Node Guide: Break

Overview

The **Break** node is a control node used to exit a loop prematurely. It acts as an interrupt mechanism inside loop structures like **For(n)** or **For(each)**, allowing you to stop further iterations when certain conditions are met.

What This Node Does

- Immediately terminates the loop when it is encountered during execution.
- Prevents further iterations from running.
- Works only inside the scope of **For(n)** or **For(each)** loop nodes.

Inputs

- This node does **not** accept any inputs.

Outputs

- This node does **not** produce any outputs.

Configuration Details

- **No configuration required**.
- Simply place the node inside a loop to enable conditional loop-breaking behavior.
- Typically used in combination with a **Condition** node to control its execution path.

When to Use

Use the **Break** node when you want to:

- Stop a loop as soon as a specific condition is met.
- Exit early from processing a list or range.
- Improve performance by avoiding unnecessary iterations once the required outcome is achieved.

Example Flow

Scenario: Iterating through a list of items and stopping once a match is found.

1. **For(Each) Node** – Loops through items in a list.
2. **Condition Node** – Checks if the current item meets your condition.
3. **Break Node** – Placed under the "Yes" path of the condition to exit the loop.

Summary

The **Break** node helps in building efficient and controlled workflows by giving you the ability to exit loops early, just like the `break` statement in traditional programming languages.