**RPA Week 7 Day 1**

**Robot Control – Orchestrator**

UiPath’s web interface, called Orchestrator, allows users to monitor, maintain and designate robots running workflows created by developers. The features discussed in this document cover:

* **Provisioning** - creates and maintains the connection with the robots.
* **Deployment** - ensures the delivery of the workflows for execution, either immediately or using schedules.
* **Configuration** - enables the creation, configuration and maintenance of groups of robots and the execution of tasks..
* **Queues** - the data that needs to be processed is broken down to indivisible operations called transactions. Queues can store any number of transactions and they facilitate their distribution, execution and monitoring.
* **Monitoring** - keeps track of robot identification data and maintains user permissions.
* **Logging** - stores and indexes the logs to an SQL database and/or ElasticSearch.
* **Inter-connectivity** - acts as the centralized point of communication for 3rd party solutions or applications, and can be used for storing activity packages, libraries and other assets (such as credentials).

Orchestrator has other benefits too, such as version control, transaction processing and storing assets.

**Robots**

With regards to RPA, a robot is considered an execution agent of different workflows built within Studio. Whilst all robot’s core function is to execute a workflow, there are several differing types of robots each with their own capabilities.

**Types of Robots**

* **Attended Robot** - This type of Robot is triggered by user events, and operates alongside a human, on the same workstation. Attended Robots are used with Orchestrator for a centralized process deployment and logging medium.
* **Unattended Robot** - Robots run unattended in virtual environments and execute any number of processes. On top of the Attended Robot capabilities, Orchestrator covers execution, monitoring, scheduling and providing support for work queues.
* **Development/ Non-Production Robot** - Has the features of an Unattended Robot, but it should be used only to connect Studio to Orchestrator, for development purposes.

**Attended Vs Unattended**

Understanding whether a robot should be attended or unattended depends on whether the user needs to interact with the robot or not. If the robot requires the user to input information, or aid the robot in making decisions based on information that must be inferred or interpreted and has no steadfast rules, then an attended robot will be needed. If the robot requires zero interaction or input from the user, then an unattended robot may be used.

Please note, that licensing will restrict the number of attended and unattended robots that can be used. With the Community Edition, the user will be allowed to use 2 attended robots, and 0 unattended robots.

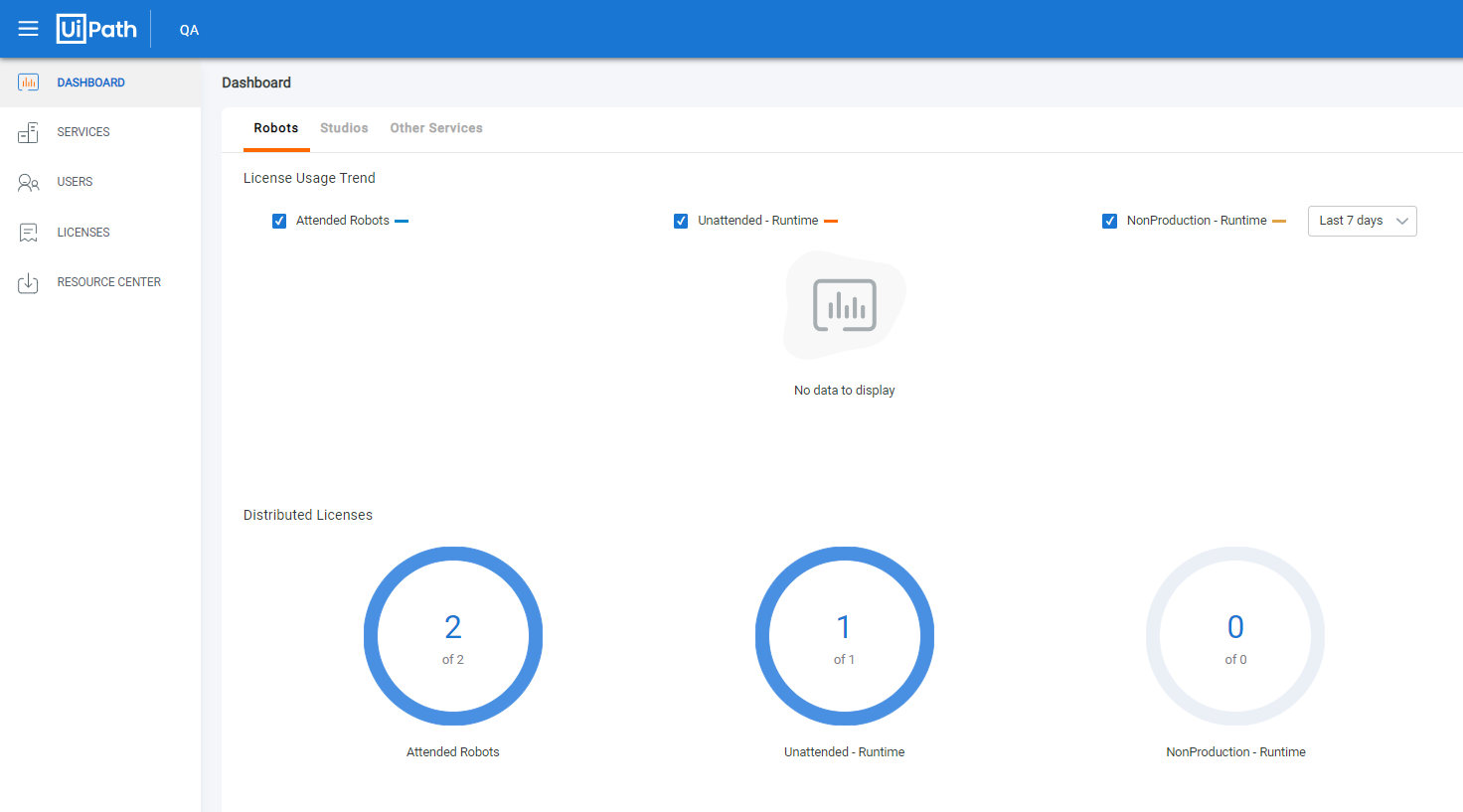
**Kinds of Robots**

Beyond the 3 types of robots, each one of them can be of either a Standard Robot, or a Floating Robot.

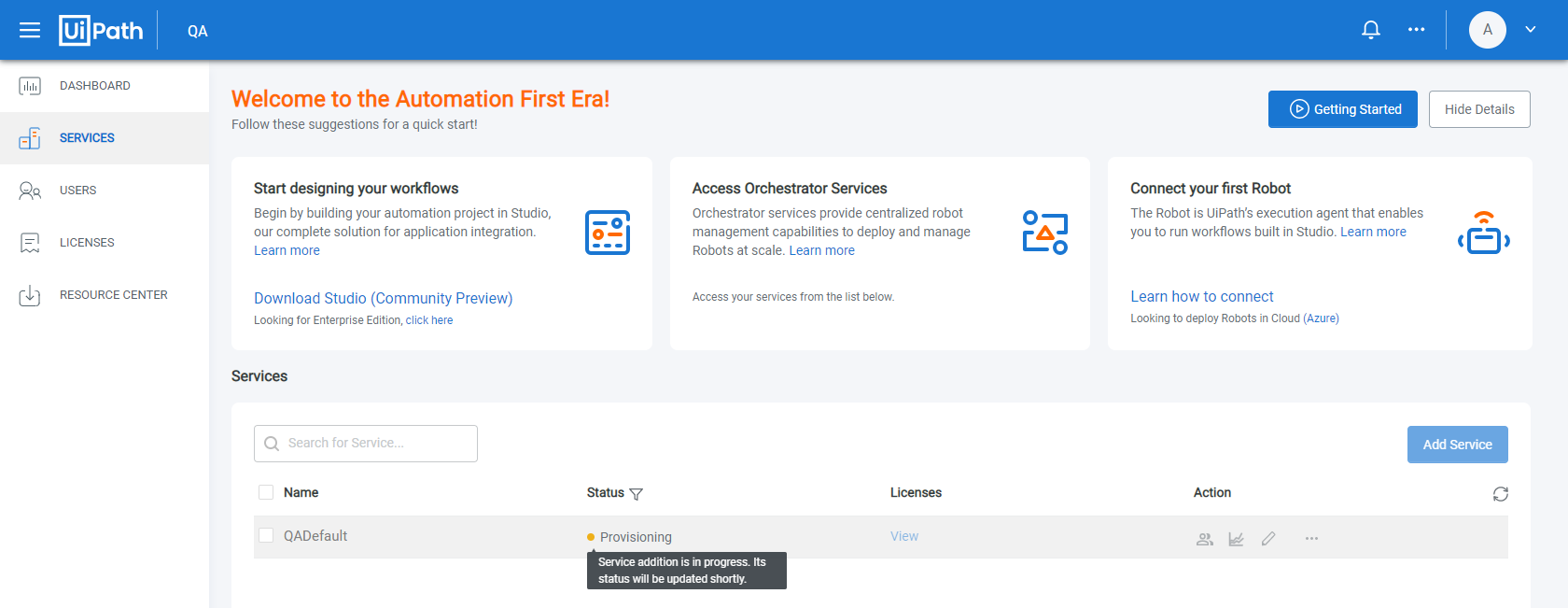
* **Standard Robot** - Works on a single Standard Machine only. It is a good choice when the machine on which the robot runs is known and will never change.
* **Floating Robot** - Works on any machine defined in Orchestrator. It is a good choice when human users work in shifts on the same machine or on different machines, and when virtual machines are being regenerated often.

**Provisioning**

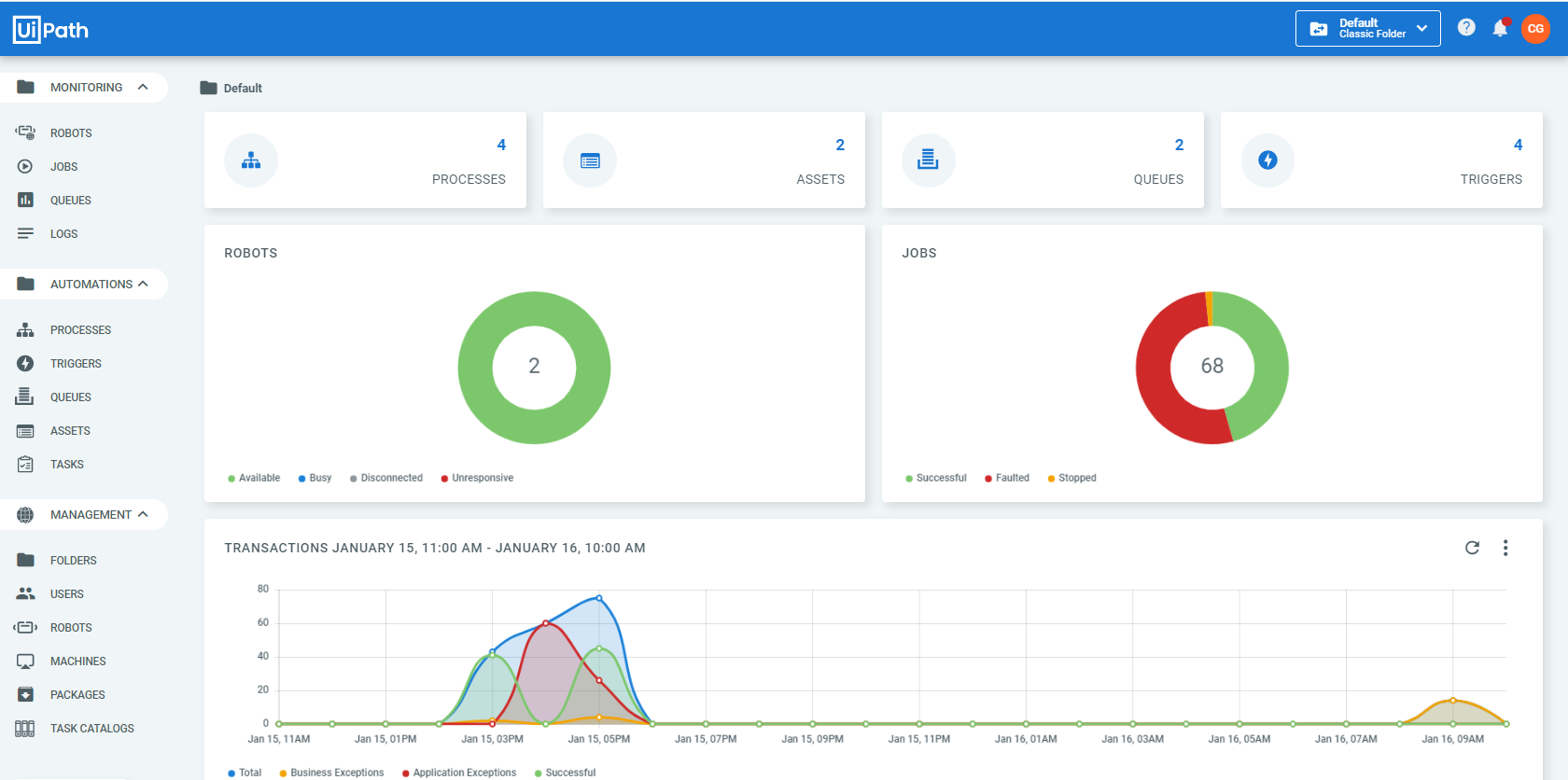
For robots to be able to execute workflows, they must be provisioned in Orchestrator with the machine that will be hosting the execution. When the user first accesses the platform.uipath.com site, they will be presented with a Dashboard.



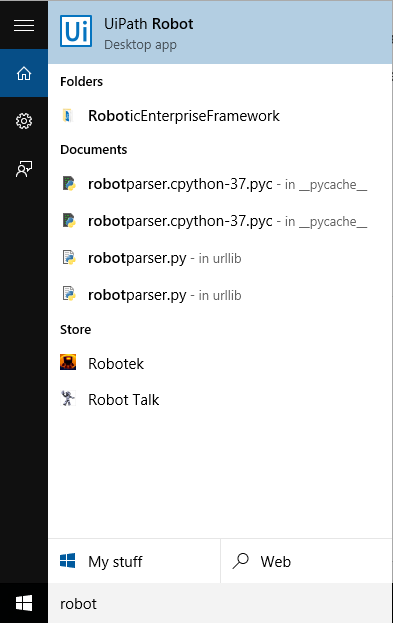
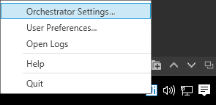
Navigating to Services on the left hand panel will navigate the user to a new interface, displaying the Services that the user has set up – there should be 1 there by default, and is the maximum number of services that can be set up on a Community Edition account.

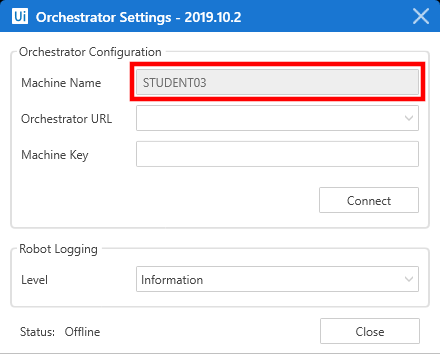


In the image above, the Service is called QADefault – clicking the name will direct the user to a new tab within their browser of choice, displaying the overview of the Orchestrator.

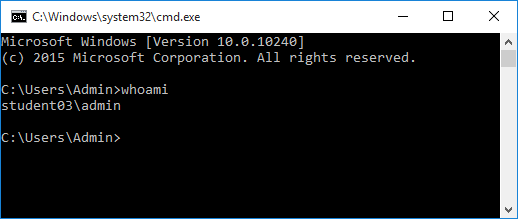


To provision the robot within UiPath, some information of the target machine needs to be acquired. First, the machine name must be captured. This can be achieved by accessing the system tray UiPath icon via the context menu (right click), and navigating to Orchestrator Settings – the machine name will be displayed. If the tray icon is not present, the user must start the Robot by typing Robot in to the start menu interface, and starting the application.





Next, the username of the account must be acquired from the target machine. This can be achieved by opening the Command Line Interface (CLI) by typing ‘command prompt’ in to start menu interface. Within the CLI using the command ‘whoami’ will yield the information required.

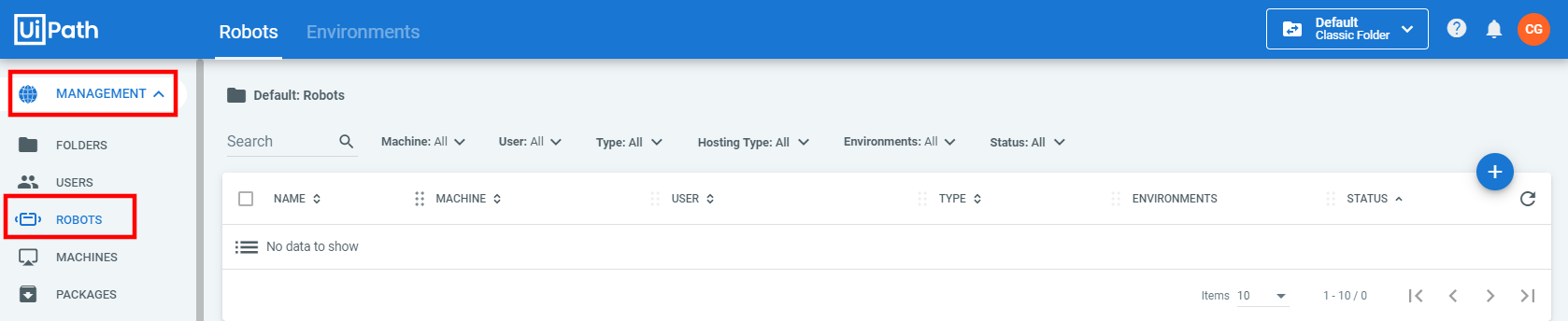


In this example, the information gethered is:

Machine name - STUDENT03

Username - student03\admin

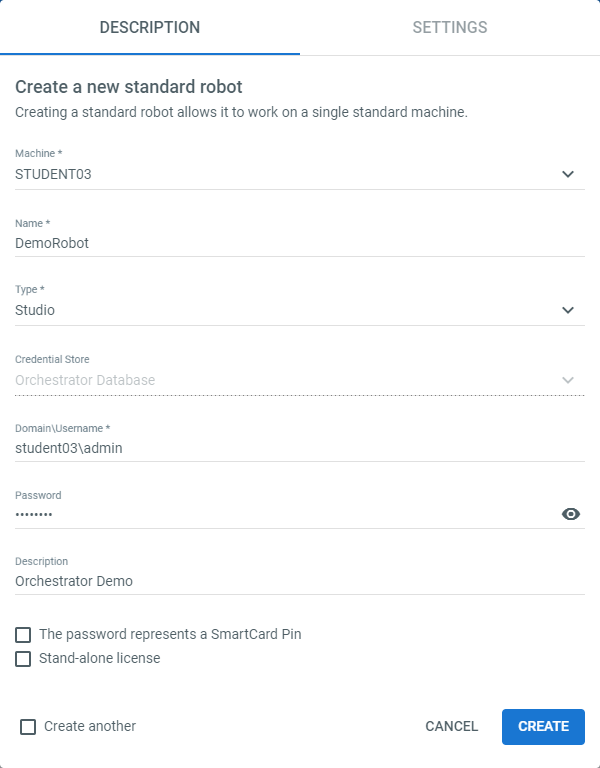
With this information, the user is able to provision the machine by navigating to Robots under the Management heading.



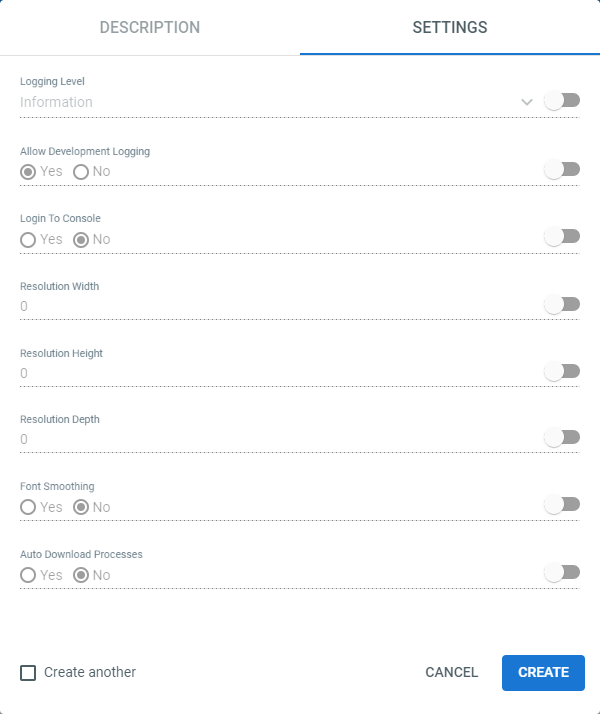
Clicking the plus (+) button, and selecting the Standard Machine button.



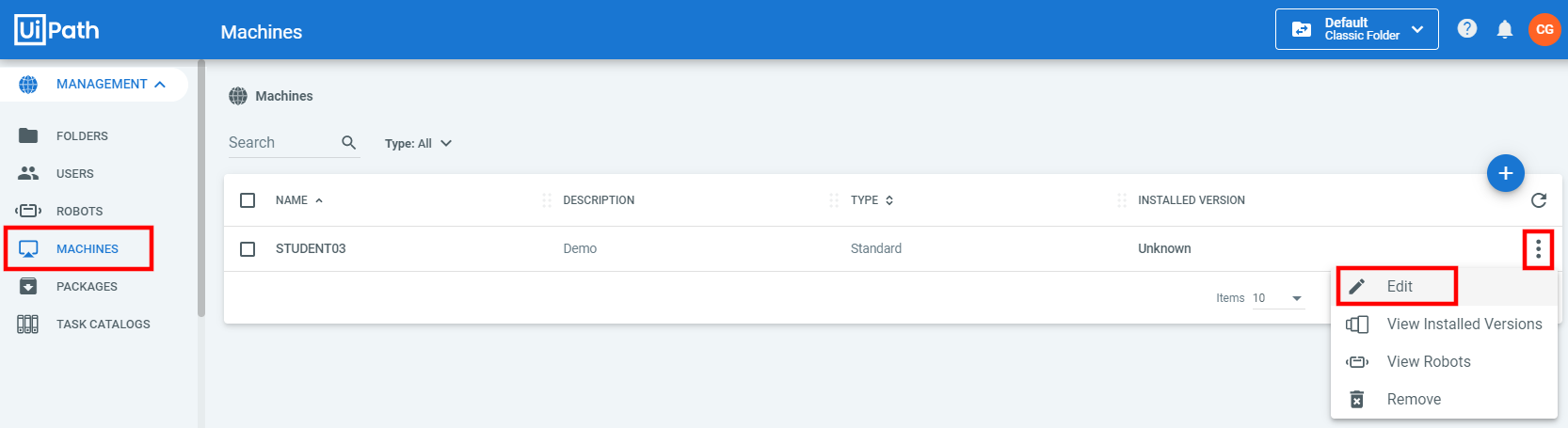
The user will be presented with a new interface, requesting the gathered information for the machine being provisioned.

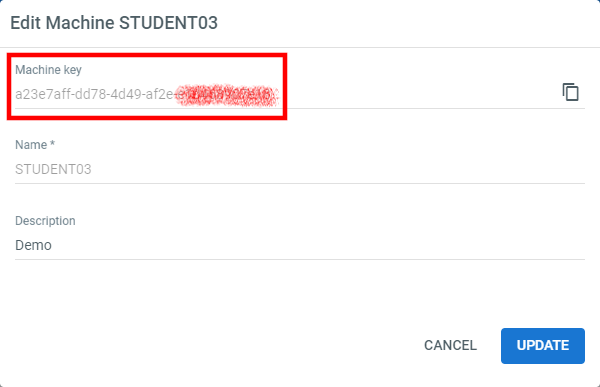


As well as allow the user to configure the settings of the target robot.

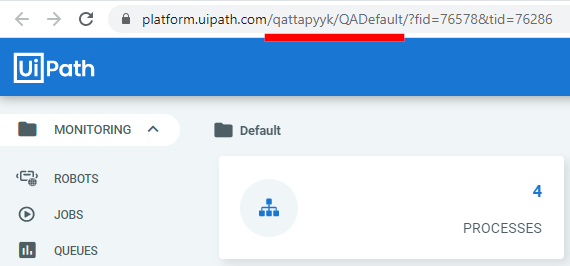


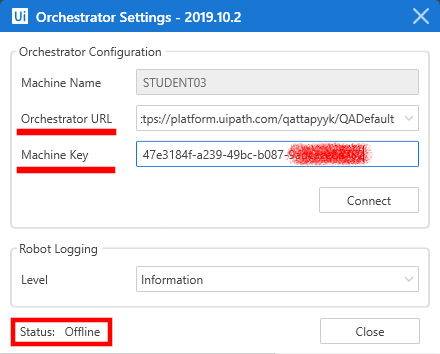
Once the machine has been set up within Orchestrator, the uniquely identifying Key must be provided to the target machine. Selecting the elipse button (…) on the row of the newly made Machine, will unveil more options. Selecting Edit will present the user with a new interface detailing the Machine, and also providing the Key.



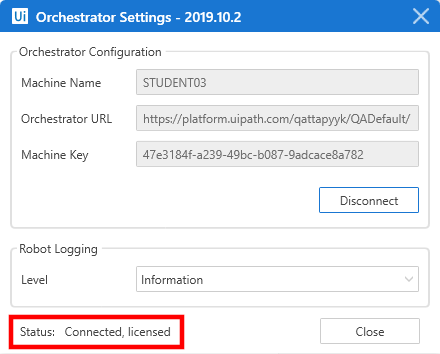


Back in the Orchestrator Settings on the target machine, the user can now supply the Key. The user must also supply the Orchestrator URL, which begins with ‘https://platform.uipath.com/’ followed by the allocated name space for the user’s Orchestrator panel, followed by the name of the target service. In this example, the full path will be ‘https://platform.uipath.com/qattapyyk/QADefault’.

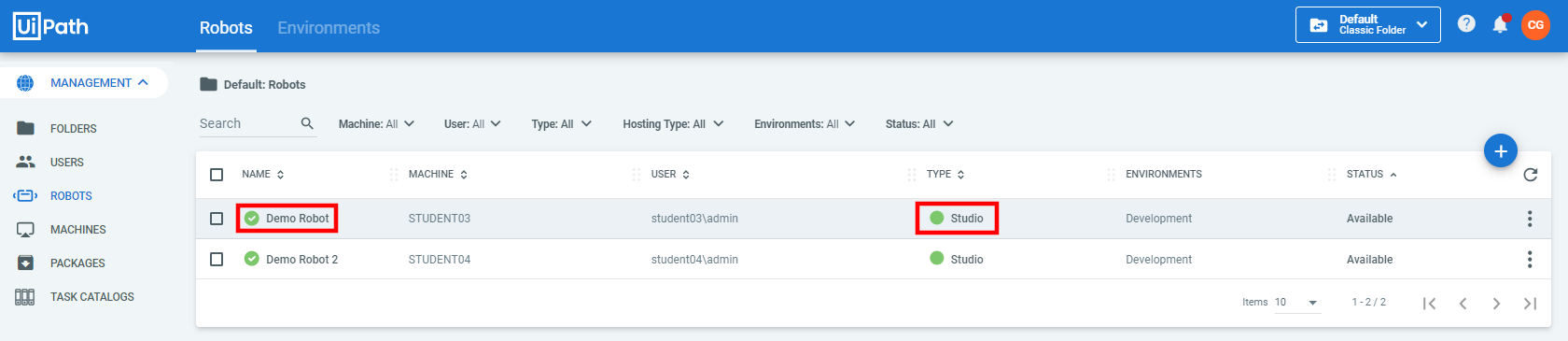




Note that the Status of the robot will always be displayed here. Once configured, clicking Connect will change the Status to ‘Connected, Licensed’.



Once connected, navigating to Robots under the Management heading, should present the user with a green tick next to the name, and a green dot next to the Type of machine it is. If the machine disconnects, or is faulted, the dot will be a different colour, or not present entirely.

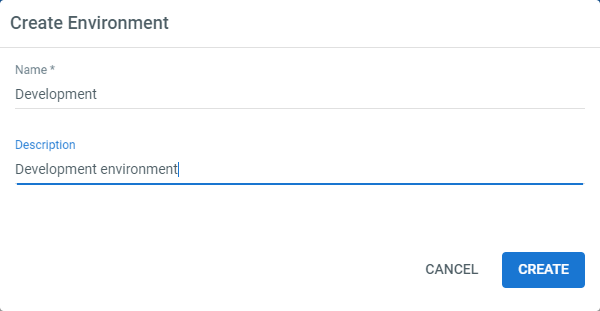


Now that the robot and machine have been set up and configured, an Environment must be made for the robots to run within. An environment is simply a group of robots defined to the organisations requirements.

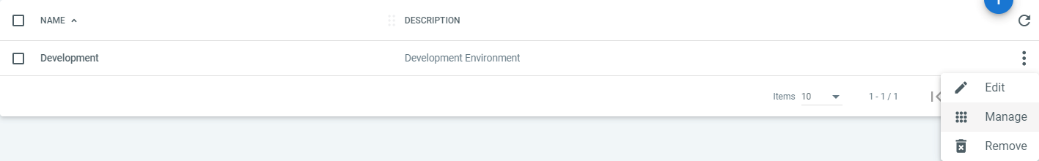
To set up an Environment, from within Robots under the Management heading, there is a a tab at the top labeled Environments, clicking this will change the display of the interface.



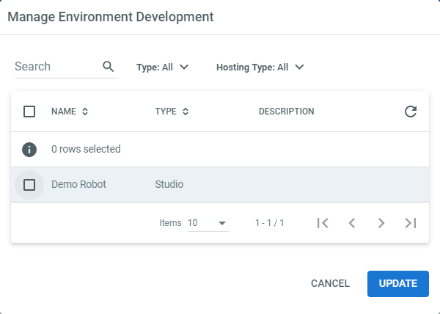
Clicking the Plus button will again present a new input interface. Completing the requested details with ‘Development’ with an accompanying description, then clicking Create will establish the Environment.



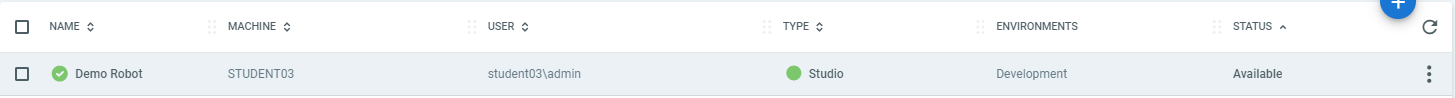
From the elipse button (…), selecting Manage will allow the user to associate a robot with the environment. A robot may only be associated with a single environment.



Selecting the check box next to the name of the robot, the clicking Update will complete the robot will then be associated with the environment.

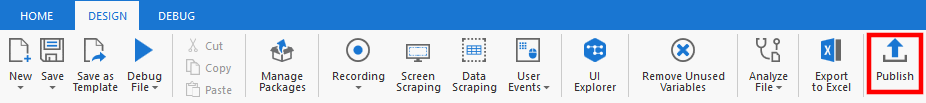


If everything is configured and connected properly, back on the Robots tab, there should be present a green tick next to name, a green dot next to type, and the name of the environment displayed. This will indicate the robot and machien are ready to execute automation pieces.



**Publishing a Workflow**

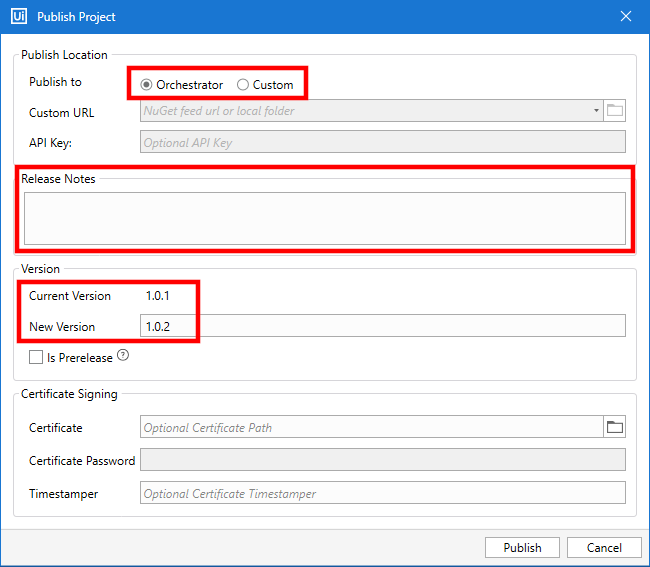
To allow a workflow to be executed at a scheduled point in time, or when certain conditions are triggered, the workflow must be published to the Orchestrator. To achieve this, the user must use the Publish button within Stuio, under the Design ribbon.



The user will be presented with a dialogue box requesting information from the user. The desired information is the Publish to location. The options are Orchestrator or Custom. Custom will publish the project to a supplied folder location, where as Orchestrator will push the workflow to the cloud, accessible from within Orchestrator web interface.

Ensure that a Release Note has been supplied so that developers or maintainers inheriting the workflow are able to understand what the workflow is, or to understand any changes that have been made.

Finally, a Version number must be supplied. The version number will automatically increase by 0.0.1 if the workflow has been published once before.



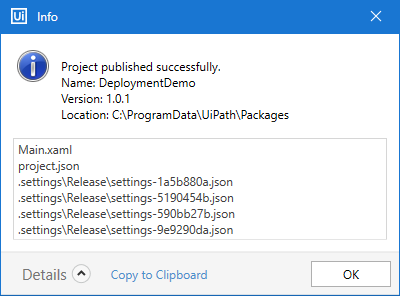
A general note on version numbering (x.y.z):

X - Major feature update or adding new features and functionality including distribution to new platform(s)

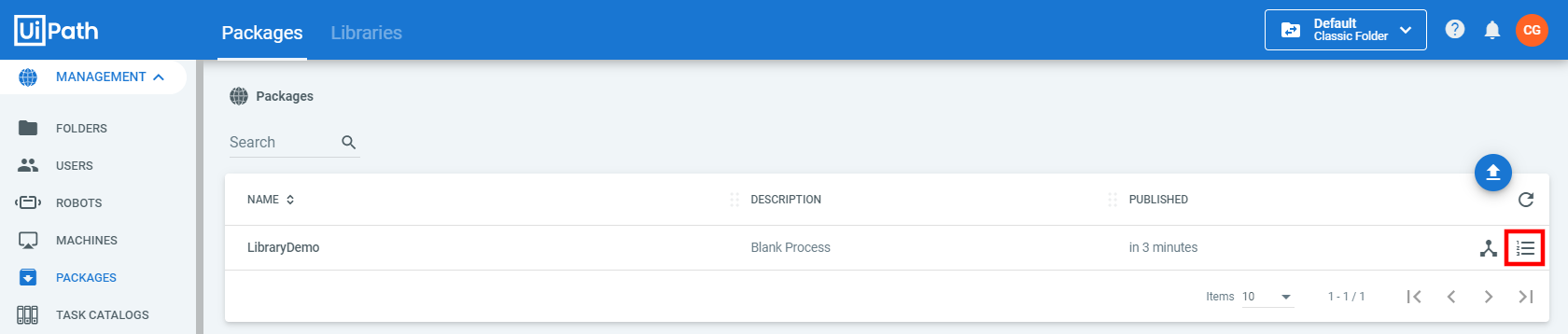
Y - Minor feature update or new small feature and funtionality

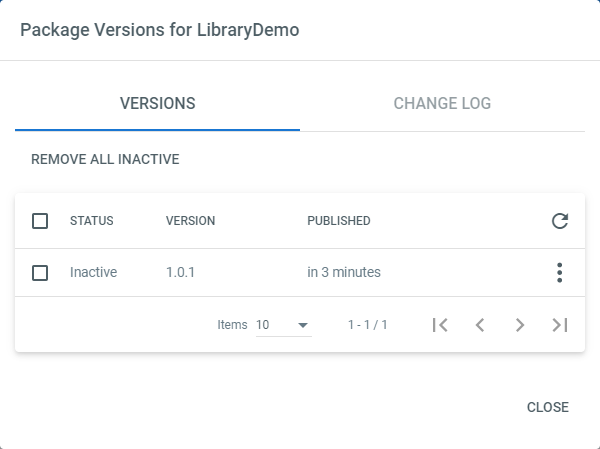
Z - Bug fixes, UI tweaks and data refreshes

Once the workflow has been publish, a confirmation screen will be presented.

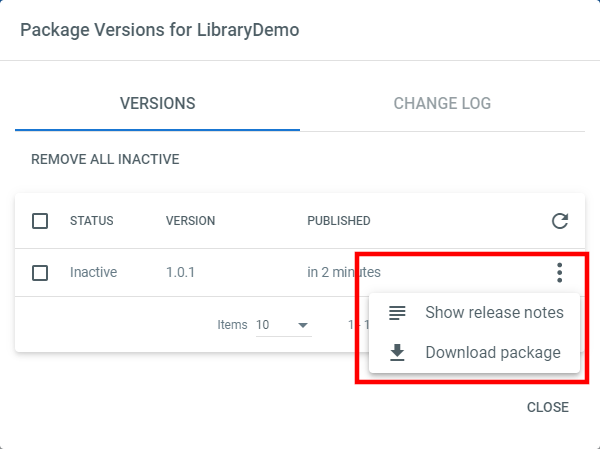


Navigating to Packages under the Management heading within Orchestrator, the uploaded workflow package will be visible. Clickin the ‘1-2-3-‘ button on the far right of the package will display a new window.

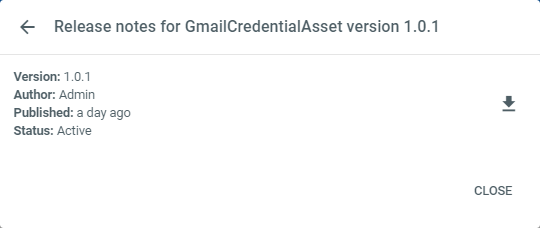




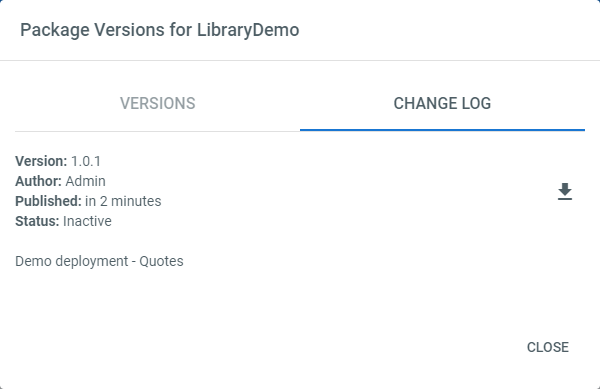
Clicking the elipse button will allow the user to either download the workflow, or to show the release notes that were previously added.



The release notes.

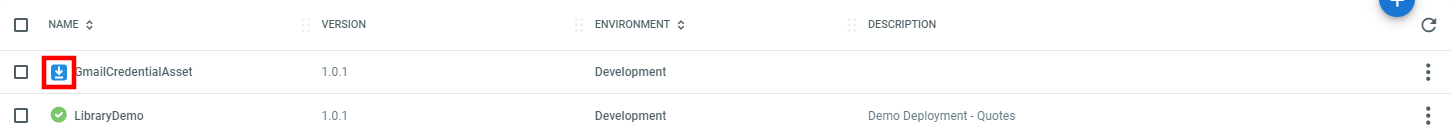


And the changelog.

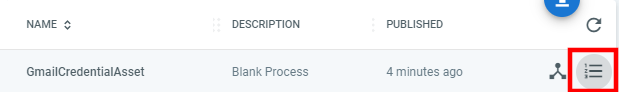


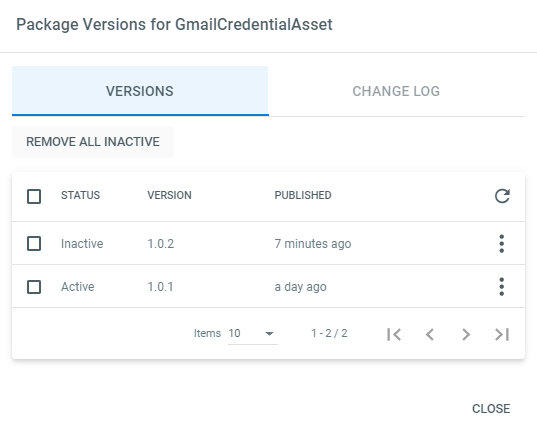
As can be seen above, the status is set to Inactive. This states that the package (workflows) have not been deployed to an environment, and will allow the user to be able to delete the workflows whilst in this state. If the status is set to Active, it has been deployed to an environment.

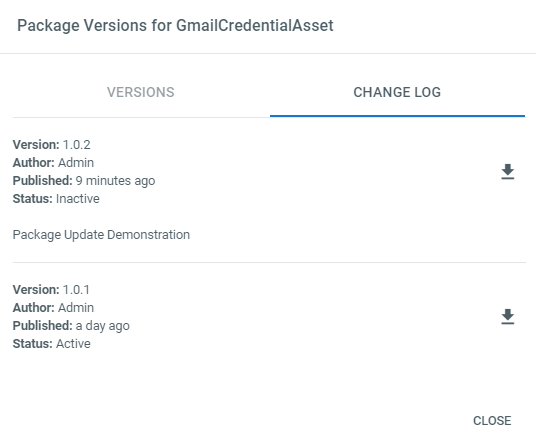
If the user needs to upload the workflow again due to changes made to it, once the project has been successfully Published from with Studio, the user must navigate to Processes under Automations to observe that an update is available, which is depicted by a blue and white icon.



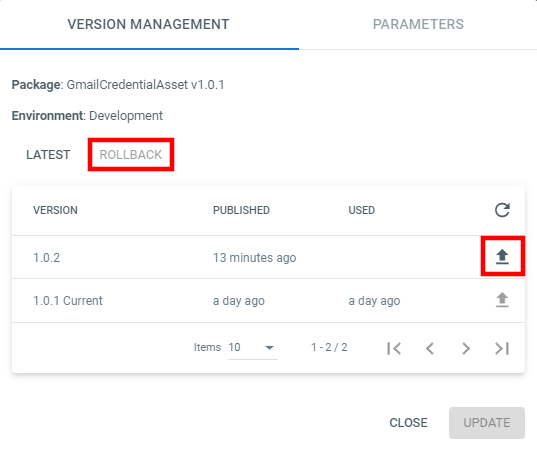
Navigating to Packages under Management and accessing the View Versions button will present the user with all of the version of the package that have been uploaded, as well as their release notes.



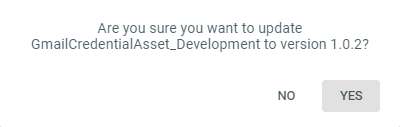




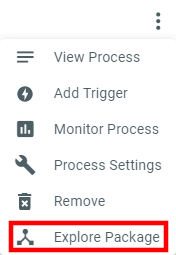
To be able to update the package that is being used by the processes, back in Processes under Automations, selecting the elipse button for the update-able package, and selecting View Process will unveil the Version Management window. From this window the user is able to update the package with the non-greyed out Use button, shaped like an Up Arrow. To undo the update(s) to the package, the Rollback feature can be utilised.

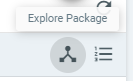


Either way, a confirmation window will appear, preventing the user from accidentaly updating or rolling back packages with mis-clicks.

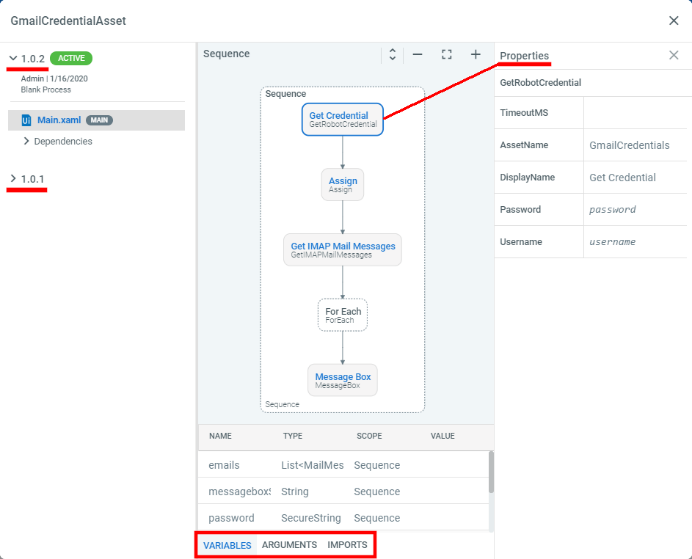


After uploading a workflow to Orchestrator, the user is able to view the workflow in a high level manner, by opening the Explore Package window. This window can be accessed from Processes under Automations, selecting the elipse button, and clicking Explore Package.

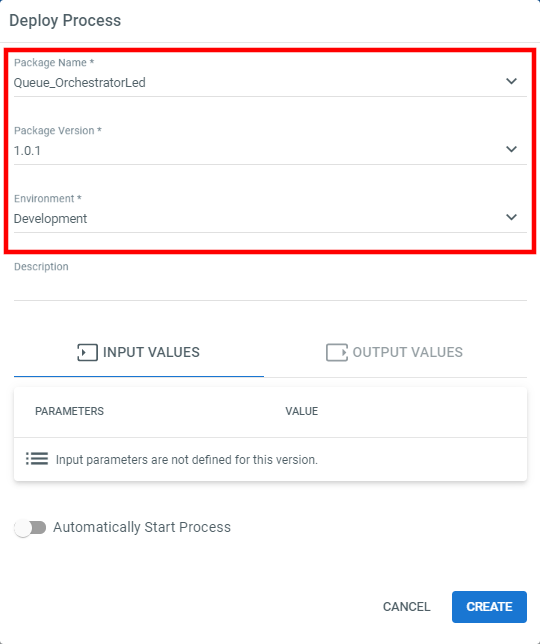
An alternative point of access is from Packages under Management, and selecting the same icon as above, next to the package in question.



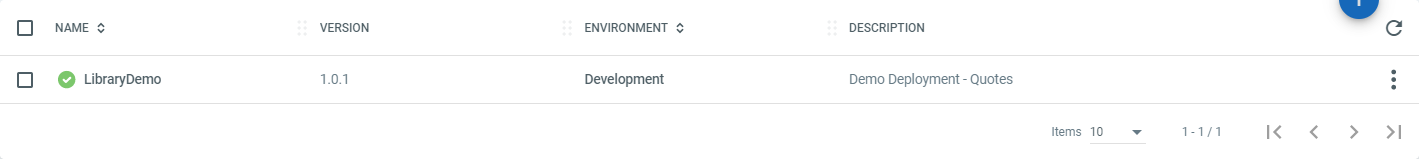
Either of these buttons will present the user with their package, which contains all of the workflows and activities. Selecting an activity will further unveil more information about that specific activity. Variables, Arguments and Imports can also be viewed from here.



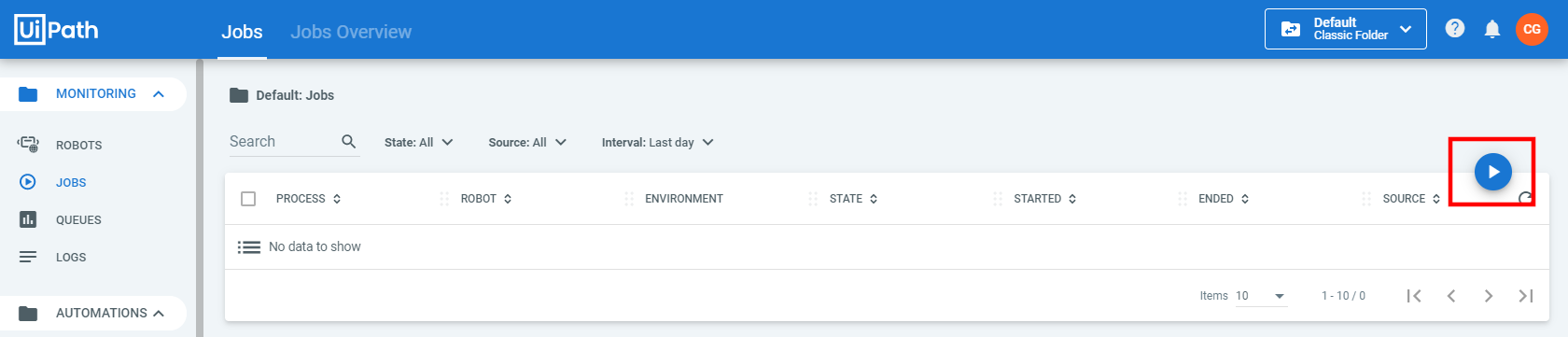
To associate the workflow(s) with an environment, the user must navigate to the Proccesses heading under the Automations title within Orchestrator. Clicking the plus button to create the link, will present the user with a new window, expecting the Package Name, Package Version and Environment to deploy to.



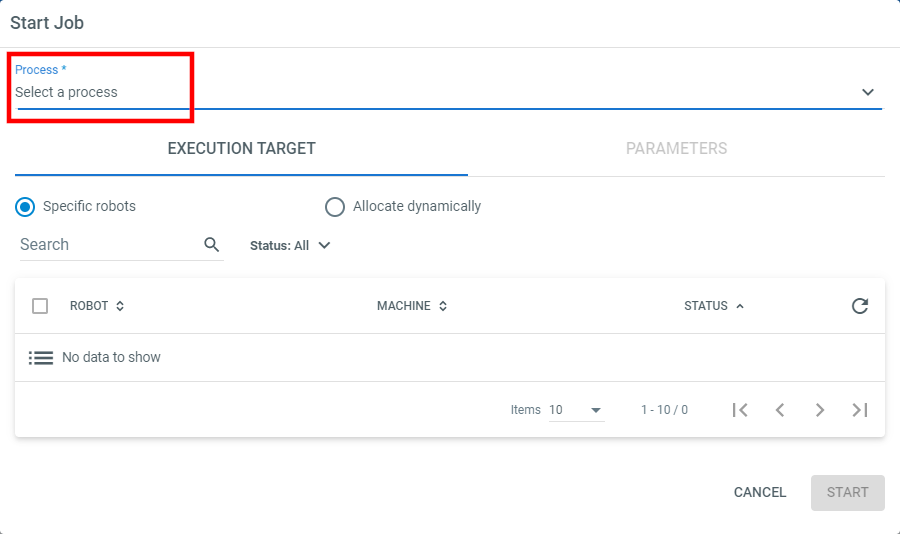
Once created, a new item will be shown in the list of proccesses.

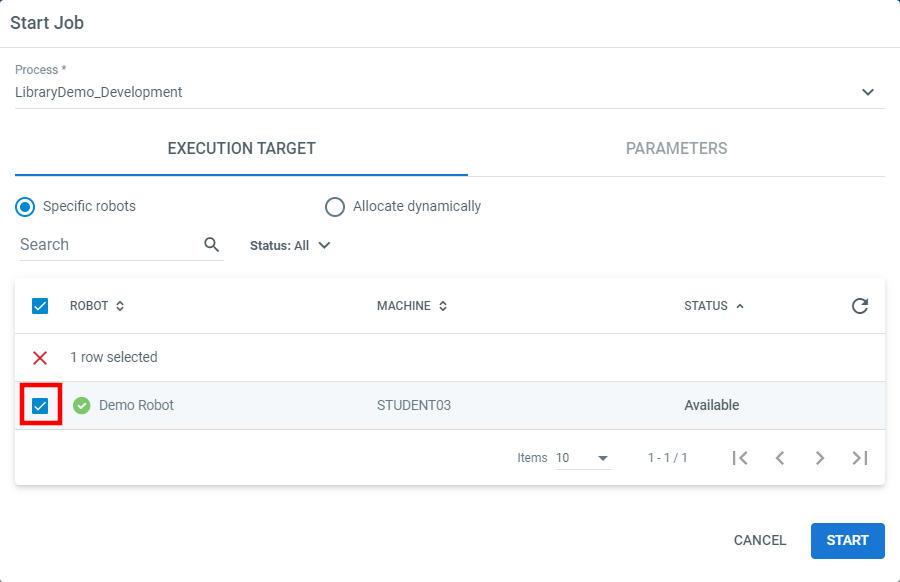


Finally, for the machine to run the process, a Job must be set up in Orchestrator. The Job interface, found under the Monitoring heading, will allow the user to execute a process (workflows) by clicking on the blue Play button in the top right hand corner.

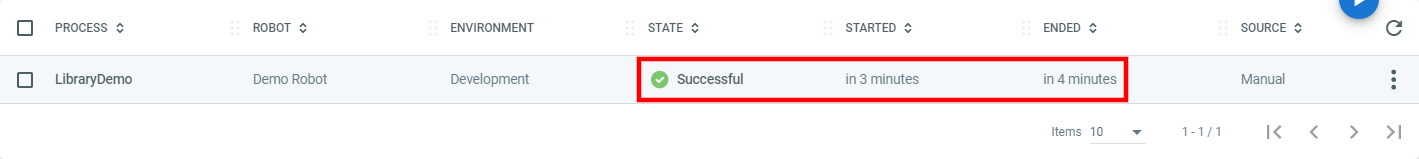


Once a proccess has been selected (the previously associated package and environment), the available robots within that environment will be displayed. From here the user can specify whether to run on specific robots, or to allocate dynamically. Dynamic allocation will allow the robot to choose the first available machine from within that environment. This can be useful when it is unknown when robot(s) will finish due to varying amount of processing to achieve.

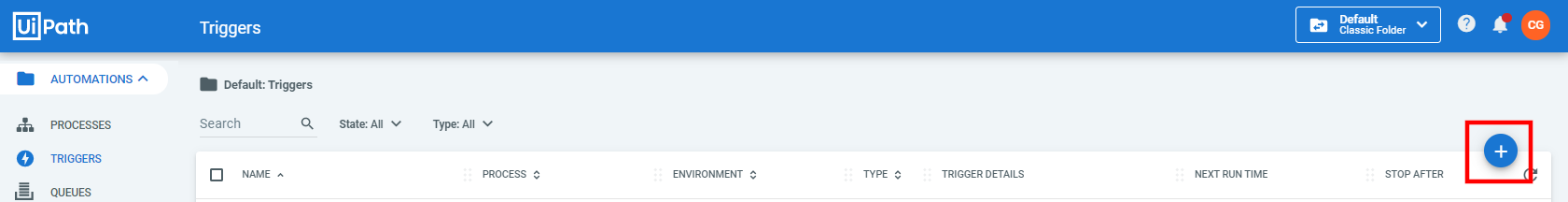


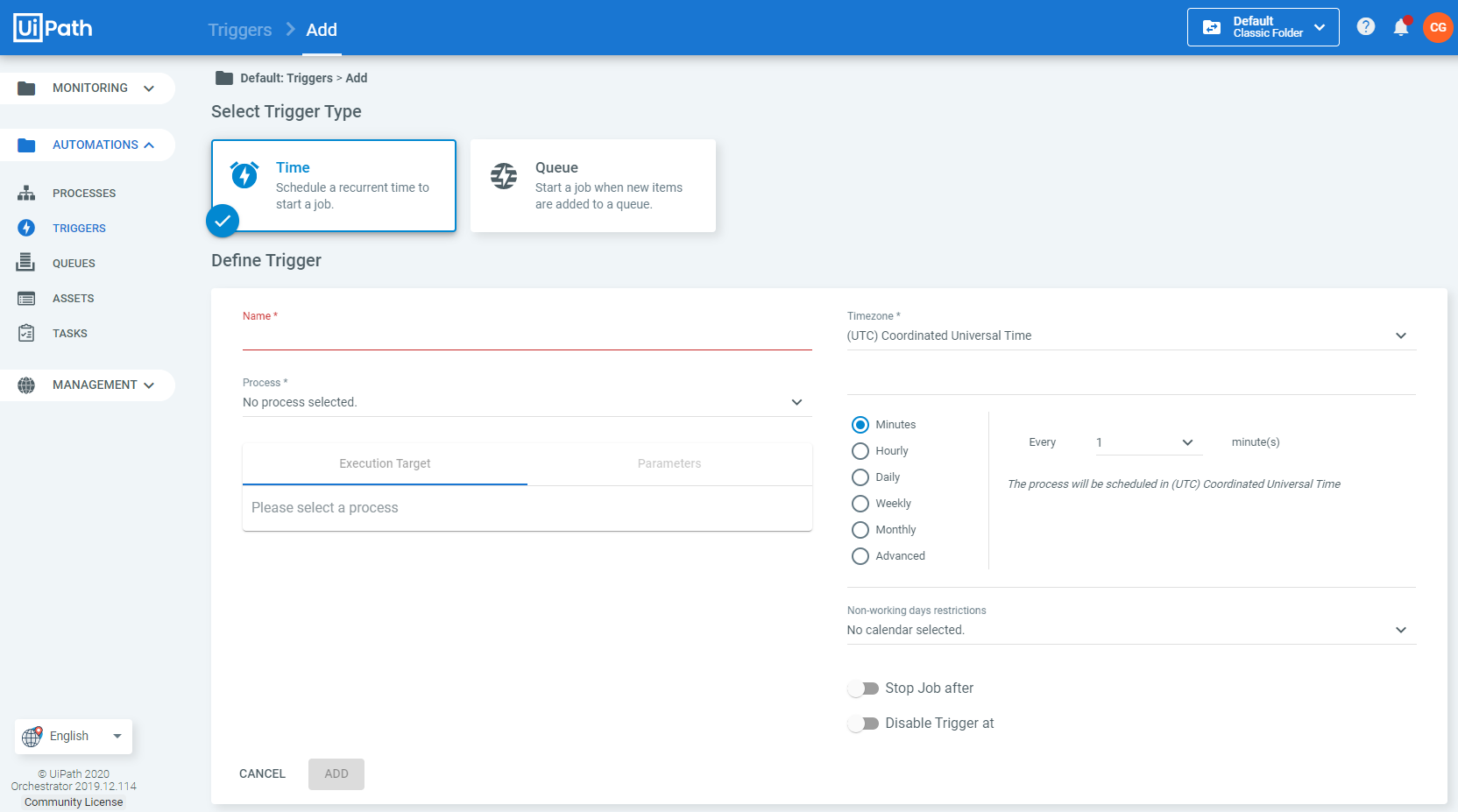


Once the Start button has been pressed, the robot will execute.



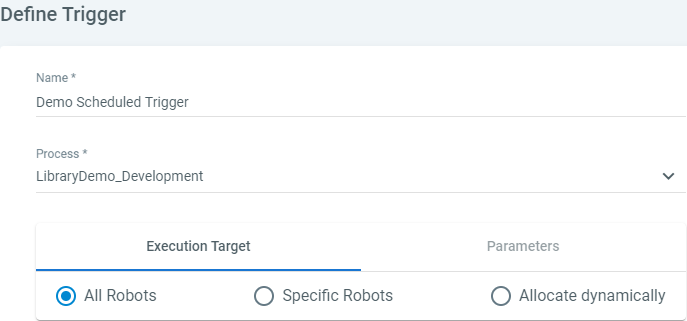
If the user wishes to Schedule a job to execute, whether that be at a fixed point in time, routinely on certain days, based on an inbound email etc. This can be achieved by navigating to Triggers under the Automations heading within Orchestrator. Clicking the familiar Plus button will allow the user to configure the Trigger(s).

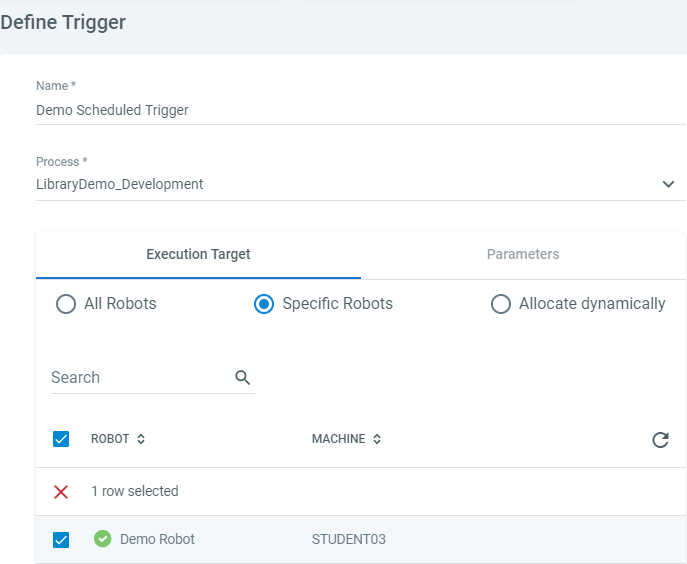


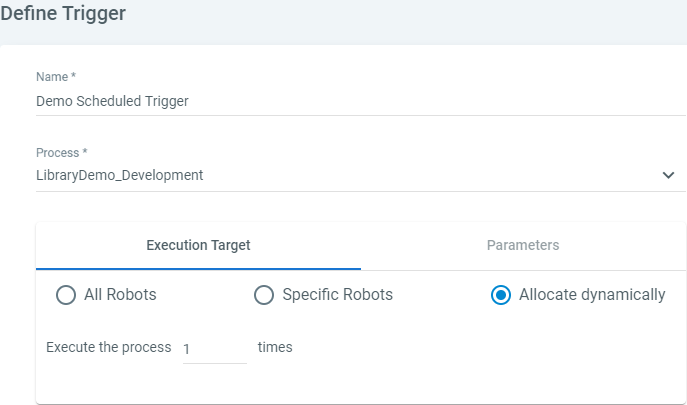


Please note, that there are two types of Triggers the user can select from, Time, and Queue. Queue will be covered in a separate handout.

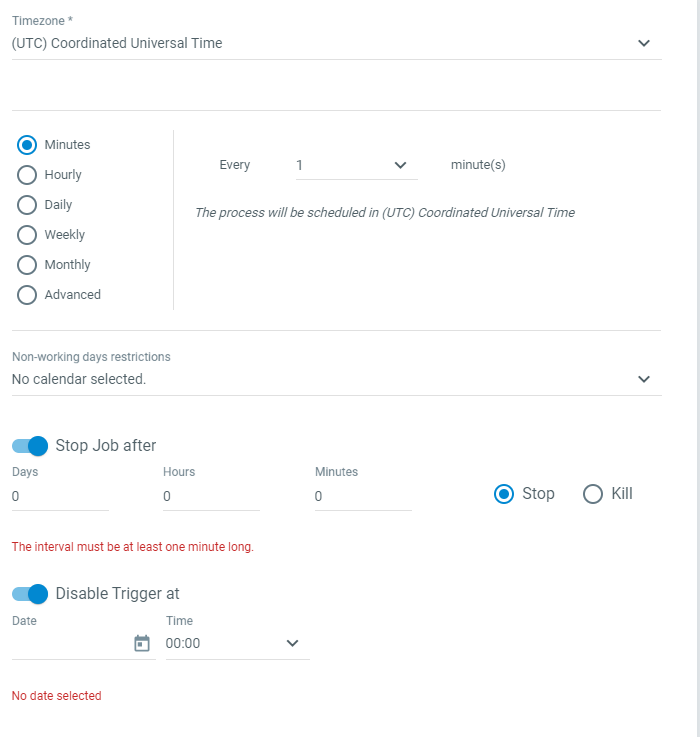
With the Time Trigger selected, the user must provide information to the form. Once the Name has been completed, and a Process has been selected, the Execution Target will change to allow the user to select either ‘All Robots’, ‘Specific Robots’ or ‘Allocate Dynamically’. If All Robots is selected, it will start to run the process on all machines within the environment that the package has been associated with. If Specific Robots is selected, the use can select as many machines as are available within the environment that the packages has been associated with. Finally, if Dynamic Allocation has been selected, Orchestrator will assign the next available machine(s), and asking the user how many executions must be performed.



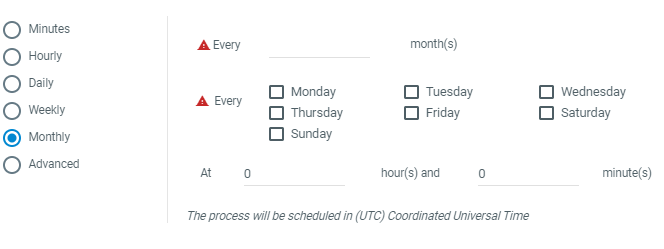
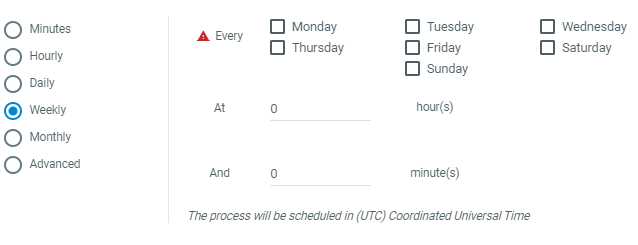
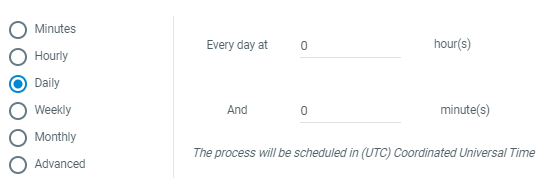
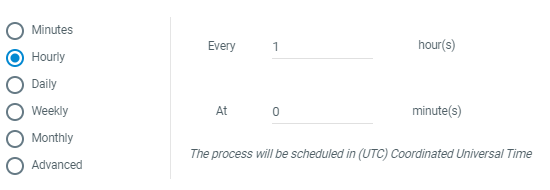
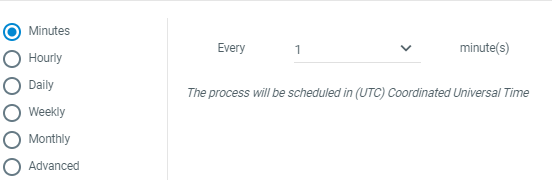




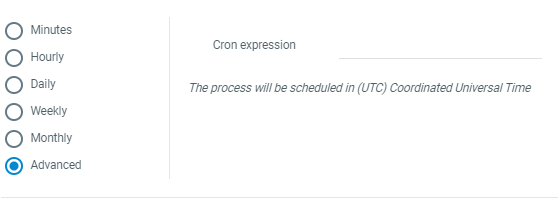
On the other side of this configuration window, the user can specify the frequency at which the job must run. The job can be scheduled to run Every x y; x being a numeric value, and y being Minutes/Hours/Daily/Weekly/Monthly and Advanced.



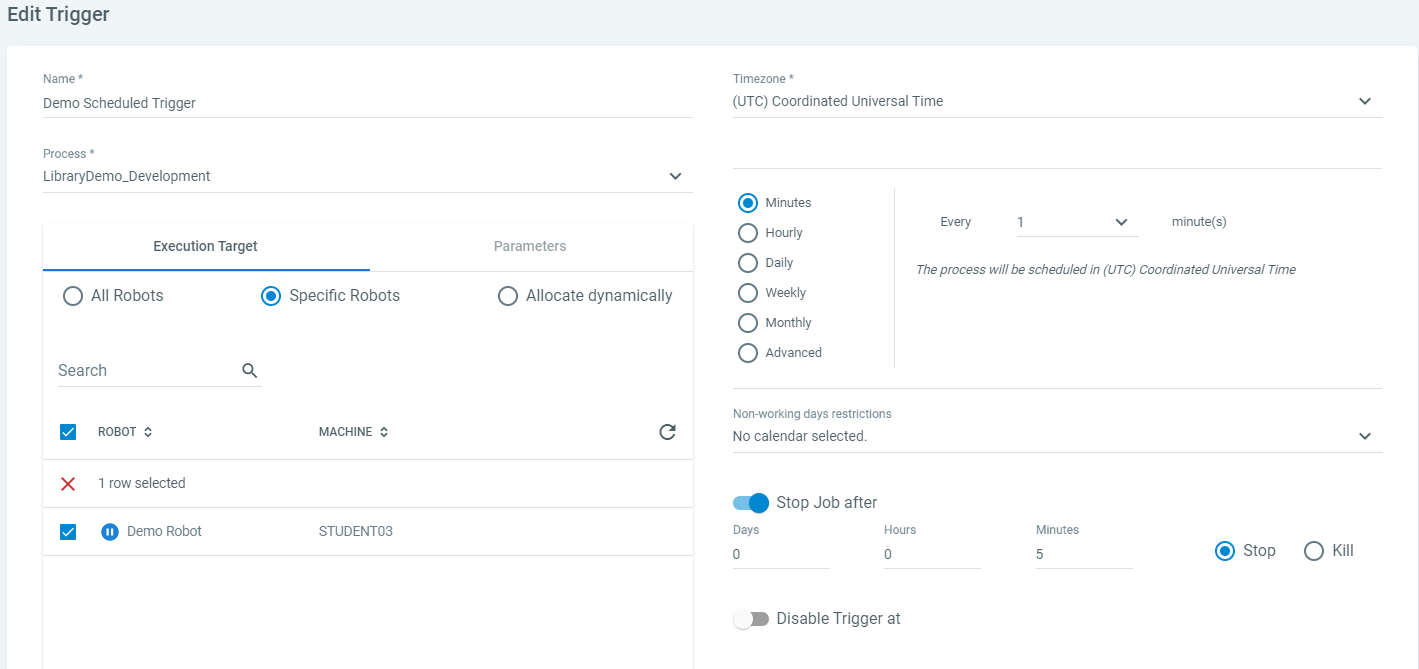
The different interval type interfaces:



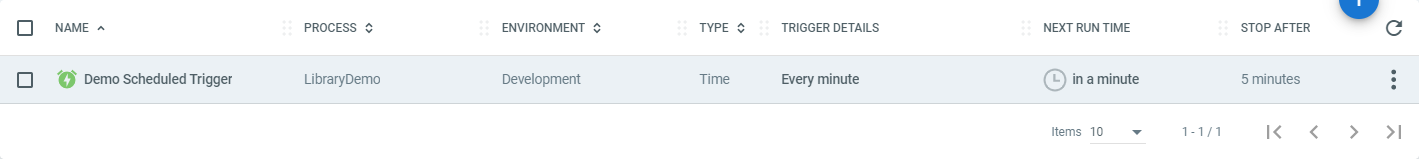
The Advanced option allows the user to write a Cron expression. See <https://healthchecks.io/docs/cron/> for more information.



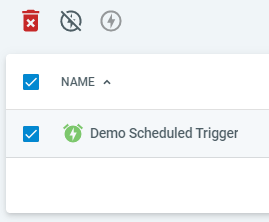
The below configuration is set up to run every minute, for 5 minutes.



Once configured and the Add button has been pressed, the user will see a new line in the familiar interface.

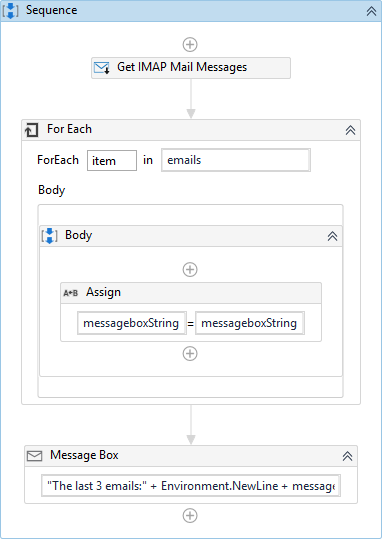


If the user selects the newly created line, new buttons will be unveiled, allowing the user to Delete, Enable or Disable the Trigger.



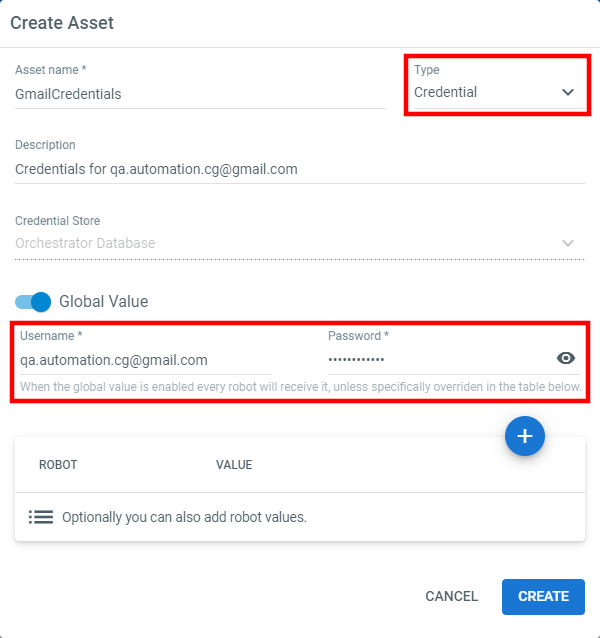
**Credential Assets**

Consider the following Sequence.

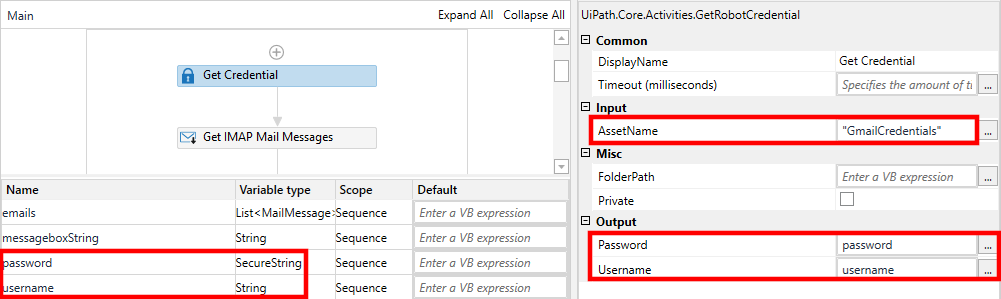
****

The user should be aware at this stage, that Get IMAP Mail Messages requires a Username and a Password for the account they are trying to access. Storing the login credentials within the script is **incredibly bad pactice** and poses a host of security issues. To circumvent this, UiPath’s Orchestrator is capable of handling a Credential Asset.

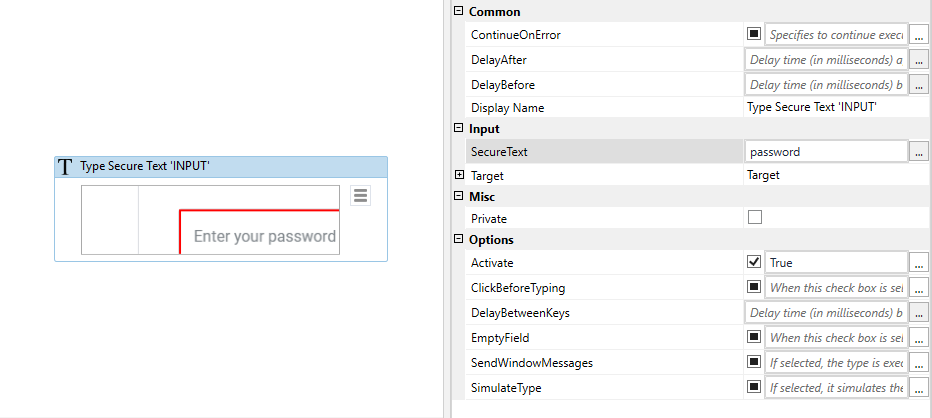
Selecting Assets under the Automations heading, the user will be able to click the Plus button from the familiar display. This will present the user with a form to complete. From the form, selecting Type Credential will change the interface slightly to allow the user to input a Username and Password. The Password will be obfuscated with star symbols. In this instance, a Gmail account is being accessed.



Within the workflow that requires the credentials, a Get Credential Activity must be used. The user must ensure that the AssetName property is written exactly as the Asset name given during the Create Asset process. Furthermore the user must create some variables to store the values within. Using the ‘Control + K’ technique within the Username and Password fields, to create the required variable types.



If these credentials are to be supplied to a User Interface, the Password must be entered with a Type Secure Text Activity must be used, supplying the Secure String Password in the SecureText Property.



If the Password needs to be passed to the Get IMAP Mail Messages, the Password Secure String will not be able to be supplied to the Activity, as it is expecting a **Plain** String. To be able to pass it a Plain String Password without compromising security (displaying plaing text passwords to developers/maintainers), a new String variable must be created, and the Password must be passed to it, utilising the System.Net.NetworkCredential method. The code required for the Expression should be:

new System.New.NetworkCredential(string.Empty, password).Password

Note, the purple password above is the name of the variable created in the Output Password field of the Get Credential Activity.

