On This Page

Authentication

Marketing API calls require an access token to be passed as a parameter in every API call.

See Access Tokens for Meta Technologies for more information on the various types of access tokens.

Get an Access Token for Your App

User Access Tokens

Graph API Explorer

You can get a user access token using the Graph API Explorer. To learn how to use the Graph API Explorer to make API calls, see the Graph API Explorer Guide.

1. In the **Meta App** field, select an app to obtain the access token for.
2. In the **User or Page** field, select **User Token**.
3. In the **Add a Permission** drop-down under **Permissions**, select the permissions you need (for example, `ads_read` and/or `ads_management`).
4. Click **Generate Access Token**. The box on top of the button is populated with the access token; store the token for later use.

Debug

To get more information in the token you just generated, click on the information icon (**i**) in front of the token to open the **Access Token Info** table, which displays some basic information about the token. Click **Open in Access Token Tool** to be redirected to the Access Token Debugger.

While debugging, you can check:

- **App ID:** The app ID mentioned in the prerequisite section.
- **Expires:** A time stamp. A short-lived token expires in an hour or two.
- **Scopes:** Contains the permissions added in the Graph API Explorer.

Extend your access token

1. Paste your token in the text box of the Access Token Debugger and click **Debug**.
2. Click **Extend Access Token** at the bottom of the **Access Token Info** table to get a long-lived token, and copy that token for later use.

Check your new token's properties using the Access Token Debugger. It should have a longer expiration time, such as "60 days", or "Never" under **Expires**. See Long-Lived Access Token for more information.

System User Access Tokens

A system user access token is a type of access token that is associated with a system user account, which is an account that is created in Meta Business Manager for the purpose of managing assets and calling the Marketing API. System user access tokens are useful for server-to-server interactions where there is no user present to authenticate. They can be used to perform actions on behalf of the business, such as reading and writing business data, managing ad campaigns, and other ad objects.

One benefit of using a system user access token is that it does not expire, so it can be used in long-running scripts or services that need to access the Marketing API. Additionally, because system user accounts are not tied to a specific individual, they can be used to provide a level of separation between personal and business activity on Meta technologies.

System user tokens are also less likely subject to invalidation for other reasons compared to the long-lived user access tokens.

See System Users for more information.

Get an Access Token for Ad Accounts you Manage

After the owner of an ad account you are going to manage clicks the **Allow** button when you prompt for permissions, they are redirected to a URL that contains the value of the `redirect_uri` parameter and an authorization code:

```
http://YOUR_URL?code=<AUTHORIZATION_CODE>
```

You can then build the URL for an API call that includes the endpoint for getting a token, your app ID, your site URL, your app secret, and the authorization code you just received:

```
https://graph.facebook.com/v23.0/oauth/access_token?
  client_id=<YOUR_APP_ID>
  &redirect_uri=<YOUR_URL>
  &client_secret=<YOUR_APP_SECRET>
  &code=<AUTHORIZATION_CODE>
```

The API response should contain the generated access token:

- If you follow the server-side authentication flow, you get a persistent token.
- If you follow the client-side authentication flow, you get a token with a finite validity period of about one to two hours. This can be exchanged for a persistent token by calling the Graph API endpoint for Extending Tokens.

If the API is to be invoked by a system user of a business, you can use a system user access token.

You can debug the access token, check for expiration, and validate the permissions granted using the access token debugger or the programmatic validation API.

Storing the Token

Your token should be safely stored in your database for subsequent API calls. Moving tokens between your client and server must be done securely over HTTPS to ensure account security. Read more about the implications of moving tokens between your clients and your server.

You should regularly check for validity of the token, and if necessary, prompt for permissions renewal. Even a persistent token can become invalid in a few cases, including the following:

- A password changes
- Permissions are revoked

As user access tokens can be invalidated or revoked anytime for some reasons, your app should expect to have a flow to re-request permission from users. Check the validity of the user token when they start your app. If necessary, re-run the authentication flow to get an updated token.

If this is not possible for your app, you may need a different way to prompt for permissions. This can happen in cases where the API calls are not directly triggered by a user interface or are made by periodically run scripts. A possible solution is to send an email with instructions.

Best Practices for Secure Credential Management

To ensure the security of user credentials and access tokens, you should adhere to the following best practices:

- **Use HTTPS:** Always transmit access tokens over secure connections (HTTPS) to prevent interception by malicious actors.
- **Store Tokens Securely:** Utilize secure storage solutions, such as encrypted databases, for storing access and refresh tokens, minimizing the risk of unauthorized access.
- **Limit Token Scope:** Request only the minimum necessary permissions, reducing the risk of overexposure to user data.
- **Implement Token Expiration:** Regularly refresh tokens and have a robust mechanism to handle expiration, ensuring continued access without exposing long-lived tokens.

Learn More

- Access Tokens
- Long-Lived Tokens
- Debugging and Errors
- Session Info Access Tokens
- Portability