Running Actions In Parallel

[FinalBuilder Professional Edition]

Motivation

FinalBuilder allows you to structure your project so that multiple actions can be run simultaneously. A graphical representation of multiple actions running in parallel is represented by this flow diagram:

? Unknown Attachment

After Task 1 finishes, tasks 2a, 2b and 2c all run simultaneously. Only when 2a, 2b and 2c all finish (the sync point), can task 3 can run.

Async Action Groups

To run actions in parallel use the "ASync Action Group" action (in the Flow Control category).

? Unknown Attachment

This action looks like a normal Action Group action, except that each of its immediate children are run simultaneously. When all the parallel child actions have completed, the ASync Action group finishes and control continues sequentially (this is the Sync Point in the above diagram.)

The flow diagram above can be represented in FinalBuilder as follows:

? Unknown Attachment

In the screenshot above, Task 1 has already completed and now all 3 of the parallel tasks are running at once. When all the parallel tasks have completed, Task 3 will run.

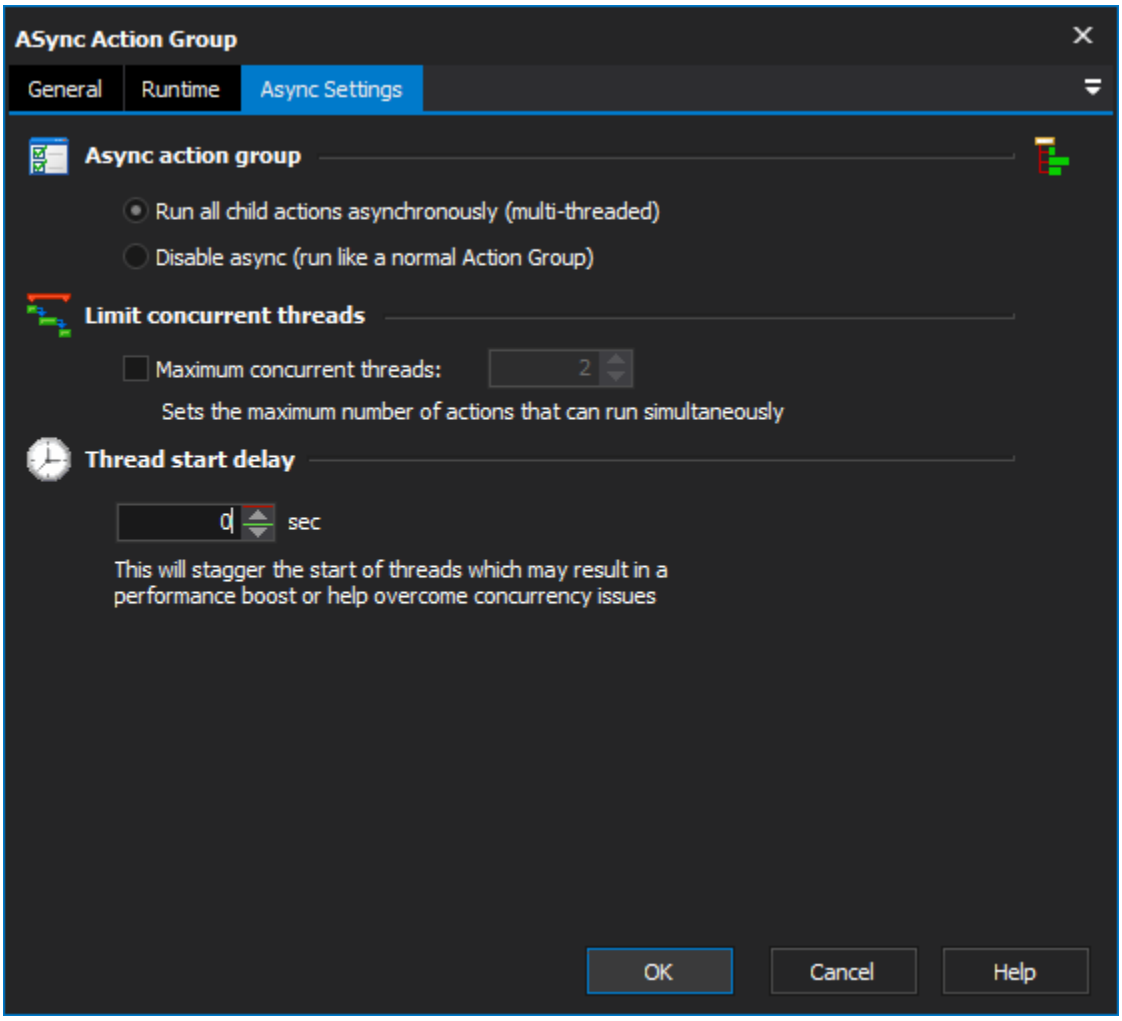
Grouping actions under Async

If actions under an ASync group have child actions, then these actions will run sequentially in the same thread as the parent (ie not asynchronously) and follow the normal processing rules:

? Unknown Attachment

(In the example above, task 2b has 4 sequential child actions. The first and second have completed, and the third is currently running.) Tasks 2a and 2c are running in parallel.

Async Action Group Properties



You can choose to Disable Async to have the Async Action Group behave like a normal (sequential) action group. This may be useful for debugging purposes, or for running the project on hardware which may not perform faster with multiple actions running simultaneously.

Setting a thread start delay allows you to "stagger" the start of the threads, which may give a performance boost or help overcome concurrency issues. Starting from the second child of the Async Action Group, each child will be started the specified number of seconds after the previous async child.

If you would like to limit the number of simultaneously running actions, then set the maximum concurrent threads. It must be at least 2, otherwise the Async group will behave like a normal Action Group.

Getting the Most from Async Action Groups

The purpose of ASync Action Group is to allow your project to run multiple actions simultaneously to reduce the total running time. This is most beneficial under one or more of the following conditions:

- More than one CPU (or more than one CPU core) is available.
- "IO bound" actions can be run together with "CPU bound" actions.
- Slow network-bound actions, or other non-CPU intensive actions, can be run together with CPU intensive actions.

It is entirely possible to slow the project down by running certain actions together, for example if you have two IO bound actions then running them together will may result in an overall reduction in performance. Similarly, it is not recommended that the number of CPU bound actions running in parallel exceeds more than one greater than the number of available CPU cores (and, in some cases, less.)

You can even run ASync Action groups inside ASync Action groups. Again, it is recommended that caution is exercised when doing this.

Limitations

You cannot have any interactive actions which require user input (eg. Prompt for Variables) under Async Action Groups.

Things to be aware of for actions running in parallel

- Reading and writing the same FinalBuilder variables (if running the same action list multiple times asynchronously, use Action List Parameters.)
- Reading and writing the same files.
- Reading and writing the same registry values.
- Some compilers (such as the Delphi compiler) and some other tools may lock certain files and a second instance may not succeed.
- The Visual SourceSafe actions cannot be used under Async group as the actions temporarily modify the SourceSafe INI files.