Processing different data types

# Processing different data types#

In this chapter, you will learn how to process different types of data using n8n core nodes.

## HTML and XML data#

You're most likely familiar with HTML and XML.

HTML vs. XML

HTML is a markup language used to describe the structure and semantics of a web page. XML looks similar to HTML, but the tag names are different, as they describe the kind of data they hold.

If you need to process HTML or XML data in your n8n workflows, use the HTML node or the XML node.

Use the HTML node to extract HTML content of a webpage by referencing CSS selectors. This is useful if you want to collect structured information from a website (web-scraping).

### HTML Exercise#

Let's get the title of the latest n8n blog post:

• Use the HTTP Request node to make a GET request to the URL https://blog.n8n.io/ (this endpoint requires no authentication).

https://blog.n8n.io/

• Connect an HTML node and configure it to extract the title of the first blog post on the page.  
Hint: If you're not familiar with CSS selectors or reading HTML, the CSS selector .post .item-title a should help!

• Hint: If you're not familiar with CSS selectors or reading HTML, the CSS selector .post .item-title a should help!

.post .item-title a

Use the XML node to convert XML to JSON and JSON to XML. This operation is useful if you work with different web services that use either XML or JSON and need to get and submit data between them in the two formats.

### XML Exercise#

In the final exercise of Chapter 1, you used an HTTP Request node to make a request to the PokéAPI. In this exercise, we'll return to that same API but we'll convert the output to XML:

• Add an HTTP Request node that makes the same request to the PokéAPI at https://pokeapi.co/api/v2/pokemon.

https://pokeapi.co/api/v2/pokemon

• Use the XML node to convert the JSON output to XML.

## Date, time, and interval data#

Date and time data types include DATE, TIME, DATETIME, TIMESTAMP, and YEAR. The dates and times can be passed in different formats, for example:

DATE

TIME

DATETIME

TIMESTAMP

YEAR

• DATE: March 29 2022, 29-03-2022, 2022/03/29

DATE

• TIME: 08:30:00, 8:30, 20:30

TIME

• DATETIME: 2022/03/29 08:30:00

DATETIME

• TIMESTAMP: 1616108400 (Unix timestamp), 1616108400000 (Unix ms timestamp)

TIMESTAMP

• YEAR: 2022, 22

YEAR

There are a few ways you can work with dates and times:

• Use the Date & Time node to convert date and time data to different formats and calculate dates.

• Use Schedule Trigger node to schedule workflows to run at a specific time, interval, or duration.

Sometimes, you might need to pause the workflow execution. This might be necessary if you know that a service doesn't process the data instantly or it's slow to return all the results. In these cases, you don't want n8n to pass incomplete data to the next node.

If you run into situations like this, use the Wait node after the node that you want to delay. The Wait node pauses the workflow execution and will resume execution:

• At a specific time.

• After a specified time interval.

• On a webhook call.

### Date Exercise#

Build a workflow that adds five days to an input date from the Customer Datastore node that you used before. Then, if the calculated date occurred after 1959, the workflow waits 1 minute before setting the calculated date as a value. The workflow should be triggered every 30 minutes.

To begin:

• Add the Customer Datastore (n8n training) node with the Get All People action selected. Return All.

• Add the Date & Time node to Round Up the created Date from the datastore to End of Month. Output this to field new-date. Include all input fields.

• Add the If node to check if that new rounded date is after 1960-01-01 00:00:00.

1960-01-01 00:00:00

• Add the Wait node to the True output of that node and set it to wait for one minute.

• Add the Edit Fields (Set) node to set a new field called outputValue to a String containing new-date. Include all input fields.

• Add the Schedule Trigger node at the beginning of the workflow to trigger it every 30 minutes. (You can keep the Manual Trigger node for testing!)

## Binary data#

Up to now, you have mainly worked with text data. But what if you want to process data that's not text, like images or PDF files? These types of files are represented in the binary numeral system, so they're considered binary data. In this form, binary data doesn't offer you useful information, so you'll need to convert it into a readable form.

In n8n, you can process binary data with the following nodes:

• HTTP Request to request and send files from/to web resources and APIs.

• Read/Write Files from Disk to read and write files from/to the machine where n8n is running.

• Convert to File to take input data and output it as a file.

• Extract From File to get data from a binary format and convert it to JSON.

Reading and writing files is only available on self-hosted n8n

Reading and writing files to disk isn't available on n8n Cloud. You'll read and write to the machine where you installed n8n. If you run n8n in Docker, your command runs in the n8n container and not the Docker host. The Read/Write Files From Disk node looks for files relative to the n8n install path. n8n recommends using absolute file paths to prevent any errors.

To read or write a binary file, you need to write the path (location) of the file in the node's File(s) Selector parameter (for the Read operation) or in the node's File Path and Name parameter (for the Write operation).

File(s) Selector

File Path and Name

Naming the right path

The file path looks slightly different depending on how you are running n8n:

• npm: ~/my\_file.json

~/my\_file.json

• n8n cloud / Docker: /tmp/my\_file.json

/tmp/my\_file.json

### Binary Exercise 1#

For our first binary exercise, let's convert a PDF file to JSON:

• Make an HTTP request to get this PDF file: https://media.kaspersky.com/pdf/Kaspersky\_Lab\_Whitepaper\_Anti\_blocker.pdf.

https://media.kaspersky.com/pdf/Kaspersky\_Lab\_Whitepaper\_Anti\_blocker.pdf.

• Use the Extract From File node to convert the file from binary to JSON.

### Binary Exercise 2#

For our second binary exercise, let's convert some JSON data to binary:

• Make an HTTP request to the Poetry DB API https://poetrydb.org/random/1.

https://poetrydb.org/random/1

• Convert the returned data from JSON to binary using the Convert to File node.

• Write the new binary file data to the machine where n8n is running using the Read/Write Files From Disk node.

• To check that it worked out, use the Read/Write Files From Disk node to read the generated binary file.