Integrate Document Information Extraction

Project name: DOX

Package version: 1.1.12

THE BEST RUN SAP

Version	Date	Description
1	October 22, 2020	Document created
1.1	February 9, 2021	Add information about clientId (environment variable, step to execute before launching the agent)
1.2	March 05, 2021	Change template Add information about SDK dependencies
1.3	December 17, 2021	Add automation to extract data from document using built-in activities
1.4	July 1, 2022	Add automation to insert data into Excel file

TABLE OF CONTENTS

INTRODUCTION

This document describes the SAP Build Process Automation sample **DOX** and provides the following information:

- Description (functional and technical)
- Version used to generate this sample

It also contains information on prerequisites, such as the steps to follow before launching the agent.

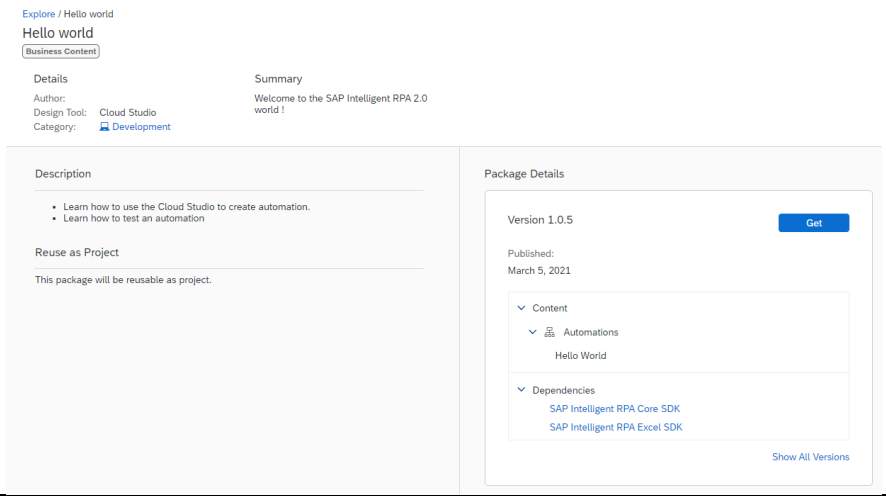
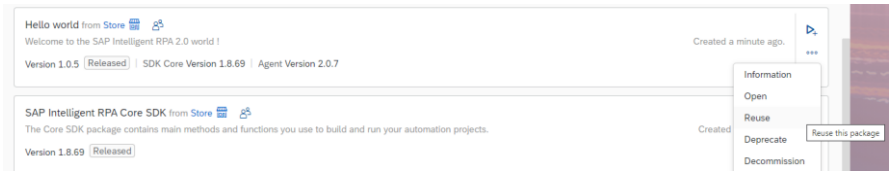
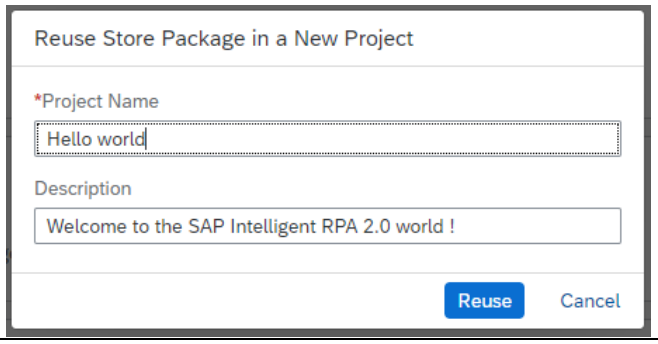
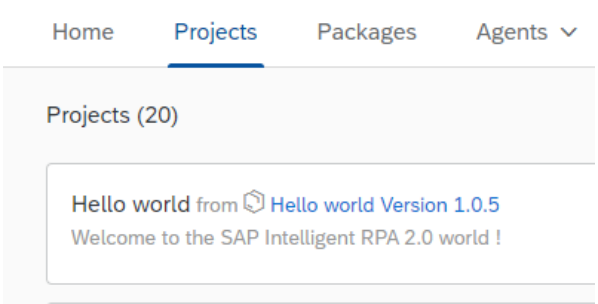
IMPORTANT RECOMMANDATION

General

To use this sample, you need to have a basic knowledge and understanding of SAP Build Process Automation tool. At the very least you need to know how to build an automation, add and modify activities and generate a package.

Reuse the sample as a new project

Note: screenshot might display a different name than the one of this sample.

From the Cloud Factory, open the Store tab and select the sample you want to retrieve. Click on the **Get** button.	
Once the package is retrieved, open the Packages tab of the Cloud Factory. Click on the Options button of the package you just retrieved and select the option **Reuse**.	
Set a name for the project to be created.	
Open the project that has just been created.	
If needed, update the content of this project, and generate a new package from it.	

You need to execute this procedure to be able to open the project and see all its content (the captured applications, the declared items, the automations, etc.).

DESCRIPTION

This package contains captures, datatype and automations that are described below. See chapter Version for more details about the version of the Desktop Agent and the SDK dependencies.

Settings

This section describes the settings of the project such as environment variables or dependent packages that are used in the automations.

Environment variables

Name	Type	Description
filePath	String	The path to the document to send to the Document Information Extraction service
clientId	String	Id of the client which was declared in Document Information Extraction service
uaaClientId	String	The uaa.cliendId field from the service key
uaaClientSecret	Password	The uaa.cliendSecret field from the service key
uaaUrl	String	The uaa.url field from the service key
serviceUrl	String	The url field from the service key

Dependent packages

N/A

Captures

This section describes the captures which were made to pilot the application in this sample. It will also describe the different methods which were used to capture the pages and declare the items.

Datatypes

This section describes the datatype used in this sample. It describes the structure of the datatype and where it is used in the automations.

doxResult

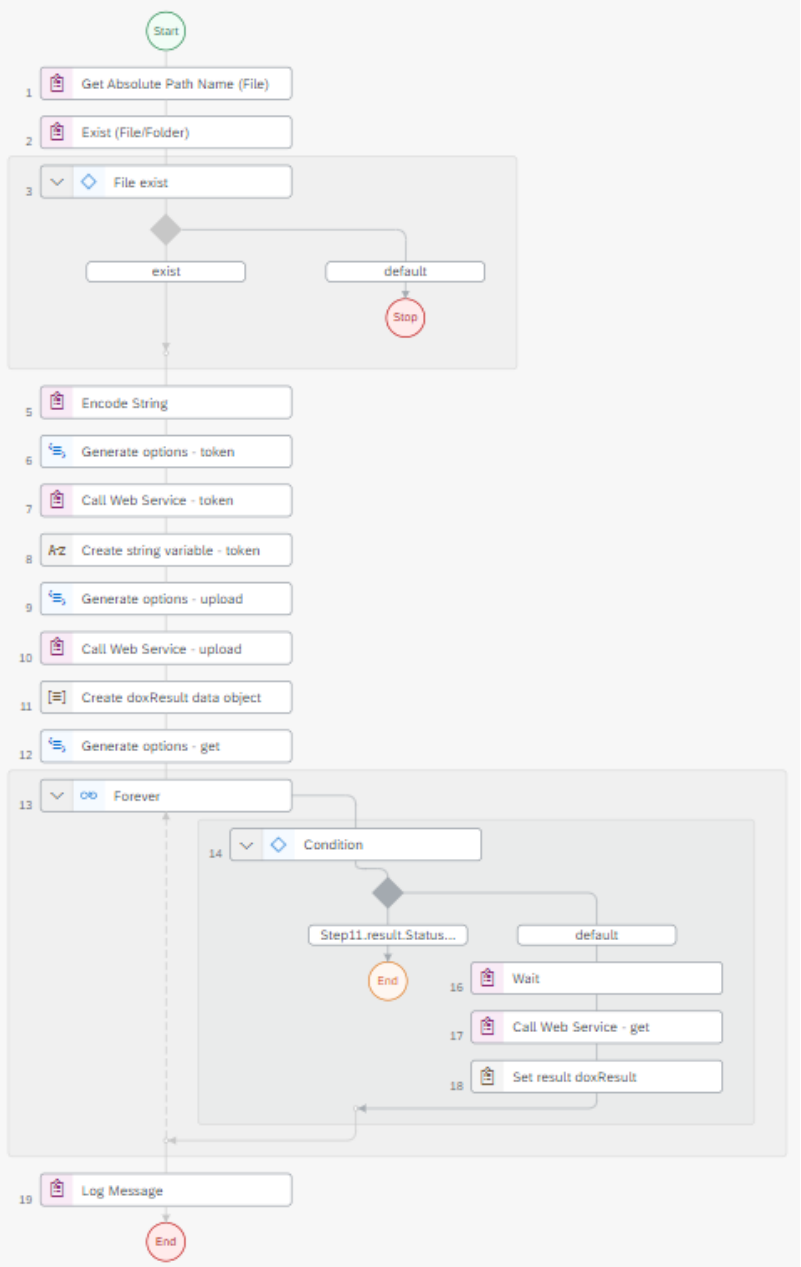
Name	Type	Description
Status	String	Required. Status of the service for the processing of the document submitted in the automation. See service documentation for more details. Note: this is the content of the status field (see documentation)
Data	Any	The data extracted from the document. See service documentation for more details. Note: this is the content of the extraction field (see documentation)

Automations

Dox

Type: Attended
Input: None
Output:

Name	Type	Description
serviceResult	doxResult	JSON containing the data extracted from the document and also the status of the extraction (DONE or FAILED). See service documentation for more details.



The agent first encodes the *clientId* and *clientSecret*.

Then it executes a query to get the token which will be used to authenticate to the service.

The file is then uploaded to the service and the document ID is retrieved from the response.

At this point, we create a variable of type *doxResult* (see datatype chapter) and we initialize the status with PENDING value as it always takes some time for the service to process the document.

Then the agent will periodically (every 3 seconds) ask the service for the results.

The `status` and `extraction` fields are stored in the *doxResult* variable (in *Status* and *Data* attributes)

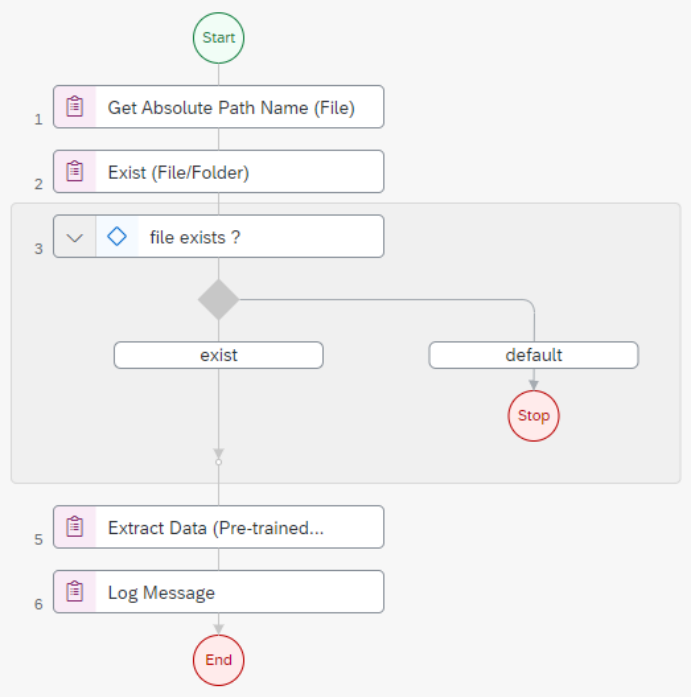
When the Status is not PENDING anymore (could be DONE or FAILED), the agent exits the loop and the automation end.

Note: when this automation is used in another one, it is highly recommended to check the value of **result.Status**.

If it is DONE, then you can proceed. Otherwise, an error was raised during the process.

Extract data using activities
Type: Attended
Input: FilePath
Output:

Name	Type	Description
data	Invoice Data	JSON containing the data extracted from the document

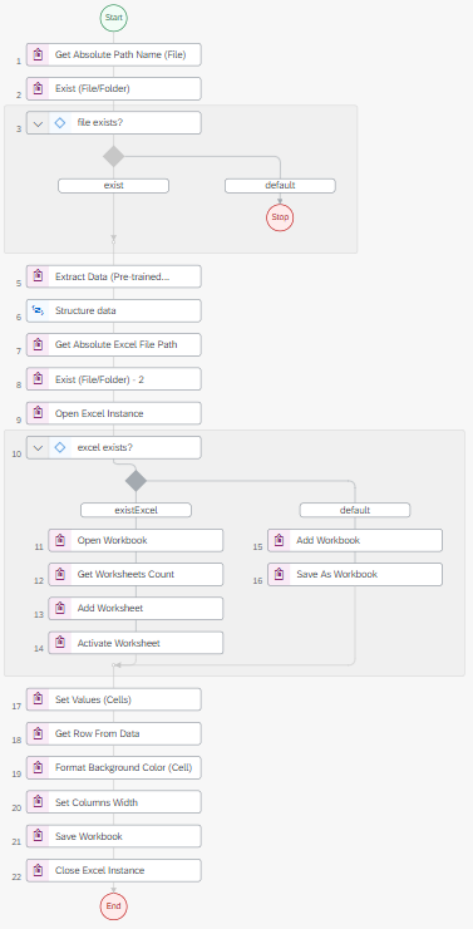


The agent first makes sure the file exists, and it will use the built-in activity to upload the document to DOX service.

The agent will continue the execution of the automation once it gets the data from the DOX service.

.

Extract data from document to Excel
Type: Attended
Input: FilePath, ExcelFilePath
Output: N/A



In a similar way, the bot will extract data from a file which path is given as input parameter.

Once it is done, it will structure this data into a table having the following format:

Header	Value	Confidence

And this table will be written in an Excel file which location is given as input parameter as well.

Note: To change the structure of data from one JS object to a collection of array of string, we are using a Custom Script activity.

VERSION

The product versions used to generate this sample are detailed below. This sample is provided “as is”, with no warranty that it will work correctly with other versions. If some versions of your software are different (such as the tool version or the target application version), you may need to recapture the application and/or update the workflow activities.

SAP Build Process Automation

This sample targets the Desktop Agent **2.0.8** or higher.

The following SDK dependencies were used to generate this sample: 1.26.51

See documentation for more details about the compatibility between SDK version and Desktop Agent.

Target application

Document Information Extraction service	Date of last test: 17 December 2021

PREREQUISITES

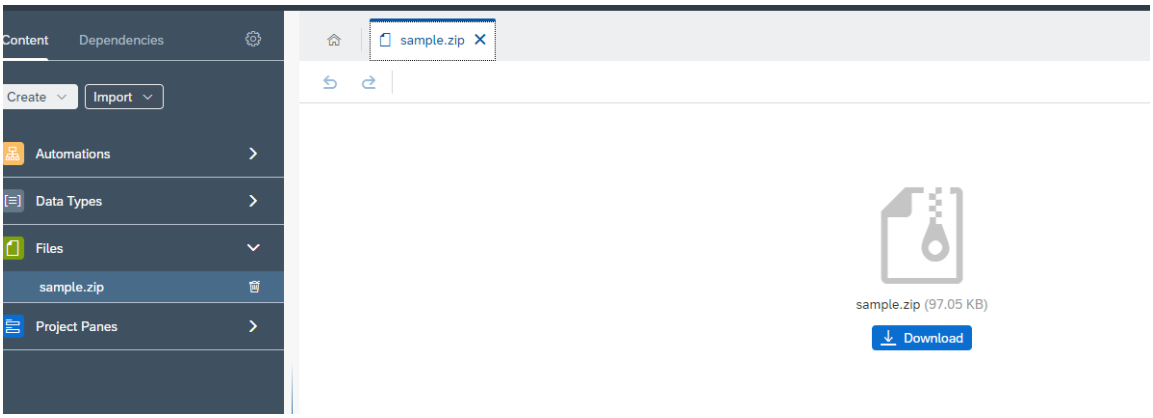
Global setup

SAP Build Process Automation must be installed in accordance with the installation guide available here. An SAP Build Process Automation Factory must be available with a suitable environment (containing an agent). All information can be found in the “Getting Started” section accessible via the above link.

Specific steps to follow before launching the agent

1. Set up an instance of the SAP Document Information Extraction service on your tenant
2. Generate the service key and get the following information from the key:
    a. url
    b. uaa.clientid
    c. uaa.clientsecret
    d. uaa.url
3. Following the documentation of the Document Information Extraction service, create a client (with a label and an id, which will be used in an environment variable)

    Note: This can be done from Postman for example, of from the service user interface. In both cases, don’t forget to retrieve the access token as described here.
4. Download the **sample.zip** archive from the sample and extract its content. Save the location of the invoice.pdf file as it will be used in an environment variable



5. When you deploy the package, set the environment variables with the following information:

filePath	The location of the **invoice.pdf** file you extracted from **sample.zip**
clientId	The id of the client that was added in the Document Information Extraction service database.
serviceUrl	url from the service key
uaaClientId	uaa.clientid from the service key
uaaClientSecret	uaa.clientsecret from the service key
uaaUrl	uaa.url from the service key

6. You can also modify the value of the options in the activity **Generate options – upload**, according to the documentation.

EXPECTED OUTPUT
You should get a variable (type *doxResult*) with:
Status = DONE
Data = content of the document

www.sap.com/contactsap

THE BEST RUN SAP®