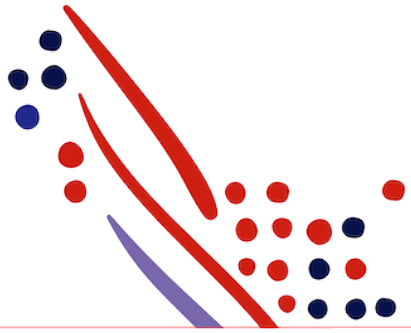
Guide

Pay Data Input API Guide for iHCM NL

Published on
Jun 30, 2022, 08:37 AM

Last modified
Jul 10, 2026, 04:06 AM

Always Designing for People™



ADP Copyright Information

ADP, the ADP logo, and Always Designing for People are trademarks of ADP, Inc.

Windows is a registered trademark of the Microsoft Corporation.

All other trademarks are the property of their respective owners.

Copyright © 2026 ADP, Inc. ADP Proprietary and Confidential - All Rights Reserved. These materials may not be reproduced in any format without the express written permission of ADP, Inc.

These materials may not be reproduced in any format without the express written permission of ADP, Inc. ADP provides this publication "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose. ADP is not responsible for any technical inaccurancies or typographical errors which may be contained in this publication. Changes are periodically made to the information herein, and such changes will be incorporated in new editions of this publication. ADP may make improvements and/or changes in the product and/or the programes described in this publication.

Published on
Jun 30, 2022, 08:37 AM

Published on
Jul 10, 2026, 04:06 AM

Table of Contents

Chapter 1

About this API

- Process Overview
- What's New in this guide
- Before You Begin
- Data type overview

Chapter 2

Use Case: sychronizing pay data with iHCM NL

- Use Case Description
- API Usage
- Application Scope
- Supported OData Query Options
- Data Entitlements
- Sequence of Interactions
- Responses
- Validations
- Determining payrun by date

Chapter 3

Appendices

- Appendix [1] - Data Dictionary for .add and .replace
- Appendix [2] - Validations performed on the Pay Data Input Api for .add and .replace
- Appendix [3] - Additional Data Dictionary entries specific for .replace
- Appendix [4] - Additional validations performed on the Pay Data Input Api specific for .replace

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 enable your application to synchronize its payment data with iHCM NL. This is done by sending **batches** of data labelled by **batch ID**. Adding a batch of data can be done through the **.add** call, which returns a batch ID. This ID is advised to be stored as it can be used later to modify data provided in the batch using the **.replace** call.

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.

Data type overview

Consult the table below for selecting an API for your desired datatype if the current one is not what you are looking for.

Data type	API
Variable fields (Vxxx)	Pay Data Input - Add/Replace
Salary fields: 2416 - salary code 2417 - salary amount	Base Remuneration - Change

Numeric fields with empty taxability: P - Percentage B - Amount F - Frequency N - Not defined U - Hours D - Days	Additional Remuneration - Change
Numeric fields with taxability I (deductions)	General Deduction Instruction - Change
Text, Date, Lookup field	Memo Instruction - Change

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: Correction values should be passed to update client data that has already been submitted to iHCM through the Pay Data Input API. For example, correcting a value of 100 units to a desired 80 units should be done by passing -20 units. Alternatively, the **.replace** call can be utilized to replace a batch of data that has been submitted through the corresponding **.add** call, provided this data has not yet been processed by the ADP central systems.

API Usage

Visit the corresponding developer resource page to explore supported calls and their possible states and to reference some example requests and responses.

Note that in the request body, the `payrollGroupCode` field is the paygroup code in iHCM and can be retrieved through the worker API in the `paygroupcode` field. The `payeeID` request body field would be the employee number in iHCM, which maps to the `payrollfilenumber` field from the worker API.

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. For both URIs there is a /meta call that does not have to be registered separately.

The following canonicals should be added to your application scope to enable the use of the Pay Data Input API:

`/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

Keep in mind that the API results are limited by the consumers' access level: an error response will be returned if a request is being made by a consumer who does not have the right privileges in terms of data access.

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 Pay Data Input API endpoint to POST the pay data information that is to be synchronized.

2. The 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, which should be accounted for during initial development. For more information, see API Common exceptions and Tips for Handling.

To enable effective use of **.replace**, the batch ID's returned by instances of **.add** should be stored as these are required as an input for **.replace**. Note that *all* fields from the **.add** action response should typically be sent with the **.replace** call as the omittance of one such field will result in its deletion. 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 database. Pay data changes should target specific employees using valid variable and authorized target fields that exist in the database and the pay data 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 database, targeted for a specific effective date. If any of these conditions are not met, an appropriate error message will be included in the result.

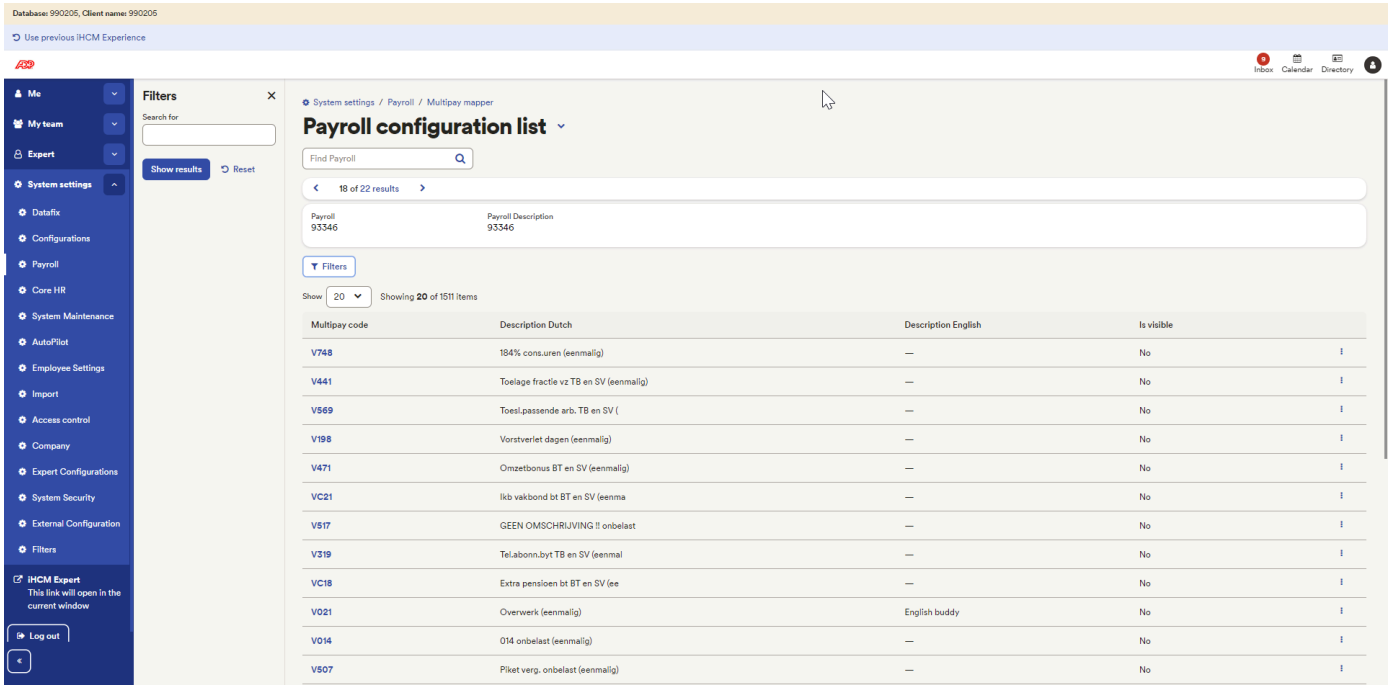
See the chapter *Validations performed* in the appendices below for a list of the validations that the API will perform on the input data. Note that all validations specified for **.add** are also applicable to **.replace**, but not the other way around.

Determining payrun by date

The Pay Data Input API uses the **payPeriodStartDate** and **payPeriodEndDate** to determine the payrun to which the datachanges will be linked. If the dates do not match, the request will be accepted by the API but the data will not be linked to a payrun. The user should then either manually link the datachanges in iHCM or wait for the creation of the payrun, in which case the changes will be linked automatically.

NB: To check the valid columns that are mapped for a specific client, either use the corresponding /meta calls or navigate the iHCM interface as follows:

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



Chapter 3

Appendices

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

Key	Sample value	Notes
data | eventContext | payrollRegionCode	"codeValue": "NL"	
data | eventContext | payrollGroupCode	"codeValue": "93346"	Paygroup code in iHCM, corresponds to the paygroupcode worker API field
data | eventContext | transform| payDataInput| payeePayInputs| payeeID	"idValue": "5000041"	Employee number in iHCM, corresponds to the payrollfilenumber worker API field
data | eventContext | transform| payDataInput| payeePayInputs| payPeriodStartDate	"2022-01-01"	Identify the target payroll run by exactly matching the start date of the pay period accroding to the paygroup's frequency.
data | eventContext | transform| payDataInput| payeePayInputs| payPeriodEndDate	"2022-01-31"	Identify the target payroll run by exactly matching the end date of the pay period accroding to the paygroup's frequency.
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs| earningCode	"codeValue": "V160"	
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs|	"rateValue": -40.01	

rate		
data | eventContext | transform| payDataInput| payeePayInputs| payInputs| earningInputs| earnedPayPeriod StartDate	"2022-01-01"	Effective date of the datachange to be sent. Choosing a date before the start date of the selected pay period will result in a backdated entry. Must be on or after the employee's contract start date.

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 assigmen t	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