STRAVA Developers

Getting Started with the Strava API

Welcome to the Strava API! This is a brief overview of how to use our API. Anyone who breaks a sweat is an athlete, so we refer to our users as athletes.

🌈 **Table of Contents**

🚴 A. Basic info about the API 🏃 🚣

The Strava REST API includes data on athletes, segments, routes, clubs, and gear. It is free to use. The Strava API does not allow you to get data for all Strava public athletes, as you can see on our website.

To get data on athletes, you will have to make an application and request that athletes sign in with Strava, and grant your application certain permissions using OAuth 2.0. You can get data on yourself without authentication for testing purposes.

Strava API usage is limited on a per-application basis using both a 15-minute and daily request limit. The default rate limit allows 200 requests every 15 minutes, with up to 2,000 requests per day.

If you’re looking to connect our athlete community to your app, and aren’t just developing for yourself, with your own data, please submit your app for review!

- Quick Links

Rate limits and pagination
Uploading files
Terms of Service
Brand Guidelines

- Examples of What to Build with the Strava API

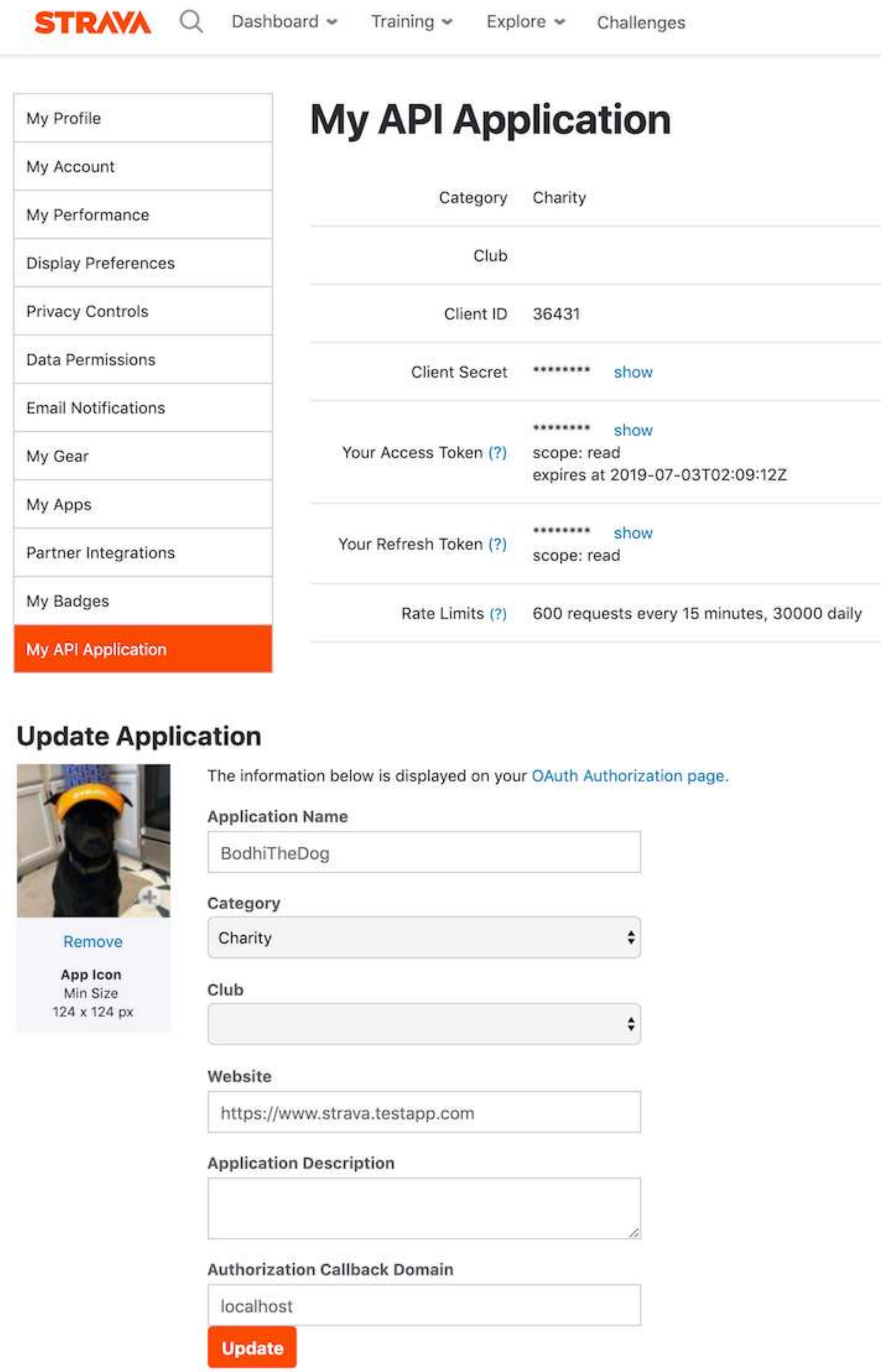
Create a print of your route
Add weather for your activities
Listen for new Strava activities using Strava webhooks
Measure how you are performing in your athletic activities

🏄 B. How to Create an account ⛷️

To start developing with the Strava API, you will need to make an application

1. If you have not already, go to https://www.strava.com/register and sign up for a Strava account.
2. After you are logged in, go to https://www.strava.com/settings/api and create an app.

3. You should see the “My API Application” page now. Here is what everything means:
    - Category: The category you chose for your application
    - Club: Will show if you have a club associated with your application
    - Client ID: Your application ID
    - Client Secret: Your client secret (please keep this confidential)
    - Authorization token: Your authorization token which will change every six hours (please keep this confidential)
    - Your Refresh token: The token you will use to get a new authorization token (please keep this confidential)
    - Rate limits: Your current rate limit
    - Authorization Callback Domain: When building your app, change “Authorization Callback Domain” to localhost or any domain. When taking your app live, change “Authorization Callback Domain” to a real domain.



My API Application

Category	Charity
Club	
Client ID	36431
Client Secret	******** show
Your Access Token (?)	******** show scope: read expires at 2019-07-03T02:09:12Z
Your Refresh Token (?)	******** show scope: read
Rate Limits (?)	600 requests every 15 minutes, 30000 daily

Update Application

App Icon
Min Size
124 x 124 px

The information below is displayed on your OAuth Authorization page.

Application Name: BodhiTheDog

Category: Charity

Club:

Website: https://www.strava.testapp.com

Application Description:

Authorization Callback Domain: localhost

🏊 C. How to Make a cURL request 🏄

1. Let's make a cURL request against the Strava API. cUrl is a command line tool and is available on all platforms. For our first request, we are going to get our profile information which is this call: https://developers.strava.com/docs/reference/#api-Athletes-getLoggedInAthlete

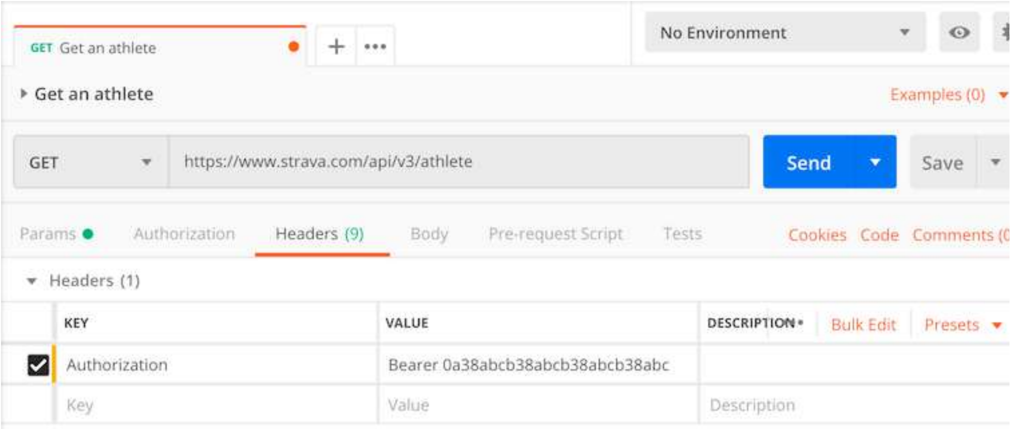
2. Access tokens are required for all requests, and can be included by adding "Authorization: Bearer #{access_token}" as a header.

3. To get your access token, go to https://www.strava.com/settings/api. Access tokens expire every six hours.

4. Make a cURL request:

```
curl -X GET \
https://www.strava.com/api/v3/athlete \
-H 'Authorization: Bearer YOURACCESSTOKEN'
```

5. If you are using Postman, it will look like this:



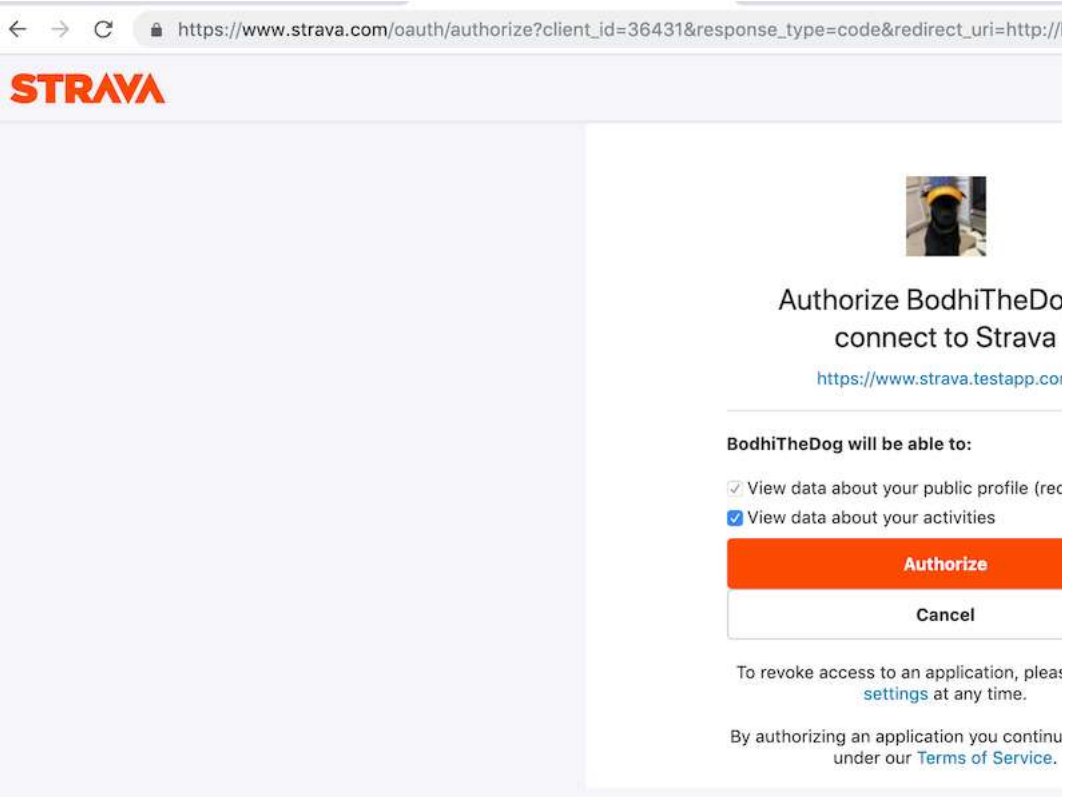
GET https://www.strava.com/api/v3/athlete

	KEY	VALUE	DESCRIPTION
☑	Authorization	Bearer 0a38abcb38abcb38abcb38abc	
	Key	Value	Description

🧗🏿‍♀️ D. How to Authenticate 🏊‍♀️

As you may have already noticed, we require authentication via OAuth 2.0 in order to request data about any athlete. OAuth 2.0 allows developers to interact with Strava athletes without having to store sensitive information. View our full documentation here.

When OAuth is initiated, the athlete is prompted by the application to log in to the Strava website and give consent to the requesting application. This is what your athlete will see:



https://www.strava.com/oauth/authorize?client_id=36431&response_type=code&redirect_uri=http://

Authorize BodhiTheDo
connect to Strava

https://www.strava.testapp.co

BodhiTheDog will be able to:

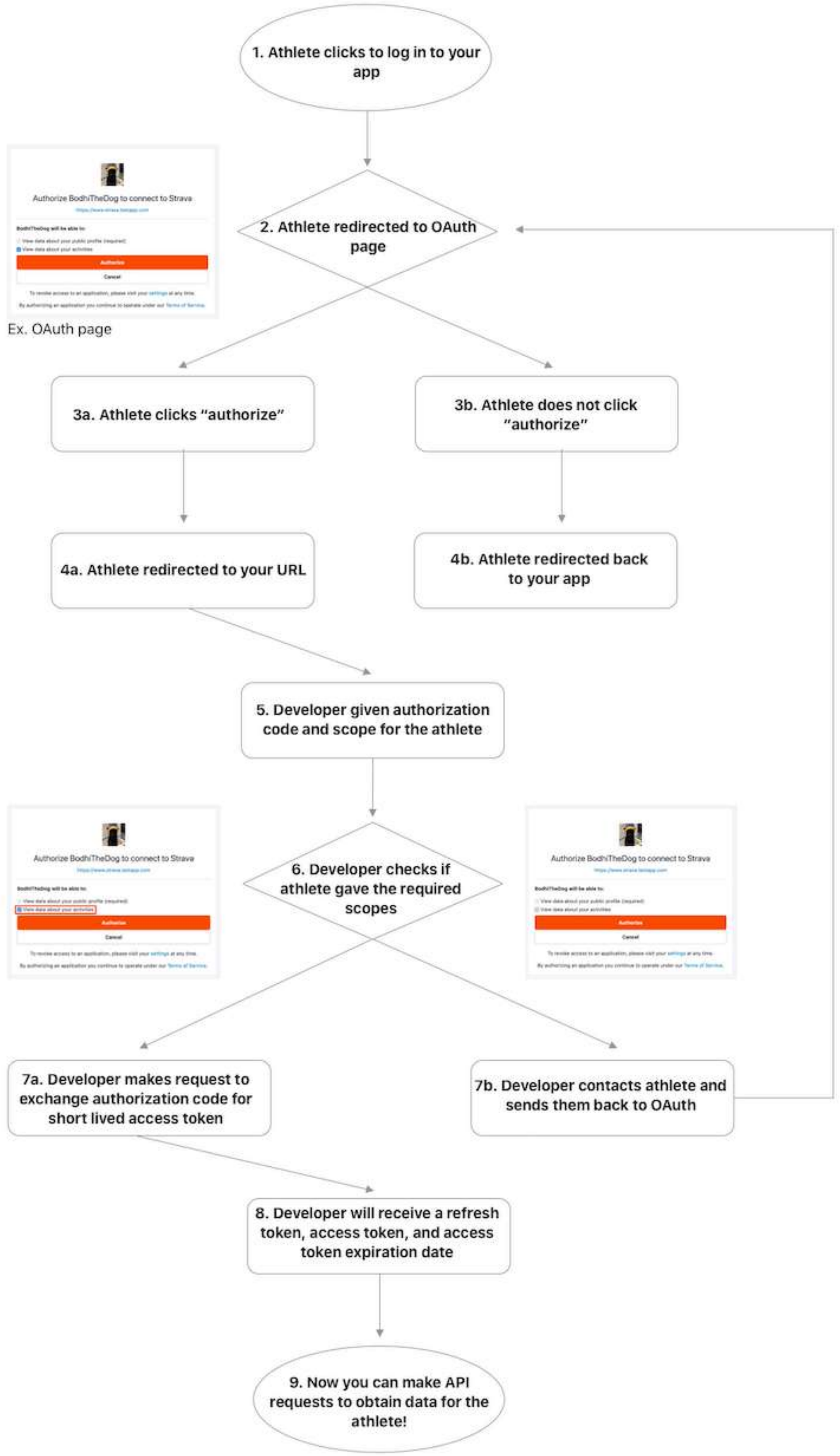
☑ View data about your public profile (rec
☑ View data about your activities

To revoke access to an application, pleas
settings at any time.

By authorizing an application you continu
under our Terms of Service.

After the athlete accepts or rejects the authorization request, Strava redirects the athlete to a URL specified by the application. If the athlete authorized the application, the URL query string will include an authorization code and the scope accepted by the athlete. Please check to see that the athlete has accepted all of your scopes. The application must complete the authentication process by exchanging the authorization code for a refresh token and short-lived access token.

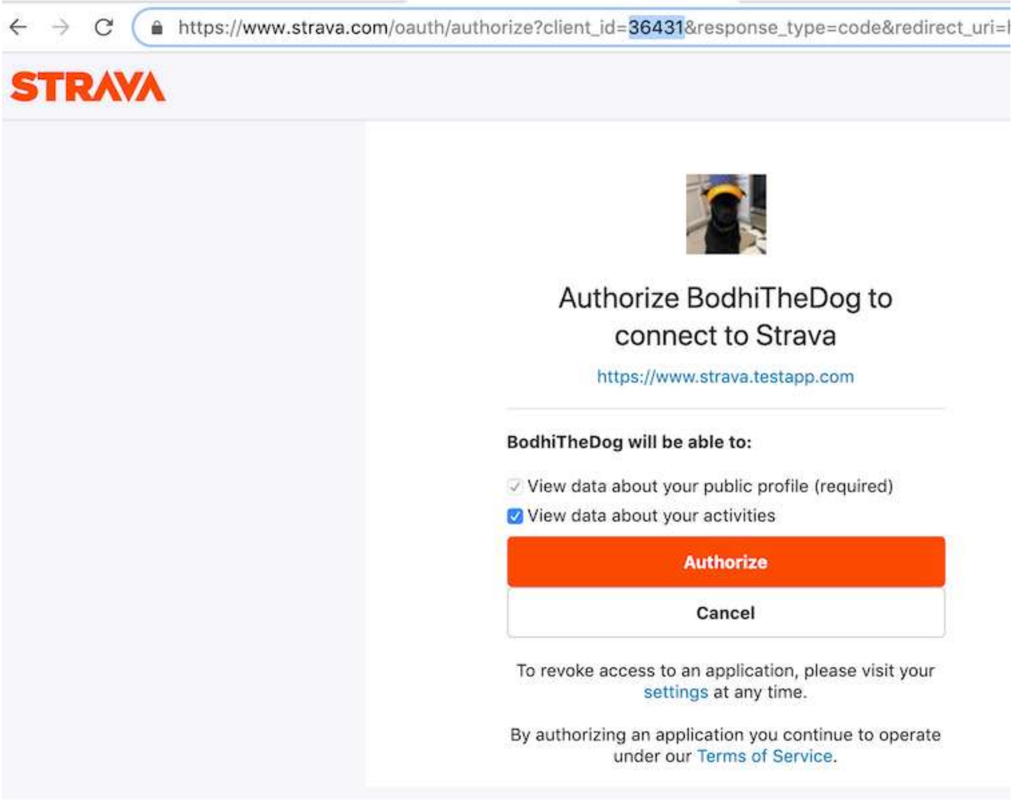
Confused? Here is a Graph:


1. Athlete clicks to log in to your app
Authorize BodhiTheDog to connect to Strava
Authorize
Cancel
Ex. OAuth page
2. Athlete redirected to OAuth page
3a. Athlete clicks "authorize"
3b. Athlete does not click "authorize"
4a. Athlete redirected to your URL
4b. Athlete redirected back to your app
5. Developer given authorization code and scope for the athlete
Authorize BodhiTheDog to connect to Strava
Authorize
Cancel
6. Developer checks if athlete gave the required scopes
Authorize BodhiTheDog to connect to Strava
Authorize
Cancel
7a. Developer makes request to exchange authorization code for short lived access token
7b. Developer contacts athlete and sends them back to OAuth
8. Developer will receive a refresh token, access token, and access token expiration date
9. Now you can make API requests to obtain data for the athlete!

For demonstration purposes only, here is how to reproduce the graph above with cURL:

1. Go to https://www.strava.com/settings/api and copy your Client ID
2. Paste your Client ID into this URL: `http://www.strava.com/oauth/authorize?client_id=[REPLACE_WITH_YOUR_CLIENT_ID]&response_type=code&redirect_uri=http://localhost/exchange_toke`
3. Go to a browser

4. Paste the URL you edited into the browser window (step 1 and 2 from the graph)
5. Hit enter
6. When you see the authorization page, click “Authorize” (step 3a from the graph)



https://www.strava.com/oauth/authorize?client_id=36431&response_type=code&redirect_uri=

Authorize BodhiTheDog to connect to Strava

https://www.strava.testapp.com

BodhiTheDog will be able to:

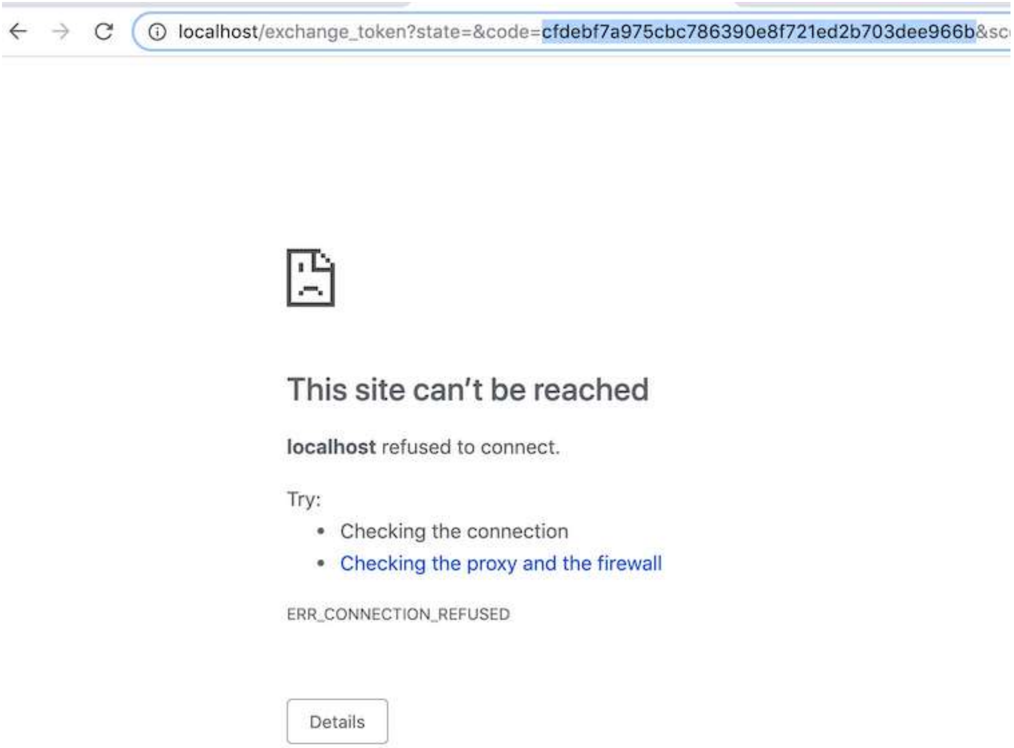
☑ View data about your public profile (required)

☑ View data about your activities

To revoke access to an application, please visit your settings at any time.

By authorizing an application you continue to operate under our Terms of Service.

7. After you click “Authorize,” you should see something like, “this site can’t be reached”
8. Stay on that page and look at the URL
9. The URL will show the authorization code and scope accepted by the athlete (step 5 from the graph)



localhost/exchange_token?state=&code=cfdebf7a975cbc786390e8f721ed2b703dee966b&sc

This site can’t be reached

localhost refused to connect.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_REFUSED

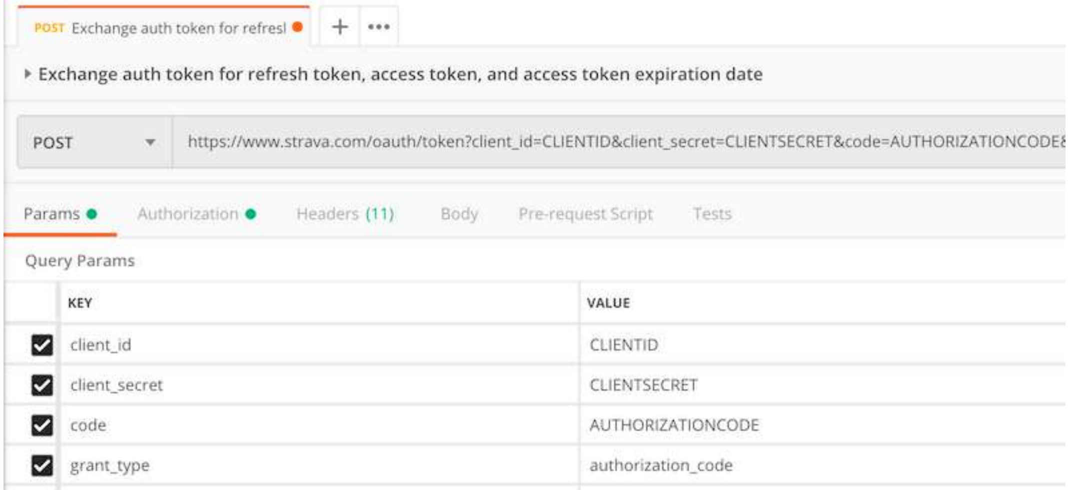
10. Copy the authorization code

11. Make a cURL request to exchange the authorization code and scope for a refresh token, access token, and access token expiration date (step 7a from the graph). Replace the client_secret and code. The response should include the refresh token, access token, and access token expiration date (step 8 from the graph).

    Sample cURL request:

```
    curl -X POST https://www.strava.com/oauth/token \
    -F client_id=YOURCLIENTID \
    -F client_secret=YOURCLIENTSECRET \
    -F code=AUTHORIZATIONCODE \
    -F grant_type=authorization_code
```

If you are using Postman, here is a sample request:



▸ Exchange auth token for refresh token, access token, and access token expiration date

POST ▾ https://www.strava.com/oauth/token?client_id=CLIENTID&client_secret=CLIENTSECRET&code=AUTHORIZATIONCODE&

Query Params

	KEY	VALUE
☑	client_id	CLIENTID
☑	client_secret	CLIENTSECRET
☑	code	AUTHORIZATIONCODE
☑	grant_type	authorization_code

Sample response:

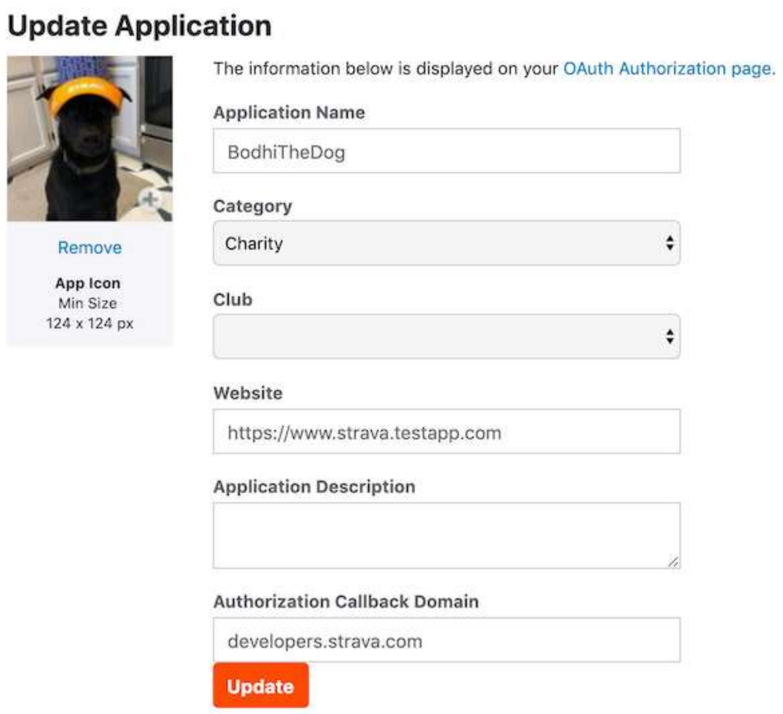
```
{
    "token_type": "Bearer",
    "expires_at": 1562908002,
    "expires_in": 21600,
    "refresh_token": "REFRESHTOKEN",
    "access_token": "ACCESSTOKEN",
    "athlete": {
        "id": 123456,
        "username": "MeowTheCat",
        "resource_state": 2,
        "firstname": "Meow",
        "lastname": "TheCat",
        "city": "",
        "state": "",
        "country": null,
        ...
    }
}
```

For more information on authentication, click here.

🏊 E. How to Use the Swagger Playground 🏄

1. The Swagger Playground is the easiest way to familiarize yourself with the Strava API by submitting HTTP requests and observing the responses before you write any client code. It will show what a response will look like with different endpoints depending on the authorization scope you receive from your athletes. To use the Playground, you will have to change your “Authorization Callback Domain” to

developers.strava.com.



Update Application

The information below is displayed on your OAuth Authorization page.

Application Name: BodhiTheDog

Category: Charity

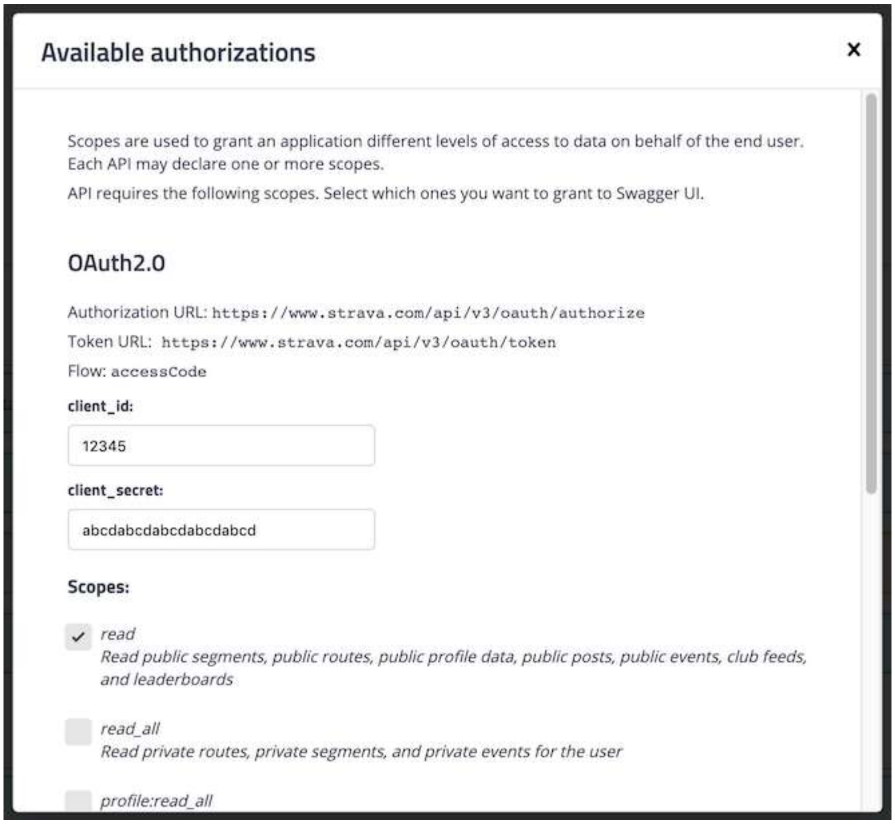
Club:

Website: https://www.strava.testapp.com

Application Description:

Authorization Callback Domain: developers.strava.com

2. Go to the Playground, https://developers.strava.com/playground, and click the green “Authorize” button on the right.
3. You will use your Client ID and Client Secret from https://www.strava.com/settings/api.



Available authorizations

Scopes are used to grant an application different levels of access to data on behalf of the end user. Each API may declare one or more scopes.

API requires the following scopes. Select which ones you want to grant to Swagger UI.

OAuth2.0

Authorization URL: `https://www.strava.com/api/v3/oauth/authorize`

Token URL: `https://www.strava.com/api/v3/oauth/token`

Flow: `accessCode`

client_id: 12345

client_secret: abcdabcdabcdabcdabcd

Scopes:

☑ *read*
Read public segments, public routes, public profile data, public posts, public events, club feeds, and leaderboards

☐ *read_all*
Read private routes, private segments, and private events for the user

☐ *profile:read_all*

🏃 F. Why Do I Need Webhooks? 🚣

- Per our API terms, you need to implement webhooks to know when an athlete has deauthorized your API application
- To avoid hitting rate limits
    - Instead of polling to see if athletes have new activities, you can subscribe to activity updates
- To know if activities have changed from public to private
- To subscribe to webhooks, please follow the instructions here

---

⛵ G. How to Get Support 🚴

Before emailing, please check our documentation first:

- Authentication: https://developers.strava.com/docs/authentication
- Webhooks: https://developers.strava.com/docs/webhooks
- API docs: https://developers.strava.com/docs/reference
- Basic info: https://developers.strava.com/docs

If you have questions, please check our developer community hub. Remember, **never** share access tokens, refresh tokens, authorization codes, or your client secret in a public forum.