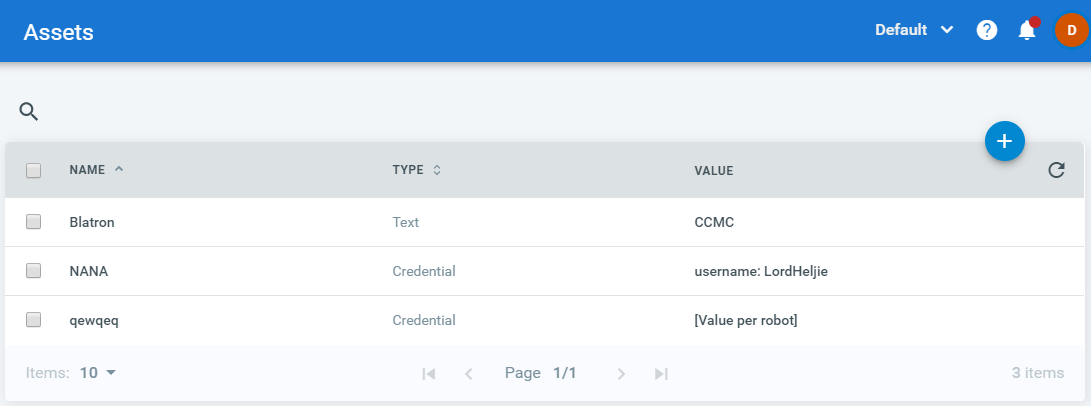
1. **About Assets (Basically assets are used in Orchestrator to configure the secure data (details which will not be exposed to any vendor)**

Assets usually represent shared variables or credentials that can be used in different automation projects. They give you the opportunity to store specific information so that the Robots can easily have access to it.

Additionally, an extra level of security is provided, as all credentials stored here are encrypted with the AES 256 algorithm. They can be invoked by RPA developers when designing a process, but their values can be hidden from them.

The **Assets** page enables you to create new assets. It also displays all previously created assets, which can be edited or deleted.

[](https://files.readme.io/15fec4f-assets_page.png)

The **Get Asset** and **Get Credential** activities used in Studio request information from Orchestrator about a specific asset, according to a provided **AssetName**. If the **AssetName** provided in Studio coincides with the name of an asset stored in the Orchestrator database, and the Robot has the required permissions, the asset information is retrieved and used by the Robot when executing the automation project. For more information, see the [Managing Assets in UiPath Studio](https://orchestrator.uipath.com/docs/managing-assets-in-studio) page.

There are four types of assets:

**Text** - stores only strings (it is not required to add quotation marks)

**Bool** - supports true or false values

**Integer** - stores only whole numbers

**Credential** - contains usernames and passwords that the Robot requires to execute particular processes, such as login details for SAP or SalesForce.

Additionally, assets can be:

**Global** - can be accessed and used by all Robots

**Per Robot** - can be accessed only by an indicated Robot (created only by selecting the **Per Robot** check box)

# Managing Assets in Studio

[SUGGEST EDITS](https://orchestrator.uipath.com/docs/managing-assets-in-studio)

Studio has two activities, **Get Asset** and **Get Credential**, that the Robot can use to extract information from assets stored in the Orchestrator database.

They are displayed in the **Activities** panel, under **Orchestrator > Assets**, and are part of the UiPath Core activities pack.

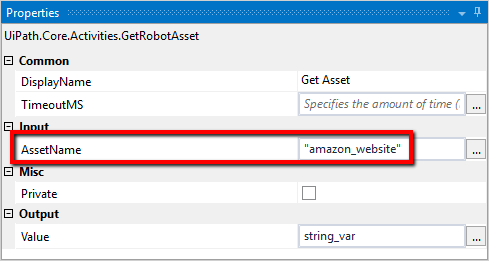
The **Get Asset** activity should be used with the String, Bool and Integer assets, while **Get Credential** should be used with Credential assets.

### Note:

The asset name is not case sensitive. For example, "Text" and “teXt” are the same.

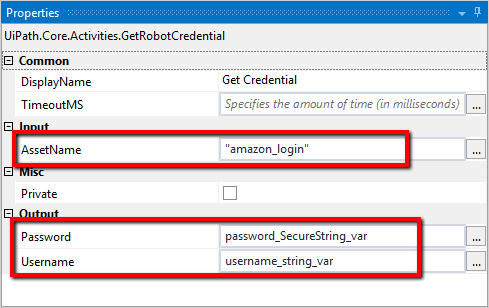
# Using the Get Asset Activity

* In Orchestrator, create a string, boolean or integer asset.
* In Studio, create a string, boolean or integer variable, depending on the type of asset you want to use from Orchestrator. For example, if you want to work with a string asset, create a string variable.
* From the **Activities** panel, drag a **Get Asset** activity to the **Main** panel.
* In the **Properties** panel, in the **AssetName** field, type the name of the Orchestrator asset you want to use, and place it between quotation marks. For example, if in Orchestrator the asset name is StringAsset, write "stringasset" in Studio.
* In the **Properties** panel, in the **Value** field, enter the variable created at step 2. This variable stores information from the specified Orchestrator asset, if the **AssetName**coincides with the asset name that is stored in the Orchestrator database, and the Robot has the required permissions.

[](https://files.readme.io/350509c-get_asset_properties.png)

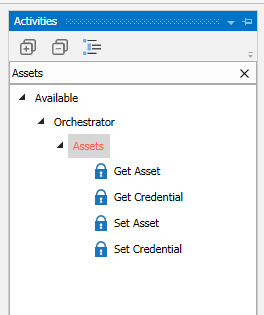
# Using the Get Credential Activity

1. In Orchestrator, create a **Credential** asset.
2. In Studio, create a String variable. This is used to store the username part of the credentials.
3. Create a SecureString variable. This is used to store the password. The SecureString is a special .NET Framework variable type that is encrypted within the framework.
4. In the **Properties** panel, in the **AssetName** field, type the name of the credential asset, as it is in Orchestrator, and place it between quotation marks. For example, "amazon\_login".
5. In the **Properties** panel, in the **Password** field, enter the SecureString variable.
6. In the **Properties** panel, in the **Username** field, enter the string variable. The credential asset stored in Orchestrator can be used by the Robot too, as long as the **AssetName**coincides with the asset name that is stored in the Orchestrator database, and the Robot has the required permissions.

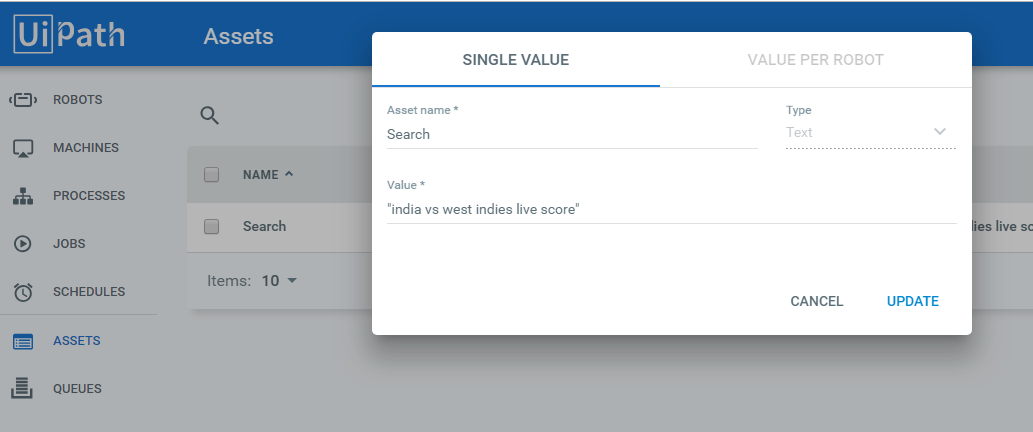
[](https://files.readme.io/b246bda-get_credential_properties.png)

**Exercise: on Get Asset Activity**

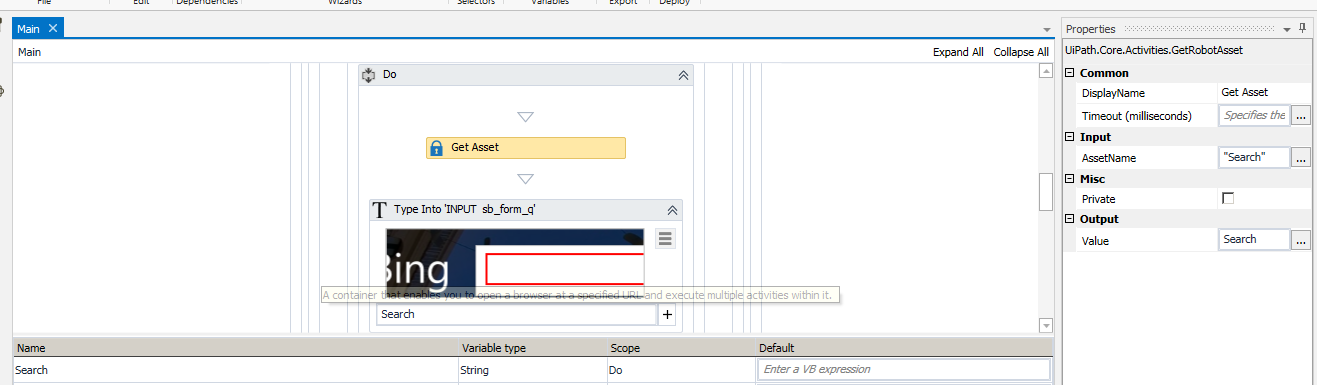
1. Select the Get Asset Activity from the Activities



1. Create the Asset of type ‘Text ’ from the Orchestrator



1. Invoke the Activity Get Asset in the project -> Specify the asset name mentioned in Orchestrator using quotation marks (example: Asset name: Search in Orchestrator should be defined in “Search ” in get assert activity Assert name) - > specify the value based on Asset Type mentioned in orchestrator (currently we used string for the type ‘Text’) and that’s it



1. Update the package in the orchestrator once after publish (shown in above screen shot on how to use the latest package and run the jobs -> it should get expected result )

For more information please refer:- <https://orchestrator.uipath.com/docs/>

