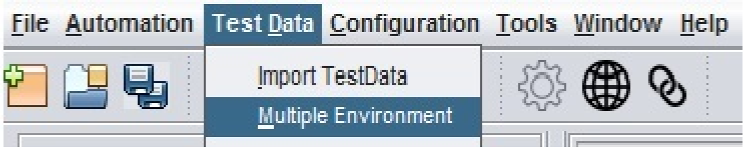
Things You Should Know

Environment Based Execution

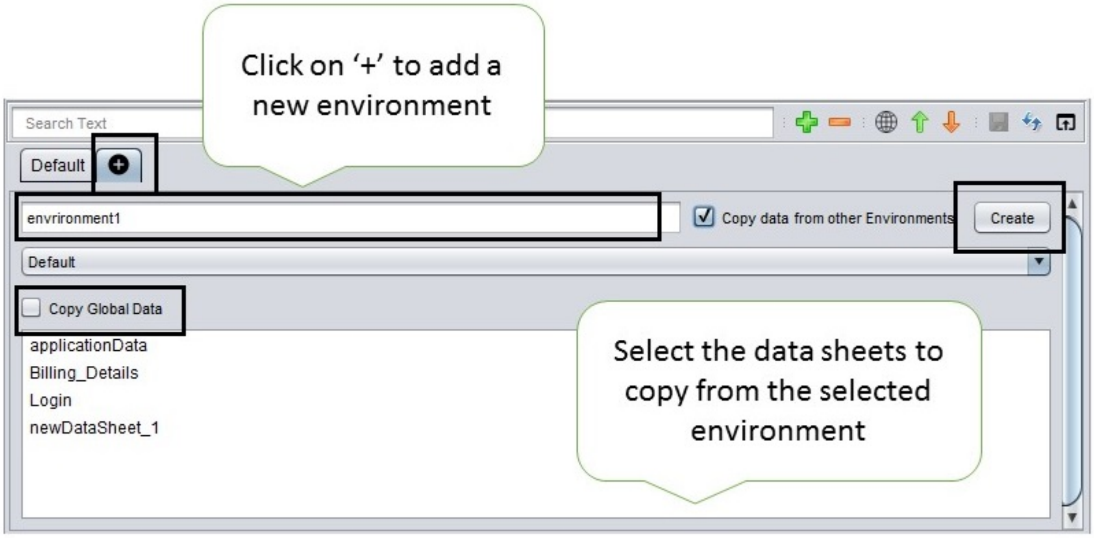
Environment based execution allows you to configure your test cases to run them in multiple environments. It is achieved by creating data sheets for each environment.

To create data sheets for environment-based execution, please follow the steps below to create new environments and add data sheets to the same

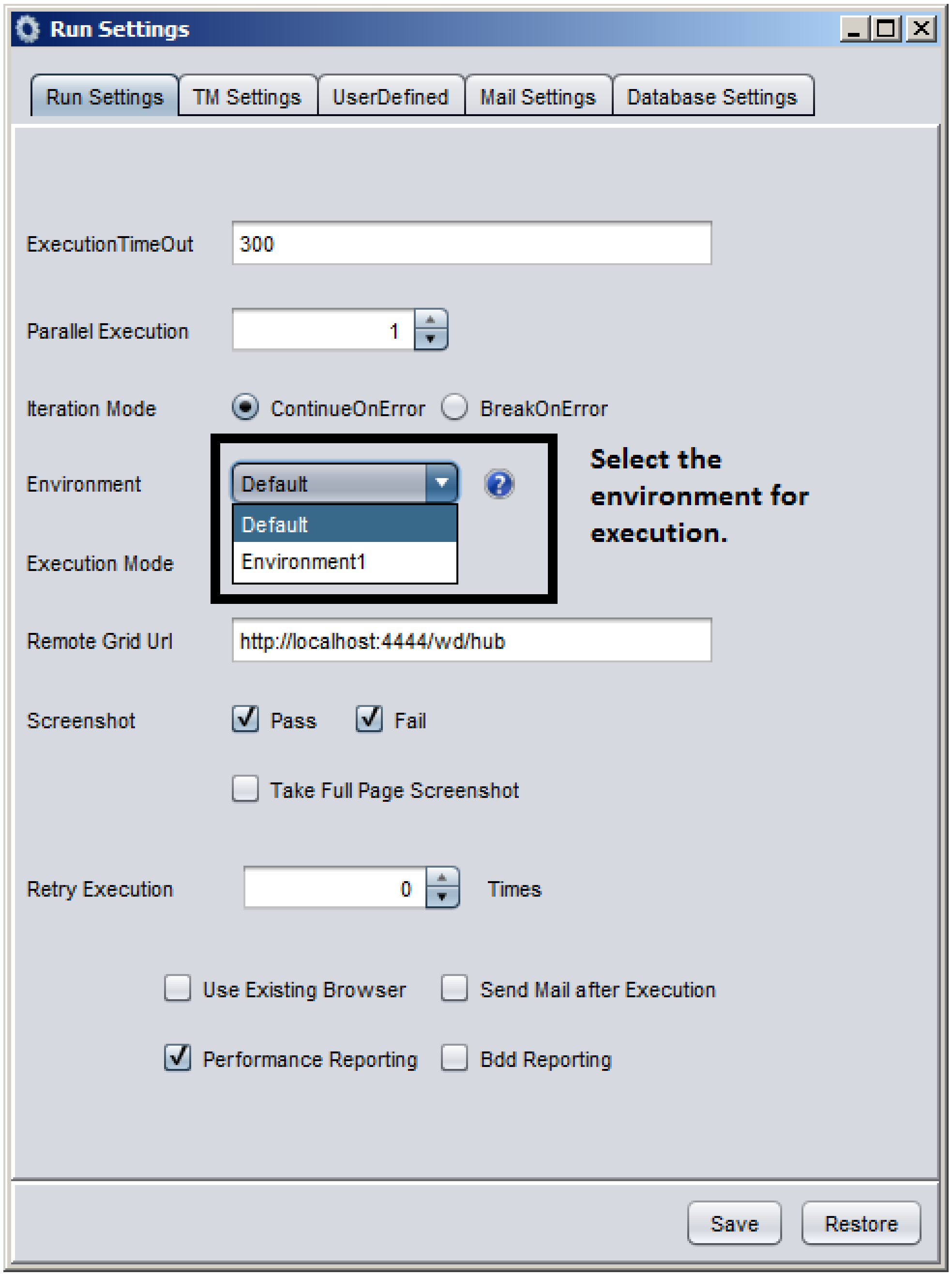
- Navigate to **Test Data>Multiple Environment** and click on the **Multiple Environment** option to enable the same.



- Click on the '+' sign, as shown in the image below to create a new environment.
- Give a new name to the environment in the respective option
- If you want to copy data sheets and global data sheet from selected environment , choose the respective option, as highlighted in the image below.
- Click on **create** to create the same


Click on '+' to add a new environment
Search Text
Default
envrironment1
☑ Copy data from other Environments
Create
Default
□ Copy Global Data
applicationData
Billing_Details
Login
newDataSheet_1
Select the data sheets to copy from the selected environment

- After this, just select the environment for execution from **Configuration>Run Settings** and choose the environment under **Environment** option



ExecutionTimeOut: 300

Parallel Execution: 1

Iteration Mode: ● ContinueOnError ○ BreakOnError

Environment: Default (Default, Environment1)

Execution Mode

Remote Grid Url: http://localhost:4444/wd/hub

Screenshot: ☑ Pass ☑ Fail

□ Take Full Page Screenshot

Retry Execution: 0 Times

□ Use Existing Browser □ Send Mail after Execution

☑ Performance Reporting □ Bdd Reporting

Select the environment for execution.

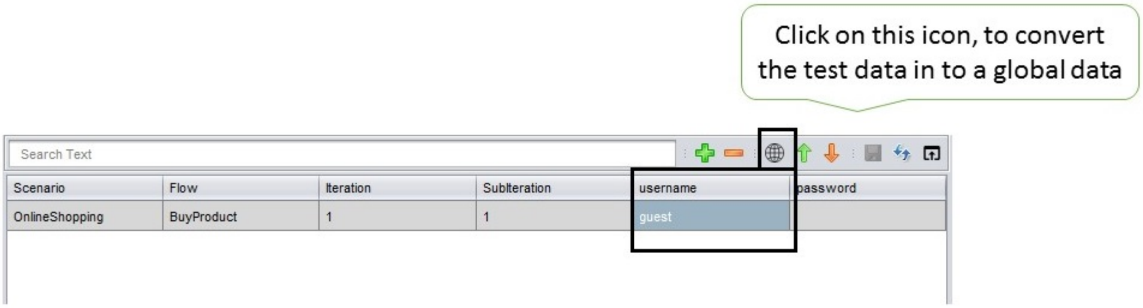
Note: You can also select a test set and save the environment based settings for the same under the **Quick Settings** section of the Execution Panel. All the test cases in the test set will run with the test data chosen from the selected environment and if the particular data sheet or the column is not available in the environment selected then it will search for the same in the **Default** set of data sheets.

How To Use Global Data Sheet?

For every project that is created in Cognizant Intelligent Test Scripter, a default global data sheet is generated. The data entered in the global data sheet can be referenced in any test data sheet within the project. This allows you to reuse the data across multiple test cases.

Where to use the Global Data?

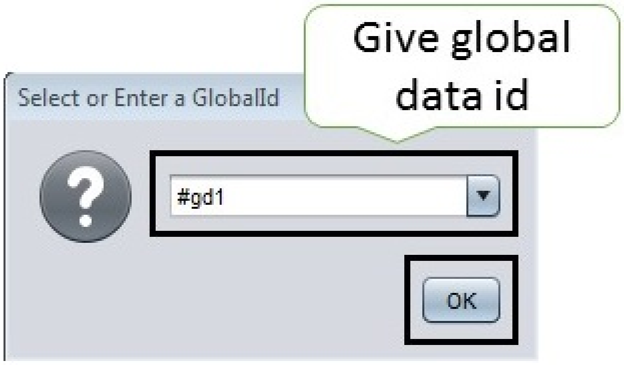
Identify the Data column name which is common across multiple data sheets.



Click on this icon, to convert the test data in to a global data

Scenario	Flow	Iteration	SubIteration	username	password
OnlineShopping	BuyProduct	1	1	guest	

Select that data value and click on **global data** icon, as shown in the image above. Give the global data id, in the respective window, as shown below,

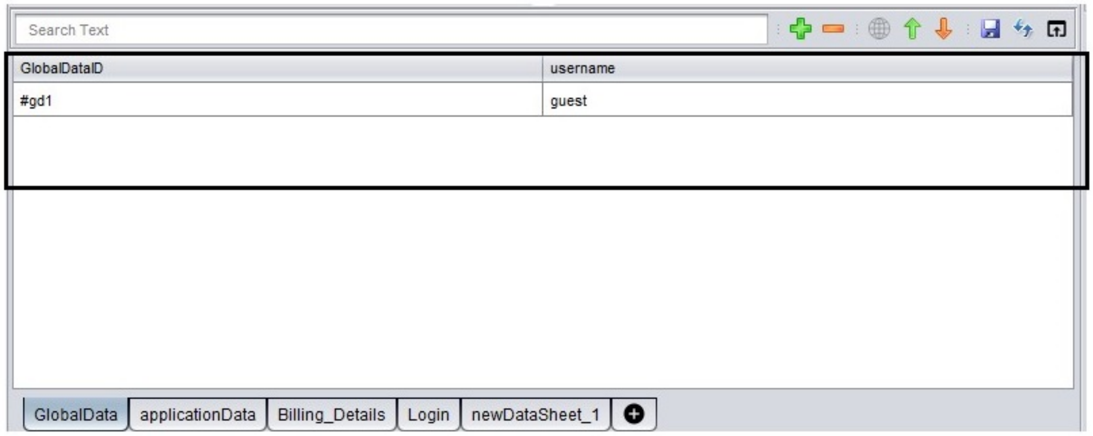


Give global data id

Select or Enter a GlobalId

#gd1

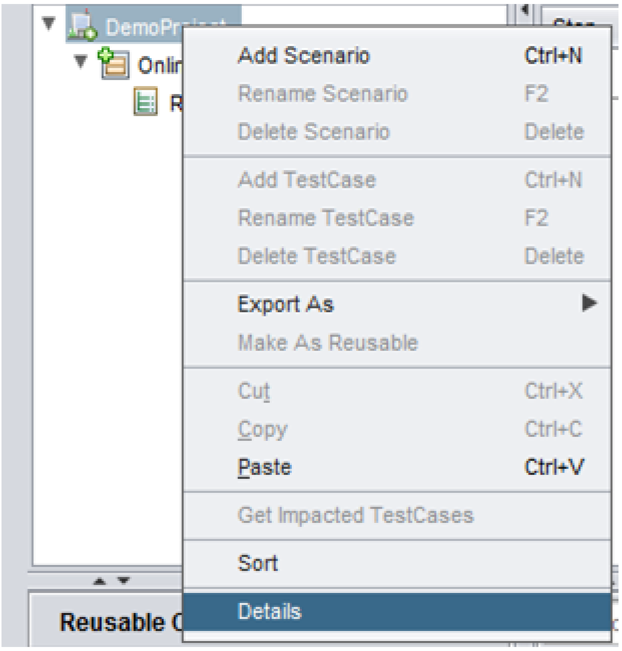
You will get the global id in the global data sheet, as shown below.



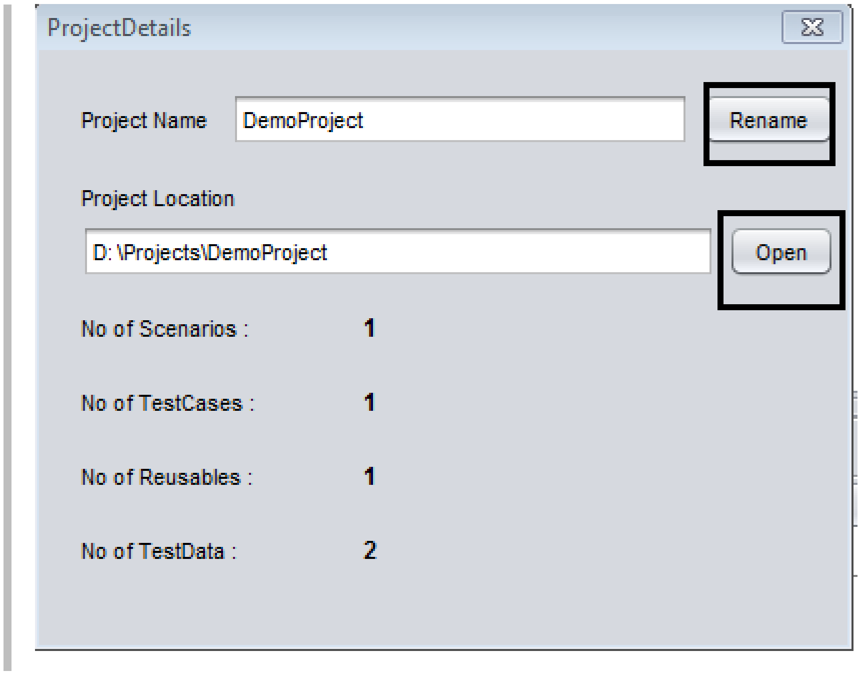
GlobalDataID	username
#gd1	guest

How To Rename Project?

- To rename your project, right click on the project name from the **Test Plan** section and choose **Details** from the context menu option.



- The **Project Details** window opens and the following options are available in the window, as shown below.



ProjectDetails

Project Name: DemoProject

Project Location: D: \Projects\DemoProject

No of Scenarios : 1

No of TestCases : 1

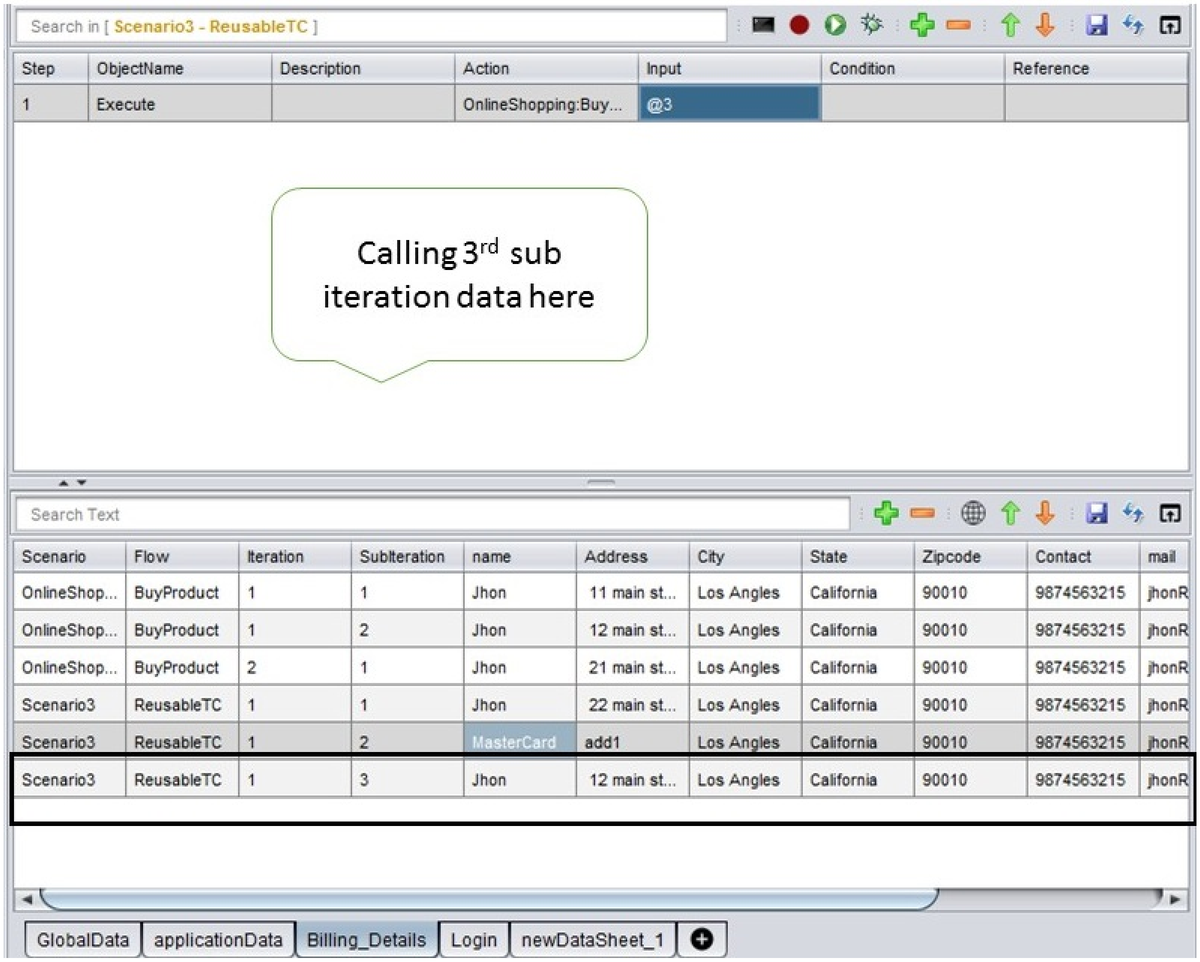
No of Reusables : 1

No of TestData : 2

- To rename your project, give the new name in the **Project Name text box** and click on **Rename**. Your project will be renamed.
- The **open** option opens the project location in your system.
- Right below these options, you have details on the total number of scenarios, test cases, reusable components and test data sheets present in the project.

How To Execute A Reusable Test Case For Desired Set Of Data From The Data Sheet?

> A reusable test case can be executed with a desired set of data by providing the sub iteration index in the Input column. It will fetch the data set from the data sheet with the desired index and use it within the test case. This is very useful when you want to execute the same reusable component, inside a single test case, with different set set of data passed each time.



Search in [Scenario3 - ReusableTC]
Calling 3rd sub iteration data here
Search Text
GlobalData
applicationData
Billing_Details
Login
newDataSheet_1

Step	ObjectName	Description	Action	Input	Condition	Reference
1	Execute		OnlineShopping:Buy...	@3		

Scenario	Flow	Iteration	SubIteration	name	Address	City	State	Zipcode	Contact	mail
OnlineShop...	BuyProduct	1	1	Jhon	11 main st...	Los Angles	California	90010	9874563215	jhonR
OnlineShop...	BuyProduct	1	2	Jhon	12 main st...	Los Angles	California	90010	9874563215	jhonR
OnlineShop...	BuyProduct	2	1	Jhon	21 main st...	Los Angles	California	90010	9874563215	jhonR
Scenario3	ReusableTC	1	1	Jhon	22 main st...	Los Angles	California	90010	9874563215	jhonR
Scenario3	ReusableTC	1	2	MasterCard	add1	Los Angles	California	90010	9874563215	jhonR
Scenario3	ReusableTC	1	3	Jhon	12 main st...	Los Angles	California	90010	9874563215	jhonR

Points To Remember While Injecting Script

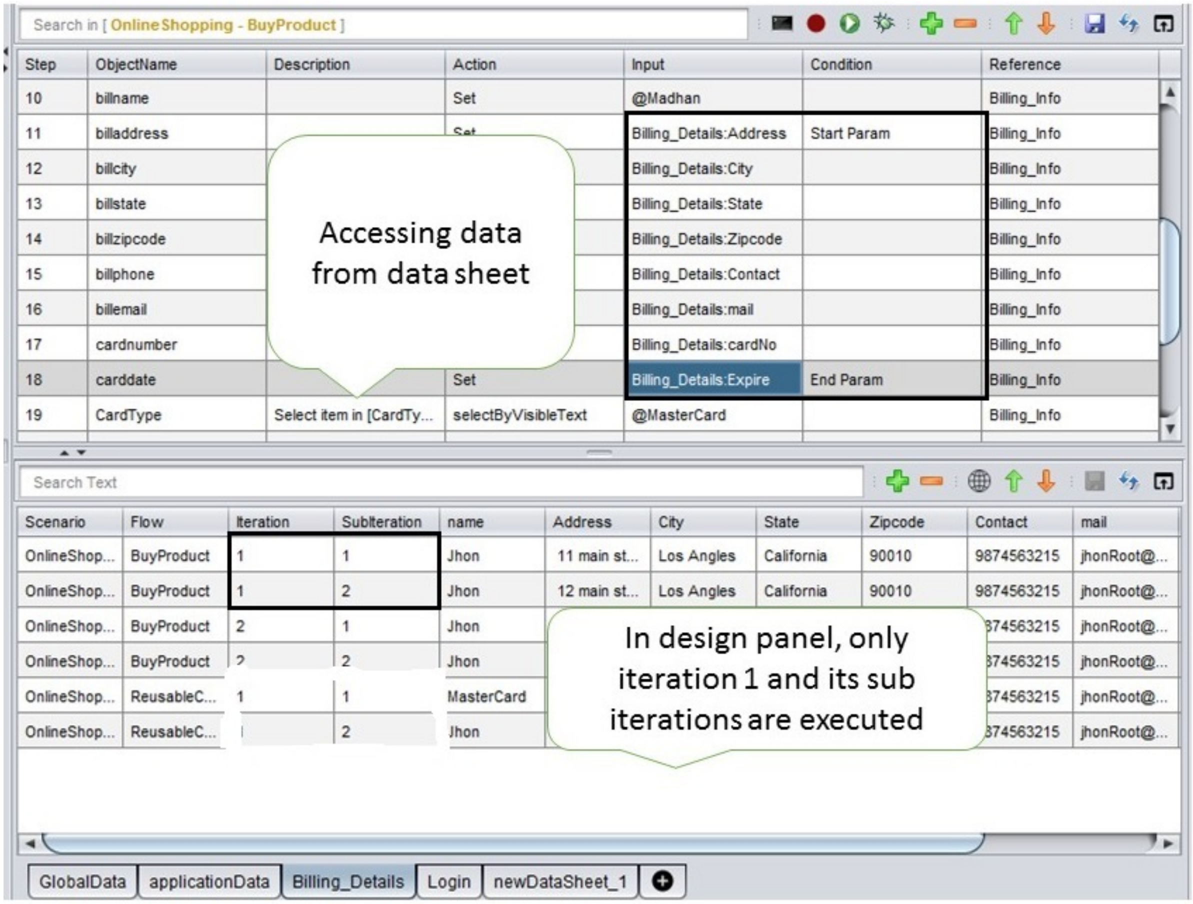
Standards to be followed while writing custom functions are described below.

- Do not provide any package declarations when you inject through **Inject Script**.
- Custom Functions should be **public.**
- Custom Functions should be of **void** type (It should not return anything)
- Custom Function should not contain parameters (use **Data** or **Input** keyword for fetching data from Cognizant Intelligent Test Scripter Input column)
- Do not inject anything that is available within the **Engine folder.**
- Use **Engine** to write custom code that requires the use of any external library.

Note: You need to have **jdk** to perform Inject script, since the .java files containing the source code needs to be compiled in to **.class** file for execution.

How To Loop?

- **NOTE:** In Test Design panel, users are only given access to Iteration 1 and its various Subiterations



Step	ObjectName	Description	Action	Input	Condition	Reference
10	billname		Set	@Madhan		Billing_Info
11	billaddress		Set	Billing_Details:Address	Start Param	Billing_Info
12	billcity			Billing_Details:City		Billing_Info
13	billstate			Billing_Details:State		Billing_Info
14	billzipcode			Billing_Details:Zipcode		Billing_Info
15	billphone			Billing_Details:Contact		Billing_Info
16	billemail			Billing_Details:mail		Billing_Info
17	cardnumber			Billing_Details:cardNo		Billing_Info
18	carddate		Set	Billing_Details:Expire	End Param	Billing_Info
19	CardType	Select item in [CardTy...	selectByVisibleText	@MasterCard		Billing_Info

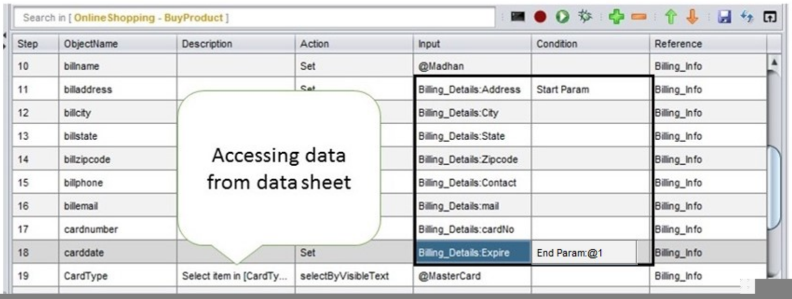
Accessing data from data sheet

Scenario	Flow	Iteration	SubIteration	name	Address	City	State	Zipcode	Contact	mail
OnlineShop...	BuyProduct	1	1	Jhon	11 main st...	Los Angles	California	90010	9874563215	jhonRoot@...
OnlineShop...	BuyProduct	1	2	Jhon	12 main st...	Los Angles	California	90010	9874563215	jhonRoot@...
OnlineShop...	BuyProduct	2	1	Jhon					374563215	jhonRoot@...
OnlineShop...	BuyProduct	2	2	Jhon					374563215	jhonRoot@...
OnlineShop...	ReusableC...	1	1	MasterCard					374563215	jhonRoot@...
OnlineShop...	ReusableC...	1	2	Jhon					374563215	jhonRoot@...

In design panel, only iteration 1 and its sub iterations are executed

- For the above image, Steps 11 to 18 will be executed 2 times (equal to the number of Sub iterations for Iteration 1)
  - For first iteration, data will be taken from Iteration 1 and Subiteration 1
  - For second iteration, data will be taken from Iteration 1 and Subiteration 2

- To repeat an entire Test Case, keep **Start Param** in the first Step of your test Case and **End Param** in the last Step of your Test Case.



Search in [OnlineShopping - BuyProduct]

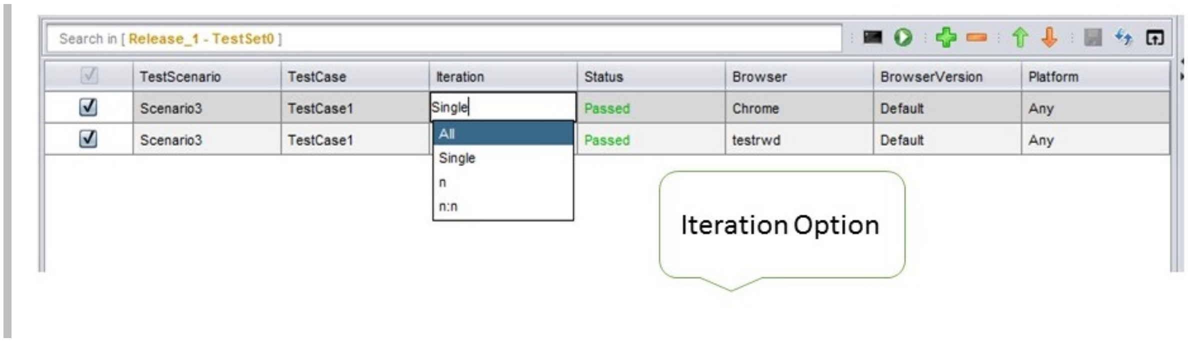
Step	ObjectName	Description	Action	Input	Condition	Reference
10	billname		Set	@Madhan		Billing_Info
11	billaddress		Set	Billing_Details:Address	Start Param	Billing_Info
12	billcity			Billing_Details:City		Billing_Info
13	billstate			Billing_Details:State		Billing_Info
14	billzipcode			Billing_Details:Zipcode		Billing_Info
15	billphone			Billing_Details:Contact		Billing_Info
16	billemail			Billing_Details:mail		Billing_Info
17	cardnumber			Billing_Details:cardNo		Billing_Info
18	carddate		Set	Billing_Details:Expire	End Param:@1	Billing_Info
19	CardType	Select item in [CardTy...	selectByVisibleText	@MasterCard		Billing_Info

Accessing data from data sheet

- In the above image, we have used **End Param:@1**. This will execute Step 11 to 18, once and not twice.
- To iterate through the entire Test Case for all the sub iterations of data in the data sheet, keep **Start Param** in the first step and **End Param** in the last step of your Test Case.

How To Use Iterations And Subiterations In The Test Execution Panel?

- **NOTE:** In the Test Execution panel, you are given access to all Iterations and their Sub iterations.



Search in [Release_1 - TestSet0]

☑	TestScenario	TestCase	Iteration	Status	Browser	BrowserVersion	Platform
☑	Scenario3	TestCase1	Single	Passed	Chrome	Default	Any
☑	Scenario3	TestCase1		Passed	testrwd	Default	Any

Iteration Option

- In the above image, there are some options given under the Iteration column
  - **All**: will Iterate through all your Iterations of your datasheet.
  - **Single**: will only Iterate through Iteration 1 of your datasheet.
  - **n:n (Range)**: will Iterate through a range of iterations.

    > For example, if you use 2:5, then Cognizant Intelligent Test Scripter will only Iterate through Iteration 2 to Iteration 5 even if there are more iterations.

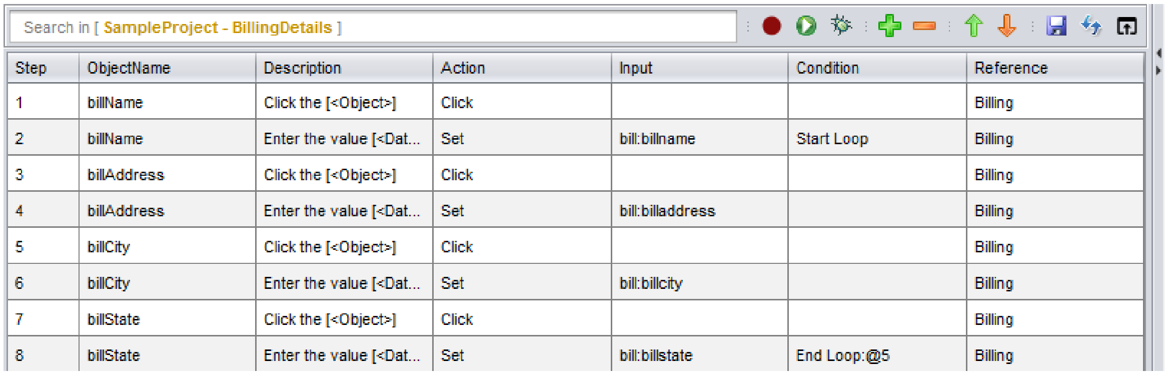
  - **n**: will allow you to Iterate a particular (nth) iteration in your datasheet.

    > For example, if you use 5, then Cognizant Intelligent Test Scripter iterate only the 5th iteration and all its sub iterations.

> **Note:** Subiterations will work according to the definition of **Start Param** and **End Param** in the Test Design panel.

Looping Without Test Datasheet

- To repeat few steps, use Start Loop and **End Loop:@n**.



Search in [SampleProject - BillingDetails]

Step	ObjectName	Description	Action	Input	Condition	Reference
1	billName	Click the [<Object>]	Click			Billing
2	billName	Enter the value [<Dat...	Set	bill:billname	Start Loop	Billing
3	billAddress	Click the [<Object>]	Click			Billing
4	billAddress	Enter the value [<Dat...	Set	bill:billaddress		Billing
5	billCity	Click the [<Object>]	Click			Billing
6	billCity	Enter the value [<Dat...	Set	bill:billcity		Billing
7	billState	Click the [<Object>]	Click			Billing
8	billState	Enter the value [<Dat...	Set	bill:billstate	End Loop:@5	Billing

- In the above image, Start Loop is at Step 2 and **End Loop:@5** is at Step 8. So, Step 2 to 8 will be repeated 5 times.

Execution Of Scripts In Safari 10

Consider the following points,taken from the official links, before performing execution in Safari,

- As a WebDriver test is executing in an Automation window, any attempts to interact with the window or web content could derail the test by unexpectedly changing the state of the page. To prevent this from happening, Safari installs a “glass pane” over the Automation window while the test is running. This blocks any stray interactions (mouse, keyboard, resizing, and so on) from affecting the Automation window.
- The old SafariDriver implementation is no longer maintained and should not be used. You do not need to download anything besides Safari 10 to get the Apple-developed driver
- Some users reported that they could not get safaridriver to run correctly after installing **Safari 10** on **El Capitan**. After installing Sierra the execution occurred
- Safari’s driver restricts the number of active WebDriver sessions. Only one Safari browser instance can be running at any given time, and only one WebDriver session can be attached to the browser instance at a time.

> **Note**: For more information, refer here [https://webkit.org/blog/6900/webdriver-support-in-safari-10/]

How To Do Web Performance Testing With Cognizant Intelligent Test Scripter?

Create Har files

- Refer the actions section WebPage Performance Actions [../actions/performanceaction.html#webpage-performance-actions] of the document to check how to use the Performance testing actions
- Navigate to **Configuration>Run Settings** and check the **Performance Reporting** check box under the **Run settings tab**.

- Once this configuration is saved, run your test set and post the execution you will get a report.Expand the report by clicking on your test case and this time the **console** icon on the bottom right will change into the **Performance Report icon.**
- To launch the **performance report** click on the **Performance Report icon**.
- The table and graph view contains the **har** files generated post execution.Click on them to get the various **page and resource timings parameters** captured by the **capturePageTimings** action.

Compare Har Files

To Compare your **Har** files, navigate to **Tools>Har Compare** and you will be redirected to the **Har Comparator** site.

Here you compare the **Har** files that you have in your project and also set **benchmarks** for achieving standards.

How To Schedule Tasks With Cognizant Intelligent Test Scripter?

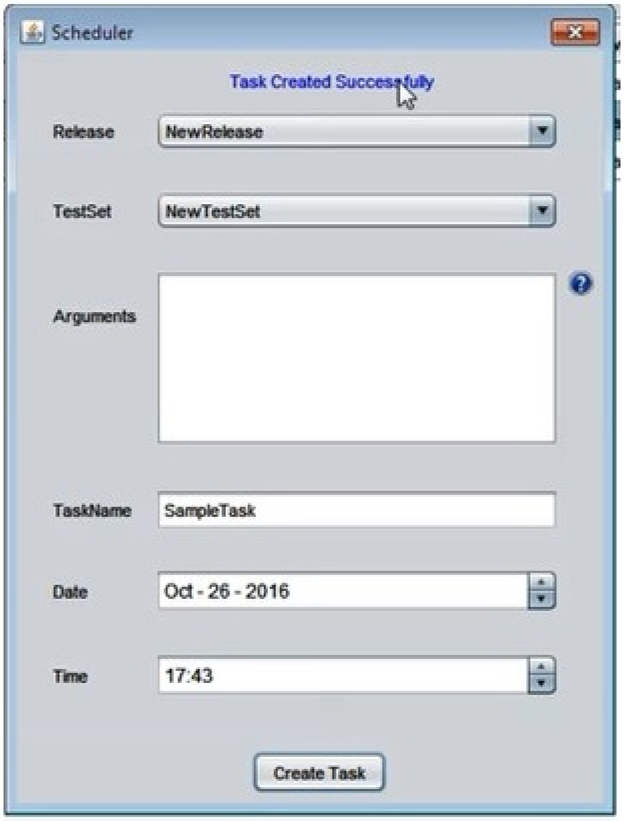
Follow the steps below to perform **schedule run**

- Launch **Cognizant Intelligent Test Scripter Scheduler** by navigating to **Configuration>Schedule Run** and configure it as per your requirements.

[redacted]

- Provide the name of your task and test set details and the time and the date on which you want to execute the specified test sets.You can also add the arguments here, these are similar to the command line arguments used in Cognizant Intelligent Test Scripter.
- Arguments that can be added are

> dont_launch_report Disables launching summary report after execution
>
> setEnv <arg Create/Set Env settings to override project Settings from Command Line.

> setVar <arg Create/Set user defined variable [-setVar "var=value"]
>
> standalone_report Create Standalone Report instead of Relative one
>
> 
>
>
>
> Task Created Successfully
>
> Release: NewRelease
>
> TestSet: NewTestSet
>
> Arguments:
>
> TaskName: SampleTask
>
> Date: Oct - 26 - 2016
>
> Time: 17:43

- Open **windows Task Scheduler** (**Start>Control Panel>System and Maintenance>Administrative Tools>Task Scheduler**) and search for the task you scheduled in Cognizant Intelligent Test Scripter and **double click** it.

- Click on **Triggers** tab then click on Edit

- Now you can configure it as per your requirements as shown below.

How To Handle Random Actions/Pop up Windows?

In some applications, there may be components which will originate at random ie. the origination of such components cannot be predicted. This section will describe how to handle such components.

Logic

Since the origin of some components cannot be predicted, check for the availability of the respective component before or after performing each step.

How To Do It?

- Open Cognizant Intelligent Test Scripter Engine in **Eclipse IDE** (or any IDE supporting java development).
- Navigate to **com.cognizant.cognizantits.engine.execution.run**.
- The following actions will be available in this class **Annotation**:
  - **beforeStepExecution**:Enter the code to handle the component here ,if you want to perform the check before the execution of each step.
  - **afterStepExecution**:Enter the code to handle the component here,if you want to perform the check after the execution of each step.

How To Change Object Properties At Run Time?

> Sometimes it is necessary to use a dynamic property for objects at runtime. Customizing the existing properties at runtime can be done by using **setObjectProperty** and **setglobalObjectProperty** functions. **setObjectProperty** function: will update property of the driving object. **setglobalObjectProperty** function: will update property for all the objects.

setObjectProperty

- **setObjectProperty** action can be used by an object.
- To provide dynamic property value, use the **Input** column.
- To provide flag for replacing the property string, use the **condition** column.

setglobalObjectProperty

- **setglobalObjectProperty** action can be used by Browser object.
- To provide dynamic property value, use the **Input** column.
- To provide flag for replacing the property string, use the **condition** column.
- Setting Global Object property is optional. To use object with global property when required, set **GlobalObject** in the **Condition** column.

> **Note**: It is also possible to use **setObjectProperty/setglobalObjectProperty** for replacing multiple variables at the same time. Follow the syntax shown in the image below for the same,

How To Navigate between Test Data Sheets?

- Right -click on any data sheet and choose the option **Search Test Data** from the context menu

- In the "Go To Test Data" window, choose the test data sheet or the global data sheet to which you want to navigate

- You will be navigated to the respective sheet after clicking on "**OK**"

How To Handle Ajax Controls?

> Your application may have certain objects that are dependent on the **Ajax** queries. Such objects can be handled using some useful methods available in Cognizant Intelligent Test Scripter such as the **Wait** method.

The below image shows a sample scenario involving Ajax controls:

- The above scenario can be handled in your test case by using wait actions as shown in the image below:

How Do I Call My AutoIT scripts in Cognizant Intelligent Test Scripter?

To call any **exe** file and execute the same directly from the UI, follow the procedure below,

- Create a test step, under the object name as **App** , action as **openApp** and give the path of the **exe** file under the **Input** column, as shown below.

Now when you run this script, the corresponding **.exe** file will be executed.

User Defined Variables In Cognizant Intelligent Test Scripter.

User defined variables can be created and defined by two ways as given below.

- Using the actions **AddVar** and **AddGlobalVar**. **AddVar** action will let you define a variable whose scope is throughout the execution of your test case and is valid for all the sub-iterations of the iteration 1 of your test case. But **AddGlobalVar** action will let you define a variable whose scope is throughout the execution of your test set. Click here [../actions/webactions.html#addvar] for more details on the given actions.
- Navigate to **Configuration>Run Settings** tab and provide your variable name and value under the **Property Name** and **Value** columns respectively, as shown in the screen shot below. Click on **save** for saving your settings.

- These variables can now be used anywhere in your project, an example of which is shown below.

- In addition to these, we have certain actions that take variables as input and will store the corresponding result in the variable provided.

How To Change Cognizant Intelligent Test Scripter Report Theme?

- Go to **Configuration>Options Theme**.
- From the **Reporting theme**, select a theme of your choice.

- You can also toggle between **Single** and **Group View.**

- You can choose which columns need to be displayed in the report.

- Show or hide columns in the report using the **show/hide columns** option. Check the respective column name checkbox to display that column or **uncheck** the respective column name checkbox to hide it.

Filtering In Report

You can filter reports using **Global Search** or **Column Search**.

Sorting In Report

You can sort the reports using the **up/down** arrows beside each column name.

How To Generate Run Time Data In Cognizant Intelligent Test Scripter

There are options available in the **Input** column to perform functions like rounding off numbers, generating a random number, concatenating strings, other arithmetic calculations and etc.

Follow the steps below to access those options.

- Consider any action that requires **Input** column. For instance let us take the **Print** action. Instead of providing the URL directly, you can concatenate two strings that lead to the URL.
- Consider the custom method below that uses the **Data** keyword which will process the concatenate function and open the URL.

Function	Syntax	Description
=Round	=Round(Decimal number)	Will round off the given decimal number
=Pow	=Pow(a,b)	Will return the value of **a^b**
=Min	=Min(a,b)	Will return the minimum number between **a** and **b**
=Max	=Max(a,b)	Will return the maximum number between **a** and **b**
=Random	=Random(a)	Will return a random number of **"a"** digits
=Random	=Random(a,b)	Will return a random number between **a** and **b**
=Concat	=Concat(a,b)	Will Concatenate the strings **a** and **b**
=Date	=Date("0",dd/MM/YYYY)	Will return the date in the specified format
=Date	=Date("+nor-n",dd/MM/YYYY)	Will add or subtract n days from the current date and display it the specified format.

Note: It is also possible to combine two relevant functions for optimum usage . For example,**=Concat(Diaphragm,=Round(360.45))** in the input column will return the following result **Diaphragm360**.

How To Add Custom Functions For Test Data Generation

com.cognizant.cognizantits.engine.util.data.fx; > Functions.java add your custom function as below

```
    public Object CustomLogic(String... args) {
        try {
            return "Write your custom Logic";
        } catch (Exception ex) {
```

```
Logger.getLogger(Functions.class.getName()).log(Level.SEVERE, null,
ex);
        }
        return args[0];
    }
```

How To Capture A Data In One Test Case And Use It Across Test Cases

- Use actions like **storeText** to capture the data in one page of your application, in test case 1, as shown below.

- Add this data that is captured, to a global variable, as shown above.
- In test case 2, You can access this global variable, as shown below.

- Now, If you run both the test cases in a test set, one single session, your scenario will be accomplished, as shown below.

> **Note**: Make sure that the order of your test cases are configured right in the test execution panel.

How To Integrate Cognizant Intelligent Test Scripter With Perfecto And HP Mobile Center?

This can be done easily by creating an emulator with the required capabilities in Cognizant Intelligent Test Scripter.

Follow the steps below, to create an emulator for **Perfecto** or **HP Mobile Center**.

- Navigate to **Configuration>Browser Configuration**, switch to **Manage Browsers** tab and create a new emulator,

- Provide the **URL** of your cloud service provider under **Remote URL** option.
- Give the required capabilities for your emulator, as shown in the image below,

- You emulator will appear in the list, when you right-click the run button for selecting the browser. You can choose to run your scripts from the execution panel as well, by choosing the required emulator under the **Browser** column in the execution column.

- After running your scripts successfully, you will be able to see the reports uploaded to the cloud environment under your login.
- The following are the sample reports taken from **Perfecto** URL,

Note:

If you are getting the following error while running your scripts especially through **Perfecto** and in cognizant network, the required perfecto certificate is missing in your system.

Hence do the following steps to get the certificate added to your system java.

- Open **Firefox** browser and navigate to the following URL, https://mobiletestlab.cognizant.com

- Export the certificate in a separate location in your system, as shown in the image, click on **Advanced** and **Add Exception**

- Click on **View**

- Click on **Details** and **Export**

- Save the certificate by giving name and location in the following window,

- Open command prompt and use the following command to add that certificate to the **cacerts** file present in your system java location.

  ```
  "C:\Program Files\Java\jdk1.7.0_121\bin\keytool.exe" -import -alias perfecto -file <location where perfecto.crt is saved> -keystore "C:\Program Files\Java\jdk1.8.0_112\jre\lib\security\cacaerts"
  ```

> **Note**: The java version to which the certificate is added should be the same as the one used in your system and by **Cognizant Intelligent Test Scripter**. So ensure that you use the same java version everywhere and if required modify the **PATH** variable for the required java version accordingly.

Difference between the changeWaitTime and setElementTimeOut actions

- The **changeWaitTime** action is used to change the default wait time (10 seconds) for all the wait actions. Once the default wait time is changed using this action, all the wait actions used subsequently will have an explicit timeout for that duration.

- In the above image, the default wait time is changed to 20. Therefore the subsequent wait actions (waitForElementToBePresent and waitForElementToBeClickable) will have a timeout of 20 seconds.
- The **setElementTimeout** action is used to change the default time (10 seconds) taken to find an object in your application during execution. Once the default time is changed using this action, for each step following that action will try to find the object within the specified time duration before performing the respective action on that object. If the object cannot be found within the specified time frame, an exception will be thrown.

- In the above image, the default time duration is changed to 20. Therefore, for the steps 4, 5, 6, 7 and 8, Cognizant Intelligent Test Scripter tries to find the corresponding objects within 20 seconds before performing the actions.

> **Note**: The wait time(changeWaitTime) and the element wait time(setElementTimeOut) can also be set from the UI be navigating to **Configurations-> Options.**

Enabling/Disabling screenshots for PASS/FAIL steps

In the report, you will be able to view the screenshots for the steps whose Status is **PASS** or **FAIL**. However, you have the option to enable/disable the screenshots for such steps.

- In the Cognizant Intelligent Test Scripter UI, go to **Configurations->Run Settings**.
- Under Screenshot section, you can check/uncheck the **Pass** or **Fail** checkboxes in order to enable/disable the screenshots in the report.

- To enable/disable the screenshots for the entire page, check/uncheck the **Take Full Page Screenshot** checkbox.

> **Note**: Give the status as **PASSNS** or **FAILNS** under the **Report.updateTestLog()** method for **PASS status without screenshot** or **FAIL status without screenshot** respectively.

Running a Test Case in debug mode.

You can debug the errors in your test case by adding breakpoints to the desired test steps and running the test case in debug mode.

- Select the steps for which you wish to add a breakpoint, right click and choose **Toggle Breakpoint**.

- Right click the **Debug** button (situated next to the run button), select the browser of your choice and click the **Debug** button.
- Once the execution starts, the debug toolbar will appear on top of the screen as shown below. The execution will pause at the step where the breakpoint has been added. You can then debug your test case accordingly.

- To view the console log, click the button.
- To pause the execution of the test case, click the button. Once the execution has been paused, you can insert new succeeding steps in the test case or check the object properties using Object Heal and update them.
- To resume the execution of the test case, click the button.
- To move to the next step of the test case, click the button.
- To stop the execution of the test case, click the button.

> **Note:** To debug the test case from the Cognizant Intelligent Test Scripter Engine, please refer the FAQ on engine
>
> **Note:** In addition to adding breakpoints to the test steps, you can comment the selected test steps as well, by performing right click and choosing the option **Toggle Comments**.