **tpl\_id**: 模板 id 单个模板请传入 2, 该任务有多个模板则 1, 2, 3, 78 逗号隔开格式（可以填写自己和子帐号共享模板 id）
2. **send\_domain**: 发件人域名
3. **send\_account**: 发件人
4. **send\_fullname**: 发件人名称
5. **maillist\_id**: 联系人分类 id。
6. **send\_qty**: 联系人发送地址数量范围最大值，默认为 0，表示全部地址，结合 **send\_qty\_start** 代表发送地址范围 2-30, 50-100, 如: **send\_qty\_start=10, send\_qty=100** 表示到从第 10-100 个地址开始发送。
7. **send\_qty\_start**: 联系人发送地址范围开始最小值，为 1，表示从第一个地址开始, 注意必须大于 1，此值不能传 0
8. **status**: 发送状态，默认为 1，等待启动发送；-1 为暂不发送；-2 暂停发送，-3 取消发送，2 为立即发送，3 发送完成 ‘0’ 为等待发送，
  - ‘1’ 为等待启动发送
  - ‘2’ 为正在发送
  - ‘3’ 为发送完成
  - ‘4’ 任务停止
  - ‘-1’ 为暂不发送
  - ‘-2’ 为暂停发送
  - ‘-3’ 为取消发送
  - ‘-4’ 为发送出错
  - ‘-5’ 为等待入库启动
9. **time**: 发送时间，必须。格式 2012-07-04%2016:08, 即 2012-07-04 16:08 经过 urlencode 转义后的值，否则空格无法识别。
10. **subs\_link**: 是否在邮件中加入退订订阅链接。默认为 1，加入
11. **track\_status**: 是否开启邮件跟踪。默认为 0，不开启，默认为 0，不开启 1 只开



启打开统计 2 开启打开统计和点击链接跟踪统计

- 12、 `track_domain`: 自定义跟踪统计链接域名
13. `send_replyto`: 指定发送回复邮箱地址。例如: `reply@126.com`
14. `hour_speed`: 指定发送速率; 取值 100 至 75000 (单位: 每小时最大发送量设置)
15. `is_need_receipt`: 阅读回执; 1 则可以阅读回执, 0 则不行. 默认不传则为 0, 注意此参数依赖于是否开通阅读回执服务, 没开通则任务无法修改此参数, 即无法进行阅读回执。

#### 备注:

1. 除参数 `username`, `password`, `do` 之外, 其它参数均可通过 `get` 或者 `post` 方式发送
2. 如果发件人是某个域名下的所有发件人, 参数 `send_domain` 为域名, `send_account` 为 `all`。如: `&send_domain=comingchina.com&send_account=all`
3. 如果发件人是所有域名下的所有发件人, 参数 `send_domain`, `send_account` 两个都为 `all`。  
如: `&send_domain=all&send_account=all`
4. 如果发件人只是某个域名下的一个发件人。 `Send_domain` 可以不需要。如: [&send\\_account=test@comingchina.com](#)

备注: 调用此接口 1 小时调用次数不能超过 100 次

#### 输出格式:

成功:

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>success</status>
  <id>新创建的任务 id</id>
  <data><![CDATA[添加群发任务成功!]]></data>
</result>
```

错误:

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
  <status>错误状态</status>
  <data><![CDATA[错误描述]]></data>
</result>
```

## 5.2 上传邮件模板

接口功能: 上传邮件模板 (可上传附件)

接口地址:

<http://www.bestedm.org/mm-ms/apinew/template.php?do=add-tpl>

GET 参数说明:

1. do: 操作类型, 必须

POST 参数说明:

1. name: 邮件名称, 必须

2. subject: 邮件主题, 必须, 多个模板主题请以|隔开; 如: 主题一|主题二

3. content: 邮件内容, 必须

4. text\_content: 纯文本内容,html 邮件内容无法显示时, 显示纯文本内容

5. encoding: 邮件发送编码, base64 或者 quoted-printable

6. file: 附件参数。Input 标签中 name 属性的值为 'file', 如果是上传多个文件, 那么 name 属性值为 'file[]'。

说明: 成功返回新建模板的 id 值

Php 上传附件演示代码:

```
<?php
```

```
/**
 * Created by api 接口对接群发新增带附件模板附件 php 演示代码.
 * User: yzg
 * Date: 2018/4/18
 * Time: 16:55
 */
function sendCurlPost ()
{
    $vars = array ();
    $vars['list_id'] = 1176005;//联系人分类 id
    $vars['name'] = 'cesshi';//模板名称
    $vars['subject'] = 'subject';//模板主题
    $vars['content'] = 'test';//模板内容
    $vars['file'] = '@' . dirname(__FILE__) . '\121.jpg';//要上传的附件

    /* 初始化并执行 curl 请求 */
    $ch = curl_init();
    $opts = array(
        CURLOPT_TIMEOUT => 50,
        CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,
        CURLOPT_USERPWD=>'你的用户名:你的密码',
        CURLOPT_RETURNTRANSFER => 1,
        CURLOPT_BINARYTRANSFER=>1,
        CURLOPT_URL =>
```

```

'http://www.bestedm.org/mm-ms/apinew/template.php?do=add-tpl',
    CURLOPT_POST          => 1,
    CURLOPT_POSTFIELDS    => $vars,
    CURLOPT_USERAGENT    => $_SERVER['HTTP_USER_AGENT'],
    CURLOPT_SSL_VERIFYPEER => FALSE,
    CURLOPT_SSL_VERIFYHOST => FALSE,
    CURLOPT_HEADER=>0
);
curl_setopt_array($ch, $opts);
$status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE); //get
status code ;
$data = curl_exec($ch);
curl_close($ch);
return $data;
}
var_dump(sendCurlPost());

```

C#示例代码:

```

string url = "http://www.bestedm.org/mm-ms/apinew/template.php?do=add-tpl";
string username = "mxxxxxx";
string password = "sxxxxxx";
string sub = "中国（上海）国际婚纱摄影器材展览会周边报价单<盟轩：客服-admin>";
string name = "admin中国（上海）国际婚纱摄影器材展览会模版";
FileStream fs = new FileStream(@"c:\test\1.html", FileMode.OpenOrCreate,
FileAccess.Read);
StreamReader sr = new StreamReader(fs, Encoding.Default);
string mcontent = sr.ReadToEnd();
sr.Close();
fs.Close();

string postData = string.Format("&name={0}&subject={1}&content={2}", name, sub,
mcontent);

byte[] data = Encoding.UTF8.GetBytes(postData);
Encoding encoding = Encoding.Default;
HttpWebRequest myReq = (HttpWebRequest)WebRequest.Create(url);
myReq.Method = "POST";
myReq.ContentType = "application/x-www-form-urlencoded; charset=UTF-8";
//注意这里的格式哦，为 "username:password"

```

```

string username_Password = username + ":" + password;
CredentialCache mycache = new CredentialCache();
mycache.Add(new Uri(url), "Basic", new NetworkCredential(username,
password));
myReq.Credentials = mycache;
myReq.Headers.Add("Authorization", "Basic " + Convert.ToBase64String(new
ASCIIEncoding().GetBytes(username_Password)));
myReq.ContentLength = data.Length;
Stream newStream = myReq.GetRequestStream();
// Send the data.
newStream.Write(data, 0, data.Length);
newStream.Close();
WebResponse wr = myReq.GetResponse();
Stream receiveStream = wr.GetResponseStream();
StreamReader reader = new StreamReader(receiveStream, Encoding.UTF8);
string content = reader.ReadToEnd();

```

### 5.3 获取邮件模板检测状态和模板信息

接口功能：查看模板是否已经可以使用以及模板大小，检测状态

接口地址：

[http://www.bestedm.org/mm-ms/apinew/template.php?do=get\\_tpl\\_status](http://www.bestedm.org/mm-ms/apinew/template.php?do=get_tpl_status)

GET 参数说明：

1、do：操作类型，必须

GET 参数说明：

2、tpl\_id：模板 id，必须

输出格式：

```

<?xml version="1.0" encoding="UTF-8"?>
<template>
  <id>53</id>
  <name>模板名称</name>
  <status>1</status>
  <colour>green</colour><update_date>2017-11-11 11:11:11</update_date>
  <size_status>1</size_status>
  <size>78</size>
  <subject>模板主题 </subject>
</template>

```

说明: size 的单位大小为 kb; size\_status=-1 表示模板大小正在统计中, size\_status=1 表示模板大小已经统计完毕; size 为真实模板邮件大小。status 为-1 代表模板正在检测中, 0 标识检测为红色模板不予通过(red), 1 表示检测通过为绿色或者黄色或者红绿色(yellow, green, red\_pass); update\_date 模板更新时间, name 代表模板名称; colour 代表模板检测后的标识颜色, 正在检测中则为空。

## 5.4 模板列表

接口功能: 获取用户模板列表

接口地址:

[http://www.bestedm.org/mm-ms/apinew/template.php?do=list\\_tpl&tpl\\_id=67](http://www.bestedm.org/mm-ms/apinew/template.php?do=list_tpl&tpl_id=67)

GET 参数说明: (红色必须)

1. 'tpl\_id': 模板 id, 如果存在, 则只获取该 id 值的模板信息。否则获取全部模板信息

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<template_list>
  <template>
    <id>53</id>
    <user_id>模板 id</user_id>
    <name>
      <![CDATA[模板名称]]>
    </name>
    <subject>
      <id>模板主题一 id</id>
      <name><![CDATA[模板主题 1]]></name>
    </subject>
    <subject>
      <id>模板主题二 id</id>
      <name><![CDATA[模板主题 2]]></name>
    </subject>
    <content>
      <![CDATA[模板内容]]>
    </content>
    <content_type>模板类型</content_type>
  </template>
  <attachments> ----- 20121206 新增附件信息
```

```

    <attachment>
      <atta_id>附件 id</atta_id>
      <atta_name>附件文件名</atta_name>
    </attachment>
  </attachments>

  <created>创建时间</created>
  <updated>更新时间</updated>
</template>

<template>
  ...
</template>

...
</template_list>

```

## 5.5 删除模板

接口功能：删除指定 id 模板

接口地址：

[http://www.bestedm.org/mm-ms/apinew/template.php?do=del-tpl&tpl\\_id=67](http://www.bestedm.org/mm-ms/apinew/template.php?do=del-tpl&tpl_id=67)

GET 参数说明：

1. ‘**tpl\_id**’：模板 id

## 5.6 修改模板

接口功能：修改指定模板

接口地址：

[http://www.bestedm.org/mm-ms/apinew/template.php?do=edit-tpl&tpl\\_id=11&name=linbihuan&subject=sub&content=con](http://www.bestedm.org/mm-ms/apinew/template.php?do=edit-tpl&tpl_id=11&name=linbihuan&subject=sub&content=con)

GET 参数说明：

1. ‘**tpl\_id**’：模板 id，也可通过 post 上传，必须
2. ‘**name**’：模板名称，也可通过 post 上传
3. ‘**subject**’：模板主题，也可通过 post 上传，多个模板主题请以|隔开；如主题一|主题二

4. 'content': 模板内容, 也可通过 post 上传
5. text\_content: 纯文本内容, html 邮件内容无法显示时, 显示纯文本内容
6. encoding: 邮件发送编码, base64 或者 quoted-printable

#### 备注:

1. tpl\_id 参数必须, name、subject、content 三个参数必须至少有一个。
2. 四个参数均可以通过 post 或者 get 方式上传

## 5.7 上传模板附件

接口功能: 上传模板附件

接口地址:

[http://www.bestedm.org/mm-ms/apinew/template.php?do=tpl-attachment-add&tpl\\_id=67](http://www.bestedm.org/mm-ms/apinew/template.php?do=tpl-attachment-add&tpl_id=67)

GET 参数说明:

1. 'tpl\_id': 模板 id, 也可通过 post 上传, 必须

POST 参数说明:

1. 'file': 附件参数名。Input 标签中 name 属性的值为 'file', 如果是多附件, 该属性值为 'file[]'

Php 上传演示代码

```
/**
 * Created by api 接口对接群发上传制定模板附件 php 演示代码.
 * User: yzg
 * Date: 2018/4/18
 * Time: 16:55
 */
function sendCurlPost ()
{
    $vars = array ();
    $vars['file'] = '@' . dirname(__FILE__) . '\122.jpg'; //要上传的附件

    /* 初始化并执行 curl 请求 */
    $ch = curl_init();
    $opts = array (
        CURLOPT_TIMEOUT => 300,
        CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,
```

```

CURLOPT_USERPWD=>'你用户名:你的密码',
CURLOPT_RETURNTRANSFER => 1,
CURLOPT_BINARYTRANSFER=>1,
CURLOPT_URL
=>
'http://www.bestedm.org/mm-ms/apinew/template.php?do=tpl-attachme
nt-add&tpl_id=255288',
CURLOPT_POST
=> 1,
CURLOPT_POSTFIELDS
=> $vars,
CURLOPT_USERAGENT
=> $_SERVER['HTTP_USER_AGENT'],
CURLOPT_SSL_VERIFYPEER
=> FALSE,
CURLOPT_SSL_VERIFYHOST
=> FALSE,
CURLOPT_HEADER=>0
);
curl_setopt_array($ch, $opts);
$status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE); //get
status code ;
$data = curl_exec($ch);
curl_close($ch);
return $data;
}
var_dump(sendCurlPost());

```

## 5.8 删除模板附件

接口功能：删除模板附件

接口地址：

[http://www.bestedm.org/mm-ms/apinew/template.php?do=tpl-attachment-del&tpl\\_id=67&attachment\\_id=1](http://www.bestedm.org/mm-ms/apinew/template.php?do=tpl-attachment-del&tpl_id=67&attachment_id=1)

GET 参数说明：

1. 'tpl\_id': 模板 id, 也可通过 post 上传, 必须
2. 'attachment\_id': 附件 id, 默认为 'all', 删除模板中所有附件

## 六、群发任务相关接口

### 6.1 群发任务列表



**接口功能：**获取用户群发任务列表

**接口地址：**

<http://www.bestedm.org/mm-ms/apinew/task.php?do=task-list>

**GET 参数说明：（红色必须）**

1. ‘do’：操作类型
2. ‘id’：任务 id，如存在该参数，则只返回该任务的信息，否则返回所有任务信息
3. ‘page’：页数，从 1 开始
4. ‘limit’：每页显示的任务数，如果没有指定，默认为 30
5. ‘date’：发送日期，如 ‘2013-01-05’
6. ‘send\_status’：任务发送状态。  
默认为 ‘all’，表示所有群发任务。
  - ‘0’ 为等待发送
  - ‘1’ 为等待启动发送
  - ‘2’ 为正在发送
  - ‘3’ 为发送完成
  - ‘4’ 任务停止
  - ‘-1’ 为暂不发送
  - ‘-2’ 为暂停发送
  - ‘-3’ 为取消发送
  - ‘-4’ 为发送出错
  - ‘-5’ 为等待入库启动

**备注：**page, limit 两个参数如果都没有指定，默认获取全部任务列表。

**输出格式：**

```
<?xml version="1.0" encoding="UTF-8"?>
<task_list>
  <task>
    <id>任务 id</id>
    <sn>任务号</sn>
    <acct_type>发件人类型 (all, domain, address) </acct_type>
    <acct_domain>发件人域名, 发件人类型为 domain 时存在</acct_domain>
    <acct_address>发件人, 发件人类型为 address 时存在</acct_address>
    <sender>发件人</sender>
    <replyto>回复地址</replyto>
    <sender_name>发件人姓名</sender_name>
    <templates>
      <template>
        <tpl_id>模板 id</tpl_id>
```

```

        <tpl_name>模板名称</tpl_name>
    </template>
    <template>
        <tpl_id>模板 id</tpl_id>
        <tpl_name>模板名称</tpl_name>
    </template>
</templates>
<addr_type>地址类型</addr_type>
<listes>
    <item>
        <list_id>联系人分类 id</list_id>
        <list_name>联系人分类名称</list_name>
    </item>
</listes>
<send_qty>预定发送数量(用户指定)</send_qty>
<send_qty_remak>预定发送数量(实际发送)</send_qty_remak>
<send_time>预定发送时间</send_time>
<time_start>实际发送开始时间</time_start>
<time_end>实际发送结束时间</time_end>
<send_count>实际发送数量</send_count>
<error_count>发送失败数量</error_count>
<send_status>发送状态</send_status>
<track_status>
    跟踪统计状态数字
    (0 不跟踪 1 跟踪邮件打开情况 2 跟踪邮件打开与链接点击情况)
</track_status>
</task>
<task>...</task>
...
</task_list>

```

## 6.2 创建群发任务

接口功能：创建群发任务

接口地址：

[http://www.bestedm.org/mm-ms/apinew/task.php?do=add-task&tpl\\_id=16&send\\_domain=comingchina.com&send\\_account=linbihuan@comingchina.com&send\\_fullname=linbihuan&maillist\\_id=5&time=2017-11-24+09%3A54%3A01](http://www.bestedm.org/mm-ms/apinew/task.php?do=add-task&tpl_id=16&send_domain=comingchina.com&send_account=linbihuan@comingchina.com&send_fullname=linbihuan&maillist_id=5&time=2017-11-24+09%3A54%3A01)

GET 参数说明：（红色必须）

1. tpl\_id: 模板 id 单个模板请传入 2, 该任务有多个模板则 1, 2, 3, 78 逗号隔开格式（可以填写自己和子帐号共享模板 id）

2. **send\_domain**: 发件人域名
3. **send\_account**: 发件人
4. **send\_fullname**: 发件人名称
5. **maillist\_id**: 联系人分类 id
6. **send\_qty\_start**: 联系人发送地址范围开始最小值, 为 1, 表示从第一个地址开始, 此值不能传 0。
7. **send\_qty**: 联系人发送地址数量范围最大值, 默认为 0, 表示全部地址, 结合 **send\_qty\_start** 代表发送地址范围 2-30, 50-100, 如: **send\_qty\_start=10, send\_qty=100** 表示到从第 10-100 个地址开始发送。
8. **status**: 发送状态, 默认为 2, 1 为等待启动发送; **2 为立即发送, -2 暂停发送, -3 取消发送, 3 发送完成**; -1 为暂不发送; -2 暂停发送, -3 取消发送, 2 为立即发送, 3 发送完成  
'0' 为等待发送,
  - '1' 为等待启动发送
  - '2' 为正在发送
  - '3' 为发送完成
  - '4' 任务停止
  - '-1' 为暂不发送
  - '-2' 为暂停发送
  - '-3' 为取消发送
  - '-4' 为发送出错
  - '-5' 为等待入库启动
9. **time**: 发送时间, 必须。格式 2017-7-4 16:08:56 或者 2017-7-4 16:08 或者 2017-7-4 请 **urlencode** 传入。当 **status=2** 时 **time** 大于等于当前时间即可立即发送。
10. **subs\_link**: 是否在邮件中加入退订订阅链接。默认为 1, 加入
11. **track\_status**: 是否开启邮件跟踪。默认为 0, 不开启 1 只开启打开统计 2 开启打开统计和点击链接跟踪统计
12. **track\_domain**: 自定义跟踪统计链接域名
13. **hour\_speed**: 指定发送速率; 取值 100 至 75000 (单位: 每小时最大发送量设置)
14. **track\_domain**: 自定义跟踪域名
15. **send\_replyto**: 指定回复地址
16. **is\_need\_receipt**: 阅读回执; 1 则可以阅读回执, 0 则不行. 默认不传则为 0, 注意此参数依赖于是否开通阅读回执服务, 没开通则任务无法修改此参数, 即无法进行阅读回执。

**备注:**

1. 除参数 **username**, **password**, **do** 之外, 其它参数均可通过 **get** 或者 **post** 方式发送
2. 如果发件人是某个域名下的所有发件人, 参数 **send\_domain** 为域名, **send\_account** 为 **all**。如: **&send\_domain=comingchina.com&send\_account=all**
3. 如果发件人是所有域名下的所有发件人, 参数 **send\_domain**, **send\_account** 两个都为 **all**。  
如: **&send\_domain=all&send\_account=all**
4. 如果发件人只是某个域名下的一个发件人。Send\_domain 可以不需要。如: **&send\_account=test@comingchina.com**

### 6.3 修改群发任务

**接口功能：**修改群发任务

**接口地址：**

[http://www.bestedm.org/mm-ms/apinew/task.php?do=edit-task&id=16&send\\_domain=comingchina.com&send\\_account=linbihuan@comingchina.com&send\\_fullname=linbihuan&maillist\\_id=5&time=2012-07-31%2012:00](http://www.bestedm.org/mm-ms/apinew/task.php?do=edit-task&id=16&send_domain=comingchina.com&send_account=linbihuan@comingchina.com&send_fullname=linbihuan&maillist_id=5&time=2012-07-31%2012:00)

**GET 参数说明：**（红色必须）

1. id: 任务 id
2. tpl\_id: 模板 id 单个模板请传入 2, 该任务有多个模板则 1, 2, 3, 78 逗号隔开格式（可以填写自己和子帐号共享模板 id）
3. send\_domain: 发件人域名，默认填 all
4. send\_account: 发件人，默认填 all
5. send\_fullname: 发件人名称
6. maillist\_id: 联系人分类 id
7. send\_qty\_start: 联系人发送地址范围开始最小值，为 1，表示从第一个地址开始，此值不能传 0。
8. send\_qty: 联系人发送地址数量范围最大值，默认为 0，表示全部地址，结合 send\_qty\_start 代表发送地址范围 2-30, 50-100, 如: **send\_qty\_start=10, send\_qty=100** 表示到从第 10-100 个地址开始发送。
9. send\_status: 发送状态，默认为 1 等待启动发送。
  - '0' 为等待发送，
  - '1' 为等待启动发送
  - '2' 为正在发送
  - '3' 为发送完成
  - '4' 任务停止
  - '-1' 为暂不发送
  - '-2' 为暂停发送
  - '-3' 为取消发送
  - '-4' 为发送出错
  - '-5' 为等待入库启动
10. send\_time: 发送时间，格式 2017-7-4 16:08:56 或者 2017-7-4 16:08 或者 2017-7-4 请 urlencode 传入。
11. subs\_link: 是否在邮件中加入退订订阅链接 (0, 1)。
12. track\_status: 是否开启邮件跟踪 (0, 1, 2)。0 是否开启邮件跟踪。默认为 0，不开启 1 只开启打开统计 2 开启打开统计和点击链接跟踪统计
13. track\_domain: 自定义跟踪域名
14. send\_replyto: 指定回复地址
15. hour\_speed: 指定发送速率；取值 100 至 75000（单位：每小时最大发送量设置）

16. `is_need_receipt`: 阅读回执; 1 则可以阅读回执, 0 则不行. 默认不传则为 0, 注意此参数依赖于是否开通阅读回执服务, 没开通则任务无法修改此参数, 即无法进行阅读回执。

#### 备注:

1. 除参数 `username`, `password`, `do` 之外, 其它参数均可通过 `get` 或者 `post` 方式发送
2. 如果发件人是某个域名下的所有发件人, 参数 `send_domain` 为域名, `send_account` 为 `all`。如: `&send_domain=comingchina.com&send_account=all`
3. 如果发件人是所有域名下的所有发件人, 参数 `send_domain`, `send_account` 两个都为 `all`。  
如: `&send_domain=all&send_account=all`
4. 如果发件人只是某个域名下的一个发件人。Send\_domain 可以不需要。如: `&send_account=test@comingchina.com`

## 6.4 删除群发任务

接口功能: 删除群发任务

接口地址:

<http://www.bestedm.org/mm-ms/apinew/task.php?do=del-task&id=19124>

GET 参数说明: (红色必须)

1. `id`: 任务 id

## 七、域名及域名下的邮箱账号

### 7.1 域名列表

接口功能: 获取域名列表

接口地址:

<http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-list>

GET 参数说明: (红色必须)

1. `customer_id`: 客户 id
2. `user_type`: 用户类型。manager 或 customer, 默认为 customer。如果是 manager, `username` 跟 `password` 两个参数必须是管理员的用户名跟密码, 需要加 `customer_id`、`user_type` 两参数

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<domain_list>
  <data>
    <customer_id>客户 id (0 为系统域名) </customer_id>
    <domain>域名</domain>
  </data>
  <data>...</data>
  ...
</domain_list>
```

## 7.2 添加域名

接口功能: 添加域名

接口地址:

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=customer-domain-add&domain=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=customer-domain-add&domain=***)

返回:

添加成功并域名配置成功返回结果

```
<result>
<status>success_and_check_ok</status>
<data>23</data>
</result>
```

添加成功但域名配置没有成功返回结果

```
<result>
<status>success</status>
<data>23</data>
</result>
```

Data 为域名主键 id

## 7.3 验证域名

接口功能: 验证域名 DNS 是否已经配置正确

接口地址:

<http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-check>

POST 参数说明:

‘domains’: 要检测域名列表。多个域名用逗号隔开格式 test.com, we.com, me.com

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<domain_list>
  <domain>
    <customer_id>2369</customer_id>
    <domain_id>3213</domain_id>
    <val>test.com</val>
    <status>3</status>
    <info>添加成功并验证通过</info>
  </domain>
  <domain>
    <customer_id>2369</customer_id>
    <domain_id>6731</domain_id>
    <val>we.com</val>
    <status>2</status>
    <info>添加成功但验证不通过</info>
  </domain>
  <domain>
    <customer_id>2369</customer_id>
    <domain_id>0</domain_id>
    <val>me.com</val>
    <status>0</status>
    <info>系统不存在并验证不通过</info>
  </domain>
  .....
</domain_list>
```

说明:

**domain\_id>0** 表示系统已经存在此域名, 否则不存在; **status=3** 表示添加成功并验证通过, **status=2** 表示添加成功但验证不通过; **status=1** 表示系统不存在但验证通过; **status=0** 表示系统中不存在并验证不通过。

如果添加域名想验证通过; 需要 DNS 服务提供商处添加相关配置记录方可验证通过。具体配置如下: 比如您的添加域名为 magvision.com

## 发信配置

请至域名 ceshi.magvision.com DNS服务提供商处添加TXT记录。

## \* 1、SPF验证

| 类型  | 主机记录  | 主域名           | 记录值                                 | 状态   |
|-----|-------|---------------|-------------------------------------|------|
| TXT | ceshi | magvision.com | v=spf1 include:spf.bestedm.org -all | 验证通过 |

说明：如果您已经添加了SPF验证，则请在SPF的TXT记录中加上 include:spf.bestedm.org

## 收信配置

请至域名 ceshi.magvision.com DNS服务提供商处添加TXT记录。

## \* 2、MX验证

| 类型 | 主机记录  | 主域名           | 记录值              | 状态   |
|----|-------|---------------|------------------|------|
| MX | ceshi | magvision.com | mail.bestedm.org | 验证通过 |

## 3、DKIM验证

验证通过

在DNS服务器上给 umail.\_domainkey.ceshi.magvision.com 添加TXT记录

```
k=rsa;
p=MIGfMA0GCsGqGSIsb3DQEBAQUAA4GNADCBiQKBgQC31fDVvUxmPwbdRUW4jzT0qs0pQn2KDcRNp70rhxAAASxySDxrbDQglyRiP0spPc0
TmkhNkqQRHjOO
/sd2X2Tbn6ulrXOG6JfInbQnEYxBwC6+yNYG+7oTRf9M2psUAwgBCH6r73GNiwccxhvo87Zv90CBDi1T0CyEHm7bMzNmwlDAQAB
```

注：域名的SPF，MX记录必须配置正确才能验证通过

## 7.4 为指定子账号用户添加域名(管理员账号才可调用)

接口功能：为指定子账号用户添加域名

接口地址：

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-add&customer\\_id=\\*&domain=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-add&customer_id=*&domain=***)

GET 参数说明：(红色必须)

1. customer\_id: 指定客户的 id 如 2
2. domain: 需要添加的域名，如 test.com

## 7.5 删除指定子帐号的域名(管理员账号才可调用)

接口功能：删除域名

接口地址：

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-del&customer\\_id=4&domain=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=domain-del&customer_id=4&domain=***)

GET 参数说明：(红色必须)

1. customer\_id: 客户 id



2. domain: 需要添加的域名, 如 test.com

## 7.6 删除域名

接口功能: 删除域名

接口地址:

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=customer-domain-del&domain=\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=customer-domain-del&domain=*)

说明: 已经验证通过的域名不能删除。

## 7.7 指定域名下的账号列表

接口功能: 获取指定域名下的账号列表

接口地址:

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-list&domain=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-list&domain=***)

GET 参数说明: (红色必须)

1. customer\_id: 客户 id
2. domain: 需要添加的域名, 如 test.com
3. user\_type: 用户类型。manager 或 customer, 默认为 customer。如果是 manager, username 跟 password 两个参数必须是管理员的用户名跟密码, 需要加 customer\_id、user\_type 两参数
4. page: 分页数, 如果存在该参数, 则为分页查找, 否则为查找全部
5. limit: 分页查找时每页显示的数据记录数目, 默认为 30

输出格式:

```
<?xml version="1.0" encoding="UTF-8"?>
<mailbox_list>
  <data>
    <customer_id>客户 id</customer_id>
    <domain>域名</domain>
    <name>账号</name>
    <mailbox>完整账号</mailbox>
  </data>
  <data>...</data>
  ...
</mailbox_list>
```

## 7.8 添加账号到指定域名

接口功能：添加账号到指定域名

接口地址：

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-add&mailbox=\\*\\*\\*&pwd=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-add&mailbox=***&pwd=***)

GET 参数说明：（红色必须）

1. **pwd**：添加账号的密码
2. **mailbox**：[需要添加的账号，如 test@test.com](#)
3. **customer\_id**：客户 id
4. **user\_type**：用户类型。manager 或 customer，默认为 customer。如果是 manager，username 跟 password 两个参数必须是管理员的用户名跟密码，需要加 customer\_id、user\_type 两参数

## 7.9 删除指定域名下的账号

接口功能：删除指定域名下的账号

接口地址：

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-del&mailbox=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-del&mailbox=***)

GET 参数说明：（红色必须）

1. **mailbox**：[需要删除的账号，如 test@test.com](#)
2. **customer\_id**：客户 id
3. **user\_type**：用户类型。manager 或 customer，默认为 customer。如果是 manager，username 跟 password 两个参数必须是管理员的用户名跟密码，需要加 customer\_id、user\_type 两参数

## 7.10 修改指定域名的账号

接口功能：修改指定域名下的账号

接口地址：

[http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-edit&mailbox=\\*\\*\\*&pwd=\\*\\*\\*](http://www.bestedm.org/mm-ms/apinew/bs.php?do=mailbox-edit&mailbox=***&pwd=***)

GET 参数说明：（红色必须）

1. pwd：修改账号的密码

2. mailbox：需要修改的账号，如 test@test.com

3. customer\_id：客户 id

4. user\_type：用户类型。manager 或 customer，默认为 customer。如果是 manager，username 跟 password 两个参数必须是管理员的用户名跟密码，需要加 customer\_id、user\_type 两参数

## 八、其他

### 8.1 客户自定义群发登录页面说明

说明：

form 表单的 action 属性为



```
#curl -u username:password
http://www.bestedm.org/mm-ms/apinew/task.php?do=list-task&date=2012-03-02
```

2. python 代码请求

```
import urllib2, base64
username = 'XXXX'
password = 'XXXXX'
url =
'http://www.bestedm.org/mm-ms/apinew/task.php?do=list-task&date=2012-03-02'
request = urllib2.Request(url)
base64string = base64.encodestring('%s:%s' % (username, password)).replace('\n',
'')
request.add_header("Authorization", "Basic %s" % base64string)
result = urllib2.urlopen(request)
```

## 9.2 C#用户认证示例代码:

```
HttpWebRequest myReq = (HttpWebRequest)HttpWebRequest.Create("接口地址");

string username="username";
string password="password";
//注意这里的格式哦, 为 "username:password"
string usernamePassword = username + ":" + password;
CredentialCache mycache = new CredentialCache();
mycache.Add(new Uri(url), "Basic", new NetworkCredential(username, password));
myReq.Credentials = mycache;
myReq.Headers.Add("Authorization", "Basic " + Convert.ToBase64String(new
ASCIIEncoding().GetBytes(usernamePassword)));
WebResponse wr = myReq.GetResponse();
Stream receiveStream = wr.GetResponseStream();
StreamReader reader = new StreamReader(receiveStream, Encoding.UTF8);
string content = reader.ReadToEnd();
```

## 9.3 java 用户认证示例代码:

```
HttpGet request = new HttpGet(URL_SECURED_BY_BASIC_AUTHENTICATION);
String auth = DEFAULT_USER + ":" + DEFAULT_PASS;
byte[] encodedAuth =
```

```

Base64 encodeBase64(auth.getBytes(Charset.forName("US-ASCII")));
String authHeader = "Basic " + new String(encodedAuth);
request.setHeader(HttpHeaders.AUTHORIZATION, authHeader);

HttpClient client = HttpClientBuilder.create().build();
HttpResponse response = client.execute(request);

int statusCode = response.getStatusLine().getStatusCode();
assertThat(statusCode, equalTo(HttpStatus.SC_OK));

```

特别注意：在调用 api 接口时如果需要传入中文请务必指定 http 头为 utf-8 格式，并把所传中文转为 utf-8 格式，如下图代码所示：

```

//验证用户
Map<String, String> requestProperties = new HashMap<String, String>();
requestProperties.put("Authorization", "Basic "+validUser());// 你的用户名密码编码
String res = "";// 响应
String boundary = "-----" + new Date().getTime(); // boundary 就是 request 头和上传文件内容的分隔符

URLConnection conn = null;
try {
    if (StringUtils.isBlank(contentType)) {
        contentType = "text/plain";
    }

    URL url = new URL(urlStr);
    conn = (URLConnection) url.openConnection();
    conn.setConnectTimeout(30000);
    conn.setReadTimeout(30000);
    conn.setDoOutput(true);
    conn.setDoInput(true);
    conn.setUseCaches(false);
    conn.setRequestMethod("POST");
    conn.setRequestProperty("Connection", "Keep-Alive");
    conn.setRequestProperty("User-Agent", "Mozilla/5.0 (Windows; U; Windows NT 6.1; zh-CN; rv:1.9.2.6)");
    conn.setRequestProperty("Content-Type", "multipart/form-data; boundary="+ boundary);
    conn.setRequestProperty("charset", "utf-8");
    if (requestProperties != null && requestProperties.size() > 0) {
        for (String key : requestProperties.keySet()) {
            conn.setRequestProperty(key, requestProperties.get(key));
        }
    }
    OutputStream out = new DataOutputStream(conn.getOutputStream());
    // 参数
    if (params != null && params.size() > 0) {
        StringBuffer strBuf = new StringBuffer();
        for (Map.Entry<String, String> entry : params.entrySet()) {
            String key = entry.getKey();
            String value = entry.getValue();
            if (StringUtils.isBlank(key) || StringUtils.isBlank(value)) {
                continue;
            }

```

## 9.4 php 用户认证示例代码

方法一：//---使用 `fsocket` 的方式验证用户名和密码

```
$url = 'api 地址' ;
$fp = fsockopen("$url", 80);
fputs($fp, "GET /downloads HTTP/1.0");
fputs($fp, "Host: $url");
fputs($fp, "Authorization: Basic " . base64_encode("user:pass") . "");
fpassthru($fp);
```

推荐方法二：//---使用 `CURL` 扩展的方式验证密码并调用接口

//---使用 `CURL` 扩展的方式

```
if(extension_loaded('curl')){
    $username = "1xx2"; //用户名
    $password = "xxxxxx"; //密码
    $url =
'http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-addr-lis
t&list_id=461253&page=1&limit=1';
    // $params = array();
    // $vars = http_build_query($params);
    //post 方式
    // $opts = array(
    //     CURLOPT_TIMEOUT => 5,
    //     CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,
    //     CURLOPT_USERPWD=>"$username:$password",
    //     CURLOPT_RETURNTRANSFER => 1,
    //     CURLOPT_URL => $url,
    //     CURLOPT_POST => 1,
    //     CURLOPT_POSTFIELDS => $vars,
    //     CURLOPT_USERAGENT => $_SERVER['HTTP_USER_AGENT']
    // );

    //get 方式
    $opts = array(
        CURLOPT_TIMEOUT => 5,
        CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,
```

```

    CURLOPT_USERPWD=>"$username:$password",
    CURLOPT_RETURNTRANSFER => 1,
    CURLOPT_URL             => $url,
    CURLOPT_HEADER          =>0,
    CURLOPT_USERAGENT       => $_SERVER['HTTP_USER_AGENT']
);

/* 初始化并执行 curl 请求 */
$ch = curl_init();
curl_setopt_array($ch, $opts);
$status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE); //get
status code
$data = curl_exec($ch);
$error = curl_error($ch);
curl_close($ch);
}

```

### 9.5 Ruby On Rails 示例代码:

```

import mx.rpc.events.FaultEvent;
import mx.rpc.events.ResultEvent;
import mx.utils.Base64Encoder;
import mx.rpc.http.HTTPService;
URLRequestDefaults.authenticate = false; //设默认为 false, 否则用户校验错误时会弹出验证框

```

```

private var result:XML;
private function initApp():void
{
    var base64enc:Base64Encoder = new Base64Encoder;
    base64enc.encode("user:password"); //用户名和密码需要 Base64 编码
    var user:String = base64enc.toString();

    var http:HTTPService = new HTTPService;
    http.addEventListener(ResultEvent.RESULT, resultHandler); //监听返回事件
    http.addEventListener(FaultEvent.FAULT, faultHandler); //监听失败事件
    http.resultFormat = "e4x"; //返回格式
    http.url = "api 地址";
    http.headers = {"Authorization":"Basic " + user};
    http.send();
}

```



```

}
private function resultHandler(e:ResultEvent):void
{
    result = XML(e.result);
    test.dataProvider = result.status; //绑定数据
}
private function faultHandler(e:ResultEvent):void
{
    //处理失败
}

```

## 9.6 JS 用户认证示例代码:

```

<script type="text/javascript">
    //需要 Base64 见: http://www.cnblogs.com/pingming/p/4165063.html
    function make_base_auth(user, password) {
        var tok = user + ':' + password;
        var hash = Base64.encode(tok);
        return "Basic " + hash;
    }

    var auth = make_base_auth('Qleelulu', 'mypassword');
    var url = 'api url address';

    // 原始 JavaScript
    xml = new XMLHttpRequest();
    xml.setRequestHeader('Authorization', auth);
    xml.open('GET', url);

    // ExtJS
    Ext.Ajax.request({
        url : url,
        method : 'GET',
        headers : { Authorization : auth }
    });

    // jQuery
    $.ajax({
        url : url,
        method : 'GET',
        beforeSend : function(req) {

```

```

        req.setRequestHeader('Authorization', auth);
    }
});
</script>

```

## 9.7 php 验证用户名密码并调用接口的示例代码

```

//---使用 CURL 扩展的方式验证用户名密码并调用接口
if(extension_loaded('curl')){
    $username = "lixxx";
    $password = "xxxxx";
    $url =
'http://202.103.191.7/mm-ms/apinew/task.php?do=add-task&tpl_id=16
3269&send_domain=all&send_account=all&send_fullname=negoo&status=
2&track_status=1&track_domain=followemail.51negoo.com&maillist_id
=241293&time=2017-11-24+10%3A38%3A40';
    $params = array();
    $params['name'] = '季度经济数据时代';
    // $params['name'] = iconv('gbk', 'utf-8', $params['name']);
    $params['subject'] = '季度经济数据时代';
    $params['content'] = '基金基, 金基金';
    $params['text_content'] = '基金基金基金, 舅舅家';
    $vars = http_build_query($params);
    //psot 方式
    // $opts = array(
    //     CURLOPT_TIMEOUT => 5,
    //     CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,
    //     CURLOPT_USERPWD=>"$username:$password",
    //     CURLOPT_RETURNTRANSFER => 1,
    //     CURLOPT_URL => $url,
    //     CURLOPT_POST => 1,
    //     CURLOPT_POSTFIELDS => $vars,
    //     CURLOPT_USERAGENT => $_SERVER['HTTP_USER_AGENT']
    // );
    //get 方式
    $opts = array(
        CURLOPT_TIMEOUT => 5,
        CURLOPT_HTTPAUTH=>CURLAUTH_BASIC,

```

```

    CURLOPT_USERPWD=>"$username:$password",
    CURLOPT_RETURNTRANSFER => 1,
    CURLOPT_URL             => $url,
    CURLOPT_HEADER          =>0,
    CURLOPT_USERAGENT       => $_SERVER['HTTP_USER_AGENT']
);
/* 初始化并执行 curl 请求 */
$ch = curl_init();
curl_setopt_array($ch, $opts);
$status_code = curl_getinfo($ch, CURLINFO_HTTP_CODE); //get
status code
$data = curl_exec($ch);
$error = curl_error($ch);
curl_close($ch);
}

```

## 9.8 java 验证用户密码并调用上传附件接口示例代码

```

public static void main(String[] args) throws Exception {
    String urlStr =
        "http://www.bestedm.org/mm-ms/apinew/mloperate.php?do=ml-
        addr-add-file";

    // 参数

    Map < String, String > params = new HashMap <> ();
    params . put("list_id", "418315");

    // 文件

    Map < String, Map < String, byte [] >> fileMap = new HashMap
    <> ();
    fileMap . put("file", new HashMap < String, byte [] > ());
    FileInputStream fis = new
    FileInputStream("address.txt");
    byte[] buf = new byte[fis . available()];
    fis . read(buf);
    fileMap . get("file") . put("test.txt", buf);

    fis.close();
    String authStr = "用户名"+" ":"密码";

```

```

byte[] autnEncByte = Base64.encodeBase64(authStr.getBytes());
String authBase64 = New String(autnEncByte);
// 请求验证用户账号密码
Map < String, String >requestProperties = new HashMap<>();
requestProperties.put("Authorization", "Basic
"+authBase64); // 你的用户名密码编码

String contentType = "multipart/form-data";

String s = fileUpload(urlStr, params, fileMap, requestProperties,
contentType);
}
/**
 * 上传文件流
 *
 * @param url
 * @param params
 * @param fileMap: name: {filename:byte[]}
 * @param requestProperties
 */
public static String fileUpload(String urlStr, Map<String, String>
params, Map<String, Map<String, byte[]>> fileMap, Map<String, String>
requestProperties, String contentType) {
    String res = ""; // 响应
    String boundary = "-----" + new
Date().getTime(); // boundary 就是 request 头和上传文件内容的分隔符
    HttpURLConnection conn = null;
    try {
        if (StringUtils.isBlank(contentType)) {
            contentType = "text/plain";
        }
        contentType = "multipart/form-data"; // TODO
        URL url = new URL(urlStr);
        conn = (HttpURLConnection) url.openConnection();
        conn.setConnectTimeout(CON_TIMEOUT);
        conn.setReadTimeout(READ_TIMEOUT);
        conn.setDoOutput(true);
        conn.setDoInput(true);
        conn.setUseCaches(false);
        conn.setRequestMethod("POST");
        conn.setRequestProperty("Connection", "Keep-Alive");

```

```

        conn.setRequestProperty("User-Agent", "Mozilla/5.0 (Windows;
U; Windows NT 6.1; zh-CN; rv:1.9.2.6)");
        conn.setRequestProperty("Content-Type",
"multipart/form-data; boundary=" + boundary);

        if(requestProperties != null && requestProperties.size() >
0){
            for (String key : requestProperties.keySet()) {
                conn.setRequestProperty(key,
requestProperties.get(key));
            }
        }

        OutputStream out = new
DataOutputStream(conn.getOutputStream());

        // 参数
        if (params != null && params.size() > 0) {
            StringBuffer strBuf = new StringBuffer();
            for (Map.Entry<String, String> entry : params.entrySet())
            {
                String key = entry.getKey();
                String value = entry.getValue();
                if (StringUtils.isBlank(key) ||
StringUtils.isBlank(value)) {
                    continue;
                }
                strBuf.append("\r\n").append("--").append(boundary).append("\r\n"
);
                strBuf.append("Content-Disposition: form-data;
name=\"" + key + "\"\r\n\r\n");
                strBuf.append(value);
            }
            out.write(strBuf.toString().getBytes());
        }

        // 上传文件
        if (fileMap != null && fileMap.size() > 0) {
            for (Map.Entry<String, Map<String, byte[]>> entry :
fileMap.entrySet()) {
                Map<String, byte[]> file = entry.getValue();
                String name = entry.getKey();// 上传表单的 name
                String filename = file.keySet().iterator().next();// 文

```

件名

```

        byte[] fileByte = file.get(filename); // 文件的字节
        if (StringUtils.isBlank(name) ||
        StringUtils.isBlank(filename) || fileByte == null || fileByte.length
        == 0) {
            continue;
        }
        StringBuffer strBuf = new StringBuffer();
        strBuf.append("\r\n").append("--").append(boundary).append("\r\n"
        );
        strBuf.append("Content-Disposition: form-data;
        name=\"" + name + "\"; filename=\"" + "address.txt" + "\"\r\n");
        // strBuf.append("Content-Type: " + contentType +
        "\r\n\r\n");// 默认 txt 文件格式
        strBuf.append("Content-Type: " + "multipart/form-data"
        + "\r\n\r\n");
        out.write(strBuf.toString().getBytes());

        ByteArrayInputStream bais = new
        ByteArrayInputStream(fileByte);
        int bytes = 0;
        byte[] bufferOut = new byte[1024];
        while ((bytes = bais.read(bufferOut)) != -1) {
            out.write(bufferOut, 0, bytes);
        }
        bais.close();
    }
}

out.write("\r\n--" + boundary + "--\r\n").getBytes());
out.flush();
out.close();

// 读取返回数据
StringBuffer strBuf = new StringBuffer();
BufferedReader reader = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
String line = null;
while ((line = reader.readLine()) != null) {
    strBuf.append(line).append("\n");
}
res = strBuf.toString();
reader.close();

```

```

        reader = null;
    } catch (Exception e) {
        logger.error("上传文件失败, 请求路径【" + urlStr + "】, 失败原因:
" + e);
        e.printStackTrace();
    } finally {
        if (conn != null) {
            conn.disconnect();
            conn = null;
        }
    }
    return res;
}

```

## 十、群发 smtp 对接方式发送邮件代码样例

### 1、php 版本 smtp 对接发送 demo 代码

```

<?php
/**
 * Created by umail
 * User: NUC
 * Date: 2018/8/22
 * Time: 14:10
 */
class smtp
{
    /* Public Variables */
    public $smtp_port;
    public $time_out;
    public $host_name;
    public $log_file;
    public $relay_host;
    public $debug;
    public $auth;
    public $user;
    public $pass;

    /* Private Variables */
    private $sock;

    /* Constructor */

```

```

// function __construct($relay_host = "", $smtp_port = 25, $auth = false, $user, $pass) //
与下作用相同
function smtp($relay_host = "", $smtp_port = 25, $auth = false, $user, $pass)
{
    $this->debug = FALSE;
    $this->smtp_port = $smtp_port;
    $this->relay_host = $relay_host;
    $this->time_out = 30; //is used in fsockopen()
    #
    $this->auth = $auth;//auth
    $this->user = $user;
    $this->pass = $pass;
    #
    $this->host_name = "localhost"; //is used in HELO command
    $this->log_file = "";

    $this->sock = FALSE;
}

/* Main Function */
function sendmail($to, $from, $subject = "", $body = "", $mailtype, $cc = "", $bcc = "",
$additional_headers = "")
{
    $header = '';
    $mail_from = $this->get_address($this->strip_comment($from));
    $body = preg_replace("/^(^|(\r\n))(\.)/", "\\1.\\3", $body);
    $header .= "MIME-Version:1.0\r\n";
    if($mailtype=="HTML"){
        $header .= "Content-Type:text/html\r\n";
    }
    $header .= "To: ".$to."\r\n";
    if ($cc != "") {
        $header .= "Cc: ".$cc."\r\n";
    }
    $header .= "From: $from<".$from.">\r\n";
    $header .= "Subject: ".$subject."\r\n";
    $header .= $additional_headers;
    $header .= "Date: ".date("r")."\r\n";
    $header .= "X-Mailer:By Redhat (PHP/".$phpversion().")\r\n";
    list($msec, $sec) = explode(" ", microtime());
    $header .= "Message-ID: <".date("YmdHis",
$sec).".$msec*1000000).".$mail_from.">\r\n";
    $TO = explode(",", $this->strip_comment($to));

```



```

if ($cc != "") {
    $TO = array_merge($TO, explode(",", $this->strip_comment($cc)));
}

if ($bcc != "") {
    $TO = array_merge($TO, explode(",", $this->strip_comment($bcc)));
}

$sent = TRUE;
foreach ($TO as $rcpt_to) {
    $rcpt_to = $this->get_address($rcpt_to);
    if (!$this->smtp_sockopen($rcpt_to)) {
        $this->log_write("Error: Cannot send email to ".$rcpt_to."\n");
        $sent = FALSE;
        continue;
    }
    if ($this->smtp_send($this->host_name, $mail_from, $rcpt_to, $header, $body))
    {
        $this->log_write("E-mail has been sent to <".$rcpt_to.">\n");
    } else {
        $this->log_write("Error: Cannot send email to <".$rcpt_to.">\n");
        $sent = FALSE;
    }
    fclose($this->sock);
    $this->log_write("Disconnected from remote host\n");
}
echo "<br>";
echo $header; //
return $sent;
}

/* Private Functions */

function smtp_send($helo, $from, $to, $header, $body = "")
{
    if (!$this->smtp_putcmd("HELO", $helo)) {
        return $this->smtp_error("sending HELO command");
    }

    #auth
    if($this->auth){
        if (!$this->smtp_putcmd("AUTH LOGIN", base64_encode($this->user))) {

```

```

        return $this->smtp_error("sending HELO command");
    }
    if (!$this->smtp_putcmd("", base64_encode($this->pass))) {
        return $this->smtp_error("sending HELO command");
    }
}
#
if (!$this->smtp_putcmd("MAIL", "FROM:<".$from.">")) {
    return $this->smtp_error("sending MAIL FROM command");
}

if (!$this->smtp_putcmd("RCPT", "TO:<".$to.">")) {
    return $this->smtp_error("sending RCPT TO command");
}

if (!$this->smtp_putcmd("DATA")) {
    return $this->smtp_error("sending DATA command");
}

if (!$this->smtp_message($header, $body)) {
    return $this->smtp_error("sending message");
}

if (!$this->smtp_eom()) {
    return $this->smtp_error("sending <CR><LF>.<CR><LF> [EOM]");
}

if (!$this->smtp_putcmd("QUIT")) {
    return $this->smtp_error("sending QUIT command");
}

return TRUE;
}

function smtp_sockopen($address)
{
    if ($this->relay_host == "") {
        return $this->smtp_sockopen_mx($address);
    } else {
        return $this->smtp_sockopen_relay();
    }
}
}

```

```

function smtp_sockopen_relay()
{
    $this->log_write("Trying to ".$this->relay_host.":".$this->smtp_port."\n");
    $this->sock = @fsockopen($this->relay_host, $this->smtp_port, $errno, $errstr,
$this->time_out);
    if (!$this->sock && $this->smtp_ok()) {
        $this->log_write("Error: Cannot connenct to relay host
".$this->relay_host."\n");
        $this->log_write("Error: ".$errstr." (".$errno.")\n");
        return FALSE;
    }
    $this->log_write("Connected to relay host ".$this->relay_host."\n");
    return TRUE;;
}

function smtp_sockopen_mx($address)
{
    $domain = preg_replace("/^\.+@([\^@]+)$/", "\\1", $address);
    if (!@getmxrr($domain, $MXHOSTS)) {
        $this->log_write("Error: Cannot resolve MX \"".$domain."\"");
        return FALSE;
    }
    foreach ($MXHOSTS as $host) {
        $this->log_write("Trying to ".$host.":".$this->smtp_port."\n");
        $this->sock = @fsockopen($host, $this->smtp_port, $errno, $errstr,
$this->time_out);
        if (!$this->sock && $this->smtp_ok()) {
            $this->log_write("Warning: Cannot connect to mx host ".$host."\n");
            $this->log_write("Error: ".$errstr." (".$errno.")\n");
            continue;
        }
        $this->log_write("Connected to mx host ".$host."\n");
        return TRUE;
    }
    $this->log_write("Error: Cannot connect to any mx hosts (".implode(" ",
$MXHOSTS).")\n");
    return FALSE;
}

function smtp_message($header, $body)
{
    fputs($this->sock, $header."\r\n".$body);
    $this->smtp_debug("> ".str_replace("\r\n", "\n."> ", $header."\n> ".$body."\n>

```

```

    ));

    return TRUE;
}

function smtp_eom()
{
    fputs($this->sock, "\r\n.\r\n");
    $this->smtp_debug(". [EOM]\n");

    return $this->smtp_ok();
}

function smtp_ok()
{
    $response = str_replace("\r\n", "", fgets($this->sock, 512));
    $this->smtp_debug($response."\n");

    if (!preg_match("/^[23]/", $response)) {
        fputs($this->sock, "QUIT\r\n");
        fgets($this->sock, 512);
        $this->log_write("Error: Remote host returned \"".$response."\"");
        return FALSE;
    }
    return TRUE;
}

function smtp_putcmd($cmd, $arg = "")
{
    if ($arg != "") {
        if ($cmd=="") $cmd = $arg;
        else $cmd = $cmd." ".$arg;
    }

    fputs($this->sock, $cmd."\r\n");
    $this->smtp_debug("> ".$cmd."\n");

    return $this->smtp_ok();
}

function smtp_error($string)
{
    $this->log_write("Error: Error occurred while ".$string."");
}

```

```

    return FALSE;
}

function log_write($message)
{
    $this->smtp_debug($message);

    if ($this->log_file == "") {
        return TRUE;
    }

    $message = date("M d H:i:s ").get_current_user(). "[" . getmypid() . "]: " . $message;
    if (!@file_exists($this->log_file) || !($fp = @fopen($this->log_file, "a"))) {
        $this->smtp_debug("Warning: Cannot open log file \"\". $this->log_file. \"\"\\n");
        return FALSE;
    }
    flock($fp, LOCK_EX);
    fputs($fp, $message);
    fclose($fp);

    return TRUE;
}

function strip_comment($address)
{
    $comment = "/\([\^()]*\)/";
    while (preg_match($comment, $address)) {
        $address = preg_replace($comment, "", $address);
    }

    return $address;
}

function get_address($address)
{
    $address = preg_replace("/([\t\r\n]+)/", "", $address);
    $address = preg_replace("/^\.*<(.)>.*$/", "\\1", $address);

    return $address;
}

function smtp_debug($message)
{

```

```

    if ($this->debug) {
        echo $message."<br>";
    }
}

function get_attach_type($image_tag) { //

    $filedata = array();

    $img_file_con=fopen($image_tag,"r");
    unset($image_data);
    while ($tem_buffer=AddSlashes(fread($img_file_con,filesize($image_tag))))
        $image_data.=$tem_buffer;
    fclose($img_file_con);

    $filedata['context'] = $image_data;
    $filedata['filename'] = basename($image_tag);

    $extension=substr($image_tag, strrpos($image_tag, "."), strlen($image_tag)-strrpos($image_t
ag, "."));
    switch($extension) {
        case ".gif":
            $filedata['type'] = "image/gif";
            break;
        case ".gz":
            $filedata['type'] = "application/x-gzip";
            break;
        case ".htm":
            $filedata['type'] = "text/html";
            break;
        case ".html":
            $filedata['type'] = "text/html";
            break;
        case ".jpg":
            $filedata['type'] = "image/jpeg";
            break;
        case ".tar":
            $filedata['type'] = "application/x-tar";
            break;
        case ".txt":
            $filedata['type'] = "text/plain";
            break;
        case ".zip":

```

```

        $filedata[' type' ] = "application/zip";
        break;
    default:
        $filedata[' type' ] = "application/octet-stream";
        break;
    }
    return $filedata;
}

} // end class

/**
 *实例化邮件类
 */
date_default_timezone_set('Asia/Shanghai'); // 'Asia/Chongqing' or 'PRC'
$smtpserver = "smtp.bestedm.org"; //SMTP 服务器
$smtpserverport =25; //SMTP 服务器端口
$smtpusermail = "xxxx@mail79.comingchina.com"; //SMTP 服务器的用户邮箱
$smtpmailto = "784885758@qq.com"; //发送给谁
$smtpuser = "xxxx@mail79.comingchina.com"; //SMTP 服务器的用户帐号
$smtppass = "xxxxx"; //SMTP 服务器的用户密码
$mailsubject = "测试 umail 邮件系统"; //邮件主题
$mailbody = "<h1>这是 smtp 测试邮件</h1>"; //邮件内容
$mailtype = "HTML"; //邮件格式 (HTML/TXT) ,TXT 为文本邮件
$smtp = new smtp($smtpserver, $smtpserverport, true, $smtpuser, $smtppass);
$smtp->debug = true; //是否显示发送的调试信息
$smtp->sendmail($smtpmailto, $smtpusermail, $mailsubject, $mailbody, $mailtype);
?>

```

## 2、python 版本 smtp 对接发送 demo 代码

```

import smtplib
from email.mime.text import MIMEText
mailto_list=["YYY@test.com"]
mail_host="smtp.bestedm.org" #设置服务器
mail_user="xxx@bestedm.org" #账号邮箱
mail_pass="密码 xxx" #密码

def send_mail(to_list,sub,content): #to_list: 收件人; sub: 主题; content: 邮件内容
    me="hello"+"<" +mail_user+">" #这里的 hello 可以任意设置, 收到信后, 将按照设置显示

```

```

    msg = MIMEText(content, _subtype='html', _charset='gb2312')    #创建一个实例,这里设置为html
格式邮件
    msg['Subject'] = sub    #设置主题
    msg['From'] = me
    msg['To'] = ",".join(to_list)
    try:
        s = smtplib.SMTP()
        s.connect(mail_host, 25) #连接 smtp 服务器
        s.login(mail_user,mail_pass) #登陆服务器
        s.sendmail(me, to_list, msg.as_string()) #发送邮件
        s.close()
        return True
    except Exception, e:
        print str(e)
        return False
if __name__ == '__main__':
    if send_mail(mailto_list,"hello","测试内容"):
        print "发送成功"
    else:
        print "发送失败"

```

### 3、java 版本 smtp 对接发送 demo 代码

第一步：引入依赖的 jar 包

```

<dependency>
    <groupId>com.sun.mail</groupId>
    <artifactId>javax.mail</artifactId>
    <version>1.5.6</version>
</dependency>

```

第二步：实体类介绍(可以直接复制，账号替换为客服分配的 smtp 账号即可)

邮件参数实体

```

import java.util.List;
import java.util.Properties;
import com.sun.mail.util.MailSSLSocketFactory;

/**
 * 邮件发送配置参数信息
 * @ClassName: MailSenderInfo
 * @Description: TODO
 * @author umail
 * @Date 2019年5月8日 下午6:27:35

```



```

*
*/
public class MailSenderInfo {
    // 发送邮件的服务器的 IP
    private String mailServerHost;
    //发送邮件的服务器端口, 该处暂时默认 25
    private String mailServerPort = "25";
    // 邮件发送者的地址
    private String fromAddress;
    // 邮件接收者的地址
    private String toAddress;
    //邮件接收者的地址集合
    private String[] toBatchAddress;
    // 登陆邮件发送服务器的用户名
    private String userName;
    //登陆邮件发送服务器的密码
    private String password;
    // 是否需要身份验证 [默认 false 不认证]
    private boolean validate = false;
    // 邮件主题
    private String subject;
    // 邮件主题
    private String[] subjects;
    // 邮件的文本内容
    private String content;
    // 邮件的文本内容
    private String[] contents;
    // 邮件附件的文件名
    private String[] attachFileNames;
    // 邮件附件的文件名 针对一邮件带多个附件关系
    private List<String[]> attachFileList;
    /**
     * 获得邮件会话属性
     */
    public Properties getProperties() {
        Properties p = new Properties();
        try {
            p.put("mail.smtp.host", this.mailServerHost);
            p.put("mail.smtp.port", this.mailServerPort);
            p.put("mail.smtp.auth", validate ? "true" : "false");
            p.setProperty("mail.transport.protocol", "smtp");
            if("smtp.qq.com".equals(this.mailServerHost)) {
                MailSSLConnectionFactory sf = new MailSSLConnectionFactory();

```

```

        sf.setTrustAllHosts(true);
        p.put("mail.smtp.ssl.enable", "true");
        p.put("mail.smtp.ssl.socketFactory", sf);
    }
} catch (Exception e) {
    e.printStackTrace();
}
return p;
}
/**
 * 发送邮件的服务器的 IP 如: smtp.bestedm.org
 */
public String getMailServerHost() {
    return mailServerHost;
}
/**
 * 发送邮件的服务器的 IP 如: smtp.bestedm.org
 */
public void setMailServerHost(String mailServerHost) {
    this.mailServerHost = mailServerHost;
}
/**
 * 发送邮件的服务器端口, 如: umail 默认 25
 */
public String getMailServerPort() {
    return mailServerPort;
}
/**
 * 发送邮件的服务器端口, 如: 网易邮箱默认 25
 */
public void setMailServerPort(String mailServerPort) {
    this.mailServerPort = mailServerPort;
}
/**
 * 是否需要身份验证 [默认 false 不认证]
 */
public boolean isValiddate() {
    return validate;
}
/**
 * 是否需要身份验证 [默认 false 不认证]
 */
public void setValidate(boolean validate) {

```

```
        this.validate = validate;
    }
    /**
     * 邮件附件的文件名
     */
    public String[] getAttachFileNames() {
        return attachFileNames;
    }
    /**
     * 邮件附件的文件名
     */
    public void setAttachFileNames(String[] fileNames) {
        this.attachFileNames = fileNames;
    }
    /**
     * 邮件发送者的邮箱地址
     */
    public String getFromAddress() {
        return fromAddress;
    }
    /**
     * 邮件发送者的邮箱地址
     */
    public void setFromAddress(String fromAddress) {
        this.fromAddress = fromAddress;
    }
    /**
     * 邮件发送者的邮箱密码
     */
    public String getPassword() {
        return password;
    }
    /**
     * 邮件发送者的邮箱密码
     */
    public void setPassword(String password) {
        this.password = password;
    }
    /**
     * 邮件接收者的地址
     */
    public String getToAddress() {
        return toAddress;
    }
}
```

```
}  
/**  
 * 邮件接收者的地址  
 */  
public void setToAddress(String toAddress) {  
    this.toAddress = toAddress;  
}  
/**  
 * 邮件接收者的地址集合  
 * @return  
 */  
public String[] getToBatchAddress() {  
    return toBatchAddress;  
}  
/**  
 * 邮件接收者的地址集合  
 * @param toBatchAddress  
 */  
public void setToBatchAddress(String[] toBatchAddress) {  
    this.toBatchAddress = toBatchAddress;  
}  
/**  
 * 登陆邮件发送服务器的用户名  
 */  
public String getUsername() {  
    return userName;  
}  
/**  
 * 登陆邮件发送服务器的用户名  
 */  
public void setUsername(String userName) {  
    this.userName = userName;  
}  
/**  
 * 邮件主题  
 */  
public String getSubject() {  
    return subject;  
}  
/**  
 * 邮件主题  
 */  
public void setSubject(String subject) {
```

```
        this.subject = subject;
    }
    /**
     * 邮件主题
     */
    public String[] getSubjects() {
        return subjects;
    }
    /**
     * 邮件主题
     */
    public void setSubjects(String[] subjects) {
        this.subjects = subjects;
    }
    /**
     * 邮件的文本内容
     */
    public String getContent() {
        return content;
    }
    /**
     * 邮件的文本内容
     */
    public void setContent(String textContent) {
        this.content = textContent;
    }
    /**
     * 邮件的文本内容
     */
    public String[] getContents() {
        return contents;
    }
    /**
     * 邮件的文本内容
     */
    public void setContents(String[] contents) {
        this.contents = contents;
    }
    /**
     * 针对一邮件多附件
     */
    public List<String[]> getAttachFileList() {
        return attachFileList;
    }
}
```

```

    }
    /**
     * 针对一邮件多附件
     */
    public void setAttachFileList(List<String[]> attachFileList) {
        this.attachFileList = attachFileList;
    }
}

```

## 邮件身份认证类

```

import javax.mail.Authenticator;
import javax.mail.PasswordAuthentication;

/**
 * 邮箱参数
 * @ClassName: MailAuthenticator
 * @Description: TODO
 * @author umail
 * @Date 2019年5月8日 下午6:22:59
 */
public class MailAuthenticator extends Authenticator{
    String userName=null;
    String password=null;

    public MailAuthenticator(){
    }
    public MailAuthenticator(String username, String password) {
        this.userName = username;
        this.password = password;
    }
    protected PasswordAuthentication getPasswordAuthentication(){
        return new PasswordAuthentication(userName, password);
    }
}

```

## 邮件发送器

```

import java.io.UnsupportedEncodingException;

```

```

import java.util.Date;
import java.util.Properties;

import javax.activation.DataHandler;
import javax.activation.FileDataSource;
import javax.mail.Address;
import javax.mail.BodyPart;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.Multipart;
import javax.mail.NoSuchProviderException;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.mail.internet.MimeUtility;

/**
 * 邮件发送器
 * @ClassName: SimpleMailSender
 * @Description: TODO
 * @author umail
 * @Date 2019 年 5 月 8 日 下午 6:23:20
 *
 */
public class SimpleMailSender {

    /**
     * 以文本格式发送邮件
     *
     * @param mailInfo
     *          待发送的邮件的信息
     */
    public boolean sendTextMail(MailSenderInfo mailInfo) {
        // 判断是否需要身份认证
        MailAuthenticator authenticator = null;
        Properties pro = mailInfo.getProperties();
        if (mailInfo.isValidate()) {
            // 如果需要身份认证，则创建一个密码验证器
            authenticator = new MailAuthenticator(mailInfo.getUserName()),

```

```

        mailInfo.getPassword());
    }
    // 根据邮件会话属性和密码验证器构造一个发送邮件的 session
    Session sendMailSession = Session
        .getDefaultInstance(pro, authenticator);
    try {
        // 根据 session 创建一个邮件消息
        Message mailMessage = new MimeMessage(sendMailSession);
        // 创建邮件发送者地址
        Address from = new InternetAddress(mailInfo.getFromAddress());
        // 设置邮件消息的发送者
        mailMessage.setFrom(from);
        // 创建邮件的接收者地址，并设置到邮件消息中
        if (mailInfo.getToAddress() != null
            && mailInfo.getToAddress().length() > 0) {
            Address to = new InternetAddress(mailInfo.getToAddress());
            // Message.RecipientType.TO 属性表示接收者的类型为 TO
            mailMessage.setRecipient(Message.RecipientType.TO, to);
        } else if (mailInfo.getToBatchAddress() != null
            && mailInfo.getToBatchAddress().length > 0) {
            final int size = mailInfo.getToBatchAddress().length;
            Address[] to = new Address[size];
            for (int i = 0; i < size; i++) {
                to[i] = new InternetAddress(mailInfo.getToBatchAddress()[i]);
            }
            // Message.RecipientType.TO 属性表示接收者的类型为 TO
            mailMessage.setRecipients(Message.RecipientType.TO, to);
        }
        // 设置邮件消息的主题
        mailMessage.setSubject(mailInfo.getSubject());
        // 设置邮件消息发送的时间
        mailMessage.setSentDate(new Date());
        // 设置邮件消息的主要内容
        String mailContent = mailInfo.getContent();
        mailMessage.setText(mailContent);
        // 发送邮件
        Transport.send(mailMessage);
        return true;
    } catch (MessagingException ex) {
        ex.printStackTrace();
    }
    return false;
}

```



```

/**
 * 以 HTML 格式发送邮件
 *
 * @param mailInfo
 *         待发送的邮件信息
 */
public boolean sendHtmlMail(MailSenderInfo mailInfo) {
    // 判断是否需要身份认证
    MailAuthenticator authenticator = null;
    Properties pro = mailInfo.getProperties();
    // 如果需要身份认证, 则创建一个密码验证器
    if (mailInfo.isValidDate()) {
        authenticator = new MailAuthenticator(mailInfo.getUserName(),
            mailInfo.getPassword());
    }
    // 根据邮件会话属性和密码验证器构造一个发送邮件的 session
    Session sendMailSession = Session
        .getDefaultInstance(pro, authenticator);
    try {
        sendMailSession.setDebug(true);
        // 根据 session 创建一个邮件消息
        Message mailMessage = new MimeMessage(sendMailSession);
        // 创建邮件发送者地址
        Address from = new InternetAddress(mailInfo.getFromAddress());
        // 设置邮件消息的发送者
        mailMessage.setFrom(from);
        // 创建邮件的接收者地址, 并设置到邮件消息中
        if (mailInfo.getToAddress() != null
            && mailInfo.getToAddress().length() > 0) {
            Address to = new InternetAddress(mailInfo.getToAddress());
            // Message.RecipientType.TO 属性表示接收者的类型为 TO
            mailMessage.setRecipient(Message.RecipientType.TO, to);
        } else if (mailInfo.getToBatchAddress() != null
            && mailInfo.getToBatchAddress().length > 0) {
            final int size = mailInfo.getToBatchAddress().length;
            Address[] to = new Address[size];
            for (int i = 0; i < size; i++) {
                to[i] = new InternetAddress(mailInfo.getToBatchAddress()[i]);
            }
            // Message.RecipientType.TO 属性表示接收者的类型为 TO
            mailMessage.setRecipients(Message.RecipientType.TO, to);
        }
    }
}

```

```

// 设置邮件消息的主题
mailMessage.setSubject(mailInfo.getSubject());
mailMessage.setSubject(MimeUtility.encodeText(mailInfo.getSubject(), "UTF-8",
"B"));

// 设置邮件消息发送的时间
mailMessage.setSentDate(new Date());
// MiniMultipart 类是一个容器类, 包含 MimeBodyPart 类型的对象
Multipart mainPart = new MimeMultipart();

//-----begin 文本-----//
// 创建一个包含 HTML 内容的 MimeBodyPart
BodyPart html = new MimeBodyPart();
// 设置 HTML 内容
html.setContent(mailInfo.getContent(), "text/html; charset=utf-8");
mainPart.addBodyPart(html);
//-----end 文本-----//

//-----begin 附件-----//
if(mailInfo.getAttachFileList() != null && mailInfo.getAttachFileList().size() >
0) {
    for (String[] files : mailInfo.getAttachFileList()) {
        for (String file : files) {
            //邮件的附件
            String fileName = file;
            if(fileName != null&&!fileName.trim().equals("")) {
                MimeBodyPart mbp = new MimeBodyPart();
                FileDataSource fileSource = new FileDataSource(fileName);
                mbp.setDataHandler(new DataHandler(fileSource));
                try {
                    mbp.setFileName(MimeUtility.encodeText(fileSource.getName()));
                } catch (UnsupportedEncodingException e) {
                    e.printStackTrace();
                }
                mainPart.addBodyPart(mbp);
            }
        }
    }
} else {
    if(mailInfo.getAttachFileNames() != null &&
mailInfo.getAttachFileNames().length > 0) {
        //邮件的附件

```

```

String fileName = mailInfo.getAttachFileNames()[0];
if(fileName != null&&!fileName.trim().equals("")) {
    MimeBodyPart mbp = new MimeBodyPart();
    FileDataSource fileSource = new FileDataSource(fileName);
    mbp.setDataHandler(new DataHandler(fileSource));
    try {
        mbp.setFileName(MimeUtility.encodeText(fileSource.getName()));
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
    mainPart.addBodyPart(mbp);
}
}
}

//-----end 附件-----//

// 将 MiniMultipart 对象设置为邮件内容
mailMessage.setContent(mainPart);
// 发送邮件
Transport.send(mailMessage);
System.out.println("发送成功!");
return true;
} catch (MessagingException ex) {
    ex.printStackTrace();
} catch (UnsupportedEncodingException e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
}
return false;
}
}

/**
 * 以 HTML 格式发送多封邮件
 *
 * @param mailInfo
 *      待发送的邮件信息
 */
public boolean sendBatchHtmlMail(MailSenderInfo mailInfo) {
    // 判断是否需要身份认证
    MailAuthenticator authenticator = null;
    Properties pro = mailInfo.getProperties();

```

```

pro.setProperty("mail.transport.protocol", "smtp");
// 如果需要身份认证, 则创建一个密码验证器
if (mailInfo.isValiddate()) {
    authenticator = new MailAuthenticator(mailInfo.getUserName(),
        mailInfo.getPassword());
}
// 根据邮件会话属性和密码验证器构造一个发送邮件的 session
Session sendMailSession = Session.getInstance(pro, authenticator);
try {
    // 发送邮件
    sendMailSession.setDebug(true);
    Transport transport = sendMailSession.getTransport();

transport.connect(mailInfo.getMailServerHost(), Integer.parseInt(mailInfo.getMailServerPort()
),
        mailInfo.getUserName(), mailInfo.getPassword());
// 创建邮件的接收者地址, 并设置到邮件消息中
if (mailInfo.getToBatchAddress() != null
    && mailInfo.getToBatchAddress().length > 0) {
    final int size = mailInfo.getToBatchAddress().length;
    for (int i = 0; i < size; i++) {
        // 根据 session 创建一个邮件消息
        Message mailMessage = new MimeMessage(sendMailSession);
        // 创建邮件发送者地址
        Address from = new InternetAddress(mailInfo.getFromAddress());
        // 设置邮件消息的发送者
        mailMessage.setFrom(from);

        // 设置邮件消息的主题
        mailMessage.setSubject(mailInfo.getSubjects()[i]);
        mailMessage.setSubject(MimeUtility.encodeText(mailInfo.getSubjects()[i],
"UTF-8", "B"));

        // 设置邮件消息发送的时间
        mailMessage.setSentDate(new Date());
        // MiniMultipart 类是一个容器类, 包含 MimeBodyPart 类型的对象
        Multipart mainPart = new MimeMultipart();

        //-----begin 文本-----//
        // 创建一个包含 HTML 内容的 MimeBodyPart
        BodyPart html = new MimeBodyPart();
        // 设置 HTML 内容
        html.setContent(mailInfo.getContents()[i], "text/html; charset=utf-8");
        mainPart.addBodyPart(html);

```

```

//-----end 文本-----//

//-----begin 附件-----//
if(mailInfo.getAttachFileList() != null &&
mailInfo.getAttachFileList().size() > 0){
    String[] files = mailInfo.getAttachFileList().get(i);
    for (String file : files) {
        //邮件的附件
        String fileName = file;
        if(fileName != null&&!fileName.trim().equals("")) {
            MimeBodyPart mbp = new MimeBodyPart();
            FileDataSource fileSource = new FileDataSource(fileName);
            mbp.setDataHandler(new DataHandler(fileSource));
            try {

mbp.setFileName(MimeUtility.encodeText(fileSource.getName()));
                //System.out.println("ceshi: " +
MimeUtility.encodeText(fileSource.getName(),"utf-8",null));
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            }
            mainPart.addBodyPart(mbp);
        }
    }
} else {
    if(mailInfo.getAttachFileNames() != null &&
mailInfo.getAttachFileNames().length > 0){
        //邮件的附件
        String fileName = mailInfo.getAttachFileNames()[i];
        if(fileName != null&&!fileName.trim().equals("")) {
            MimeBodyPart mbp = new MimeBodyPart();
            FileDataSource fileSource = new FileDataSource(fileName);
            mbp.setDataHandler(new DataHandler(fileSource));
            try {

mbp.setFileName(MimeUtility.encodeText(fileSource.getName()));
                //System.out.println("ceshi: " +
MimeUtility.encodeText(fileSource.getName(),"utf-8",null));
            } catch (UnsupportedEncodingException e) {
                e.printStackTrace();
            }
            mainPart.addBodyPart(mbp);
        }
    }
}

```

```

    }
}
//-----end 附件-----//

// 将 MiniMultipart 对象设置为邮件内容
mailMessage.setContent(mainPart);

    Address[] to = new Address[]{new
InternetAddress(mailInfo.getToBatchAddress()[i])};
    mailMessage.setRecipient(Message.RecipientType.TO, to[0]);
    transport.sendMessage(mailMessage, to);
    }
}
transport.close();
System.out.println("发送成功!");
return true;
} catch (AddressException e) {
    e.printStackTrace();
} catch (NoSuchProviderException e) {
    e.printStackTrace();
} catch (MessagingException e) {
    e.printStackTrace();
} catch (UnsupportedEncodingException e) {
    e.printStackTrace();
}

return false;
}
}
}

```

### 测试代码:

```

import java.util.Date;
import java.util.Properties;
import javax.mail.Address;
import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

import com.only.mate.email.MailAuthenticator;
import com.only.mate.email.MailSenderInfo;

```

```

import com.only.mate.email.SimpleMailSender;
import com.sun.mail.util.MailSSLSocketFactory;

public class EmailTest {
    private MailSenderInfo mailSenderInfo;

    public static void main(String[] args) throws Exception {
        EmailTest test = new EmailTest();
        // test.sendTextEmail();
        // test.sendTextEmail();
        test.sendHtmlEmail();
    }

    {
        mailSenderInfo = new MailSenderInfo();
        mailSenderInfo.setMailServerHost("smtp.bestedm.org");
        mailSenderInfo.setMailServerPort("465");
        mailSenderInfo.setUsername("*****@xxx.com");
        mailSenderInfo.setPassword("*****");
        mailSenderInfo.setFromAddress("*****@xxx.com");
        mailSenderInfo.setToAddress("*****@xxx.com");
        mailSenderInfo.setValidate(true);
        mailSenderInfo.setSubject("主题-你猜猜?");
        Date date = new Date();
        String content = String.format(
            "<!DOCTYPE html>"+
            "<html lang=\"en\">"+
            "<head>"+
            "<meta charset=\"UTF-8\" />"+
            "<title></title>"+
            "</head>"+
            "<style type=\"text/css\">hml,body{margin: 0;padding: 0;font-size:
14px;}.container{width: 880px;margin:0 auto;background: #e7f5ff;height:800px;padding-top:
80px;margin-top: 20px;}.container-con{width:680px;margin:0
auto;background:#fff;height:600px;padding:20px;}.email-top{font-size:
14px;}.email-top>span{color:#000;font-weight: bold;}.email-top2{font-size: 14px;padding-left:
16px;margin-bottom: 30px;}.email-con{padding:20px;}.email-con>p{line-height:
20px;}.top-img{background:url(\"images/tt0_03.png\") no-repeat;background-size: cover;
width:722px;height:100px;margin:0 auto;}.fpptwe{line-height: 30px;}.footer{float:
right;}.jingao{font-size: 12px;color:#888}</style>"+
            "<body>"+
            "<div class=\"container\">"+
            "<div class=\"top-img\"></div>"+

```

```

"<div class=\"container-con\">"+
    "<p class=\"email-top\">"+
        "尊敬的 XX 女士"+
    "</p>"+
    "<p class=\"email-top2\">您好! </p>"+
    "<div class=\"email-con\">"+
        "<p>您所提交“XXX”的申请, 已通过.</p>"+
        "<p>"+
            "请及时前往去享受! <span>%s</span>"+
        "</p>"+
        "<img
src='http://img.mp.itc.cn/upload/20160326/73a64c935e7d4c9594bdf86d76399226_th.jpg' />"+
        "</div>"+
    "<p class=\"jingao\">(这是一封系统自动发送的邮件, 请不要直接回
复。) </p>"+
    "<div class=\"footer\">"+
        "<p>爱的港湾</p>"+
        "<span>%tF %tT</span>"+
    "</div>"+
    "</p>"+
    "</div>"+
    "</div>"+
    "</body>"+
    "</html>", "❤️️️️️", date, date);
mailSenderInfo.setContent(content);
// mailSenderInfo.setContent("其实只是好玩而已");

mailSenderInfo.setAttachFileNames(new String[]
{"C:/Users/OnlyMate/Pictures/MyPhoneTheme.jpg"});
}
public void sendTextEmail() throws Exception {
    SimpleMailSender sender = new SimpleMailSender();
    sender.sendTextMail(mailSenderInfo);
}
public void sendHtmlEmail() throws Exception {
    SimpleMailSender sender = new SimpleMailSender();
    sender.sendHtmlMail(mailSenderInfo);
}
public void sendTextEmail1() throws Exception {
    Properties props = new Properties();
    // 开启 debug 调试
    props.setProperty("mail.debug", "false");
    // 发送服务器需要身份验证

```



```

props.setProperty("mail.smtp.auth", "true");
// 使用 STARTTLS 安全连接
props.put("mail.smtp.starttls.enable", "true");
// 设置邮件服务器主机名
props.setProperty("mail.smtp.host", mailSenderInfo.getMailServerHost());
// 设置邮件服务器端口
props.put("mail.smtp.port", mailSenderInfo.getMailServerPort());
// 发送邮件协议名称
props.setProperty("mail.transport.protocol", "smtp");

MailSSLConnectionFactory sf = new MailSSLConnectionFactory();
sf.setTrustAllHosts(true);
props.put("mail.smtp.ssl.enable", "true");
props.put("mail.smtp.ssl.socketFactory", sf);

// 如果需要身份认证, 则创建一个密码验证器
MailAuthenticator authenticator = new MailAuthenticator(mailSenderInfo.getUserName(),
    mailSenderInfo.getPassword());
// 根据邮件会话属性和密码验证器构造一个发送邮件的 session
Session session = Session.getDefaultInstance(props, authenticator);

Message msg = new MimeMessage(session);
msg.setSubject("主题-你猜猜? ");
StringBuilder builder = new StringBuilder();
builder.append("测试邮件: 我用 Java 代码给你发送了一份邮件! 我的♥你收到了吗? ");
msg.setText(builder.toString());
msg.setFrom(new InternetAddress(mailSenderInfo.getUserName()));
Address to = new InternetAddress(mailSenderInfo.getToAddress());
// Message.RecipientType.TO 属性表示接收者的类型为 TO
msg.setRecipient(Message.RecipientType.TO, to);

Transport.send(msg);
/*Transport transport = session.getTransport("smtp");
transport.connect(mailSenderInfo.getMailServerHost(), mailSenderInfo.getUserName(),
    "irfydcgrkxembii");

transport.sendMessage(msg, new Address[] { new InternetAddress("*****@qq.com") });
transport.close();*/
}
}

```

## Smtplib 对接代码 demo 参考



java、php、python群发smtp对接demo代码.rar

U-Mail 邮件群发