

Robotic Process Automation with Automation Anywhere





Table of Contents

1. About Automation Anywhere: 4
 2. Installing Automation Anywhere: 19
 3. Overview of Automation Anywhere Control Room: 44
 4. Overview of the Automation Anywhere Development Interface: 69
 5. Building Your First Bot: 120
 6. Introducing Variables: 150
 7. Interacting with Applications: 187
 8. String Manipulation and List Variables: 237
 9. Working with Conditional Logic, Loops, and the Filesystem: 287
- 



Table of Contents

- 10: Working with XML Files: 323
 - 11: Automating Excel: 385
 - 12: Automation Using Word: 425
 - 13: Working with Emails: 455
 - 14: Working with PDF Files: 503
 - 15: Working with Databases: 543
 - 16: Building Modular Bots and Sub-Tasks: 591
 - 17: Running External Scripts: 647
 - 18: Managing Errors: 687
- 

About Automation Anywhere



About Automation Anywhere

We will cover the following topics in this lesson:

- What is robotic process automation?
- Overview of Automation Anywhere
- Automation Anywhere Versions
- Community Edition

Technical requirements

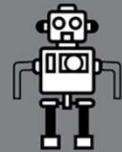
In order to use AA Community Edition, the following requirements are necessary:

- Windows OS version 7 or higher
- A processor with a minimum speed of 3 GHz
- A minimum of 4 GB RAM
- Internet Explorer v10 or higher, or Chrome v49 or higher
- An internet connection with a minimum speed of 10 Mb/second

What is robotic process automation?

- You probably already know what RPA is, but we will go through a quick overview here.
- The words automation or robot usually conjure up images of a physical machine performing repetitive tasks. We began to see this type of automation years ago, particularly in manufacturing. Physical robotic machines were built to help automate tasks usually done by humans.
- This form of industrial manufacturing automation was later adopted by many other industries including logistics, distribution, and packaging.

What is robotic process automation?

RPA	TRADITIONAL AUTOMATION
	
Requires no modifications in the current IT infrastructure	Requires certain customisations in current IT infrastructure
Lower costs of implementation when RPA is adopted	An option for companies with a flexible budget
A more efficient option since it can make improvements instantly	Requires more time, effort and considerable manpower
Eliminate the need to change the current processes in the system	Have to change the processes in the current system



Opening email
and attachments



Logging into web/
enterprise applications



Reading and writing
to databases



Copying and pasting



Filling in forms



Moving files and folders



Following "if/then"
decisions/rules



Collecting social
media statistics



Extracting structured
data from documents



Making calculations



Connecting to
system APIs



Scraping data from
the web

Overview of Automation Anywhere

	BluePrism	UiPath	Automation Anywhere
Free Community Edition	No	Yes	Yes
Front Office Development	No	Yes	Yes
Back Office Development	Yes	Yes	Yes
User Friendly Interface	Yes	Yes	Yes
Drag & Drop Development Feature	Yes	Yes	Yes
Interface Recorder Development Feature	No	Yes	Yes
Certification Available	Yes	Yes	Yes
Training Academy	No	Yes	Yes
Cloud Based Development	No	Yes	Yes
Bot Store Marketplace	No	Yes	Yes

The Digital Workforce

- A bot is referred to by AA as a Digital Worker as it clones the actions of a human to perform a given task. A Digital Worker is a member of the team designed to carry out a process just the same as any human worker.
- In a working environment, a team can consist of both humans and bots, hence the bot being referred to as a Digital Worker. As more bots are built within an organization, you can see a Digital Workforce being created.
- These bots can work side by side with a human or can be deployed to run on their own.

IQ Bot

- As well as utilizing condition-based decisions, more and more processes require a certain level of cognitive intelligence to make decisions.
- An example of this would be when dealing with unstructured data, A common scenario would be based on invoices, they all tend to have the same type of data such as supplier, items, costings, and dates, but the layout and format will vary between different suppliers.
- The consistency is not present when handling multiple suppliers. AA has developed a product called IQ Bot.
- This bot uses cognitive automation with RPA to learn how to handle unstructured data.

Bot Insight

- Designing and building bots is not the complete story, AA has also developed a platform that produces real-time analytics about your Digital Workforce, processes, and business-level processes, This is all a part of the Bot Insights tool, the RPA analytics tool for AA.
- Bot Insights is broken down into two categories: operational analytics and business intelligence.
- As bots are deployed, as well as executing tasks, they also process data, This data is related to each specific process and can provide valuable insight.
- Bot Insight analyzes this data and transforms it into meaningful insights.

Bot Store

- AA is the first RPA vendor to have a fully operational Bot Store.
- Bot Store is an online store with a collection of Digital Workers, The bots available here are built by independent developers from all around the world as well as AA themselves.
- AA Bot Store won the Silver award in the 2019 Edison Awards for developing the world's first and largest enterprise automation marketplace.
- These are complete bots out of the box that will perform a specific task or role.
- They are available as bots for specific applications, categories, or business processes.

Mobile Bot

- AA has also released a mobile app to work with your bots.
- It allows you to manage your Digital Workers from your mobile device.
- Bot Insight is available on the mobile app.
- This app will give you live alerts on bot performance as well as business insights on bot data.
- You can control your bots from the app including starting and stopping them.
- It also provides a platform for you to connect with the wider AA RPA community.

Automation Anywhere versions

ENTERPRISE VERSION 11	ENTERPRISE A2019	COMMUNITY EDITION A2019
Designed as an on-premises enterprise level RPA platform	Designed as a cloud-based enterprise level RPA platform	Designed for the developer and student
Free trial for 30 days	Free trial for 30 days	Free for small businesses, developers, and students
Included components: Enterprise – RPA on-premises IQ Bot, Bot Insight, Bot Store, Mobile App	Included components: Enterprise – fully web-based RPA in the cloud IQ Bot, Bot Insight, Bot Store, Mobile App	Included components: Enterprise – fully web-based RPA in the cloud IQ Bot, Bot Insight, Bot Store

Community Edition

- AA Community Edition is the latest free version available.
- The version prior, AA v11.x, used a client-server architecture where the management was done through the web-based Control Room app while the bot development was done through a client application installed on the desktop.
- Community Version is a fully cloud-based solution.
- The bot management and building are all done through the web application.
- No development client is installed on your desktop.

Summary

- You will now have a good understanding of AA and its competitors as well as what AA's capabilities are.
- Having registered with AA to use the free Community Edition , you must be keen to get AA up and running on your machine.
- In the next lesson, you will be guided through the installation process of AA.

Installing Automation Anywhere



Installing Automation Anywhere

We will cover the following topics in this lesson:

- Connecting to Control Room
- Preparing your device
- Configuring profile and device credentials

Technical requirements

In order to install the Automation Anywhere Community Edition Bot agent, the following requirements are necessary:

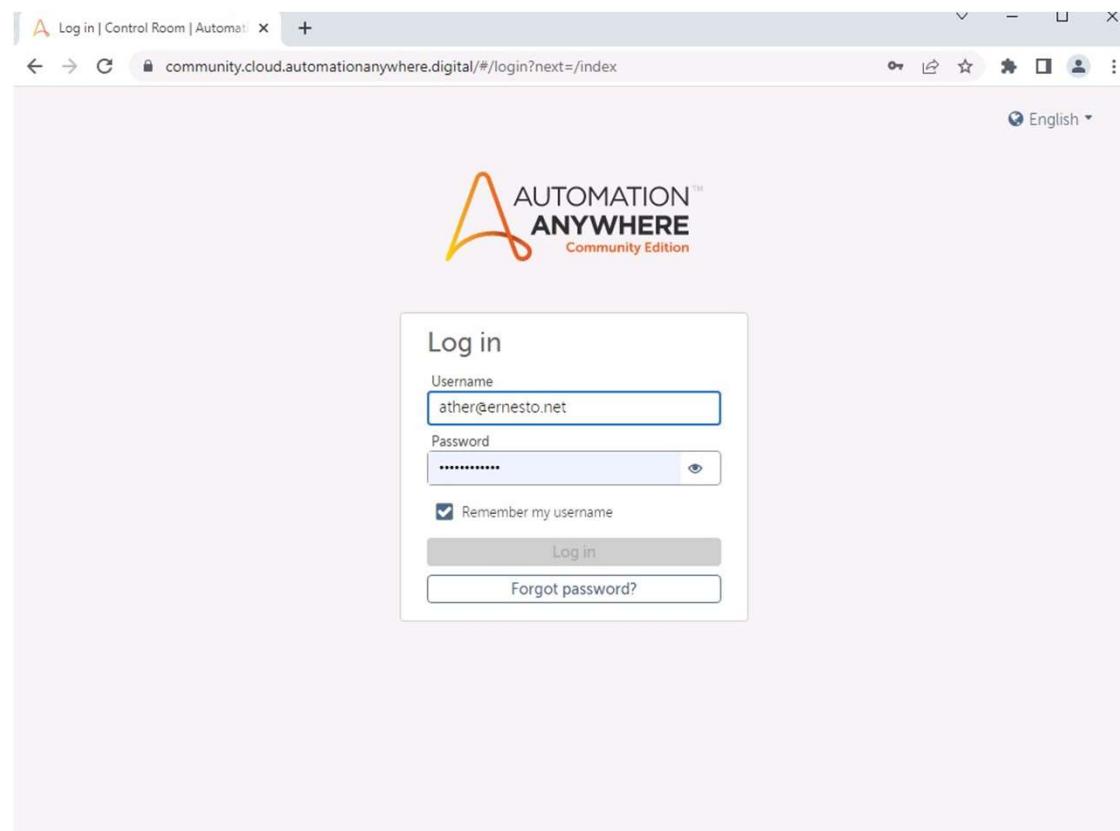
- Windows OS version 7 or higher
- A processor with a minimum speed of 3 GHz
- A minimum of 4 GB RAM
- At least 100 MB hard disk space
- Internet Explorer v10 or higher, or Chrome v49 or higher
- A minimum screen resolution of 1024*768
- An internet connection with a minimum speed of 10 Mb/second
- Have completed the registration process with Automation Anywhere for Community Edition AA 2019

Connecting to Control Room

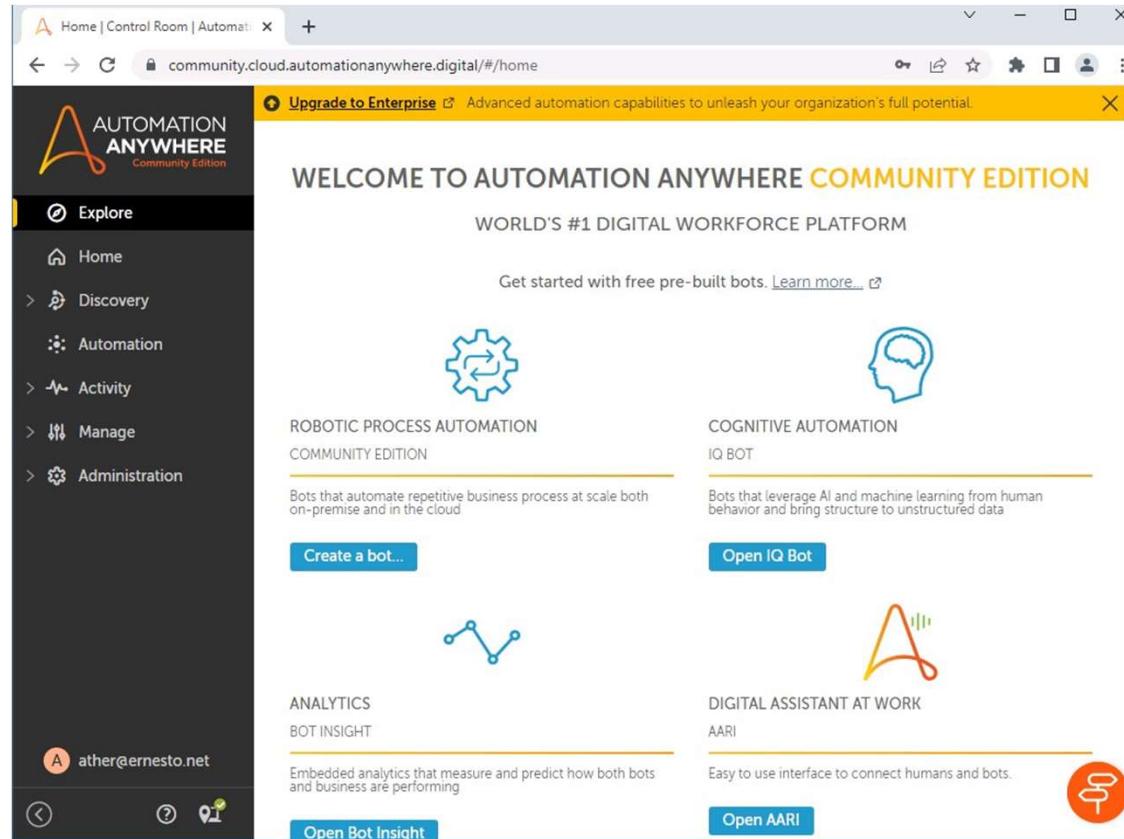
Continuing from lesson 1, About Automation Anywhere, after your registration with Automation Anywhere is complete, you should have received an email with the following details:

- Your Control Room URL
- Your username
- Your password

Launching and logging into Control Room

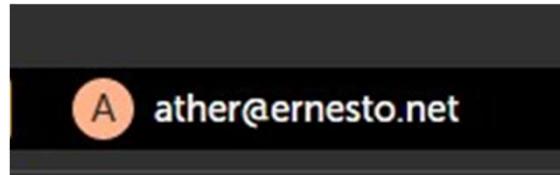


Launching and logging into Control Room

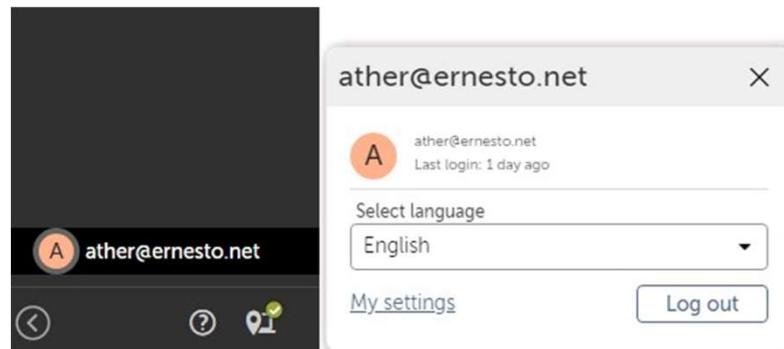


Updating your profile and password

- In the top right-hand corner of the Home page, click on your profile icon:



- From the dialog box, click on Go to My settings:



Updating your profile and password

- Will take you to your profile details interface; to make any changes to your profile, click on Edit:

The screenshot shows the 'My Settings' interface. At the top, there's a user icon (an orange circle with a white letter 'A') and the email address 'ather@ernesto.net'. To the right, there's a language dropdown set to 'English'. Below this, the 'My profile' section is expanded, showing 'General details' and 'Contact info'. Under 'General details', 'First name' and 'Last name' both show 'N/A'. There's an 'Edit...' button next to 'Last name'. Under 'Contact info', the 'Email' field shows 'ather@ernesto.net'. In the 'Change password' section, there's a 'Password' field with five asterisks. At the bottom of the profile section, it says 'Modified by' followed by the user's name and email again. To the right, it shows 'Last login' as '1 day ago' and 'Last modified' as '1 day ago'. Below the profile section, there's a partially visible 'Devices' section.

Updating your profile and password

- Will allow you to update your details.
- Here, you can update your password; once you have made your changes, click on Save changes to apply them:

My Settings

ather@ernesto.net

English ▾

My profile

General details

First name (optional) Last name (optional)
Maximum = 50 characters Maximum = 50 characters

Contact info

Contact email Confirm contact email
ather@ernesto.net ather@ernesto.net
Maximum = 255 characters

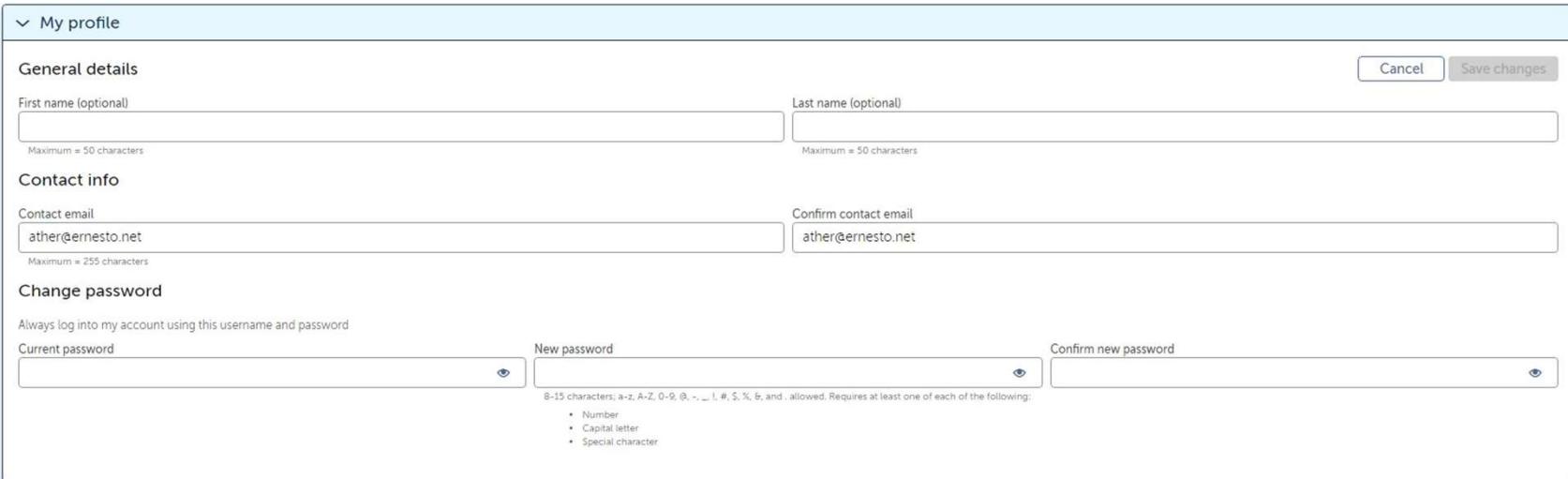
Change password

Always log into my account using this username and password

Current password New password Confirm new password
8-15 characters, a-z, A-Z, 0-9, @, ., _, #, \$, %, &, and . allowed. Requires at least one of each of the following:

- Number
- Capital letter
- Special character

Cancel Save changes



Preparing your device

To add a device, two stages are involved:

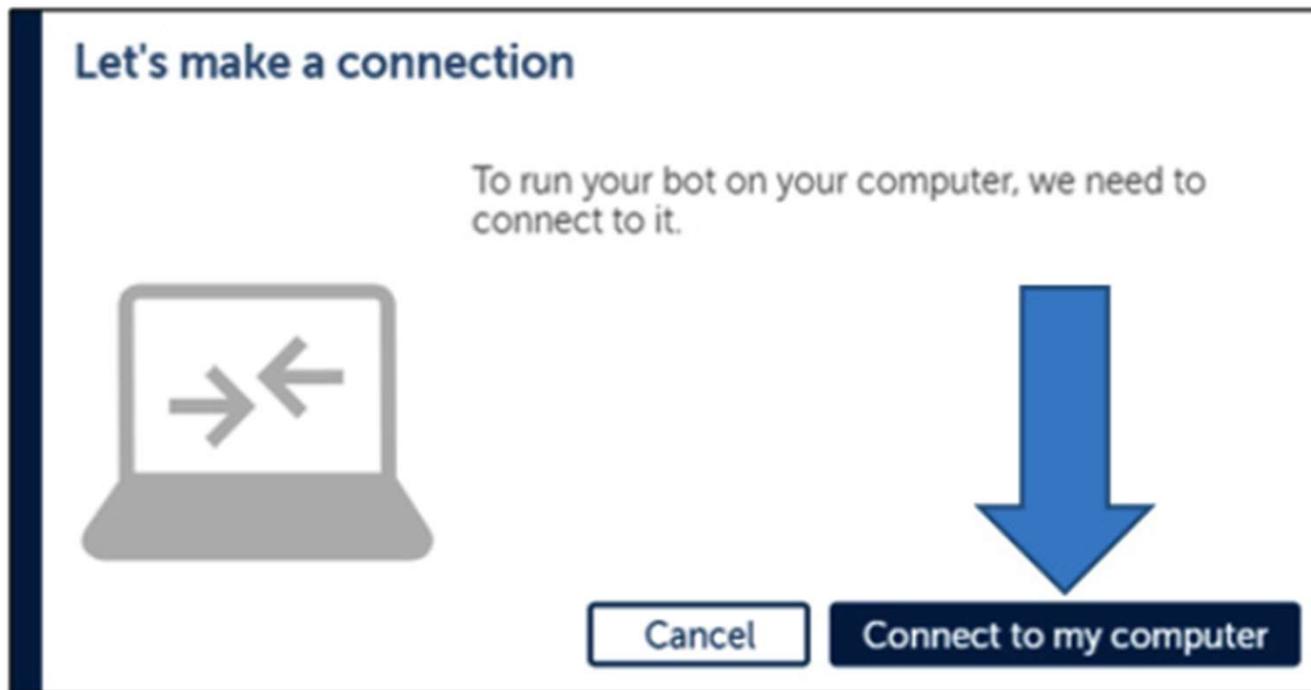
- Installing a Bot agent
- Enabling the extension

Installing a Bot agent

The screenshot shows the Automation Anywhere Community Edition web interface. The left sidebar menu includes Home, Discovery, Processes, Opportunities, Automation, Activity, Manage (with Learning Instances), Devices (which is selected and highlighted in yellow), Global values, Credentials, Packages, and Administration. The main content area is titled 'Devices' and shows a table with one device entry. The table columns are Status, Device name, Default users, Device nickname, and Lifespan. A yellow banner at the top right encourages upgrading to Enterprise with the text: 'Upgrade to Enterprise Advanced automation capabilities to unleash your organization's full potential.' Below the banner are buttons for 'Run bot now...' and 'Connect local device...'. A search bar at the top allows filtering by 'Device name'.

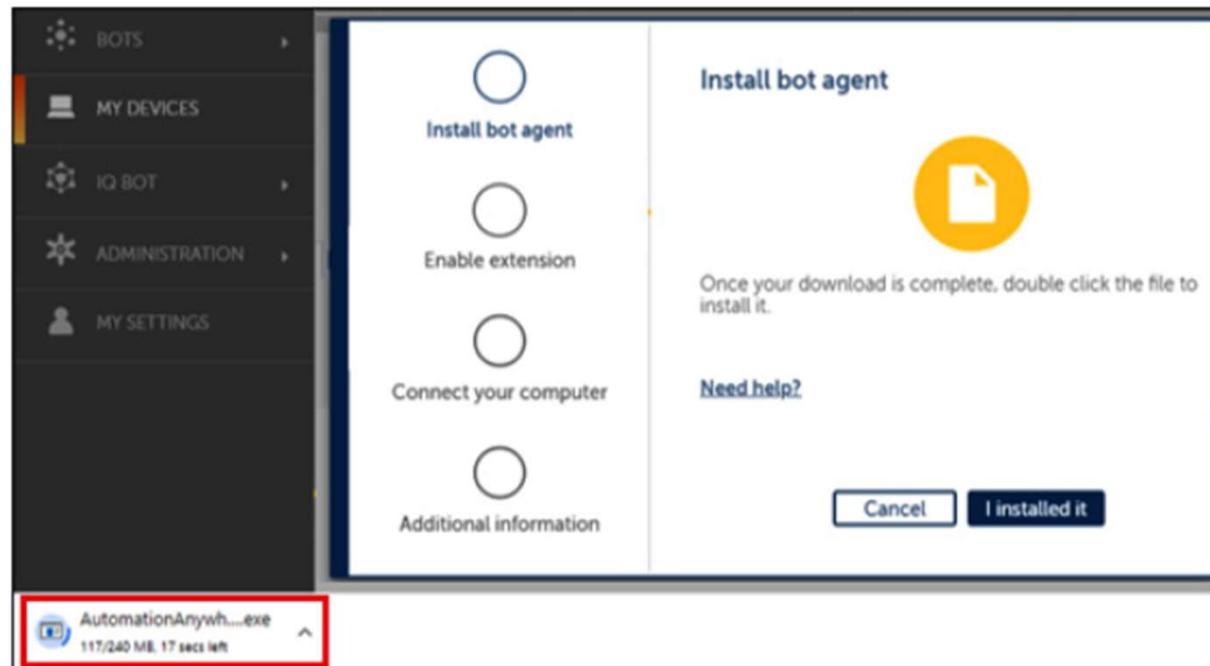
Installing a Bot agent

The connection wizard will pop up. Click on **Connect to my computer**:



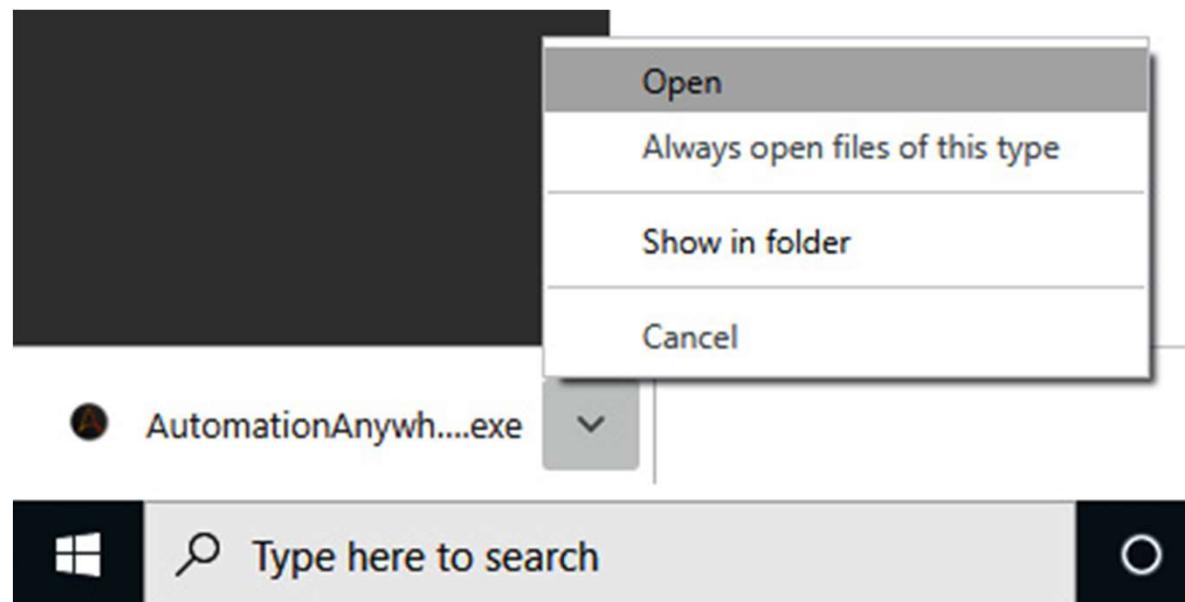
Installing a Bot agent

The wizard will start to download the Bot agent (into your default downloads folder) as shown in the following screenshot:



Installing a Bot agent

- Once the Bot agent has been downloaded, right-click on the downloaded file's icon and select Open:



Installing a Bot agent



Install bot agent



Enable extension



Connect your computer



Additional information

Enable Chrome extension

This will let bots run on this computer without device credentials. Without the extension a username and password will be required.

[Need help?](#)

Cancel

I enabled it

Enabling the extension



Install bot agent



Enable extension



Connect your computer



Additional information

Chrome extension not enabled

It might help if you get the extension from the Chrome store:

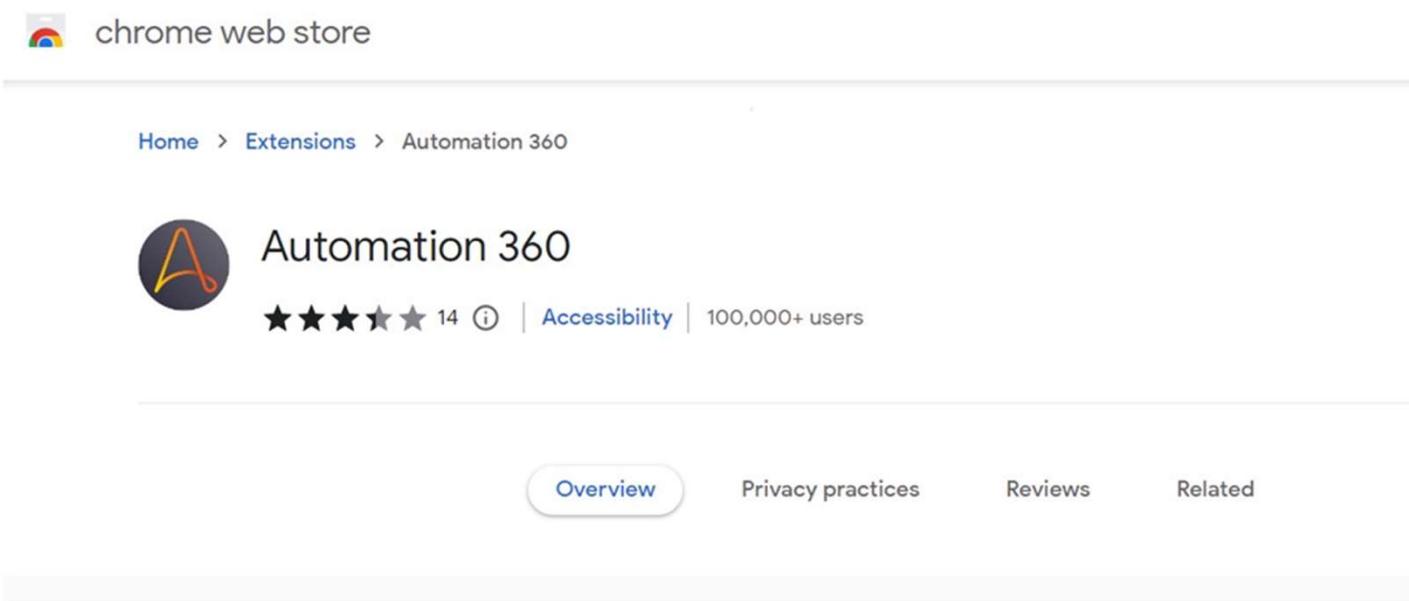
1. [Get the Automation Anywhere extension.](#)
2. Click "Add extension" to add it and enable it.
3. Once the extension is installed and enabled, we will automatically move to the next step.

[Skip this step.](#) I will add my username and password manually.

Cancel

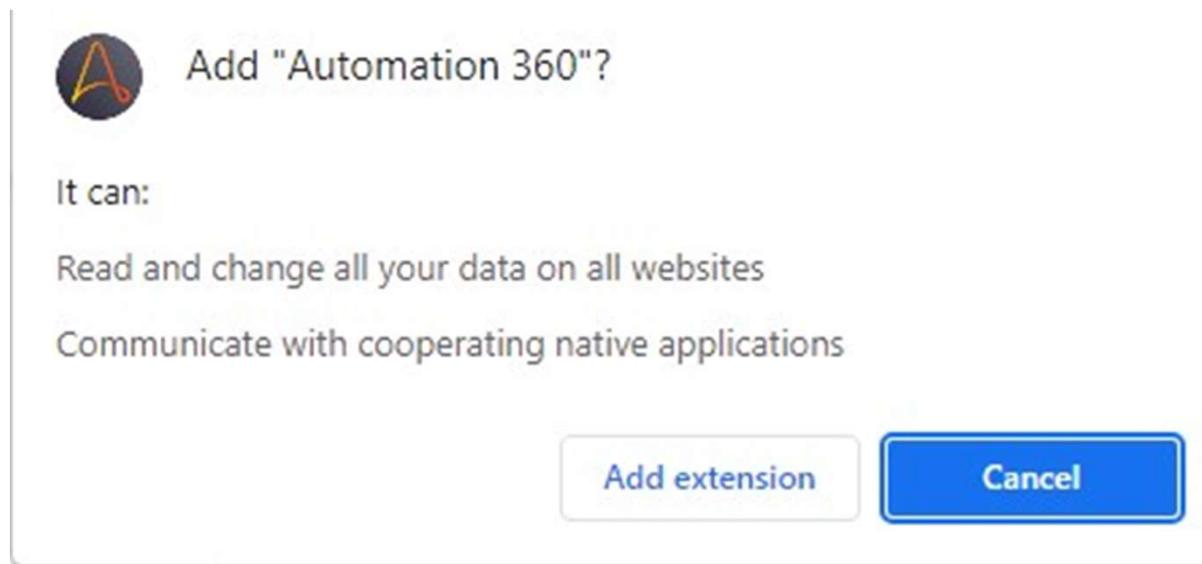
Enabling the extension

- Click on the Get the Automation Anywhere extension link to navigate to the Chrome Web Store:

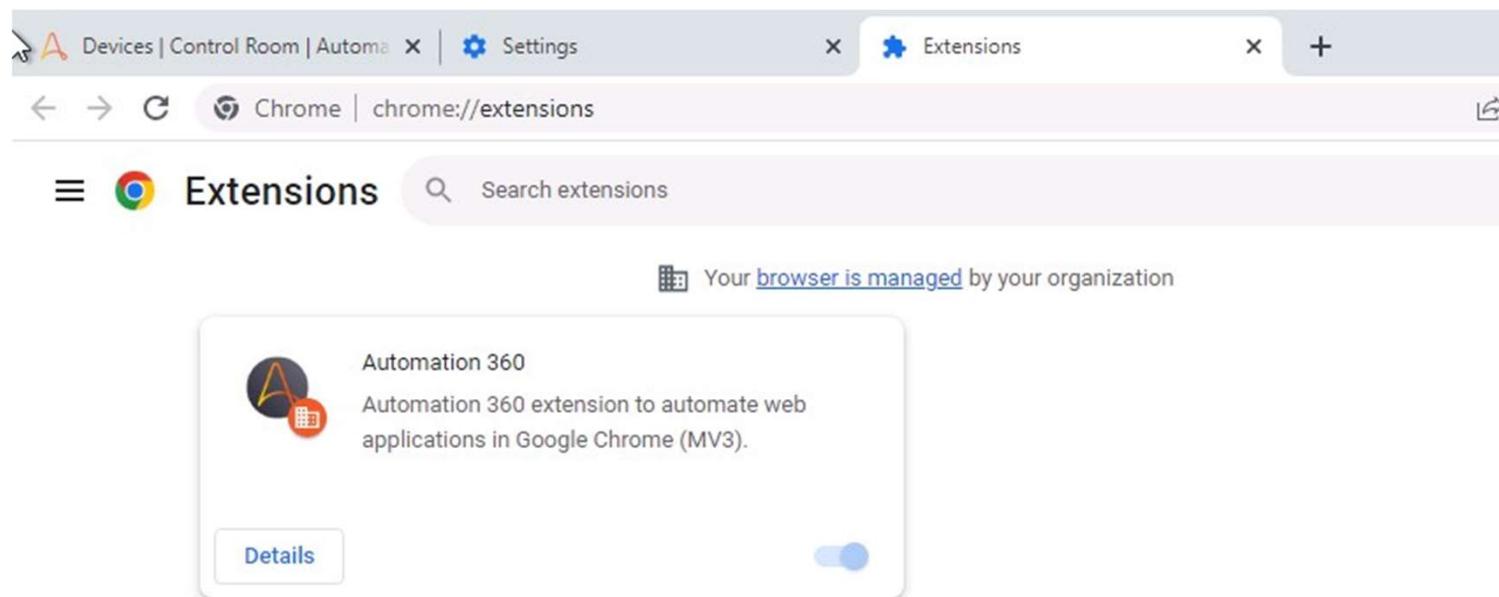


Enabling the extension

- Click on Add to Chrome button and you will get the following prompt; click on Add extension:



Enabling the extension



Enabling the extension

- Once the extension has been enabled, the progress indicator will be updated with further green ticks, shown as follows:



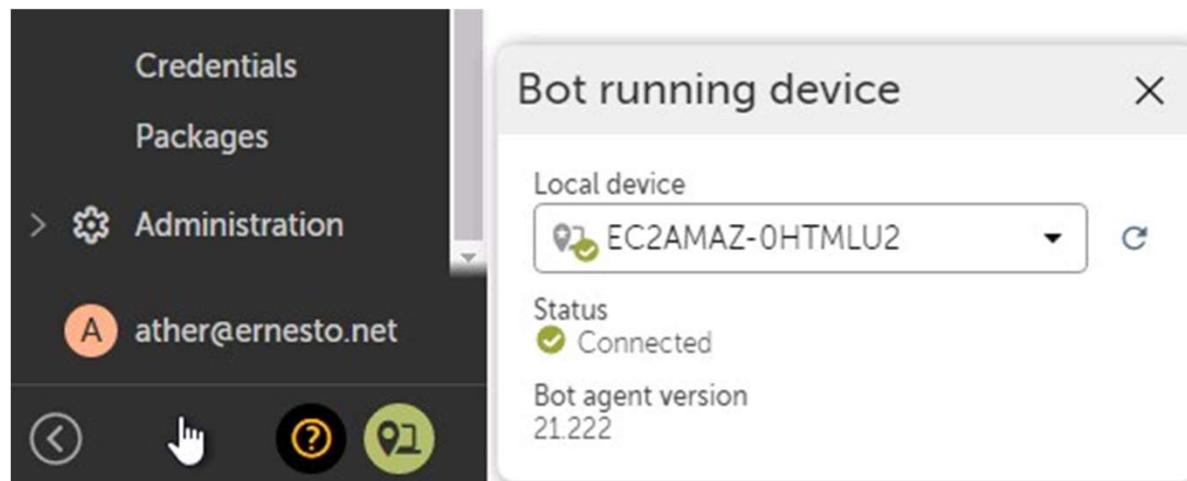
Enabling the extension

The screenshot shows the Automation Anywhere Community Edition web interface. The left sidebar has a dark theme with white icons and text. It includes links for Home, Discovery, Processes, Opportunities, Automation, Activity, Manage (with Learning Instances), and Devices, where 'Devices' is currently selected. The main content area has a yellow header bar with an 'Upgrade to Enterprise' button and a message about advanced automation capabilities. Below this is a section titled 'Devices' with a search bar and a 'Run bot now...' button. A table lists one device entry:

Status	Device name	Default users	Device nickname	Lifespan
Connected	EC2AMAZ-0HTMLU2	ather@ernesto.net	--	Persis :

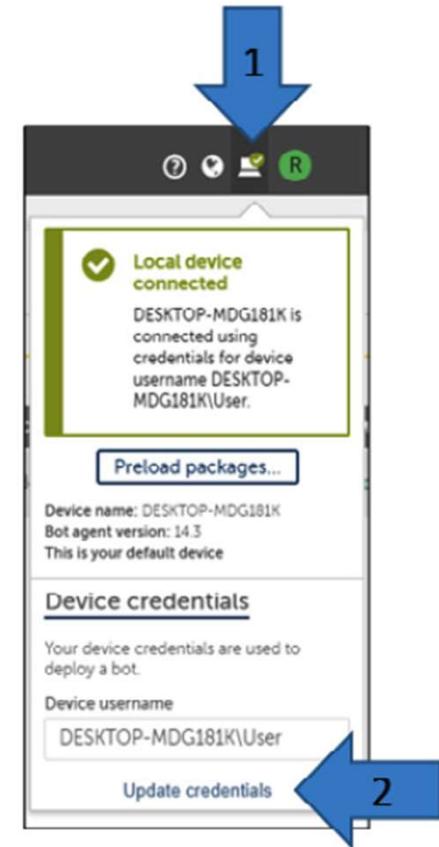
Enabling the extension

- The Bot agent should now be successfully installed on your device.
- You will also notice a little green tick against your local device icon on the top pane:



Configuring profile and device credentials

- Once in Control Room, from the Home screen, click on the local device icon, followed by Update credentials



Configuring profile and device credentials

- The dialog will allow you to update the Device password.
- Enter the user password for the device and click on Update

Device credentials

Your device credentials are used to deploy a bot.

Device username
DESKTOP-MDG181K\User 

Max characters = 255

Device password 

Device password is used for remote login.

Summary

- We are all ready to get going now.
- You will be comfortable in the future with setting up and configuring a device with Automation Anywhere Control Room, and can configure your device credentials as well as update your user profile.
- You could have a number of devices installed in Control Room.
- Once a bot is built, it can be deployed to any one of them or even a number of devices.
- Installing and configuring devices is an essential part of deploying and testing bots, and Control Room provides a centralized location to manage all your devices easily.

Overview of Automation Anywhere Control Room



Automation Anywhere Control Room

We will cover the following sections of Control Room in this lesson:

- Exploring the home screen
- Understanding the dashboard
- Viewing RPA activity
- Managing bots
- Managing My Devices
- Managing user administration

Technical requirements

- Windows operating system version 7 or higher
- A processor with a minimum speed of 3 GHz
- Minimum of 4 GB RAM
- At least 100 MB hard disk space
- Internet Explorer v10 or higher OR Chrome v49 or higher
- A minimum screen resolution of 1024*768
- An internet connection with a minimum speed of 10 Mb/sec
- Completed registration with Automation Anywhere Community Edition
- Logged on successfully to Automation Anywhere Community Edition
- A successfully registered local device

Exploring the home screen

- You will notice shortcuts to some of the exciting features available for you to explore:

The screenshot shows the home screen of the Automation Anywhere Community Edition web interface. The left sidebar has a dark theme with white text and icons. It includes sections for Explore (Home, Discovery, Processes, Opportunities, Automation, Activity), Manage (Learning Instances, Devices, Global values, Credentials, Packages), and Administration. A user profile for 'ather@ernesto.net' is at the bottom. The main content area has a light blue header with a yellow bar containing an 'Upgrade to Enterprise' button. Below this is a yellow banner with the text 'WELCOME TO AUTOMATION ANYWHERE COMMUNITY EDITION' and 'WORLD'S #1 DIGITAL WORKFORCE PLATFORM'. It features five main service cards: 'ROBOTIC PROCESS AUTOMATION COMMUNITY EDITION' with a gear icon and 'Create a bot...', 'COGNITIVE AUTOMATION IQ BOT' with a brain icon and 'Open IQ Bot', 'ANALYTICS BOT INSIGHT' with a line graph icon and 'Open Bot Insight', 'DIGITAL ASSISTANT AT WORK AARI' with a stylized 'A' icon and 'Open AARI', and 'PROCESS DISCOVERY DISCOVERY BOT' with a gear icon and 'Open Discovery Bot'. Each card has a brief description below it.

Exploring the home screen

- There are a number of options and information available in the top-right panel, These are as follows:



Access further documentation and help



Set your language preference for the application



View/Set local bot agent and device credentials



rpa_training@skysoftuk.net

View/Set user profile

Understanding the dashboard

The screenshot shows the Automation Anywhere Community Edition dashboard. The left sidebar includes links for Explore, Home (which is selected), Discovery, Automation, Activity, Manage, and Administration. The main content area features three main sections: 'Getting started' (with a 'Create a bot' button), 'Recently visited pages' (listing Home, My Settings, Devices, Packages, and Test | Edit Task Bot), and 'Insights' (describing Bot Insight). Below these are four metrics: '# of Task Bots created' (1), 'Most used actions' (represented by a yellow circle), '# of Task Bots run' (1), and 'Average time spent to create a Task Bot (across all users)' (15 min).

Understanding the dashboard

The available shortcuts on the dashboard are as follows:

- Creating a new bot
- Launching bot insights
- Your recently visited pages

Viewing RPA activity

The screenshot shows the Automation Anywhere web interface with the title bar "In progress activity | Control Room". The URL in the address bar is "community.cloud.automationanywhere.digital/#/activity/inprogress". A yellow banner at the top right says "Upgrade to Enterprise" with a link to "Advanced automation capabilities to unleash your organization's full potential".

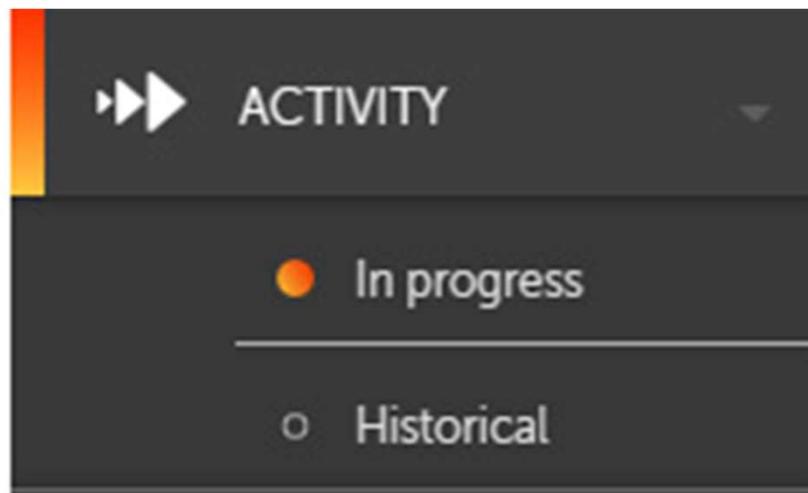
The left sidebar has a dark theme with the following navigation items:

- AUTOMATION ANYWHERE Community Edition
- Explore
- Home
- Discovery
- Automation
- Activity
 - In progress
 - Historical
 - Insights
- Manage
- Administration

The main content area is titled "In progress activity". It features a search bar with "Status" dropdown and "Choose" button. Below is a table header for "Activity (0)" with columns: Status, Item name, Automation priority, Progress, Current action, Bot, Activity type, Started on, Device, and Username. A large circular button with two right-pointing arrows is centered. Below it is the text: "When there is activity, it will automatically appear here."

Viewing RPA activity

- There are two sub-sections to the Activity page, showing In progress and Historical:



Viewing RPA activity

- A number of options are available through the icons on the right, just above the activity list:



Export List to CSV file



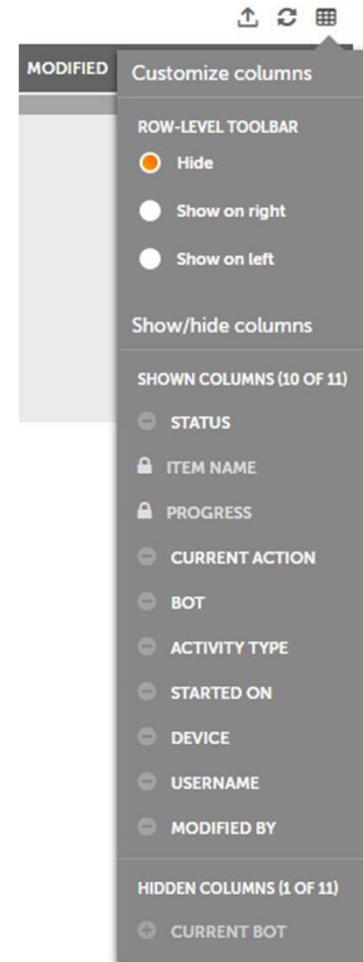
Refresh List



View/Hide Columns in List

Viewing RPA activity

- You can see all the columns that are available.
- Each one can be set to be visible or hidden:



Viewing RPA activity

In progress activity

Status ▼ Choose

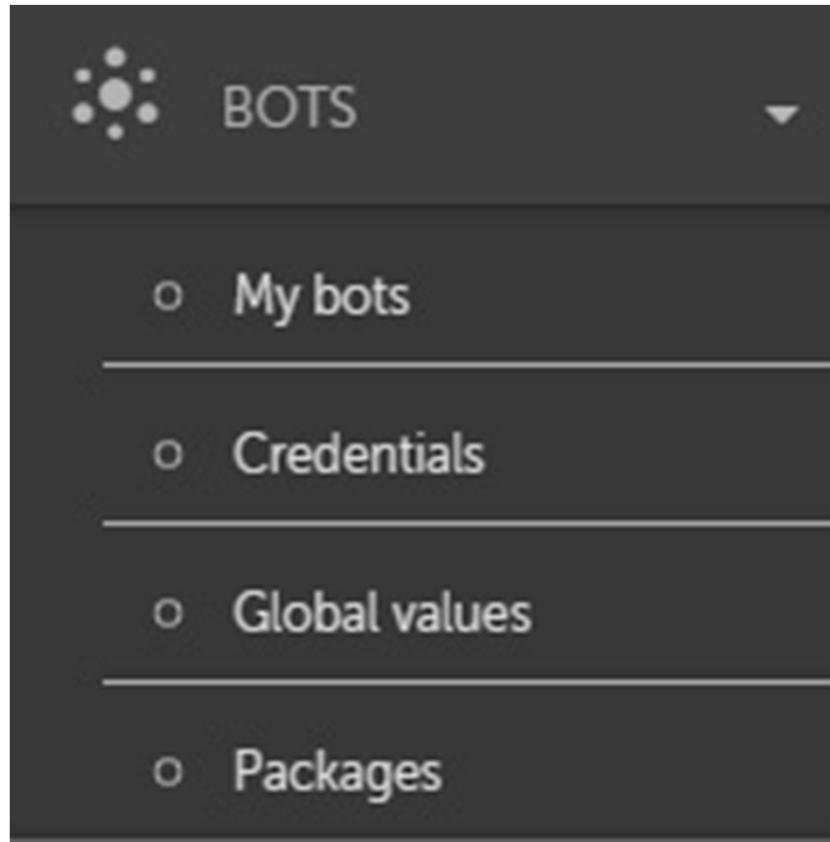
Status

- Current action
- Current bot
- Bot
- Activity type
- Device

NAME	CURRENT ACTION	PROGRESS
------	----------------	----------

Managing bots

- Bot management and deployment are performed in this section.
- This is broken down into four sub-sections, which are accessible from the main menu pane on the left:



My bots

- The folder structure is displayed on the left pane and the contents on the right:

The screenshot shows the 'Control Room' interface with the title 'Control Room' at the top. On the right side of the top bar are icons for help, user profile, and email (rpa_training@skysoftuk.net). Below the title, the navigation path 'Bots > My bots' is shown. The main area is titled 'My bots' with a 'Create a bot...' button. On the left, there's a tree view under 'Folders' showing a single folder named 'Bots'. The right pane displays a table titled 'Files and folders (1)' with one item: 'Sample_bots'. The table has columns for Name, Status, Size, Last Modified, and Modified By. The 'Modified By' column shows 'N/A'. The 'Last Modified' column shows '12:23:57 GMT 2020-02-14'. A search bar at the top of the right pane is set to 'Name'.

Name	Status	Size	Last Modified	Modified By
Sample_bots	N/A	N/A	12:23:57 GMT 2020-02-14	N/A

My bots

- There are a few options available for each file and folder within this interface, as shown in the following screenshot:



Create new bot



Delete checked item



Create sub-folder



Refresh list



Upload files



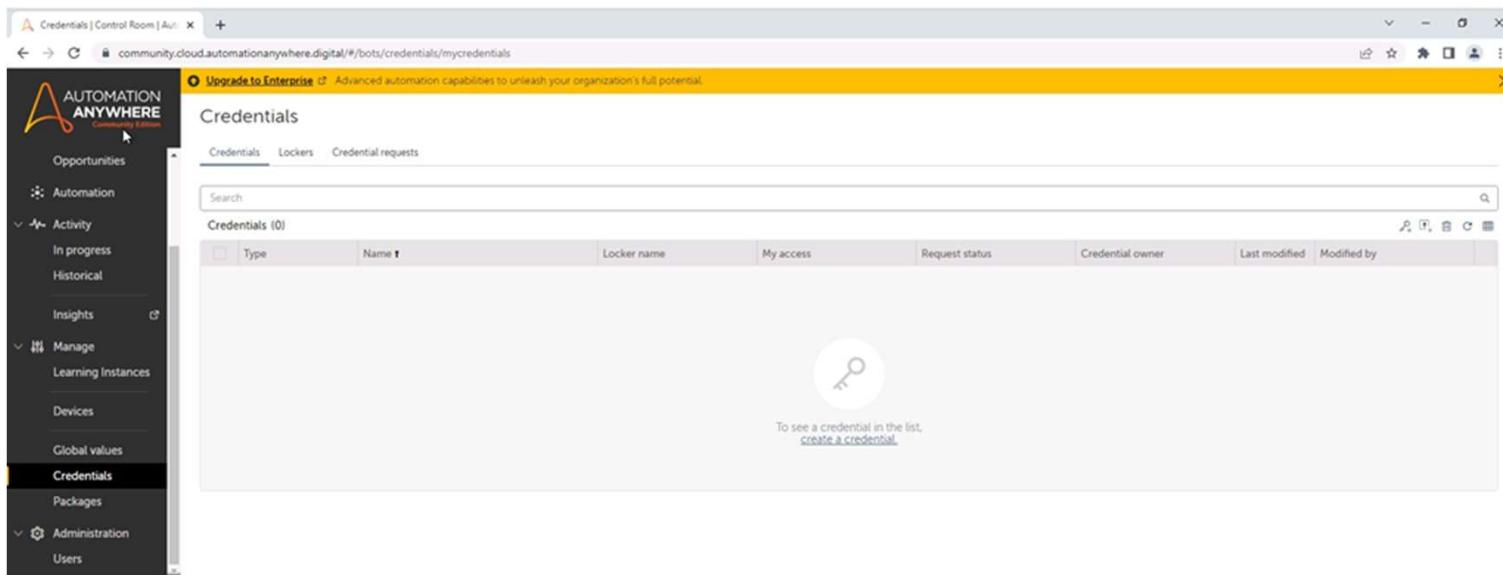
View/Hide columns

Credentials

- Create a credential with the required attributes; that is, a username/password.
- Add this credential to a locker.
- Grant the bot access to the locker.
- The bot can now get this specific credential to access the application.

Credentials

- The final CREDENTIAL REQUESTS tab is an informative tab showing all the requests that have been made for any credentials.
- You can see in the following screenshot the Credentials interface showing the three different tabs:



Credentials

- The options available for credentials are shown in the following screenshot:



Create a credential



Create locker with checked item



Delete checked item



Refresh list



View/Hide columns

Packages

All packages

Name	STA...	NAME ↑	# OF ACTIONS	VERSION	
		Analyze	2	2.1.0-20200204-154550	⋮
		Application	1	2.0.0-20200131-085947	⋮
		AWS Comprehend NLP (Beta)	4	0.1.0-20191001-174008	⋮
		AWS Comprehend NLP (Beta)	4	0.2.0-20191107-111107	⋮
		Boolean	6	2.0.0-20200131-085949	⋮
		Bot Migration	1	1.1.0-20200208-020245	⋮
		Browser	3	2.0.0-20200127-180439	⋮
		Clipboard	3	2.0.0-20200131-085958	⋮

Packages

Email

[**< Back**](#)

This page shows the package details of the selected version. You can also select a different package version and update the installed version.

Versions

2.0.0-20200206-135926 (Default)

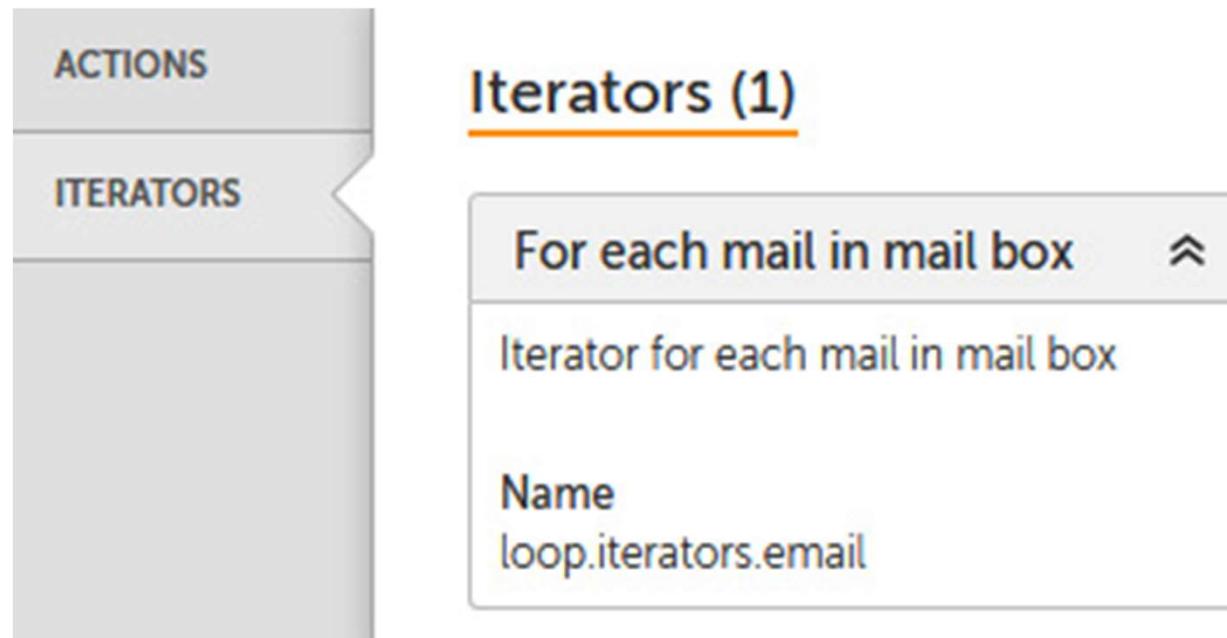


PACKAGE DETAILS				
Name	Description	Version	Status	
Email	Provides actions to perform email operations.	2.0.0-20200206-135926	Default	

ACTIONS	Actions (13)
ITERATORS	<ul style="list-style-type: none">✉ Change status✉ Check if folder exists✉ Delete all✉ Delete✉ Disconnect✉ Connect✉ Forward

Packages

- In the following screenshot, we can see that an iterator is also available to help with our bot:



Managing your devices

- By selecting the MY DEVICES option from the menu pane, you will see the following interface:

The screenshot shows the 'Control Room' interface for 'Devices > My devices'. The left sidebar has a dark theme with orange highlights for 'MY DEVICES'. The main area has a light background. At the top right, there are user icons and the email 'rpa_training@skysoftuk.net'. Below the header, the title 'My devices' is underlined. A search bar with 'Device name' dropdown and 'Search' button is present. A toolbar with icons for 'Run bot now...', 'Print', 'Upload', 'Download', 'Refresh', and 'Grid View' is above the table. The table has columns: STATUS, DEVICE NAME, DEVICE NICKNAME, DEVICE TYPE, DEFAULT USERS, BOT AGENT VERSION, and DEVICE POOL. One row is shown: 'Connected' (with a green checkmark), 'DESKTOP-MDG181K', '--', 'Single user', '14.3', '--', and three vertical dots. The bottom right corner of the table has a small orange icon.

Managing user administration

The screenshot shows the Automation Anywhere Control Room interface. The left sidebar has a dark theme with orange highlights for the selected 'Users' option under 'ADMINISTRATION'. The main area is titled 'Control Room' and 'Administration > Users'. It displays a table titled 'All users' with one entry:

	US...	USERNAME ↑	FIRST NAME	LAST NAME	DESCRIPTION	ROLES	DEVICE LICENSE	USER STATUS	LICENSE STATUS
<input type="checkbox"/>		rpa_training	RPA	Training	--	CE_user 1 more...	Bot creator	Enabled	Verified

Managing user administration

The screenshot shows the Automation Anywhere Control Room interface. The left sidebar menu includes HOME, DASHBOARD, ACTIVITY, BOTS (with sub-options My bots, Credentials, Packages), MY DEVICES, and ADMINISTRATION (with sub-option Users selected). The main content area is titled "Control Room" and shows the "View user" page for the email address "rpa_training@skysoftuk.net". The page displays "USER DETAILS" with fields: First name (RPA), Last name (Training), Description (--); Email (rpa_training), Password (****), User status (Enabled); License (Bot creator), License status (Verified), device (DESKTOP-MDG181K); Auto login (Cannot auto login). Below this is a "Roles (2)" section listing "NAME" (AAE_Bot Insight Expert, CE_user). At the bottom is a "GENERAL DETAILS" section showing Last modified (12:46:13 GMT 2020-02-14), Modified by (rpa_training), Object type (User), and User type (Bot creator).

USER DETAILS			
First name	RPA	Last name	Training
Email	rpa_training	Password	****
License	Bot creator	License status	Verified
Auto login	Cannot auto login	device	DESKTOP-MDG181K

Roles (2)	
NAME	AAE_Bot Insight Expert
NAME	CE_user

GENERAL DETAILS			
Last modified	12:46:13 GMT 2020-02-14	Modified by	rpa_training
Object type	User	User type	Bot creator

Summary

- You should now be comfortable with Automation Anywhere's user interface, having a clearer understanding of all the configurations needed to design and support your bot.
- You will understand why features such as security, reusability, and a simple interface make Automation Anywhere an award-winning RPA tool and one of the industry leaders.
- You should now have an overview of the Control Room interface and its features.

Overview of the Automation Anywhere Development Interface



Automation Anywhere Development Interface

In this lesson, we will cover the following topics:

- Bot development interface
- What can a bot do?
- Programming techniques using Automation Anywhere
- Variables and triggers
- Debugging and dependencies

Technical requirements

- Windows OS version 7 or higher
- A processor with a minimum speed of 3 GHz
- A minimum of 4 GB RAM
- At least 100 MB of hard disk space
- Web browser: Internet Explorer v10 or higher or Chrome v49 or higher
- A minimum screen resolution of 1024*768
- An internet connection with a minimum speed of 10 Mb/sec
- Completed registration with Automation Anywhere Community Edition
- Successful logon to Automation Anywhere Community Edition
- Successful registration of a local device

Bot development interface

- The development interface is where all the magic happens.
- This is where bots are created, edited, and debugged. In order to look at what it has to offer, you need to start by creating a new bot.
- Let's dive straight into creating a bot so that we can explore the development interface.

Creating a new bot

The screenshot shows the Automation Anywhere web interface. The left sidebar has a dark theme with the following navigation items:

- Explore
- Home
- Discovery
- Processes
- Opportunities
- Automation** (selected)
- Activity
- In progress
- Historical
- Insights
- Manage
- Learning Instances

The main content area is titled "Automation". It shows a "Folders" section with a tree view:

- Bots
 - Document Workspace Pr...
 - Sample bots** (selected)

A yellow banner at the top right says "Upgrade to Enterprise" and "Advanced automation capabilities to unleash your organization's full potential". A "Create new" button dropdown menu is open, showing options: Bot, Form, and Process.

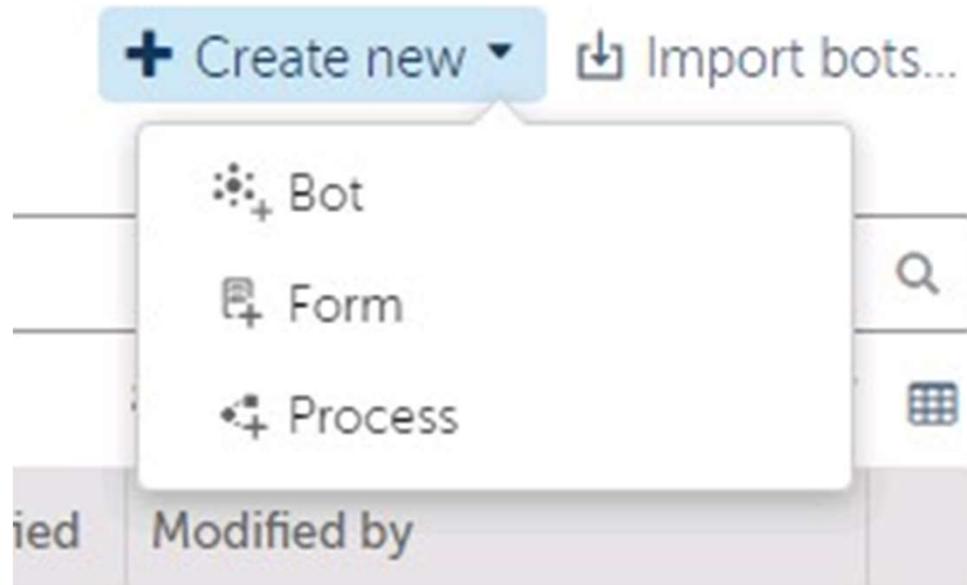
The main table area displays a single row with the following columns:

Name	Type	Name	Status	Size	Last modified	Modified by
	t1	t2				

A large circular icon with a grid pattern is centered below the table. Below it, text reads: "To see bots or folders here, [create a bot](#) or [create a subfolder](#)".

Creating a new bot

- Click on the Create a bot icon from the icon options in the top-right corner:



Creating a new bot

- The following bot properties dialog is presented:

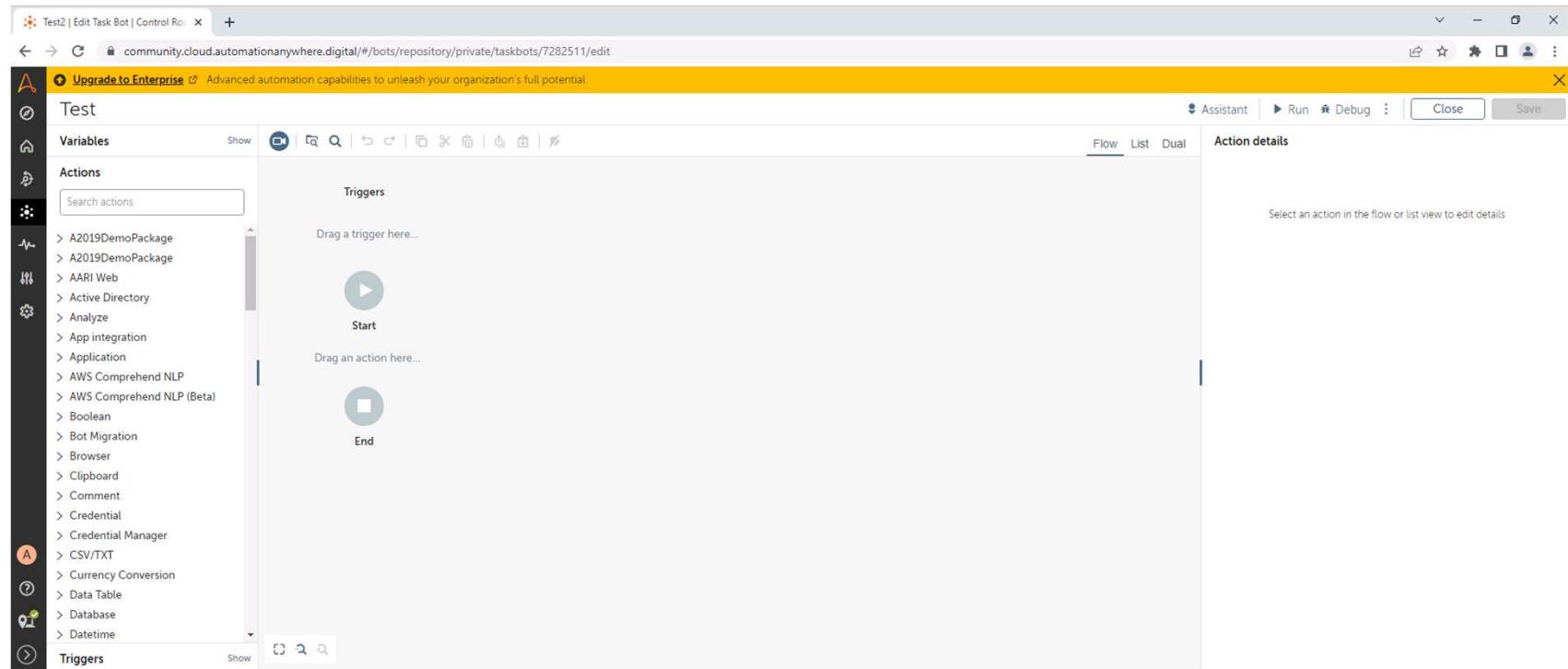
Create Task Bot

Name **Description (optional)**

Max characters = 50 Max characters = 255

Folder **Browse...**

Creating a new bot

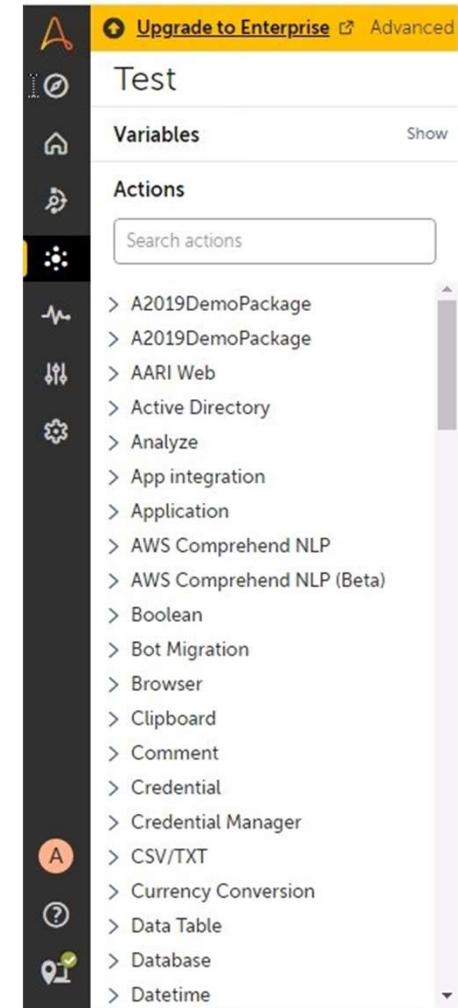


Creating a new bot

The screenshot shows the Automation Anywhere Community Edition Control Room interface. At the top left is the logo and text "AUTOMATION ANYWHERE Community Edition". To the right is a breadcrumb navigation "Control Room" and "My bots > Edit Task Bot". A large blue arrow points upwards from the bottom of the slide towards the "Edit Task Bot" link. On the left, there's a vertical navigation menu with "HOME", "DASHBOARD", "ACTIVITY", and "BOTS". The "BOTS" item has a dropdown menu with "My bots" selected, indicated by an orange dot. On the right, under the heading "Actions", there's a search bar labeled "Search actions" and three collapsed dropdown menus: "Analyze", "Application", and "AWS Comprehend NLP (Beta)".

What can a bot do?

- All the options are listed on the left pane, These actions are grouped into categories known as packages.
- Each package is a collection of individual actions.
- Here you can see the list pane showing all the available packages:



Programming techniques using Automation Anywhere

The screenshot shows the Automation Anywhere Control Room interface. At the top, a dark header bar displays the title "Control Room". Below it, a navigation bar shows the path "Bots > My bots > Edit Task Bot". The main area is titled "Test". At the top of this area, there are three tabs: "Flow", "List" (which is underlined, indicating it is the active tab), and "Dual". On the left, there is a sidebar with the heading "Actions" and a search bar labeled "Search actions". To the right of the sidebar is a toolbar with various icons: a video camera, a laptop, a document, a delete symbol, a file, a lightning bolt, a circular arrow, and a refresh symbol. Below the toolbar, the word "Triggers" is visible.

Programming techniques using Automation Anywhere

The screenshot shows the Control Room interface for a bot named "Edit Task Bot". The main area is titled "Test" and includes buttons for "Run", "Debug", "Analyze", "Close", and "Save". Above the main area are tabs: "Flow", "List" (which is selected), and "Dual". The left side features a sidebar titled "Actions" with a search bar and a list of actions. The "Message box" action is highlighted with a blue arrow labeled "2" pointing to it. A tooltip for "Message box" states: "Displays a message box". The right side of the interface contains a workspace with a toolbar at the top and a central area for dragging triggers and actions. A trigger icon is shown with a blue arrow labeled "1" pointing to it, and a tooltip says "Drag a trigger here...". Below the trigger is a placeholder for an action with the text "Drag an action here...".

Programming techniques using Automation Anywhere

The screenshot shows the Automation Anywhere Control Room interface. At the top, it displays 'Control Room', 'Bots > My bots > Private > Edit Task Bot', and a user profile 'rpa_training'. Below the header, there's a title 'Test' and buttons for 'Run', 'Debug', 'Analyze', 'Cancel', and 'Save'. The main area is divided into sections: 'Variables' (with a 'Show' button), 'Actions' (with a 'Search actions' input and a list including 'Legacy automation', 'List', 'Log To File', 'Loop', 'ML - Microsoft Anomaly Dete...', 'MS Word', 'Message box', 'Microsoft LUIS NLP (Beta)', 'Mouse', 'Number', 'NumberUtils', 'OCR', 'Office 365 Calendar', and 'Triggers'), 'Flow' (with tabs for 'List' and 'Dual'), 'Triggers' (with a placeholder 'Drag a trigger here...'), 'Start' (with a selected 'Message box' action), and 'End'. To the right, there's a detailed configuration panel for the 'Message box' action, which includes fields for 'Message box window title' (set to 'Automation Anywhere Enterprise Client'), 'Message to display' (set to 'Hello World'), 'Scrollbar after lines' (set to '# 30'), and a checkbox for 'Close message box after' (unchecked). A 'Seconds' field is also present.

Programming techniques using Automation Anywhere

- Set the following properties for the Message box action:

Message box

Displays a message box

Enter the message box window title

” Test Message Box 1

Enter the message to display

” Hello World

Scrollbar after lines

30

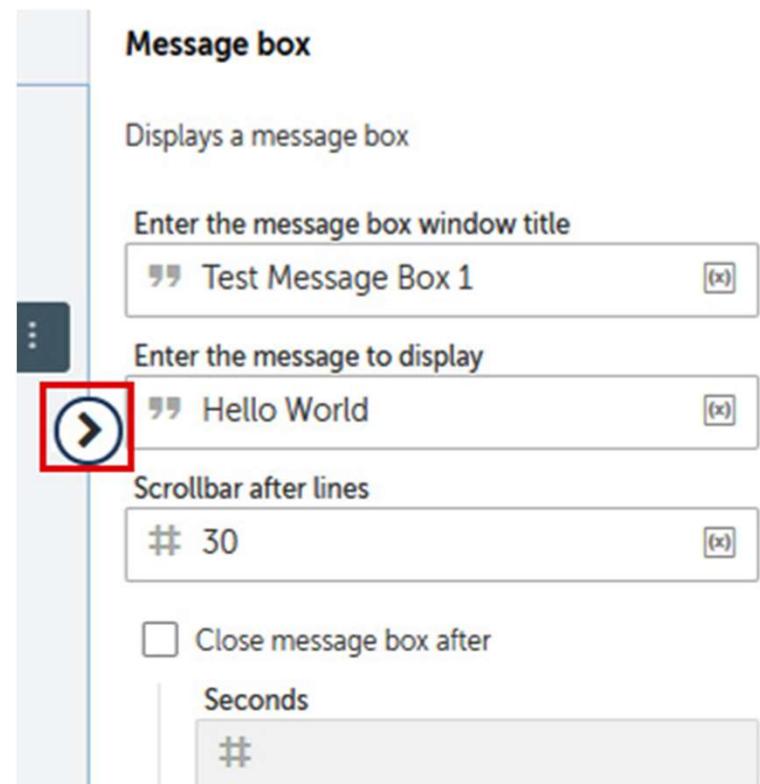
Close message box after

Seconds

2

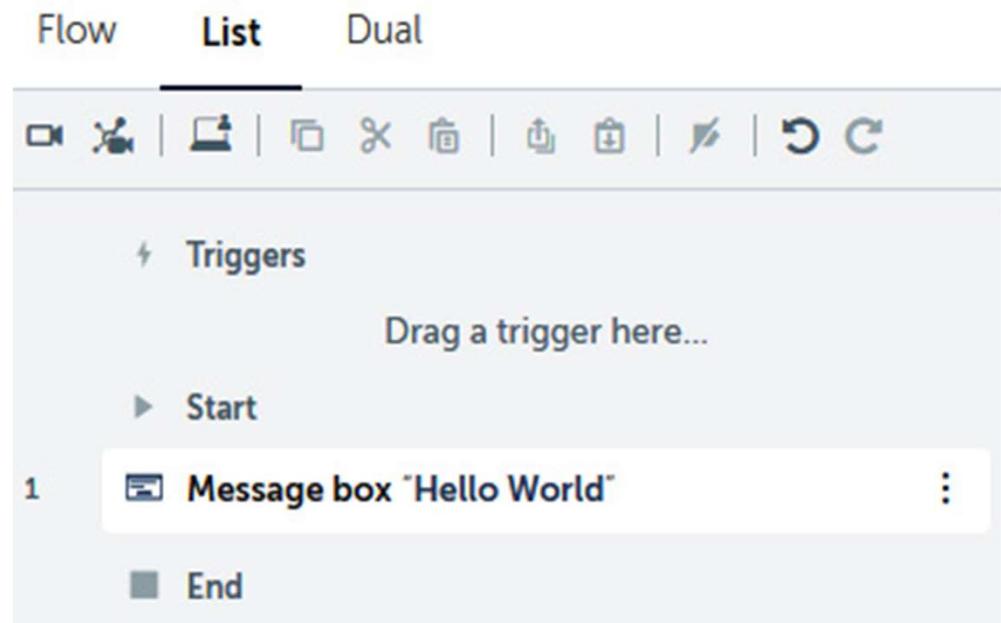
Programming techniques using Automation Anywhere

- You can collapse the properties pane when you are not using it.
- Just hover over and click the collapse icon to hide it:



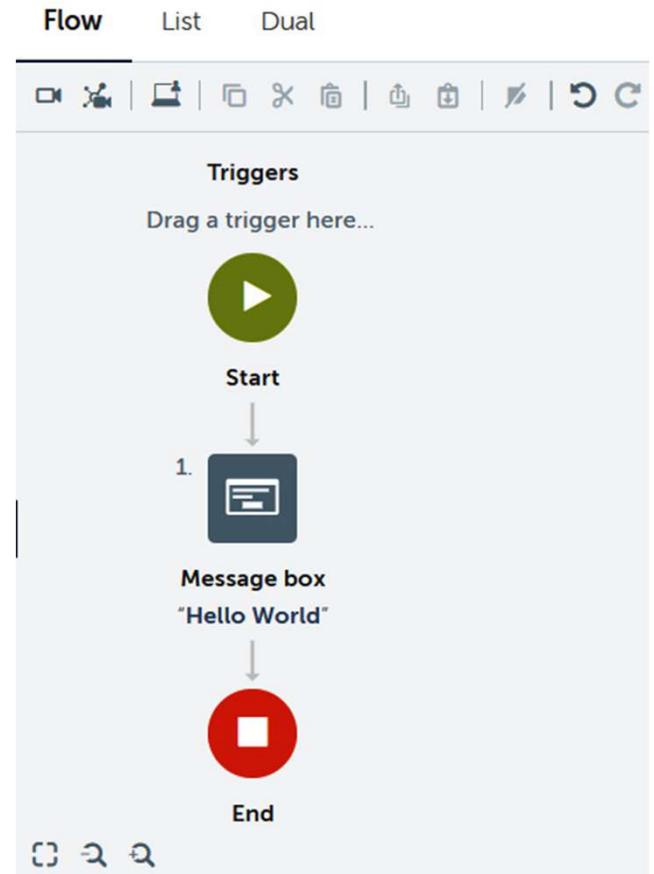
List view

- You have used this so far to add the Message box action.
- In this view, all actions are listed, similar to a script.
- The line number provides the logical workflow of the task:



Flow view

- The Flow view shows your task actions as a flowchart.
- As you can see, this provides a more visual process flow of how your bot works through the workflow.
- The line numbers are omitted as they are not necessary:



Flow view

- You can change the size of the flow diagram using the sizing options on the bottom left.
- These options are as follows:



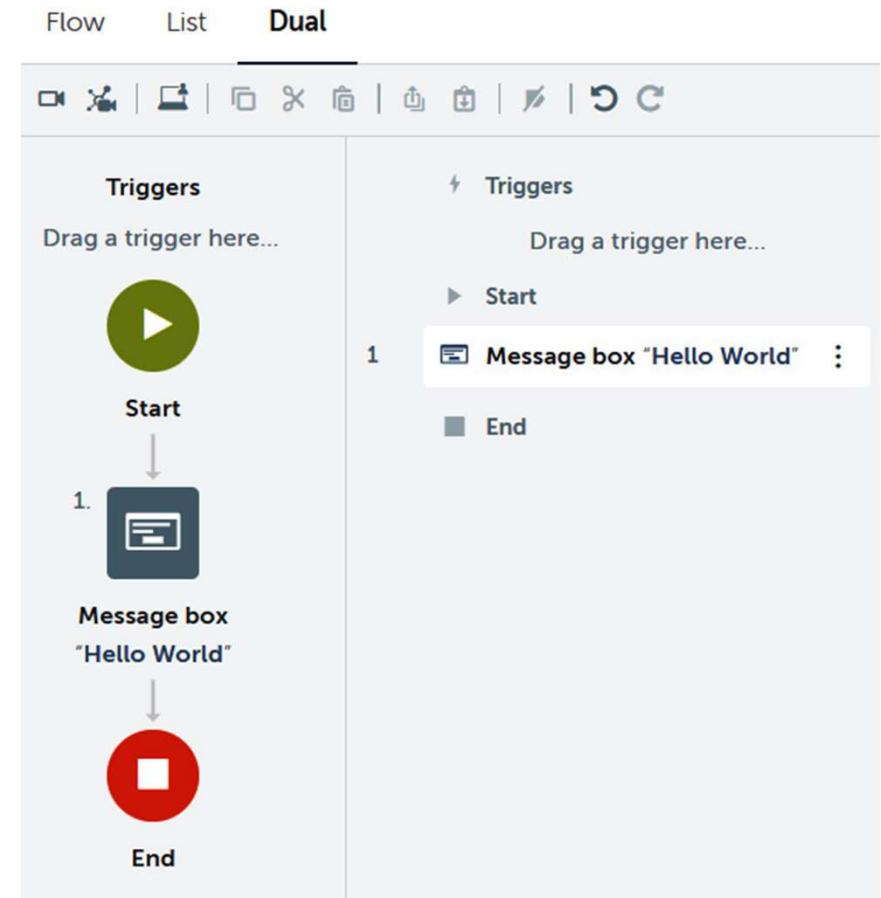
Zoom fit

Zoom in

Zoom out

Dual view

- If you select an action on either view, it will also be selected in the other view.
- This makes navigation and editing much easier:



Dual view

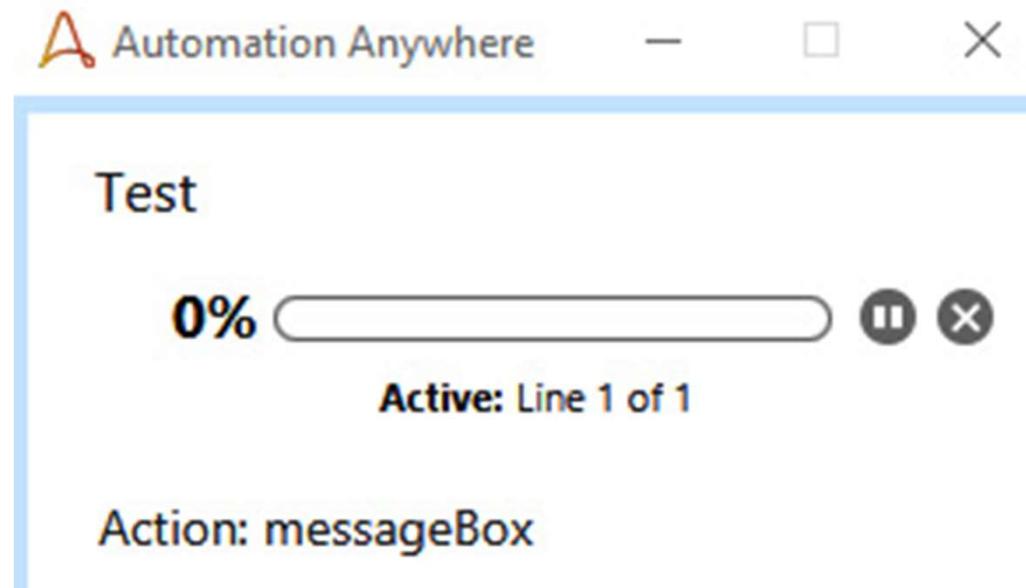
- Click on the Save button in the top-right pane:



- Click on Run to execute the bot.

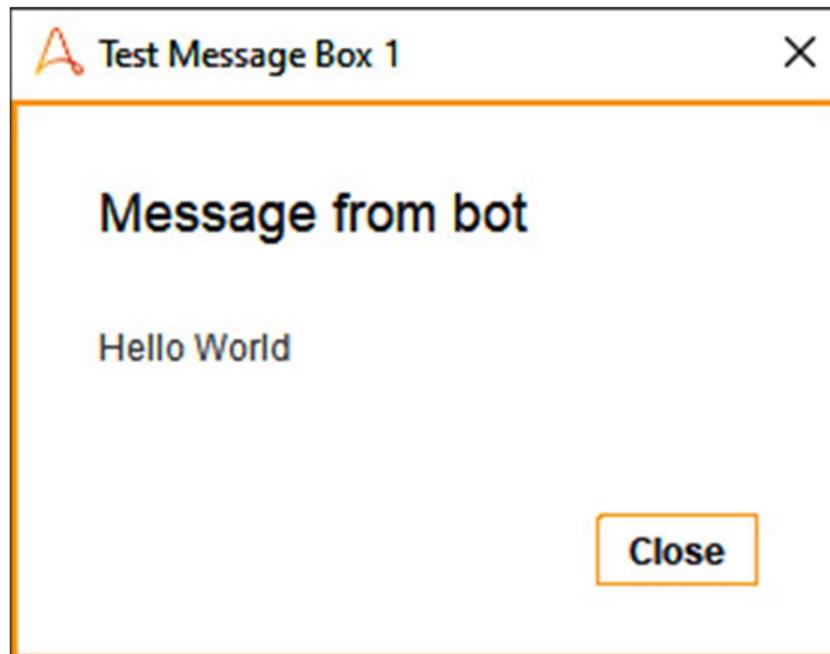
Dual view

- It is useful for testing and debugging your bot:



Dual view

- The Message Box should be displayed as your bot runs the action:



Dual view

- Click on the Close button in Message Box.
- Once the bot has completed, you will be presented with the following:

Your bot has run successfully!



Close

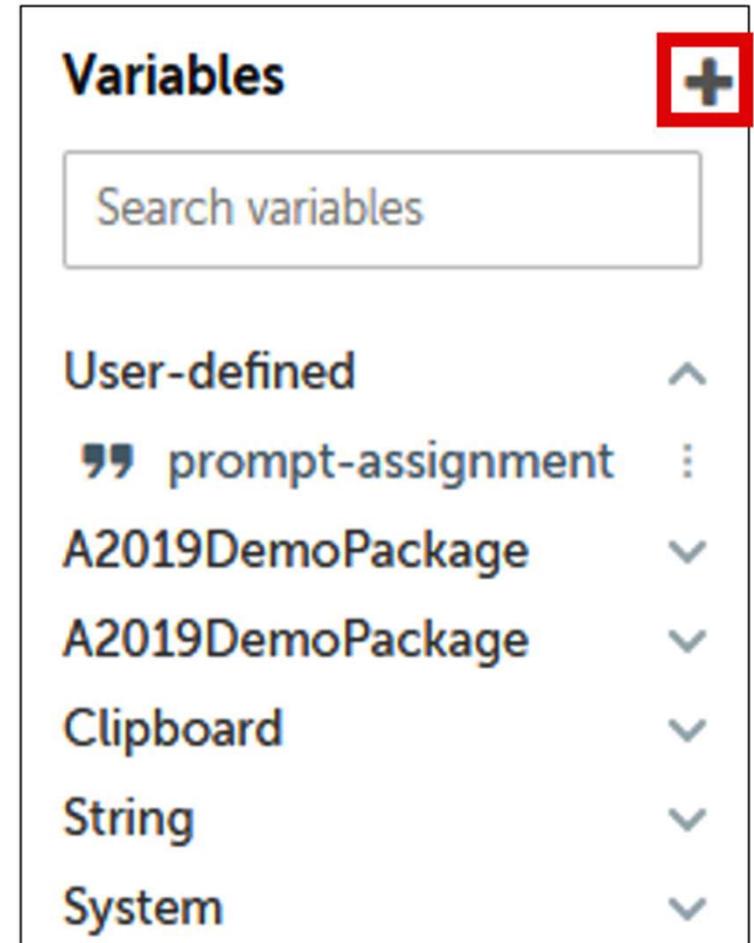
Variables and triggers

- Click on Show for the Variables tab from the option pane on the left:

The screenshot shows the 'Variables' tab selected in the top navigation bar of a software interface. A red box highlights the 'Show' button. Below the tab, the word 'Actions' is displayed. A search bar contains the placeholder text 'Search actions'. To the right of the search bar is a vertical scroll bar. Below the search bar, three items are listed with dropdown arrows: 'Legacy automation', 'List', and 'Log To File'. The 'Legacy automation' item has an upward arrow icon to its right.

Variables and triggers

- Click on the + icon to create a new variable:



The screenshot shows a 'Variables' panel with a search bar at the top. Below the search bar, there is a section titled 'User-defined' with a collapse arrow. Underneath are several variable entries, each with a collapse arrow: 'prompt-assignment', 'A2019DemoPackage', 'Clipboard', 'String', and 'System'. In the top right corner of the panel, there is a red-bordered '+' icon.

User-defined	
prompt-assignment	⋮
A2019DemoPackage	▼
A2019DemoPackage	▼
Clipboard	▼
String	▼
System	▼

Variables and triggers

- Will launch the Create variable dialog box:

Create variable

Cancel

Create

▲ Name

Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

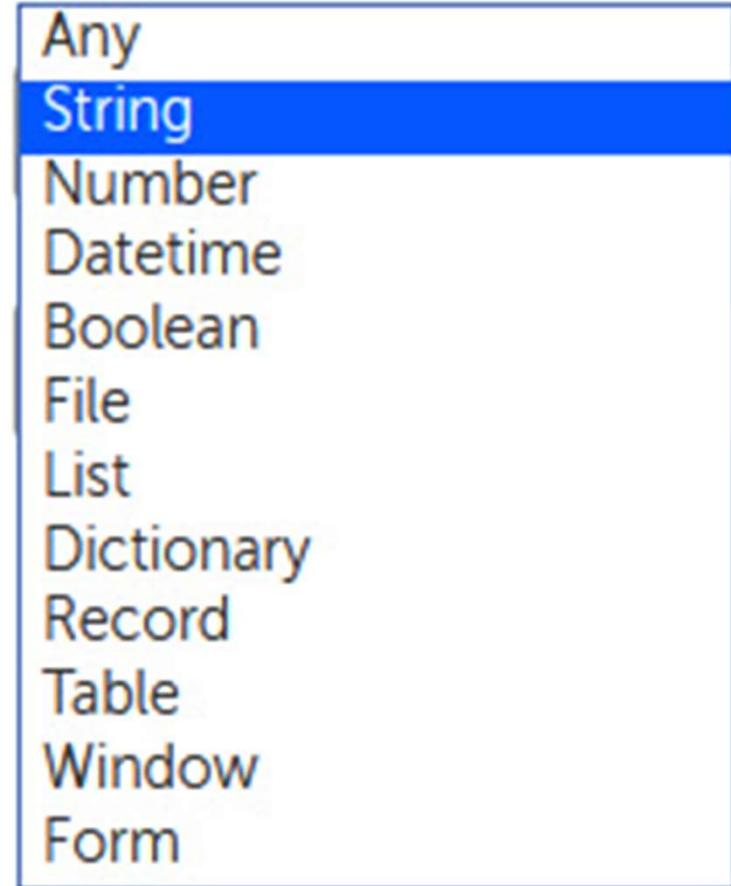
Type

String

Default value

Variables and triggers

- You will notice from the Type variable drop-down list all the different data types that are available for your variables:



Variables and triggers

- The Create variable dialog box should look similar to this:

Create variable

Name
strName
Max characters = 50

Description (optional)
Max characters = 255

Use as input

Use as output

Constant (read-only)

Type
String

Default value
Husan

Cancel **Create**

Variables and triggers

Flow List Dual

The screenshot shows a software interface with a toolbar at the top containing various icons. Below the toolbar, there are three main sections: 'Triggers' (selected), 'Start', and a numbered list item '1'. The '1' item contains a 'Message box "Hello World"' trigger, which is currently selected. To the right of the list is a detailed configuration panel for the selected trigger.

Message box

Displays a message box

Enter the message box window title
"Test Message Box"

Enter the message to display
"Hello World"

Scrollbar after lines
30

Close message box after
Seconds
#

Variables and triggers

- Modify the following property value:

Enter the message to display: Hello \$strName\$

- The property value should look like this:

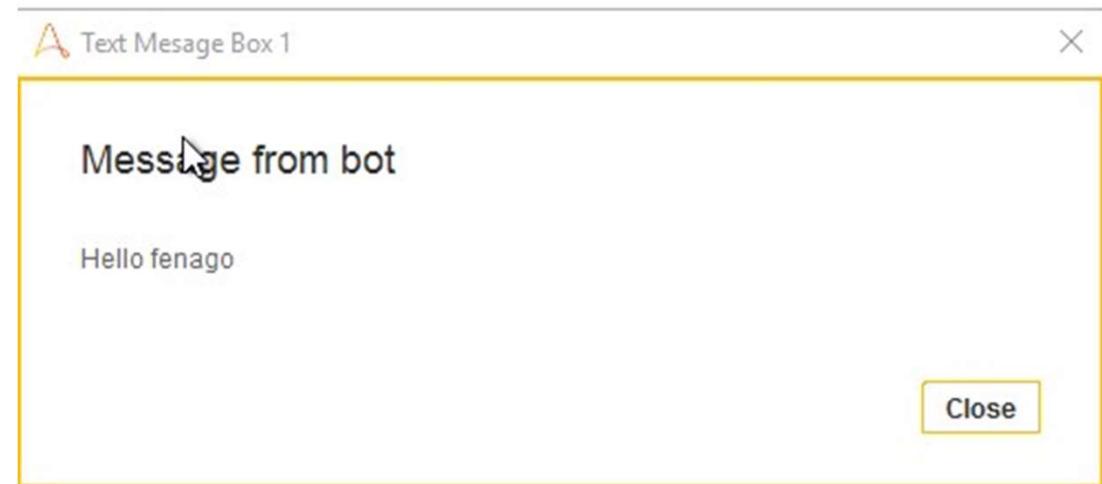
Enter the message to display

” Hello \$strName\$



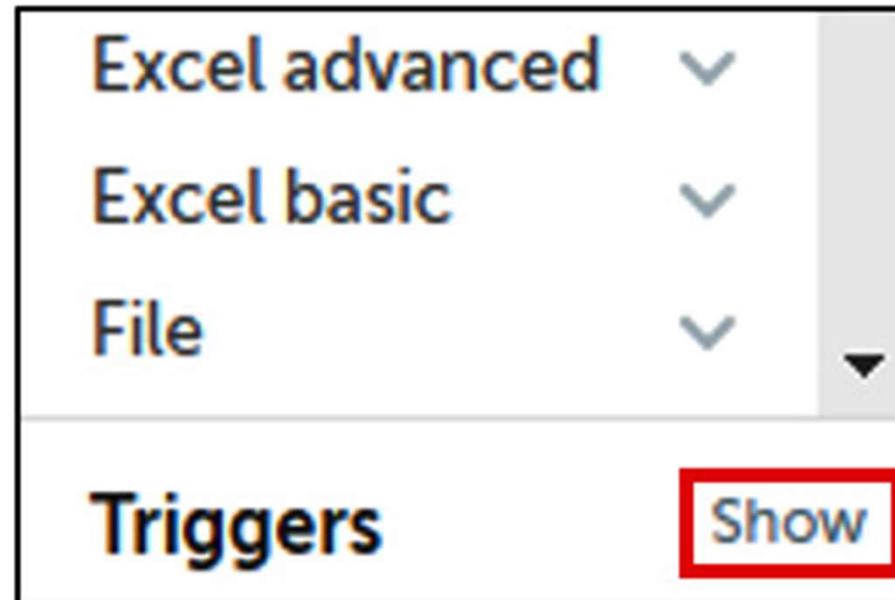
Variables and triggers

- Now, let's run the bot by clicking on Run.
- You should get a Message Box with your name:



Variables and triggers

- Click on Show on the Triggers tab from the option pane on the left:



Variables and triggers

- Expand the Files & folders trigger group:

The screenshot shows a 'Triggers' interface with a search bar at the top. Below the search bar, there is a list of trigger categories. The 'Files & folders' category is highlighted with a red border. Underneath it, two sub-options are listed: 'File trigger' and 'Folder trigger'. Other categories shown include 'A2019DemoPackage' (expanded), 'Email trigger', 'Hot key', and 'Interface trigger'.

Triggers	
Search triggers	
A2019DemoPackage	▼
A2019DemoPackage	▼
Email trigger	▼
Files & folders	▲
File trigger	
Folder trigger	
Hot key	▼
Interface trigger	▼

Variables and triggers

Flow List **List** Dual

Variables Show

Actions Show

Triggers

Search triggers

A2019DemoPackage

A2019DemoPackage

Email trigger

Files & folders

- File trigger
- Folder trigger

Hot key

Interface trigger

Triggers

Files & folders: File trigger when a file is created

Start

1 Message box "Hello \$strName\$"

End

The screenshot shows the Microsoft Power Automate interface in 'List' mode. On the left, there's a sidebar with sections for 'Variables', 'Actions', and 'Triggers'. Under 'Triggers', there's a search bar and a list of trigger types: A2019DemoPackage, A2019DemoPackage, Email trigger, Files & folders (which is expanded, showing 'File trigger' and 'Folder trigger'), Hot key, and Interface trigger. On the right, the main area shows a single trigger item: 'Files & folders: File trigger when a file is created'. This item has a status icon (yellow warning), a delete icon, and a more options icon. Below it, the flow starts with 'Start', followed by a step 'Message box "Hello \$strName\$"', and ends with 'End'.

Variables and triggers

Set the following properties for the new
Files & folders: File trigger:
File: C:\RPA\TriggerFile.txt

Start the bot when the file is...: modified

- The trigger properties should look similar to this:

Files & folders: File trigger

Triggers on a file event.

File

C:\RPA\TriggerFile.txt

Browse...

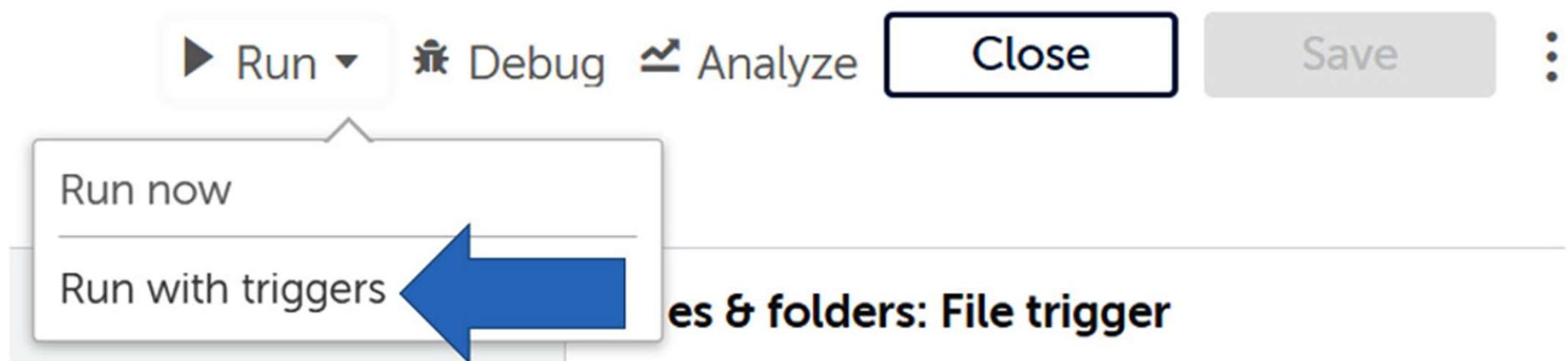
Start the bot when the file is...

modified



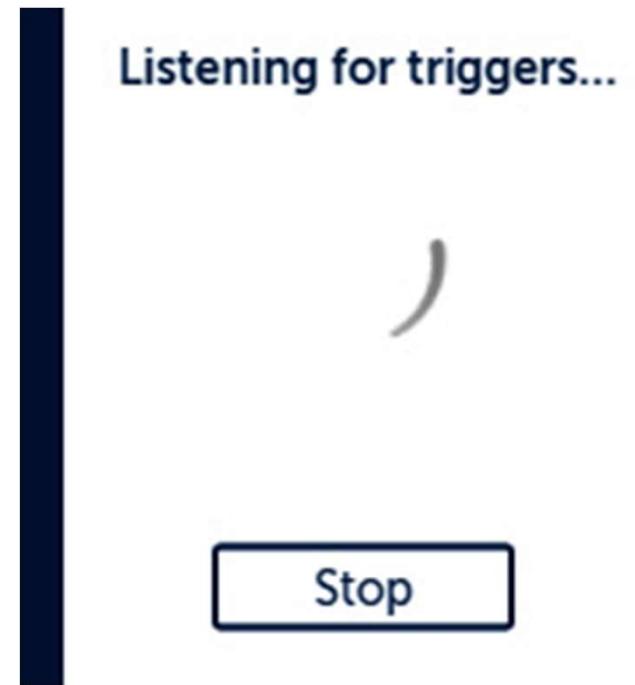
Variables and triggers

- To run the bot with triggers, click on Run and, from the dropdown, select Run with triggers:



Variables and triggers

- The bot will be deployed to your device. It will then wait for the trigger event to be true.
- This is known as listening for triggers. The following message box should appear as it listens for the trigger:



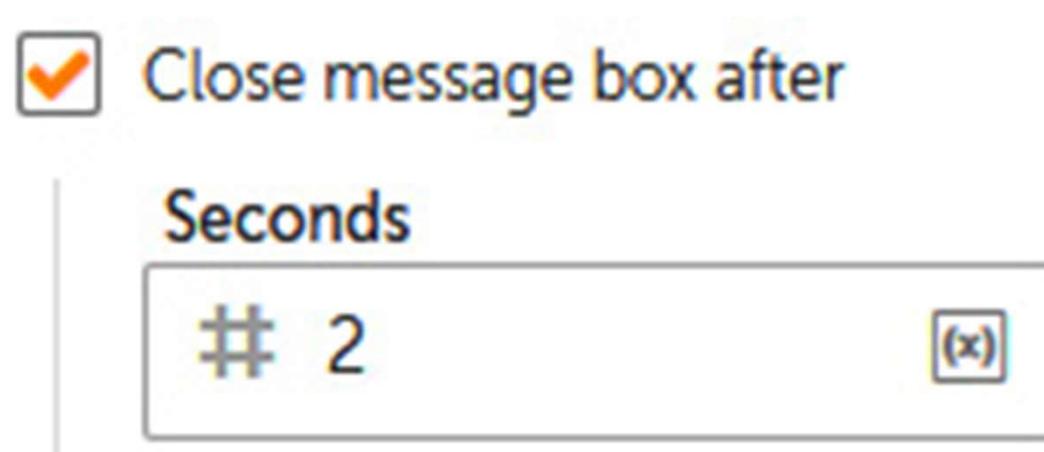
Variables and triggers

- Open, edit, and save the TriggerFile.txt trigger file that we created.
- You will notice the bot start and execute once you have saved the file:

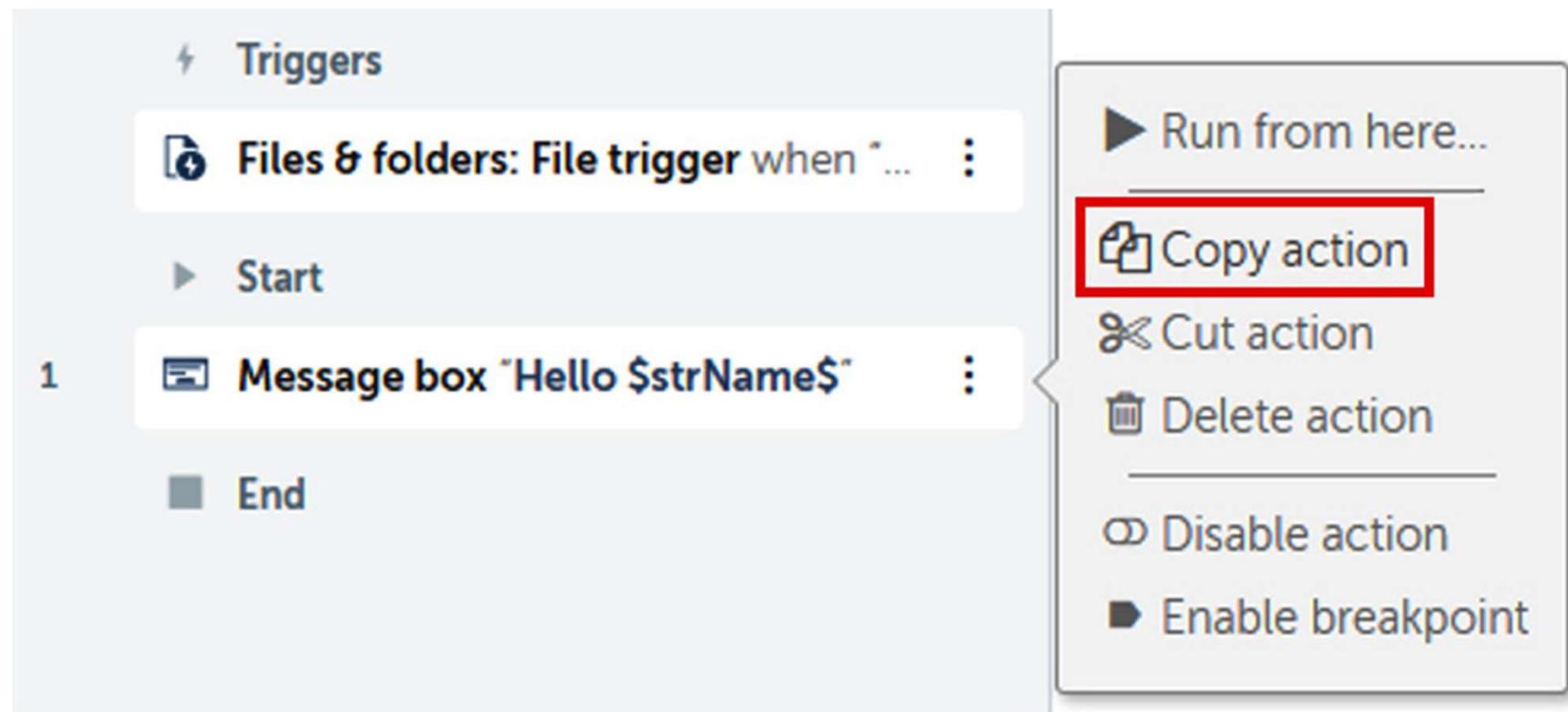


Debugging and dependencies

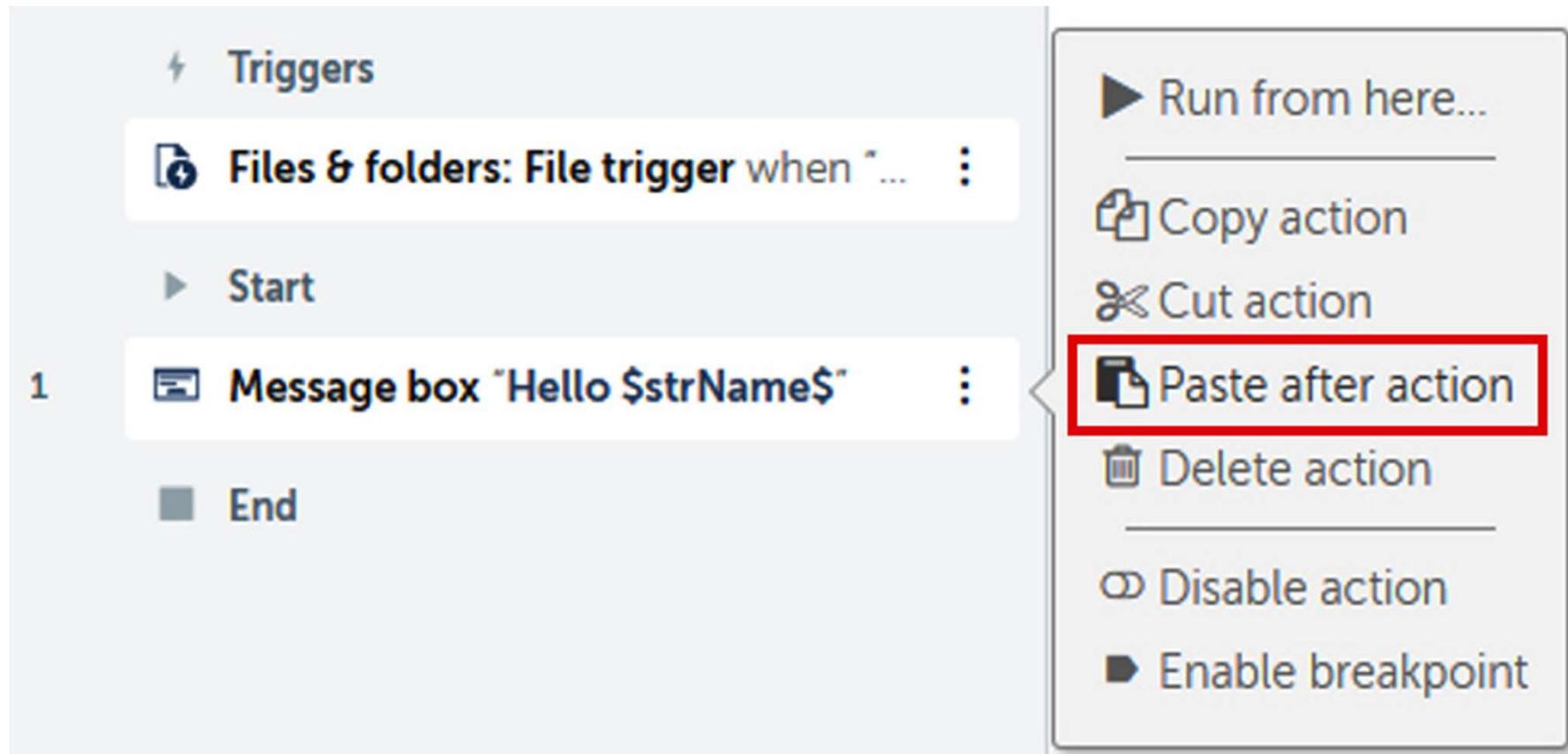
- Navigate to the properties of our Message box action on line 1 and update the following properties:



Debugging and dependencies

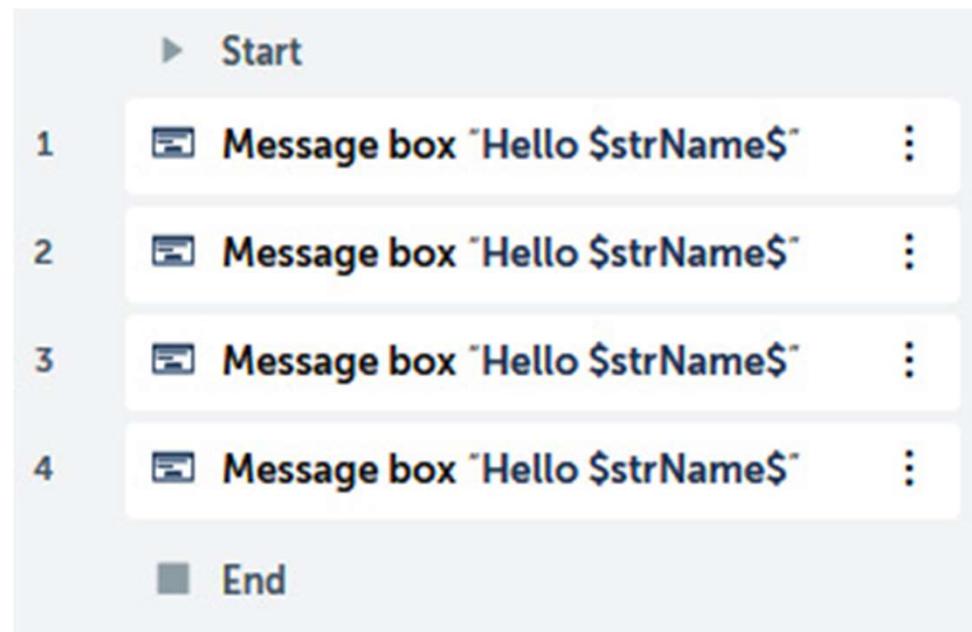


Debugging and dependencies



Debugging and dependencies

- Repeat last step two more times.
- The development interface should look like this:



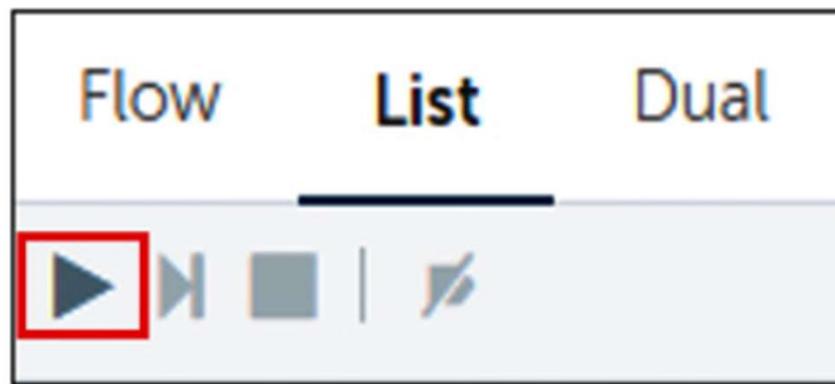
Switch to debug mode by clicking on **Debug** from the top-right menu:



Figure 4.39 – Switching to debug mode

Debugging and dependencies

Once in debug mode, you can run the bot from the start icon above the development window, as shown in the following screenshot:



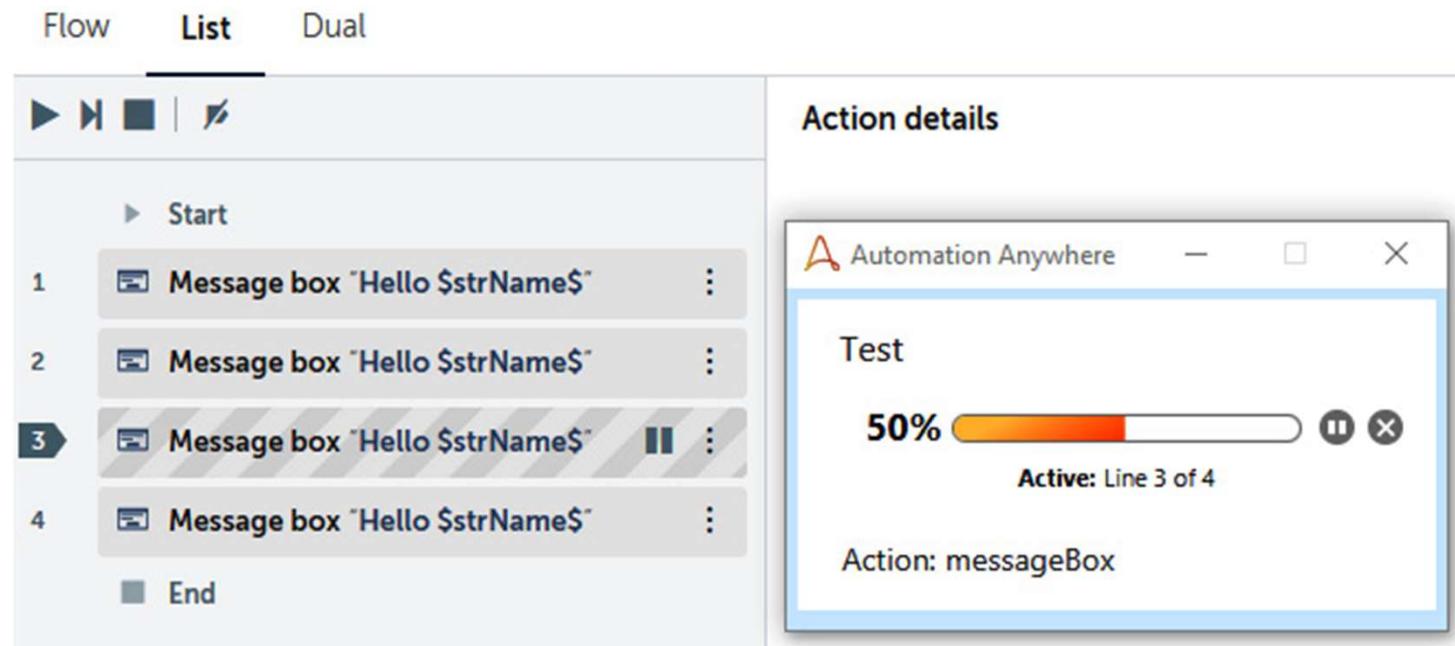
Debugging and dependencies

- We will add a breakpoint to the Message box action on line 3. To do this, select Enable breakpoint from the action line options:



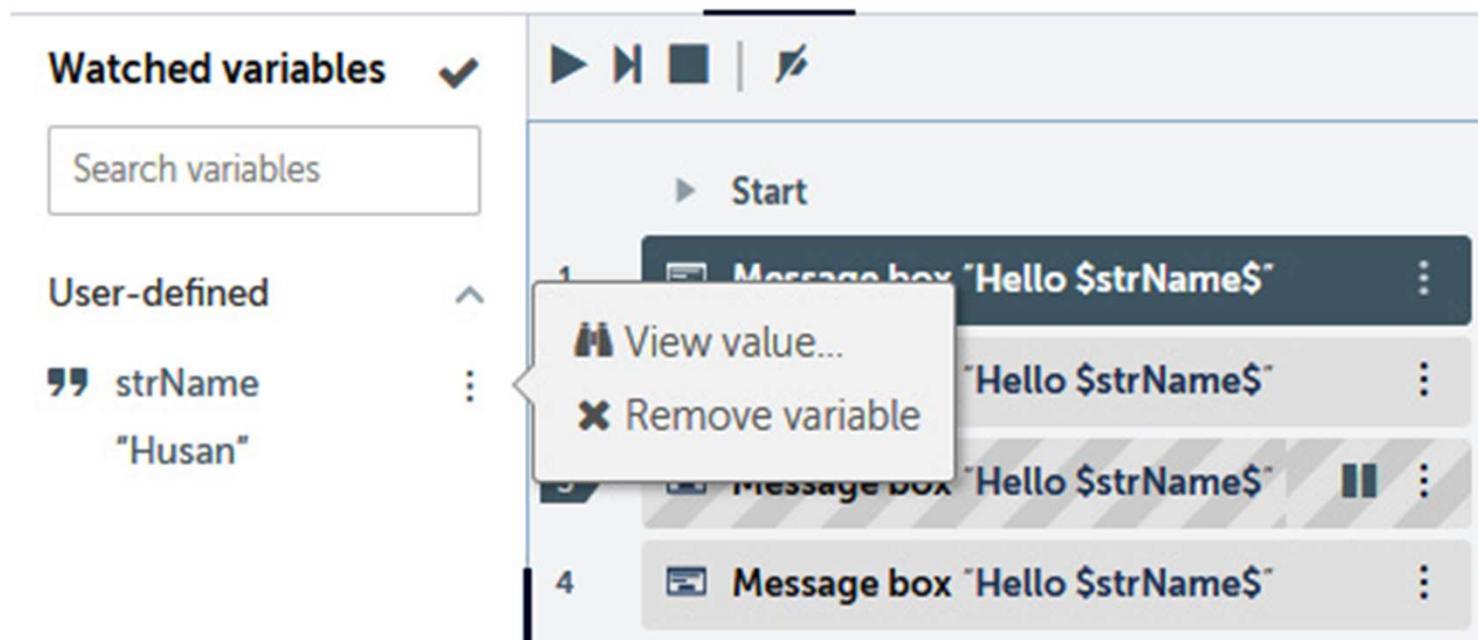
Debugging and dependencies

- Now run the bot and you will notice the bot pause at the breakpoint action:



Debugging and dependencies

- The breakpoint also allows you to view any values assigned variables during runtime:

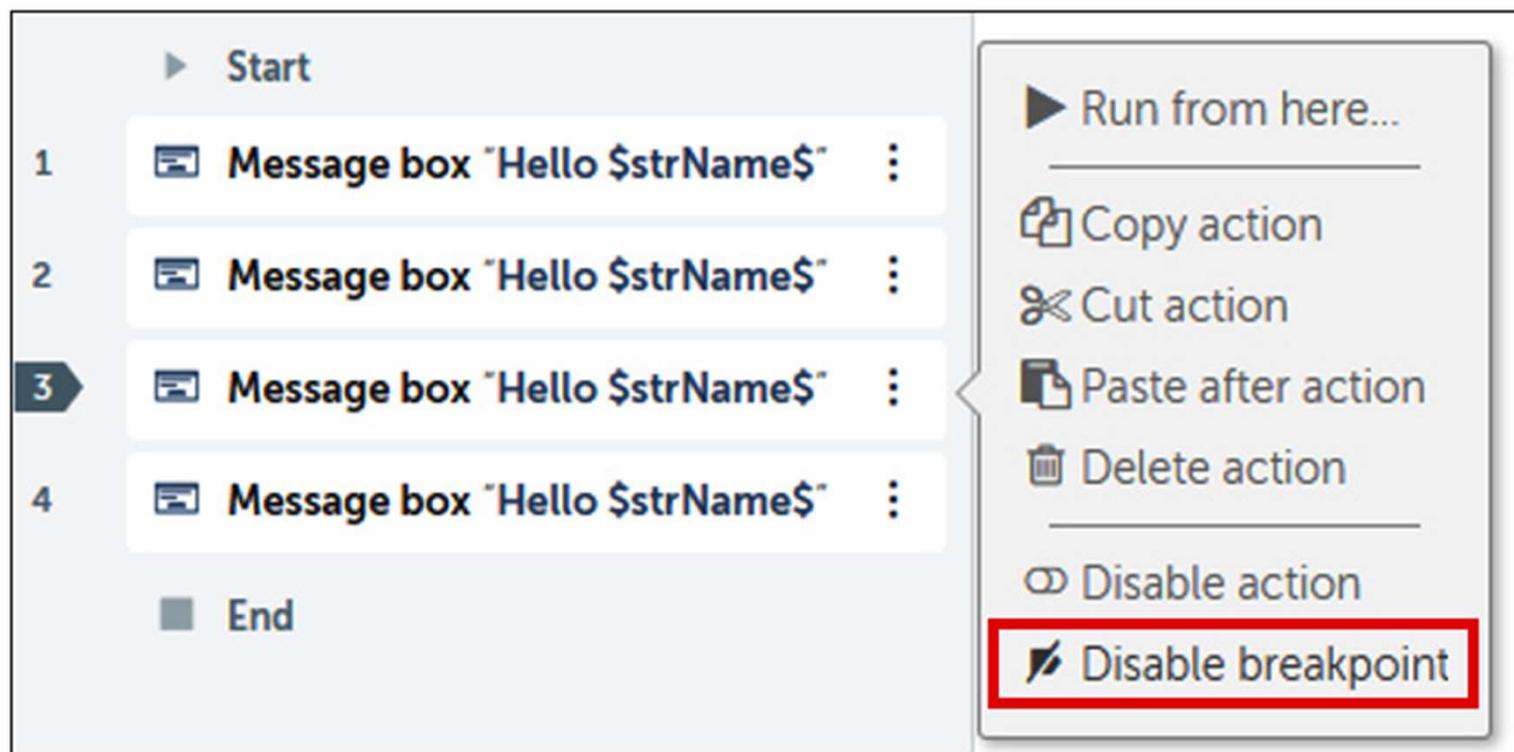


Debugging and dependencies

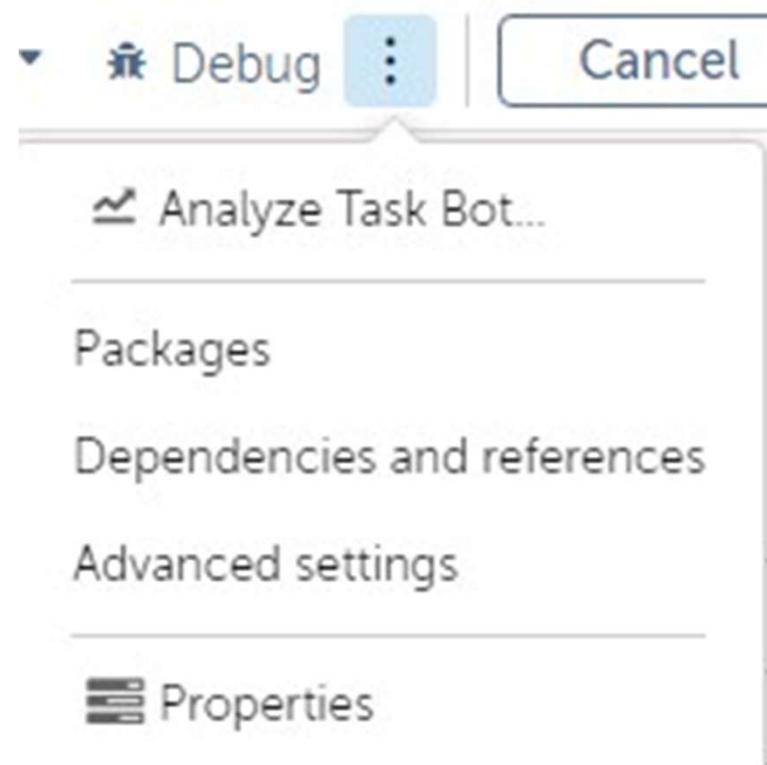
- Once the bot has completed, we can exit debug mode by clicking on Exit debug to take you back to the normal development interface:



Debugging and dependencies



Debugging and dependencies



Debugging and dependencies

The screenshot shows the Control Room interface with the following details:

- Control Room Header:** Includes a menu icon, user status (green checkmark), and email (rpa_training@skysoftuk.net).
- Breadcrumbs:** Bots > My bots > Edit Task Bot.
- Title:** Test.
- Buttons:** Close, Save, and a three-dot menu.
- Folders:** A tree view showing a 'Bots' folder expanded, with a 'Sample bots' subfolder selected.
- Available files:** A table with columns: Name (dropdown), Search, TYPE, NAME (sorted), PATH, and SIZE. It shows 0 results.
- Selected files:** A table with columns: Name (dropdown), Search, TYPE, NAME (sorted), PATH, and SIZE. It shows 0 results.
- GENERAL DETAILS:** A summary section with the following data:

Last modified 12:47:10 BST 2020-04-01	Modified by rpa_training@skysoftuk.net	Object type Task Bot
---	---	-------------------------

Summary

- You should now be comfortable with the Automation Anywhere development interface.
- We have gone through creating a simple bot. You have created variables and triggers for your bot.
- In addition, you have also acquired some hands-on experience of using the debugging tool for Automation Anywhere.
- Having created your initial bot and gotten an overview of all the other actions available, you must be keen to start implementing more complex actions in your bot.

Building Your First Bot



Building Your First Bot

In this lesson, we will cover the following topics:

- Downloading sample data from GitHub
- Understanding your automation task
- Creating and reading a CSV file
- Performing basic arithmetic calculations
- Appending records to a CSV file

Technical requirements

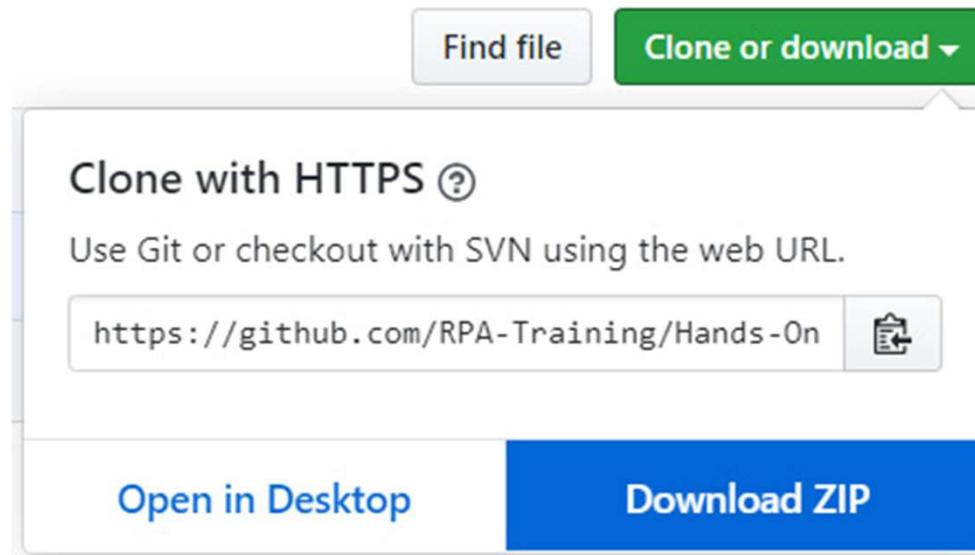
- Windows OS version 7 or higher
- A processor with a minimum speed of 3 GHz
- A minimum of 4 GB RAM
- At least 100 MB of hard disk space
- Internet Explorer v10 or higher OR Chrome v49 or higher
- A minimum screen resolution of 1024*768
- An internet connection with a minimum speed of 10 Mb/sec
- Completed registration with Automation Anywhere Community Edition
- Successful login to Automation Anywhere Community Edition
- Successful registration of a local device

Downloading sample data from GitHub

The screenshot shows a GitHub repository page. At the top, there's a navigation bar with links to 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', 'Pricing', a search bar, and 'Sign in'/'Sign up' buttons. Below the navigation is the repository header: 'RPA-Training / Hands-On-RPA-with-AA'. It shows 'Watch 1', 'Star 0', and 'Fork 0'. The main navigation tabs are 'Code' (selected), 'Issues 0', 'Pull requests 0', 'Actions', 'Projects 0', 'Security', and 'Insights'. A prominent 'Join GitHub today' banner is displayed in the center, featuring a 'Sign up' button. Below the banner, it says: 'GitHub is home to over 40 million developers working together to host and review code, manage projects, and build software together.' On the left, there's a sidebar with a tree view icon and a checkmark. On the right, there's a 'Dismiss' button and a yellow square icon. Below the banner, a message states: 'No description, website, or topics provided.' Key statistics are listed: '5 commits', '2 branches', '0 packages', '0 releases', and '1 contributor'. A dropdown menu for the branch 'Sample-Data' is open. At the bottom, there's a summary of the branch status: 'This branch is 4 commits ahead, 1 commit behind master.', along with 'Find file', 'Clone or download', 'Pull request', and 'Compare' buttons. Below that, a file list shows 'RPA-Training Add files via upload' and 'Chapter05_InputData.csv' with a timestamp of 'Latest commit 23e206a 4 minutes ago'. Another row shows 'Add files via upload' and '4 minutes ago'.

Downloading sample data from GitHub

- To download, click on the green Clone or download button. This will open a small dialog box, and then click on Download ZIP:

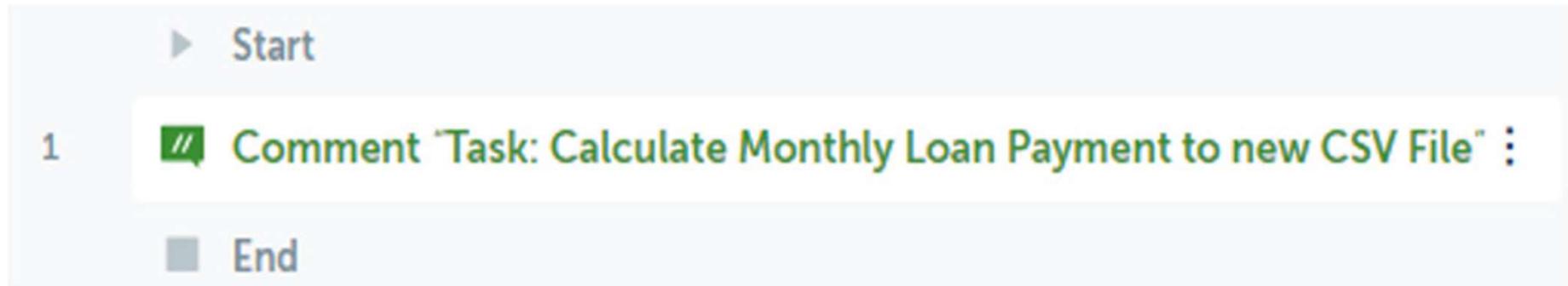


Creating your first bot

- Log in to Control Room.
- Create a new bot and call it lesson 5 - FirstBot in the \Bot\ folder.
- Add a Comment action as line 1; we will use this as our bot description comment.
- Set the Comment property's text as "Task: Calculate Monthly Loan Payment to new CSV File".

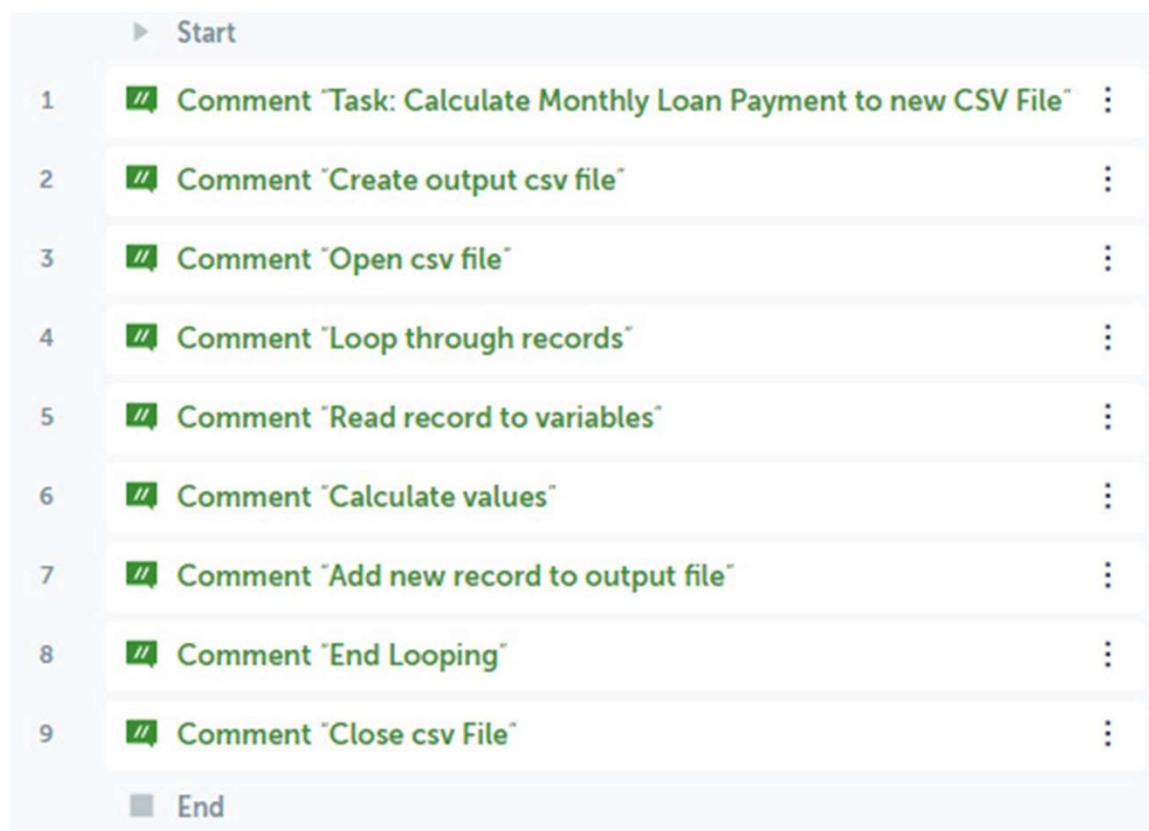
Creating your first bot

- Click on Save. The development interface should look like this:



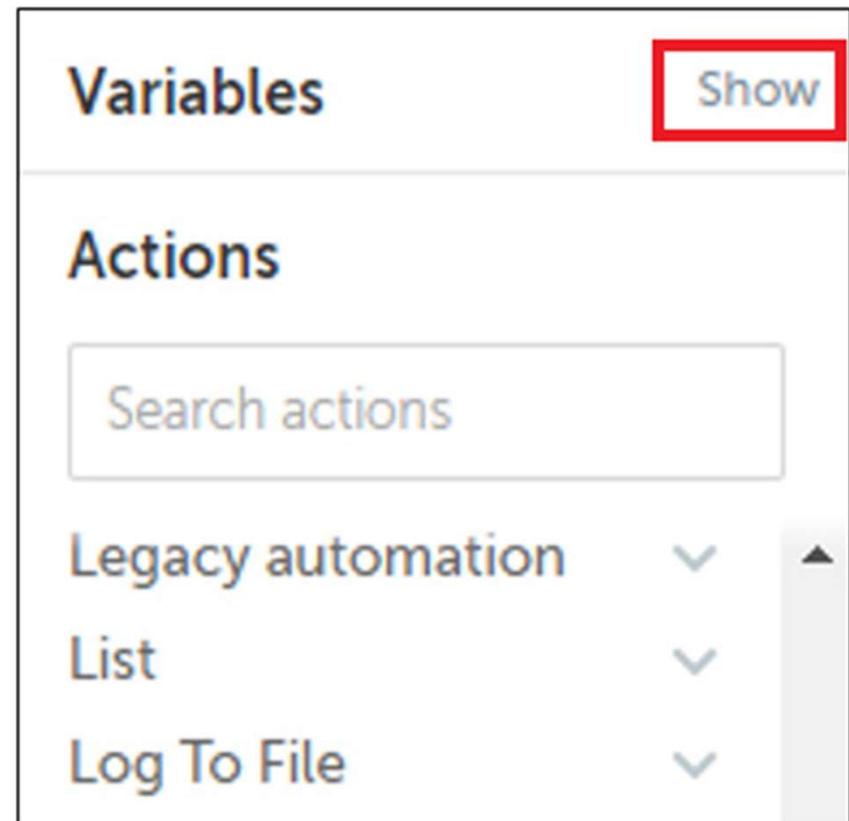
Creating your first bot

- Add a new Comment action as "Close csv file" on line 9 and click on Save.
- Your bot should look something like this:



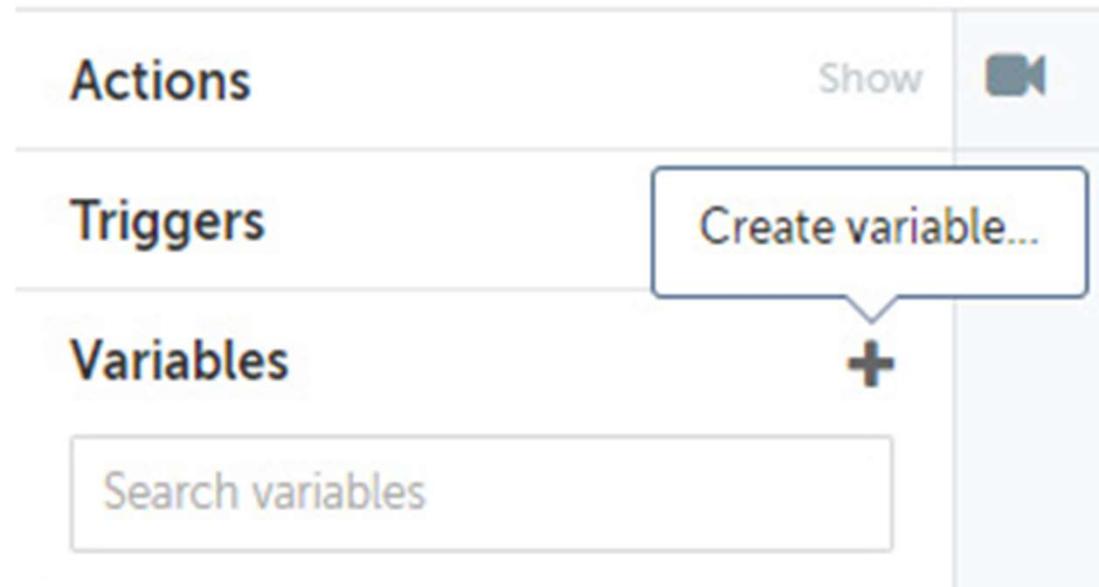
Creating bot variables

- Click on the Show option for the Variables tab from the option pane on the left



Creating bot variables

- Click on the + icon to create a variable:



Creating bot variables

- The Create variable dialog will appear.
- Give your new variable the name strRef, set it as a String type, and then click on Create:

Create variable

Name
strRef

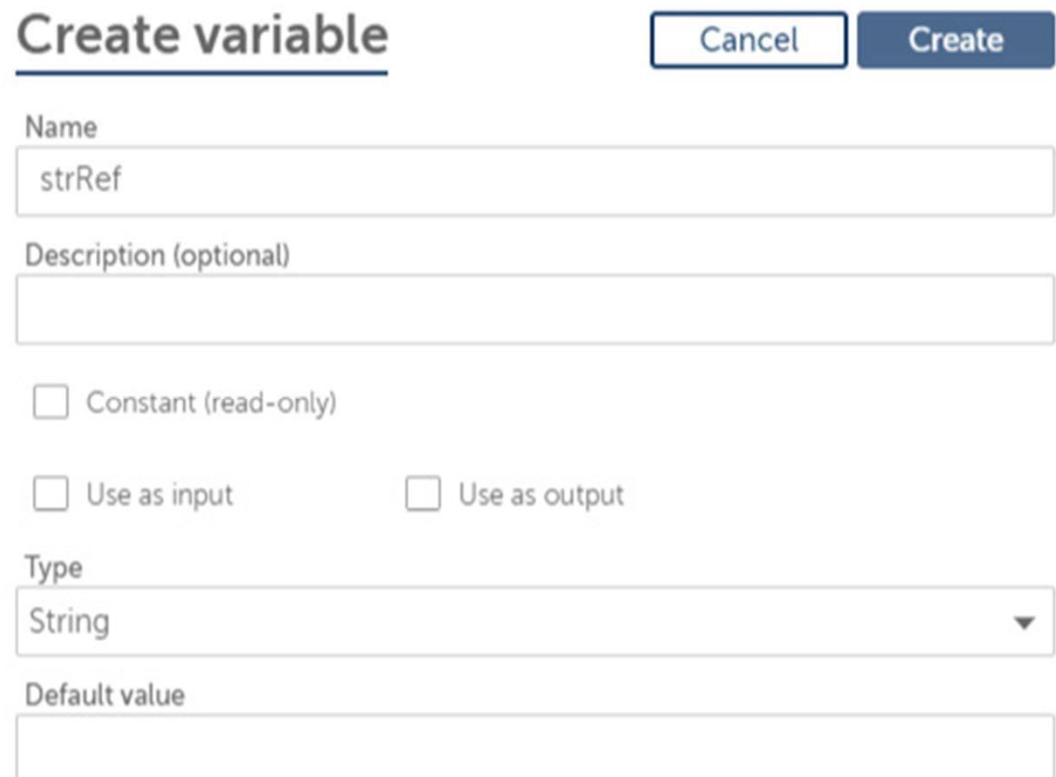
Description (optional)

Constant (read-only)

Use as input Use as output

Type
String

Default value



Creating bot variables

- Create another new variable named recLoan as the Record type.
- You should now have created the following seven variables:

Variables	
Search variables	
	User-defined ^
”	strRef :
#	numAmount :
#	numYears :
#	numInterest :
#	numMonthly :
■	recLoan :
”	strMonthly :

Creating and reading a CSV file

The screenshot shows the Microsoft Power Automate interface with the 'List' tab selected. On the left, the 'Actions' pane is open, displaying various actions such as IQ Bot, JavaScript, Legacy automation, List, Log To File (with 'Log to file' selected), Loop, Message box, Microsoft LUIS NLP (Beta), Mouse, Number, NumberUtils, OCR, and Office 365 Calendar. Below these are sections for 'Triggers' and 'Variables'. The main workspace shows a flow starting with a 'Start' trigger, followed by ten steps: 1. Comment 'Task: Calculate Monthly Loan Payment to new CSV File', 2. Comment 'Create output csv file', 3. Log to file (which is highlighted in blue), 4. Comment 'Open csv file', 5. Comment 'Loop through records', 6. Comment 'Read record to variables', 7. Comment 'Calculate values', 8. Comment 'Add new record to output file', 9. Comment 'End Looping', and 10. Comment 'Close csv File'. There is also a placeholder 'Drag a trigger here...' at the top right.

Creating and reading a CSV file

Log to file

Logs any text into a file

File path

“ C:\Hands-On-RPA-with-AA-Sample-Data

(x)

Browse...

Enter text to log

“ Reference,Monthly Amount

(x)

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

ANSI

▼

Opening and closing a CSV file

	▶ Start	
1	〃 Comment 'Task: Calculate Monthly Loan Payment to new CSV File'	⋮
2	〃 Comment 'Create output csv file'	⋮
3	log Log to file 'Reference,Monthly Amount' to 'C:\Hands-On-RPA-with-AA-Sample-Data\'	⋮
4	〃 Comment 'Open csv file'	⋮
5	CSV/TXT: Open	⚠ ⋮
6	〃 Comment 'Loop through records'	⋮
7	〃 Comment 'Read record to variables'	⋮
8	〃 Comment 'Calculate values'	⋮
9	〃 Comment 'Add new record to output file'	⋮
10	〃 Comment 'End Looping'	⋮
11	〃 Comment 'Close csv File'	⋮
12	CSV/TXT: Close csv/txt "Default"	⋮
	▀ End	

Opening and closing a CSV file

CSV/TXT: Open

Opens a CSV/TXT file

Session name

InputData1

File path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

[x]

Browse...

Required extensions: ".csv", ".txt", ".tsv"

Contains header

Delimiter

Comma

Tab

Regional list separator

Newline

Other

Specific Delimiter (optional)

[x]

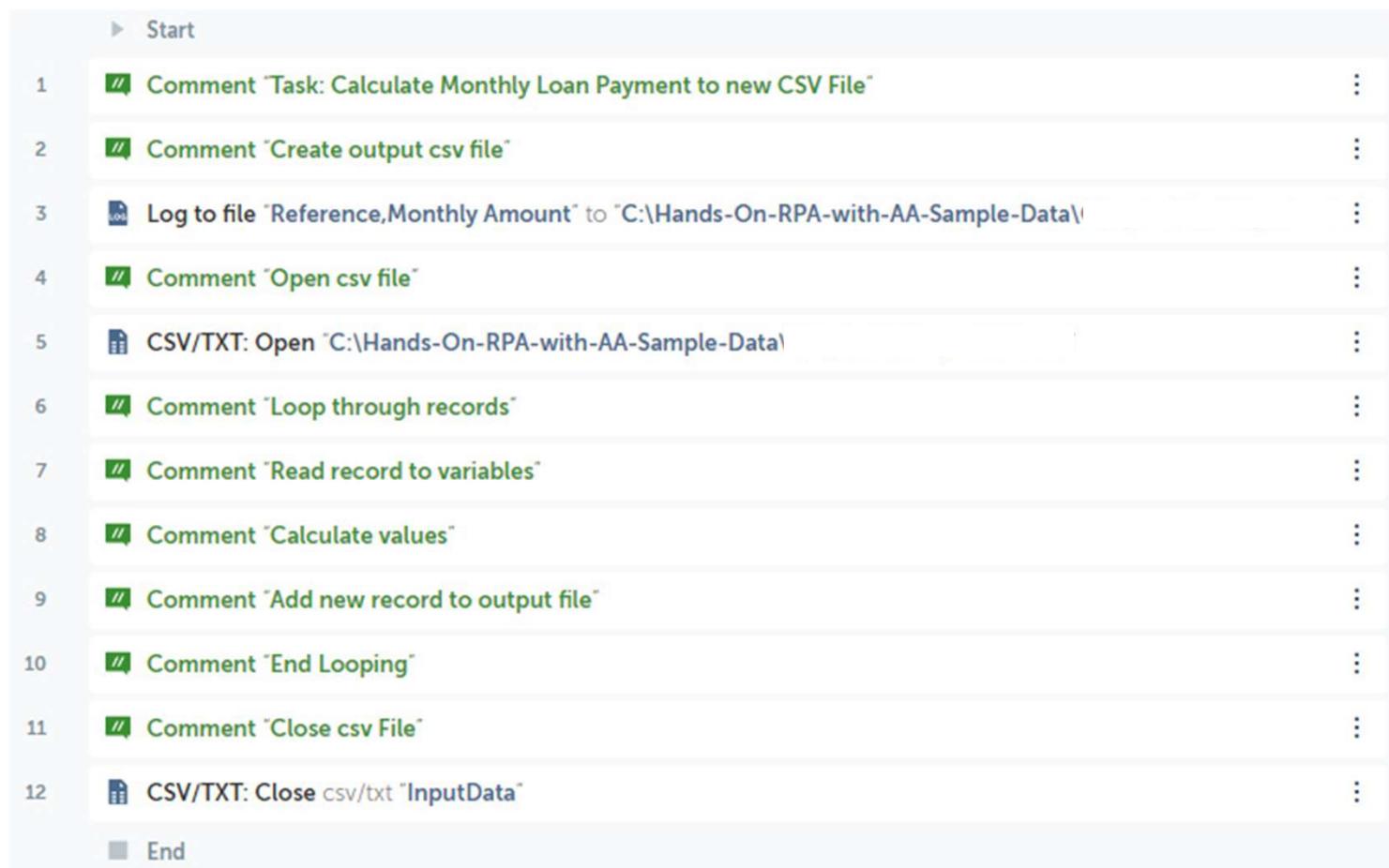
Trim leading spaces

Trim trailing spaces

Encoding

UTF-8

Opening and closing a CSV file



Looping through rows in a CSV file

- Assign the current row to this variable: recLoan – Record
- The action properties dialog should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in CSV/TXT

Iterator for each row in CSV/TXT

Session name

InputData

Assign the current row to this variable

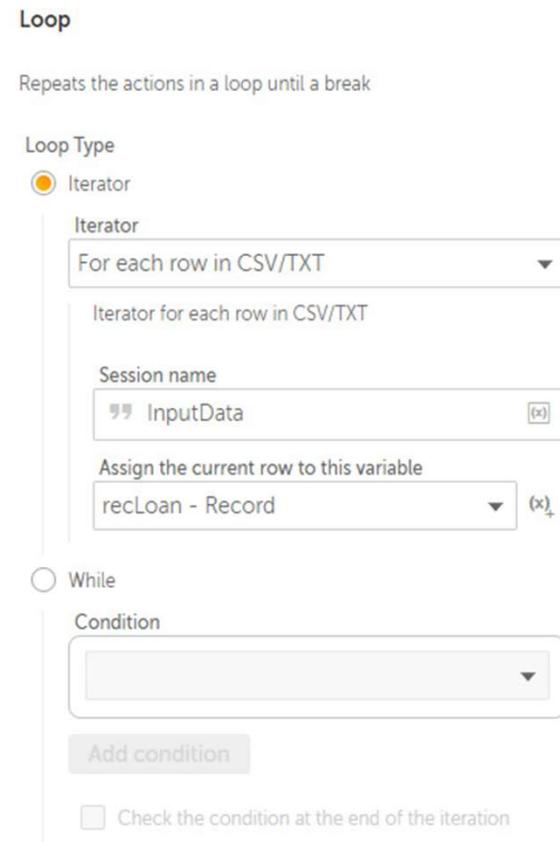
recLoan - Record

While

Condition

Add condition

Check the condition at the end of the iteration



Looping through rows in a CSV file

```
> Start
1  # Comment 'Task: Calculate Monthly Loan Payment to new CSV File'
2  ## Comment 'Create output csv file'
3  ! Log to file 'Reference,Monthly Amount' to 'C:\Hands-On-RPA-with-AA-Sample-Data'
4  ## Comment 'Open csv file'
5  ! CSV/TXT: Open 'C:\Hands-On-RPA-with-AA-Sample-Data'
6  ## Comment 'Loop through records'
7  -> Loop for each row in csv/txt
8    ## Comment 'Read record to variables'
9    ## Comment 'Calculate values'
10   ## Comment 'Add new record to output file'
11   ## Comment 'End Looping'
12   ## Comment 'Close csv File'
13   ! CSV/TXT: Close csv/txt 'InputData'
      End
```

Looping through rows in a CSV file

- Select the destination string variable: strRef - String
- The properties should look like the following screenshot:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

” \$recLoan[0]\$

(x)

Select the destination string variable

strRef - String

▼

(x)

Looping through rows in a CSV file

- Drag the Number: Assign action just below line 9, ensuring it is within the Loop action on line 7.
- Set the following properties for the Number: Assign action on line 10:Select the source string variable/value: \$recLoan[1]\$
- Select the destination number variable: numAmount - Number
- The properties should look like the following screenshot:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$recLoan[1]\$

Specify value to assign to number

Select the destination number variable

numAmount - Number

(x) +

Looping through rows in a CSV file

- Set the following properties for the Number: Assign action on line 11:Select the source string variable/value: \$recLoan[2]\$
- Select the destination number variable: numYears - Number
- The properties should look like the following screenshot:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$recLoan[2]\$ (x)

Specify value to assign to number

Select the destination number variable

numYears - Number ▼ (x) +

Looping through rows in a CSV file

- Set the following properties for the Number: Assign action on line 12:
- Select the source string variable/value: \$recLoan[3]\$
- Select the destination number variable: numInterest - Number
- The properties should look like the following screenshot:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$recLoan[3]\$

Specify value to assign to number

Select the destination number variable

numInterest - Number



Looping through rows in a CSV file

```
> Start
1  // Comment 'Task: Calculate Monthly Loan Payment to new CSV File'
2  // Comment 'Create output csv file'
3  ↗ Log to file 'Reference,Monthly Amount' to 'C:\Hands-On-RPA-with-AA-Sample-Data'
4  // Comment 'Open csv file'
5  ↗ CSV/TXT: Open 'C:\Hands-On-RPA-with-AA-Sample-Data\' 
6  // Comment 'Loop through records'
7  ↘ Loop for each row in csv/txt
8  // Comment 'Read record to variables'
9  # String: Assign $recLoan[0]$ to $strRef$
10 # Number: Assign $recLoan[1]$ to $numAmount$
11 # Number: Assign $recLoan[2]$ to $numYears$
12 # Number: Assign $recLoan[3]$ to $numInterest$
13 // Comment 'Calculate values'
14 // Comment 'Add new record to output file'
15 // Comment 'End Looping'
16 // Comment 'Close csv File'
17 ↗ CSV/TXT: Close csv/txt "InputData"
█ End
```

Performing basic arithmetic calculations

- Set the following properties for the Number: Assign action on line 14:Select the source string variable/value: (\$numAmount\$ + \$numInterest\$)/(12 * \$numYears\$)
- Select the destination number variable: numMonthly - Number
- The action properties dialog should look like this:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

(\$numAmount\$ + \$numInterest\$) / (12 * \$numYears\$) (x)

Specify value to assign to number

Select the destination number variable

numMonthly - Number ▼ (x) +

Appending records to a CSV file

- Set the following properties for the String: Assign action on line 16:
- Enter a number: \$numMonthly\$
- Enter number of digits after decimal: 2
- Assign the output to variable: strMonthly - String
- The action properties dialog should look like this:

Number: To string

Converts a user specified number to a string

Enter a number

\$numMonthly\$ (x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

2 (x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

strMonthly - String (x) +

Appending records to a CSV file

Log to file

Logs any text into a file

File path

“ C:\Hands-On-RPA-with-AA-Sample-Data\

(x)

Browse...

Enter text to log

“ \$strRef\$, \$strMonthly\$

(x)

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

ANSI

▼

Appending records to a CSV file

```
> Start
1  # Comment 'Task: Calculate Monthly Loan Payment to new CSV File'
2  # Comment 'Create output csv file'
3  Log to file 'Reference,Monthly Amount' to 'C:\Hands-On-RPA-with-AA-Sample-Data\
4  # Comment 'Open csv file'
5  CSV/TXT: Open 'C:\Hands-On-RPA-with-AA-Sample-Data\
6  # Comment 'Loop through records'
7  -  Loop for each row in csv/btx
8    # Comment 'Read record to variables'
9    String: Assign $recLoan[0]$ to $strRef$
10   # Number: Assign $recLoan[1]$ to $numAmount$
11   # Number: Assign $recLoan[2]$ to $numYears$
12   # Number: Assign $recLoan[3]$ to $numInterest$
13   # Comment 'Calculate values'
14   # Number: Assign '$numAmount$ + $numInte...' to $numMonthly$
15   # Comment 'Add new record to output file'
16   # Number: To string convert $numMonthly$ to a string datatype and assign output to $strMonthly$
17   Log to file '$strRef$', $strMonthly$ to 'C:\Hands-On-RPA-with-AA-Sample-Data\
18   # Comment 'End Looping'
19   # Comment 'Close csv File'
20   CSV/TXT: Close csv/btx 'InputData'
  End
```

Appending records to a CSV file

- You have done some great work, Go ahead and run your first bot.
- Your first bot should create the output CSV file containing the calculated Monthly Payments.
- The output file should look like this in Excel:

	A	B
1	Reference	Monthly Amount
2	508-001	176.66
3	523-679	147.22
4	524-602	110.41
5	534-001	265.00
6	302-170	298.33
7	230-614	53.00
8	550-205	176.66

Summary

- You have built your first bot using Automation Anywhere.
- You are now a lot closer to becoming an RPA developer, having gained some valuable knowledge with the help of a real-life business process.
- You should be very comfortable with using actions to make your bot perform tasks.
- We have already looked at comma-separated files, in other words, reading, creating, and appending.

Introducing Variables



Introducing Variables

In this lesson, we will cover the following topics:

- Working with different variable types
- Using message boxes and prompts
- Converting data types

Working with different variable types

- There are other data types available in and each type is represented by a specific icon.
- Here, you can see the icons for each type of variable:

 Boolean	 List
 Credential	 Number
 Datetime	 Record
 Dictionary	 String
 File	 Table
 Form	 Window

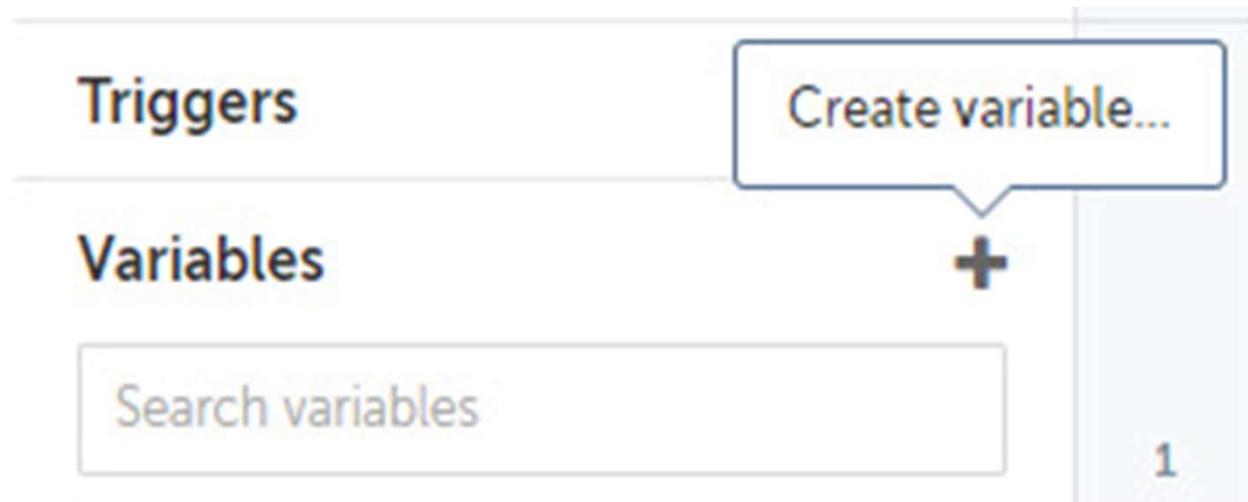
Working with different variable types

- Like most development platforms, it is good practice to use a naming convention when creating variable names.
- This course will be using the following variable naming notation:

Data Type	Variable name Prefix
Number	num
String	str
Boolean	bln
Date\Time	dte
Record	rec
List	lst
Table	tbl
Dictionary	dct

Using the String variable type

- Log in to the Control Room.
- Create a new bot and call it lesson 6 - Variables in the \Bot\ folder.
- Expand the Variables pane from the options on the left and select + to create a new variable:



Using the String variable type

- The Create variable dialog will appear.
- Call this variable strFirstName and set it as a String type.
- Once the details are entered, click on Create, The dialog should look like this:

Create variable

Name
strFirstName
Max characters = 50

Description (optional)
Max characters = 255

Use as input
 Use as output
 Constant (read-only)

Type
String

Default value

Using the String variable type

- Create another new variable named strFullscreen as a String type.
- Your variable list should appear as follows:

The screenshot shows a user interface for managing variables. At the top, there is a header labeled "Variables" with a plus sign icon for adding new variables. Below the header is a search bar with the placeholder text "Search variables". The main area displays a list of variables under the heading "User-defined". The list contains three entries, each consisting of a double quotes icon, a variable name, and a more options icon. The variable names are "strFirstName", "strFullscreen", and "strSurname".

User-defined
” strFirstName
” strFullscreen
” strSurname

Using the String variable type

- Add a new Comment action as "-----" on line 4 and click on Save.
- Your bot should now look like this:



Using the String variable type

- Set the following properties for the String: Assign action on line 2:
- Select the source string variable(s)/ value (optional): John
- Select the destination string variable: strFirstName - String
- The action properties dialog should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

“ John

(x)

Select the destination string variable

strFirstName - String

▼

(x)
+

Using the String variable type

- Set the following properties for the String: Assign action on line 3:
- Select the source string variable(s)/ value (optional): Smith
- Select the destination string variable: strSurname - String
- The action properties dialog should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

“ Smith (x)

Select the destination string variable

strSurname - String ▼ (x)

Using the String variable type

- Set the following properties for the String: Assign action on line 5:
- Select the source string variable(s)/ value (optional): \$strFirstName\$ \$strSurname\$
- Select the destination string variable: strFullname - String
- The action properties dialog should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s) / value (optional)

“ ” \$strFirstName\$ \$strSurname\$

Select the destination string variable

strFullname - String

Using the String variable type

- Set the following properties for the Message box action on line 7:
- Enter the message box window title: Merged variables
- Enter the message to display: \$strFullscreen\$
- The action properties dialog should look like this:

Message box

Displays a message box

Enter the message box window title

” Merged Variables

Enter the message to display

” \$strFullscreen\$

Using the String variable type

- Click on Save, The development interface should look something like this:

1	// Comment "String Variables"	:
2	" String: Assign "John" to \$strFirstName\$:
3	" String: Assign "Smith" to \$strSurname\$:
4	// Comment "Merge variables"	:
5	" String: Assign "\$strFirstName\$ \$strSurname\$" to \$strFullname\$:
6	// Comment "Show Output"	:
7	Message box \$strFullname\$:
8	// Comment "-----"	:

Using the Datetime variable type

Create variable

Cancel

Create

Name

dteChristmas

Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

Type

Datetime

Default value (optional)

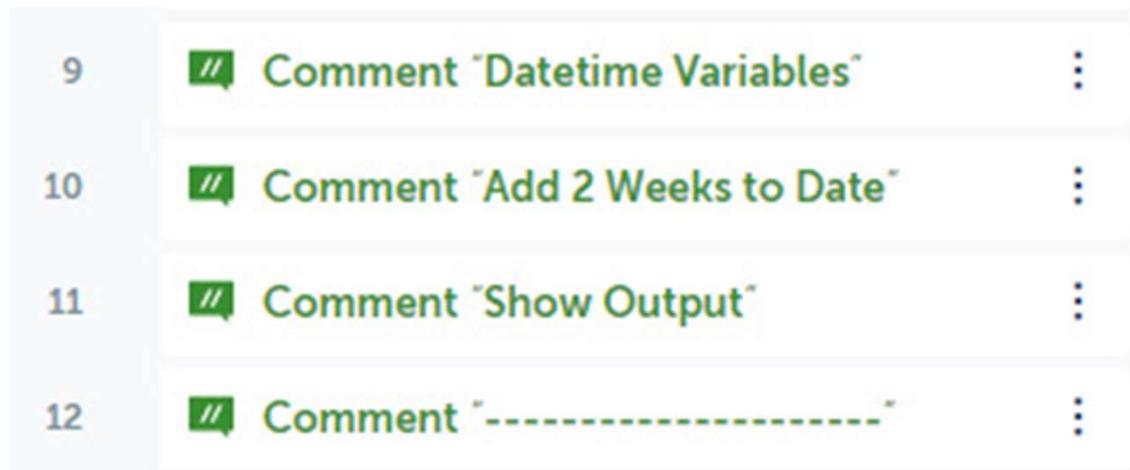
12/25/2019

12:00 AM

BST (UTC+1:00) London, Europe

Using the Datetime variable type

- Add a new Comment action as "-----" on line 12 and click on Save.
- Your bot should look like this:



Using the Datetime variable type

Datetime: Add

Adds a specified time unit to a given date and time

Source date and time variable

dteChristmas - Datetime ▾ (x) 

Time value to add

2 

Maximum value for addition is "69948627" hours

Time unit to add

Weeks ▾

Assign the output to a variable

dteChristmasPlus2Weeks - Datetime ▾ (x) 

Using the Datetime variable type

Datetime: To string

Converts a datetime value to a string value and assigns it to a string variable

Source date and time variable

dteChristmasPlus2Weeks - Datetime ▾ (x)₊

Select date time format

Formats

▼

Custom format

▼ DD MMM YYYY (x)₊

Assign the output to a variable

strDate - String ▾ (x)₊

Using the Datetime variable type

- Set the following properties for the Message box action on line 14:
Enter the message box window title: Datetime variables
- Enter the message to display: \$strDate\$
- The action properties dialog should look like this:

Message box

Displays a message box

Enter the message box window title

” Datetime Variables

Enter the message to display

” \$strDate\$

Using the Datetime variable type

- 9 Comment "Datetime Variables" : ...
- 10 Comment "Add 2 Weeks to Date" : ...
- 11 Datetime: Add 2 Weeks to \$dteChristmas\$ and assign result to \$dteChristmasPlus2Weeks\$: ...
- 12 Comment "Show Output" : ...
- 13 Datetime: To string Convert \$dteChristmasPlus2Weeks\$ and assign result to \$strDate\$: ...
- 14 Message box \$strDate\$: ...
- 15 Comment "-----" : ...

Using the Boolean variable type

- The Create variable dialog will appear. Set the following values:

Name: blnLeapYear

Type: Boolean

Default value: True

The dialog should look like this:

Create variable Cancel **Create**

Name
blnLeapYear
Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

Type
Boolean

Default value
False **True**

Using the Boolean variable type

- Add another Comment action as "-----" as line 20 and click on Save.
- Your bot should look like this:

16	// Comment "Boolean Variables"	:
17	// Comment "Assign Boolean Value"	:
18	// Comment "Invert Boolean Value"	:
19	// Comment "Show Output"	:
20	// Comment "-----"	:

Using the Boolean variable type

- Set the following properties for the Boolean: Assign action on line 18:
- Select the source Boolean variable/ value: Constant values
- Constant values: True
- Select the destination Boolean variable: blnLeapYear - Boolean
- The action properties should look like this:

Boolean: Assign

Assigns the source boolean variable's value or the user defined value to the destination boolean variable

Select the source boolean variable/ value

Constant values

True

False

Variable value

Select the destination boolean variable

blnLeapYear - Boolean



Using the Boolean variable type

- Set the following properties for the Boolean: Invert action on line 20:

Select the Boolean variable to be inverted: Variable

Value: \$blnLeapYear\$

Assign the output: blnLeapYear - Boolean

The Boolean: Invert action properties should look like this:

Boolean: Invert

Inverts a boolean variable's value i.e. converts True to False and False to True and assigns the output to a variable (same or different)

Select the boolean variable to be inverted

False True Variable

 \$blnLeapYear\$ (x)

Assign the output to

blnLeapYear - Boolean (x)

Using the Boolean variable type

- Set the following properties for the Boolean: To string action on line 22:

Select Boolean variable:

blnLeapYear - Boolean

Select the string variable to store the result: strLeapYear - String

The action properties should look like this:

Boolean: To string

Converts a boolean value to string and assigns it to a string variable

Select boolean variable

blnLeapYear - Boolean

Select the string variable to store the result

strLeapYear - String

Using the Boolean variable type

- Set the following properties for the Message box action on line 23:
- Enter the message box window title: Boolean variables
- Enter the message to display: \$strLeapYear\$
- The action properties dialog should look like this:

Message box

Displays a message box

Enter the message box window title

” Boolean Variables

Enter the message to display

” \$strLeapYear\$

Using the Boolean variable type

16	#[Comment "Boolean Variables"	:
17	#[Comment "Assign Boolean Value"	:
18	FLAG Boolean: Assign True to \$blnLeapYear\$:
19	#[Comment "Invert Boolean Value"	:
20	FLAG Boolean: Invert value of boolean variable \$blnLeapYear\$ and assign result to \$bl...	:
21	#[Comment "Show Output"	:
22	FLAG Boolean: To string \$blnLeapYear\$ and assign result to a \$strLeapYear\$:
23	INFO Message box \$strLeapYear\$:
24	#[Comment "-----"	:

Using the Number variable type

- Add another Comment action as "-----" as line 29 and click on Save. Your bot should look like this:

25	#[Comment "Number Variables"	:
26	#[Comment "Assign Random Value"	:
27	#[Comment "Apply Formula"	:
28	#[Comment "Show Output"	:
29	#[Comment "-----"	:

Using the Number variable type

- Set the following properties for the Number: Random action on line 27:
- Beginning of range: 1
- End of range: 100
- Save the outcome to a number variable: numRandom - Number
- The action properties should look like this:

Number: Random

Assigns a random number to a number variable

Beginning of range:

1

(x)

Accepts decimal and negative value.

End of range:

100

(x)

Must be larger than beginning of range.

Save the outcome to a number variable

numRandom - Number

▼

(x)

+

Using the Number variable type

- Set the following properties for the Number: Assign action on line 29:
- Select the source string variable: `($numRandom$/2) + 25`
- Select the destination number variable: numResult - Number
- The action properties should look like this:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

`($numRandom$/2) + 25` (x)

Specify value to assign to number

Select the destination number variable

`numResult - Number` ▼ (x) +

Using the Number variable type

- Set the following properties for the Number: To string action on line 31:
- Enter a number: \$numResult\$
- Enter number of digits after decimal: 2
- Assign the output to variable: strResult - String
- The action properties should look like this:

Number: To string

Converts a user specified number to a string

Enter a number

\$numResult\$

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

2

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

strResult - String

Using the Number variable type

- Set the following properties for the Message box action on line 32:
- Enter the message box window title: Number variables
- Enter the message to display: \$strResult\$
- The action properties dialog should look like this:

Message box

Displays a message box

Enter the message box window title

” Number Variables

Enter the message to display

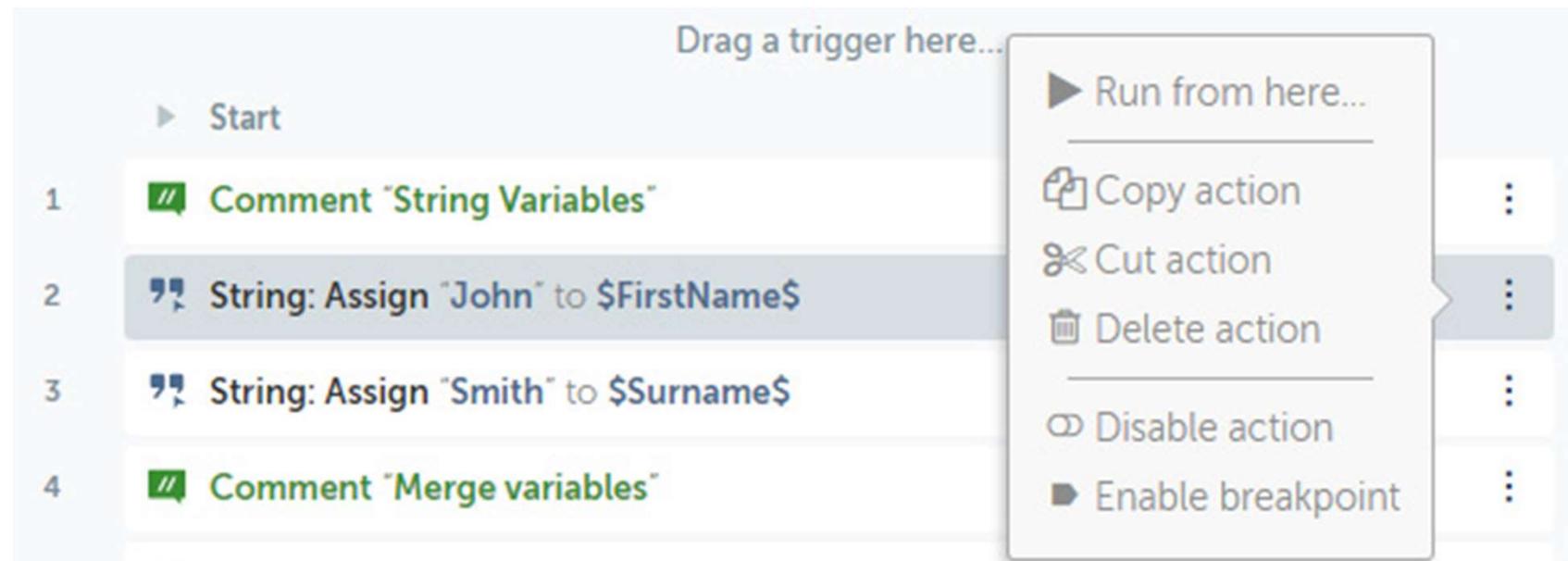
” \$strResult\$

Using the Number variable type

25	Comment "Number Variables"	⋮
26	Comment "Assign Random Value"	⋮
27	Number: Random Assign a random number from beginning of range 1 to end of range 100 to \$numRandom\$	⋮
28	Comment "Apply Formula"	⋮
29	Number: Assign <code>"(\$numRandom\$/2) + 25"</code> to \$numResult\$	⋮
30	Comment "Show Output"	⋮
31	Number: To string convert \$numResult\$ to a string datatype and assign output to \$strResult\$	⋮
32	Message box \$strResult\$	⋮
33	Comment "-----"	⋮

Using message boxes and prompts

- Select the options menu for line 2:



Using message boxes and prompts

- Set the following properties for the Prompt: For value action on line 3:
- Prompt window caption: Prompt for String
- Prompt message: Enter Firstname:
- Assign the value to a variable: strFirstName - String
- The action properties should look like this:

Prompt: For value

Prompts user for entering a value

Prompt window caption

” Prompt for String (x)

Prompt message

” Enter Firstname: (x)

Mask keystroke

Assign the value to a variable

strFirstName - String ▼ (x) +

Using message boxes and prompts

- 1 Comment "String Variables" : ...
- 2 String: Assign "John" to \$strFirstName\$ ∅ :
- 3 Prompt: For value : ...
- 4 String: Assign "Smith" to \$strSurname\$ ∅ :
- 5 Prompt: For value : ...
- 6 Comment "Merge variables" : ...
- 7 String: Assign "\$strFirstName\$ \$strSurname\$" to \$strfullname\$: ...
- 8 Comment "Show Output" : ...
- 9 Message box \$strfullname\$: ...
- 10 Comment "-----" : ...

Converting data types

Let's look at the data conversions performed by the current bot. The first data conversion the bot performs is converting a **Datetime** data type to a **String** data type. Take a look at line **15** from the development interface:

```
15  📅 Datetime: To string Convert $dteChristmasPlus2Weeks$ and assign result to $strDate$ :
```

- Converting a Datetime data type to String

The next data conversion performed is from a **Boolean** data type to a **String** data type. This is demonstrated on line **24** of the development interface:

```
24  🏟 Boolean: To string $blnLeapYear$ and assign result to a $strLeapYear$ :
```

- Converting a Boolean data type to String

Finally, the bot converts a **Number** data type to a **String** data type. This is on line **33** of the development interface:

```
33  # Number: To string convert $numResult$ to a string datatype and assign output to $strResul... :
```

Summary

- To recap, you have learned how to create different types of variables from numbers, Booleans, and dates.
- Your knowledge has been expanded by assigning values to these variables as well as performing calculations.
- We also explored the reasons for using, and how to convert, different data types.
- Having also created prompts and message boxes, you are well on your way to start learning some more exciting Automation Anywhere packages and actions.

Interacting with Applications



Interacting with Applications

In this lesson, we will cover the following topics:

- Automating web applications
- Automating desktop applications
- Simulating keystrokes

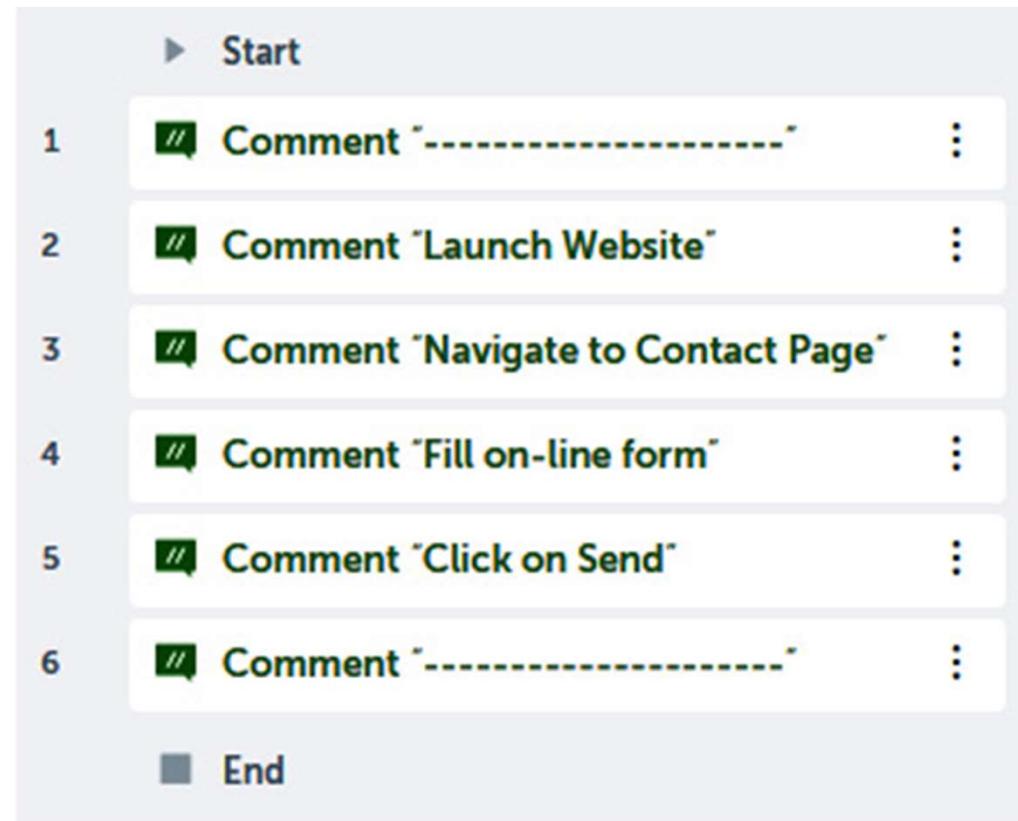
Interacting with Applications

- By the end of this lesson, you will have the skills needed to build bots that can launch applications, navigate through various application interfaces, interact with buttons and checkboxes, and read and enter data.
- This lesson will be using the following packages:

<input checked="" type="checkbox"/> Application	▼
<input checked="" type="checkbox"/> Browser	▼
<input checked="" type="checkbox"/> Comment	▼
<input checked="" type="checkbox"/> Message box	▼
<input checked="" type="checkbox"/> Recorder	▼
<input checked="" type="checkbox"/> Simulate keystrokes	▼
<input checked="" type="checkbox"/> Window	▼

Automating web applications

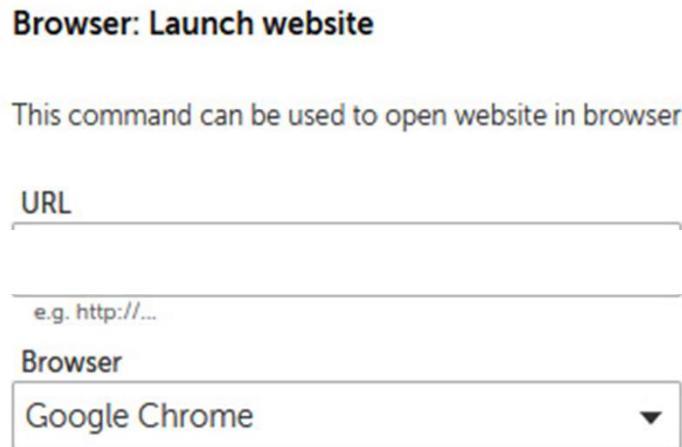
- Add a new Comment action as "-----" on line 6 and click on Save.
- Your bot should look like this:



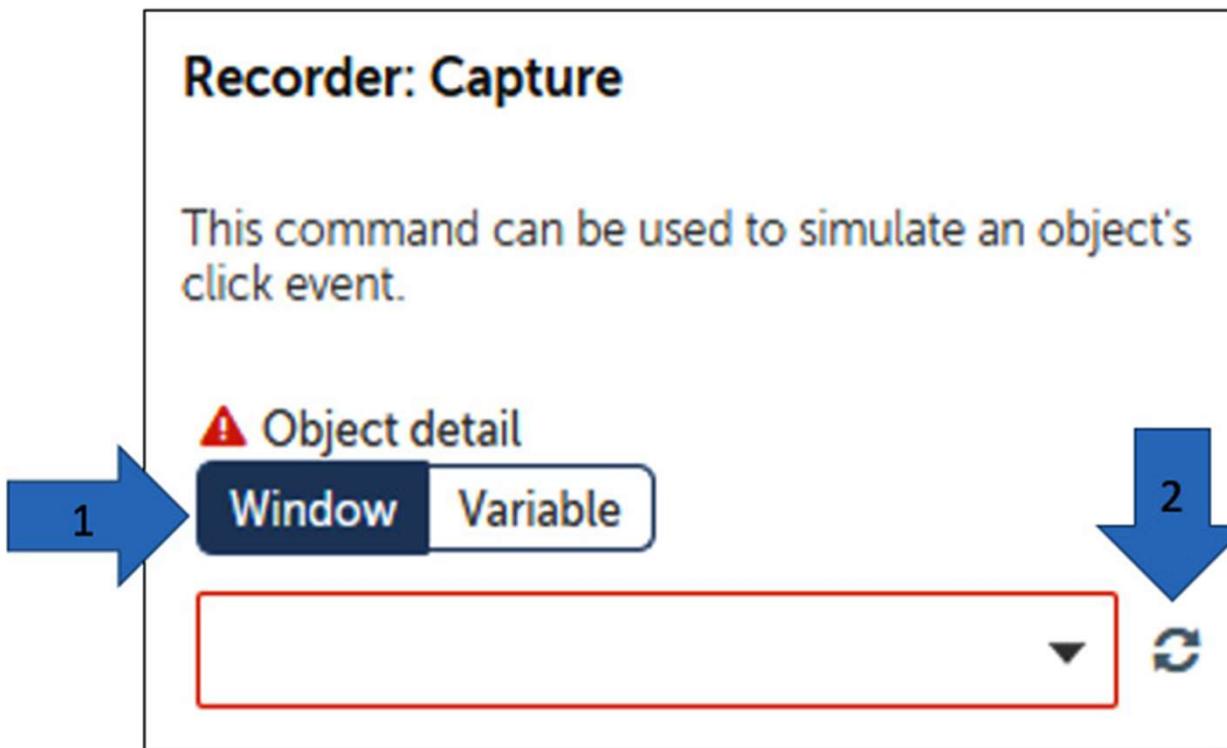
Automating web applications

Set the following properties for the Browser: Launch website action on line 3:

- URL: <http://fenago.net/>
- Browser: Google ChromeThe Browser: Launch website action properties should look like this:



Automating web applications



Automating web applications

- The drop-down list will show all windows that are currently open.
- Select Home - Google Chrome.
- The Recorder: Capture action properties should look like this:

Recorder: Capture

This command can be used to simulate an object's click event.

Object detail

Window Variable

Home - Google Chrome

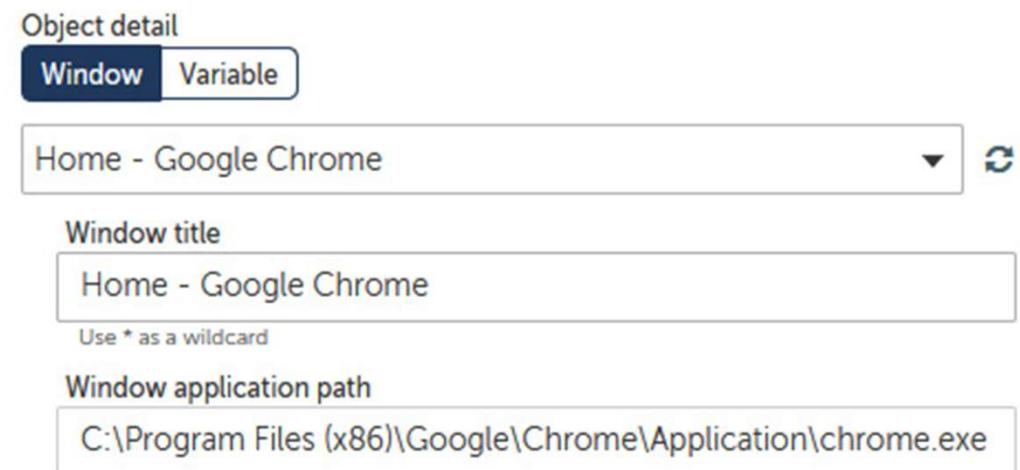
Window title

Home - Google Chrome

Use * as a wildcard

Window application path

C:\Program Files (x86)\Google\Chrome\Application\chrome.exe



Automating web applications



The Bots Are Coming...



Are you ready for the Digital Workforce?

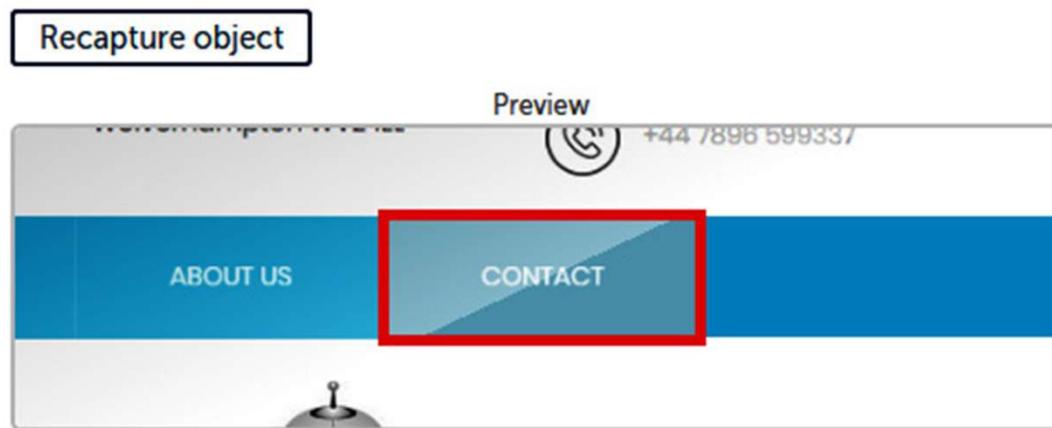
Robotic Processing Automation (RPA) is currently one of the fastest growing technologies around the world. Automation of repetitive, tedious and boring tasks can help your business get most out of your workforce.

What is Robotic Process Automation?

Commonly known as RPA it enables organisations and businesses to 'teach' computers to execute repetitive and time consuming tasks. Being a software based tool, it is easily able to connect and integrate with most desktop applications as well as integration with back-end systems via API's.

Automating web applications

- Once the correct object has been identified with the red border, click to select it.
- Once clicked, the bot will capture all the attributes it needs.
- You will see the selected object in the preview section of the properties, It should look something like this:



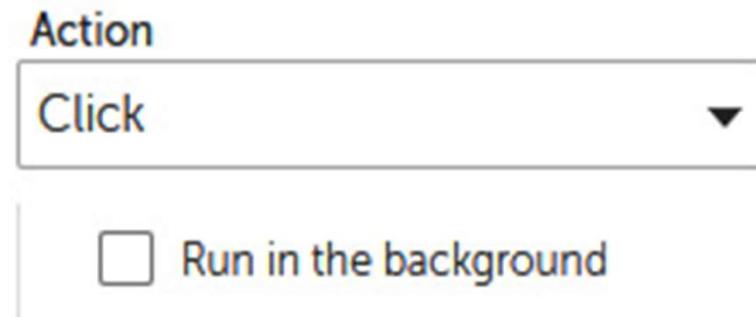
Automating web applications

- The next step is to look at the attributes used to identify the object.
- If we collapse the Object properties list, you will see only the checked attributes used to identify the object.
- The properties should look like this:

Object properties (8 of 48)	
Name	Value
Control Type	LINK
Technology Type	HTML
HTML Tag	” A
DOMXPath	” /html/body/div[4]/div[3]/div[1]/div[1]/div[1] ◀ ▶
HTML HasFrame	” false
HTML InnerText	” CONTACT
HTML Href	”
Path	” 5 3 1 1 1 1 1 1 1 1 -1 1

Automating web applications

- Now that the bot has found the CONTACT tab, the next step is to get the bot to click on it.
- This is where we set the Action property, This should be set to Click, as shown here:



Automating web applications

- We have set the first bot interaction with the website.
- The development interface for this step should look like this:

2	Comment 'Launch Website'	:
3	Browser: Launch website	
4	Comment 'Navigate to Contact Page'	:
5	Recorder: Capture Click on link HTML InnerText "CONTACT" in window "Home - Google Chrome"	:

Automating web applications

HOME SERVICES TRAINING ABOUT US CONTACT

Get In Touch With Us

We are more than happy to help however we can so, please do not hesitate to contact us

(location pin)

(phone receiver)

(envelope)

Name *

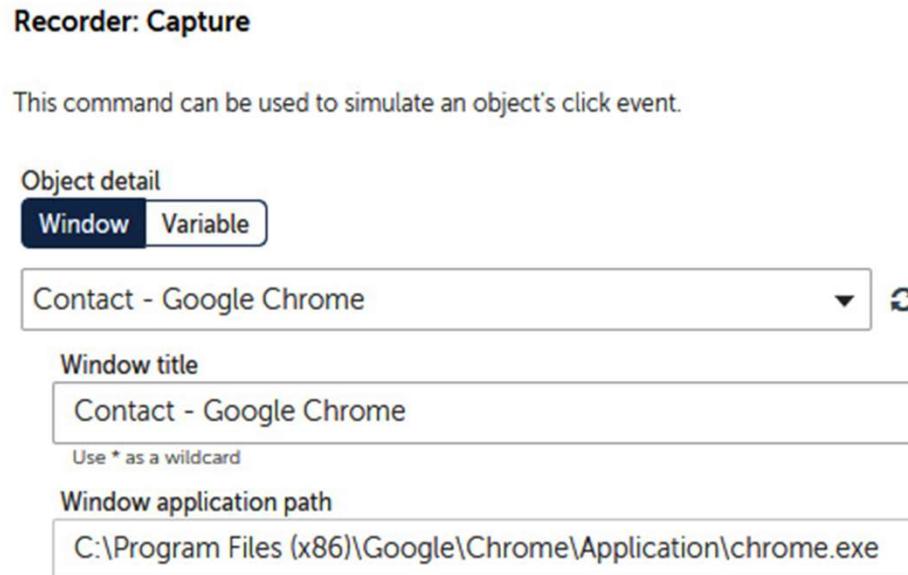
Email *

Message *

Send

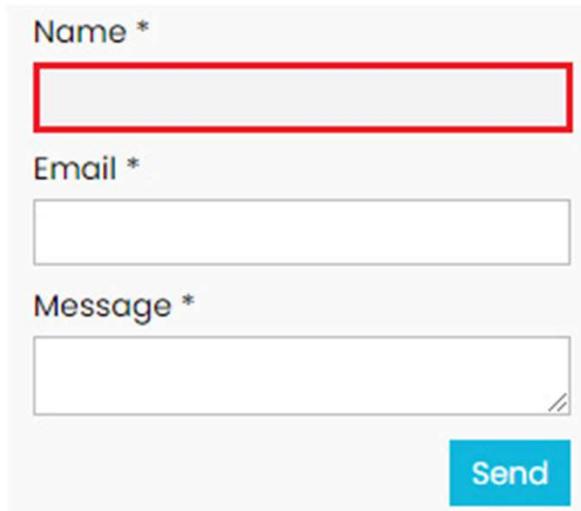
Automating web applications

- Refresh the windows drop-down list and select Contact - Google Chrome.
- The action properties should look like this:



Automating web applications

- When the Contact web page appears, hover the mouse over the Name textbox until it has a red border around it, as shown in the following screenshot:



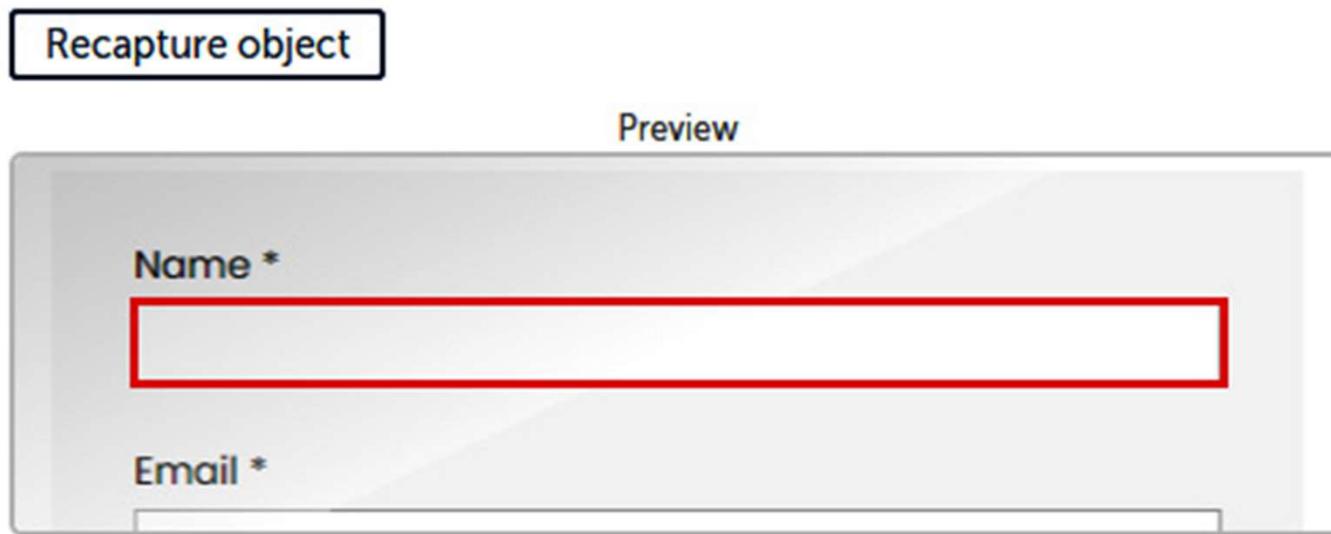
A screenshot of a contact form. It contains three input fields: 'Name *' with a red border, 'Email *', and 'Message *'. Below the fields is a blue 'Send' button.

Name *
Email *
Message *

Send

Automating web applications

- Click in the red border to capture it.
- Once captured, check the preview to ensure that the correct object has been captured, as shown in the following screenshot:



Automating web applications

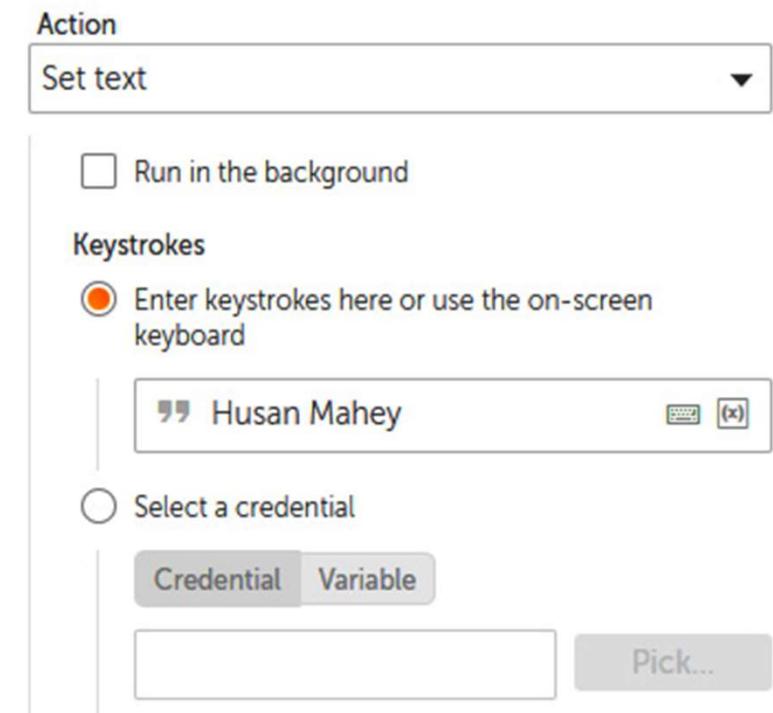
- This time we want to populate the textbox with your name. Set the following properties for the Recorder:
Capture action on line 7:

Action: Set text

Keystrokes: Enter keystrokes here or use the on-screen keyboard

Value: (Enter your name)

The properties should look like the following screenshot:



Automating web applications

- When the Contact web page appears, hover the mouse over the Email textbox until it has a red border around it, as shown in the following screenshot:

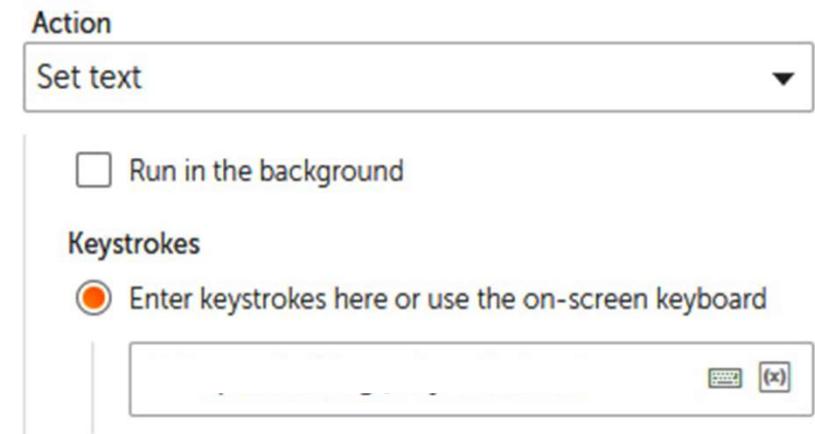
The screenshot shows a contact form with three fields: 'Name *' (empty), 'Email *' (highlighted with a red border), and 'Message *' (empty). A blue 'Send' button is at the bottom right.

Name *	<input type="text"/>
Email *	<input type="text"/>
Message *	<input type="text"/>

Send

Automating web applications

- This time we want to populate the textbox with your email address.
Set the following properties for the Recorder: Capture action on line 8:
- Action: Set text
- Keystrokes: Enter keystrokes here or use the on-screen keyboard
- Value: (Enter your email address)
- The properties should look similar to the following screenshot:



Automating web applications

- When the Contact web page appears, hover the mouse over the Message textbox until it has a red border around it, as shown in the following screenshot:

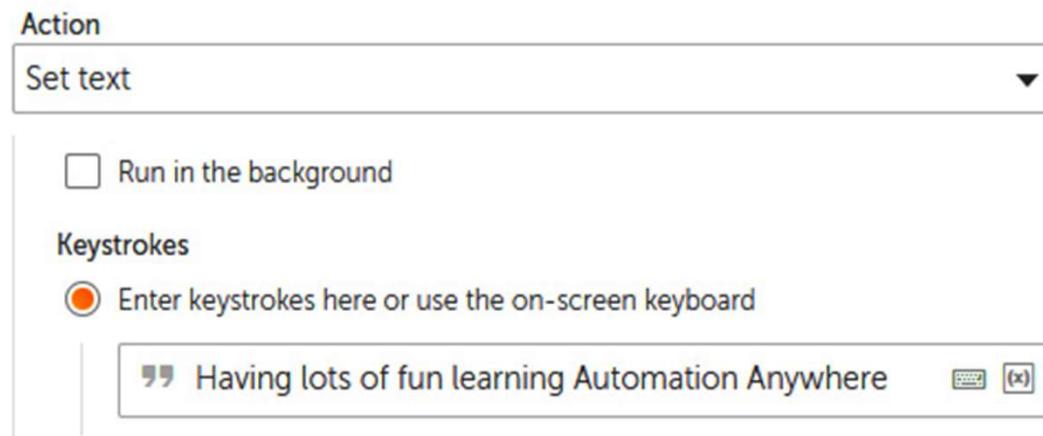
The screenshot shows a contact form with three fields: 'Name *', 'Email *', and 'Message *'. The 'Message' field is highlighted with a red border, indicating it is the current target for the mouse cursor. A blue 'Send' button is located at the bottom right.

Name *	<input type="text"/>
Email *	<input type="text"/>
Message *	<input type="text"/>

Send

Automating web applications

- This time we want to populate the textbox with your email address. Set the following properties for the Recorder: Capture action on line 9:
- Action: Set text
- Keystrokes: Enter keystrokes here or use the on-screen keyboard
- Value: Having lots of fun learning Automation Anywhere
- The properties should look similar to the following screenshot:



Automating web applications

- Click on Save. Your development interface for this section should look like this

6	Comment 'Fill on-line form'	:
7	Recorder: Capture Set text on textbox HTML Name "Name" in window "Contact - Google ..."	:
8	Recorder: Capture Set text on textbox HTML Name "Email" in window "Contact - Google C..."	:
9	Recorder: Capture Set text on textbox HTML Name "Message" in window "Contact - Googl..."	:

Automating web applications

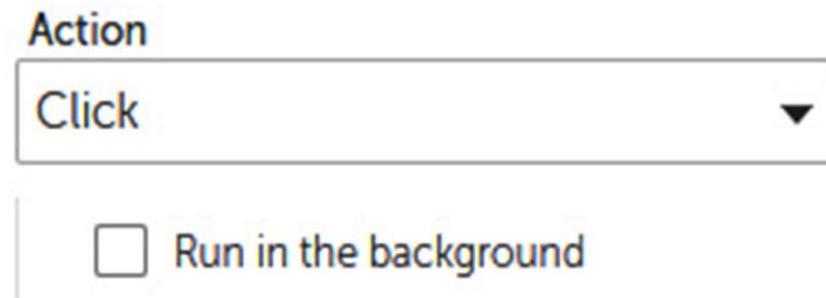
- When the Contact web page appears, hover the mouse over the Send button until it has a red border around it, as shown in the following screenshot:

A screenshot of a contact form with three input fields: Name, Email, and Message, followed by a Send button. The Send button is highlighted with a red border, indicating it is the target for automation.

Name *	<input type="text"/>
Email *	<input type="text"/>
Message *	<input type="text"/>
Send	

Automating web applications

- This time we want the bot to click the button.
- To do this, set the following properties for the Recorder: Capture action on line 11:
- Action: Click
- The properties should look similar to the following screenshot:



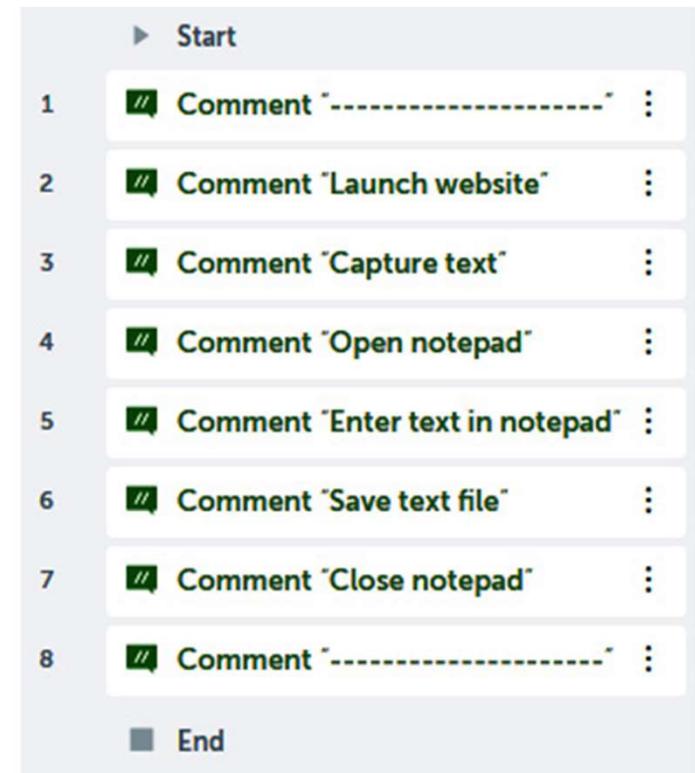
Automating web applications

- Click on Save.
- The development interface for this section should look like this:

10	>// Comment "Click on Send"	:
11	■■■ Recorder: Capture Click on button in window "Contact - Google Chrome"	:
12	✉ Message box "Bot has sent an email successfully"	:
13	// Comment "-----"	:

Automating desktop applications

- Add a new Comment action as "-----" on line 8 and click on Save.
- Your bot should look like this:



Automating desktop applications

- Set the following properties for the Browser: Launch website action on line 3:URL: <http://fenago.net/>
- Browser: Google Browser
- The Browser: Launch website action properties should look like this:

Browser: Launch website

This command can be used to open website in browser

URL

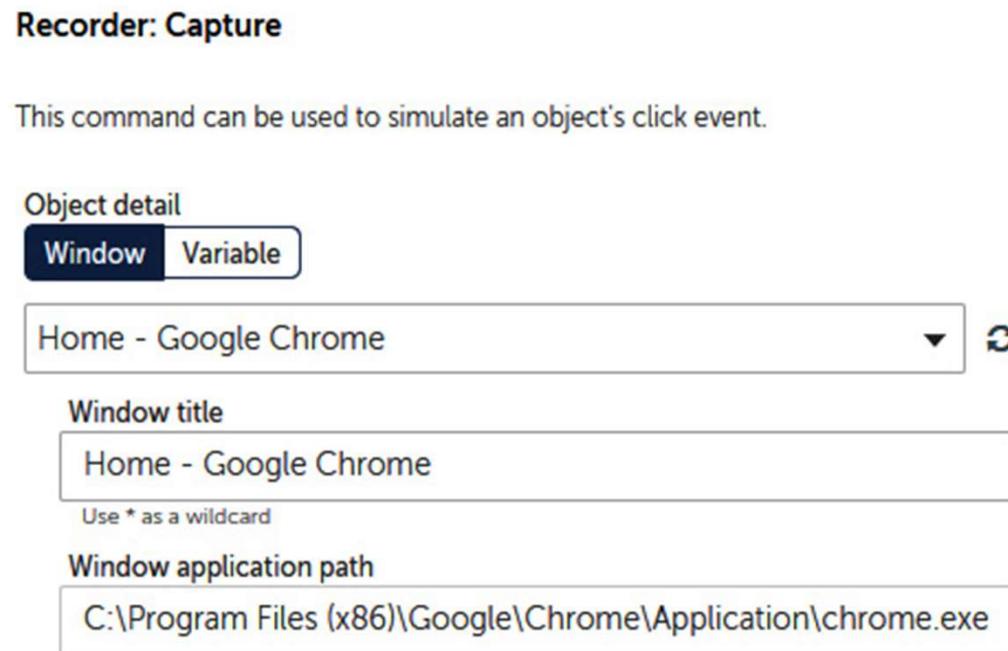
e.g. <http://>...

Browser

Google Chrome ▾

Automating desktop applications

- Refresh the windows drop-down list and select Home - Google Chrome.
- The action properties should look like this



Automating desktop applications



The Bots Are Coming...



Are you ready for the Digital Workforce?

Robotic Processing Automation (RPA) is currently one of the fastest growing technologies around the world. Automation of repetitive, tedious and boring tasks can help your business get most out of your workforce.

What is Robotic Process Automation?

Commonly known as RPA it enables organisations and businesses to 'teach' computers to execute repetitive and time consuming tasks. Being a software based tool, it is easily able to connect and integrate with most desktop applications as well as integration with back-end systems via API's.

Automating desktop applications

As we want to extract the text, we need to identify which property contains this. If we look through the object properties, we can identify the correct property. Once identified, make a note of it. When I captured this, the text was in the **HTML InnerText** property, as you can see in the following screenshot:

HTML InnerText  Commonly known as RPA it enables organisations

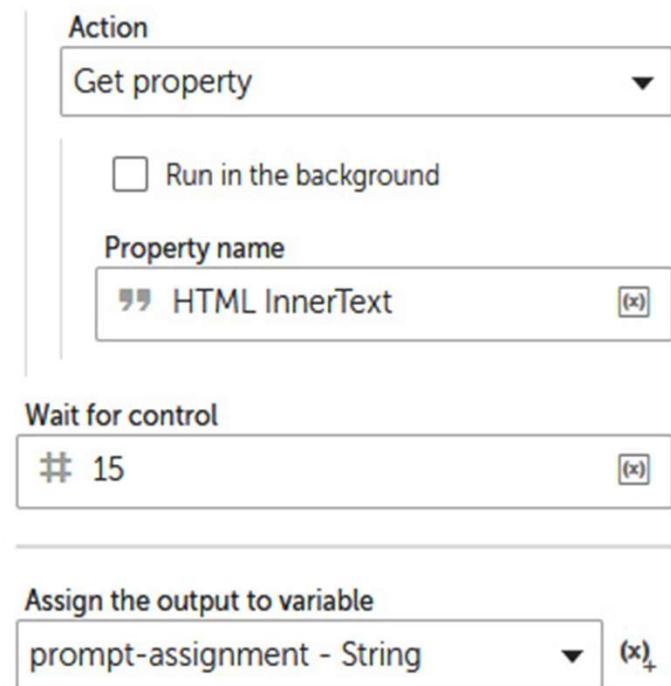
– HTML InnerText property

When you capture the paragraph, you may not necessarily see the text in the same attribute. In some cases, it may be in the **Name** attribute, as shown in the following screenshot:

Name  Commonly known as RPA it enables organisations

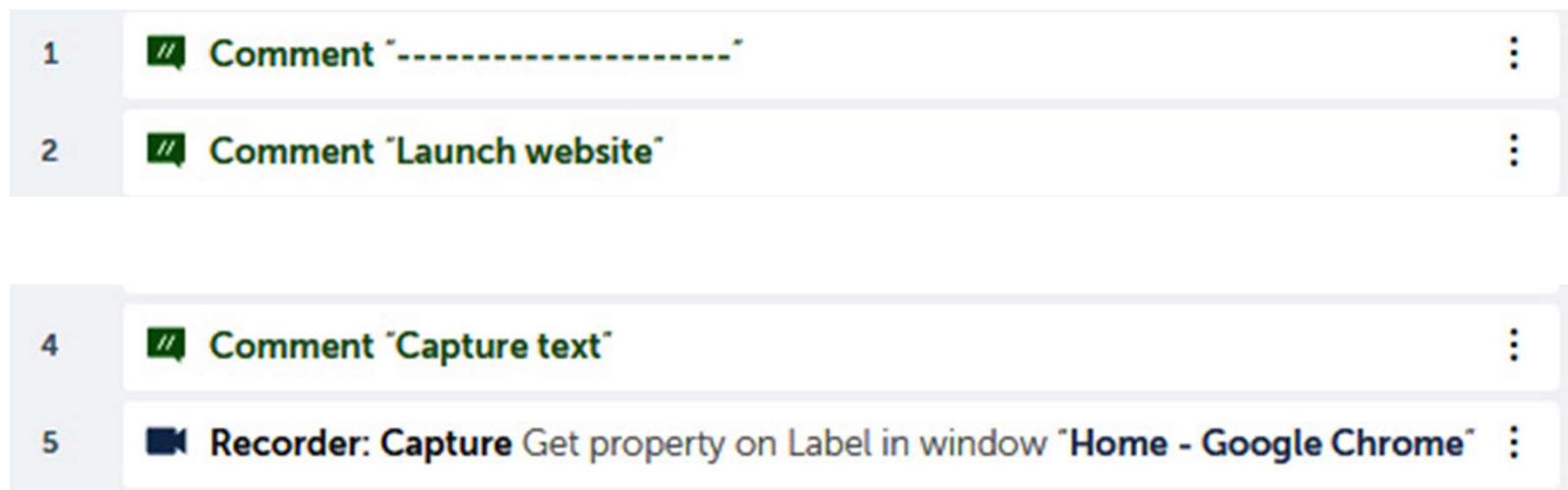
Automating desktop applications

- Set the following properties for the Recorder:
Capture action on line 5:
- Action: Get property
- Property name: HTML InnerText (if your property was Name, then set as Name)
- Assign the output to variable: prompt-assignment - String
- The properties should look similar to the following screenshot:



Automating desktop applications

- Click on Save, Your development interface should look something like this:



Working with Notepad

Application: Open program/file

Opens program/file

Location of the program/file

“ C:\Windows\System32\notepad.exe

e.g.“...\\excel.exe”

Start in path (optional)

“

e.g.“C:\\My Folder”

Parameters (optional)

“

e.g. /r “E:\\My Folder\\test.xls”

Working with Notepad

Recorder: Capture

This command can be used to simulate an object's click event.

Object detail

Window Variable

Untitled - Notepad ▾ 

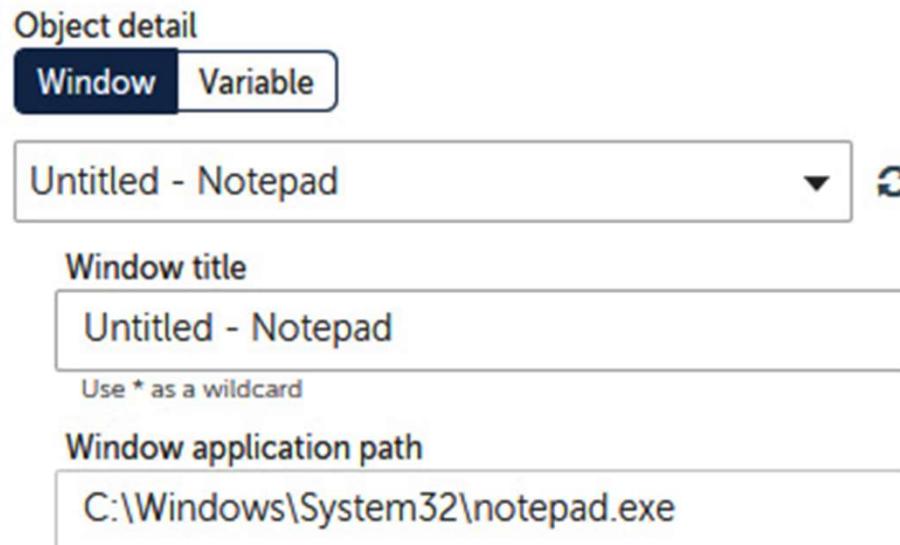
Window title

Untitled - Notepad

Use * as a wildcard

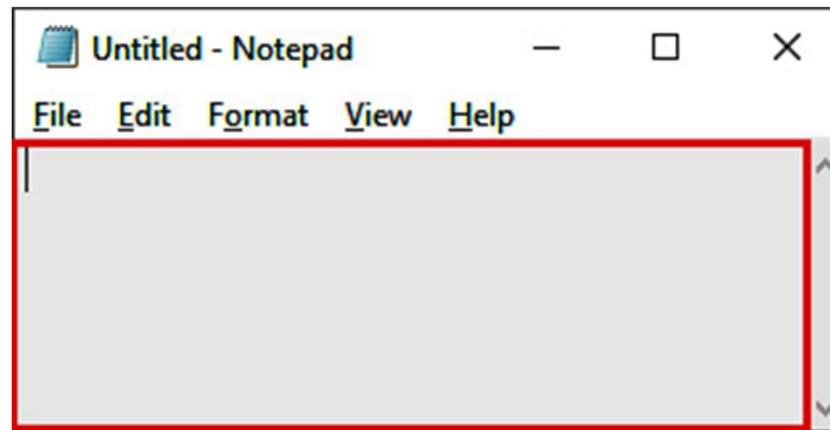
Window application path

C:\Windows\System32\notepad.exe



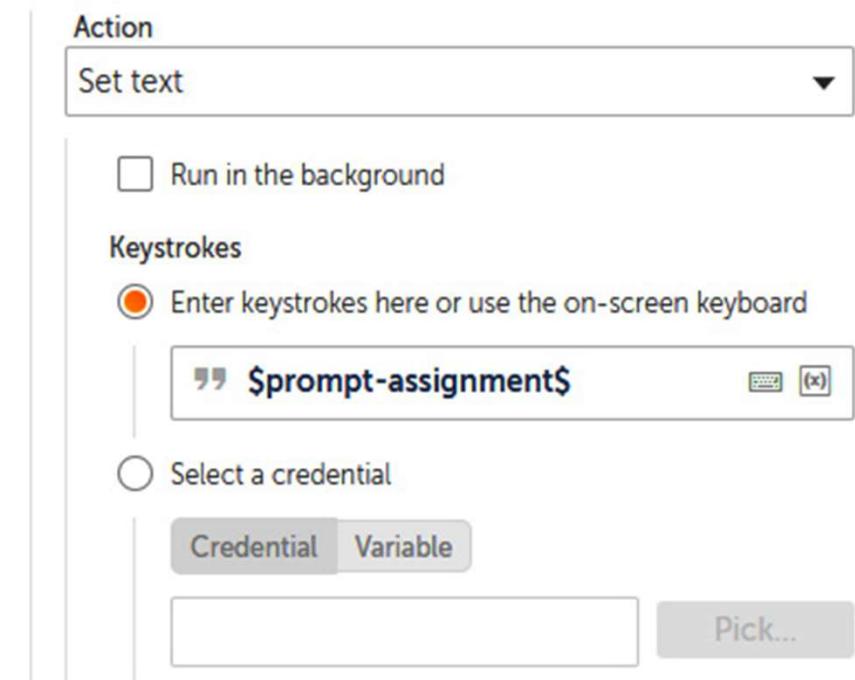
Working with Notepad

- To capture the text entry pane, click on Capture object.
- When Notepad appears, hover the mouse over the text entry pane until it has a red border around it, as follows:



Working with Notepad

- We want to enter the contents of our variable in this pane, Set the following properties for the Recorder: Capture action on line 9:
- Action: Set text
- Keystrokes: \$prompt-assignment\$
- The properties should look similar to the following screenshot:

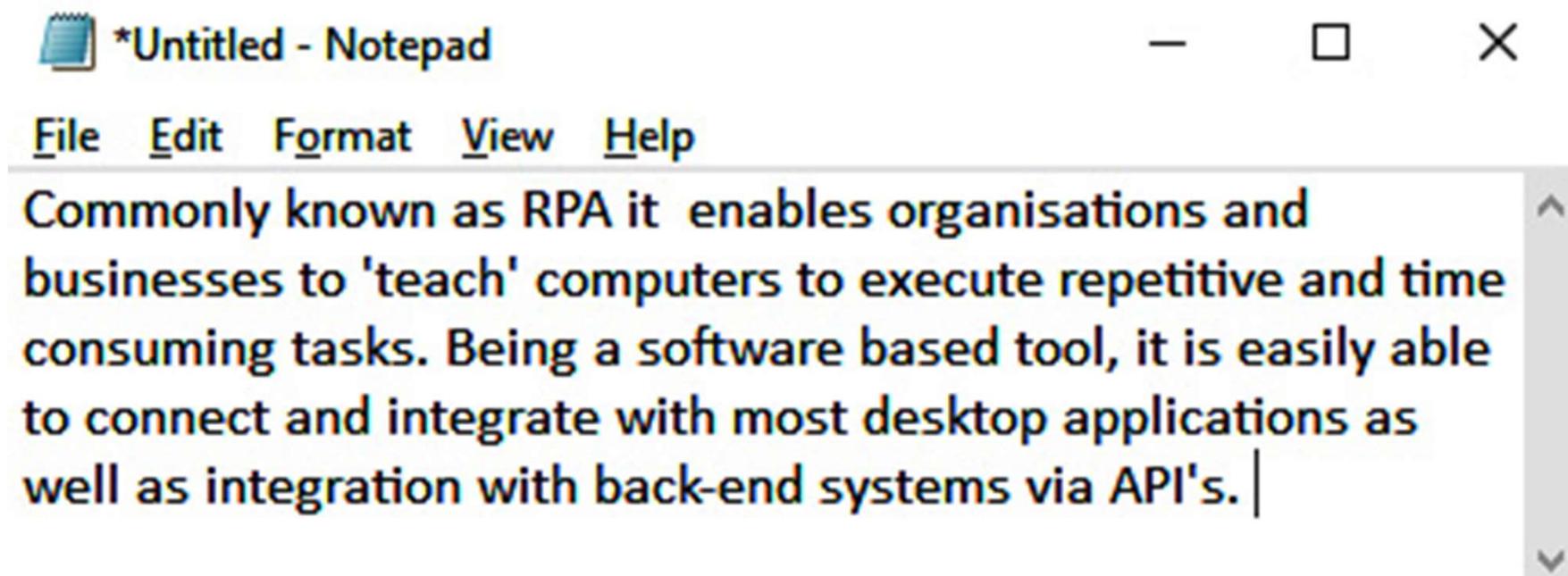


Working with Notepad

- Click on Save.
- Your development interface should look something like this:

6	Comment "Open notepad"	:
7	Application: Open program/file "C:\Windows\System32\notepad.exe"	:
8	Comment "Enter text in notepad"	:
9	Recorder: Capture Set text on textbox in window "Untitled - Notepad"	:

Simulating keystrokes



Simulating keystrokes

Simulate keystrokes

Inserts keystrokes into a selected window

Select window

Window Variable

*Untitled - Notepad



Window title

*Untitled - Notepad

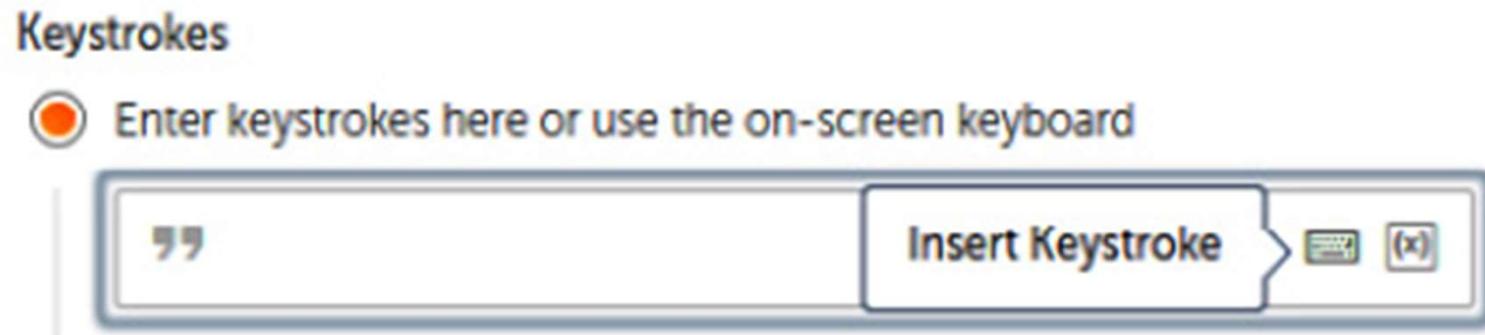
Use * as a wildcard

Window application path

C:\Windows\System32\notepad.exe

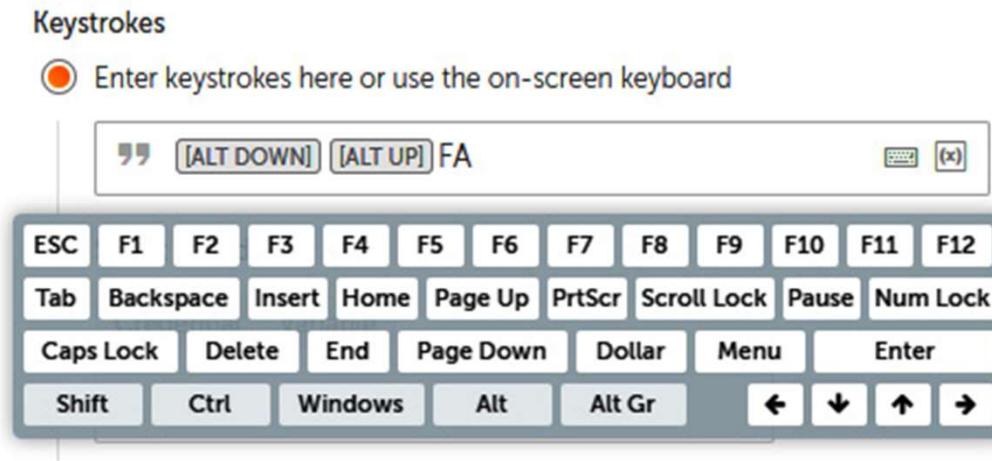
Simulating keystrokes

- To assign the keystroke sequence of Alt + F + A, set the Keystrokes property to Enter keystrokes here or use the on-screen keyboard.
- Click on the keyboard icon, as shown in the following screenshot:

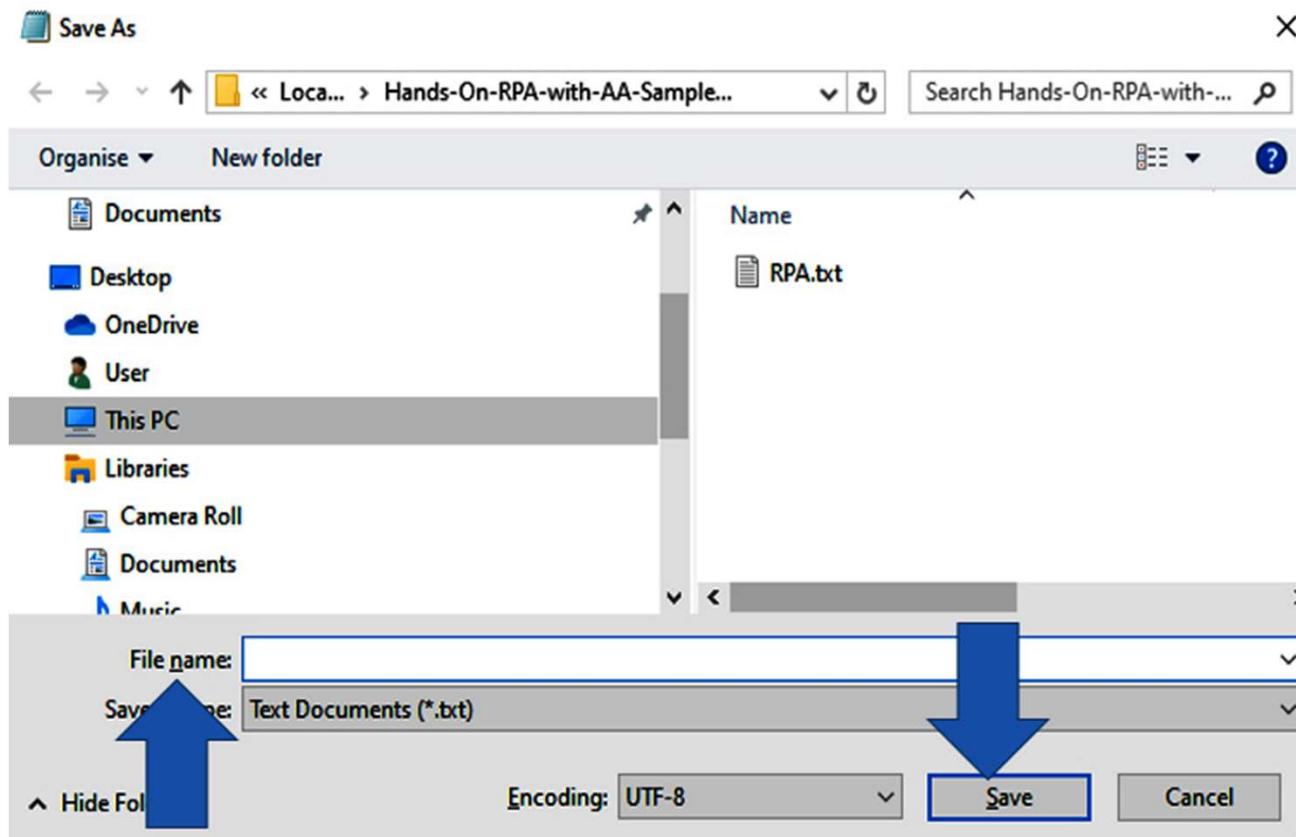


Simulating keystrokes

- The keyboard will appear with all the special keys.
- Any alphanumeric keys can just be typed in the desired case, Select the sequence Alt + F + A.
- This property should look like this:



Simulating keystrokes



Simulating keystrokes

- Refresh the windows drop-down list and select Save As.
- The action properties should look like this:

Simulate keystrokes

Inserts keystrokes into a selected window

Select window

Window Variable

Save As



Window title

Save As

Use * as a wildcard

Window application path

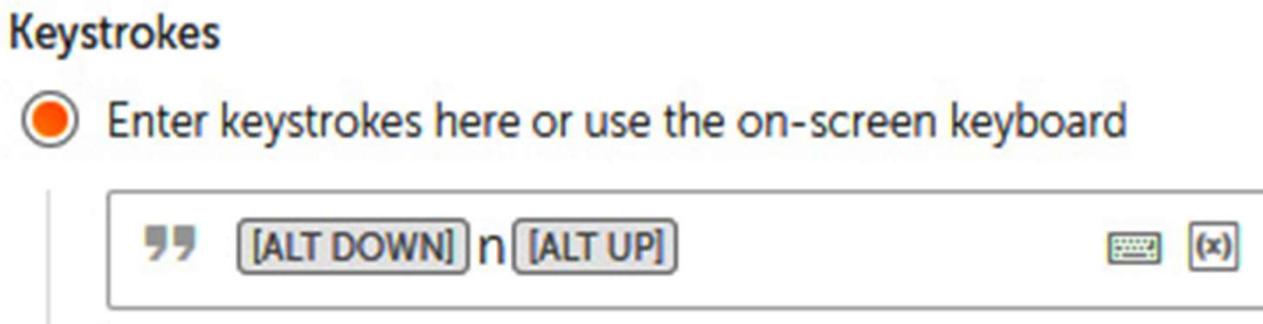
C:\Windows\System32\notepad.exe

Window type

Dialog

Simulating keystrokes

- To assign the keystroke sequence of Alt + n, set the Keystrokes property to Enter keystrokes here or use the on-screen keyboard.
- Once the keyboard appears, select the sequence Alt + n, This property should look like this:



Simulating keystrokes

- Once the keyboard appears, enter C:\Hands-On-RPA-with-AA-Sample-Data\lesson07.txt, as shown in the following screenshot:

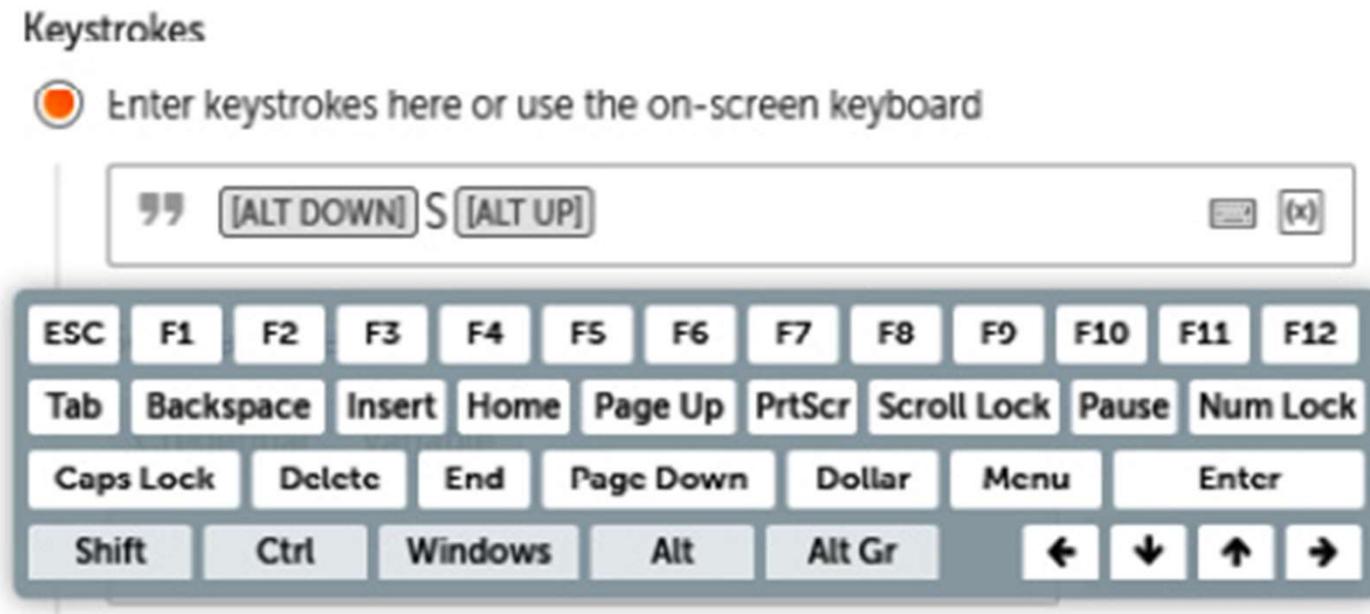
Keystrokes

- Enter keystrokes here or use the on-screen keyboard



Simulating keystrokes

- Once the keyboard appears, select the sequence Alt + S.
- This property should look like this:



Simulating keystrokes

- Click on Save. Your development interface for this section should look something like this:

10	Comment "Save text file"	:
11	Simulate keystrokes "[ALT DOWN][ALT UP]FA" on window "*Untitled - Notepad"	:
12	Simulate keystrokes "[ALT DOWN]n[ALT UP]" on window "Save As"	:
13	Simulate keystrokes "C:\Hands-On-RPA-with-AA..." on window "Save As"	:
14	Simulate keystrokes "[ALT DOWN]S[ALT UP]" on window "Save As"	:

Simulating keystrokes

- Refresh the windows drop-down list and select lesson07.txt –
- Notepad. The properties should look like this:

Window: Close

Closes a window

Window
Window Variable

.lxl - Notepad ▾ 

Window title

.txt - Notepad
Use * as a wildcard

Window application path

C:\Windows\System32\notepad.exe

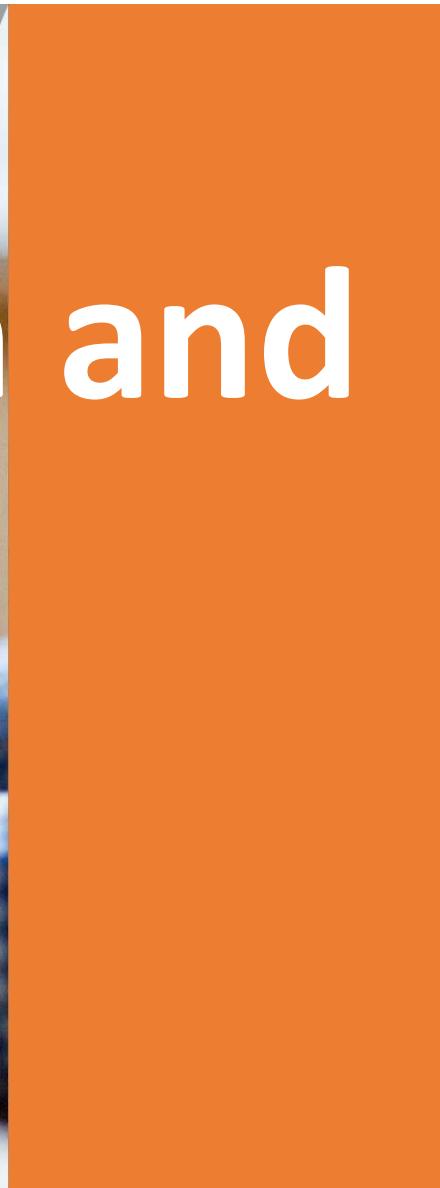
Simulating keystrokes

▶ Start	
1	〃 Comment "-----"
2	〃 Comment "Launch website"
3	↗ Browser: Launch website
4	〃 Comment "Capture text"
5	■ Recorder: Capture Get property on Label in window "Home - Google Chrome"
6	〃 Comment "Open notepad"
7	⌚ Application: Open program/file "C:\Windows\System32\notepad.exe"
8	〃 Comment "Enter text in notepad"
9	■ Recorder: Capture Set text on textbox in window "Untitled - Notepad"
10	〃 Comment "Save text file"
11	■ Simulate keystrokes "[ALT DOWN][ALT UP]FA" on window "*Untitled - Notepad"
12	■ Simulate keystrokes "[ALT DOWN]n[ALT UP]" on window "Save As"
13	■ Simulate keystrokes "C:\Hands-On-RPA-with-AA..." on window "Save As"
14	■ Simulate keystrokes "[ALT DOWN]S[ALT UP]" on window "Save As"
15	〃 Comment "Close notepad"
16	□ Window: Close the "Untitled - Notepad" window
17	〃 Comment "-----"
	▀ End

Summary

- This lesson has covered some key elements of implementing RPA in relation to your daily tasks.
- Understanding how to interact with web and desktop applications is what we humans do.
- All the tasks that we perform while sitting in front of a computer involve interacting with an application of some sort.
- This interaction may involve selecting or clicking on objects as well as entering inputs using the keyboard.
- The walk-throughs in this lesson have given you the practical knowledge to enable you to create bots that navigate through applications, as well as read and enter text.

String Manipulation and List Variables



String Manipulation and List Variables

In this lesson, we will cover the following topics:

- Manipulating strings
- Creating and looping through List variables
- Applying simple conditional logic

String Manipulation and List Variables

 CSV/TXT	 Log To File
 Comment	 Loop
 If	 Message box
 List	 String

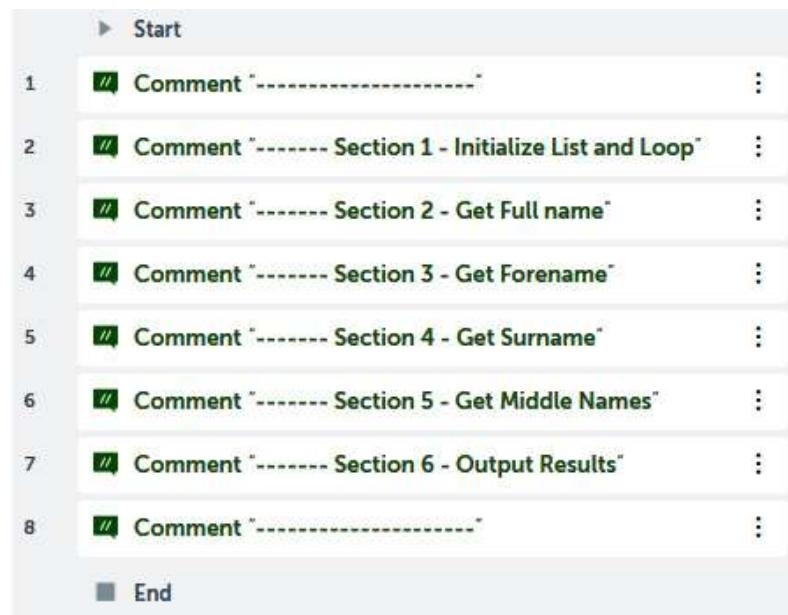
Manipulating strings

While working through this walk-through, we will look at the following string manipulation actions:

- Replace
- Find
- Substring
- Split
- Trim
- Uppercase
- Lowercase

Manipulating strings

- Add a new Comment action of "-----" on line 8 and click on Save.
- Now, your bot should look like this:



Section 1 – Initializing lists and loops

Create variable

Cancel

Create

Name

tblSourceText

Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

Type

Table

Default value (optional)

Edit table (1x1)

Section 1 – Initializing lists and loops

Create variable

Cancel

Create

Name

lstSourceList

Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

Type

List

Subtype

String

Default value (optional)

This list is empty



Section 1 – Initializing lists and loops

- To store each full name from the list, create a String type variable called strFullName.
- The initial variable list should look like this:

Variables +

Search variables

User-defined ^

- list lstSourceList
- String strFullName
- Table tblSourceText

Section 1 – Initializing lists and loops

- The source list of names that we will be using is stored in the lesson08_InputData.txt file in our root folder, which can be found at C:\Hands-On-RPA-with-AA-Sample-Data.
- The content of this file is a single string containing the following data:

"husan lal mahey, priya mahey, sonam mahey, ravinder raj lal mahey,
sunita kumari mahey, manisha Mahey"

Section 1 – Initializing lists and loops

- Set the following properties for the CSV/TXT: Open action on line 3:

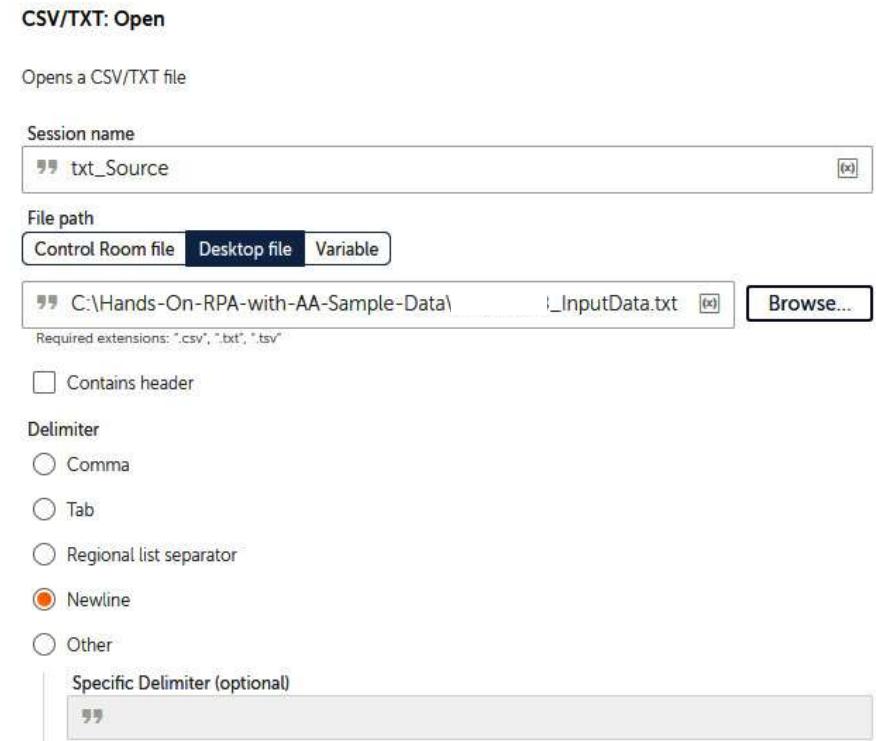
Session name: txt_Source

File path: Desktop file – C:\Hands-On-RPA-with-AA-Sample-Data\lesson08_InputData.txt

Contains header: Unchecked

Delimiter: Newline

The properties should look like this:



Section 1 – Initializing lists and loops

- Set the following properties for the CSV/TXT: Read action on line 4:

Session name: txt_Source

Assign value to the variable:
tblSourceText - Table

The properties should look like this:

CSV/TXT: Read

Reads the entire content of a CSV file

Session name

 (x)

Assign value to the variable

 ▼ (x)

Section 1 – Initializing lists and loops

Set the following properties for the CSV/TXT: Close action on line 5:

- Session name: txt_Source
- The properties should look like this:

CSV/TXT: Close

Closes CSV/TXT session

Session name

 txt_Source	
---	--

Applying the Split action

- When a string needs to be separated into a List variable, we can apply the Split action.
- We have already created our List type variable; that is, `IstSourceList`.
- To split a variable, it is essential we know what character to use for the split, Let's take a look at our source string:

husan lal mahey, priya mahey, sonam mahey, ravinder raj lal mahey,
sunita kumari mahey, manisha mahey

Applying the Split action

- Set the following properties for the String: Split action on line 6:
- Source string: \$tblSourceText[0][0]\$
- Delimiter: ,
- Delimiter is: Not case sensitive
- Split into substrings: All possible
- Assign the output to list variable: lstSourceList – List of Strings
- The properties should look like this:

String: Split

Splits the source string into multiple strings using a delimiter.

Source string
\$tblSourceText[0][0]\$

Delimiter
,

Delimiter is
 Case sensitive
 Not case sensitive

Split into substrings
 All possible
 Only
#

Assign the output to list variable
lstSourceList - List of Strings

Applying the Split action

- Click on Save. Your development interface for this section should look like this:

2	Comment "----- Section 1 - Initialize List and Loop"	:
3	CSV/TXT: Open "C:\Hands-On-RPA-with-AA-Sample-Data\"	:
4	CSV/TXT: Read data and assign to \$tblSourceText\$:
5	CSV/TXT: Close csv/txt "txt_Source"	:
6	String: Split \$tblSourceText[0][0]\$ with delimiter ";" and assign the result to \$lstSourceList\$:

Looping through lists

- Drag the Loop action from the Loop package just below line 6.
- Set the following properties for the Loop action on line 7:
 - Loop Type: Iterator
 - Iterator: For each item in the list
 - list: lstSourceList - List
 - For: All items in the list
 - Assign the current value to variable: strFullName - String
 - The properties should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each item in the list

iterate list

list

lstSourceList - List

(x)

For

All items in the list

Range

From index (optional)

#

To index (optional)

#

Assign the current value to variable

strFullName - String

(x)

Looping through lists

Message box

Displays a message box

Enter the message box window title

String Manipulation



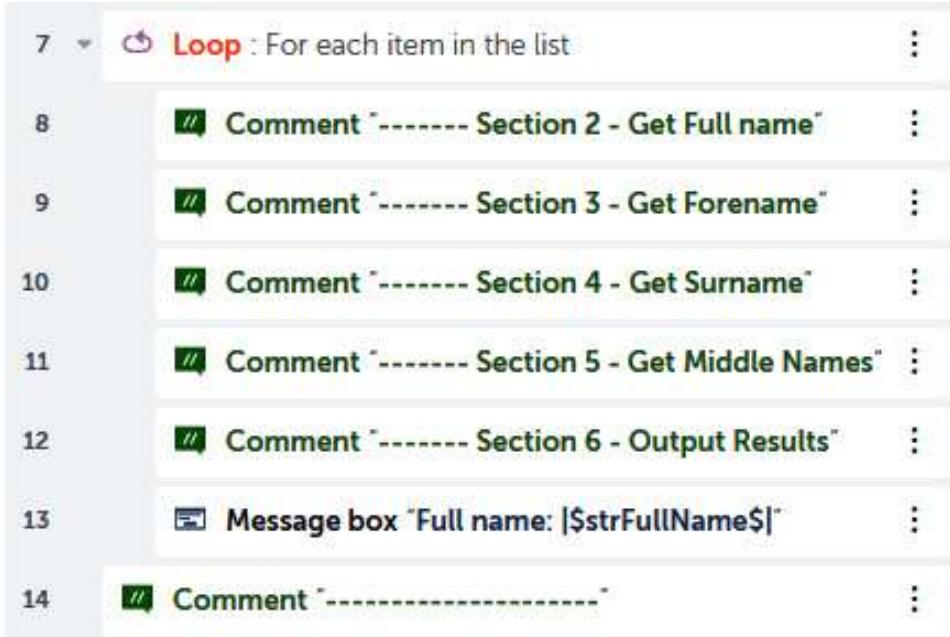
Enter the message to display

Full name: |\$strFullName\$|



Looping through lists

- Click on Save.
- Your development window for this section should look like this:



The screenshot shows a software development environment with a code editor. The code is organized into a loop structure. The loop starts at line 7 with a 'Loop' block, which contains six 'Comment' blocks (lines 8-13) and ends at line 14 with another 'Comment' block. Line 13 contains a 'Message box' block. The code is as follows:

```
7 Loop :For each item in the list
8  // Comment ----- Section 2 - Get Full name
9  // Comment ----- Section 3 - Get Forename
10 // Comment ----- Section 4 - Get Surname
11 // Comment ----- Section 5 - Get Middle Names
12 // Comment ----- Section 6 - Output Results
13  Message box "Full name: $strFullName$"
14 // Comment -----
```

Section 2 – Getting full names

- Set the following properties for the String: Trim action on line 9:
- Source String: \$strFullName\$
- Trim from the beginning: Checked
- Trim from the end: Checked
- Assign the output to variable: strFullName - String
- The properties should look like this:

String: Trim

Trims blanks and whitespaces from a given string.

Source string

“ \$strFullName\$ ” (x)

Trim from the beginning

Trim from the end

Assign the output to variable

strFullName - String ▼ (x)

Applying uppercase to a string

- Set the following properties for the String: Uppercase action on line 10:Source
String: \$strFullName\$
- Assign the output to variable: strFullName - String
- The properties should look like this:

String: Uppercase

Converts the source string to upper case.

Source string

” \$strFullName\$ (x)

Assign the output to variable

strFullName - String ▼ (x)

Applying uppercase to a string

- Click on Save.
- Your development interface for this section should look like this:

```
8  // Comment "----- Section 2 - Get Full name"
9  ?? String: Trim $strFullName$ and assign the result to $strFullName$
10 ?? String: Uppercase Convert $strFullName$ to upper case and assign the result to $strFullName$
```

Section 3 – Getting forenames

- It would be a good idea to add these variables to our message box to help us with progress testing.
- Edit the message display property of the message box on line 15 to the following:

Message box

Displays a message box

Enter the message box window title

” String Manipulation

Enter the message to display

” Full name: |\$strFullName\$|
Initial: |\$strInitial\$|
Forename:|\$strForename\$|

Section 3 – Getting forenames

String: Extract text

Extracts a sub-string between two given strings specified by 'Before' and 'After'.

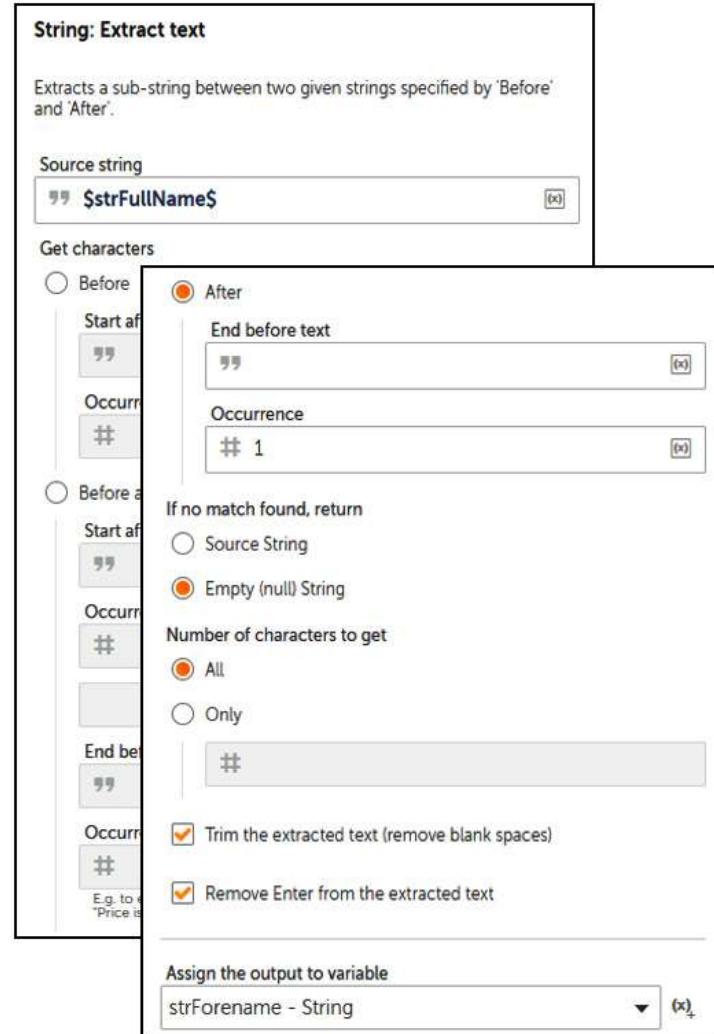
Source string
\$strFullName\$

Get characters

After
Start after
Occurrences
End before text
Occurrence
If no match found, return
Source String
Empty (null) String

Before a
Start after
Occurrences
End before
Occurrences
E.g. to extract "Price is \$100" from "The price is \$100."
Number of characters to get
All
Only
Trim the extracted text (remove blank spaces)
Remove Enter from the extracted text

Assign the output to variable
strForename - String



Section 3 – Getting forenames

String: Substring

Extracts a sub-string from a given string.

Source string

“ \$strForename\$

(x)

Start from

1

(x)

Length (optional)

1

(x)

Assign the output to variable

strInitial - String

▼

(x)

Section 3 – Getting forenames

String: Substring

Extracts a sub-string from a given string.

Source string

” \$strForename\$

(x)

Start from

2

(x)

Length (optional)

#

(x)

Assign the output to variable

strForename - String

▼

(x)

Section 3 – Getting forenames

String: Lowercase

Converts the source string to lower case.

Source string

“ \$strForename\$ ”

(x)

Assign the output to variable

strForename - String

▼

(x)

Section 3 – Getting forenames

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

“ \$strInitial\$\$strForename\$



Select the destination string variable

strForename - String



(x)₄

Section 3 – Getting forenames

- | | | |
|----|--|---|
| 11 | Comment ----- Section 3 - Get Forename | : |
| 12 | ?? String: Extract text Source string \$strFullName\$: Extract sub-string after "" | : |
| 13 | ?? String: Substring : Extract substring from the \$strForename\$ string | : |
| 14 | ?? String: Substring : Extract substring from the \$strForename\$ string | : |
| 15 | ?? String: Lowercase : Convert the \$strForename\$ to lowercase | : |
| 16 | ?? String: Assign "\$strInitial\$strForenam..." to \$strForename\$ | : |

Section 4 – Getting surnames

- Set the following properties for the String: Find action on line 18:
- Source string variable: \$strFullName\$
- Find string: (\w+)\$\$
- When finding: Do not match case
- The "find string" is: A regular expression
- Start from: 1
- Assign the output to variable: numLoc - Number
- The properties should look like this:

String: Find

Locates a given string within the source string.

Source string

\$strFullName\$

(x)

Find string

(\w+)\$\$

(x)

When finding

Match case

Do not match case

The "find string" is

A regular expression

Not a regular expression

Start from (optional)

1

(x)

Assign the output to variable

numLoc - Number

(x)

Section 4 – Getting surnames

- Set the following properties for the String: Substring action on line 19:Source String: \$strFullName\$
- Start from: \$numLoc\$
- Assign the output to variable: strSurname - String
- The properties should look like this:

String: Substring

Extracts a sub-string from a given string.

Source string

” \$strFullName\$

Start from

\$numLoc\$

Length (optional)

#

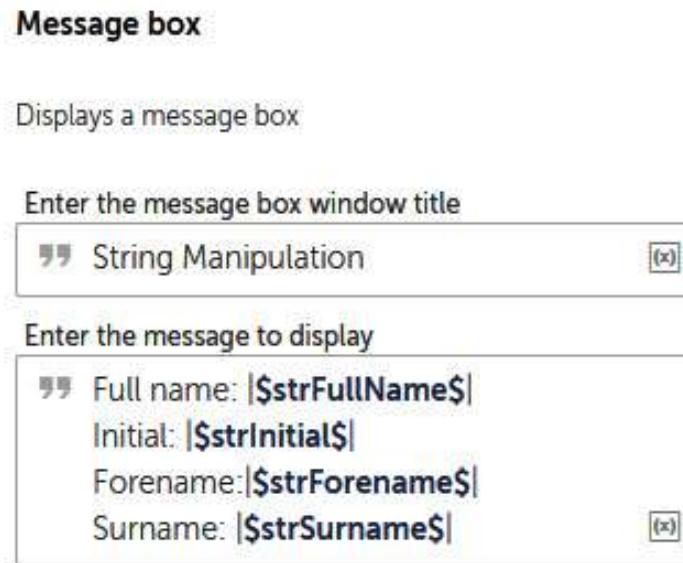
Assign the output to variable

strSurname - String

(x) +

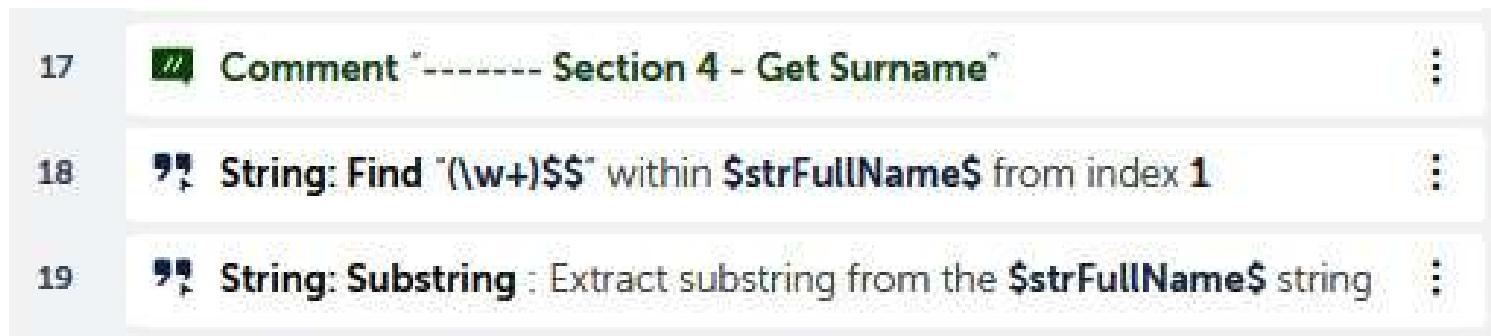
Section 4 – Getting surnames

- Now that the surname has been extracted, add it to the final message box on line 22.
- Edit the message display property of the message box so that it includes the following:



Section 4 – Getting surnames

- Great progress! That's Section 4 – Getting surnames complete. Run the bot to test it.
- We now have the forename and the surname in the correct format. The development interface for this section should look like this:



```
17  Comment ----- Section 4 - Get Surname
18  String: Find '(\w+)$' within $strFullName$ from index 1
19  String: Substring : Extract substring from the $strFullName$ string
```

Section 4 – Getting surnames

String: Replace

Replaces specified part of a 'Source string' with a 'Replacement string'

Source string

\$strFullName\$

(x)

Find string

\$strForename\$

(x)

When finding

- Match case
- Do not match case

The "find string" is

- A regular expression
- Not a regular expression

Start from (optional)

1

(x)

Count (optional)

-1

(x)

Replace with (optional)

##

(x)

Assign the output to variable

strMiddleNames - String

(x)

▼

Section 4 – Getting surnames

- Set the following properties for the String Replace action on line 22:

Source String: \$strMiddleNames\$

Find string: \$strSurname\$

When finding: Do not match case

The "find string" is: Not a regular expression

Start from: 1

Replace with: (enter space)

Assign the output to variable: strMiddleNames

- String

The properties should look like this:

String: Replace

Replaces specified part of a 'Source string' with a 'Replacement string'

Source string

\$strMiddleNames\$

Find string

\$strSurname\$

When finding

Match case

Do not match case

The "find string" is

A regular expression

Not a regular expression

Start from (optional)

1

Count (optional)

-1

Replace with (optional)

" "

Assign the output to variable

strMiddleNames - String

Section 4 – Getting surnames

- Set the following properties for the String: Trim action on line 23:
- Source String: \$strMiddleNames\$
- Trim from the beginning: Checked
- Trim from the end: Checked
- Assign the output to variable: strMiddleNames - String
- The properties should look like this:

String: Trim

Trims blanks and whitespaces from a given string.

Source string

“ \$strMiddleNames\$ ” (x)

Trim from the beginning

Trim from the end

Assign the output to variable

strMiddleNames - String ▼ (x)

Section 4 – Getting surnames

- The middle names have now been extracted to a variable.
- Let's add this variable to the final message box on line 25.
- Edit the message display property of the Message box so that it includes this:

Message box

Displays a message box

Enter the message box window title

” String Manipulation

Enter the message to display

” Full name: |\$strFullName\$|
Initial: |\$strInitial\$|
Forename:|\$strForename\$|
Middle Names: |\$strMiddleNames\$|
Surname: |\$strSurname\$|

Assigning a null value to a string

- Set the following properties for the String: Assign action on line 24:
- Select the source string variable: (leave empty)
- Select the destination string variable: strInitial - String
- The properties should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

 (x)

Select the destination string variable

 (x)

Applying a simple logical condition

- Set the following properties for the If action on line 25:
- Condition: String condition
- Source value: \$strMiddleNames\$
- Operator: Not equal to
- Target value: (leave blank)
- The properties should look like this:

If

Runs a sequence of actions if a condition is true

Condition

String condition

Checks the string variable condition.

Source value (optional)
\$strMiddleNames\$

Operator
Not equal to(≠)

Target value (optional)
''

Match case

Add condition

Implementing a nested loop

- Create two String type variables called strCurrentMiddleName and strCurrentMiddleNameInitial.
- Your variables list should now look similar to this:

User-defined

```
■ lstMiddleNames
■ lstSourceList
# numLoc
„ strCurrentMiddleName
„ strCurrentMiddleNameInitial
„ strForename
„ strFullName
„ strInitial
„ strMiddleNames
„ strSurname
■ tblSourceText
```

Implementing a nested loop

- Set the following properties for the String: Split action on line 26:
- Source string: \$strMiddleNames\$
- Delimiter: (enter a space)
- Delimiter is: Not case sensitive
- Split into substrings: All possible
- Assign the output to list variable: lstMiddleNames - List of Strings
- The properties should look like this:

String: Split

Splits the source string into multiple strings using a delimiter.

Source string

\$strMiddleNames\$

(x)

Delimiter

" "

(x)

Delimiter is

- Case sensitive
 Not case sensitive

Split into substrings

- All possible
 Only

#

Assign the output to list variable

lstMiddleNames - List of Strings

▼

(x)

Implementing a nested loop

- Set the following properties for the Loop action on line 27:
- Loop Type: Iterator
- Iterator: For each item in the list
- List: lstMiddleNames - List
- For: All items in the list
- Assign the current value to variable: strCurrentMiddleName - String
- The properties should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each item in the list

iterate list

list

lstMiddleNames - List

(x)

For

All items in the list

Range

From index (optional)

#

To index (optional)

#

Implementing a nested loop

- Set the following properties for the String: Substring action on line 28:
- Source String:
\$strCurrentMiddleName\$
- Start from: 1
- Length: 1
- Assign the output to variable:
strCurrentMiddleNameInitial - String
- The properties should look like this:

String: Substring

Extracts a sub-string from a given string.

Source string

\$strCurrentMiddleName\$ (x)

Start from

1 (x)

Length (optional)

1 (x)

Assign the output to variable

strCurrentMiddleNameInitial - String (x)

Implementing a nested loop

- Set the following properties for the String: Assign action on line 29:
- Select the source string variable(s)/ value (optional):
\$strInitial\$\$strCurrentMiddleName
Initial\$
- Select the destination string variable: strInitial - String
- The properties should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

\$strInitial\$\$strCurrentMiddleNameInitial\$

Select the destination string variable

strInitial - String

Implementing a nested loop

```
20  // Comment ----- Section 5 - Get Middle Names
21  " String: Replace $strForename$ with "" in $strFullName$ and assign the result to $strMiddleNames$ ...
22  " String: Replace $strSurname$ with "" in $strMiddleNames$ and assign the result to $strMiddleNames$ ...
23  " String: Trim $strMiddleNames$ and assign the result to $strMiddleNames$ ...
24  " String: Assign "" to $strInitial$
25  ◇ If string $strMiddleNames$ Not equal to(≠) ""
26  " String: Split $strMiddleNames$ with delimiter "" and assign the result to $lstMiddleNames$ ...
27  - ◇ Loop : For each item in the list
28      " String: Substring : Extract substring from the $strCurrentMiddleName$ string ...
29      " String: Assign "$strInitial$$strCurrent..." to $strInitial$ ...
```

Section 6 – Outputting the results

- A CSV file with headers will need to be created just before the primary loop as we only want this to be created once.
- The record should be added while we're within the primary loop so that it's created once per individual.
- Once we've done this, we will need to identify the sequence of our name items as required. The output should be in the following format:

Surname in uppercase, Forename in Proper case, Middle name initials in uppercase

Section 6 – Outputting the results

Log to file

Logs any text into a file

File path

(x)Browse...

Enter text to log

(x)

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

▼

Section 6 – Outputting the results

Log to file

Logs any text into a file

File path

C:\Hands-On-RPA-with-AA-Sample-Data\

(x)

Browse...

Enter text to log

\$strSurname\$, \$strForename\$ \$strInitial\$

(x)

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

ANSI

▼

Section 6 – Outputting the results

- Finally, delete the Message box action on line 33 and click on Save.
- Congratulations – you have completed your bot! The development interface for this final section should look like this:

2	#[Comment "----- Section 1 - Initialize List and Loop"	:
3	Log to file "Surname, Forename/Initi..." to "C:\Hands-On-RPA-with-AA-Sample-Data\Output.csv"	:
31	#[Comment "----- Section 6 - Output Results"	:
32	Log to file "\$strSurname\$, \$strForen..." to "C:\Hands-On-RPA-with-AA-Sample-Data\Output.csv"	:

Section 6 – Outputting the results

- Now, it's time to test the bot. When it's executed, you will get the lesson08_Output.csv file at C:\Hands-On-RPA-with-AA-Sample-Data\lesson08_Output.csv.
- The contents should look like this:

	A	B
1	Surname	Forename/Initials
2	MAHEY	Husan L
3	MAHEY	Priya
4	MAHEY	Sonam
5	MAHEY	Ravinder RL
6	MAHEY	Sunita K
7	MAHEY	Manisha

Summary

- To recap, we have covered a number of useful actions, including extracting a specific substring from a string via locations, extracting a substring from a string via specific text, finding a specific substring within a string, replacing parts of a string
- Converting strings into upper or lowercase, trimming leading/trailing spaces from strings, using regular expressions to find string patterns, splitting strings with a specific delimiter, concatenating strings, creating list variables, looping through list variables, and finally, understanding simple logical conditions.

Working with Conditional Logic, Loops, and the Filesystem



Working with Conditional Logic, Loops, and the Filesystem

In this lesson, we will cover the following topics:

- Applying different types of loops
- Applying logical conditions
- Working with the filesystem

Working with Conditional Logic, Loops, and the Filesystem

 Comment	 List
 CSV/TXT	 Log To File
 File	 Loop
 Folder	 String

Applying different types of loops

- Add a new Comment action called "-----" on line 9 and click on Save, Our initial development interface should look like this:

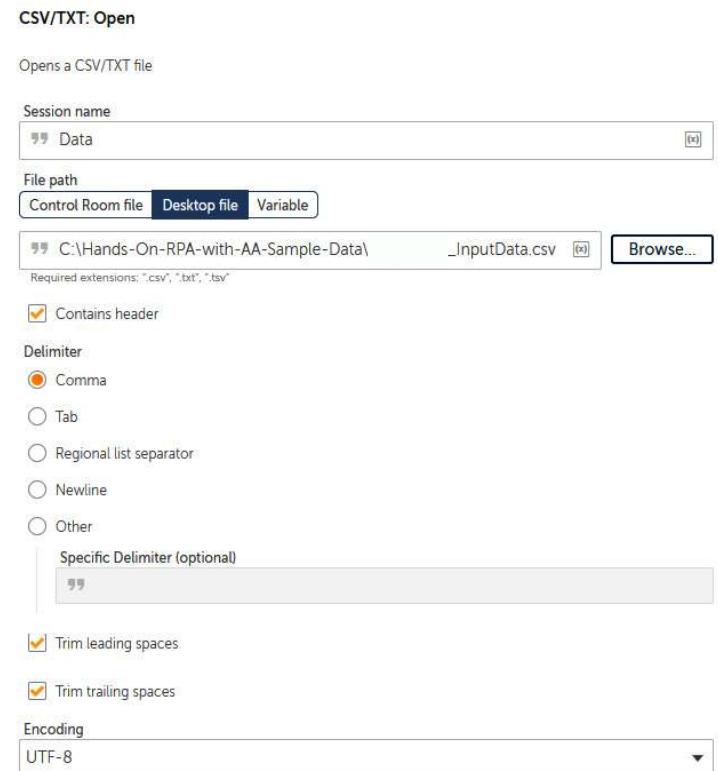
1	#[Comment "-----"	:
2	#[Comment "----- Section 1 - Open the Source file"	:
3	#[Comment "----- Section 2 - Loop through each row"	:
4	#[Comment "----- Section 3 - Get Surname initial & identify group"	:
5	#[Comment "----- Section 4 - Check if sub folder exists"	:
6	#[Comment "----- Section 4a - Create sub folder & output file if it doesn't exist"	:
7	#[Comment "----- Section 5 - Update output file"	:
8	#[Comment "----- Section 6 - Close the Source file"	:
9	#[Comment "-----"	:

Section 1 – Opening the source file

Set the following properties for the CSV/TXT:

Open action on line 3:

- Session name: Data
- File path: Desktop file – C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_InputData.csv
- Contains header: Checked
- Delimiter: Comma
- Trim leading spaces: Checked
- Trim trailing spaces: Checked
- The properties window should look like this:



Section 1 – Opening the source file

- Click on Save.
- The development window for this section should look like this:



Section 2 – Looping through each row

- Create a new variable named recSource as a Record type.
- The Create variable dialog should look like this:

Create variable

Name
recSource
Max characters = 50

Description (optional)

Max characters = 255

Constant (read-only)

Use as input Use as output

Type
Record

Default value (optional)
Record

+

Section 2 – Looping through each row

Set the following properties for the Loop action on line 5:

- Loop Type: Iterator
- Iterator: For each row in CSV/TXT
- Session name: Data
- Assign the current row to this variable: recSource - Record
- The properties window should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in CSV/TXT

Iterator for each row in CSV/TXT

Session name

” Data

Assign the current row to this variable

recSource - Record

Section 2 – Looping through each row

- Click on Save.
- The completed development window should look like this:

1	Comment "-----"	:
2	Comment "----- Section 1 - Open the Source file"	:
3	CSV/TXT: Open "C:\Hands-On-RPA-with-AA-Sample-Data_\InputData.csv"	:
4	Comment "----- Section 2 - Loop through each row"	:
5	Loop for each row in csv/txt	:
6	Comment "----- Section 3 - Get Surname initial & identify group"	:
7	Comment "----- Section 4 - Check if sub folder exists"	:
8	Comment "----- Section 4a - Create sub folder & output file if it doesn't exist"	:
9	Comment "----- Section 5 - Update output file"	:
10	Comment "----- Section 6 - Close the Source file"	:
11	Comment "-----"	:

Section 3 – Getting the surname initial and identifying the group

- Set the following properties for the String:
Substring action on line 7:Source String
name: \$recSource[1]\$
- Start from: 1
- Length: 1
- Assign the output to variable: strRefInitial - String
- The properties window should look like this:

String: Substring

Extracts a sub-string from a given string.

Source string

\$recSource[1]\$

(x)

Start from

1

(x)

Length (optional)

1

(x)

Assign the output to variable

strRefInitial - String

▼

(x)

Applying logical conditions

A2019DEMPACKAGE
Window Exists demo
False condition demo
A2019DEMPACKAGE
Window Exists demo
False condition demo
APPLICATION
Application is not running
Application is running
BOOLEAN
Boolean condition
DATA TABLE
Data table is empty
Number of columns
Number of rows
DATETIME
Date Condition
DICTIONARY
Check key
Check for a single value

VBSCRIPT
Script is successful
Script is unsuccessful
WINDOW
Window exists
Window does not exist

FILE
File date
File exists
File does not exist
File size
FOLDER
Folder does not exist
Folder exists
IMAGE RECOGNITION
Image file is NOT found in image file
Image file is NOT found in window
Window is NOT found in image file
Window is NOT found in window
Image file is found in image file
Image file is found in window
Window is found in image file
Window is found in window

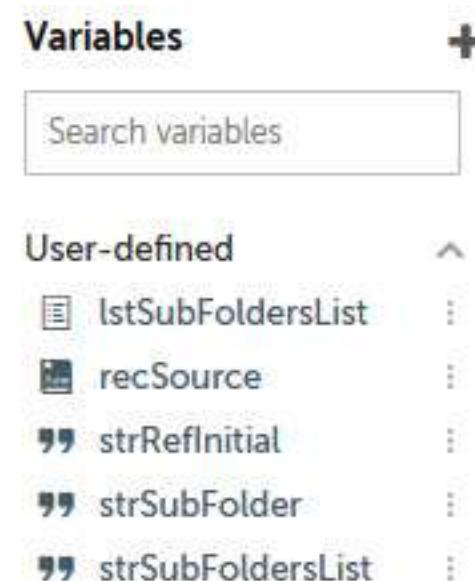
STRING
String condition
TASK BOT
Task successful
Task unsuccessful
UTILS
Not Empty String
Empty String
Compare Record Value by Index
List contains Record value

JAVASCRIPT
Script is successful
Script is unsuccessful
LEGACY AUTOMATION
Web control exists
Web control does not exist
Window control is active
Window control does not exist
Window control exists
Window control is not active
Script is unsuccessful
Script is successful
Child window does not exist
Child window exists
LIST
List variable
NUMBER
Number condition

PING
Ping is successful
Ping is unsuccessful
PROCESS DISCOVERY
Object
RECORDER
Object
SERVICE
Service is not running
Service is running

Applying different types of loops

- Create a String type variable named strSubFolder for storing the allocated subfolder.
- The variables list should look like this:



Applying different types of loops

Set the following properties for the String: Assign action on line 8:

- Select the source string variable/value:
ABCD,EFGH,IJKL,MNOP,QRST,UVW
X,YZ
- Select the destination string variable: strSubFoldersList - String
- The properties window should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

“ ABCD,EFGH,IJKL,MNOP,QRST,UVWX,YZ

Select the destination string variable

strSubFoldersList - String

Applying different types of loops

Set the following properties for the String: Assign action on line 9:

- Source string: \$strSubFoldersList\$
- Delimiter: ,
- Assign the output to list variable:
lstSubFoldersList - List of Strings
- The properties window should look like this:

String: Split

Splits the source string into multiple strings using a delimiter.

Source string
\$strSubFoldersList\$

Delimiter
,

Delimiter is
 Case sensitive
 Not case sensitive

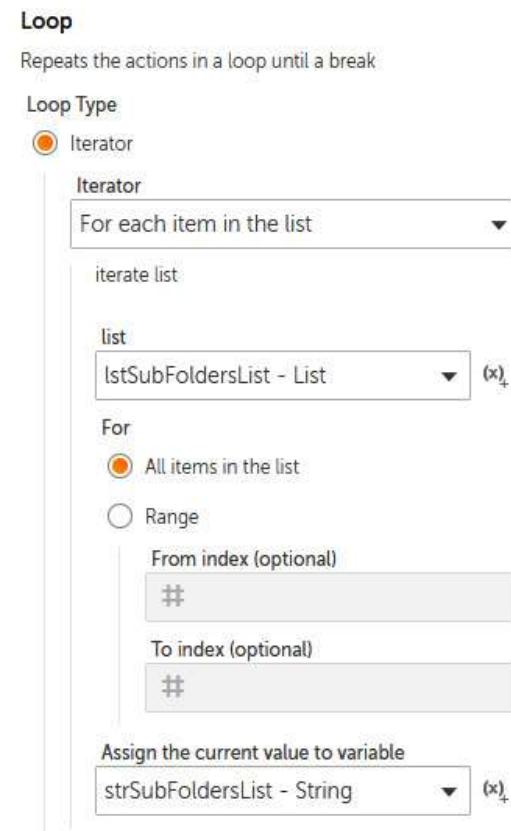
Split into substrings
 All possible
 Only
#

Assign the output to list variable
lstSubFoldersList - List of Strings

Applying different types of loops

Set the following properties for the Loop action on line 10:

- Loop Type: Iterator
- Iterator: For each item in the list
- List: lstSubFoldersList - List
- For: All items in the list
- Assign the current value to variable: strSubFoldersList - String
- The properties window should look like this:



Applying different types of loops

Set the following properties for the If action on line 11:

- Condition: String condition
- Source value: strSubFoldersList
- Operator: Includes
- Target value: \$strReflinitial\$
- The properties window should look like this:

If

Runs a sequence of actions if a condition is true

Condition

String condition

Checks the string variable condition.

Source value (optional)
\$strSubFoldersList\$

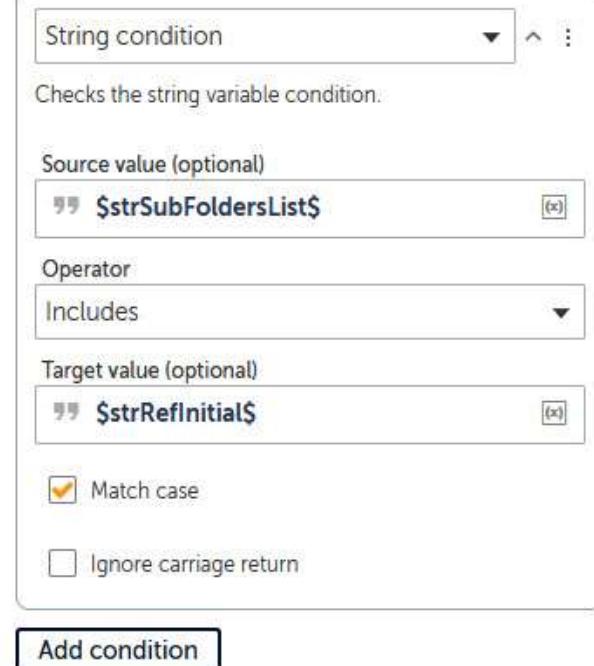
Operator
Includes

Target value (optional)
\$strReflinitial\$

Match case

Ignore carriage return

Add condition



Applying different types of loops

Set the following properties for the String: Assign action on line 12:

- Select the source string variable(s)/value (optional): \$strSubFoldersList\$
- Select the destination string variable: strSubFolder - String
- The properties window should look like this:

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

” \$strSubFoldersList\$ (x)

Select the destination string variable

strSubFolder - String ▼ (x)

Applying different types of loops

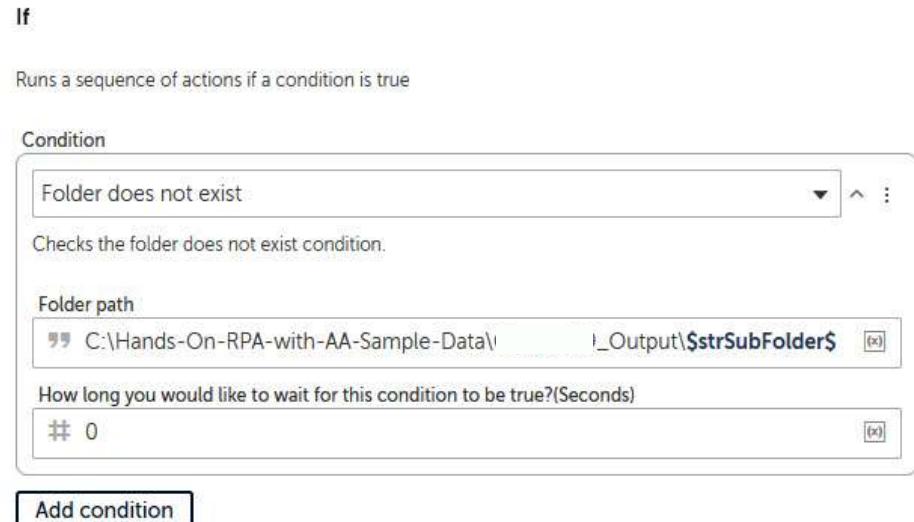
- The bot is at a stage now where it has identified the subfolder group for the working record using the surname initial.
 - This brings Section 3 – Getting the surname initial and identifying the group, to an end.
 - The development interface for this section should look like this:

6	Comment ----- Section 3 - Get Surname initial & identify group	:
7	" String: Substring : Extract substring from the \$recSource[1]\$ string	:
8	" String: Assign "ABCD,EFGH,IJKL,MNOP,QRS..." to \$strSubFoldersList\$:
9	" String: Split \$strSubFoldersList\$ with delimiter ";" and assign the result to \$lstSubFold...	:
10	Loop : For each item in the list	:
11	If string \$strSubFoldersList\$ Includes \$strRefInitial\$:
12	String: Assign \$strSubFoldersList\$ to \$strSubFolder\$:

Section 4 – Checking or creating a subfolder

Set the following properties for the If action on line 14:

- Condition: Folder does not exist
- Folder path: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$
- The properties window should look like this:



Section 4 – Checking or creating a subfolder

- The properties window should look like this:

Folder: Create

Creates a folder

Folder

„ C:\Hands-On-RPA-with-AA-Sample-Data\

_Output\\$strSubFolder\$

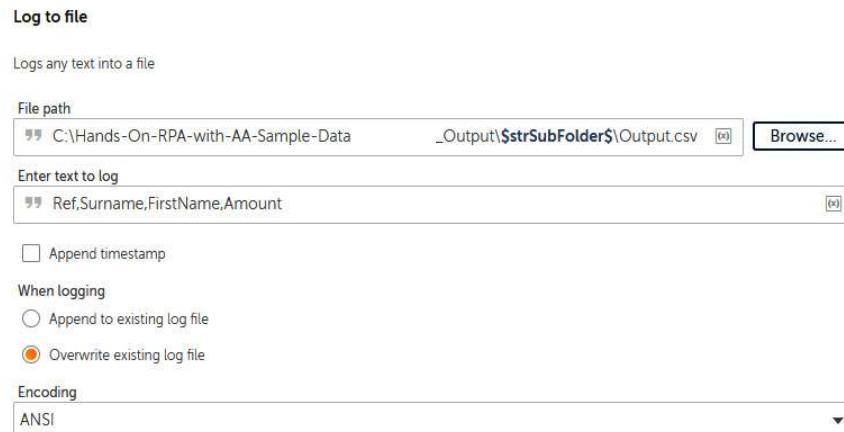
e.g. C:\MyDoc\MyNewFolder

Overwrite an existing folder

Section 4 – Checking or creating a subfolder

Set the following properties for the Log to file action on line 17:

- File path: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$\Output.csv
- Enter text to log: Ref,Surname,FirstName,Amount
- When logging: Overwrite existing log file
- The properties window should look like this:



Section 4 – Checking or creating a subfolder

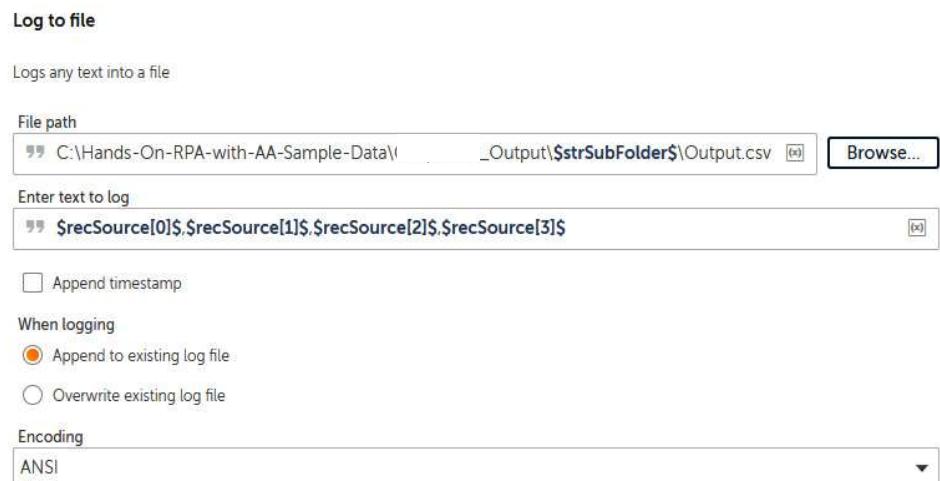
- Checking or creating a subfolder, complete! Our bot will create the necessary subfolder and file if needed.
- The development window for this section should look like this:

```
13  ┌─ Comment "----- Section 4 - Check if sub folder exists"
14  ▀ └─ ◊ If folder does not exist at "C:\Hands-On-RPA-with-AA-Sample-Data\"
15    ┌─ Comment "----- Section 4a - Create sub folder & output file if it doesn't exist"
16    └─ ┌─ Folder: Create "C:\Hands-On-RPA-with-AA-Sample-Data\"
17    └─ ┌─ Log to file "Ref,Surname,FirstName,A..." to "C:\Hands-On-RPA-with-AA-Sample-Data\"
      └─ └─
```

Section 5 – Updating the output file

Set the following properties for the Log to file action on line 19:

- File path: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$sSubFolder\$\Output.csv
- Enter text to log:
\$recSource[0]\$, \$recSource[1]\$, \$recSource[2]\$, \$recSource[3]\$
- When logging: Append existing log file
- The properties window should look like this:



Section 5 – Updating the output file

- And that's it – great work! Our bot is now complete.
- The development interface for this section should look like this

```
18  // Comment "----- Section 5 - Update output file" :  
19  Log to file "$recSource[0]$,${recSour...}" to "C:\Hands-On-RPA-with-AA-Sample-Data\  
           _Output\$strSubFolder$\Output.csv" :
```

Section 6 – Closing the source file

- Set the following properties for the CSV/TXT: Close action on line 21:Session name: Data
- The properties window should look like this:



Section 6 – Closing the source file

- And that's it – great work! Our bot is now complete.
- The development interface for this section should look like this:



Working with the filesystem

- There may be instances where we must create, delete, move, or rename files and folders.
- Automation Anywhere has two packages: one for files and one for folders.
- In the following screenshot, you can see the actions that are available for managing the filesystem in Automation Anywhere:

File	Folder
📄 Assign	📁 Zip
📄 Copy	📁 Copy
📄 Create	📁 Create
📄 Delete	📁 Unzip
📄 Download CR file	📁 Delete
📄 Open	📁 Open
📄 Print	📁 Rename
📄 Print multiple files	
📄 Rename	

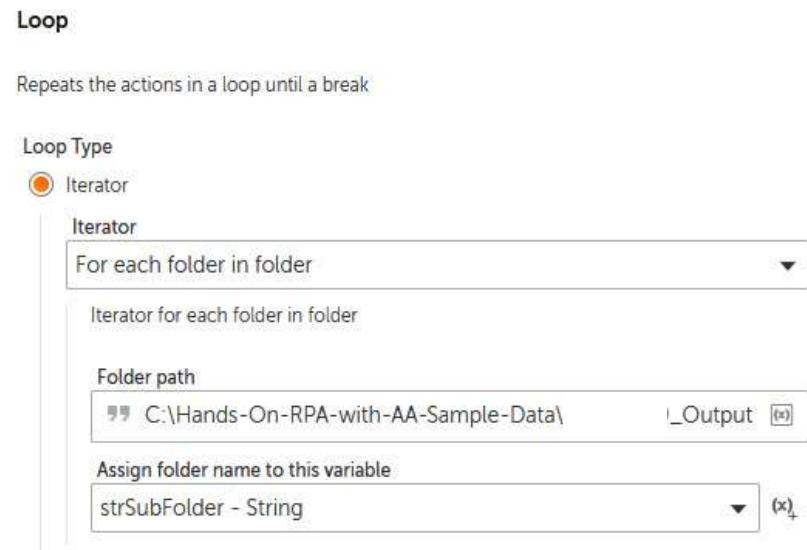
Working with the filesystem

- Add a new Comment action called "-----" on line 7 and click on Save.
- Your bot should look like this:

1	#[Comment -----	:
2	#[Comment ----- Section 1 - Loop through Sub Folders	:
3	#[Comment ----- Section 2 - Rename output file	:
4	#[Comment ----- Section 3 - Copy output file	:
5	#[Comment ----- Section 4 - Delete file	:
6	#[Comment ----- Section 5 - Delete Sub Folder	:
7	#[Comment -----	:

Section 1 – Looping through subfolders

- Set the following properties for the Loop action on line 3:Loop Type: For each folder in folder
- Folder path: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output
- Assign folder name to this variable: strSubFolder - String
- The properties window should look like this:



Section 1 – Looping through subfolders

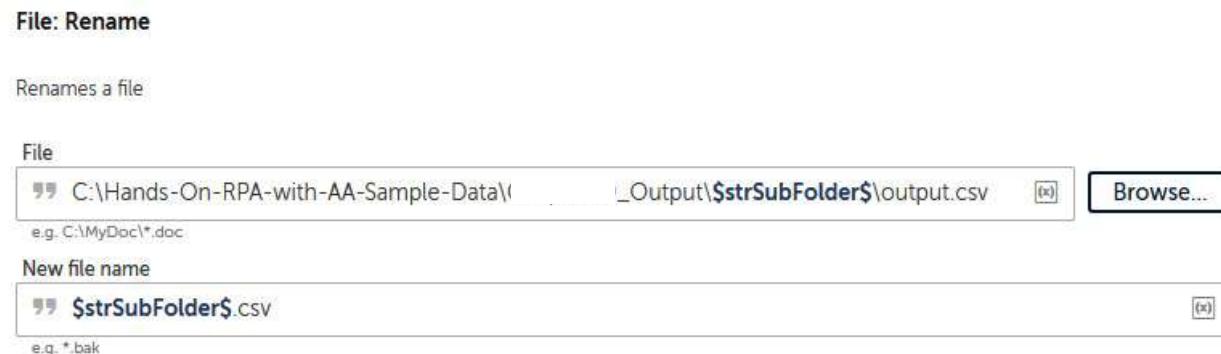
- Since all the other sections will be completed within this Loop, drag Comment lines 4 to 7 so that they are within the Loop element on line 3.
- The development interface should look like this:

```
1  // Comment "-----"
2  // Comment "----- Section 1 - Loop through Sub Folders"
3  ↗ Loop for each folder and assign folder name to $sSubFolder$ :
4  // Comment "----- Section 2 - Rename output file"
5  // Comment "----- Section 3 - Copy output file"
6  // Comment "----- Section 4 -Delete file"
7  // Comment "----- Section 5 - Delete Sub Folder"
8  // Comment "-----"
```

Section 2 – Renaming the output file

Set the following properties for the File: Rename action on line 5:

- File: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$\output.csv
- New file name: \$strSubFolder\$.csv
- The properties window should look like this:



Section 3 – Copying the output file

- Set the following properties for the File: Copy action on line 7:
- Source file: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$sSubFolder\$\\$strSubFolder\$.csv
- Destination file/folder: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$.csv
- The properties window should look like this:



Section 4 – Deleting the output file

Set the following properties for the File: Delete action on line 9:

- File: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$\\$strSubFolder\$.csv
- The properties window should look like this:

File: Delete

Deletes a file

File

C:\Hands-On-RPA-with-AA-Sample-Data\

_Output\\$strSubFolder\$\\$strSubFolder\$.csv



Browse...

e.g. C:\MyDoc*.doc

Section 5 – Deleting the subfolder

- Set the following properties for the Folder: Delete action on line 11:
- Folder: C:\Hands-On-RPA-with-AA-Sample-Data\lesson09_Output\\$strSubFolder\$
- The properties window should look like this:

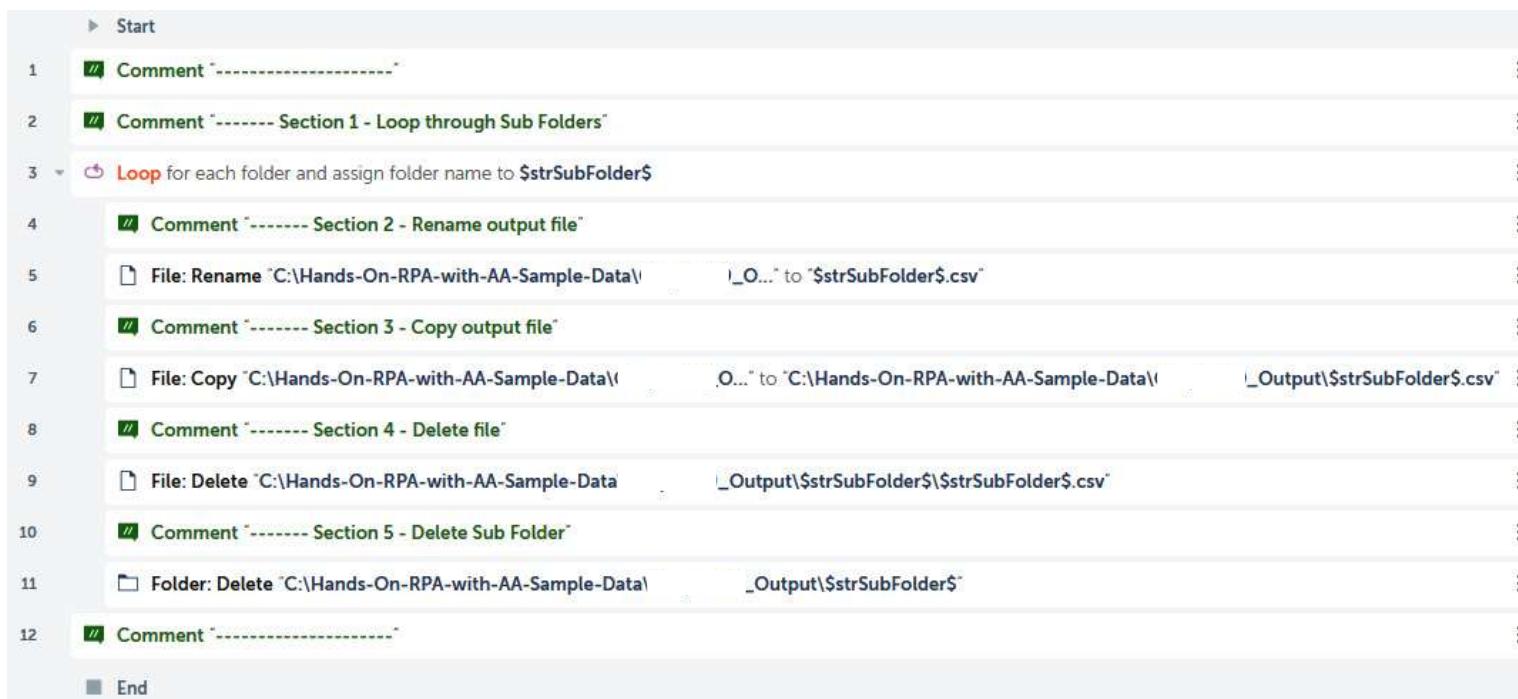
Folder: Delete

Deletes a folder



Section 5 – Deleting the subfolder

- Now that the bot is complete, you can go ahead and run it. The development interface should look like this:



The screenshot shows a sequence of 12 steps in a development interface:

- 1 Comment "-----"
- 2 Comment "----- Section 1 - Loop through Sub Folders"
- 3 Loop for each folder and assign folder name to \$strSubFolder\$
 - 4 Comment "----- Section 2 - Rename output file"
 - 5 File: Rename "C:\Hands-On-RPA-with-AA-Sample-Data__O..." to "\$strSubFolder\$.csv"
 - 6 Comment "----- Section 3 - Copy output file"
 - 7 File: Copy "C:\Hands-On-RPA-with-AA-Sample-Data__O..." to "C:\Hands-On-RPA-with-AA-Sample-Data__Output\\$strSubFolder\$\\$strSubFolder\$.csv"
 - 8 Comment "----- Section 4 - Delete file"
 - 9 File: Delete "C:\Hands-On-RPA-with-AA-Sample-Data__Output\\$strSubFolder\$\\$strSubFolder\$.csv"
 - 10 Comment "----- Section 5 - Delete Sub Folder"
 - 11 Folder: Delete "C:\Hands-On-RPA-with-AA-Sample-Data__Output\\$strSubFolder\$"
 - 12 Comment "-----"
- End

Summary

- We have covered a lot of packages. Just to recap once more, first, we built some nested loops that went through lists, records, and folders. We also discovered a new Record data type variable.
- You were then given some exposure to all the other types of looping available, all of which we will cover in later lessons.
- The walk-throughs also included building some rule-based decisions with conditional statements, using variables, and checking if files exist.

Working with XML Files



Working with XML Files

- To change things up a little, we will not be building a single bot to demonstrate XML actions. Instead, we will go through the individual actions within the XML package.
- Each walk-through will provide a step-by-step guide that shows you how to use each action. We will be using the following packages:

<input checked="" type="checkbox"/> Comment	<input checked="" type="checkbox"/> NumberUtils
<input checked="" type="checkbox"/> If	<input checked="" type="checkbox"/> Step
<input checked="" type="checkbox"/> Loop	<input checked="" type="checkbox"/> String
<input checked="" type="checkbox"/> Message box	<input checked="" type="checkbox"/> XML

Working with XML Files

In this lesson, we will cover the following topics:

- Starting, validating, and ending XML sessions
- Reading and updating XML nodes
- Inserting and deleting XML nodes
- Executing XPath functions

Starting, validating, and ending XML sessions

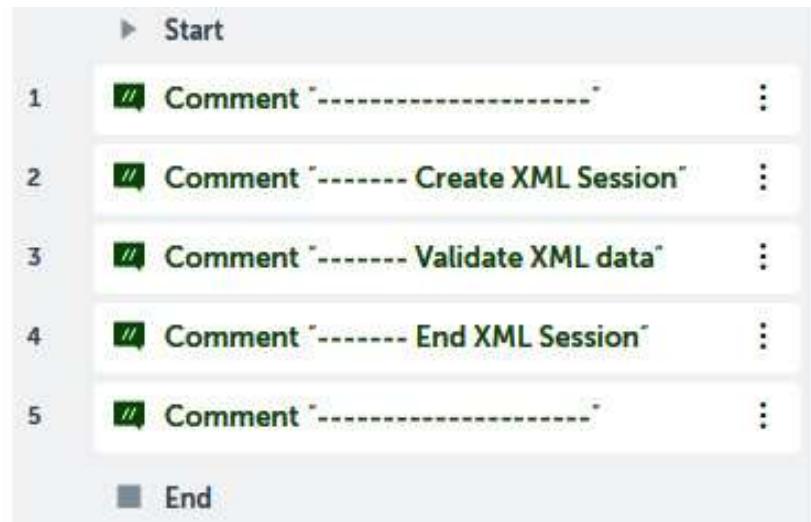


The screenshot shows a Windows Explorer window with several files listed in the title bar: Class1.cs, credentials.txt, Structure.txt, new 1.txt, and Chapter10_SampleFile.xml. The Chapter10_SampleFile.xml file is highlighted with a red border, indicating it is the active file. The main pane displays the XML code:

```
1 <?xml version="1.0"?>
2 <catalog>
3   <book id="bk101">
4     <author>Gambardella, Matthew</author>
5     <title>XML Developer's Guide</title>
6     <genre>Computer</genre>
7     <price>44.95</price>
8     <publish_date>2000-10-01</publish_date>
9     <description>An in-depth look at creating applications
10    with XML.</description>
11  </book>
12  <book id="bk102">
13    <author>Ralls, Kim</author>
14    <title>Midnight Rain</title>
15    <genre>Fantasy</genre>
16    <price>5.95</price>
17    <publish_date>2000-12-16</publish_date>
18    <description>A former architect battles corporate zombies,
19    an evil sorceress, and her own childhood to become queen
20    of the world.</description>
21  </book>
```

Starting, validating, and ending XML sessions

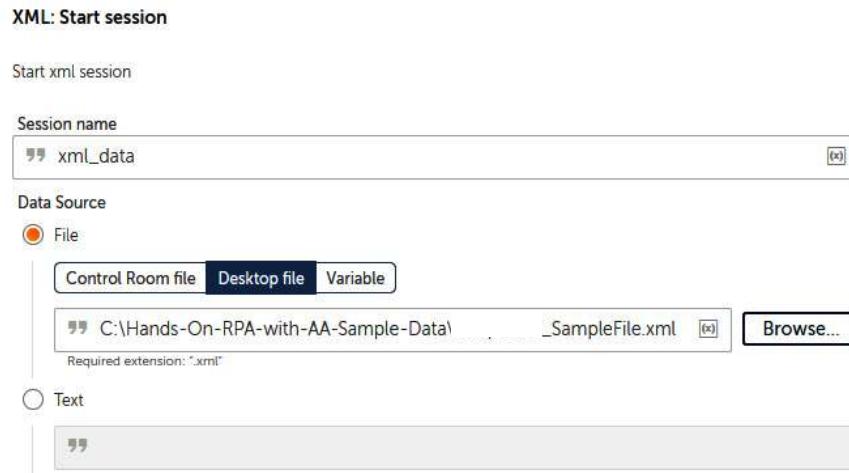
- Add another Comment action called "-----" on line 5 and click on Save.
- Our initial development interface should look like this:



Starting, validating, and ending XML sessions

Set the following properties for the XML: Start session action on line 3:Session name: xml_data

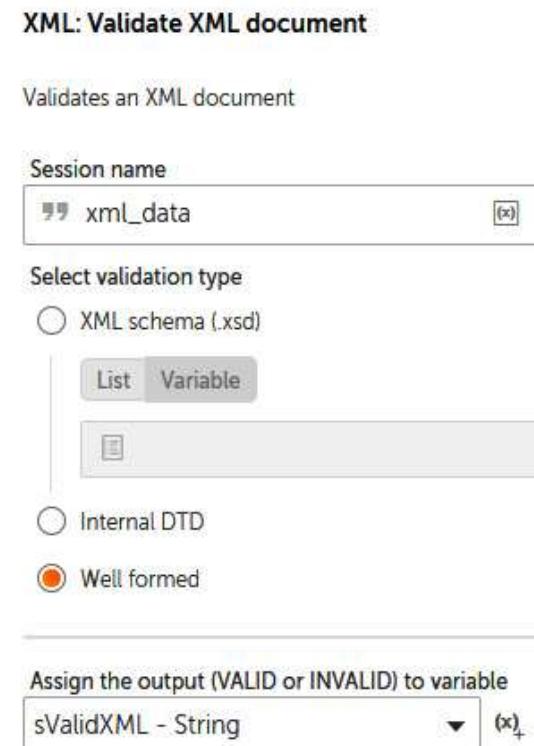
- Data Source: File
- Desktop file: C:\Hands-On-RPA-with-AA-Sample-Data\lesson10_SampleFile.xml
- The properties window should look like this:



Starting, validating, and ending XML sessions

Set the following properties for the XML:
Validate XML document action on
line 5:Session name: xml_data

- Select validation type: Well formed
- Assign the output to variable: sValidXML - String
- The properties window should look like this:



Starting, validating, and ending XML sessions

Set the following properties for the If action on line 6:

- Condition: String condition
- Source value: \$sValidXML\$
- Operator: Equals to (=)
- Target value: VALID
- Match case: Checked
- The properties window should look like this:

If

Runs a sequence of actions if a condition is true

Condition

String condition

Checks the string variable condition.

Source value (optional)
\$sValidXML\$

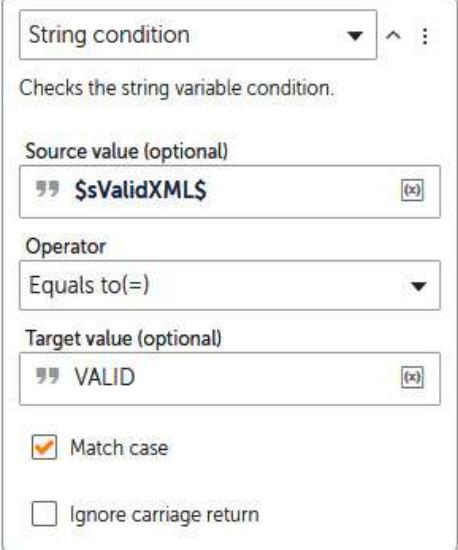
Operator
Equals to(=)

Target value (optional)
VALID

Match case

Ignore carriage return

Add condition



Starting, validating, and ending XML sessions

Set the Message box action properties on line 7 to the following:

- Enter the message box window title: XML - Validation Check
- Enter the message to display: Result: \$sValidXML\$ - Bot will continue...
- The properties window should look like this:

Message box

Displays a message box

Enter the message box window title

XML - Validation Check

Enter the message to display

Result: \$sValidXML\$ - Bot will continue...

Scrollbar after lines

30

Close message box after

Seconds

#

Starting, validating, and ending XML sessions

Set the Message box action properties on line 9 to the following:

- Enter the message box window title: XML – Validation Check
- Enter the message to display: Result: \$sValidXML\$ - Bot will Stop!
- The properties window should look like this:

Message box

Displays a message box

Enter the message box window title

XML - Validation Check

Enter the message to display

Result: \$sValidXML\$ - Bot will stop!

Scrollbar after lines

30

Close message box after

Seconds

#

Starting, validating, and ending XML sessions

Set the following properties for the XML: End session action on line 11:

- Session name: `xml_data`
- The properties window should look like this:

XML: End session

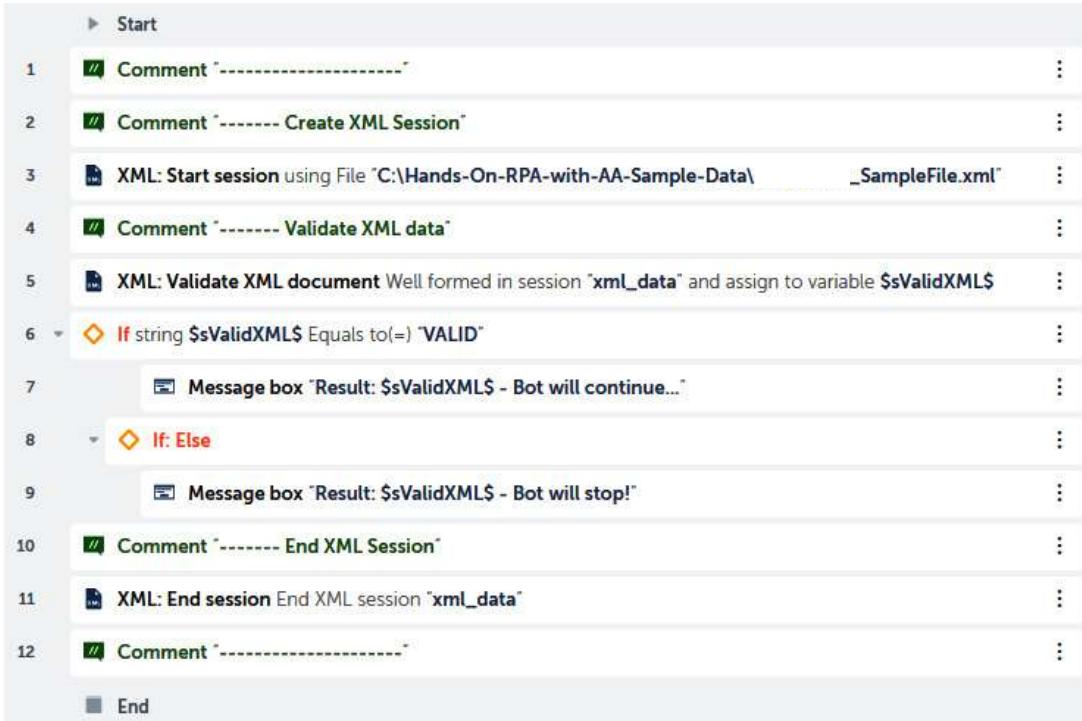
End XML session

Session name

 (x)

Starting, validating, and ending XML sessions

- Click on Save.
- Your development interface should look like this:



The screenshot shows a sequence of 12 numbered steps in a RPA development interface:

1. Comment "-----"
2. Comment "----- Create XML Session"
3. XML: Start session using File "C:\Hands-On-RPA-with-AA-Sample-Data\ _SampleFile.xml"
4. Comment "----- Validate XML data"
5. XML: Validate XML document Well formed in session "xml_data" and assign to variable \$sValidXML\$
6. If string \$sValidXML\$ Equals to(=) "VALID"
 - 7. Message box "Result: \$sValidXML\$ - Bot will continue..."
 - 8. If: Else
 - 9. Message box "Result: \$sValidXML\$ - Bot will stop!"
10. Comment "----- End XML Session"
11. XML: End session End XML session "xml_data"
12. Comment "-----"

The interface includes a "Start" button at the top left and an "End" button at the bottom right.

Reading XML nodes

- There may be many instances where you need to look up a single value from your XML data stream, especially when the XML data is being used for reference purposes.
- We will continue to build on our walk-through by providing you with steps on how to read a specific single node, as well as multiple nodes, from our XML file.

Reading a single node

- Set the Title property of the Step action on line 8 to Reading a Single Node.
- The properties window should look like this:

Step

Runs a sequence of commands.

Title (optional)

” Reading a Single Node



Reading a single node

Set the following properties for the XML: Get single node action on line 9:

- Session name: xml_data
- XPath expression: book[2]/title
- Assign the output to variable: sTitle - String
- The properties window should look like this:

XML: Get single node

Fetch value of a specific node from the xml

Session name

” xml_data (x)

XPath expression

” book[2]/title (x)

e.g. /bookstore/book/title

Attribute (optional)

” (x)

Assign the output to variable

sTitle - String ▼ (x)

Reading a single node

Set the Message box action properties on line 10 to the following:

- Enter the message box window title: Reading a Single Node
- Enter the message to display: Book Title: |\$sTitle\$|
- The properties window should look like this:

Message box

Displays a message box

Enter the message box window title

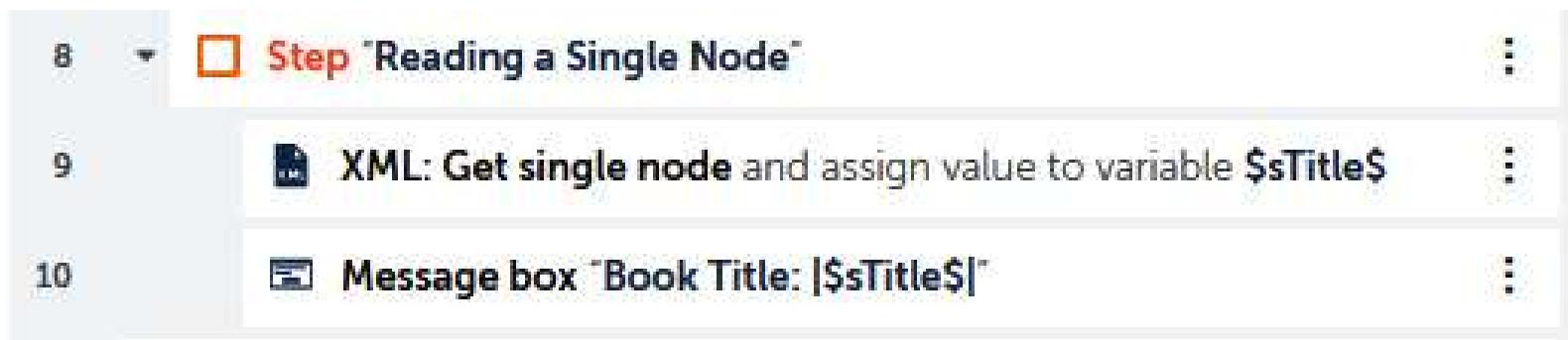
” Reading a Single node

Enter the message to display

” Book Title: |\$sTitle\$|

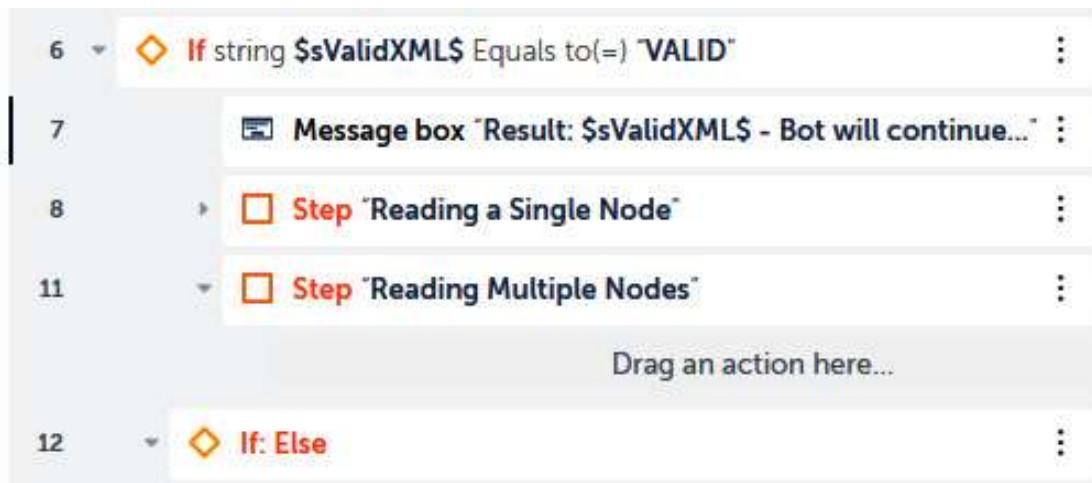
Reading a single node

- Great work – you have just created your first complete Step! This Step has been tasked with reading a single node.
- In this example, we specified the second record and the node name. Remember that you can always replace these hard values with variables if needed.
- Your Step action in the development interface should look like this:



Reading multiple nodes

- Set the Title property of this new Step action on line 11 to Reading Multiple Nodes.
- Your development interface for this section should look like this:



Reading multiple nodes

Set the following properties for the XML:
Get multiple nodes action on line 12:

- Session name: xml_data
- XPath expression: book/title
- Get each node: Text value
- The properties window should look like this:

XML: Get multiple nodes

Fetch value from multiple xml nodes

Session name

XPath Expression

For example //bookstore/book

Get each node

- Text value
- Xpath expression
- Specific attribute name

Reading multiple nodes

Set the following properties for the Loop action on line 13:

- Iterator: For each Node in a XML Dataset
- Session name: xml_data
- Assign the current row to this variable:
sTitle - String
- The properties window should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each Node in a XML Dataset ▾

Iterator for each node in XML Dataset

Session name

xml_data (x)

Assign the current row to this variable

sTitle - String ▾ (x)

Reading multiple nodes

Set the following properties for the Message Box action on line 14:

- Enter the message box window title: Reading Multiple Nodes
- Enter the message to display: Book Title: |\$sTitle\$|
- The properties window should look like this:

Message box

Displays a message box

Enter the message box window title

” Reading Multiple Nodes

Enter the message to display

” Book Title: |\$sTitle\$|

Reading multiple nodes

- Click on Save.
- Your development window should look like this:



Reading multiple nodes

```
> Start
1  // Comment -----
2  // Comment ----- Create XML Session
3  XML: Start session using File "C:\Hands-On-RPA-with-AA-Sample-Data\ _SampleFile.xml"
4  // Comment ----- Validate XML data
5  XML: Validate XML document Well formed in session "xml_data" and assign to variable $sValidXML$ 
6  ◊ If string $sValidXML$ Equals to(=) "VALID"
7    ☐ Message box "Result: $sValidXML$ - Bot will continue..."
8    □ Step "Reading a Single Node"
11   □ Step "Reading Multiple Nodes"
15   ◊ If: Else
16     ☐ Message box "Result: $sValidXML$ - Bot will stop!"
17   // Comment ----- End XML Session
18   XML: End session End XML session "xml_data"
19   // Comment -----
      End
```

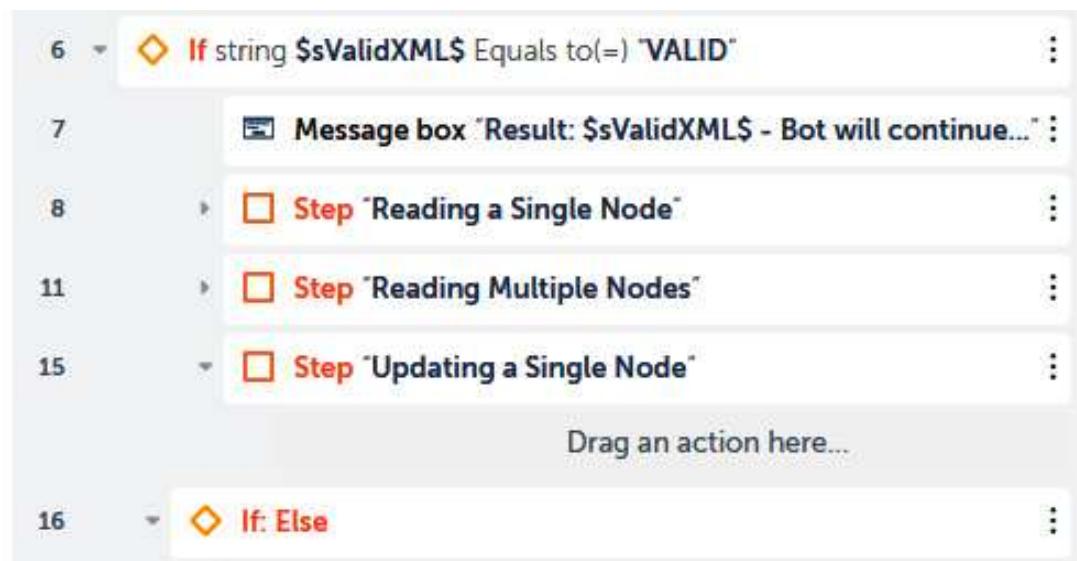
Updating a single node

- From our sample XML file, we will be updating the genre for the first record.
- It is currently set to Computer. We will update this to Computer – Software.
- We know that the record number is 1, and we also know the node is genre, In our XML file, this record currently looks like this:

```
<book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
    with XML.</description>
</book>
```

Updating a single node

- Set the Title property of the Step action on line 15 to Updating a Single Node.
- Your development interface should look like this:



Reading multiple nodes

Set the following properties for the XML: Update node action on line 16:

- Session name: xml_data
- XPath expression: book[1]/genre
- New value: Computer - Software
- The properties window should look like this:

XML: Update node

Update specific node in the xml

Session name

XPath expression

For example //bookstore/book

New value

Reading multiple nodes

XML: Save session data

Save XML session data

Session name

xml_data (x)

Write XML data

File path

C:\Hands-On-RPA-with-AA-Sample-Data\ _SampleFile.xml (x)

Browse...

Required extension: ".xml"

Overwrite

Assign the output to variable

sXML_DataStream - String

(x)

Reading multiple nodes

Message box

Displays a message box

Enter the message box window title

" Updating a Single Node



Enter the message to display

" Record Number: 1

Previous Genre: Computer

New Genre: Computer Programming



Reading multiple nodes

- That wasn't too difficult! With that, you have learned how to update a single node, as long as you know which record it is for.
- Your Step action should look like this in the development interface:

15	<input type="checkbox"/> Step "Updating a Single Node"	:
16	XML: Update node Update node value "Computer - Software" at xpath location "book[1]/genre" in session "xml_data"	:
17	XML: Save session data assigned to variable \$sXML_DataStream\$ write session data into file "C:\Hands-On-RPA-with-AA-Sample..."	:
18	Message box "Record Number: 1 Previous Genre: Computer New Genre: Computer Programming"	:

Reading multiple nodes

- You should be getting the hang of this by now.
- Once you have run the bot, have a look at your XML file; the first record should have been updated, It should look as follows:

```
<book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer - Software</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
    with XML.</description>
</book>
```

Updating multiple nodes

Let's start this walk-through by executing the following steps:

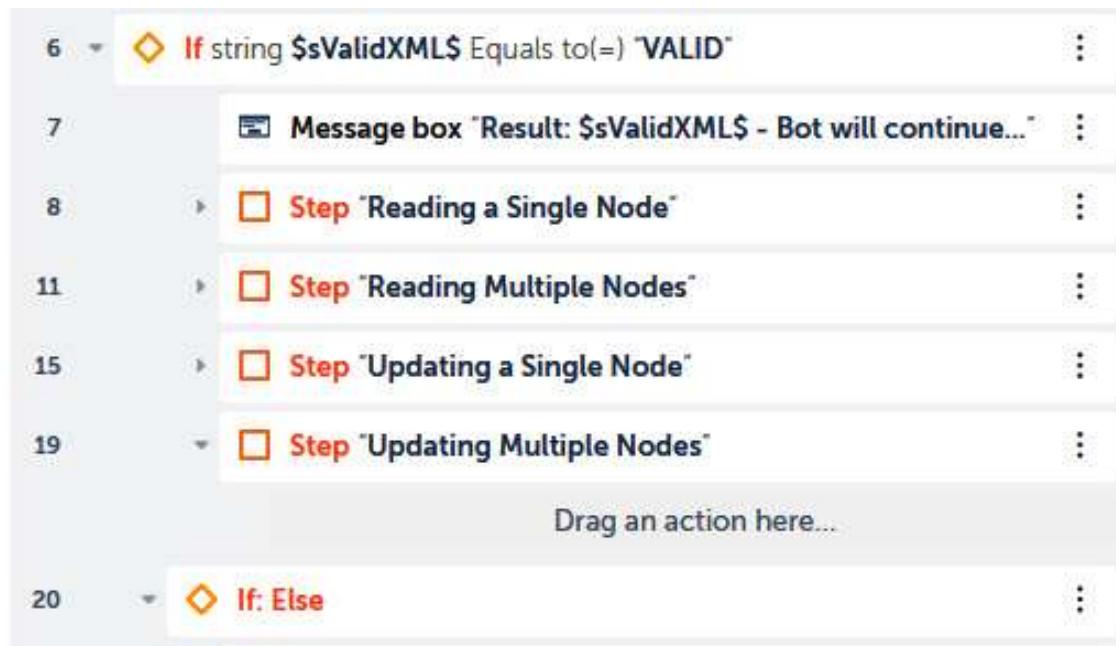
- Create three String type variables called sRecordNum, sOldPrice, and sNewPrice.
- Create three Number type variables called nRecordNum, nOldPrice, and nNewPrice.
- Your variable list should look like this:

User-defined

```
# nNewPrice  
# nOldPrice  
# nRecordNum  
## sNewPrice  
## sOldPrice  
## sRecordNum  
## sTitle  
## sValidXML  
## sXML_DataStream
```

Updating multiple nodes

- Set the Title property of this new Step action on line 19 to Updating Multiple Nodes.
- Your development interface should look like this:



Updating multiple nodes

Set the following properties for the XML: Get multiple nodes action on line 20:

- Session name: xml_data
- XPath expression: book/price
- Get each node: Text value
- The properties window should look like this:

XML: Get multiple nodes

Fetch value from multiple xml nodes

Session name

” xml_data

XPath Expression

” book/price

For example //bookstore/book

Get each node

Text value

Xpath expression

Specific attribute name

”

Updating multiple nodes

Set the following properties for the Loop action on line 21:

- Iterator: For each Node in a XML Dataset
- Session name: xml_data
- Assign the current row to this variable: sOldPrice - String
- The properties window should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each Node in a XML Dataset

Iterator for each node in XML Dataset

Session name

xml_data

Assign the current row to this variable

sOldPrice - String

Updating multiple nodes

Set the following properties for the Number: Assign action on line 22:

- Select the source string variable/ value: \$nRecordNum\$ + 1
- Select the destination number variable: nRecordNum – Number

The properties window should look like this:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$nRecordNum\$ + 1 (x)

Specify value to assign to number

Select the destination number variable

nRecordNum - Number ▼ (x)

Updating multiple nodes

Set the following properties for the Number: To string action on line 23:

- Enter a number: \$nRecordNum\$
- Enter the number of digits after decimal: 0
- Assign the output to variable: sRecordNum - String
- The properties window should look like this:

Number: To string

Converts a user specified number to a string

Enter a number

\$nRecordNum\$

(x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

0

(x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

sRecordNum - String

▼

(x)

Updating multiple nodes

Set the following properties for the String: To number action on line 24:

- Enter the string: \$sOldPrice\$
- Assign the output to variable: nOldPrice - Number
- The properties window should look like this:

String: To number

Converts a string to a number

Enter the string

” \$sOldPrice\$ (x)

String entered must be a valid number

Assign the output to variable

nOldPrice - Number ▼ (x)

Updating multiple nodes

Set the following properties for the NumberUtils: Calc action on line 25:

- Expression to be calculated:
\$nOldPrice\$ * 1.1
- Calculated Expression:
nNewPrice - Number
- The properties window should look like this:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$nOldPrice\$ * 1.1

Specify value to assign to number

Select the destination number variable

nNewPrice - Number

(x) 

Updating multiple nodes

Set the following properties for the Number: To string action on line 26:

- Enter a number: \$nNewPrice\$
- Enter number of digits after decimal: 2
- Assign the output to variable: nNewPrice – String
- The properties window should look like this:

Number: To string

Converts a user specified number to a string

Enter a number

\$nNewPrice\$ (x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

2 (x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

sNewPrice - String (x) +

Updating multiple nodes

Set the following properties for the XML: Update node action on line 27:

- Session name: XML_data
- XPath expression:
book[\$sRecordNum\$]/price
- New value: \$sNewPrice\$
- The properties window should look like this:

XML: Update node

Update specific node in the xml

Session name

xml_data

XPath expression

book[\$sRecordNum\$]/price

For example //bookstore/book

New value

\$sNewPrice\$

Updating multiple nodes

Message box

Displays a message box

Enter the message box window title

Updating Multiple Nodes

Enter the message to display

Record Number: \$sRecordNum\$
Old Price: \$sOldPrice\$
New Price: \$sNewPrice\$

Scrollbar after lines

30

Close message box after

Seconds

5

Updating multiple nodes

XML: Save session data

Save XML session data

Session name

xml_data (x)

Write XML data

File path

C:\Hands-On-RPA-with-AA-Sample-Data\

_SampleFile.xml (x)

Browse...

Required extension: ".xml"

Overwrite

Assign the output to variable

sXML_DataStream - String

▼ (x)

Updating multiple nodes

```
19      □ Step "Updating Multiple Nodes"
20          XML: Get multiple nodes Text value from xpath location "book/price" session "xml_data"
21          ▶ ⏪ Loop Each node In a XML Dataset "xml_data"
22              # Number: Assign "$nRecordNum$ + 1" to $nRecordNum$
23              # Number: To string convert $nRecordNum$ to a string datatype and assign output to $sRecordNum$
24              # String: To number Convert string $sOldPrice$ to a number and assign it to number variable $nOldPrice$
25              # Number: Assign '$nOldPrice$ * 1.1' to $nNewPrice$
26              # Number: To string convert $nNewPrice$ to a string datatype and assign output to $sNewPrice$
27              XML: Update node Update node value $sNewPrice$ at xpath location "book[$sRecordNum$/price" in session "xml_data"
28              Message box "Record Number: $sRecordNum$ Old Price: $sOldPrice$ New Price: $sNewPrice$"
29              XML: Save session data assigned to variable $sXML_DataStream$ write session data into file "C:\Hands-On-RPA-with-AA-Sa...
```

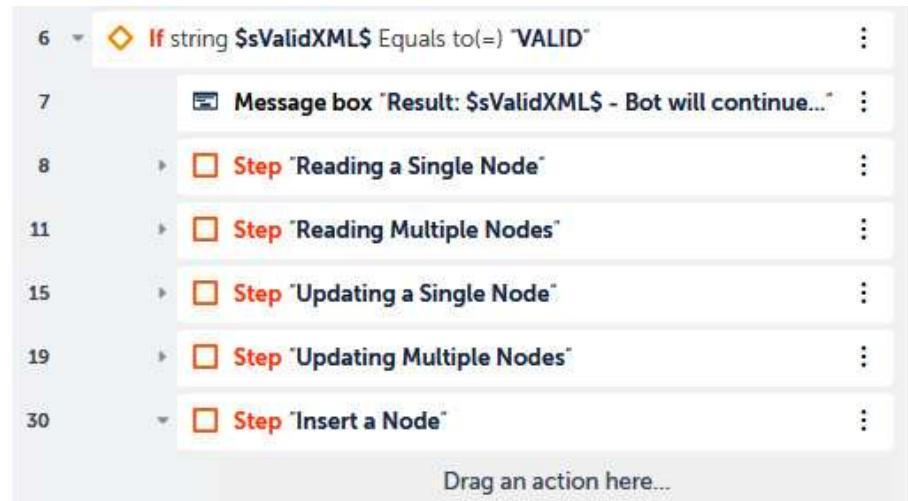
Updating multiple nodes

```
1  // Comment "-----"  
2  // Comment "----- Create XML Session"  
3  XML: Start session using File 'C:\Hands-On-RPA-with-AA-Sample-Data\....._SampleFile.xml'  
4  // Comment "----- Validate XML data"  
5  XML: Validate XML document Well formed in session "xml_data" and assign to variable $sValidXML$  
6  □ If string $sValidXML$ Equals to(=) "VALID"  
7    └ Message box "Result: $sValidXML$ - Bot will continue..."  
8    □ Step "Reading a Single Node"  
11   □ Step "Reading Multiple Nodes"  
15   □ Step "Updating a Single Node"  
19   □ Step "Updating Multiple Nodes"  
30  □ If: Else  
31    └ Message box "Result: $sValidXML$ - Bot will stop!"  
32  // Comment "----- End XML Session"  
33  XML: End session End XML session "xml_data"  
34  // Comment "-----"
```

Inserting XML nodes

- Add a Step action to line 30, ensuring it is aligned with the previous Step action on line 19.
- Set the Title property of this Step action on line 30 to Inserting a Node.

Your development interface should look like this:



Inserting XML nodes

Set the following properties for the XML: Insert node action on line 31:

- Session name: xml_data
- XPath Expression: book
- Node name: format
- Node value: Paperback
- If node name is present then: Insert it anyways
- Insert node location: End of child nodes

The properties window should look like this:

XML: Insert node

Insert node within xml

Session name

xml_data (x)

XPath Expression

book (x)

For example //bookstore/book

Node name

format (x)

Node value (optional)

Paperback (x)

If node name is present then

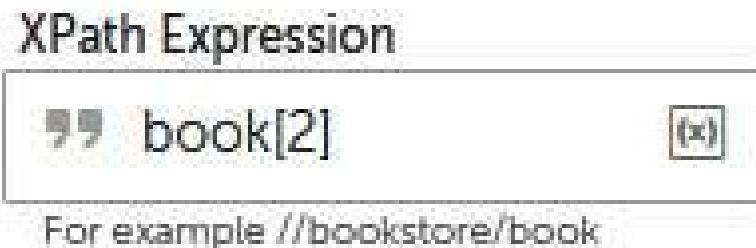
Insert it anyways ▼

Insert node location

End of the child nodes ▼

Inserting XML nodes

- There may be instances where you may want to just insert a node for a specific record.
- To do this, simply specify the record number when setting the XPath Expression property, as follows:



Inserting XML nodes

XML: Save session data

Save XML session data

Session name

xml_data (x)

Write XML data

File path

C:\Hands-On-RPA-with-AA-Sample-Data\ .SampleFile.xml (x)

Browse...

Required extension: ".xml"

Overwrite

Assign the output to variable

sXML_DataStream - String

▼ (x)

Inserting XML nodes

Message box

Displays a message box

Enter the message box window title

99 Inserting a Node



Enter the message to display

99 Node Inserted: format, value Paperback

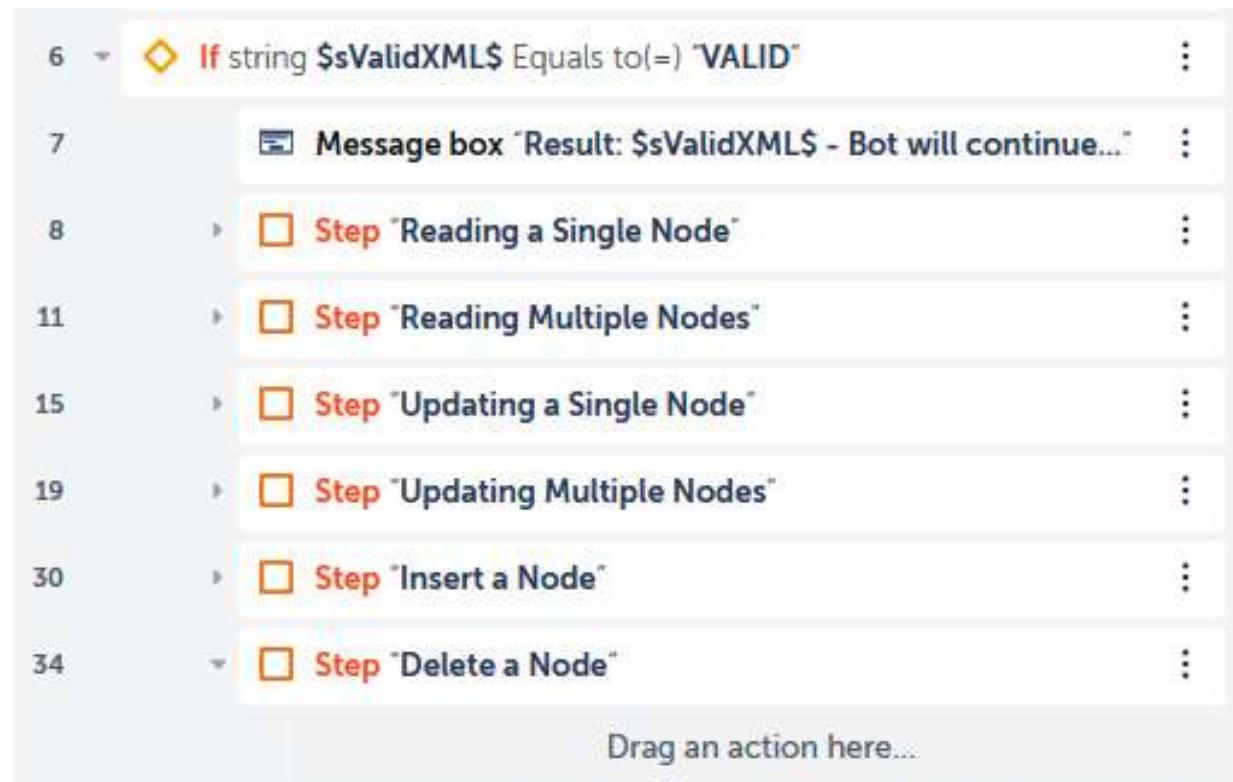


Inserting XML nodes

- Your Step action should look like this in the development interface:

30	<input type="checkbox"/> Step 'Insert a Node'	:
31	XML: Insert node Insert node with name "format" and value "Paperback" at xpath location "book" in session "xml_data"	:
32	XML: Save session data assigned to variable \$sXML_DataStream\$ write session data into file "C:\Hands-On-RPA-with-AA-Sample..."	:
33	Message box "Node Inserted: format, value Paperback"	:

Deleting XML nodes



Deleting XML nodes

Set the following properties for the XML: Delete node action on line 35:

- Session name: xml_data
- XPath Expression: book/format
- The properties window should look like this:

XML: Delete node

Delete specific node from the xml.

Session name

xml_data

XPath expression

book/format

For example //bookstore/book

Attribute (optional)

Deleting XML nodes

- In cases where you need to delete a node for a specific record, just specify the record number when setting the XPath Expression property, as shown in the following screenshot:



Deleting XML nodes

XML: Save session data

Save XML session data

Session name:

xml_data (x)

Write XML data

File path

C:\Hands-On-RPA-with-AA-Sample-Data\

SampleFile.xml (x)

Browse...

Required extension: ".xml"

Overwrite

Assign the output to variable

sXML_DataStream - String

(x)

Deleting XML nodes

Message box

Displays a message box

Enter the message box window title

” Deleting a Node

(x)

Enter the message to display

” Node Deleted: format, value Paperback

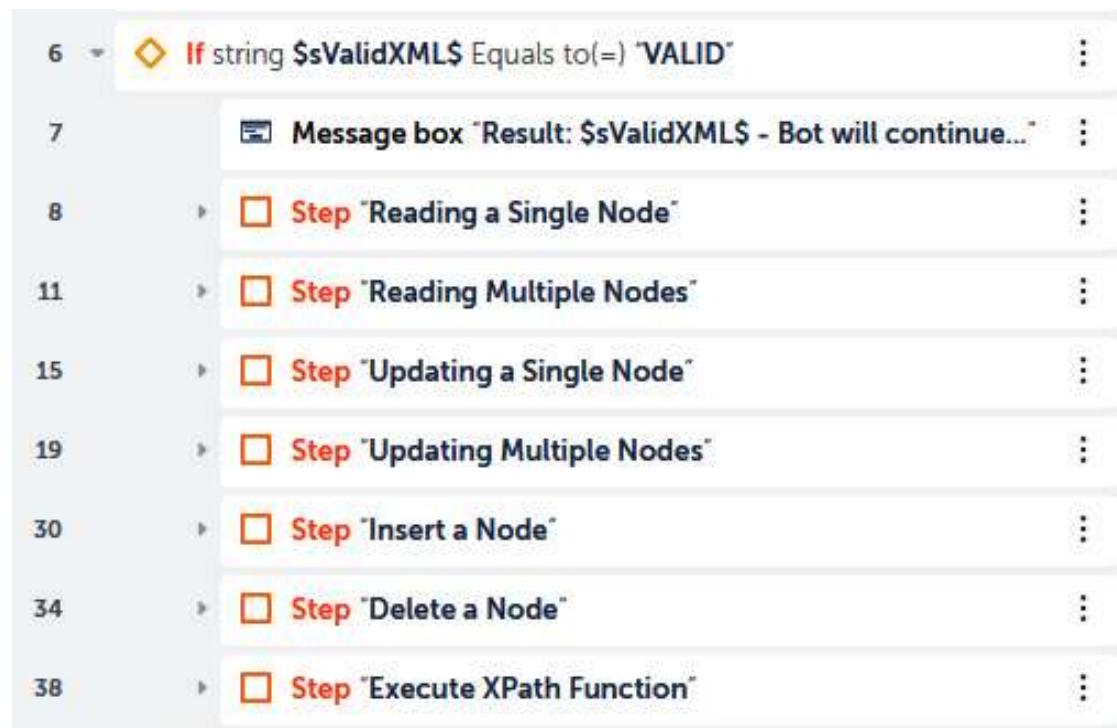
(x)

Deleting XML nodes

- Click on Save.
- Your Step action should look like this in the development interface:

34	<input type="checkbox"/> Step "Delete a Node"	⋮
35	XML: Delete node Delete node attribute "" at xpath location "book/format" in session "xml_data"	⋮
36	XML: Save session data assigned to variable \$sXML_DataStream\$ write session data into file "C:\Hands-On-RPA-...	⋮
37	Message box "Node Deleted: format, value Paperback"	⋮

Executing XPath functions



Executing XPath functions

XML: Execute XPath function

Executes the XPath function on the XML

Session name

xml_data

(x)

XPath expression

count(/book)

(x)

e.g. /bookstore/book/title

Assign the output to variable

sBookCount - String

▼

(x)

Executing XPath functions

Message box

Displays a message box

Enter the message box window title

” Execute XPath Function



Enter the message to display

” Number of Books: \$sBookCount\$



Executing XPath functions

- Click on Save.
- Your Step action should look like this in the development interface:

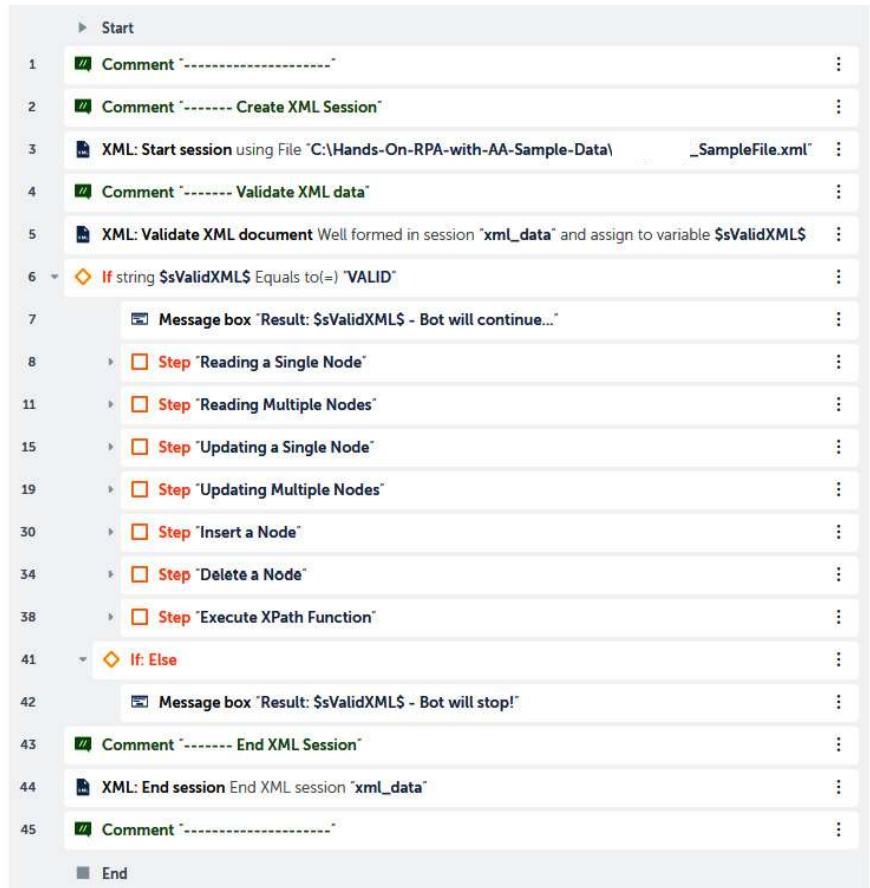
The screenshot shows a software interface for developing workflows or scripts. It displays a sequence of numbered steps:

- Step 38: Step "Execute XPath Function"
- Step 39: XML: Execute XPath function and assign value to variable \$sBookCount\$
- Step 40: Message box "Number of Books: \$sBookCount\$"

Each step has a small icon to its left and a vertical ellipsis button to its right.

Executing XPath functions

- Now, for the final time, run the bot. You should get a result of 5.
- There are five book records in our sample XML file.
- Collapse all the steps; your development interface should look like this:



Summary

- We have covered every action that is available in the XML package and learned some great skills.
- This lesson's walk-throughs have taken you through a multitude of exercises, including reading single and multiple nodes from XML data streams.
- You also learned how to update single and multiple nodes within XML files.
- Furthermore, you've learned how to insert and delete nodes and values from XML files.

Automating Excel



Automating Excel

In this lesson, we will cover the following:

- Opening, closing, and saving Excel workbooks
- Reading and writing data within Excel worksheets
- Working with data in Excel
- Running macros

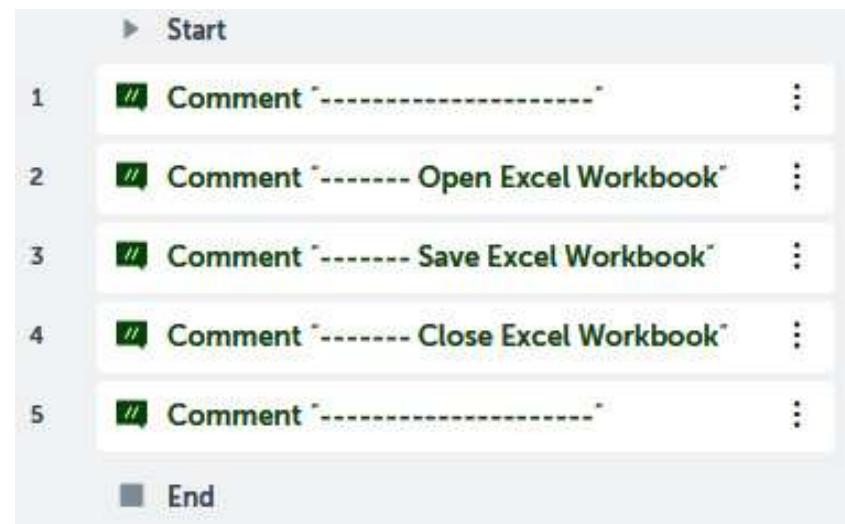
Automating Excel

- We will be taking the same approach as the previous lesson, with a number of walk-throughs exploring the different Excel actions.
- As before, we will be building our bots with multiple steps to help us keep things structured.
- We will be using the following packages:

	Comment	
	Excel advanced	
	Excel basic	
	Loop	
	Message box	
	Number	
	Step	

Opening, closing, and saving Excel workbooks

- Add a new Comment action as "-----" on line 5 and click on Save.
- Our initial development interface should look like this:



Opening, closing, and saving Excel workbooks

Excel basic: Open

Opens an excel spreadsheet

Session name

xl_data

e.g. Session1 or S1

File path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

Catalog.xlsx

Browse...

Required extension: ".xlsx"

e.g. C:\Working\Excel1.xlsx

Specific sheet name

Catalog

e.g. Sheet1 or SHEET1

Open in

Read-only mode

Read-write mode

Password is required

To open

Credential Variable Insecure string

Pick...

Sheet contains a header

Opening, closing, and saving Excel workbooks

- To save the workbook, drag the Excel basic: Save workbook action just below line 4.
- Set the following properties for the Excel basic: Save workbook action on line 5:
- Session name: xl_data

Excel basic: Save workbook

Saves an excel spreadsheet

Session name

” xl_data



e.g. Session1 or S1

Opening, closing, and saving Excel workbooks

Set the following properties for the Excel basic: Close action on line 7:

- Session name: xl_data
- Save changes when closing file: Unchecked
- The properties should look like this:

Excel basic: Close

Closes an excel spreadsheet

Session name

"xl_data"

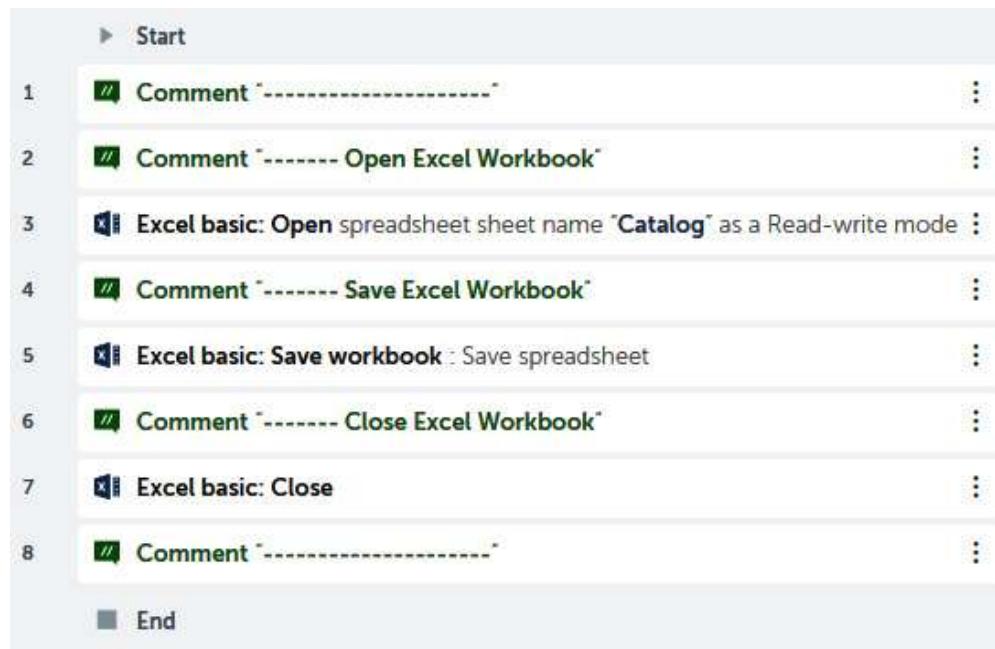
(x)

e.g. Session1 or S1

Save changes when closing file

Opening, closing, and saving Excel workbooks

- We have looked at the basics—opening, saving, and closing an Excel workbook.
- Your development window should look like this:



Reading from Excel worksheets

- Data in Excel is usually presented as a table, which means it consists of a fixed set of columns with each row as a record or transaction.
- The sample data file has a worksheet named Catalog.
- The dataset looks like this:

	A	B	C	D	E	F
1	ID	Author	Title	Genre	Price	Publish Date
2	bk101	Gambardella, Matthew	XML Developer's Guide	Computer - Software	44.95	01/10/2000
3	bk102	Ralls, Kim	Midnight Rain	Fantasy	5.95	16/12/2000
4	bk103	Corets, Eva	Maeve Ascendant	Fantasy	5.95	17/11/2000
5	bk104	Corets, Eva	Oberon's Legacy	Fantasy	5.95	10/03/2001
6	bk105	Corets, Eva	The Sundered Grail	Fantasy	5.95	10/09/2001



Reading from Excel worksheets

- Set the Title property of the Step on line 4 as Read Worksheet Records.
- Click on Save.
- Your development interface should look like this:



Reading from Excel worksheets

- Set the following properties for the Loop action on line 5:Loop Type*: For each row in worksheet

*When selecting this, ensure you select the Loop Type from the EXCEL BASIC package as this is the one used to create this session

EXCEL ADVANCED

For each row in worksheet

EXCEL BASIC

For each row in worksheet

Reading from Excel worksheets

- Session name: xl_data
- Loop through: All rows
- Assign the current value to this variable: recBook - Record

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in worksheet

Iterator for each row in Excel Output(s) will be assigned to a record variable

Session name

„ xl_data

(x)

Loop through

All rows

(▼)

Assign the current value to this variable

recBook - Record

(▼)

(x)

Reading from Excel worksheets

Message box

Displays a message box

Enter the message box window title

" " Reading Excel Worksheet

Enter the message to display

" " Title: \$recBook[2]\$ - Price: \$recBook[4]\$

Scrollbar after lines

30

Close message box after

Seconds

4

Reading from Excel worksheets

- Click on Save, the development interface for this section should look like this:



Reading from Excel worksheets

- Set the following properties for the Number: Increment action on line 6:
Enter number: \$numRecCount\$
- Enter increment value: 1
- Assign the output to variable: numRecCount - Number

The properties should look like this:

Number: Increment

Increments a number by specified value

Enter number

\$numRecCount\$

Enter increment value

1

Increments number by value (e.g. 1)

Assign the output to variable

numRecCount - Number

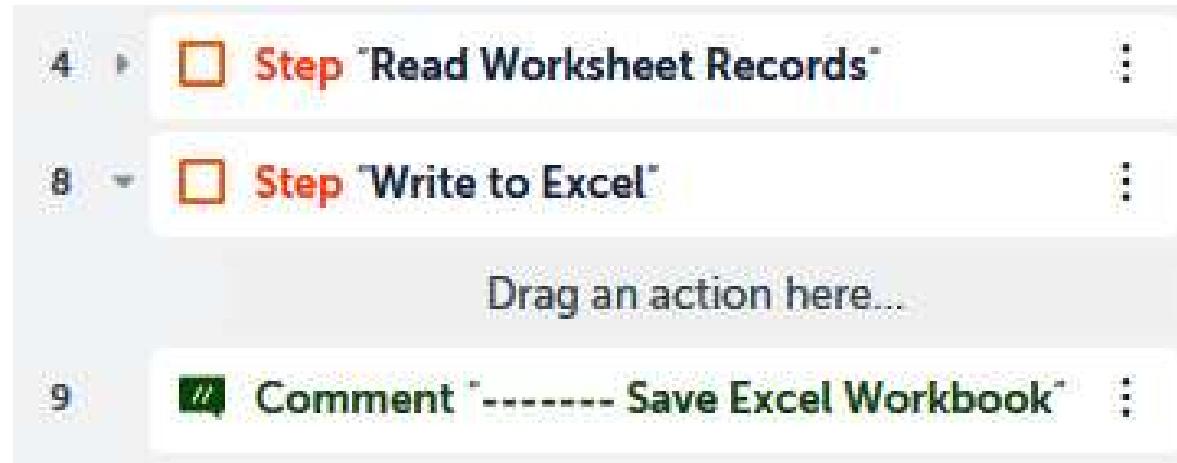
Reading from Excel worksheets

- Click on Save.
- The development interface for the Loop should look like this:



Reading from Excel worksheets

- Set the Title property of this Step on line 8 as Write to Excel.
- Click on Save and collapse the Step on line 4.
- Your development interface should look like this:



Reading from Excel worksheets

Set the following properties for the Number:

Assign action on line 9:

- Select the source string variable/ value:
\$numRecCount\$ + 2
- Select the destination number variable:
numResultRow - Number
- The properties should look like this:

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$numRecCount\$ + 2 (x)

Specify value to assign to number

Select the destination number variable

numResultRow - Number ▼ (x)

Reading from Excel worksheets

Set the following properties for the Number: To string action on line 10:

- Enter a number: \$numRecCount\$
- Enter number of digits after decimal: 0
- Assign the output to variable: strRecCount - String
- The properties should look like this:

Number: To string

Converts a user specified number to a string

Enter a number

\$numRecCount\$ (x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

0 (x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

strRecCount - String ▼ (x)

Reading from Excel worksheets

Number: To string

Converts a user specified number to a string

Enter a number

\$numResultRow\$ (x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

0 (x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

strResultRow - String ▼ (x)

Reading from Excel worksheets

Set the following properties for the Excel basic: Set cell action on line 12:
Session name: xl_data

- Use: Specific cell – A\$strResultRow\$
- Value to set: Total:
- The properties should look like this:

Excel basic: Set cell

Sets a value in a given cell of an excel spreadsheet

Session name

„ xl_data (x)

e.g. Session1 or S1

Use

Active cell

Specific cell

„ A\$strResultRow\$ (x)

e.g., A5 or B10

Value to set

„ Total: (x)

e.g., Original

Reading from Excel worksheets

Set the following properties for the Excel basic: Set cell action on line 13:Session name: xl_data

- Use: Specific cell – B\$strResultRow\$
- Value to set: \$strRecCount\$
- The properties should look like this:

Excel basic: Set cell

Sets a value in a given cell of an excel spreadsheet

Session name

” xl_data (x)

e.g. Session1 or S1

Use

Active cell

Specific cell

” B\$strResultRow\$ (x)

e.g., A5 or B10

Value to set

” \$strRecCount\$ (x)

e.g., Original

Reading from Excel worksheets

- Click on Save.
- The development interface should look like this:

8	<input type="checkbox"/> Step "Write to Excel"	:
9	# Number: Assign "\$numRecCount\$ + 2" to \$numResultRow\$:
10	# Number: To string convert \$numRecCount\$ to a string datatype and assign output to \$strRecCount\$:
11	# Number: To string convert \$numResultRow\$ to a string datatype and assign output to \$strResultRow\$:
12	Excel basic: Set cell : Set value of Specific cell "A\$strResultRow\$" to "Total:"	:
13	Excel basic: Set cell : Set value of Specific cell "B\$strResultRow\$" to \$strRecCount\$:
14	Comment "----- Save Excel Workbook"	:

Reading from Excel worksheets

	A	B	C	D	E	F
1	ID	Author	Title	Genre	Price	Publish Date
2	bk101	Gambardella, Matthew	XML Developer's Guide	Computer - Software	44.95	01/10/2000
3	bk102	Ralls, Kim	Midnight Rain	Fantasy	5.95	16/12/2000
4	bk103	Corets, Eva	Maeve Ascendant	Fantasy	5.95	17/11/2000
5	bk104	Corets, Eva	Oberon's Legacy	Fantasy	5.95	10/03/2001
6	bk105	Corets, Eva	The Sundered Grail	Fantasy	5.95	10/09/2001
7	Total:	5				

... Catalog Sheet2 +

Working with data in Excel

	A	B	C	D	E
1	Segment	Product	Units Sold	Sale Price	Date
2	Midmarket	Paseo	549	£ 15.00	01/09/2013
3	Small Business	Paseo	788	£ 300.00	01/09/2013
4	Government	VTT	1527	£ 350.00	01/09/2013
5	Enterprise	Carretera	330	£ 125.00	01/09/2013

Extract Sheet1 +

Working with data in Excel

- Add a new Comment action as "-----" on line 8 and click on Save.
- Our development interface should look like this:

1	Comment -----	:
2	Comment ----- Open Excel Worksheet	:
3	Comment ----- Inserting a Column	:
4	Comment ----- Setting Cell Formula	:
5	Comment ----- Sorting Data	:
6	Comment ----- Finding Empty Cell	:
7	Comment ----- Save & Close Excel	:
8	Comment -----	:

Working with data in Excel

Excel advanced: Open

Opens an excel spreadsheet. This action works with xlsx, xls, xlsb, xlsm and csv files.

Session name

xl_data

e.g. Session1 or S1

File path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data

SampleData.xlsx

Browse...

Required extensions: ".xlsx", ".xls", ".xlsm", ".xlsb", ".csv"

e.g. C:\Working\Excel1.xlsx

Specific sheet name

Extract

e.g. Sheet1 or SHEET1

Open in

Read-only mode

Read-write mode

Password is required

To open (optional)

Credential Variable Insecure string

Pick...

To edit (optional)

Credential Variable Insecure string

Pick...

Sheet contains a header

Load Add-ins

Inserting a column

Set the following properties for the Excel advanced: Insert table column action on line 5:Session name: xl_data

- Table name: Data
- Column name: Amount
- Column position: 5
- The properties should look like this:

Excel advanced: Insert table column

Inserts a table column in a spreadsheet. This action works with xlsx, xls, xlsb and xlsm files.

Session name	<input type="text" value="xl_data"/> <input type="button" value="({})"/>
e.g. Session1 or S1	
Table name	<input type="text" value="Data"/> <input type="button" value="({})"/>
e.g. Table1	
Column name (optional)	<input type="text" value="Amount"/> <input type="button" value="({})"/>
Column position (at)	<input type="text" value="# 5"/> <input type="button" value="({})"/>

Setting a cell formula

Set the following properties for the Excel advanced: Set cell formula action on line 7:Session name: xl_data

- Set cell formula for: Specific cell – E2
- Enter formula for specific cell: =[@[Units Sold]]*[@[Sale Price]]
- The properties should look like this:

Excel advanced: Set cell formula

Sets the formula of a given cell. This action works with xlsx, xls, xlsb, xlsm and csv files.

Session name

„ xl_data (x)

e.g. Session1 or S1

Set cell formula for

Active cell

Specific cell

„ E2 (x)

e.g. A5 or B10

Enter formula for specific cell

„ =[@[@[Units Sold]]*[@[Sale Price]]] (x)

Sorting data

Set the following properties for the Excel advanced: Sort table action on line 9:Session name: xl_data

- Table name: Data
- Sort for: Column name - Segment
- Sort order: Text - A-to-Z
- The properties should look like this:

Excel advanced: Sort table

Sorts a table within an Excel sheet. This action works with xlsx, xls, xlsb and xlsm files.

Session name

„ xl_data (x)

e.g. Session1 or S1

Table name

„ Data (x)

e.g. Table1

Sort for

Column name

„ Segment (x)

e.g. Column1

Column position

(x)

e.g. 2

Sort order

Number

▼

Text

A-to-Z ▼

Finding empty cells

Set the following properties for the Excel advanced: Find next empty cell action on line 11:

- Session name: xl_data
- Transverse by: column
- Start from: specific cell
- Cell address: E1
- Assign the output to variable: strTotalCell - String
- The properties should look like this:

Excel advanced: Find next empty cell

Finds next empty cell. This action works with xlsx, xls, xlsb and xlsm files.

Session name
"xl_data"

e.g. Session1 or S1

Traverse by
 row
 column

Start from
 active cell
 specific cell

Cell address
"E1"

e.g. A5 or B10

Assign the output to variable
strTotalCell - String

Working with data in Excel

Set the following properties for the Excel advanced: Set cell formula action on line 12:Session name: xl_data

- Set cell formula for: Specific cell – \$strTotalCell\$
- Enter formula for specific cell: =SUM(Data[Amount])
- The properties should look like this:

Excel advanced: Set cell formula

Sets the formula of a given cell. This action works with xlsx, xls, xlsb, xlsm and csv files.

Session name

“ xl_data (x)

e.g. Session1 or S1

Set cell formula for

Active cell

Specific cell

“ \$strTotalCell\$ (x)

e.g. A5 or B10

Enter formula for specific cell

“ =SUM(Data[Amount]) (x)

Saving and closing the workbook

Set the following properties for the Excel advanced: Close action on line 14:

- Session name: xl_data
- Save changes when closing file:
Checked
- The properties should look like this:

Excel advanced: Close

Closes an excel spreadsheet. This action works with xlsx, xls, xlsb, xlsm and csv files.

Session name

e.g. Session1 or S1

Save changes when closing file

Saving and closing the workbook

1	Comment "-----"	⋮
2	Comment "----- Open Excel Worksheet"	⋮
3	Excel advanced: Open "C:\Hands-On-RPA-with-AA-Sample-Data\SampleData.xlsx"	⋮
4	Comment "----- Inserting a Column"	⋮
5	Excel advanced: Insert table column "Amount" to table "Data" into current worksheet	⋮
6	Comment "----- Setting Cell Formula"	⋮
7	Excel advanced: Set cell formula "=@[Units Sold])*@[Sal...]" into Specific cell "E2"	⋮
8	Comment "----- Sorting Data"	⋮
9	Excel advanced: Sort table "Data" for column "Segment" with sort order A-to-Z	⋮
10	Comment "----- Finding Empty Cell"	⋮
11	Excel advanced: Find next empty cell in column from specific cell from "E1" and store value to \$strTotalCell\$	⋮
12	Excel advanced: Set cell formula "=SUM(Data[Amount])" into Specific cell \$strTotalCell\$	⋮
13	Comment "----- Save & Close Excel"	⋮
14	Excel advanced: Close	⋮
15	Comment "-----"	⋮

Saving and closing the workbook

	A	B	C	D	E	F
1	Segment	Product	Units Sold	Sale Price	Amount	Date
82	Small Business	VTT	2151	£ 300.00	£ 645,300.00	01/09/2014
83	Small Business	VTT	986	£ 300.00	£ 295,800.00	01/09/2014
84	Small Business	Paseo	2905	£ 300.00	£ 871,500.00	01/11/2014
85					£ 15,445,109.00	

Extract Sheet1 +

Running macros

- Add a new Comment action as "-----" on line 4 and click on Save.
- Our development interface should look like this:

1	# Comment -----	:
2	# Comment ----- Open Excel Worksheet	:
3	# Comment ----- Run Macro	:
4	# Comment -----	:

Running macros

Excel advanced: Open

Opens an excel spreadsheet. This action works with xlsx, xls, xlsm and csv files.

Session name

xl_data

e.g. Session1 or S1

File path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data

SampleData.xlsx

Browse...

Required extensions: ".xlsx", ".xls", ".xlsm", ".xlst", ".csv"

e.g. C:\Working\Excel1.xlsx

Specific sheet name

Extract

e.g. Sheet1 or SHEET1

Open in

Read-only mode

Read-write mode

Password is required

To open (optional)

Credential Variable Insecure string

Pick...

To edit (optional)

Credential Variable Insecure string

Pick...

Sheet contains a header

Load Add-ins

Running macros

Set the following properties for the Excel advanced: Run macro action on line 5:Session name: xl_data

- Macro name: procFilterSegment
- Macro arguments: Enterprise
- The properties should look like this:

Excel advanced: Run macro

Runs a macro in an excel worksheet. This action works with xlsx, xls, xlsb and xlsm files.

Session name

„ xl_data

e.g. Session1 or S1

Macro name

„ procFilterSegment

e.g. ConvertData

Macro arguments (optional)

„ Enterprise

e.g. Arg1,Arg2,Arg3

Running macros

- Click on Save.
- The development interface should look like this:



The screenshot shows a development interface with a list of steps. The steps are numbered 1 through 6. Step 1 is a comment. Step 2 is another comment. Step 3 is an Excel advanced action to open a file named 'SampleData.xlsxm'. Step 4 is a comment. Step 5 is an Excel advanced action to run a macro named 'procFilterSegment' with an argument 'Enterprise'. Step 6 is a final comment. The interface has a 'Start' button at the top left and an 'End' button at the bottom left. There are three vertical dots on the right side of each step row.

Step	Action	Details	⋮
1	Comment	"-----"	
2	Comment	"----- Open Excel Worksheet"	
3	Excel advanced	Open "C:\Hands-On-RPA-with-AA-Sample-Data\SampleData.xlsxm"	
4	Comment	"----- Run Macro"	
5	Excel advanced	Run macro : "procFilterSegment" with arguments "Enterprise"	
6	Comment	"-----"	

Summary

- A wide selection of actions from both the Excel basic and advanced package has been covered in this lesson.
- The walk-throughs will have given you the opportunity to build three individual bots all performing different tasks.
- You should be confident with opening, closing, and saving Excel documents, as well as reading and writing to them.
- The skills gained should also include working with data within Excel documents, such as sorting, inserting columns, adding formulas, as well as running macros.

Automation Using Word



Automation Using Word

 Comment	 List	 Step
 Datetime	 Loop	 String
 Excel advanced	 MS Word	 System
 File	 Number	

Automation Using Word

In this lesson, we will cover the following topics:

- Understanding the manual process
- Creating new Word documents
- Inserting text in Word documents
- Inserting paragraphs in Word documents
- Replacing text in Word documents

Understanding the manual process

- From this role specification, the process for our bot is shown in the following figure:
 1. Understanding the manual process.
 2. Reading source data.
 1. Opening Excel workbook.
 2. Reading and assigning column names to a list variable.
 3. Looping through each Excel record.
 1. Creating the output letter.
 1. Creating a new Word document from the template.
 2. Updating the output letter.
 1. Inserting the system date in the Word document.
 2. Adding additional paragraphs in the Word documents.
 3. Looping through each column from the list.
 1. Replacing placeholder in Word with the value from Excel.
 3. Closing Excel data source.

Understanding the manual process

- Add a new Comment action as "-----" on line 5 and click on Save.
- Your initial development interface should look like this:

1	Comment -----	:
2	Comment ----- Read source data	:
3	Comment ----- Create output letter	:
4	Comment ----- Update output letter	:
5	Comment -----	:

Understanding the manual process

- Add a step just below line 11, ensuring it is aligned with the step on line 5, set the Title property to Close Excel Workbook, and click on Save.
- The development interface should look like this:

1	Comment "-----"	:
2	Comment "----- Read source data"	:
3	Step "Open Excel Workbook"	:
4	Step "Get Column Names"	:
5	Step "Read each loan record"	:
6	Comment "----- Create output letter"	:
7	Step "Create new word document"	:
8	Comment "----- Update output letter"	:
9	Step "Insert Date"	:
10	Step "Add Paragraphs"	:
11	Step "Replace placeholders with values"	:
12	Step "Close Excel Workbook"	:
13	Comment "-----"	:

Reading the source data

- The source data is stored in the lesson12_LoanData.xlsx file, which has a worksheet called Approved.
- The data looks as shown in the following screenshot:

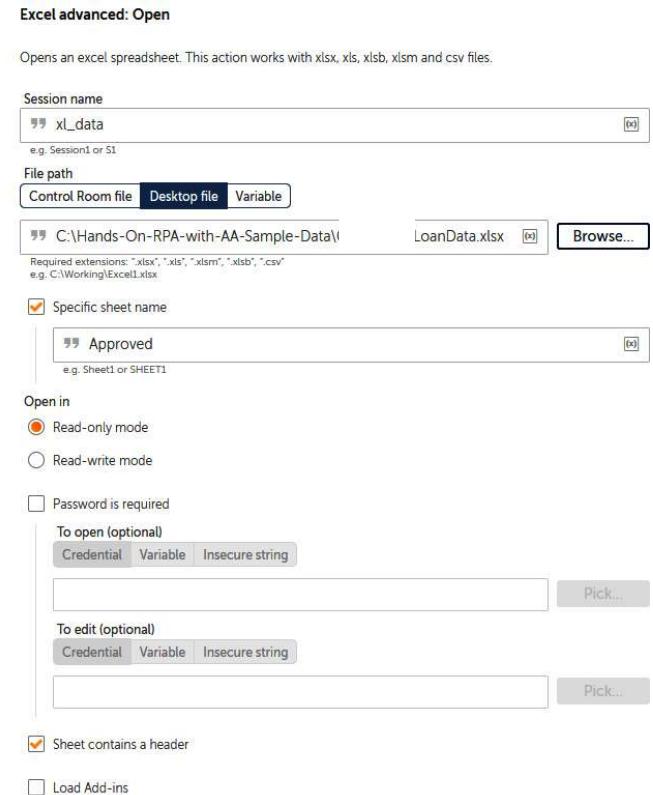
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	[Ref]	[Title]	[Forename]	[Surname]	[Address]	[City]	[County]	[Postcode]	[Amount]	[Term]	[Interest]	[Payable]	[Monthly]
2	Q298	Mr	John	Pince	21 Hobart St	Parkham	Devon	EX39 5DJ	£5,000.00	24	3.40%	£5,176.08	£ 215.67
3	Q299	Mrs	Vannessa	Casper	45 Bradfield St	Newquay	Cornwall	TR7 1LS	£3,500.00	12	8.50%	£3,657.12	£ 304.76
4	Q300	Miss	Sarah	Mchughes	73 Parkfield Rd	Parwich	Derbyshire	DE6 1QN	£7,500.00	48	3.00%	£7,961.76	£ 165.87
5	Q301	Dr	David	Hawkin	30 Aughton St	Norton Canes	Staffordshire	WS11 9RH	£8,000.00	60	3.00%	£8,616.00	£ 143.60
6	Q302	Mr	Roger	Day	7 Richmond St	Hilton	Aberdeenshire	AB24 2RR	£4,000.00	36	8.50%	£4,524.48	£ 125.68

Approved

Reading the source data

Set the following properties for the Excel advanced: Open action on line 4:Session name: xl_data

- File path: Desktop file – C:\Hands-On-RPA-with-AA-Sample-Data\lesson12_LoanData.xlsx
- Specific sheet name: Checked – Approved
- Open in: Read-only mode
- Sheet contains a header: Checked
- The properties should look like this:



Getting column names

Set the following properties for the Excel advanced: Read row action on line 6:Session name: xl_data

- Cell option: From specific cell
- Cell address: A1
- Read full row: Checked
- Read option: Read cell value
- Assign the output to variable: lstColumns – List of Strings
- The properties should look like this:

Excel advanced: Read row

Reads values from a row. This action works with xlsx, xls, xlsb and xlsm files.

Session name

„ xl_data

e.g. Session1 or S1



Cell option

From active cell

From specific cell

Cell address

„ A1



e.g. A5 or B10

Read full row

Read option

Read visible text in cell

e.g. 50% will be read as 50%

Read cell value

e.g. 50% will be read as 50

Assign the output to variable

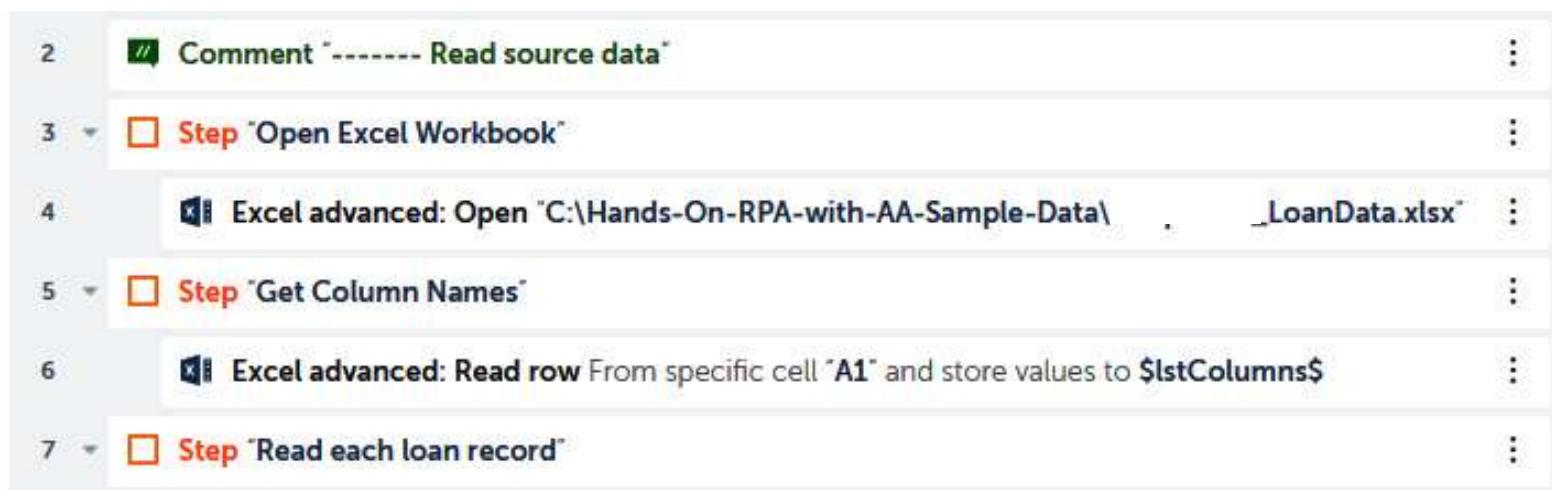
lstColumns - List of Strings



(x)

Getting column names

- Click on Save.
- The development interface for this section should look like this:



Reading the loan records

Set the following properties for the Loop action on line 8:Iterator: For each row in worksheet (Excel advanced)

- Session name: xl_data
- Loop through: All rows
- Read option: Read cell value
- Assign the current value to this variable: recLoan - Record
- The properties should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in worksheet ▾

Iterator for each row in Excel. Output(s) will be assigned to a record variable

Session name

xl_data ▾

Loop through

All rows ▾

Read option

Read visible text in cell

e.g. 50% will be read as 50%

Read cell value

e.g. 50% will be read as 50

Assign the current value to this variable

recLoan - Record ▾

Reading the loan records

- Click on Save.
- The development interface for this section should look like this:

1	#[Comment "-----"	⋮
2	#[Comment "----- Read source data"	⋮
3	▶ ⚡ Step "Open Excel Workbook"	⋮
5	▶ ⚡ Step "Get Column Names"	⋮
7	▶ ⚡ Step "Read each loan record"	⋮
8	▶ ⚡ Loop : For each row in worksheet and assign to \$recLoan\$ 9 #[Comment "----- Create output letter"	⋮
10	10 ▶ ⚡ Step "Create new word document"	⋮
11	11 #[Comment "----- Update output letter"	⋮
12	12 ▶ ⚡ Step "Insert Date"	⋮
13	13 ▶ ⚡ Step "Add Paragraphs"	⋮
14	14 ▶ ⚡ Step "Replace placeholders with values"	⋮
15	15 ▶ ⚡ Step "Close Excel Workbook"	⋮
16	16 #[Comment "-----"	⋮

Creating an output letter

- We have a Word document named `lesson12_Template.docx` to use as a template.
- The document looks as shown in the following figure:



[Title] [Forename] [Surname]
[Address]
[City]
[County]
[Postcode]

Our Ref: [Ref]

Dear [Forename],

Congratulations! Based on the information provided by you, we are pleased to inform you that you have been pre-approved for your recent loan application.

Loan Details

Amount: [Amount]

Term: [Term] months

Interest Rate (APR): [Interest]

Total Payable: [Payable]

Monthly Payment: [Monthly]

Creating an output letter

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

C:\Hands-On-RPA-with-AA-Sample-Data\

\\$recLoan[0].docx



Select the destination string variable

strLoanLetter - String



Creating an output letter

File: Copy

Copies a file

Source file

C:\Hands-On-RPA-with-AA-Sample-Data\template.docx

e.g. C:\MyDoc*.doc

Destination file/folder

\$strLoanLetter\$

e.g. C:\Backup\ , C:\Backup*.doc

Overwrite existing files

Creating an output letter

Click on Save.

- A new Word document should now be created and named as the reference number for each specific record.
- The development interface for this section should look like this:

The screenshot shows a sequence of steps in a RPA development interface:

- Step 9: Comment "----- Create output letter"
- Step 10: Step "Create new word document"
- Step 11: String: Assign "C:\Hands-On-RPA-with-AA..." to \$strLoanLetter\$
- Step 12: File: Copy "C:\Hands-On-RPA-with-AA-Sample-Data\T..." to \$strLoanLetter\$

Creating an output letter

Set the following properties for the Datetime: To string action on line 15:Source date and time variable: System:Date - Datetime

- Select date time format: Custom format – d MMM YYYY
- Assign the output to a variable: strDate - String
- The properties should look like this:

Datetime: To string

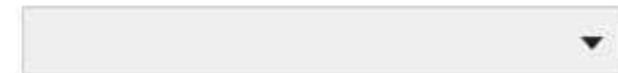
Converts a datetime value to a string value and assigns it to a string variable

Source date and time variable

System:Date - Datetime

Select date time format

Formats



Custom format

“ d MMM YYYY

Assign the output to a variable

strDate - String

Creating an output letter

MS Word: Insert Text

Insert Text at Bookmark Position in MS Word

Select the Word document

Control Room file Desktop file Variable

“ \$strLoanLetter\$

(x)

Browse...

Enter Bookmark Name

“ bmDate

(x)

Enter Text to be Inserted at Bookmark position

“ \$strDate\$

(x)

Creating an output letter

- Click on Save.
- The development interface for this section should look like this:

The screenshot shows a development interface with a list of numbered steps:

- 13: Comment "----- Update output letter"
- 14: Step "Insert Date"
- 15: Datetime: To string Convert \$System:Date\$ and assign result to \$strDate\$
- 16: MS Word: Insert Text
- 17: Step "Add Paragraphs"
- 18: Step "Replace placeholders with values"

Adding paragraphs

Please check that all the details are correct and if you wish to proceed, please contact our pre-approval customer service office on 0800 000 0000. Lines are open Monday to Friday 9:00 – 17:00.

Yours sincerely,

Jack Money

Loan Approvals Manager

Adding paragraphs

MS Word: Add Paragraph

Add Paragraph in Existing MS Word Document

Select the Word document

Control Room file Desktop file Variable

” \$strLoanLetter\$

Please write paragraph or select variable

” Please check that all the details are correct

Adding paragraphs

- Click on Save.
- Your development interface for this section should look like this:

17	<input type="checkbox"/> Step "Add Paragraphs"	:
18	■ MS Word: Add Paragraph	:
19	■ MS Word: Add Paragraph	:
20	■ MS Word: Add Paragraph	:
21	■ MS Word: Add Paragraph	:
22	■ MS Word: Add Paragraph	:
23	■ MS Word: Add Paragraph	:
24	<input type="checkbox"/> Step "Replace placeholders with values"	:

Replacing text

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

-1

(x)

Specify value to assign to number

Select the destination number variable

numColumnIndex - Number



(x)

Replacing text

Set the following properties for the Loop action on line 26:Iterator: For each value in record

- Record variable: recLoan - Record
- Assign the current value to this variable: strValue - String
- The properties should look like this:

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each value in record

Iterator for each value in record

Record variable

recLoan - Record

(x)₊

Assign the current value to this variable

strValue - String

(x)₊

Replacing text

Set the following properties for the Number: Increment action on line 26:
Enter number: \$numColumnIndex\$

- Enter increment value: 1
- Assign the output to variable: numColumnIndex - Number
- The properties should look like this:

Number: Increment

Increments a number by specified value

Enter number

\$numColumnIndex\$ (x)

Enter increment value

1 (x)

Increments number by value (e.g. 1)

Assign the output to variable

numColumnIndex - Number ▼ (x)

Replacing text

List: Get item

Gets an item from the List from a given index position

List variable

lstColumns - List



(x)
+

Index number

\$numColumnIndex\$

(x)

Assign the output to variable

strPlaceHolder - String



(x)
+

MS Word: Replace Text

Replace Existing text in MS Word Document

Select the Word document

Control Room file Desktop file Variable

\$strLoanLetter\$

Enter Text to be replaced

\$strPlaceHolder\$

Enter new Text

\$strValue\$

Replacing text

Replacing text

Excel advanced: Close

Closes an excel spreadsheet. This action works with xlsx, xls, xlsm and csv files.

Session name

99 xl_data

(x)

e.g. Session1 or S1

Save changes when closing file

Replacing text

24	☐ Step "Replace placeholders with values"	:
25	# Number: Assign -1 to \$numColumnIndex\$:
26	⌚ Loop	:
27	# Number: Increment \$numColumnIndex\$ by 1 and assign result to a \$numColumnIndex\$ variable	:
28	>List: Get item from position \$numColumnIndex\$ in \$lstColumns\$:
29	FLAG MS Word: Replace Text	:
30	☐ Step "Close Excel Workbook"	:
31	Excel advanced: Close	:

Summary

- In this lesson, we learned how Automation Anywhere can be used to automate tasks using Word. We learned how to add paragraphs, insert text using bookmarks, and replace text.
- The bot that we built in this lesson has also shown us how Excel and Word can be used together to fully automate a business role end to end.
- All this automation provides a more effective way to perform the same task without the risk of errors and also increases efficiency as it reduces the manual effort involved.

Working with Emails



Working with Emails

-  Comment 
-  Email 
-  Folder 
-  If 
-  Loop 
-  Message box 
-  Step 

Working with Emails

In this lesson, we will cover the following:

- Connecting to mailboxes
- Reading emails and attachments
- Sending emails and attachments

Connecting to mailboxes

In the first walk-through, you will learn how to create a new session by connecting to the following:

- An Outlook mailbox
- An POP3/IMAP mailbox
- An EWS mailbox

Connecting to mailboxes

- Add a new Comment action on line 6 as "-----" and click on Save.
- Your initial development interface should look like this:



Connecting to Outlook

Email: Connect

Connects to an email server

Session name

99 EmailOutlook

e.g. Session1 or S1

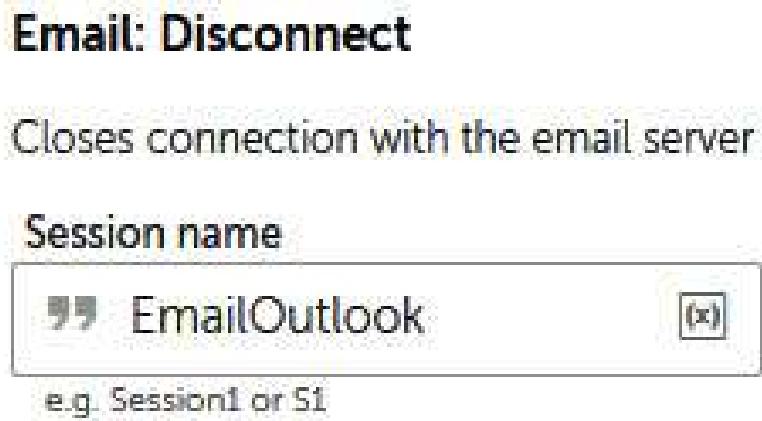
Connect to



Outlook

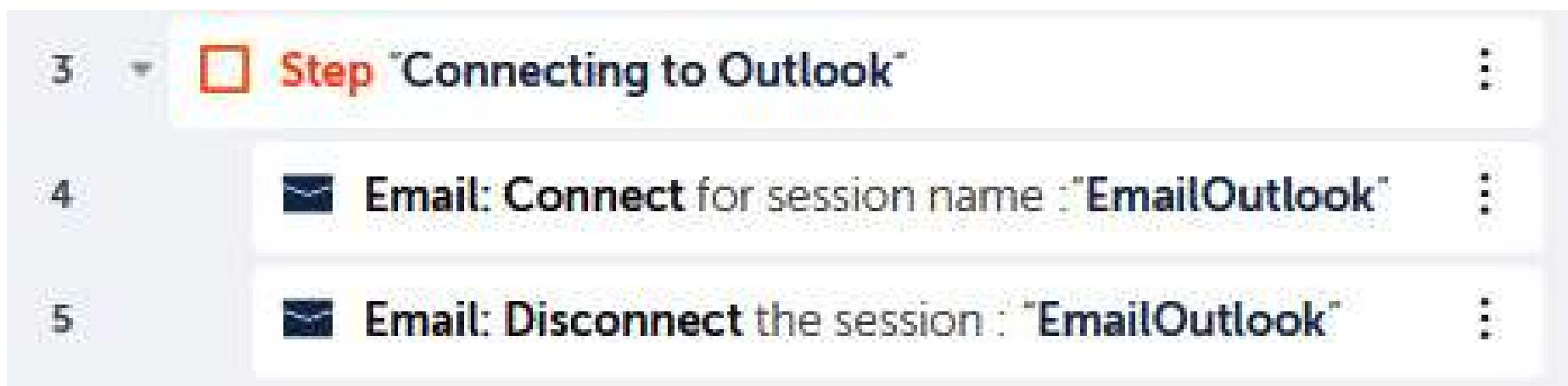
Connecting to Outlook

- Set the following property for the Email: Disconnect action on line 5:
- Session name: EmailOutlook
- The property should look like this:



Connecting to Outlook

- Click on Save and your development interface for this section should look like this:



Connecting to a POP3/IMAP mailbox

- To give an example of this information, if you were connecting to a Gmail account, the details would be as shown in the following figure:

POP3 Settings	
Incoming Mail Server:	pop.gmail.com
Requires SSL:	Yes
Port:	995
Email address	*****@gmail.com
Password	*****

IMAP Settings	
Incoming Mail Server:	imap.gmail.com
Requires SSL:	Yes
Port:	993
Email address	*****@gmail.com
Password	*****

Connecting to a POP3/IMAP mailbox

Email: Connect
Connects to an email server

Session name

e.g. Session1 or S1

Connect to
 Outlook
 Email server

Host

eg: outlook.office365.com, etc.

Port

eg: 993, 995 etc.

Username

Password

Use secure connection(SSL/TLS)

Protocol
 IMAP
 POP3

Connecting to a POP3/IMAP mailbox

- Set the following property for the Email: Disconnect action on line 5:Session name: EmailSession
- The property should look like this:

Email: Disconnect

Closes connection with the email server

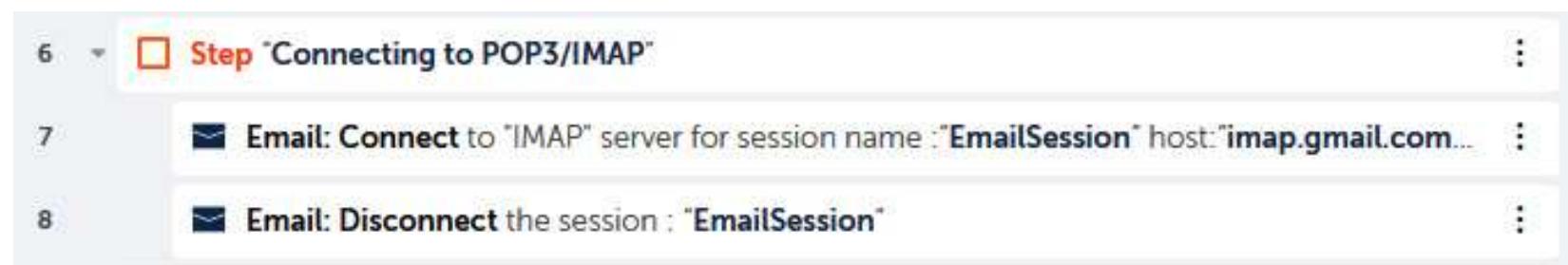
Session name

” EmailSession

e.g. Session1 or S1

Connecting to a POP3/IMAP mailbox

- Click on Save.
- Your development interface for this section should look like this:



Connecting to an EWS mailbox

- To give an example of this information, if you were connecting to an Outlook Exchange account, the details would be as shown in the following figure:

Exchange Server Settings	
Email:	*****@outlook.com
Password:	*****
Server:	outlook.com
Version:	Exchange 2010

Connecting to an EWS mailbox

Email: Connect

Connects to an email server

Session name

„ EmailSessionEWS (x)

e.g. Session1 or S1

Connect to

EWS server

Username

Credential Variable Insecure string Insecure string

„ *****@outlook.com (x)

Password

Credential Variable Insecure string Insecure string

„ ***** (x)

Enter Domain name (optional)

„ outlook.com (x)

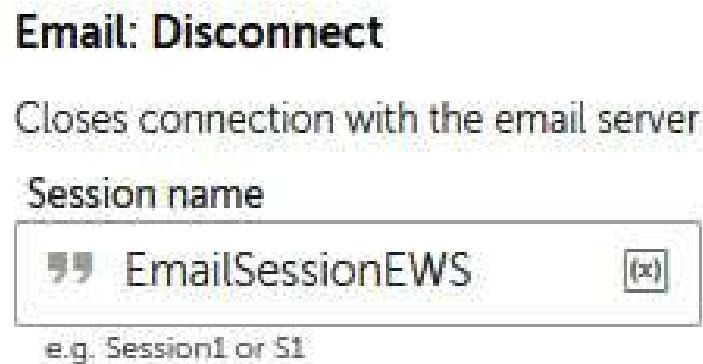
e.g. smtp.office365.com

Exchange Version

Exchange2010 ▼

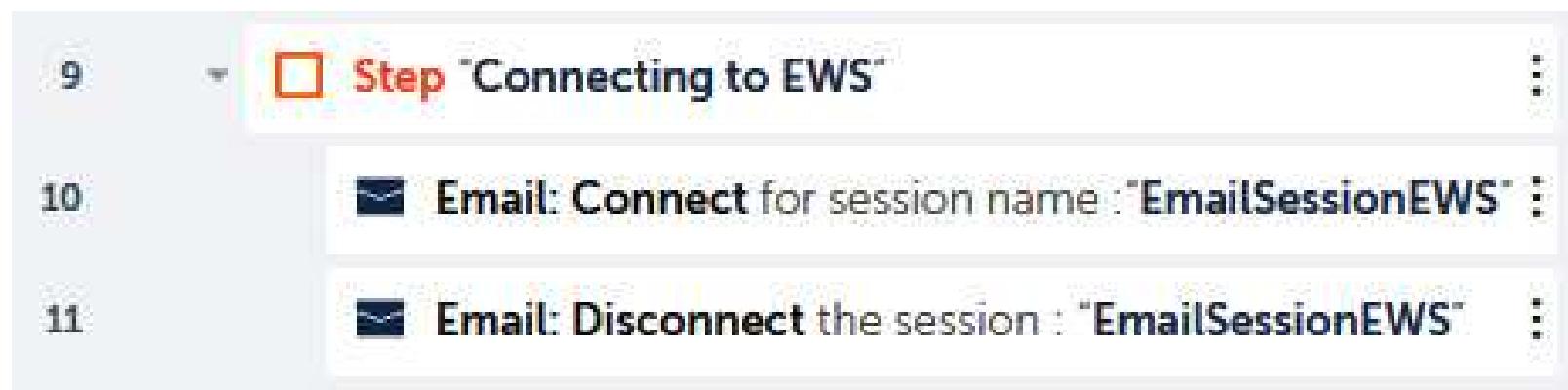
Connecting to an EWS mailbox

- Set the following property for the Email: Disconnect action on line 11:
- Session name: EmailSessionEWS
- The property should look like this:



Connecting to an EWS mailbox

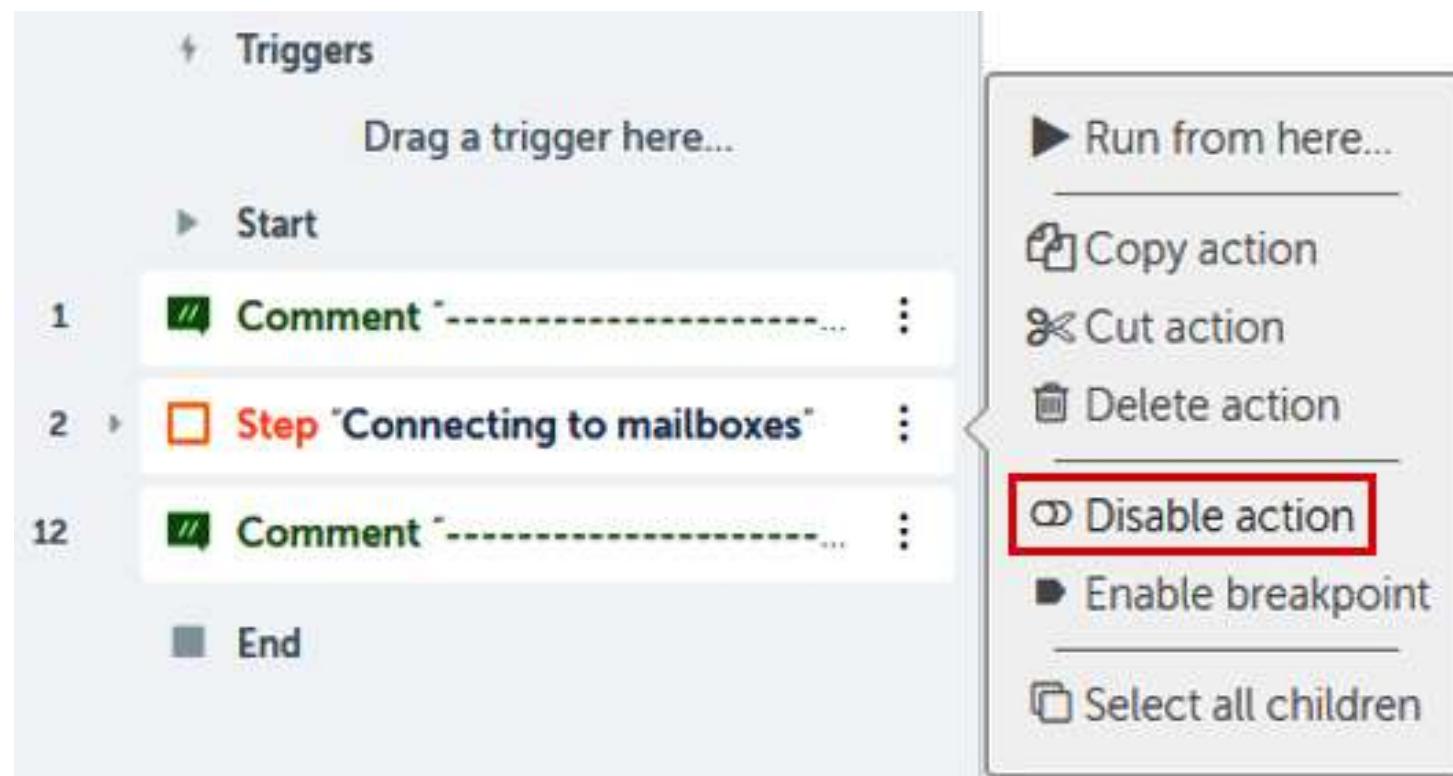
- Click on Save and your development interface for this section should look like this:



Connecting to an EWS mailbox

1	Comment -----	⋮
2	Step "Connecting to mailboxes"	⋮
3	Step "Connecting to Outlook"	⋮
4	Email: Connect for session name : "EmailOutlook"	⋮
5	Email: Disconnect the session : "EmailOutlook"	⋮
6	Step "Connecting to POP3/IMAP"	⋮
7	Email: Connect to "IMAP" server for session name : "EmailSession" host:"ima..."	⋮
8	Email: Disconnect the session : "EmailSession"	⋮
9	Step "Connecting to EWS"	⋮
10	Email: Connect for session name : "EmailSessionEWS"	⋮
11	Email: Disconnect the session : "EmailSessionEWS"	⋮
12	Comment -----	⋮

Connecting to an EWS mailbox

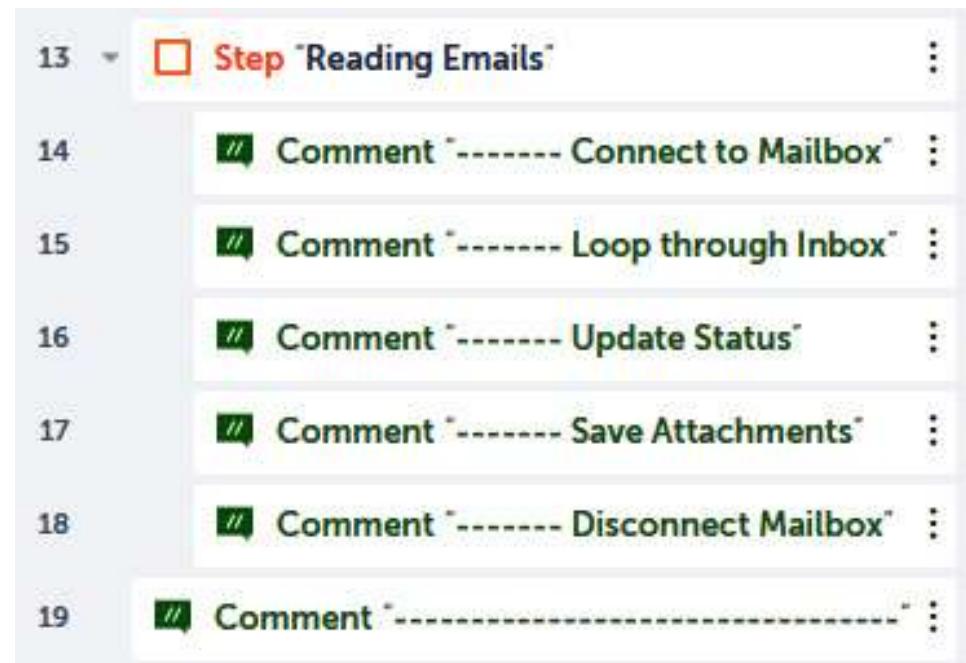


Reading emails and attachments

Key	Value
emailSubject	Email Subject
emailFrom	Senders Email
emailTo	Who the Email is addressed to
emailCc	Any Email CC's
emailBcc	Any Email Bcc's
emailMessage	Email Message
emailReceivedTime	Time Email received
emailReceivedDate	Date Email received

Reading emails and attachments

- Add a new Comment action as "-----" on line 19 and click on Save.
- Your initial development interface should look like this:



Reading emails and attachments

Set the following properties for the Email: Connect action on line 15:Session name: EmailSession

- Connect to: Outlook
- The properties should look like this:

Email: Connect

Connects to an email server

Session name

 (x)

e.g. Session1 or S1

Connect to



Outlook

Reading emails and attachments

- Set the following property for the Email: Disconnect action on line 20:
- Session name: EmailSession
- The property should look like this:



Reading emails and attachments

- Click on Save and your development interface for this section should look like the following:

13	<input type="checkbox"/> Step "Reading Emails"	:
14	Comment "----- Connect to Mailbox"	:
15	Email: Connect for session name : "EmailSession"	:
16	Comment "----- Loop through Inbox"	:
17	Comment "----- Update Status"	:
18	Comment "----- Save Attachments"	:
19	Comment "----- Disconnect Mailbox"	:
20	Email: Disconnect the session : "EmailSession"	:
21	Comment "-----"	:

Looping through emails from a folder

Create variable

Cancel

Create

Name

dctEmail

Max characters = 50

Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)

Type

Dictionary

Subtype

String

Default value (optional)

This dictionary is empty



Looping through emails from a folder

Set the following properties for the Loop action on line 17: Loop Type: Iterator

- Iterator: For each mail in mail box
- Session name: EmailSession
- Type of email to get: ALL
- Message format: PLAINTEXT
- Assign the current value to variable: dctEmail
 - Dictionary of Strings
- The properties should look like this:

Loop
Repeats the actions in a loop until a break

Loop Type
 Iterator

Iterator
For each mail in mail box

Iterator for each mail in mail box

Session name
EmailSession

Type of email to get
 ALL
 READ
 UNREAD

Message format
 HTML
 PLAINTEXT

Assign the current value to variable (optional)
dctEmail - Dictionary of Strings

Applying filters and specifying folders

Update the loop properties for the Loop action on line 17:

- Type of email to get: UNREAD
- From a specific folder (optional): Inbox
- The properties should look like this:

Type of email to get

ALL

READ

UNREAD

For POP3 protocol all message will be fetched

From a specific folder (optional)

 Inbox



e.g. `Inbox/folder1;Inbox/folder2 or Inbox/test*`. For POP3 fetching from `Inbox only`

Specifying the email status

- You have just updated the email status filter, but you can see, as per the following screenshot, that you have an option to select either ALL, READ, or UNREAD emails only:

Type of email to get

ALL

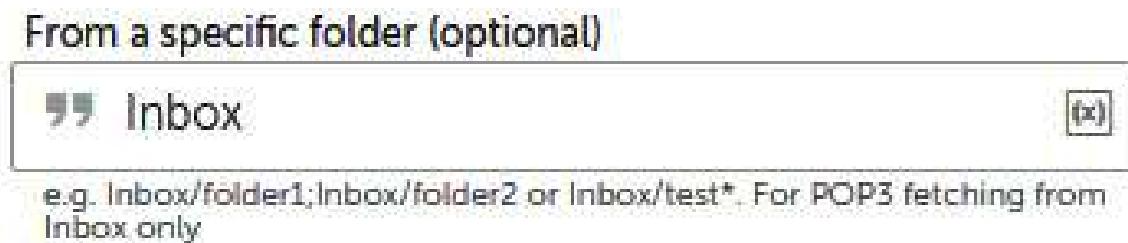
READ

UNREAD

For POP3 protocol all message will be fetched

Specifying the mailbox folders

- You can set a specific folder here.
- Wildcard characters can also be used as part of the folder name.
- Nested subfolders can be specified using the Inbox/SubFolder1 format.
- If needed, variables can also be used to enter this value, The following screenshot shows how to set Inbox as the specified folder:



Specifying the email subject line text

To set a particular text in the subject line, you can do so here. If needed, **String** type variables can also be used to enter this value:

When subject contains (optional)

„	(x)
---	-----

e.g. subject1;subject2

Email loop – subject filter

Specifying the email sender

Here, you can look at emails from a specific sender. Enter the sender's email address (you can also enter multiple email addresses separated by a semicolon). If needed, **String** type variables can also be used to enter this value:

From specific senders (optional)

„	(x)
---	-----

e.g. john@abc.com;Mary@xyz.com

Specifying the email sender

Here, you can look at emails from a specific sender. Enter the sender's email address (you can also enter multiple email addresses separated by a semicolon). If needed, **String** type variables can also be used to enter this value:

From specific senders (optional)

 (x)
e.g. john@abc.com;Mary@xyz.com

- Email loop – sender filter

Specifying the date and time of the received email

You can also set a date/time filter. Again, variables can be used here, but they need to be of the **Datetime** type:

When received date is on or after (optional)

 (x)
▼

When received date is before (optional)

 (x)
▼

Specifying the mailbox folders

- Click on Save and your development window should be looking like this:

13	□ Step "Reading Emails"	⋮
14	#[Comment "----- Connect to Mailbox"	⋮
15	✉ Email: Connect for session name : "EmailSession"	⋮
16	#[Comment "----- Loop through Inbox"	⋮
17	⌚ Loop through UNREAD emails for session: "EmailSession"	⋮
18	#[Comment "----- Update Status"	⋮
19	#[Comment "----- Save Attachments"	⋮
20	#[Comment "----- Disconnect Mailbox"	⋮
21	✉ Email: Disconnect the session : "EmailSession"	⋮
22	#[Comment "-----"	⋮

Updating the email status

Set the following properties for the Email: Change status action on line 19:

- Session name: EmailSession
- Change status to: Read
- The properties should look like this:

Email: Change status

Changes email status to Read/Unread. Use this action inside a loop

Session name

EmailSession



Change status to

Read

Unread

Saving attachments

If

Runs a sequence of actions if a condition is true

Condition

Folder does not exist

Checks the folder does not exist condition.

Folder path

“ C:\Hands-On-RPA-with-AA-Sample-Data”

Emails

How long you would like to wait for this condition to be true?(Seconds)

0

Add condition

Saving attachments

Folder: Create

Creates a folder

Folder

C:\Hands-On-RPA-with-AA-Sample-Data\

e.g. C:\MyDoc\MyNewFolder

Overwrite an existing folder

Saving attachments

Email: Save attachments

Saves all attachments of a single email. Use this action inside a loop.

Save attachments to folder

„ C:\Hands-On-RPA-with-AA-Sample-Data\	_Emails <input type="button" value="x"/>
--	--

D:/Emails



Overwrite file(s)

Saving attachments

Message box

Displays a message box

Enter the message box window title

"Reading Emails"

Enter the message to display

"subject: \${dctEmail{emailSubject}}"

From: \${dctEmail{emailFrom}}

Message: \${dctEmail{emailMessage}}

Scrollbar after lines

30

Close message box after

Seconds

5

Saving attachments

```
12  " Comment "-----"  
13  □ Step "Reading Emails"  
14  " Comment "----- Connect to Mailbox"  
15  ⏱ Email: Connect for session name : "EmailSession"  
16  " Comment "----- Loop through Inbox"  
17  ▷ Loop through UNREAD emails for session: "EmailSession"  
18  " Comment "----- Update Status"  
19  ⏱ Email: Change status to "Read"  
20  " Comment "----- Save Attachments"  
21  ▷ If folder does not exist at "C:\Hands-On-RPA-with-AA-Sample-Data\  
22    □ Folder: Create "C:\Hands-On-RPA-with-AA-Sample-Data\Chapter13_Emails"  
23    ⏱ Email: Save attachments from an email in "C:\Hands-On-RPA-with-AA-Sample-D...  
24    ⏱ Message box "subject: |$dctEmail(emailSubject)| From: |$dctEmail(emailFrom)...  
25  " Comment "----- Disconnect Mailbox"  
26  ⏱ Email: Disconnect the session : "EmailSession"  
27  " Comment "-----"
```

Sending emails and attachments

- To give an example of this information, if you were connecting to a Gmail account, the outgoing server IMAP details would be as shown in the following figure:

Gmail IMAP Settings	
Outgoing Mail Server:	smtp.gmail.com
Requires SSL:	Yes
Requires Authentication:	Yes
Port for SSL:	465
Port for TLS/STARTTLS:	587
Email address:	*****@gmail.com
Password:	*****

Sending emails and attachments

- Add a new Comment action as "-----" on line 32, and click on Save.
- Your development interface should look as in the following screenshot:



The screenshot shows a development interface with a list of code steps. Step 28 is highlighted in orange and labeled 'Step "Sending Emails"'. Steps 29 through 32 are green and labeled 'Comment "-----"'. Step 32 is partially visible at the bottom.

28	Step "Sending Emails"
29	Comment "----- Sending an Email"
30	Comment "----- Forwarding a Email"
31	Comment "----- Replying to a email"
32	Comment "-----"

Sending an email

There are a number of properties to set for the Email: Send action. Starting with the recipient and subject details, set the following properties for the Email:

Send action on line 30:To
address: *****@gmail.com (the email
address you are sending to)

- Subject: RPA – Sending Emails
- These property settings should look like this:

Email: Send

Sends an email

To address

“*****@gmail.com

Use comma for multiple email ids

Cc (optional)

“

Use comma for multiple email ids

Bcc (optional)

“

Use comma for multiple email ids

Subject

“RPA - Sending Emails

Sending an email

- Send email as: Plain text
- Message: This is message sent from your RPA Bot
- These property settings should look like this:

Email: Send

Send email as

- Plain text
- Html

Message

” This is message sent from your RPA Bot (x)

Sending an email

- Send email via: Email server
- From address: *****@gmail.com (email address you are sending from)
- Email server host: smtp.gmail.com
- Email server port: 587
- Use secure connection (SSL/TLS): True
- My server requires authentication: True
- Username (optional): Insecure string – *****@gmail.com (enter your email address)
- Password (optional): Insecure string – ***** (enter your email password)
- These properties should look like this:

Email: Send

Send email via

Email server

From address

*****@gmail.com

Email server host

smtp.gmail.com

eq: smtp-mail.outlook.com,smtp.gmail.com ,etc.

Email server port

587

eq: 587

Use secure connection (SSL/TLS)

True

My server requires authentication

True

Username (optional)

Insecure string

*****@gmail.com

Password (optional)

Insecure string

Attaching a document to an email

- To add an attachment, update the following property for the Email:
Send action on line 30:Attachment (optional): Desktop file –
C:\Hands-On-RPA-with-AA-Sample-Data\lesson05_InputData.csv
- The property should look like this:



Forwarding an email

Set the following properties for the Email:

Forward action on line 32: To

address: *****@gmail.com (the email address you are forwarding the message to)

- Send email as: Plain text
- Message (optional): This email is forwarded by your RPA Bot
- Send email via: Outlook
- The properties should look like this:

Email: Forward

Forwards an email with the same subject. Use this action inside a loop.

To address

*****@gmail.com [x]

Use comma for multiple email ids

Cc (optional)

[x]

Use comma for multiple email ids

Bcc (optional)

[x]

Use comma for multiple email ids

Attachment (optional)

Control Room file Desktop file Variable

Browse...

Validate if attachment is missing

Send email as

Plain text

Html

Message (optional)

***** This email is forwarded by your RPA Bot [x]

The email body will automatically be appended to the message.

Include Go Green message at the end of the email

Send email via

Outlook ▼

Replying to an email

- Set the following properties for the Email: Reply action on line 34:Send email as: Plain text
- Message (optional): This email is a reply by your RPA Bot
- Send email via: Outlook
- The properties should look like this:

Email: Reply

Rplies to an email sender with the same subject. Use this action inside a loop.

Cc (optional)

 [x]

Use comma for multiple email ids

Bcc (optional)

 [x]

Use comma for multiple email ids

Attachment (optional)

Control Room file Desktop file Variable

Browse...

Validate if attachment is missing

Send email as

Plain text

Html

Message (optional)

 [x]

The email body will automatically be appended to the message.

Include Go Green message at the end of the email

Send email via

 ▼

Replying to an email

- Click on Save.
- The development window for the Reading Emails section should look like this:

29	Comment "----- Sending an Email"	:
30	Email: Send an email to "*****@gmail.com" with subject : "RPA - Sending Emails"	:
31	Comment "----- Forwarding a Email"	:
32	Email: Forward an email in current session with Plain text	:
33	Comment "----- Replying to a email"	:
34	Email: Reply to an email in current session with Plain text	:
35	Comment "-----"	:

Summary

- This lesson has been about everything related to email.
- The walk-throughs have demonstrated how to connect to the different types of mail servers, as well as using Outlook.
- You learned how to read from any email folder using specified criteria, such as a certain sender or a particular value in the subject line.
- You further built your knowledge of learning how to send emails, including replying and forwarding emails.
- That's not all; we also included how to save and add attachments to our emails.

Working with PDF Files



Working with PDF Files

- In this lesson, we will be using the following packages:



Working with PDF Files

In this lesson, we will cover the following topics:

- Extracting text and images
- Splitting and merging documents
- Encrypting and decrypting documents
- Using the PDF dictionary

Extracting text from a PDF file

- Add a new Comment action as "-----" on line 3, and click on Save.
- Your initial development interface should look as in the following screenshot:



Extracting text from a PDF file

PDF: Extract text

Extracts text from a PDF file and saves it into a text file.

PDF path

Control Room file Desktop file Variable

„ C:\Hands-On-RPA-with-AA-Sample-Data\

_Letter.pdf

Required extension: ".pdf"

User password (optional)

Credential Variable Insecure string

Owner password (optional)

Credential Variable Insecure string

Extracting text from a PDF file

Continue setting the properties. Next, specify the page range and format of the text required. To do this, set the following properties for the PDF: Extract text action on line 3:

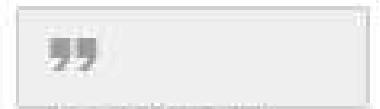
- Text type: Plain text (Structured text would keep the layout – that is, tabs and spaces)
- Page range: All pages
- The properties should look as in the following screenshot:

PDF: Extract text

Text type



Page range



”

(e.g. 1, 3, 5-12)

Extracting text from a PDF file

The final property to set is to specify the output text file. To do this, set the following properties for the PDF: Extract text action on line 3:

- Export data to text file: C:\Hands-On-RPA-with-AA-Sample-Data\lesson14_Letter.txt
- Overwrite files with the same name: Checked
- The properties should look as in the following screenshot:

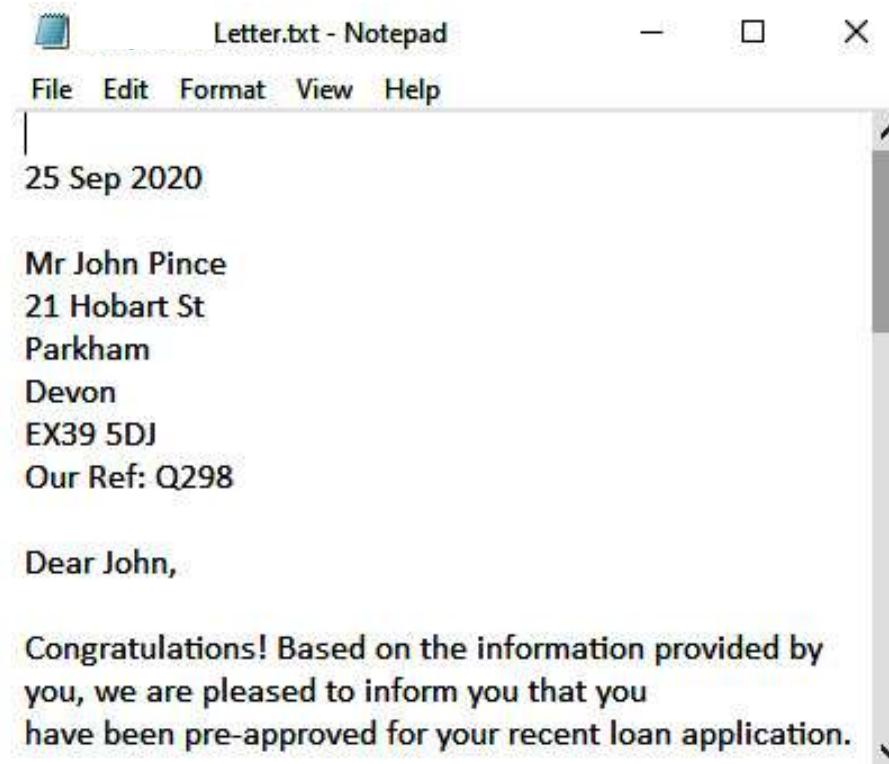


Extracting text from a PDF file

- Click on Save.
- The development interface for this section should look as in the following screenshot:



Extracting text from a PDF file



Extracting an image from a PDF file

PDF: Extract image

Saves PDF document as an image file.

PDF path

Control Room file Desktop file Variable

“ C:\Hands-On-RPA-with-AA-Sample-Data\

Chart.pdf

Required extension: ".pdf"

User password (optional)

Credential Variable Insecure string

Owner password (optional)

Credential Variable Insecure string

Extracting an image from a PDF file

PDF: Extract image

Page range

All pages

Pages

“ ”

(e.g. 1, 3, 5-12)

Type of image to be converted to

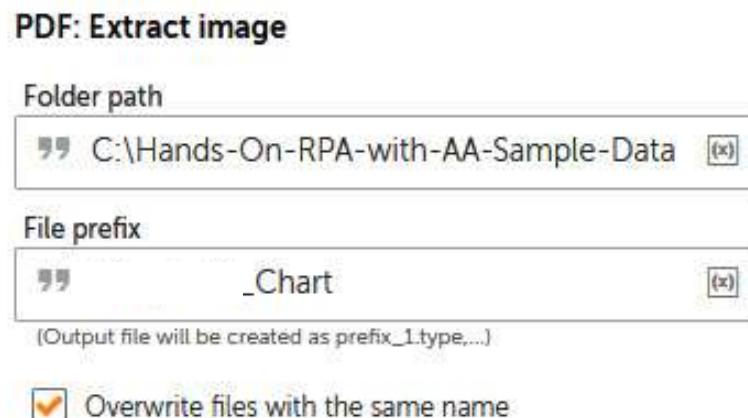
JPEG ▾

JPEG quality

100

Extracting an image from a PDF file

- File prefix: lesson14_Chart
- Overwrite files with the same name: Checked
- The properties should look as in the following screenshot:



Extracting an image from a PDF file

- X Resolution(dpi): 200
- Y Resolution(dpi): 200
- Image output: Color
- Color property: True color (32 bits)
- The properties should look as in the following screenshot:

PDF: Extract image

X Resolution(dpi)

200

Y Resolution(dpi)

200

Image output

Color

Color property

True color (32 bits)

Grayscale

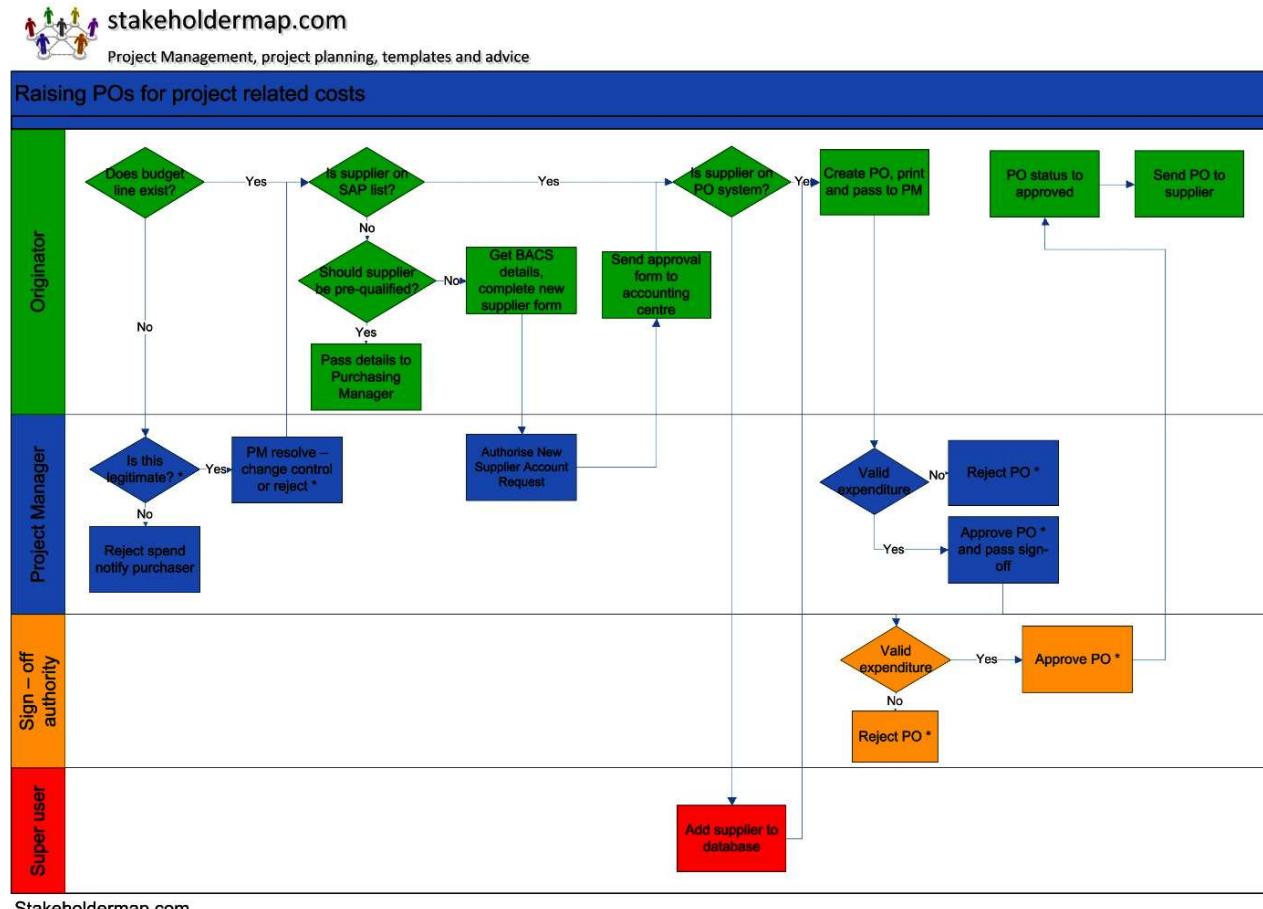
Color property

Extracting an image from a PDF file

- Click on Save.
- The development interface for this section should look as in the following screenshot:



Extracting an image from a PDF file



stakeholdermap.com

Splitting a PDF file

PDF: Split document

Splits a PDF file into multiple PDF files.

PDF path

Control Room file

Desktop file

Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

Games.pdf

Required extension: ".pdf"

User password (optional)

Credential

Variable

Insecure string

Owner password (optional)

Credential

Variable

Insecure string

Splitting a PDF file

PDF: Split document

Output file creation options

- Number of pages per extracted PDF

1

(x)

- Single file with selected pages

" "

(e.g. 1, 3, 5-12)

- Blank page as a separator

- Bookmark level per file

Bookmark Level

#

Splitting a PDF file

PDF: Split document

Folder path

“ C:\Hands-On-RPA-with-AA-Sample-Data

File prefix

“ _GamesSplit

(Output file will be created as prefix_1.pdf,...)

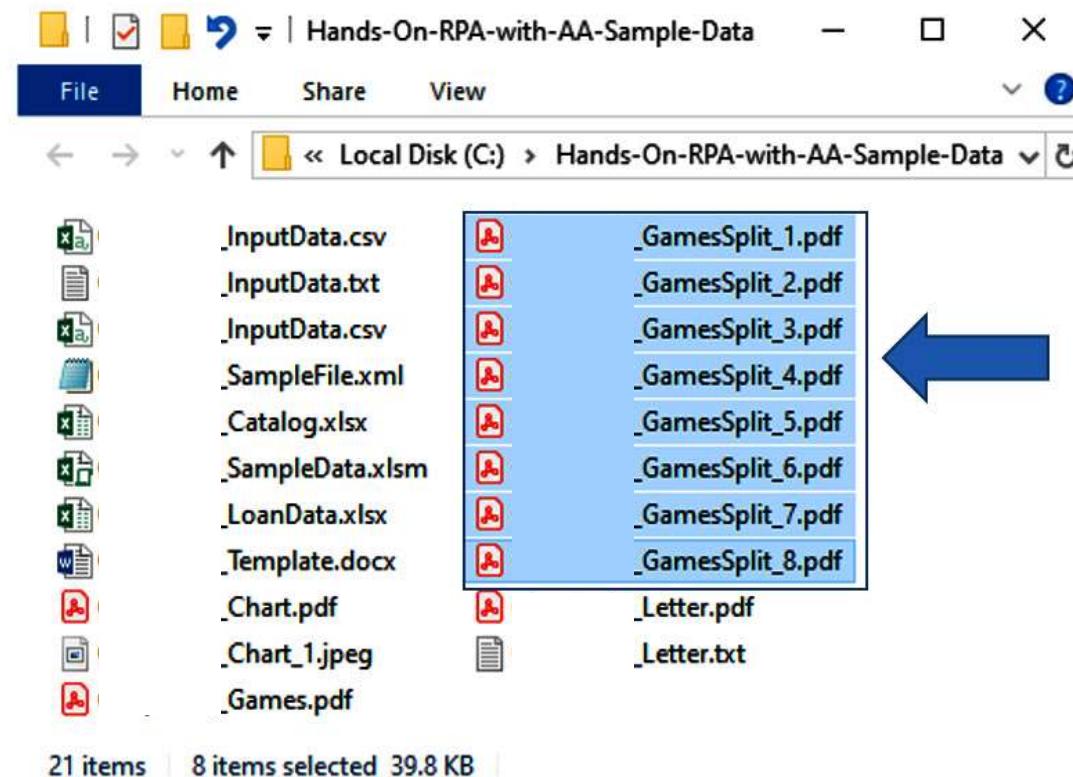
Overwrite files with the same name

Splitting a PDF file

- Click on Save.
- The development interface for this section should look as in the following screenshot:



Splitting a PDF file



Merging multiple PDF files

- We need to specify the individual PDF files that we want merging together.
- To do this, click on the Add PDF document button from the properties of the PDF: Merge documents action on line 9:

PDF: Merge documents

Merges multiple PDF documents into a single PDF document



Merging multiple PDF files

Add PDF document Cancel Add

PDF path

Control Room file Desktop file Variable

Browse...

Required extension: ".pdf"

User password (optional)

Credential Variable Insecure string

Pick...

Owner password (optional)

Credential Variable Insecure string

Pick...

Pages

All pages
 Specific Pages

(e.g. 1, 3, 5-12)

Merging multiple PDF files

PDF: Merge documents

Merges multiple PDF documents into a single PDF document

PDF documents (8)

File	Pages	Specific Pages	⋮
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_1.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_2.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_3.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_4.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_5.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_6.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_7.pdf	All pages	--
C:\Hands-On-RPA-with-AA-Sample-Data\	_GamesSplit_8.pdf	All pages	--

Add PDF document

Merging multiple PDF files

- Output file path: C:\Hands-On-RPA-with-AA-Sample-Data\lesson14_GamesMerged.pdf
- Overwrite existing file: Checked
- The properties should look as in the following screenshot:

PDF: Merge documents

Output file path

C:\Hands-On-RPA-with-AA-Sample-Data _GamesMerged.pdf

Required extension: ".pdf"
e.g. C:\Users\Admin\Test.pdf

Overwrite existing file

Merging multiple PDF files

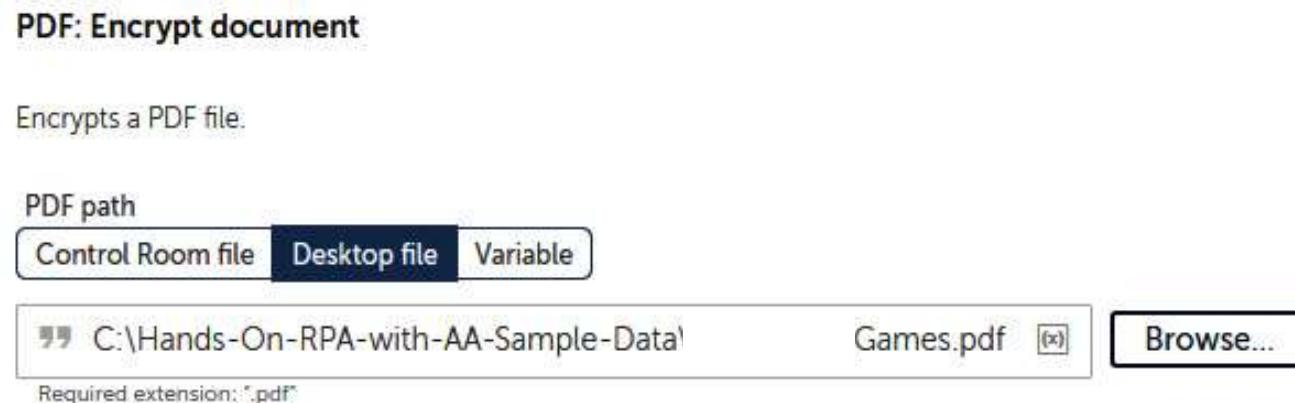
- Click on Save.
- The development interface for this section should look as in the following screenshot:



Encrypting a PDF file

Firstly, we need to specify the PDF file that will be used. To do this, set the following property for the PDF: Encrypt document action on line 11:

- PDF path: Desktop file – C:\Hands-On-RPA-with-AA-Sample-Data\lesson14_Games.pdf
- The property should look as in the following screenshot:



Encrypting a PDF file

PDF: Encrypt document

User password (optional)

Credential

Variable

Insecure string

” Password

(x)

Owner password (optional)

Credential

Variable

Insecure string

Pick...

At least one password out of user password field and owner password field needs to be added.

Encrypting a PDF file

- Next, we need to specify which permissions the password applies to; for this example, we will apply permissions to all.
- To do this, set the following property for the PDF: Encrypt document action on line 11:
- User Permissions to Apply: Check all options

The property should look as in the following screenshot:

PDF: Encrypt document

User Permissions to Apply

- Print
- Modify
- Copy
- Form Fill
- Document Assembly
- Annotation
- Accessibility

Encrypting a PDF file

- Now, we need to specify the level of encryption we want, For this example, we will apply RC4 128-bit encryption.
- To apply this to all, set the following property for the PDF: Encrypt document action on line 11:
- Encryption level: AES 128-bit

The property should look as in the following screenshot:

PDF: Encrypt document

Encryption level

- RC4 40-bit
- RC4 128-bit
- AES 128-bit

Encrypting a PDF file

PDF: Encrypt document

Save encrypted PDF as

"C:\Hands-On-RPA-with-AA-Sample-Dat

_GamesEncrypt.pdf

Required extension: ".pdf"

Overwrite files with the same name

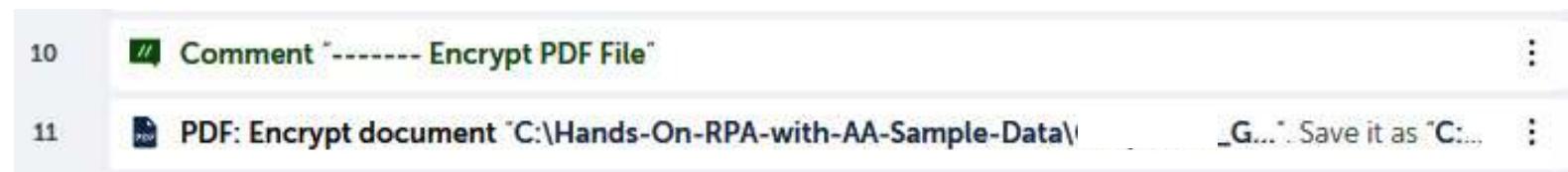
– PDF: Encrypt document – output file properties

12. Click on **Save**. The development interface for this section should look as in the following screenshot:



Encrypting a PDF file

- Click on Save.
- The development interface for this section should look as in the following screenshot:



Decrypting PDF files

PDF: Decrypt document

Decrypts a PDF file.

PDF path

Control Room file

Desktop file

Variable

“ C:\Hands-On-RPA-with-AA-Sample-Data\ _GamesEncrypt.pdf

Required extension: “.pdf”

Decrypting PDF files

PDF: Decrypt document

User/Owner password (optional)

Credential

Variable

Insecure string

” Password



Decrypting PDF files

PDF: Decrypt document

Save the decrypted PDF file as

C:\Hands-On-RPA-with-AA-Sample-Data\ GamesDecrypt.pdf Required extension: ".pdf"

Overwrite files with the same name

– PDF: Decrypt document – output file properties

8. Click on **Save**. The development interface for this section should look as in the following screenshot:



Using the PDF dictionary

Key	Value
pdfTitle	Document Title
pdfFilename	Document Filename
pdfSubject	Document Subject
pdfAuthor	Document Author

Using the PDF dictionary

Create variable

Name
dctPDF
Max characters = 50

Description (optional)
Max characters = 255

Use as input

Use as output

Constant (read-only)

Type Dictionary **Subtype** String

Default value (optional)
This dictionary is empty

+

Cancel **Create**

Using the PDF dictionary

PDF: Decrypt document

Assign PDF properties to a dictionary variable (optional)

dctPDF - Dictionary of Strings



(x)₊

Note: The PDF file name, title, author and subject can be accessed via this dictionary

Using the PDF dictionary

Message box

Displays a message box

Enter the message box window title

” PDF Dictionary

(x)

Enter the message to display

” Title: |\$dctPDF{pdfTitle}\$|

Filename: |\$dctPDF{pdfFilename}\$|

Subject: |\$dctPDF{pdfSubject}\$|

Author: |\$dctPDF{pdfAuthor}\$|

(x)

Using the PDF dictionary

- Click on Save.
- The complete development interface for this lesson should look as in the following screenshot:

1	Comment "-----"	⋮
2	Comment "----- Extract Text"	⋮
3	PDF: Extract text from "C:\Hands-On-RPA-with-AA-Sample-Data" _L..." to "C:\Hands...	⋮
4	Comment "----- Extract Image"	⋮
5	PDF: Extract image "C:\Hands-On-RPA-with-AA-Sample-Data\" ,C..." to a JPEG imag...	⋮
6	Comment "----- Split File"	⋮
7	PDF: Split document "C:\Hands-On-RPA-with-AA-Sample-Data" _G..." as "	⋮
8	Comment "----- Merge Files"	⋮
9	PDF: Merge documents into "C:\Hands-On-RPA-with-AA-Sample-Data\" G..."	⋮
10	Comment "----- Encrypt PDF File"	⋮
11	PDF: Encrypt document "C:\Hands-On-RPA-with-AA-Sample-Data" _G...". Save it as "...	⋮
12	Comment "----- Decrypt PDF File"	⋮
13	PDF: Decrypt document "C:\Hands-On-RPA-with-AA-Sample-Data\" _G...". Save it as ...	⋮
14	Message box "Title: \$dctPDF{pdfTitle} \$ Filename: \$dctPDF{pdfFilename} \$ Subject: \$dctPDF...	⋮
15	Comment "-----"	⋮

Using the PDF dictionary

- When you are ready, please run the bot.
- The bot will now display a message box containing the document properties.
- It should look as in the following screenshot:



Summary

- In this lesson, you have learned how to add automation to your PDF-related tasks.
- The walk-throughs have enabled you to build a bot that performs many actions against PDF documents.
- This has included extracting text and images from PDF documents, as well as splitting and merging multiple documents.
- This lesson also demonstrated how to encrypt and decrypt files and how to get access to the document properties by using a Dictionary type variable.

Working with Databases



Working with Databases

- In this lesson, we will be using the following packages:



Working with Databases

In this lesson, we will cover the following topics:

- Connecting to, and disconnecting from, databases
- Reading data from databases
- Updating databases

Connecting to, and disconnecting from, databases

When working with databases, the first thing you would do with most development platforms is to establish a connection with the database. There are many types of databases that are used, the most widely used being the following:

- SQL or Oracle
- SQLite
- MS Access

Connecting to, and disconnecting from, databases

- Add a new Comment action as "--" on line 7 and click on Save.
- Your initial development interface should look like the following screenshot:



Connecting to a SQL/Oracle database

Firstly, we need to specify the Session details.

To do this, set the following properties for the Database: Connect action on line 4:
Session name: db_Session

- Connection mode: User defined (Default is used for connection strings only)
- The properties should look like the following screenshot:

Database: Connect

Connects to a database

Session name

db_Session

Connection mode

User defined

Connecting to a SQL/Oracle database

SQL Server Settings

Database Type: Microsoft SQL Server

Enter a server name: MDG181K\SQL_DB

Database name: dbSales

Username: Insecure string – *****

Password: Insecure string – *****

Instance name:

Oracle Database Settings

Database Type: Oracle

Enter a server name: MDG181K\ORA_DB

Oracle system id:

Username: Insecure string – *****

Password: Insecure string – *****

Port: 1521

Connecting to a SQL/Oracle database

Connecting to SQL Server

Database: Connect

Database type
Microsoft SQL Server

Enter a server name
MDG181K\SQL_DB

Database name (optional)
dbSales

User name
Insecure string

Password
Insecure string

Instance name (optional)

Connecting to Oracle Server

Database: Connect

Database type
Oracle

Enter a server name
MDG181K\ORA_DB

Oracle system id(sid) (optional)

User name
Insecure string

Password
Insecure string

Port
1521

Connecting to a SQL/Oracle database

Database: Disconnect

Disconnects from the database

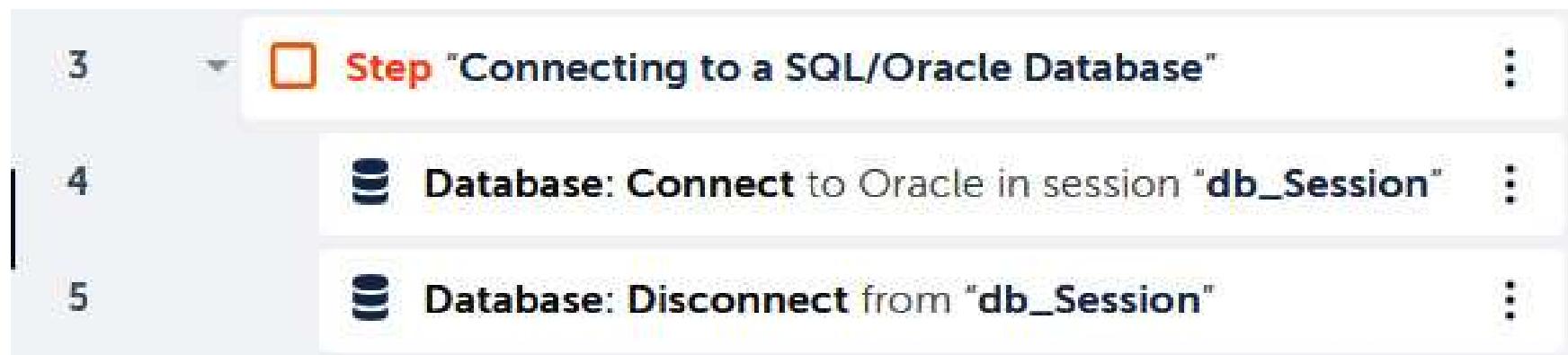
Session name

'' db_Session



Connecting to a SQL/Oracle database

- Click on Save.
- Your development interface for this section should look like the following screenshot:



Connecting to a SQLite database

Firstly, we need to specify the session details. To do this, set the following properties for the Database: Connect action on line 7:Session name: db_SQLite

- Connection mode: User defined (Default is used for connection strings only)
- The properties should look like the following screenshot:

Database: Connect

Connects to a database

Session name

db_SQLite

Connection mode

User defined

Connecting to a SQLite database

Database: Connect

Database type

SQLite

Database file path

Control Room file

Desktop file

Variable

“ C:\Hands-On-RPA-with-AA-Sample-Data\

.SQLite.db

Required extension: ".db"
Only .db files are allowed.

Connecting to a SQLite database

- Set the following property for the Database: Disconnect action on line 8:Session name: db_SqLite
- The property should look like the following screenshot:



Connecting to a SQLite database

- Click on Save.
- Your development interface for this section should look like the following screenshot:



Connecting to an Access database

We first need to specify the session details. To do this, set the following properties for the Database:

Connect action on line 10:Session name: db_Access

- Connection mode: User defined (Default is used for connection strings only)
- The properties should look like the following screenshot:

Database: Connect

Connects to a database

Session name

db_Access	(x)
-----------	-----

Connection mode

User defined	▼
--------------	---

Connecting to an Access database

Database: Connect

Database type

Microsoft Access



Database file path

Control Room file

Desktop file

Variable

„ C:\Hands-On-RPA-with-AA-Sample-Data\

.Access.accdb

Browse...

Required extensions: ".accdb", ".mdb"

Only .accdb and .mdb files are allowed.

Connecting to an Access database

Set the following property for the Database: Disconnect action on line 11:

- Session name: db_Access
- The property should look like the following screenshot:

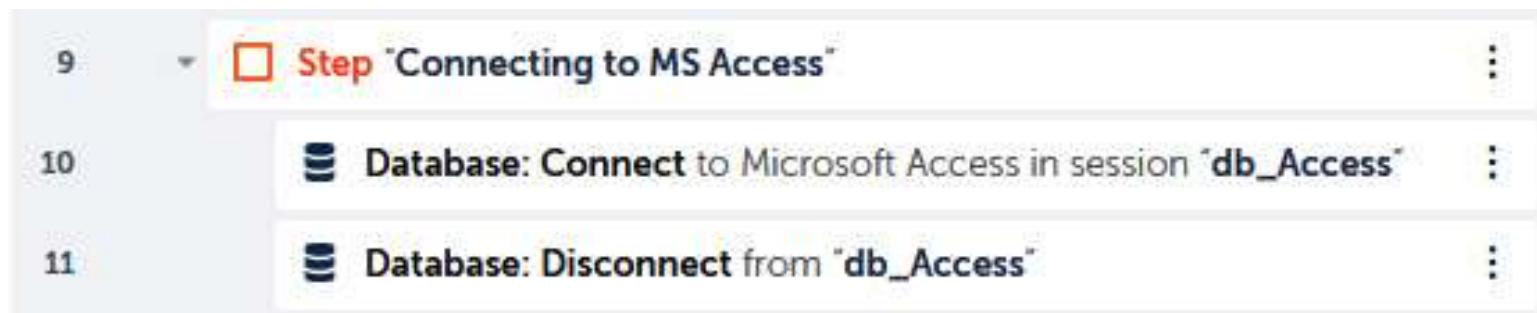
Database: Disconnect

Disconnects from the database

Session name

Connecting to an Access database

- Click on Save.
- Your development interface for this section should look like the following screenshot:



Connecting using a connection string

Firstly, we need to specify the session details.

To do this, set the following properties for the Database: Connect action on line 13:Session name: db_ConStr

- Connection mode: Default
- The properties should look like the following screenshot:

Database: Connect

Connects to a database

Session name

” db_ConStr

Connection mode

Default

Connecting using a connection string

- Set the following property for the Database: Connect action on line 13:Connection string: Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\Hands-On-RPA-with-AA-Sample-Data\lesson12_LoanData.xlsx;Extended Properties="Excel 12.0 Xml;HDR=YES;IMEX=1";
- The property should look like the following screenshot:



Connecting using a connection string

Database: Disconnect

Disconnects from the database

Session name

'' db_ConStr



Connecting using a connection string

- Click on Save, Your development interface for this section should look like the following screenshot:



Reading data from databases

	Field Name	Data Type
!	ID	AutoNumber
	Country	Short Text
	Item Type	Short Text
	Sales Channel	Short Text
	Order Priority	Short Text
	Order Date	Date/Time
	Order ID	Number
	Ship Date	Date/Time
	Units Sold	Number
	Unit Price	Currency
	Unit Cost	Currency
	Total Revenue	Currency
	Total Cost	Currency
	Total Profit	Currency

Reading data from databases

- To achieve this, we will run the following SQL statement:

```
SELECT [Item Type] As Type, Count([Item Type]) as Orders, Sum([Units Sold]) as Quantity, Sum([Total Profit]) as Profit FROM tblSales GROUP BY [Item Type] ORDER BY [Item Type]
```

Reading data from databases

- Keeping up with tidy scene, it would make sense to collapse the Step in line 2.
- Your development interface should look like the following screenshot:



Reading data from databases

Database: Connect

Connects to a database

Session name

db_Access

Connection mode

User defined

Database type

Microsoft Access

Database file path

Control Room file

Desktop file

Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

_Access.accdb

Required extensions: ".accdb", ".mdb"
Only .accdb and .mdb files are allowed.

Reading data from databases

Database: Read from

Retrieves data from the database

Session name

db_Access

[x]

Enter SELECT Statement

SELECT [Item Type] As Type, Count([Item Type]) as Orders, Sum([Units Sold]) as Quan

[x]

Maximum number of records to fetch (optional)

#

[x]

Timeout for the query in seconds (optional)

#

[x]

Export data to CSV

File Path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

Sales.csv

[x]

Browse...

Required extension: ".csv"

Encoding (optional)

ANSI

▼

Export data with header

When saving

Overwrite existing file

Append existing file

Reading data from databases

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in a SQL query dataset ▾

Iterator for each row in sql query dataset.

Session name

” db_Access (x)

Assign the current row to this variable

recData - Record ▾ (x)

Reading data from databases

Message box

Displays a message box

Enter the message box window title

” Reading a SQL Dataset

Enter the message to display

” Type: |\$recData[0]\$|
Orders: |\$recData[1]\$|
Quantity: |\$recData[2]\$|
Profit: |\$recData[3]\$|

Scrollbar after lines

30

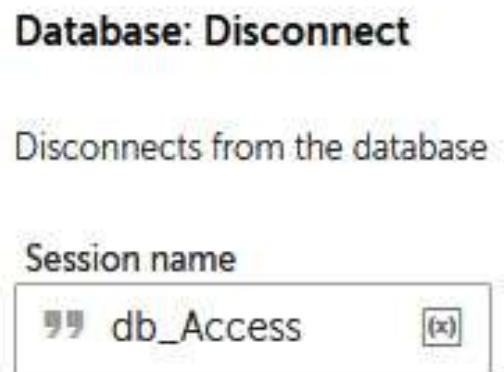
Close message box after

Seconds

5

Reading data from databases

- Set the following property for the Database: Disconnect action on line 20:Session name: db_Access
- The property should look like the following screenshot:



Reading data from databases

- Click on Save.
- Your development interface for this section should look like the following screenshot:



Reading data from databases

	A	B	C	D
1	Item Type	ORDERS	QUANTITY	PROFIT
2	Baby Food	7	40545	3886643.7
3	Beverages	8	56708	888047.28
4	Cereal	7	25877	2292443.43
5	Clothes	13	71260	5233334.4
6	Cosmetics	13	83718	14556048.66
7	Fruits	10	49998	120495.18
8	Household	9	44727	7412605.71
9	Meat	2	10675	610610
10	Office Supplies	12	46967	5929583.75
11	Personal Care	10	48708	1220622.48
12	Snacks	3	13637	751944.18
13	Vegetables	6	20051	1265819.63

Chapter15_Sales

Reading data from databases

SQL statement to retrieve data using the Excel connection string used earlier:

```
Connection string: SELECT * FROM [Approved$]$
```

```
SQL Statement: SELECT * FROM [Approved$]$
```

The output CSV file should look like this:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	(Ref)	(Title)	(Forename)	(Surname)	(Address)	(City)	(County)	(Postcode)	(Amount)	(Term)	(Interest)	(Payable)	(Monthly)
2	Q298	Mr	John	Pince	21 Hobart	Parkhar Devon		EX39 5DJ	5000	24	0.034	5176.08	215.67
3	Q299	Mrs	Vannessa	Casper	45 Bradfiel	Newquay Cornwall		TR7 1LS	3500	12	0.085	3657.12	304.76
4	Q300	Miss	Sarah	Mchughes	73 Parkfield	Parwick Derbyshire	DE6 1QN		7500	48	0.03	7961.76	165.87
5	Q301	Dr	David	Hawkin	30 Aughtor Norton	Staffordshir	WS11 9RH		8000	60	0.03	8616	143.6
6	Q302	Mr	Roger	Day	7 Richmon	Hilton Aberdeens	AB24 2RR		4000	36	0.085	4524.48	125.68

– Output CSV file using the Excel connection string

SQL statement to retrieve data from the sample SQLite database:

```
Connection string: SQLite Database - Chapter15_SQLite.db
```

```
SQL Statement: SELECT * FROM playlists
```

Reading data from databases

- The output CSV file should look like this:

	A	B
1	PlaylistId	Name
2		1 Music
3		2 Movies
4		3 TV Shows
5		4 Audiobooks
6		5 90's Music
7		6 Audiobooks
8		7 Movies
9		8 Music
10		9 Music Videos
11		10 TV Shows
12		11 Brazilian Music
13		12 Classical
14		13 Classical 101 - Deep Cuts
15		14 Classical 101 - Next Steps
16		15 Classical 101 - The Basics
17		16 Grunge
18		17 Heavy Metal Classic
19		18 On-The-Go 1

Updating databases

You have done a great job so far, but now we will actually start editing the data in our databases. You will learn how to execute the following types of SQL statements:

- Insert
- Update
- Delete

Updating databases

Database: Connect

Connects to a database

Session name

db_Access

Connection mode

User defined

Database type

Microsoft Access

Database file path

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

.Access.accdb

Required extensions: ".accdb", ".mdb"
Only .accdb and .mdb files are allowed.

Updating databases

- Set the following property for the Database:
Disconnect action on
line 26:Session
name: db_Access
- The property should look like the following screenshot:

Database: Disconnect

Disconnects from the database

Session name

Updating databases

- Click on Save, Your development interface for this section should look like the following screenshot:

21	<input type="checkbox"/> Step "Updating to Databases"	:
22	<input type="checkbox"/> Database: Connect to Microsoft Access in session "db_Access"	:
23	<input type="checkbox"/> Step "Inserting data - INSERT Statement"	:
24	<input type="checkbox"/> Step "Updating data - UPDATE Statement"	:
25	<input type="checkbox"/> Step "Deleting data - DELETE Statement"	:
26	<input type="checkbox"/> Database: Disconnect from "db_Access"	:
27	<input type="checkbox"/> Comment "-----"	:

Inserting data

- The Access database, lesson15_Access.accdb, has a table called `tblTypes`.
- This table consists of all the product types sold, There is only one field in this table, called `ItemType`.
- Currently, there are 12 records in this table, as you can see:

tblTypes	ItemType
	Baby Food
	Beverages
	Cereal
	Clothes
	Cosmetics
	Fruits
	Household
	Meat
	Office Supplies
	Personal Care
*	Snacks
	Vegetables

Inserting data

- In the walk-through, our bot will insert a new type of product, Electrical.
- This will bring the total record count to 13.
- To insert this new record, the SQL statement would be as follows:

```
INSERT INTO tblTypes (ItemType) VALUES 'Electrical';
```

Inserting data

Database: Insert/Update/Delete

Executes a statement at the database

Session name

‣ db_Access

(x)

Statement

‣ INSERT INTO tblTypes (ItemType) VALUES 'Electrical';

(x)

Timeout for the query in seconds (optional)

#

(x)

Updating data

- For this walk-through, we will continue working with the `tblTypes` table, One item in the table is Fruits.
- We will configure our bot to update this value to Fresh Fruits.
- The SQL statement to apply this update would be as follows:

```
UPDATE tblTypes SET ItemType = "Fresh Fruits" WHERE ItemType = "Fruits"
```

Updating data

Database: Insert/Update/Delete

Executes a statement at the database

Session name

” db_Access

Statement

” UPDATE tblTypes SET ItemType = "Fresh Fruits" WHERE ItemType = "Fruits"

Timeout for the query in seconds (optional)

#

Deleting data

- This is the final walk-through for this lesson; you will learn how to apply a Delete SQL statement.
- Again, we will work on the same table, tblTypes.
- In this example, we want the bot to delete the record where the ItemType value is Cereal.
- The SQL statement to apply this Delete statement would be as follows:

```
DELETE FROM tblTypes WHERE ItemType ="Cereal";
```

Deleting data

Database: Insert/Update/Delete

Executes a statement at the database

Session name

” db_Access

(x)

Statement

” DELETE FROM tblTypes WHERE ItemType = "Cereal";

(x)

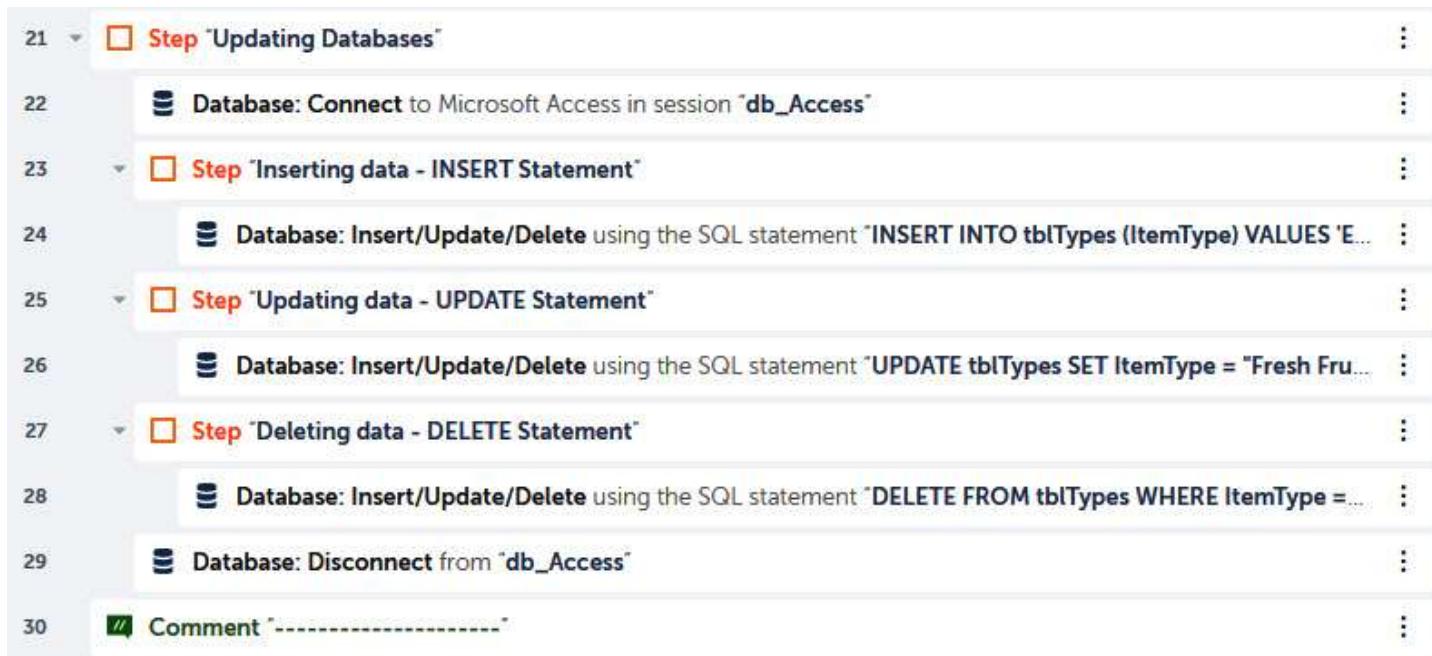
Timeout for the query in seconds (optional)

#

(x)

Deleting data

- Click on Save, The development interface for this entire section should look like the following screenshot:



The screenshot shows a list of numbered steps in a development interface:

- 21 □ Step "Updating Databases"
- 22 Database: Connect to Microsoft Access in session "db_Access"
- 23 □ Step "Inserting data - INSERT Statement"
- 24 Database: Insert/Update/Delete using the SQL statement "INSERT INTO tblTypes (ItemType) VALUES 'E..."
- 25 □ Step "Updating data - UPDATE Statement"
- 26 Database: Insert/Update/Delete using the SQL statement "UPDATE tblTypes SET ItemType = "Fresh Fru..."
- 27 □ Step "Deleting data - DELETE Statement"
- 28 Database: Insert/Update/Delete using the SQL statement "DELETE FROM tblTypes WHERE ItemType = ..."
- 29 Database: Disconnect from "db_Access"
- 30 // Comment "-----"

Deleting data

tblTypes
ItemType
Baby Food
Beverages
Clothes
Cosmetics
Fresh Fruits
Household
Meat
Office Supplies
Personal Care
Snacks
Vegetables
Electrical
*

Summary

- This lesson has taken you into the world of automation with databases.
- As you worked through the different walk-throughs, you have learned how to connect to various types of databases, including using specific connection strings.
- The lesson showed you how to connect to Excel as a database.
- Aside from connecting, you have acquired the skills needed to read, update, insert, and delete data from your databases.
- All this automation will help reduce the time you spend on manual tasks daily.

Building Modular Bots and Sub-Tasks



Building Modular Bots and Sub-Tasks

- In this lesson, we will be using the following packages:

<input checked="" type="checkbox"/> Comment	<input checked="" type="checkbox"/> List
<input checked="" type="checkbox"/> Data Table	<input checked="" type="checkbox"/> Loop
<input checked="" type="checkbox"/> Database	<input checked="" type="checkbox"/> Number
<input checked="" type="checkbox"/> Excel advanced	<input checked="" type="checkbox"/> Step
<input checked="" type="checkbox"/> File	<input checked="" type="checkbox"/> String
<input checked="" type="checkbox"/> If	<input checked="" type="checkbox"/> Task Bot

Building Modular Bots and Sub-Tasks

In this lesson, we will cover the following:

- Designing modular task bots
- Running sub-task bots
- Passing variables between main and sub-task bots
- A working example walk-through

Designing modular task bots

1. Check if the `Output.xlsx` file exists.
 1. If it does, then delete the `Output.xlsx` file.
2. Open an Excel session with the `Output.xlsx-xl_Session` file.
3. Open a SQLite database session with the `Chapter15_SQLite.db-db_Sqlite` file.
 1. Retrieve all non-system table names from `db_Sqlite` to `dataset-1`.
 1. Loop through each table name from `dataset-1`.
 2. Retrieve a maximum of 20 records from current table name in `dataset-1` to `dataset-2`.
 1. Create a new worksheet in `xl_Session` Excel session as current table name.
 2. Output `dataset-2` to worksheet current table name.
 3. Close the `db_Sqlite` SQLite database session.
 4. Close the `xl_Session` Excel session.

Designing modular task bots

1	## Comment -----	:
2	□ Step 'Open Excel Session - xl_Session'	:
3	□ Step 'Open database Session - db_SQLite'	:
4	□ Step 'Get table names from db_SQLite to table'	:
5	□ Step 'Loop each record from table'	:
6	□ Step 'Retrieve dataset for current record to table'	:
7	□ Step 'Create worksheet in xl_Session as current record'	:
8	□ Step 'Output table to worksheet'	:
9	□ Step 'Close database Session - db_SQLite'	:
10	□ Step 'Close Excel Session - xl_Session'	:

Designing modular task bots

Sub-Task Bot 1: Create a new Excel Workbook

1. Check if the output file exists.
2. If it does, then delete the output file.
 1. Create a new output file (opens a new session).
3. Close the Excel session.

Sub-Task Bot 2: Get non-system table names from SQLite database

1. Create a SQLite database session.
 1. Run a SQL statement to extract all non-system table names.
 2. Loop through each table name.
 1. Create a comma-separated string of all the extracted table names.
2. Close the database session.

Sub-Task Bot 3: Copy table data to a new worksheet

1. Create a SQLite database session.
 1. Run a SQL statement to extract data from the specified table.
 2. Create an Excel session to the specified workbook.
 1. Create new worksheet as table name.
 2. Export table data to worksheet.
3. Close Excel session.
2. Close database session.

Designing modular task bots

- With these three sub-task bots, our main task bot would be like the controller bot.
- It would run the sub-task bots in the correct order so the bot performs the complete process without any issues.
- The main task bot design would be as shown in the following figure:

Main-Task Bot: Export SQLite database non-system table data to Excel

1. Run Sub-Task 1.
2. Run Sub-Task 2.
3. Assign comma-separated table names string to a list.
4. Loop through table names list.
 1. Run Sub-Task 3.

Passing variables between main and sub-task-bots

Create variable

Name
|
Max characters = 50

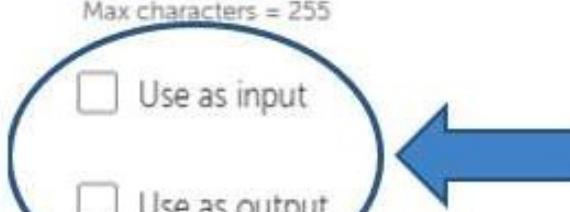
Description (optional)

Max characters = 255

Use as input

Use as output

Constant (read-only)



- These settings define whether this variable will be provided as an input value and/or it will be outputted as part of the Dictionary variable to the calling task.
- The inputs and outputs for each task should be as follows.

Building a bot – lesson16_Sub_CreateNewExcel

Set the following properties for the If action on line 5:Condition: File exists

- File path: \$strFile_OutputXL\$
- How long you would like to wait for this condition to be true?: 0
- The properties should look as shown in the following figure:



Building a bot – lesson16_Sub_CreateNewExcel

Set the following properties for the File: Delete action on line 7:File: \$strFile_OutputXL\$

- The properties should look as shown in the following figure:



Building a bot – lesson16_Sub_CreateNewExcel

Excel advanced: Create workbook

Creates an Excel workbook. This action works with xlsx, xls, xlsm and csv files.

Session name

xl_Session



e.g. Session1 or S1

File path

\$strFile_OutputXLS\$



Browse...

Required extensions: ".xlsx", ".xls", ".xlsm", ".csv"

e.g. C:\Working\Excel1.xlsx Folder(s) will be created if it doesn't exist

Building a bot – lesson16_Sub_CreateNewExcel

Excel advanced: Close

Closes an excel spreadsheet. This action works with xlsx, xls, xlsm and csv files.

Session name

 (x)

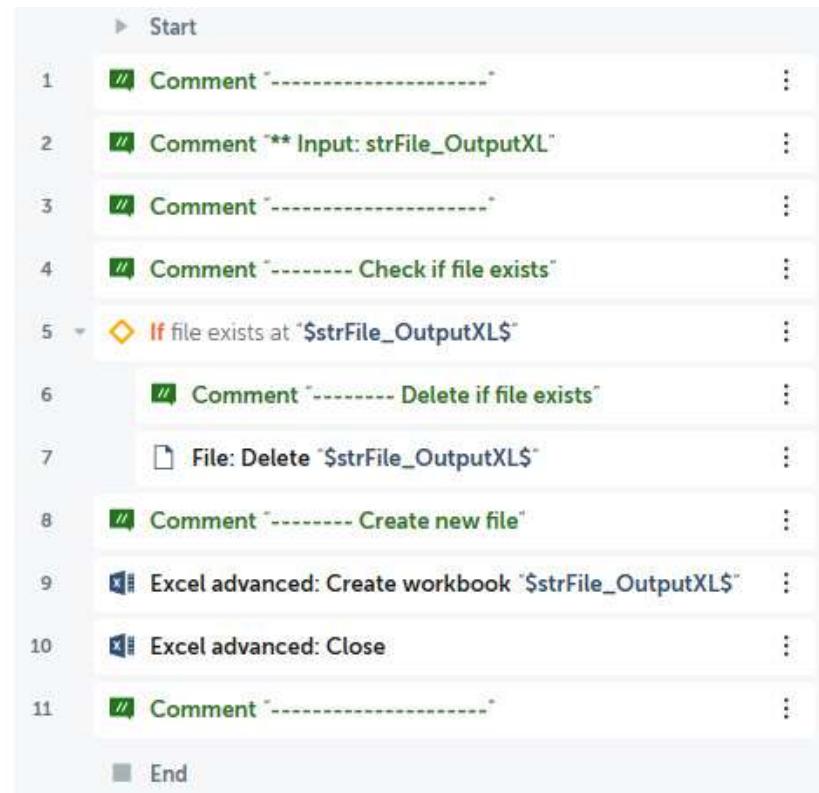
e.g. Session1 or S1



Save changes when closing file

Building a bot – lesson16_Sub_CreateNewExcel

- Click on Save.
- Add a new Comment action just below line 10 as "-----" and click on Save.
- The development interface should look as shown in the following figure:



Building a bot – lesson16_Sub_GetSqLiteTableNames

- This bot connects to a SQLite database and then runs a SQL statement to get all the non-system table names.
- The statement we are using is as follows:

```
SELECT name FROM sqlite_master WHERE type='table' and name Not Like 'sqlite%';
```

Building a bot – lesson16_Sub_GetSQLiteTableNames

- Add a new Comment action as "-----" on line 8 and click on Save; your initial development interface should look like the following figure:



Building a bot – lesson16_Sub_GetSqlLiteTableNames

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

''

(x)

Select the destination string variable

strTableNames - String



(x)



Building a bot – lesson16_Sub_GetSqliteTableNames

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

0

(x)

Specify value to assign to number

Select the destination number variable

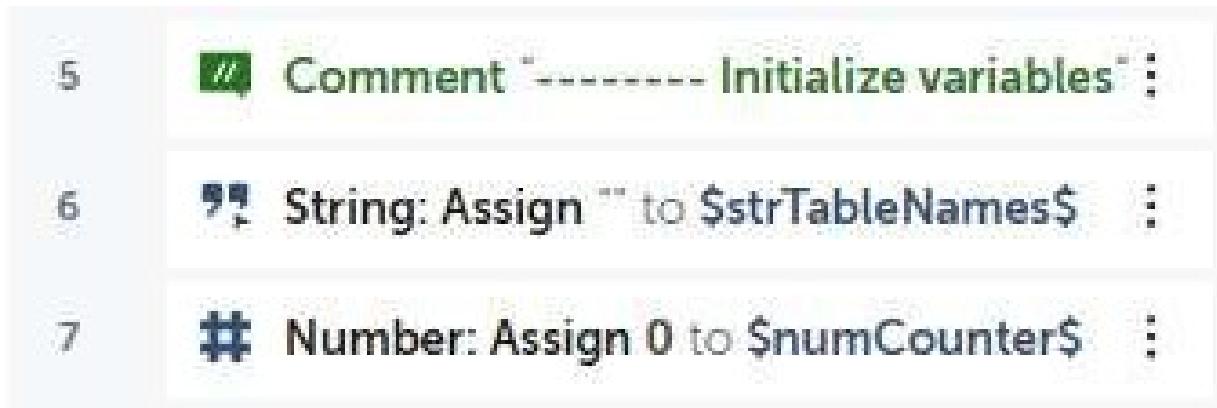
numCounter - Number

▼

(x)

Building a bot – lesson16_Sub_GetSqlLiteTableNames

- Click on Save. This section of the development interface should look as shown in the following figure:



```
5  # Comment ----- Initialize variables
6  ?? String: Assign "" to $strTableNames$ :
7  ## Number: Assign 0 to $numCounter$ :
```

Building a bot – lesson16_Sub_GetSQLiteTableNames

Database: Connect

Connects to a database

Session name

“ db_SqLite ”

Connection mode

User defined

Database type

SqLite

Database file path

Control Room file Desktop file Variable

“ \$strFile_SqLiteDB\$ ”

Required extension: “.db”
Only .db files are allowed.

Building a bot – lesson16_Sub_GetSQLiteTableNames

- Set the following properties for the Database: Read from action on line 10:Session name: db_SQLite
- Enter SELECT Statement: SELECT name FROM sqlite_master WHERE type='table' and name Not Like 'sqlite%';
- The properties should look as shown in the following figure:



Building a bot – lesson16_Sub_GetSQLiteTableNames

- Click on Save; this section of the development interface should look as shown in the following figure:



The screenshot shows a list of steps in a development interface:

- 8 Comment "----- Get table names"
- 9 Database: Connect to SQLite in session "db_SQLite"
- 10 Database: Read from database using the SQL statement "SELECT name FR...

Building a bot – lesson16_Sub_G etSQLiteTableNa mes

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in a SQL query dataset ▾

Iterator for each row in sql query dataset.

Session name

db_SQLite (x)

Assign the current row to this variable

recTableName - Record ▾ (x)

Building a bot – lesson16_Sub_G etSQLiteTableNa mes

Number: Increment

Increments a number by specified value

Enter number

\$numCounter\$



Enter increment value

1



Increments number by value (e.g. 1)

Assign the output to variable

numCounter - Number



Building a bot – lesson16_Sub_GetSqlLiteTableNames

If

Runs a sequence of actions if a condition is true

Condition

Number condition

Checks the number variable condition.

Source value

\$numCounter\$

Operator

Equals to(=)

Target value

1

Add condition

Building a bot – lesson16_Sub_GetSqlLiteTableNames

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

\$recTableName[0]\$



Select the destination string variable

strTableNames - String



Building a bot – lesson16_Sub_GetSqlLiteTableNames

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

\$strTableNames\$, \$recTableName[0]\$

(x)

Select the destination string variable

strTableNames - String

▼

(x)
+

Building a bot – lesson16_Sub_GetSQLiteTableNames

- Set the following properties for the Database: Disconnect action on line 18:Session name: db_SQLite
- The properties should look as shown in the following figure:



Building a bot – lesson16_Sub_GetSQLiteTableNames

```
11  // Comment "----- Create comma separated string"
12  ⏴ Loop : For each row in a SQL query dataset
13    # Number: Increment $numCounter$ by 1 and assign result to a $numCounter$ variable
14    ⏴ ◇ If number $numCounter$ Equals to(=) 1
15      " String: Assign $recTableName[0]$ to $strTableNames$
16      ⏴ ◇ If: Else
17        " String: Assign "$strTableNames$,$recTab..." to $strTableNames$
18    Database: Disconnect from "db_SQLite"
19  // Comment "-----"
```

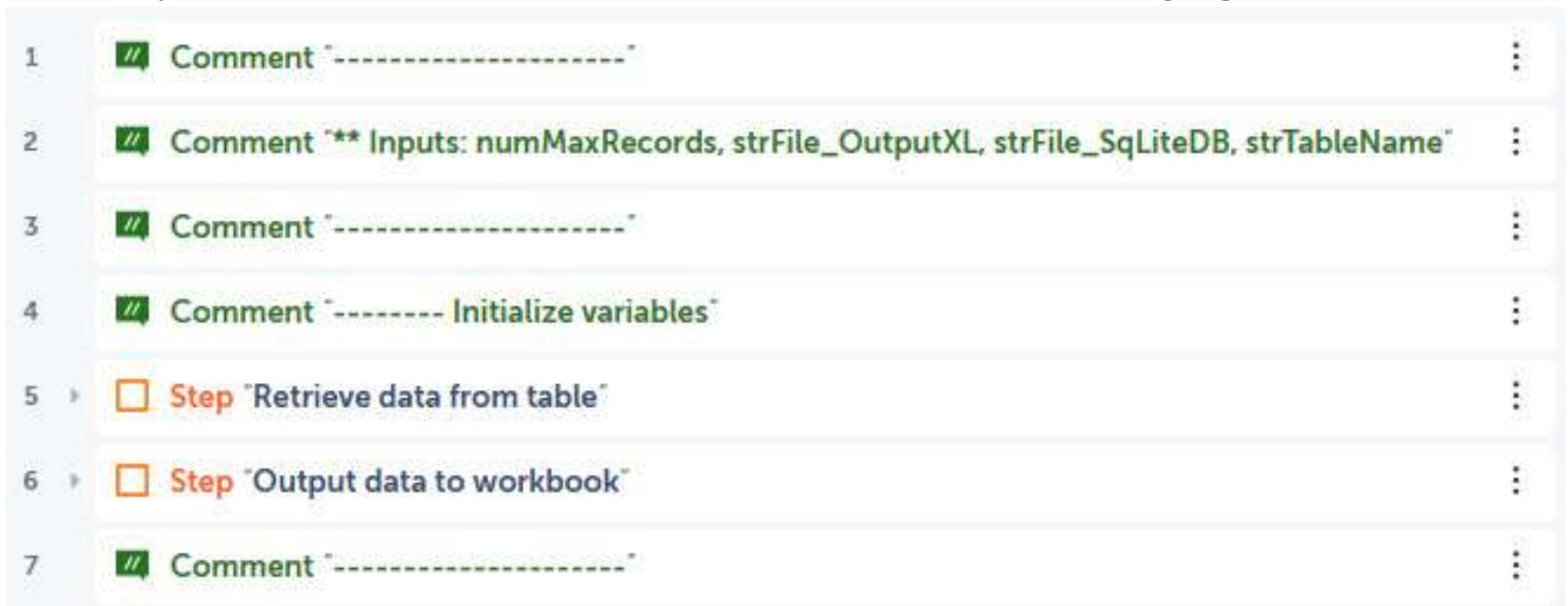
Building a bot – lesson16_Sub_CopySqLiteTableToExcel

- From this, the bot needs to connect to the database and extract data from the given table.
- It should then output all that data to a new worksheet on the given workbook.
- The SQL query that we will be using to get all the data is as follows:

Select * from \$strTableName\$

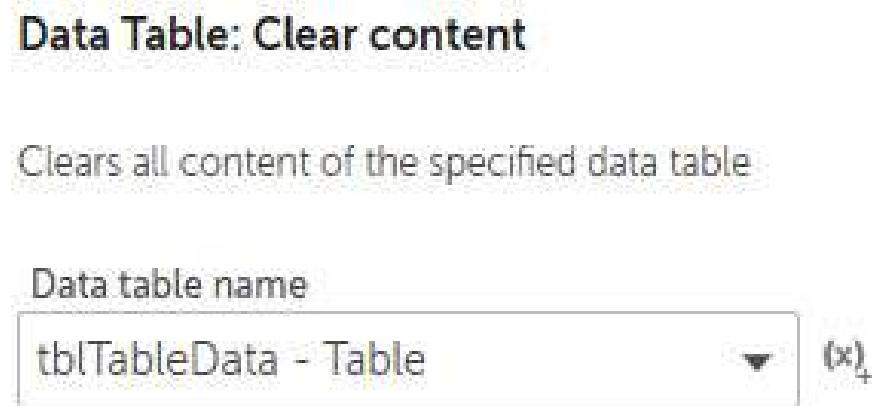
Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Add a new Comment action on line 7 as, "-----" and click on Save; your initial development interface should look as shown in the following figure:



Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Set the following properties for the Data Table: Clear content action on line 5:Data table name: tblTableData - Table
- The properties should look as shown in the following figure:



Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Click on Save; the development interface for this section should look as shown in the following figure:



Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Database: Connect

Connects to a database

Session name

“ db_SqLite ”

Connection mode

User defined

Database type

SQLite

Database file path

Control Room file **Desktop file** Variable

“ \$strFile_SQLiteDB\$ ”

Required extension: “.db”
Only .db files are allowed.

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Database: Export to data table

Exports data from a database to a user specified data table

Session name

“ db_SQLite ”

(x)

Enter SELECT Statement

“ Select * from \$strTableName\$ ”

(x)

Maximum number of records to fetch (optional)

“ # \$numMaxRecords\$ ”

(x)

Timeout for the query in seconds (optional)

“ # ”

(x)

Assigned to

tblTableData - Table

▼

(x)

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Set the following properties for the Database: Disconnect action on line 9:Session name: db_SQLite
- The properties should look as shown in the following figure:

Database: Disconnect

Disconnects from the database

Session name

db_SQLite



Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Click on Save; the development interface for this section should look as shown in the following figure:



Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Set the following properties for the Excel advanced: Open action on line 11:

- Session name: xl_Session
- File path: Desktop file - \$strFile_OutputXL\$
- Open in: Read-write mode
- The properties should look as shown in the following figure:

Excel advanced: Open

Opens an excel spreadsheet. This action works with .xlsx, .xls, .xlsb, .xlsm and .csv files.

Session name

xl_Session

e.g. Session1 or S1

File path

Control Room file Desktop file Variable

\$strFile_OutputXL\$

Required extensions: ".xlsx", ".xls", ".xlsm", ".xlsb", ".csv"
e.g. C:\Working\Excel1.xlsx

Specific sheet name

e.g. Sheet1 or SHEET1

Open in

Read-only mode

Read-write mode

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Set the following properties for the Excel advanced: Create worksheet action on line 12:

- Session name: xl_Session
- Create sheet by: Name – \$strTableName\$
- The properties should look as shown in the following figure:

Excel advanced: Create worksheet

Creates an excel worksheet. This action works with xlsx, xls, xlsb and xlsm files.

Session name

" xl_Session

e.g. Session1 or S1

Create sheet by

Index

#

e.g. 1 or 3

Name

" \$strTableName\$

e.g. Sheet1

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Set the following properties for the Excel advanced: Write from data table action on line 13:Session name: xl_Session

- Enter data table variable: tblTableData - Table
- Enter worksheet name: Specific worksheet – \$strTableName\$
- Specify the first cell: A1
- The properties should look as shown in the following figure:

Excel advanced: Write from data table

Write a data table's contents into a specified worksheet. This action works with xlsx, xls, xlsb and xlsm files.

Session name

„ xl_Session „

e.g. Session1 or S1

Enter data table variable

tblTableData - Table

(x)

(x)

Enter worksheet name:

Active worksheet

Specific worksheet

„ \$strTableName\$ „

(x)

e.g. Sheet1

Specify the first cell

„ A1 „

(x)

e.g. A5 or B10

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

Excel advanced: Close

Closes an excel spreadsheet. This action works with xlsx, xls, xlsm and csv files.

Session name

XL_Session



e.g. Session1 or S1



Save changes when closing file

Building a bot – lesson16_Sub_CopySQLiteTableToExcel

- Click on Save, the development interface for this section should look as shown in the following figure:

10	□ Step "Output data to workbook"	:
11	☒ Excel advanced: Open "\$strFile_OutputXL\$"	:
12	☒ Excel advanced: Create worksheet with name \$strTableName\$:
13	☒ Excel advanced: Write from data table \$tblTableData\$ to worksheet \$strTableName\$:
14	☒ Excel advanced: Close	:
15	〃 Comment "-----"	:

Building a bot – lesson16_Main_SQLiteToExcel

- Add a new Comment action as "-----" on line 8 and click on Save.
- Your initial development interface should look as shown in the following figure:



The screenshot shows a list of steps in a bot development environment. The steps are numbered 1 through 8. Steps 1, 2, 3, and 8 are green 'Comment' blocks with the text '-----'. Step 4 is a green 'Comment' block with the text '----- Initialize variables'. Steps 5, 6, and 7 are orange 'Step' blocks with the labels 'Create Output Workbook', 'Get table names from SQLite database', and 'Output to Excel' respectively. Each step has a vertical ellipsis icon to its right.

```
1 // Comment -----
2 // Comment ** outputs: numMaxRecords, strFile_OutputXL, s...
3 // Comment -----
4 // Comment ----- Initialize variables
5 > □ Step "Create Output Workbook"
6 > □ Step "Get table names from SQLite database"
7 > □ Step "Output to Excel"
8 // Comment -----
```

Building a bot – lesson16_Main_SQLiteToExcel

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

C:\Hands-On-RPA-with-AA-Sample-Data\

_Output.xlsx



Select the destination string variable

strFile_OutputXL - String



Building a bot – lesson16_Main_SQLiteToExcel

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

C:\Hands-On-RPA-with-AA-Sample-Data\SQLite.db

SQLite.db



Select the destination string variable

strFile_SQLiteDB - String



Building a bot – lesson16_Main_SQLiteToExcel

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

20

(x)

Specify value to assign to number:

Select the destination number variable

numMaxRecords - Number



(x)
+

Building a bot – lesson16_Main_SQLiteToExcel

- Click on Save; this section of the development interface should look as shown in the following figure:

```
4  ## Comment ----- Initialize variables-----  
5  ## String: Assign "C:\Hands-On-RPA-with-AA..." to $strFile_OutputXL$  
6  ## String: Assign "C:\Hands-On-RPA-with-AA..." to $strFile_SQLiteDB$  
7  ## Number: Assign 20 to $numMaxRecords$
```

Building a bot – lesson16_Main_SQLiteToExcel

Task Bot: Run

Runs the selected task bot.

Task Bot to run

Current Task Bot Control Room file Variable

Bots\

Input values

Set strFile_OutputXL
 \$strFile_OutputXL\$

Do not repeat

Building a bot – lesson16_Main_SQLiteToExcel

- Click on Save; this section of the development interface should look as shown in the following figure:



Building a bot – lesson16_Main_SQLiteToExcel

Task Bot: Run

Runs the selected task bot.

Task Bot to run

Bots\

.Sub_GetSQLite

Input values

Set strFile_SQLiteDB

” \$strFile_SQLiteDB\$

Do not repeat

Assign the output to variable (optional)

dctTableNames - Dictionary of Strings

Building a bot – lesson16_Main_SQLiteToExcel

- Click on Save; this section of the development interface should look as shown in the following figure:



Building a bot – lesson16_Main_S qLiteToExcel

String: Split

Splits the source string into multiple strings using a delimiter.

Source string

“ \$dctTableNames{strTableNames}\$ ”

(x)

Delimiter

“ , ”

(x)

Delimiter is

- Case sensitive
- Not case sensitive

Split into substrings

- All possible
- Only

#

Assign the output to list variable

IstTableNames - List of Strings

▼

(x)

Building a bot – lesson16_Main_S qLiteToExcel

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each item in the list

iterate list

list

IstTableNames - List

(x)

For

All items in the list

Range

From index (optional)

#

To index (optional)

#

Assign the current value to variable

strTableName - String

(x)

Building a bot – lesson16_Main_S qLiteToExcel

Task Bot: Run

Runs the selected task bot.

Task Bot to run

Current Task Bot Control Room file Variable

Bots\ _Sub_CopySQLiteTableToExcel

Input values

Set strFile_OutputXL

`# $strFile_OutputXL$`

(x)

Set strFile_SQLiteDB

`# $strFile_SQLiteDB$`

(x)

Set strTableName

`# $strTableName$`

(x)

Set numMaxRecords

`# $numMaxRecords$`

(x)

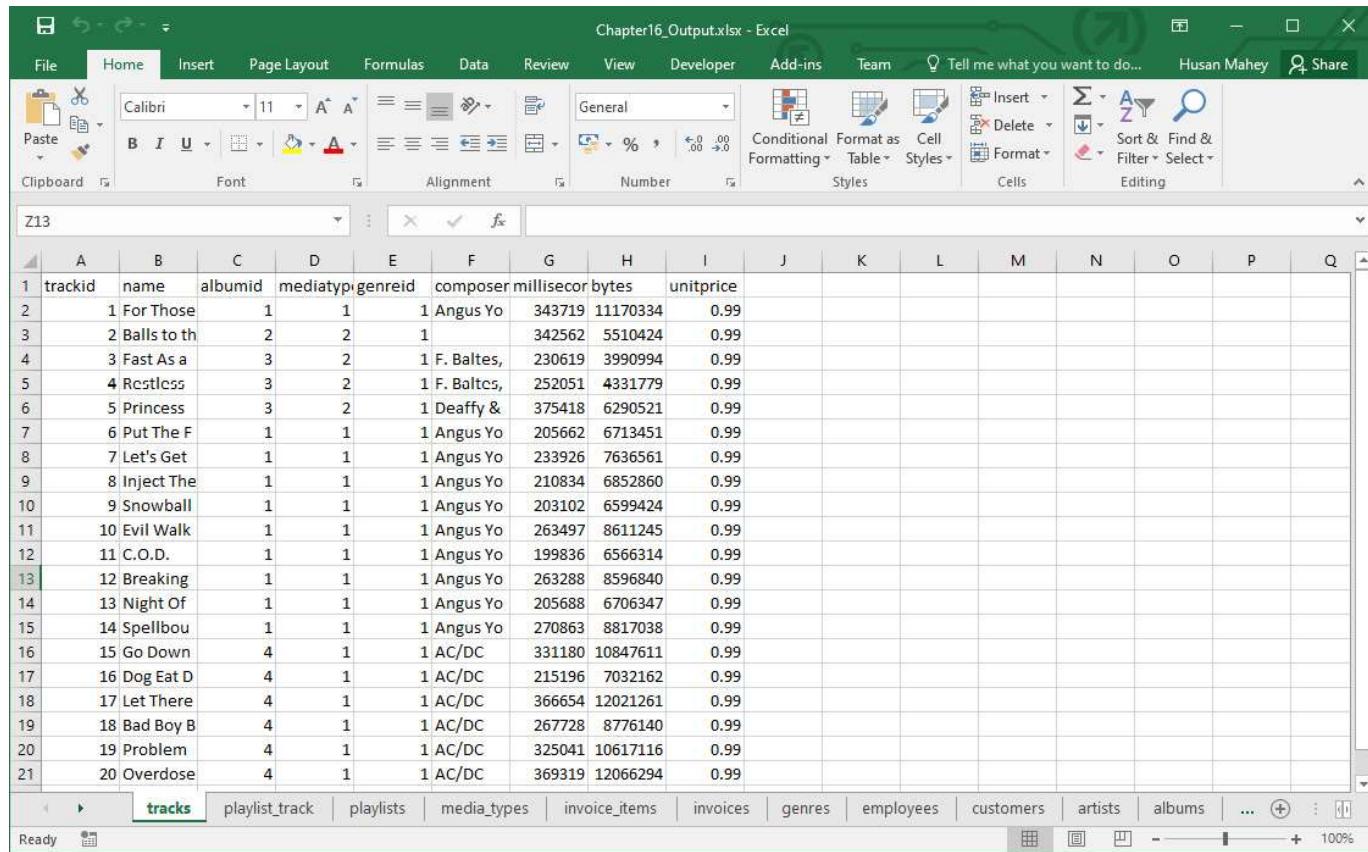
Do not repeat

Building a bot – lesson16_Main_SQLiteToExcel

- Click on Save.
- This section of the development interface should look as shown in the following figure:

```
12  □ Step "Output to Excel"
13      " String: Split $dctTableNames{strTableNames}$ with delimiter "," and assign the result to ...
14  ▶ ⚡ Loop : For each item in the list
15      ⚡ Task Bot: Run "Bots\Chapter16_Sub_CopySQLiteTableToExcel" and assign output to ...
16  // Comment "-----"
```

Building a bot – lesson16_Main_SQLiteToExcel



The screenshot shows a Microsoft Excel spreadsheet titled "Chapter16_Output.xlsx - Excel". The active sheet is named "tracks". The table contains 21 rows of data with the following columns: trackid, name, albumid, mediatype, genreid, composer, millisec, bytes, and unitprice. The data includes various tracks by artists like Angus Yo, F. Baltes, and AC/DC.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	trackid	name	albumid	mediatype	genreid	composer	millisec	bytes	unitprice								
2	1	For Those	1	1	1	Angus Yo	343719	11170334	0.99								
3	2	Balls to th	2	2	1		342562	5510424	0.99								
4	3	Fast As a	3	2	1	F. Baltes,	230619	3990994	0.99								
5	4	Rcticss	3	2	1	F. Baltcs,	252051	4331779	0.99								
6	5	Princess	3	2	1	Deaffy &	375418	6290521	0.99								
7	6	Put The F	1	1	1	Angus Yo	205662	6713451	0.99								
8	7	Let's Get	1	1	1	Angus Yo	233926	7636561	0.99								
9	8	Inject The	1	1	1	Angus Yo	210834	6852860	0.99								
10	9	Snowball	1	1	1	Angus Yo	203102	6599424	0.99								
11	10	Evil Walk	1	1	1	Angus Yo	263497	8611245	0.99								
12	11	C.O.D.	1	1	1	Angus Yo	199836	6566314	0.99								
13	12	Breaking	1	1	1	Angus Yo	263288	8596840	0.99								
14	13	Night Of	1	1	1	Angus Yo	205688	6706347	0.99								
15	14	Spellbou	1	1	1	Angus Yo	270863	8817038	0.99								
16	15	Go Down	4	1	1	AC/DC	331180	10847611	0.99								
17	16	Dog Eat D	4	1	1	AC/DC	215196	7032162	0.99								
18	17	Let There	4	1	1	AC/DC	366654	12021261	0.99								
19	18	Bad Boy B	4	1	1	AC/DC	267728	8776140	0.99								
20	19	Problem	4	1	1	AC/DC	325041	10617116	0.99								
21	20	Overdose	4	1	1	AC/DC	369319	12066294	0.99								

Summary

- There has been a lot covered in this lesson, giving you the skills needed to understand and design modular bots.
- Taking this approach will be a stepping stone to having your own library of smaller sub-bots.
- This saves you from a lot of redevelopment effort, especially when automating larger and complete processes.
- You have learned how to run a sub-bot from within a bot, as well as how to pass parameters between these bots.
- The real-life scenario walk-through provided practical experience of how this actually works in the real world.

Running External Scripts



Running External Scripts

- In this lesson, we will be using the following packages:

 Comment	 Python script
 List	 Step
 Message box	 String
 Number	 VBScript

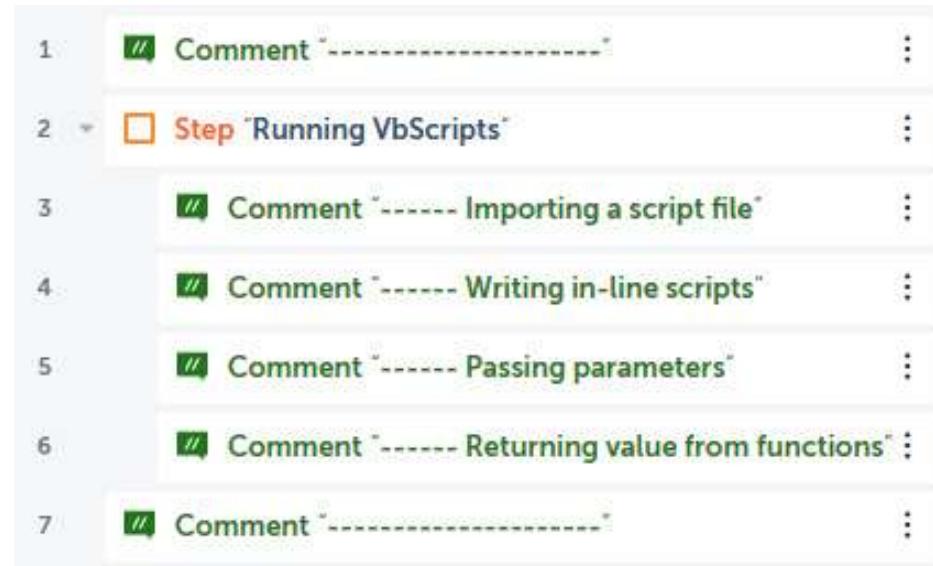
Running External Scripts

In this lesson, we will cover the following:

- Running VBScripts
- Running Python scripts

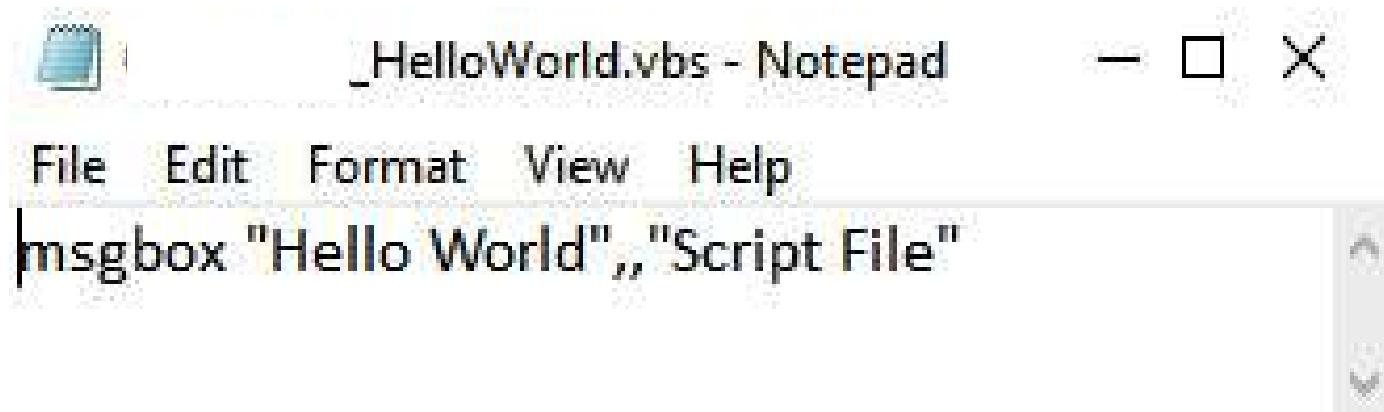
Running VBScripts

- Add a new Comment action below line 6 as "-----", ensuring it is not within the Step on line 2 and click on Save.
- The development interface should look like this:



Importing a script file

- You can run this file independently just by double-clicking on it. The file only has one line of code as shown here:



A screenshot of a Windows Notepad window titled '_HelloWorld.vbs - Notepad'. The window contains the following code:

```
File Edit Format View Help
msgbox "Hello World", "Script File"
```

Importing a script file

VBScript: Open

Opens a VBScript

New VBScript session

vbs_Session (x)

Use the session name to refer to this file in other VBScript actions

VBScript

Import existing file

VBScript file

Control Room file Desktop file Variable

vbs C:\Hands-On-RPA-with-AA-Sample-Data\HelloWorld.vbs (x) Browse...

Required extension: ".vbs"

Manual input

Enter script here

1

Importing a script file

- Set the following properties for the VBScript: Run function action on line 5:VBScript session: vbs_Session
- The properties should look as shown in the following screenshot:

VBScript: Run function

Executes a VBScript function

VBScript session

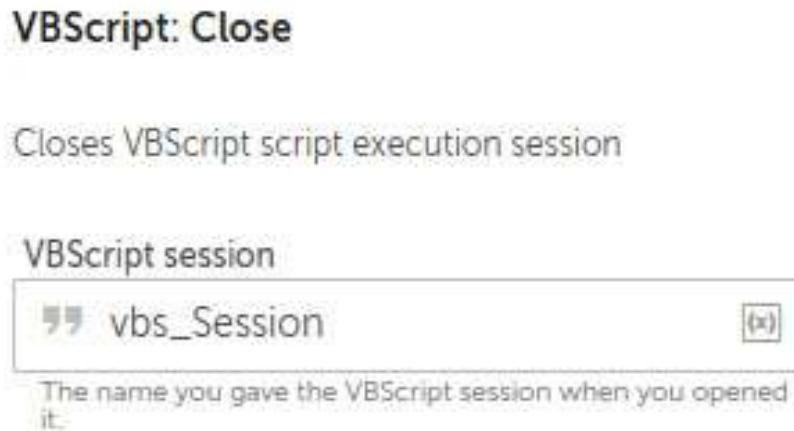
vbs_Session



The name you gave the VBScript session when you opened it.

Importing a script file

- Set the following properties for the VBScript: Close action on line 6:VBScript session: vbs_Session
- The properties should look as shown in the following screenshot:



Importing a script file

- Click on Save.
- The development interface for this section should look like this:



```
3  # Comment ----- Importing a script file
4  $ VBScript: Open VBScript "C:\Hands-On-RPA-with-AA-Sample-Data\
5  $ VBScript: Run function
6  $ VBScript: Close VBScript "vbs_Session"
```

Importing a script file

- Your bot will now run the VBScript file, Give it a test.
- The VBScript should show the following message box:



Writing inline scripts

VBScript: Open

Opens a VBScript

New VBScript session

 vbs_Session (x)

Use the session name to refer to this file in other VBScript actions

VBScript

Import existing file

VBScript file

[Control Room file](#) [Desktop file](#) [Variable](#)

[Browse...](#)

Manual input

Enter script here

```
1 msgbox "Hello World","","In-Line Script"
```

Writing inline scripts

VBScript: Run function

Executes a VBScript function

VBScript session

99 vbs_Session



The name you gave the VBScript session when you opened it.

Writing inline scripts

VBScript: Close

Closes VBScript script execution session

VBScript session

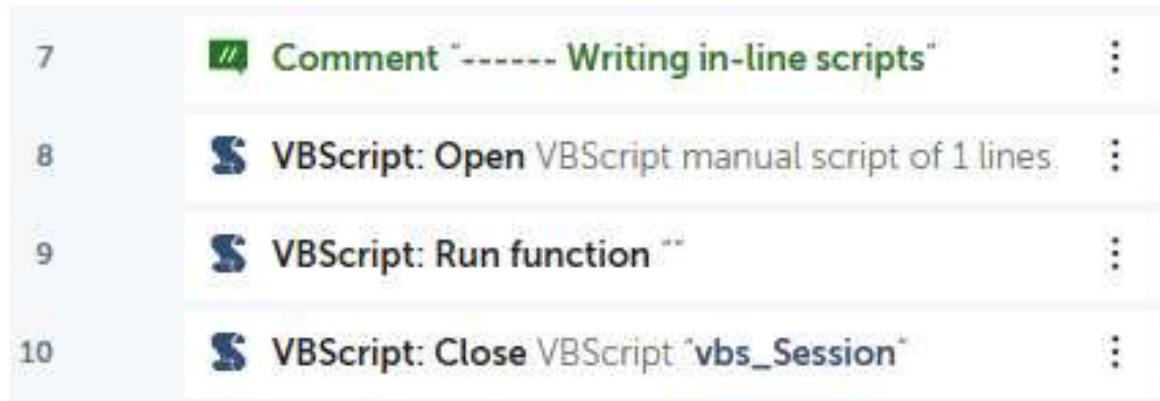
?? vbs_Session



The name you gave the VBScript session when you opened it.

Writing inline scripts

- Click on Save.
- The development interface for this section should look like this:



Writing inline scripts

- Although this is a simple one-line script, it demonstrates how to write your code directly within Automation Anywhere.
- When you run the bot, the VBScript will show the following message box:



Passing parameters

- When passing parameters to a VBScript, they need to be in the form of a List type variable.
- A sample script file is included in the GitHub repository, which takes two parameters.
- This file is lesson17_InputParamters.vbs.
- To use a parameter within a VBScript, you need to utilize the following syntax:

`WScript.Arguments(0)`

Passing parameters

- The index number (0) represents the item in the List and it uses a zero index, so the first value is indexed at 0.
- The contents of the script file are as follows:

```
strValue01 = WScript.Arguments(0)  
strValue02 = WScript.Arguments(1)  
msgbox "Hello " & strValue01 & " " & strValue02,"Input Parameters"
```

Passing parameters

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

“ Husan



Select the destination string variable

strFirstname - String



Passing parameters

String: Assign

Assign or Concatenate the given strings

Select the source string variable(s)/ value (optional)

99 Mahey

(x)

Select the destination string variable

strSurname - String



(x)

Passing parameters

List: Add item

Adds an item to the list at a given index position

List variable

lstParameters - List



(x) +

Item to be added

strFirstname - String



(x) +

Add item

To end of list

At list index

#

Passing parameters

VBScript: Open

Opens a VBScript

New VBScript session

„ vbs_Session (x)

Use the session name to refer to this file in other VBScript actions

VBScript

Import existing file

VBScript file

Control Room file Desktop file Variable

„ C:\Hands-On-RPA-with-AA-Sample-Data\ „ InputParamters.vbs (x) Browse...

Required extension: ".vbs"

Manual input

Enter script here

```
1
```

Passing parameters

VBScript: Run function

Executes a VBScript function

VBScript session

vbs_Session

The name you gave the VBScript session when you opened it.

Enter name of function to be executed (optional)

e.g. AddNumbers

Parameters (optional)

IstParameters - List

Passing parameters

- Set the following properties for the VBScript: Close action on line 18:VBScript session: vbs_Session
- The properties should look as shown in the following screenshot:

VBScript: Close

Closes VBScript script execution session

VBScript session

99 vbs_Session



The name you gave the VBScript session when you opened it.

Passing parameters

- Click on Save.
- The development interface for this section should look like this:

11	Comment ----- Passing parameters	:
12	String: Assign "Husan" to \$strFirstname\$:
13	String: Assign "Mahey" to \$strSurname\$:
14	List: Add item \$strFirstname\$ to \$lstParameters\$:
15	List: Add item \$strSurname\$ to \$lstParameters\$:
16	VBScript: Open VBScript "C:\Hands-On-RPA-with-AA-Sample-Data"	:
17	VBScript: Run function --	:
18	VBScript: Close VBScript "vbs_Session"	:

Returning values from functions

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

25

(x)

Specify value to assign to number

Select the destination number variable

numValue - Number

▼

(x)

Returning values from functions

VBScript: Open

Opens a VBScript

New VBScript session

“ vbs_Session (x)

Use the session name to refer to this file in other VBScript actions.

VBScript

Import existing file

VBScript file

Control Room file Desktop file Variable

“ C:\Hands-On-RPA-with-AA-Sample-Data\ _Functions.vbs (x) Browse...

Required extension: “.vbs”

Manual input

Enter script here

1

Returning values from functions

VBScript: Run function

Executes a VBScript function

VBScript session

vbs_Session (x)

The name you gave the VBScript session when you opened it.

Enter name of function to be executed (optional)

procSquareRoot (x)

e.g. AddNumbers

Parameters (optional)

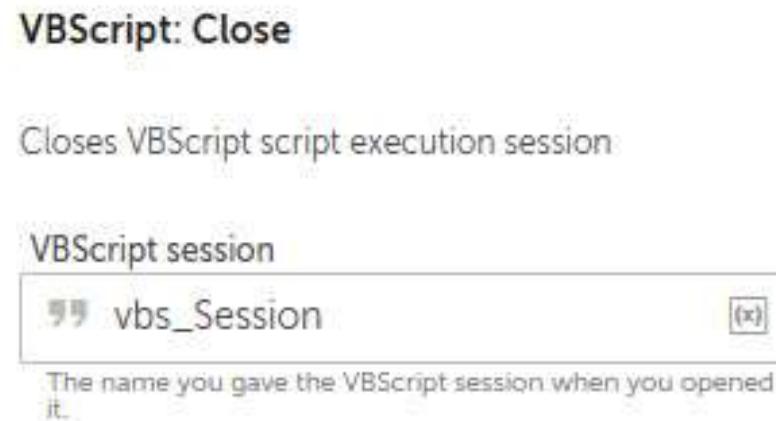
numValue - Number ▼ (x) +

Assign the output to variable (optional)

strReturnValue - String ▼ (x) +

Returning values from functions

- Set the following properties for the VBScript: Close action on line 23:VBScript session: vbs_Session
- The properties should look as shown in the following screenshot:



Returning values from functions

Message box

Displays a message box

Enter the message box window title

Returning values from a VbScript



Enter the message to display

Returned value: \$strReturnValue\$



Scrollbar after lines

30

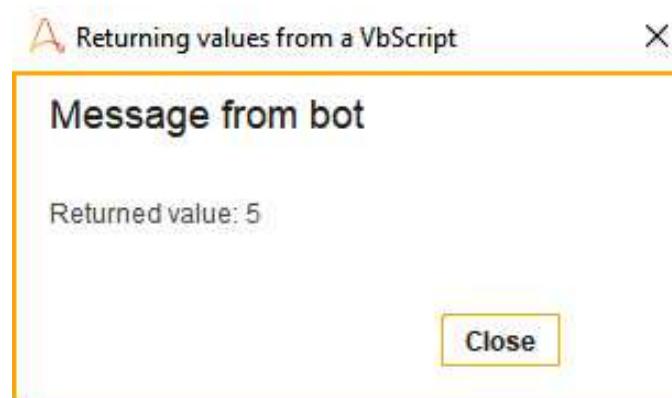


Returning values from functions

- 19 // Comment ----- Returning value from functions
- 20 # Number: Assign 25 to \$numValue\$
- 21 \$ VBScript: Open VBScript "C:\Hands-On-RPA-with-AA-Sample-Data\("
- 22 \$ VBScript: Run function "procSquareRoot"
- 23 \$ VBScript: Close VBScript "vbs_Session"
- 24 # Message box "Returned value: \$strReturnValue\$"
- 25 // Comment -----

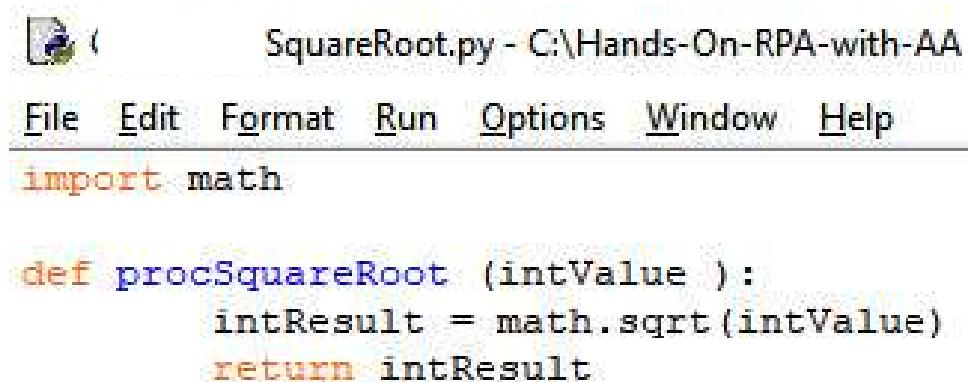
Returning values from functions

- The function will calculate the square root and return the results back to Automation Anywhere.
- The following message box should appear showing the results:



Running Python scripts

- The bot will run a function called procSquareRoot from the Python script file lesson17_SquareRoot.py.
- The function takes a value as an input, It then calculates the square root of this value and returns the result.
- The contents of the Python file look like the following screenshot:



```
File Edit Format Run Options Window Help
import math

def procSquareRoot (intValue ):
    intResult = math.sqrt(intValue)
    return intResult
```

Running Python scripts

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

36

(x)

Specify value to assign to number

Select the destination number variable

numValue - Number



(x)
+

Running Python scripts

Python script: Open

Opens a Python script

New Python session

py_Session [x]

Use the session name to refer to this file in other Python actions

Python

Import existing file

Python file

Control Room file Desktop file Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

SquareRoot.py [x]

Browse...

Required extension: ".py"

Manual input

Enter script here

1

Python runtime version

2

3

Running Python scripts

Python script: Execute function

Executes a Python function

Python session

py_Session (x)

The name you gave the Python session when you opened it.

Enter name of function to be executed

procSquareRoot (x)

e.g. AddNumbers

Arguments to the function (optional)

numValue - Number ▼ (x) +

Supports 0 or 1 argument

Assign the output to variable (optional)

strReturnValue - String ▼ (x) +

Running Python scripts

Set the following properties for the Python script: Close action on line 29:

- Python session: py_Session
- The properties should look as shown in the following screenshot:

Python script: Close

Closes Python script execution session

Python session

py_Session



The name you gave the Python session when you opened it.

Running Python scripts

Message box

Displays a message box

Enter the message box window title

Returning values from a Python Script

(x)

Enter the message to display

\$strReturnValue\$

(x)

Scrollbar after lines

30

(x)

Running Python scripts

- Click on Save.
- The development interface for this section should look like this:

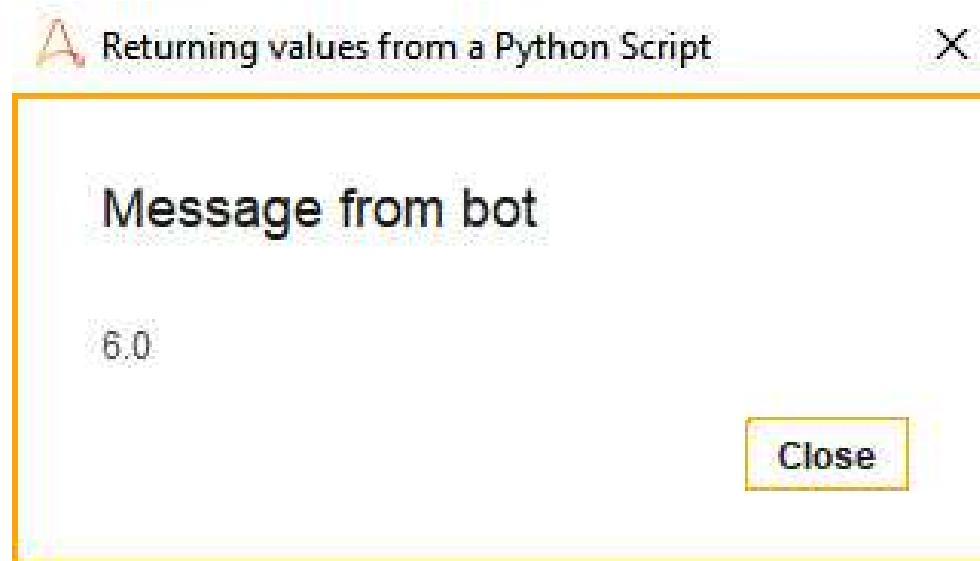


The screenshot shows a software interface with a list of steps numbered 25 to 31. Step 25 is collapsed. Step 26 is a 'Number: Assign 36 to \$numValue\$'. Step 27 is a 'Python script: Open Python script "C:\Hands-On-RPA-with-AA-Sample-Data"'. Step 28 is a 'Python script: Execute function "procSquareRoot" with parameter \$numValue\$'. Step 29 is a 'Python script: Close Python "py_Session"'. Step 30 is a 'Message box \$strReturnValue\$'. Step 31 is a 'Comment "-----"'.

Step	Action
25	Step "Python Scripts"
26	Number: Assign 36 to \$numValue\$
27	Python script: Open Python script "C:\Hands-On-RPA-with-AA-Sample-Data"
28	Python script: Execute function "procSquareRoot" with parameter \$numValue\$
29	Python script: Close Python "py_Session"
30	Message box \$strReturnValue\$
31	Comment "-----"

Running Python scripts

- When you run the bot, the result should be a message box showing a result of 6.0 as this is the square root of 36, as shown in the following screenshot:



Summary

- In this lesson, we looked at external scripts, There will be some exceptional instances when the actions within Automation Anywhere won't perform a specific action.
- An example could be calculating the square root of a value, In order to achieve this, we can still rely on Automation Anywhere to provide a solution.
- An ideal solution would be to use an external script. Whether it's a VBScript or a Python script, Automation Anywhere can handle it.
- You learned how to run scripts as well as how to pass parameters and receive return values.

Managing Errors



Managing Errors

- In this lesson, we will be using the following packages:

 CSV/TXT	 Loop
 Comment	 Number
 Error handler	 Step
 If	 String
 Log To File	

Managing Errors

In this lesson, we will cover the following:

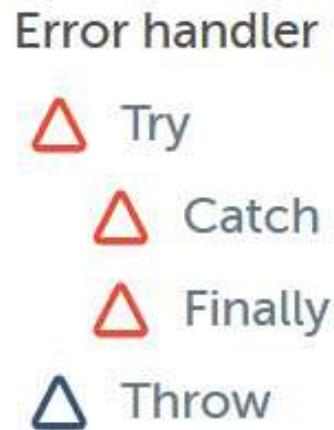
- Error handling
- Understanding Try, Catch, Finally, and Throw actions
- Building an error-handling routine

Error handling

- Pretty much all development platforms will have some sort of error handling functions.
- Having error handling in your bots is crucial when building resilient automation.
- The purpose of managing any errors or exceptions is to keep your bot processing.
- As an example, if your bot is processing a large file unattended and there is an error within the first few records, your bot will stop processing and you won't be aware of this until you next check your bot.

Error handling

- Automation Anywhere comes with an Error handler package.
- This package has four actions, which are designed to help build a robust error-handling routine:



Understanding Try, Catch, Finally, and Throw actions

- The actions used for handling errors are all designed to manage different aspects of each error.
- We know a bot runs a sequence of actions. Each bot can have multiple error-handling routines.
- If there is a particular part of your bot that is vulnerable to an error, you would wrap those actions within a Try action.

Building an error-handling routine

	A	B	C
1	Segment	Product	Price
2	Enterprise	Amarilla	125
3	Enterprise	Carretera	50
4	Enterprise	Montana	100
5	Enterprise	Paseo	15
6	Enterprise	Velo	350
7	Enterprise	VTT	40
8	Government	Amarilla	125

Building an error-handling routine

1	Comment -----	
2	Comment ----- create new csv output file	
3	Comment ----- open products csv file and read each row	
4	Step calculate new price and update file	
5	Comment ----- close products csv file	
6	Comment -----	

Building an error-handling routine

- Create a Record type variable called recProduct.
- The variables should look as shown in the following screenshot:

numNewPrice

numPrice

recProduct

strNewPrice

Building an error-handling routine

Log to file

Logs any text into a file.

File path

„ C:\Hands-On-RPA-with-AA-Sample-Data\ .UpdatedProducts.csv

Enter text to log

„ Segment,Product,Price

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

ANSI

Building an error-handling routine

CSV/TXT: Open

Opens a CSV/TXT file

Session name

csv_Session



File path

Control Room file

Desktop file

Variable

C:\Hands-On-RPA-with-AA-Sample-Data\

roducts.csv



Browse...

Required extensions: ".csv", ".txt", ".tsv"

Contains header

Delimiter

Comma

Building an error-handling routine

CSV/TXT: Close

Closes CSV/TXT session

Session name



CSV_Session



Building an error-handling routine

- 1 // Comment "-----"
- 2 // Comment "----- create new csv output file"
- 3 Log to file "Segment,Product,Price" to "C:\Hands-On-RPA-with-AA-Sample-Data\C...
- 4 // Comment "----- open products csv file and read each row"
- 5 CSV/TXT: Open "C:\Hands-On-RPA-with-AA-Sample-Data\"_Products.csv"
- 6 Step "calculate new price and update file"
- 7 // Comment "----- close products csv file"
- 8 CSV/TXT: Close csv/txt "csv_Session"
- 9 // Comment "-----"

Building an error-handling routine

Loop

Repeats the actions in a loop until a break

Loop Type

Iterator

Iterator

For each row in CSV/TXT

Iterator for each row in CSV/TXT

Session name

CSV_Session

(x)

Assign the current row to this variable

recProduct - Record

(x)

(x)

Building an error-handling routine

String: To number

Converts a string to a number

Enter the string

99 \$recProduct[2]\$

String entered must be a valid number

Assign the output to variable

numPrice - Number



(x)
+

Building an error-handling routine

Number: Assign

Assigns user specified number to number variable

Select the source string variable/ value

\$numPrice\$ * 0.9



Specify value to assign to number:

Select the destination number variable

numNewPrice - Number



Building an error-handling routine

Number: To string

Converts a user specified number to a string

Enter a number

\$numNewPrice\$ (x)

Specify number to convert to string e.g. 35

Enter number of digits after decimal (number format)

2 (x)

e.g for number 35.265, enter the number of digits after decimal as 3

Assign the output to variable

strNewPrice - String ▼ (x) +

Building an error-handling routine

Log to file

Logs any text into a file

File path

C:\Hands-On-RPA-with-AA-Sample-Data\ _UpdatedProducts.csv

Enter text to log

\$recProduct[0]\$, \$recProduct[1]\$, \$strNewPrice\$

Append timestamp

When logging

Append to existing log file

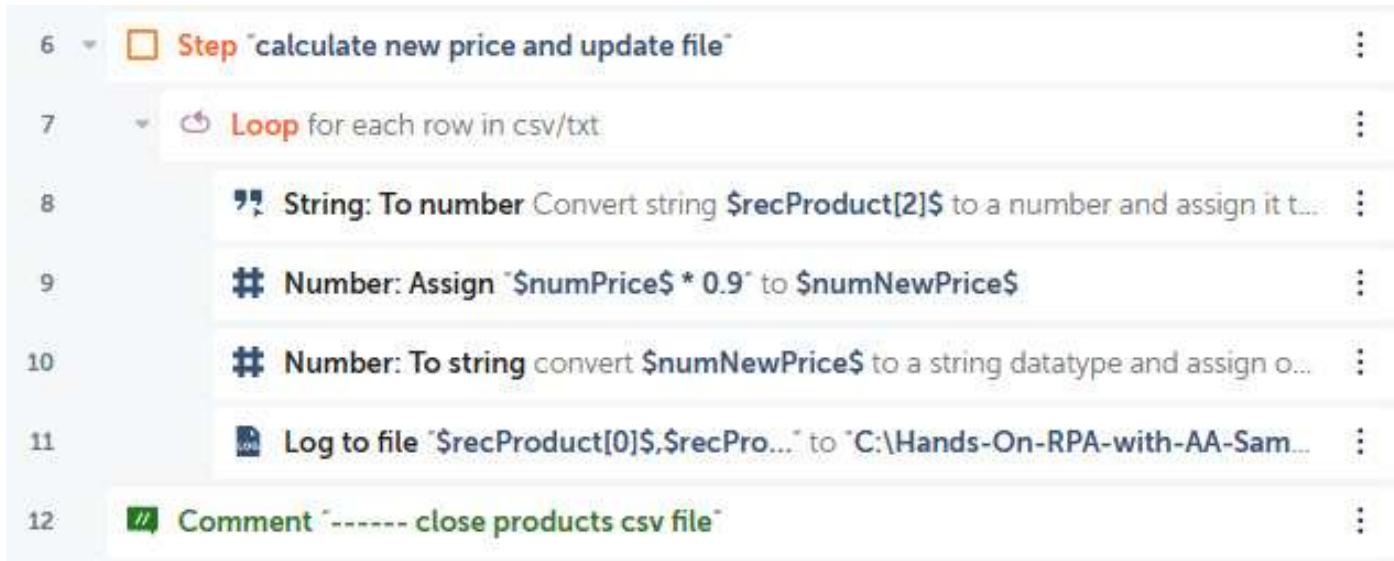
Overwrite existing log file

Encoding

ANSI

Building an error-handling routine

- Click on Save.
- The development interface for this section should look like this:



The screenshot shows a sequence of steps in a RPA development environment:

- Step 6: Step "calculate new price and update file"
- Step 7: Loop for each row in csv/txt
 - String: To number Convert string \$recProduct[2]\$ to a number and assign it to \$numPrice\$
 - Number: Assign "\$numPrice\$ * 0.9" to \$numNewPrice\$
 - Number: To string convert \$numNewPrice\$ to a string datatype and assign it to \$recProduct[2]\$
 - Log to file '\$recProduct[0]\$,\$recPro...' to 'C:\Hands-On-RPA-with-AA-Sam...'
- Step 12: Comment "----- close products csv file"

Building an error-handling routine

	A	B	C
1	Segment	Product	Price
2	Enterprise	Amarilla	112.5
3	Enterprise	Carretera	45
4	Enterprise	Montana	90
5	Enterprise	Paseo	13.5
6	Enterprise	Velo	315
7	Enterprise	VTT	36
8	Government	Amarilla	112.5

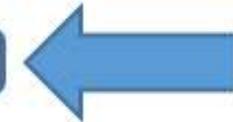
Modifying the input file and introducing an error

Original

Segment	Product	Price
Enterprise	Amarilla	125
Enterprise	Carretera	50
Enterprise	Montana	100
Enterprise	Paseo	15
Enterprise	Velo	350
Enterprise	VTT	40
Government	Amarilla	125
Government	Carretera	50
Government	Montana	100

Updated

Segment	Product	Price
Enterprise	Amarilla	125
Enterprise	Carretera	50
Enterprise	Montana	100
Enterprise	Paseo	
Enterprise	Velo	350
Enterprise	VTT	40
Government	Amarilla	125
Government	Carretera	50
Government	Montana	100



Modifying the input file and introducing an error



There was a problem at line 8

This may be due to the following reason:

The input entered in 'sourceString' is incorrect.

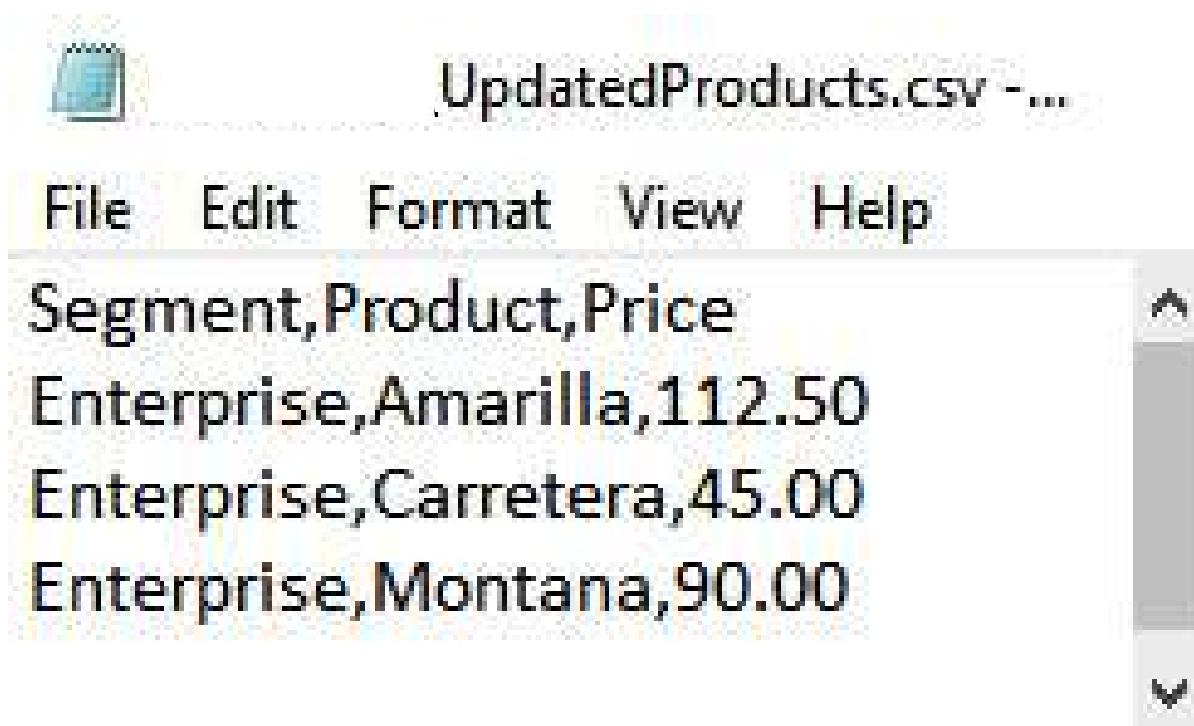
To continue, edit the bot and fix the error. Then, try again.

If you continue to see this message, please contact your system administrator.

Code: `bot.execution.error`

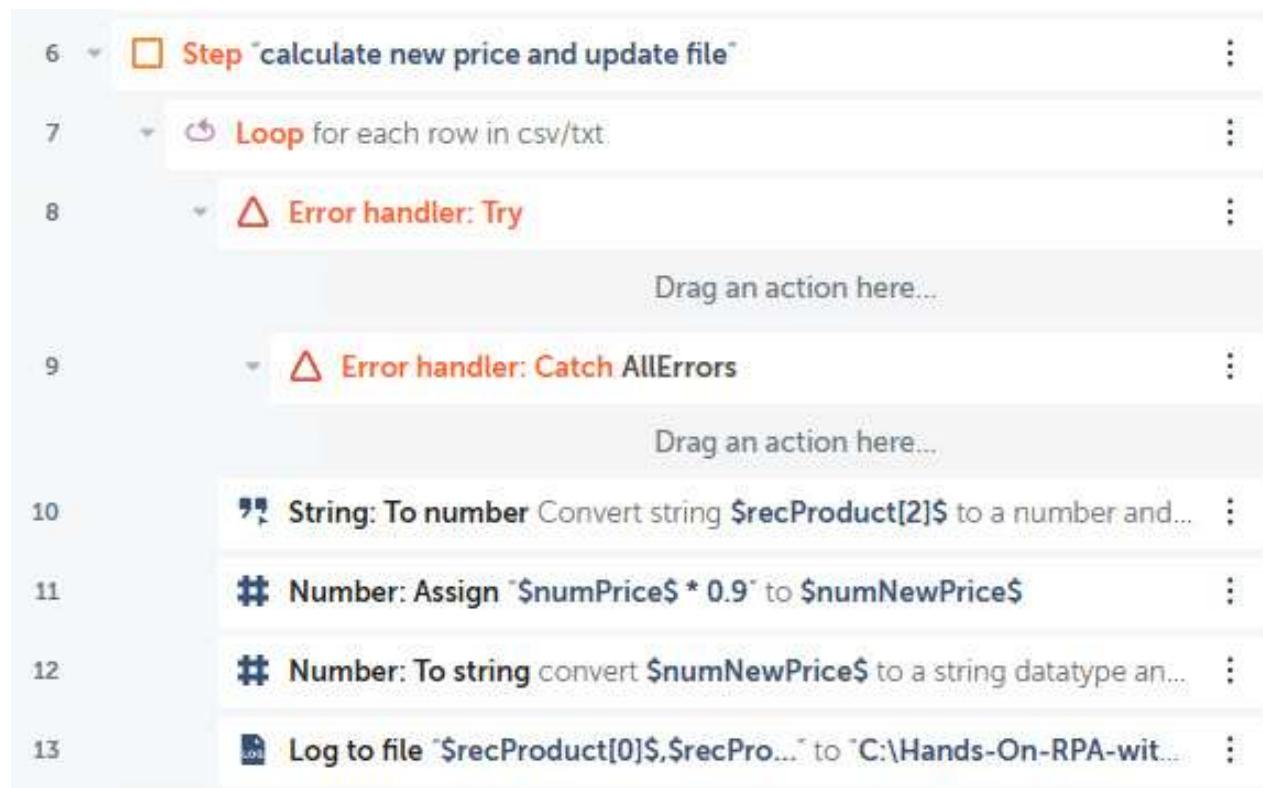
Close

Modifying the input file and introducing an error



Segment	Product	Price
Enterprise	Amarilla	112.50
Enterprise	Carretera	45.00
Enterprise	Montana	90.00

Modifying the input file and introducing an error



Modifying the input file and introducing an error

```
8   △ Error handler: Try
9     " String: To number Convert string $recProduct[2]$ to a nu...
10    # Number: Assign "$numPrice$ * 0.9" to $numNewPrice$
11    # Number: To string convert $numNewPrice$ to a string dat...
12    Log to file "$recProduct[0]$,$recPro..." to "C:\Hands-On-...
13   △ Error handler: Catch AllErrors
14   # Comment "----- close products csv file"
```

Modifying the input file and introducing an error

Error handler: Catch

Run a sequence of commands if the commands in Try fail with an exception.

Exception

AllErrors ▼

Indicates a generic bot exception when a bot is run
Code: AA-BotException-111-111

Assign exception message to (optional)

strErrDesc - String ▼ (x)

Assign line number to (optional)

numErrLine - Number ▼ (x)

Modifying the input file and introducing an error

Number: To string

Converts a user specified number to a string

Enter a number

\$numErrLine\$

(x)

Specify number to convert to string e.g. 35.

Enter number of digits after decimal (number format)

0

(x)

e.g for number 35.265, enter the number of digits after decimal as
3

Assign the output to variable

strErrLine - String

▼

(x)

Modifying the input file and introducing an error

Log to file

Logs any text into a file

File path

„ C:\Hands-On-RPA-with-AA-Sample-Data\

_ErrorLog.csv



Browse...

Enter text to log

„ Desc: \$strErrDesc\$, Line: \$strErrLine\$, (Record: \$recProduct[0]\$,\$recProduct[1]\$,\$recProduct[2]\$) (x)

Append timestamp

When logging

Append to existing log file

Overwrite existing log file

Encoding

ANSI

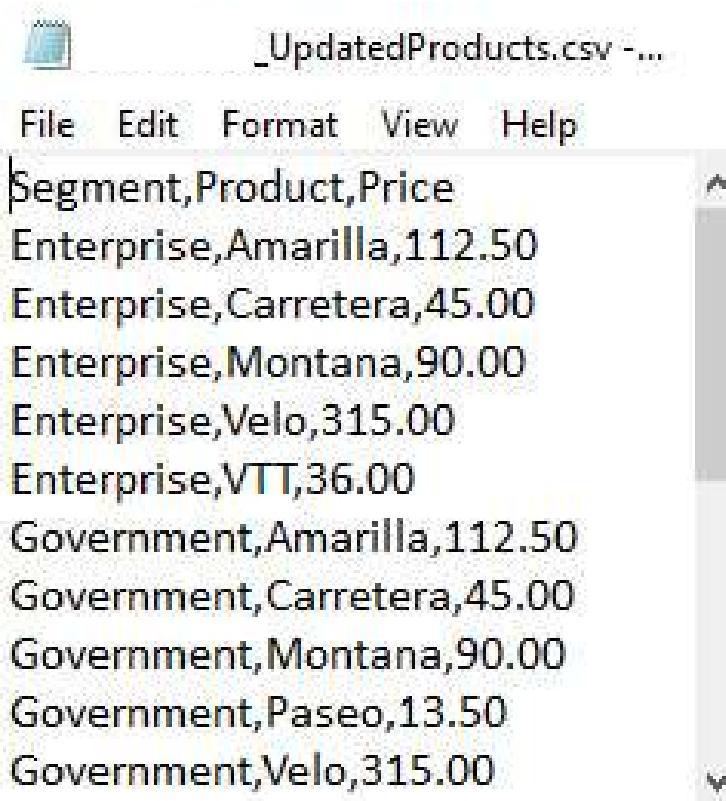


Modifying the input file and introducing an error

- Click on Save.
- The development interface for this section should look like this:

```
13      ▲ Error handler: Catch AllErrors
14      # Number: To string convert $numErrLine$ to a string dataty...
15      □ Log to file "Desc: $strErrDesc$, Lin..." to "C:\Hands-On-RP...
```

Modifying the input file and introducing an error

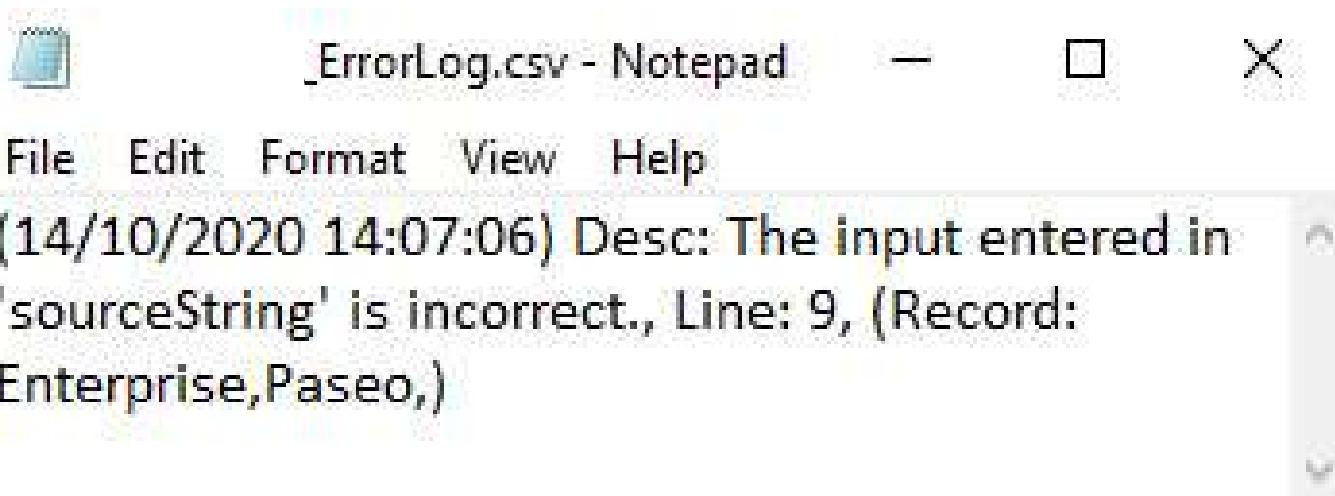


The screenshot shows a CSV file titled '_UpdatedProducts.csv' in a window. The window has a menu bar with File, Edit, Format, View, and Help. The main area displays the following data:

Segment	Product	Price
Enterprise	Amarilla	112.50
Enterprise	Carretera	45.00
Enterprise	Montana	90.00
Enterprise	Velo	315.00
Enterprise	VTT	36.00
Government	Amarilla	112.50
Government	Carretera	45.00
Government	Montana	90.00
Government	Paseo	13.50
Government	Velo	315.00

Modifying the input file and introducing an error

- There will also be an error log file generated, lesson18_ErrorLog.csv.
- This file will have the details of the error and the invalid record:



Summary

- This lesson has been all about building robust, resilient bots that can get up and continue even when they fall over.
- We have explored the Error handler package and the actions available, The walk-through provided the practical skills to actually build your own error-handling routine.
- Having a good error-handling routine is key to the success of your bots, No matter how great the functionality of your bot, one thing we cannot guarantee is the quality of any source inputs.