Apple Music API / Generating Developer Tokens

Article

Generating Developer Tokens

Generate a developer token needed to make requests to Apple Music API.

Overview

To make requests to the Apple Music API, you need to authorize yourself as a trusted developer and member of the Apple Developer Program. The header of every Apple Music API request requires a signed developer token.

There are two paths to generate developer tokens:

- If you’re developing an app for Apple platforms (iOS, tvOS, watchOS or macOS), the recommended way to integrate with Apple Music is to use MusicKit for Swift, following the steps provided in Using Automatic Developer Token Generation for Apple Music API.
- Follow the directions below to create and manage developer tokens for other platforms.

Create a Developer Token

A developer token is a signed token used to authenticate a developer in Apple Music requests. Creating a MusicKit identifier and private key allows you to use a developer token to authenticate yourself as a trusted developer and member of the Apple Developer Program.

The Apple Music API supports the JSON Web Token (JWT) specification, so you can pass statements and metadata called *claims*. For more information, see the JWT specification and the available libraries for generating signed JWTs.

Construct a developer token as a JSON object whose header contains:

- The algorithm (`alg`) you use to sign the token, which should have a value of `ES256`
- A 10-character key identifier (`kid`) key, obtained from your developer account

Important

Apple Music supports only developer tokens signed with the ES256 algorithm. Apple Music rejects unsecured developer tokens or developer tokens signed with other algorithms. These rejections result in a `401` error code.

In the *claims* payload of the token, include:

- The *issuer* (`iss`) registered claim key, whose value is your 10-character Team ID, obtained from your developer account
- The *issued at* (`iat`) registered claim key, whose value indicates the time at which the token was generated, in terms of the number of seconds since epoch, in UTC
- The *expiration time* (`exp`) registered claim key, whose value must not be greater than `15777000` (6 months in seconds) from the current Unix time on the server
- Optional, but recommended for web clients, use the *origin claim* (`origin`). Only use this JWT if the origin header of the request matches one of the values in the array. This addition helps prevent unauthorized use of the tokens. For example: `“origin”:[”https://example.com”,”https://music.example.com”]`.

Tip

To locate your Team ID, sign in to your developer account, and click Membership in the sidebar. Your Team ID appears in the Membership Information section under the team name.

A decoded developer token has the following format.

```
{
     "alg": "ES256",
     "kid": "ABC123DEFG"
}
{
     "iss": "DEF123GHIJ",
     "iat": 1437179036,
     "exp": 1493298100
}
```

After you create the token, sign it with your MusicKit private key using the ES256 algorithm.

Note

ES256 is the JSON Web Algorithms (JWA) name for the Elliptic Curve Digital Signature Algorithm (ECDSA) with the P-256 curve and the SHA-256 hash.

Authorize Requests

A developer token is used to authorize all Apple Music API requests. If you manage this directly, in all requests, pass the `Authorization: Bearer` header set to the developer token.

```
curl -v -H 'Authorization: Bearer [developer token]' "https://api.music.apple.com/v1
```

To sign in and authenticate requests for an Apple Music subscriber, see User Authentication for MusicKit. For more information about requests, responses, and error handling, see Handling Requests and Responses.

Request Rate Limiting

Apple Music API limits the number of requests your app can make using a developer token within a specific period of time. If this limit is exceeded, you'll temporarily receive `429 Too Many Requests` error responses for requests that use the token. This error resolves itself shortly after the request rate has reduced.

See Also

Essentials

- User Authentication for MusicKit

  Authenticate requests for user data using the Music User Token.

- Handling Requests and Responses

  Write a request and handle responses from the API.

- Handling Resource Representation and Relationships

  Fetch resources with extended attributes and included relationships and relationship views.

- Storefronts and Localization

Pick a region-specific geographic location from which to retrieve catalog information, or retrieve information from the user’s personal library.

Common Objects

Understand the common JSON objects that framework responses contain.

Managing Content Ratings, Alternate Versions, and Equivalencies

Handle multiple and alternate versions of content.

Fetching Resources by Page

Use pagination to fetch the next set of objects.