RPA Bootcamp
Your First Bot

Lab 01.00a

⌛90-120-min

Lab Objectives

In this lab you will be working in RPA Desktop Design Studio to learn to automate the following tasks:

- Reading data from an HTML Table
- Saving data to an Excel File
- Send an email with an attachment.

> ***Lab Dependency:*** *Requires installing RPA Desktop Design Studio (Windows only)*
>
> ***Lab Dependency:*** *Requires installing Service Now Google Chrome Extension*

Scenario

Users are faced with the highly repetitive task of checking a website every half an hour, extracting data from that website, and sending an update via email. This process is tedious, repetitive, and error-prone, so we want to automate it to save time and effort.

Requirements Summary

User Stories

- **STRY010101 – Check the exchange rate**

  As an end user, I need to copy the current exchange rate from a website into Excel, and send the update Excel file via email

Story Mode:

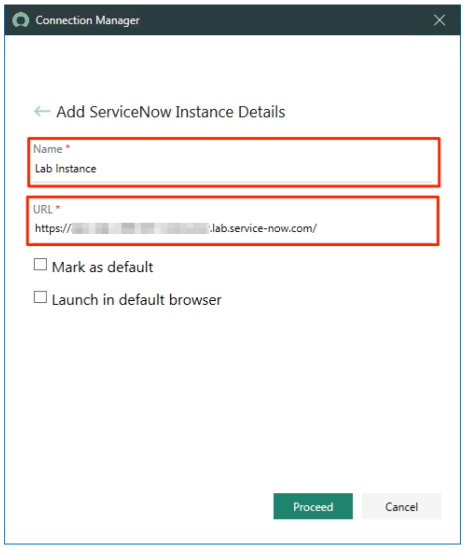
Check the Exchange Rate

A. Create a Currency Bot

Create and Configure a Project

0. Open **RPA Desktop Design Studio**

1. Enter your instance details on the Connection Manager Dialog



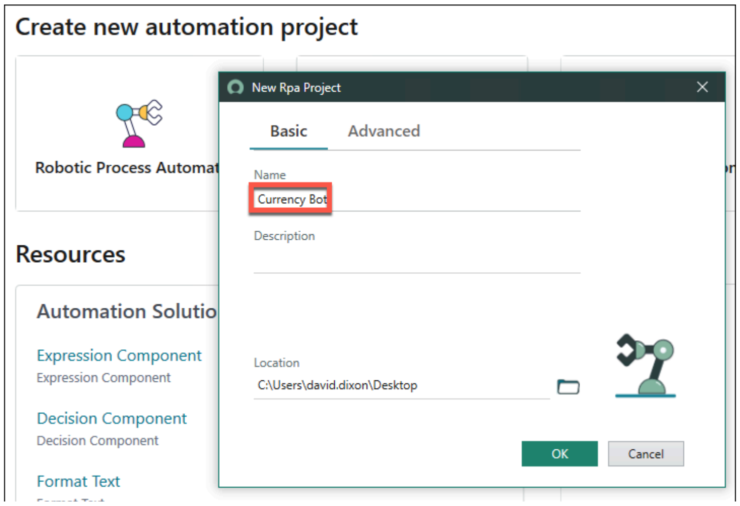
Name *
Lab Instance

URL *
https://[redacted].lab.service-now.com/

Note: The first start might take some time as RPA Desktop Design Studio is updating plugins to match the instance version.

2. From the Home screen, choose to create a new Unattended Automation project.

3. Name the project **Currency Bot** and save it to the default location.


Create new automation project

Robotic Process Automat

Resources

Automation Solutio

Expression Component

Expression Component

Decision Component

Decision Component

Format Text

New Rpa Project

Basic | Advanced

Name: Currency Bot

Description

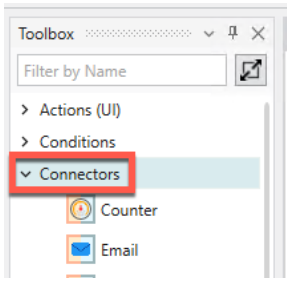
Location: C:\Users\david.dixon\Desktop

OK | Cancel

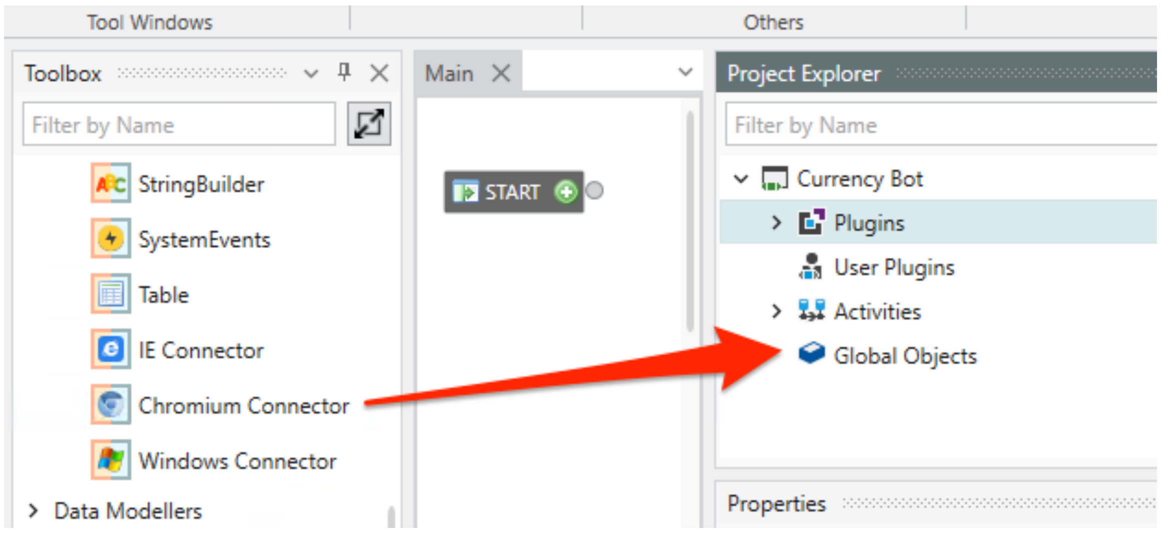
Note: *Note: The default storage location for your automation is on the user's desktop. You can change this to any folder you like.*

- This lab will use Google Chrome as browser engine. To enable add the **Chromium Connector** to the project. This connector can work with any chromium based browser.

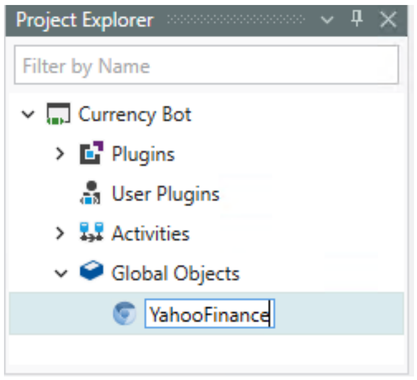
In the **Toolbox**, expand **Connectors**



4. Drag the **Chromium Connector** from the **Toolbox** to **Global Objects** in the **Project Explorer** on the right



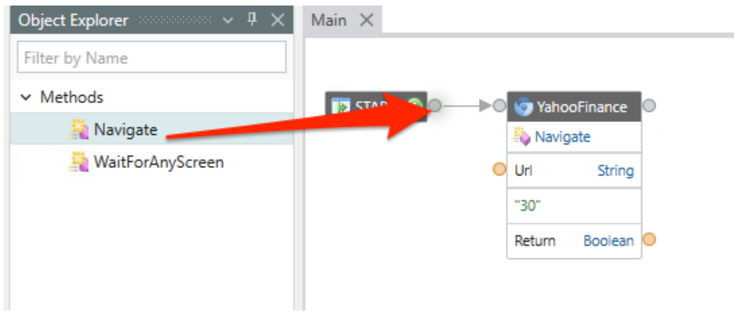
5. Right-click the newly added Chromium Connector (WebApplication) in Project Explorer, and **Rename** it as **YahooFinance**



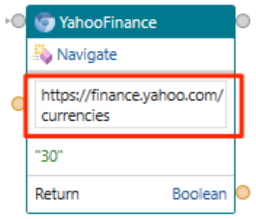
Configure the Chromium Connector

0. Double-click on the newly created **YahooFinance** object and ensure that the **Methods** update in the **Object Explorer**.

1. Drag the **Navigate** method from the Object Explorer to the control port so that it connects to the **START** activity.



2. Double-click the **Url** property and add the value https://finance.yahoo.com/currencies

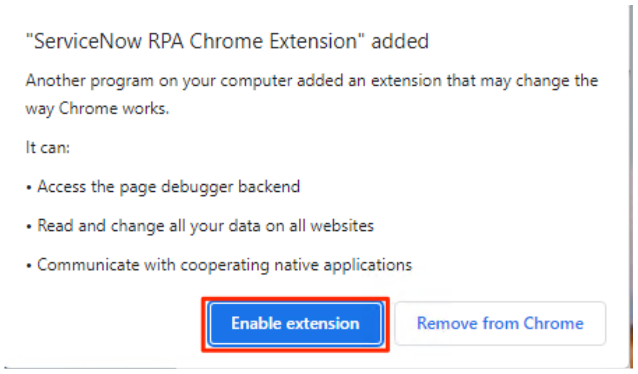


3. Right-click the **YahooFinance** component and select **Run From Here**. Confirm that **Google Chrome** opens and navigates to the specified website.

4. Enabling the chrome extension

> ***Note:***
> *Note: The provided virtual machine has this extension already loaded. If you work on your own machine you will need to go the Chrome Web Store and download the extension first.*

Once the page is loaded chrome will inform you that ServiceNow RPA Chrome Extension has been added. **Enable extension** to allow it access to the website's components.



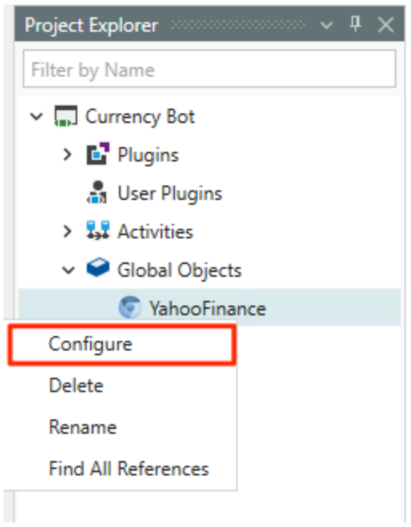
"ServiceNow RPA Chrome Extension" added

Another program on your computer added an extension that may change the way Chrome works.

It can:

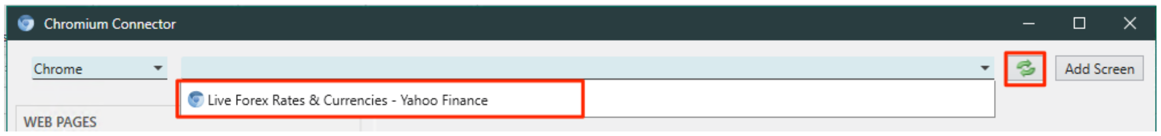
- Access the page debugger backend
- Read and change all your data on all websites
- Communicate with cooperating native applications

1. With Chrome still open, go back into **RPA Desktop Design Studio**, **right-click** on the **YahooFinance** connector in **Project Explorer** and select **Configure**



5. Next to the browser selector (Chrome) on the **AVAILABLE WEB PAGES** drop-down, select the green **refresh button**, and confirm that the drop-down contains the web page that was just opened by your automation.

> ***Note:*** *This step requires that you first have the Service Now Google Chrome Extension installed. Refer to the pre-workshop installation instructions if you have not yet done so. You also must have a web page open in Google Chrome.*

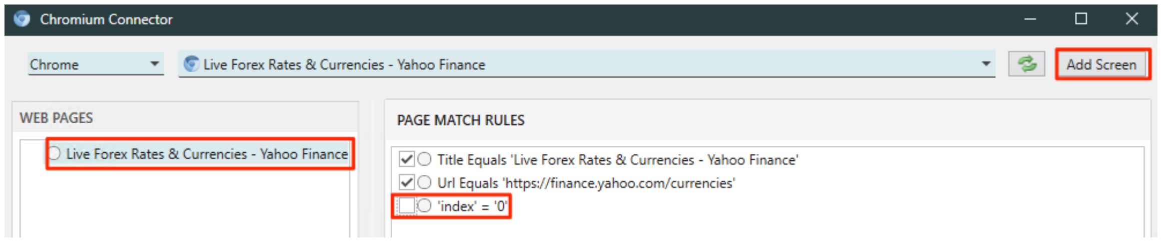


6. Select the Yahoo Finance webpage and choose **Add Screen** so that it is added to the **WEB PAGES** pane on the top left.

7. Click on the web page on the left to load the **PAGE MATCH RULES.**

   One of the created page match rules is 'index' = '0'. In the case that a browser has multiple tabs open with the same page, it will select the very first one – index 0. For most automations you should expect that only one tab is open and hence remove this checkbox.

   Uncheck 'index = 0'



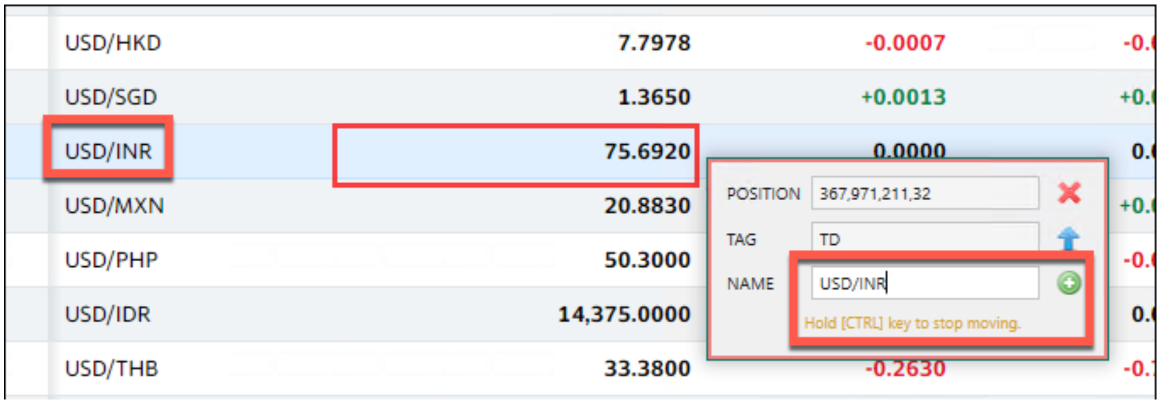
Chromium Connector
Chrome
Live Forex Rates & Currencies - Yahoo Finance
Add Screen
WEB PAGES
Live Forex Rates & Currencies - Yahoo Finance
PAGE MATCH RULES

- [✓] Title Equals 'Live Forex Rates & Currencies - Yahoo Finance'
- [✓] Url Equals 'https://finance.yahoo.com/currencies'
- [ ] 'index' = '0'

8. Right click the web page at the top left and select "Refresh". The page and selected match rules should turn green.

9. Right click the web page again and select "Capture Element" – the Chrome window should come into focus, with a dialog for selecting an element.

10. Hover over the **Last Price** field for the **USD/INR** row, select the CTRL key, and give the element the name USD/INR



USD/HKD	7.7978	-0.0007	-0.
USD/SGD	1.3650	+0.0013	+0.
USD/INR	75.6920	0.0000	0.
USD/MXN	20.8830		+0.
USD/PHP	50.3000		-0.
USD/IDR	14,375.0000		0.
USD/THB	33.3800	-0.2630	-0.

POSITION 367,971,211,32

TAG TD

NAME USD/INR

Hold [CTRL] key to stop moving.

11. With the **Last Price** still selected, click the **green plus icon** to capture the element. The name field will clear.

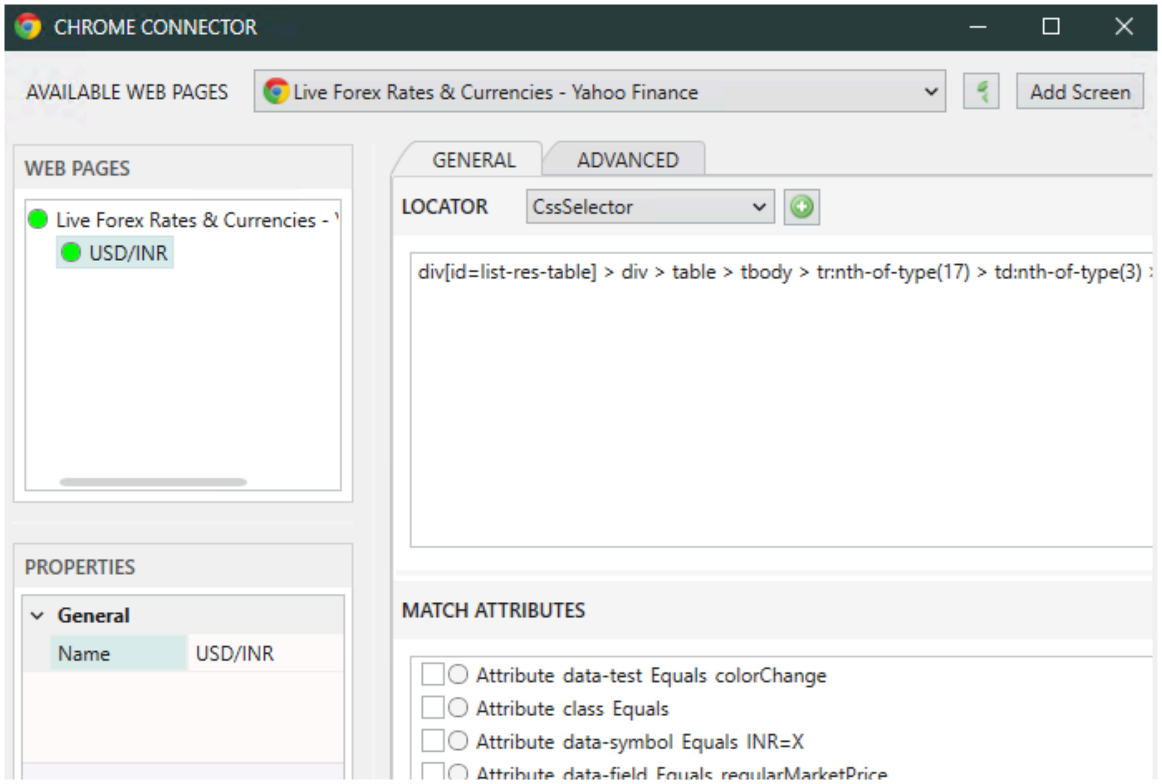
> ***Note:*** *Note: Ensure you select the 'TD' tag when clicking the plus icon.*

12. Close the element capture dialog with the **red X** icon.

13. In the **CHROME CONNECTOR** dialog, Double-click the required element(expand if its not directly visible) again in the top left so that **USD/INR** displays, right-click **USD/INR** element, and choose **Refresh**. The icon next to the element should turn green.

14. Double-click the **USD/INR** element and review the available **LOCATOR** options and **MATCH ATTRIBUTES**. These will be used by the bot to identify the element on the page.

> ***Note:*** *Depending on the nature of the page you are working with, you may want to modify the LOCATOR types and MATCH ATTRIBUTES to ensure that you are using a method that will identify your element if the page changes.*



AVAILABLE WEB PAGES: Live Forex Rates & Currencies - Yahoo Finance

Live Forex Rates & Currencies -
USD/INR

Name: USD/INR

LOCATOR: CssSelector

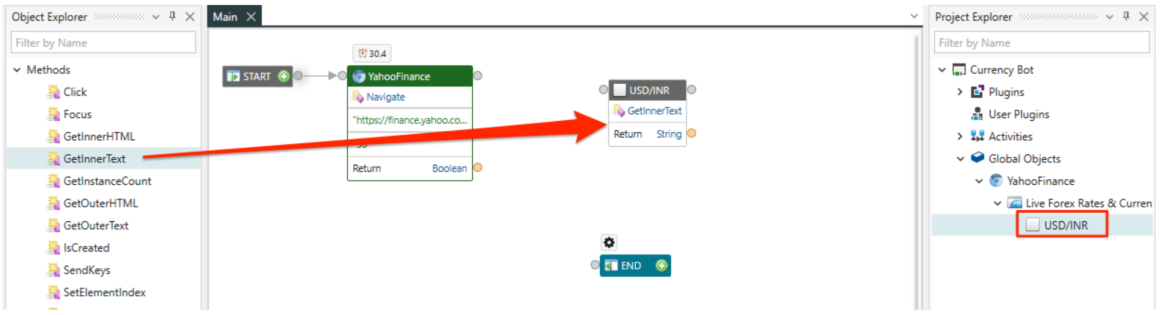
```
div[id=list-res-table] > div > table > tbody > tr:nth-of-type(17) > td:nth-of-type(3) :
```

MATCH ATTRIBUTES

- Attribute data-test Equals colorChange
- Attribute class Equals
- Attribute data-symbol Equals INR=X
- Attribute data-field Equals regularMarketPrice

15. Select **OK** to close the **CHROME CONNECTOR** dialog.

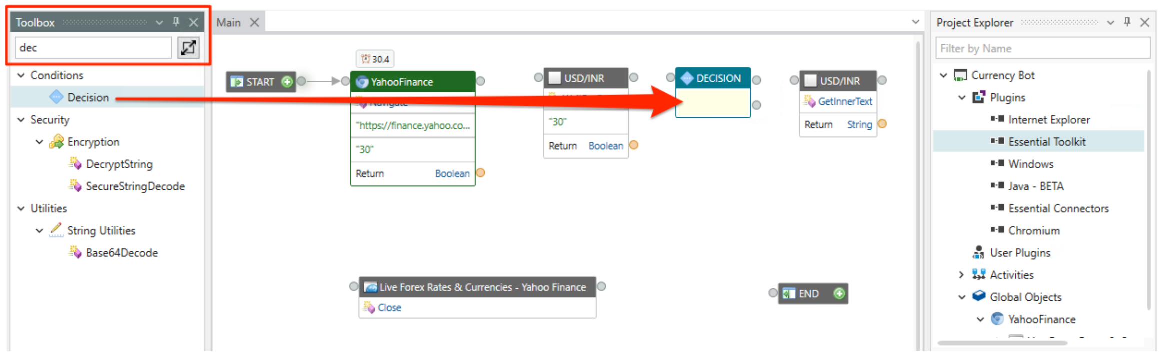
16. In Project Explorer, navigate to the **USD/INR** field, double-click, and drag the **GetInnerText** method from the **Object Explorer** on to your diagram.



17. With **USD/INR** still selected in **Project Explorer**, drag the **Wait for Create** method on to your diagram and place it somewhere before the **GetInnerText** method.

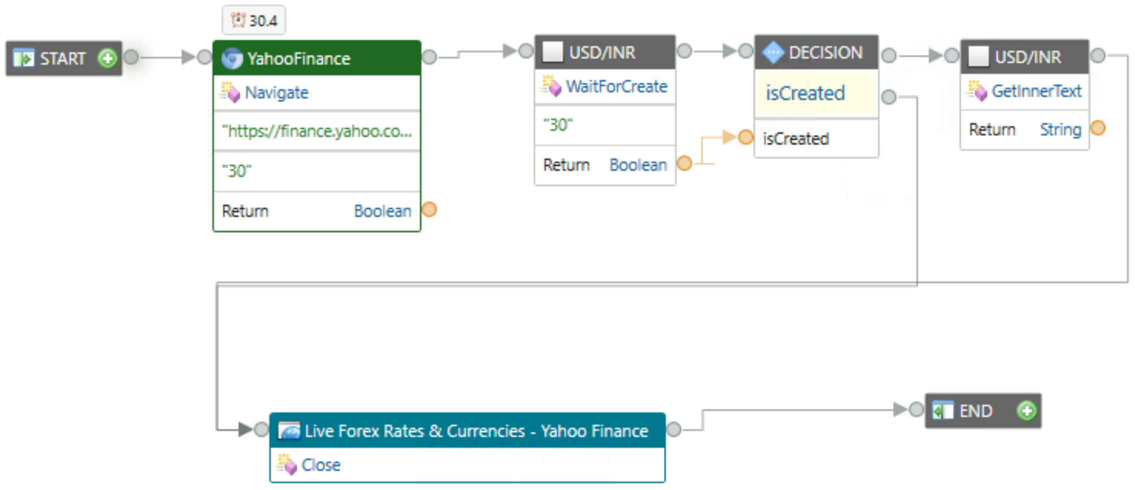
18. Double-click the page component “**Live Forex Rates…**” in Project Explorer and drag the **Close** method from **Object Explorer** on to your diagram.

19. In the **Toolbox**, search for **Decision**, and drag that condition component on to your diagram, between the **WaitForCreate** and **GetInnerText**.

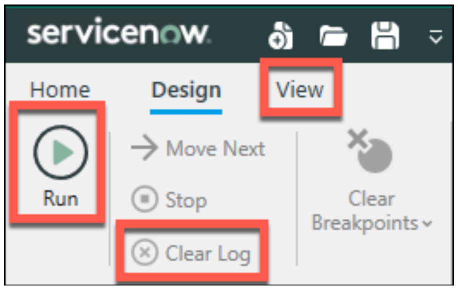


20. Double-click the empty text field on the **Decision** condition and give it a name of **isCreated**.

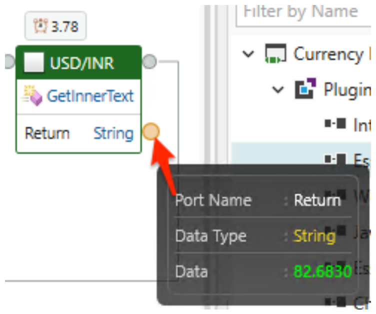
21. Connect the components in your diagram and adjust the order and layout so it looks roughly as shown below



22. On the **Design** tab, clear the log and **Run** your automation. You should see the browser load the page, and eventually all activities should complete successfully and turn green.



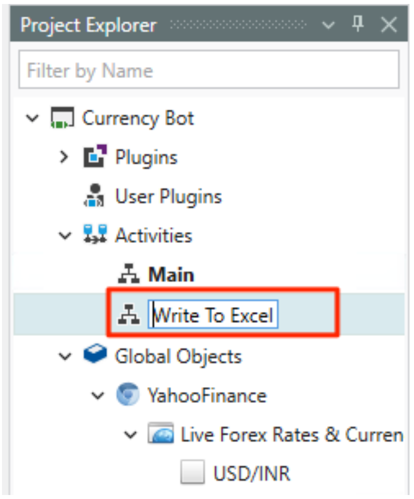
To validate your result you can hover on the yellow data port of the GetInnerText components result which should show the conversion rate of USD to INR as of today.



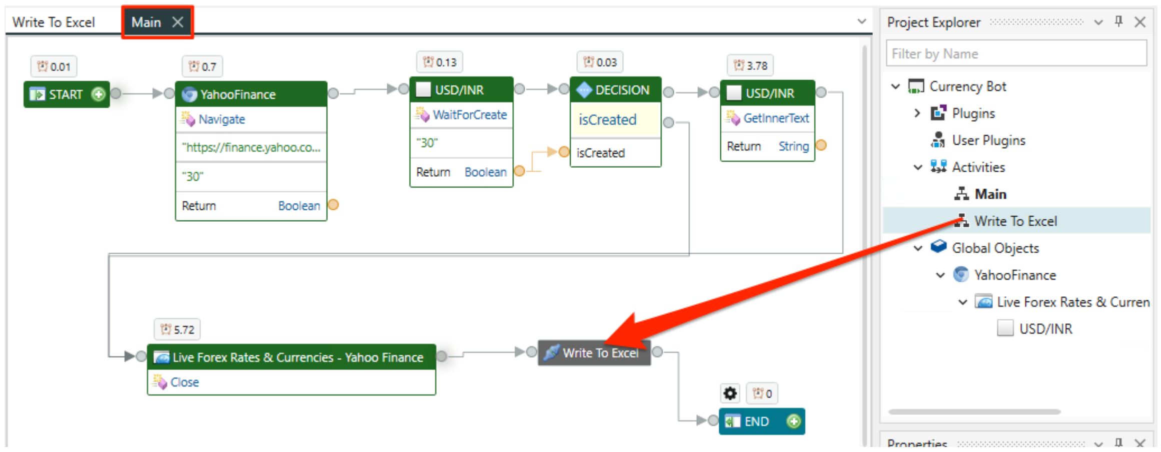
Port Name : Return
Data Type : String
Data : 82.6830

B. Build the Excel Bot

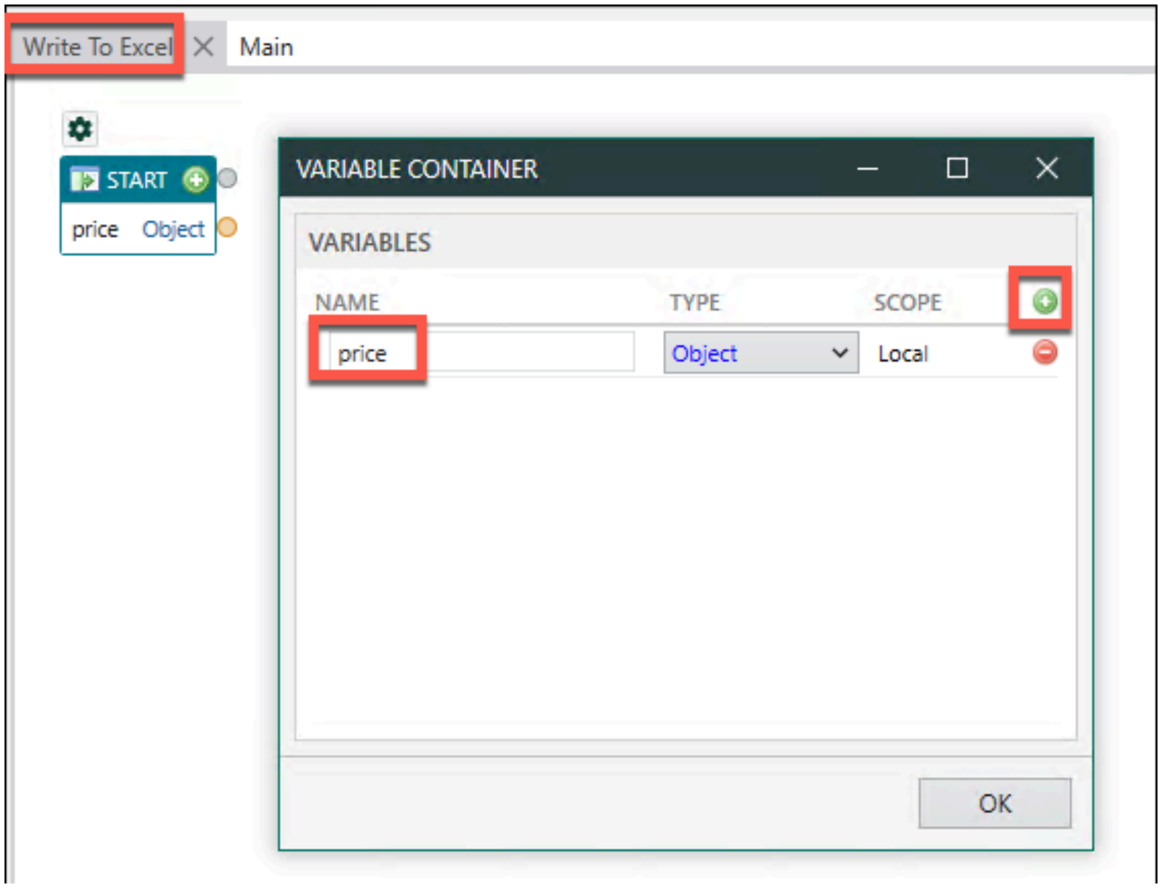
0. In the **Project Explorer**, right click **Activities** and select **New Activity**. Create an activity and rename it as **Write to Excel**.



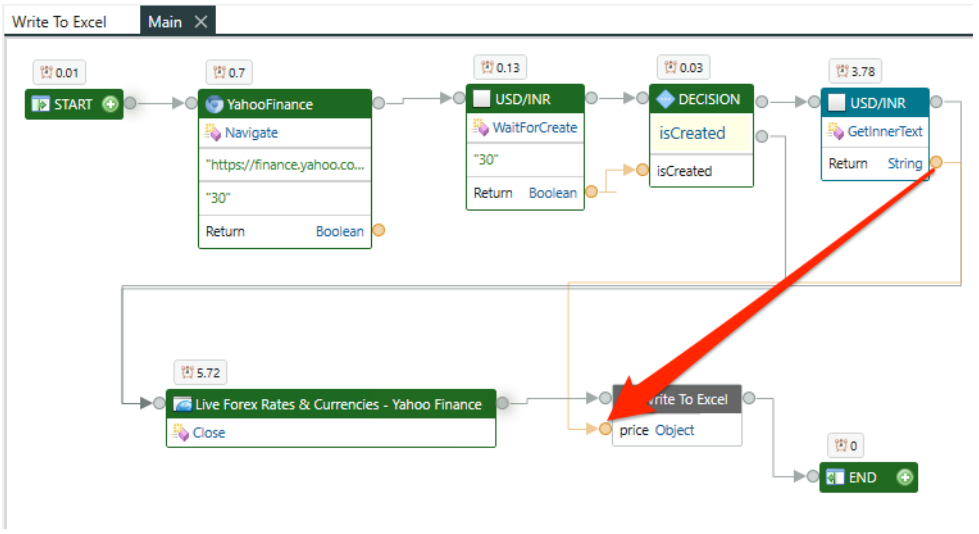
1. Go back to the **Main** diagram and drag your new **Write to Excel** activity on to the diagram in between the **Close** activity and the **END**.



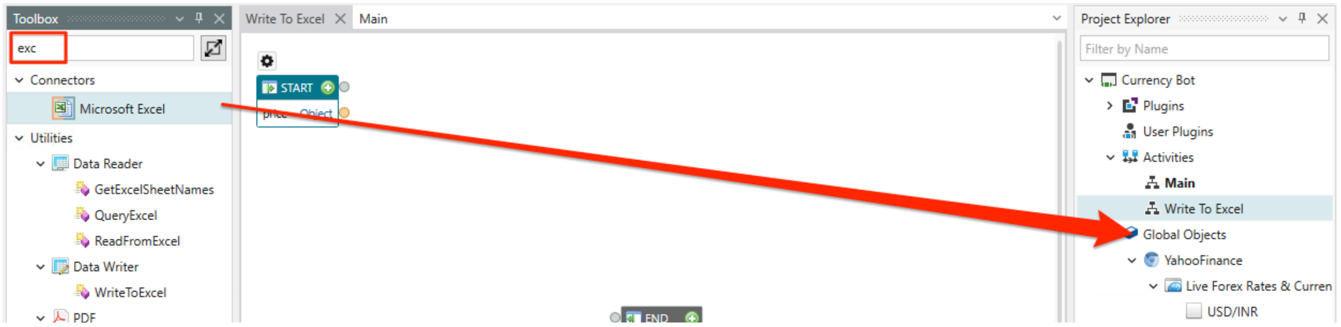
2. Back in the **Write To Excel** diagram, double click the **START** component to open the **VARIABLE CONTAINER** dialog, and add a variable called **price** using the **green plus icon,** and click **OK** to close the dialog.



3. Back in the Main diagram, connect the Return port on the **GetInnerText** component to the price variable on the **WriteToExcel** component, as shown below.

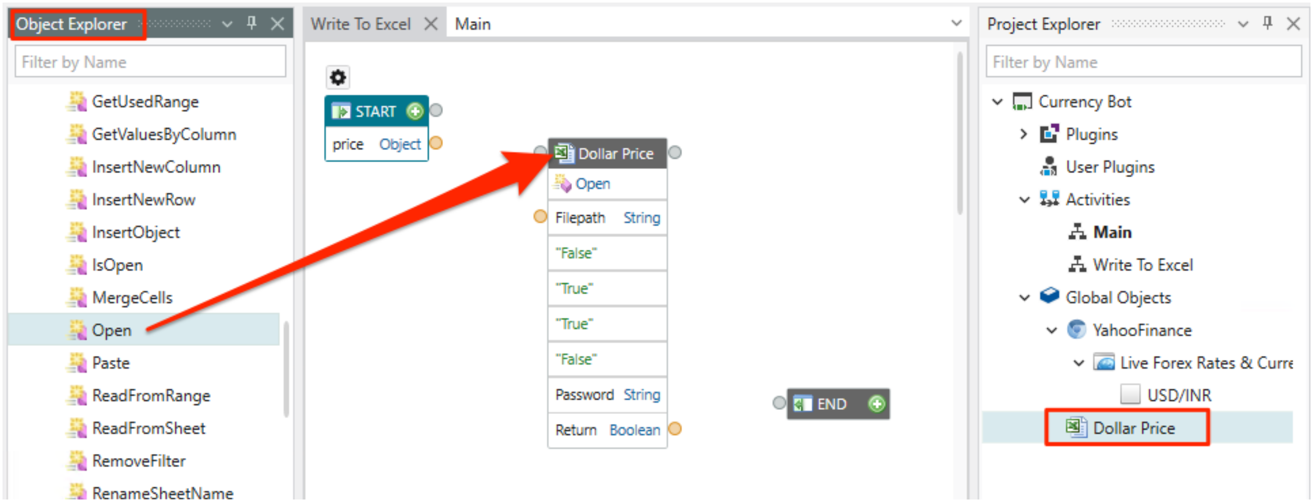


4. With the Write to Excel diagram open, go to the **Toolbox** and search for the **Microsoft Excel** connector, and drag it to **Global Objects** in the **Project Explorer**.



5. Rename your new connector under **Global Objects** as **Dollar Price**.

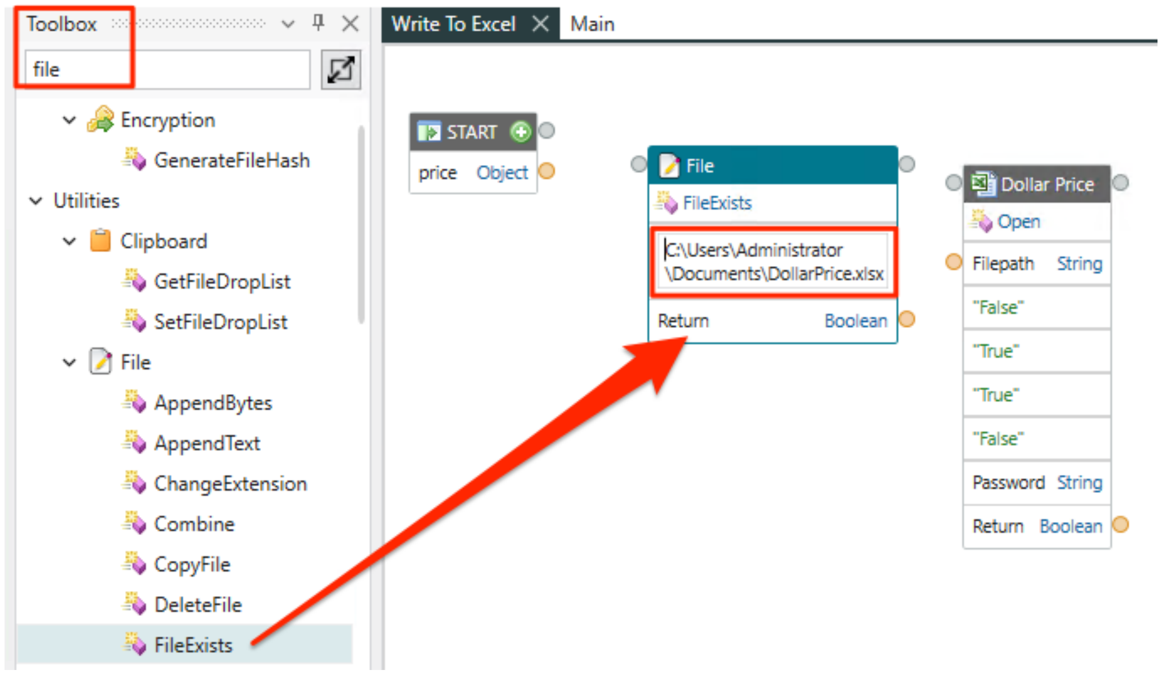
6. Double-click the **Dollar Price** connector to show the Excel methods in the **Object Explorer**, and drag the **Open** method on to the diagram.



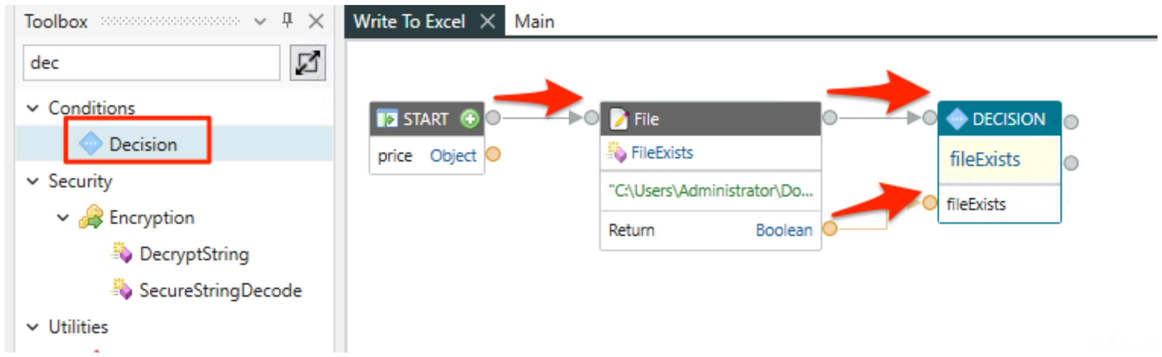
The open component has one input parameter wether to open an existing file or create a new one. To get the correct input we need to add a file check and decision component in the following steps.

7. From the **Toolbox**, drag the **FilesExists** component on to your diagram between **START** and **Open**, and give it any valid path and a file name of **DollarPrice.xlsx**

> ***Note:*** *The screenshot uses the Administrators Documents folder. If you work on your PC the username and or path might vary.*



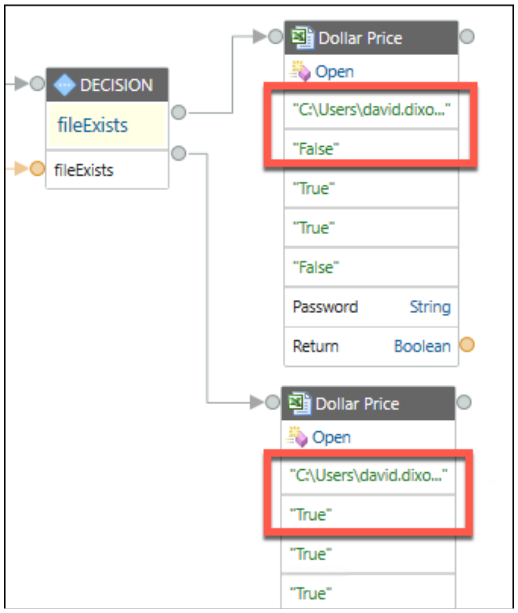
8. Add a **Decision** condition from the **Toolbox** and connect the components as shown:



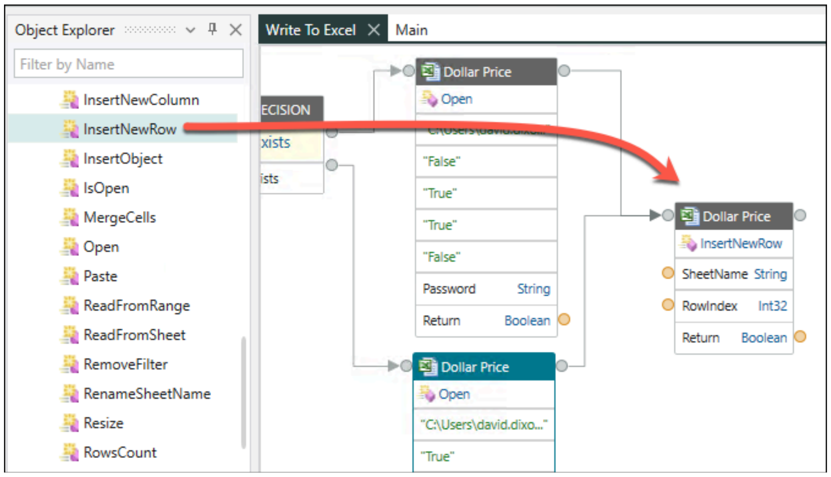
9. From **Object Explorer** create a second **Open** method and connect the two **Open** components to the two ports on the **Decision** component.

10. Set the **Filepath** properties on both **Open** components to the full path and filename of your file. Same value is on the **FileExists** component.

Set the **CreateNewFile** property on the top activity to **False**, and set it to **True** on the bottom activity. The resulting diagram should be as shown below:



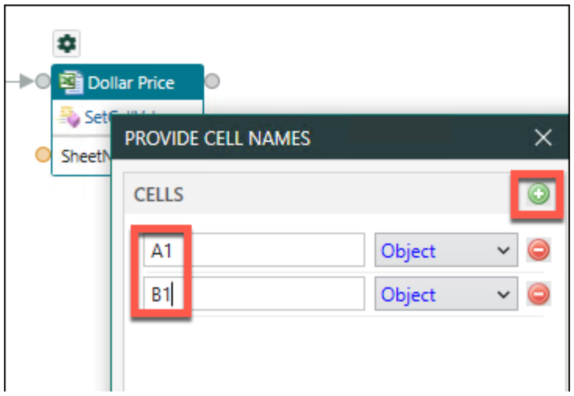
11. From **Object Explorer**, add the **InsertNewRow** method to your diagram and connect it to both **Open** activites.



12. Set the **SheetName** property on **InsertNewRow** to **Sheet1**, and the **RowIndex** property to **1. This will add a new row on top of the sheet named 'Sheet1'.**

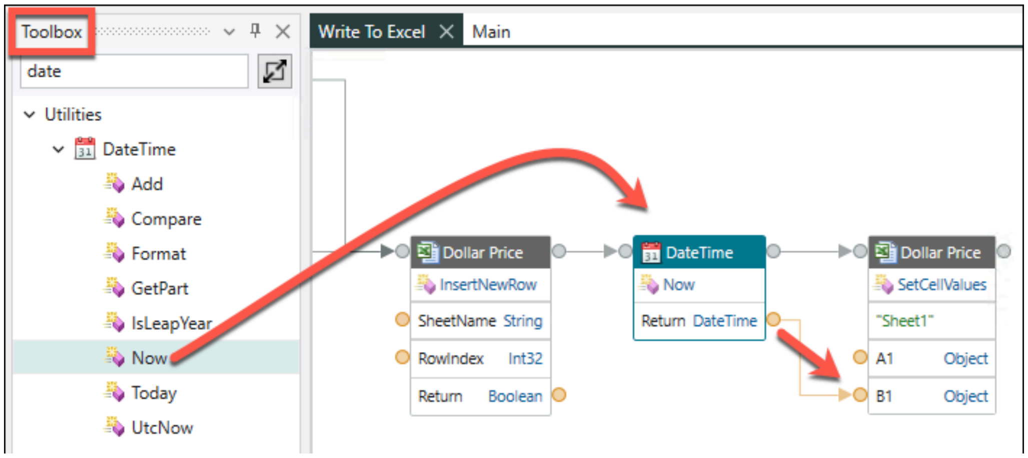
> ***Note:*** *If your Excel is not setup in English language, you will need to replace 'Sheet1' with your local equivalent. Some of the following steps will again refer to 'Sheet1', change them accordingly.*

13. Drag a **SetCellValues** method on to the diagram from the object explorer, double-click it, and in the **PROVIDE CELL NAMES** dialog, use the green plus icon to specify cells of **A1** and **B1**



14. Close the **PROVIDE CELL NAMES** dialog and specify a **SheetName** parameter of **Sheet1**.

15. From the **Toolbox**, add a **Now** DateTime Utility to the diagram after insert new row component, and connect the **Return** port to **B1** on **SetCellValues**, as shown.



Note: *If you drop the new component right on the connection – make sure it is highlighted, the gray control flow will be connected automatically.*

16. Connect the **price** object on the Start activity to **A1** on SetCellValues.

17. Add a **Close** method from the Object Explorer connected to the **SetCellValues** method and the **END** activity, so that the file will be closed.

Note: *Notice, the Close has a default parameter for save changes set to true.*

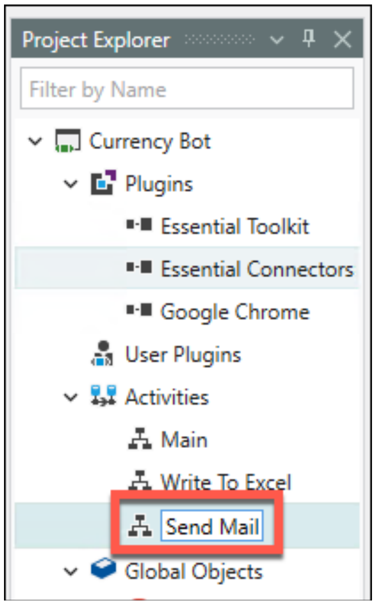
18. Clear the log, close open chrome windows if any and run your automation.

Email a File

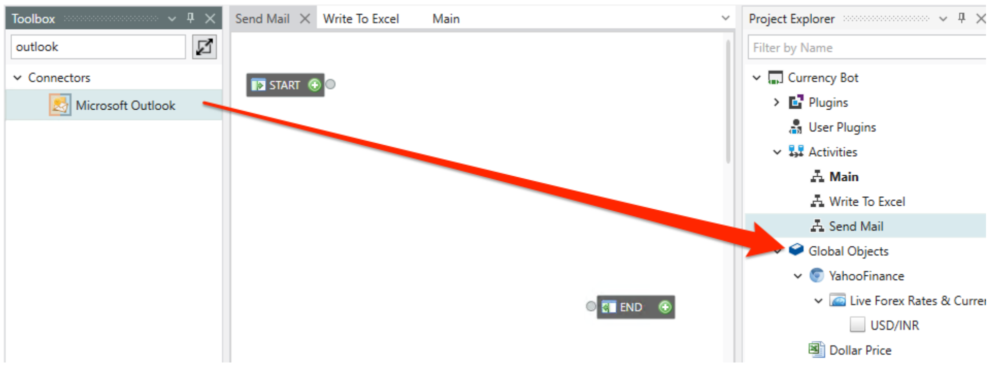
C. Build the Email Bot

Note: *This lab will only work if you can link an email account to the Outlook installed on the virtual machine or if run from your regular windows machine with Outlook already configured. If you do not want to configure your corporate account you can also use any other one like Gmail, GMX, etc. Follow their instructions on how to setup Outlook.*

0. In the **Project Explorer**, right-click **Activities** and add a new activity called **Send Mail**.

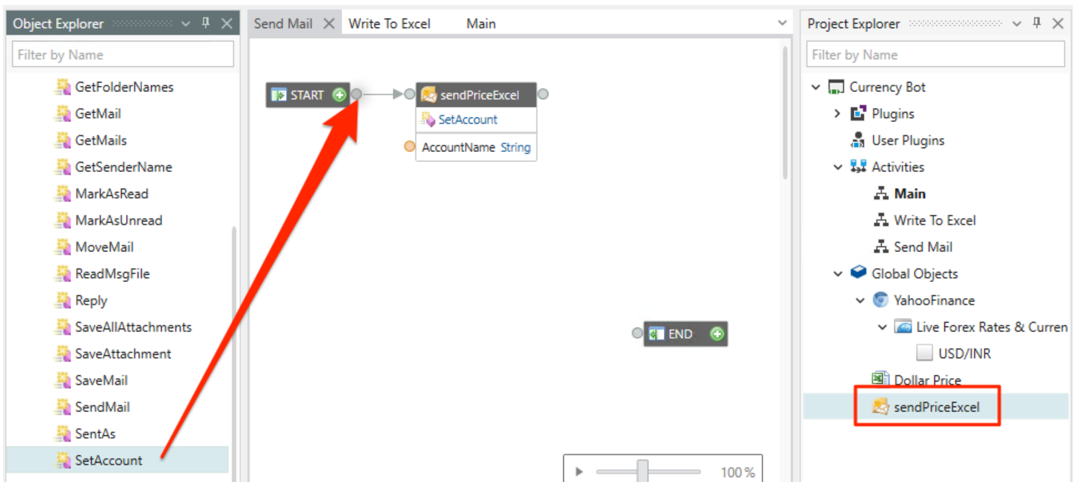


1. From the **Toolbox**, drag the **Microsoft Outlook** connector to **Global Objects** in **Project Explorer**



2. Rename the new connector **sendPriceExcel**

3. Double-click the **sendPriceExcel** connector, and from the **Object Explorer**, drag the **SetAccount** method on to your diagram, connected to the **START** component.



4. Add your email address as the **AccountName** parameter on SetAccount.

5. Drag a **SendMail** method on to your diagram connected to **SetAccount**.

6. Configure the following parameters on the SendMail method:

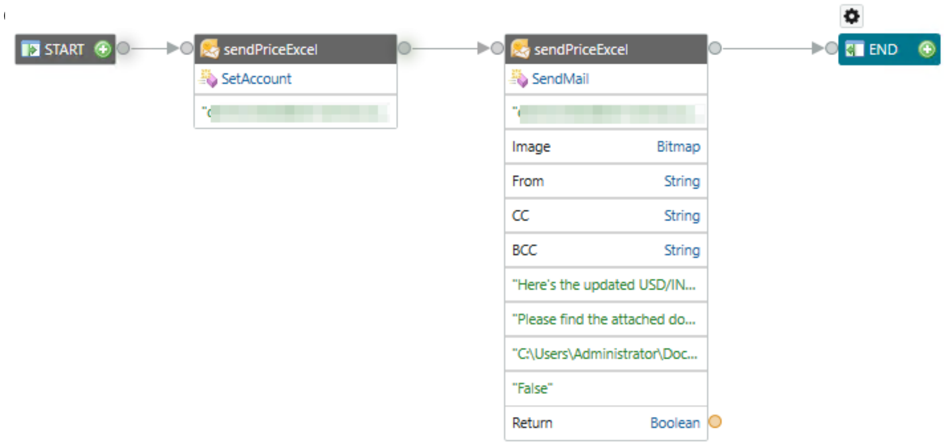
ToEmailId : < ***your email address >***

Subject : **Here's the updated USD/INR spreadsheet**

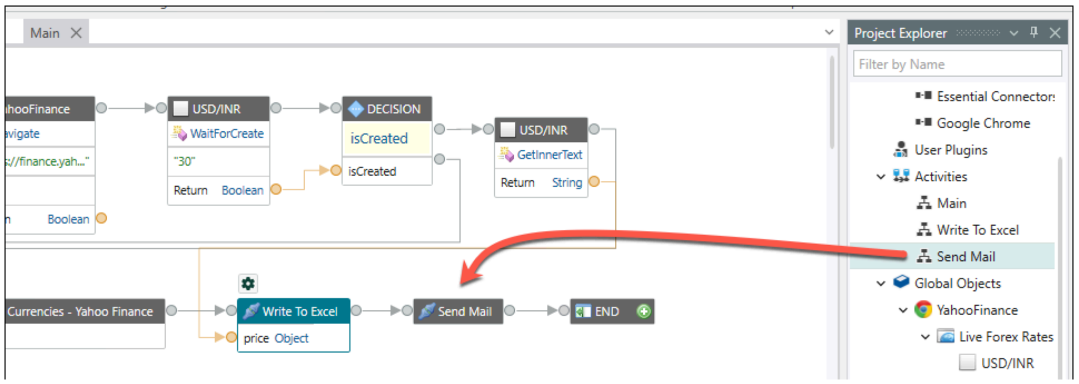
Body : **Please find the attached document**

Attachment : < ***Full path and filename of your Excel file from the previous exercise*** >

7. Connect the SendMail component to the End, you activity will look similar to this

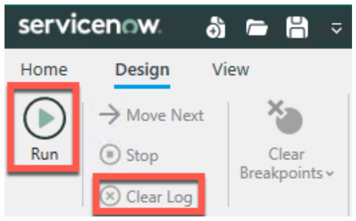


8. Back on the **Main** diagram, add the **Send Mail** activity in between **Write to Excel** and **END**.



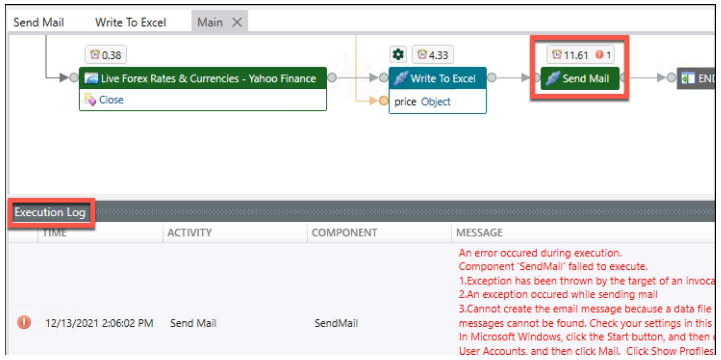
Lab Verification

0. **Clear the log** and select **Run**. You can see your robots execute and check the webpage, and you should see your Excel file get updated.



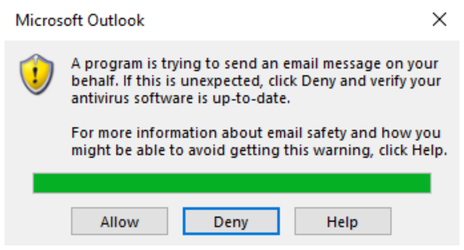
1. Each executed component on the Main diagram should turn green, as well as the activities in within Write To Excel and Send Mail.

2. Failing components are marked by a red error icon, and more information for troubleshooting is available in the Execution Log



TIME	ACTIVITY	COMPONENT	MESSAGE
12/13/2021 2:06:02 PM	Send Mail	SendMail	An error occured during execution. Component 'SendMail' failed to execute. 1.Exception has been thrown by the target of an invoca 2.An exception occured while sending mail 3.Cannot create the email message because a data file messages cannot be found. Check your settings in this In Microsoft Windows, click the Start button, and then User Accounts. and then click Mail. Click Show Profiles

3. When sending email the first time you might see a dialog from Outlook to confirm this automatic action, click Allow if


Microsoft Outlook

A program is trying to send an email message on your behalf. If this is unexpected, click Deny and verify your antivirus software is up-to-date.

For more information about email safety and how you might be able to avoid getting this warning, click Help.

Congratulations on completing the lab!