MODULE 5 QB

-PP

1) Explain Try catch activity for handling exceptions.

Ans)

- Exception handling is a way to handle exceptions for a process that the program or the procedure has failed to execute.
- For handling exceptions in a program, the best practice is to use the Try catch activity.
- The Try catch activity can be found in the Activities panel.
- By dragging and dropping the Try catch activity into the workspace, we can handle exceptions.
- For handling errors in the Try catch block, we can divide the whole process into four parts just to make it simpler:
  - ➢ Drag and drop the Try catch activity
  - ➢ Try block
  - ➢ Catch block
  - ➢ Finally block

These steps are illustrated in the following example:

1. <u>Drag and drop the Try catch activity:</u>

- Create a blank project.
- Drag and drop the Flowchart activity into the Designer panel.
- Search for the Try catch activity in the Activities panel and drag it into the Flowchart.
- Set it as the Start node.

2. <u>Try:</u>

- When we double-click on the Try catch activity, space for the Try activity appears.
- Inside the Try block, we have to drop the activity we want to perform.
- Drop a Write line activity to test the working of Try Catch block.

3. <u>Catches:</u>

- Inside the Catches activity, first we have to click on Add new Catch.
- Then click on Add Exception option, from which we have to select the type of exception.
- In most cases, System.Exception is preferred.
- Say the execution fails: for example, the Click activity is unable to be executed because of the unavailability of a UI element.
- In such a case, we can use the Catches block in order to either view the error that has occurred or for an alternative method to be used if that particular error occurs.
- After selecting System.Exception, now inside the exception block, we will drop a Message box activity.
- Entering exception.ToString will display the error that occurred during execution.

4. <u>Finally:</u>

- When we have defined the exception for our sequence, the Finally block will always work, regardless of whether the execution was successful or not.
- Suppose we want to display a message to the user notifying that the process is complete.
- To make sure that the whole Try catch activity is executed, we will just drop a Message box activity in the area provided in the Finally block.

2. List and explain common exceptions and ways to handle them.

Ans)

1. Unavailability of UI element
2. Handling runtime exceptions
3. Orbit reference not set to the instant of an object
4. Index was outside the bounds of an array. Index out of the range
5. Image not found in the provided timeout
6. Click Generic error

Unavailability of UI element:

- When working on UiPath, especially on the web, we may encounter this type of error.
- This is because the UI element was not found due to the dynamic behavior of the web page.
- To handle this exception, we have to make changes in the selector attributes or we have to add new attributes to the selector so that the UI element can be easily found.
- For example, if we have a variable which is dynamically changing, we can use a wildcard so that it can be easily found by the robot.
- Another way is to attach it to the live element.

Handling runtime exceptions:

- We may encounter runtime errors while working in UiPath.
- To rectify these errors, one of the best practices is to use the Try catch activity, which can be used to handle exceptions at runtime.
- By keeping an alternative inside the catch block, we can also overcome this error.
- So placing your sequence or workflow inside the Try catch activity will help you handle runtime exceptions.

Orbit reference not set to the instant of an object:

- This type of error usually occurs when the default value required for some variable is not provided.
- In that case, we are required to give a default value to the required variable.
- In the empty area indicated, just type the default value of the variable in order to overcome this error.

Index was outside the bounds of an array. Index out of the range:

- This error occurs when we try to iterate array elements by an index which is out of range.
- This happens when we are not aware of the size of the array and we just randomly type the index to access the element.
- To resolve this, we must check the size of the indexes of the array or the collective list.

Image not found in the provided timeout:

- This type of exception is thrown because the image was not found.
- This may be due to a change of environment, such as resolution or theme settings.
- In this case, using some selector attribute or indicating an anchor will work well.
- Indicate Anchor will help us indicate the UI element nearby so that the recorder can identify the correct image.

Click Generic error:

- This type of error occurs when the environment in which we are trying to use the Click activity does not support Simulate or Send message activity.
- Sometimes, either SimulateClick or the SendWindowMessages may be checked.
- In both cases, when an exception is thrown we just have to uncheck the appropriate box.

3. Explain Client Logging and Server Logging.

Ans)

Client logging:

- Client logs basically enable a server to record connections.
- These logs can be used by content providers in various scenarios, such as to generate billing, to trace media server usage or to deliver suitable quality content depending on the speed of the client's server.
- For client logging in UiPath, we have an NLog configuration file which makes it easy and flexible to integrate with databases, servers or any other NLog targets.
- Logging can be configured with this NLog.config file.
- UiPath Studio, Robot and workflow execution generate log messages on the client side:
  - ➢ Messages which are produced by the workflow execution are logged with the execution logging source.
  - ➢ Messages produced by UiPath Studio are logged as Studio Source.
  - ➢ Messages produced by UiPath Robot are logged as Robot logging Source.
  - ➢ We can also access these logs from UiPath Studio.
- We can access the stored logs by clicking on Open Logs in the EXECUTE option.
- By default, these Logs are saved in %Local App %\Uipath\Logs :
  - ➢ The automatic logging mechanism for all errors generated, including values of variables and arguments, may be enabled in the UiRobot.exe config file, which is present in C drive of the system, by setting the Log parameter from 0 to 1 inside the <Switches> section.
  - ➢ We have two activities that can be used for logging - Log message and WriteLine activities.

Server logging:

- If you have configured the UiPath server, then all logs generated by the execution are also sent to the server.
- You can take a screenshot anytime by pressing Ctrl + PrtScrn.

4. List and explain the various debugging techniques.

Ans) There are various techniques for debugging. They are:

- Setting breakpoints
- Slow step
- Highlighting
- Break

1) Setting breakpoints:

- While debugging a workflow, we can set breakpoints in between if we want to run the program up to a specific location.
- This is useful when we have to stop before an activity ends completely.
- In such a case, we should use a breakpoint on the previous activity.
- The highlighted region indicates the breakpoint since the execution stops just after the breakpoint.
- In order to continue any further, we have to click on the Continue button on the top corner indicated by the arrow.
- When we click on Step into, the relevant part will start to execute.
- After we click on Step over, execution will jump to the next part, and so on.

2) Slow step:

- This is an activity in the EXECUTE block through which we can reduce the execution speed of a particular process or activity.
- This way, we can identify each and every process and keep an eye on where to find the error.
- In the Output panel, all activities or steps can be viewed.

3) Highlighting:

- Highlighting is used to highlight the steps we have taken during automation and to identify each and every step in the workflow.
- It is very useful while debugging.
- Its panel can be found in the Options menu of the Execute section in the Ribbon.

4) Break:

- This activity is used to break a process at a certain point.
- Suppose we have a sequence performing seven activities together and if we want to break the execution at a certain activity, we can use the Break activity.
- While debugging, an option for Break is available.
- We can break at any point we want to.
- If we want to continue any further, we just have to click on Continue.
- Or we can stop the execution at that point by clicking on the Stop option.

5. Describe Collecting crash dumps.

Ans)

- Collecting crash dumps basically refers to collecting information when your UiPath Studio crashes.
- We can enable and disable crash dumps.
- These dumps provide us with information regarding the UiPath crash.
- Memory dumps are of two types- full dumps and mini dumps.
- Full dumps provide us with complete information about the encountered crash.
- Mini dumps provide us with just the main information regarding the crash.
- When a crash is encountered, we first have to identify the process which has crashed.
- Usually, a dialog will appear on the screen indicating the nature of the crash and the application involved.
- A UiPath process could crash, such as UiStudio.exe, Uiexplorer.exe or Uilauncher.exe, or the target application you want to automate may crash.

Enabling crash dumps:

The following are the steps to enable crash dumps:

1. To enable crash dumps, we first have to download the EnableFullDump.erg file for full dumps from

https://cdn2.hubspot.net/hubfs/416323/QuickAnswers/EnableFullDump.reg?t=1513326308120

or the EnableMiniDump.erg file from

https://cdn2.hubspot.net/hubfs/416323/QuickAnswers/EnableMiniDump.reg?t=1513326308120

2. Double-click the file and click Yes. Administrator rights are needed to access the registry settings.

3. The dumps folder is %TEMP% whose complete path is like C:\\users2;username\AppData\Local\TEMP

4. When the application crashes, you will find the .dmp file in the TEMP folder.

5. For example, if UiExplorer crashes then a file such as UiExplorer.exe.7429.dmp will be found in the TEMP folder.

Disabling crash dumps:

To disable crash dumps, perform the following steps:

1. Download the DisableDump.reg file from

https://cdn2.hubspot.net/hubfs/416323/QuickAnswers/EnableFullDump.reg?t=1513326308120

2. Double-click the file and click Yes to disable crash dumps. Administration rights are needed for this action.