



Build your first IBM RPA bot

Hands-on Lab

Version 1.3 (for on-prem deployment)

For BP Virtual Workshop

Authors:

Jukka Juselius - jukka.juselius@fi.ibm.com

Paul Pacholski – pacholsk@ca.ibm.com

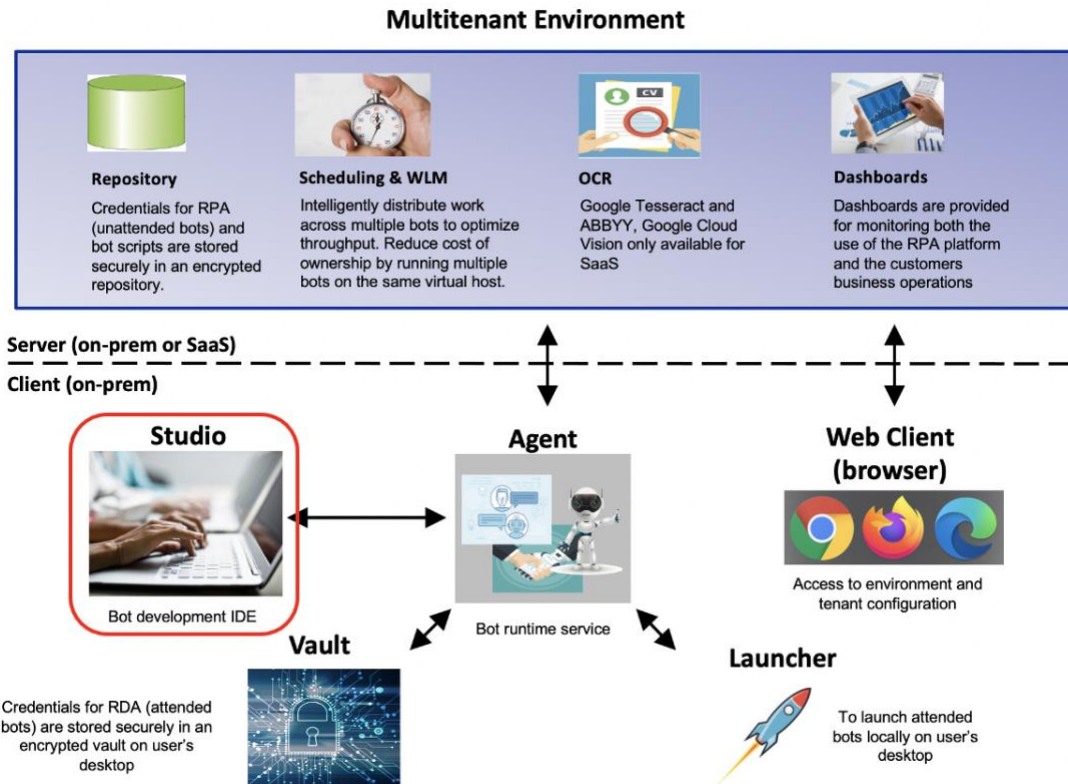
Changes for BP Virtual Workshop by Jukka Juselius

Table of Contents

1. Introduction	4
Use case	4
Prerequisites	4
2. Lab Instructions.....	5
Start IBM RPA Studio	5
Create a new automation	5
Create your bot script - Login to JK Automation website.....	6
Test your automation and continue implementation	13
Read CSV	14
Store row data in variables	16
Set Sales Leads input fields	19
Define Subroutines	23
Add loop to iterate through the data table	25
Logout from JK automation website	28
3. Work with Completed Automation	31
Open completed lab	31
Check input csv file location	31
Run the automation.....	31

1. Introduction

This hands-on lab you will learn how to use IBM RPA Studio to create a simple RPA automation.



Use case

In this exercise you will build a robot to automate processing of sales leads that arrive in a CSV/Excel format. Each row of the file represents a separate sales lead. The sales leads need to be manually entered (copy/paste) by an analyst into the online opportunity system of record (JK Automation Sales Leads). This task is error prone and the analysts repeatedly ask if this can be automated, but it has never been a company priority. Until now 😊

Prerequisites

Nothing. Your environment is already configured, and everything is set. Ready to start?

2. Lab Instructions

Start IBM RPA Studio

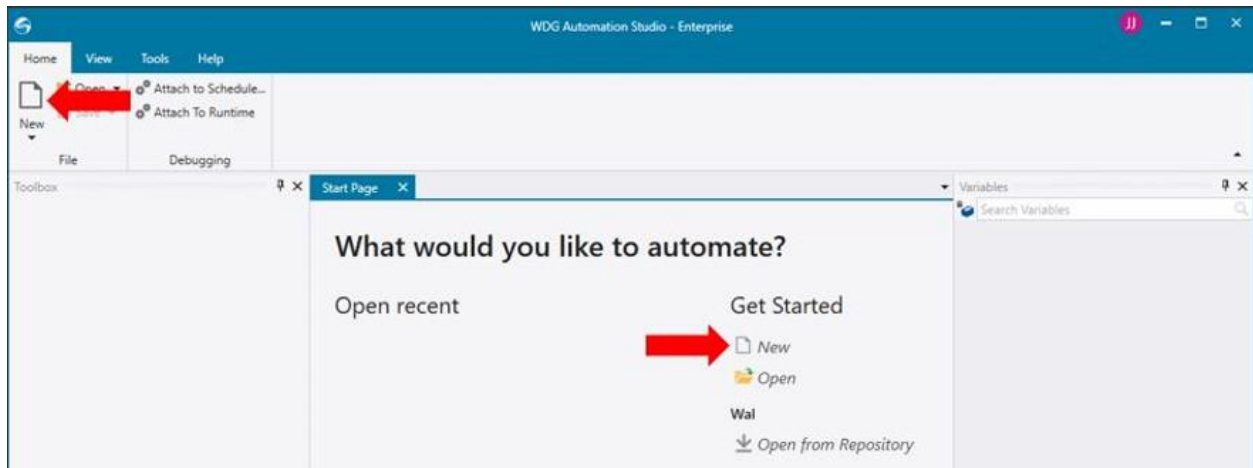
_1. Double-click the IBM RPA Studio shortcut in your desktop.



_2. Login with username **admin@ibm.com** and with password **password**.

Create a new automation

_1. Create a new WAL file by clicking the **New** icon from the top toolbar or **New** under "Get Started" section.

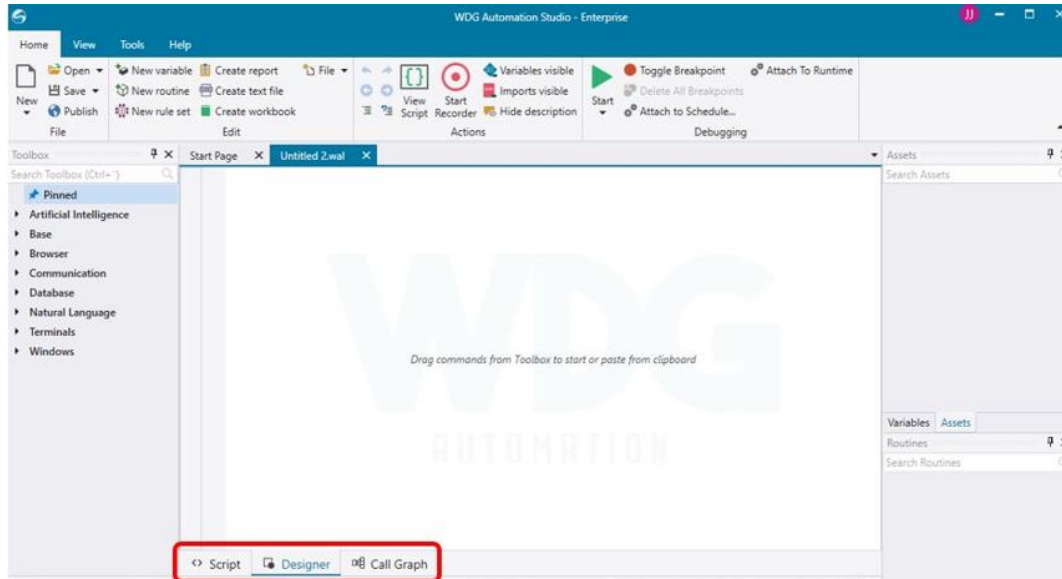


_2. Select WAL file and click Open.

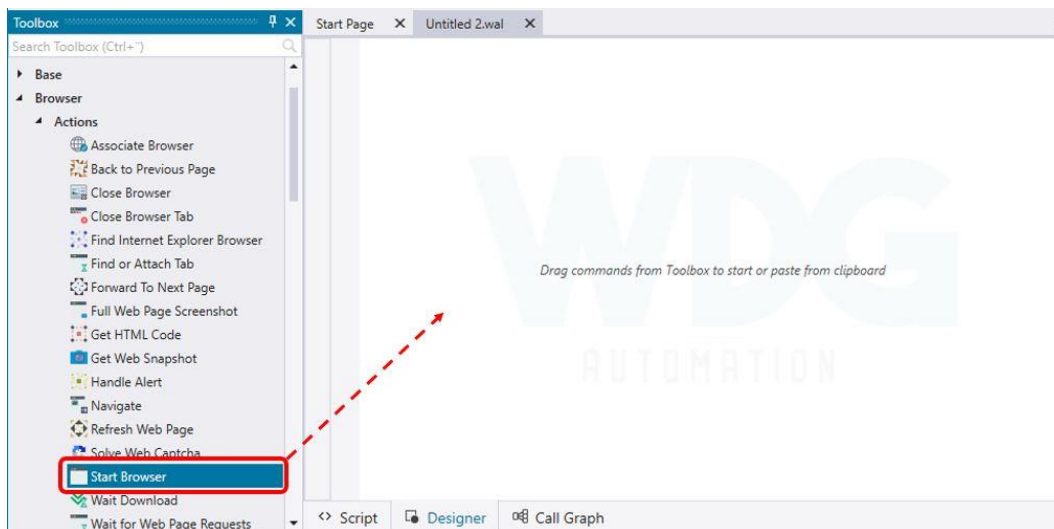


Create your bot script - Login to JK Automation website

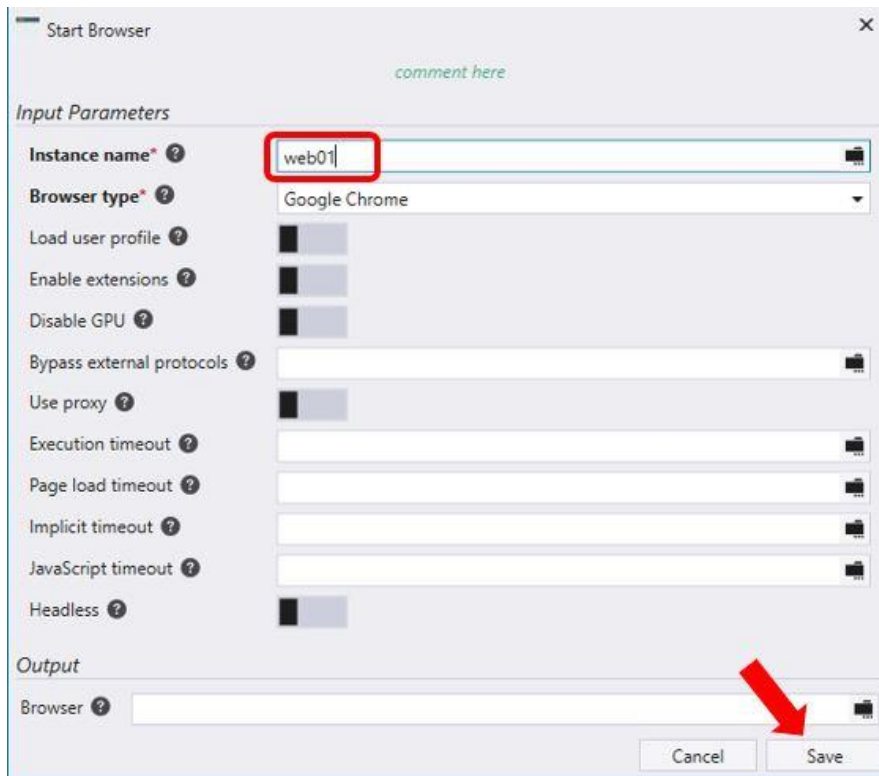
_1. You should now have your Studio opened with an empty WAL file in your Designer view. Note that you can change your view between *Script*, *Designer* and *Call Graph*.



_2. Add Start Browser command from your Toolbox to your Designer view. You can find the command under **Browser** > **Actions**. Drag and drop it to your canvas.



_3. The command configuration opens. You need to set name to your browser instance (for example `web01`). Keep `Google Chrome` selected as your browser type, if you have Chrome installed. You can also change the browser to one you want to use. Click `Save` button.



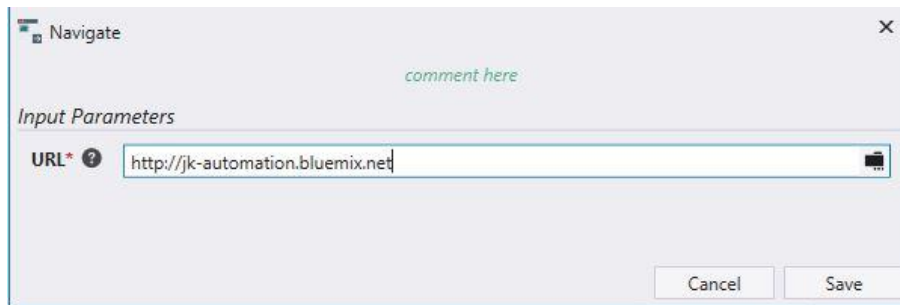
The screenshot shows the 'Start Browser' configuration window. It has a title bar with a close button. Below the title bar is a green text prompt 'comment here'. The 'Input Parameters' section contains several fields: 'Instance name*' (text input with 'web01' and a red box around it), 'Browser type*' (dropdown menu showing 'Google Chrome'), 'Load user profile' (checkbox), 'Enable extensions' (checkbox), 'Disable GPU' (checkbox), 'Bypass external protocols' (text input), 'Use proxy' (checkbox), 'Execution timeout' (text input), 'Page load timeout' (text input), 'Implicit timeout' (text input), 'JavaScript timeout' (text input), and 'Headless' (checkbox). The 'Output' section has a 'Browser' field. At the bottom are 'Cancel' and 'Save' buttons. A red arrow points to the 'Save' button.

_4. Close Browser configuration window pops up automatically. Make sure to set the same instance name value that you used for `Start Browser`. Also **set `Keep browser open` as `enabled`**. This helps you during the implementation so that browser is not closed when you test your automation and you can continue using it for defining next actions for your automation. Click `Save` button to close the configuration window.

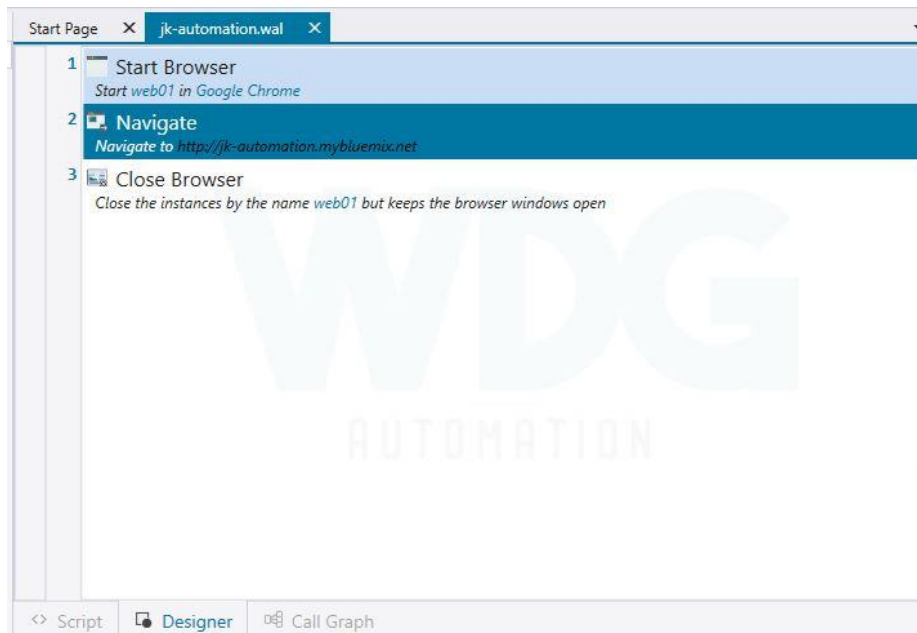


The screenshot shows the 'Close Browser' configuration window. It has a title bar with a close button. Below the title bar is a green text prompt 'comment here'. The 'Input Parameters' section contains two fields: 'Instance name*' (text input with 'web01') and 'Keep browser open' (checkbox, which is checked). At the bottom are 'Cancel' and 'Save' buttons. The 'Save' button is highlighted with a light blue background.

_5. Next command we want is *Navigate*. You can find it also under *Browser > Actions*. Drag and drop it below the *Start Browser* command to your Designer view. Set the URL to <http://jk-automation.mybluemix.net> and click the *Save* button.



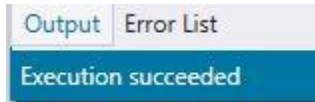
_6. **Save** your work by hitting *Ctrl+S* or the *Save* icon from the top toolbar. Navigate to *Desktop → IBM RPA Lab Resources → Lab1 – Build your first IBM RPA bot* and save your automation WAL file using name **jk-automation**. Your automation should now look as follows.



_7. **Run** your automation to test that it works and JK Automation website is opened. Click the green *Run* icon form the top toolbar. Alternatively, you can hit *F5*.



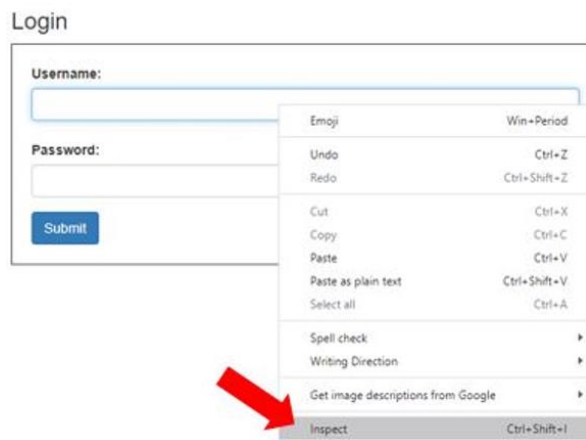
8. You should see new Chrome Browser window opened and JK Automation login page opened. **Note!** For the first time this might take a couple of seconds. When you go back to your Studio window, you should see `Execution succeeded` text at the bottom left-hand side corner. ***Make sure to leave the browser window open!***



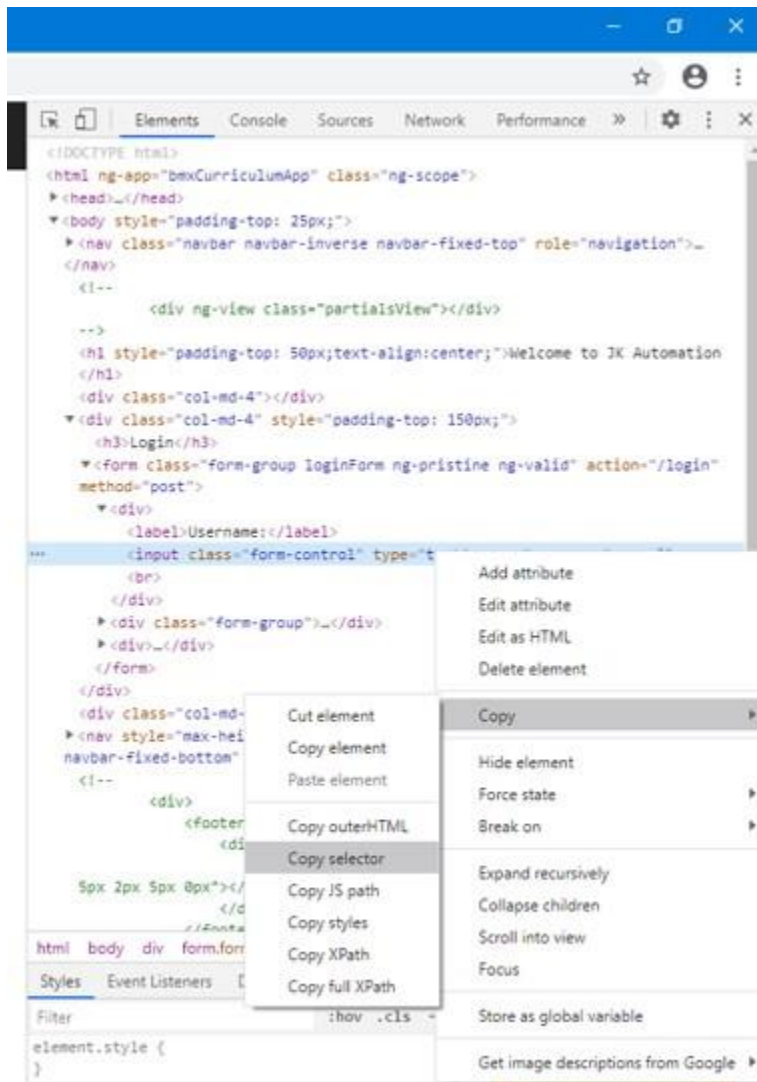
9. Nice! Your first run with IBM RPA 👍 Let's keep on going! Next we will need to automate the login to JK Automation website.

IBM RPA has a web recorder plugin for Chrome, but for this lab we're doing things manually to learn how we can select web elements just using basic tools. IBM RPA supports several selectors for fields: id, name, css, xpath, id + name.

10. Open JK Automation browser window that you should have open. **Right-click** the Username field and from the opened menu select `Inspect`.



11. This open your browsers element inspector with the Username field element selected.
Right-click the selected element, select Copy > Copy selector. This will copy the CSS Selector for the element to your clipboard.



_12. Go back to your IBM RPA Studio and add Set Value to Field (Browser > Fields > Set Value to Field) command to your automation just under Navigation command.

_13. When the command configuration window opens set *Value* to "**whatever**" (username you use does not matter), *Selector Type* to **Css** and *Css* to **the value from your clipboard** (the selector you copied from element inspector). Finally click *Save*.

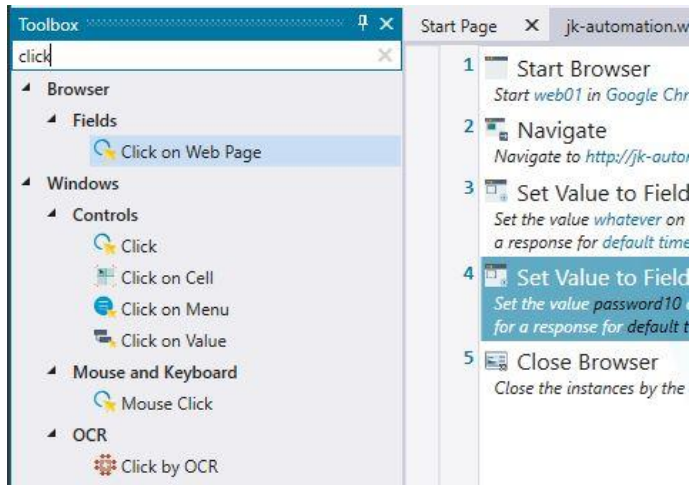


_14. Similarly add new Set Value to Field command below the previous for the Password field (you need to copy the selector for it as well). Set *Value* to **password10**.

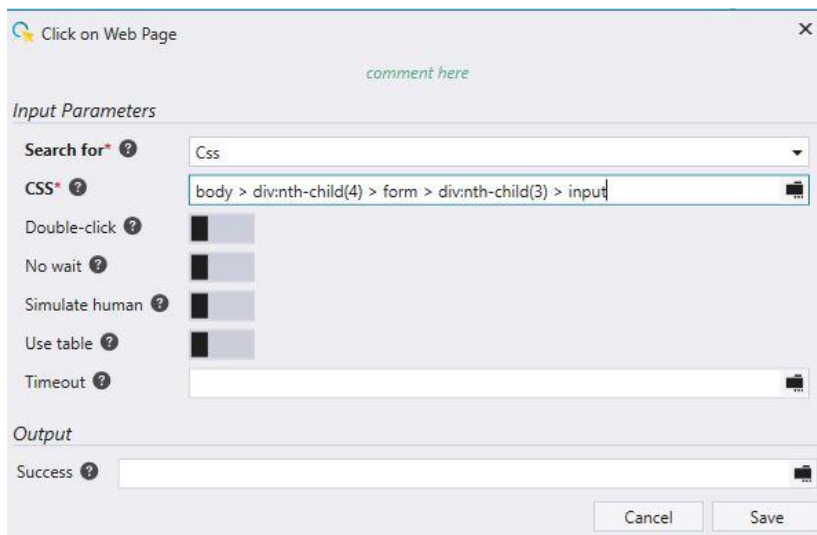


Now, let's finish the login sequence by adding a command to click the *Submit* button in the JK Automation login page.

_15. We'll do that by adding command `Click on Web Page` after the `Set Value to Field` commands. **Note** that you can use toolbox search field to find commands. To display all the commands related to clicking, type **click** to the search field.



_16. Get the selector for the Submit button from your browsers element inspector as you earlier did for the Username and Password fields.



Your automation should look like as follows. Nice, good job!

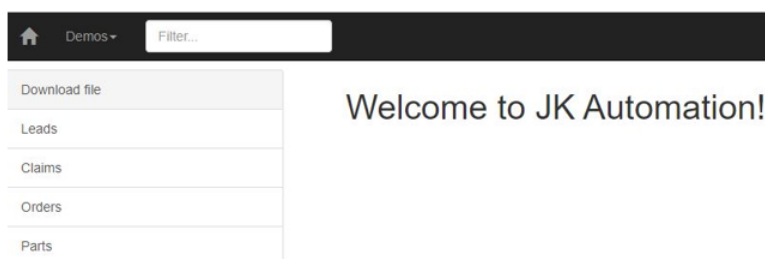


Test your automation and continue implementation

- _1. Save your work.
- _2. Close your browser with JK Automation login page
- _3. **Run** your automation to test that it works and JK Automation website is opened. Click the green Run icon from the top toolbar. Alternatively, you can hit F5.

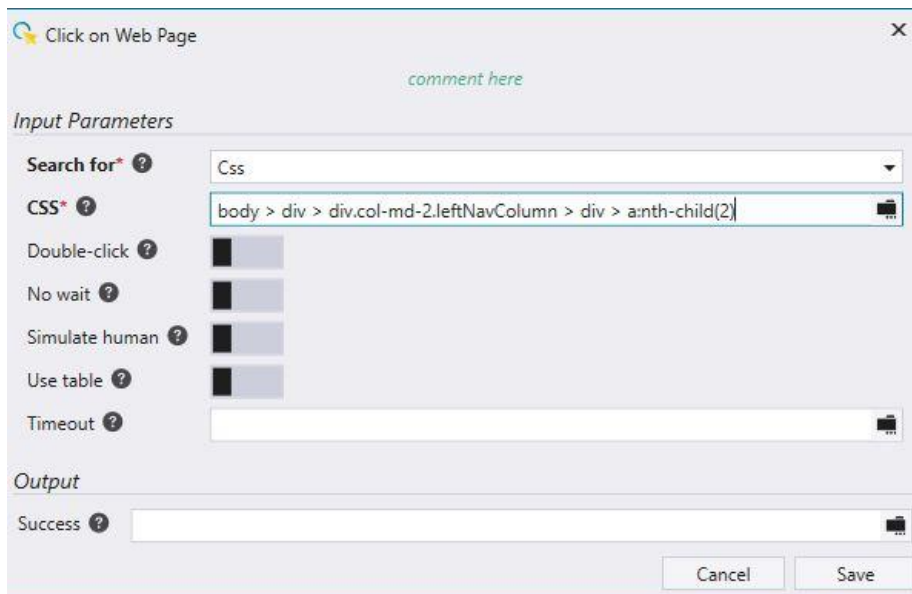
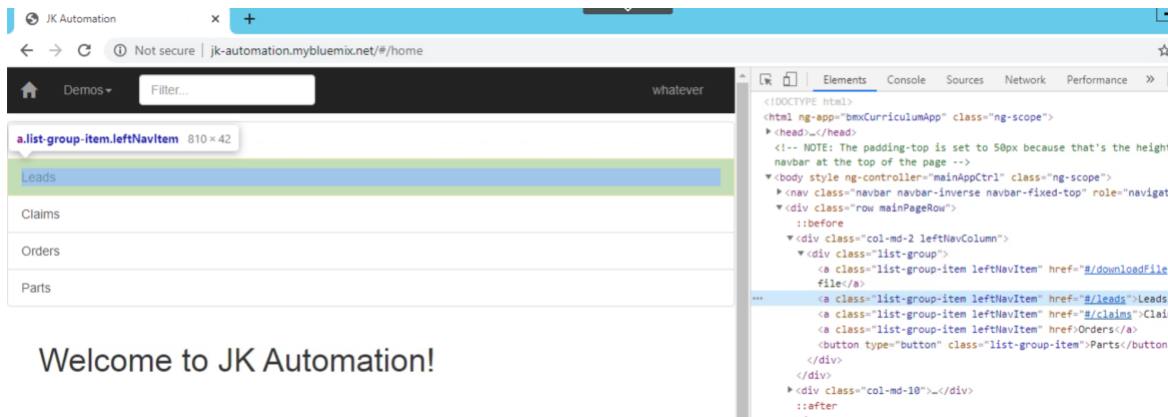


You should see your bot executing, opening the JK Automation web page and logging in with the information that you used to develop the login sequence. When your bot is finished, you should see the JK Automation welcome page.



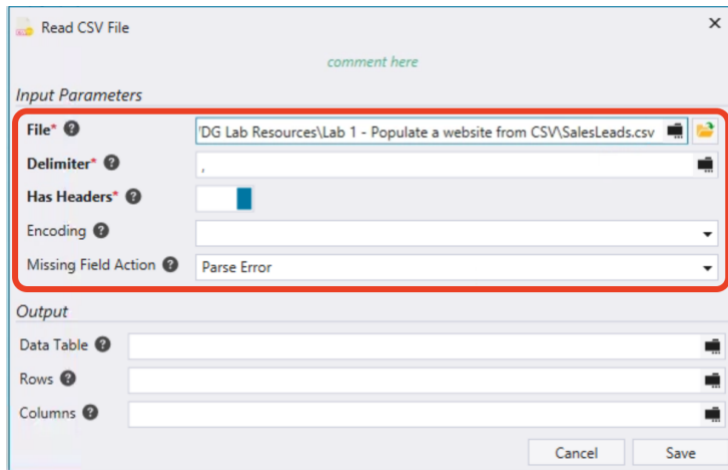
Read CSV

_1. Since we want to handle *Leads*, add a Click on Web Page command to your automation to click the *Leads* link in the left-hand side menu. Yet again, use your browsers element inspector to get the needed selector for the *Leads* link.



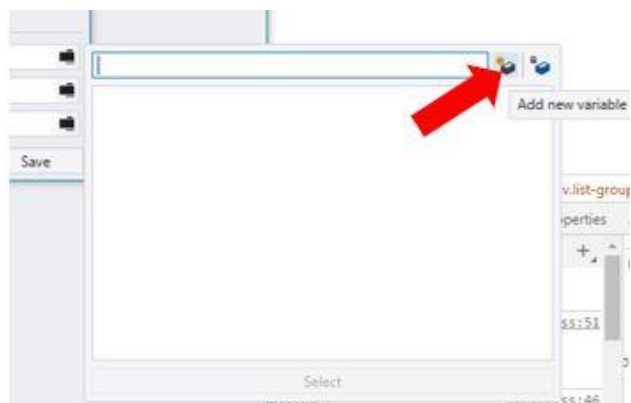
2. Use toolbox search to find `Read CSV File` command and add it after your last `Click on Web Page` command.

3. When the command configuration window opens, use the folder browse icon to select the file **SalesLeads.csv** from folder *Desktop → IBM RPA Lab Resources → Lab 1 – Build your first IBM RPA bot*. Leave all the other selections as they are in default.

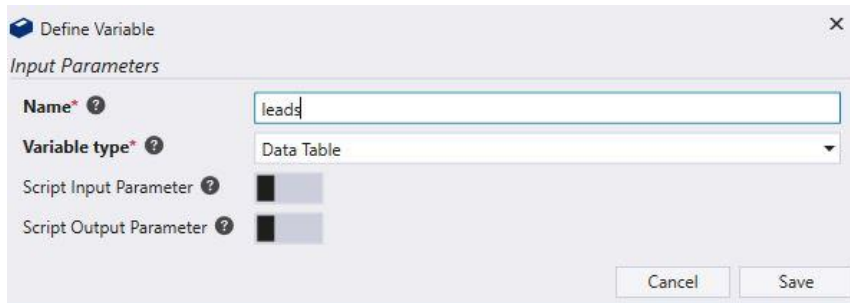


There are three outputs for the command: *Data Table* (holds the data), *Rows* (number of rows in the data table) and *Columns* (number of columns in the data table). We want to store these to variables.

4. First, **Click** the folder icon beside the *Data Table* field to open variable list (as shown in picture above). Next, click `Add new variable` icon and the `Define Variable` window opens.

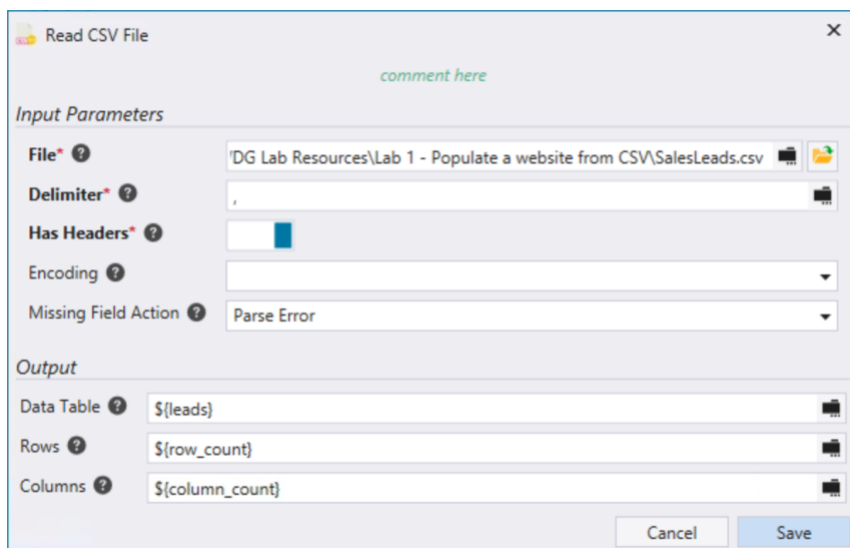


_5. Name the variable as **leads**. Notice that the variable type is automatically set to match the output type (Data Table). Click **Save**.



The 'Define Variable' dialog box is shown. It has a title bar with a close button. Below the title bar is the 'Input Parameters' section. It contains three fields: 'Name' with the value 'leads', 'Variable type' with a dropdown menu set to 'Data Table', and two checkboxes for 'Script Input Parameter' and 'Script Output Parameter', both of which are unchecked. At the bottom right are 'Cancel' and 'Save' buttons.

_6. Repeat the previous steps to create variables also for other outputs *Rows* and *Columns*. Name them **row_count** and **column_count**, respectively. Your **Read CSV File** command configuration should now look as follows. Click **Save**.



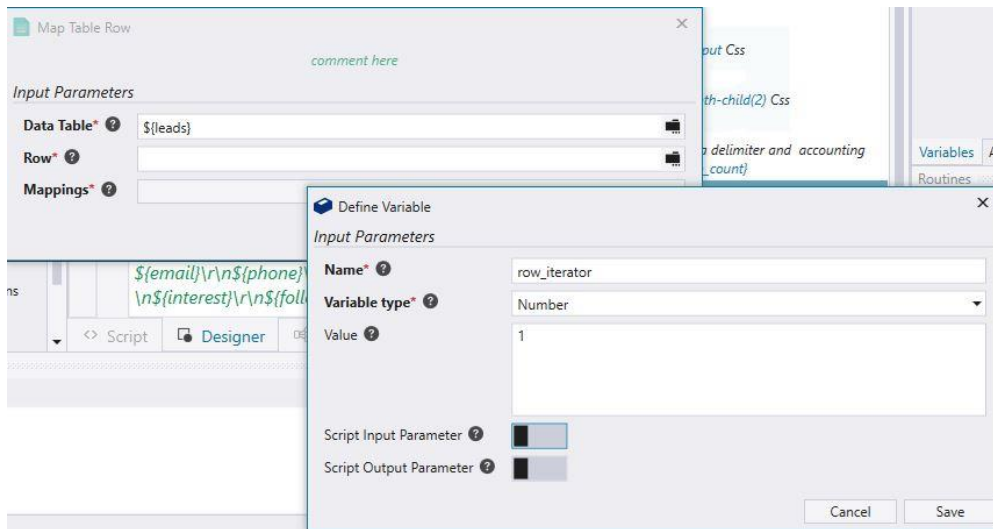
The 'Read CSV File' dialog box is shown. It has a title bar with a close button. Below the title bar is a green text label 'comment here'. The 'Input Parameters' section contains: 'File' with a file path, 'Delimiter' with a dropdown set to comma, 'Has Headers' with a checked checkbox, 'Encoding' with a dropdown, and 'Missing Field Action' with a dropdown set to 'Parse Error'. The 'Output' section contains three fields: 'Data Table' with the value '\${leads}', 'Rows' with the value '\${row_count}', and 'Columns' with the value '\${column_count}'. At the bottom right are 'Cancel' and 'Save' buttons.

Store row data in variables

Now, we obviously want to iterate through all the rows within the data table and insert the data to JK Automation Sales Leads page. Let's focus first on getting a lead read from the data table.

_1. Use your toolbox search to find **Map Table Row** command and then drag and drop it under the **Read CSV File** command in your Designer view.

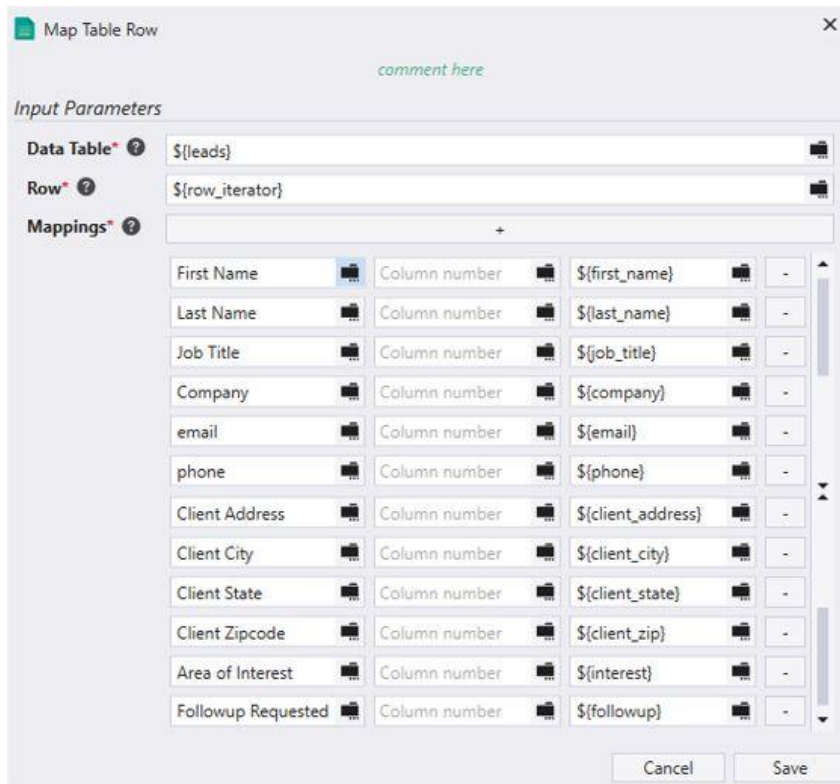
2. When the configuration window opens, select variable **leads** to Data Table and then create a new variable for Row called row_iterator (since we want a variable to help us iterate through the data table) and set the default value for it to **1**.



Next we need to create mappings for the data table row. If you open the CSV file that we're using for the lab with Excel / Notepad, you can see that there is a header row (with column names) followed by 12 data rows and also 12 different columns: **First Name, Last Name, Job Title, Company, email, phone, Client Address, Client City, Client State, Client Zipcode, Area of Interest and Followup Requested**.

1	First Name,Last Name,Job Title,Company,email,phone,Client Address,Client City,Client State,Client Zipcode,Area of Interest,Followup Requested
2	Dave,Wakeman,Director IT,Wakeman Aviation Enterprises,dwakeman@us.ibm.com,515-555-4981,9321 Fleur Drive,Des Moines,IA,50309,Robotic Proc
3	Pramod,Prince,IT Developer,United Third Bank,pramod@unitedthirdbank.com,612-555-2341,332 Hennipen Ave,Minneapolis,MN,55404,Microservice
4	Otto,Pilot,CIO,"Autonomous Cars, Inc.",otto@autocars.com,669-555-8221,9876 Infinite Loop,San Joes,CA,95002,Autonomous Cars,Yes
5	Stu,Liebowitz,Vice President,Craymore Computers,stu@craymore.com,512-555-1488,11501 Burnet Rd,Austin,TX,78702,Business Process Managemen
6	Sheri,Williams,Director of Operations,Piston Manufacturing,sheri@pistonmfg.com,313-555-2991,503 Fasbinder Parkway,Detroit,MI,48202,IT Service P
7	Monika,Datar,Data Scientist,Covfefe Pharmaceuticals,monika@covfefepharma.com,212-555-5591,51 Astor Place,New York,NY,10026,Machine Learnir
8	Jeff,Goodhue,Director of Programming,Dojo Networks,jeff@dojo.com,424-555-4398,33912 Sunset Blvd,Culver City,CA,90233,Beating Ken Jennings at
9	Satoshi,Nakamoto,Lead Developer,Programming Anonymous,satoshi@coinbit.com,929-555-2358,3928 Crypto Lane,Seattle,WA,98114,blockchain,No
10	Timothy,McGee,Data Scientist,Coach K Communications,tim@coachk.com,919-555-0720,928 hoopsville Lane,Raleigh,NC,27676,Data Science,Yes
11	Josh,Bockman,Director of IT,HealthFirst Patient Services,josh@healthfirst.com,617-555-1103,321 King St,Boston,MA,2132,Business Rules,Yes
12	Stuart,Jones,Enterprise Architect,Wessex Financial Services,stuart@wessex.com,630-555-9227,71 S Wacker,Chicago,IL,60624,Cloud,Yes
13	Richard,Scott,Consultant,Scott Professional Consulting,Richard@scottconsulting.com,602-555-1194,119 Barkley Drive,Phoenix,AZ,85028,Security Solu

_3. Click plus sign (+) within your Map Table Row configuration window to add all the needed mappings for your row. On the left side, you need to type in the Column Name from the CSV file (watch out for typos) to identify the data for the mapping, on the right side, you need to first create a new variable to hold the data within the bot. **Map and name the variables as follows.**



TIP:

To save time, using the **script view** in your Studio you can cut and paste the following line:

```
mapTableRow --dataTable ${leads} --row ${row_iterator} --mappings "name=First
Name=${first_name},name=Last Name=${last_name},name=Job
Title=${job_title},name=Company=${company},name=email=${email},name=phone=${phone},name=Client
Address=${client_address},name=Client City=${client_city},name=Client
Zipcode=${client_zipcode},name=Area of Interest=${interest},name=Followup
Requested=${followup},name=Client State=${client_state}"
```

If do this, you also need to copy the new variable definitions to the start of your script:

```
defVar --name first_name --type String
defVar --name last_name --type String
defVar --name job_title --type String
defVar --name company --type String
defVar --name email --type String
defVar --name phone --type String
```

```
defVar --name client_address --type String
defVar --name client_city --type String
defVar --name client_zipcode --type String
defVar --name interest --type String
defVar --name followup --type String
defVar --name client_state --type String
```

You can find both of these from the lab folder as text files that you use for copying and pasting.

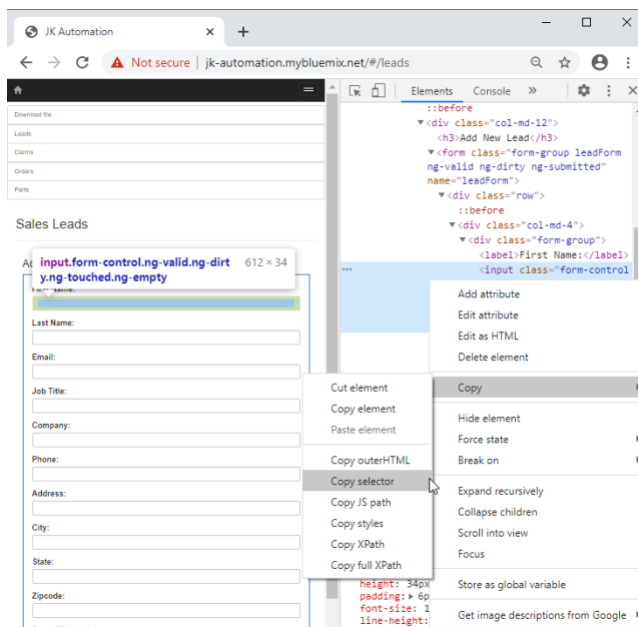
_4. Click **Save** to save and close the configuration. Now we need to insert the data to JK Automation Sales Leads page that you should still have open in your browser. Click **Leads** from the left-hand side navigation menu, if not already on Sales Leads page.

If you have closed JK Automation web page, you can always run your current automation to get the JK Automation Sales Leads page opened and continue from there. For each run, a new browser will be opened. While Studio is able to deal with these multiple open browser windows without issues, it helps to close all but one from time to time to avoid confusing the developer.

Set Sales Leads input fields

You will use a series of **Set Value to Field** commands to insert data that we just extracted to JK Automation Sales Leads page and to its matching fields. Let's go through the first mapping together.

_1. Right-click the **First Name** input field in the web page, select **Inspect > Copy > Copy Selector**.



- _2. Add Set Value to Field command to your automation under Map Table Row command.
- _3. For Selector Type select **Css**, for **Css** copy the CSS Selector from (as shown in Step 1), for Value use the extracted variable **first_name**. Enable Simulate Human option. Click Save.

The screenshot shows a dialog box titled "Set Value to Field" with a close button (X) in the top right corner. Below the title bar, there is a green text placeholder "comment here". The dialog is divided into two main sections: "Input Parameters" and "Output".

Input Parameters:

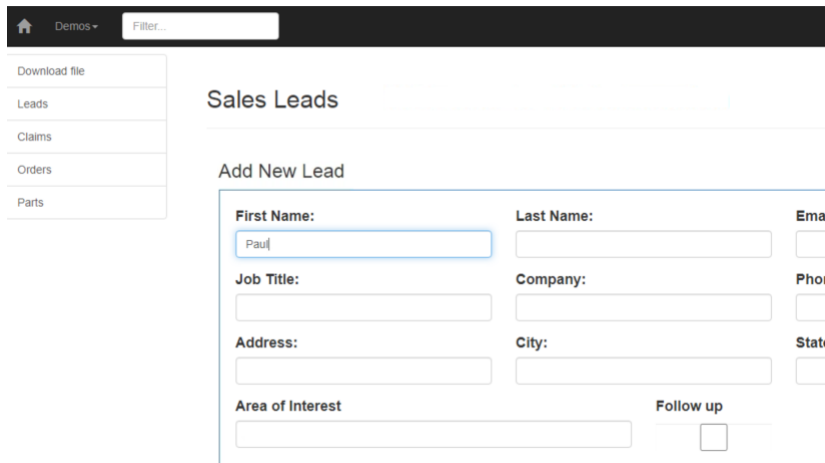
- Value* ?**: A text input field containing "\${first_name}" with a selection icon on the right.
- Selector Type* ?**: A dropdown menu currently set to "Css".
- Css* ?**: A text input field containing ".div:nth-child(1) > div > form > div:nth-child(1) > div:nth-child(1) > div > input" with a selection icon on the right.
- Simulate Human ?**: A toggle switch that is currently turned on (blue).
- usetable ?**: A checkbox that is currently unchecked.
- Timeout ?**: A text input field that is empty with a selection icon on the right.

Output:

- Success ?**: A text input field that is empty with a selection icon on the right.

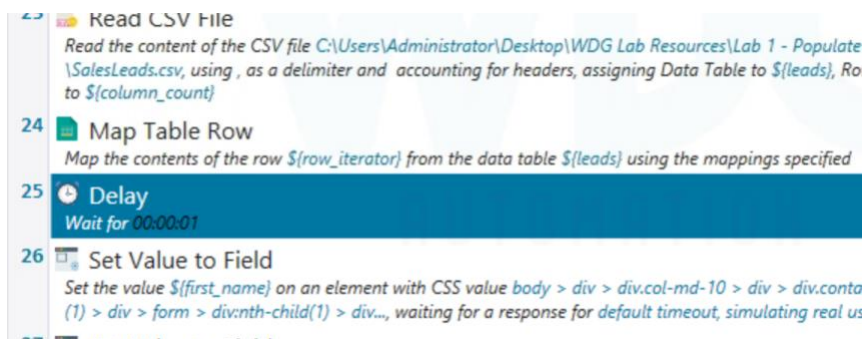
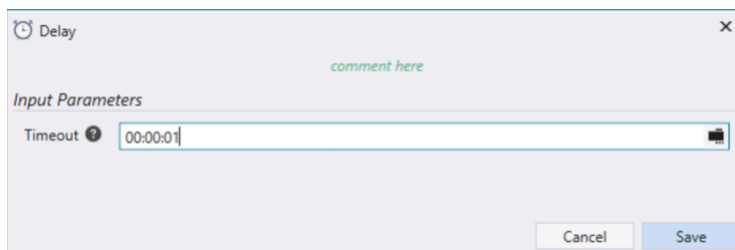
At the bottom right of the dialog, there are two buttons: "Cancel" and "Save".

4. Close the browser with JK Automation site and run your automation to test it. When the script finishes, you should see the Sales Lead page with an entry to First Name input field from your first sales lead.



NOTE! If you do not see First Name field filled in, the automation seems to be too “quick” for the web site, this is quite normal. For demo, we can fix this by adding a small delay before starting to fill in the input fields. Normally you would need to wait for certain elements to appear and be ready. We’ll do this in easy way now 😊

Drag and drop **Delay** command *after the Map Table Row* command and *before the first Set Value to Field* command. Set time out to **00:00:01** for one second.

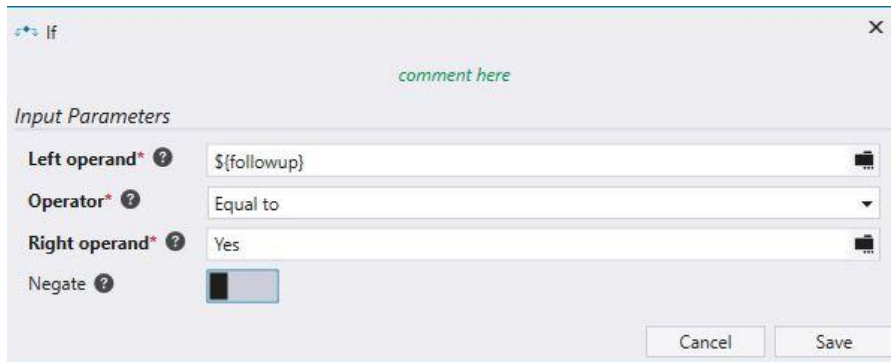


Close the browser with JK Automation and **run** your script again to verify that also the first name field gets populated.

_5. Now, set all the remaining 11 input fields (Area of Interest being the last one) as you did with the First Name field. Remember to enable Simulate Human option.

Note, you can use copy-paste in your Designer view to easily copy commands.

_6. Add **If** command (Under *Base > Flow Control*) to your automation under the last Set Value to Field command. Configure it using the **followup** variable as follows.



The screenshot shows the 'If' command configuration window. It has a title bar with a close button. Below the title bar is a green text label 'comment here'. The 'Input Parameters' section contains four fields: 'Left operand*' with the value '\${followup}', 'Operator*' with the value 'Equal to', 'Right operand*' with the value 'Yes', and 'Negate*' with an unchecked checkbox. At the bottom right are 'Cancel' and 'Save' buttons.

_7. Add Click on Web Page command between *If* and *End If*.

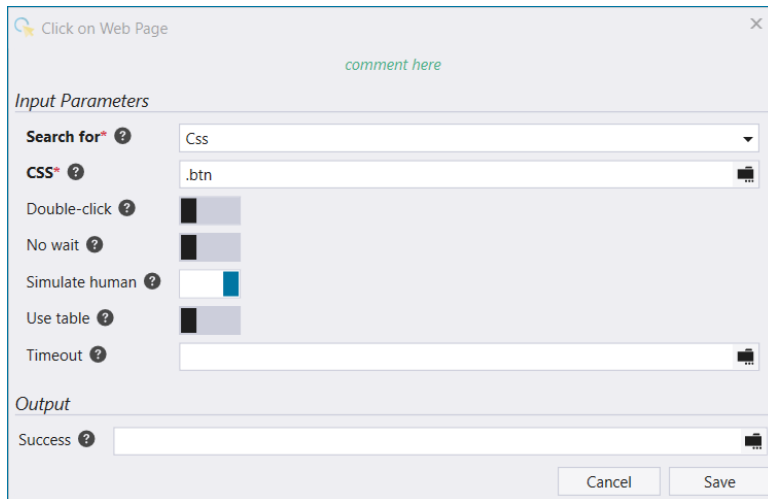
_8. For Selector Type select **Css**, for **Css** copy the CSS Selector for the **Follow up** check box in the Sales Leads web page. Enable Simulate Human option. Click Save.



The screenshot shows the 'Click on Web Page' command configuration window. It has a title bar with a close button. Below the title bar is a green text label 'comment here'. The 'Input Parameters' section contains several fields: 'Search for*' with a dropdown set to 'Css', 'CSS*' with the value '> div:nth-child(1) > div > form > div:nth-child(4) > div:nth-child(2) > div > input', 'Double-click*' with an unchecked checkbox, 'No wait*' with an unchecked checkbox, 'Simulate human*' with a checked checkbox, 'Use table*' with an unchecked checkbox, and 'Timeout*' with an empty text field. The 'Output' section contains a 'Success*' field with an empty text field. At the bottom right are 'Cancel' and 'Save' buttons.

_9. Add Click on Web Page command under *End If* to click the **Submit** button in the Sales Leads web page.

_10. For Selector Type select **Css**, for **Css** copy the CSS Selector for the **Submit** button web. Enable **Simulate Human** option. Click **Save**.



Click on Web Page

comment here

Input Parameters

Search for* ?

CSS* ?

Double-click ? ☐

No wait ? ☐

Simulate human ? ☒

Use table ? ☐

Timeout ?

Output

Success ? ☐

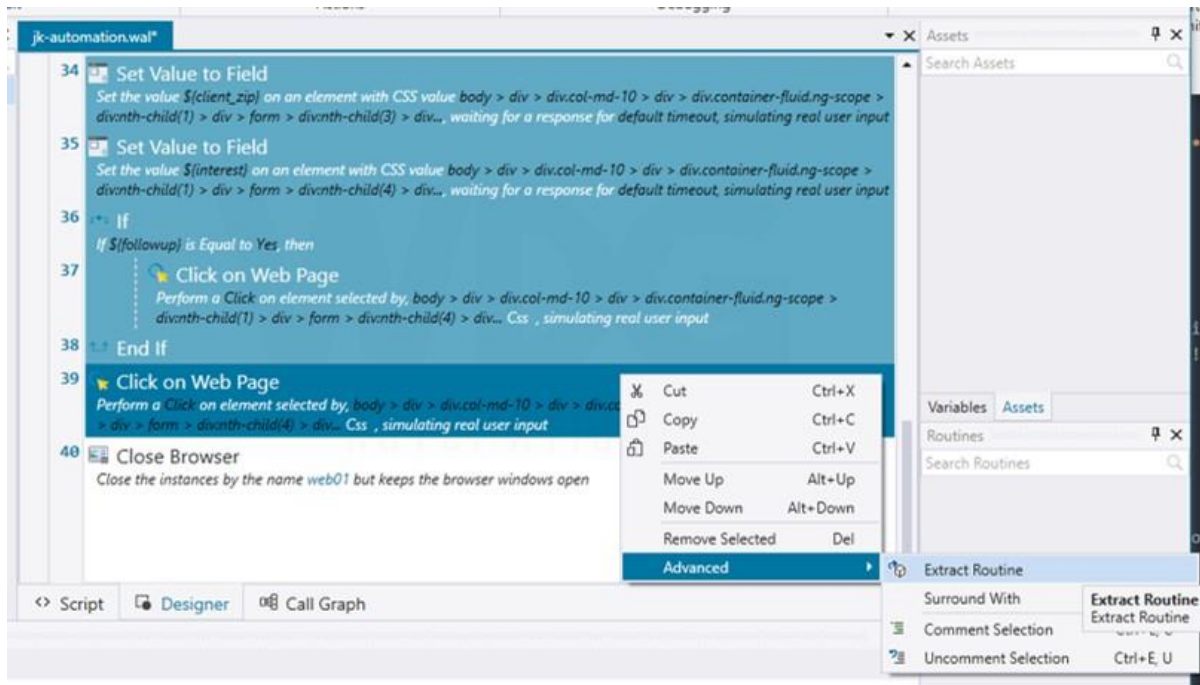
Cancel Save

_11. **Save** your work, **close your browser** showing the JK Automation web site and **run** your automation by hitting the **Run** icon in the top toolbar. You should see your automation executing and entering the first sales lead (Paul Pacholski) to the JK Automation Sales Leads page. Nice!

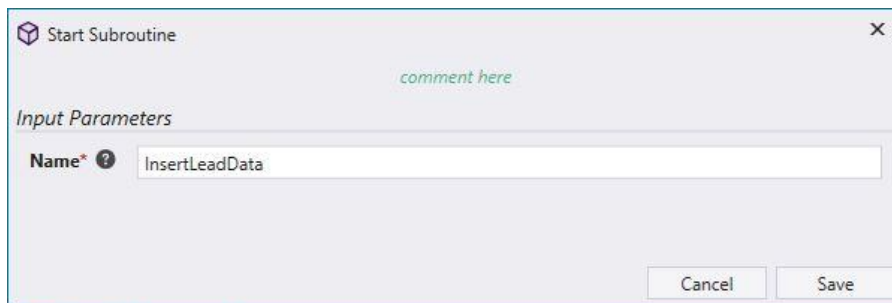
Define Subroutines

IBM RPA Studio allows you to define and group parts of your automation as sub-routines. This can be useful to group logic parts of your automation script and make your automation more easily understandable.

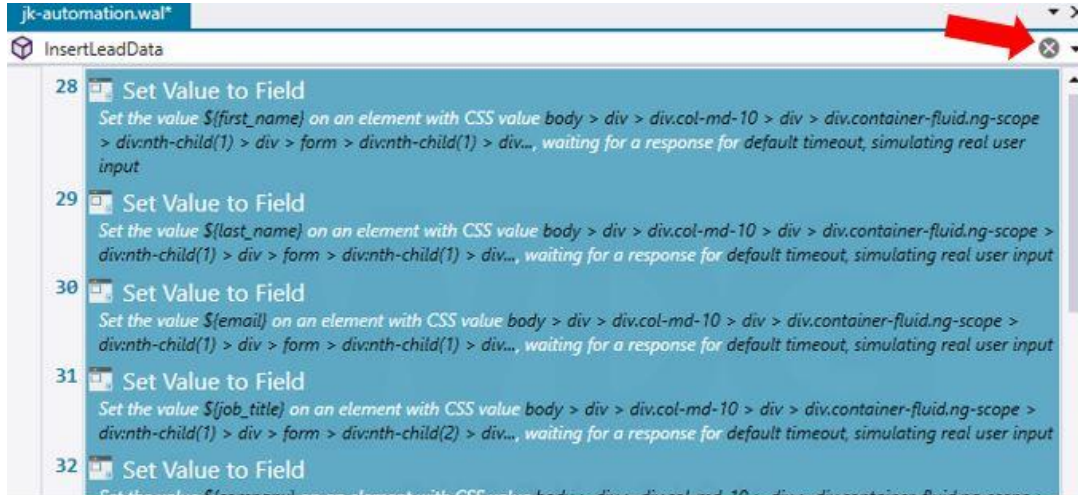
_1. In your WGD Studio Designer view, select (*Shift+Click*) **all commands** between *Map Table Row* and *Close Browser* (**NOTE!** Do **NOT** select *Map Table Row* and *Close Browser*!), right-click one of the selected commands and select **Advanced** > **Extract Routine**.



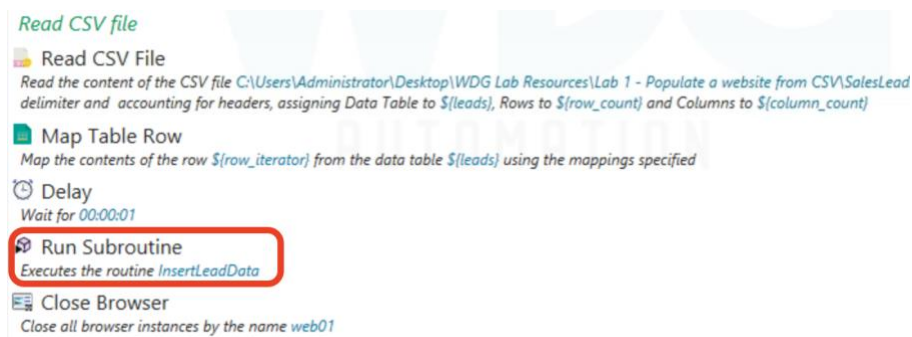
_2. Name your sub-routine as **InsertLeadData**. Click Save.



_3. Studio will show the extracted sub-routine. You can close it and return to "main" automation by clicking the *close icon* after the name of the sub-routine.



_4. You can now see the extracted sub-routine in your "main" automation script.

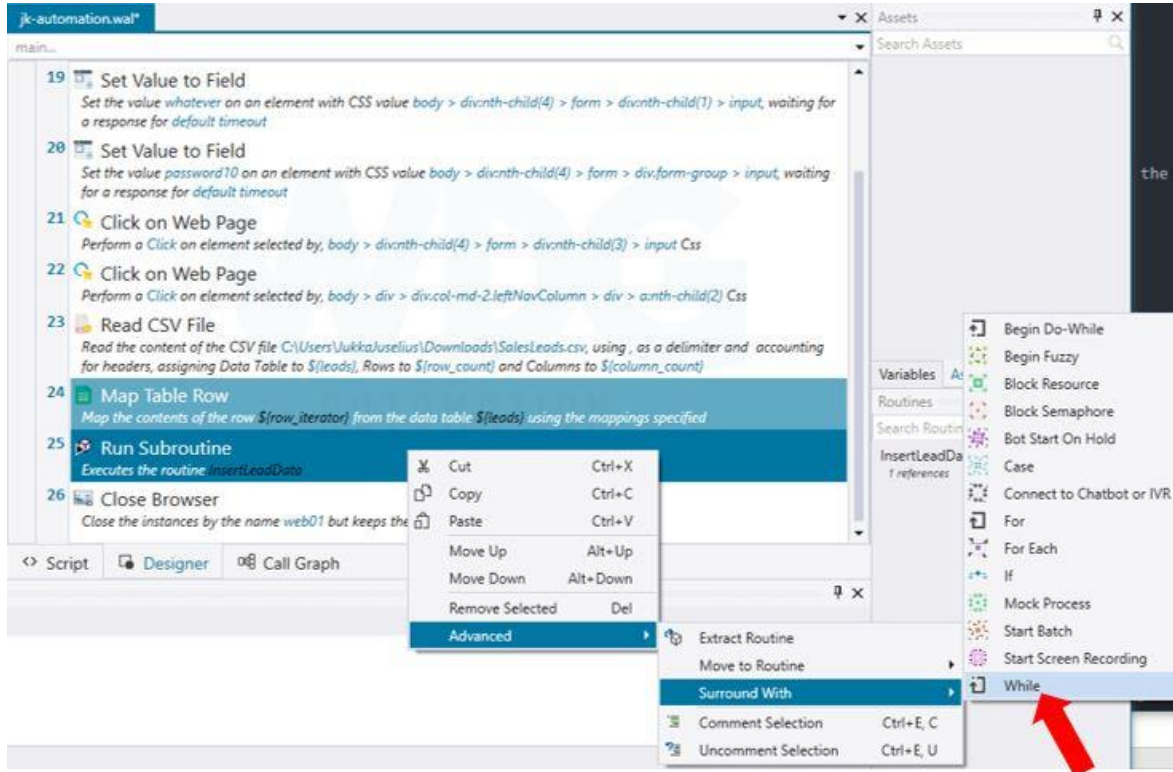


Note, variables in IBM RPA Studio are global. This means that you do not need to map variables between your main automation script and its sub-routines.

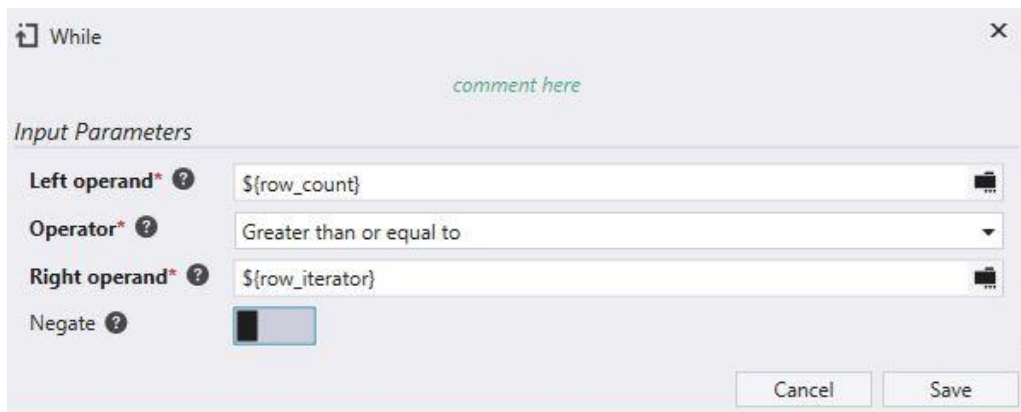
Add loop to iterate through the data table

We have our automation almost ready. One of thing we still need to do, is to add a looping structure to iterate through all the sales leads within the data table that we read from the CSV file. IBM RPA studio offers several structures to do this. We will use a `while` loop and use the `row_count` variable to go through all the records.

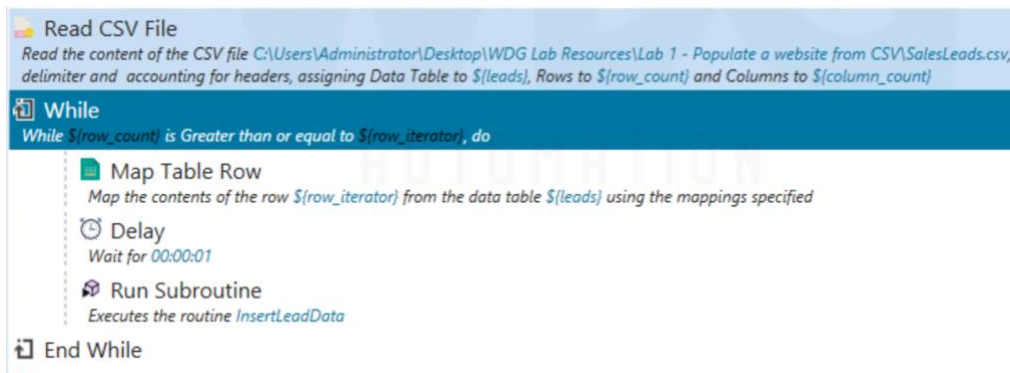
1. Select the Map Table Row and the Run Subroutine commands, **right-click** one of the selected commands, and select **Advanced** > **Surround With** > **While**.



2. Configure the While command using the **row_count** and **row_iterator** variables as follows. Click Save.

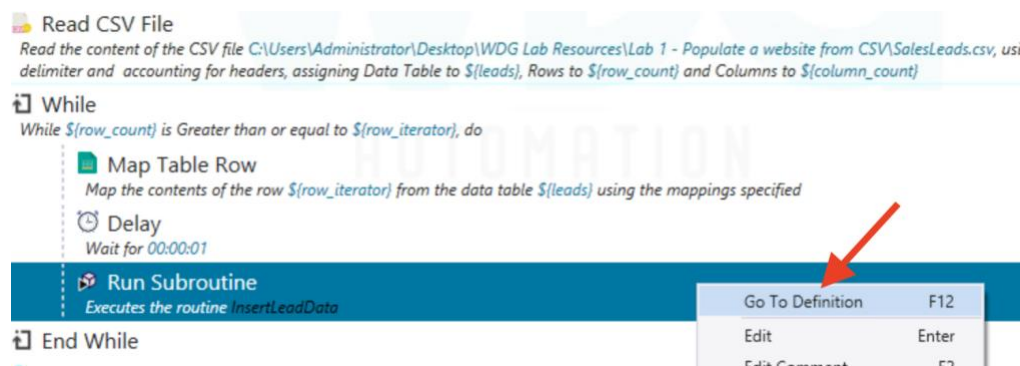


You will now see the While loop in your main automation script.



Good job! But something is missing... Right! We need still need to make sure that we increase our **row_iterator** value for each iteration.

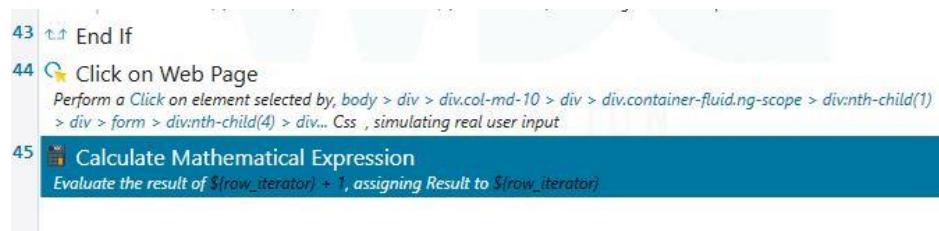
_3. **Right-click** the Run Subroutine inside the While loop and select **Go To Definition**.



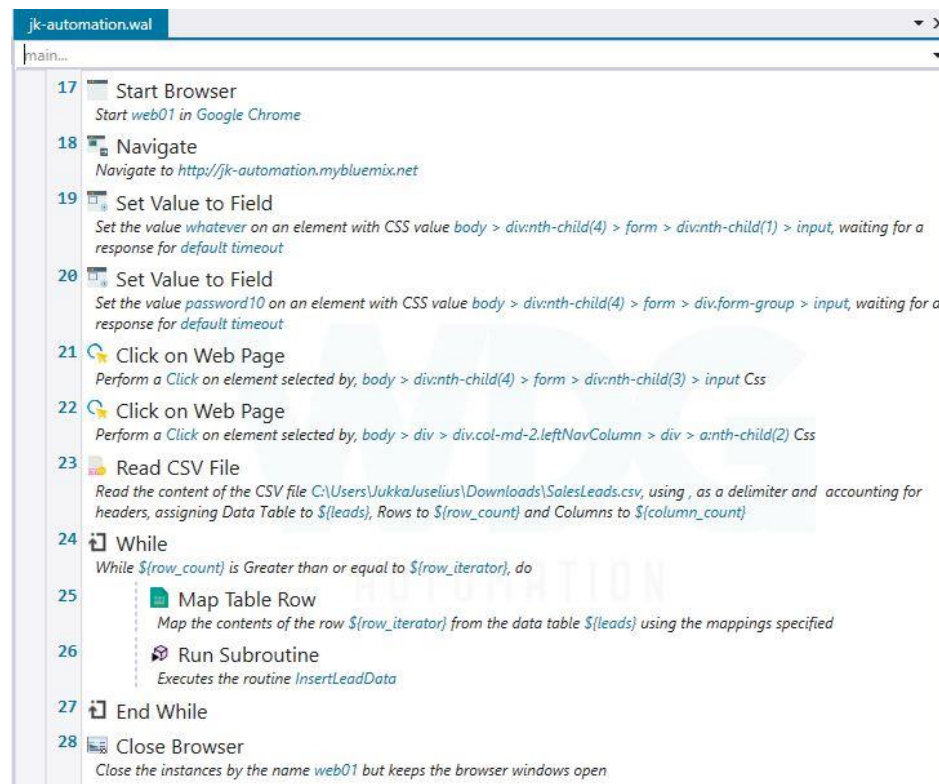
_4. Add Calculate Mathematical Expression command (Base > Numeric) as the last command of the sub-routine. Configure it as follows. We want to add +1 at the end of each iteration. Don't forget to set the output result to \${row_iterator}!



You should now have the command added to your sub-routine.



5. Close the sub-routine to go back to main automation script. **Save** your work.



6. Close your current browser window showing the JK Automation web site and **Run** your automation to test that it works and JK Automation website is opened. Click the green Run icon from the top toolbar. Alternatively, you can hit F5.



You should see the automation / bot executing and adding all the sales leads to the JK Automation web site. When the bot finished running you should now see all 12 rows from the excel spreadsheet in the Sales Leads table, first one being *Paul* and the last one *Stuart*.

Logout from JK automation website

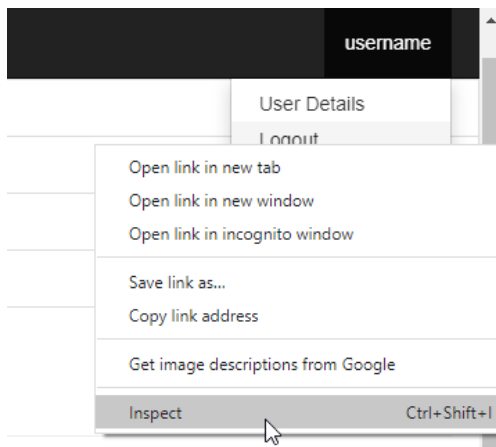
Recall that the Close Browser command was configured to left browser open for testing.

To complete our automation. Let's implement logout from the JK Automation site by adding one Click on Web Page command at the end of your automation, just before the Close Browser command.

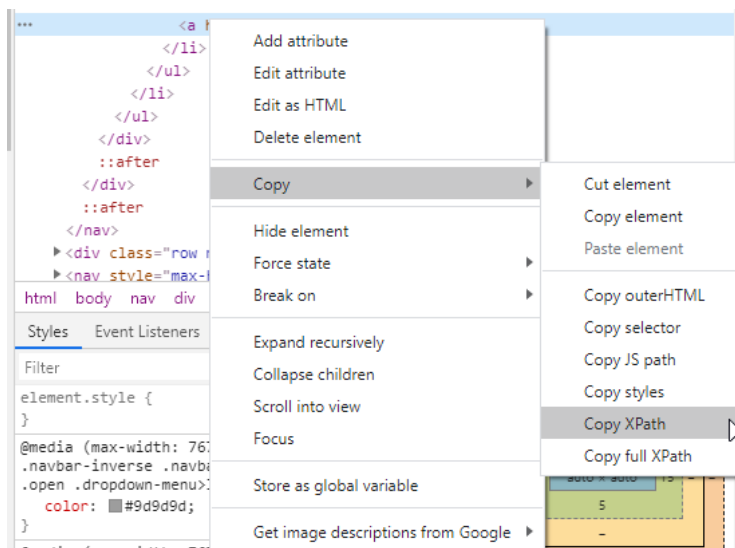
_1. Click your **username** on the right-hand side upper corner of the web page.



_2. Select the **logout link** from the menu, right click and then select **Inspect**.

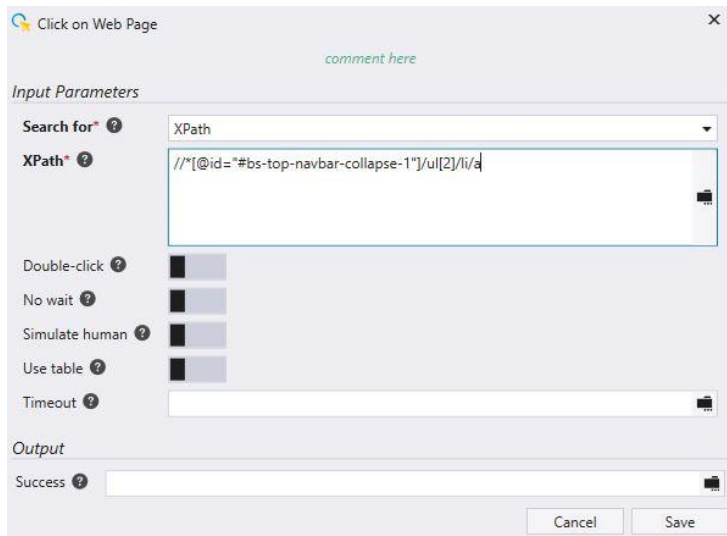


_3. Select **Copy > Copy XPath**.



_4. Add **Click on Web Page** command after *End While* command.

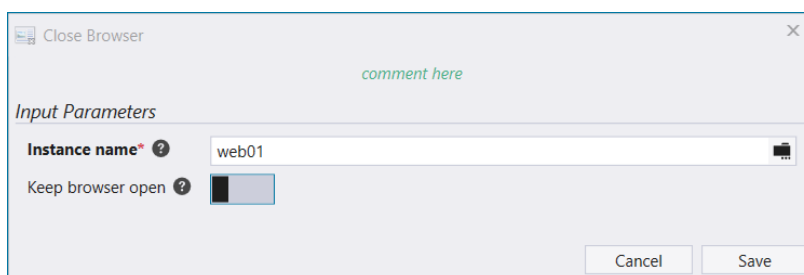
_5. Set *Search for* to **XPath** and *Xpath* to **the value from your clipboard** (the XPath you copied from element inspector). Finally click *Save*.



The last three commands in the Designer should now look similar to this:



_6. Double click **Close Browser** command at the end of your automation and change the **Keep browser open** **off** and click *Save*.



_7. Test your automation again. You should now see the browser close after the automation completes.

Well done! Congrats finishing the exercise! Hope you learned some of the basics on how to use IBM RPA Studio. Keep on automating 😊

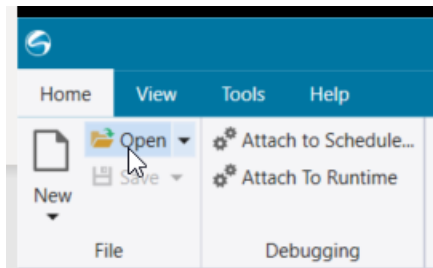
THIS COMPLETES THIS HANDS-ON LAB

3. Work with Completed Automation

If you just want to run the automation without authoring it or if you just want to look at a completed solution, follow these instructions below.

Open completed lab

1. In IBM RPA Studio, Select **Open**.



2. Navigate to **sales-lead-automation-completed.wal** in folder *Desktop* → *IBM RPA Lab Resources* → *Lab 1 – Build your first IBM RPA bot* and click **Open**.

Check input csv file location

1. Double click on the **Read CSV File** command



2. Make sure that the location of the file is set to *Desktop* → *IBM RPA Lab Resources* → *Lab 1 – Build your first IBM RPA bot* → **SalesLeads.csv**.

Run the automation

1. Click the green **Run** icon from the top toolbar. Alternatively, you can hit F5.

