Coordinating Actions using Signals

In Flux, a signal is an indicator that announces something significant has occurred within the workflow itself. Within the workflow you can configure your actions and triggers to monitor for certain signals. This allows you to prevent from running or bypass actions if a certain flow is reached, or even use conditional flows to take an alternate action if some signal is raised. Signals are a kind of "GO TO" within a workflow, allowing portions of a workflows to be bypassed.

In this way, signals allow you to make actions and triggers dependent on each others outcomes. If you have an action in your workflow that requires another task to complete successfully, or a certain step that should only execute if a different flow was followed, you can use signals to make your action react accordingly.

Generally speaking, there are two components to signals:

- A flow that raises a signal.
- A trigger or action that monitors for the signal, and either:
    - Takes a certain flow based on the signal it receives, or
    - Stops executing (if the signal was raised early enough, it can also stop the action from running altogether)

Signals are not transferred between workflows, so a signal raised in one workflow cannot be picked up in a different workflow. Signals also cannot be used to transmit data — a signal is only an indicator that a certain flow was followed.

Special Characters

Signals may not contain the special character '. Workflows using signals that contain this special character cannot be saved or exported.

Raising Signals

Signals are raised on the flows in a workflow. When the engine follows a flow, it will raise all of the signals that flow is configured for; those signals can then be picked up by any monitoring action or trigger.

NOTE: a workflow will never have more than one instance of a particular signal active, so if a flow raises a signal that already exists in the workflow and has not been cleared, the new signal will be ignored.

Designer

In the Designer, you can configure a flow to raise signals by doing the following:

1. Open the flow's dialog that should raise the signal(s) and select "Flow Properties" from the menu.
2. In the box labeled "Signals To Raise", enter the name of the signal you want to raise. This name will match the name you provide to the action or trigger that will monitor for this signal.
3. If you want to raise an additional signal, click the "Add" button and enter the signal name in the box that appears. Repeat this for any additional signals you want to add.

Monitoring for Signals

You can configure an action to monitor for a signal in one of two ways - by setting its "Signals to Monitor" property, or by adding a signal flow from the action.

When an action is monitoring for a signal and that signal arrives, the action's behavior will depend on whether it has started executing yet, and whether there are any outgoing signal flows.

If the action has not started executing when the signal is raised, the signal will prevent the action from executing altogether. If the signal has already been raised when the workflow reaches the action, it will not execute the action. If there are any signal flows, the workflow will follow them; otherwise the workflow (or execution branch) will stop at that point.

NOTE: Process Actions can be exempt from this — if you set the "destroy on signal" property for the process action, then Flux will kill the process and stop the action immediately. After that, the workflow will follow signal flows or stop the execution branch as normal.

If the action has already started when the signal is raised, the engine will wait for the action to complete normally. Once the action has finished, the workflow will follow any signal flows from the action. If there are no signal flows, the workflow (or execution branch) will stop when the action completes.

If a trigger is monitoring for a signal but the trigger has already started polling, it will immediately stop polling once the signal is raised. If there are any signal flows, the workflow will follow them; otherwise, the workflow (or execution branch) will stop at the trigger.

This same behavior applies if an action is monitoring for multiple signals and two or more of those signals are received simultaneously. The action will follow all appropriate signal flows for each signal it receives.

An action will automatically clear any signal it receives as long as it is the only action monitoring for that signal. If multiple actions are monitoring for the same signal, however, you must clear the signal yourself.

As an example, consider a workflow with two actions (action *A* and action *B*) that run in parallel. Further suppose that if action *A* encounters an error, it must notify action *B* to take corrective action. This can be achieved by creating an error flow from action *A* that raises a signal ("SIGNAL" in this case), and having action *B* monitor for the signal using a signal flow (a flow with a signal condition that looks for "SIGNAL").

As soon as action *A* encounters an error, it will raise the signal "SIGNAL". If action *B* has not started executing yet, the workflow will skip executing action *B* and instead follow its signal flow to take corrective action. If action *B* has already started, the signal flow will be followed as soon action action *B* has completed normally.

For an example that demonstrates how to use signals to cancel an action / actions when another action or condition occurs, see the "abort" example, located in the */examples/end_users/abort* directory under your Flux installation directory.

Signal Flows

You can create a signal flow by giving your flow a signal condition and providing the name of the signal that should trigger the flow. A signal flow will only be followed if the given signal is raised.

You can create a signal flow in the Designer by doing the following:

1. Create a new flow.
2. Open the flow's dialog and select "Flow Properties" from the menu.
3. In the "Flow Condition" box, select the "SIGNAL" option.
4. In the dialog box that appears, enter the name of the signal that should be associated with this flow.
5. Repeat steps 1-4 for each additional signal flow you want to create.

Clearing Signals

If there is only one action or trigger monitoring for a signal in your workflow, the signal will be automatically cleared as soon as the action or trigger receives it. This prevents actions and triggers that execute in a loop from accidentally picking up signals that they have already received.

If there are multiple actions or triggers monitoring for a signal, however, you must clear the signals yourself. You can do this by setting the "Signals to Clear" property on the flow.

Designer

In the Designer, you can configure a flow to clear a signal by doing the following:

1. Open the flow's dialog that should clear the signal and select "Flow Properties" from the menu.
2. In the box labeled "Signals To Clear", enter the name of the signal you want to clear when this flow is followed.
3. If you want to clear an additional signal, click the "Add" button and enter the name of the signal. Repeat this for each signal you want to clear.

Signals and Join Points

Signals take precedence over join points. If an action is a join point and is *also* monitoring for a signal, the action will follow any outgoing signal flows as soon as the signal arrives, regardless of whether the join condition has been satisfied or not.

For example, consider the diagram below. In this case actions *A*, *B*, and *C* are all flowing into a join point, action *J*. Action *J* has the join condition "*B and C*" , meaning that the join condition is only satisfied when actions *B* and *C* have completed and flowed into action *J*.

Action *J* is also monitoring for a signal that is raised on the flow from action *A* to action *J*. This means that if action *A* finishes, it will raise the signal that action *J* is monitoring for.

If action *A* raises its signal before actions *B* and *C* both complete, then action *J* will immediately follow its signal flow, even though the join condition is not yet satisfied.