Health Gorilla Clinical Network
OAuth 2.0 Guide

Date: January 17, 2018
Version: 1.2

Health Gorilla API uses OAuth 2.0 protocol for authentication and authorization. Health Gorilla supports base OAuth 2.0 scenarios for client-side applications.

To begin you need to contact Health Gorilla support@healthgorilla.com or your dedicated implementation representative to obtain your organization's OAuth 2.0 credentials and an access token to be used to call Health Gorilla API.

Resources:

- OAuth 2.0 home page - http://oauth.net/2/
- Protocol Specification - http://tools.ietf.org/html/rfc6749

Contents:

---

1. Obtaining OAuth 2.0 credentials

You must request OAuth 2.0 credentials from support@healthgorilla.com to be able to make OAuth 2.0 requests.

In order to obtain the credentials you will need to provide the following info:

Property Name	Description	Example
Available Callback URLs	List of URLs that your client will be able to specify in redirect_uri property when making OAuth 2.0 requests. Mandatory. **Callback URLs must use HTTPS protocol.**	“https://yoursite1.com, https://yoursite2.com”
Default callback URL	The default callback URL used by Health Gorilla OAuth 2.0 Authorization Server when OAuth request does not contain the redirect_uri parameter. Optional. **Callback URL must use HTTPS protocol.**	https://yoursite.com/callback
Attribution Logo	Logo that will be visible to the user during authorization.	
Web Site URL	This is a unique entity that identifies your resources.	https://<yoursite.com>

When the request is completed, you will receive the following:

Property Name	Description	Example

Client ID	Your unique client application identifier.	*qpgW44*
Client Secret	Secret. It is your password that your client application will use to authenticate with Health Gorilla CLinical Network. **Important! You must treat the Client Secret in accordance with your Health Gorilla License Agreement.**	*eeVk7vcq*
Scopes	Available scopes is a *comma-separated* list of permissions that identifies Clinical Network APIs for your client application's access. Only the API calls defined in **Scopes** will be enabled for your application.	get_results, get_profile

2. Getting an Access Token

To access Health Gorilla APIs you must obtain an access token that is issued to grant access to the requested APIs. A single access token can be used to access multiple APIs and access is identified by the **Scopes** variable linked to your specific token issued by Health Gorilla. When your application requests an access token, the request should contain the desired scopes; you will be able to request access only within the scopes that were allowed for you client application during registration.

There are several ways to get an access token from Health Gorilla OAuth 2.0 authorization server. They vary based on the type of client application and you can choose the one that applies to your application. Below you can find examples describing two possible authorization flows. More details can be found in OAuth 2.0 protocol specification.

Available authorization flows are:

1. Authorization Code Grant flow
2. Implicit Grant flow

Health Gorilla OAuth 2.0 authorization endpoint is:
`https://healthgorilla.com/oauth/authorize`

Authorization service is accessible only over SSL; plain text HTTP calls are refused. This endpoint accepts only HTTPS GET requests.

To start, the client should send HTTPS GET request to Health Gorilla OAuth 2.0 authorization endpoint. The request depends on the chosen authorization flow.

The following table lists common parameters for the authorization request:

Parameter	Values	Description
response_type	code or token	A predefined OAuth 2.0 constant. code for Authorization Code Grant flow token for Implicit Grant flow
client_id	The client identifier	Identifies the client that is making the request. This is the unique Client ID that you got after registration
redirect_uri	The callback URL	Defines the destination where to send response to. If specified then it should **exactly match** one of the callback URLs that you provided during registration. If not specified then the default callback URL will be used to respond
scope	A list of space-delimited permissions that the client is requesting	Defines the access rights to gain. Your application can request only scope that the client got after registration. If empty then the default scope is requested

state	Any string	A value used by the client to maintain state between the request and callback. Recommended to prevent XSRF attacks

Also it is recommended to include additional parameters that are described in Section 2.3.

Notes:

- The access token has limited lifetime and will expire after some time of inactivity. Also resource server may invalidate access token to force the client to repeat authorization. If refresh token has been issued by the authorization server then the application can request new access token (with the same grants) without authorization. Otherwise, the client application should go through the authorization flow again.
- You can always validate an access token using Health Gorilla info endpoint (see Section 4 for more details).
- An access token can be manually revoked using Health Gorilla revoke endpoint (see Section 5 for more details).

2.1. Authorization Code Grant

This is a two-step authorization flow used to obtain both access tokens and refresh tokens and is optimized for confidential web server applications.

2.1.1. Authorization Request

The first HTTPS GET request should be sent to Health Gorilla OAuth 2.0 authorization endpoint.

The set of supported query string parameters are:

Parameter	Values	Description
response_type	code	A predefined constant
Common authorization parameters		

It is recommended to include additional parameters described in Section 2.3.

Below you can find an example of the request:

```
https://healthgorilla.com/oauth/authorize?
  response_type=code&
  client_id=qpgW44&
  redirect_uri=https%3A%2F%2Fmy-oauth2.com%2Fcallback&
  scope=get_results&
  state=127&
  hg_user_first_name=Tom&
  hg_user_last_name=Sawyer
  hg_user_email=tomsawyer@mail.com
```

2.1.2. Authorization Response

HealthGorilla OAuth 2.0 Authorization Server will return an authorization code if the user is authenticated and has been granted the requested access rights. The response will be sent to redirect_url that was passed in the request (or to the default callback if the redirect_url was omitted). The response contains an authorization code and the state parameter (if present).
Below you can find an example of that response:

```
https://my-oauth2.com/callback?
  code=JDsd2jd24dsfDDFfd43r&
  state=127
```

When the client web server receives the authorization code, it must call Health Gorilla token endpoint to exchange its authorization code for an access token. Go to Section 2.1.4 to continue.

2.1.3. Authorization Error Response

If the authorization request was rejected or if any errors occurred, the authorization server returns an error response to the client.

Authorization Server returns an error code response in two ways:

- If the OAuth client (client_id) has not been authorized or callback URL (redirect_url) is invalid, then server returns response with HTTP status 400.
- Otherwise, the authorization server sends error response to redirect_url.

An example of the error response:

```
https://my-oauth2.com/callback?
  error=invalid_scope&
  state=127
```

2.1.4. Access Token Request

Once the authorization token is received the web server should send HTTPS POST request to Health Gorilla OAuth 2.0 token endpoint to obtain an access token and a refresh token.

You can find Health Gorilla OAuth 2.0 token endpoint description in Section 3.

HTTPS POST request should have the following parameters:

Parameter	Description
grant_type	authorization_code

redirect_uri	The callback URI that was specified in the original request
client_id	The client ID obtained during registration
client_secret	The client secret obtained during registration
code	The authorization code to exchange

Below you can find an example of that request:

```
POST /oauth/token HTTP/1.1
Host: www.healthgorilla.com
Content-Type: application/x-www-form-urlencoded
grant_type=authorization_code&
code=JDsd2jd24dsfDDFfd43r&
client_id=qpgW44&
client_secret=eeVk7vcq&
redirect_uri=https%3A%2F%2Fmy-oauth2.com%2Fcallback
```

2.1.5. Access Token Response

Health Gorilla OAuth 2.0 Token Service returns response as a JSON array.

A successful response contains the following fields:

Field	Description
access_token	The access token issued by the authorization server.
refresh_token	The refresh token, which can be used to obtain new access tokens using the same authorization grant.
expires_in	The lifetime in seconds of the access token.
token_type	Bearer
scope	The access rights that were granted and bound to the given access token.

Below you can find an example of the successful response

```
{
  "access_token": "sldfksgkkG78df78dgfjhnfknkn",
  "expires_in": 3600,
  "token_type": "Bearer",
  "refresh_token": "HKj87975vlJjkjhfH766",
  "scope": "get_results",
}
```

An error response contains the following fields:

Field	Description
error	An error code.
error_description	An additional information. Optional.

Below you can find an example of the error response

```
{
  "error": "invalid_grant",
  "error_description": "Invalid authorization code",
}
```

2.2. Implicit Grant

The implicit grant type is used to obtain access tokens (it does not support the issuance of refresh tokens) and is optimized for public clients known to operate a particular redirection URI. Usually this method is used by JavaScript-based applications that cannot keep a secret.

2.2.1. Request

The first HTTPS GET request should be sent to Health Gorilla OAuth 2.0 authorization endpoint. The set of supported query string parameters are:

Parameter	Values	Description
response_type	token	A predefined constant.
Common authorization parameters		

Also it is recommended to include an additional parameters that are described in Section 2.3.

Below you can find an example of that request:

```
https://healthgorilla.com/oauth/authorize?
  response_type=token&
  client_id=qpgW44&
  redirect_uri=https%3A%2F%2Fmy-oauth2.com%2Fcallback&
  scope=get_results&
  state=86&
  hg_user_first_name=Sarah&
  hg_user_last_name=Connor&
  hg_user_dob=01/01/1966
```

2.2.2. Response

Authorization server returns an access token as the result of the authorization request, if the user is authenticated and grants requested access rights. The authorization server redirects the user-agent back to the client using the redirection URI provided earlier. The redirection URI includes the access token in the URI fragment.

A successful response contains the following fields:

Field	Description
access_token	The access token issued by the authorization server.
expires_in	The lifetime in seconds of the access token.
token_type	Bearer
scope	The access rights that were granted and bound to the given access token.
state	The exact value received from the client.

Below you can find an example of that response:

```
https://my-oauth2.com/callback#
  access_token=2YotnFZFEjr1zCsicMWpAA&
  token_type=Bearer&
  expires_in=3600&
  scope=get_results&
  state=86
```

Once the access token received in that way then it should be validated using Health Gorilla Info endpoint. Tokens received on the fragment must be validated to prevent confused deputy attacks.

2.2.3. Error Response

If authorization request was rejected or if any error occurred then authorization server returns an error response to the client.
Authorization Server can do it in two ways. They are:

- If OAuth client (client_id) has not been authorized or callback URL (redirect_url) is invalid, then server returns HTTP response with status HTTP 400 to the client.
- Otherwise authorization server sends error response to redirect_url.

This is an example of that error response:

```
https://my-oauth2.com/callback#
```

```
error=access_denied&
error_description=Canceled%20by%20the%20user&
state=86
```

2.3. Extended authentication parameters

Health Gorilla Authorization Server extends OAuth 2.0 protocol. We define an additional parameters that client can specify to speed up authorization. Application can send these parameters if client knows the user that requires is trying to authenticate. It is an optional part, but recommended.

Here you can find the set of additional parameters:

Parameter	Values	Description
hg_user_first_name	First name	User first name if known.
hg_user_last_name	Last name	User last name if known.
hg_user_dob	Date of birth	User date of birth if known. Format: YYYYMMDD
hg_user_email	Email address	User email if known.

2.4. JSON Web Token for Authentication and Authorization Grants

JWT can be used for authentication and authorization mechanism and allows to request an access token.

This method is usually used when the remote service needs to obtain an access token without user intervention. A trusted relationship between the services must be established prior to this method utilization.

Specifications:

- Assertion Framework - https://tools.ietf.org/html/rfc7521
- JSON Web Token - https://tools.ietf.org/html/rfc7523

2.4.1. Issue JWT

JWT token must contains the following claims:

Claim	Values	Description

iss	An unique identifier for the entity that issued the JWT	Must match to a client URL which you sent to Health Gorilla during registration
aud	https://healthgorilla.com/oauth/token	Health Gorilla OAuth 2.0 token endpoint
sub	The is ID of the user that should be authenticated by JWT	Health Gorilla support will provide you this value
exp	‘expiration time’ claim that limits the time window during which the JWT can be used	
nbf	‘not before’ claim that identifies the time before which the token MUST NOT be accepted for processing	
iat	‘issued at’ claim that identifies the time at which the JWT was issued	

JWT token must contain the following header parameters:

Parameter	Values
typ	JWT

JWT must be signed with Client Secret that you received from Health Gorilla support after registration using HS256 (HMAC using SHA-256) algorithm.

2.4.2. Request

The web server should send HTTPS POST request to Health Gorilla OAuth 2.0 token endpoint to obtain an access token and a refresh token and include JWT to the request.

HTTPS POST request should have the following parameters:

Parameter	Description
grant_type	urn:ietf:params:oauth:grant-type:jwt-bearer
assertion	JWT

client_id	The client ID obtained during registration.
scope	A list of space-delimited permissions that the client is requesting.

Below you can find an example of that request:

```
POST /oauth/token HTTP/1.1
Host: www.healthgorilla.com
Content-Type: application/x-www-form-urlencoded
grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Ajwt-bearer&
client_id=qpgW44&
scope=user%2F*.*&
assertion=eyJhbGciOiJFUzI1NiIsImtpZCI6IjE2In0.
J9l-ZhwP[...omitted for brevity...].
eyJpc3Mi[...omitted for brevity...]
```

2.4.3. Response

Health Gorilla OAuth 2.0 Token Service returns response as a JSON array.
You can find details here - 2.1.5. Access Token Response

3. Refreshing the Access Token

If the authorization server issued a refresh token to the client, then client can make a refresh request to the token endpoint to obtain a new access token. The newly issued access token will have the same permissions as the previous one, so scope cannot be extended. Authorization server may return a new refresh token to use in further requests. The refresh token has fixed lifetime and request may be rejected by the server, in this case client should send a new authorization request.

Health Gorilla OAuth 2.0 token endpoint is:
`https://healthgorilla.com/oauth/token`
This service is accessible only over SSL, and HTTP calls are refused. This endpoint accepts only HTTPS POST requests.

3.1. Request

Client should send HTTPS POST request to Health Gorilla OAuth 2.0 token endpoint.

HTTPS POST request must contain the following parameters:

Parameter	Description
grant_type	refresh_token
client_id	The client ID obtained during registration

client_secret	The client secret obtained during registration
refresh_token	The secure refresh token issued by the authorization server

Below you can find an example of that request:

```
POST /oauth/token HTTP/1.1
Host: www.healthgorilla.com
Content-Type: application/x-www-form-urlencoded
grant_type=refresh_token&
client_id=qpgW44&
client_secret=eeVk7vcq&
refresh_token=HKj87975vlJjkjhfH766
```

3.2. Response

Health Gorilla OAuth 2.0 Token Service returns response as a JSON array.

A successful response contains the following fields:

Field	Description
access_token	The new access token issued by the authorization server
refresh_token	The new refresh token, which can be used to obtain new access tokens using the same authorization grant
expires_in	The lifetime in seconds of the access token
token_type	Bearer
scope	The access rights that were granted and bound to the given access token

Below you can find an example of the successful response

```
{
  "access_token": "sldfksgkkG78df78dgfjhnfknkn",
  "expires_in": 3600,
  "token_type": "Bearer",
  "refresh_token": "HKj87975vlJjkjhfH766",
  "scope": "get_results",
}
```

An error response contains the following fields:

Field	Description
error	An error code
error_description	Additional information (optional)

Below you can find an example of the error response:

```
{
  "error": "invalid_grant",
  "error_description": "Invalid refresh  token",
}
```

4. Validating the Access Token

Sometimes the client may need to validate an access token, for example to resolve confused deputy problem. Health Gorilla provides service that allows to verify the given access token.

Health Gorilla OAuth 2.0 Info endpoint is:
`https://healthgorilla.com/oauth/info`
This service is accessible only over SSL, and HTTP calls are refused.

For validating the application should make the request to the endpoint and pass the desired access token in the access_token parameter.

Below you can find an example of such a request:

```
https://healthgorilla.com/oauth/info?
  access_token=sldfksgkkG78df78dgfjhnfknkn
```

The Token Endpoint returns response as a JSON array.

If the token is valid then the status code of the response is 200. And the response contains the following fields:

Value	Description
client_name	The bane of the client that is identified as an owner of the token
client_id	The ID of the client that is identified as an owner of the token
expires_in	The lifetime in seconds of the access token

scope	The list of granted scopes, space-delimited

The client **must ensure** that client_id matches exactly the identifier that the client received after registration.

An example of such a response is shown below:

```
{
  "client_name": "My App",
  "client_id": "45f4OJJdf",
  "expires_in": 3602,
  "scope": "profile send_orders",
}
```

If validation fails for any reason - the token is expired, invalidated etc. - the status code of the response is 400.

An example error response is shown below:

```
{
  "error": "invalid_request",
}
```

5. Revoking the Token

Sometimes client may need to revoke its access token. For example, when the client application needs to be deactivated. Health Gorilla provides service that allows clients to invalidate any access tokens or refresh tokens.

Health Gorilla OAuth 2.0 Revoke endpoint is:
`https://healthgorilla.com/oauth/cancel`
This service is accessible only over SSL, HTTP calls are refused.

To revoke the token, an application should call the Revoke endpoint and pass the token in the token parameter. You can pass access token or a refresh token. If the client makes the request and passes an access token then a corresponding refresh token will also be invalidated.

Below you can find an example of such a request:

```
https://healthgorilla.com/oauth/cancel?
  token=sldfksgkkG78df78dgfjhnfknkn
```

The Revoke Endpoint returns the response with 200 status code. Or the error HTTP response if any errors had occurred.

6. Accessing Protected Resources

The client accesses Health Gorilla API by presenting the access token in the request.

Access Token can be passed in a two ways:

Passing access token as the query parameter:

```
https://healthgorilla.com/result?
  access_token=sldfksgkkG78dsdff787&
  id=jdkfjsdjfkjksdfgjk767676f
```

Passing access token in the header:

```
GET /result HTTP/1.1
Authorization: Bearer sldfksgkkG78dsdff787
Host: healthgorilla.com
```

If the specified access token is not accepted, the endpoint rejects the request and returns an error to the client. Below you can find some common errors that the server may return. If the access token is expired then the status code of the response is 401 and the result looks like the following:

```
HTTP/1.1 401 Unauthorized
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=435311D972BC5C1896C; Path=/; HttpOnly
WWW-Authenticate: Bearer
realm="PatientServiceServlet",error="expired_token"
Content-Length: 0
Date: Tue, 17 Mar 2015 14:33:16 GMT
```

If the access token isn't valid (it was never issued or was invalidated) then the status code of the response is 401 and the result looks like the following:

```
HTTP/1.1 401 Unauthorized
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=435311D972BC5C1896C; Path=/; HttpOnly
WWW-Authenticate: Bearer
realm="PatientServiceServlet",error="invalid_token"
Content-Length: 0
Date: Tue, 17 Mar 2015 14:33:16 GMT
```

If the access token does not have the requested access rights then the status code of the response is 403 and the result looks like the following:

```
HTTP/1.1 403 Forbidden
```

```
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=435311D972BC5C1896C; Path=/; HttpOnly
WWW-Authenticate: Bearer
realm="PatientServiceServlet",error="insufficient_scope"
Content-Length: 0
Date: Tue, 17 Mar 2015 14:33:16 GMT
```