How-To Guides
OAuth
OAuth Overview
Using OAuth
Fullscript.js
Fullscript Redirect
Webhooks
Events

OAuth overview

When developing a Fullscript integration, your app's Fullscript API interactions are done on behalf of Fullscript users. Until your app is authorized by at least one Fullscript user, there's very little you can do with our APIs.

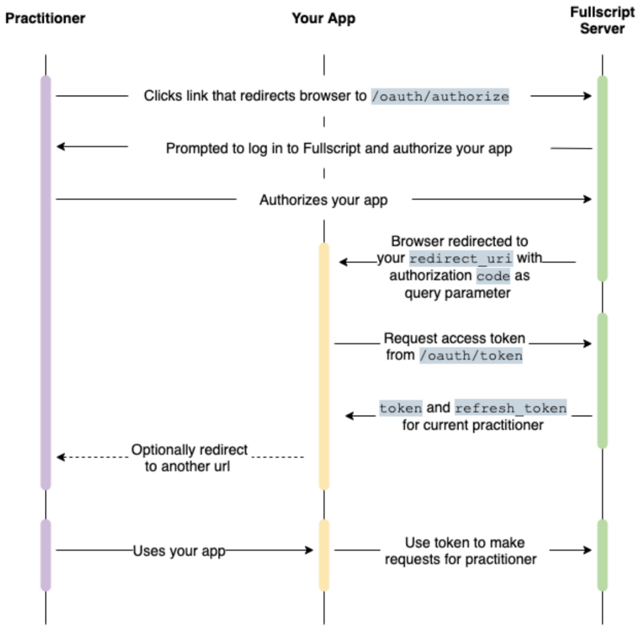
Related Topics

OAuth 2.0

OAuth scheme

Fullscript uses the OAuth 2.0 protocol with role-based access control. So, one of the first things to set up in your app is the OAuth flow.

We support the standard authorization code flow grant type. In this scheme, you obtain an authorization code from our API endpoint by asking each user to authorize your app to access their Fullscript private data. Your app then exchanges this authorization code for a secure access token that is used to access all the other API endpoints on behalf of the authorizing user.


Practitioner
Your App
Fullscript Server
Clicks link that redirects browser to /oauth/authorize
Prompted to log in to Fullscript and authorize your app
Authorizes your app
Browser redirected to your redirect_uri with authorization code as query parameter
Request access token from /oauth/token
token and refresh_token for current practitioner
Optionally redirect to another url
Uses your app
Use token to make requests for practitioner

OAuth steps

Let's take a closer look at the OAuth authorization code flow. It can be broken down into 3 steps.

1. Users grant access

Fullscript users (practitioners and office staff) are asked to grant your application access to their Fullscript account

In your app, add a place for users to trigger the authorization process to connect their Fullscript account. This can be a **Connect my Fullscript account** button in a special settings page, or it can be triggered via your app's usual **add a Fullscript treatment plan** flow.

When the link is clicked by a user who hasn't completed authorization or doesn't have a valid access or refresh token, redirect them to our OAuth page.

See Request an auth code for implementation details.

Button assets

We have a pre-created button pack you can use. Or, use our logo files to create your own. Here are three of our most commonly used colors:

- green (#88B04B) ■
- coal (#2E3A47) ■
- forest (#1B533F) ■

2. Redirect back to your app

As part of the OAuth setup, you'll give us a `redirect_uri`, which is an endpoint provided by your app. If the practitioner authorizes your application, we'll forward them to your `redirect_uri` with a **one-time-use authorization code** in the url query parameters.

You have **10 minutes** to exchange this authorization code for a secure access token (described in step 3, below) or the authorization code expires and you'll need to start back at step 1.

> **Tip**
> Generally, app developers don't expost the `redirect_uri` page to users. They store the authorization code, then immediately redirect the user to another spot in their app.

3. Token exchange (the OAuth dance 🕺)

Behind the scenes, your app exchanges the temporary auth code for an **access token** and can then access the Fullscript API.

OAuth access tokens expire in **2 hours**. But they also come with a non-expiring refresh token so you can refresh an expired token when needed.

See Request an access token and Refresh an access token for implementation details

> **FYI**
> Applications created before August 26, 2021 use OAuth 2.0 with **clinic** level authorization instead of role-based access control. This guide outlines the differences (if any) for apps using clinic level authorization. Look for **FYI** blocks like this one.

‹ How-to guides

Setup ›