How to use XML files

Project name: How to use XML files

Package version: 1.0.4

THE BEST RUN SAP

Version	Date	Description
1	March 11, 2021	Document created

TABLE OF CONTENTS

INTRODUCTION

This document describes the SAP Build Process Automation sample **How to use XML files** 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.

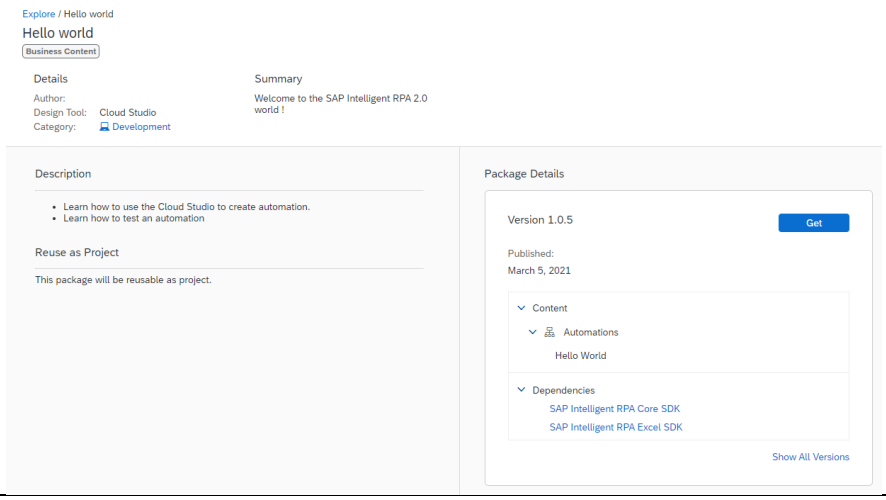
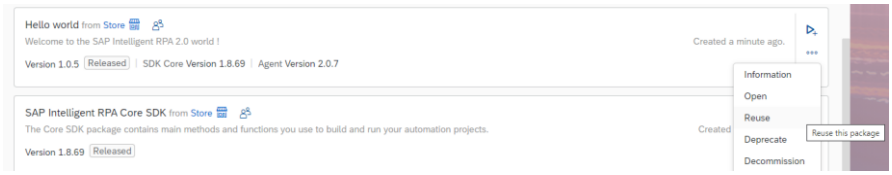
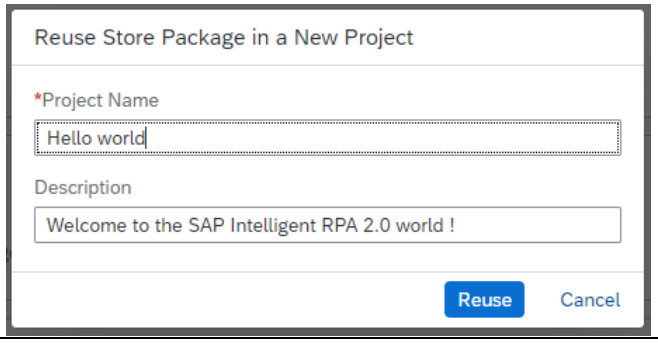
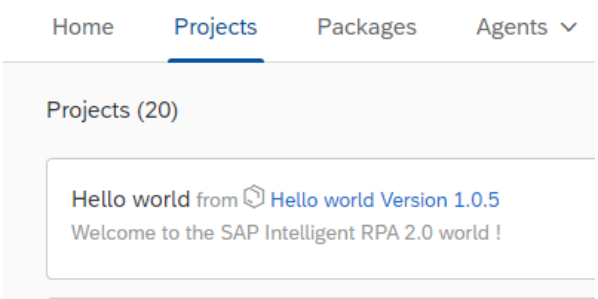
IMPORTANT RECOMMANDATION

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

Name	Description	Type
BookCatalog	List of books that are to be used in the project	Books catalog
FilePath_Read	Path of the XML file to be read in the automation	String
FilePath_Write	Path of the XML file to be written in the automation	String

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.

Book Information

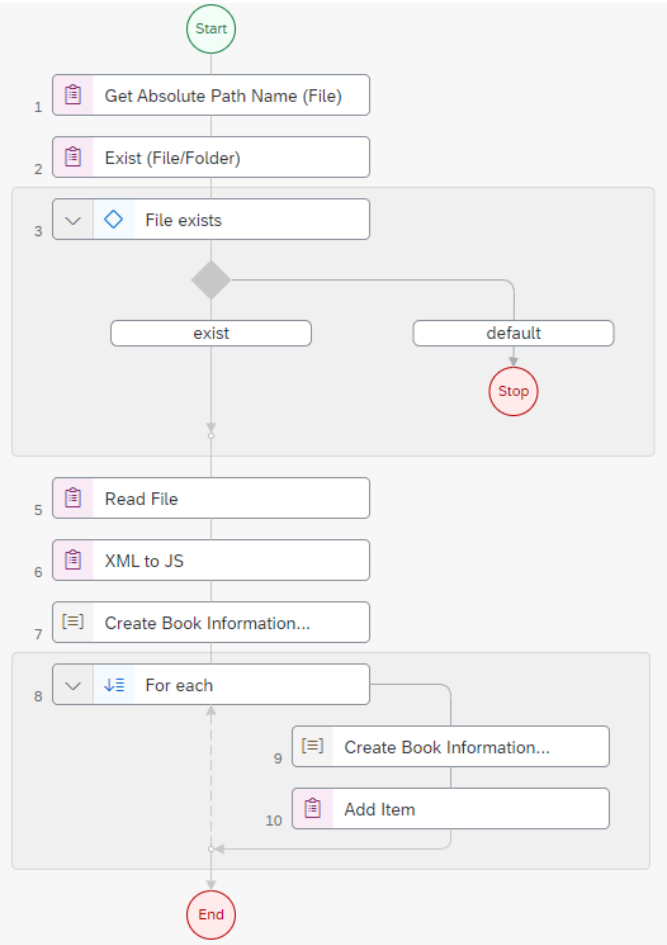
Name of attribute	Type	Description
Id	String	
author	String	
title	String	
genre	String	
price	Number	
publish_date	String	
description	String	

Books catalog

Name of attribute	Type	Description
book	Book Information [ ]	List of Book Information

Automations

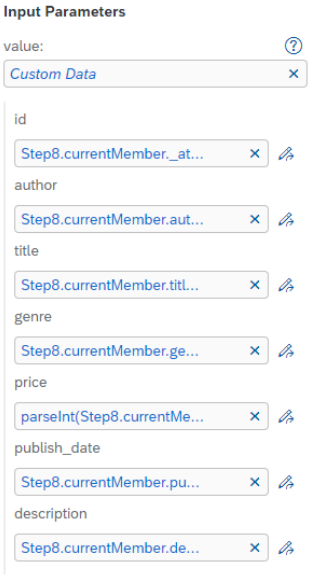
Read file and get JS object
Type: Attended
Input: None
Output: List of *Book Information* objects



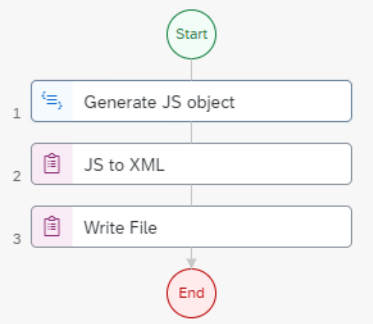
The agent reads an XML file (path of the file is an environment variable) and parse it to create a JS object.

Then it will create a list of **Book Information** objects using these data.

Note: The input XML file contains data as attributes and as node text. It's the reason why the setup of activity Create Book Information is the following :



Write JS object to XML file
Type: Attended
Input: None
Output: None



Using environment variable as input parameter, the agent creates a JS object with a specific structure to be able to write it as XML in a file.

The content of the Generate JS object activity is the following:

```
let lst = catalog.book.map(b=>{
    let res = {
        _attributes:{}
    };
    for(var att in b){
        if (b.hasOwnProperty(att)){
            if (att == 'id'){
                // 'id' should be an attribute of the 'book' tag
                res._attributes[att] = b[att];
```

```
            } else {
            // other information should be text of their own xml tag
                res[att] = { _text : b[att] };
            }
        }
    }
    return res;
});
return { catalog: { book: lst } };
```

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 **2.0.9** or higher.

The following SDK dependencies were used to generate this sample:

irpa_core	1.8.69
irpa_excel	1.8.69
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

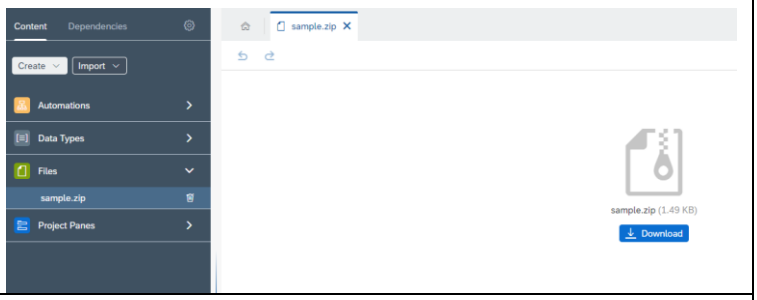
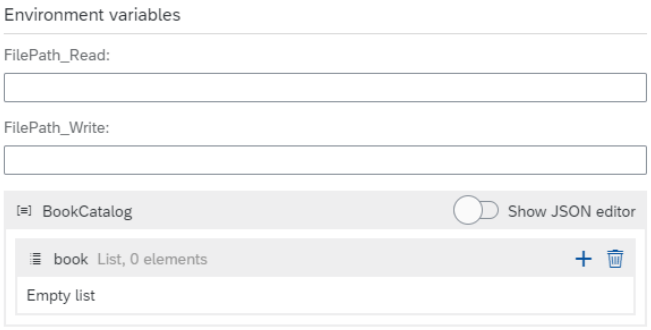
N/A

PREREQUISITES

Global setup

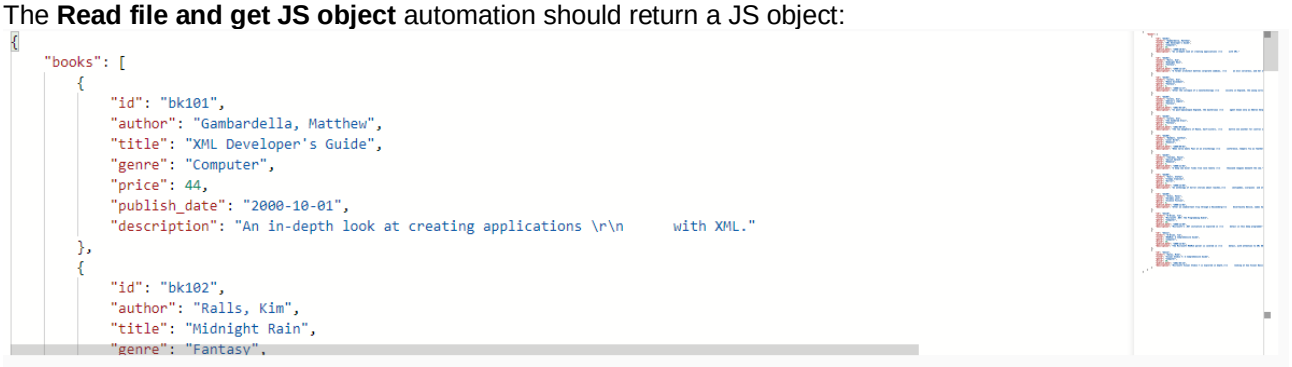
SAP Build Process Automation must be installed in accordance with the installation guide available here. An SAP Build Process Automation Factory must be available with a suitable environment (containing an agent). All information can be found in the “Getting Started” section accessible via the above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

Specific steps to follow before launching the agent

Download the **sample.zip** archive of the project and extract its content.	
When you deploy your package, set the values for the environment variables. **FilePath_Read** should be set with the location of the file you extracted in the previous step **FilePath_Write** should be set with the path of the file which will be generated. Ex: C:\Temp\output.xml **BookCatalog** should be set with whatever value you might want.	

EXPECTED OUTPUT

The **Read file and get JS object** automation should return a JS object:




```
{
    "books": [
        {
            "id": "bk101",
            "author": "Gambardella, Matthew",
            "title": "XML Developer's Guide",
            "genre": "Computer",
            "price": 44,
            "publish_date": "2000-10-01",
            "description": "An in-depth look at creating applications \r\n      with XML."
        },
        {
            "id": "bk102",
            "author": "Ralls, Kim",
            "title": "Midnight Rain",
            "genre": "Fantasy",
```


The **Write JS object to XML** file automation should produce an XML file at the location specified by the environment variable **FilePath_Write**. This file should have a structure like the following:

```
<catalog>
    <book id="bk1">
        <author>Douglas Adams</author>
        <title>The Hitchhiker's Guide to the Galaxy</title>
    </book>
</catalog>
```

www.sap.com/contactsap

THE BEST RUN SAP®