Enterprise Server-to-Server API workflow

The purpose of this document is to illustrate the overall process and technical workflow for Adobe Enterprise partners and customers who want to integrate with the Adobe Stock API. The Stock API allows you to interact with Adobe Stock programmatically, rather than through a user interface.

Document contents

Introduction

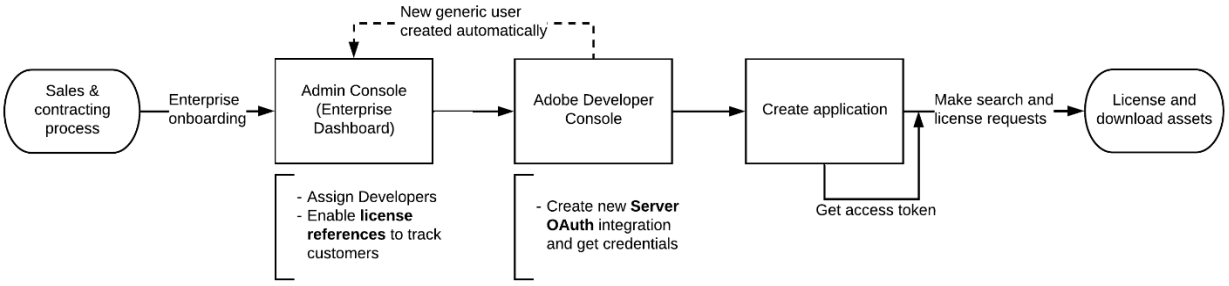
This paper outlines the workflow for any external integrator who wishes to license assets on behalf of its users. This could include:

- **Enterprise customers**. Integrator is an Enterprise Adobe customer who uses Stock for internal applications, such as for its branded websites. When it licenses assets, they are available for the entire organization.
- **Retailers (e.g., Print on Demand).** This partner produces commercial goods using Adobe Stock images and sells them directly to consumers. Unlike the Enterprise use case, licensed assets are only available per user and must be re-licensed to be used again.
- **Marketing platforms (e.g., social media aggregators and website builders).** There are multiple use cases, but the goal for the partner is to increase the value of their platform or application by offering the convenience of direct Stock access within their platform.

Some distinguishing factors of the Enterprise Server-to-server workflow:

- Partners must be on-boarded as Adobe Enterprise customers. As described below, applications must create a *server-to-server* integration to allow asset licensing via the API, and only Enterprise customers may create OAuth Server-to-Server credentials.
    - This account allows the application to license assets on behalf of all the users in their organization or on behalf of their customers. The end users would not need to interact directly with Adobe's API or even need Adobe accounts.
- It is assumed that communication between the application and Adobe will be performed server-to-server, as opposed to client webpage-to-server. This ensures the highest level of security.

Process overview



- The process begins with a contracting process with the Adobe sales team. While it is possible to test the search APIs without an Enterprise account, the licensing APIs require an account which has a quota of credits or an active subscription.
    - Sales and contracting are beyond the scope of this document. To get more information, please visit https://www.adobe.com/creativecloud/business/enterprise.html.
- Once the account is provisioned and the partner goes through a brief onboarding process, a new Enterprise organization is created which grants access to the Adobe Admin Console. This is where the administrator can add users to the Developer role, allowing them to create integrations for the organization.
- Developers would then sign into the separate Adobe Developer Console, which is used to create application integrations. The result of this process is an API key and a set of credentials which are used to authenticate requests.
- Now the integrator can start writing an application that uses the Stock API to make search queries and programmatically generate an access token that allows the app to make authenticated license requests and fetch license history.
- When an end user interacts with the application and chooses to license an item, the application licenses an asset on their behalf and can download the asset for delivery.

Adobe Admin Console overview

Enterprise customers interact with their Adobe account via the Admin Console, where they can add new users, see their active entitlements, and manage their products. This is where any or all the following activities might occur:

- Create a System Administrator account for the integrator or assign them to the Developers role. This account will be used to create credentials for the application; however, this account will *not* be used to make API requests. Instead, the Developer Console will generate a technical user account which will be used to authenticate API calls (see *Developer Console workflow*, below).
    - Note that it is not necessary to add additional users via the Admin Console unless they need the ability to license assets directly from the Stock website, or re-download assets for fulfillment. It is expected that your application will interact with the Stock API in place of user interaction.
    - Admin Console documentation: https://helpx.adobe.com/enterprise/help/aedash.html
    - Assigning developers: https://helpx.adobe.com/enterprise/using/manage-developers.html
- Optionally, enable reference fields that get assigned as extra metadata for each transaction, simplifying tracking and reporting. For example, in a retail or marketing tools use case, the customer's name and reference can be specified as part of the license request. This could potentially be added to an invoice for that customer.
    - References are configured on the **Permissions** tab of the product configuration. If these options do not appear in the Console, contact the Adobe team to enable this feature.
- Reassign quota to multiple internal brands. By default, all Stock credits are allocated to a single product profile, usually called "Default Adobe Stock configuration." It is possible to create additional profiles and change the distribution of those credits. For example, if there are two distinct branded websites sharing the quota and you wish to keep them separate, you can assign each 50% of the quota (or whatever ratio you want). Each profile will have a separate license history for tracking purposes.
    - Note that the application configuration is also tied to the profile. If you have two separate brands, each with its own quota, you will need to create two server-to-server integrations (see *Developer Console workflow*, below). The application and logic could remain the same, but you would typically have two configuration settings for the different sites.
    - Managing product profiles on Admin Console: https://helpx.adobe.com/enterprise/using/manage-product-profiles.html

Developer Console workflow

From this point forward, it is assumed that the developer team will take over the remaining tasks. The lead developer will use the Developer Console to create one or more server-to-server integrations,[1] each of which will generate a unique API key and credentials used to generate authentication tokens.

Before you begin

To simplify testing, it's recommended that you also consider installing these additional tools.

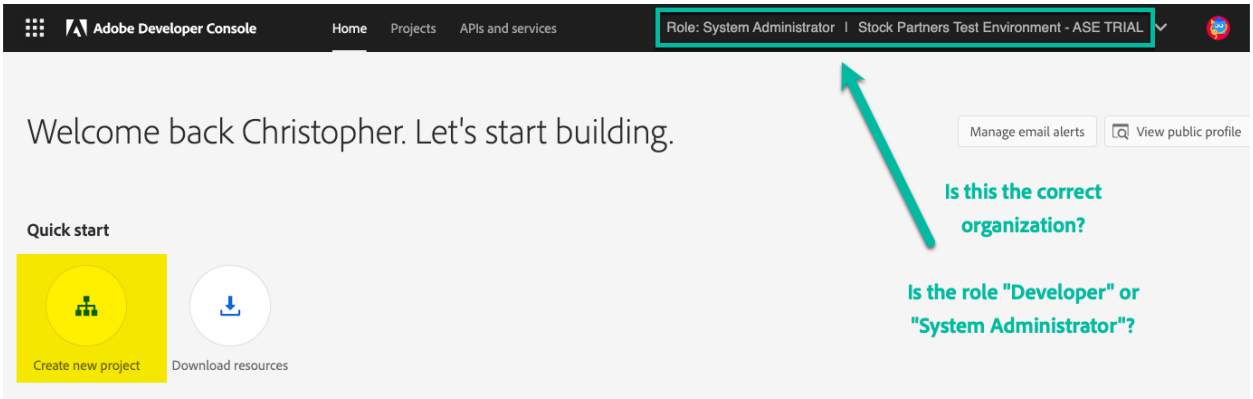
- **Postman**. Postman is a free application available in a standalone version and as a desktop app. It allows you to test API requests, attaching required headers and parameters, and save your requests for future testing. Also, it can export requests in different formats, including cURL, PHP and JavaScript code.
    - https://www.getpostman.com
- **Secure web server.** As you begin to test and integrate the API into your main application, you will need an environment able to send and receive secure (HTTPS) requests. For local testing, you can use a self-signed certificate (see instructions under *Authentication workflow*, below) that you trust on your machine. You can either run a full-fledged local server using Apache, or it is easy to create a simple one using Node.js or Python.
    - Apache (use alone, or with PHP or Apache Tomcat): https://www.apache.org/
    - Node JS (use the Express module): https://nodejs.org/en/
    - Python (SimpleHTTPServer module): https://www.python.org/
    - Installing a self-signed certificate.
        - Windows: https://blogs.technet.microsoft.com/sbs/2008/05/08/installing-a-self-signed-certificate-as-a-trusted-root-ca-in-windows-vista/
        - Mac: http://tosbourn.com/getting-os-x-to-trust-self-signed-ssl-certificates/

Creating the integration in Developer Console

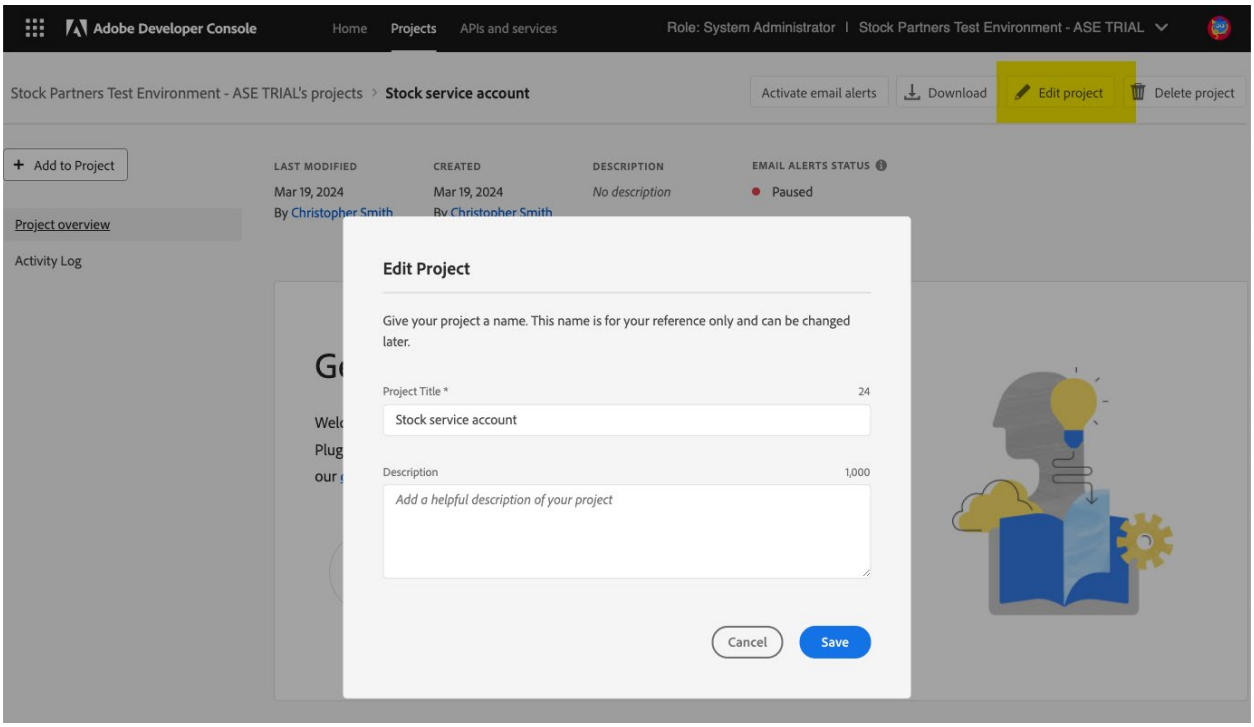
1. Access the Developer Console here: https://developer.adobe.com/console/home.
    - Sign in using either a System Administrator or Developer login created in the Admin Console, above. Note that other types of administrators cannot perform this task.
    - For customers with multiple organizations, the user must sign into the same organization that has a contract for Adobe Stock.

[1] Multiple configurations are necessary if you have more than one product profile configured. Another common reason for creating separate configurations is for development and production. Although both will ultimately point to the same production instance, a new profile with very little quota (e.g., ten or fewer credits) could be created for test purposes.

- If your only organization is your email address (e.g., john-smith@xyz.com), then you do not have sufficient access, and must contact your System Administrator.

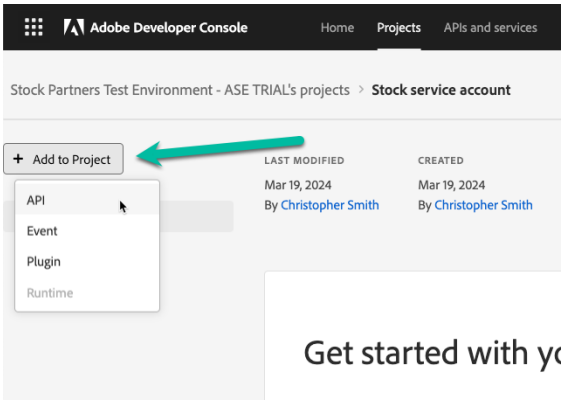


2. Create a new project or open an existing one. Projects are containers that can hold one or more API keys for different API products. By default, the project is given a placeholder name. To rename it, use the **Edit project** link.

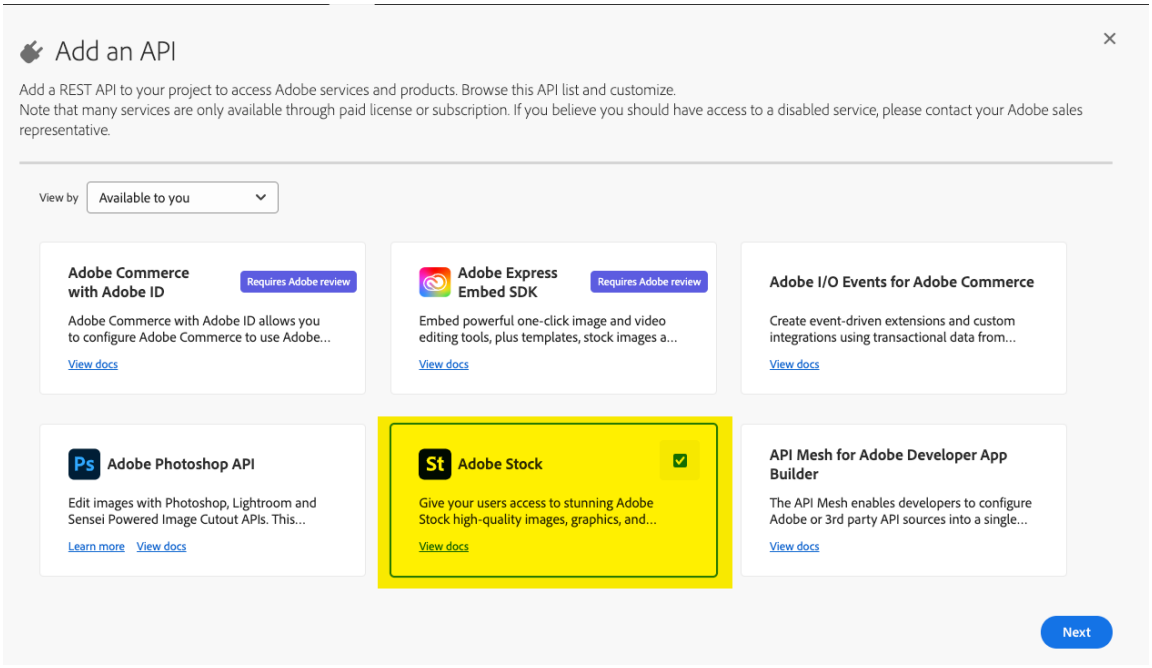


- Docs: https://developer.adobe.com/developer-console/docs/guides/projects/projects-empty

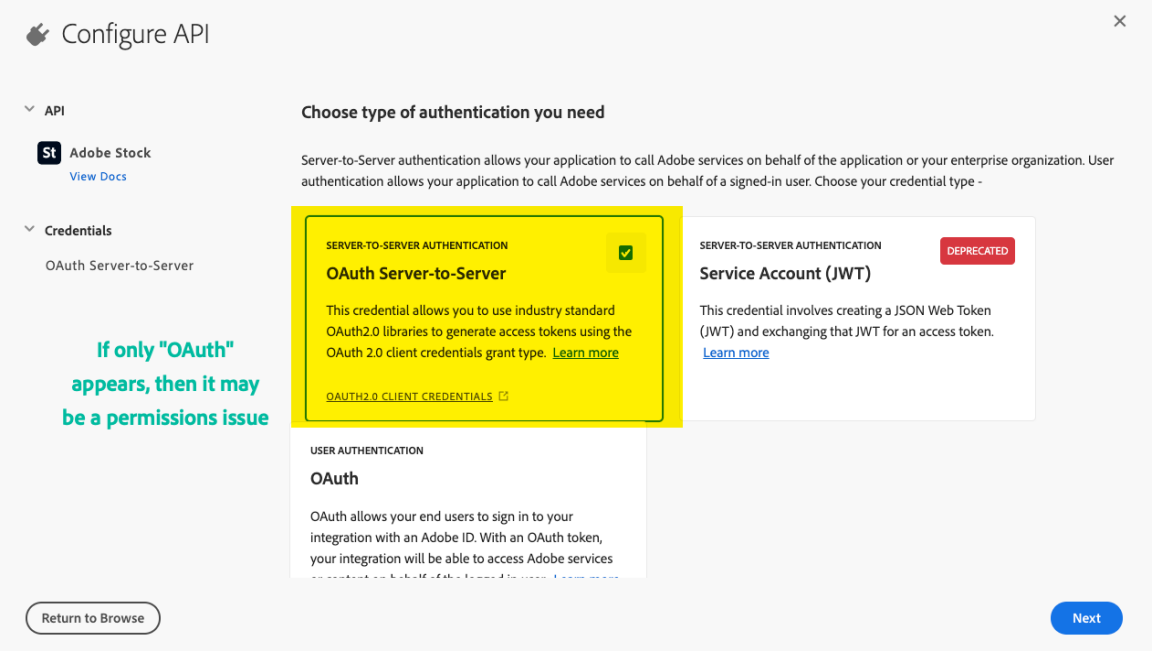
- Select **+ Add to Project > API**.



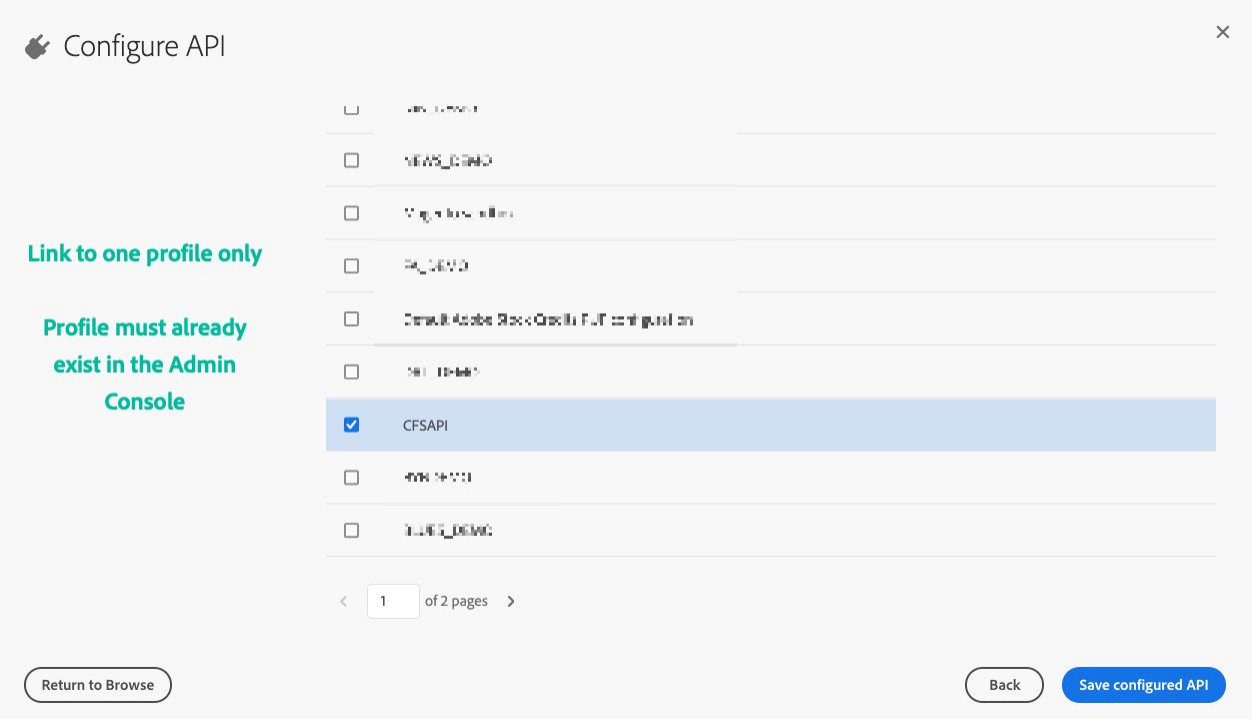
3. Chose **Adobe Stock** and click **Next**.
    - Docs: https://developer.adobe.com/developer-console/docs/guides/services/services-add-api-oauth-s2s



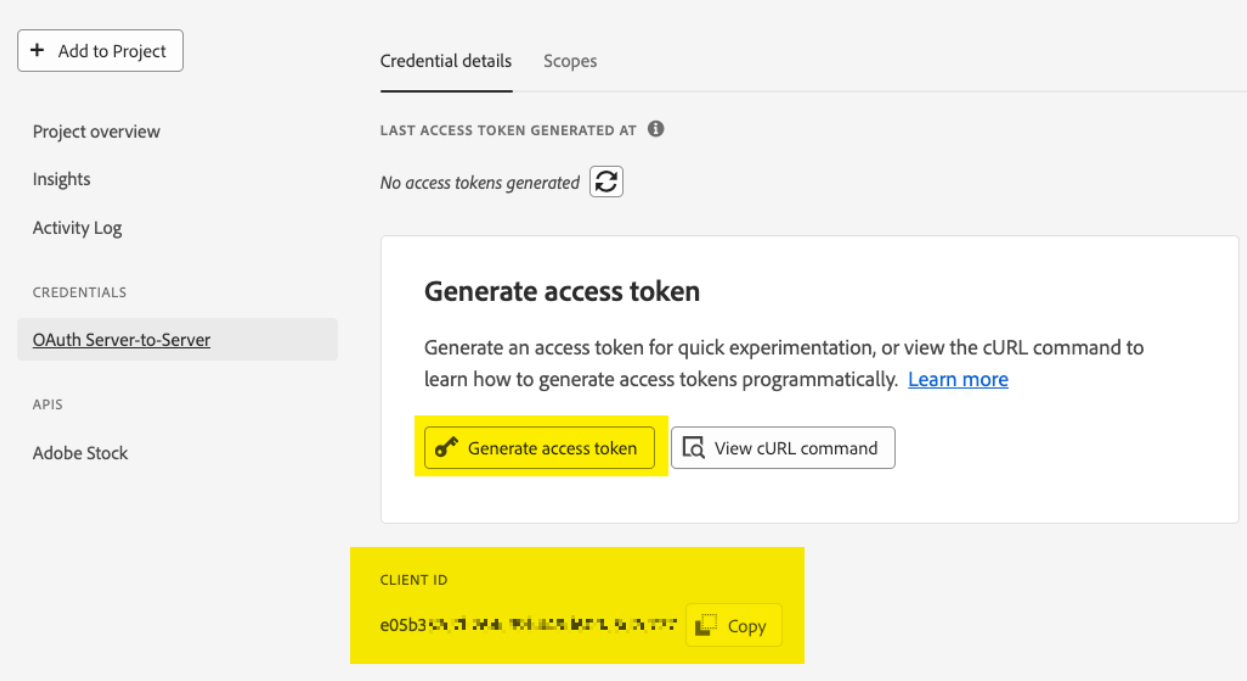
4. Select **OAuth Server to Server** to create a service account and **Next**.
    - If OAuth Server to Server is not available, then you either do not have sufficient permissions, or you are not in the Adobe Stock organization, or you do not have the Stock for Enterprise product.



5. Next you must link this integration to an existing Stock product profile (see *Admin Console overview*, above) created in the Admin Console and click **Save configured API**. Once assigned, an *API credentials* tab will appear in the Admin Console under that profile. This profile will be used for any licensing operations, and its name will appear in the Stock License History dashboard.
    - To learn more about product profiles, see Create/edit a product profile for Adobe Stock. The profile can be changed later in the Developer Console.
    - Even though it is possible, do not link the service account to multiple profiles. Keep in mind that for downloading, every member of a Stock profile can download any asset previously licensed by the organization, unless the admin has prohibited this ability.



6. This opens a screen with your integration details. You can click on **Generate Access Token,** and it will lead you to the credential overview page. This page shows you all details about the credential you just created, including when this credential was last used to generate an access token. Furthermore, it gives you an easy way to generate access tokens, view the cURL command you can use to generate access tokens programmatically, and view scopes per service.
    - To make API calls, you will need the API key (Client ID), and to authenticate your licensing requests, you will need an access token.
    - You may generate an access token on this page, but it will expire in 24 hours. To generate a new token programmatically, use the sample cURL or see the authentication documentation for the Developer Console.
    - Docs: https://developer.adobe.com/developer-console/docs/guides/authentication/ServerToServerAuthentication/#oauth-server-to-server-credential

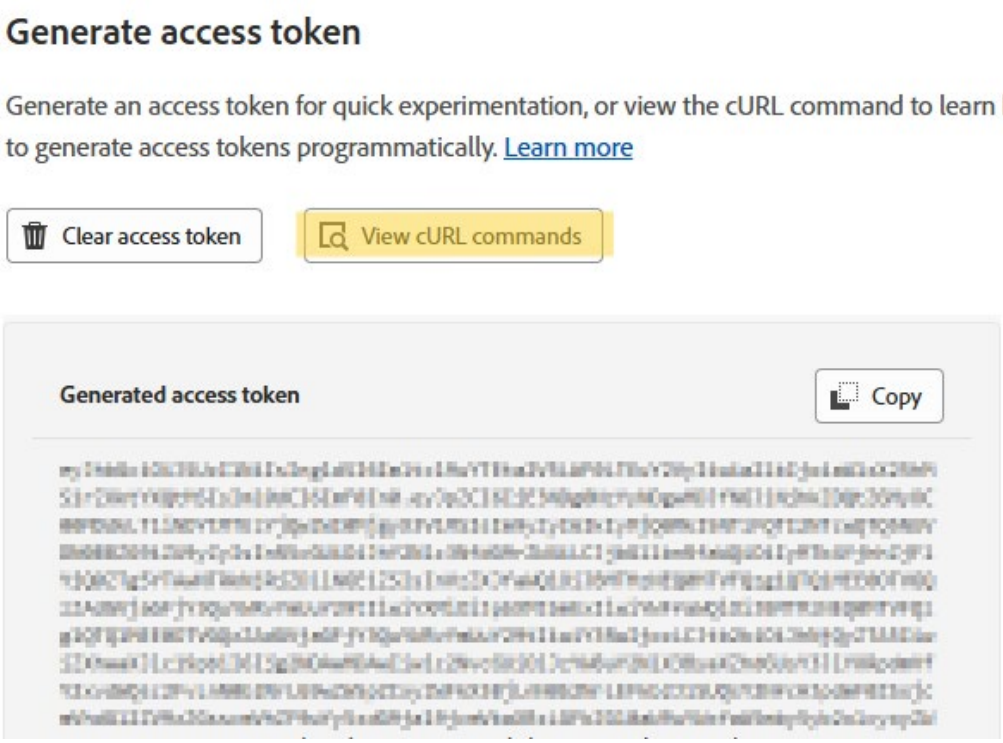


7. Copy the **Generated access token** to a secure location. Unlike the API key, the token identifies the application as belonging to the organization, and can be used to authorize licensing and consumption of credits.
    - API keys are considered confidential, although they are easily replaceable. Store them with other secrets for your applications.
    - Access tokens and client secrets should be stored with the highest security. If they were to leak, a hacker could use them to impersonate your organization and access your Stock downloads.

Authentication workflow

While the Search API only requires an API key, accessing licensing functions requires an access token to authorize those actions. If you only need an access token for testing, you can generate it on demand from the OAuth Server-to-Server tab in your project.

Access tokens expire after 24 hours. This cannot be changed, so the only option is to generate a new token using code or by manually generating. If you click the **View cURL commands** button, it will generate the necessary code.



Example of a cURL that fetches an access token

```
curl -X POST 'https://ims-na1.adobelogin.com/ims/token/v3' -H
'Content-Type: application/x-www-form-urlencoded' -d
'grant_type=client_credentials&client_id={YOUR_CLIENT_ID}&client_sec
ret={YOUR_SECRET}&scope=sao.cce_private,creative_cloud,sao.stock,ope
nid,read_pc.stock,AdobeID,cc_private,creative_sdk,read_pc.stock_cred
its,additional_info.roles,read_organizations'
```

The Adobe identity service will respond with a JSON message that includes your access token.

```
{
  "access_token": "{TOKEN}",
  "token_type": "bearer",
  "expires_in": 863999
}
```

This access token will be used for every licensing operation. While it is optional for search requests, if supplied it will return whether the asset has already been licensed.

The access token has a type “Bearer,” so this is how it would be used in a request to a License API such as License History:

Example of using a Bearer access token

```
GET /Rest/Libraries/1/Member/LicenseHistory HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer {ACCESS_TOKEN}
```

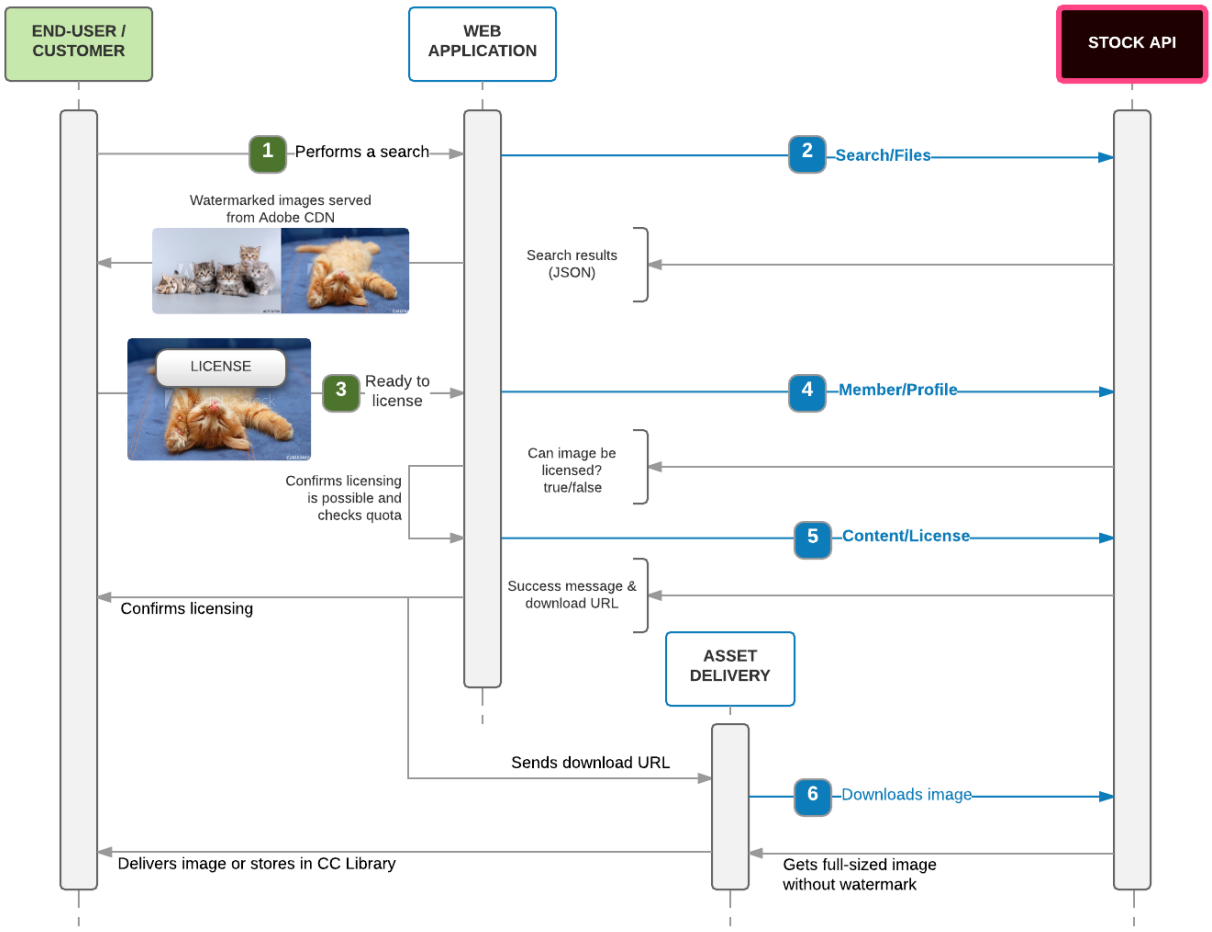
Application flow details

The application will make and receive RESTful calls between the end-user and Adobe Stock, including licensing and download of assets. Since the Stock API is unaware of the end-users in this flow, asset licenses will be granted to the application (technically, to the parent organization), and credits will be deducted from the quota. For a discussion of quotas, see *Adobe Admin Console overview*, above.

The diagram below illustrates the overall process, with requests to the Stock API colored in blue. In each case, there are some mandatory parameters or headers that must be supplied, such as the access token that is sent with the License API requests and download URL for the full-sized asset.

In summary, the typical flow is:

- **Search**. Call the Search API and get back results. This call can either be made with authentication or not—i.e., using the bearer token generated in the *Authentication workflow* section, above—with the difference that authenticated search requests can also return whether the asset is already licensed. This could be used to display a "licensed" flag or badge on the asset.
- **Get profile**. Gets the current remaining quota and confirms that the asset can be licensed.
- **License**. Gets a license for the asset and deducts credit.
- **Download**. Delivers the asset itself to the user or for a fulfillment process.



Search

1. The workflow begins when the end-user (or customer) performs a search on the partner's site for Adobe Stock assets.
2. The web application submits a **Search/Files** request to the Stock API.

- Endpoint
  - `https://stock.adobe.io/Rest/Media/1/Search/Files`
- Required headers
  - `x-Product` : arbitrary string identifying your app
  - `x-api-key` : generated when you created the integration on I/O Console

- Parameters [1]
  - The API can accept multiple search parameters and result filters. At minimum, search requires at least one `search_parameters[]` command.
  - Search results will return a maximum of 100 assets per request. For pagination, see https://www.adobe.io/apis/creativecloud/stock/docs/api/search.html#paginate.

Search/Files request

```
GET /Rest/Media/1/Search/Files?locale=en-US
&search_parameters[words]=kittens HTTP/1.1
Host: stock.adobe.io
X-Product: CFSTest/0.1
x-api-key: ...0e3f
```

- Response: Stock API responds with a JSON object containing the number of results, and an array of metadata for each file.
  - The API limits the number of files returned in the search, therefore to get more results, you must use the pagination mechanism when requesting subsequent pages.
  - Some of the most important metadata returned is the title of each asset, the Stock content/media ID, and URLs to watermarked previews and thumbnails.

[1] For more information on search parameters and filtering results, see https://developer.adobe.com/stock/docs/api/11-search-reference/.

Search/Files response[1] (truncated to show one file result)


```
{
  "nb_results": 247038,
  "files": [
    {
      "id": 75950374,
      "title": "five kittens",
      "width": 2500,
      "height": 1667,
      "creator_name": "adyafoto",
      "creator_id": 205216144,
      "thumbnail_url":
"https://as2.ftcdn.net/jpg/00/75/95/03/500_F_75950374_yNANaKsbx7oLzG
JUFSzXW7j5cGoiKDT9.jpg",
      "thumbnail_html_tag": "<img
src=\"https://as2.ftcdn.net/jpg/00/75/95/03/500_F_75950374_yNANaKsbx
7oLzGJUFSzXW7j5cGoiKDT9.jpg\" alt=\"five kittens\" title=\"Photo:
five kittens\" zoom_ratio=\"1.25\" zoom_depth_max=\"2\" />",
      "thumbnail_width": 500,
      "thumbnail_height": 334,
      "media_type_id": 1,
      "vector_type": null,
      "content_type": "image/jpeg",
      "category": {
        "id": 44,
        "name": "Cats"
      },
      "premium_level_id": 0
    }, { ... },
]}
```


3. The web application parses the results and shows thumbnails for each asset and any other data (such as keywords) that may help the user refine their search. Some filters are necessary from a business perspective. For example, if photos are only being offered, there is a parameter to restrict search only to images and not return vector illustrations or videos. Similarly, if the application cannot sell Premium content for contractual reasons, there is a filter to restrict the search to Standard items only.

Get member profile

4. When the end-user is ready to get an unwatermarked version of the asset (whether it will be used directly by the user, or on behalf of the user on a printable good), the asset must be licensed from Adobe Stock. The application submits a **Member/Profile** request to the Stock API with the content ID of the asset. The purpose is to confirm it is possible to license this asset and perhaps get the total number of credits available. Note that when using a server-to-server integration, the "profile" request does not get the profile of the end-user, but of the product profile. The quota returned corresponds to the linked product profile in the Admin Console.

- Endpoint
  - `https://stock.adobe.io/Rest/Libraries/1/Member/Profile`
- Required headers [1]
  - Same headers as Search requests.
  - `Authentication` : **`Bearer`** access token generated by Adobe's identity service. This is a long base-64 encoded string representation. See *Authentication workflow*, above.
- Parameters
  - `content_id` : Asset's unique identifier, obtained from a search response's `id` attribute.
  - `license` : Stock licensing state for the asset (defaults to **`Standard`**.)
  - `locale` : Default is **`en_US`**.

Member/Profile request

```
GET /Rest/Libraries/1/Member/Profile?content_id=75428610
&license=Standard&locale=en_US HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

- Response: The Stock API returns JSON indicating the remaining credits, and whether it is possible to license the asset.
  - It should normally be *possible* to license an asset, if that type is allowed by the permissions in the product profile and there are available credits. As mentioned earlier, if the contract or profile does not permit licensing Premium content but the application did not filter out these assets, then it may *not* be possible to license the asset. The `"state": "possible"` field indicates that this asset *can* be licensed.
  - If `"requires_checkout"` is `true`, that means the asset cannot be licensed using the API method, and must be licensed via the Stock website.
  - These issues are easiest to troubleshoot using the Adobe Stock website because it should report a helpful message to the user. The user should check that they are in the correct product profile, locate the asset, and attempt to license it. If it is possible via the website but not the API, then contact Adobe Support (via the Admin Console) to raise a ticket for the Stock API.

[1] For more information on headers and parameters for license requests, see https://developer.adobe.com/stock/docs/getting-started/apps/06-licensing-assets/.

Member/Profile response


```
{
    "available_entitlement": {
        "quota": 100,
        "license_type_id": 16,
        "has_credit_model": true,
        "has_agency_model": false,
        "is_cce": true,
        "full_entitlement_quota": {
            "credits_quota": 100
        }
    },
    "member": {
        "stock_id": 41003529
    },
    "purchase_options": {
        "state": "possible",
        "requires_checkout": false,
        "message": "This will use 1 of your 100 credits."
    }
}
```


License

5. The application now sends a **Content/License** request, which will license the asset and deduct credits. Calling this URL again on the same asset will not trigger a license action but will deliver the URL, unless the application fits into a specific retail use case (see below).

- Endpoint
  - `https://stock.adobe.io/Rest/Libraries/1/Content/License`
- Required headers: Same as **Member/Profile**.
- Parameters: Same as **Member/Profile**, unless the contract or licensing stipulates that the file must be re-licensed for multiple uses. This is common in the Print on Demand use case, where the retailer may need to license the asset again for each additional customer or additional print application. In this case, an optional parameter is added:
  - `license_again` : **`true`** | **`false`**[1]

[1] In most cases, setting the value to `false` is the same as not setting it at all.

Content/License request

```
GET /Rest/Libraries/1/Content/License?content_id=75950374
&license=Standard HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

- Response: Stock API responds with a JSON object indicating that licensing was successful and the updated quota. Among the information returned is the **URL to download the full asset**, which the web application will store in its database and/or forward to its fulfillment/manufacturing division.

Content/License response (partial)

```
{
    "available_entitlement": {
        "quota": 79, ...
        "full_entitlement_quota": {
            "image_quota": 80,
            "credits_quota": 34
        }
    },
    "contents": {
        "75950374": {
            "content_id": "75950374",
            "size": "Comp",
            "purchase_details": {
                "state": "just_purchased",
                "license": "Standard",
                "date": "2017-09-05 23:47:14",
                "url":
"https://stock.adobe.com/Rest/Libraries/Download/75950374/1",
                "content_type": "image/jpeg",
                "width": 2500,
                "height": 1667
            },...
        }
    }
}
```

Download

6. To finish the workflow, the application or fulfillment team will download the asset without a watermark. In the previous response returned from the Content/License request (above), the download URL is highlighted.

- Endpoint:

    - `https://stock.adobe.io/Rest/Libraries/Download/{content_id}/1`
  - Required headers: Same as **Member/Profile**.
  - Parameters: None.
  - Response: Stock server will respond with a 200 and/or 302 response, depending on the type of asset. Files will either be delivered directly from the URL, or the server will issue a redirect to a download destination, which the application or browser must follow to receive the file.

Download request

```
GET /Rest/Libraries/Download/75950374/1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

Note that the access token does not need to be the same one generated during the purchase flow; it simply needs to contain the same credentials. Also, the licensing and download do not need to happen simultaneously, and once licensed, the file may be re-downloaded without penalty.

Additional workflows

In addition to the Search and License workflow, Enterprise customers commonly use the Stock API to fetch their download license history. For a complete discussion of this topic, see the

- Get license history (API method).
  - Endpoint
    - `https://stock.adobe.io/Rest/Libraries/1/Member/LicenseHistory`
  - Required headers: Same as **Member/Profile**.
  - Parameters: None

Member/LicenseHistory request

```
GET /Rest/Libraries/1/Member/LicenseHistory HTTP/1.1
Host: stock.adobe.io
X-Product: CFS_Test_1.0
x-api-key: ...0e3f
Authorization: Bearer eyJ4NX...ztcSQ
```

  - Response: The returned JSON response will resemble a Search API response, where you will receive an array of file metadata.

Member/LicenseHistory response


```
{
  "nb_results": 16,
  "files": [
    {
      "license": "Standard",
      "license_date": "6/19/17, 5:32 PM",
      "download_url":
"https://stock.adobe.com/Download/DownloadFileDirectly/mTFwPEzGtGYQN
Gpm9SnEzAtqlF4hHKTu",
      "id": 24683483,
      "title": "kitten sleeps on the back",...
      "content_type": "image/jpeg",
      "height": 1805,
      "width": 2715
    },
    { ... }
  ]
}
```


References

- Adobe Stock API getting started and reference documentation: https://developer.adobe.com/stock/docs/getting-started/
- Workflow guides:
  - Search: https://developer.adobe.com/stock/docs/getting-started/apps/05-search-for-assets/
  - Licensing: https://developer.adobe.com/stock/docs/getting-started/apps/06-licensing-assets/
- Technical FAQ: https://developer.adobe.com/stock/docs/faq/
- Integrating Adobe Stock with an Enterprise DAM: https://developer.adobe.com/stock/docs/Stock-DAM-integrations.pdf
- API technical support: Email stockapis@adobe.com

Adobe

Adobe Systems Incorporated
345 Park Avenue
San Jose, CA 95110-2704
USA
www.adobe.com

5/15