有道云笔记 Open API 说明文档

版本	作者	概要	日期
0.1	李崇欣	起始版本，网易内部 API 说明	2011-9-13
0.2	李崇欣	反馈修订并添加 API 使用示例	2011-9-16
0.3	陈超	修改安全机制模块，添加 OAuth 认证	2011-11-17
0.4	李崇欣	API 接口修改	2011-12-7
0.4.4	李崇欣	添加 User 接口及每个应用的默认笔记本	2012-5-29
1.0	李崇欣	文档整理及接口更新	2012-7-27
1.1	李猛	增加 OAuth2.0 授权的文档	2013-5-28

概述

有道云笔记旨在以云存储技术帮助用户建立一个可以轻松访问、安全存储的云笔记空间，解决了个人资料和信息跨平台跨地点的管理问题，目前已经提供了桌面版、网页版以及部分手机型号的版本。但是用户对笔记的需求不仅仅局限于这几种情形，对于很多非笔记应用，用户仍然可能有跨平台跨设备的存储需求。通过开放的 API，第三方应用只需要进行简单的开发，就可以通过标准的 web 协议对有道云笔记的数据进行安全的访问与修改，而不需要搭设和维护运存储服务，这大大降低了第三方的开发与运营成本，从而将更多的精力专注于应用本身。

本文档从技术角度对有道云笔记的开放 API 进行说明，从而方便开发者的理解和使用。

数据模型

目前有道云笔记的数据模型包括用户、笔记本、笔记以及附件四部分，下面对它们分别进行说明：

用户

每一个有道云笔记的账号（目前我们主要使用网易通行证作为账号，但是同时也支持用户使用新浪微博账号进行登录，以后也将允许用户使用 QQ 账号进行登录）都在系统中对应着一份用户信息以及相应的配置，包括该用户邮箱、笔记总空间大小、当前使用空间大小、默认笔记本、在线时间以及注册时间、上次修改时间等一系列信息。

笔记本

用户通过笔记本对所有笔记进行组织管理，每一篇笔记必须属于某一个笔记本，同时为了方便用户的区分与定位，每个笔记本也必须具有唯一的名字。目前有道云笔记已经支持将多个笔记本归类为一个笔记组，但是尚未对 OpenAPI 开放。

对于同一篇笔记，可以在两个笔记本之间进行转移，但是却不允许出现在两个笔记本中。此外，每个用户在任何时候都有一个笔记本被标记为默认笔记本，如果一个笔记在创建时没有特别指明笔记本，它将被放入默认笔记本中。删除一

个笔记本将删除该笔记本下的所有笔记。

虽然有道云笔记的客户端同时提供了同步笔记本与本地笔记本两种类型，但是由于本地笔记本仅存储在用户的本机而没有上传至服务器，因此 OpenAPI 只能创建或者访问用户同步笔记本中的笔记。

笔记

一篇笔记由一段超文本以及相关联的附件（例如图片、文本、PPT、PDF 等）所组成，其中超文本的内容按照一种标记语言的格式进行存储，每个附件所在的位置都对应着一个 tag，包含着该附件路径，从而可以根据该信息找到对应附件的数据。同时，每个笔记还包含一些属性信息，例如题目、作者、来源 URL 以及创建修改时间等，这些信息可以用来对笔记进行查找、过滤以及排序。

当一个笔记被删除时，它将被放入回收站，回收站中的笔记在一定的时间内仍然可以访问（Open API 不可以访问删除的笔记），但是不可以进行编辑，如果没有进行恢复，回收站的笔记将在一定的时间后（目前为两个月）被清除。

笔记标记语言格式的详细说明见附录 A。

附件

笔记附件是附属于笔记的二进制数据，无法脱离笔记单独存在，因此如果仅上传附件而不添加相应的笔记，则该附件会被定期的空间回收所删除。笔记附件类似于邮件的附件，但是与邮件的附件不同的是，笔记的附件没有与笔记的内容进行整体编码，而只是在笔记内容超文本的对应位置放置了一个 tag 对其进行引用，附件的数据则单独进行存储。因此如果将附件 tag 在两个笔记之间进行拷贝，这两个笔记将共享同一个附件，而对附件的修改也将同时反映在两个笔记之中。不过目前我们尚不提供附件修改的 API，每次上传的附件都是一个全新的附件，因此还不存在这样的问题。随着 API 的完善，以后将提供类似的 API，因此如果第三方应用希望避免这种情况，需要自行对附件拷贝进行处理。另一方面，如果共享附件的两篇笔记中有一篇笔记被删除，不会对另外一篇笔记及其附件造成影响。

目前笔记的附件支持除了 exe、com、cmd、bat、sys 以外的所有文件格式，大小限制为 100MB。

授权机制

为了对用户的数据进行保护，大部分 API 的访问都需要获得用户的身份信息，我们采用 OAuth 的方式对第三方应用进行授权，只有经过用户授权的应用才可以对用户的数据进行访问。目前我们支持的 OAuth 版本为 1.0a 和 OAuth 2.0，推荐用户使用 OAuth2.0 授权的方式，该方式更简洁也更安全。

申请 Consumer Key

对于所有需要使用 Open API 的应用，在使用 Open API 之前首先要在系统内申请一个应用程序，申请者需要提交申请人姓名、email、应用名称、应用主页、callback URL 以及是否限制 callbackURL 等信息，我们会生成相应的 Consumer Key 和 Consumer Secret，作为每个应用的身份标识。应用名称和 Consumer Key 在系统中均是唯一的，在进行请求时，第三方应用需要将 Consumer Key 等验证信息加入请求中，而系统收到请求后，便可以根据 Consumer Key 和 Consumer Secret 对请求进行验证与记录，具体请求方式参见根据 OAuth 版本的不同有所不同，请分别参见 OAuth1.0a 授权说明和 OAuth2.0 授权说明。

OAuth 1.0a 请求授权

授权流程

在获取 Consumer Key 和 Consumer Secret 后，应用程序便可以通过 OAuth 授权访问用户数据，一个完整的 OAuth 授权流程如下图所示：

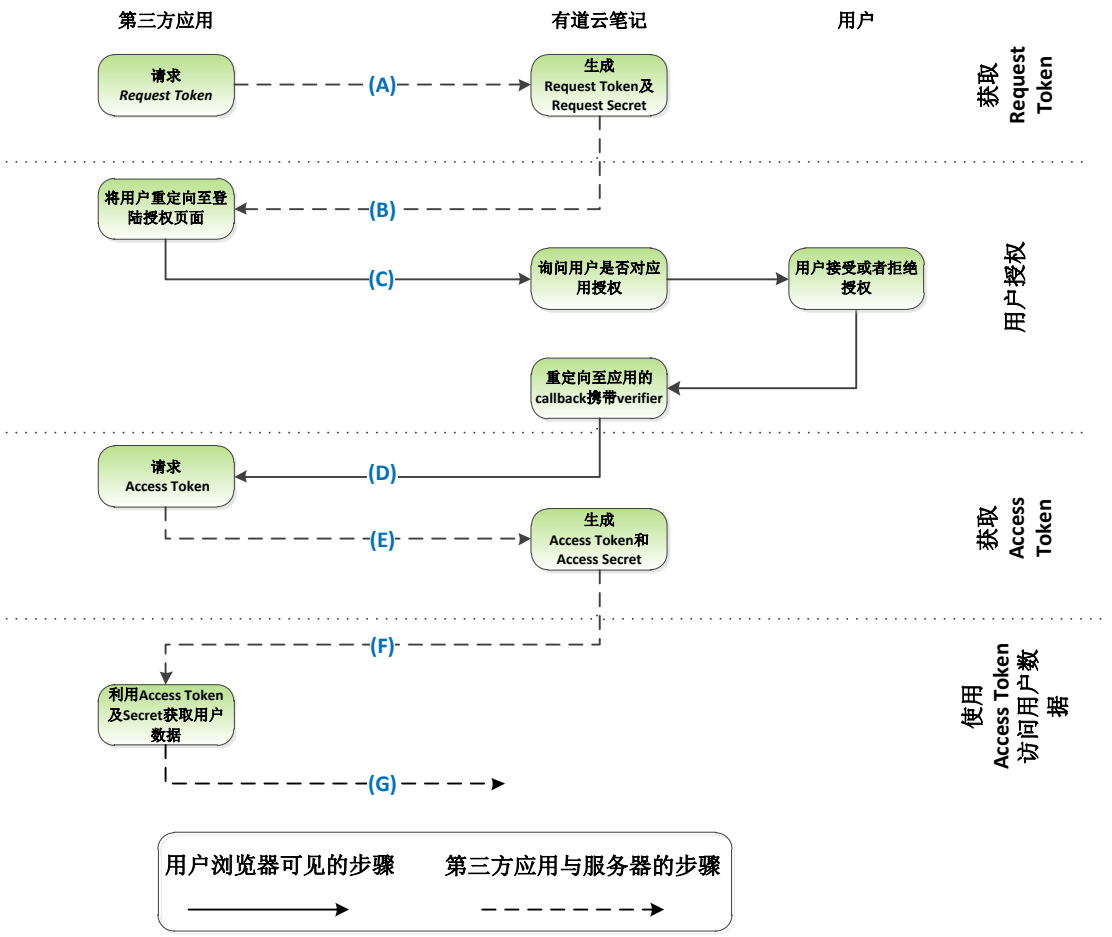

图 **1** OAuth 授权流程

应用程序首先利用 Consumer Key 与 Consumer Secret 从有道云笔记 Server 端获取 Request Token 及 Request Secret。

获取 Request Token 及 Secret 后，应用程序便可以将用户重定向至有道云笔记的用户登录授权页面，此时用户登录并选择接受或者拒绝对应用程序的授权，如果用户选择接受授权有道云笔记会重定向至应用程序提供的 callback 页面，并且将 verifier 作为参数一同传递过去；如果用户选择拒绝授权则终止整个授权过程。

应用程序利用 Request Token 以及刚刚得到的 verifier 应用程序便可以请求 Access Token，而有道云笔记在验证过 Request Token 及 verifier 后便会生成 Access Token 及 Secret 返回给应用程序，之后应用程序便可以使用 Access Token 及 Secret 访问用户的数据。通常 Access Token 具有一定的有效期，在有效期内应用程序可以一直访问该用户的数据，而当 Access Token 过期后，应用程序则需要再次走一遍授权流程。

在上述过程中，如果发生了任何异常，有道云笔记都会返回对应的错误码及错误信息，详细的错误信息说明见附录 B。

授权请求说明

在进行 OAuth 授权请求时，OAuth 相关的参数可以放置在 query string 和 http header 中，这里我们推荐使用 http header 的方式，这样可以将 OAuth 相关的参数与接口非 OAuth 参数区别对待，其中 Header name 为 **Authorization**，而 Header value 则为 OAuth 相关参数组成的值，格式如下：

OAuth[空格][parameterName=”value”],[空格][parameterName=”value”]…

此外在发送时还有以下几点需要注意：

1. parameterName 和 value 必须进行 Percent Encoding (做法为先进行 URL Encoding 后再要将”+”替换为”%20”,将”*”替换为”%2A”，将”%7E”替换为”~”)
2. value 的引号为必须
3. 发送请求时参数可以是无序的，但是在签名时这些参数有一定的顺序要求，详见附录 D
4. 请求的 baseURL 为有道云笔记的 domain，对于线上环境该 baseURL 为 note.youdao.com，而测试环境为 notesandbox.youdao.com，下同

一个包含 OAuth 参数的 Header 示例如下所示：

```
Authorization: OAuth oauth_token="64e4f0c25029dd6687ea74cd64e9640a",
oauth_consumer_key="2456f9dd37e162ffe237c8b88739925f",
oauth_signature_method="HMAC-SHA1", oauth_timestamp="1343381467",
oauth_nonce="952606576956129", oauth_version="1.0",
oauth_signature="%2FLQhn0OCf3lCMPSAAMVW35aUka0%3D"
```

获取笔记服务器时间

在 OAuth 的认证流程中需要传入参数 oauth_timestamp，通过该参数来验证请求是否超时，若用第三方应用的本地时间，有可能出现时间不一致的情况导致认证失败，所以提供接口来获取到当前笔记服务器的时间。

- **URL：**http://[baseURL]/oauth/time
- **请求方式：**GET
- **返回结果：**服务器的时间，以 json 的形式返回，示例如下：

```
{"unit":"second","oauth_timestamp":1368609463}
```

请求 Request Token

- **URL：**http://[baseURL]/oauth/request_token
- **请求方式：**GET
- **OAuth 参数：**

参数名	参数说明
oauth_callback	回调 url，该 url 可以为 oob,代表不回调，另外 url 还可以跟其他参数
oauth_consumer_key	申请应用时拿到的 Consumer Key
oauth_signature_method	签名方法，支持 HMAC-SHA1
oauth_timestamp	时间戳，当前时间，单位秒
oauth_nonce	随机串，为了防止重放攻击，5 分钟内同一用户同一应用同一时间发来的请求中 oauth_nonce 应不同
oauth_version	1.0
oauth_signature	签名，使用 Consumer Secret +'&'作为签名的 key， 而签名的内容是除 oath_signature 以外的请求 url 内容，计算签名完毕后，使用 base64 编码，具体签名方法见附录 D

- **返回结果：**请求成功时返回 oauth_token 和 oauth_token_secret，即 Request Token 与 Secret，结果示例：

```
oauth_token=8b1676312a97aaa762ad743853a3c814&oauth_token_secret=71eb91f4d40b896c0aed9820cb240f76
```

请求失败时返回 http status code 500 以及 error message

请求用户登录授权

- **URL：** http://[baseURL]/oauth/authorize
- **请求方式：**GET
- **OAuth 参数：**（该处的参数必须以 query string 的形式发送）

参数名	参数说明
oauth_token	请求 request_token 时返回的 oauth_token

- **返回结果：**当用户成功登录并且选择接受授权后，返回 oauth_token 与 oauth_verifier，否则返回出错信息
- **备注：**

1. 应用将用户定向至授权页面时，如果已经携带了网易或者有道的 cookie 信息，则将直接询问用户是否授权，否则会重新定向到网易的通行证页面先进行登录。
2. 如果 oauth_callback 为 oob，则不会产生回调，用户会在有道云笔记的授权页面上看到授权码，该授权码即为授权后的 verifier，用户需要手工将该授权码黏贴至应用程序中完成 OAuth 授权程序。

请求 Access Token

- **URL：**http://[baseURL]/oauth/access_token
- **请求方式：**GET
- **OAuth 参数：**

参数名	参数说明
oauth_consumer_key	第三方应用的 Consumer Key
oauth_token	请求 request_token 时返回的 oauth_token
oauth_verifier	用户授权后得到的 oauth_verifier，或者是授权页面显示的授权码
oauth_signature_method	签名方法，支持 HMAC-SHA1
oauth_timestamp	时间戳，当前时间，单位毫秒
oauth_nonce	随机串，为了防止重放攻击，5 分钟内同一用户同一应用同一时间发来的请求中 oauth_nonce 应不同
oauth_version	1.0
oauth_signature	使用 consumerSecret + '&' + oauth_token_secret（该 secret 为请求 request_token 时返回的 oauth_token_secret，即 Request Token Secret）字串作为 key，而签名的内容是除 oauth_signature 以外的请求 url 内容，计算签名完毕后，使用 base64 编码，具体签名方法见附录 D

- **返回结果：**请求成功时返回 oauth_token 和 oauth_token_secret，即为 Access Token 与 Secret，结果示例：

```
oauth_token=f7063a40e0c480c99930131526d5d31c&oauth_token_secret=71eb91f4d40b896c0aed9820cb240f76
```

请求失败时返回 http status code 500 以及 error message

访问 OpenAPI

在获得 Access Token 及 Secret 后，应用程序便可以通过 OpenAPI 访问、更新用户的数据，在进行 OpenAPI 的请求时，应用程序除了需要添加 OpenAPI 接口所需要的参数，仍然需要在 Header 中添加 OAuth 相关的参数以便对请求来源进行校验。

- **OAuth 参数：**

参数名	参数说明
oauth_consumer_key	第三方应用的 Consumer Key
oauth_token	请求 access_token 时返回的 oauth_token
oauth_signature_method	签名方法，支持 HMAC-SHA1
oauth_timestamp	时间戳，当前时间，单位毫秒
oauth_nonce	随机串，为了防止重放攻击，5 分钟内同一用户同一应用同一时间发来的请求中 oauth_nonce 应不同
oauth_version	1.0
oauth_signature	使用 consumerSecret + '&' + oauth_token_secret (该 secret 为请求 access_token 时返回的 oauth_token_secret，即 Access Token Secret)字串作为 key，而签名的内容是除 oauth_signature 以外的请求 url 以及参数的内容，计算签名完毕后，使用 base64 编码，具体签名方法见附录 D

需要注意的是，在进行 OpenAPI 请求时，签名的内容根据 OpenAPI 接口的不同而有所不同：

1. 对于请求方式为 GET 的接口，在计算签名时除了 OAuth 本身的参数以及 OpenAPI 的 URL，例如获取用户信息接口，还需要将该接口 query string 中的其他所有非 OAuth 参数加入到 base string 中一起进行签名
2. 对于请求方式为 POST 且 Content Type 为 application/x-www-form-urlencoded 的接口，例如创建笔记本接口，在计算签名时除了 OAuth 本身的参数以及 OpenAPI 的 URL，还需要将该接口 content body 中的其他所有非 OAuth 参数加入到 base string 中一起进行签名，例如笔记本 name
3. 对于请求方式为 POST 且 Content Type 为 multipart/form-data 的接口，例如创建笔记接口，在计算签名时只需要加入 OAuth 本身的参数及 OpenAPI 的 URL，而不需要考虑 content body 中的参数

具体 API 的请求方式以及 Content Type 见下文关于 OpenAPI 的详细说明。

目前已经有一些 OAuth 相关的类库提供了计算签名以及获取 Request Token 和 Access Token 的方法，同时我们也提供了一些常见语言的 SDK 以便应用开发者可以快速的开发，详见附录 C。

OAuth 2.0 请求授权

OAuth 2.0 说明

OAuth 2.0 较 1.0 整个授权验证流程更简单更安全，是未来主要的用户身份验证和授权方式，因此有道云笔记开放平台对这种授权方式进行了支持，并推荐开发者采用这一协议，在 OAuth 2.0 的授权及使用流程中都需要使用 https 协议。

授权流程

第三方应用在获取到 consumerKey 和 consumerSecret 之后，便可以通过 OAuth 2.0 授权访问用户数据，完整的 OAuth2.0 授权流程如下图所示：

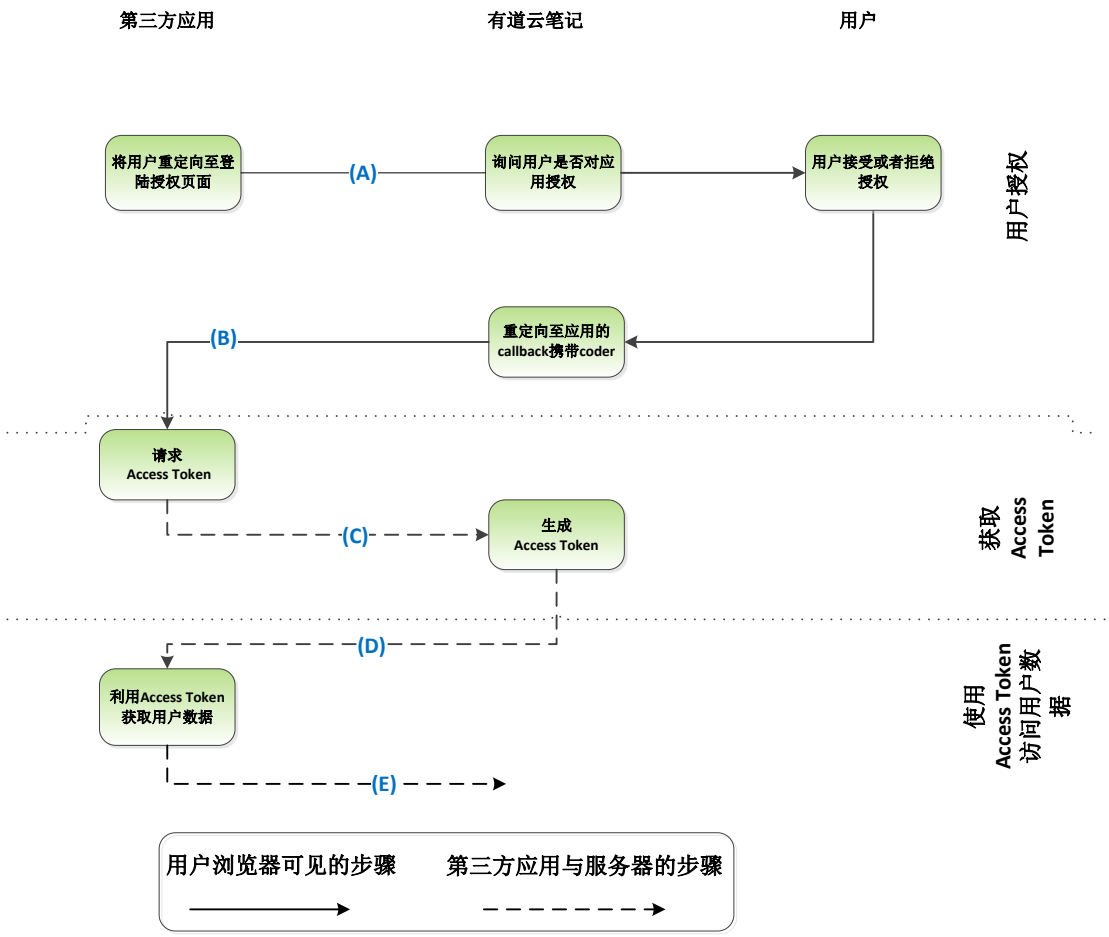

图 **2** OAuth 2.0 授权流程

应用程序首先利用 Consumer Key 请求用户授权，便可以将用户重定向至有道云笔记的用户登录授权页面，此时用户登录并选择接受或者拒绝对应用程序的授权，如果用户选择接受授权有道云笔记会重定向至应用程序提供的 callback 页面，并且将 code 作为参数一同传递过去；如果用户选择拒绝授权则终止整个授权过程。

利用刚刚得到的 code 应用程序便可以请求 Access Token，而有道云笔记在验证过 code 后便会生成 Access Token 返回给应用程序，之后应用程序便可以使用 Access Token 访问用户的数据。

在上述过程中，如果发生了任何异常，有道云笔记都会返回对应的错误码及错误信息，OAuth2.0 相关的错误信息说明见附录 E。

请求用户登录授权

- **URL：** https://[baseURL]/oauth/authorize2

- **请求方式：**GET
- **OAuth 参数：**（该处的参数必须以 query string 的形式发送，以下参数都为必传）

参数名	参数说明
client_id	申请的 consumerKey
response_type	当前只支持 code
redirect_uri	授权成功之后的跳转地址，在申请 consumerKey 的时候会要求填入 domains，只要该参数值属于所填的 domains（没有 domain 的时候采用 home page）中的任意一个即可。如果该应用没有 domain 信息，则该参数需要填固定的 http://note.youdao.com/redirect 网址才能通过认证。
state	该参数必填，并且会回传该参数。
display	默认会根据 userAgent 进行自动适配，可选值 web/mobile，用于适配 pc 或 mobile 设备

- **返回结果：**当用户成功登录并且选择接受授权后，通过 redirect_uri 的方式将 code 返回，否则返回出错信息。

备注：

请求成功后的跳转 url 格式如下：

http://note.youdao.com/test?state=state&code=060148d3e406a80ac5f89e28f4ac633a

请求 Access Token

- **URL：** https://[baseURL]/oauth/access2
- **请求方式：**GET
- **OAuth 参数：**（该处的参数必须以 query string 的形式发送，以下参数都必填）

参数名	参数说明
client_id	申请的 consumerKey
client_secret	对应的 consumerSecret
grant_type	当前只支持 authorization_code
redirect_uri	填入的规范和上面授权接口中的规则一致。
code	授权成功取得到的 code

- **返回结果：**调用成功返回 access token，否则返回出错信息

备注：

正常返回结果的格式如下：
{"accessToken":"b43de33483491bc86090d6a707d6c3d3"}

OAuth1.0a 的 AccessToken 置换成 OAuth2.0 的 AccessToken

- **URL：** https://[baseURL]/oauth/replace
- **请求方式：**GET
- **OAuth 参数：**（该处的参数必须以 query string 的形式发送，以下参数都必填）

参数名	参数说明
client_id	申请的 consumerKey
client_ secret	对应的 consumerSecret
token	OAuth1.0a 中授权成功后获取到的 token
token_secret	OAuth1.0a 中授权成功后获取到的 token secret

- 返回结果：调用成功返回 access token，否则返回出错信息

备注：
该接口在调用成功之后会将对应的 OAuth1.0a 的 token 置为失效。
正常返回结果的格式如下：
{"accessToken":"b43de33483491bc86090d6a707d6c3d3"}

访问 OpenAPI

对于普通的 post 和 get 请求，只需要增加 oauth_token 参数即可，参数值为去得到的 accessToken。对于 Content-Type 为 multipart/form-data 的 post 请求，需要在访问中增加一个名为 Authorization 的 header，格式为：OAuth oauth_token="%accessToken%"，其中%accessToken%即为取到的 accessToken。
具体 API 的请求方式以及 Content Type 见下文关于 OpenAPI 的详细说明。用 OAuth2.0 协议访问 OpenAPI 时需要使用 https 协议。

用户操作 API

查看用户信息

- **URL：**https://[baseURL]/yws/open/user/get.json
- **请求方式：**GET
- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**无
- **返回结果：**操作成功时 http 状态为 200，并返回用户信息；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

  ```
  {
    "user" : "test@163.com",          // 用户 ID
    "total_size" : "10293736383",     // 用户总的空间大小，单位字节
    "used_size" : "1024"              // 用户已经使用了的空间大小，单位字节
    "register_time" : "1323310917650"       // 用户注册时间，单位毫秒
    "last_login_time" : "1323310949120"     // 用户最后登录时间，单位毫秒
    "last_modify_time" : "1323310978120"    // 用户最后修改时间，单位毫秒
    "default_notebook" : "/AB384D734"       // 该应用的默认笔记本
  }
  ```

  注：对于每个第三方应用，都分别在用户的笔记空间中对应一个默认笔记本，笔记本名可以在申请应用时指定，如果不指定则默认为“来自<应用名称>”，这样在使用 OpenAPI 创建笔记时，如果第三方应用不指定笔记本，则自动创建在该默认笔记本中。

笔记本操作 API

查看用户全部笔记本

- **URL：**https://[baseURL]/yws/open/notebook/all.json
- **请求方式：**POST
- **Content-Type：**application/x-www-form-urlencoded

- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**无
- **返回结果：**操作成功时 http 状态为 200，并返回笔记本列表，第一个笔记本为该应用的默认笔记本；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

```
[
  {“path” : “/4AF64012E9864C”,          // 笔记本的路径
   “name” : “笔记本 1”,                 // 笔记本的名称
   “notes_num” : “3”                    // 该笔记本中笔记的数目
   “create_time” : “1323310917”         // 笔记本的创建时间，单位秒
   “modify_time” : “1323310949”         // 笔记本的最后修改时间，单位秒
  },
  {“path” : “/5AB0C6B33BD1162”,
   “name” : “笔记本 2”,
   “notes_num” : “10”
   “create_time” : “1323311244”
   “modify_time”: “1323310349”
  }
]
```

列出笔记本下的笔记

- **URL：**https://[baseURL]/yws/open/notebook/list.json
- **请求方式：**POST
- **Content-Type：**application/x-www-form-urlencoded
- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
notebook	笔记本路径	是	String

- **返回结果：**操作成功时 http 状态为 200，并返回该笔记本下的笔记列表（只有笔记路径，笔记内容需要通过笔记接口获取）；失败时 http 状态 500 并返回错误码和错误信息，详见附录 B。

结果示例：

[

“/4AF64012E9864C/123TDE3DC”,　　// 笔记路径

“/4AF64012E9864C/DC12FTDE3”,

“/4AF64012E9864C/FE23TDE3C”

]

创建笔记本

- **URL：** https://[baseURL]/yws/open/notebook/create.json
- **请求方式：** POST
- **Content-Type：** application/x-www-form-urlencoded
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
name	笔记本名称	是	String
create_time	创建时间，单位为秒，如果不指定则使用系统时间	否	Long

- **返回结果：** 操作成功时 http 状态 200，并返回新创建笔记本的路径；失败时 http 状态 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

  {“path” : “/4AF64012E9864C”}

注：对于创建时间，主要作用在于对移动端的应用，笔记本的创建时间和同步时间可能差别较大，因此如果不指定该时间，笔记本的创建时间则会显示为同步时间而非本身的创建时间。

删除笔记本

- **URL：** https://[baseURL]/yws/open/notebook/delete.json
- **请求方式：** POST
- **Content-Type：** application/x-www-form-urlencoded
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)

- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
notebook	笔记本路径	是	String
modify_time	删除时间，单位为秒，如果不指定则使用系统时间	否	Long

- **返回结果：**操作成功时 http 状态 200，无返回数据；失败时 http 状态 500 并返回错误码和错误信息，详见附录 B。

笔记操作 API

创建笔记

- **URL：** https://[baseURL]/yws/open/note/create.json
- **请求方式：**POST
- **Content-Type：**multipart/form-data
- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
source	笔记来源 URL	否	String
author	笔记作者	否	String
title	笔记标题	否	String
content	笔记正文（笔记格式见附录 A）	是	String
create_time	创建时间，单位为秒，如果不指定则使用系统时间	否	Long
notebook	该新建笔记所属的笔记本路径	否	String

- **返回结果：**操作成功时 http 状态为 200，并返回新建笔记的路径；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

  {“path” : “/4AF64012E9864C/FE89D12134E”}

  注：

  1. 如果第三方应用没有指定 notebook，则在该第三方应用对应的默认笔记本中创建该笔记。

2. 对于创建时间，主要作用在于对移动端的应用，笔记的创建时间和同步时间可能差别较大，因此如果不指定该时间，笔记的创建时间则会显示为同步时间而非本身的创建时间。

查看笔记

- **URL：** https://[baseURL]/yws/open/note/get.json
- **请求方式：** POST
- **Content-Type：** application/x-www-form-urlencoded
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
path	笔记路径	是	String

- **返回结果：** 操作成功时 http 状态为 200，并返回该笔记的相关信息；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

  ```
  {
  “title” : “工作记录”,                    // 笔记标题
  “author” : “Tom”,                        // 笔记作者
  “source” : “http://note.youdao.com”,     // 笔记来源 URL
  “size” : “1024”,                         // 笔记大小，包括笔记正文及附件
  “create_time” : “1323310917”             // 笔记的创建时间，单位秒
  “modify_time” : “1323310949”             // 笔记的最后修改时间，单位秒
   “content” : “<p>This is a test note</p>  // 笔记正文
  }
  ```

修改笔记

- **URL：** https://[baseURL]/yws/open/note/update.json
- **请求方式：** POST
- **Content-Type：** multipart/form-data
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
path	修改笔记的路径	是	String
source	修改后的笔记来源 URL	否	String
author	修改后的笔记作者	否	String
title	修改后的笔记标题	否	String
content	修改后的笔记正文（笔记格式见附录 A）	是	String
modify_time	修改时间，单位为秒，如果不指定则使用系统时间	否	Long

- **返回结果：**操作成功时 http 状态为 200，无返回结果；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。
  注：对于修改时间，主要作用在于对移动端的应用，笔记的修改时间和同步时间可能差别较大，因此如果不指定该时间，笔记的修改时间则会显示为同步时间而非本身的修改时间。

移动笔记

- **URL：** https://[baseURL]/yws/open/note/move.json
- **请求方式：**POST
- **Content-Type：**application/x-www-form-urlencoded
- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
path	想要移动的笔记原路径	是	String
notebook	目标笔记本的路径	是	String

- **返回结果：**操作成功时 http 状态为 200，返回移动后的笔记路径；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。
  结果示例：
  {“path”: “/5AB0C6B33BD1162/FE89D12134E”}

删除笔记

- **URL：** https://[baseURL]/yws/open/note/delete.json
- **请求方式：**POST

- **Content-Type：** application/x-www-form-urlencoded
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
path	想要删除的笔记原路径	是	String
modify_time	修改时间，单位为秒，如果不指定则使用系统时间	否	Long

- **返回结果：** 操作成功时 http 状态为 200，无返回结果；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

分享操作 API

分享笔记链接

- **URL：** https://[baseURL]/yws/open/share/publish.json
- **请求方式：** POST
- **Content-Type：** application/x-www-form-urlencoded
- **支持格式：** JSON
- **是否需要用户认证：** 是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
path	想要分享的笔记的路径	是	String

- **返回结果：** 操作成功时 http 状态为 200，并返回该分享笔记的链接；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  结果示例：

  {“url”:
  “http://note.youdao.com/share/?id=a5ae59ffda1e11caa43124288890b44d&type=note”}

  注：通过这种方式分享的笔记，任何人都可以通过该链接查看该笔记

附件操作 API

上传附件或图片

- **URL：**https://[baseURL]/yws/open/resource/upload.json
- **请求方式：**POST
- **Content-Type：**multipart/form-data
- **支持格式：**JSON
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **请求参数：**

参数名	参数说明	是否必须	数据类型/限制
file	上传的附件文件（包括附件的文件名、类型、大小、数据等）	是	Multipart File/附件大小限制 25M

- **返回结果：**操作成功时 http 状态为 200，对于图片，返回该图片的链接 URL，对于普通附件，服务器会为该附件生成一个图标文件，因此返回结果包括该附件对应的链接 URL 以及图标的 URL；失败时 http 状态为 500 并返回错误码和错误信息，详见附录 B。

  示例：

  图片附件

  {"url" : "http://[baseURL]/yws/open/resource/28/d83f547f3055"}

  普通附件

  {
  "url" : "http://[baseURL]/yws/open/resource/28/d83f547f3055" // 附件 URL
  "src": " http://[baseURL]/yws/open/resource/29/7f2bd6fa2795" // 图标 URL
  }

  需要注意的是，在附件上传完成后，使用者应该紧接着更新对应的笔记，否则不被任何笔记引用的附件将被定期的空间回收所删除。

下载附件/图片/图标

- **URL：**https://[baseURL]/yws/open/resource/download/...
- **请求方式：**GET
- **是否需要用户认证：**是 (关于登录授权，参见请求用户授权)
- **返回结果：**操作成功时，response 返回附件及图片的二进制流；失败时 http

状态为 500 并返回错误码和错误信息，详见附录 B。

- **备注：**
  1. 当打开笔记时，对于笔记中的附件及图片链接，只需要添加用户的 OAuth 认证信息即可以获取对应的附件及图片，同理在上传附件后，返回的附件及图片链接也可以通过这种方式进行访问。
  2. 附件及图片的下载支持断点续传

附录 A：有道云笔记内容格式

笔记内容的格式采用类 HTML 的标记语言，绝大部分格式相关 tag 均采用标准的 HTML Tag，唯一需要不同的是附件及图片 Tag。在格式上图片 Tag 与标准的 img tag 相同，一个图片资源的 tag 如下：

```
<img src="http://note.youdao.com/yws/open/resource/22/7f2bd6fa2795441c">
```

但是由于对应的接口需要用户认证，因此在获取图片资源时，需要加入 OAuth 的用户认证信息。

附件通常以图标在笔记中进行显示，因此我们扩展了图片 Tag <img>，除了 img 本身的 src 属性，额外加入了一个属性 path，其中 src 对应着该附件的图标 URL，而 path 则为该附件的 URL，如下所示：

```
<img path="http://note.youdao.com/yws/open/resource/21/9fe604c7c90642fd8f"
src="http://note.youdao.com/yws/open/resource/22/7f2bd6fa2795441c">
```

同理，也需要加入 OAuth 的用户认证信息，才可以正常访问附件资源及其图标，如果第三方应用希望用户能够下载对应附件，则需要自行对用户的行为进行响应。

附录 B：错误码及错误信息说明

同理，当 Open API 的请求在执行时出现任何异常或错误时，系统将返回 Http 500，应用程序可以根据此判断请求是否成功执行。当应用程序在进行 OAuth 认证，或者未经认证便调用 Open API 时，系统会返回 OAuth 错误信息，OAuth 的错误码与错误信息将以 json 的格式进行返回，各个错误码及错误信息说明如下：

错误码	错误描述及可能原因
1001	token_rejected：consumer 没有找到，consumerKey 错误， consumer

	没有创建，或者 request token 或者 access token 不存在于 server 中，或者没有获得授权的 token，或者 token 过期
1002	parameter_rejected：这个会返回具体原因 key 是 oauth_parameters_rejected， value 是一个 parameter 列表
1003	version_rejected：OAuth 版本号不是 1.0
1004	timestamp_refused：timestamp 非 5 分钟内进行的请求
1005	nonce_used：5 分钟内该 nonce 已经被使用
1006	parameter_absent：有些参数没有发送给 server
1007	signature_invalid：oauth 签名异常，可能原因没有签名，签名方法和签名不一致等
1008	signature_method_rejected：oauth server 没有提供该签名方法
1009	access_state_error：获取 token 时状态混乱，详细的信息会告知当前的状态 [AUTHORIZED | REQUEST | ACCESS | INVALID]
1010	consumer_rejected：该 consumer 在 server 中不存在，或者被禁掉。或者 consumerKey 改变，但是还没有反应到 server 的数据库中。
1011	accessor_rejected：由 token 得到的 accessor 构造异常，或者 accessor 中缺失某些属性，或者转换失败
1012	callback_error：request token 传来的 callback 和以后的 callback 不一致。
1013	callback_domain_error：当限定 callback 后，发来的 callback 与原设置的 callback 的域不相同
1014	verifier_error：Verifier 不一致错误
1015	permission_denied：无授权的 token，或者授权后没有获取过 access token

同理，当 Open API 的请求在后台执行出现异常或错误时，系统也将同时错误码与错误信息以 json 的格式一同返回，应用程序可以根据错误码得到相应的原因，常见的错误码说明如下：

错误码	错误描述及可能原因
206	UNKNOWN_URI：错误的 URI 请求
207	AUTHENTICATION_FAILURE：用户认证错误
209	RESOURCE_NOT_EXIST：请求的资源（笔记本、笔记、附件等）不存在
210	USER_SPACE_FULL：用户空间已满
214	INVALID_PARAMETER：错误的传入参数，必须的参数为空等
220	USER_NOT_EXISTS：用户不存在
221	USER_ALREADY_EXISTS：用户已经存在，不能重复注册
225	PARENT_NOT_EXIST：笔记本尚不存在，不能在该笔记本下创建笔记

231	RESOURCE_ALREADY_EXIST：该笔记或者附件已经存在，不能重复创建
304	NOTE_ALREADY_DELETED：笔记已经被删除
307	INVALID_APPLICATION：无效的第三方应用，未经过用户授权或者授权已过期，在这种情况下，应用程序应该提示用户再进行一次用户授权。

示例：当查看笔记路径错误时

{
"error" : "209",
"message" : " Required note does not exist, path=/5AB0C6B33BD12/FE89D12134E"
}

附录 C: OAuth 开源类库及有道云笔记 SDK

开源类库

ActionScript/Flash
oauth-as3 http://code.google.com/p/oauth-as3/

A flex oauth client
http://www.arcgis.com/home/item.html?id=ff6ffa302ad04a7194999f2ad08250d7

C/C++
QTweetLib http://github.com/minimoog/QTweetLib
libOAuth http://liboauth.sourceforge.net/

clojure
clj-oauth http://github.com/mattrepl/clj-oauth

.net
oauth-dot-net http://code.google.com/p/oauth-dot-net/
DotNetOpenAuth http://www.dotnetopenauth.net/

Erlang
erlang-oauth http://github.com/tim/erlang-oauth

Java
Scrible http://github.com/fernandezpablo85/scribe-java
oauth-signpost http://code.google.com/p/oauth-signpost/

javascript
oauth in js http://oauth.googlecode.com/svn/code/javascript/

Objective-C/Cocoa & iPhone programming
OAuthCore http://bitbucket.org/atebits/oauthcore
MPOAuthConnection http://code.google.com/p/mpoauthconnection/
Objective-C OAuth http://oauth.googlecode.com/svn/code/obj-c/

Perl
Net::OAuth http://oauth.googlecode.com/svn/code/perl/

PHP
tmhOAuth http://github.com/themattharris/tmhOAuth
oauth-php http://code.google.com/p/oauth-php/

Python
python-oauth2 http://github.com/brosner/python-oauth2

Qt
qOauth http://github.com/ayoy/qoauth

Ruby
Oauth ruby gem http://oauth.rubyforge.org/

Scala
DataBinder Dispatch http://dispatch.databinder.net/About

SDK

Java

http://code.google.com/p/ynote-sdk-java/

https://github.com/lichongxin/YNote-Java-SDK

PHP

http://code.google.com/p/ynote-sdk-java/downloads/list

iOS

即将发布

PS：同时也欢迎第三发应用开发者提供其他语言的 SDK

附录 D: OAuth 签名方法

1. 签名的 base string 由以下内容组成：
   a. 请求方法名，例如 GET/POST 等， 加密时方法名大写并进行特殊替换（详见 3）。
   b. URL (query parameter 之前的部分，不包括 ?)，http 或者 https 需要全部小写， http 如果带有 80 端口，或者 https 带有 443 端口，需要将其省略。其他端口则不能省略。例如：

      http://local:9999/requestToken
      ⇨ http://local:9999/requestToken
      http://local:80/requestToken
      ⇨ http://local/requestToken
      https://local:443/requestToken
      ⇨ https://local/requestToken

   c. 参数(query parameter 中的参数以及 OAuth 参数)，首先将这些参数进行特殊替换后，**按字典序排序升序排列**。如果两个 key 相同，则以 value 特殊替换后字典序为准。key 与 value 之间使用”=”连接，如果 value==null，则使用空字符串””替代，两个参数对之间使用”&”连接，连接后形成的最终字符串的开头和结尾没有”&”。

   将以上 a,b,c 三步进行特殊替换后，使用&连接就得到了签名的 base string，

最终形成的字符串的开头和结尾都没有”&”。

2. 签名的 key 是 ConsumerSecret + & + TokenSecret，如果 TokenSecret 还没有得到，则使用空字符串””。 ConsumerSecret 和 TokenSecret 需先做特殊替换，然后再进行签名。
3. 特殊替换是将 String 进行 URL 编码后，将”+”替换为”%20”,将”*”替换为”%2A”,将”%7E”替换为”~”.
4. 签名方法可以使用 HMAC-SHA1,PLAINTEXT, RSA_SHA1 三种。其中 PLAINTEXT 为 2 中的 key. 该方法不提倡使用。

得到签名内容后使用 Base64 进行编码，编码后的字符串即为 oauth_signature。

Examples：

1. 以获取 request_token 为例，该请求的基本信息如下：

请求 URL：http://note.youdao.com/oauth/request_token
请求方式：GET
OAuth 参数：oauth_consumer_key，oauth_signature_method，oauth_timestamp，oauth_nonce，oauth_version

对应签名的 base string 为

```
GET&http%3A%2F%2Fnote.youdao.com%2Foauth%2Frequest_token&oauth_consumer_key%3D2456f9dd37e162ffe237c8b88739925f%26oauth_nonce%3D1209317042071280%26oauth_signature_method%3DHMAC-SHA1%26oauth_timestamp%3D1343638182%26oauth_version%3D1.0
```

2. 以创建笔记本为例，该请求的基本信息如下：

请求 URL：http://note.youdao.com/yws/open/notebook/create.json
请求方式：POST
Content Type：application/x-www-form-urlencoded
请求参数：name=New_Notebook
OAuth 参数：oauth_consumer_key，oauth_signature_method，oauth_timestamp，oauth_nonce，oauth_version

对应签名的 base string 为

```
POST&http%3A%2F%2Fnote.youdao.com%2Fyws%2Fopen%2Fnotebook%2Fcreate.j
son&name%3DNew_Notebook%26oauth_consumer_key%3D2456f9dd37e162ffe237c
8b88739925f%26oauth_nonce%3D1209327240426679%26oauth_signature_metho
d%3DHMAC-SHA1%26oauth_timestamp%3D1343638192%26oauth_token%3D4948a92
00d25424566682af4ac8b2c4b%26oauth_version%3D1.0
```

附录 E: Oauth 2.0 错误码及错误信息说明

当 Open API 的请求在执行时出现任何异常或错误时，系统将返回 Http 500，应用程序可以根据此判断请求是否成功执行。当应用程序在进行 OAuth 2.0 认证，或者未经认证便调用 Open API 时，系统会返回 OAuth 错误信息，OAuth 的错误码与错误信息将以 json 的格式进行返回，各个错误码及错误信息说明如下：

错误码	错误描述及可能原因
1200	empty_clientId：请求参数中缺少了 client_id 参数
1201	empty_clientSecret：请求参数中缺少了 clientSecret 参数
1202	client_id_miss_match：输入的 client_id 与期望值不一致
1203	expire_authorization_code：授权 code 已过期
1204	unsupported_response_type：不支持的 response_type 类型
1205	invalid_authorization_code：授权 code 无效
1206	redirectUri_has_fragment：redirect_uri 中含有”#”
1207	invalid_redirectUri：redirect_uri 不匹配
1208	empty_redirectUri：redirect_uri 为空
1209	unauthorized_client：用户未对该应用授权
1210	unsupported_grant_type：不支持的 grant_type 类型
1211	unsupported_response_type：不支持的 response_type 类型
1212	empty_state：需要传入 state 参数
1213	token_secret_missing：通过 OAuth 1.0a 的 access_token 置换成 OAuth 2.0 可用的时候需要 token_secret 参数
1214	token_secret_miss_match：token_secret 不匹配
1215	client_secret_miss_match：client_secret 不匹配