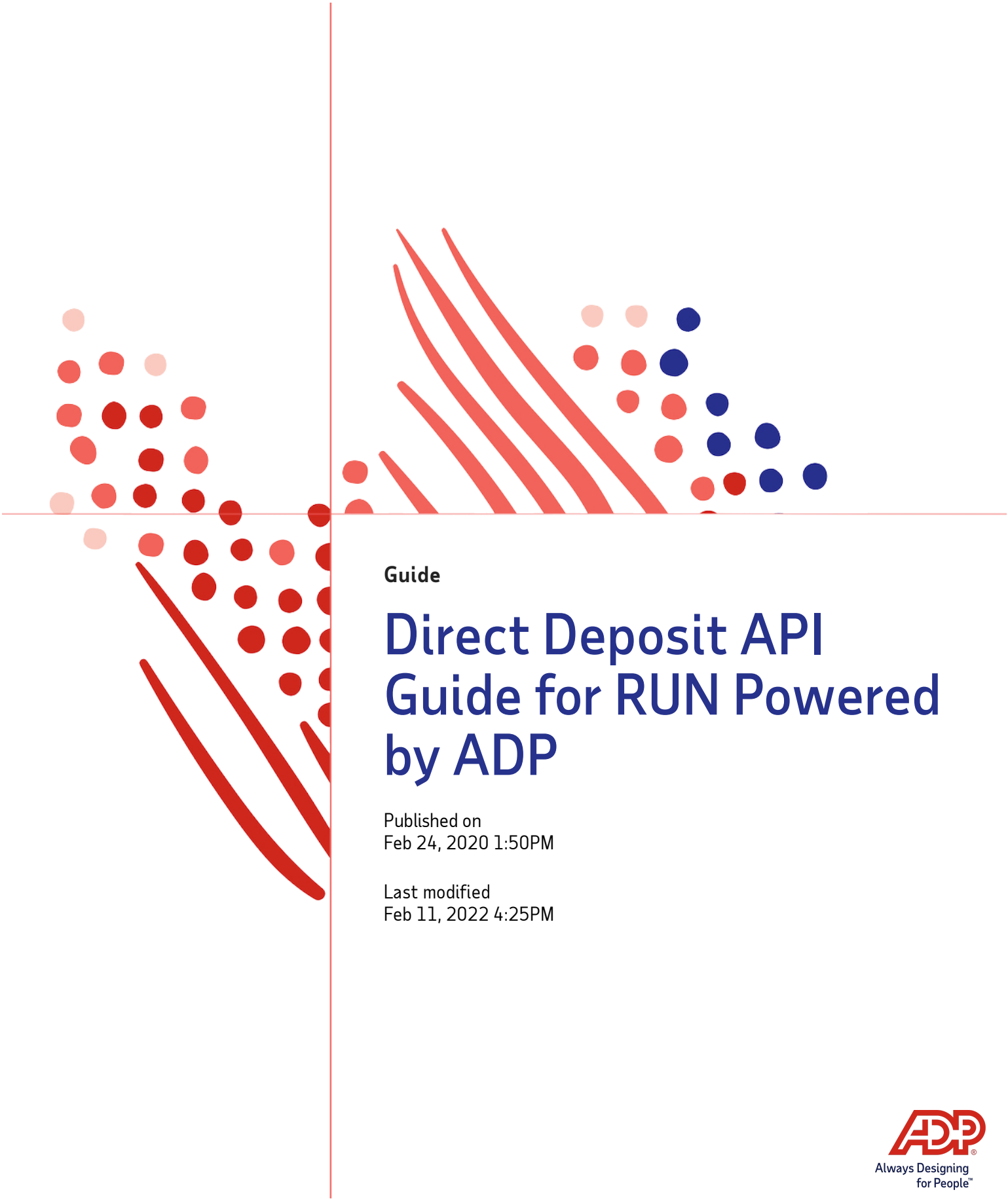


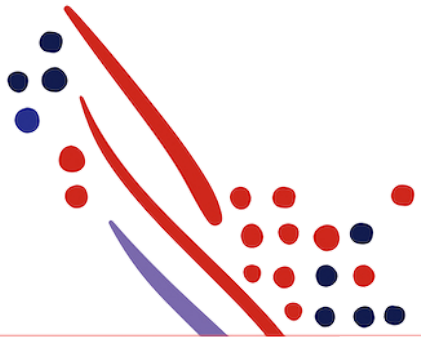
Guide

Direct Deposit API Guide for RUN Powered by ADP

Published on
Feb 24, 2020 1:50PM

Last modified
Feb 11, 2022 4:25PM

Always Designing for People™



ADP Copyright Information

Published on
Feb 24, 2020 1:50PM

Last modified
Feb 11, 2022 4:25PM

Table of Contents

Impacted API

Issue Description

Suggested Workaround

Chapter 1

About the API

Summary

The Direct Deposit Application Programming Interface (API) enables users to view, add, update and remove employee's direct deposit setup.

What's New in this Guide?

Additions and updates to the Direct Deposit API Guide for RUN Powered by ADP include the following:

Feb 15, 2022

Updated API endpoint and samples using API Explorer information

March 13, 2020

Added the End User Application, also referred to as Business to Consumer (B2C) Application support.

Supported Product Version and Customer Base

This is available for workers in the United States.

Process Overview

Typical B2B Application Integration Flow

	Actor	Task Description
1	Your Application	Retrieves and synchronizes employee data using the Worker Management API.
2	Your Application	Retrieves direct deposit data for an employee using the employee's unique ADP Associate Object Identifier (AOID).
3	Your Application	Sends updated direct deposit data to RUN Powered by ADP.

Important: Practitioners can only retrieve and update an employee's direct deposit records with Employee Self Service (ESS) access enabled and an associated email ID on RUN Powered by ADP.

Client Practitioners need to contact ADP Service to ensure their employees are allowed to self-configure direct deposit.

Typical B2C Application Integration Flow

	Actor	Task Description
1	User	Login to your application.
2	User	Authenticate using the employer-issued ADP credential and consent to allow your application to retrieve the individual's ADP-managed direct deposit data for that employer.
3	Your Application	Use the Direct Deposit API to retrieve the individual's direct deposit setup data.
4	User	In your application, view direct deposit setup data.
5	User	In your application, update direct deposit setup data.
6	Your Application	Use the Direct Deposit API to replace direct deposit setup data for the user under the employer.

Recommended Reading

For more information, see *Common API Operations for RUN Powered by ADP* at https://developers.adp.com/articles/guides/common-api-operations-for-run-powered-by-adp.

Chapter 2

API Details

Endpoints and samples

Pay Distribution API for RUN Powered by ADP

Application Scope

The canonical URI corresponding to the API needs to be added in the Consumer Application Registry (CAR) for the subscription following which a user can access this API and make successful API calls.

The following canonical needs to be added to your application scope to enable this use case:

1. **/payroll/payrollManagement/payrollInstructionManagement/payDistributionManagement/worker.payDistribution.read**
2. **/payroll/payrollManagement/payrollInstructionManagement/payDistributionManagement/worker.payDistribution.change**

Supported Actors

Request Parameter roleCode value	Usage
practitioner	Retrieves an employee's direct deposit details. A system user is considered a practitioner.
employee	Retrieves your own direct deposit details.

Sequence of Interactions

The following are common request flows

1. Retrieve pay distributions/direct deposits for interested worker
2. Retrieve **events/payroll/v1/worker.pay-distribution.change/meta** to the ADP API endpoint.
3. The ADP API endpoint responds to your consumer application with the meta payload. Your consumer application processes the meta payload to validate
   data included in the payload (step 3 in the above figure) and prompts the user to fix any validation issues. Otherwise, compose the payload for the **events/payroll/v1/worker.pay-distribution.change** request.
4. The ADP API endpoint responds to the consumer application concerning the details of **events/payroll/v1/worker.pay-distribution.change**.

Data Dictionary

Resources listed in the following table can be accessed in the RUN Powered by ADP UI by selecting **Employees > Payroll > Direct Deposit**.

Schema Location	Field Name in RUN UI	Required(Y/N)?	Note
/events/serviceCategory Code			
/events/serviceCategory Code/codeValue			
/events/eventNameCode			
/events/eventNameCode /codeValue			
/events/data/			
/events/data/eventCont ext/			
/events/data/eventCont ext/worker			
/events/data/eventCont ext/worker/associateOI D			

/events/data/transform/payDistribution			
/events/data/transform/payDistribution/requestedStartDate			Indicates the current date.
/events/data/transform/payDistribution/distributionStatusCode			
/events/data/transform/payDistribution/distributionStatusCode/codeValue			
/events/data/transform/payDistribution/distributionStatusCode/shortName			
/events/data/transform/payDistribution/distributionStatusCode/longName			
/events/data/transform/payDistribution/distributionInstructions			
/events/data/transform/payDistribution/distributionInstructions/itemID			
/events/data/transform/payDistribution/distributionInstructions/precedenceCode		Y	
/events/data/transform/payDistribution/distributionInstructions/precedenceCode/codeValue			The valid values are **1**, **2**, **3**, and **4**.
/events/data/transform/payDistribution/distributionInstructions/precedenceCode/shortName			
/events/data/transform/payDistribution/distributionInstructions/precedenceCode/longName			
/events/data/transform/payDistribution/distributionInstructions/paymentMethodCode			
/events/data/transform/			The valid value is **D**.

payDistribution/distributionInstructions/paymentMethodCode/codeValue			
/events/data/transform/payDistribution/distributionInstructions/paymentMethodCode/shortName			The valid value is **Direct Deposit**.
/events/data/transform/payDistribution/distributionInstructions/paymentMethodCode/longName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/nameCode			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/nameCode/codeValue	**Bank Name**		
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/nameCode/shortName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/nameCode/longName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/routingTransitID		Y	
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialPart	**Routing Number**		

y/routingTransitID/idValue			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/routingTransitID/schemeCode			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/routingTransitID/schemeCode/codeValue			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/routingTransitID/schemeCode/shortName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialParty/routingTransitID/schemeCode/longName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/accountNumber	**Account Number**	Y	
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/accountName	**Nickname**		
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/typeCode			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/typeCode/codeValue	**Account Type**		The valid values are **C** and **S**.
/events/data/transform/			The valid values are **Checking** and **Savings**.

payDistribution/distributionInstructions/depositAccount/financialAccount/typeCode/shortName			
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/typeCode/longName			The valid values are **Checking** and **Savings**.
/events/data/transform/payDistribution/distributionInstructions/depositAccount/financialAccount/currencyCode			
/events/data/transform/payDistribution/distributionInstructions/distributionAmount			
/events/data/transform/payDistribution/distributionInstructions/distributionAmount/amountValue	**Amount**		
/events/data/transform/payDistribution/distributionInstructions/distributionAmount/currencyCode			
/events/data/transform/payDistribution/distributionInstructions/distributionPercentage	**Amount**		
/events/data/transform/payDistribution/distributionInstructions/remainingBalanceIndicator			
/events/data/transform/payDistribution/distributionInstructions/prenoteBypassIndicator			

Fields Not Supported

Schema Location	Field Name
NA	**Deposit Distribution**

Common Rejected Requests

You may encounter exceptions outside your common success scenarios. You must account for the following exceptions during your initial development.

Response Code	**Condition**	**messageTxt**
Response Code	**Condition**	**messageTxt**

400 Bad Request	The **Financial Account** is missing in the request payload.	"userMessage": {
		"messageTxt": "Account information is not provided.;"
		}

400 Bad Request	Length of the **Account Number** is less than 4.	"userMessage": {
		"messageTxt": "Account number length cannot be less than 4.;"
		}

400 Bad Request	The **Account Number** has invalid characters. Other than Alphanumerics, it contains a space and & character.	"userMessage": {
		"messageTxt": "Account number has invalid characters.;"
		}

400 Bad Request	The **Nickname** has invalid characters. Other than Alphanumerics, it contains a space and & character.	"userMessage": {
		"messageTxt": "Nickname has invalid characters.;"
		}

400 Bad Request	The **Bank Account Type** is missing	"userMessage": {
		"messageTxt": "Bank Account Type is not provided.;"
		}

400 Bad Request	The **Bank Account Type** is invalid, other than **S** or **C**.	"userMessage": {
		"messageTxt": "Bank Account Type should either be Checking or Savings.;"
		}

400 Bad Request	The **Distribution Percentage** has more than two decimals.	"userMessage": {
		"messageTxt": "Only two decimal places are allowed in Distribution percentage.;"
		}

400 Bad Request	The **Distribution Percentage** and **Distribution Amount** are 0.	"userMessage": {
		"messageTxt": "A valid pay distribution type was not provided.;"
		}

400 Bad Request	The **Distribution Percentage** is greater than 0 and the **remainingBalanceIndicator** is True.	"userMessage": {
		"messageTxt": "A valid pay distribution type was not provided.;"
		}

400 Bad Request	Both the **Distribution Percentage** and **Distribution Amount** are provided.	"userMessage": { "messageTxt": "You can enter a percentage or an amount, not

		both.;" **}**
400 Bad Request	The **Distribution Amount** has more than two decimals.	**"userMessage": {** **"messageTxt": "Only two decimal places allowed in Distribution Amount.;"** **}**
400 Bad Request	The **Payment Method** name is invalid. Other than the **Direct Deposit**.	**"userMessage": {** **"messageTxt": "A valid payment method code was not provided.;"** **}**

400 Bad Request	The **Precedence Code** is invalid. More than 4 or less than 1, null, or is not integer.	"userMessage": {
		"messageTxt": "A valid precedence code was not provided.;"
		}

400 Bad Request	The **Distribution Percentages** for all the direct deposits sum up to more than 100.	"userMessage": {
		"messageTxt": "Total distribution cannot exceed 100%.;"
		}

400 Bad Request	**More than one Full Net type distribution provided when the following occurs:**
	• **Remainder** indicator is set to 1 and the **Amount** and **Percentage** fields are blank.
	• **Remainder** indicator set to 1 and **Percentage** field set to 100.
		"userMessage": {
		"messageTxt": "More than one direct deposit can not be added if 'All of my pay' is selected.;"
		}

400 Bad Request	More than one **remainderBalanceIndicators** are used in the pay distribution payload.	"userMessage": {
		"messageTxt": "More than one remainder type distribution cannot be set.;"
		}

400 Bad Request	Same precedence number is provided for more than one direct deposit.	"userMessage": {
		"messageTxt": "The same precedence cannot be set for multiple accounts.;"
		}

400 Bad Request	The **Financial Party** is missing in the request payload.	"userMessage": {
		"messageTxt": "Routing information is not provided.;"
		}

400 Bad Request	The invalid routing id is provided. The null or length not equal to 9.	"userMessage": {
		"messageTxt": "Valid Routing Number is not provided.;"
		}

400 Bad Request	The **Routing Number** starts with 5.	"userMessage": {
		"messageTxt": "Routing Number cannot start with 5.;"
		}

400 Bad Request	The **Bank Name** has more than 40 characters.	"userMessage": {
		"messageTxt": "Maximum length of the Bank name cannot exceed 40.;"
		}

400 Bad Request	Invalid characters entered in the **Bank Name**. The **Bank Name** contains characters other than Alphanumerics, space, and &.	"userMessage": {
		"messageTxt": "Bank Name has invalid characters.;"
		}

400 Bad Request	The PayDistribution is missing in the request payload.	"userMessage": {
		"messageTxt": "No pay distribution information is provided."
		}

400 Bad Request	The **Associate ID** is not set in the **Event Context**.	"userMessage": {
		"messageTxt": "Event context associateoid is not set."
		}

401 Unauthorized	The employee does not have ESS access.	"userMessage": {
		"messageTxt": "Direct Deposit is disabled for employees"
		}

	Invalid Credentials provided.	"userMessage": {
		"messageTxt": "Invalid authorization credentials"
		}

404 Not Found	Invalid employee AOID provided.	"userMessage": {
		"messageTxt": "Cannot find person using the given Associateoid - {{aoid}}"
		}

Chapter 3

Frequently Asked Questions

Question 1: How many types of direct deposits are supported and what is the maximum number of direct deposits that can be added through the API?

There are four kinds of direct deposit distributions supported by RUN Powered by ADP and the maximum limit supported through the Direct Deposit API is also four. Users can view these in RUN Powered by ADP by selecting Employees > Payroll > Direct Deposit > Deposit Distribution.

- **Full Net** - This gets selected in RUN Powered by ADP when:
  - **distributionAmount/amountValue** is set to employees full net amount
    Or
  - **distributionPercentage** is given as 100
    Or
  - **remainingBalanceIndicator** is set to true (when there are no other direct deposits configured for the employee)
- **Partial-$** - This gets selected in RUN Powered by ADP when **distributionAmount/amountValue** is set to some partial amount.
- **Partial-%** - This gets selected in RUN when **distributionPercentage** is given less than 100
- **Remainders** (Deposits the remaining amount to the account configured) - This gets selected in RUN Powered by ADP when **remainingBalanceIndicator** is set to true

Question 2: Which APIs can my application use to retrieve the actual amount getting distributed into each direct deposit accounts for a payroll run?

You application can use the Payroll Output API. See the sample PayrollOutput_Res.json response.

Question 3: Can my application add new direct deposit configuration for a worker with a future start date?

Yes, it's possible to add a direct deposit for an employee with future hire-date as long as the person is fully onboarded (not in the In Progress state, and was given the Employee Self Service access). See Process Overview for more information.

Question 4: Are direct deposit accounts verified?

ADP works with a third party to verify employees' direct deposit bank accounts. This verification checks the Account Number & Routing Number, usually in few seconds, whenever a new direct deposit is added to RUN Powered by ADP and marks it as one of the following:

- **Yes** - Validated, which Saves Direct Deposit details in RUN Powered by ADP.
- **No** - Direct deposit should not be saved in RUN Powered by ADP.
- **Don't know** - When it's not possible to confirm or deny. RUN Powered by ADP saves these, but marks them as **not-verified**.

These results are available to the practitioner in the RUN Powered by ADP login on each employee's **Direct Deposit** page.

API has a **prenotebypassIndicator** flag which should be set to **true** if the partner can verify and send the bank account details.

Question 5: Are there different API end points to use for add, update, and delete direct deposits?

The **/events/payroll/v1/worker.pay-distribution.change** end point must be used for add, update, and delete direct deposits as incremental updates are not allowed. For example, if there are three direct deposit accounts to be configured and two are already configured in the system, your application should send details of all three with their precedence code when updating the Direct Deposit Distribution. To remove all direct deposit information, your application should send a blank payload.

To remove a single direct deposit account when there are, for example, three available direct deposit accounts configured, your application should send the new change request with the distribution information of the remaining two accounts.

Question 6: How do you work with precedenceCode in the API Payload?

Precedence code tells RUN Powered by ADP the preferred direct deposit mode for the worker if more than one direct deposits are present. The **precedenceCode** can have values from 1 - 4.

For example, if a partner application is trying to set up four direct deposit accounts for a an employee, they could send details of all four in the **Change Pay Distribution** method call. When building the payload they could have the direct deposits positioned in a random manner, but assign a unique **precedenceCode** (between 1 to 4) for each direct deposit. On a successful POST, direct deposits would appear as **Direct Deposit One**, **Direct Deposit Two**, **Direct Deposit Three**, and **Direct Deposit Four** in the UI mapping to **precedenceCode** 1, 2, 3, and 4 respectively.

Note: The lower **precedenceCode** always gets the priority.

Chapter 4

Known Issues and Limitations

Information: This section details all reported and known issues, so developers can be aware during implementation.

Issue #1: Event Notifications are not supported by the Direct Deposit API

Issue Description

The current version of the Direct Deposit API does not support event notifications for the Pay Distribution/Direct Deposit information changes on the workaround shown in the next section.

Suggested Workaround

Your application should regularly retrieve all employee's direct deposit information from RUN Powered by ADP if data needs to be synchronized between your application and RUN Powered by ADP.

Issue #2: META response does not provide codeList for precedenceCode and paymentMethodCode

Impacted API

Method	URI	roleCode
GET	events/payroll/v1/worker.pay-distribution.change/meta	Practitioner
POST	events/payroll/v1/worker.pay-distribution.change	Practitioner

Issue Description

Change meta response doesn't provide any codeList for precedenceCode and paymentMethodCode.

Suggested Workaround

All clients have the same value, so your application could hard code the following values for your needs:

- Valid codeValues for precedenceCode are **1**, **2**, **3**, and **4**.
- Valid Value for paymentMethodCode/codeValue is **D**.
- Valid Value for paymentMethodCode/shortName is **Direct Deposit**.