

Lab Guide

Exploring the RPA Asynchronous API

Nigel T. Crowther

Hands-on Lab

Version 1.0 for General Availability





NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

© Copyright International Business Machines Corporation 2020.

This document may not be reproduced in whole or in part without the prior written permission of IBM.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



Table of Contents

1 Introduction	4
2 The RPA Asynchronous API Overview	5
2.1 Prerequisites	6
2.1.1 Download Lab Materials.....	6
2.1.2 Download Cygwin	7
2.1.3 Optional - Download SoapUI.....	8
2.1.4 Optional - Download Eclipse IDE	9
3 Exploring the RPA API.....	10
3.1 Create a bot process.....	10
3.2 Test the RPA API from PowerShell.....	10
3.2.1 Step 1 – Host Selection	10
3.2.2 Step 2 – Enter your tenant credentials	11
3.2.3 Step 3 – Select the tenant.....	11
3.2.4 Step 4 – Select the process	12
3.2.5 Step 5 – Edit the payload.....	12
3.2.6 Step 6 - Run the bot.....	13
3.2.7 Step 7 – View the Result.....	13
3.2.8 View the RPA API Curl	15
3.2.9 Run curl.....	15
3.3 Testing the API from SoapUI	17
3.3.1 Open SoapUI.....	17
3.3.2 Setting Project Variables.....	19
3.3.3 Run the test suite	20
3.3.4 Examine the test suite flow.....	23
3.4 Generating an Open API Specification.....	25
3.4.1 Start Eclipse	25
3.4.2 Run curl from OpenApi	32
3.4.3 Test Api with JUnit.....	35
3.5 Integrate to the World!	38



1 Introduction

In this lab you will **explore the RPA Asynchronous API**. You will learn how to:

- Apply tools such as PowerShell, *curl* and *SoapUI* to test the RPA API
- Generate an *OpenApi* specification to integrate your bot to the modern world.
- Test the API using *Java* and *Junit*.

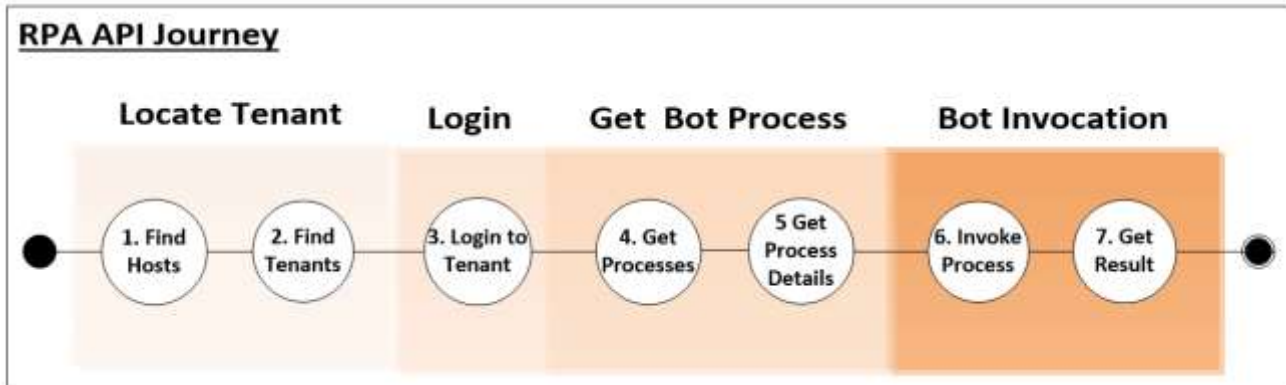
The lab is based on the IBM RPA API documentation here:

<https://www.ibm.com/docs/en/rpa/21.0?topic=automation-rpa-api-reference>

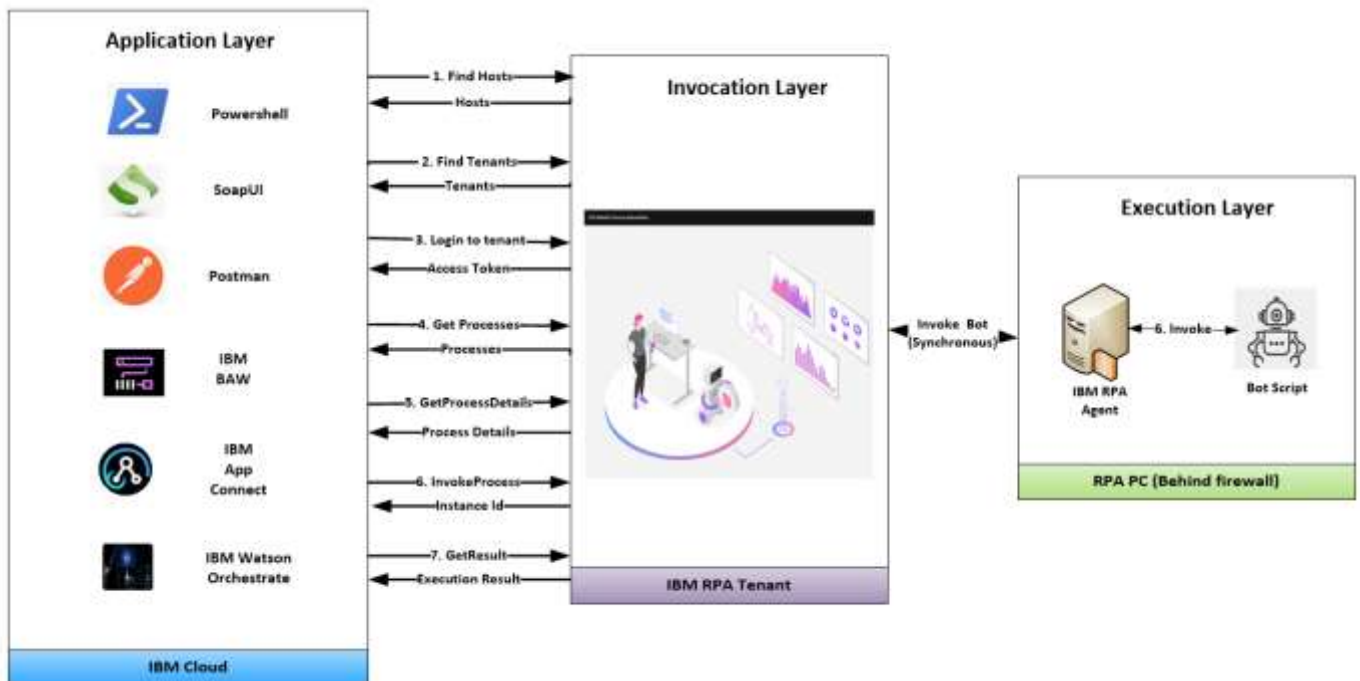


2 The RPA Asynchronous API Overview

To run a bot Asynchronously you follow a journey. The journey starts with finding a host and ends with bot execution. The following flow diagram describes each step:



The API is invoked against the RPA tenant. The tenant authenticates the caller, locates the process in which the bot resides and then permits invocation of the bot. The tenant finds the designated computer and invokes the internal synchronous API to run the bot. Once the bot is run, the result is retrieved asynchronously by invoking *GetResult*. This interaction is depicted below:



You will explore this API flow in the next sections.



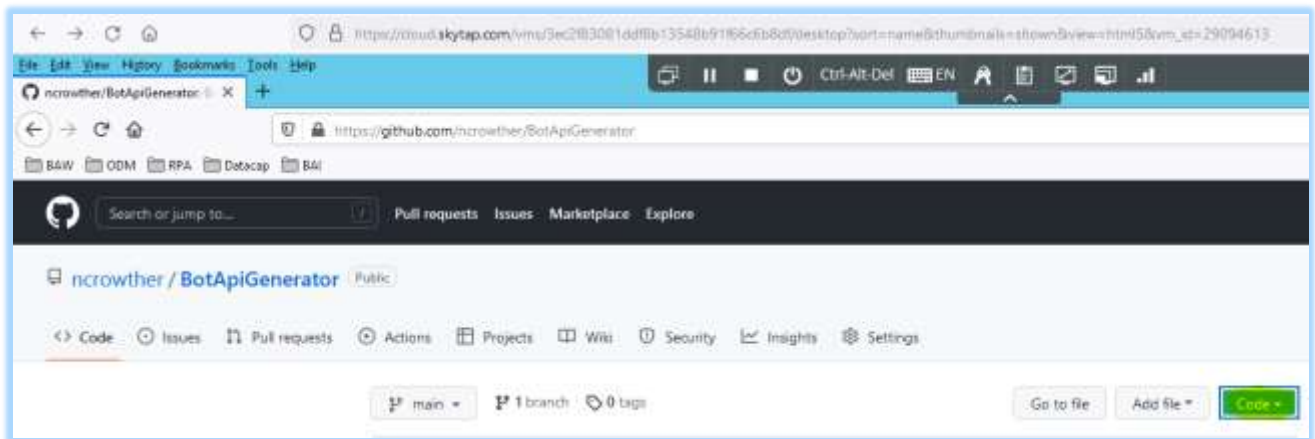
2.1 Prerequisites

2.1.1 Download Lab Materials

The prerequisite material for this lab is stored in **Github**. Using a browser, navigate to the folder:

<https://github.com/ncrowther/BotApiGenerator>

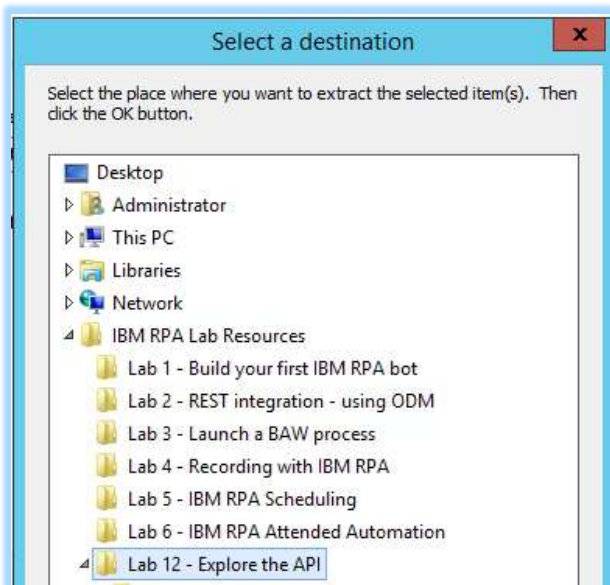
You should see the following:



Click on the **Code** button, and then *Download Zip*. Save the file. Once downloaded, right-click the zip and select *extract all....* Extract the zip to a local folder. If using the Skytap image, we recommend:

C:\Users\Administrator\Desktop\IBM RPA Lab Resources\Lab 12 - Explore the API

See below:



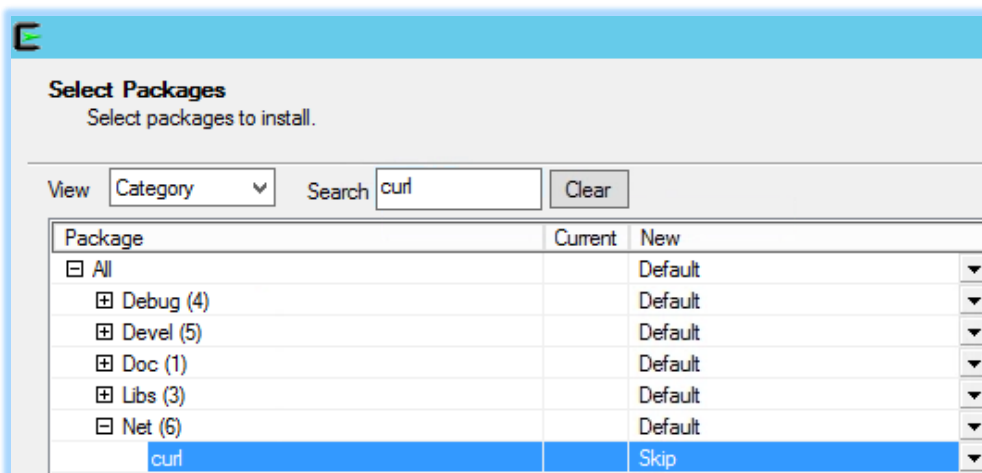
2.1.2 Download Cygwin

Using a browser, download and run:

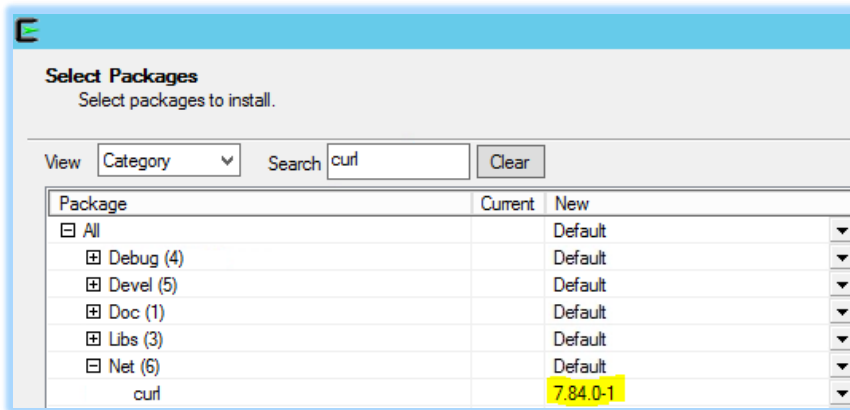
https://www.cygwin.com/setup-x86_64.exe

Use all default installation settings. When it comes to selecting a download site, select your most local server.

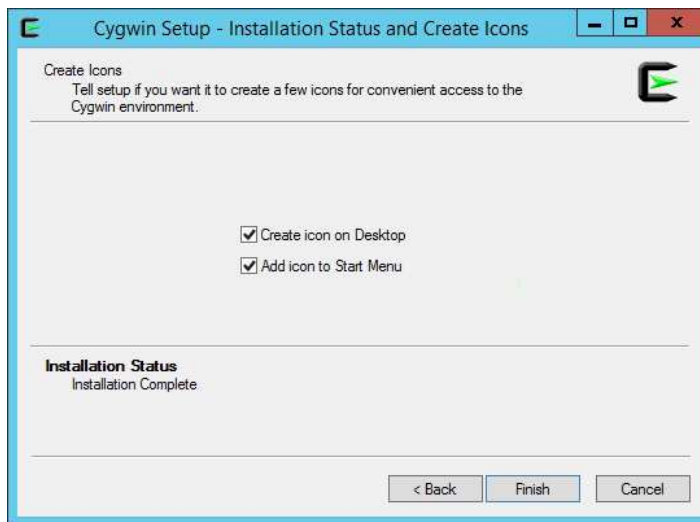
When selecting packages, enter **curl** as the search criteria and expand the package tree. You should see the following:



Curl is in Skip state which means it won't be installed by default. Change 'Skip' to the latest version as highlighted below.



Now click *Next* and continue pressing defaults until the installation window starts. After a minute you should see this:



Press Finish. The install is complete!

2.1.3 Optional - Download SoapUI

This step is not required if you are using the Skytap image.

You can download an open source version of SoapUI from the following location:

<https://www.soapui.org/downloads/soapui/>

Click on *Download SoapUI Open Source*.

Save the zip, extract and install the software using the default settings.



2.1.4 Optional - Download Eclipse IDE

This step is not required if you are using the Skytap image.

You can download Eclipse from the following location:

<https://www.eclipse.org/downloads/>



3 Exploring the RPA API

3.1 Create a bot process

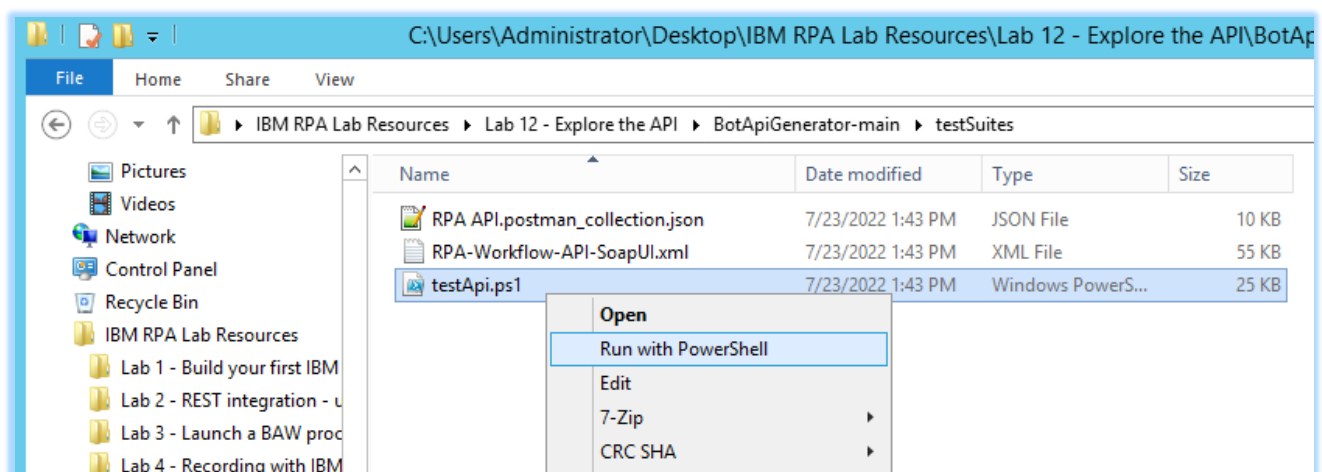
To start your exploration, you will need to create a bot process. Please follow steps **2.1 to 2.4** in lab:

https://github.com/juseljuk/IBM-RPA-Toolkit-for-BAW/blob/master/downloads/Using%20IBM%20RPA%20with%20IBM%20BAW%201_1.pdf

3.2 Test the RPA API from PowerShell

Navigate to the Folder in which you unzipped the GitHub Repo in the pre-requisites step:

Within the *testsuites* folder, Right-click *testApi.ps1* and select *Run with PowerShell*:

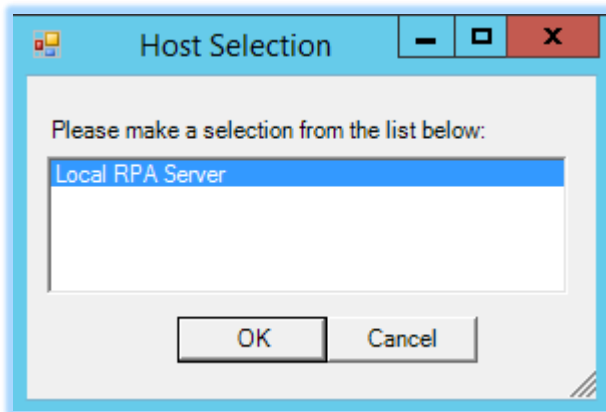


Ignore the security warning and press *Open*. A series of dialog boxes should appear. Let's follow the steps.

3.2.1 Step 1 – Host Selection

If you are running the Skytap image you will not be able to see the SaaS tenants. Press OK when the message “*Unable to find SaaS tenants*” appears.

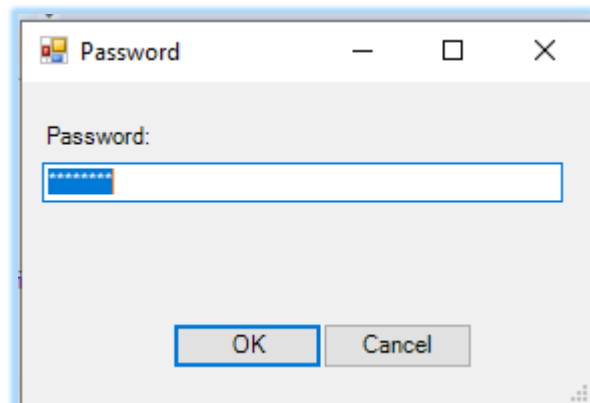
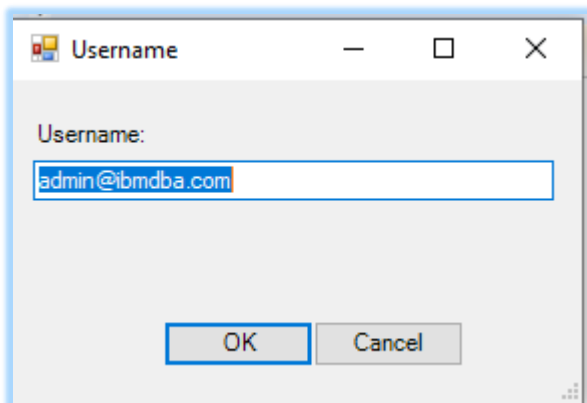
Select *Local RPA Server*



Tip: You need to select it so that it turns blue as shown above.

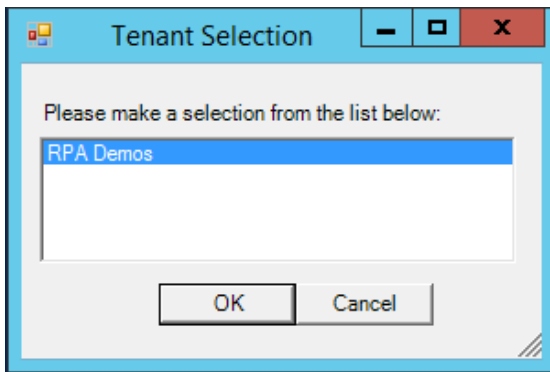
3.2.2 Step 2 – Enter your tenant credentials

The username is admin@ibmdba.com, the password is *passw0rd*. These should be set as default values:



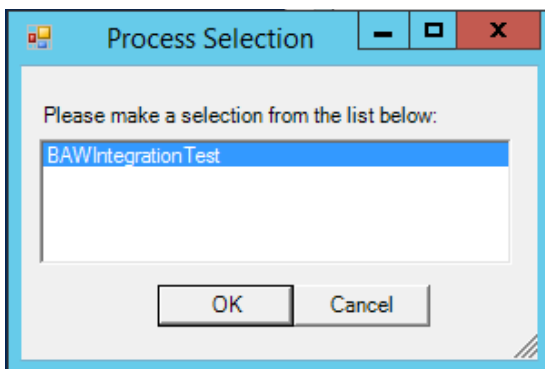
3.2.3 Step 3 – Select the tenant

Select tenant. If you are running within the Skytap image you only have one option. Select *RPA Demos*:



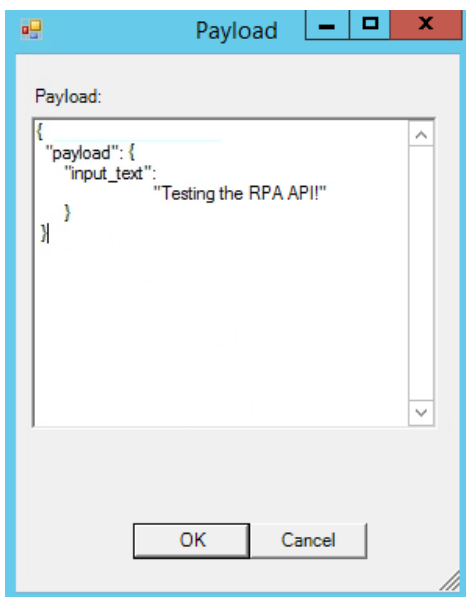
3.2.4 Step 4 – Select the process

Select the process you defined in the earlier step:



3.2.5 Step 5 – Edit the payload

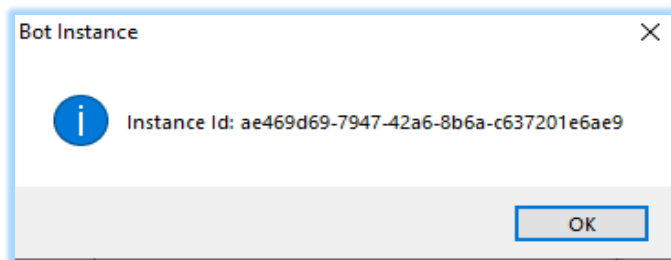
The payload will be generated from the bot input parameters. You can modify the input value if you wish:





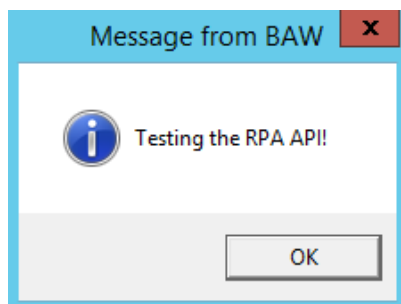
3.2.6 Step 6 - Run the bot

The bot runs and a dialog box appears showing the instance id of the running process

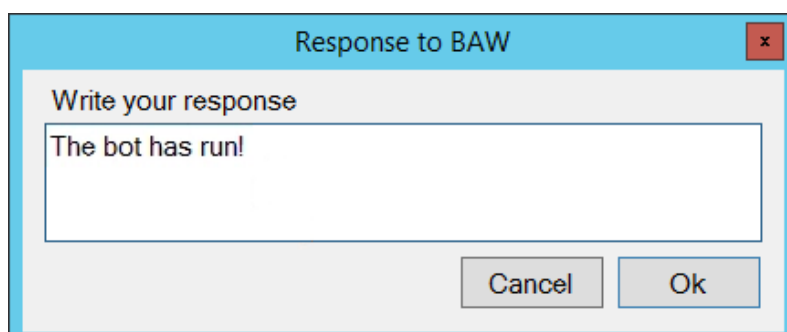


3.2.7 Step 7 – View the Result

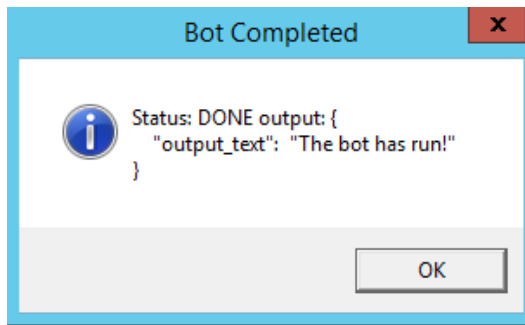
After a second, the bot runs, and you will see the following message:



Click OK and type your response:



The bot may take several seconds to run, in which case it will either be in *new* or *processing* state. When completed, the state will have a status of *done* and you will see the following dialog:



Press OK to finish.

Congratulations, you have run your first bot using PowerShell to drive the asynchronous API!

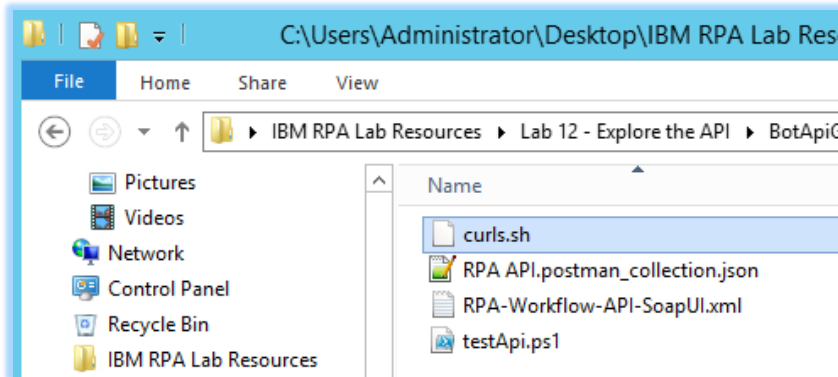
Next you will view the actual RPA API commands executed.



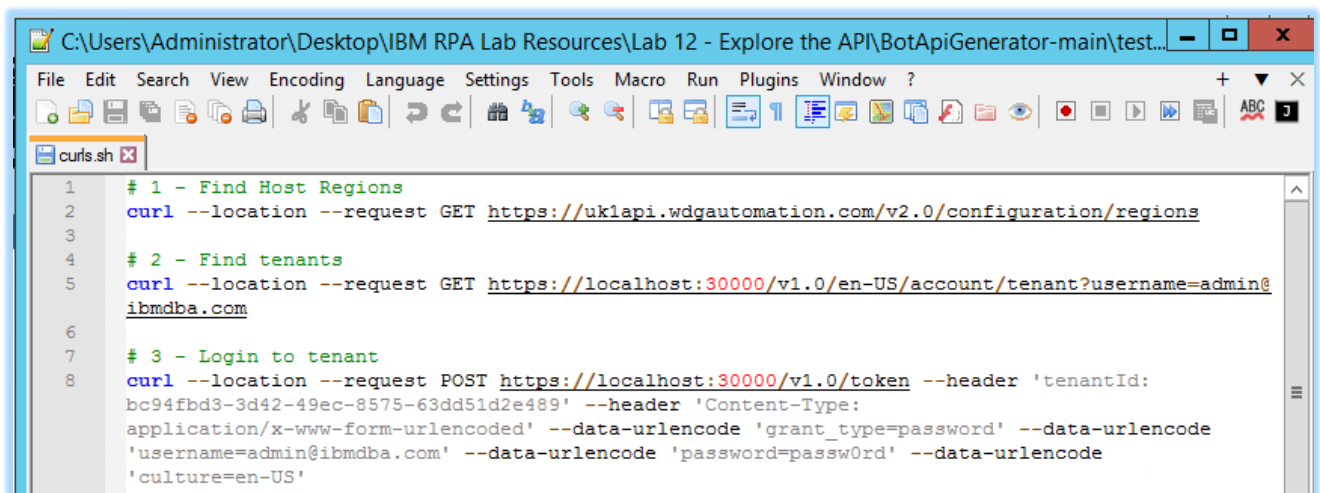
3.2.8 View the RPA API Curl

In this section you will use curl to call the RPA API. Curl is the de facto tool to test APIs.

Navigate to the directory in which you ran PowerShell. Right click *curls.sh* and edit using *Notepad++*.



You will see the curl executed in the previous step.



3.2.9 Run curl

From the desktop, open



A Unix terminal should appear.



Paste each curl command into the terminal and hit return. Examine the result of each command to verify it is working. Finally, your terminal should look like this:

```
Administrator@IBMBAW ~  
$ curl -k --location --request GET https://localhost:30000/v2.0/workspace/bc94fbd3-3d42-49ec-8575-63dd51d2e489/process/b6bfea68-7f55-43c6-821e-e5968a9f0c2f/instance/eb0a3dd3-2119-4ffe-a0c5-fbe724eae71b --header 'Authorization: Bearer 15PdWmZXG08krhJVyRmxDcW8KdH2fTHw~pu6cobTG1Tio9T2AiORIdhMAZngYL4iGDORWRhrm4RdcgeKKLDjjQke4er5Ex1d_wYPRZ~HmUqkyuhnMRpAq6VPPdkimV6dCZi6_Ix7d8bf18916XW9tmjw8NDDQ7SaxhyLtPpzAkVc3TEp9N9o9Pj55NZ05a~xgDGojaZuzWxc4Av6f8x1CLOIT12Rmxm5kV0zac9FgPCU7qzI96nLd3cC9kMxgPQ5wHM_9jkEFFITDRj8wYGVk1NFUD9aw7FJxcL9o2d4ZTw6HiawOHpVYC65TGs0EJmMIvVti6TFEyaKteDES_zKjVxuCFaz0~k5m121pCrF0pLsgtUqPD1SmrTuum_7b2gf40B71L58I4AeUdKB8CHC1iCu6x16zqp8xCwzTsxo0Xcd_XZAZqTyA1ntkKNi1WwTsw5LQjG6hounuG0wmB0WdcQnFTL37FwtLoA1IkVtsJTSKqFwn9krG7fb5JoHjYE~vSyVYUdRwXP2tpbrUJEmiCFxq_jp33PEauaynMp8xpdepbkskfhwTcv7LxDrTI2EdR20dti~gicDmZ1w~aYzJwwQLhxKYZY4nv_PtURuATVq9xBXj~L4In9qF~_augSS0DK8AJe0628bFhJ~rbNhx0GZ4Wv6MzDx5AtGsFL5_Q0801AJpE7zss7uVyhdNE2XRgAEgd~D7VWvwUAeEIA2hNN9pde2ueXD00u5BjbNp5e1wHZvn9PNPctCsMFFMgz1NIy0Y5nxnMEY1q5vx2NRbQnX_7Vvp~6W9btFJr261nEb1gQq5m6rG1C7JLpciRq9Ccu_Aj6hYGo3watgtvw8Z4YqoGCU0NgAJD35pXAM1IjkiZy1HEcW0CRID59qdUFd51UOVYX_M0oDpgsP9KU58g5tD5jak_2FIKYiIhqrFeXh5cDojUHQddq1HtikhWNpqEb3ZOG6MMWqL_8JUhr5kgd~uQPjPxqAyHXzdv1TzanjCkWKs9R3558EN6LOtyc' --data '{"status": "done", "variables": [{"value": "string", "name": "input_text"}], "outputs": {"output_text": "ggg"}}'  
Administrator@IBMBAW ~  
$
```

Congratulations, You have run a bot using curl!



3.3 Testing the API from SoapUI

SoapUI is an open-source graphical tool for testing APIs. In this section you will apply SoapUI to test the RPA API.

3.3.1 Open SoapUI

Find Soap UI and run it:



Once open, click the *Import* button.



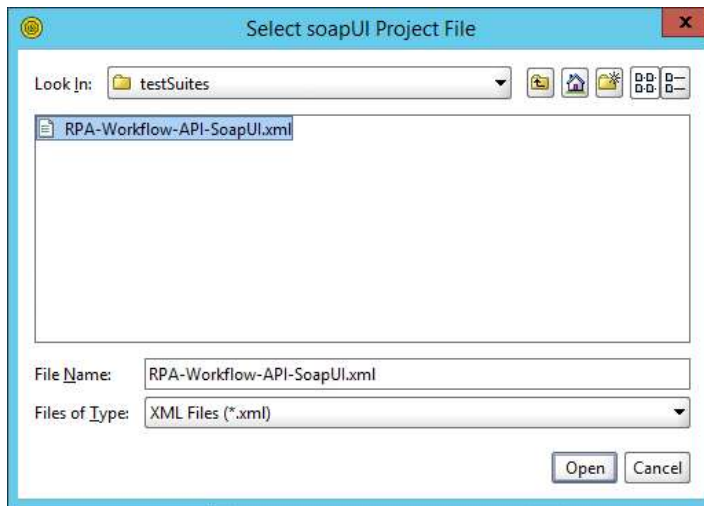


Navigate to folder:

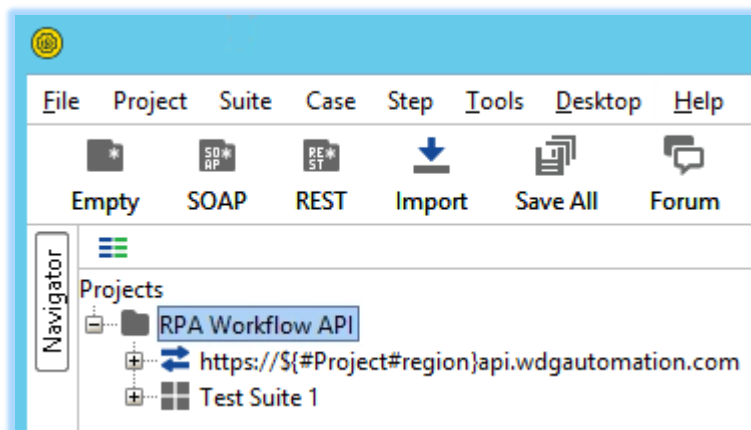
C:\Users\Administrator\Desktop\IBM RPA Lab Resources\Lab 12 -
API\BotApiGenerator-main\testSuites

Note: This location may vary depending on where you downloaded your lab in step 2.1.1

Select *RPA-Workflow-API-SoapUI.xml* and press *Open*:



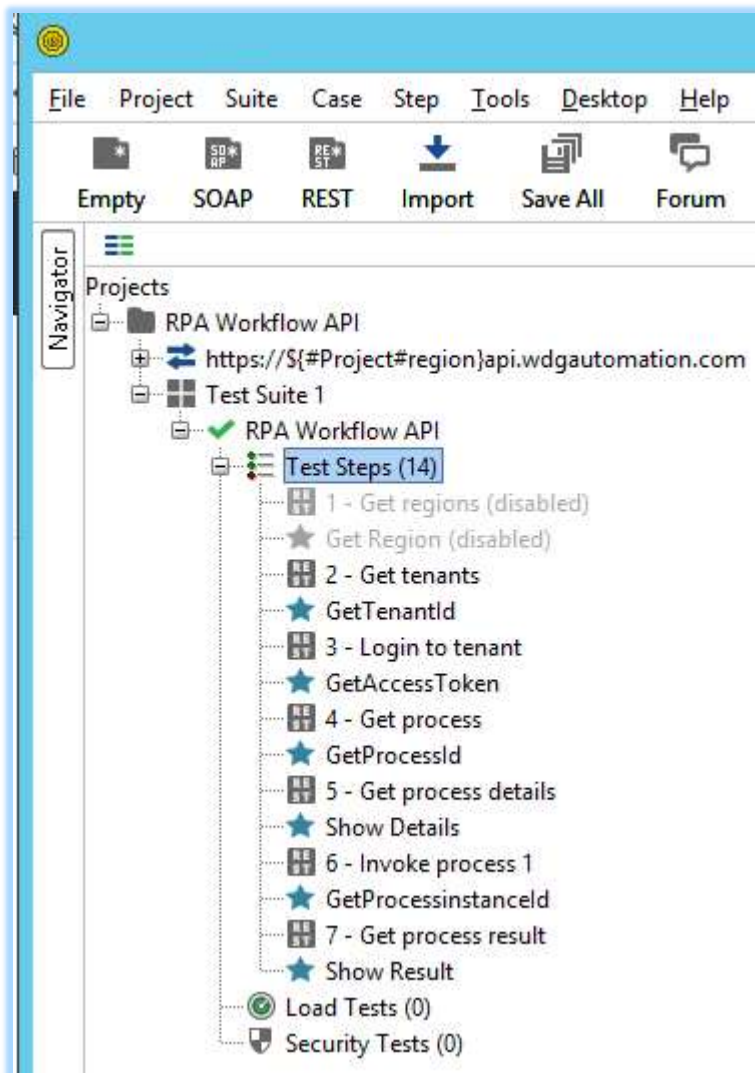
Ignore the version warning.
Verify the file is imported:



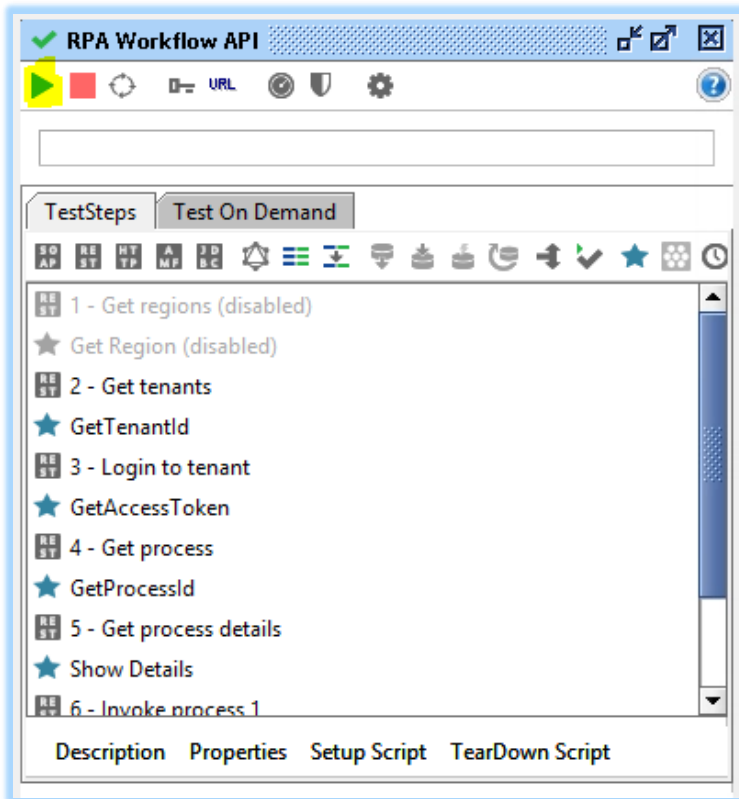


3.3.3 Run the test suite

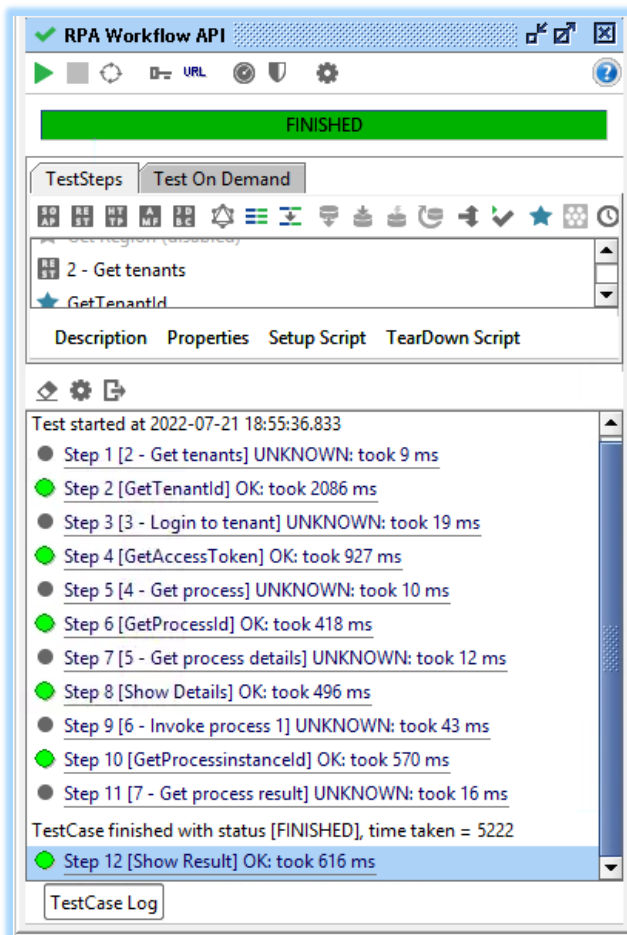
In the Projects tree, navigate to *Test Steps*. See below:



Double-click *Test steps* to open the *RPA Workflow API* test runner. Click *Run* to invoke the RPA API test suite. See below:



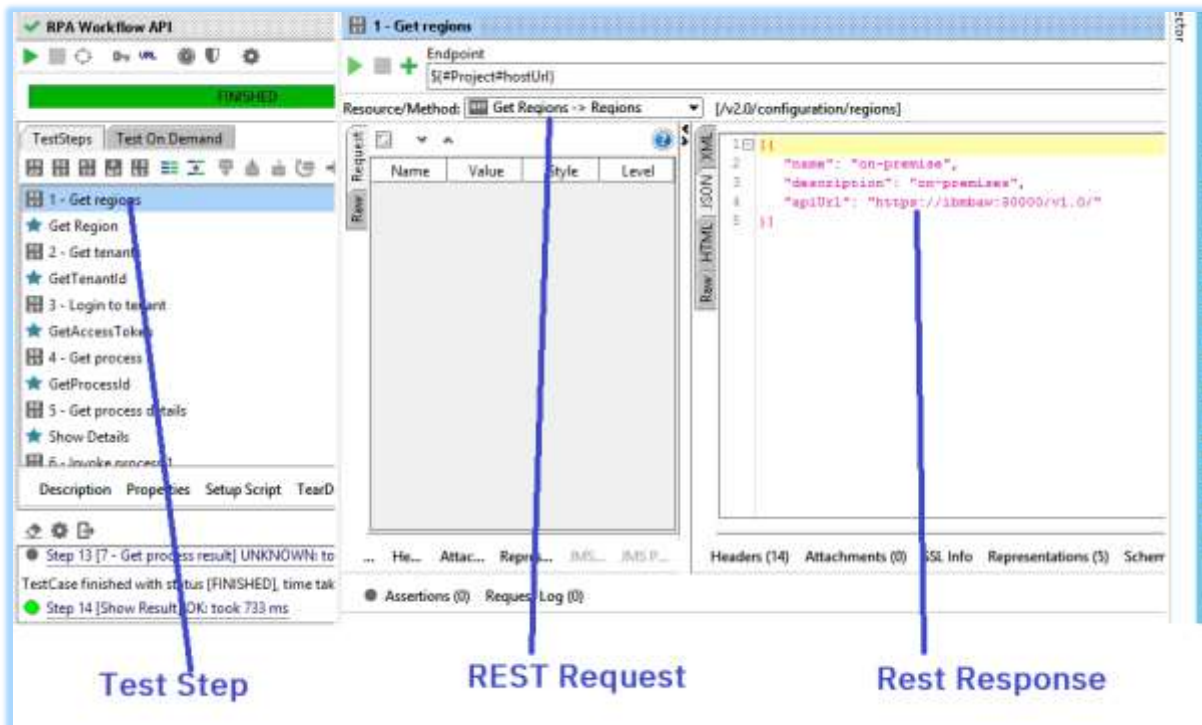
When running, you will be prompted for a series of dialog boxes showing you the API journey. When complete, you should see the following:





3.3.4 Examine the test suite flow.

The SoapUI test suite orchestrates the flow of execution by setting variables in one step and reading them in the next. You can verify this by double-clicking **step 1 – Get Regions**:



You can see the invocation request in the center panel and the invocation result on the right panel.

Tip: The result is JSON, so you need to select the JSON tab on the right panel.

Now select the next step, **Get Region**. This step is written in *Groovy Script*. It parses the JSON response from previous step and sets a project variable to be used in the next step. See below:



The screenshot displays the SoapUI interface. On the left, the 'Navigator' pane shows a project tree with 'RPA Workflow API' and 'Test Suite 1'. Under 'Test Steps (14)', the 'Get Region' step is selected. The main editor shows the Groovy code for this step. Two yellow boxes highlight specific parts of the code: one for parsing the JSON response into variables and another for setting a project property. Blue arrows point from text labels at the bottom to these highlighted sections.

```
1 import com.eviware.soapui.model.testsuite.TestCaseRunContext
2 import net.sf.json.groovy.JsonSlurper
3
4 //Check if the response
5 assert context.response, 'Response is empty or null'
6
7 log.info("context.response: " + context.response)
8
9 def jsonResonponse = new JsonSlurper().parseText(context.response)
10
11 region = 0
12 regionName = jsonResonponse[region].name
13 regionUrl = jsonResonponse[region].apiUrl
14
15 if (regionUrl) {
16     regionUrl = regionUrl.replace("/v1.0/", "")
17     log.info("Found Region: " + regionName)
18     log.info("Found region Url: " + regionUrl)
19     testRunner.testCase.testSuite.project.setPropertyValue( "hostUrl", regionUrl )
20 }
21
22 def ui = com.eviware.soapui.support.UISupport
23 ui.showInfoMessage( "selected region is: " + regionUrl )
```

Parse Json Response into variables

Set project variable

Now follow the remainder of the suite to see the full API journey.



3.4 Generating an Open API Specification

Up until now you used the raw REST RPA API. But there is a better way! Modern applications allow interaction with APIs through the *OpenAPI* specification. This specification defines APIs in a human readable way which is also readable by computers.

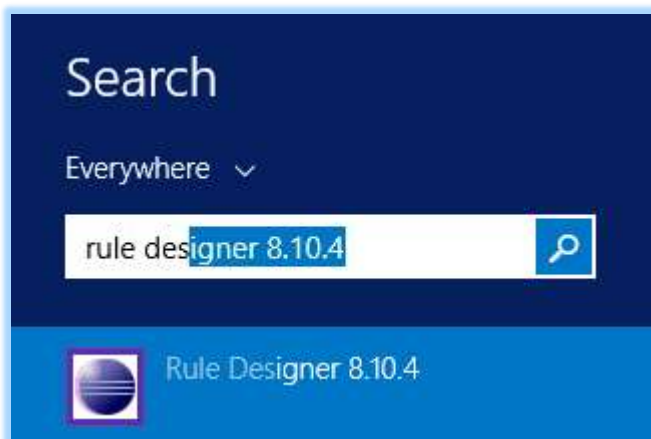
The advantages of using OpenAPI are:

- It's a universal API definition language understood by most modern applications
- It can be easily viewed by humans in tools such as <https://editor.swagger.io/>
- The definition language is rigorous, so avoids the creation of sloppy APIs.

Let's go ahead and generate an OpenAPI spec for the bot you created.

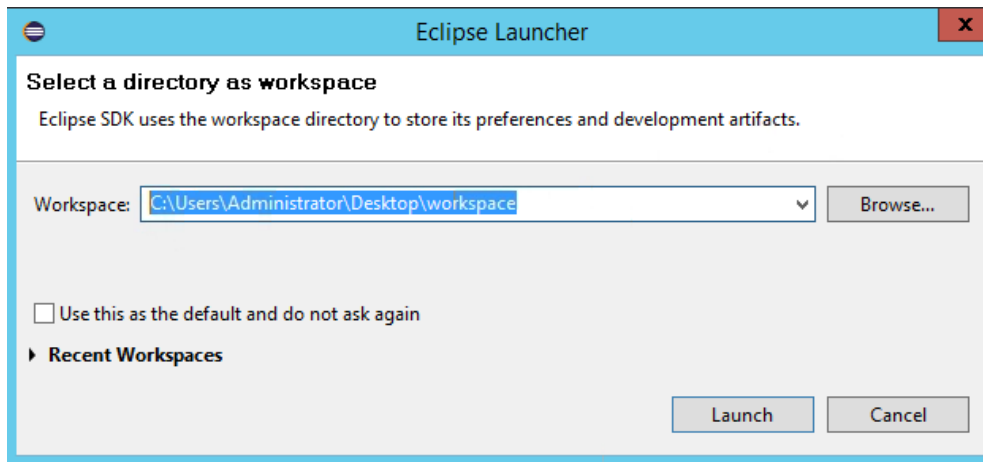
3.4.1 Start Eclipse

If you are running in the Skyap image, you can access Eclipse by running Rule Designer. Search for *Rule Designer* in the Windows search and then double click to start.



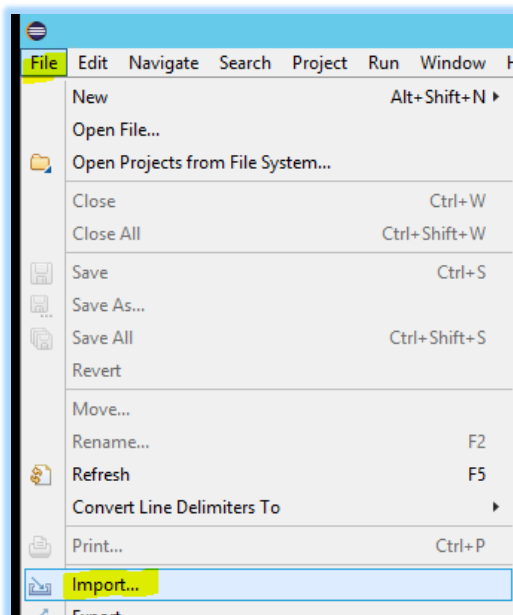
Create a workspace under

C:\Users\Administrator\Desktop\workspace

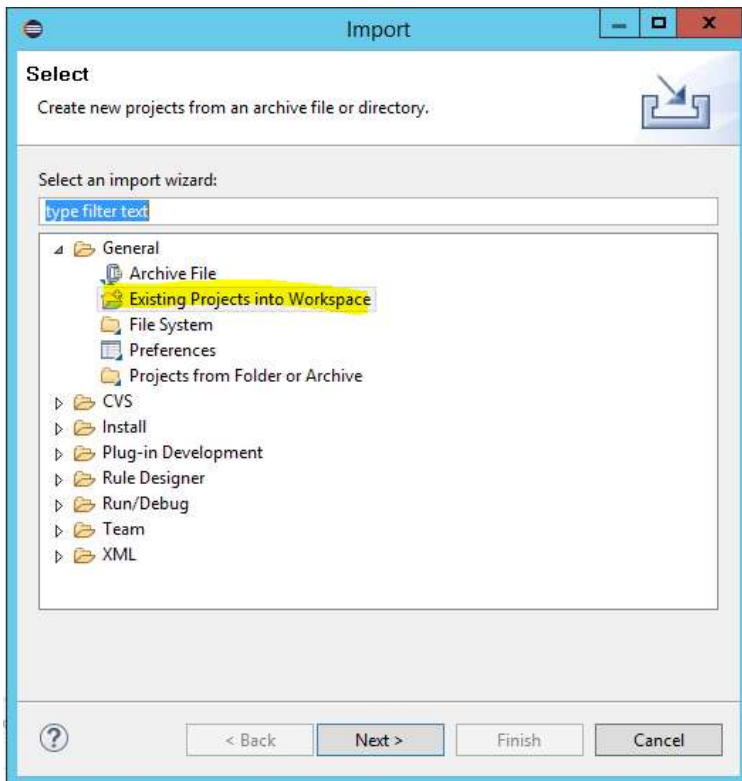


Press *Launch*.

Once Eclipse is open, select *File->Import*:



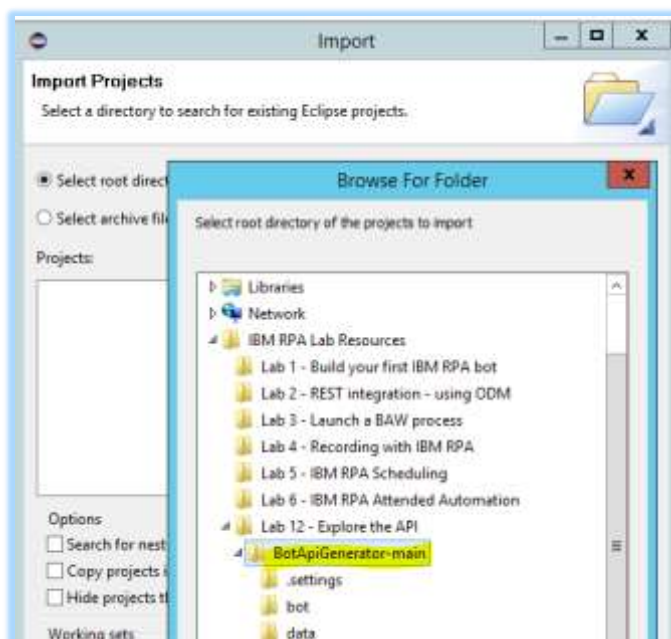
Select *General->Existing Projects into Workspace*:



Press *Next*. Select the folder

C:\Users\Administrator\Desktop\IBM RPA Lab Resources\Lab 12 - API\BotApiGenerator-main

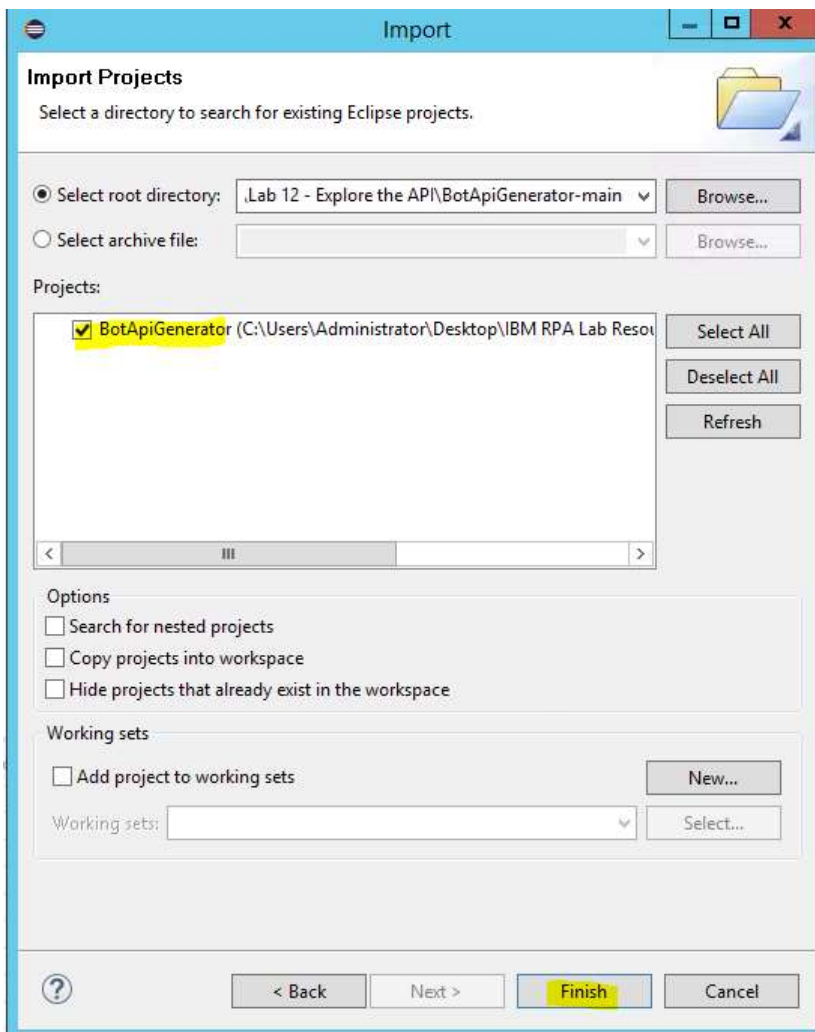
Note: This location may vary depending on where you downloaded your lab in step 2.1.1



Press *OK*.

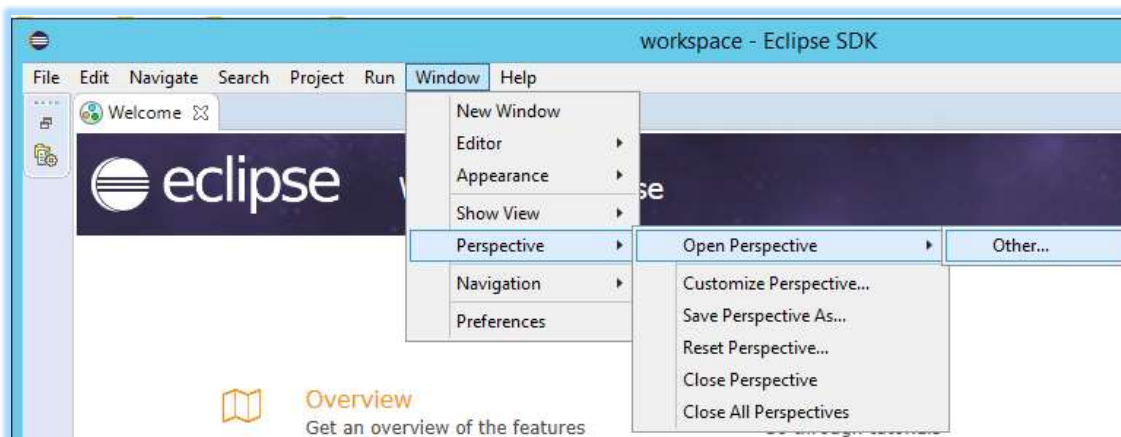


Select *BotApiGenerator* and then press Finish:



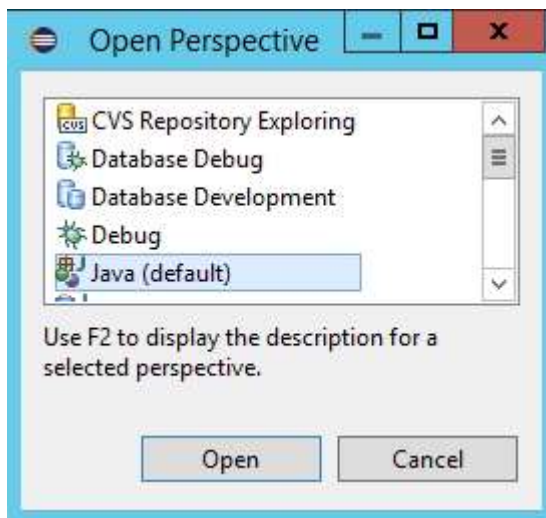
Set the perspective to Java

Navigate to Window->Perspective->Open Perspective->Other



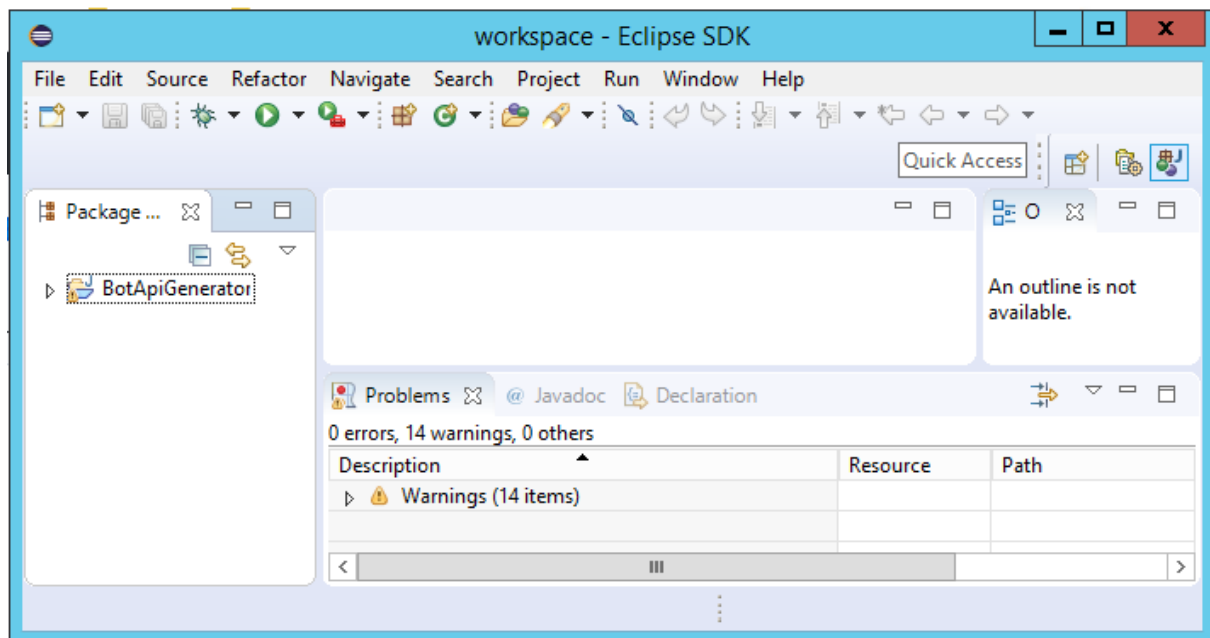


Then choose *Java (default)*:

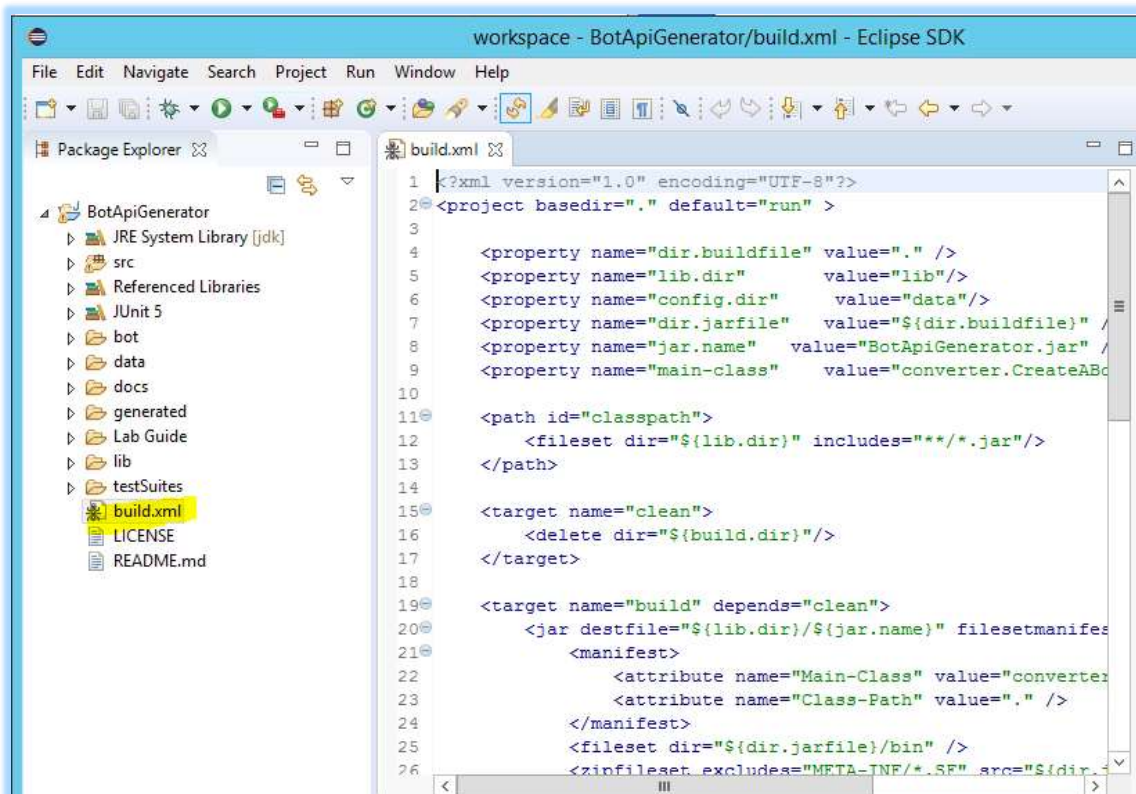


Press *Open*.

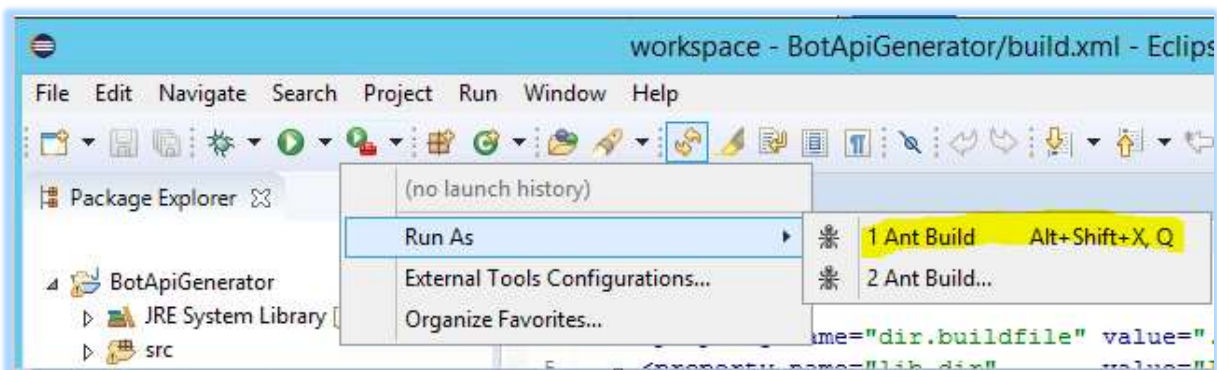
Close the *Welcome* tab. You should see this:



Expand the Package hierarchy on the left until you find *build.xml*. Double-click to open. You should see this:



Execute the *build.xml* by selecting the *External Tools* icon , then select *Run As->Ant Build*.



This runs the API generator. You should see the following output in the Console:



```

Problems @ Javadoc Declaration Console
<terminated> BotApiGenerator build.xml [Ant Build] C:\IBM\ODM8103\jdk\bin\javaw.exe (Jul 26, 2022, 3:15:57 PM)

[java] apiHost:
[java] apiPayload:
[java] resUser:
[java] resPwd:
[java] RpaConfig [id=null, startId=null, name=BAWIntegrationTest, system=null,
[java] doRest: result is {"access_token":"GTQLjV7LRu9jT1lE3eMWYJhjoOROBq0g2OSK
[java] >> doRest: command=GET, urlString=https://ibmbaw:30000/v2.0/workspace/b
[java] doRest: result is {"results":[{"id":"b6bfea68-7f55-43c6-821e-e5968a9f0c
[java] >> doRest: command=GET, urlString=https://ibmbaw:30000/v2.0/workspace/b
[java] doRest: result is {"id":"b6bfea68-7f55-43c6-821e-e5968a9f0c2f","name":"
[java] BotInfo: BotInfo [workspaceId=bc94fbd3-3d42-49ec-8575-63dd51d2e489, pro
[java] **GENERATING CODE FOR BAWIntegrationTest
[java] Internal API generated in .\generated\Bawintegrationtest_internal.yaml
[java] **GENERATING CODE FOR BAWIntegrationTest
[java] External API generated in .\generated\Bawintegrationtest_external.yaml
[java] **GENERATING CODE FOR BAWIntegrationTest
[java] JUnit test generated in .\generated\BawintegrationtestTest.java
BUILD SUCCESSFUL
Total time: 1 second

```

Tip: If you see an error, it could be that the name of your process does not match that defined in the *data/config.json*. Edit this file to match your system.

Refresh the *BotApiGenerator* folder by hitting F5.
Open *generated/Bawintegrationtest_external.yaml*:

```

Package Explorer
BotApiGenerator
├── JRE System Library [jdk]
├── src
├── Referenced Libraries
├── JUnit 5
├── bot
├── data
├── docs
├── generated
│   ├── Bawintegrationtest_external.yaml
│   ├── Bawintegrationtest_internal.yaml
│   └── BawintegrationtestTest.java
├── Lab Guide
├── lib
├── testSuites
├── build.xml
├── LICENSE
└── README.md

build.xml
Bawintegrationtest_external.yaml
1 openapi: 3.0.1
2 info:
3   title: BAWIntegrationTest RPA API
4   description: |-
5     This API presents an authenticated interface
6     The **IBM RPA API V2.0 API** provides s
7
8   * **The Authentication APIs** provide a JSC
9
10  * **The Tenant management APIs** provide s
11
12  * **The Process management APIs** provide
13
14  * **The Workspace APIs** provide endpoints
15  contact:
16    email: ncrowther@uk.ibm.com
17  version: 1.0.0
18 externalDocs:
19  description: Find out more about Swagger

```

The contents of this file contain the generated OpenAPI specification for your bot. Copy the contents of the entire file into the clipboard.

Open URL

<https://editor.swagger.io/>



Paste into the left panel. You should see the Open API specification for your bot:

The screenshot shows the Swagger Editor interface. The left panel contains the OpenAPI specification for the BAWIntegrationTest RPA API. The right panel displays a graphical overview of the API, including the Application Layer, Invocation Layer, and Execution Layer. The Application Layer includes components like IBM App Connect, IBM BPM, and IBM Decision Services. The Invocation Layer shows the BAW RPA Tenant. The Execution Layer includes the BAW RPA Agent and Bot scripts. The diagram illustrates the flow from the Application Layer through the Invocation Layer to the Execution Layer.

Take a moment to explore the API.

3.4.2 Run curl from OpenApi

You can use the *Swagger Editor* to create curl. Navigate to *Login to RPA tenant* in the Swagger Editor:

The screenshot shows the 'Login to RPA tenant' endpoint in the Swagger Editor. The endpoint is a POST request to `/v1.0/token`. The description states: 'Asynchronously runs a process on a RPA tenant specified in the URL. All requests are authenticated using a bearer token'. The 'Parameters' section is visible at the bottom, and a 'Try it out' button is highlighted.

Expand the command. Press “*try it out*”. Then press *Execute*. See below:



☐ Send empty value

culture
string

The code of the language. See Supported languages for the supported language codes

en-US

☐ Send empty value

Execute Clear

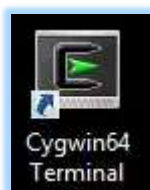
Responses

Curl

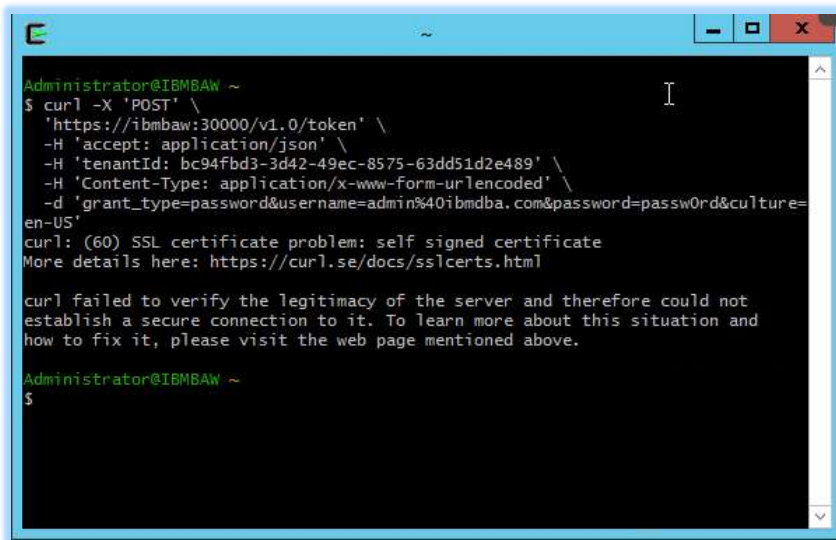
```
curl -X 'POST' \
  'https://ibmbaw:30000/v1.0/token' \
  -H 'accept: application/json' \
  -H 'tenantId: bc94fbd3-3d42-49ec-8575-63dd51d2e489' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d 'grant_type=password&username=admin%40ibmdba.com&password=passw0rd&culture=en-US'
```

Copy the generated curl using the clipboard icon.

From the desktop, open a Unix terminal:



Paste the curl command into the terminal and hit return. If you are using Skytap, then you will see the following error:



```
Administrator@IBMBAW ~  
$ curl -X 'POST' \\  
  'https://ibmbaw:30000/v1.0/token' \\  
  -H 'accept: application/json' \\  
  -H 'tenantId: bc94fbd3-3d42-49ec-8575-63dd51d2e489' \\  
  -H 'Content-Type: application/x-www-form-urlencoded' \\  
  -d 'grant_type=password&username=admin%40ibmdba.com&password=passw0rd&culture=en-US'  
curl: (60) SSL certificate problem: self signed certificate  
More details here: https://curl.se/docs/sslcerts.html  
  
curl failed to verify the legitimacy of the server and therefore could not  
establish a secure connection to it. To learn more about this situation and  
how to fix it, please visit the web page mentioned above.  
  
Administrator@IBMBAW ~  
$
```

La

This is not a problem! In Skytap you can ignore certificate errors by adding the **-k** flag to the beginning of the curl command as shown below:

```
curl -k -X 'POST'    'https://ibmbaw:30000/v1.0/token'    -H 'accept:  
application/json'    -H 'tenantId: bc94fbd3-3d42-49ec-8575-63dd51d2e489'  
-H 'Content-Type: application/x-www-form-urlencoded'    -d  
'grant_type=password&username=admin%40ibmdba.com&password=passw0rd&cultur  
e=en-US'
```

Now the curl should run successfully!

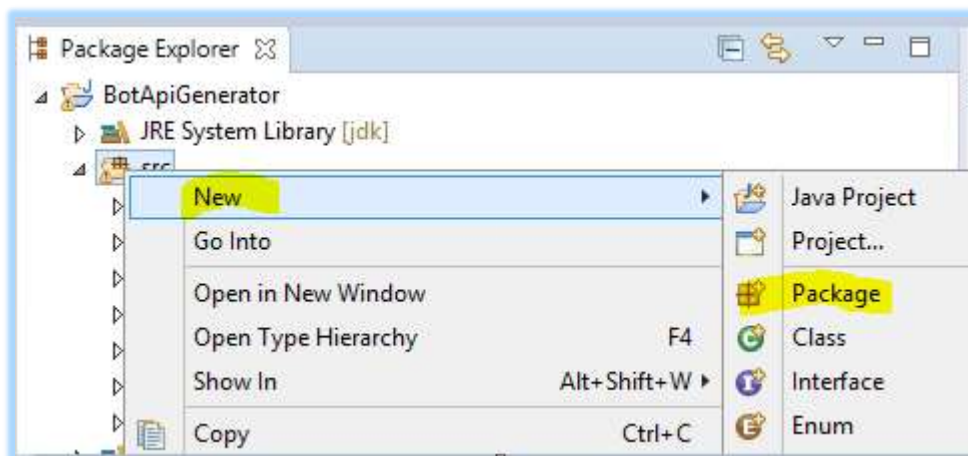


3.4.3 Test Api with JUnit

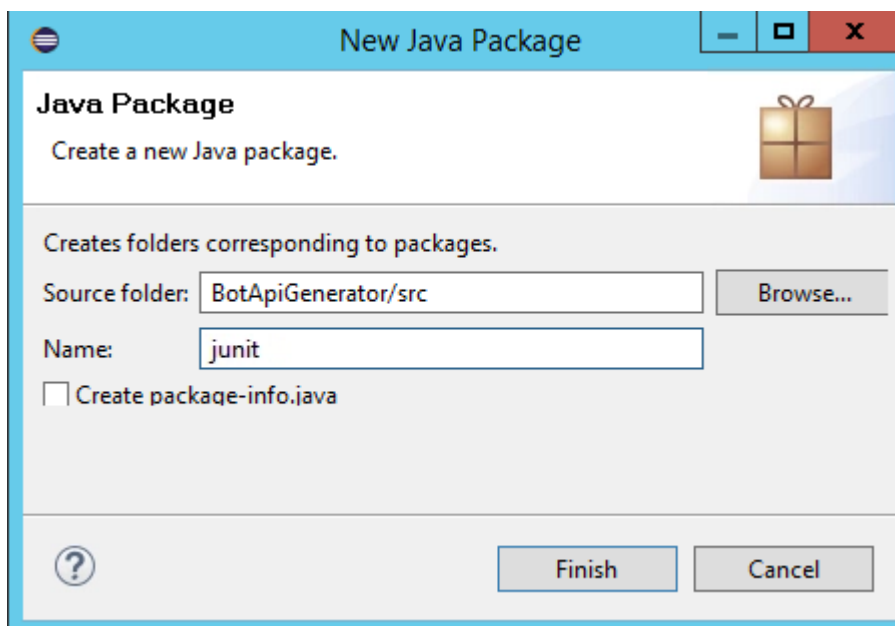
In this step you will test the RPA API using Junit. Junit is the original test framework and is still one of the world's most popular.

Revisit the Eclipse IDE you ran in 3.4.1.

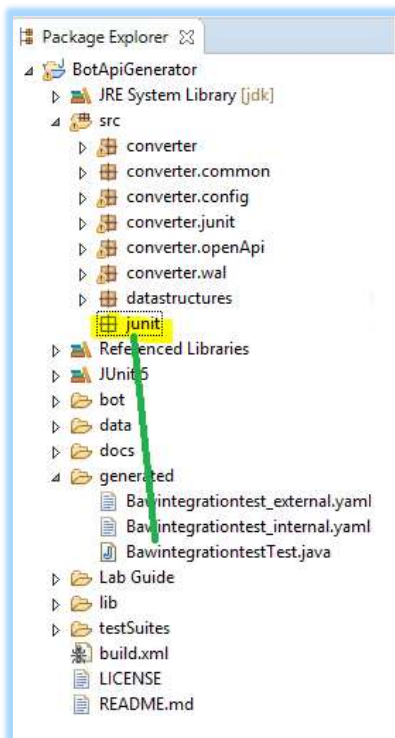
Create a new package under the *src* folder:



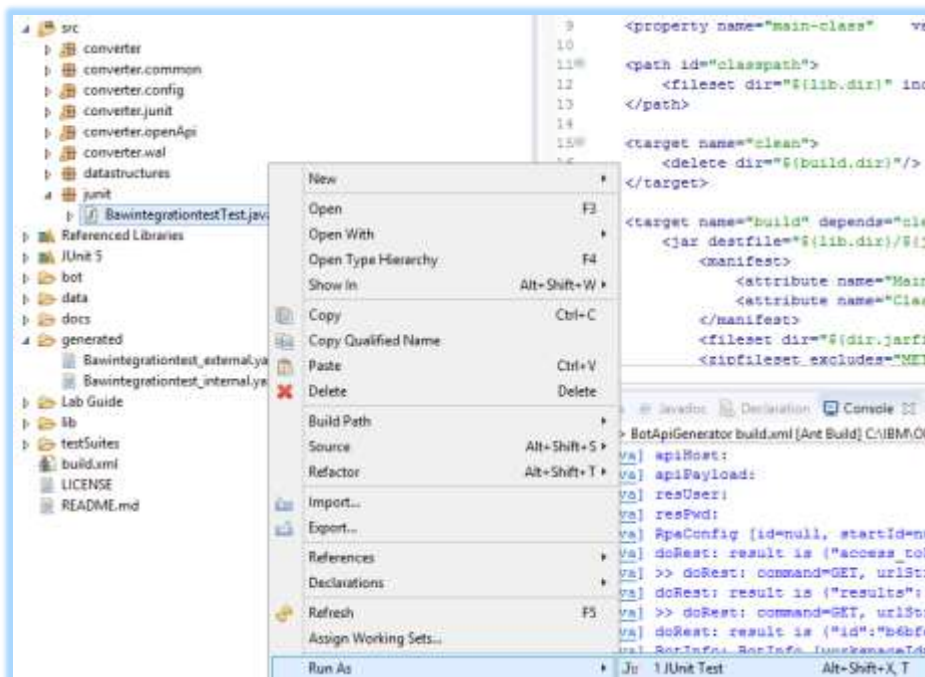
Set the name of the folder to *junit* and press Finish:



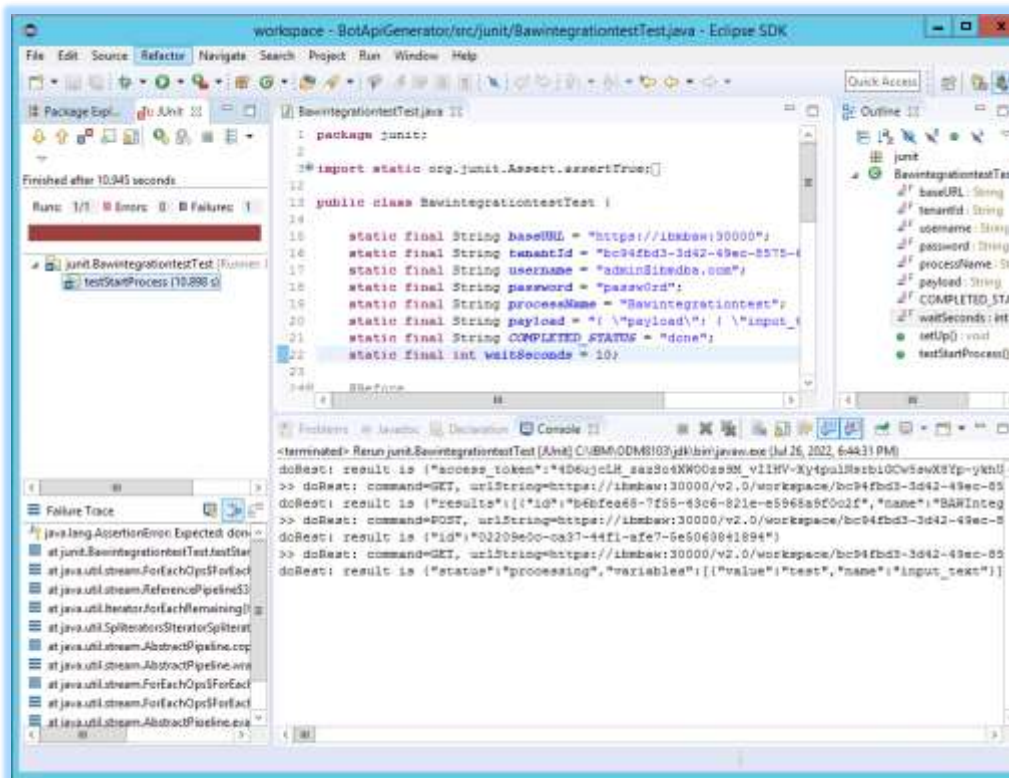
In the Package Explorer drag and drop *generated/BawintegrationTest.java* into the *junit* folder, as shown below:



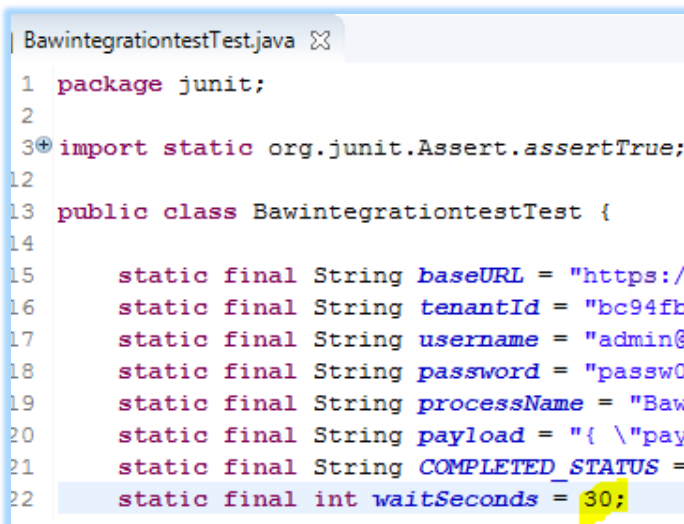
Now right-click *BawintegrationTest.java* and run as a Junit test:



JUnit will test the bot using the RPA API. You will see two dialog boxes appear. **Do not do anything**, just wait. After 20 seconds you will see a fail:



This is expected. The test failed because the wait period was exceeded. Edit line 22 in *BawintegrationTest.java*, and increase the value to 30.



Re-run the test. This time, enter the values requested by the bot within 30 seconds and it should pass:



```
workspace - BotApiGenerator/src/junit/BawintegrationtestTest.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
JUnit
Package Explorer
JUnit
Finished after 30.922 seconds
Runs: 1/1 Errors: 0 Failures: 0
junit.BawintegrationtestTest [Runner:]
BawintegrationtestTest.java
1 package junit;
2
3 import static org.junit.Assert.assertTrue;
4
5 public class BawintegrationtestTest {
6
7     static final String baseUrl = "https://ibmbaw:30000";
8     static final String tenantId = "bc94fbd3-3d42-49ec-8575-6";
9     static final String username = "admin@ibmdba.com";
10    static final String password = "password";
11    static final String processName = "Bawintegrationtest";
12    static final String payload = "{ \"payload\": { \"input_text\"";
13    static final String COMPLETED_STATUS = "done";
14    static final int waitSeconds = 30;
15
16    @Before
17    public void setUp() {
18        // ...
19    }
20
21    @Test
22    public void testStartProcess() {
23        // ...
24    }
25
26    @After
27    public void tearDown() {
28        // ...
29    }
30
31    }
32
33 }
Outline
junit
BawintegrationtestTest
    baseUrl: String
    tenantId: String
    username: String
    password: String
    processName: String
    payload: String
    COMPLETED_STATUS: String
    waitSeconds: int
    setUp(): void
    testStartProcess(): void
Problems
Declaration
Console
<terminated> BawintegrationtestTest [JUnit] C:\IBM\ODMS103\jdk\bin\java.exe (Jul 26, 2022, 6:37:40 PM)
doRest: result is {"access token":"3dsPRMqmbowr-zQvDfAOngh8FzEaYw7Tl3RuU16Roup4iVnoma76Wp
>> doRest: command=GET, urlString=https://ibmbaw:30000/v2.0/workspace/bc94fbd3-3d42-49ec-85
doRest: result is {"results":[{"id":"b6bfae68-7f55-43c6-821e-e5968a9f0c2f","name":"BawInteg
>> doRest: command=POST, urlString=https://ibmbaw:30000/v2.0/workspace/bc94fbd3-3d42-49ec-8
doRest: result is {"id":"c9000ea4-29f9-43ef-a610-09e2226a057"}
>> doRest: command=GET, urlString=https://ibmbaw:30000/v2.0/workspace/bc94fbd3-3d42-49ec-85
doRest: result is {"status":"done","variables":[{"value":"test","name":"input_text"},"outp
Its working!
```

Can you see any problems testing this bot? It relies on human input which fails if the human is too slow. JUnit is not ideal for this scenario. It should be used for testing bots that do not require user input.

Nicely done! You have tested the RPA API with JUnit.

3.5 Integrate to the World!

In this lab you should now be familiar with the RPA Asynchronous API. You called it using PowerShell, Curl and SoapUI. You then generated an OpenApi specification to make your API ready to plug into modern applications. Finally, you tested the API using JUnit.

Now that you have completed this lab, you can start creating your own bot integration. The generator provided in the lab should help you create APIs for any bot. Give it a try and integrate your bots to the rest of the world!

If you have feedback, please contact:

ncrowther@uk.ibm.com



This concludes the lab.