Introduction

This guide was designed to assist Marketing Partners focused on the SMB vertical design architecture and integrate with the Ads API. The audience of this guide are partners who are interested in building a self-serve ads management platform.

The goals of this guide are as follows

Chapter 1: Understanding Access Tokens
A quick introduction to the OAuth2 and access tokens. We will quickly go over what an access token represents in the context of your platform.

Chapter 2: Payments
Under the self-serve model, the advertiser will pay for their Facebook spend. The partner can charge a percentage or flat rate on top of that separately. It's keen to note that the payment credentials must be supplied to both Facebook and the partner.

Chapter 3: Ad Account Ownership
The ad accounts are owned by the advertiser. They can be the user's personal ad account or an ad account they have access to through Business Manager.

Chapter 4: Assets
The Facebook Page should be owned by the advertiser. The images, videos, custom audiences, and other ad account specific objects can be owned by the partner.

Chapter 5: Access Token Handling
Refreshing access tokens, long-lived tokens, and storing access tokens.

Chapter 6: Business Manager
Business Manager, while not necessary for this implementation, may be helpful to address certain use cases (when a small business customer becomes not so small).

Chapter 7: Generating a Preview
Generating a preview to upsell users on Facebook Ads.
Understanding Access Tokens

This section is intended to be a brief recap of what an access token represents in the context of this implementation. For a more technical and deeper exploration into access tokens, you can reference our access token overview.

What is an access token?

An access token is the representation of a user (or page) in the context of an application. This means API calls made by the access token look like the user taking actions.

How do I get an access token?

The simplest way to retrieve an access token is by using Facebook's Javascript SDK. Implementing this means your platform will invoke an authentication flow with the advertiser, requesting permissions for ads_management and manage_pages.
Do I need to add the advertiser as an "advertiser"?

No, ad accounts have an option to add another user as an admin or moderator. This grants them access to these assets within the Facebook interfaces. *This flow does not apply to the API.*

You do not need additional permissions granted within the interfaces. The access token you retrieve allows you to act as the user, who is presumably already an admin on both assets.
Payments

Payments for Facebook Ads will be paid by the advertiser. Under this model, the partner must charge their fee separately from the ad spend. This means two things:

- The advertiser owns the ad account.
- The advertiser has updated payment credentials on that ad account.

Creating an ad account

Ad accounts can be under the Business Manager. The next chapter will cover this topic in greater detail. The first step should be verifying if a business manager exists:

curl -G \
-d "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/me/businesses"

If not, you can create it using the same call but as a POST request.

curl \
-F "name=Pomni Media" \
-F "vertical=ADVERTISING" \
-F "primary_page=<PAGE_ID>" \
-F "timezone_id=1" \
-F "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/<USER_ID>/businesses"
The vertical is a constant string of one of the following types: ADVERTISING, AUTOMOTIVE, CONSUMER_PACKAGED_GOODS, ECOMMERCE, EDUCATION, ENERGY_AND_UTILITIES, ENTERTAINMENT_AND_MEDIA, FINANCIAL_SERVICES, GAMING, GOVERNMENT_AND_POLITICS, MARKETING, ORGANIZATIONS_AND_ASSOCIATIONS, PROFESSIONAL_SERVICES, RETAIL, TECHNOLOGY, TELECOM, TRAVEL, OTHER

To create an ad account, the partner should make a POST request to the adaccounts edge on the business node.

```bash
curl
-F "name=MyAdAccount"
-F "currency=USD"
-F "timezone_id=1"
-F "end_advertiser=<END_ADVERTISER_ID>"
-F "media_agency=<MEDIA_AGENCY_ID>"
-F "partner=NONE"
-F "access_token=<ACCESS_TOKEN>"

"https://graph.facebook.com/<API_VERSION>/<BUSINESS_ID>/adaccount"
```

You can also make a GET request to the same edge if you want to use an existing ad account (assuming a Business Manager was already available).

**Adding Payment Credentials**

This payment dialog is under limited testing. Attaching payment credentials to an ad account is not available through the API. Programmatically attaching payment credentials can have an unintended side effect where the advertiser is unaware
they are being charged in two places (with partner and with Facebook). However, we have provided a flow to easily allow partners to attach payment information to an ad account within their platform.

Using the Javascript SDK, the partner can invoke a popup window to request payment information.

This dialog can be invoked with the following javascript:

```javascript
FB.ui({
  method: 'ads_payments',
  ad_account_id: 'act_81940859402',
  display: 'popup',
}, function(response) {
  // callback
});
```
Ad Account Ownership

Under this model, the ad account must be owned by the advertiser. As a consumer of the API, you can read ad accounts from either the user's account or their business manager. If you want to assist in the creation of ad accounts, you may only do that in Business Manager. Check out this guide to learn more about creating ad accounts.

The alternative option (which this guide doesn't cover) is for the partner to create ad accounts within their own Business Manager and map the ad accounts to advertisers. While one is not necessarily better than the other, there are a few key distinctions here.

1. The advertiser can see their ad spend within Facebook's ads interfaces.
   There is a transparency concern when the ad account is not owned by the advertiser and the spend is passed down to them. By policy, the ad spend must be itemized on the invoice, but in a model where the advertiser owns the ad account, this is not a concern.

2. The advertiser owns all their images, videos, custom audiences, etc.
   This means all the assets the advertiser provided to the partner for ads creation is accessible through Facebook interfaces.

3. The advertiser pays for their ad spend and pays the partner separately.
   There is a bit of friction here as the advertiser must enter payment information twice. However, this is less of a concern when Facebook Ads are an upsell (later down the funnel) to the partner's main offering.

4. The partner does not have to set up a line of credit to pay for all their customers up front.
   If the partner pays the bill and passes the cost down, they must have a line of credit large enough to cover the cost ahead of time.

Assets

Pages, images, and videos all belong to the advertiser. Because pages represent the user's business, it is imperative that the advertiser own their page. If an advertiser outgrows a platform, they should be able to take their page with them.
Creating a page

Page creation is only available for Standard Access partners.

curl \n  -F "name=My New Page" \n  -F "category_enum=" \n  -F "picture=<PROFILE_PICTURE_URL>" \n  -F "cover_photo=<COVER_PHOTO_URL>" \n  -F "about=<ABOUT_MY_PAGE>" \n  -F "access_token=<ACCESS_TOKEN>" \n  "https://graph.facebook.com/<API_VERSION>/me/accounts"

Read the list of pages available from a user. Note this is /accounts and not /adaccounts.

curl -G \n  -d "access_token=<ACCESS_TOKEN>" \n  "https://graph.facebook.com/<API_VERSION>/me/accounts"

The response from this call will contain a Page Access Token.

Access Token Handling

Related Topics
- Back to Overview
- Understanding Access Tokens
This section goes over in greater detail the process for handling access tokens. For the most part, you should not need to sync data into your own databases, so you can rely on the API to power your interface.

After receiving the access token as covered in Chapter 1, you can ideally store this in session storage, local storage, or a cookie.

Once the user is authenticated to your application, the javascript SDK will handle refreshing tokens for the user’s session, but **you must make sure you are correctly updating the token when the SDK reauthenticates.**

From here, you can make API calls using this token which will now be accessible from server-side or client-side languages.

**Storing the Access Token**

Normally, the token you retrieve will expire in a few hours and will not be that useful to store. However, if you need to store the token to make automated calls while the user is not necessarily logged into your application, you can request a *long-lived token*.

The API call to retrieve the long-lived token should be made server-side as it requires your *app_secret*.

**Business Manager**

For a basic implementation, a partner focused on the small business vertical will not need to integrate with business manager. Business manager comes into play if you want to make a *lot* of programmatic calls using *System Users*.

**Using System Users**

You must have *Standard Access* to use System Users.

1. **The partner must have a business manager.**

A business manager will grant you access to *system users*, which have more relaxed constraints on rate limiting. This allows you to programmatically make read calls for insights or metadata on a cron.
Keep in mind that making calls on a cron can be done with a long-lived token, but it is subject to user rate limiting which may be a concern for large advertisers.

2. The partner must request access to the advertiser's ad account.
You can request access to the advertiser's ad account using the following API call. This makes your system user and ADVERTISER on the account.

```
curl \
-F "adaccount_id=act_<AD_ACCOUNT_ID>" \
-F "access_type=AGENCY" \
-F "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/<BUSINESS_ID>/adaccounts"
```

3. The partner must request access to the advertiser's page.
The same flow exists for the advertiser's page as well.

```
curl \
-F "page_id=<PAGE_ID>" \
-F "access_type=OWNER" \
-F "access_token=<ACCESS_TOKEN>" \
"https://graph.facebook.com/<API_VERSION>/<BUSINESS_ID>/pages"
```

4. Make API calls using the system user token.
Now that you have the system user as an ADVERTISER on the advertiser's assets, the system user's token is authenticated to make calls against the ad account or page.

You can learn more about setting up system users or getting their tokens here.
Generating a Preview

Often times, an SMB who uses your platform can be upsold on Facebook Ads. In the instance your platform hosts the SMB's assets (images, products, etc), you can generate a preview of what their ad may look like with text and images.

It is not necessary to fully render the ad on your own, instead the following API call is available.

For ease of reading, the creative object has been truncated with object_story_spec stripped out. It's representation is documented below the API call.

```
curl \
-d "ad_format=DESKTOP_FEED_STANDARD" \n-d "creative={"actor_name": "{PAGE_NAME}"}, "object_story_spec": "<OBJECT STORY_SPEC>", "actor_image_url": "{<LINK TO IMAGE}>"}" \n-d "access_token": "{ACCESS_TOKEN}" \n"https://graph.facebook.com/<API_VERSION>/generatepreviews"
```

The object_story_spec object looks as follows (represented in JSON):

```
{
    "object_story_spec":{
        "link_data":{
            "actor_image_url": "<IMAGE_URL>",
            "actor_name": "<NAME>",
            "call_to_action":{
        
```
The response from this call provides an HTML output that contains an iframe.

```html
{  
  "body": "<iframe  
    width="540"  
    height="450"  
    scrolling="yes"  
    style="border: none;"  
  ></iframe>"
}
```
When embedded, the iframe will render a page post ad unit like the ones that are available on desktop.

---

It's fig season! Not sure what to do with figs? Here's a great dessert recipe to share.

**Fig Tart with Almonds**

The simplicity of this tart perfectly accentuates ripe figs. If you don't have enough time to make a handmade crust, pick up one of Jasper's pre-made pie crusts.

WWW.JASPERS-MARKET.COM 

Learn More