# **达观数据语义理解和自动标签平台软件V1.0**

# **概述**

达观数据为应用方提供了功能强大、使用灵活的文本挖掘服务，通过简单易用的HTTP接口，帮助应用方有效分析大量的文本数据，提高应用方的处理效率。  
达观文本挖掘系统提供四种文本挖掘服务：  
1. 文本标签自动提取：从文本数据中自动提取出较为重要的关键词标签。  
2. 文本自动归类：自动判别文章所属类别，并给出相应的置信度，比如判断一篇文章是属于娱乐八卦、时事政治还是数码科技等。  
3. 文本自动审核：自动判别文章内容是否政治违规或者是否情色违规，同时给出违规的严重程度。  
4. 垃圾评论自动过滤：自动判断用户评论数据是否为垃圾评论，并给出置信度。

# **注意事项**

您需要完成以下步骤后，才可调用达观数据相关服务。  
1. 如您还没有开通达观账号，请先开通，账号开通请参考文档：[Step 1：开通账号](http://doc.datagrand.com/quick-start/step1)。  
2. 获取您的appid 及appname，请参考文档：[获取AppId及AppName](http://doc.datagrand.com/quick-start/step1" \l "%E8%8E%B7%E5%8F%96AppId%E3%80%81AppName)。  
3. 设置IP白名单，请参考文档：[设置IP白名单](http://doc.datagrand.com/quick-start/step1" \l "%E8%AE%BE%E7%BD%AEIP%E7%99%BD%E5%90%8D%E5%8D%95)。

# **数据统计和结果反馈**

数据统计和结果反馈目前仅限于文本自动归类、文本自动审核和垃圾评论自动过滤三种服务中。  
在上述操作正确执行，服务运行正常情况下，可以登录到[大数据平台](https://data.datagrand.com/signup/" \l "/login)查看统计数据。  


同时，用户可在系统后台查看到并且进行结果反馈操作。  


结果反馈请点击右侧的操作按钮。反馈结果会上传系统用于模型训练和效果提升。  


如需将反馈结果实时响应给应用方，可以添加回调地址。回调地址是应用方提供的，接受消息推送的地址。添加回调地址之后，一旦用户在系统平台提交结果反馈，反馈结果会通过POST方式的HTTP请求，推送到用户填的回调地址，用户可以根据推送内容去进行相关处理。反馈结果格式见各服务模块。  


# **接口列表**

## **文本标签自动提取**

### 接口描述

从文本数据中自动提取出较为重要的、标识文本核心内容的关键词标签。

### URL

http://taggingapi.datagrand.com/tagging/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要提标签的文本数据 |
| title | string | 否 | 要提标签的文本标题 |
| textid | string | 否 | 标记该文本的唯一ID |

### Http 返回结果说明

| **字段** | **类型** | **描述** |  |
| --- | --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |  |
| tag\_list | string | 提标签结果，为一个列表的json字符串，列表每一项是一个文本标签和权重，权重取值范围是0~1000 |  |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |  |
| errors | string | 错误信息 |  |

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&text=AlphaGo首战胜李世石，科学家：机器人超过人类很正常&title=围棋比赛&textid=18429488533' 'http://taggingapi.datagrand.com/tagging/meituan' |

成功返回示例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | {  "status": "OK",  "tag\_list": [{  "tag": "围棋比赛",  "weight": 1000  }, {  "tag": "alphago",  "weight": 233  }, {  "tag": "李世石",  "weight": 156  }, {  "tag": "机器人",  "weight": 145  }, {  "tag": "科学家",  "weight": 113  }, {  "tag": "人类",  "weight": 103  }, {  "tag": "战胜",  "weight": 95  }, {  "tag": "正常",  "weight": 82  }, {  "tag": "超过",  "weight": 63  }, {  "tag": "很",  "weight": 11  }],  "request\_id": "148642065805100373587",  } |

错误返回示例：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 1012,  "message": "text not found"  },  "request\_id": "1422348739084222300234072"  } |

警告返回:

## **文本自动归类**

### 接口描述

自动理解文本内容并判定所属类别，给出相应的置信度，比如判断一篇文章是属于娱乐八卦、时事政治还是数码科技等。

### URL

http://classifyapi.datagrand.com/classify/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要进行归类的文本数据 |
| title | string | 否 | 要进行归类的文本标题 |
| textid | string | 否 | 标记该文本的唯一ID |

### Http 返回结果说明

| **字段** | **类型** | **描述** |
| --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |
| result | string | 归类结果，为一个列表，列表的每一项是类别名称和置信度的对，形如 [(“新闻”, 0.88), (“娱乐”, 0.22)]。置信度是0到1之间的浮点数，值越大代表置信度越高。分类结果按照置信度从高到低排序。 |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |
| errors | string | 错误信息 |

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&text=何润东夫妇婚后首度甜蜜同行&textid=1885469469545' 'http://classifyapi.datagrand.com/classify/meituan' |

成功返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | {  "status": "OK",  "result": [  ["娱乐", 0.7328],  ["时事", 0.1900],  ["科技", 0.0012]  ],  "request\_id": "148642065805100373587",  } |

错误返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 1012,  "message": "text not found"  },  "request\_id": "1422348739084222300234072"  } |

警告返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | {  "status": "WARN",  "result": [  ["娱乐", 0.7328],  ["时事", 0.1900],  ["科技", 0.0012]  ],  "errors": {  "code": 8012,  "message": "some warning information"  },  "request\_id": "1422348739084222300234072"  } |

### 结果反馈推送POST请求格式

| **参数** | **类型** | **描述** |
| --- | --- | --- |
| textid | string | 标记该文本的唯一ID |
| feedback\_info | string | 反馈结果。json格式。例：{"is\_cate\_entertainment":1, "is\_cate\_ sports ":1}。其中1表示是文本属于该分类（此处为娱乐分类） |

key的形式如is\_cate\_【类别】。key包括如下：

社会: is\_cate\_society  
财经: is\_cate\_finance  
健康: is\_cate\_health  
美食: is\_cate\_food\_num  
时尚: is\_cate\_fashion  
教育: is\_cate\_education  
育儿: is\_cate\_baby  
房产家居: is\_cate\_house  
文化生活: is\_cate\_culture  
旅游摄影: is\_cate\_travel  
搞笑: is\_cate\_humor  
情感: is\_cate\_emotion  
娱乐: is\_cate\_entertainment  
科技: is\_cate\_science  
数码: is\_cate\_digital  
互联网: is\_cate\_internet  
游戏: is\_cate\_games  
汽车: is\_cate\_automobile  
体育运动: is\_cate\_sports |

## **文本自动审核（政治、色情等）**

### 接口描述

自动判别文章内容是否政治违规和是否情色违规，同时给出违规的严重程度。

### URL

http://auditapi.datagrand.com/audit/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要进行审核的文本数据 |
| title | string | 否 | 要进行审核的文本标题 |
| type\_list | string | 是 | 为一个列表的json字符串，列表每一项是要做审核的类型代号，“politic”为政治类审核，“porn”为情色类审核，列表不能为空 |
| textid | string | 否 | 标记该文本的唯一ID |
| suspect | int | 否 | 控制是否返回各审核类型的嫌疑文本，适合长文本数据，值为1表示打开该功能，审核分值高于0.4会返回该审核类型的嫌疑文本。 |
| userid | string | 否 | 登录用户ID |
| cid | string | 否 | 未登录用户ID。pc用户是cookieid，安卓APP 是imei ，苹果APP是UDID |
| mac | string | 否 | 用户的mac地址 |
| user | string | 否 | 用户名称或昵称 |
| ip | string | 否 | 文本提交IP |

### Http 返回结果说明

| **参数** | **类型** | **描述** |
| --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |
| result | string | 审核结果，为一个字典，字典的键（key）是审核类型代号，值（value）是违规严重程度（0到1之间的浮点数），值越大代表违规程度越严重。 |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |
| suspect | string | 嫌疑文本，请求参数添加suspect=1才会返回该结果。为一个字典，字典的键（key）是审核类型，值是该审核类型对应的嫌疑文本。 |
| errors | string | 错误信息 |

说明：  
(1) 如果调用时type\_list里有“politic”，返回值的result将包含“politic”（涉政程度）和“reaction”（政治违规程度）两个key，返回值的“politic”表示文本与政治相关的程度有多少，不表示文本是否政治违规，需要知道文本的政治违规嫌疑程度有多大，请使用“reaction”。  
(2) 审核分值解读：  
涉黄（porn）：正常（0-0.5），轻微涉黄（0.5-0.75）， 涉黄（0.75-1.0）  
涉政（politic）：正常（0 – 0.5），轻微涉政（0.5 – 0.75），涉政（ 0.75 – 1.0）  
反动（reaction）：正常（0 – 0.5），轻微反动（0.5 – 0.75），反动（ 0.75 – 1.0）

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&type\_list=["politic","porn"]&text=毛新宇：敌对势力借网络攻击毛泽东思想&textid=165930459549503953&suspect=1' 'http://auditapi.datagrand.com/audit/meituan' |

成功返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | {      "status": "OK",      "suspect": {          "reaction\_text": "",          "porn\_text": "",          "politic\_text": "毛新宇敌对势力借网络攻击毛泽东思想"      },      "result": {          "politic": 0.963,          "reaction": 0.2494,          "porn": 0.0852      },      "request\_id": "1516074215308639"  } |

错误返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 1012,  "message": "text not found"  },  "request\_id": "1422348739084222300234072"  } |

警告返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | {  "status": "WARN",  "result": {  "politic": 0.723,  "porn": 0.088,  "reaction": 0.00  },  "errors": {  "code": 8012,  "message": "illegal type"  },  "request\_id": "1422348739084222300234072"  } |

### 结果反馈推送POST请求格式

| **参数** | **类型** | **描述** |
| --- | --- | --- |
| textid | string | 标记该文本的唯一ID |
| feedback\_info | string | 反馈结果。json格式。示例：{"is\_reaction":0,"is\_porn":null,"is\_politic":1}其中1表示是涉黄/涉政/反动，0表示正常，null表示未标记。 |

## **垃圾评论自动过滤**

### 接口描述

自动判断用户评论数据是否为垃圾评论，并给出置信度。

### URL

http://commentapi.datagrand.com/bad\_comment/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要进行垃圾评论判断的文本数据 |
| title | string | 否 | 评论的标题 |
| textid | string | 否 | 标记该文本的唯一ID |
| userid | string | 否 | 发表评论的登录用户ID |
| user\_type | string | 否 | 用户类型，包括”common”（通用）和”zhubo”（主播）。默认是”common”。 |
| cid | string | 否 | 未登录用户ID。pc用户是cookieid，安卓APP 是imei ，苹果APP是UDID |
| mac | string | 否 | 用户的mac地址 |
| user | string | 否 | 用户名称或昵称 |
| ip | string | 否 | 评论提交IP |
| type\_list | string | 否 | 垃圾评论识别种类，包括广告(“ad”)和低质量文本(“nonsense”)，该字段为json格式，形如[“ad”,”nonsense”] 或 [“ad”]，默认为[“ad”]。若指定该参数，则返回结果只包含指定类型的识别结果。 |

### Http 返回结果说明

| **字段** | **类型** | **描述** |
| --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |
| score | string | 判定为垃圾评论的置信度（0到1之间的浮点数），值越大代表是垃圾评论的可能性越大。 |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |
| errors | string | 错误信息 |

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&text=3m每天百分之1利息，60元起步，有需要可以联系我，Q49663537，或者关注百度贴吧，老马平台吧！&title=顶顶顶&textid=435386945382932&user=巧儿' 'http://commentapi.datagrand.com/bad\_comment/meituan' |

成功返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | {  "status": "OK",  "weight\_ad": 0.6200, *//广告评论权重， 0到1之间。数值越大，该评论为广告评论的概率越大*  "is\_ad": 1, *//0为正常文本，1为广告评论*  "request\_id": "148642065805100373587",  } |

错误返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 1012,  "message": "text not found"  },  "request\_id": "1422348739084222300234072"  } |

警告返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | {  "status": "WARN",  "weight\_ad": 0.6200,  "is\_ad": 1,  "errors": {  "code": 8012,  "message": "some warning information"  },  "request\_id": "1422348739084222300234072"  } |

### 结果反馈推送POST请求格式

| **参数** | **类型** | **描述** |
| --- | --- | --- |
| textid | string | 标记该文本的唯一ID |
| feedback\_info | string | 反馈结果。json格式。示例：{"is\_ad":1}其中1表示是垃圾评论，0表示非垃圾评论，null表示未标记。 |

## **文本审核、评论过滤合并接口**

### 接口描述

一次调用同时返回文本审核和评论过滤的判别结果。

### URL

http://commentaggreapi.datagrand.com/commentaggre/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要进行评论判断的文本数据 |
| title | string | 否 | 评论的标题 |
| textid | string | 否 | 标记该文本的唯一ID |
| userid | string | 否 | 发表评论的登录用户ID |
| user\_type | string | 否 | 用户类型，包括”common”（通用）和”zhubo”（主播）。默认是”common”。 |
| cid | string | 否 | 未登录用户ID。pc用户是cookieid，安卓APP 是imei ，苹果APP是UDID |
| mac | string | 否 | 用户的mac地址 |
| user | string | 否 | 用户名称或昵称 |
| ip | string | 否 | 评论提交IP |

### Http 返回结果说明

| **字段** | **类型** | **描述** |
| --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |
| result | string | 返回结果，weight\_ad（广告指数）、is\_ad（是否广告）、reaction（反动指数）、 porn（涉黄指数）和politic（涉政指数）等字段含义见文章审核和垃圾评论的返回 |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |
| errors | string | 错误信息 |

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&text=3m每天百分之1利息，60元起步，有需要可以联系我，Q49663537，或者关注百度贴吧，老马平台吧！&title=顶顶顶&textid=435386945382932&user=巧儿' 'http://commentaggreapi.datagrand.com/commentaggre/datagrand' |

成功返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | {  "status": "OK",  "result": {  "is\_ad": 1, *//0为正常文本，1为广告评论*  "weight\_ad": 0.5818, *//广告评论权重*  "reaction": 0.0097, *//反动指数*  "porn": 0.0897, *//涉黄指数*  "politic": 0.0097 *//涉政指数*  },  "request\_id": "1470899110107282"  } |

错误返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 8011,  "message": "some warning information"  },  "request\_id": "1470900887957456"  } |

警告返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | {  "status": "WARN",  "result": {  "is\_ad": 1,  "weight\_ad": 0.5818,  "reaction": 0.0097,  "porn": 0.0897,  "politic": 0.0097  },  "errors": {  "code": 8012,  "message": "some warning information"  },  "request\_id": "1422348739084222300234072"  } |

## **文本情感分析**

### 接口描述

自动判断文本的情感倾向为正面或者负面，并给出情感倾向的程度。

### URL

http://sentimentapi.datagrand.com/sentiment/YOUR\_APP\_NAME

### Http Method

POST

### Http 返回格式

JSON

### Http 请求参数说明

| **参数** | **类型** | **是否必需** | **描述** |
| --- | --- | --- | --- |
| appid | int | 是 | 应用的id |
| text | string | 是 | 要进行归类的文本数据 |
| title | string | 否 | 要进行归类的文本标题 |
| textid | string | 否 | 标记该文本的唯一ID |

### Http 返回结果说明

| **字段** | **类型** | **描述** |
| --- | --- | --- |
| status | string | 执行结果，OK为成功，FAIL为失败，WARN为有部分非重要字段异常，请根据返回错误信息进行排查。 |
| result | string | 情感分析结果，为一个字典，字典的键（key）是”positive”(正面)和”negative”(负面)，值（value）是对应的情感分类的置信度（0到1之间的浮点数），值越大代表置信度越高。 |
| request\_id | string | 该条上报记录的序号，仅用于排查问题使用 |
| errors | string | 错误信息 |

### 示例

CURL调用示例：

|  |  |
| --- | --- |
| 1  2 | curl -i -X POST -d 'appid=12345&text=何润东夫妇婚后首度甜蜜同行&textid=1885469469545' 'http://sentimentapi.datagrand.com/sentiment/meituan' |

成功返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "OK",  "result": {  "positive": 0.772,  "negative": 0.228  },  "request\_id": "148642065805100373587",  } |

错误返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | {  "status": "FAIL",  "errors": {  "code": 1012,  "message": "text not found"  },  "request\_id": "1422348739084222300234072"  } |

警告返回示例:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | {  "status": "WARN",  "result": {  "positive": 0.772,  "negative": 0.228  },  "errors": {  "code": 8012,  "message": "some warning information"  },  "request\_id": "1422348739084222300234072"  } |

# **HTTP请求示例代码**

## **JAVA**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91 | **package** com.datagrand.datareport.test;    import java.io.IOException;  import java.util.ArrayList;  import java.util.HashMap;  import java.util.List;  import java.util.Map;  import java.util.Set;  import java.util.Map.Entry;    import org.apache.http.Header;  import org.apache.http.HttpEntity;  import org.apache.http.HttpResponse;  import org.apache.http.HttpStatus;  import org.apache.http.NameValuePair;  import org.apache.http.StatusLine;  import org.apache.http.client.ClientProtocolException;  import org.apache.http.client.HttpClient;  import org.apache.http.client.entity.GzipDecompressingEntity;  import org.apache.http.client.entity.UrlEncodedFormEntity;  import org.apache.http.client.methods.HttpPost;  import org.apache.http.impl.client.DefaultHttpClient;  import org.apache.http.message.BasicNameValuePair;  import org.apache.http.params.HttpConnectionParams;  import org.apache.http.params.HttpParams;  import org.apache.http.util.EntityUtils;    @SuppressWarnings("deprecation")  public **class** TestHttp{    @SuppressWarnings("resource")  public **String** post(**String** reqURL, Map<**String**, **String**> params)  throws ClientProtocolException, IOException{  **String** responseContent = "";    HttpPost httpPost = **new** HttpPost(reqURL);  **if** (params != **null**) {  List nvps = **new** ArrayList();  Set<Entry<**String**, **String**>> paramEntrys = params.entrySet();  **for** (Entry<**String**, **String**> entry : paramEntrys) {  nvps.add(**new** BasicNameValuePair(entry.getKey(), entry.getValue()));  }  httpPost.setEntity(**new** UrlEncodedFormEntity(nvps, "utf-8"));  }    httpPost.setHeader("User-Agent", "datagrand/datareport/java sdk v1.0.0");  httpPost.setHeader("Content-Type","application/x-www-form-urlencoded");    HttpClient httpClient = **new** DefaultHttpClient();  HttpParams httpParams = httpClient.getParams();  HttpConnectionParams.setSoTimeout(httpParams, 60\*1000);  HttpConnectionParams.setConnectionTimeout(httpParams, 60\*1000);    HttpResponse response = httpClient.execute(httpPost);  StatusLine status = response.getStatusLine();  **if** (status.getStatusCode() >= HttpStatus.SC\_MULTIPLE\_CHOICES) {  System.out.printf(  "Did not receive successful HTTP response: status code = {}, status message = {}",  status.getStatusCode(), status.getReasonPhrase());  httpPost.abort();  }    HttpEntity entity = response.getEntity();  **if** (entity != **null**) {  responseContent = EntityUtils.toString(entity, "utf-8");  EntityUtils.consume(entity);  } **else** {  System.out.printf("Http entity is null! request url is {},response status is {}", reqURL, response.getStatusLine());  }  **return** responseContent;  }    public static **void** main(**String**[] args){  TestHttp obj = **new** TestHttp();  Map<**String**,**String**> params = **new** HashMap<**String**,**String**>();  params.put("appid", "12345");  params.put("title", "顶顶顶");  params.put("textid", "435386945382932");  params.put("text", "3m每天百分之1利息，60元起步，有需要可以联系我，Q49663537，或者关注百度贴吧，老马平台吧！");    **String** res;  **try** {  res = obj.post("http://commentapi.datagrand.com/bad\_comment/meituan", params);  System.out.println(res);  } **catch** (Exception e) {    e.printStackTrace();  }  }  } |

## **PHP**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | <!--?php **function** reportItemsByCurl($url, $params = **array**()) { $options = **array**( CURLOPT\_HTTP\_VERSION => 'CURL\_HTTP\_VERSION\_1\_1',<br ?--> CURLOPT\_CONNECTTIMEOUT => 60,  CURLOPT\_TIMEOUT => 60,  CURLOPT\_CUSTOMREQUEST => 'POST',  CURLOPT\_HEADER => **false**,  CURLOPT\_RETURNTRANSFER => **true**,  CURLOPT\_HTTPHEADER => **array**('Expect:'),  CURLOPT\_POSTFIELDS => $params,  );  $session = curl\_init($url);  curl\_setopt\_array($session, $options);  $response = curl\_exec($session);  curl\_close($session);  **return** $response;  }    $YOUR\_APP\_ID = 12345;  $YOUR\_APP\_NAME = 'meituan';  $data = **array**(  'appid' => $YOUR\_APP\_ID,  'title' => '顶顶顶',  'textid' => '435386945382932',  'text' => '3m每天百分之1利息，60元起步，有需要可以联系我，Q49663537，或者关注百度贴吧，老马平台吧！',  );    $returnDG = reportItemsByCurl('http://commentapi.datagrand.com/bad\_comment/' . $YOUR\_APP\_NAME, $data);  echo $returnDG; |