File System

Project name: File System - Cloud Studio

Package version: 1.0.1

THE BEST RUN SAP

Version	Date	Description
1	July 19, 2021	Document created

TABLE OF CONTENTS

INTRODUCTION

This document describes the SAP Build Process Automation sample **File System - Cloud Studio** 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.

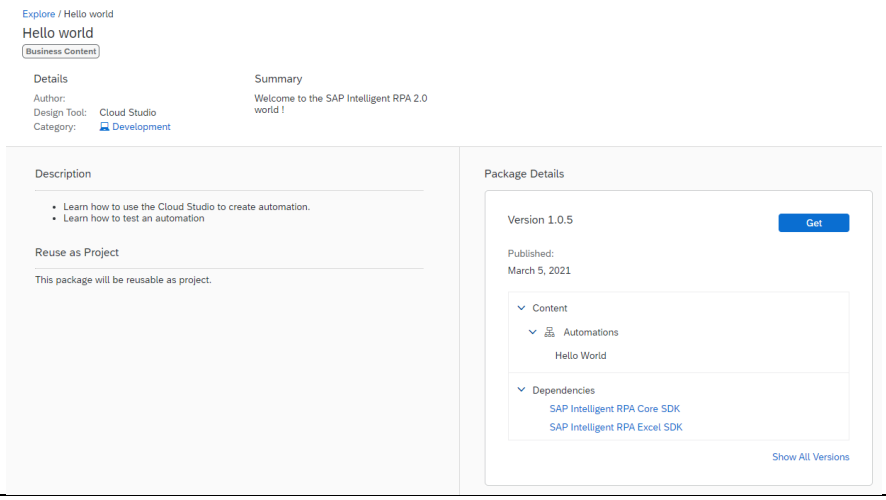
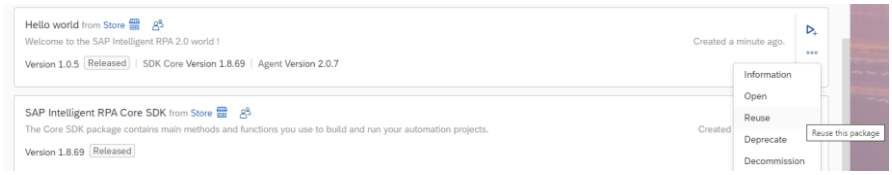
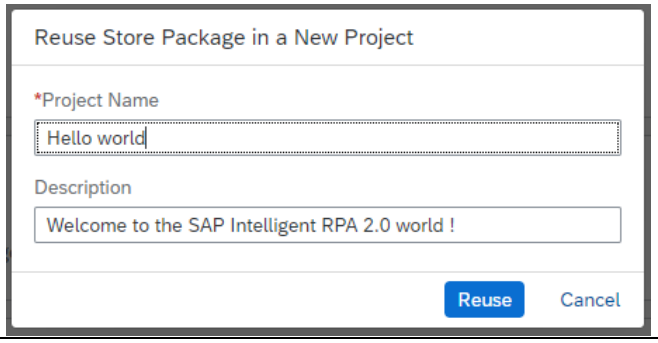
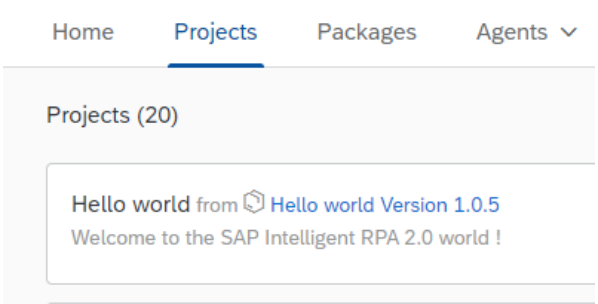
IMPORTANT RECOMMANDATION

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 automation.

Environment variables

N/A

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.

N/A

Datatypes

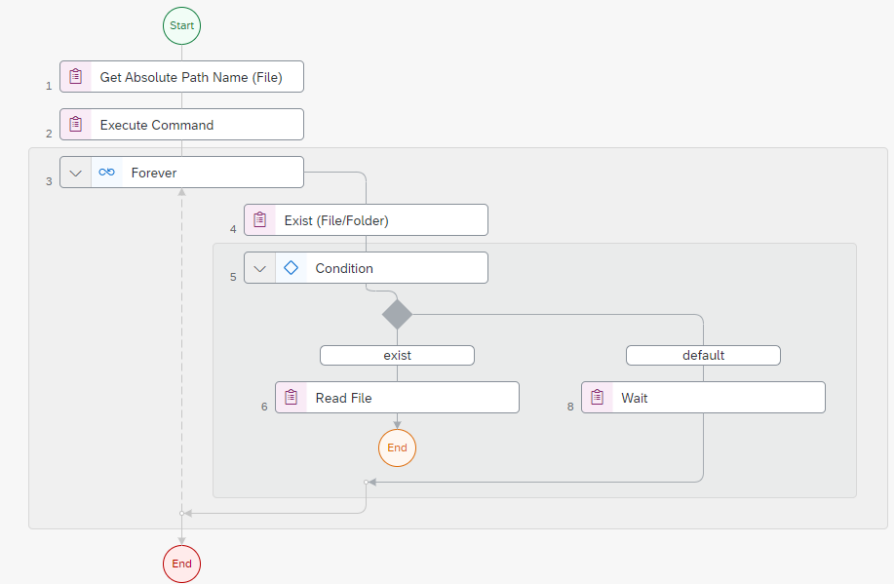
This section describes the datatype used in this sample. It describes the structure of the datatype and where it is used in the automations.

Excel data

Name of attribute	Type	Description
Attribute1	String	
Attribute2	String	
Attribute3	String	
Attribute4	String	
Attribute5	String	

Automations

Execute batch command
Triggerable: No
Input: OutputFileName (type = String), outputPath (type = String)
Output: None

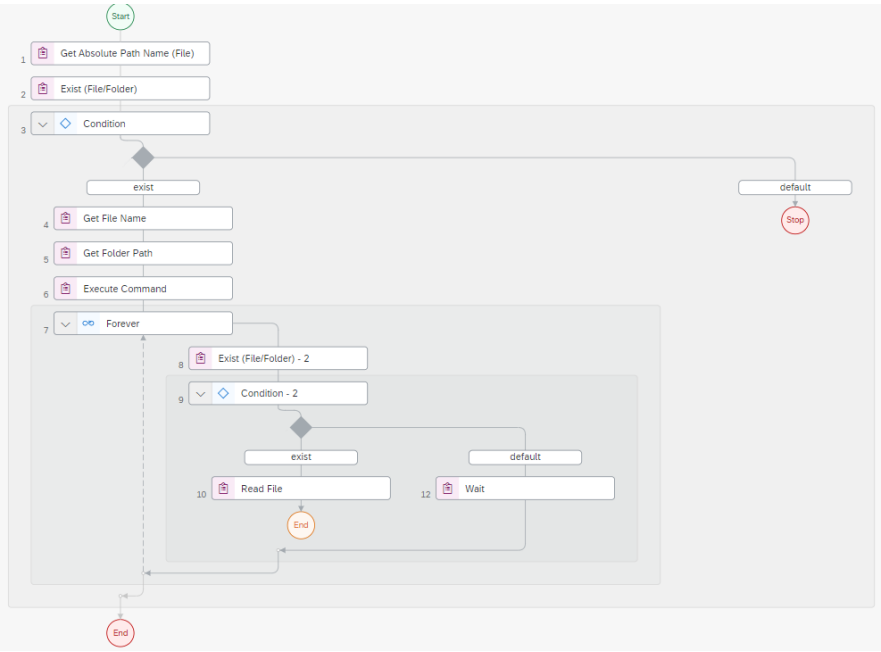


The agent will use the outputPath given as input to get an Absolute Path of it to ensure that the path will be formatted for the next steps.

Then it will execute a simple command that will print an “Hello” and redirect the output in the file given as input. If the file doesn’t exist, it will be created.

Next, we do a forever condition to wait for the end of the execution of the command. We check if the output file exists, if it exists, we read it. This way we can retrieve the output of the command.

Execute PowerShell script
Type: Attended
Input: scriptPath (type = String), outputFIleName (type = String)
Output: None

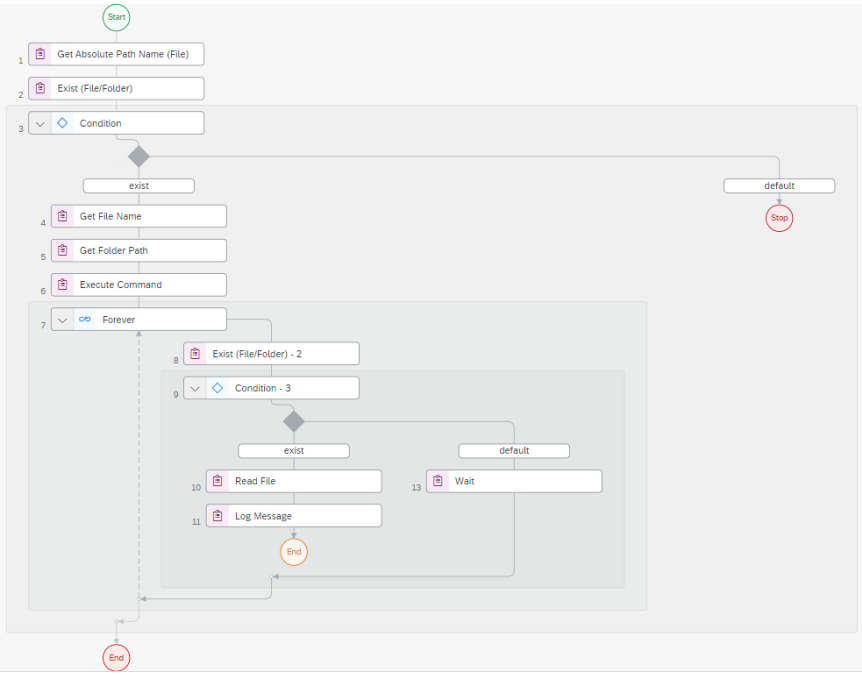


The agent gets the absolute path of the input scriptPath so we are sure to have a good path for the next steps. In addition, we test if the script file exists, if not we stop the automation.

To execute the script, we retrieve first the name of the script file and the folder path, then we execute the script by executing the command in a powershell and redirecting the output in an output file given as input of the automation. The output file is in the same folder as the script.

Next, we do a forever loop to wait for the end of execution of the command by checking if the output file exist or not. When it is created by the execute command step, we read the file to retrieve the output of the script

Execute python script
Type: Attended
Input: scriptPath (type = String), outputFileName (type = String), inputScript (type = String)
Output: None

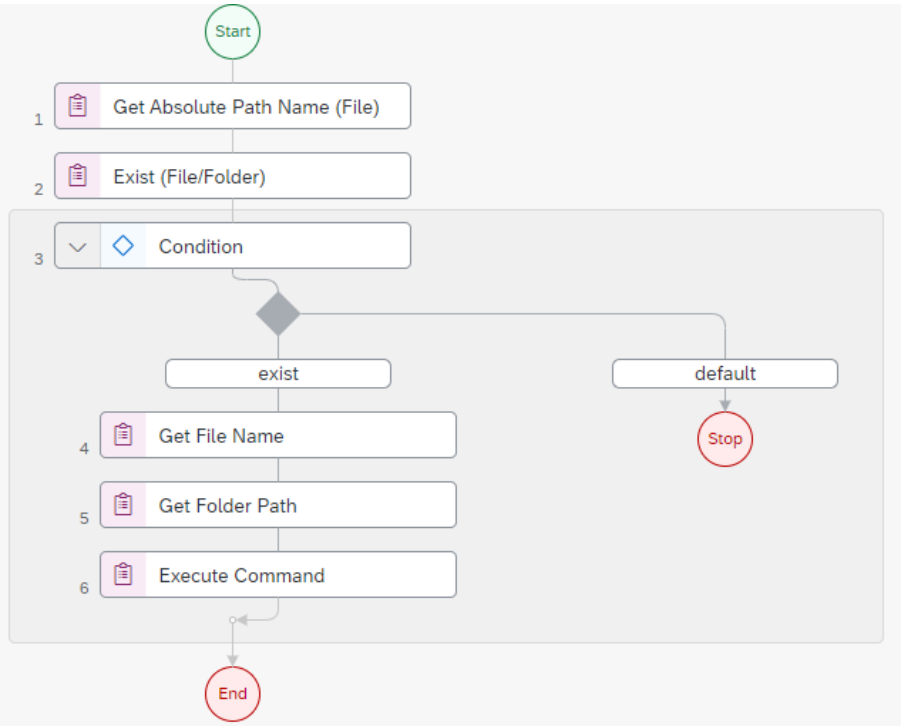


In this automation, we test if the sciptPath file exist, if not we stop the execution of the automation.

To execute the script, we retrieve first the name of the script file and the folder path. Then we execute the script by executing the command in a powershell and redirecting the output in an output file given as input of the automation. The output file is in the same folder as the script.

Next, we do a forever loop to wait for the end of execution of the command by checking if the output file exist or not. When it is created by the execute command step, we read the file to retrieve the output of the script and then we log the output.

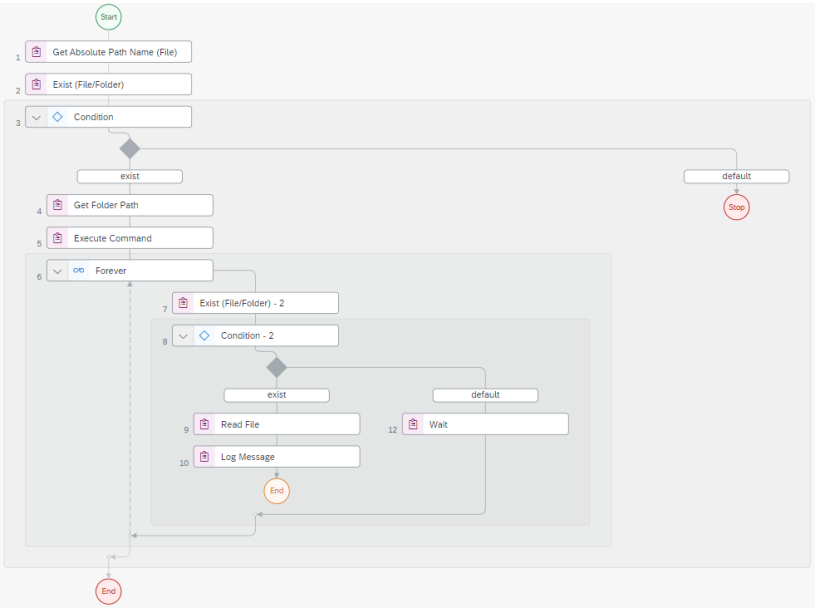
Execute VB script
Type: Attended
Input: scriptPath (type = String)
Output: None



In this automation, we get the absolute path of the script and then test if the file exists, if not we stop the workflow.

To execute the script, we retrieve first the name of the script file and the folder path. Then we execute the script by executing the command in a powershell.

Get metadata by command line
Type: Attended
Input: filePath (type = String), outputFileName (type = String)
Output: None

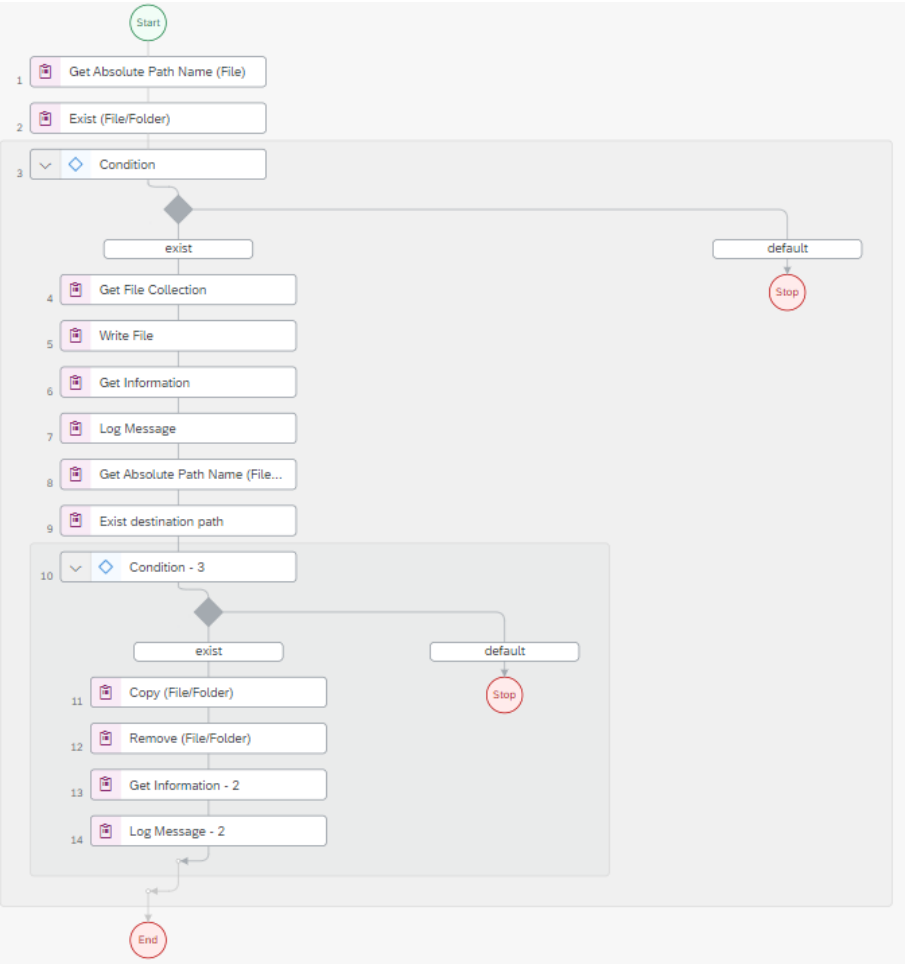


In this automation, we get the absolute path of the script and then test if the file exists, if not we stop the workflow.

We retrieve the folder path and then execute the command line to get the metadate. Here in this example we use the Get-Item command to get information of the mode of the file given as input. We redirect the output of the command in the output file.

Then in a forever loop we check of the output file exists, if it exist, we read it and log the content.

Move Folder
Type: Attended
Input: folderPath (type = String), newFolderName (type = String), destinationPath (type = String)
Output: None

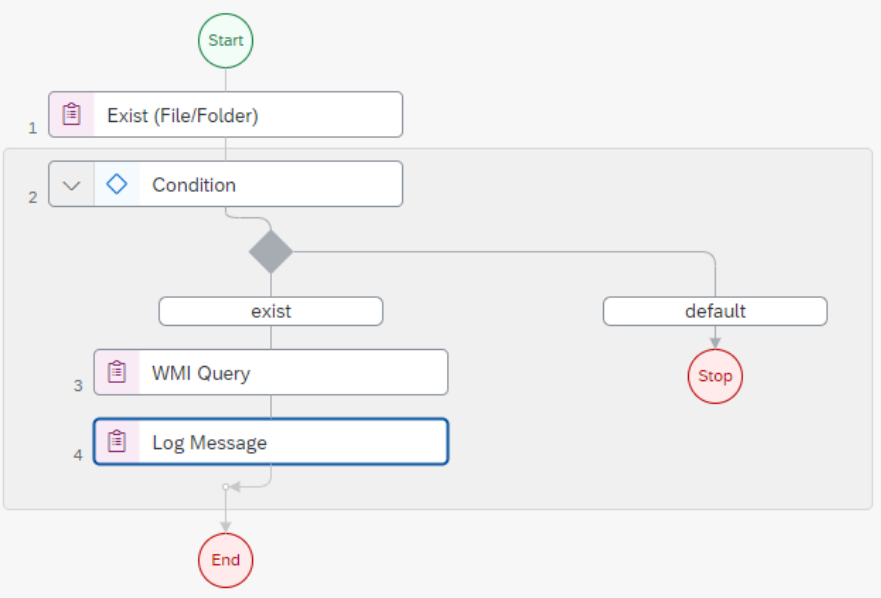


In this automation, we get the absolute path of the folder and then test if it exists, if not we stop the workflow.

Next, we get the files in the folder with the Get File Collection activity, we create a file “destination.txt” where we then write the destination path given as input. We do a Get Information on the folder to have the time of the last modification on the folder, we log this information. Then we get the absolute path of the destination and check if the destination exists. Next, we copy the folder to the destination and remove the destination file in the copied folder.

Lastly, we do a Get Information activity on the new folder and we print the last time it was modified.

WMI Query
Type: Attended
Input: filePath (type = String)
Output: None



In this automation, we test if the filePath given exists, if not we stop the workflow.

Next, we do the WMI Query activity to retrieve information. The path given to this activity must have double backslashes like this “C:\\example\\example.txt”

Lastly, we log the information retrieve by the query.

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 **2.0.14** or higher.

The following SDK dependencies were used to generate this sample:

irpa_core	1.13.60
irpa_excel	1.13.60
irpa_outlook	N/A
irpa_pdf	N/A
irpa_ui5	N/A
irpa_word	N/A

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

Target application

N/A

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

- Download the **sample.zip** archive from the sample and extract its content
- You will find example of simple script to test the automation as well as a folder for the move folder automation.

EXPECTED OUTPUT
N/A

www.sap.com/contactsap

THE BEST RUN SAP®