

★★★★ Day 99 ★★★★★
of Our Daily UiPath Q&A
Challenge!

Welcome to Day 99 of our exciting
UiPath Q&A Challenge! Each day,
we'll be answering one key
question to help you master
UiPath and revolutionize your
automation journey. 🚀

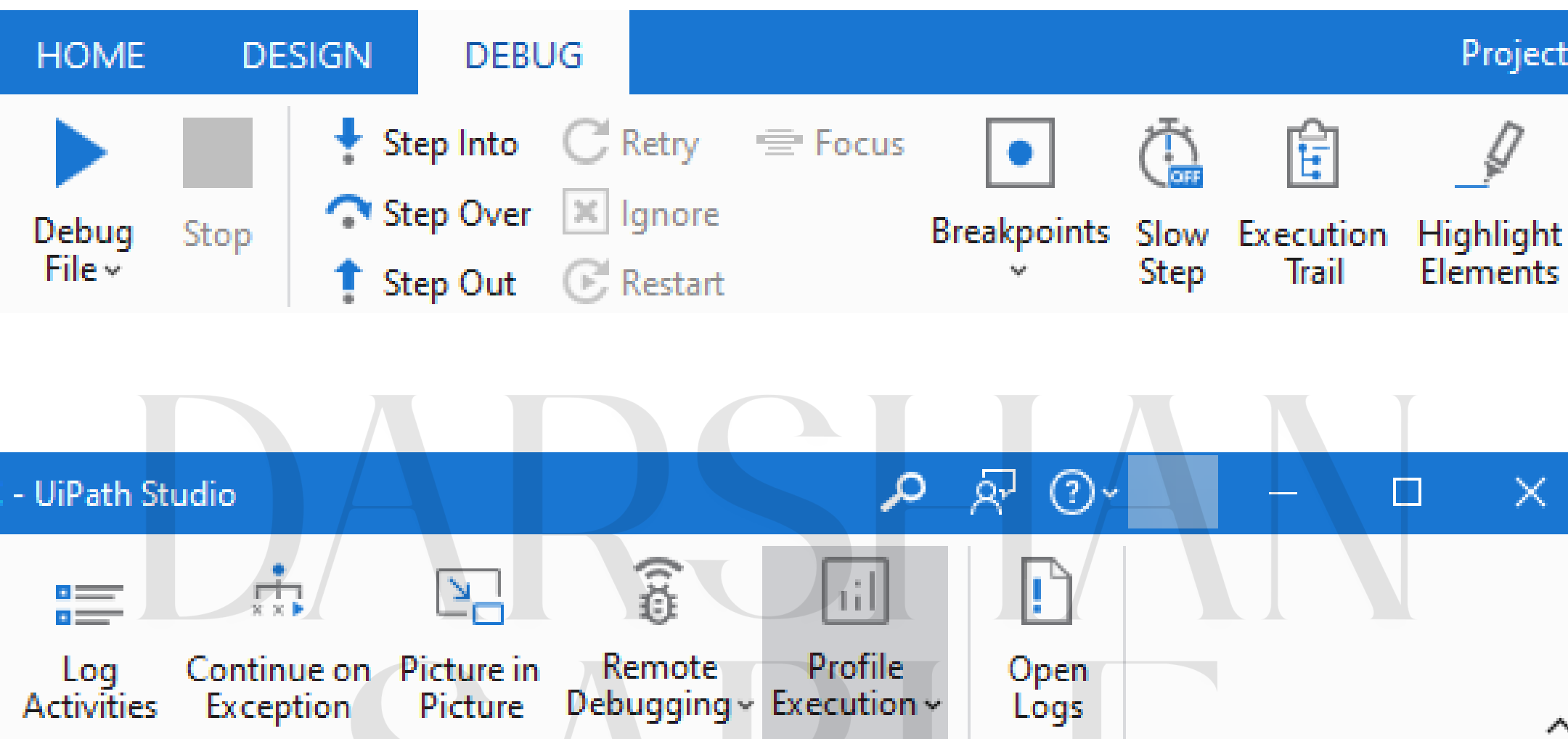
🚫 Question for Today

Explain Debugging Actions in
UiPath Studio ?

Debugging Actions

Debugging of a single file or the whole project can be performed both from the Design or Debug ribbon tabs.

However, the debugging process is not available if project files have validation errors.



Step Into

- Use Step Into to debug activities one at a time. When this action is triggered, the debugger opens and highlights the activity before it is executed.
- When Step Into is used with Invoke Workflow File activities, the workflow is opened in a new tab in ReadOnly mode and each activity is executed one by one.
- The keyboard shortcut for Step Into is F11.

Step Over

- Unlike the Step Into action, Step Over does not open the current container. When used, the action debugs the next activity, highlighting containers (such as flowcharts, sequences, or Invoke Workflow File activities) without opening them.
- This action comes in handy for skipping analysis of large containers which are unlikely to trigger any issues during execution.
- Step Over is available using the F10 keyboard shortcut.

Step Out

- As the name suggests, this action is used for stepping out and pausing the execution at the level of the current container. Step Out completes the execution of activities in the current container, before pausing the debugging. This option works well with nested sequences.
- Step Out is available using the Shift + F11 keyboard shortcut.

Retry

- Retry re-executes the previous activity, and throws the exception if it's encountered again. The activity which threw the exception is highlighted and details about the error are shown in the Locals and Call Stack panels.

Ignore

- The Ignore action can be used to ignore an encountered exception and continue the execution from the next activity so that the rest of the workflow can be debugged.
- This action is useful when jumping over the activity that threw the exception and continuing debugging the remaining part of the project.

Restart

- Restart is available after an exception was thrown and the debug process is paused. The action is used for restarting the debugging process from the first activity of the project. Use Slow Step to slow down the debugging speed and properly inspect activities as they are executed.
- Please take into consideration that when using this option after using the Run from this Activity action, the debugging is restarted from the previously indicated activity.

Break

- Break allows you to pause the debugging process at any given moment. The activity which is being debugged remains highlighted when paused. Once this happens, you can choose to Continue, Step Into, Step Over, or Stop the debugging process.
- It is recommended to use Break along with Slow Step so that you know exactly when debugging needs to be paused.
- An alternative to using Slow Step in this situation is to keep an eye on the Output panel and use Break on the activity that is currently being debugged.

Focus

- Focus Execution Point helps you return to the current breakpoint or the activity that caused an error during debugging. The Focus button is used after navigating through the process, as an easy way to return to the activity that caused the error and resume the debugging process.
- Alternatively, when debugging is paused because a breakpoint was reached, Focus can be used for returning to said breakpoint, after navigating through activities contained in the automation process.
- A third case is when the debugging is paused either after using Step Into or Step Over and then navigating through the process. In this case, Focus returns to the activity that paused the debugging process.
- From the Breakpoints context menu, you can select Focus to highlight the activity with the breakpoint.

Slow Step

- Slow Step enables you to take a closer look at any activity during debugging. While this action is enabled, activities are highlighted in the debugging process. Moreover, containers such as flowcharts, sequences, or Invoke Workflow File activities are opened. This is similar to using Step Into, but without having to pause the debugging process.
- Slow Step can be activated both before or during the debugging process. Activating the action does not pause debugging.
- Although called Slow Step, the action comes with 4 different speeds. The selected speed step runs the debugging process slower than the previous one. For example, debugging with Slow Step at 1x runs it the slowest, and fastest at 4x. In other words, the speed dictates how fast the debugger jumps from one activity to the next.
- Each time you click Slow Step the speed changes by one step. You can easily tell by the icon, which updates accordingly.

Execution Trail

- The Execution Trail ribbon button is disabled by default. When enabled, it shows the exact execution path at debugging. As the process is executed, each activity is highlighted and marked in the Designer panel, showing you the execution as it happens:
- executed activities are marked and highlighted in green;
- activities that were not executed are not marked in any way;
- activities that threw an exception are marked and highlighted in red.

Highlight Elements

- If enabled, UI elements are highlighted during debugging. The option can be used both with regular and step-by-step debugging.

Continue on Exception

- This debugging feature is disabled by default. When disabled in the ribbon, it throws the execution error and stops the debugging, highlights the activity which threw the exception, and logs the exception in the **Output** panel. If a Global Exception Handler was previously set in the project, the exception is passed on to the handler.
- When enabled, the exception is logged in the **Output** panel, the execution continues.

Log Activities

- If enabled, debugged activities are displayed as Trace logs in the Output panel. Note that Highlight Elements and Log Activities options can only be toggled before debugging, and persist when reopening the automation project. This is not applicable for invoked workflows unless these files are opened in the Designer panel.
- Logs are automatically sent to Orchestrator if connected, but you can have them stored locally by disabling the Allow Development Logging option from the Robot Settings tab in the Add or Edit user window.
- Disabling Log Activities can be a way to send smaller log files to Orchestrator.
- When running processes from Studio, the logs sent to Orchestrator are always Trace when Log Activities is disabled, and Verbose when Log Activities is enabled. This overrides both the Robot and the Orchestrator setting.

Remote Debugging

- When this feature is enabled, all run and debug operations are performed on a specified remote robot instead of the robot installed locally, allowing you to test the automation on different environments. For more information, see Remote Debugging.

Profile Execution

- You can identify performance bottlenecks in the workflow when you debug the file. For more information, see Profile Execution.

Open Logs


- Clicking Open Logs brings up the %localappdata%\UiPath\Logs folder where logs are locally stored. The naming format of log files is YYYY-DD-MM_Component.log (such as 2018-09-12_Execution.log, or 2018-09-12_Studio.log). Read more about logging here.

Picture in Picture

The Picture in Picture ribbon option in the Debug tab is available for both executing and debugging processes or libraries in a separate session on your machine.

If enabled, whenever you select Run or Run File, Debug or Debug File the process starts either in a separate session or in a virtual desktop in the user session. If Picture in Picture is disabled, debugging and execution is performed in the current session.

Having the option to run a process in Picture in Picture (PiP) can be very useful in attended automation. Verify whether a process runs successfully in PiP, and then update the project settings to indicate if it can be executed using this feature after it is published:

1. In the Project panel, click Settings  docs image to open the Project Settings window.
2. In the General tab:
 - PiP Options - Indicate whether the project was tested using Picture in Picture and whether it should start in PiP by default.
 1. Tested for PiP usage; Starting in PiP - The automation has been approved to run in PiP mode. When run, it starts in PiP by default.
 2. Tested for PiP usage; Not starting in PiP by default - The automation has been approved to run in PiP mode. When run, it starts in the main session or desktop by default.
 3. Not tested for PiP usage - The automation has not been approved to run in PiP mode. When run, it starts in the main session or desktop by default. If run in PiP, a dialog informs the user it was not tested using this feature and prompts for confirmation before proceeding.
 - PiP Type - Select how to isolate the automation from the user session when running the project in PiP: New Session (child session on the machine) or New Desktop (virtual desktop in the user session).