Instructions:

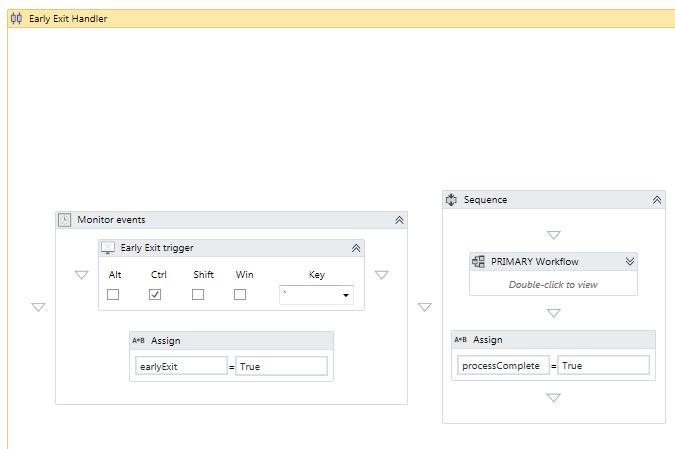
EARLY EXIT HANDLER (REF Extension)

Summary:

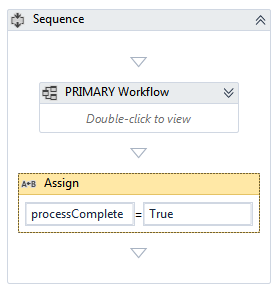
The Early Exit Handler is a framework that enable a smooth exit from any automated process prior to the natural “end” of the process. This handler works well as an extension to the Robotic Enterprise Framework already endorsed by UIPath, requiring only slight alterations to the framework.

Step-by-Step Instructions:

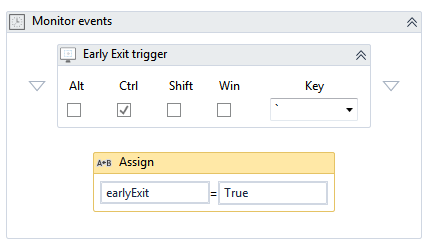
1. The Early Exit Handler consists of a single Parallel statement containing the user’s primary workflow alongside a “Monitor” activity that waits for one or more user-defined early exit triggers.



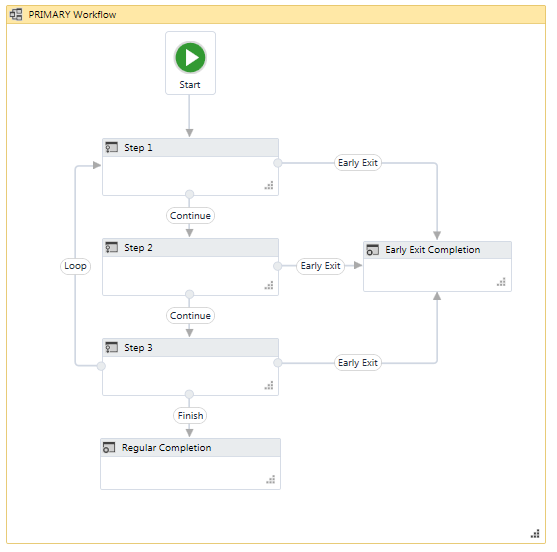
1. The Early Exit Handler requires two Boolean variables:
   1. The Parallel statement should contain a Boolean variable (e.g. *processComplete*) to determine when the Early Exit Handler should finish running. . The “CompletionCondition” for the Parallel statement should be when *processComplete* = TRUE. This Boolean should be predefined as NOTHING or FALSE, and only assigned to TRUE as the very last step following the user’s primary workflow:



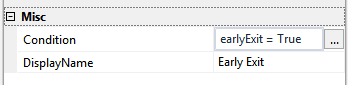
1. A second Boolean (e.g. *earlyExit*), also within the Parallel statement, should only be set to TRUE if the early exit trigger within the “Monitor” activity is activated.



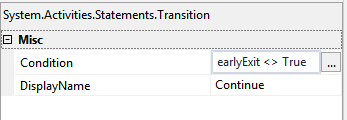
1. The primary workflow for the Early Exit Handler works best as a State Machine using the following setup:



1. Within the State Machine the process should be divided up into different States. At the end of each State, one transition (using the condition *earlyExit* = TRUE) will lead to an “Early Exit Completion” Final State. *earlyExit* <> TRUE should be part of the condition for all other transitions.



vs



1. If the early exit trigger is activated, the current State will finish running, then the State Machine will transition to the Early Exit Final State where you can perform whatever wrap up activities may be necessary in an early exit situation.
2. If the early exit trigger is *not* activated, the process will run its course and complete normally.

Alternative Applications and Alterations:

* The pop-up handler can be set to run indefinitely or as part of a user-assisted process. In such situations, an additional parallel should be added (e.g. a “Monitor” event waiting for a specific key combination, or a timed-delay) to determine when the process is complete. (An example of such a workflow is included in the documentation.)
* Although my sample project does not use the Robotic Enterprise Framework (for clarity-of-concept reasons), using REF as the workflow within the Early Exit Handler is an excellent option. In particular, by inserting a Try Catch within each State to handle both System and Business Exceptions, the user can create a 100% error-free process that can be safely exited at any time by wrapping the REF in an Early Exit Handler.
* Other triggers besides a “Hotkey Trigger” activity can be used to trigger an early exit. For instance, a “File Change Trigger” activity or a time-based trigger (see sample project for an example) could also be used.

Issues Addressed:

* The Early Termination framework will only exit the process at the end of each State. (Unless the user uses open-ended flowchart decision-trees as described in more detail below.) This is intentional. Although the Early Exit Handler’s parallel Boolean can be used to force an exit even in the middle of a State, this will occasionally result in an uncatchable error that will break the robot, terminating it with an error rather than the desired, smooth early exit. These uncatchable errors are particularly likely if the Primary workflow contains activities with potentially long timeout waits, such as “Find Element”.

Performance Notes:

* If your process is long and involved, divide it up into additional States to allow the Exit Early to function as quickly as possible.
* If an overarching flowchart is placed directly within each State, open-ended Flow Decisions can also be used to push the process to close more quickly when the early exit trigger is activated. (E.g. The Flow Decision condition would be *earlyExit* = TRUE. If TRUE the Flow Decision ends, sending the process out of both the Flowchart and State. If FALSE then continue with the process as usual.)
* If a user desires to use a time limit as the early exit trigger, they should not use a small looped delay and counter (see commented out example in sample project) rather than a single “Delay Until” or “Delay” activity. This way if the primary workflow completes more quickly than anticipated it will finish normally and in a timely manner. (Otherwise the robot will wait until the “Delay Until” or “Delay” is up before exiting the Early Exit Handler, regardless of when the primary workflow is completed.)

Sample Use Cases:

* A user will be running an extremely long front-office robot but may need to shut down the computer midway through. The user can create the process within an Early Exit Handler, using a hotkey trigger to wrap up the process early, if necessary.
* A process can continue running until 5:00 when the server will be going down. At that point the robot should wrap up its activities (close applications, etc.) whether or not it has completed its entire run. The Early Exit Handler can help make this potential exit seamless.