Boolean

These nodes work with **true/false** values — the yes/no answers your scene produces, such as whether a door is open, a switch is on, or a trainee is standing in the right place. An operator takes one or two of these true/false values and works out a new one, so you can combine several conditions into a single answer and react to it.

What's on this page

- **And Operation** — the answer is true only when *both* values are true.
- **Or Operation** — the answer is true when *at least one* value is true.
- **Not Operation** — flips a single value to its opposite.

The **And** and **Or** nodes hand you the answer in two ways at once: a true/false **Result** you can read whenever you like, and two triggers — **OnTrue** and **OnFalse** — that fire at the moment the answer changes. Use whichever suits your scene.

True / false logic

 And Operation

Checks two true/false values and tells you whether *both* of them are true.

What it does

This node looks at two true/false values you feed in and combines them: the result is true only when **both** are true. If either one is false — or both are — the result is false. Think of it as the word “and” in everyday speech: the door is unlocked *and* the trainee is wearing a helmet.

The node keeps watching both inputs and updates as they change. The **Result** output always holds the current answer, ready to read at any time. The two trigger outputs fire only at the moment the answer flips: **OnTrue** fires the instant the combined answer becomes true, and **OnFalse** fires the instant it becomes false. If an input changes but the overall answer stays the same, neither trigger fires.

Inputs

Port	Type	What to connect
Boolean A	True / false	The first true/false value you want to check — for example, whether a switch is on.
Boolean B	True / false	The second true/false value to check alongside the first — for example, whether a safety guard is closed. Both must be true for the result to be true.

Outputs

Port	Type	What you get
OnTrue	Trigger	Fires once at the moment both values become true. Wire this to whatever should happen when the combined answer turns true.
OnFalse	Trigger	Fires once at the moment the answer turns false — that is, when at least one of the two values stops being true. Wire this to whatever should happen when the combined answer turns false.
Result	True / false	The current combined answer: true when both values are true, otherwise false. You can read this at any time.

Example

Boolean A input	Machine power is on — `true`
Boolean B input	Safety guard starts open (`false`), then the trainee closes it (`true`)

OnTrue output	Fires the instant the guard closes — now both are true, so the answer flips to true and this trigger fires once (you could use it to start the machine)
Result output	`true`

Tips

- Use the **Result** output when you just need the current answer somewhere else; use **OnTrue** / **OnFalse** when you want something to happen the moment the answer changes.
- The triggers fire only on a change. If both values are already true when your scene starts, **OnTrue** won't fire until the answer leaves and returns to true.
- Need to check that *either* value is true instead of both? Use the **Or Operation** node. Need to flip a single true/false value? Use the **Not Operation** node.

Or Operation

Checks two true/false values and tells you whether at least one of them is true.

What it does

This node looks at its two true/false inputs and works out a single combined answer. The combined answer is **true** whenever at least one of the inputs is true, and only becomes **false** when both inputs are false. You can read this answer at any time from the Result output.

The node keeps watching both inputs and re-checks the answer whenever either one changes. The moment the combined answer flips to true, the **OnTrue** trigger fires once; the moment it flips back to false, the **OnFalse** trigger fires once. If an input changes but the overall answer stays the same, neither trigger fires.

Inputs

Port	Type	What to connect
Boolean A	True / false	The first true/false value to check — for example whether a door is open.
Boolean B	True / false	The second true/false value to check — for example whether a window is open.

Outputs

Port	Type	What you get
OnTrue	Trigger	Fires once at the moment the combined answer becomes true — that is, when at least one input has just turned true. Connect this to whatever should happen when that switch-on moment occurs.
OnFalse	Trigger	Fires once at the moment the combined answer becomes false — that is, when both inputs have just become false. Connect this to whatever should happen when that switch-off moment occurs.
Result	True / false	The current combined answer: true if at least one input is true, false only when both are false. You can read this at any time.

Example

Boolean A input	Door is open — `false`
Boolean B input	Window is open — changes from `false` to `true`
Result output	`true` (at least one is now open)
OnTrue output	Fires once, because the combined answer just flipped from false to true — you could use it to start a “something is open” warning.

Tips

- Use **Result** when you just want the current answer to feed into another node, and use **OnTrue** / **OnFalse** when you want something to happen the moment the answer changes.
- The triggers fire only on a change. If the answer is already true and an input change keeps it true, **OnTrue** will not fire again.
- If you need the answer to be true only when *both* inputs are true, use the **And Operation** node instead.



Not Operation

Flips a true/false value to its opposite.

What it does

This node takes a single true/false value and gives you back the reverse. If the value coming in is true, the result is false. If it comes in as false, the result is true.

It’s handy when you want to react to the opposite of something — for example, doing something only when a door is *not* open, or when a switch is *not* turned on. The value you connect is not changed; you simply get a new, flipped result out.

Inputs

Port	Type	What to connect
Input	True / false	The true/false value you want to flip — for example, whether a light is on or whether an object is visible.

Outputs

Port	Type	What you get
Output	True / false	The opposite of what you connected: `false` when the input is true, and `true` when the input is false.

Example

Input	`true` (the door is open)
Output	`false` — so “the door is not open” is false

Tips

- Use this when you want to act on the opposite of a condition without setting up a separate check — for example, to know when something is *not* happening.

Related

- **Branch** — use the **Result** from any of these nodes to send your scene down one path or another.
- Chain these three together to build a bigger check — for example, flip a value with **Not Operation** before feeding it into **And Operation** or **Or Operation**.

Revision #3
Created 2026-06-09 11:49:17 UTC by Rafat
Updated 2026-10-25 09:23:48 UTC by Rafat