

Authentication & Authorization for RESTful Doodle

This document builds on [RESTfulDoodle.pdf](#), and its topic is authentication & authorization for RESTful Doodle.

Prerequisites

The OAuth protocol is used for both authentication and authorization. We kindly refer developers without prior OAuth experience to the documentation (in particular the specification) available at <http://oauth.net/documentation>.

History

1.0.0	
-------	--

Contents

Prerequisites.....	1
History	1
Credentials.....	3
Legs and Resources	3
Request Token.....	3
URL	3
Request parameters.....	3
Response in case of success	3
Response in case of failure.....	3
Requirements for the request to succeed	4
Example	4
Access Token	4
URL	4
Request parameters.....	4
Response in case of success	4
Response in case of failure.....	4
Requirements for the request to succeed	5
Example	5
Accessing /polls	5
Functional aspects.....	5

Security aspects.....	5
Request parameters.....	5
Response in case of failure.....	5
Requirements for the request to succeed.....	6
Example.....	6
Accessing /users.....	7
Functional aspects.....	7
Request.....	7
Response.....	7
Security aspects.....	7
Parameter location within HTTP requests.....	7

Credentials

In order to successfully authenticate against the Doodle API (i.e., the OAuth provider), consumers require a valid consumer key and a valid consumer secret.

For development purposes, credentials can be created at <https://doodle.com/mydoodle/consumer/credentials.html> (as well as at <https://doodle-test.com/mydoodle/consumer/credentials.html>).

Every consumer has a limited number of read (GET) and write (POST, PUT) requests that it can submit. If more are submitted, the HTTP status code "402 PAYMENT REQUIRED" is returned.

Credentials that allow for many read and write requests are issued upon request (<http://doodle.com/about/contact.html>).

Legs and Resources

The Doodle API uses both 2-legged and 3-legged OAuth:

- Access to resources at "/polls" requires no authorization by any user so that a request token can immediately be exchanged for an access token.
- Access to resources at "/users" requires authorization by a user.

While polls are explicitly accessed by appending their ID to "/polls/", users are implicitly accessed by matching the access token to a user.

Request Token

URL

/oauth/requesttoken

Request parameters

oauth_consumer_key,
oauth_signature_method,
oauth_signature,
oauth_timestamp,
oauth_nonce,
oauth_version (optional)

Response in case of success

Status: 200 OK
Body: oauth_token (*Request Token*) and oauth_token_secret, both form encoded

Response in case of failure

Status: 401 Unauthorized
Header: WWW-Authenticate OAuth realm="http://doodle.com/api1"
Body: error message

Requirements for the request to succeed

The consumer key must be known to the server.

The request must be signed using HMAC-SHA1.

The signature must be verifiable using the consumer secret corresponding to the consumer key.

The time stamp must be within five minutes of the time on the server.

Example

```
GET /api1/oauth/requesttoken HTTP/1.1
```

```
Authorization: OAuth realm="", oauth_consumer_key="someconsumer",
```

```
oauth_signature_method="HMAC-SHA1", oauth_timestamp="1221836992",
```

```
oauth_nonce="30494169086665", oauth_signature="laL0ulw9K4MOQE%2Fu4V108%2FDAYHA%3D"
```

```
User-Agent: Jakarta Commons-HttpClient/3.1
```

```
Host: localhost:8080
```

```
HTTP/1.1 200 OK
```

```
Server: Apache-Coyote/1.1
```

```
Content-Type: text/plain
```

```
Content-Length: 94
```

```
Date: Fri, 19 Sep 2008 15:09:52 GMT
```

```
oauth_token=57ef3fa3afe93c7eed154efd475698c&oauth_token_secret=c97e34294587f4e8f91e5d89ec3d1b4
```

Access Token

URL

```
/oauth/accesstoken
```

Request parameters

oauth_consumer_key,

oauth_token,

oauth_signature_method,

oauth_signature,

oauth_timestamp,

oauth_nonce,

oauth_version (optional)

Response in case of success

Status: 200 OK

Body: oauth_token (*Access Token*) and oauth_token_secret, both form encoded

Response in case of failure

Status: 401 Unauthorized

Header: WWW-Authenticate OAuth realm="http://doodle.com/api1"

Body: error message

Requirements for the request to succeed

The `oauth_token` must be a previously granted *Request Token* and must not be older than ten minutes.

The request must be signed using HMAC-SHA1.

The signature must be verifiable using the token secret supplied with the *Request Token*. Note that this is a different secret than the consumer secret.

The time stamp must be within five minutes of the time on the server.

Example

```
GET / api1/oauth/accesstoken HTTP/1.1
```

```
Authorization: OAuth realm="", oauth_token="57ef3fa3afe93c7eed154efd475698c",  
oauth_consumer_key="someconsumer", oauth_signature_method="HMAC-SHA1",  
oauth_timestamp="1221836992", oauth_nonce="30494597633077",  
oauth_signature="YBl4I2d8Di79wD08%2FnjkiJugts%3D"
```

```
User-Agent: Jakarta Commons-HttpClient/3.1
```

```
Host: localhost:8080
```

```
HTTP/1.1 200 OK
```

```
Server: Apache-Coyote/1.1
```

```
Content-Type: text/plain
```

```
Content-Length: 95
```

```
Date: Fri, 19 Sep 2008 15:09:52 GMT
```

```
auth_token=31b886ef2e708328a196e39b34575e63&oauth_token_secret=c97e34294587f4e8f91e5  
d89ec3d1b4
```

Accessing /polls

Functional aspects

Cf. [RESTfulDoodle.pdf](#)

Security aspects

Request parameters

`oauth_consumer_key`,
`oauth_token`,
`oauth_signature_method`,
`oauth_signature`,
`oauth_timestamp`,
`oauth_nonce`,
`oauth_version` (optional)

Response in case of failure

Status: 401 UNAUTHORIZED

Header: WWW-Authenticate OAuth realm="http://doodle.com/api1"

Body: error message

Note that in the HTTP protocol, the terms “authentication” and “authorization” are used poorly. In particular, “401 Unauthorized” should in most cases read “401 Unauthenticated”.

Requirements for the request to succeed

The `oauth_token` must be a previously granted *Access Token* and must not be older than sixty minutes.

The request must be signed using HMAC-SHA1.

The signature must be verifiable using the token secret supplied with the *Access Token*. Note that this is a different secret than the consumer secret, but it can be identical to the *Request Token* secret.

The time stamp must be within five minutes of the time on the server.

Example

```
POST /api1/polls HTTP/1.1
```

```
Authorization: OAuth realm="", oauth_token="31b886ef2e708328a196e39b34575e63",
```

```
oauth_consumer_key="someconsumer", oauth_signature_method="HMAC-SHA1",
```

```
oauth_timestamp="1221836992", oauth_nonce="30494611596960",
```

```
oauth_signature="1DbA3iNsrydvMp9vpptfbb3s9Xg%3D"
```

```
User-Agent: Jakarta Commons-HttpClient/3.1
```

```
Host: localhost:8080
```

```
Content-Length: 282
```

```
<poll xmlns="http://doodle.com/xsd1">
```

```
<type>TEXT</type>
```

```
<hidden>>false</hidden>
```

```
<levels>2</levels>
```

```
<title>PPP</title>
```

```
<description>yum!</description>
```

```
<initiator><name>John Smith</name></initiator>
```

```
<options><option>Pasta</option><option>Pizza</option><option>Polenta</option></options>
```

```
</poll>
```

```
HTTP/1.1 201 Created
```

```
Server: Apache-Coyote/1.1
```

```
Content-Location: bxp65qxx62bhexm
```

```
x-DoodleKey: emxgsai6
```

```
Content-Length: 0
```

```
Date: Fri, 19 Sep 2008 15:09:52 GMT
```

Accessing /users

Functional aspects

Request

Resource

/users

Response

Body

The user encoded in XML according to schema user.xsd.

Example:

```
<user xmlns="http://doodle.com/xsd1">
<userId>p13vdm5er44r6oqn228h6lzmklepkylg</userId>
<name>Paul</name>
<polls>
<poll><pollId>BSPeckqdxsdnsm8r</pollId></poll>
<poll><pollId>BSPmx9khhbixeqhzb</pollId></poll>
</polls>
<calendarUris></calendarUris>
</user>
```

Security aspects

1. When requesting a request token, the consumer needs to specify what user data it wants to read by including an additional parameter:
 - Parameter name: `doodle_get`
 - Parameter value: `userId` and/or `name` and/or `initiatedPolls` and/or `participatedPolls` (separated by |)
 - Example: `doodle_get=name|initiatedPolls|participatedPolls`
2. A user has to authorize access to this data. To this end, the consumer needs to direct the user by including the request token and a callback URL to:
`https://doodle.com/mydoodle/consumer/authorize.html?oauth_token=REQUESTTOKEN&oauth_callback=CALLBACKURL`
Note that the user may authorize access to fewer pieces of data than the consumer requested.

Parameter location within HTTP requests

OAuth parameters can be sent to the server URL-encoded using GET requests, form-encoded using POST or PUT requests, or in the HTTP headers. That said, putting parameters into the request body interferes with some API calls, and encoding them in the URL is considered unfavorable by the OAuth specification and considered bad practice. Therefore, and as shown in the examples, it is highly recommended to supply OAuth parameters as HTTP headers.