Transactions, Rendezvous and Checkpoints

1. **Transactions** – To measure rate and response time of the user actions.
2. **Text and Image checks** – To ensure that the response returned is correct.
3. **Rendezvous points** – To make Vusers perform a task simultaneously.

# Transactions

The mere replay of performance scripts will only put a load on the application but does not measure anything. The end goal of any performance script is to inject load and measure the rate and response time of user actions on the application.

**For Example**, how much time a product search took when 100 users are on the application

To measure these, we have to insert transactions in a VuGen script (after the test, we can see the rate and response time of these transactions in the results).

‘A transaction is an end to end measurement of a user action (or a group of user actions) on an application’.

**Transactions can be inserted in two ways:**

1. During Recording
2. After Recording (where we manually insert the transaction functions in the script)

## Steps to insert a transaction

1. Start recording
2. Click on Insert start transaction
3. VuGen prompts for transaction name. Enter any meaningful name.
4. Once the flow is recorded, click on insert end transaction

Stop the recording and see the generated script. We can see that transaction start and transaction end functions are inserted into the script and all the request(s) corresponding to the recorded user action are enclosed in between these functions.

The syntax of the ‘transaction start’ and ‘transaction end’ functions is pretty simple. Both have transaction name as the first argument. The ‘end transaction’ function has one more attribute – ‘LR\_AUTO’, this instructs VuGen to Pass or Fail a transaction automatically (instead of us explicitly specifying a condition for a pass or fail).

If the load runner encounters any errors in executing requests within a transaction then it fails the transaction otherwise it passes the transaction.

* Wasted time is the time taken to execute the transaction lines.
* Transactions can also be added to existing script using Design 🡪 Insert 🡪Start/End Transactions.

# Text And Image Checkpoints

## Text Check

Text checks are used to verify if the response returned for the request is correct or not. In the ‘Correlation’ tutorial, we saw that sometimes VuGen may not show any error but the returned response may not be the correct one. So, to ensure that the response is correct, we use text checks.

For Example, when we login into the web Tours application, we go to home page. Let’s verify in the script that we are getting the correct response (homepage) or not when we log in.

To do this, we have to identify some text in the response (homepage here) which can say if the response is correct or not. ‘Welcome’ is one such text.

For text check, we use ‘**web\_reg\_find’** function just before the request.

**The syntax of this function is:**

web\_reg\_find(“Text=”,”SaveCount=”,”Fail=”,LAST);

Example:

web\_reg\_find("Text=**Dogs**","Fail=**NotFound**",**LAST**);

‘Text’- Used to specify what text to search for.

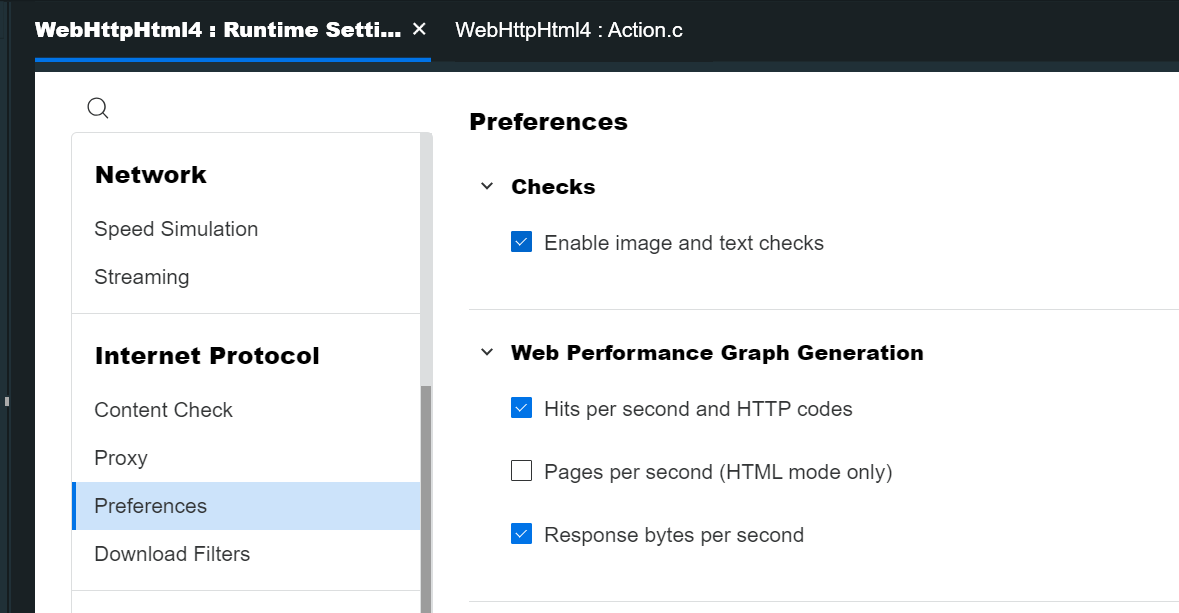
‘SaveCount’ – Saves the number of occurrences of the specified text into a parameter that we specify.

‘Fail’ – Takes one of the two values -‘Found’ and ‘NotFound’ and fails the script accordingly.

So the function, web\_reg\_find(“Text=Welcome”,”SaveCount=WelcomeCount”,”Fail=NotFound”, LAST) searches for the text ‘Welcome’ in the response and saves the number of occurrences into the parameter ‘WelcomeCount’ (and fails the script if the text is not found).

## Image Check

In order to perform Image check, we need to enable the option in Runtime settings.



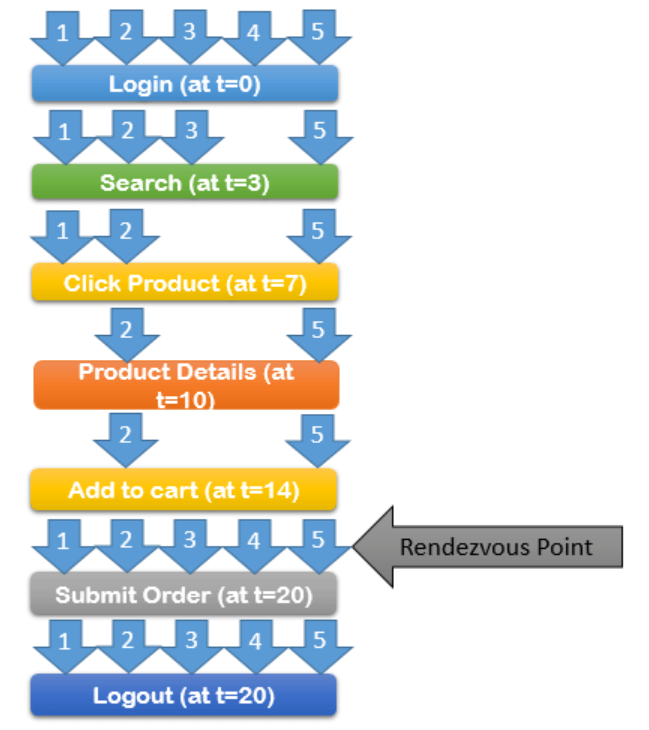
Add below line for Image check

web\_image\_check("web\_image\_check", "Src=**../images/splash.gif**",**LAST**);

# Rendezvous Point

In LoadRunner, Rendezvous point is used to force the Vusers to perform the simultaneous task during the test execution. It generates intense user load on the server for a particular functionality/page and instructs LoadRunner to measure server performance under such situation. Rendezvous point instructs Vusers to wait during test execution for multiple Vusers to arrive at a certain point, in order that they may simultaneously perform a task.

Suppose you want to measure how an online shopping portal application performs when five Vusers submit the order of a product simultaneously. To emulate the required user load on the server, you instruct all the Vusers to halt before ‘Submit Order’ transaction. Once all five Vusers arrived, LoadRunner will release them simultaneously i.e. exactly at the same time.



# LoadRunner Utility Functions

Following is the list of important LoadRunner Utility Functions which are frequently used in the script.

**1. lr\_abort:** This function stops the execution. If any specific error comes and Vuser needs to be stopped then lr\_abort is used.

**2. lr\_advance\_param:** It returns the next value available in the parameter list. Once this function is called, the next available value in the parameter list is passed in the next occurrence of the parameter. It is not suitable for Table type parameters (having multiple values in a row).

**3. lr\_continue\_on\_error:** This function is used when the runtime setting option “Continue on Error” needs to be overridden for a specific transaction/code. The input arguments are 0, 1, 2, 3 and 4.

**4. lr\_convert\_string\_encoding:** To convert the string into a specific format like utf-8 and ucs-2.

**5. lr\_convert\_double\_to\_integer:** To convert the string value of double datatype into integer datatype.

**6. lr\_db\_connect:** To connect with the database.

**7. lr\_db\_disconnect:** To disconnect from the database.

**8. lr\_db\_getValue:** To retrieve a value from a dataset.

**9. lr\_db\_executeSQLStatement:** To execute the SQL statement.

**10. lr\_debug\_message:** Instruct LR to write the debug message to the log file.

**11. lr\_enable\_ip\_spoofing:** To enable IP spoofing.

**12. lr\_start\_transaction:** To start a new transaction.

**13. lr\_end\_transaction:** To end a started (open) transaction.

**14. lr\_start\_sub\_transaction:** To start a new sub-transaction under a main transaction

**15. lr\_end\_sub\_transaction:** Call this function to end a started (open) sub-transaction before ending the main transaction.

**16. lr\_stop\_transaction:** To stop a transaction.

**17. lr\_resume\_transaction:** To re-start the transaction. It is used after the lr\_stop\_transaction function only.

**18. lr\_start\_timer:** To start a timer.

**19. lr\_end\_timer:** To end a started timer.

**20. lr\_error\_message:** To write an error message in the output window or log file.

**22. lr\_exit:** To exit the Vuser from a script, iteration and action.

**23. lr\_get\_host\_name:** To return the name of the machine on which the script is executing.

**24. lr\_get\_transaction\_duration:** This function provides the duration of the transaction. Using transaction duration another operation can be performed.

**25. lr\_get\_transaction\_status:** It is as similar as lr\_get\_transaction\_duration function. The only difference is it returns the transaction status.

**26. lr\_get\_transaction\_wasted\_time:** To get the elapsed time for running the logic inside a transaction.

**27. lr\_get\_vuser\_ip:** To get the Vuser IP address. It is helpful when IP spoofing is enabled.

**28. lr\_log\_message:** Instruct LR to write logs in a file.

**29. lr\_message:** To send the message to the output window as well as the log file.

**30. lr\_next\_row:** It advances the value of all the parameters listed in a row.

**31. lr\_output\_message:** To print the message in the log file and output window. It is specially used for debugging purposes to verify the parameter value.

**32. lr\_paramarr\_random:** It is used to pick a random value from the list (array) of correlation parameter values.

**33. lr\_param\_unique:** To generate a unique string and save it in a parameter that can be accessed through its name.

**34. lr\_rendezvous:** To insert a rendezvous point into the script where a desired number of Vusers halted and released in a group.

**35. lr\_save\_datetime:** To get the current date and time in a required format. The value is saved into a parameter. An offset can also be used to generate past or future dates and times.

**36. lr\_save\_timestamp:** To get the current timestamp in epoch format.

**37. lr\_save\_string:** To save a string into a parameter.

**38. lr\_think\_time:** To insert a wait time or delay. The LR pauses for the given time and then resumes the execution.

**39. lr\_whoami:** This function gives the information of the Vuser.