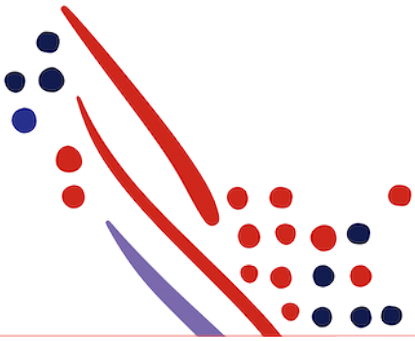
Guide

ADP Marketplace Integration Standards

Published on
Jul 23, 2021 9:42AM

Last modified
Apr 18, 2023 2:58PM

Always Designing for People™



ADP Copyright Information

Published on
Jul 23, 2021 9:42AM

Last modified
Apr 18, 2023 2:58PM

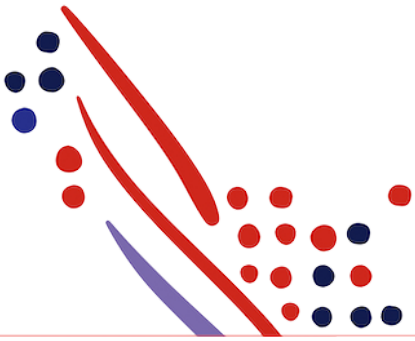


Table of Contents

Chapter 1

Overview

Integrations with ADP Marketplace will have to follow the standards as required to call the ADP API's. This article provides the list of standards and what is expected from the partner integrations in terms of requirements. Please make sure that your integration follows the standards from this document. ADP Marketplace Technical Advisory team will review your demo of the integration based on these standards as well. Integrations are required to meet these standards to be reviewed by ADP security team and to go-live.

Chapter 2

Client Credential Management

Client Credentails

Your integration will have to manage the client credentials that were retrieved through https://api.adp.com/events/core/v1/consumer-application-subscription-credentials.read. Retrieved client credentials should be stored securely within your application. Your application cannot call the credentials.read API for every API call that was made on clients' behalf.

Access Token and Token Management

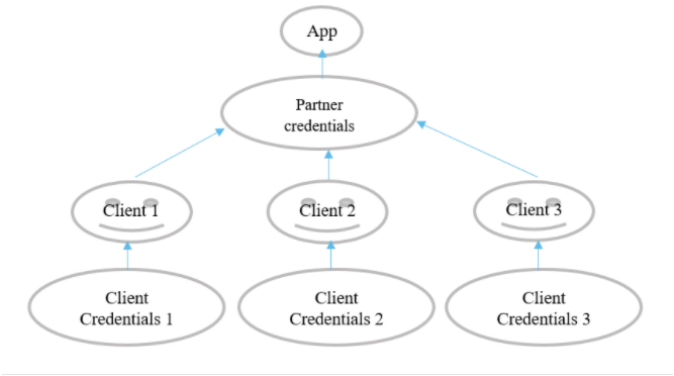
Your integration will have to make the access token calls in two different scenarios. First scenario is for calling https://api.adp.com/events/core/v1/consumer-application-subscription-credentials.read with your partner credentials. The second scenario is for calling the data APIs using the client credentials on behalf of each client.

Note that the access token is valid for 60 minutes, so you will have to reuse the access token if you are going to make the API calls within 60 minutes. Your integration cannot call the access token for every API call unless it's separated by 60 minutes.

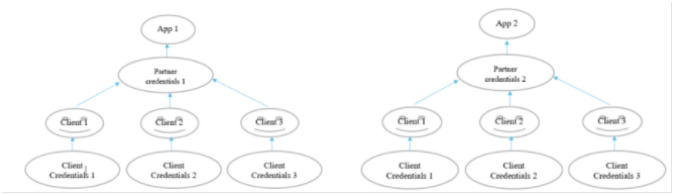
Multi-Tenancy

As a consuming integration application, your integration should implement multi-tenancy within your application for integrating with ADP. For every integration you list in the ADP Marketplace storefront, you will be provided with partner credentials, and you will be retrieving client credentials for each client who subscribes to your app.

For example, if you list one integrated app on ADP Marketplace, you will get one set of partner credentials, and you will retrieve one set of client credentials for each client that subscribes to your app. Below is a hypothetical representation of a credentials scenario when you list one app.



If you list two integrated apps on ADP Marketplace, you will be provided with two sets of partner credentials (one for each integration), and you will retrieve a maximum of two client credentials, depending upon the number of apps to which a client subscribes. Below is a hypothetical representation of a credentials scenario when you list two apps.



Chapter 3

ADP Marketplace API Best Practice

Meta API Calls

You will use the ADP meta APIs on two different scenarios:

1. **During the code development time.** Meta provides the rules that must be imposed on an interface for a given operation and within a given context. Each API will have a meta schema with all the possible rules that can be imposed. The meta API returns the relevant properties in a given context and the validation and presentation rules for each.

2. **During the run time of the interface.** An API user must use the meta API response in conjunction with an events schema to post a valid event body.

Event Notifications

Your application that relies on ADP worker data (/hr/v2/workers) should integrate with ADP events notification, instead of extracting the data daily or on an hourly basis. Integrating with ADP events notification will help the consuming application to listen for the changes and update on its side.

Here is how the ADP worker data extraction should be implemented by the consuming application:

1. **Initial extract:** After a client subscribes to the integration, your application should extract the ADP worker data (/hr/v2/workers) one time.

2. **Continuous update:** After the initial extract, consuming applications should listen to the changes through the ADP events notification, and update the worker data on its side. For the purpose of changes, e.g., address change, name change, email change, etc., you should not be calling

the (/hr/v2/workers) on a continuous basis.

3. **On demand:** Your application should also have a mechanism to extract the worker data through an on-demand basis. This is very similar to the Initial extract, but should be used only when there was a loss of data due to an outage or similar situation.

4. Event notifications are triggered at the time the data is changed in the SOR, not on the effective date.

5. Event notifications will be deleted from the message queue after five days if they are not consumed.

6. Read the event notification queue with a polling interval of five minutes. This is a minimum time requirement. Depending upon the integration, polling interval can be increased.

7. You will have to implement the logic of updating only the latest value based on the timestamp. You should check notification's DateTime and ensure only the newest notification are processed for a given notification type by the receiver. You should isolate the notification for timestamp (by type of event, org id and AOID).

Pagination

ADP APIs use the pagination concept when retrieving large data (e.g., /hr/v2/workers). The support level varies between different SORs in terms of how many records would be extracted per call. Therefore, your interface must use the ODATA mechanism to retrieve the records successfully.

See this post for more information on ODATA.

Caching

ADP APIs have the mechanism of caching when you retrieve a large volume of data. The caching duration varies between APIs and SORs that are being integrated.

For example, the ADP API proxy caches the worker API (successful) response for 24 hours. So, if the same request comes in within 24 hours, they will be given cached response. You can handle this in one of two ways:

1. Try using "If-None-Match: none" in the header.

2. Add ?preventCache=1 to the URI.

Exception Handling

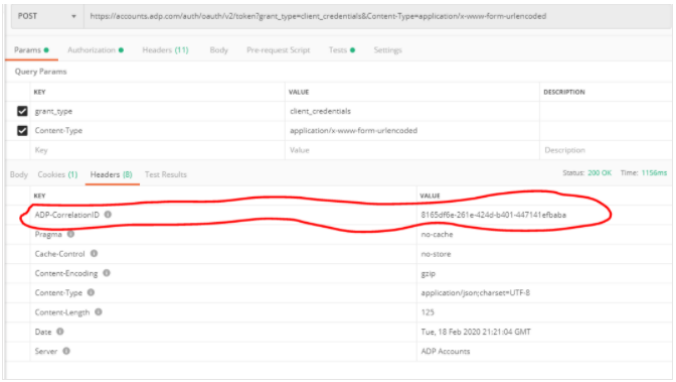
Developers must implement exception handling to make sure the integrations take the appropriate actions in case of exceptions. Please follow the directions in this guide to implement the exception handling in your code.

Creating users in partner applications using ADP worker API

You must not add users into your application using the user assignment functionality. Please refer to this article on how to add non-admin users to your application.

Correlation IDs

ADP generates a unique identifier by name correlation ID for every API call. You should log the ADP correlation ID for each of the ADP API calls. In case of issues, this is the ID ADP can use to track the issues. This is a header parameter provided in the response for each of the API call, as seen here:



ADP Marketplace Maintenance Windows

ADP Marketplace gateway and SOR's have a regular weekly schedules for planned maintenance. Please check the article **"Working with an ADP Product Maintenance Schedule"** to make sure that you are making API calls according to this schedule.