



Plex API for Developers

PLEX SYSTEMS, INC. PRIVILEGED AND CONFIDENTIAL INFORMATION.

Copyright ©2017. Plex Systems, Inc. All Rights Reserved.

PERMISSION TO COPY, MODIFY, AND/OR DISTRIBUTE THIS DOCUMENT WITHOUT THE PRIOR WRITTEN CONSENT OF PLEX SYSTEMS, INC. IS STRICTLY PROHIBITED. NO PART OF THIS DOCUMENT MAY BE REPRODUCED, STORED IN A RETRIEVAL SYSTEM, DISSEMINATED, OR TRANSMITTED IN ANY FORM OR BY ANY MEANS, ELECTRONIC, MECHANICAL, PHOTOCOPYING, RECORDING, OR OTHERWISE, WITHOUT THE PRIOR WRITTEN PERMISSION OF PLEX SYSTEMS, INC.

Contents

PLEX API FOR DEVELOPERS	1
API development process	1
Plex API terms and concepts	2
About REST web services	2
API URL structure	2
Request headers	2
HAL-formatted responses	4
UTC time	4
Plex API terms.....	5
Getting started with the Plex API	9
Create a Plex service account	9
Obtain a client ID and secret.....	10
Sign up for a Plex API developer account	11
Test your account.....	11
Navigating the Plex Developer Portal.....	15
Sign in to the Plex Developer Portal.....	15
View announcements	16
View and test an API	16
View and subscribe to an adapter	18
Access connectors	19
Access your profile	19
Using the Plex Workcenter Lookup app	20
Plex Workcenter Lookup app requirements	20
Add the Plex Workcenter Lookup app.....	21
Run the Plex Workcenter Lookup app	21
Troubleshooting.....	24
400 - Bad request	24
401 - Expired token.....	24
401 - Invalid client secret.....	25
403 - Access restricted resource	25
403 - Failed to get authorization token	26
403 - Usage Quota	26
404 - Resource not found	26
429 - Usage limiting.....	27
500 - Unknown error.....	27

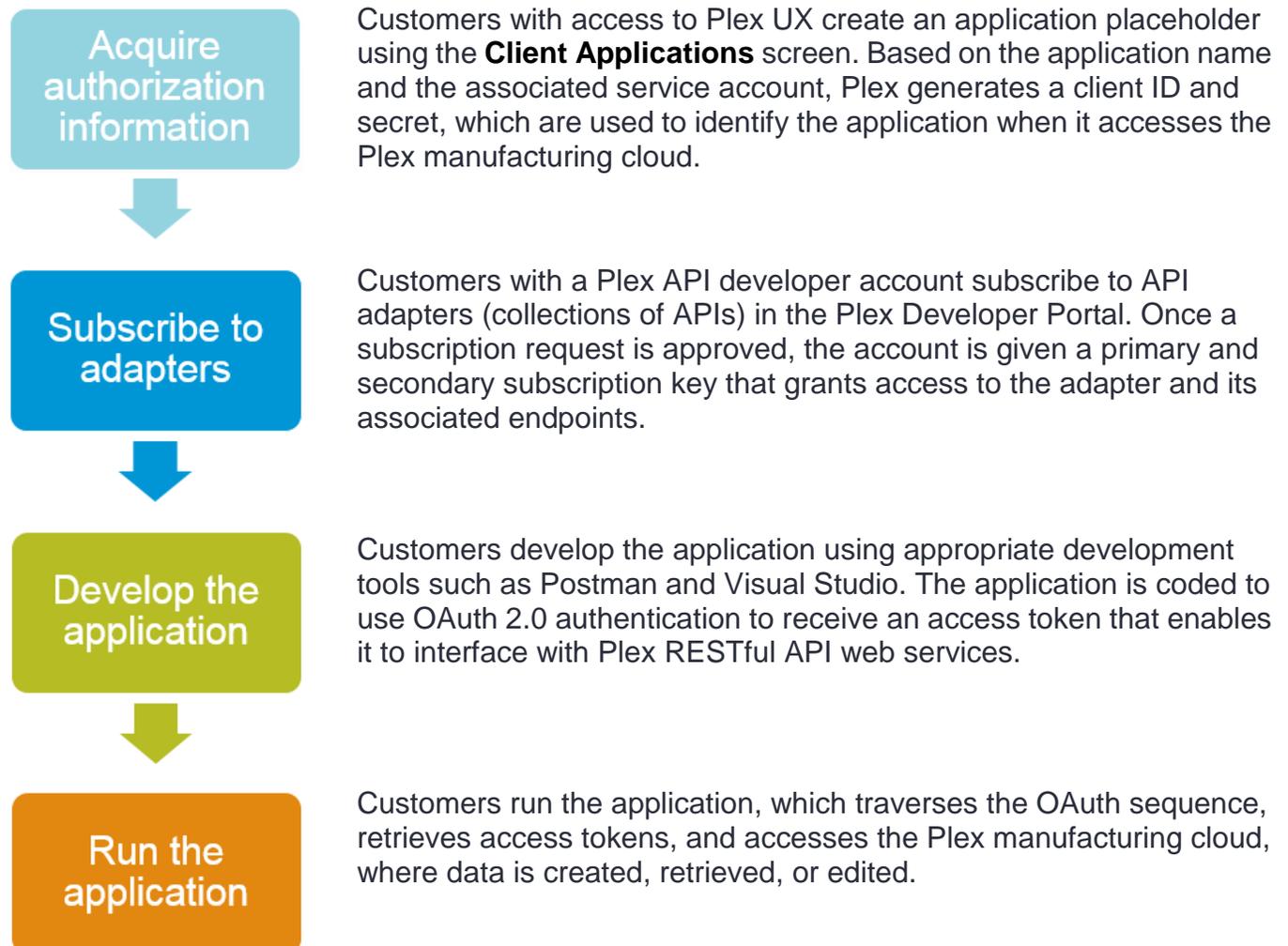
Plex API for Developers

The Plex RESTful web service application programming interface (API) is a collection of public endpoints that provide access to resources and data in the Plex manufacturing cloud. Designed according to the representational state transfer (REST) software architectural style, the Plex API uses standard HTTP verbs and a RESTful endpoint structure. Authorization is handled using OAuth 2.0.

You can access the Plex Developer Portal at <https://dev.plex.com/>.

API DEVELOPMENT PROCESS

The high level process for application development in the Plex RESTful application framework typically includes the following tasks:



PLEX API TERMS AND CONCEPTS

This section defines fundamental terms and concepts for the Plex API.

About REST web services

Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induce desirable properties that enable services to work best on the web, such as performance, scalability, and modifiability.

Key points about REST include:

- REST is a web service like SOAP but arguably better.
- Plex implements REST request and response messages as Hypertext Application Language (HAL) and JavaScript Object Notation (JSON).
- REST uses self-describing messages.
- REST provides stateful interactions through hyperlinks.

API URL structure

Managed endpoints have the following form:

```
https://api.plex.com.net[/test/]{web API URL suffix}/{version}/items[/]{ItemKey}]?[filters]
```

Example:

```
https://api.plex.com.net/test/Production/v1/WorkcenterProductionLogs/  
{WorkcenterKey}?EndDate={EndDate}&StartDate={StartDate}
```

Request headers

Subscription key header

All Plex API requests require a valid subscription key header. The request header **Ocp-Apim-Subscription-Key** is a subscription key which provides access to the API. These subscription keys are located in your account profile on the Plex Developer Portal.

If you neglect to include the subscription key header in an API request, you will receive the following response:

```
{  
  "statusCode": 401,  
  "message": "Access denied due to missing subscription key. Make sure to include  
  subscription key when making requests to an API."  
}
```

If your header has an invalid subscription key, you will receive the following response:

```
{
  "statusCode": 401,
  "message": "Access denied due to invalid subscription key. Make sure to provide
  a valid key for an active subscription."
}
```

Bearer token header

All Plex API requests require a valid bearer token in the header. The OAuth2 bearer token is obtained from the Authorization API, which is the OAuth2 token endpoint. Note this resource requires your Plex-provided client ID and the URL-encoded form of your Plex-provided client secret.

The bearer token header consists of two related values: **Authorization** and **Bearer [your bearer token]**.

If you neglect to include the authorization header, you will receive the following response:

```
{
  "statusCode": 401,
  "message": "Not authorized: Authorization header value not present."
}
```

If you neglect to include the bearer token, you will receive the following response:

```
{
  "statusCode": 401,
  "message": "Unauthorized. Access token is missing or invalid. Verify the token
  is in the Authorization Header with the token type 'Bearer'. Make certain the
  token is not expired. Use an inspection tool such as https://jwt.io/ and verify
  all expected claims are present, including exp, iat, nbf, etc. Make sure the
  token was requested from the Plex token endpoint with resource
  http://api.plex.com/."
}
```

HAL-formatted responses

The typical JSON response is consistently delivered in a simple format that helps to make the Plex API easy to explore and learn.

The basic format of the response with the generic application/hal+json media type is the following:

- Links
- Embedded returned resources
- State information (instance record count and total record count)

Example:

```
{
  "links": [
    {
      "rel": "self",
      "href": "../...",
      "method": "GET"
    },
    {
      "rel": "next",
      "href": "../...&offset=10&limit=10",
      "method": "GET"
    },
    {
      "rel": "last",
      "href": "../...&offset=10&limit=10",
      "method": "GET"
    }
  ],
  "embedded": [
    {
      FIRST RETURNED OBJECT
    },
    {
      SECOND RETURNED OBJECT
    },
    ...,
    {
      NTH RETURNED OBJECT
    }
  ],
  "total": ##,
  "count": ##
}
```

UTC time

Plex API responses return all dates in Coordinated Universal Time (UTC). Client applications must take this into account.

Plex API terms

adapter

An adapter is a set of related Plex Connect APIs. You must subscribe to an adapter before you can access it.

adapter subscription

High level access to Plex Connect adapters is managed using a subscription system. A confirmed subscription to an adapter grants access to the adapter and is used as an identifier for analytics, auditing, and any usage constraints.

API

An Application Programming Interface (API) is a set of routines, protocols, and tools for building applications. A Plex API in the Plex Developer Portal is a collection of related endpoints analogous to one or more Plex software modules.

authorization code grant

An OAuth 2.0 authentication flow where access is delegated to a client application. The transaction is protected and mediated by a code grant, which is exchanged for an access token. The client application never sees the user's credentials and the user agent never sees the access token.

authorization header

With OAuth 2.0 authorization over HTTP, the client application uses the HTTP header to submit a bearer token to the API.

bearer token

A bearer token is a unique string that is used to access OAuth 2.0 protected resources.

client credentials grant

A form of authentication known as 2-legged OAuth. Instead of user names and passwords, access tokens are used to gain access to PCN data. The OAuth 2.0 client credentials grant flow permits an unattended, confidential client to use its own credentials to authenticate when calling another web service, instead of impersonating a user. In this scenario, the client is typically a middle-tier web service, daemon service, or web site.

business process API

These practical APIs have usability characteristics that distinguish them from infrastructure-level system APIs, such as requiring minimal programming to be usable by a broader range of developers, including HTML designers and higher-level coders

client ID

A client ID is a unique string, typically a machine-generated Globally Unique Identifier (GUID), which represents a registered client application.

client secret

A client secret is a unique string that is used as a password for a registered client application when the client application is on a confidential client. The client secret has an expiration date and should be kept secure.

connector

A connector is a finished product that relies on one or more Plex Connect APIs. Connectors are pre-built and certified integrations with solution alliance partners, where both solutions are integrated in a manner that is within the scope of an alliance agreement. Connectors are supported by Plex and are licensable in conjunction with integrating solutions into Plex.

Developer Portal

The Plex Developer Portal is the site where developers can learn about and experiment with Plex APIs. The portal also provides developers with the opportunity to collaborate and share their experiences with the Plex API.

endpoint

The “endpoint” of the API is a unique URL that represents a resource.

Hypertext Application Language (HAL)

The Hypertext Application Language (HAL) is an Internet Draft standard convention that defines links to resources within JSON. HAL imposes a structured way to represent resource links in some of our standard data as part of the HATEOAS constraint of the REST application architecture.

JSON response

JSON (JavaScript Object Notation) is an open-standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs. Response bodies from the API are in JSON format.

operation

The combination of an endpoint with an HTTP verb.

primary key

In Plex normalized database tables, a unique, “primary” key of one or more attributes identifies each data row.

profile

The set of subscriptions, identifiers, and other attributes of a Plex Developer Portal user.

resource

A resource is an object with attributes, relationships to other resources, and a set of methods that operate on it. It is similar to an object instance in an object-oriented programming language, with the important difference that only a few standard methods are defined for the resource (corresponding to the standard HTTP GET, POST, PUT and DELETE methods), while an object instance typically has many methods.

REST

Representational state transfer (REST) is a software architectural style that focuses on component roles and a specific set of interactions between data elements rather than implementation details. Web services that conform to REST constraints are known as *RESTful*.

REST systems typically communicate over HTTP using the same HTTP verbs that web browsers use to retrieve web pages and to send data to remote servers.

RESTful verbs include:

GET <resource>

PUT <resource>

POST <resource>

DELETE <resource>

secondary key

An additional user-controlled subscription key.

service account

For the purposes of 2-legged OAuth 2.0 (2LO), a service account is a type of Plex user account.

sprocs

Structured Query Language (SQL) stored procedures. A stored procedure is an application subroutine that is used to interact with a database.

system API

Service APIs provide internal interfaces into specific, low-level capabilities such as database, messaging, and storage.

token

In the OAuth authorization standard, tokens are signed strings that are used to validate access authorization.

token endpoint

The token endpoint is the URL from which a proper request returns a token.

GETTING STARTED WITH THE PLEX API

This section contains information on the steps necessary to obtain access to the Plex API for use with custom applications.

Create a Plex service account

You must create a Plex service account to associate with Plex API applications that use the client credentials grant.

1. In Plex UX, create a new user account:
 - a. Access the **User Manager**.
 - b. On the **Action Bar**, click **Add**.
 - c. On the **New User** screen, provide the following information, at a minimum:
 - **First Name**
 - **Last Name**
 - **User ID**
 - **Password**
 - **Position**
 - **Department**
 - d. In the **Security Roles** section, select the **Member** check box for each security role that you want to assign to the service account.

Member	Admin	Role
<input type="checkbox"/>	<input type="checkbox"/>	Accounting - Chart of Accounts
<input type="checkbox"/>	<input type="checkbox"/>	Accounting
<input type="checkbox"/>	<input type="checkbox"/>	Buyer
<input type="checkbox"/>	<input type="checkbox"/>	Community Create
<input type="checkbox"/>	<input type="checkbox"/>	Engineer
<input type="checkbox"/>	<input type="checkbox"/>	Everyone

Note: The service account must have sufficient security privileges for the API applications that are associated with the account to interact with related areas of the system. For example, if you have a custom application that uses the API to list part records, you must assign to the service account the security roles that are used by your organization to control access to appropriate actions from the Part List module.

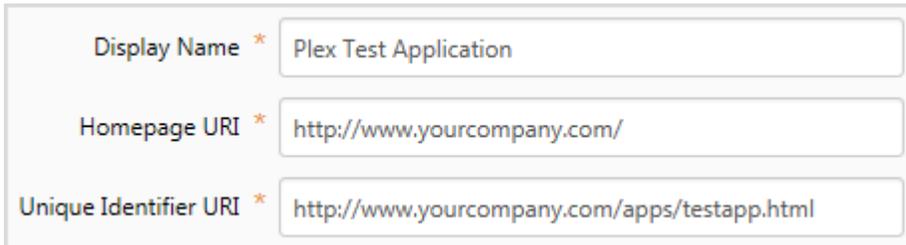
- e. Click **OK**.
The service account is created.

2. Give the user account access to Plex UX:
 - a. Access the **Cloud Access** screen.
 - b. Adjust the search filters to locate the user account, and click **Search**.
 - c. In the search results, in the **Name** column, click the account name.
 - d. On the **User Access Setup Detail** screen, for **Default Customer**, specify the PCN.
 - e. Click **OK**.

Obtain a client ID and secret

You must register your application with Plex to receive a client ID and secret. The client ID and secret are used to obtain an OAuth 2.0 bearer token, which your application will use to access the Plex manufacturing cloud. A client ID and secret are also required to try out Plex API operations within the Plex Developer Portal.

1. In Plex UX, access the **Client Applications** screen.
2. On the **Action Bar**, click **Add**.
3. On the **Client Applications** form, for **Display Name**, type your application name.
4. In the **Homepage URI** box, type the full address of your application.
5. In the **Unique Identifier URI** box, type an identifier.



The screenshot shows a form with three input fields, each with a red asterisk indicating it is required. The first field is labeled 'Display Name' and contains the text 'Plex Test Application'. The second field is labeled 'Homepage URI' and contains the text 'http://www.yourcompany.com/'. The third field is labeled 'Unique Identifier URI' and contains the text 'http://www.yourcompany.com/apps/testapp.html'.

6. If your account is associated with multiple PCNs, select the PCN that you want to use.
7. In the **PUN** box, type the Plex User ID for the account.
8. If you want to limit the account to test tier data, select the **Test Only** check box.

Note: With **Test Only** turned on, calls to managed production endpoints return a response stating that the application is not allowed to access production resources.

9. Click **Apply**.

The client ID is generated for your application and the screen updates to display the information. This can take several minutes to complete.

10. In the **Client Credentials** section, click **Secrets Grid**.
11. On the **Client Applications Secrets** screen, on the **Action Bar**, click **Add**.
On the **Client Applications Add Secret** screen, a generated secret is displayed.
12. Record the secret and keep it in a secure place.

Important: Once you click **OK** and navigate away from the **Client Applications Add Secret** screen, you cannot retrieve the secret.

13. Click **OK**.

Sign up for a Plex API developer account

You must have a Plex API developer account before you can access Plex RESTful APIs.

1. In Plex, create a **USR** to request an API developer account. Include the email address that you want to associate with the account.

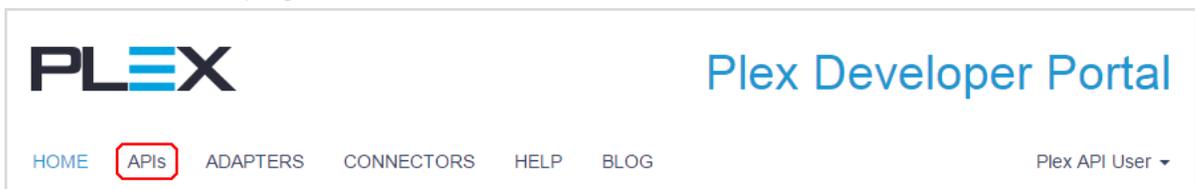
Plex will review your request and send a confirmation email to your email address.

2. In the confirmation email, click the verification link.
3. Update the **USR** and wait for Plex to process your verified user account.

Test your account

If you have a Plex API developer account, you can test Plex APIs directly within the Plex Developer Portal.

1. In Plex UX, *generate a client ID and secret* (see page 10).
2. In the Plex Developer Portal, obtain a bearer token:
 - a. Access the **APIs** page.



- b. In the list of APIs, click **Authorization**.
Details of the Authorization API are displayed.
- c. Click **Try it**.
The page updates to display links and form elements.
- d. In the **Request body**, do the following:
 - For **client_id**, replace “paste_client_id” with your client ID.
 - For **client_secret**, replace “paste_client_secret” with your client secret.

```

1 grant_type=client_credentials
2 &
3   client_id=paste_client_id
4 &
5   client_secret=paste_client_secret
6 &
7   resource=http%3A%2F%2Fapi.plex.com%2F

```

```

1 grant_type=client_credentials
2 &
3   client_id=d04499a8-765b-413a-22cf-37f572d46bac
4 &
5   client_secret=0a71265f-4ecc-474c-9397-be22442b9602
6 &
7   resource=http%3A%2F%2Fapi.plex.com%2F

```



- d. In the first text box, type “Authorization”.

Headers

Ocp-Apim-Subscription-Key

Authorization

- e. In the second text box, type “Bearer [bearer token]”, where [bearer token] is the access token string you obtained earlier.

Headers

Ocp-Apim-Subscription-Key

Authorization

- f. At the bottom of the page, click **Send**.

The API is submitted and the page updates with the response.

```

Pragma: no-cache
Cache-Control: no-cache
Date: Mon, 20 Jun 2016 20:48:08 GMT
Set-Cookie: BIGipServerpmc_rest_webservices_http_prod=1259352586.20480.0000; path=/; Httponly; Secure
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Content-Length: 201
Content-Type: application/json; charset=utf-8
Expires: -1

{
  "links": [
    {
      "rel": "self",
      "href": "/Platform/Login/CompanyProfile",
      "method": "GET"
    }
  ],
  "embedded": [
    {
      "name": "Technical Documentation PCN Automotive",
      "plexusCustomerCode": "Tech Doc Auto"
    }
  ],
  "total": 1,
  "count": 1
}

```

NAVIGATING THE PLEX DEVELOPER PORTAL

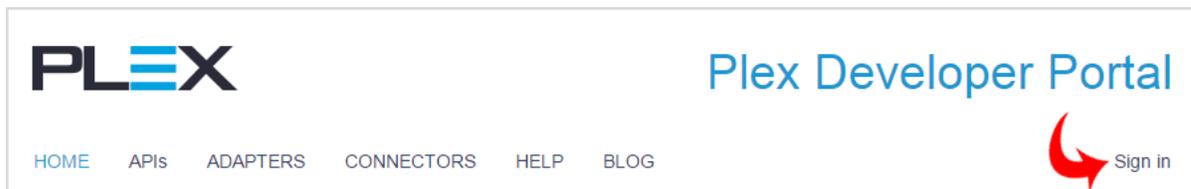
The Plex Developer Portal provides detailed information on using the Plex API in your custom applications, including the following:

- API definitions
- Sample code
- The ability to test each API directly in the portal
- Adapter subscription management
- Access to connectors

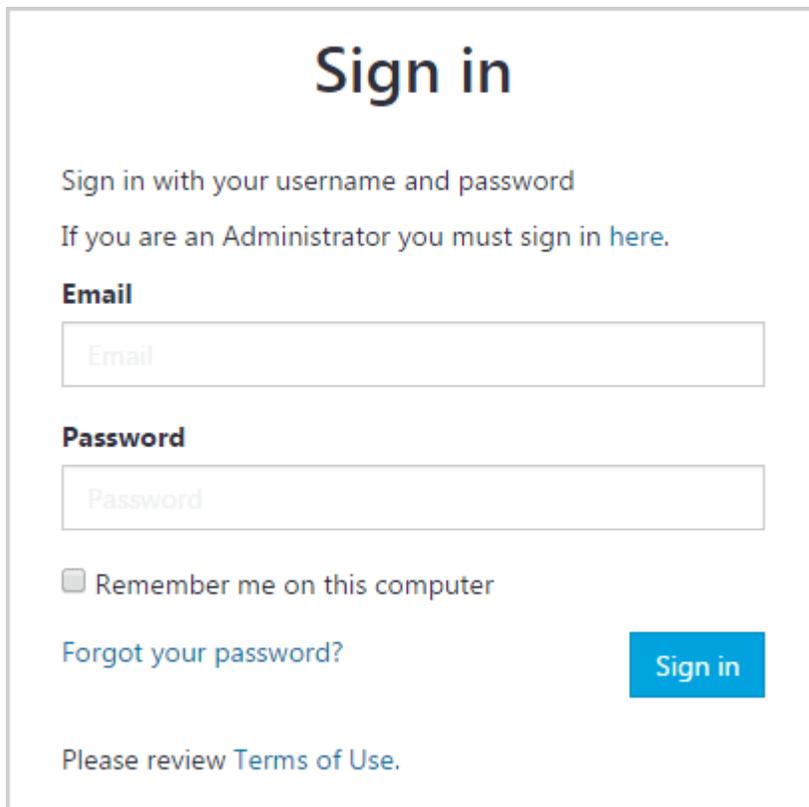
Sign in to the Plex Developer Portal

You must sign in to the Plex Developer Portal to subscribe to API adapters and to access information such as subscription keys, documentation, and the blog.

1. In the upper right corner of any page in the portal, click **Sign in**.



2. Type in your information and log in to the portal.

The image shows a 'Sign in' form. At the top, it says 'Sign in' in a large font. Below that, it says 'Sign in with your username and password' and 'If you are an Administrator you must sign in here.' There are two input fields: 'Email' and 'Password'. Below the 'Password' field is a checkbox labeled 'Remember me on this computer'. To the left of the 'Sign in' button is a link for 'Forgot your password?'. At the bottom left, there is a link to 'Please review Terms of Use.' The 'Sign in' button is a blue rectangle with white text.

View announcements

You can access official announcements on the **Blog** page of the Plex Developer Portal.

1. On any page in the portal, click **BLOG**.

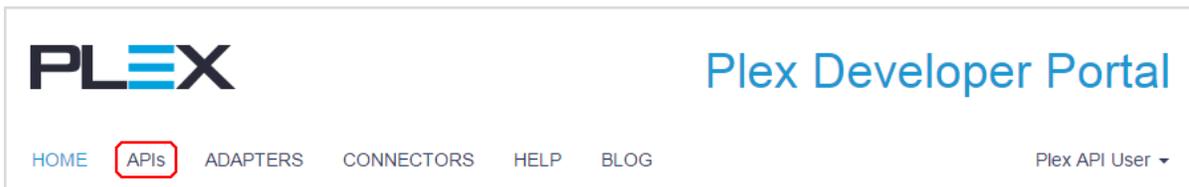


2. On the **Blog** page, click the announcement that you want to view.

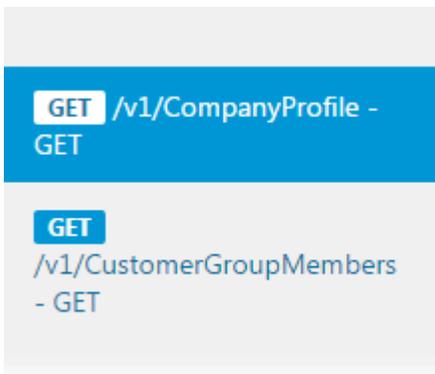
View and test an API

Plex API definitions are accessed from the **APIs** page of the Plex Developer Portal.

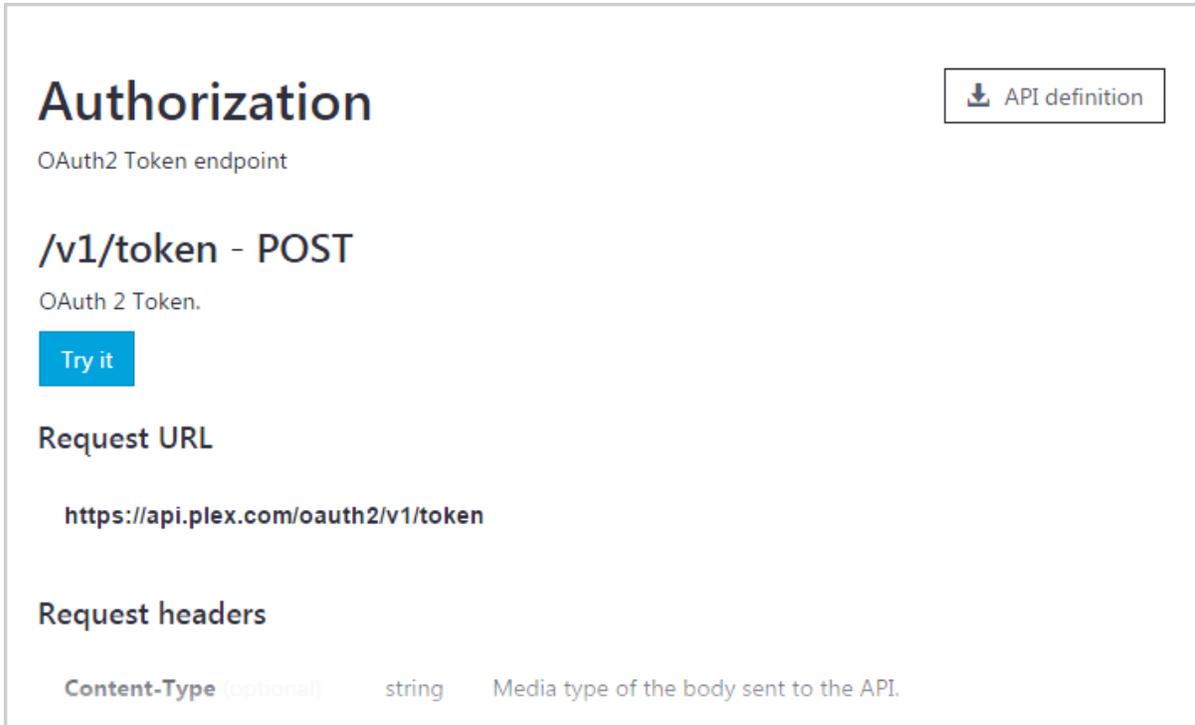
1. On any page of the portal, click **APIs**.



2. If you want to search for an API, in the upper right corner of the **APIs** page, type your search terms and press Enter.
3. Click the name of an API.
The information that is associated with the API is displayed.
4. If the API consists of multiple endpoints, click the endpoint that you want to investigate on the left side of the page.



5. The details of the API appear in the main body of the page. This includes the following:
 - The request URL
 - Request headers
 - Sample responses
 - Code samples



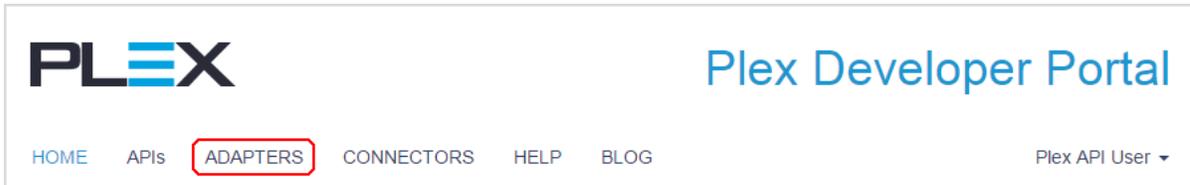
The screenshot shows the API documentation for the **Authorization** endpoint, specifically the **/v1/token - POST** method. The page is titled "Authorization" and is identified as an "OAuth2 Token endpoint". A button labeled "API definition" with a download icon is in the top right corner. Below the endpoint name, it says "OAuth 2 Token." and a blue "Try it" button is visible. The "Request URL" section shows the URL `https://api.plex.com/oauth2/v1/token`. The "Request headers" section lists `Content-Type` as optional, with a type of `string` and a description: "Media type of the body sent to the API."

6. To export the API definition as a Swagger or WDAL file, click **API definition**, and choose the appropriate option.
7. To test the API, click **Try it**.
The page updates to display links and form elements.
Note: You must subscribe to an adapter that contains the API before you can test it.
8. To specify query parameter values or add a new query parameter, do the following:
 - a. In the **Query parameters** section, if there are predefined parameters, type the necessary information in the text boxes.
 - b. To add a new parameter, click **Add parameter**.
 - c. In the text boxes that are displayed, type in the parameter name and value.
9. To specify header values or add a new header, do the following:
 - a. In the **Headers** section, if there are predefined headers, type the necessary information in the text boxes.
 - b. To add a new header, click **Add header**.
 - c. In the text boxes that are displayed, type in the header name and value.
Note: Most APIs require a bearer token for authorization. This is generated using the OAuth2 API.
10. At the bottom of the page, click **Send**.

View and subscribe to an adapter

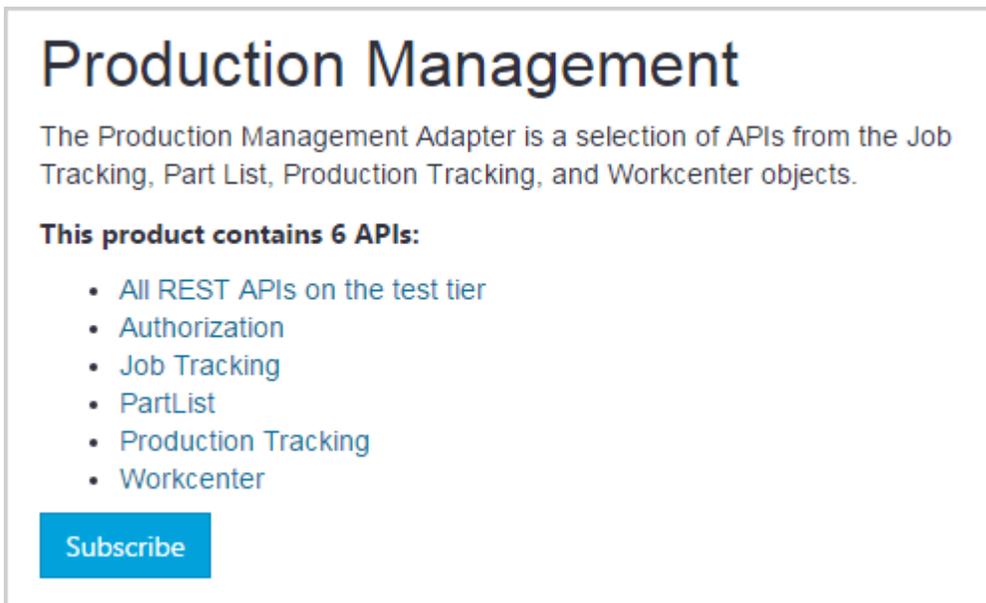
Adapters are collections or related APIs. Some adapters must be subscribed to before they become accessible to API developers.

1. On any page of the portal, click **ADAPTERS**.



2. If you want to search for an adapter, in the upper right corner of the **Adapters** page, type your search terms and press Enter.
3. Click the name of the adapter that you want to access.

A list of the APIs that are associated with the adapter is displayed.

A screenshot of the "Production Management" adapter page. The title "Production Management" is at the top in a large, bold font. Below the title, there is a paragraph: "The Production Management Adapter is a selection of APIs from the Job Tracking, Part List, Production Tracking, and Workcenter objects." Underneath this paragraph, it says "This product contains 6 APIs:" followed by a bulleted list of API names: All REST APIs on the test tier, Authorization, Job Tracking, PartList, Production Tracking, and Workcenter. At the bottom left of the page, there is a blue button labeled "Subscribe".

4. To view the details of an API, click the link to the API.
5. To subscribe to the adapter, click **Subscribe**.

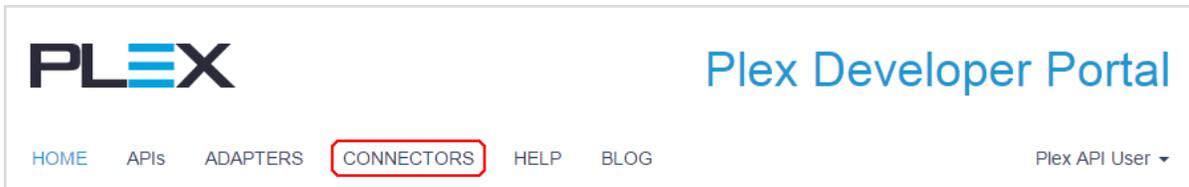
The subscription request is submitted to Plex and you should receive an email acknowledgment. Notification of subscription approval is also sent by email.

After your subscription request is approved, you can access your subscription keys in your Plex Development Portal profile.

Access connectors

Connectors are prebuilt applications created using a Plex integration solution.

1. On any page of the portal, click **CONNECTORS**.

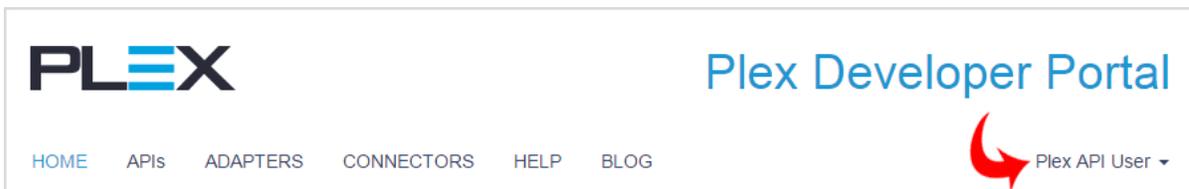


2. On the **Connectors** page, click the name of a connector for information on how to access it.

Access your profile

Your Plex developer account profile includes details such as your email address, active subscriptions, and analytics reports.

1. On the right side of the portal, click your name.



Your profile page is displayed.

2. Do any of the following:

TO	DO THIS
Change your password for logging in to the portal	Click Change password .
Change your email address or the name associated with your account	Click Change account information .
View a list of the adapters that you've subscribed to	Scroll down the page to the Your subscriptions section.
Cancel a subscription	<ol style="list-style-type: none"> 1. Locate the subscription in the Your subscriptions section of the page. 2. Click Cancel.

TO	DO THIS
View the keys associated with a subscription	<ol style="list-style-type: none"> 1. Locate the subscription in the Your subscriptions section of the page. 2. For Primary key or Secondary key, click Show. 3. If you want to generate a new key, click Regenerate.
View analytics about usage of your API account	Click Analytics reports .
Close your API account	<ol style="list-style-type: none"> 1. Locate the subscription in the Your subscriptions section of the page. 2. Scroll down to the bottom of the page. 3. Click Close account.

USING THE PLEX WORKCENTER LOOKUP APP

The Plex Workcenter Lookup app pulls workcenter data from the Plex Manufacturing Cloud. The Workcenter Lookup app enables you to do the following:

- Securely connect to the built-in sample company of the Plex Manufacturing Cloud (no credentials needed).
- Choose a workcenter by typing in the workcenter code.
- View data from the workcenter directly in the app.
- Insert the workcenter data directly into a Microsoft Excel document.

Plex Workcenter Lookup app requirements

To access the Plex Manufacturing Cloud using the Workcenter Lookup app, your PCN do the following:

- Subscribe to the Plex Integration Framework.
- Enable the Plex UX Rest API module.
- Configure a Plex user account to access Plex UX.

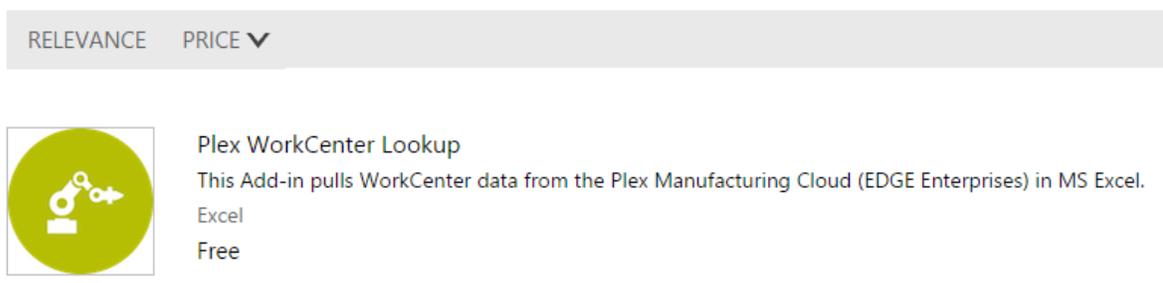
Add the Plex Workcenter Lookup app

You can add the Plex Workcenter Lookup app to your Office configuration from the Microsoft Office Store.

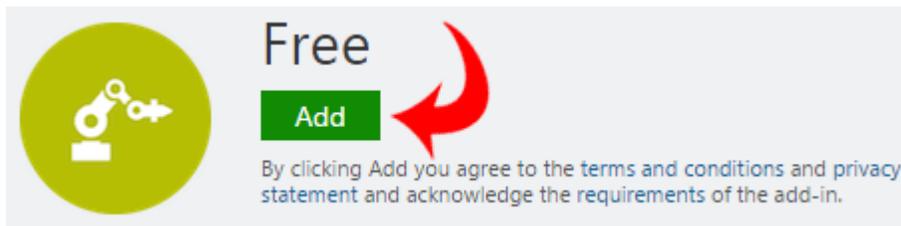
1. Access the Microsoft Office Store:
https://store.office.com
2. Sign in to your Office account.
3. Search for “Plex Workcenter Lookup”.



4. In the search results, click the link to the Plex Workcenter Lookup app information page.



5. On the Plex Workcenter Lookup app information page, review the requirements information to ensure that your system meets the necessary requirements.
6. Click **Add**.



7. On the confirmation page, click **Continue**.
Your Office account is updated.

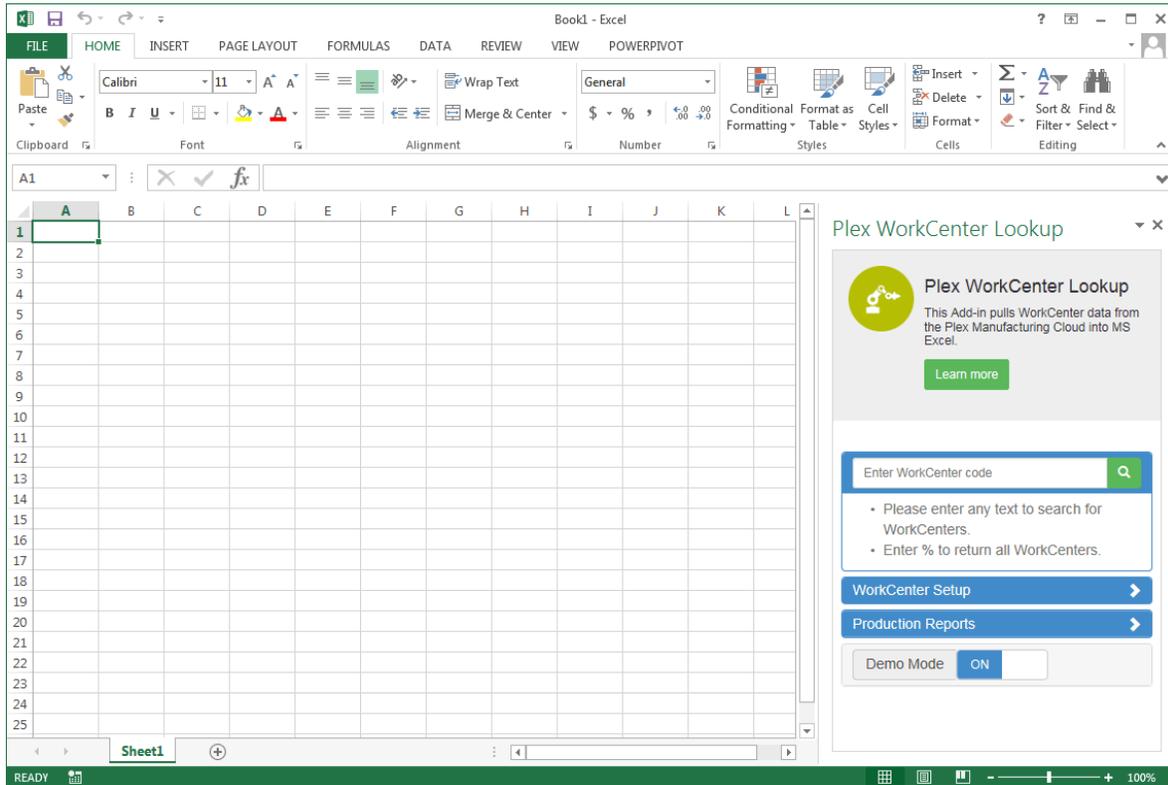
Run the Plex Workcenter Lookup app

Run the Plex Workcenter Lookup app in Microsoft Excel to retrieve workcenter information from Plex.

1. Open a Microsoft Excel worksheet.
2. In Excel, ensure that you are signed in to your Microsoft Office account.
3. On the **Insert** tab, click **My Apps**.
The **Apps for Office** dialog box is displayed.

4. Do either of the following:
 - If the Plex Workcenter Lookup app is shown, click the app.
 - Click **Store** to access the Office store, search for “Plex Workcenter Lookup”, and click the app.
5. When you run the app for the first time, click **Trust It**.

The Plex Workcenter Lookup app appears in your Microsoft application.

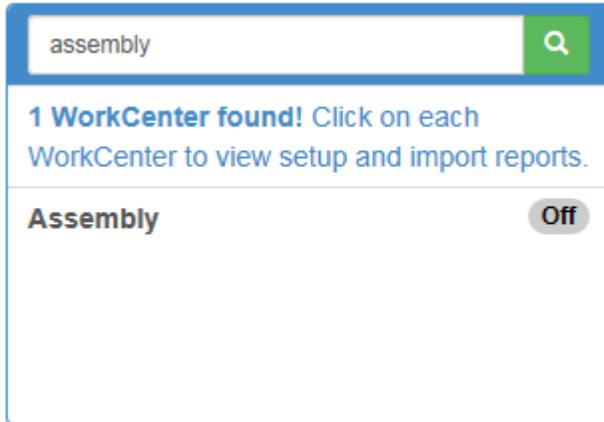


6. If you don't have a Plex API developer account, set **Demo Mode** to **ON**.

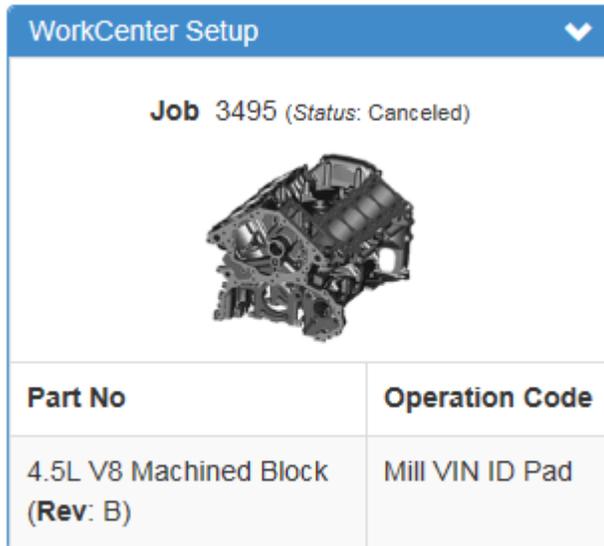


7. If you have a Plex API developer account, do the following:
 - a. Set **Demo Mode** to **OFF**.
Demo Mode is turned off, and text boxes for entering your account information are displayed.
 - b. For **User ID**, **Password**, and **Company Code**, type your Plex login information.
Note: Ensure the **Company Code** that you specify is the same as the main PCN associated with the **User ID**.
 - c. For **Subscription Key**, type the key that is associated with your Plex API Production Management subscription. You can obtain the key on your profile page within the Plex Developer Portal. You can use either the primary or the secondary key that is associated with the subscription.

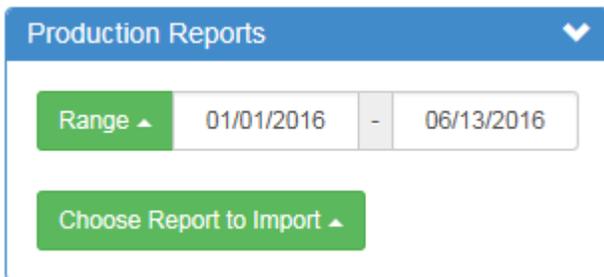
8. In the search box at the top of the app, do one of the following:
 - To display a list of all workcenters, type “%”.
 - To select a specific workcenter, type in the name of the workcenter.



9. Select the workcenter that you want to investigate.
10. Expand the **Workcenter Setup** section for information on the job currently associated with the workcenter.



11. Expand the **Production Reports** section to choose a report to import and specify a time frame.



TROUBLESHOOTING

This section provides troubleshooting information for the Plex API.

400 - Bad request

400 errors can arise for specific resources and generally indicate one or more missing request parameters. If you receive this error, verify that you used the valid audience for obtaining a token, such as <http://api.plex.com>.

The following error is specific to the token endpoint <https://api.plex.com/oauth2/v1/token>:

Example:

```
{
  "error": "invalid_resource",
  "error_description": "AADSTS50001: The application named http://api.plex.cm/ was not found in the tenant named 190f4bb0-c84d-4d0a-959a-a29c19923917. This can happen if the application has not been installed by the administrator of the tenant or consented to by any user in the tenant. You might have sent your authentication request to the wrong tenant.\r\nTrace ID: 6e5996f0-5bf1-48d2-9811-843bf30fce12\r\nCorrelation ID: 65a5f3a8-3ac0-44e0-edef-c6736521c497\r\nTimestamp: 2016-08-12 20:09:05Z",
  "error_codes": [50001],
  "timestamp": "2016-08-12 20:09:05Z",
  "trace_id": "6e5996f0-5bf1-48d2-9811-843bf30fce12",
  "correlation_id": "65a5f3a8-3ac0-44e0-edef-c6736521c497"
}
```

401 - Expired token

A 401 error can occur when you try to access the system using an expired access token, an invalid token, or without a token at all.

If you receive this error, refresh your access token, and try again to access the system.

Example:

```
{
  "status": 401,
  "Message": "Unauthorized. Access token is missing or invalid. Verify the token is in the Authorization Header with the token type 'Bearer'. Make certain the token is not expired. Use an inspection tool such as https://jwt.io/ and verify all expected claims are present, including exp, iat, nbf, etc. Make sure the token was requested from the Plex token endpoint with resource http://api.plex.com/."
}
```

401 - Invalid client secret

A 401 invalid client secret error can occur when you try to access the system using an expired or otherwise invalid client secret.

If you receive this error, you must obtain a new client secret for your application. Note that you can obtain more than one secret at once and implement rolling secrets in your application so that a single expired secret doesn't stop the application from operating.

Example:

```
{
  "error": "invalid_client",
  "error_description": "AADSTS70002: Error validating credentials. AADSTS50012: Invalid client secret is provided.\r\nTrace ID: 06cc4967-c269-4c85-9324-406f0250819c\r\nCorrelation ID: 64de8f58-1bfc-4bd4-8545-909841531eca\r\nTimestamp: 2016-12-12 15:30:03Z",
  "error_codes": [
    70002,
    50012
  ],
  "timestamp": "2016-12-12 15:30:03Z",
  "trace_id": "06cc4967-c269-4c85-9324-406f0250819c",
  "correlation_id": "64de8f58-1bfc-4bd4-8545-909841531eca"
}
```

403 – Access restricted resource

A 403 error can occur when your application attempts to access Plex functionality without having the necessary security privileges.

If you receive this error, you must adjust the security configuration for your service account, and try again to access the system.

Example:

```
{
  "type": null,
  "title": "Access Restricted Resource",
  "status": 403,
  "detail": "PCN does not have access to Controller 'Parts'",
  "instance": "[relative path]"
}
```

403 - Failed to get authorization token

If you receive this error, ensure that you are using the entire access token, and try again to access the system. If the error is recurrent, contact Plex.

Example:

```
{
  "type": null,
  "title": "Access Restricted Resource",
  "status": 403,
  "detail": "Failed to get authorization token",
  "instance": "/ProductionTracking/JobTrackingSystem/DcJobs"
}
```

403 - Usage Quota

We put a quota on API requests to protect the system from receiving more data than it can handle and to ensure an equitable distribution of system resources. A 403 error can occur when your application attempts to exceed this quota. The 403 message tells you how much time you must wait until you can make another request.

Example:

```
{
  "statusCode": 403,
  "message": "Out of call volume quota. Quota will be replenished in 02:28:27."
}
```

404 – Resource not found

A 404 error can occur when your application attempts to access Plex functionality using an incorrect request URL or if no data exists to return.

If an error message consists of only the `statusCode` and `message` properties, it's coming from the managed layer of the REST solution. In this case, the message "Resource not found" refers to the endpoint itself, as the managed layer does not hold data only endpoints. If you receive this error, adjust the request URL to the correct location, and try again to access the system.

Example:

```
{
  "statusCode": 404,
  "message": "Resource not found"
}
```

If an error message does not contain the `message` property but does include the `type`, `title`, `detail`, and `instance`, the endpoint exists although data is not present to return.

Example:

```
{
  "type": null,
  "title": "Resource Not Found",
  "status": 404,
  "detail": "Resource not found. Workcenters",
  "instance": "/ProductionTracking/WorkcenterList/Workcenters"
}
```

429 - Usage limiting

We put limits and quotas on API requests to protect the system from receiving more data than it can handle and to ensure an equitable distribution of system resources. A 429 error can occur when your application attempts to exceed these limits. The 429 message tells you how many seconds you must wait until you can make another request.

Example:

```
{
  "statusCode": 429,
  "message": "Rate limit is exceeded. Try again in 46 seconds."
}
```

Note: A data source also can be limited by UX constraints, typically to 1000 records.

500 - Unknown error

500 errors are unexpected. If you can reproduce the error, create a USR to report it to Plex. Include the steps to duplicate the issue, but do not include client secrets, passwords, or subscription keys.

Some 500 errors may require you to introduce fault tolerance, especially for a high-volume, around the clock solution. These can occur in situations like the following:

- A hopefully momentarily blip of server to server communication takes place in our data center. During scheduled maintenance this problem can last through the maintenance window. Note this does not indicate that the resource has timed out, but a failure to connect to the SQL server itself, which may be unreachable for planned maintenance.

Example:

```
{
  "type": null,
  "title": "Unknown Error",
  "status": 500,
  "detail": "An attempt to establish a SQL connection has timed out.
  Connection Timeout Expired."
}
```

- A specific network issue occurs, possibly during planned maintenance.

Example:

```
{
  "type":null,
  "title":"Unknown Error",
  "status":500,
  "detail":"An attempt to use the SQL data store has failed.  A network-related
  or instance-specific error occurred while establishing a connection to SQL
  Server. The server was not found or was not accessible."
}
```

- A request times out.

Example:

```
{
  "type":null,
  "title":"Unknown Error",
  "status":500,
  "detail":"Timeout expired.  The timeout period elapsed prior to completion
  of the operation or the server is not responding."
}
```

- An unexpected error arises around a single return property, such as “Part_No”. In reporting such an error, the endpoint is required.

Example:

```
{
  "type":null,
  "title":"Unknown Error",
  "status":500,
  "detail":"Part_No"
}
```

Note: All the earlier examples are from the backend, the Plex data center. An error from the managed layer is uniquely formatted:

```
{
  "statusCode": 500,
  "message": "Internal server error",
  "activityId": "ed99b04f-ca70-4be6-8626-7ddd0be6cb4f"
}
```