Code node documentation

# Code node#

Use the Code node to write custom JavaScript or Python and run it as a step in your workflow.

Coding in n8n

This page gives usage information about the Code node. For more guidance on coding in n8n, refer to the Code section. It includes:

• Reference documentation on Built-in methods and variables

• Guidance on Handling dates and Querying JSON

• A growing collection of examples in the Cookbook

Examples and templates

For usage examples and templates to help you get started, refer to n8n's Code integrations page.

Function and Function Item nodes

The Code node replaces the Function and Function Item nodes from version 0.198.0. If you're using an older version of n8n, you can still view the Function node documentation and Function Item node documentation.

## Usage#

How to use the Code node.

### Choose a mode#

There are two modes:

• Run Once for All Items: this is the default. When your workflow runs, the code in the code node executes once, regardless of how many input items there are.

• Run Once for Each Item: choose this if you want your code to run for every input item.

## JavaScript#

The Code node supports Node.js.

### Supported JavaScript features#

The Code node supports:

• Promises. Instead of returning the items directly, you can return a promise which resolves accordingly.

• Writing to your browser console using console.log. This is useful for debugging and troubleshooting your workflows.

console.log

### External libraries#

If you self-host n8n, you can import and use built-in and external npm modules in the Code node. To learn how to enable external modules, refer to the Enable modules in Code node guide.

If you use n8n Cloud, you can't import external npm modules. n8n makes two modules available for you:

• crypto Node.js module

• moment npm package

### Built-in methods and variables#

n8n provides built-in methods and variables for working with data and accessing n8n data. Refer to Built-in methods and variables for more information.

The syntax to use the built-in methods and variables is $variableName or $methodName(). Type $ in the Code node or expressions editor to see a list of suggested methods and variables.

$variableName

$methodName()

$

### Keyboard shortcuts#

The Code node editing environment supports time-saving and useful keyboard shortcuts for a range of operations from autocompletion to code-folding and using multiple-cursors. A full list can be found in the list of keyboard shortcuts.

## Python#

n8n added Python support in version 1.0. It doesn't include a Python executable. Instead, n8n provides Python support using Pyodide, which is a port of CPython to WebAssembly. This limits the available Python packages to the Packages included with Pyodide. n8n downloads the package automatically the first time you use it.

Slower than JavaScript

The Code node takes longer to process Python than JavaScript. This is due to the extra compilation steps.

### Built-in methods and variables#

n8n provides built-in methods and variables for working with data and accessing n8n data. Refer to Built-in methods and variables for more information.

The syntax to use the built-in methods and variables is \_variableName or \_methodName(). Type \_ in the Code node to see a list of suggested methods and variables.

\_variableName

\_methodName()

\_

### Keyboard shortcuts#

The Code node editing environment supports time-saving and useful keyboard shortcuts for a range of operations from autocompletion to code-folding and using multiple-cursors. A full list can be found in the list of keyboard shortcuts.

## File system and HTTP requests#

You can't access the file system or make HTTP requests. Use the following nodes instead:

• Read/Write File From Disk

• HTTP Request

## Coding in n8n#

There are two places where you can use code in n8n: the Code node and the expressions editor. When using either area, there are some key concepts you need to know, as well as some built-in methods and variables to help with common tasks.

### Key concepts#

When working with the Code node, you need to understand the following concepts:

• Data structure: understand the data you receive in the Code node, and requirements for outputting data from the node.

• Item linking: learn how data items work, and how to link to items from previous nodes. You need to handle item linking in your code when the number of input and output items doesn't match.

### Built-in methods and variables#

n8n includes built-in methods and variables. These provide support for:

• Accessing specific item data

• Accessing data about workflows, executions, and your n8n environment

• Convenience variables to help with data and time

Refer to Built-in methods and variables for more information.

## Use AI in the Code node#

Feature availability

AI assistance in the Code node is available to Cloud users. It isn't available in self-hosted n8n.

AI generated code overwrites your code

If you've already written some code on the Code tab, the AI generated code will replace it. n8n recommends using AI as a starting point to create your initial code, then editing it as needed.

To use ChatGPT to generate code in the Code node:

• In the Code node, set Language to JavaScript.

• Select the Ask AI tab.

• Write your query.

• Select Generate Code. n8n sends your query to ChatGPT, then displays the result in the Code tab.

## Common issues#

For common questions or issues and suggested solutions, refer to Common Issues.