Guide-3

Techwars 2025-2026

Introduction

An API is a set of functions allowing the creation of applications that access the features or data.

For our purposes, we will be using free API services to access data online. This data will be stored on Google Sheets and can then be accessed by Grafana, which will display graphs of the data.

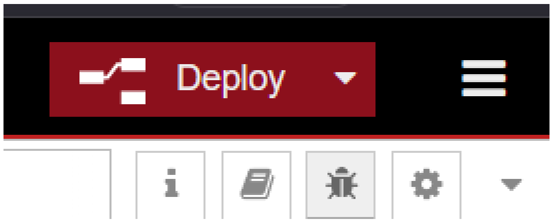
Example: Pokemon API

This example walks through pokeapi.co, getting information about the Eevee Pokemon and storing it in Google Sheets

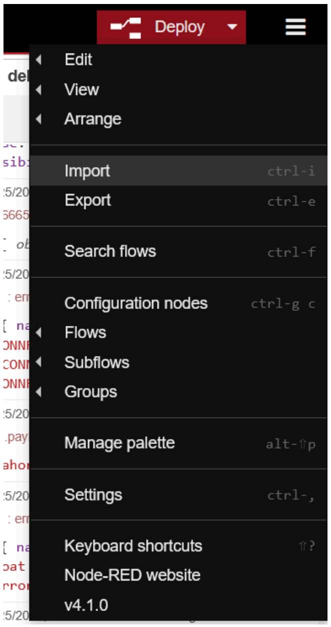
URL	https://pokeapi.co/api/v2/pokemon/{POKEMON_NAME}
Example	https://pokeapi.co/api/v2/pokemon/eevee
Explanation	Obtains information about a chosen Pokemon.

Accessing The Example

Open NodeRED, click on the three white lines on the top-left.



Next, click on **import**.



In the menu that opens (called **Clipboard**) paste the code from the examples below.

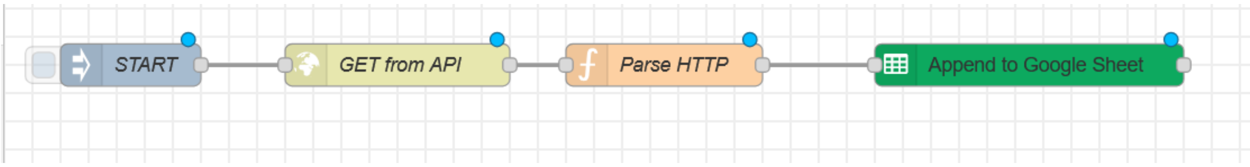
Copy and paste the code given below into NodeRED. An explanation is given below.
You do **not** need to understand this code or write it on your own. Just paste it.

Code to copy:

```
[
    {
        "id": "f6f2187d.f17ca8",
        "type": "tab",
        "label": "Flow 1",
        "disabled": false,
        "info": ""
    },
    {
        "id": "3cc11d24.ff01a2",
        "type": "comment",
        "z": "f6f2187d.f17ca8",
        "name": "NOTICE: This is an example that pulls pokemon data and sends it to Google
Sheets",
        "info": "",
        "x": 310,
        "y": 80,
        "wires": []
    },
    {
        "id": "3b4eee15e09f0787",
        "type": "inject",
        "z": "f6f2187d.f17ca8",
        "name": "START",
        "props": [
            {
                "p": "payload"
            },
            {
                "p": "topic",
                "vt": "str"
            }
        ],
        "repeat": "",
        "crontab": "",
        "once": false,
        "onceDelay": 0.1,
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "x": 90,
        "y": 200,
        "wires": [
            [
                "6c48fb729c65bc1c"
            ]
        ]
    },
    {
        "id": "6c48fb729c65bc1c",
        "type": "http request",
        "z": "f6f2187d.f17ca8",
        "name": "GET from API",
        "method": "GET",
        "ret": "obj",
        "paytoqs": "ignore",
        "url": "https://pokeapi.co/api/v2/pokemon/eevee",
        "tls": "",
        "persist": false,
        "proxy": "",
        "insecureHTTPParser": false,
        "authType": "",
        "senderr": false,
        "headers": [],
        "x": 280,
        "y": 200,
        "wires": [
            [
                "7aa777e6b9ae7918"
            ]
        ]
    },
    {
        "id": "883653f535b62262",
        "type": "GSheet",
        "z": "f6f2187d.f17ca8",
        "creds": "90f6e151a24390cc",
        "method": "append",
        "action": "",
        "sheet": "1rKgodAUeoObXzyWEbta0zIzyIC9Ivb2ArbjuGGg-Pd0",
        "cells": "Table1!A:D",
        "flatten": false,
        "name": "Append to Google Sheet",
        "x": 730,
        "y": 200,
        "wires": [
            []
        ]
    },
    {
        "id": "7aa777e6b9ae7918",
        "type": "function",
        "z": "f6f2187d.f17ca8",
        "name": "Parse HTTP",
        "func": "\n\nmsg.payload = [\n    [msg.payload.abilities[0].ability.name,\n    msg.payload.abilities[0].ability.url,\n    msg.payload.abilities[0].slot\n]\n];\nreturn msg;\n",
        "outputs": 1,
        "timeout": 0,
        "noerr": 0,
        "initialize": "",
        "finalize": "",
        "libs": [],
        "x": 470,
        "y": 200,
        "wires": [
            [
                "883653f535b62262"
            ]
        ]
    },
    {
        "id": "90f6e151a24390cc",
        "type": "gauth",
        "name": "techwars@big-genre-482208-i8.iam.gserviceaccount.com"
    },
    {
        "id": "a75aa4277cecb712",
        "type": "global-config",
        "env": [],
        "modules": {
            "node-red-contrib-google-sheets": "1.1.2"
        }
    }
]
```

Code Explanation

After pasting, your NodeRED window should show this:

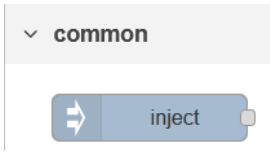


Clicking on the button on the left of **START** will run the code. Clicking **deploy** will save your code.

Explanations for each Node

START: Inject Node

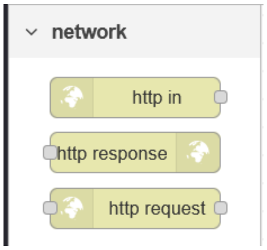
START represents an Inject Node. This starts your program.



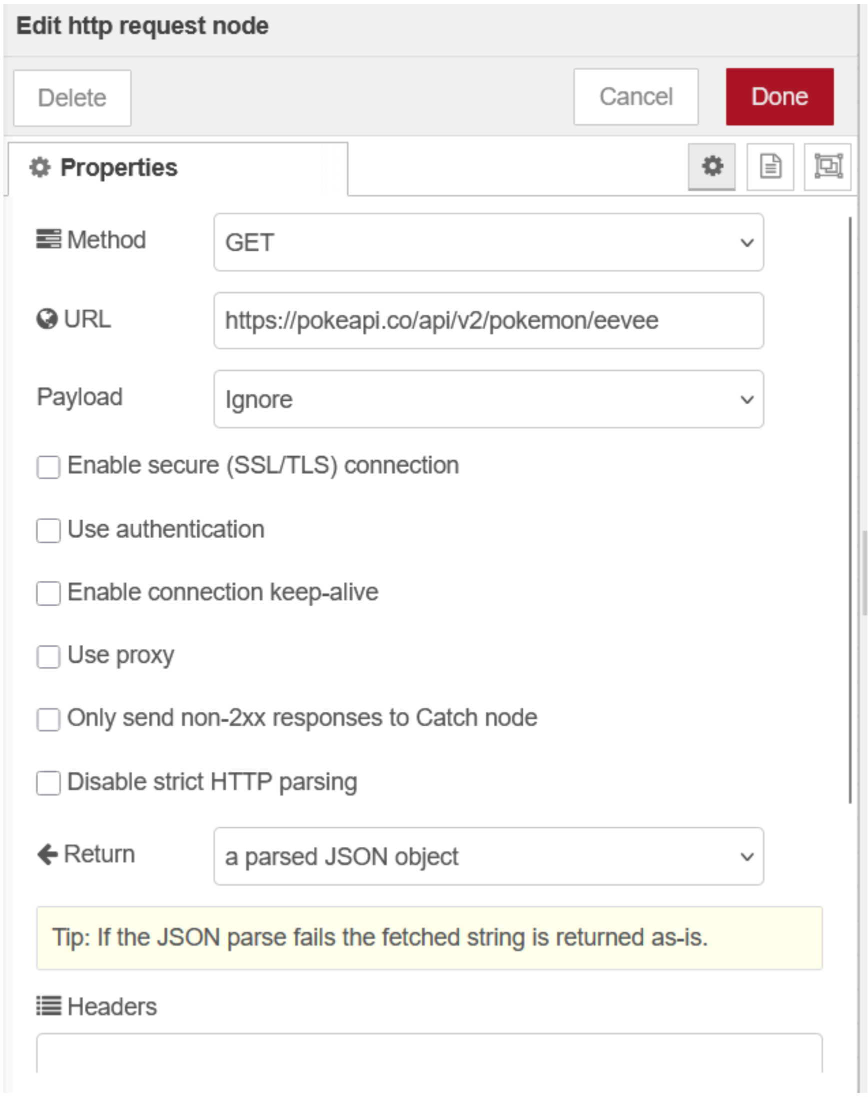
You can set it to loop as well, so it runs automatically.

GET from API: http request Node

An http request Node is capable of multiple functions, allowing it to send and get data to and from websites.
For our purposes, the **GET** function is necessary. This connects to the Pokemon API and gets all relevant data from it.



Double-click the **Get from API** node. You should see this:



Edit http request node

Method: GET

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

Payload: Ignore

□ Enable secure (SSL/TLS) connection

□ Use authentication

□ Enable connection keep-alive

□ Use proxy

□ Only send non-2xx responses to Catch node

□ Disable strict HTTP parsing

Return: a parsed JSON object

Tip: If the JSON parse fails the fetched string is returned as-is.

Headers

Here is an explanation of each entry:

Method	GET	This ensures that your node gets data from the chosen url.
URL	https://pokeapi.co/api/v2/pokemon/eevee	This obtains data from pokeapi.co You can replace this with your chosen API website
Return	a parsed JSON object	Make sure this is selected, as it makes it easier to deal with the data in later Nodes.

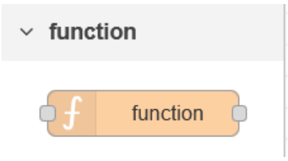
You may ignore all other options, but make sure you understand these three.

Parse HTTP: Function Node

A function Node is similar to a function in Python. It can do almost anything imaginable.
For our purposes, it is being used to take the data about the Eevee from **GET from API** and convert it into data that makes sense to Google Sheets.

Function Nodes use **Javascript**. You do not need to know about this. Just know the following:

- Each line ends with a semicolon: “;”
- **msg.payload** is the variable that stores all your important data.



After you run **GET from API**, the function Node will get this as its input:

```
{
   "abilities": [
      {
         "ability": {
            "name": "run-away",
            "url": "https://pokeapi.co/api/v2/ability/50/"
         },
         "is_hidden": false,
         "slot": 1
```

```
    },
    {
      "ability": {
        "name": "adaptability",
        "url": "https://pokeapi.co/api/v2/ability/91/"
      },
      "is_hidden": false,
      "slot": 2
    },
    {
      "ability": {
        "name": "anticipation",
        "url": "https://pokeapi.co/api/v2/ability/107/"
      },
      "is_hidden": true,
      "slot": 3
    }
  ],
```

Looking at the output above, try to understand what it represents using the following table:

Example	What it is	What it does	Accessing Data

Curly Braces {}
E.g.
msg.payload = {
“Abilities”: 1
};
	An **object**	This is very similar to a Python dictionary.
		Items are stored as a key-value pair.
			Use a dot “.”
			E.g.
			MyObject.Abilities
			Gives 1, because Abilities is the key and 1 is the value

Square brackets [] msg.payload = [1, 2, 3];	An **array**	This is very similar to a Python list. It stores a bunch of items.	Use [Index] Index is a number that counts the position in the list, starting with **0.** [1, 2, 3] 1 is at position 0, So its index is 0. E.g. **msg.payload[2]** returns 3

Helpful videos:

Objects: https://www.youtube.com/watch?v=Io7o91qLzxc

Arrays: https://www.youtube.com/watch?v=yQ1fz8LY354

Now, returning to **Parse HTTP**

The code in Parse HTTP is given below:

```
msg.payload = [
    [msg.payload.abilities[0].ability.name,
    msg.payload.abilities[0].ability.url,
    msg.payload.abilities[0].slot
]
];
return msg;
```

This means the **msg.payload** now stores a **list** as shown by the square brackets “[]”

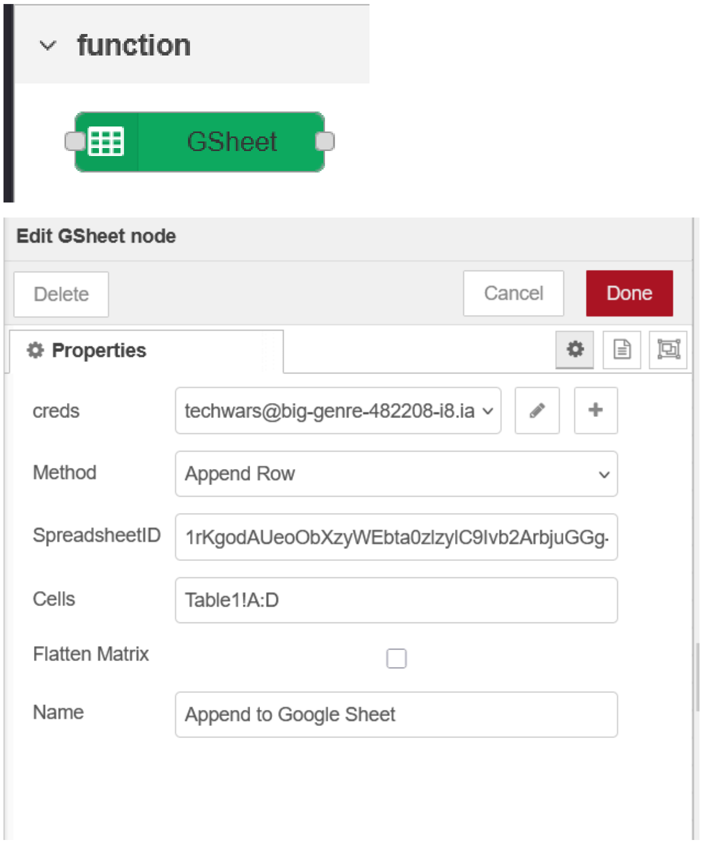
Within the list is the ability at position **0.**

.name, .url and **.slot** get the name, url and slot of the Pokemon’s ability respectively.

Now, you have a list that stores the name, url and slot of your chosen ability.

Append to Google Sheet: GSheet Node

A more detailed explanation on this is available on the main guide.



creds: techwars@big-genre-482208-i8.ia

Method: Append Row

SpreadsheetID: 1rKgodAUeoObXzyWEbta0zlzylC9Ivb2ArbjuGGg-

Cells: Table1!A:D

Flatten Matrix: ☐

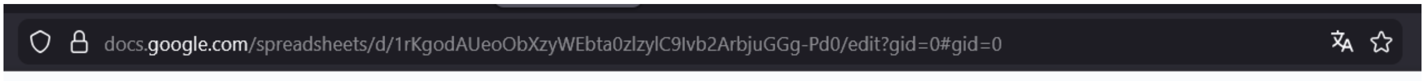
Name: Append to Google Sheet

Creds	Contains the JSON information copy-pasted from your Google Service account. Please view the main guide for details.	
Method	Append Row	This adds a new row to your chosen Google Sheet
Spreadsheet ID	1rKgodAUeoObXzyWEbta0zlzylC9Ivb2ArbjuGGg-Pd0	This is the id of your Google Sheet
Cells	Table1!A:D	Format: SheetName!FromCell:ToCell
Name	Append to Google Sheet	This is just the name of your node. You may write

		whatever you wish.

Spreadsheet ID

Your web-browser will show this url when you open a Google Sheet:



docs.google.com/spreadsheets/d/1rKgodAUeoObXzyWEbta0zlzylC9lvb2ArbjuGGg-Pd0/edit?gid=0#gid=0

Or,

```
https://docs.google.com/spreadsheets/d/1rKgodAUeoObXzyWEbta0zlzylC9lvb2ArbjuGGg-Pd0/edit?gid=0#gid=0
```

From this the id you need is between **/d/** and **/edit?**

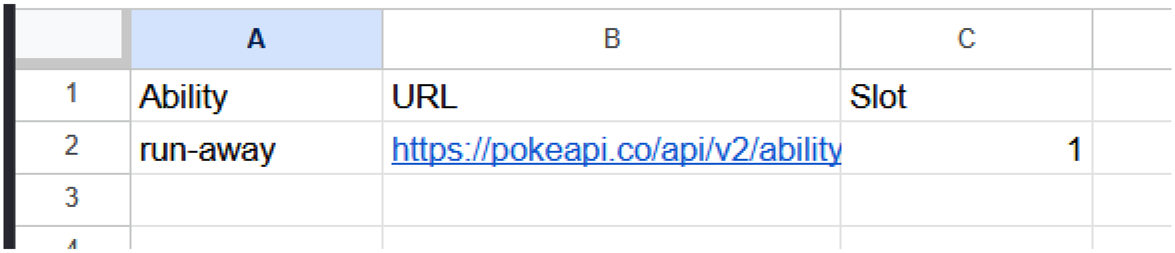
Or, https://docs.google.com/spreadsheets**/d/*YOUR_ID_HERE*/edit?**gid=0#gid=0

Formatting Cells

When getting data from a Google Sheet, you use **SheetName!FromCell:ToCell**

For example, if your sheet was called Sheet2, and you wanted data from A33 to C34, you would use:
Sheet2!A33:C34

Finally, this takes the data from the last node and sends it to the Google Sheet. Or, the Google Sheet receives data about the Eevee's ability. The output should look like this:



	A	B	C
1	Ability	URL	Slot
2	run-away	https://pokeapi.co/api/v2/ability	1
3			
4			

You can test the example code in this open access Google Sheet, which is already connected:

docs.google.com/spreadsheets/d/1rKgodAUeoObXzyWEbta0zlzylC9lvb2ArbjuGGg-Pd0/edit?gid=0#gid=0

Appendix

This example is not usable in Grafana. Why?

This is because Grafana requires a column for **Time**. Or, a different column to compare against.

Add the following code to the **Parse HTTP** node so that it adds the current time to each entry:

```
const time = new Date();

msg.payload = [
    [
    time,
    msg.payload.abilities[0].ability.name,
    msg.payload.abilities[0].ability.url,
    msg.payload.abilities[0].slot
]
];
return msg;
```

new Date();
adds the current date to a variable called time. time is then stored in msg.payload.

Useful Links

Node Red Essentials Playlist	https://www.youtube.com/watch?v=ksGeUD26Mw0&list=PLyNBB9VCLmo1hyO-4fIZ08gqFcXBkHy-6
Understanding Dashboards in Grafana	https://youtu.be/vTiIkdDwT-0?si=BYZ8Kdqx7faoELgN
Most Commonly Used Visualisations in Grafana	https://www.youtube.com/watch?v=JwF6FgeotaU
Getting Started with Google Sheets Data Source Plugin	https://www.youtube.com/watch?v=hqeqeQFrtSA

That’s a Wrap

Think about how you can use a timestamp for Pokemon data. Could it be used on a per-trainer basis? Could it be used to show when a Pokemon was first caught?

You can do the same with other types of data, be it football match stats, anime popularity or weather data.

Use the main guide alongside the API list to complete your Dashboard.

Good luck!