Switch

# Switch#

Use the Switch node to route a workflow conditionally based on comparison operations. It's similar to the IF node, but supports multiple output routes.

## Node parameters#

Select the Mode the node should use:

• Rules: Select this mode to build a matching rule for each output.

• Expression: Select this mode to write an expression to return the output index programmatically.

Node configuration depends on the Mode you select.

### Rules#

To configure the node with this operation, use these parameters:

• Create Routing Rules to define comparison conditions.  
Use the data type dropdown to select the data type and comparison operation type for your condition. For example, to create a rules for dates after a particular date, select Date & Time > is after.  
The fields and values to enter into the condition change based on the data type and comparison you select. Refer to Available data type comparisons for a full list of all comparisons by data type.

• Use the data type dropdown to select the data type and comparison operation type for your condition. For example, to create a rules for dates after a particular date, select Date & Time > is after.

• The fields and values to enter into the condition change based on the data type and comparison you select. Refer to Available data type comparisons for a full list of all comparisons by data type.

• Rename Output: Turn this control on to rename the output field to put matching data into. Enter your desired Output Name.

Select Add Routing Rule to add more rules.

You can further configure the node with this operation using these Options:

• Fallback Output: Choose how to route the workflow when an item doesn't match any of the rules or conditions.  
None: Ignore the item. This is the default behavior.  
Extra Output: Send items to an extra, separate output.  
Output 0: Send items to the same output as those matching the first rule.

• None: Ignore the item. This is the default behavior.

• Extra Output: Send items to an extra, separate output.

• Output 0: Send items to the same output as those matching the first rule.

• Ignore Case: Set whether to ignore letter case when evaluating conditions (turned on) or enforce letter case (turned off).

• Less Strict Type Validation: Set whether you want n8n to attempt to convert value types based on the operator you choose (turned on) or not (turned off).

• Send data to all matching outputs: Set whether to send data to all outputs meeting conditions (turned on) or whether to send the data to the first output matching the conditions (turned off).

### Expression#

To configure the node with this operation, use these parameters:

• Number of Outputs: Set how many outputs the node should have.

• Output Index: Create an expression to calculate which input item should be routed to which output. The expression must return a number.

## Templates and examples#

by Jimleuk

by Eduard

by Jonathan

## Related resources#

Refer to Splitting with conditionals for more information on using conditionals to create complex logic in n8n.

## Available data type comparisons#

### String#

String data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty

• is equal to

• is not equal to

• contains

• does not contain

• starts with

• does not start with

• ends with

• does not end with

• matches regex

• does not match regex

### Number#

Number data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty

• is equal to

• is not equal to

• is greater than

• is less than

• is greater than or equal to

• is less than or equal to

### Date & Time#

Date & Time data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty

• is equal to

• is not equal to

• is after

• is before

• is after or equal to

• is before or equal to

### Boolean#

Boolean data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty

• is true

• is false

• is equal to

• is not equal to

### Array#

Array data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty

• contains

• does not contain

• length equal to

• length not equal to

• length greater than

• length less than

• length greater than or equal to

• length less than or equal to

### Object#

Object data type supports these comparisons:

• exists

• does not exist

• is empty

• is not empty