**Advanced VuGen Tutorial – Transactions, Rendezvous points, Parameters etc**

We have seen how the VuGen component can be used to generate user scripts based on business scenarios. In this **advanced VuGen tutorial**, we will see how we can change the generated VUGEN scripts. These changes are required every now and then to make the script more dynamic in nature.

A script normally just records a pre-defined scenario. But if you just keep on running that script with that pre-defined scenario, it will not be a realistic scenario. In a real time scenario, you might have users which enter different sorts of data against an application. Our script should be able emulate such use case scenario’s so that the test is more or less close to the real time scenario.

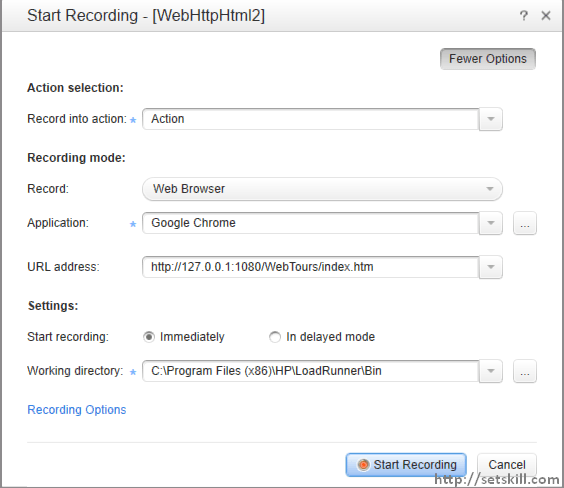
In this tutorial, you will learn about:

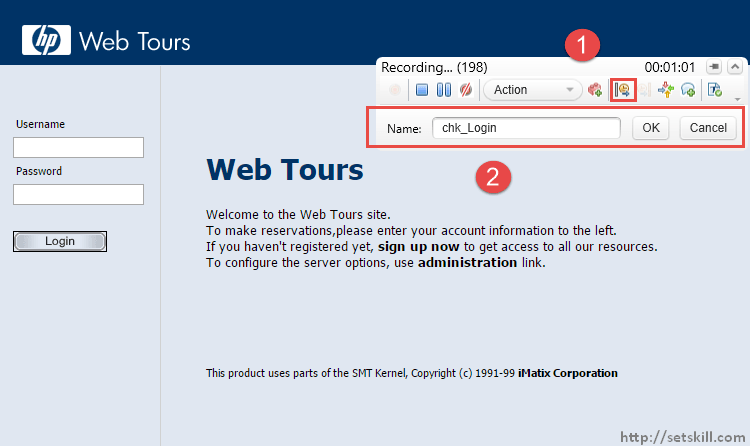
1. [How to add Transactions](http://setskill.com/hp-loadrunner/advanced-vugen-tutorial-transactions-rendezvous-points-parameters-etc/#How_to_add_Transactions)
2. [How to insert Rendezvous points](http://setskill.com/hp-loadrunner/advanced-vugen-tutorial-transactions-rendezvous-points-parameters-etc/#How_to_insert_Rendezvous_points)
3. [How to add Comments](http://setskill.com/hp-loadrunner/advanced-vugen-tutorial-transactions-rendezvous-points-parameters-etc/#How_to_add_Comments)
4. [How to add Parameters](http://setskill.com/hp-loadrunner/advanced-vugen-tutorial-transactions-rendezvous-points-parameters-etc/#How_to_add_Parameters)
5. [How to configure Runtime settings](http://setskill.com/hp-loadrunner/advanced-vugen-tutorial-transactions-rendezvous-points-parameters-etc/#How_to_configure_Runtime_settings)

**How to add Transactions**

As we have seen in an earlier section, a transaction is used to logically separate the different parts of a business scenario. For example, for the login of an application, you can define a separate transaction and similarly for the logout for the application. Let’s again see how we can achieve this through the VUGEN tool.

Step 1) Open the VUGEN tool and choose to create a new script.

Step 2) Begin the recording process with the Web Tours application

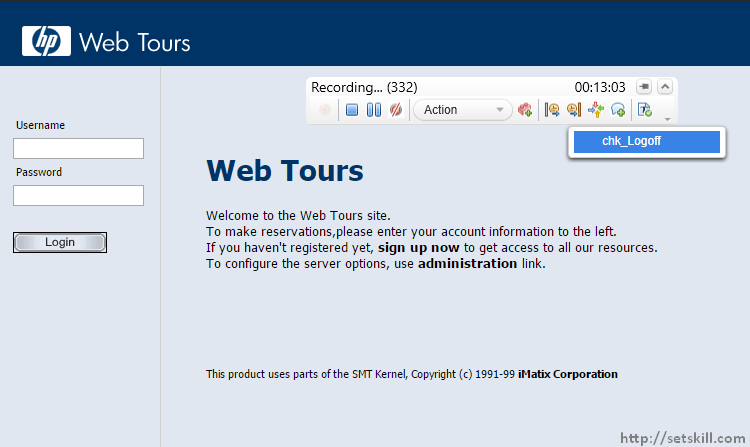
Step 3) After you start recording, before you actually log into the Web Tours application, click the button to add a transaction.

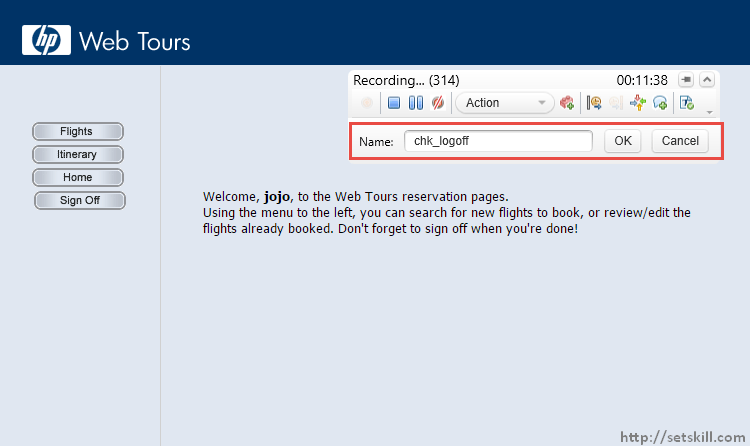
Enter the transaction name as chk\_Login and click on the OK button.

Step 4) Now log onto the application with the same user credentials of

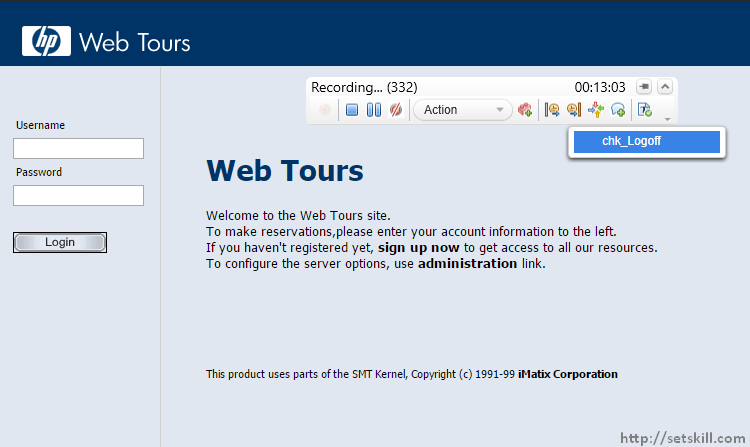
User name – jojo

Password – bean

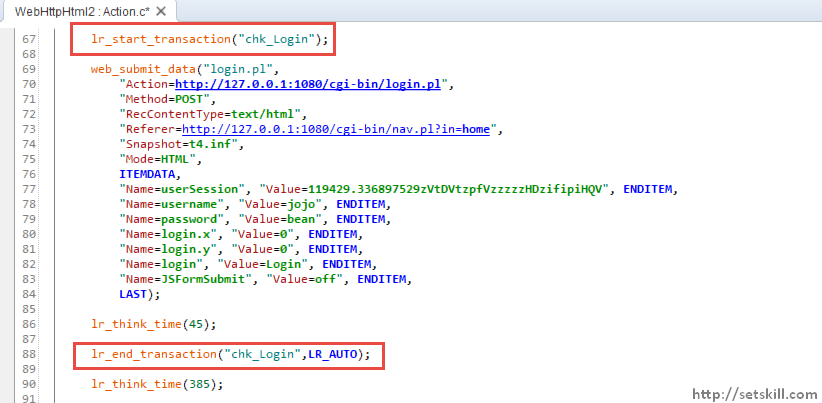
Once you have logged into the application, you can now choose to close the transaction by clicking the Close Transaction button and choosing the chk\_Login transaction name which was entered earlier.

Step 5) Now log off from the Web Tours application. But before we do this, let’s again create a transaction checkpoint for the log out phase.

Once we enter the logoff checkpoint, let’s actually log out from the application.

Step 6) Once we have logged out of the application, let’s close the checkpoint of chk\_Logoff.

Let’s now stop the recording and see what was generated as part of the script.

In the script we will notice that there is a lr\_start\_transction and lr\_end\_transaction statement added to the script.

The lr\_start\_transction statement is used to indicate the beginning of a transaction. In this transaction, we are actually performing the login to the application. The end of the transaction is marked by the lr\_end\_transaction statement.

The transaction statements have the name of the transaction which in this case was chk\_Login. In the end transaction statement there is an extra parameter passed which signifies on whether the transaction passed or failed. The possible values are

* LR\_AUTO
* LR\_PASS
* LR\_FAIL

Please note the below mentioned points when it comes to name transactions

1. All transaction names are case-sensitive, since the entire code for the VUGEN script is in C language.
2. The default status entered by loadrunner is LR\_AUTO, so Loadrunner will decide on how to handle errors. If you want to write custom code to handle error or success conditions, then modify the status to LR\_PASS or LR\_FAIL accordingly.
3. Remember to close the transaction after creating one, else you will get errors in your script.

**How to insert Rendezvous points**

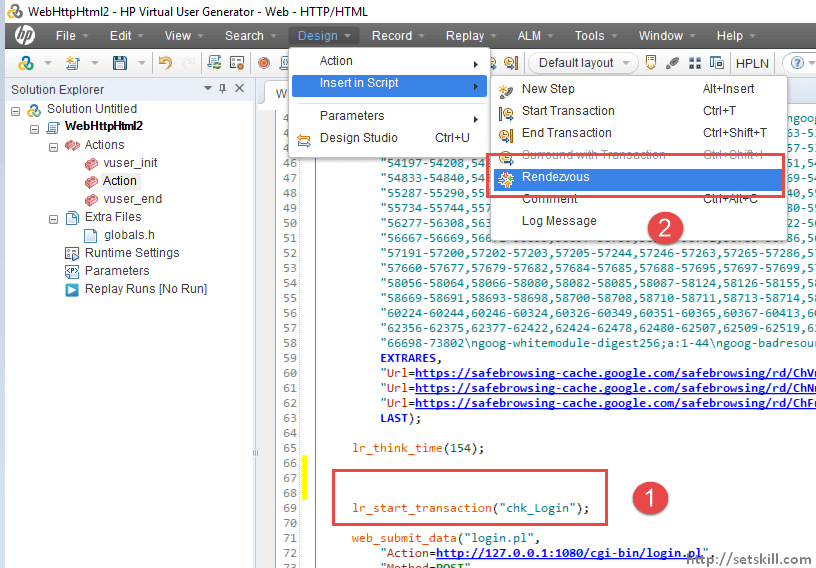
By default when many users are simulated with a particular business scenario, they could be executing different parts of the scenario at different points in time.

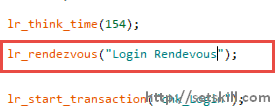
So in our Web tours application, you can have 3 users who are logging into the application at the 1stminute. Then the next 3 users could be logging in at the next minute.

But supposed if you really wanted to load test your application and see what would happen if all of the 6 users were to login at the same time into your application. How can you simulate this? The answer is, by using Rendezvous points. These points will instruct all the VUGEN users to converge at a point and wait to perform an action at the same time.

To insert a rendezvous point, perform the following steps

Step 1) After the script is generated, first decide where to add the rendezvous point. For our previously generated script, we can insert a rendezvous point just before the login checkpoint. So this would instruct all the users to log into the application at the same point in time.

Then click on the menu option Design > Insert in Script > Rendezvous.

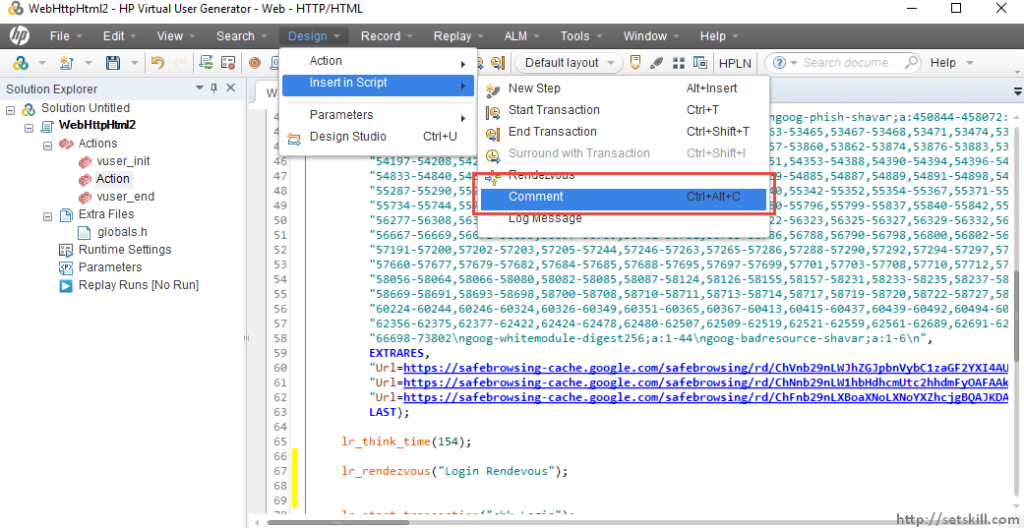
Step 2) Once the Rendezvous point has been added to the script, give a name to the point.

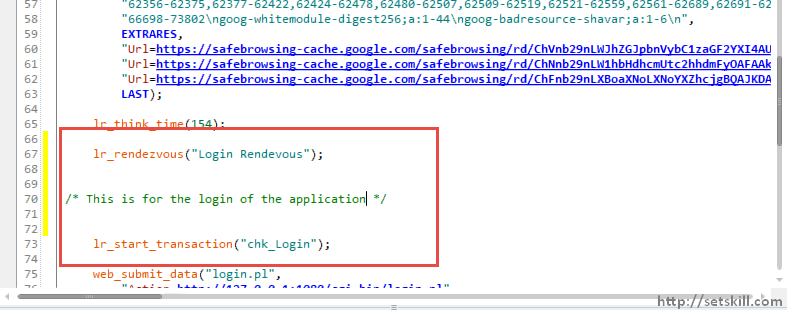
You now have a successful Rendevous point in your script.

**How to add Comments**

Comments can be added to the script to make it more readable for testers. This is a good coding practice to keep on adding comments to the script. To add a comment in the script, perform the following steps

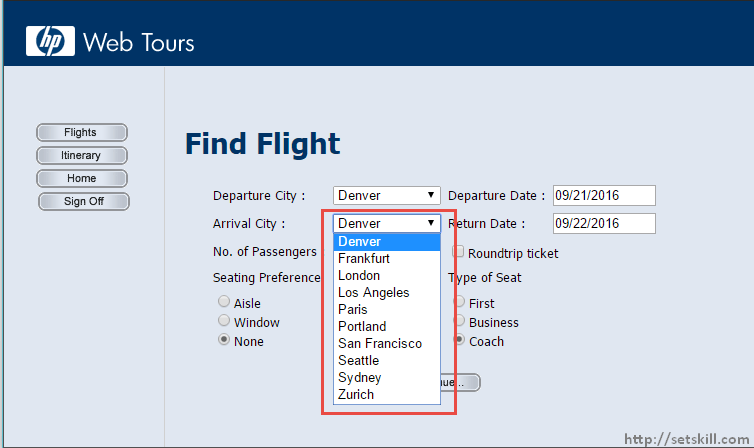
Step 1) After the script is generated, first decide where to add the comments. For our previously generated script, we can insert a comment just before the login checkpoint.

Then click on the menu option Design > Insert in Script > Comment.

Once the comment statement has been added, enter the required comments as shown below.

**How to add Parameters**

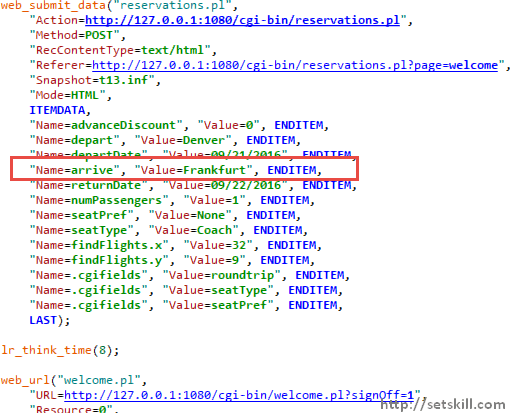
Adding parameters is an important part in any script. Parameters are used to make the script more dynamic in nature. Let’s take an example of our Webtours application.

Now during the recording of our business scenario, let’s say we entered Frankfurt as our destination to find for a flight.

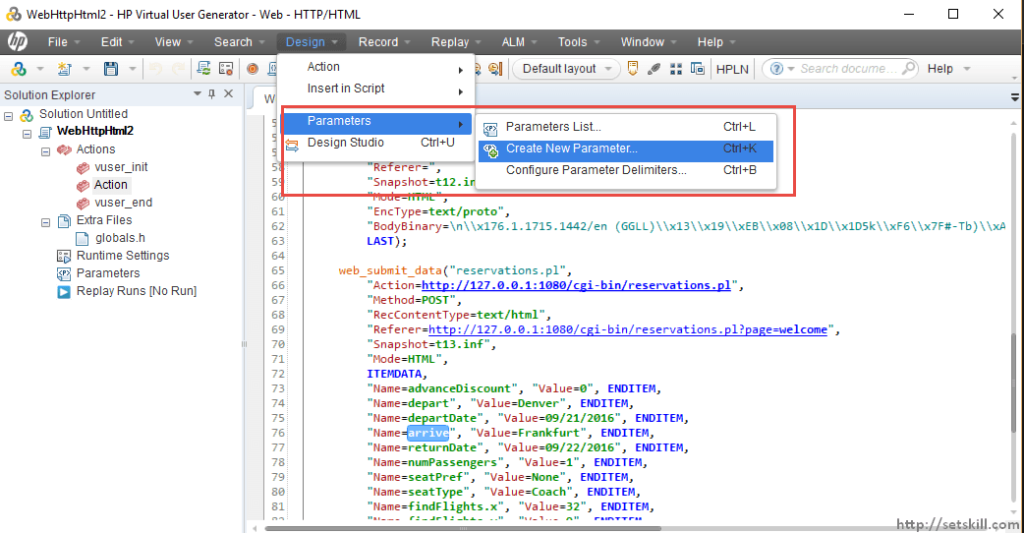
But in a real time scenario, a user could be choosing any flight from the drop down list to find for a flight. So how can we simulate different flights for our use case scenario? We do this by making the Flight as a parameter in our script. And then that parameter will pick up the flight data dynamically to simulate this real time scenario. So let’s follow the below mentioned steps to have this in place.

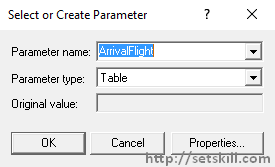
Step 1) Let’s first record our script with the following business scenario

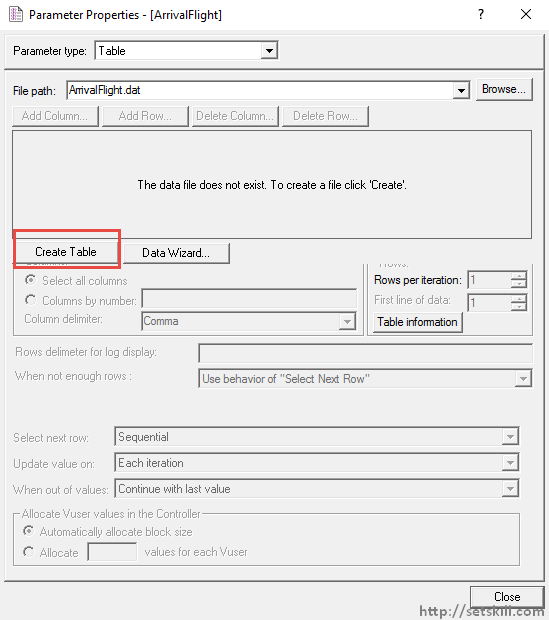
1. Login into our Web Tours application
2. Choose the Flights menu option
3. Choose the Arrival Airport as Frankfurt and Find for flights
4. After getting the desired result, log off from the application.
5. Stop the scripting process.

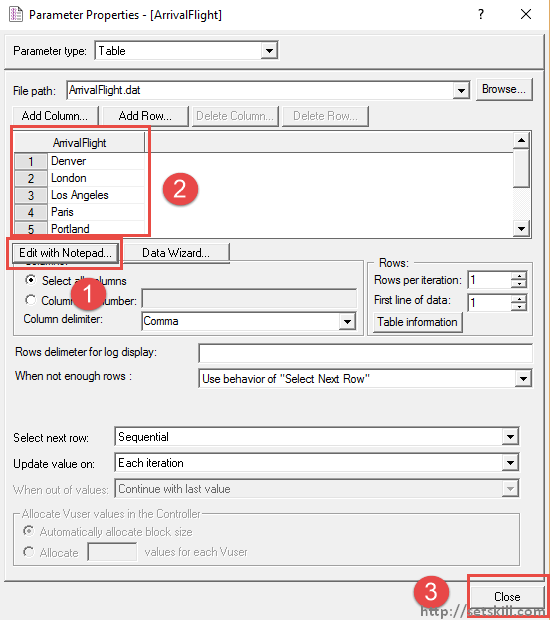
If you look into the script, you will see an ITEMDATA name of arrive having a value of “Frankfurt”. This is the place where the Flight information is being taken for the search criteria.

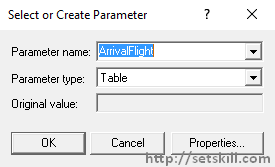
Now it’s time to parameterize this value so that it can dynamically be picked up. Let’s perform the below mentioned steps to have this in place.

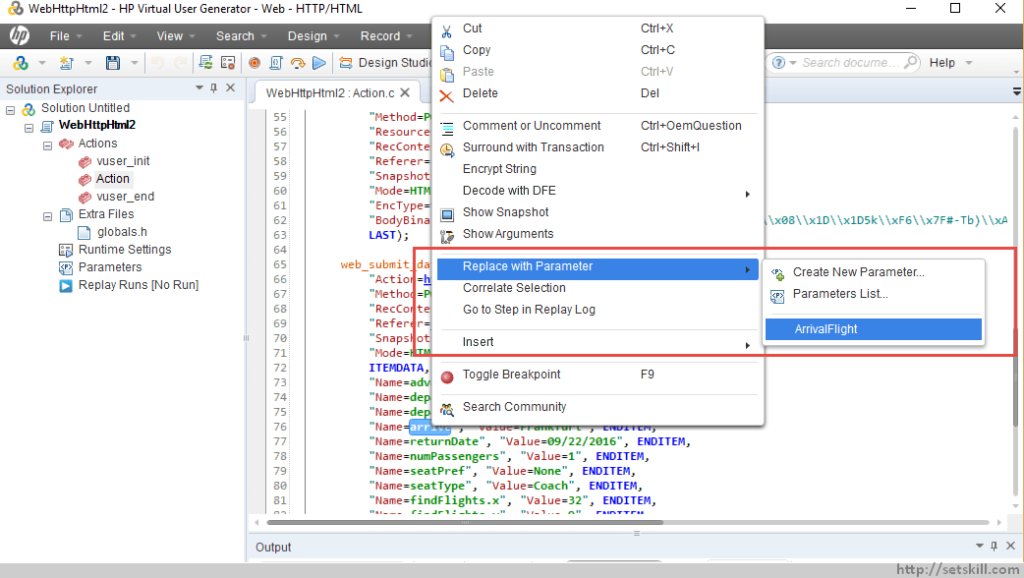
Step 1) Creating the dataset. This dataset will contain the flight values which will be replaced in the script with the help of the parameter. In the editor, choose the menu option of Parameters->Create New Parameter.

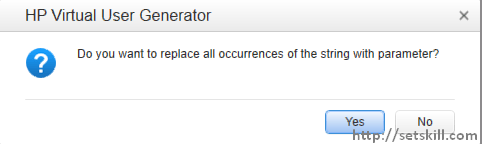
Step 2) You will then be presented with a Create Parameter Dialog box. Just give a parameter name of ArrivalFlight. Ensure the Parameter type is set to ‘Table’. Then click on the Properties button.

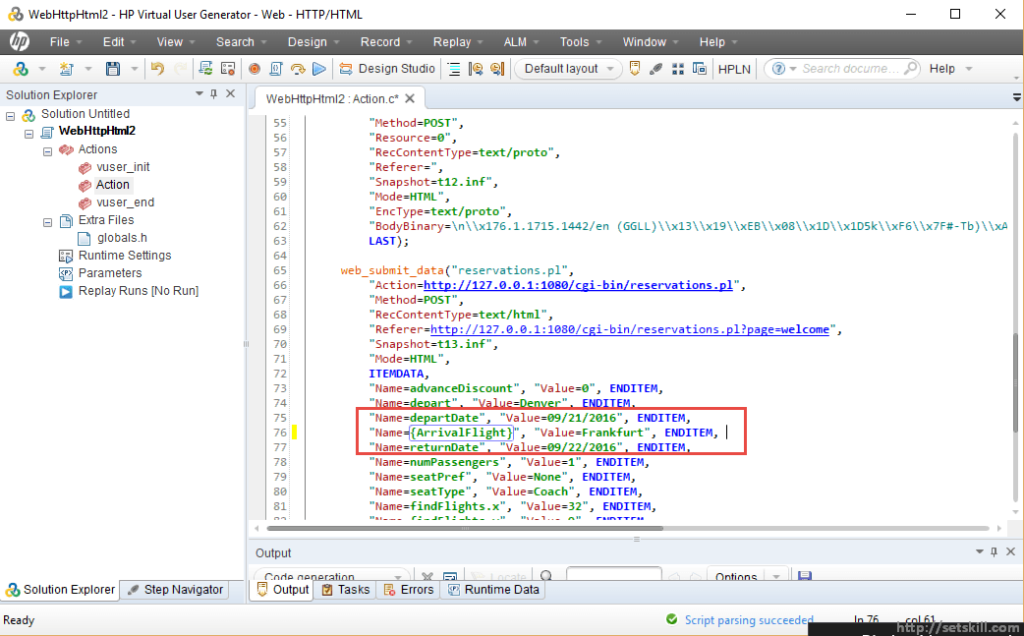
Step 3) Once you click on the Properties button in the previous screen, you will get the dialog box shown below. This will assist in creation of the test data. Click on the Create table button.

Step 4) Once you click on the Create table button, click on the ‘Edit with Notepad’ button. Then enter all the possible flight values such as Denver, London,and Los Angeles etc.

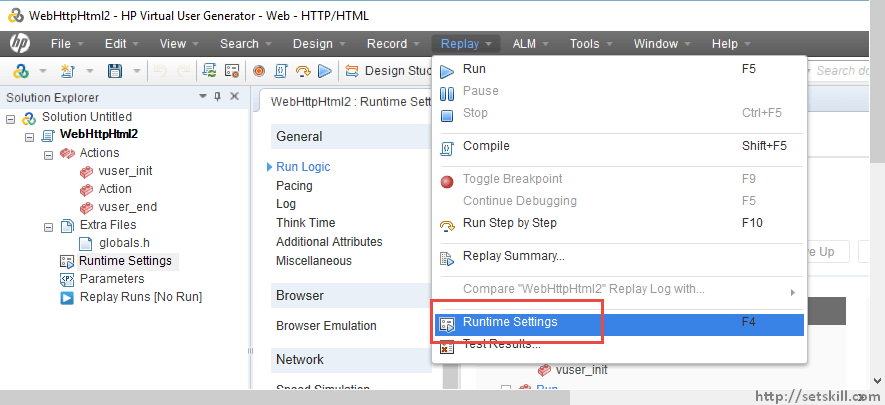
Once done, you can click on the Close button at the bottom of the screen. You will again be presented with the below ‘Create Parameter’ dialog box. Just click on the OK button.

Step 5) Now go to the script, highlight the ‘arrive’ name and right click and choose the menu option of Replace with Parameter->Arrival Flight.

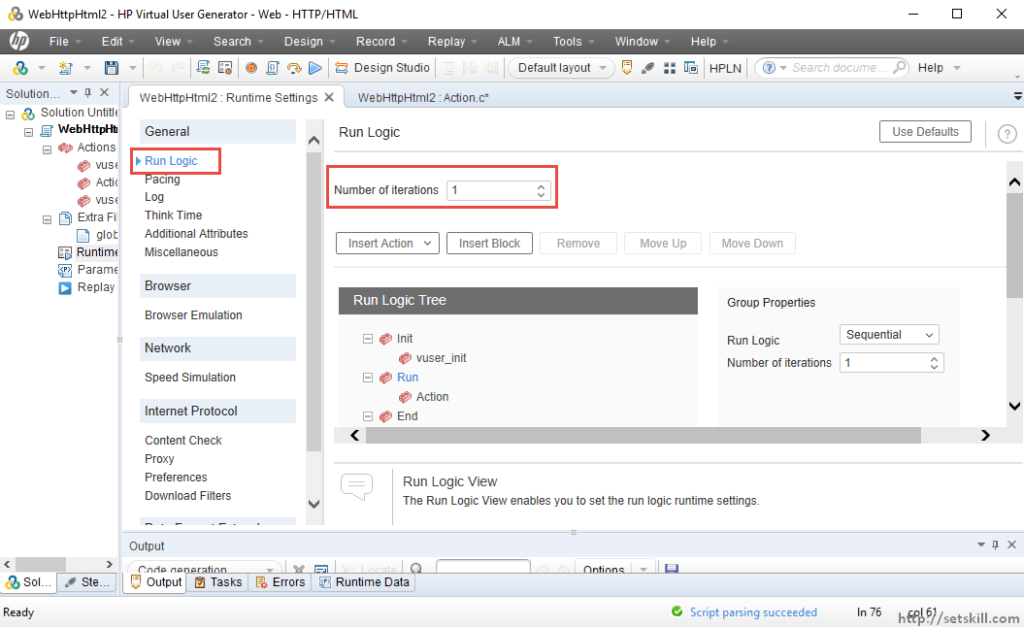
As a test team member, it is important to know what is the entire process involved when working with LoadRunner scripts. Below is the normal process followed when developing any load testing script. You will then be prompted on whether you want the replacement to occur. Click on the ‘Yes’ button.

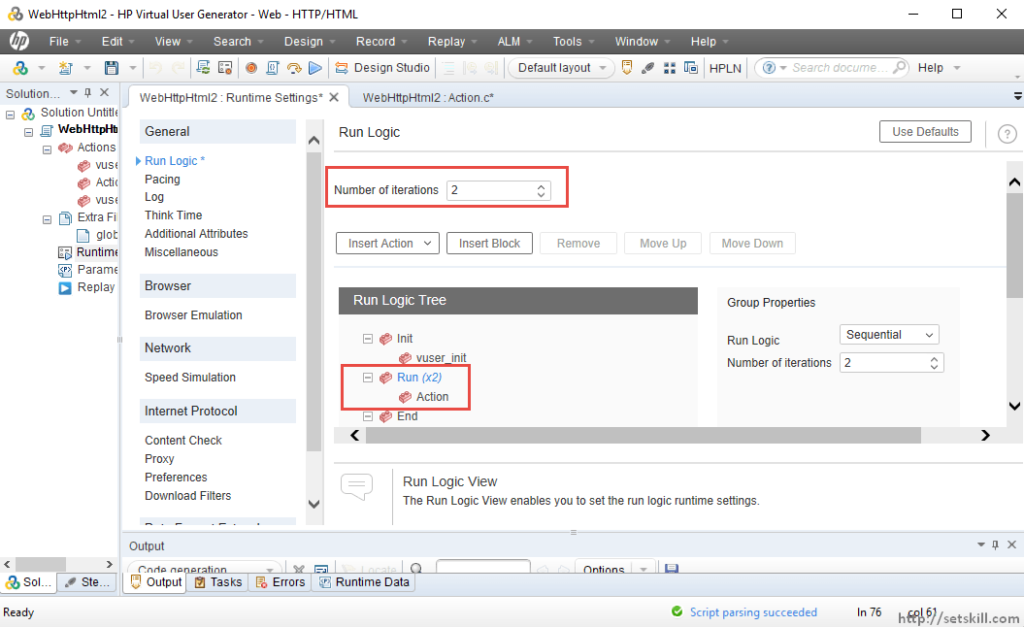
Once done, you will now see the parameter has been replaced in the script. Now whenever the script runs, for each test run, it will pick up the values from the parameter table.

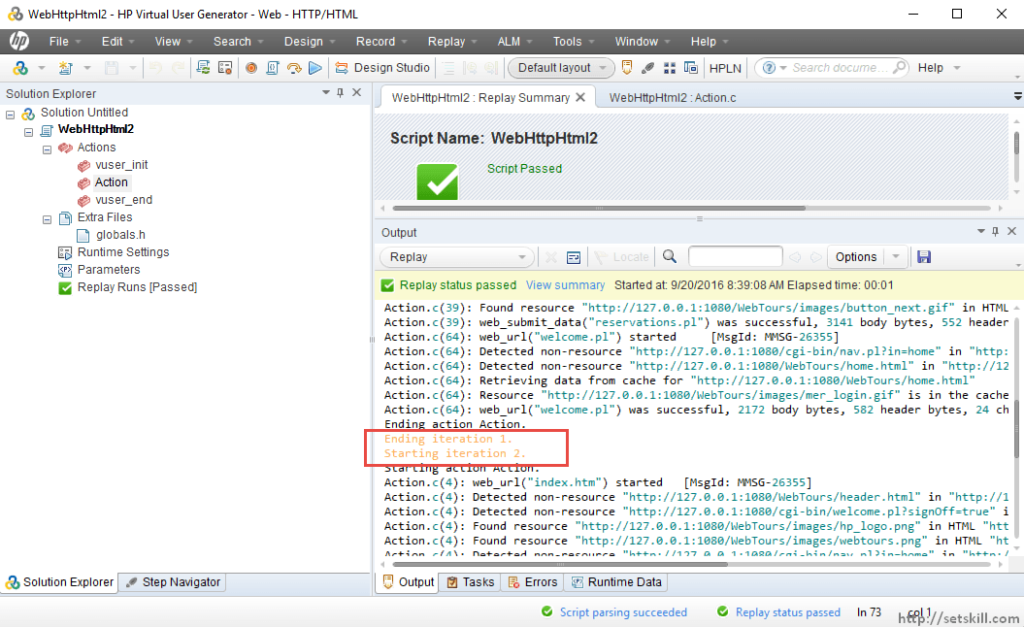
**How to configure Runtime settings**

Loadrunner has a couple of settings when it comes to running VUGEN scripts. Let’s discuss below the different Run settings available in VUGEN. To access the VUGEN Run settings, choose the menu option of Replay->Runtime settings.

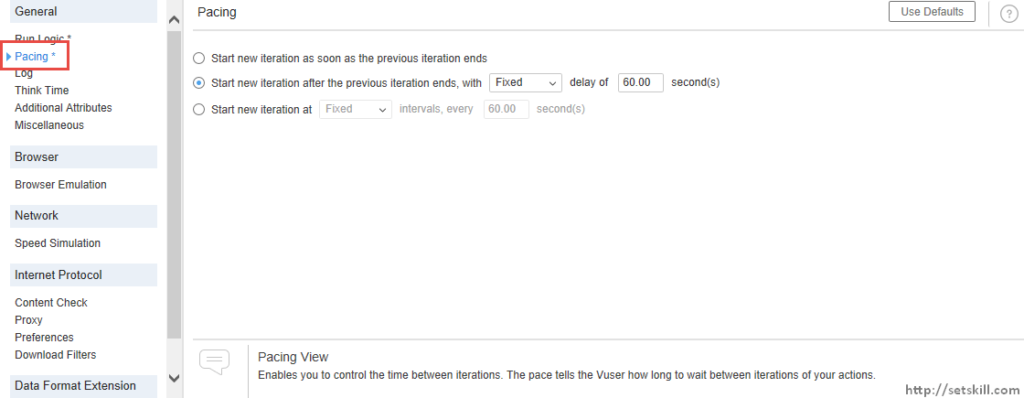
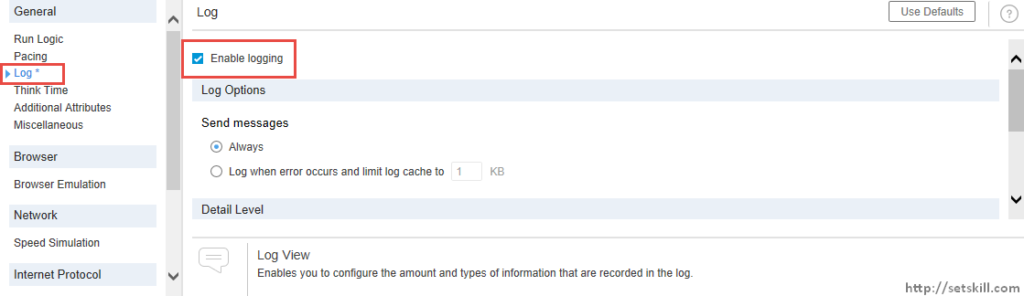
You will then be presented with the following run time settings.

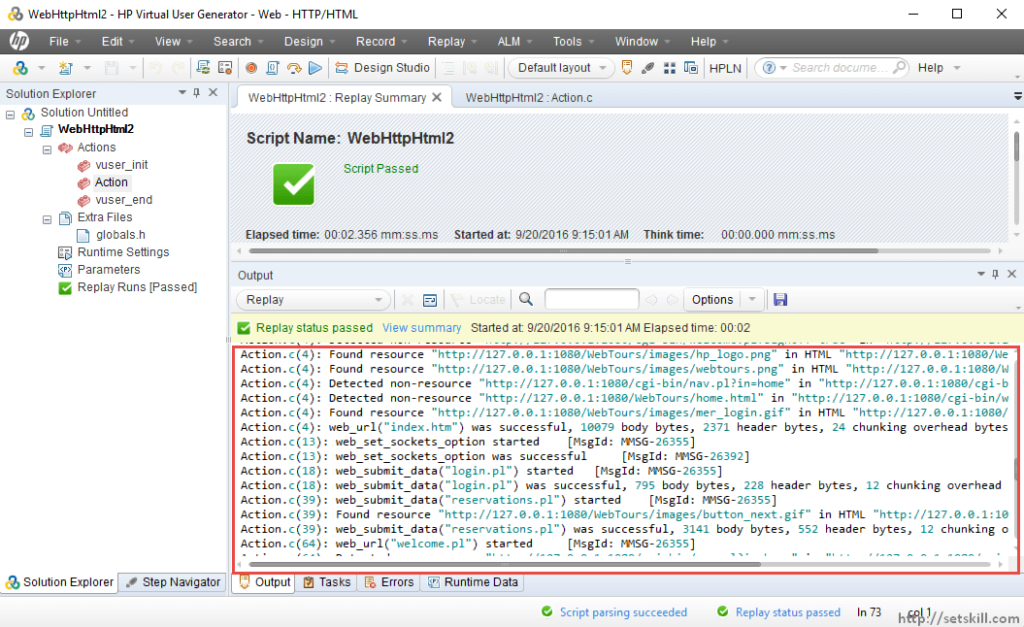
* **Run Logic** – In the Run Logic section, you can decide the number of iterations that need to run for each method in your VUGEN script.

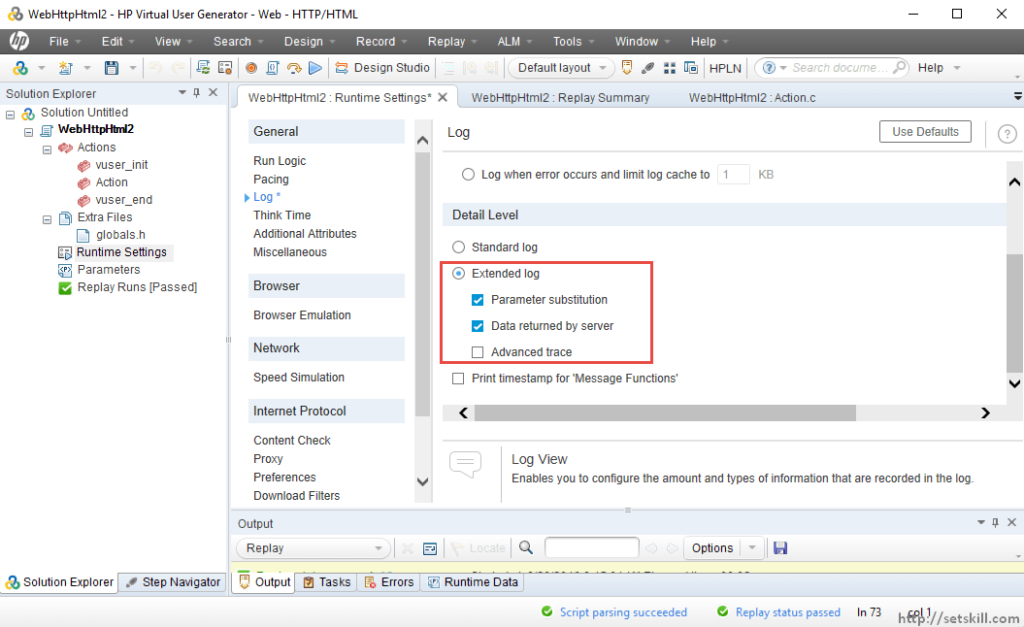
If you change the Number of iterations to 2, you will see that the Run method will now have the number of iterations of 2.

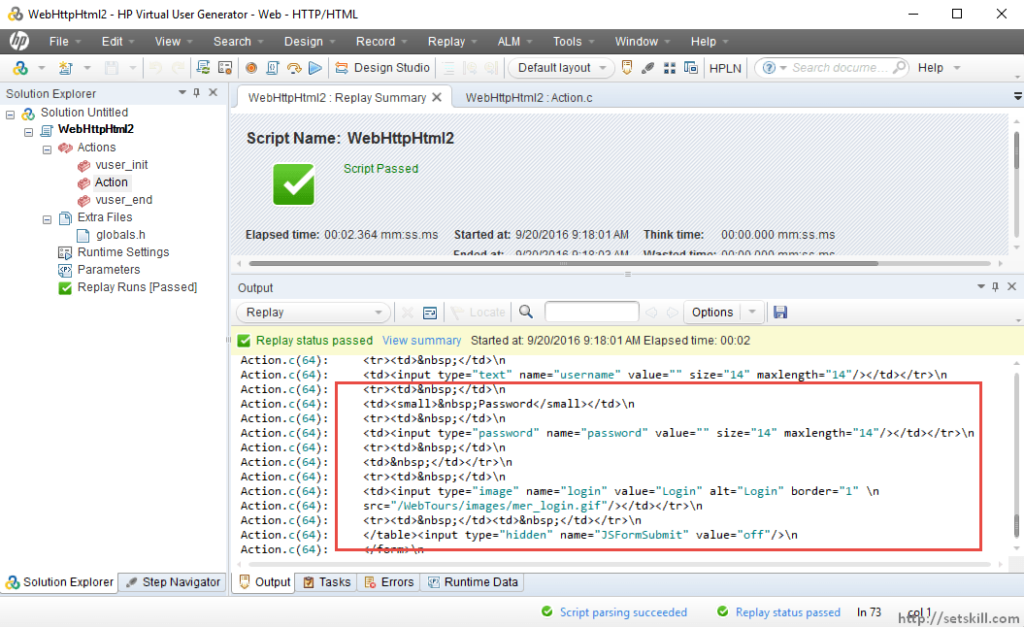
When you run you script and see the script replay log, you will then see that the script will run twice because of this setting.

You can also add more Action methods and more blocks if required. This is only if you wish to logically segregate the script for maintenance purposes. For example you could create a separate Action method for say the Find Flight method only.

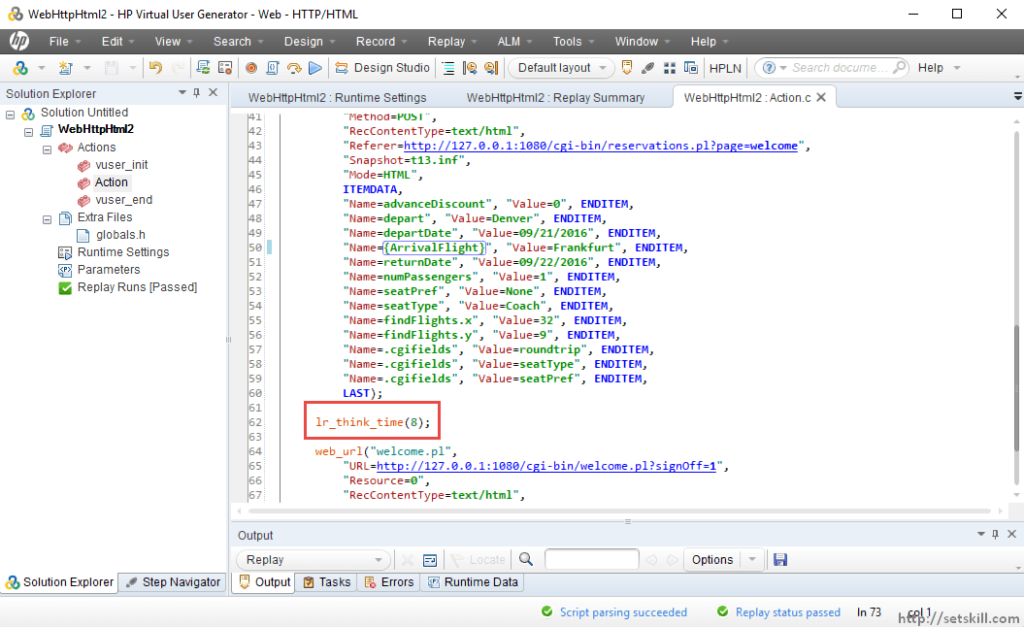
* **Pacing** – The pacing setting allows one to decide the amount of pacing between transactions. In a real time scenario, sometime it is not a case wherein one transaction runs directly after another. There is normally a gap or pause in between. The pacing settings allows one to add that gap or pause between transactions.
* **Log** – This gives you the ability to control the level of logging for scripts in the LoadRunner editor.

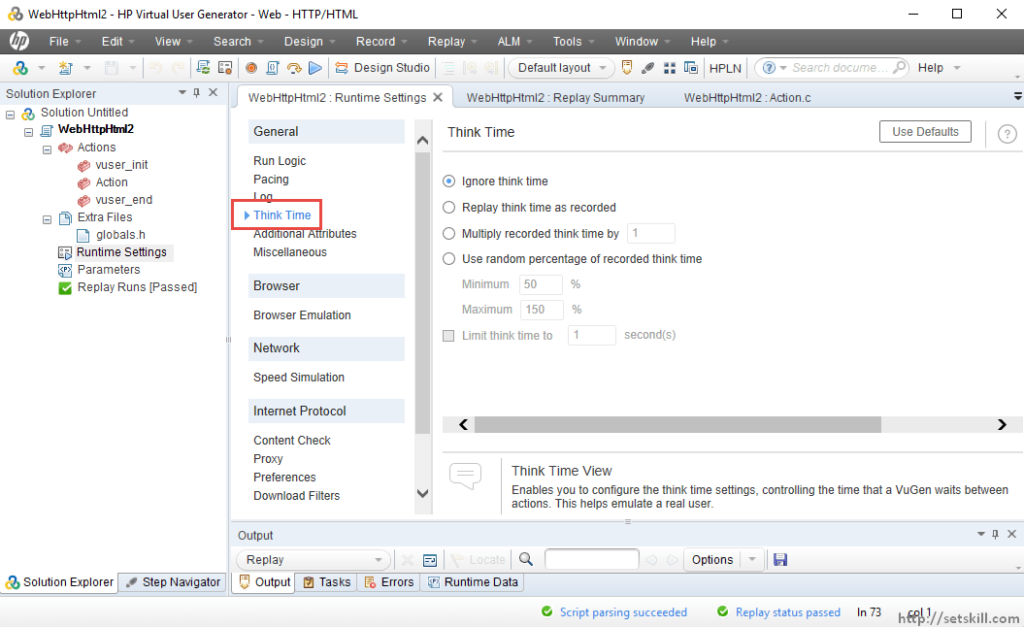
When logging is enabled and you run the scripts, you will be able to see the results in the Output window. If logging is disabled you will not be able to see any results in the Output window.

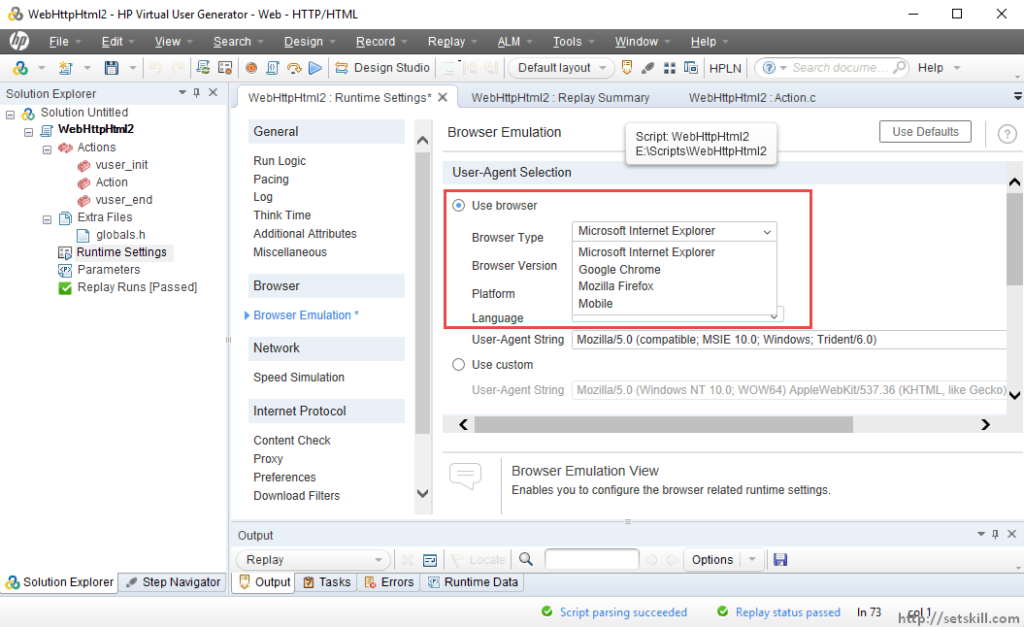
You also have the facility to enable advanced logging by enabling extended log.

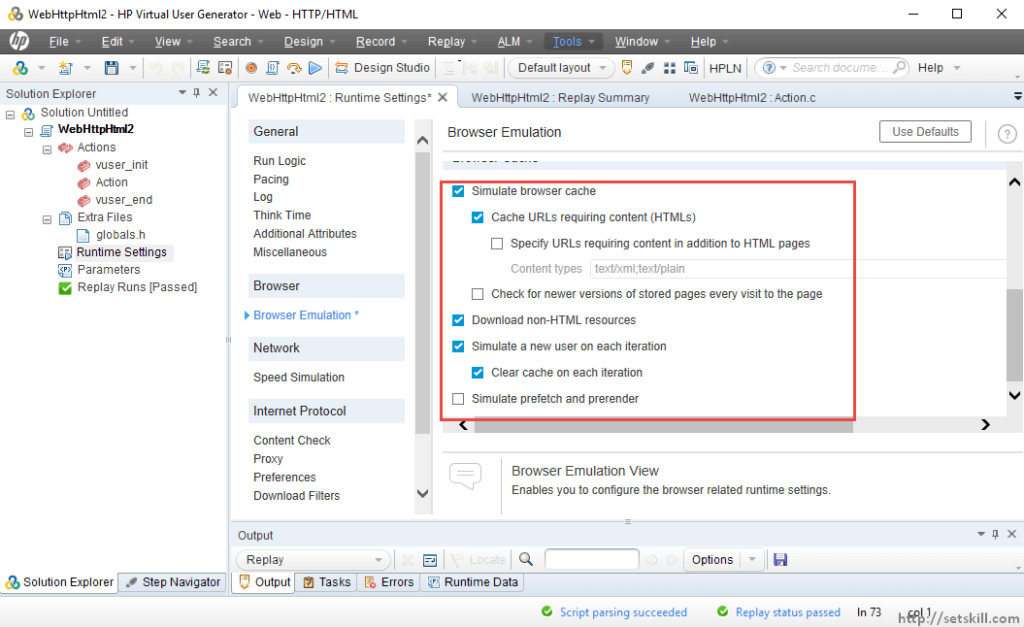
So for example, if you enable the option to return data from the server, when you run your script, the output window will contain all the data that is returned from the server. Here you can see the actual HTML code which is returned by the server.

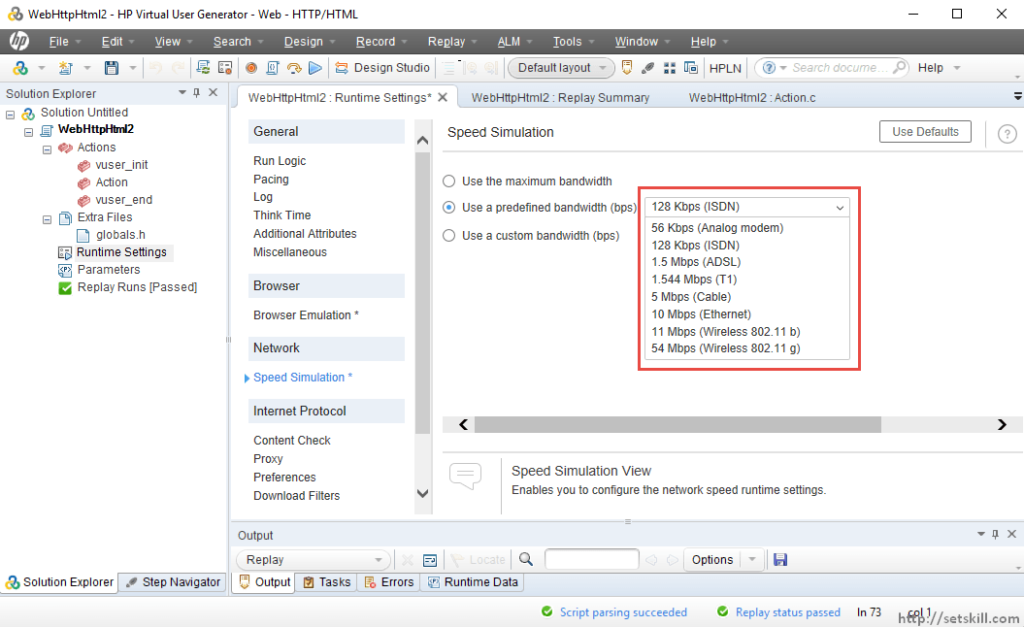
* **Think time**– Think time is the time induced in the script for any sort of delay which is normally incurred by the user. For example, when a user is deciding which flight option to choose, there might be some time before he can click the next button and move onto the next screen. To simulate this user scenario, there is a concept known as think time or delay that is inserted in the script.

You will see these think times inserted in the script automatically by Load runner when the script is recorded.

The run settings have different options on how the think time should be handled. There are different options available on how think time should be handled in the script.

* **Browser Emulation** – In the browser emulation setting, you actually have the facility to choose the type of browser to use for emulation when the script is run.

You also have advanced settings in which you can control how the browser should work. For example, the cache settings ensure that results are cached just as browser’s do in a real time scenario.

* **Network Simulation** – This is an important setting which helps simulate transactions over low bandwidth settings. Suppose your company has a low bandwidth network connection and you wanted to know how the application would perform under this circumstance, this is the place where you can choose the required bandwidth settings to simulate.
* **Proxy settings**– The proxy settings can be configured if in case you’re company uses a custom network or proxy server to route connections to your application or to the internet. Here you have the option to set the options accordingly.