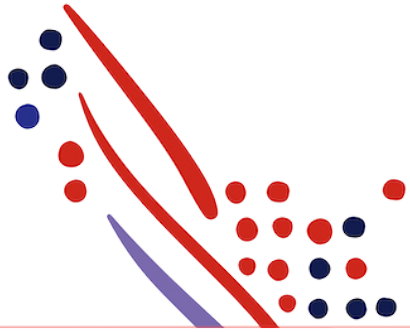
Guide

Pay Data Input API Guide for iHCM NL (pilot)

Published on
Jun 30, 2022, 08:37 AM

Last modified
Sep 04, 2023, 11:59 AM

Always Designing for People™



ADP Copyright Information

Published on
Jun 30, 2022, 08:37 AM

Published on
Sep 04, 2023, 11:59 AM

Table of Contents

Chapter 1

About this API

Process Overview

The following table shows how your data connector application would be used by a client.

	Actor	Task Description
1	Client Practitioner	Present the iHCM NL application with payroll data that has changed in your system, to be synchronized with the iHCM NL application
2	Client Practitioner	Check the results as presented by the Pay Data Input API for warnings and errors to evaluate the success rate of the API call
3	Client Practitioner	Verify the Information in ADP iHCM NL

What's New in this guide

In this section we will announce any new revisions to the Pay Distribution API Guide.

	Date	Description	Section
1	30/06/2022	Creation	Chapter 1-2, appendix
2	20/10/2022	Update: key dictionary, JSON samples	Chapter 2, appendix
3	25/05/2023	Added .replace documentation	Chapter 2, appendix

Before You Begin

ADP provides the Pay Data Input API to ensure your application can synchronize its payment data with iHCM NL. This can be done by either adding data (targeting the [..]add API clall) to a new batch, in wich case the client is advised to store the batch ID returned, because later a call to the [..]/replace API can be utilized to replace the data added by the batch earlier, using the batch ID provided by add. This way, the client can correct erroneous data.

Please note that only **variable** data from your system can be presented to iHCM NL using the Pay Data Input API. The system will try to match and import all data presented and will return information on data successfully imported and on problems encountered.

Chapter 2

Use Case: sychronizing pay data with iHCM NL

Use Case Description

The Pay Data Input API is used to synchronize **variable** pay data of the clients' system with iHCM NL: several paygroups, each with separate employees, and each employee with one or more entries for paydata can be offered to the API, which will process the correct data and return information on missing or faulty data.

NB: If you would like to use the Pay Data Input API for *updating* client data passed earlier to ADP by way of this API, you should pass correction values. For example, correcting a value of 100 units to a desired 80 units should be done by passing -20 units. Alternatively, the [..]replace API call can be utilized, to

replace a batch of data presented by the [..]/add API call earlier (provided this data has not yet been processed to the ADP central systems)

API Usage

Method	Type	Context	Use Case	Uniform Resource Identifier (URI)	Description	GitHub Sample Response Payload
POST	.add	NL	When you want to update iHCM NL with current Pay as now stored in your local system	/events/payroll/v1 /pay-data-input.add	Returns you all Pay Data that is successfully synchronized with iHCM NL, as well as error mesages	Successful Add, request sample Successful Add, response sample
POST	.add	NL	Same as above, but with an error message being returned	/events/payroll/v1 /pay-data-input.add	Returns all Pay Data that is successfully synchronized with iHCM NL, as well as error messages concerning incorrect keys or data that could not be synchronized	Unsuccessful Add, request sample Unsuccessful Add, response sample
POST	.replace	NL	When you want to correct data in iHCM NL that has been added with an .add call	/events/payroll/v1 /pay-data-input.replace	Given a batch id returned by .add, replace the corresponding batch by deleting the original and creating a new one **with the same id** based on the input data	Successful replace, request sample Successful replace, response sample
POST	.replace	NL	Same as above, but with an error message being returned	/events/payroll/v1 /pay-data-input.replace	Returns a warning that the designated target batch to mnreplace could not be found	Unsuccessful replace, request sample Unsuccessful replace, response sample
GET	.addmeta	NL	When you want to know the schema which the .add call adheres to	/events/payroll/v1 /pay-data-input.add/meta	Returns the schema for the .add call	Successful meta return sample for .add

GET	.replacemeta	NL	When you want to know the schema which the .replace call adheres to	/events/payroll/v1 /pay-data-input.replace/meta	Returns the schema for the .replace call	Successful meta return sample for .replace

Application Scope

The canonical URI corresponding to the API needs to be added in the Consumer Application Registry (CAR) for the subscription after which a user can access this API and make successful calls. Please note that the /meta calls do not have to be registered separately.

The following canonicals need to be added to your application scope to enable this use case:

/payroll/payrollManagement/payrollProcessing/payDataInputManagement/pay-data-input.add
/payroll/payrollManagement/payrollProcessing/payDataInputManagement/pay-data-input.replace

Supported OData Query Options

There is no support for OData parameters.

Data Entitlements

It is important to keep in mind that the API results are bound to the consumer's rights and population. Every consumer is part of a population where the consumer has the right to access associate's data. If the request is being made for an associate that does not belong to the population, then an error response will be returned.

Sequence of Interactions

Perform the following sequence of interactions to retrieve full worker information for a collection of workers:

1. Your consumer application makes a request to the ADP API endpoint to Post the Pay data information to synchronize.
2. The ADP API endpoint responds to your consumer application with information about the successfully synchronized data and errors concerning data that could not be synchronized.

Responses

You may encounter exceptions outside your common success scenarios. You must account for these exceptions during your initial development. For more information, see API Common exceptions and Tips for Handling.

To enable effective use of .replace, you are advised to store the batch id's returned by instances of .add, because these are required as an input for .replace. At the time of writing, there is no mechanism to retrieve a list of id's, but this is something that might be introduced in the future.

In the response body of .add, the batch id can be accessed by navigating the path events/data/output/payDataInput/itemId

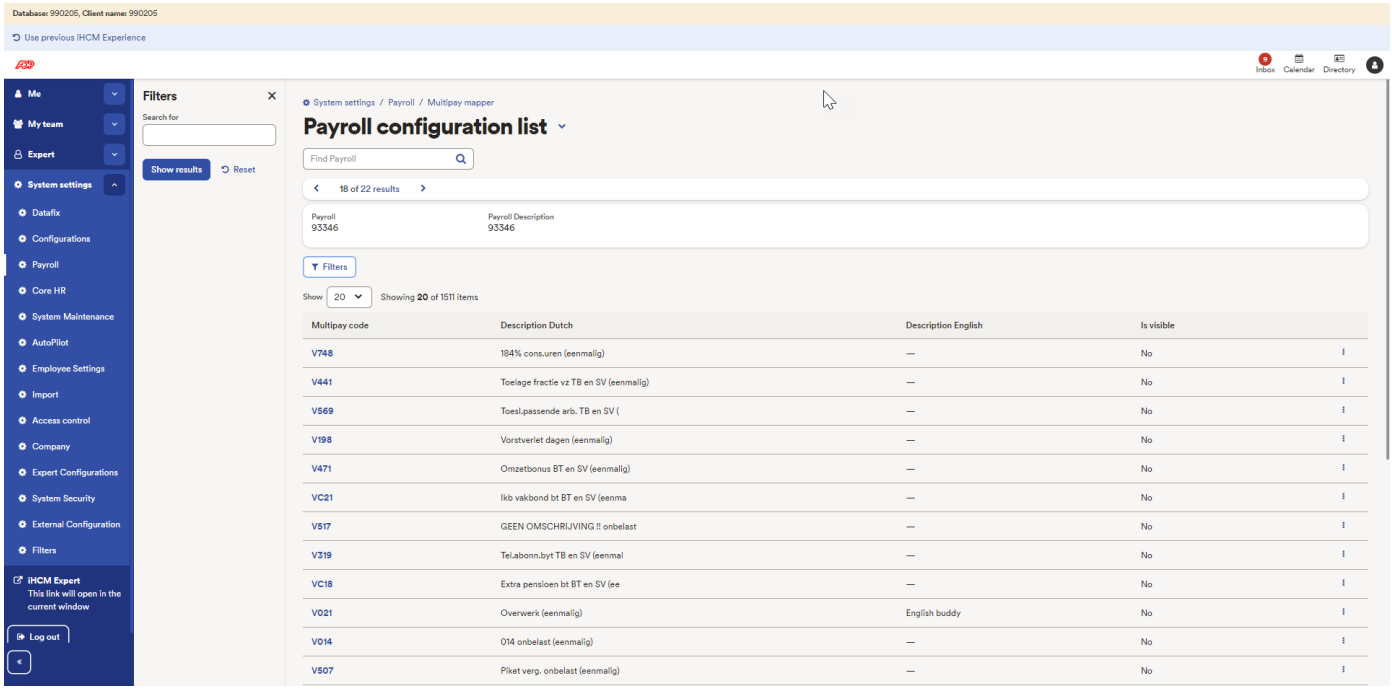
Validations

The API call should target valid paygroups that the user is authorized to update data for, using the correct numerical paygroup code, as present in the targeted DB. Pay changes should target specific employees using valid variable and authorized target fields in the DB and pay changes should not be repeated and should consist of a non-empty value with the length and the decimal places in the range as expected by the DB, targeted for a specific effective date. If any of these conditions are not met, the error messages as shown in the section "API Usage" above will be included in the result.

See *Validations performed* in the appendices for a list of the validations the API will perform on the input data. Note that all validations specified for .add are also applicable to .replace, but not the other way around.

NB: To check in the iHCM application what valid columns are mapped for a specific client, login on said DB, using iHCM, and then find your way through the menu tree as follows:

System settings > Payroll > Multipay mapper [Select a paygroup]



Chapter 3

Appendices

Appendix [1] - Data Dictionary for .add and .replace

Key	Sample value
data | eventContext | payrollRegionCode	"codeValue": "NL"
data | eventContext | payrollGroupCode	"codeValue": "93346"
data | eventContext | transform| payDataInput| payeePayInputs| payeeID	"idValue": "5000041"
data | eventContext | transform| payDataInput| payeePayInputs| payPeriodStartDate	"2022-01-01"

data | eventContext | transform| payDataInput| payeePayInputs| payPeriodEndDate	"2022-01-31"
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs| earningCode	"codeValue": "V160"
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs| rate	"rateValue": -40.01
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs| earnedPayPeriodStartDate	"2022-01-01"

Appendix [2] - Validations performed on the Pay Data Input Api for .add and .replace

Element Validated	Validation on Element
Paygroup number	The Api validates if the target iHCM data base has **any valid paygroups**
Paygroup number	The Api validates if the paygroup code passed with the input data represents a **valid numerical iHCM paygroup** code
Paygroup number	The Api validates if the paygroup code passed with the input data represents a iHCM paygroup code the user has **access** to
Employee number	The Api validates if the paygroup code passed with the input data represents a **valid numerical** iHCM employee number

Payment date	The API validates that the **begin and end date** for a payment data change that are passed with the input match a valid iHCM payment date range. If not, either the first available valid payment date range is selected from the iCHM database, and if none is found, changes are entered in the database without such a range, awaiting later linking to a valid payment date range
Paygroup assigment	The API validates if the paygroup code and employee code passed in the input constitute a **valid contract** in the iHCM database
Pay data Field	The API validates that the Pay Data **target field** is set
Pay data Field	The API validates that the Pay Data target field set constitutes a **valid target field** in the iHCM database
Pay data Field	The API validates that the Pay Data target field set constitutes a **variable field** in the iHCM database
Pay data Field	The API validates that the Pay Data target field set constitutes a **field enabled for the specified paygroup** in the iHCM database
Pay data Value	The API validates that the Pay Data value passed is **not set twice**
Pay data Value	The API validates that the Pay Data value passed is **not empty**
Pay data Value	The API validates if the Pay Data value passed has the **effective date** set
Pay data Value	The API validates that the Pay Data value passed has an effective date that is set **on or after** the start date of the contract
Pay data Value	The API validates that the Pay Data value passed has an effective earning date that is set **before the end date** of the pay period. Note that if the effective earning date is set before the start date of the pay period, it will be marked as retroactive.
Pay data Value	The API validates if the **number of decimals** of the Pay Data value passed does not exceed the number of decimals allowed for the targeted field in the iHCM database
Pay data Value	The API validates if the **length** of the Pay Data value passed does not exceed the length of the targeted field in the iHCM database

Appendix [3] - Additional Data Dictionary entries, specific for .replace

Key	Sample value
data| eventContext| payDataInput| itemID	"23-05-01"

Appendix [4] - Additional validations performed on the Pay Data Input Api, specific for .replace

Element Validated	Validation on Element
Batch id	The API validates if the batch id passed with the input data is **not empty**
Batch id	The API validates if the batch id passed with the input data is **properly formatted**
Batch id	The API validates if a call containing multiple batch id's contains **no duplicate id's**
Pay data state	The API validates if the Pay Data contains **no commited data changes**
Pay data state	The API validates if the supplied data changes are **not already assigned to a blocked payrun**
Pay data state	The API validates **even if** it contains new data changes that cannot be coupled to a (potentially blocked) payrun or if it would replace data changes that have not yet been assigned to a payrun
All	The API call validation is **all or nothing**: if one of the conditions fails, the entire batch will fail