

Hands-on NodeJS

- + JavaScript
- + Debug(WebStorm)
- + TypeScript

arnaud.nauwynck@gmail.com

Esilv 2023

This document:

<https://github.com/Arnaud-Nauwynck/presentations/tree/main/web/tp1-nodejs.pdf>

Objectives

- Discover NodeJs using practical « Hello World » developper code
- Using Simple baby steps, step by step
- Setup your web environment
- Become familiar with the Web FrontEnd ecosystem

Outline

- Download + Install NodeJs
- Run NodeJS from the command line
- Discover and run basic javascript program
- Discover NPM Node Package Manager
- Install package.json dependencies
- Install WebStorm IDE
(students may choose any IDE, VisualStudio Code)
- Debug Javascript program
- TypeScript to JavaScript from the IDE
- Tsc compiler
- Debug TypeScript

Google NodeJs

← → C google.com/search?q=nodejs&rlz=1C1CHBF_frFR928FR928&oq=nodejs&aqs=chrome..69i57j69i59j0i20i263i512j0i512j0i10i512j69i60l3.1583j0j9&sourceid=chrome&ie=UTF-8 ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋ ⌊ ⌋

Google

nodejs

Videos Images Tutorial Course Documentation Download W3schools Framework Jobs All filters ▾ Tools SafeSearch ▾

About 267,000,000 results (0.29 seconds)

 [Node.js](https://nodejs.org)
https://nodejs.org > ... ::

Node.js

Node.js® is an open-source, cross-platform JavaScript runtime environment. Download for Linux (x64). 18.17.1 LTS Recommended For Most ...

You've visited this page many times. Last visit: 7/9/2023



Download

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript ...

Docs

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript ...

Téléchargements

Dernière version LTS: 18.17.1 (includes npm 9.6.7 ...)

About

As an asynchronous event-driven JavaScript runtime, Node.js is ...


More images

Node.js

Platform

Node.js is a cross-platform, open-source server environment that can run on Windows, Linux, Unix,

Download NodeJs

nodejs.org/en/download/current

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Downloads

Latest Current Version: 20.5.1 (includes npm 9.8.0)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS	Current
Recommended For Most Users	Latest Features
 Windows Installer node-v20.5.1-x64.msi	 macOS Installer node-v20.5.1.pkg
	 Source Code node-v20.5.1.tar.gz

Windows Installer (.msi)

Windows Binary (.zip)

macOS Installer (.pkg)

macOS Binary (.tar.gz)

Linux Binaries (x64)

Linux Binaries (ARM)

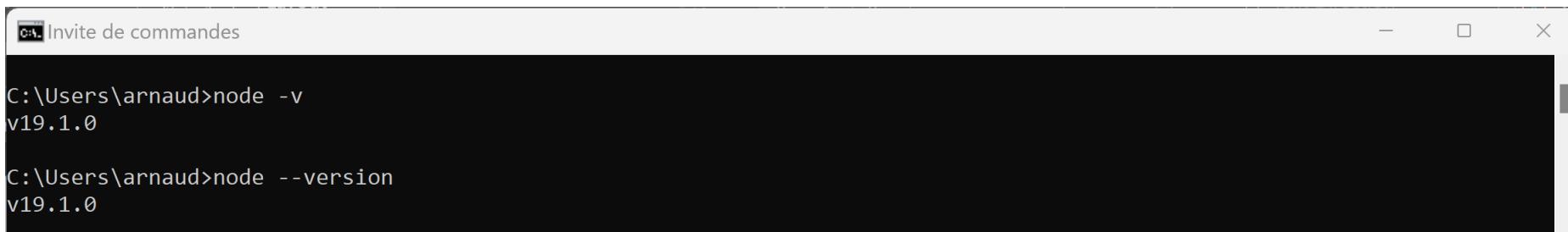
Source Code

32-bit	64-bit	ARM64
32-bit	64-bit	ARM64
64-bit / ARM64		
64-bit		ARM64
64-bit		
ARMv7		ARMv8
node-v20.5.1.tar.gz		

Additional Platforms

Check NodeJs exe (check version)

C:> node -v

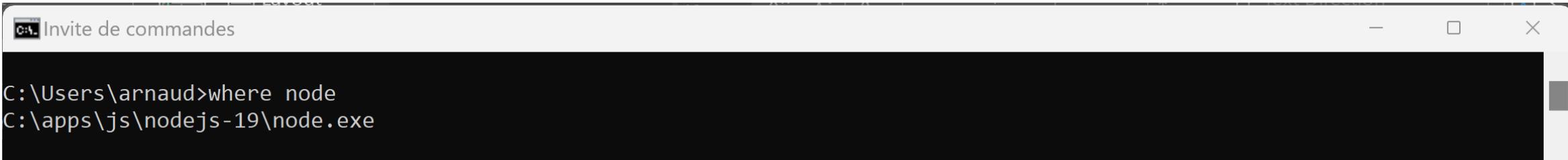


```
C:\Users\arnaud>node -v
v19.1.0

C:\Users\arnaud>node --version
v19.1.0
```

Several installed ?
... which one will be resolved

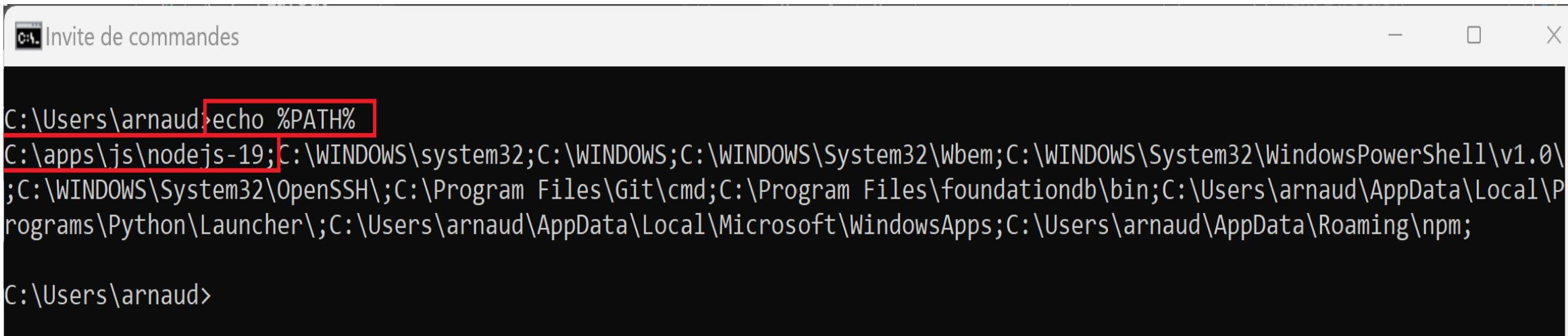
C:> where node



```
C:\Users\arnaud>where node
C:\apps\js\nodejs-19\node.exe
```

Check NodeJS is in « %PATH% »(win cmd terminal)

echo %PATH%

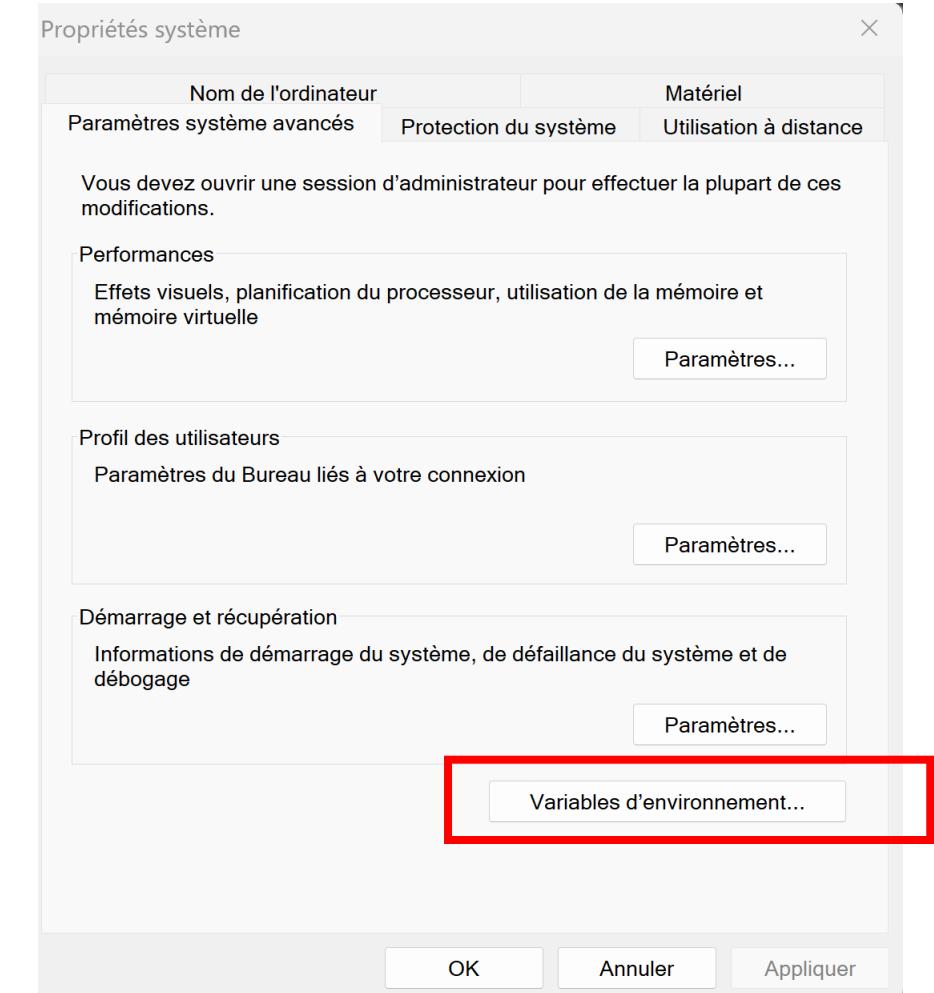
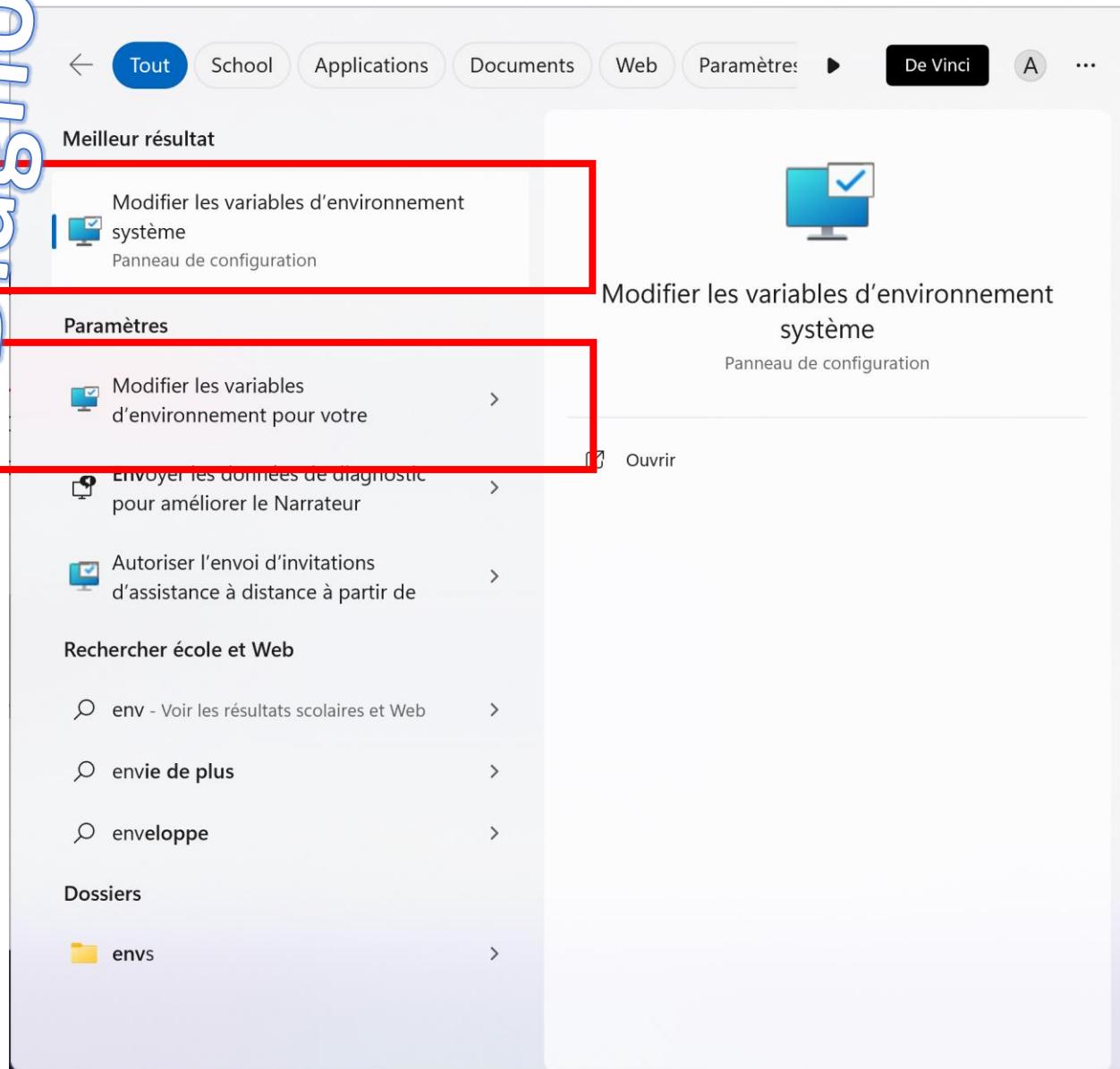


A screenshot of a Windows Command Prompt window titled "Invite de commandes". The window shows the command "echo %PATH%" entered at the prompt, followed by the output of the environment variable PATH. The output lists several directory paths, including "C:\apps\js\nodejs-19;" which is highlighted with a red rectangle. The entire window has a blue decorative border.

```
C:\Users\arnaud>echo %PATH%
C:\apps\js\nodejs-19;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\
;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Git\cmd;C:\Program Files\foundationdb\bin;C:\Users\arnaud\AppData\Local\Programs\Python\Launcher\;C:\Users\arnaud\AppData\Local\Microsoft\WindowsApps;C:\Users\arnaud\AppData\Roaming\npm;
C:\Users\arnaud>
```

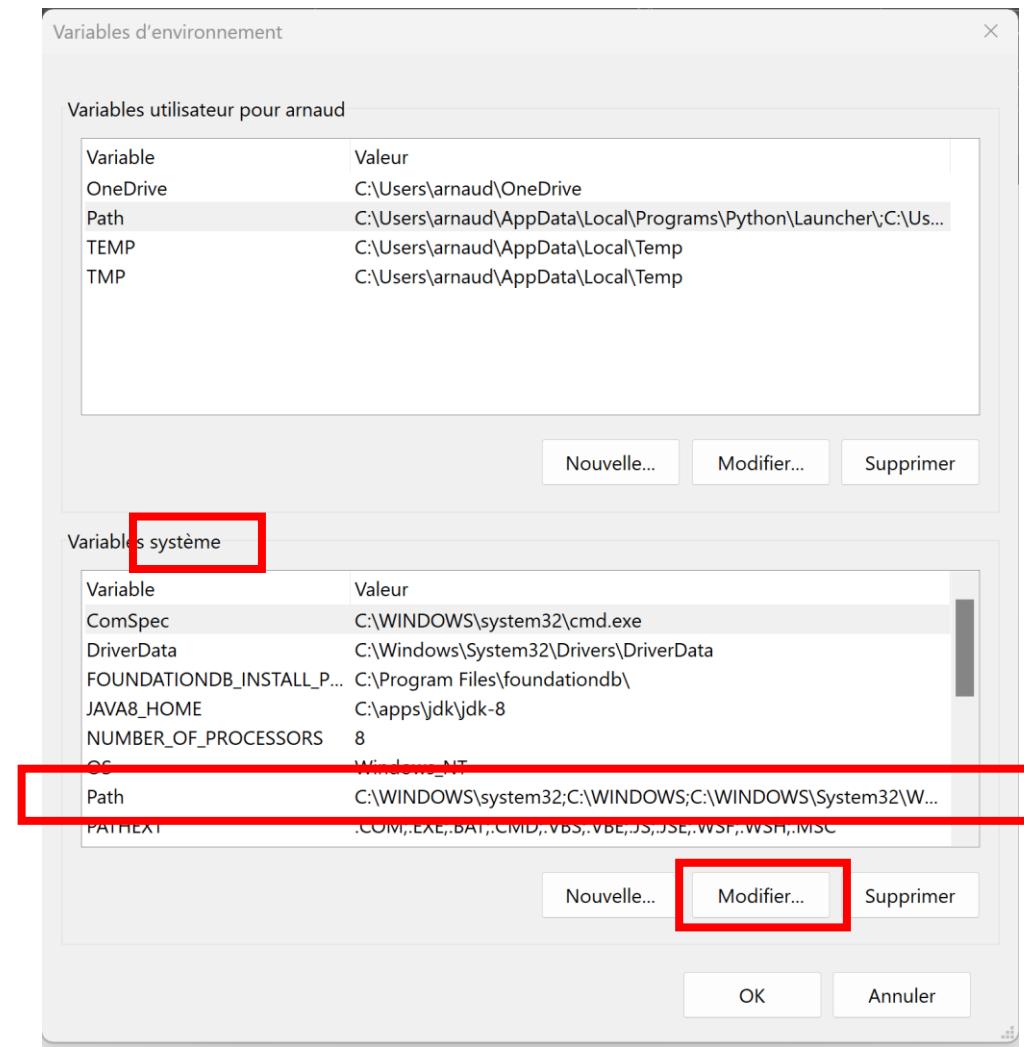
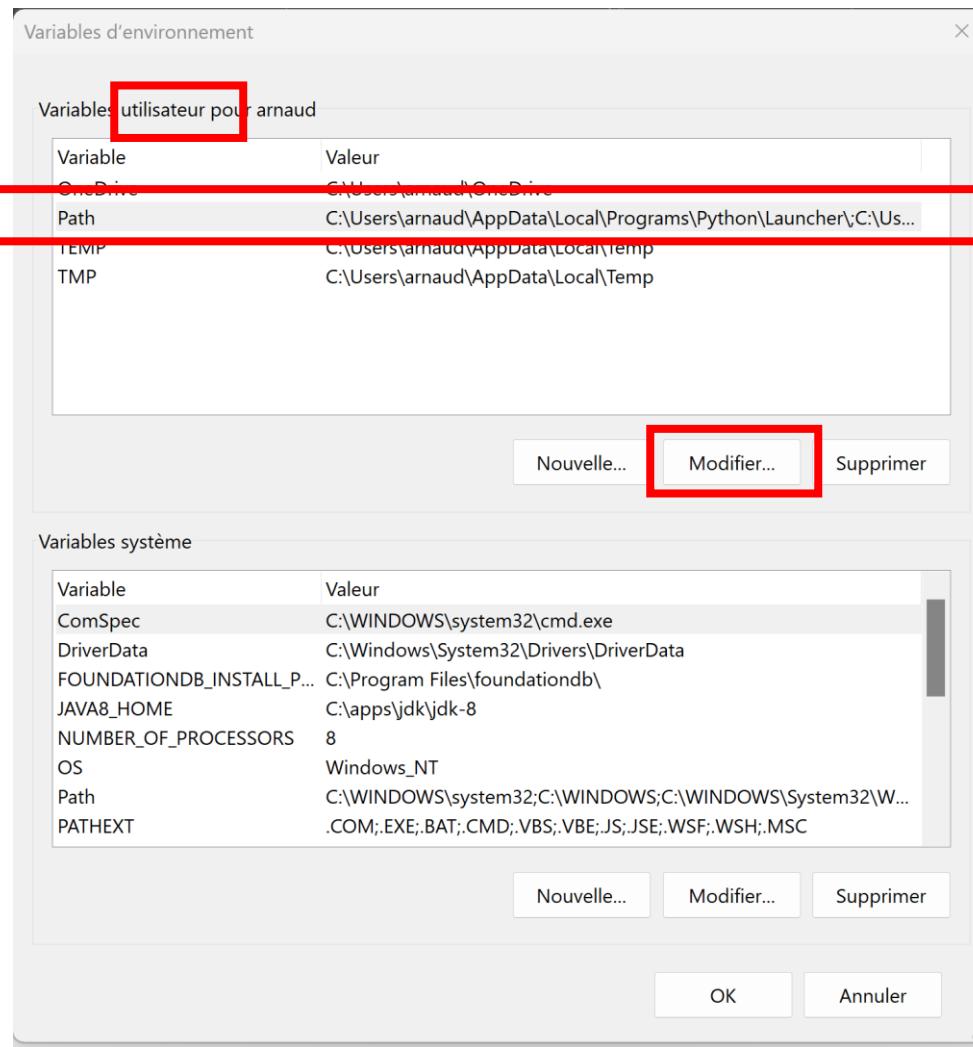
System Diagnostic

Edit System or User Environment Variables

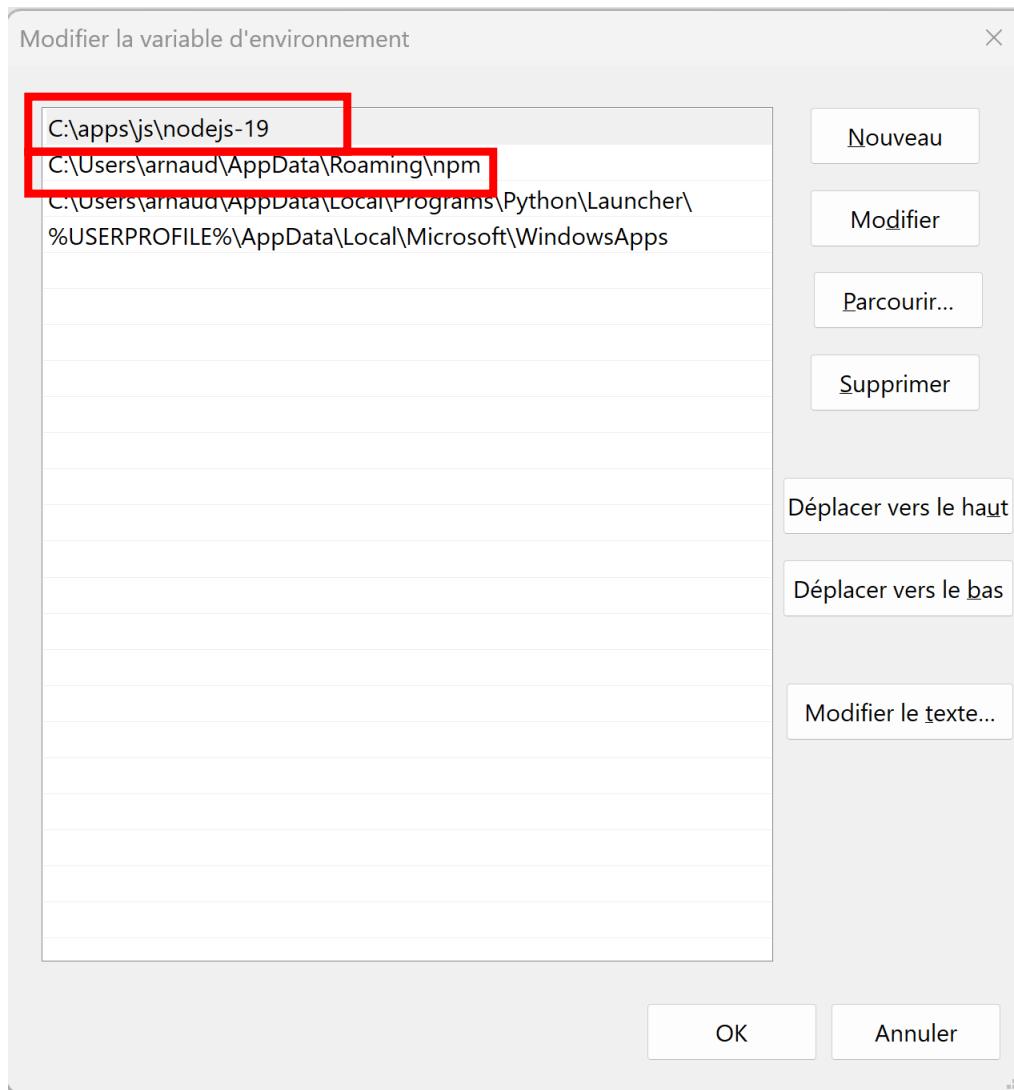


System Diagnostic

Edit {user | system} PATH environment variable



Prepend(=max priority) or Append to PATH



**Then restart
your terminals & apps !!**

(env vars are inherited during program starts)

Override in terminal to add in « %PATH% »
(may edit + call setenv.cmd)

set PATH=c:\your-node-dir;%PATH%

The screenshot shows a Windows Command Prompt window titled "Invite de commandes". The window displays the following text:

```
c:\ Invité de commandes
Microsoft Windows [version 10.0.22621.2134]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\arnaud>echo %PATH%
C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Git\cmd;C:\Program Files\foundationdb\bin;C:\Users\arnaud\AppData\Local\Programs\Python\Launcher\;C:\Users\arnaud\AppData\Local\Microsoft\WindowsApps;C:\Users\arnaud\AppData\Roaming\npm;

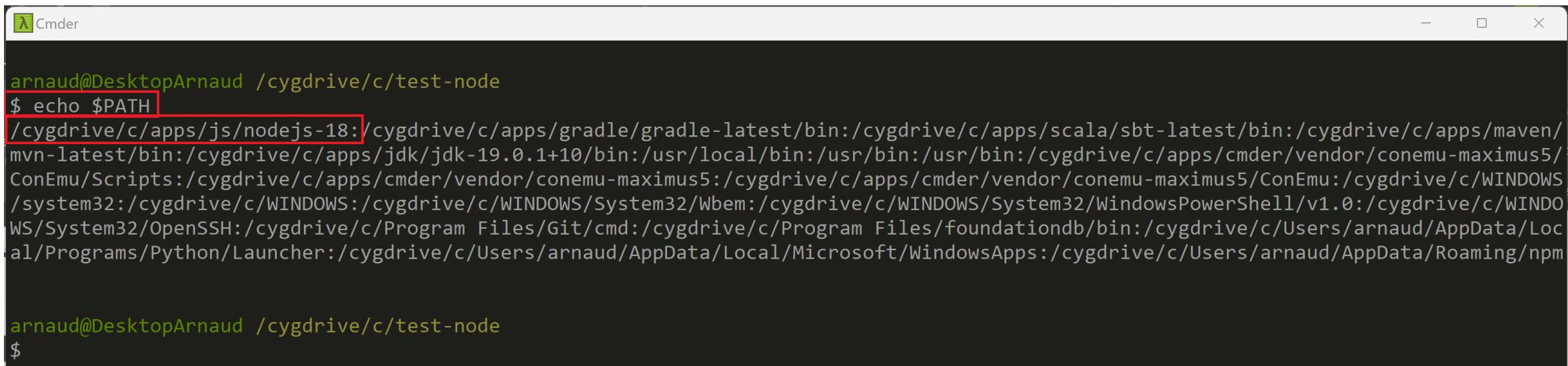
C:\Users\arnaud>set PATH=C:\apps\js\nodejs-19;%PATH%
C:\Users\arnaud>echo %PATH%
C:\apps\js\nodejs-19;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Git\cmd;C:\Program Files\foundationdb\bin;C:\Users\arnaud\AppData\Local\Programs\Python\Launcher\;C:\Users\arnaud\AppData\Local\Microsoft\WindowsApps;C:\Users\arnaud\AppData\Roaming\npm;

C:\Users\arnaud>
```

The commands entered by the user are highlighted with red boxes: "echo %PATH%", "set PATH=C:\apps\js\nodejs-19;%PATH%", and "echo %PATH%". The resulting PATH environment variable output also has red boxes around the first two entries: "C:\apps\js\nodejs-19;" and "C:\WINDOWS\system32;".

Case 2... using Cmder > Bash\$ Check NodeJS « \$PATH »

echo \$PATH



```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ echo $PATH
/cygdrive/c/apps/js/nodejs-18:/cygdrive/c/apps/gradle/gradle-latest/bin:/cygdrive/c/apps/scala/sbt-latest/bin:/cygdrive/c/apps/maven/mvn-latest/bin:/cygdrive/c/apps/jdk/jdk-19.0.1+10/bin:/usr/local/bin:/usr/bin:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu/Scripts:/cygdrive/c/apps/cmder/vendor/conemu-maximus5:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu:/cygdrive/c/WINDOWS/system32:/cygdrive/c/WINDOWS:/cygdrive/c/WINDOWS/System32/Wbem:/cygdrive/c/WINDOWS/System32/WindowsPowerShell/v1.0:/cygdrive/c/WINDOWS/System32/OpenSSH:/cygdrive/c/Program Files/Git/cmd:/cygdrive/c/Program Files/foundationdb/bin:/cygdrive/c/Users/arnaud/AppData/Local/Programs/Python/Launcher:/cygdrive/c/Users/arnaud/AppData/Local/Microsoft/WindowsApps:/cygdrive/c/Users/arnaud/AppData/Roaming/npm

arnaud@DesktopArnaud /cygdrive/c/test-node
$
```

Case 2 ... using Cmder Bash, Adding in \$PATH (may edit in your \$HOME/.profile or .bashrc)

export PATH=/c/your-node-dir:\$PATH

The screenshot shows a Cmder terminal window with the following session:

```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ echo $PATH
/cygdrive/c/apps/gradle/gradle-latest/bin:/cygdrive/c/apps/scala/sbt-latest/bin:/cygdrive/c/apps/maven/mvn-latest/bin:/cygdrive/c/apps/jdk/jdk-19.0.1+10/bin:/usr/local/bin:/usr/bin:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu/Scripts:/cygdrive/c/apps/cmder/vendor/conemu-maximus5:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu:/cygdrive/c/Windows/system32:/cygdrive/c/Windows:/cygdrive/c/Windows/System32/Wbem:/cygdrive/c/Windows/System32/WindowsPowerShell/v1.0:/cygdrive/c/Windows/System32/OpenSSH:/cygdrive/c/Program Files/Git/cmd:/cygdrive/c/Program Files/foundationdb/bin:/cygdrive/c/Users/arnaud/AppData/Local/Programs/Python/Launcher:/cygdrive/c/Users/arnaud/AppData/Local/Microsoft/WindowsApps:/cygdrive/c/Users/arnaud/AppData/Roaming/npm

arnaud@DesktopArnaud /cygdrive/c/test-node
$ node
c:\apps\cygwin64\bin\bash: node: command not found

arnaud@DesktopArnaud /cygdrive/c/test-node
$ export PATH=/cygdrive/c/apps/js/nodejs-19:$PATH

arnaud@DesktopArnaud /cygdrive/c/test-node
$ echo $PATH
/cygdrive/c/apps/js/nodejs-19:/cygdrive/c/apps/gradle/gradle-latest/bin:/cygdrive/c/apps/scala/sbt-latest/bin:/cygdrive/c/apps/maven/mvn-latest/bin:/cygdrive/c/apps/jdk/jdk-19.0.1+10/bin:/usr/local/bin:/usr/bin:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu/Scripts:/cygdrive/c/apps/cmder/vendor/conemu-maximus5:/cygdrive/c/apps/cmder/vendor/conemu-maximus5/ConEmu:/cygdrive/c/Windows/system32:/cygdrive/c/Windows:/cygdrive/c/Windows/System32/Wbem:/cygdrive/c/Windows/System32/WindowsPowerShell/v1.0:/cygdrive/c/Windows/System32/OpenSSH:/cygdrive/c/Program Files/Git/cmd:/cygdrive/c/Program Files/foundationdb/bin:/cygdrive/c/Users/arnaud/AppData/Local/Programs/Python/Launcher:/cygdrive/c/Users/arnaud/AppData/Local/Microsoft/WindowsApps:/cygdrive/c/Users/arnaud/AppData/Roaming/npm

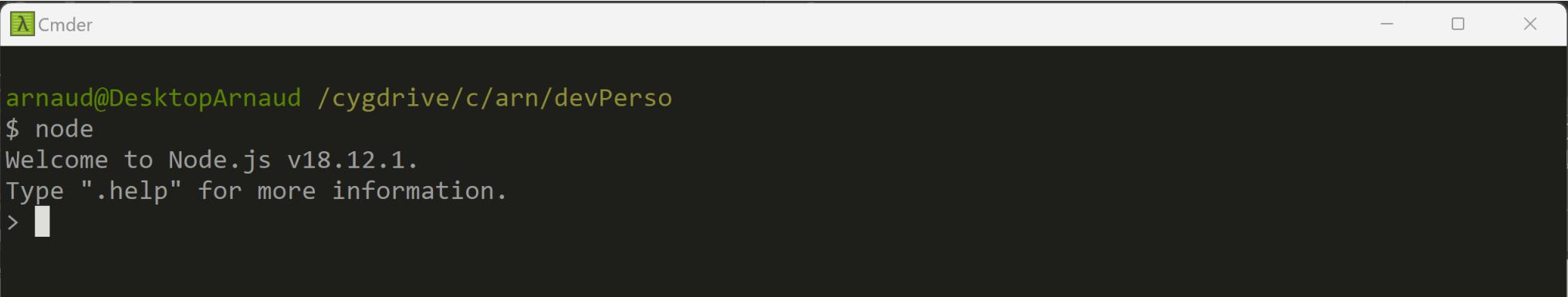
arnaud@DesktopArnaud /cygdrive/c/test-node
$ node -v
v19.1.0

arnaud@DesktopArnaud /cygdrive/c/test-node
$
```

The terminal shows the user's home directory as /cygdrive/c/test-node. They first check the current PATH, which includes various system and developer paths. They then attempt to run the node command, which fails because it is not in the PATH. To fix this, they add the path to their local node installation (c:\apps\js\nodejs-19) to the PATH variable using the export command. After doing so, they run node -v again, and it successfully outputs v19.1.0, indicating the command is now available.

Start NodeJs interactive prompt

C:> node

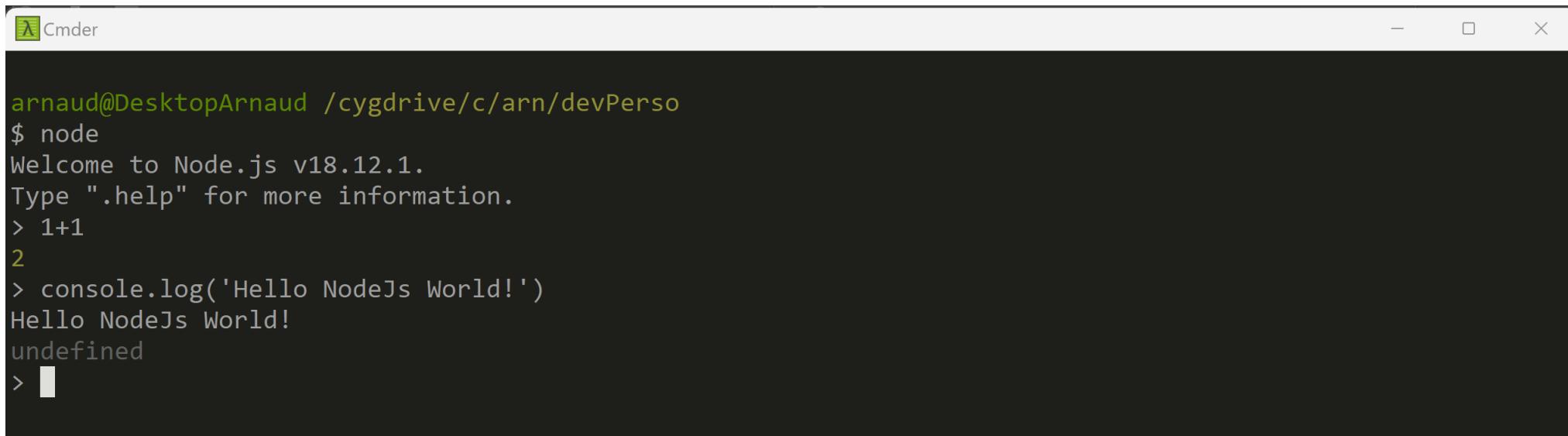


A screenshot of a terminal window titled "Cmder". The window shows a black terminal session. The user has run the command "\$ node" which has output the standard Node.js welcome message: "Welcome to Node.js v18.12.1. Type ".help" for more information." A cursor is visible at the end of the line starting with ">".

```
arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ node
Welcome to Node.js v18.12.1.
Type ".help" for more information.
> 
```

Play with interactive Javascript eval

1+1
console.log('Hello')

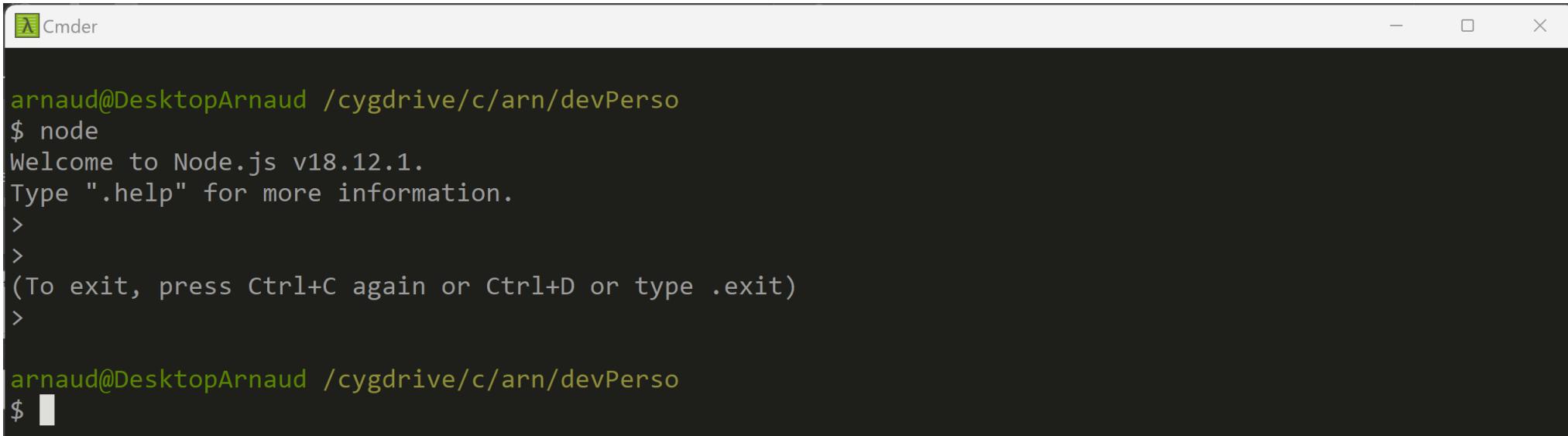


A screenshot of a terminal window titled "Cmder". The window shows a Node.js interactive session. The user has run the command "\$ node" and is now in the Node.js REPL. They have entered the expression "1+1" which evaluates to "2". They then ran the command "console.log('Hello NodeJS World!')". The output "Hello NodeJS World!" was displayed, followed by "undefined". A small white square icon is visible at the bottom left of the terminal window.

```
arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ node
Welcome to Node.js v18.12.1.
Type ".help" for more information.
> 1+1
2
> console.log('Hello NodeJS World!')
Hello NodeJS World!
undefined
> 
```

Stop Prompt ...

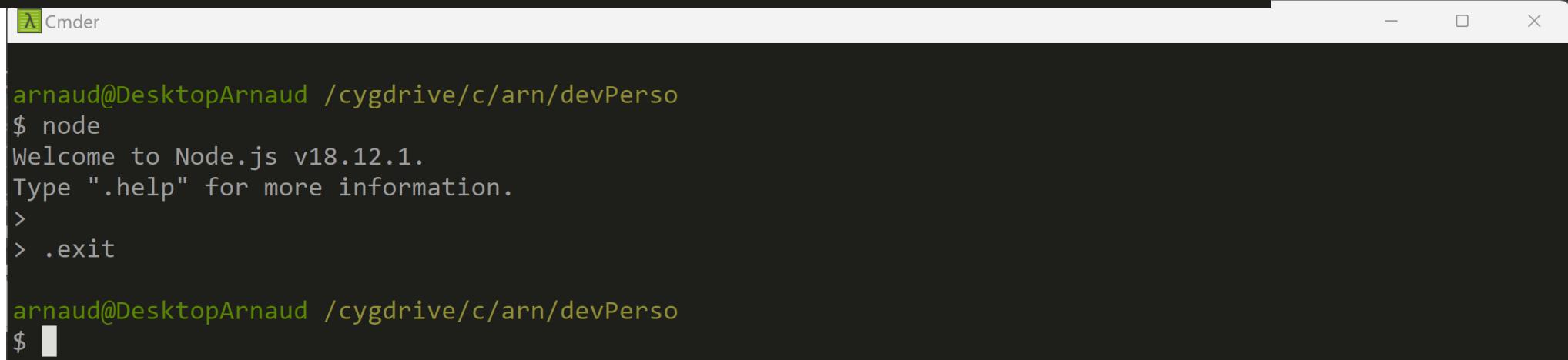
Ctrl+C Ctrl+C or .exit



A screenshot of a terminal window titled "Cmder". The window shows a command-line interface for Node.js version 18.12.1. The user has typed "\$ node" and is at the Node.js prompt, which displays "Welcome to Node.js v18.12.1." and "Type ".help" for more information.". The terminal also shows the message "(To exit, press Ctrl+C again or Ctrl+D or type .exit)". The cursor is currently at the end of the line where the user typed "\$".

```
arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ node
Welcome to Node.js v18.12.1.
Type ".help" for more information.
>
>
(To exit, press Ctrl+C again or Ctrl+D or type .exit)
>

arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ █
```



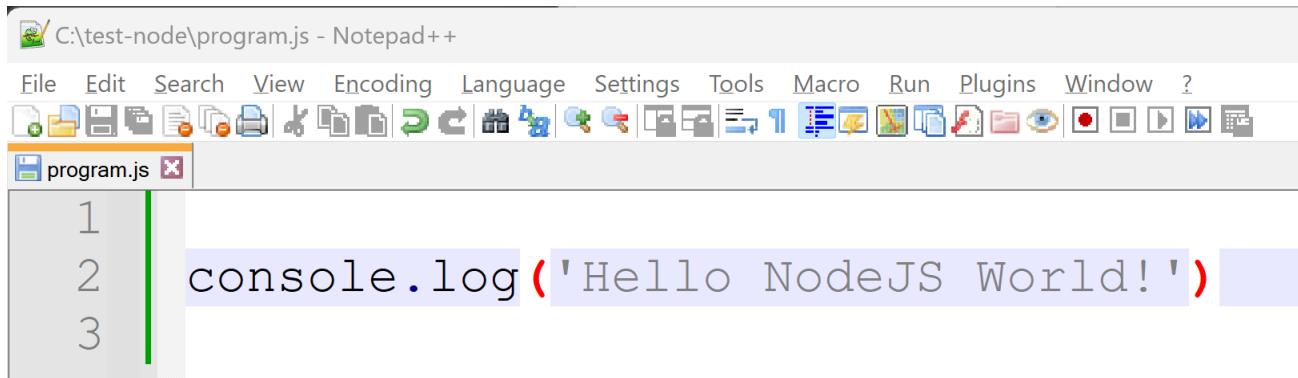
A screenshot of a terminal window titled "Cmder". The window shows a command-line interface for Node.js version 18.12.1. The user has typed "\$ node" and is at the Node.js prompt, which displays "Welcome to Node.js v18.12.1." and "Type ".help" for more information.". The user then typed ".exit" at the prompt. The terminal shows the message "(To exit, press Ctrl+C again or Ctrl+D or type .exit)". The cursor is currently at the end of the line where the user typed ".exit".

```
arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ node
Welcome to Node.js v18.12.1.
Type ".help" for more information.
>
> .exit

arnaud@DesktopArnaud /cygdrive/c/arn/devPerso
$ █
```

Start NodeJs on a « program.js » file

C:> node program.js



A screenshot of a terminal window titled "Cmder". The window shows the following session:

```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ node program.js
Hello NodeJS World!

arnaud@DesktopArnaud /cygdrive/c/test-node
$
```

Coffee Break (?)

Node ... need Package !!!

Npm = Node Package Manager

google.com/search?q=node+packages&rlz=1C1CHBF_frFR928FR928&oq=node+packages&aqs=chrome..69i57j0i512l5j0i390i650l2.5132j0j15&sourceid=chrome&ie=UTF-8

The screenshot shows a Google search results page for the query "node packages". The top navigation bar includes the Google logo, a search bar with the query "node packages", and various filters like Videos, Images, Shopping, News, Maps, Books, Flights, and Finance. Below the search bar, it says "About 539,000,000 results (0.60 seconds)". The first result is a link to "npm.js" with the URL "https://www.npmjs.com". The snippet describes the free npm Registry as the center of JavaScript code sharing, with over two million packages. Below the snippet are links to "Package", "About packages and modules", "Packages and modules", and "Using npm packages in your ...". To the right of the search results is a large image of the npm logo and some screenshots of the npm interface.

About 539,000,000 results (0.60 seconds)

npm
The free npm Registry has become the center of JavaScript code sharing, and with more than two million packages, the largest software registry in the world. Our ...

Package
Intro. This module provides an easy and simple way to export ...

About packages and modules
About modules. A module is any file or directory in the ...

Packages and modules
Documentation for the npm registry, website, and command-line ...

Using npm packages in your ...
If you are creating a Node.js module, you can use a package ...

npm
Computer program

npm is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node.js. It consists of a command line client, also called npm, and an online database of public and paid-for private packages,

C:> npm

```
Cmder
arnaud@DesktopArnaud /cygdrive/c/test-node
$ npm
npm <command>

Usage:

npm install      install all the dependencies in your project
npm install <foo> add the <foo> dependency to your project
npm test         run this project's tests
npm run <foo>   run the script named <foo>
npm <command> -h quick help on <command>
npm -l          display usage info for all commands
npm help <term> search for help on <term> (in a browser)
npm help npm    more involved overview (in a browser)

All commands:

access, adduser, audit, bugs, cache, ci, completion,
config, dedupe, deprecate, diff, dist-tag, docs, doctor,
edit, exec, explain, explore, find-dupes, fund, get, help,
help-search, hook, init, install, install-ci-test,
install-test, link, ll, login, logout, ls, org, outdated,
owner, pack, ping, pkg, prefix, profile, prune, publish,
query, rebuild, repo, restart, root, run-script, search,
set, shrinkwrap, star, stars, start, stop, team, test,
token, uninstall, unpublish, unstar, update, version, view,
whoami

Specify configs in the ini-formatted file:
  C:\apps\cygwin64\home\arnaud\.npmrc
```

\$ npm init

```
$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

```
See `npm help init` for definitive documentation on these fields
and exactly what they do.
```

```
Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
```

```
package name: (test-node)
```

```
version: (1.0.0)
```

```
description:
```

```
entry point: (index.js)
```

```
test command:
```

```
git repository:
```

```
keywords:
```

```
author:
```

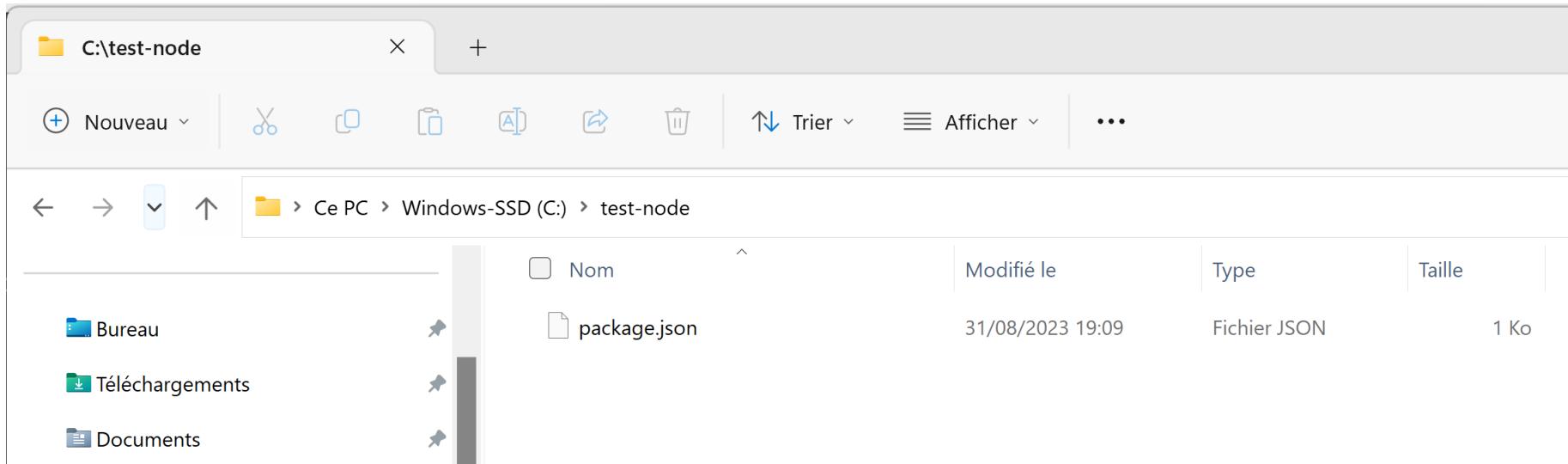
```
license: (ISC)
```

```
About to write to C:\arn\devPerso\Presentations\web\test-node\package.json:
```

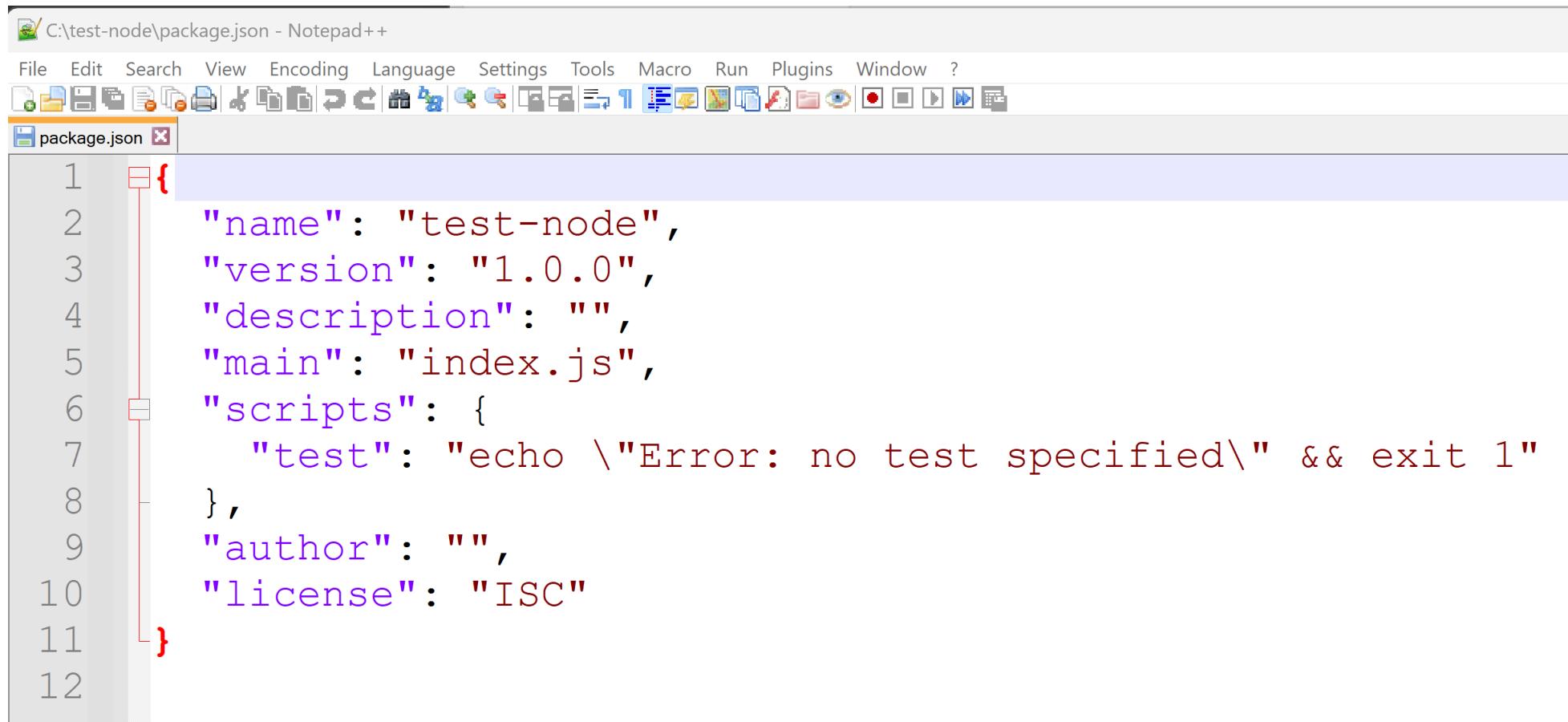
```
{
  "name": "test-node",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

```
Is this OK? (yes) █
```

npm init => open File explorer
=> 1 File « package.json »



package.json



A screenshot of the Notepad++ text editor showing the contents of a `package.json` file. The file path is `C:\test-node\package.json`. The code is as follows:

```
1 {  
2   "name": "test-node",  
3   "version": "1.0.0",  
4   "description": "",  
5   "main": "index.js",  
6   "scripts": {  
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"  
8   },  
9   "author": "",  
10  "license": "ISC"  
11 }  
12
```

The code is color-coded: curly braces (`{`, `}`) are red, strings are purple, and the file number column (1-12) is light blue.

Search Package dependencies ? ...

Npm search ... Registry (=Repository)

A screenshot of a Google search results page. The search bar at the top contains the query "npm search". Below the search bar, there are several navigation links: Images, Videos, News, Shopping, Books, Maps, Flights, and Finance. The main search results section starts with a message indicating approximately 192 million results found in 0.39 seconds. The first result is a link to "npm Docs" from "https://docs.npmjs.com", which points to the "cli > commands > npm-search" section. The second result is a link to "npm-search" with a brief description: "Search the registry for packages matching the search terms. **npm search** performs a linear, incremental, lexically-ordered search through package metadata for ...". Below this, there are links to "Synopsis · Description · Configuration". The third result is a link to "npm.js" from "https://www.npmjs.com", which points to the "npm" section. The description for this result states: "The free **npm** Registry has become the center of JavaScript code sharing, and with more than two million packages, the largest software registry in the world. Our ...". Below this, there are links to "Express · React · About npm · Npm Docs".

← → C google.com/search?q=npm+search&sca_esv=561694184&rlz=1C1CHBF_frFR928FR928&sxsrf=AB5stBiQAUFBtC8abXy4KljzfKDHH2f3cg%3A16

Google

npm search

X |

Images Videos News Shopping Books Maps Flights Finance

About 192,000,000 results (0.39 seconds)

[npm Docs](#)
<https://docs.npmjs.com> › cli › commands › npm-search

[:::](#)

[npm-search](#)

Search the registry for packages matching the search terms. **npm search** performs a linear, incremental, lexically-ordered search through package metadata for ...

[Synopsis](#) · [Description](#) · [Configuration](#)

[npm.js](#)
<https://www.npmjs.com>

[:::](#)

[npm](#)

The free **npm** Registry has become the center of JavaScript code sharing, and with more than two million packages, the largest software registry in the world. Our ...

[Express](#) · [React](#) · [About npm](#) · [Npm Docs](#)

<https://npmjs.com>

A screenshot of the npmjs.com homepage. The top navigation bar includes links for Pro, Teams, Pricing, and Documentation. A search bar with a placeholder 'Search packages' and a 'Search' button is present. On the left, there's a user profile for 'No Pants Monkey'. The main banner features a large, bold white text 'Build amazing things' on a red-to-orange gradient background with abstract geometric shapes. Below the banner, a paragraph of text reads: 'We're GitHub, the company behind the npm Registry and npm CLI. We offer those to the community for free, but our day job is building and selling useful tools for developers like you.' At the bottom, the text 'Take your JavaScript' is partially visible.

Search « express » package

The screenshot shows the npmjs.com search interface. The search bar at the top contains the query "express". Below the search bar, a list of packages is displayed, each with a name, description, and version number. To the right of the list is a large, colorful graphic consisting of overlapping orange and red shapes.

Package	Description	Version
<code>express</code>	Fast, unopinionated, minimalist web framework	4.18.2
<code>express-validator</code>	Express middleware for the validator module.	7.0.1
<code>express-handlebars</code>	A Handlebars view engine for Express which doesn't suck.	7.1.2
<code>express-session</code>	Simple session middleware for Express	1.17.3
<code>express-brute</code>	A brute-force protection middleware for express routes that rate limits incoming requests	1.0.1
<code>express-favicon</code>	Favicon middleware for express-like applications	2.0.4
<code>express-rate-limit</code>	Basic IP rate-limiting middleware for Express. Use to limit repeated requests to public APIs and/or endpoints such as password reset.	6.10.0
<code>express-promise-router</code>	A lightweight wrapper for Express 4's Router that allows middleware to return promises	4.1.1
<code>express-fileupload</code>	Simple express file upload middleware that wraps around Busboy	1.4.0
<code>express-openapi-validator</code>	Automatically validate API requests and responses with OpenAPI 3 and Express.	5.0.6

Package express ... version / doc / install

npmjs.com/package/express

Pro Teams Pricing Documentation

Nestable Processes Mutate

Search packages

Search Sign Up Sign In

express DT

4.18.2 • Public • Published a year ago

Readme Code Beta 31 Dependencies 72,994 Dependents 270 Versions

express

Fast, unopinionated, minimalist web framework for [Node.js](#).

npm v4.18.2 install size 1.89 MB downloads 123.2M/month

```
const express = require('express')
const app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})
```

Install

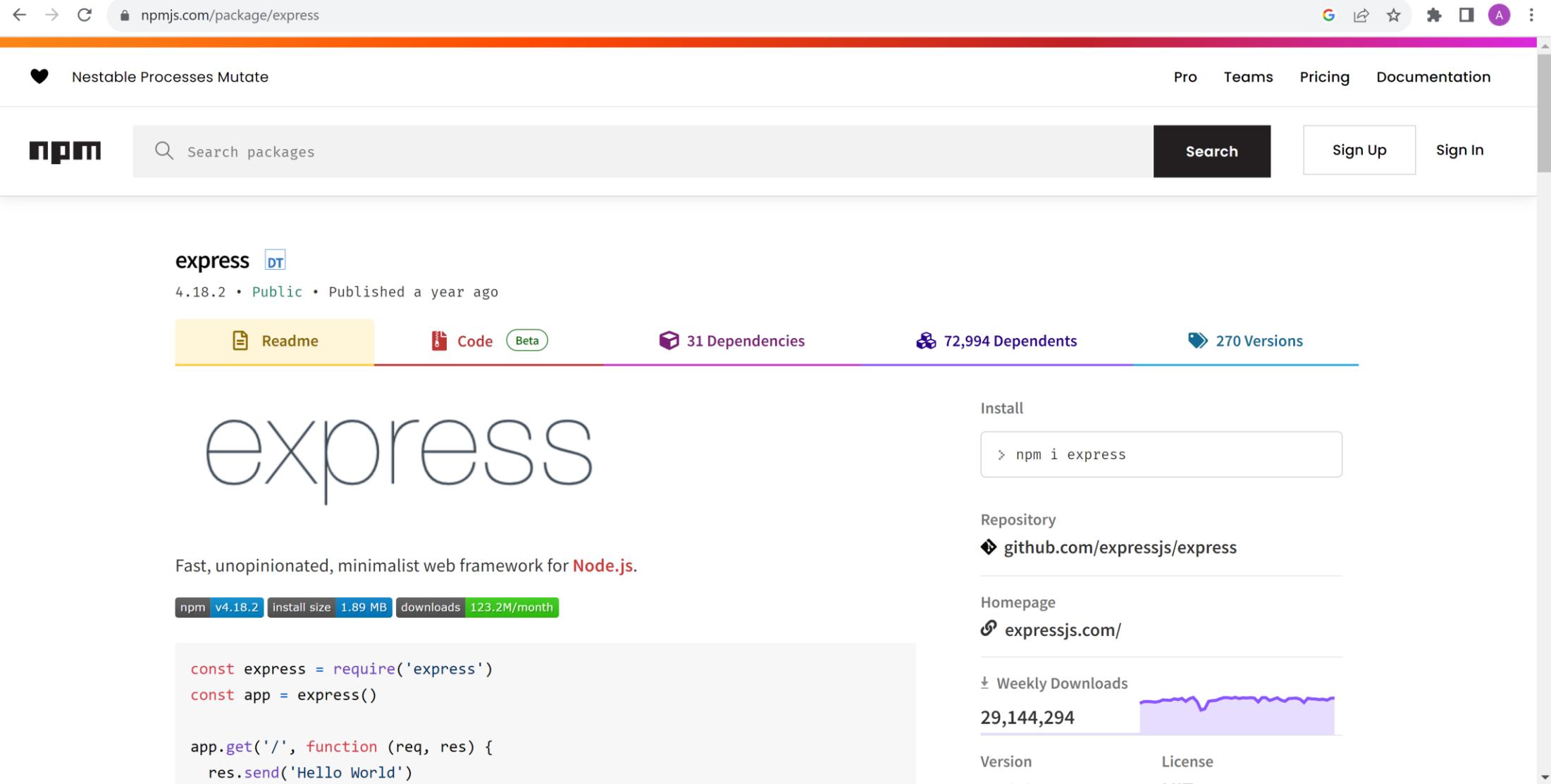
```
npm i express
```

Repository github.com/expressjs/express

Homepage expressjs.com/

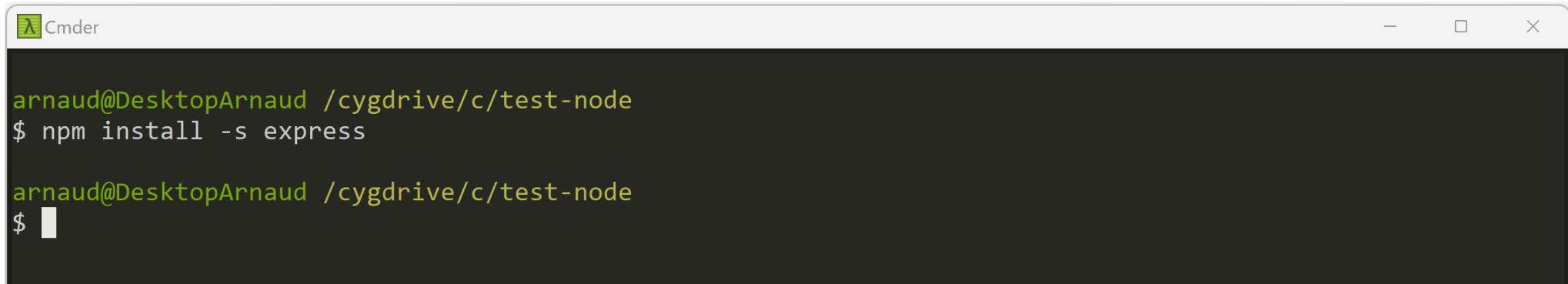
Weekly Downloads 29,144,294

Version License



Add npm dependency « express »

C:> npm install –s express
(equivalent: npm install --save express)



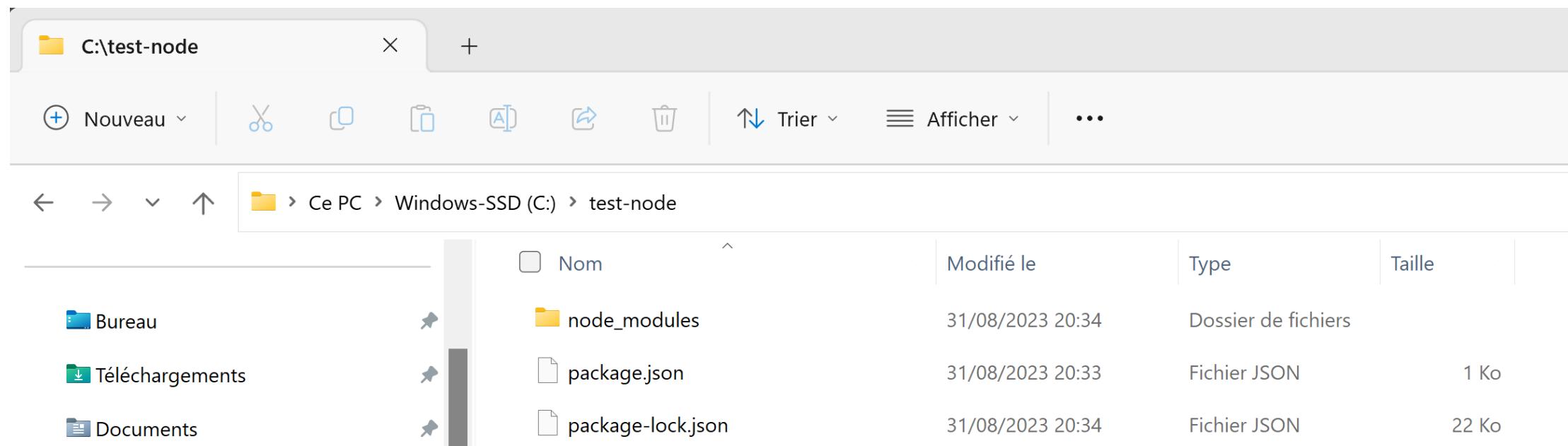
```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ npm install -s express

arnaud@DesktopArnaud /cygdrive/c/test-node
$ █
```

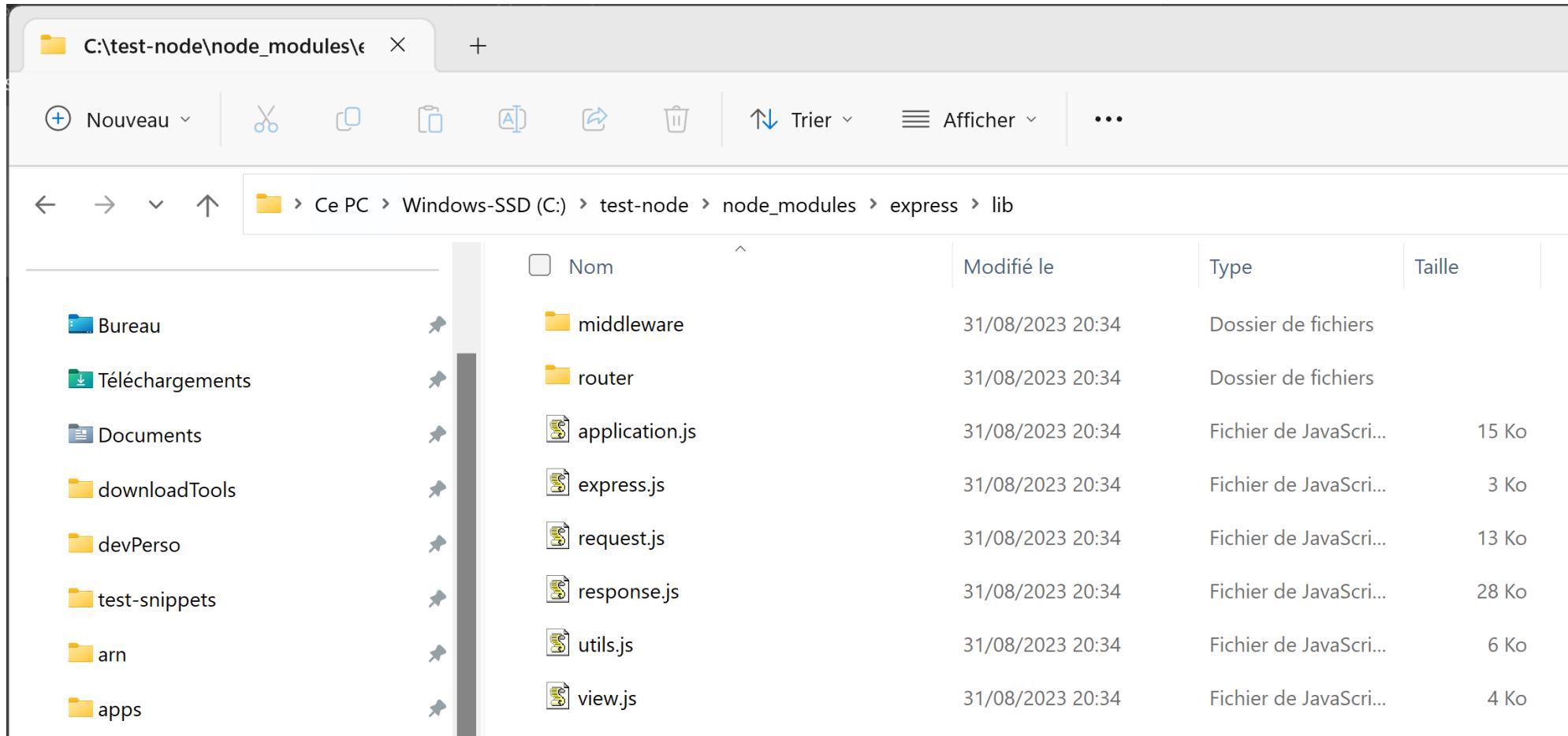
A screenshot of a terminal window titled "Cmder". The window has a dark background and light-colored text. It shows a command-line session where the user is in a directory called "test-node". The user types the command "npm install -s express" and presses Enter. The terminal then displays the command again followed by a prompt symbol (\$) and a small black square cursor.

npm --save => save to package.json file
+ package-lock.json

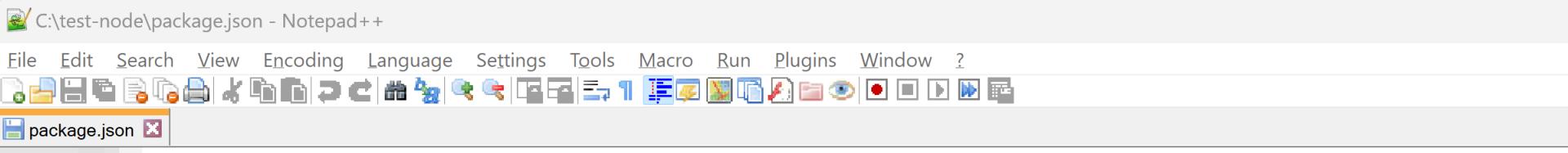
+ download **/** to node_modules/



node_modules/express/lib/**/*.js



... 1 Line added in package.json dependencies: { «pck-name» : « version », ... }



C:\test-node\package.json - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

package.json

```
1  {
2      "name": "test-node",
3      "version": "1.0.0",
4      "description": "",
5      "main": "index.js",
6      "scripts": {
7          "test": "echo \\\"Error: no test specified\\\" && exit 1"
8      },
9      "author": "",
10     "license": "ISC",
11     "dependencies": {
12         "express": "^4.18.2"
13     }
14 }
15 }
```

<https://docs.npmjs.com/about-semantic-versioning>

Using semantic versioning to specify update types your package can accept

You can specify which update types your package can accept from dependencies in your package's `package.json` file.

For example, to specify acceptable version ranges up to 1.0.4, use the following syntax:

- Patch releases: `1.0` or `1.0.x` or `~1.0.4`
- Minor releases: `1` or `1.x` or `^1.0.4`
- Major releases: `*` or `x`

For more information on semantic versioning syntax, see the [npm semver calculator](#).

Example

```
"dependencies": {  
  "my_dep": "^1.0.0",  
  "another_dep": "~2.2.0"  
},
```



Locking versions number ... package-lock.json

The screenshot shows the Notepad++ interface with the title bar "C:\test-node\package-lock.json - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and Help. The toolbar contains various icons for file operations like Open, Save, Find, and Copy. There are two tabs open: "package.json" and "package-lock.json". The "package-lock.json" tab is active and displays a large JSON object representing the dependency tree for the project. The JSON structure includes fields for name, version, lockfileVersion, requires, packages, and various node modules like accept, array-flatten, body-parser, bytes, call-bind, content-disposition, and content-type. Each node module entry contains details such as version, resolved URL, integrity hash, dependencies, engines, and peerDependencies. A red box highlights the right edge of the code editor, indicating scrollable content.

```
1 {  
2   "name": "test-node",  
3   "version": "1.0.0",  
4   "lockfileVersion": 3,  
5   "requires": true,  
6   "packages": [  
7     {  
8       "name": "test-node",  
9       "version": "1.0.0",  
10      "license": null,  
11      "dependencies": {}  
12      "peerDependencies": {}  
13    }  
14  ],  
15  "node_modules/accepts": {  
16    "version": "1.1.0",  
17    "resolved": "https://registry.npmjs.org/accepts/-/accepts-1.1.0.tgz",  
18    "integrity": "sha512-pYAtchTz2m2VXuvSD3DPO/Gy+U+sOAI1A7SmRuvw+NAC5aeXEo+NHoV-----F7rON16qcaX3Uuemaww+7+SJLw==",  
19    "dependencies": {}  
20    "mime-types": "2.1.34",  
21    "negotiator": "0.6.3"  
22  },  
23  "engines": {  
24    "node": ">= 0.6"  
25  },  
26  "node_modules/array-flatten": {  
27    "version": "1.1.1",  
28    "resolved": "https://registry.npmjs.org/array-flatten/-/array-flatten-1.1.1.tgz",  
29    "integrity": "sha512-pYQgRaod3UVm51jHwynguOwAvYPhx8nNmI+NqRcK6CxwpUsfjhIdKiHiBqg==",  
30  },  
31  "node_modules/body-parser": {  
32    "version": "1.20.1",  
33    "resolved": "https://registry.npmjs.org/body-parser/-/body-parser-1.20.1.tgz",  
34    "integrity": "sha512-+WjzAbTbYwjeOwTWCQc37VuImWiRae5RyTpCyDS51MdTwSz1OpDE67srw/Hy3Sfz1zfDQw+3txg7gNtW==",  
35    "dependencies": {}  
36    "bycrypt": "3.2.0",  
37    "content-type": "1.0.4",  
38    "debug": "2.6.9",  
39    "depd": "2.0.0",  
40    "destroy": "1.2.0",  
41    "http-errors": "2.0.0",  
42    "isarray": "0.4.2",  
43    "on-finished": "2.4.1",  
44    "qs": "6.11.0",  
45    "raw-body": "2.5.1",  
46    "type-is": "1.6.18",  
47    "unpipe": "1.0.0"  
48  },  
49  "engines": {  
50    "node": ">= 0.8",  
51    "npm": "1.2.000 || >= 1.4.16"  
52  },  
53  "node_modules/bytes": {  
54    "version": "3.2.2",  
55    "resolved": "https://registry.npmjs.org/bytes/-/bytes-3.2.2.tgz",  
56    "integrity": "sha512-Nf7TyzTx6SyRJObOA7956r8cr2+OjAC5dtBwSFQoeX58NhCu8PbzGkNNSjtTSi6fzO6FopBdcYbEg==",  
57    "engines": {}  
58    "node": ">= 0.8"  
59  },  
60  "node_modules/call-bind": {  
61    "version": "1.0.2",  
62    "resolved": "https://registry.npmjs.org/call-bind/-/call-bind-1.0.2.tgz",  
63    "integrity": "sha512-T0+jFcIhB5WGbFYeuctwmTk6eO1lzmzicreWj+0a7LJfughEsoT9ncpOfzMQzxfJQGgueWJGTYsqrA==",  
64    "dependencies": {}  
65    "function-bind": "1.1.1",  
66    "get-intrinsic": "1.0.2"  
67  },  
68  "funding": {  
69    "url": "https://github.com/sponsors/ljharb"  
70  },  
71  "node_modules/content-disposition": {  
72    "version": "0.5.4",  
73    "resolved": "https://registry.npmjs.org/content-disposition/-/content-disposition-0.5.4.tgz",  
74    "integrity": "sha512-Pve2TNUGw04cxIA1WbzieZtAL/lhehaWB7tguJh4/E5DqhtThma3KZNlaAWA8cFihHzM2ULevkwsRqk+tSQ==",  
75    "dependencies": {}  
76    "safe-buffer": "5.2.1"  
77  },  
78  "engines": {  
79    "node": ">= 0.6"  
80  },  
81  "node_modules/content-type": {  
82    "version": "1.0.5",  
83    "resolved": "https://registry.npmjs.org/content-type/-/content-type-1.0.5.tgz",  
84    "integrity": "sha512-nTjQfcBFElKGKV4YDQWcfmLZE81ldF0pApTvyFGvbcR6P/VAdSG7N+0Tr8QqlU0tFadDefK4NtQwOA==",  
85    "engines": {}  
86  }
```

File is **HUGE** ...
Like expanded package/package/package !!!
ONLY « express » → 605 lines



The screenshot shows a Notepad++ window with three tabs open: package.json, package.json, and package-lock.json. The package-lock.json tab is active and displays the following JSON content:

```
579     "node_modules/unpipe": {  
580         "version": "1.0.0",  
581         "resolved": "https://registry.npmjs.org/unpipe/-/unpipe-1.0.0.tgz",  
582         "integrity": "sha512-pjy2bYhSsufwWlKwPc+l3cN7+wuJlK6uz0YdJE0lQDb16jo/YlPi/  
583         "engines": {  
584             "node": ">= 0.8"  
585         }  
586     },  
587     "node_modules/utils-merge": {  
588         "version": "1.0.1",  
589         "resolved": "https://registry.npmjs.org/utils-merge/-/utils-merge-1.0.1.tgz",  
590         "integrity": "sha512-pMZTvIkTld+TFGvDOqodOclx0QWkkgi6Tdoa8gC8ffGAAqz9pzPT:  
591         "engines": {  
592             "node": ">= 0.4.0"  
593         }  
594     },  
595     "node_modules/vary": {  
596         "version": "1.1.2",  
597         "resolved": "https://registry.npmjs.org/vary/-/vary-1.1.2.tgz",  
598         "integrity": "sha512-BNGbWLfd0eUPabhkXUVm0j8uuvREyTh5ovRa/dyow/BqAbZJyC+5:  
599         "engines": {  
600             "node": ">= 0.8"  
601         }  
602     },  
603 }  
604 }  
605 }
```

`node_modules/*/*/*/*/*.js`

Directory `node_modules`

1/ may be BIG

.... ~300 Mega is frequent with few dependencies

\$ du -sh node_modules

2/ can be deleted + re-downloaded safely

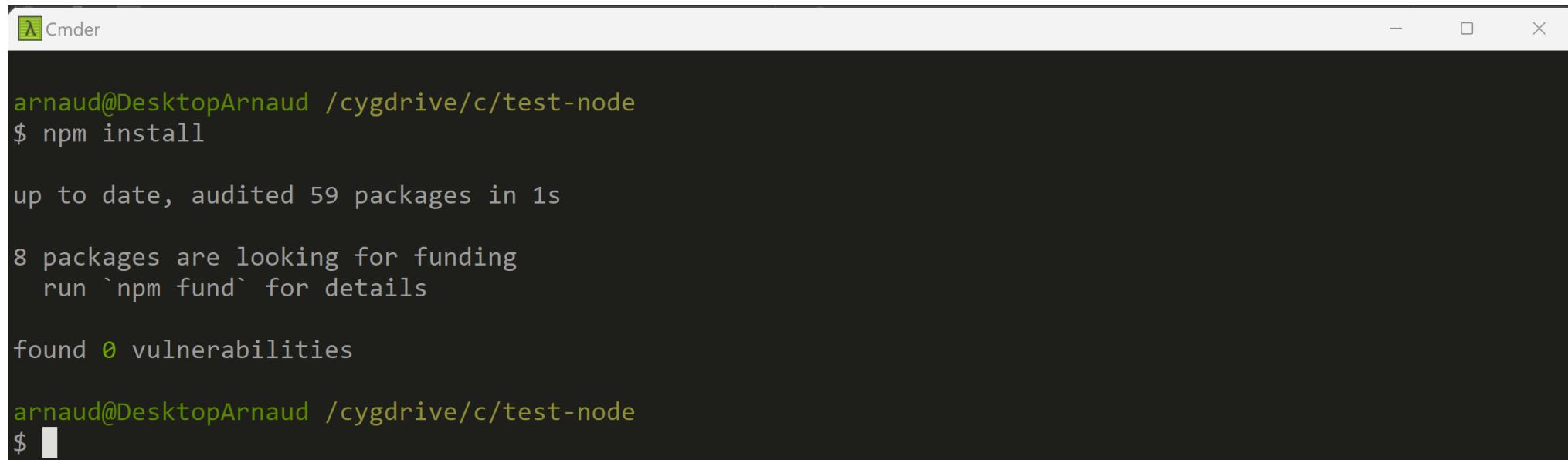
\$ rm -rf node_modules; npm install

3/ should NEVER be committed to any git repository

\$ echo node_modules/ >> .gitignore

After getting some web project source code...
=> downloads to re-fill node_modules/**
as defined in package.json

\$ npm install



```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ npm install

up to date, audited 59 packages in 1s

8 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

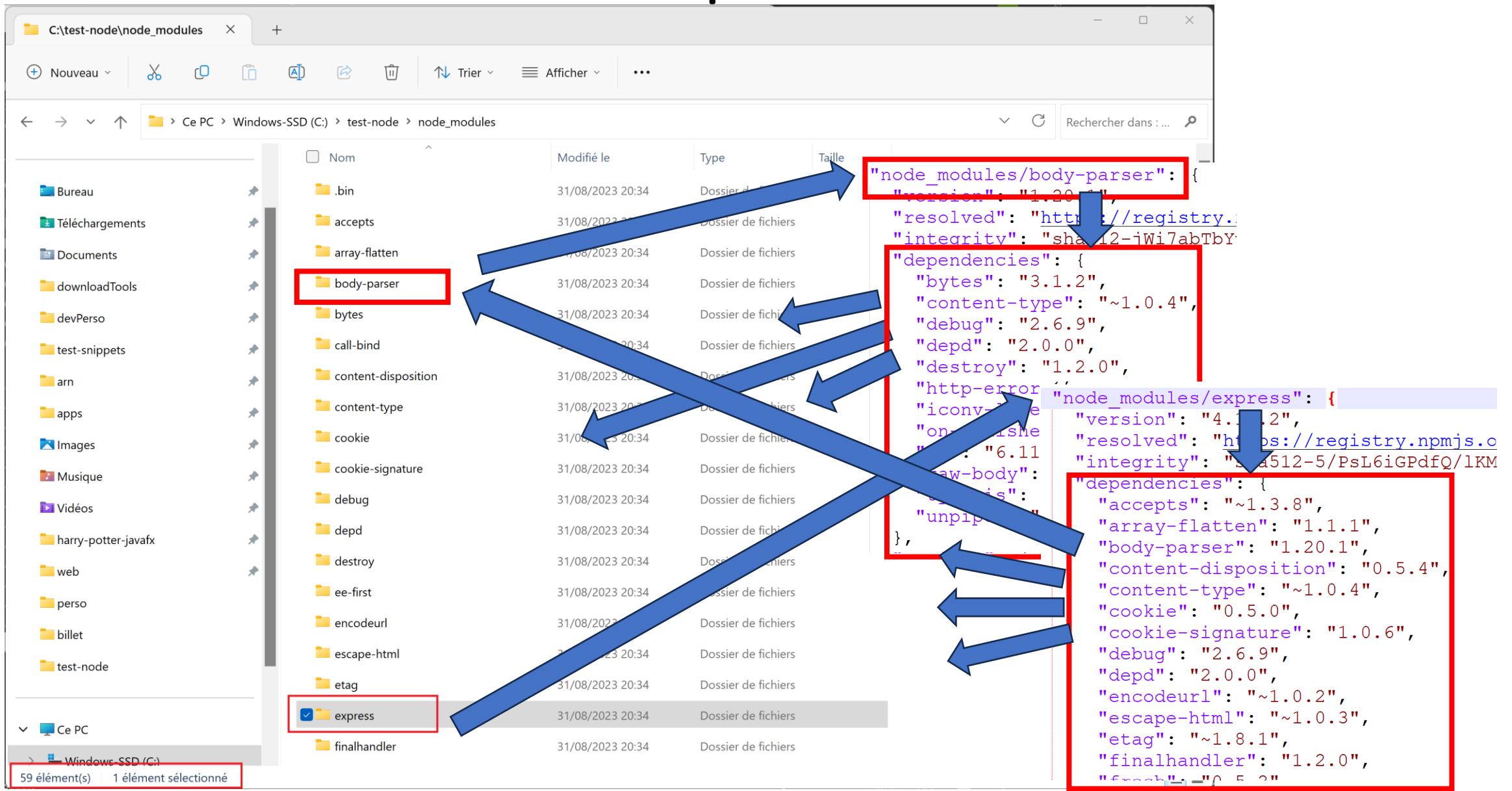
arnaud@DesktopArnaud /cygdrive/c/test-node
$ █
```

node_modules/express/package.json

The image shows two side-by-side Notepad++ windows displaying the same JSON file, `package.json`, for the `express` module. The left window covers lines 1 to 22, and the right window covers lines 27 to 50. Both windows have identical toolbars and menubars.

```
1 {  
2   "name": "express",  
3   "description": "Fast, unopinionated, minimalist web framework",  
4   "version": "4.18.2",  
5   "author": "TJ Holowaychuk <tj@vision-media.ca>",  
6   "contributors": [  
7     "Aaron Heckmann <aaron.heckmann+github@gmail.com>",  
8     "Ciaran Jessup <ciaranj@gmail.com>",  
9     "Douglas Christopher Wilson <doug@somethingdoug.com>",  
10    "Guillermo Rauch <rauchg@gmail.com>",  
11    "Jonathan Ong <me@jongleberry.com>",  
12    "Roman Shtylman <shtylman+expressjs@gmail.com>",  
13    "Young Jae Sim <hanul@hanul.me>"  
14  ],  
15  "license": "MIT",  
16  "repository": "expressjs/express",  
17  "homepage": "http://expressjs.com/",  
18  "keywords": [  
19    "express",  
20    "framework",  
21    "sinatra",  
22    "web"  
],  
23  "dependencies": {  
24    "accepts": "~1.3.8",  
25    "array-flatten": "1.1.1",  
26    "body-parser": "1.20.1",  
27    "content-disposition": "0.5.4",  
28    "content-type": "~1.0.4",  
29    "cookie": "0.5.0",  
30    "cookie-signature": "1.0.6",  
31    "debug": "2.6.9",  
32    "depd": "2.0.0",  
33    "encodeurl": "~1.0.2",  
34    "escape-html": "~1.0.3",  
35    "etag": "~1.8.1",  
36    "finalhandler": "1.2.0",  
37    "fresh": "0.5.2",  
38    "http-errors": "2.0.0",  
39    "merge-descriptors": "1.0.1",  
40    "methods": "~1.1.2",  
41    "on-finished": "2.4.1",  
42    "parseurl": "~1.3.3",  
43    "path-to-regexp": "0.1.7",  
44  },  
45  "devDependencies": {  
46    "body-parser": "1.19.0",  
47    "content-disposition": "0.5.4",  
48    "content-type": "1.2.7",  
49    "cookie": "0.5.0",  
50    "cookie-signature": "1.0.6",  
51    "debug": "2.6.9",  
52    "depd": "2.0.0",  
53    "encodeurl": "1.0.2",  
54    "escape-html": "1.0.3",  
55    "etag": "1.8.1",  
56    "finalhandler": "1.2.0",  
57    "fresh": "0.5.2",  
58    "http-errors": "2.0.0",  
59    "merge-descriptors": "1.0.1",  
60    "methods": "1.1.2",  
61    "on-finished": "2.4.1",  
62    "parseurl": "1.3.3",  
63    "path-to-regexp": "0.1.7",  
64    "serve-static": "1.13.1",  
65    "statuses": "1.5.0",  
66    "type-is": "1.6.17",  
67    "util-deprecate": "1.0.2",  
68    "vary": "1.0.3",  
69    "ws": "3.4.3",  
70  },  
71  "optionalDependencies": {  
72    "array-flatten": "1.1.1",  
73    "body-parser": "1.19.0",  
74    "content-disposition": "0.5.4",  
75    "content-type": "1.2.7",  
76    "cookie": "0.5.0",  
77    "cookie-signature": "1.0.6",  
78    "debug": "2.6.9",  
79    "depd": "2.0.0",  
80    "encodeurl": "1.0.2",  
81    "escape-html": "1.0.3",  
82    "etag": "1.8.1",  
83    "finalhandler": "1.2.0",  
84    "fresh": "0.5.2",  
85    "http-errors": "2.0.0",  
86    "merge-descriptors": "1.0.1",  
87    "methods": "1.1.2",  
88    "on-finished": "2.4.1",  
89    "parseurl": "1.3.3",  
90    "path-to-regexp": "0.1.7",  
91    "serve-static": "1.13.1",  
92    "statuses": "1.5.0",  
93    "type-is": "1.6.17",  
94    "util-deprecate": "1.0.2",  
95    "vary": "1.0.3",  
96    "ws": "3.4.3",  
97  }  
},  
98  "scripts": {  
99    "start": "node ./bin/www",  
100   "test": "node ./bin/test"  
101 },  
102 }
```

package.json → package.json → ... package.json
transitive dependencies



npm ls --all

```
arneau@DesktopArnaud /cygdrive/c/test-node
$ npm ls --all
test-node@1.0.0 C:\test-node
`-- express@4.18.2
  +-+ accepts@1.3.8
  | +-+ mime-types@2.1.35
  | | `-- mime-db@1.52.0
  | | `-- negotiator@0.6.3
  | +-+ array-flatten@1.1.1
  | +-+ body-parser@1.20.1
  | | +-+ bytes@3.1.2
  | | +-+ content-type@1.0.5 deduped
  | | +-+ debug@2.6.9 deduped
  | | +-+ depd@2.0.0 deduped
  | | +-+ destroy@1.2.0
  | | +-+ http-errors@2.0.0 deduped
  | | +-+ iconv-lite@0.4.24
  | | | `-- safer-buffer@2.1.2
  | | +-+ on-finished@2.4.1 deduped
  | | +-+ qs@6.11.0 deduped
  | | +-+ raw-body@2.5.1
  | | | +-+ bytes@3.1.2 deduped
  | | | +-+ http-errors@2.0.0 deduped
  | | | +-+ iconv-lite@0.4.24 deduped
  | | | `-- unpipe@1.0.0 deduped
  | | +-+ type-is@1.6.18 deduped
  | | | `-- unpipe@1.0.0
  | | +-+ content-disposition@0.5.4
  | | | `-- safe-buffer@5.2.1 deduped
  | | +-+ content-type@1.0.5
  | | +-+ cookie-signature@1.0.6
  | | +-+ cookie@0.5.0
  | | +-+ debug@2.6.9
  | | | `-- ms@2.0.0
  | | +-+ depd@2.0.0
```

Coffee break (?)

Choose a Text Editor

Notepad++

SublimeText

Vi Notepad

Wordpad

IntelliJ Ultimate

WebStorm

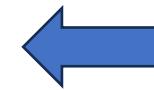
VisualStudio Code

Eclipse

NetBeans

Choose (~~NOT~~ a Text Editor) a modern IDE

IntelliJ Ultimate



€€€\$\$\$ (or 30 days)

WebStorm



€€\$\$ (or 30 days)

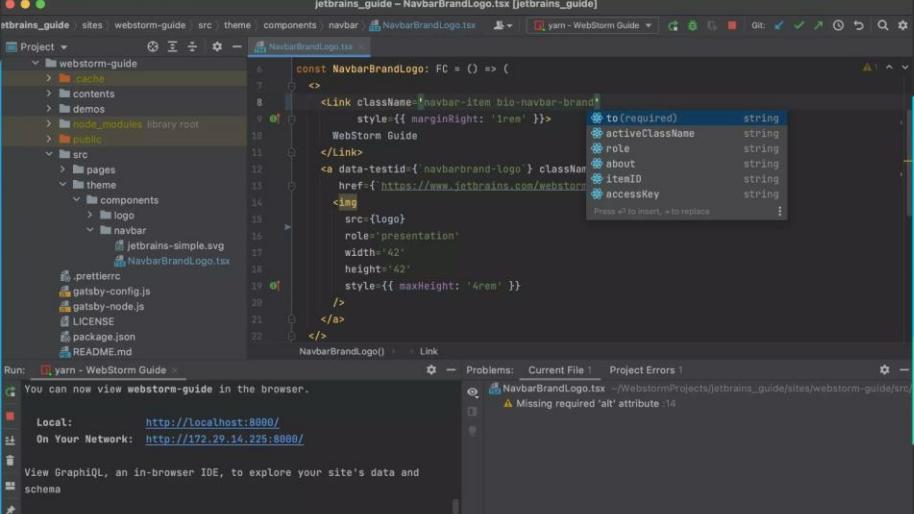
VisualStudio Code

Eclipse + plugins

Eclipse (plugins not pre-installed)

IntelliJ (No web js Support)

<https://www.jetbrains.com/webstorm/>

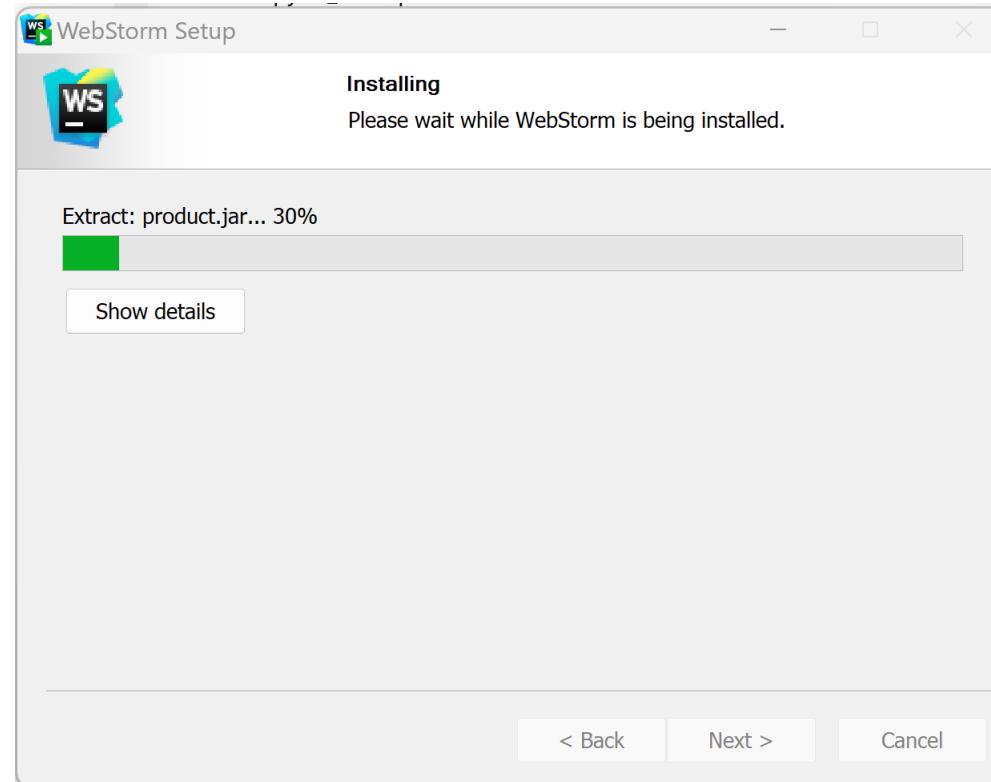


The screenshot shows the WebStorm IDE interface. On the left, there's a dark sidebar with the 'JET BRAINS' logo and a horizontal bar with a minus sign. The main content area features a large 'WebStorm' logo with a stylized 'WS' icon. Below it, the text 'The smartest JavaScript IDE' is displayed. To the right, a large blue and cyan geometric graphic is overlaid. The central part of the screen shows the IDE's code editor with a file named 'NavbarBrandLogo.tsx'. The code is as follows:

```
const NavbarBrandLogo: FC = () => {
  <Link className="navbar-item bio-navbar-brand" style={{ marginRight: '1rem' }}>
    WebStorm Guide
    <a data-testid="navbarbrand-logo" className="navbar-item" href="https://www.jetbrains.com/webstorm/">
      <img alt="jetbrains-simple.svg" src={logo} role="presentation" width="42" height="42" style={{ maxHeight: '4rem' }}/>
    </a>
  </Link>
}
```

The IDE also displays a 'Project' tree on the left, showing a 'webstorm-guide' folder containing 'cache', 'contents', 'demos', 'node_modules', 'public', 'src', 'theme', and 'theme/components/navbar'. The 'NavbarBrandLogo.tsx' file is selected. At the bottom, there are tabs for 'Run', 'Problems', and 'Current File'. A message in the 'Run' tab says 'You can now view webstorm-guide in the browser.' Below the editor, there are two buttons: 'Download' and 'Take a tour'. At the very bottom, a purple footer bar contains the text 'JetBrains tools are FREE for education! Enjoy a professional developer experience using industry-leading tools to learn, teach and collaborate' and a 'Get started now' button.

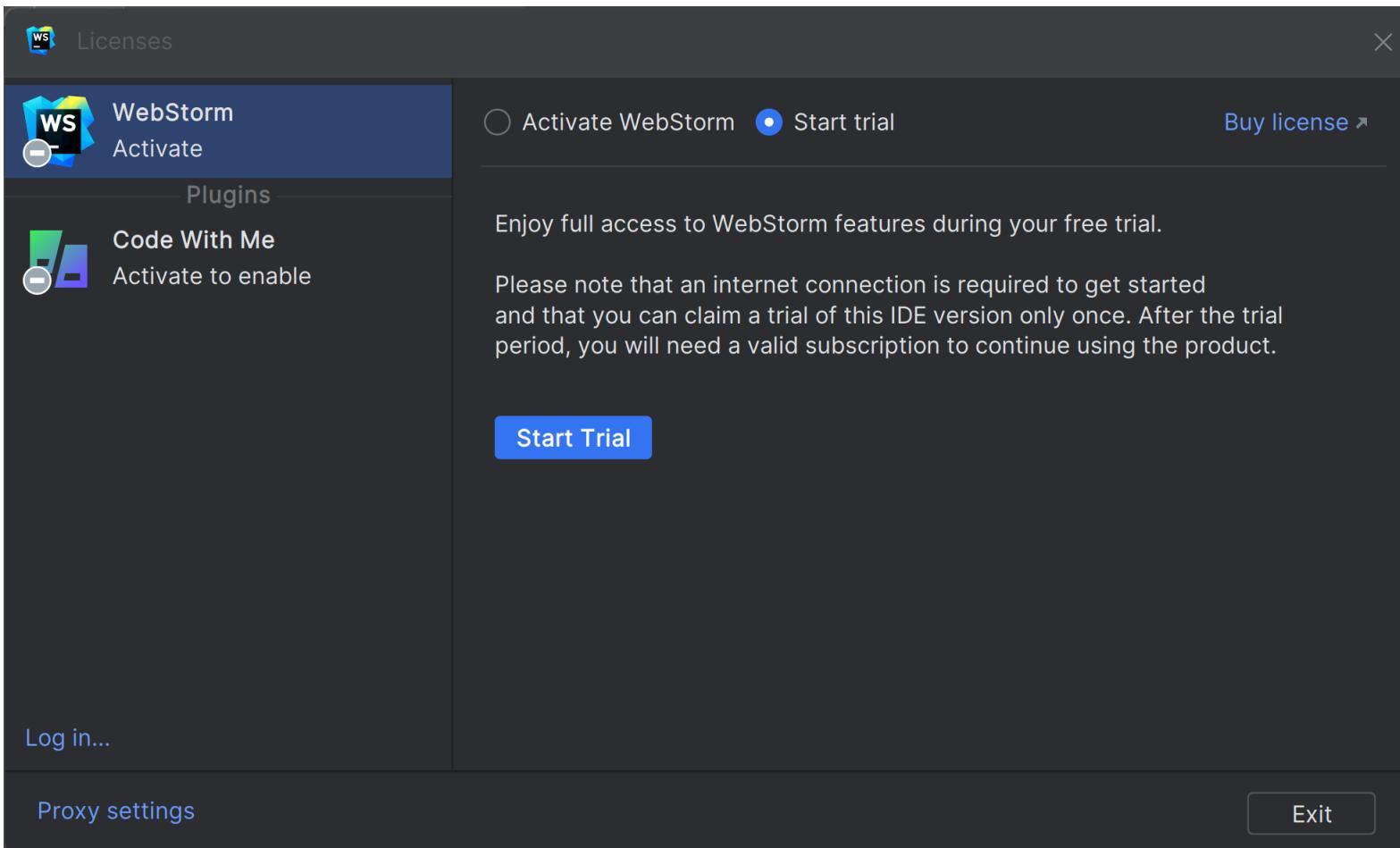
Download 30 days free evaluation + Install ...



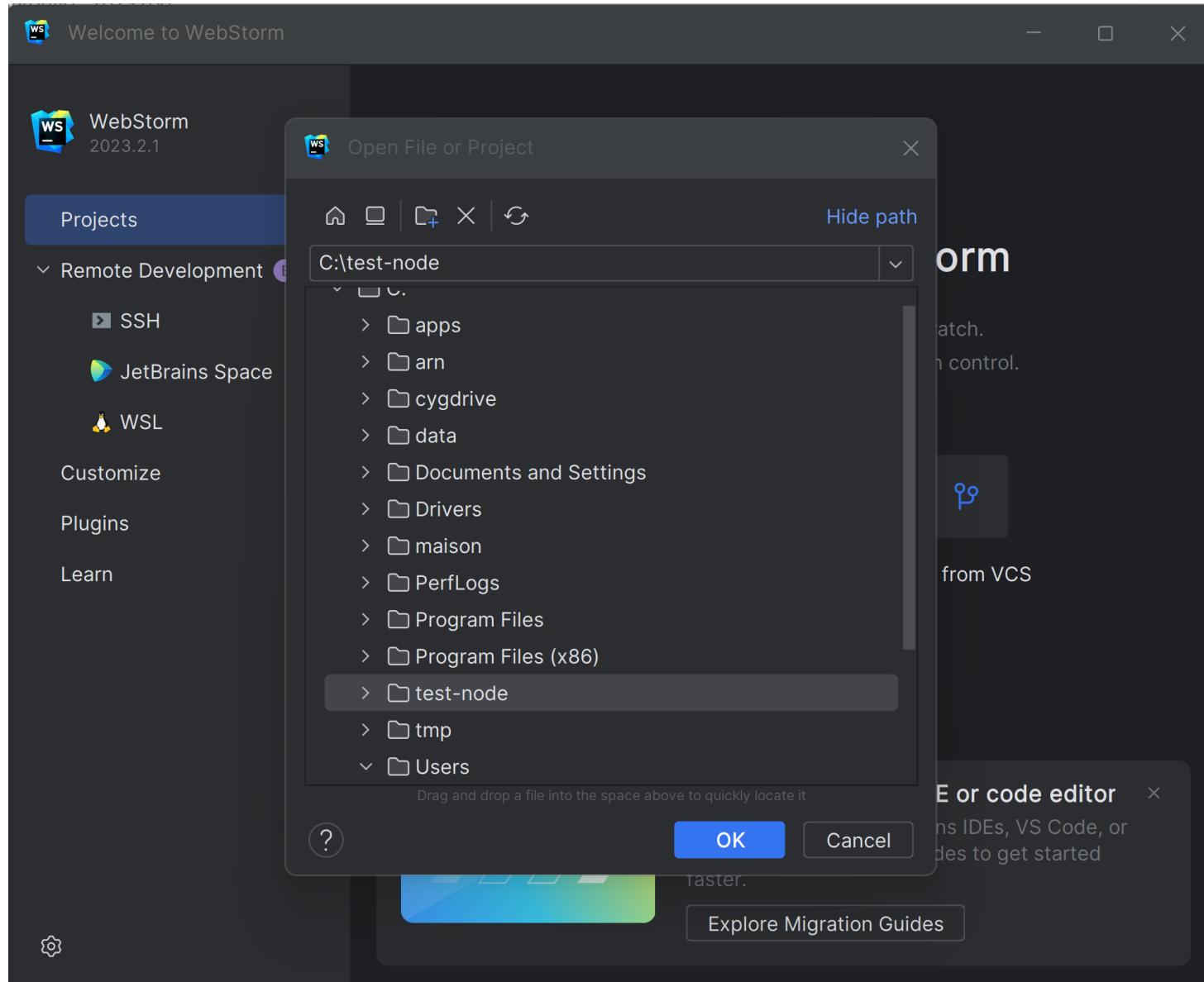
Notice Licenses:

IntelliJ Ultimate = WebStorm + IntelliJ +

IntelliJ Community = .. NO web !



Open Project



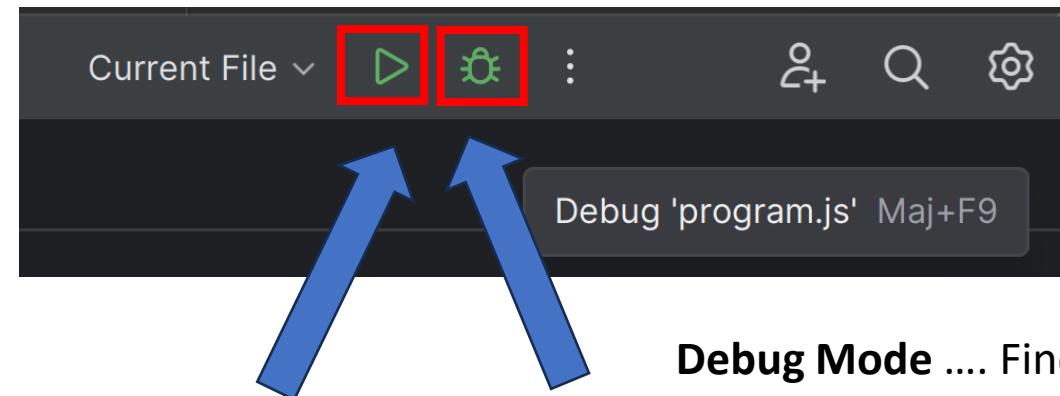
Edit program.js File

The screenshot shows the Visual Studio Code interface with a dark theme. The left sidebar displays a project structure for 'test-node' located at 'C:\test-node'. The 'node_modules' folder is highlighted. The main editor area shows the 'program.js' file with the following code:

```
1
2 console.log('Hello NodeJS World!')
3
```

The code is syntax-highlighted, with 'console' in purple and 'log' in blue. A yellow lightbulb icon is positioned next to the second line of code. The bottom status bar indicates the file is 'Current File', the path is 'test-node > program.js', and the encoding is 'UTF-8'.

Click on « Run » or « Debug » ?



Run Mode

Start and forget

Can NOT activate any breakpoint

... can only see logs, and Kill

Why do you use an IDE Debugger

Use directly « C:> node » ??

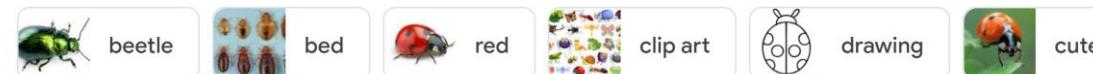
Debug Mode Find the « bug »

Can put breakpoints !

Google

bug

< All Images Videos News Books : More



W Wikipedia
Coccinellidae - Wikipedia



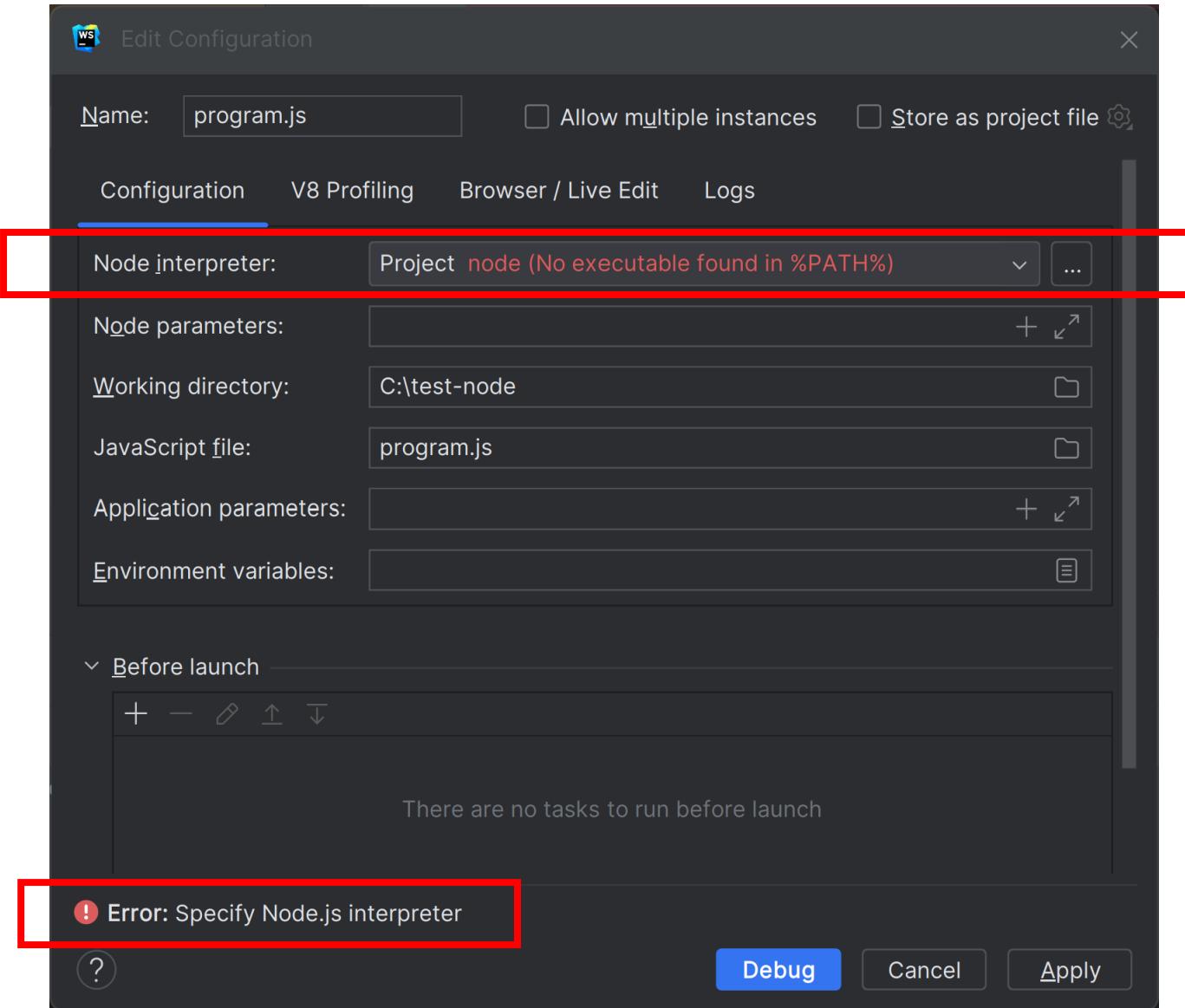
W Wikipedia
Bed bug - Wikipedia



W Wikipedia
Scutelleridae - Wikipedia

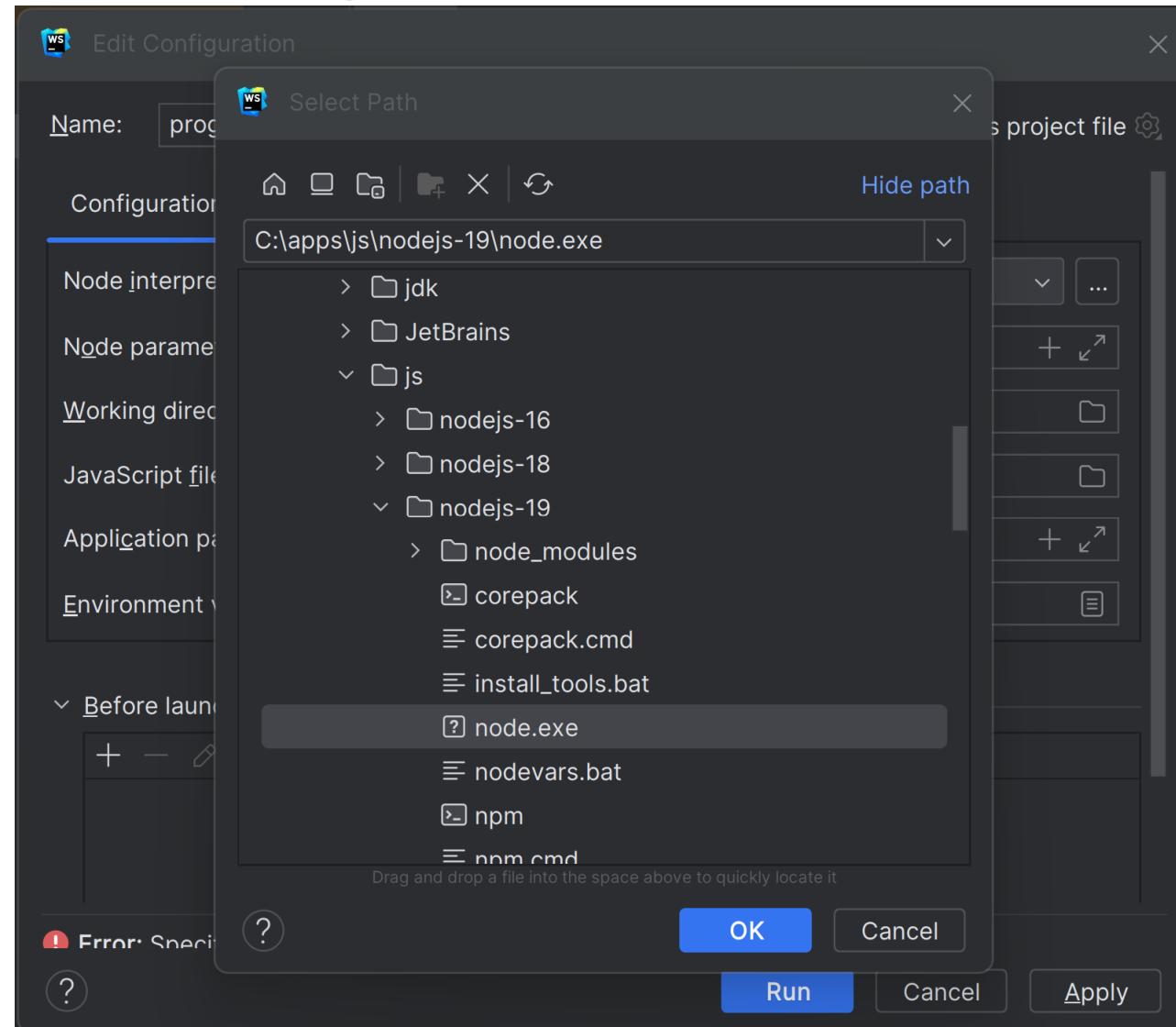
System Diagnostic

... Case undetected « node » not in %PATH%



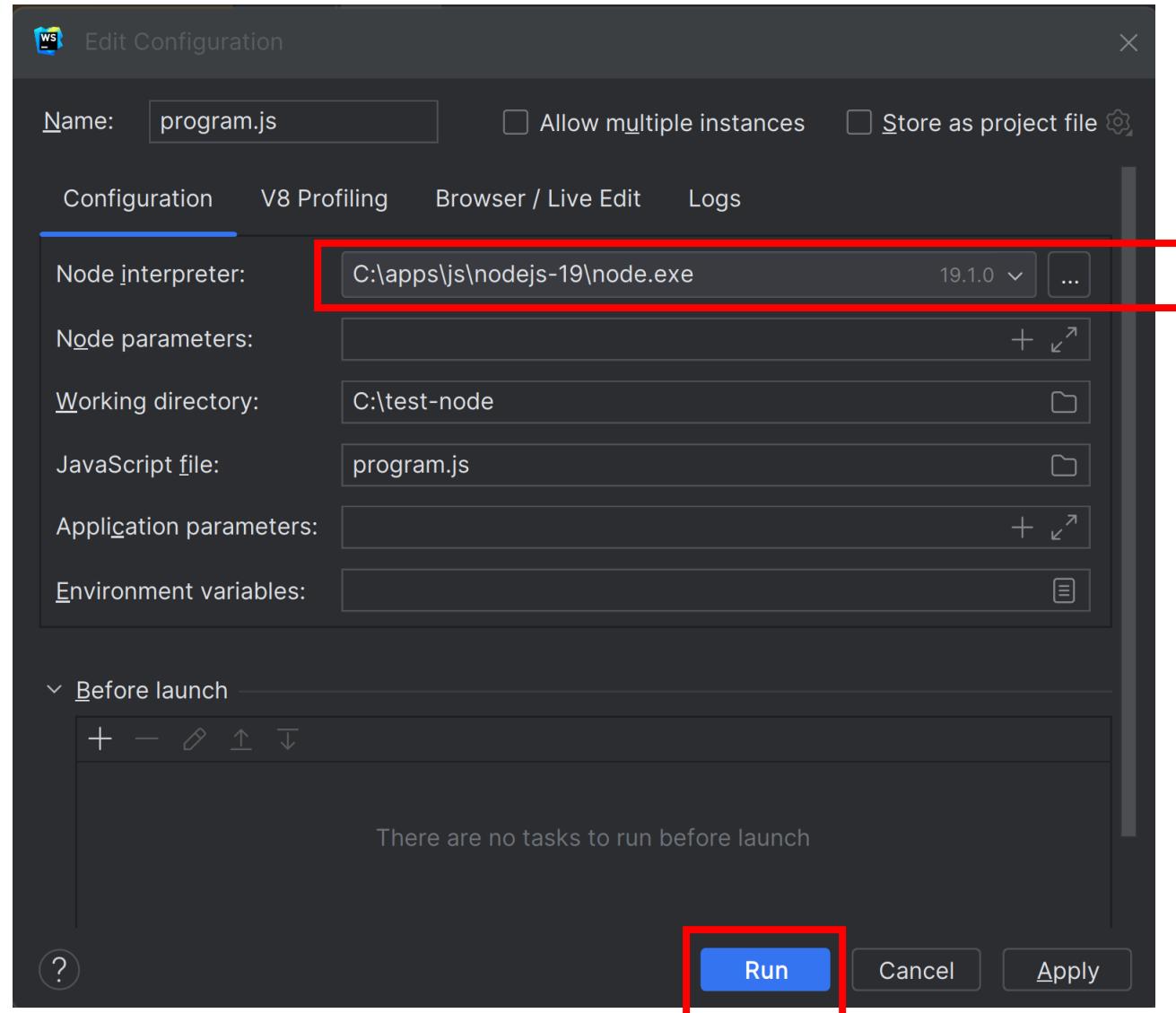
System Diagnostic

Configure ... Add node



System Diagnostic

Edit Run/Debug configuration...



« Run » ... Program Finished + console print

The screenshot shows the Visual Studio Code interface with a dark theme. The left sidebar displays a project structure for 'test-node' located at 'C:\test-node'. The 'node_modules' folder is highlighted as the 'library root'. Inside, there are files for 'package.json' and 'package-lock.json', and a script file named 'program.js'. The main editor area shows the contents of 'program.js':

```
1
2 console.log('Hello NodeJS World!')
3
```

Below the editor is the 'Run' view, which lists the currently selected task as 'program.js'. The terminal at the bottom shows the execution of the script and its output:

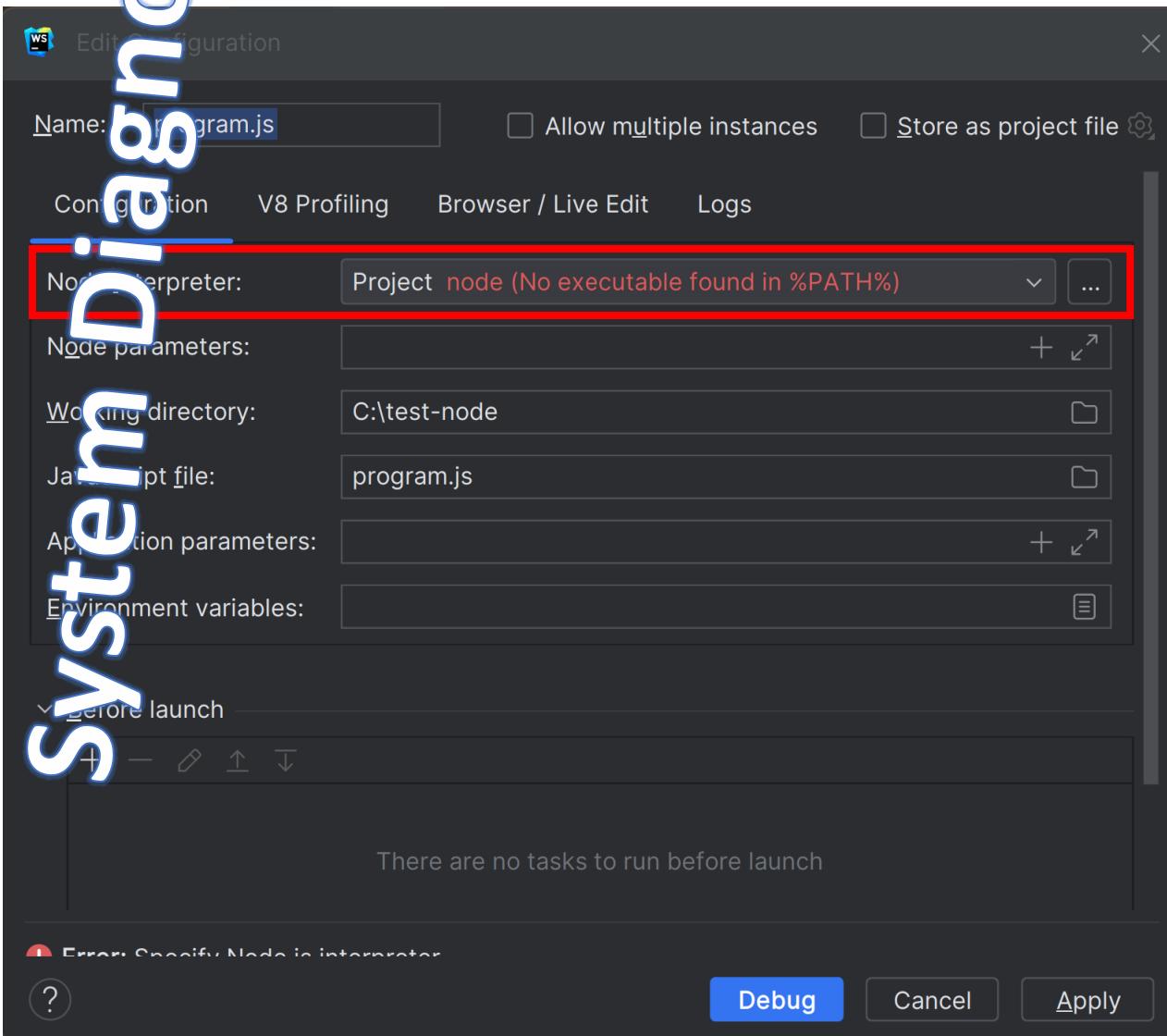
```
C:\apps\js\nodejs-19\node.exe C:\test-node\program.js
Hello NodeJS World!
Process finished with exit code 0
```

A red box highlights the terminal output, specifically the line 'Hello NodeJS World!', which corresponds to the console.log statement in the code.

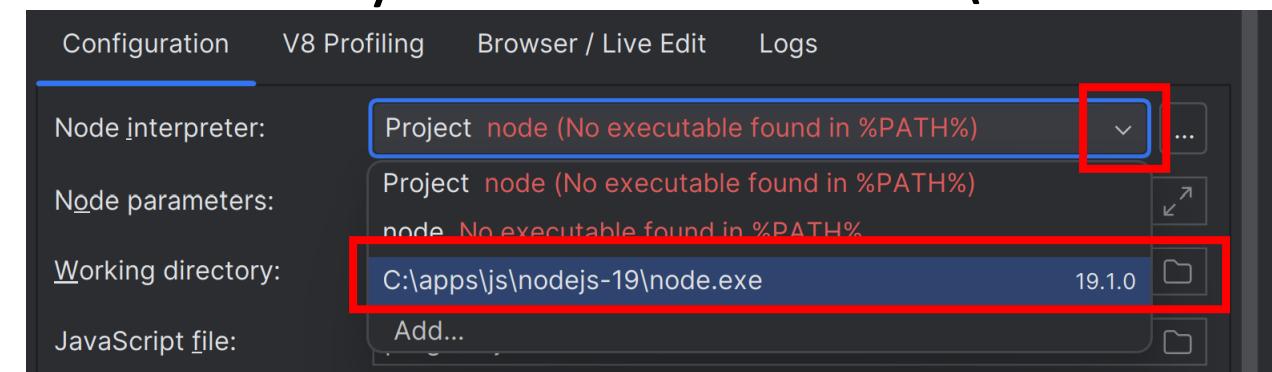
System Diagnostic

Redo « Run » or « Debug »

Need to re-edit node in %PATH% ???

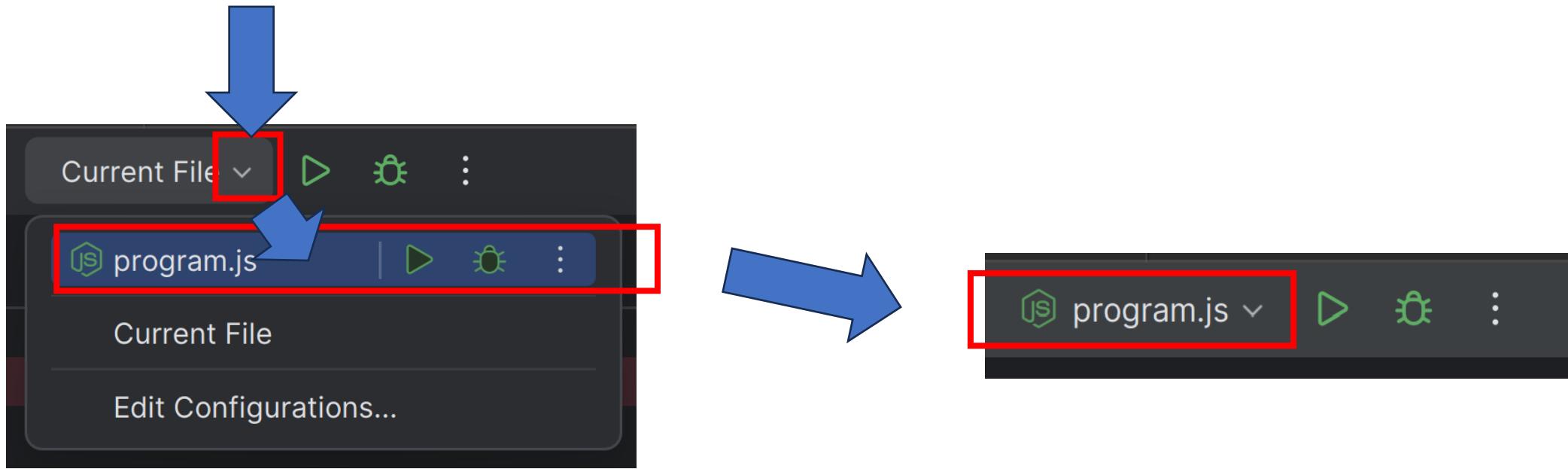


At least, can re-select
already added choice « C:\... » !

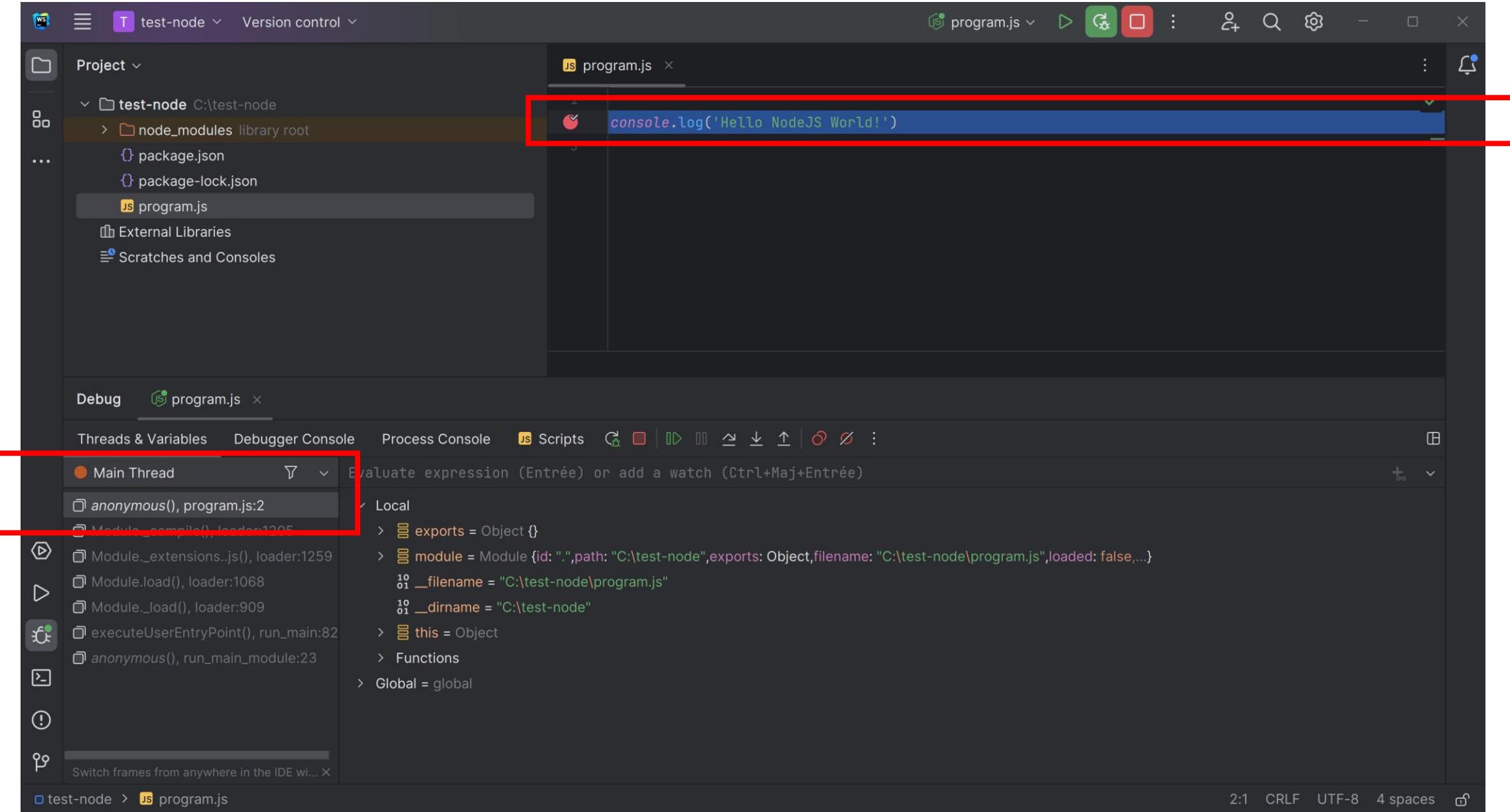


System Diagnostic

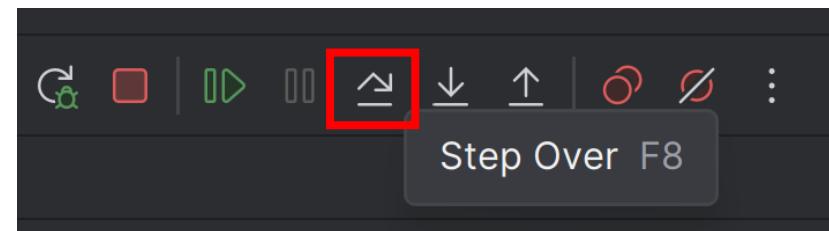
Change configuration « your-config »
instead of « Current File »



Debug « Mode » + Breakpoint => Program Paused

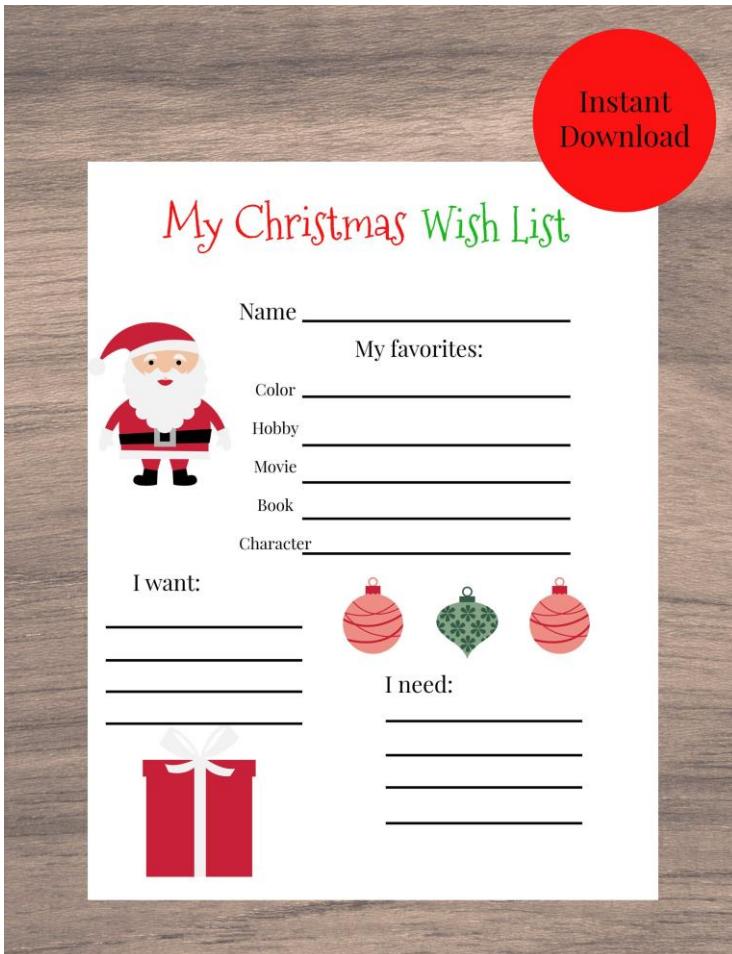


Step Into(F7)
< Step Over(F8)
< Step Out (Maj F8)
< Resume (F9)



Coffee Break

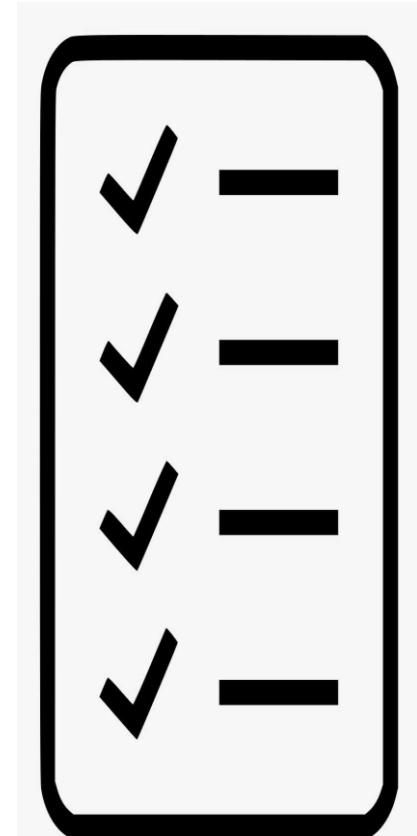
Complains on JavaScript ?



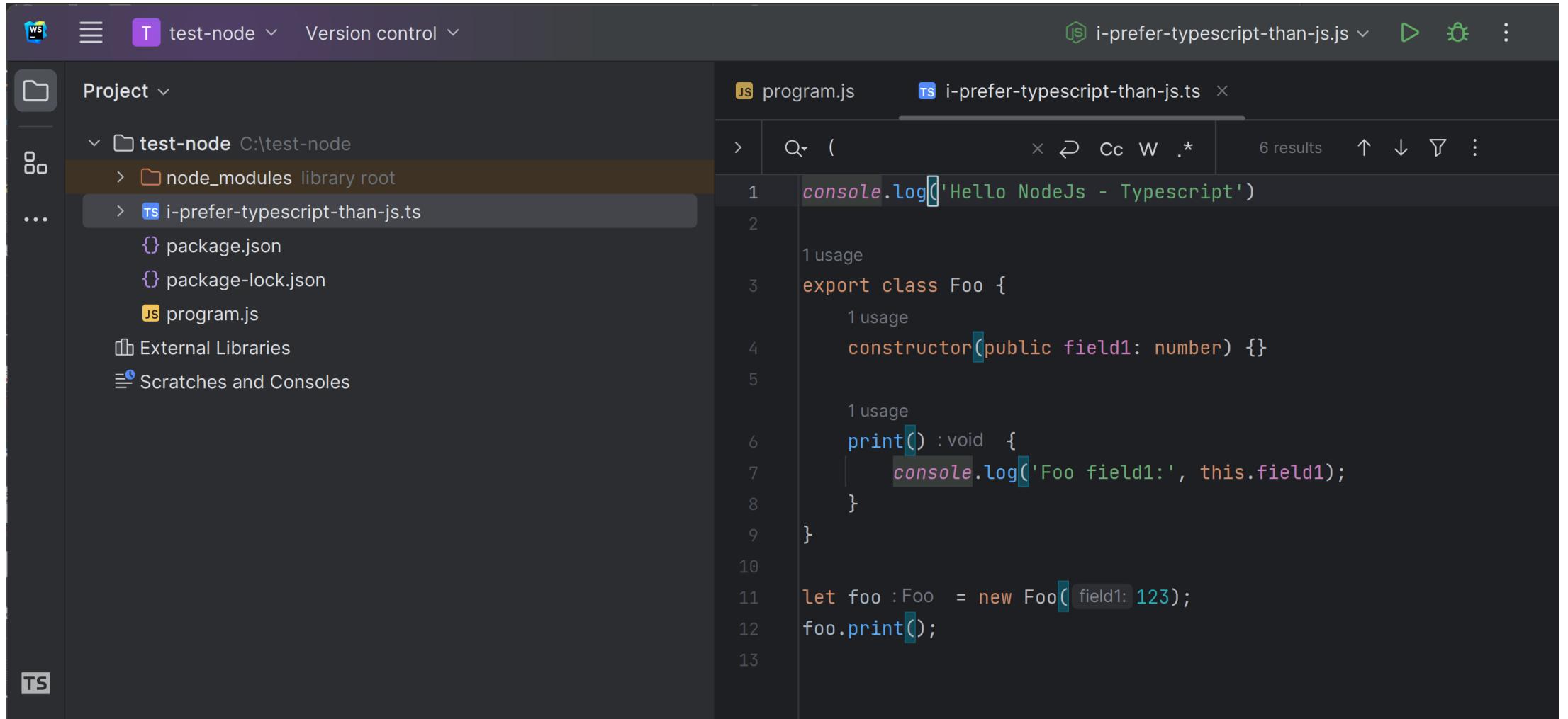
for Christmas Wish List, ask

- Types declarations
(standard types, structural types, union types)
- **Compiled-time Type checking**
- With Implicit types inference
- **Object-Oriented Classes**
- Templates
- Spread operators
- Syntactic sugars
- ..

TypeScript



Edit a TypeScript *.ts File

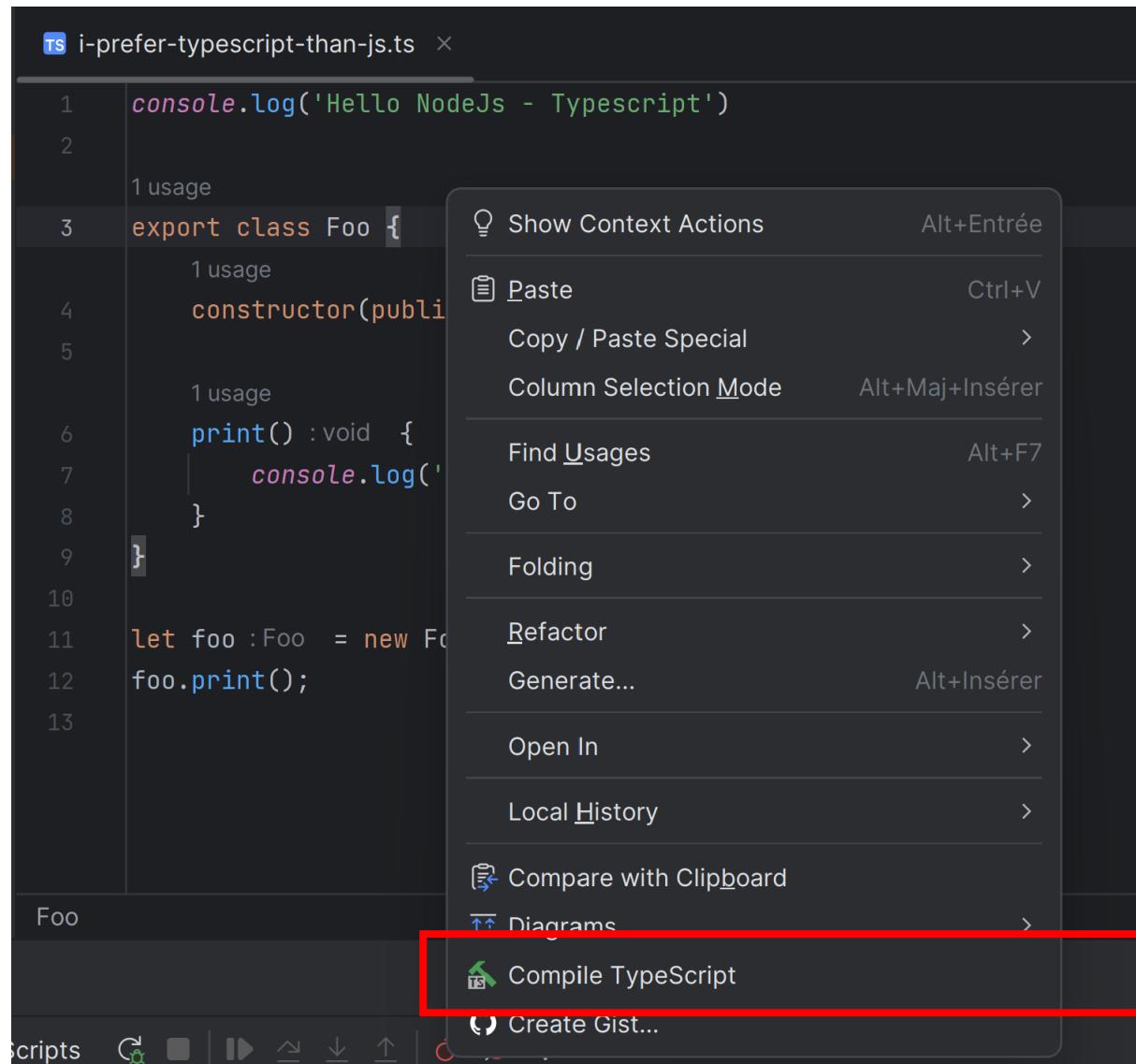


The screenshot shows the Visual Studio Code interface with a dark theme. The left sidebar displays a project structure for 'test-node' located at 'C:\test-node'. The 'node_modules' folder is highlighted. Other files listed include 'i-prefer-typescript-than-js.ts', 'package.json', 'package-lock.json', and 'program.js'. The main editor area shows two tabs: 'program.js' and 'i-prefer-typescript-than-js.ts'. The 'i-prefer-typescript-than-js.ts' tab is active and contains the following code:

```
1 console.log('Hello NodeJs - Typescript')
2
3 export class Foo {
4     constructor(public field1: number) {}
5
6     print(): void {
7         console.log('Foo field1:', this.field1);
8     }
9 }
10
11 let foo : Foo = new Foo( field1: 123);
12 foo.print();
```

The code uses TypeScript syntax like classes, methods, and imports. The 'print()' method is annotated with a type of 'void'. The 'field1' parameter in the constructor is annotated with a type of 'number'. The variable 'foo' is annotated with a type of 'Foo'. The 'console.log()' statements output the string 'Hello NodeJs - Typescript' and 'Foo field1:' followed by the value '123'.

Right Click « Compile TypeScript »



Open Item « .ts » > Sub Item « .js » see generated Js !!

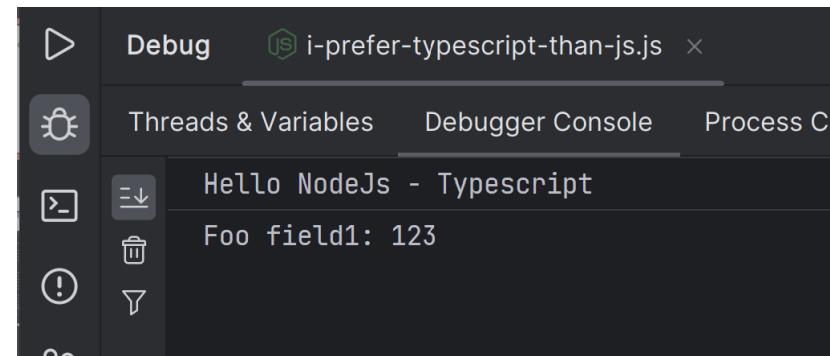
The screenshot shows the Visual Studio Code interface. On the left, the Project Explorer sidebar displays a folder structure for a project named 'test-node' located at 'C:\test-node'. Inside this folder are 'node_modules', 'i-prefer-typescript-than-js.ts', and 'i-prefer-typescript-than-js.js'. The 'i-prefer-typescript-than-js.ts' file is currently open in the main editor area. The 'i-prefer-typescript-than-js.js' file is also visible in the tabs bar above the editor. The code in the editor is as follows:

```
1 "use strict";
2 Object.defineProperty(exports, "__esModule", { value: true });
3 exports.Foo = void 0;
4 console.log('Hello NodeJs - Typescript');
5 var Foo = /** @class */ (function () {
6     2 usages
7     function Foo(field1) {
8         this.field1 = field1;
9     }
10    Foo.prototype.print = function () {
11        console.log('Foo field1:', this.field1);
12    };
13    return Foo;
14}());
15 exports.Foo = Foo;
16 var foo = new Foo(123);
17 foo.print();
```

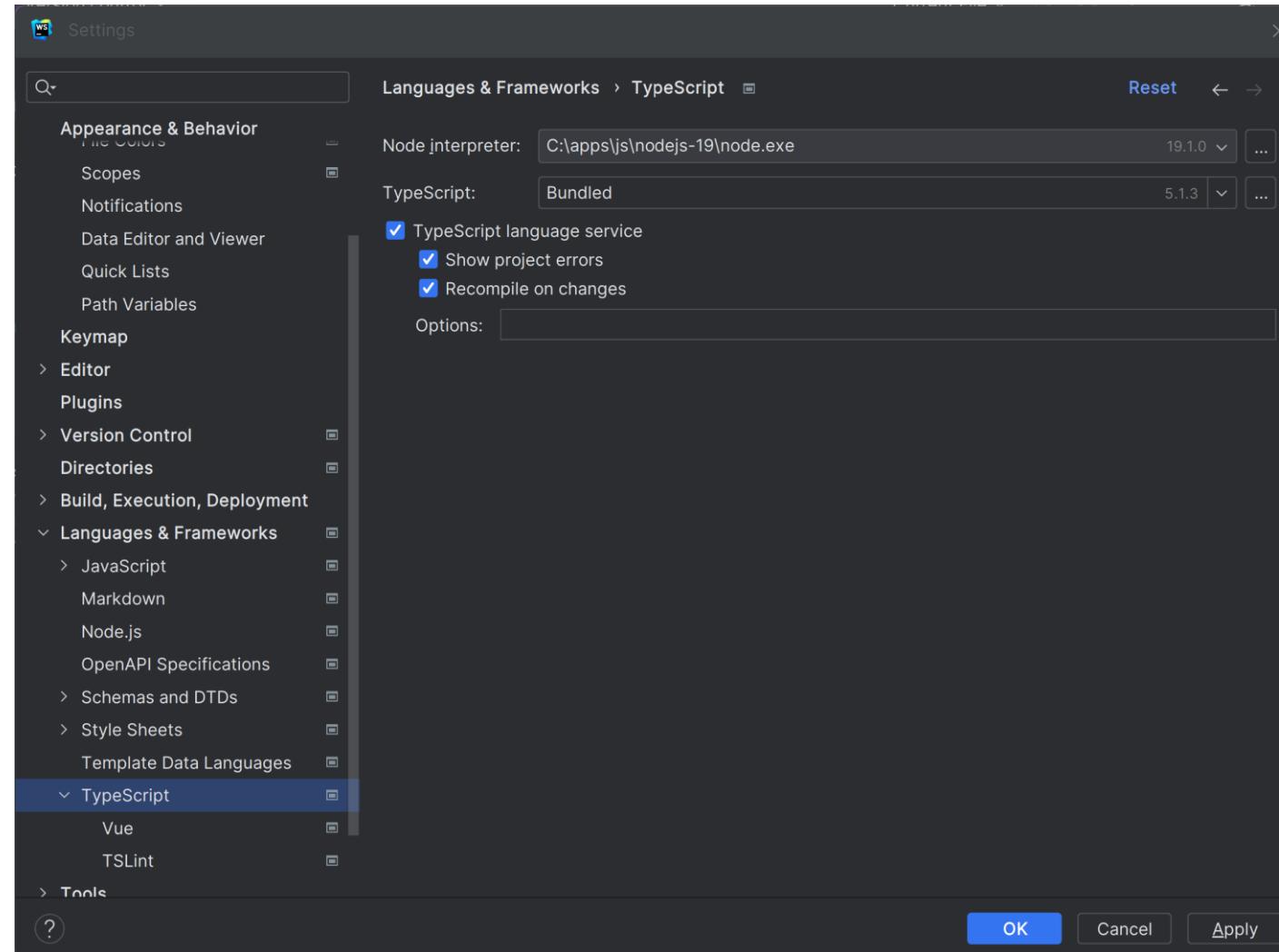
A red box highlights the entire code block in the editor, indicating the generated JavaScript code. Another red box highlights the 'i-prefer-typescript-than-js.ts' tab in the tabs bar, emphasizing the source TypeScript file.

Debug Ts program ... (No Run/Debug/breakpoint in *.ts, but ok in *.js)

need « .map » file for mapping ts line X <-> js line Y



Edit Settings > TypeScript > Recompile on changes



Edit on Left *.ts ... see real time change in right *.js

The image shows a split-screen code editor with two panes. The left pane is titled "i-prefer-typescript-than-js.ts" and contains TypeScript code. The right pane is titled "i-prefer-typescript-than-js.js" and contains the corresponding generated JavaScript code. A vertical red bar indicates the current cursor position.

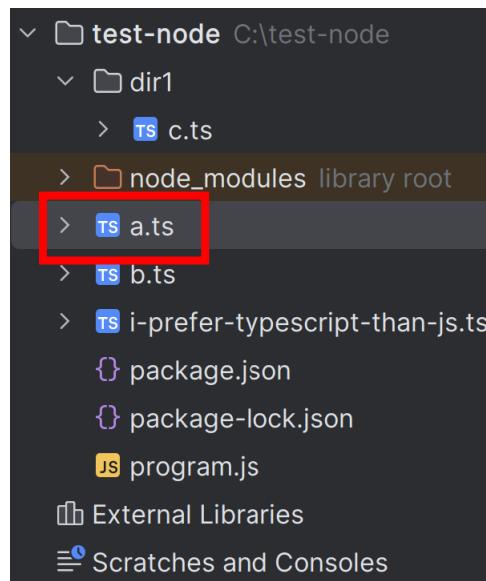
Left Pane (TypeScript):

```
ts i-prefer-typescript-than-js.ts
1 console.log('Hello NodeJs - Typescript') 1 1 1
2
3 export class Foo {
4     field2: number = 1;
5
6     constructor(public field1: number) {}
7
8     print(): void {
9         console.log('Foo field1:', this.field1);
10        console.log('Foo field2:', this.field2);
11        console.log('Foo fieldxxxx:', this.fieldxxxx);
12    }
13}
14
15let foo : Foo = new Foo({ field1: 123 });
16foo.print();
17
```

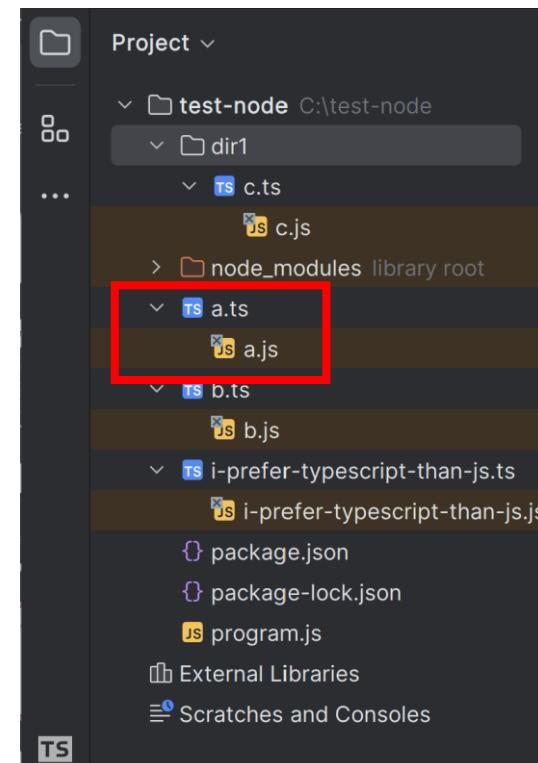
Right Pane (JavaScript):

```
js i-prefer-typescript-than-js.js
1 "use strict";
2
3 Object.defineProperty(exports, "__esModule", { value: true });
4 exports.Foo = void 0;
5
6 console.log('Hello NodeJs - Typescript');
7
8 var Foo = /** @class */ (function () {
9     2 usages
10     function Foo(field1) {
11         this.field1 = field1;
12         this.field2 = 1;
13     }
14
15     Foo.prototype.print = function () {
16         console.log('Foo field1:', this.field1);
17         console.log('Foo field2:', this.field2);
18         console.log('Foo fieldxxxx:', this.fieldxxxx);
19     };
20
21     return Foo;
22 }());
23
24 exports.Foo = Foo;
25
26 var foo = new Foo(123);
27 foo.print();
```

Edit many **/*.ts ... see corresponding **/*.js



Expand
views
→



Check
on FileSystem
→

Nom	Modifié le	Type	Taille
.idea	31/08/2023 23:29	Dossier de fichiers	
dir1	31/08/2023 23:30	Dossier de fichiers	
node_modules	31/08/2023 20:34	Dossier de fichiers	
a.js	31/08/2023 23:30	Fichier de JavaScript	0 Ko
a.ts	31/08/2023 23:30	Fichier TS	0 Ko
b.js	31/08/2023 23:30	Fichier de JavaScript	0 Ko
b.ts	31/08/2023 23:30	Fichier TS	0 Ko
i-prefer-typescript-than-js.js	31/08/2023 23:29	Fichier de JavaScript	1 Ko
i-prefer-typescript-than-js.ts	31/08/2023 23:28	Fichier TS	1 Ko
package.json	31/08/2023 20:33	Fichier JSON	1 Ko
package-lock.json	31/08/2023 20:51	Fichier JSON	22 Ko
program.js	31/08/2023 20:12	Fichier de JavaScript	1 Ko

tsc = TypeScript Compiler

... same as in IDE, but in batch mode, for CI-CD

tsc is internally used by WebStorm

... / IntelliJ / VisualStudio Code / Eclipse !!

tsc even generalised it as « language server »

=> You have exact same compilation in all IDEs !!

<https://www.typescriptlang.org/docs/handbook/typescript-tooling-in-5-minutes.html>

The screenshot shows the TypeScript handbook page. The header includes the TypeScript logo and navigation links for Download, Docs, Handbook, Community, Playground, and Tools. A sidebar on the left under 'Get Started' contains links for 'TS for the New Programmer', 'TypeScript for JS Programmers', 'TS for Java/C# Programmers', 'TS for Functional Programmers', 'TypeScript Tooling in 5 minutes' (which is highlighted), 'Handbook', 'Reference', 'Tutorials', 'What's New', 'Declaration Files', 'JavaScript', and 'Project Configuration'. The main content area features a large heading 'TypeScript Tooling in 5 minutes'. Below it, a paragraph reads 'Let's get started by building a simple web application with TypeScript.' A section titled 'Installing TypeScript' follows, with a note about two installation methods: via npm or Visual Studio plugins. It also mentions that Visual Studio 2017 and 2015 Update 3 support TypeScript by default. A command-line instruction '`> npm install -g typescript`' is shown at the bottom.

Let's get started by building a simple web application with TypeScript.

Installing TypeScript

There are two main ways to add TypeScript to your project:

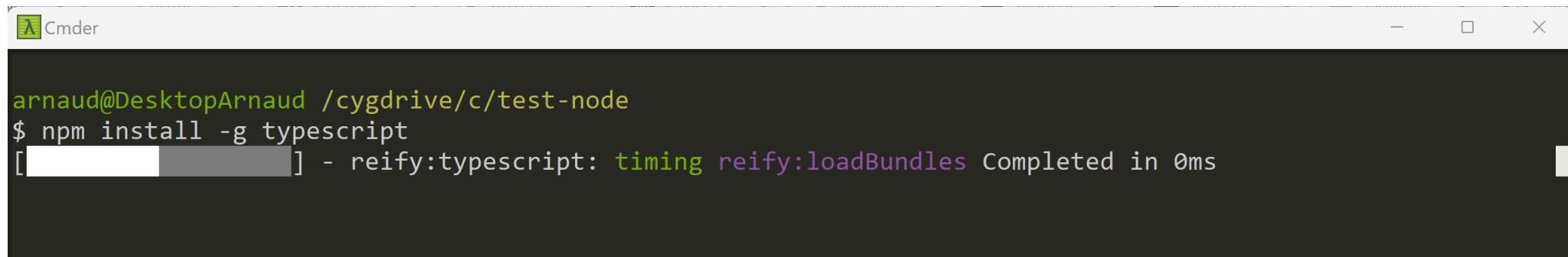
- Via npm (the Node.js package manager)
- By installing TypeScript's Visual Studio plugins

Visual Studio 2017 and Visual Studio 2015 Update 3 include TypeScript language support by default but does not include the TypeScript compiler, `tsc`. If you didn't install TypeScript with Visual Studio, you can still [download it](#).

For npm users:

```
> npm install -g typescript
```

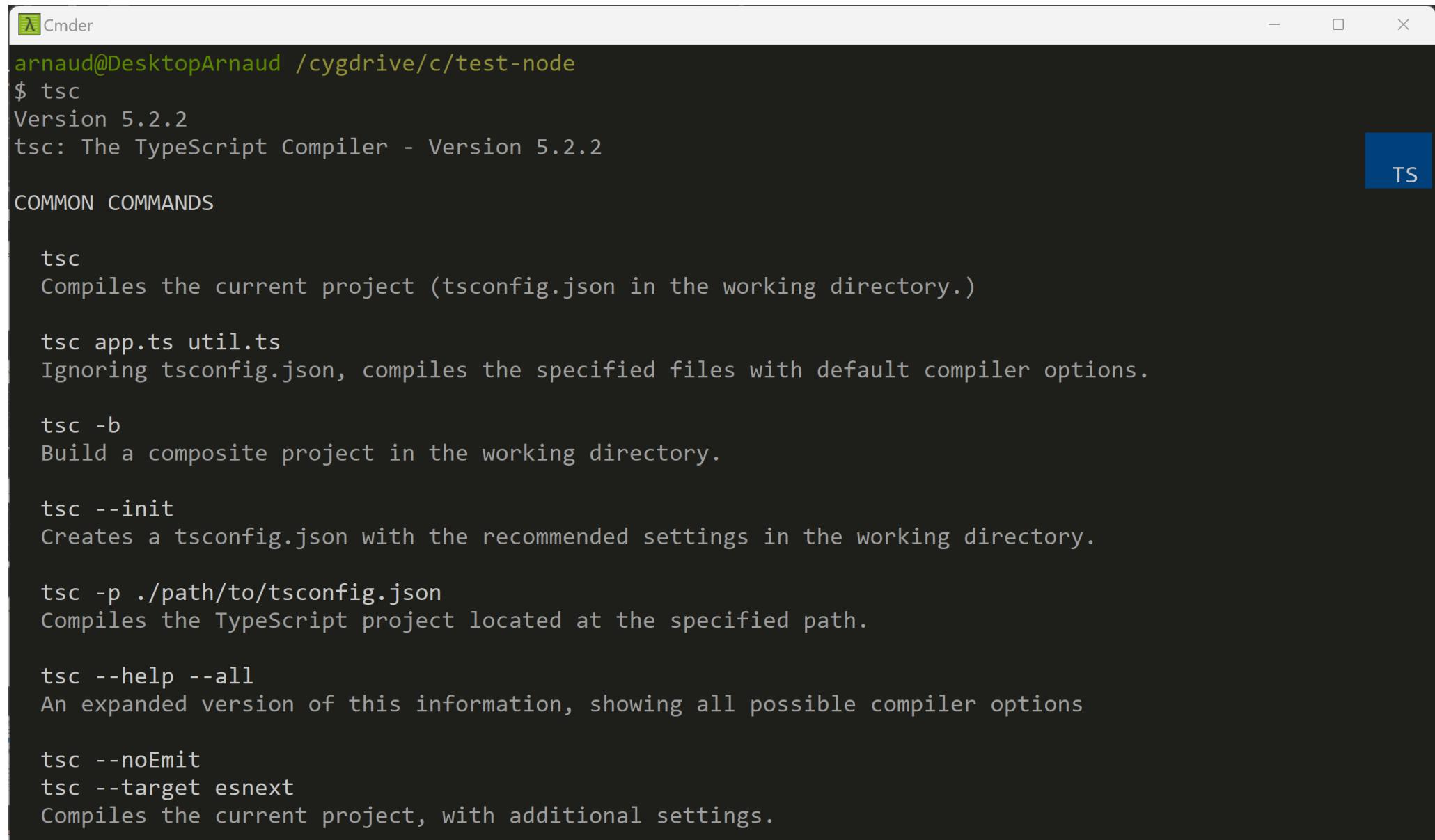
\$ npm install -g typescript
(« -g » for « --global » ... add tsc in node %PATH%)



The screenshot shows a Cmder terminal window with a dark theme. The title bar says "Cmder". The command line shows the user's path as "arnaud@DesktopArnaud /cygdrive/c/test-node" followed by the command "\$ npm install -g typescript". Below the command, a progress bar indicates the process is completed in 0ms. The progress bar is mostly grey with a small green section at the beginning.

```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ npm install -g typescript
[██████████] - reify:typescript: timing reify:loadBundles Completed in 0ms
```

\$ tsc



A screenshot of a terminal window titled "Cmder". The window shows the command \$ tsc followed by its usage information. The usage information includes examples for common commands like tsc, tsc -b, and tsc --init, as well as more specific options like --noEmit and --target esnext. A blue "TS" button is visible in the bottom right corner of the terminal window.

```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ tsc
Version 5.2.2
tsc: The TypeScript Compiler - Version 5.2.2

COMMON COMMANDS

tsc
Compiles the current project (tsconfig.json in the working directory.)

tsc app.ts util.ts
Ignoring tsconfig.json, compiles the specified files with default compiler options.

tsc -b
Build a composite project in the working directory.

tsc --init
Creates a tsconfig.json with the recommended settings in the working directory.

tsc -p ./path/to/tsconfig.json
Compiles the TypeScript project located at the specified path.

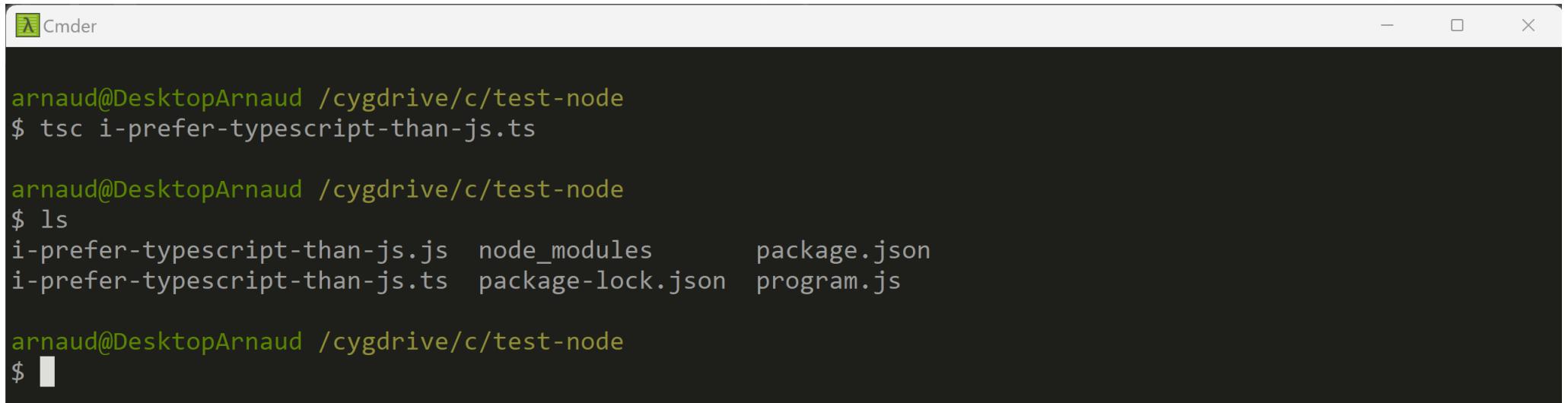
tsc --help --all
An expanded version of this information, showing all possible compiler options

tsc --noEmit
tsc --target esnext
Compiles the current project, with additional settings.
```

3 uses cases ...

- 1/ compile 1 single .ts file
- 2/ compile **/*.ts in project
- 3/ « -w » real-time watch to recompile on change

\$ tsc single-file.ts



The screenshot shows a terminal window titled "Cmder". The command \$ tsc single-file.ts is run, followed by an ls command to show the resulting files. The terminal window has a dark background with light-colored text.

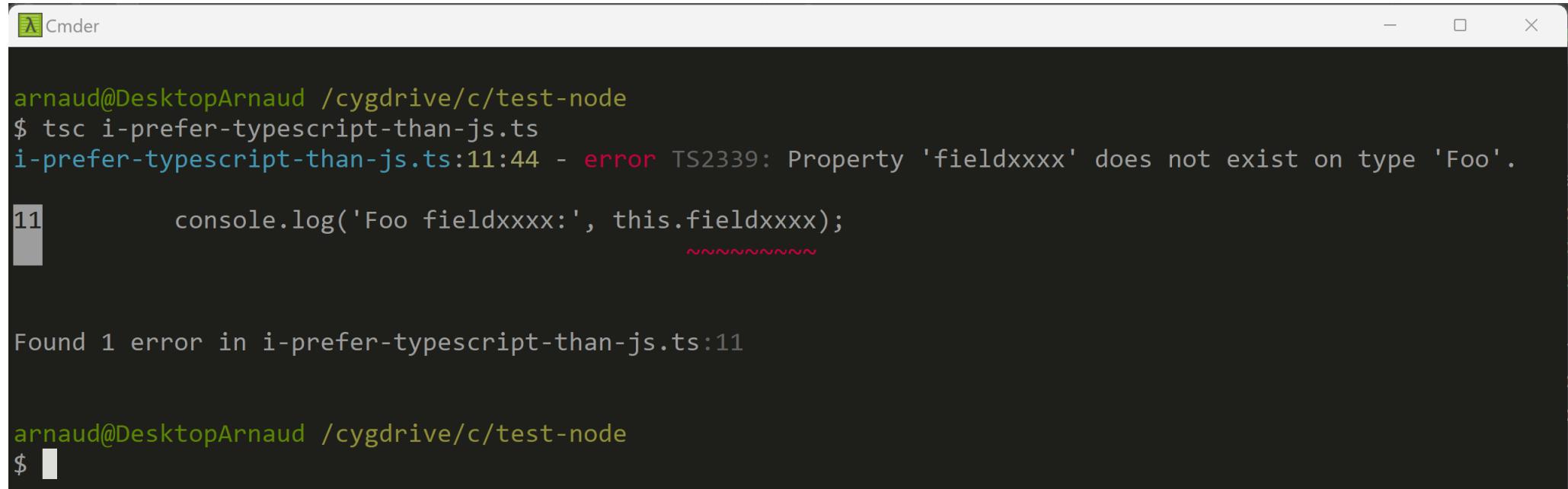
```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ tsc i-prefer-typescript-than-js.ts

arnaud@DesktopArnaud /cygdrive/c/test-node
$ ls
i-prefer-typescript-than-js.js  node_modules      package.json
i-prefer-typescript-than-js.ts   package-lock.json  program.js

arnaud@DesktopArnaud /cygdrive/c/test-node
$ █
```

Tsc Compiles ... find ERRORS messages at compile-time

Compare... remember typo found at runtime ?
<https://my-broken-web-site-deployed-in-PROD.html>

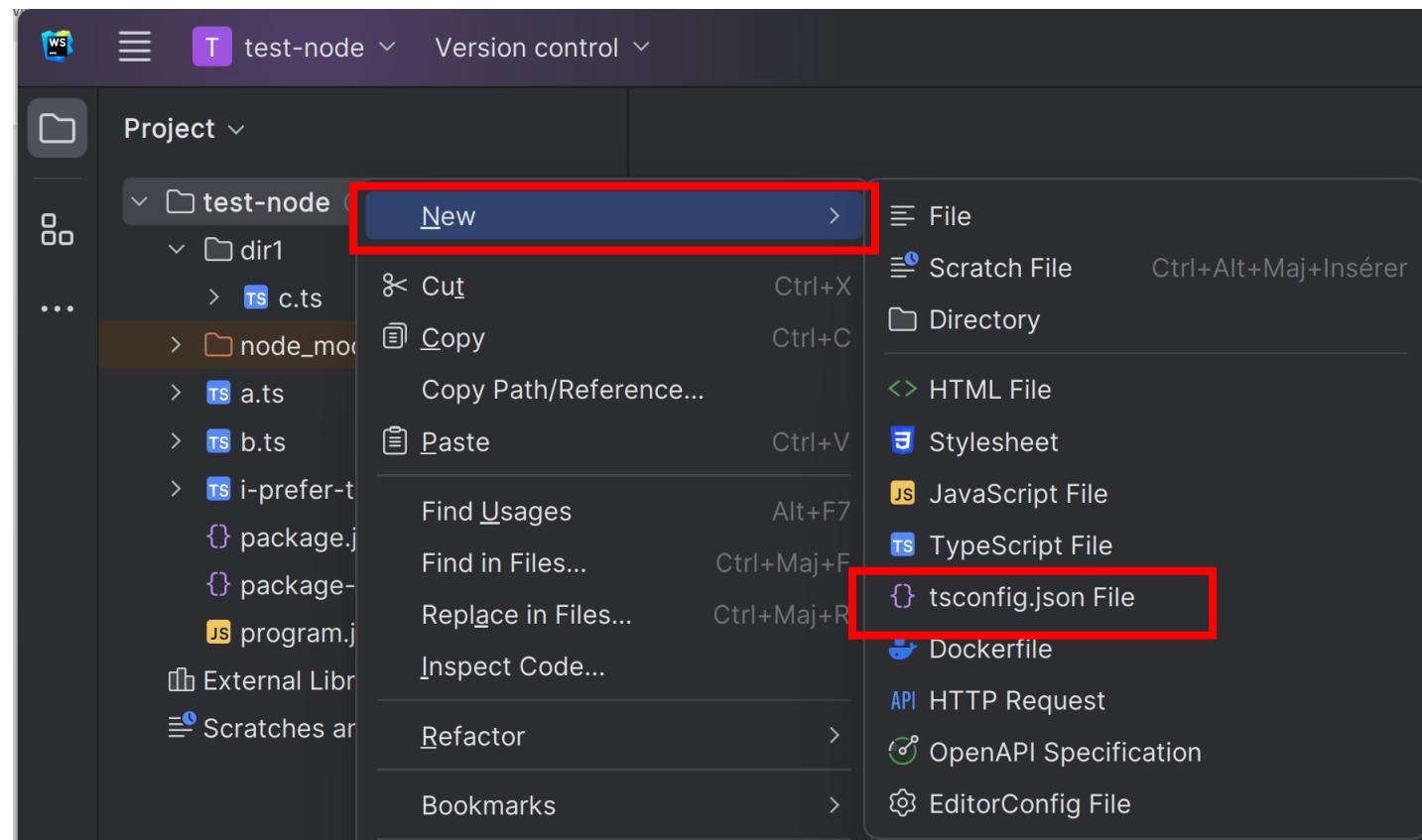


The screenshot shows a terminal window titled 'Cmder' with a dark theme. The command line shows the user is in their home directory and runs 'tsc'. The output indicates a compilation error:

```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ tsc i-prefer-typescript-than-js.ts
i-prefer-typescript-than-js.ts:11:44 - error TS2339: Property 'fieldxxxx' does not exist on type 'Foo'.
11     console.log('Foo fieldxxxx:', this.fieldxxxx);
                         ~~~~~~
Found 1 error in i-prefer-typescript-than-js.ts:11

arnaud@DesktopArnaud /cygdrive/c/test-node
$
```

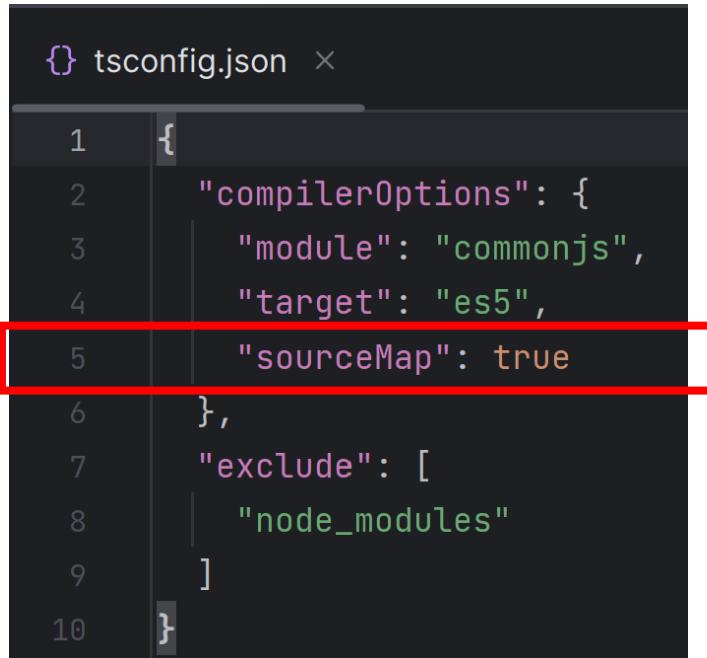
Create tsc configuration file: « tsconfig.json »



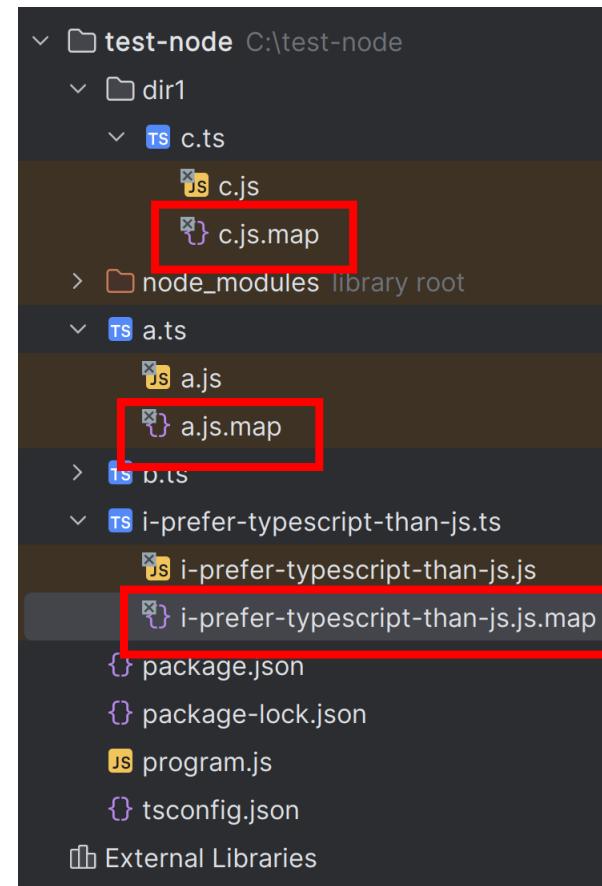
Default tsconfig.json

```
{} tsconfig.json ×  
1 {  
2   "compilerOptions": {  
3     "module": "commonjs",  
4     "target": "es5",  
5     "sourceMap": true  
6   },  
7   "exclude": [  
8     "node_modules"  
9   ]  
10 }
```

Notice sourceMap:true in tsconfig.json ... and **/*.js.map files generated



```
{} tsconfig.json ×  
1  {  
2      "compilerOptions": {  
3          "module": "commonjs",  
4          "target": "es5",  
5          "sourceMap": true  
6      },  
7      "exclude": [  
8          "node_modules"  
9      ]  
10 }
```

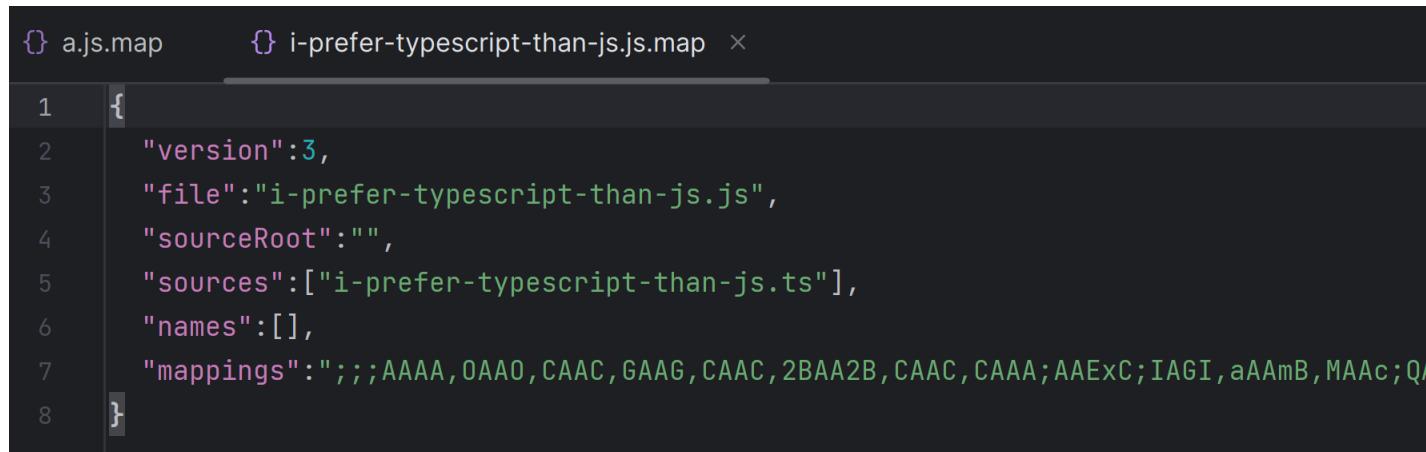


.map file

... = « lineNumber » debugging info
between source « *.ts » and generated « *.js »

Analogy with java:

between source « *.java » and generated « *.class »



The screenshot shows a code editor window with two tabs at the top: 'a.js.map' and 'i-prefer-typescript-than-js.js.map'. The 'i-prefer-typescript-than-js.js.map' tab is active, displaying the following JSON code:

```
1 {  
2   "version":3,  
3   "file":"i-prefer-typescript-than-js.js",  
4   "sourceRoot": "",  
5   "sources":["i-prefer-typescript-than-js.ts"],  
6   "names":[],  
7   "mappings":";;;AAAA,OAAO,CAAC,GAAG,CAAC,2BAA2B,CAAC,CAA;AAExC;IAGI,aAAmB,MAAc;Q/  
8 }
```

Remark and Analogy

JavaScript is the « low-level » assembler language (~x64) on the Web
Supported by all « web browsers » AND « node » (~intel/amd processors)



No developer code anymore in assembly language directly
(they prefer high-level C, C++, Java, C#, Python, ...)



Web developer prefers TypeScript rather than Js directly ..

Remark of The Remark ... WebAssembly

JavaScript is such an **horrible monster langage**
(with **eval** and strange permissive things)

W3c standard « WebAssembly » use only a SUBSET of JS
... that can be really statically typed / analysed / understood
then transformed to native x64 code by « compiler »

« WebAssembly » is trully a fast assembly langage
almost as fast as optimised « normal » code

.map Goodies ... can put breakpoint on *.ts (still debugging *.js for launch)

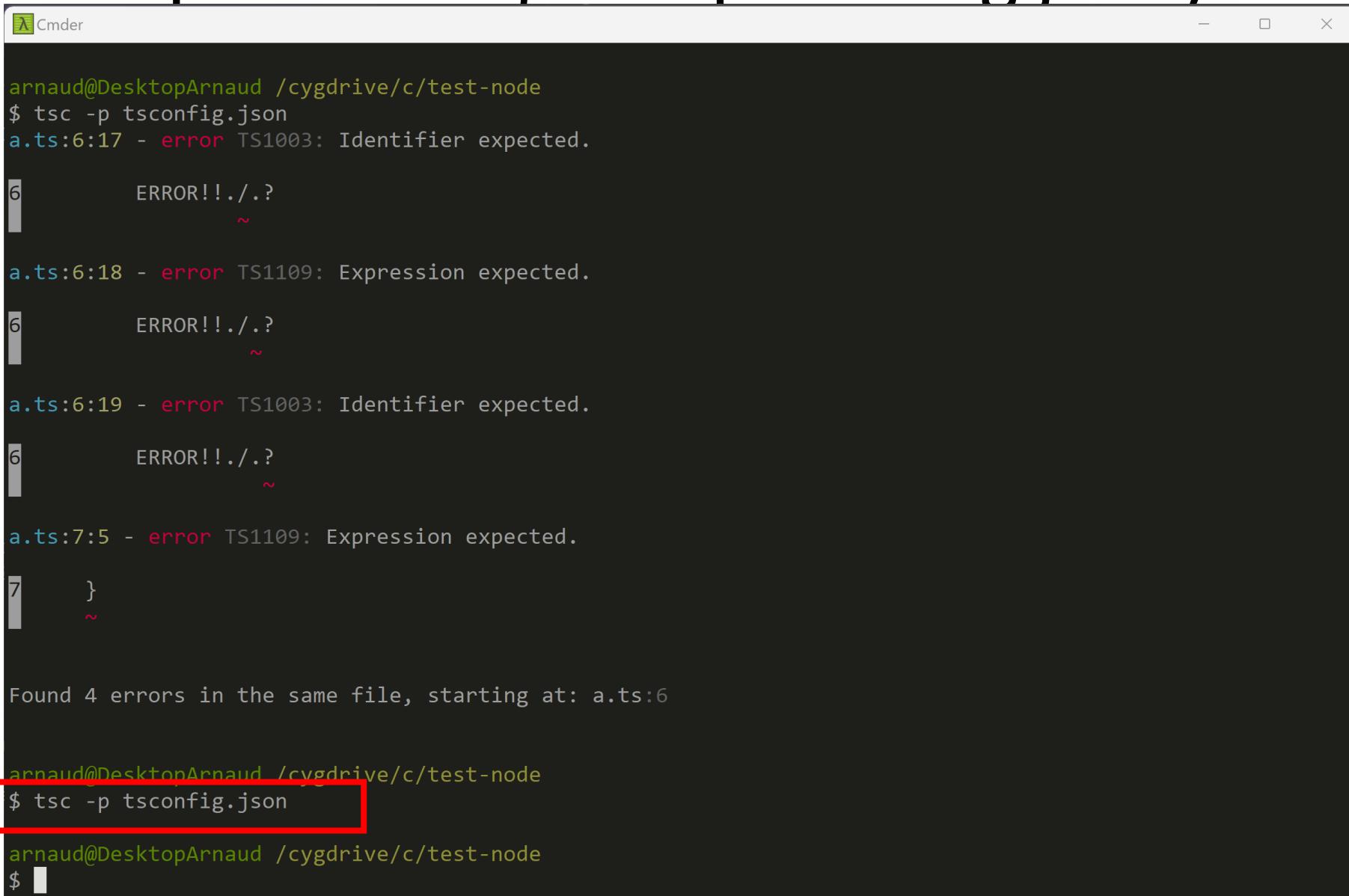
The screenshot shows a Visual Studio Code (VS Code) interface with the following details:

- Project Explorer:** Shows a folder structure for "test-node". Inside "test-node", there are "dir1" and "c.ts". The "node_modules" folder is listed as the "library root". Other files include "a.ts", "a.js", "a.js.map", "b.ts", "i-prefer-typescript-than-js.ts", "i-prefer-typescript-than-js.js", "package.json", "package-lock.json", "program.js", and "tsconfig.json".
- Editor:** The "i-prefer-typescript-than-js.ts" file is open. A red circular breakpoint icon is placed on the line containing the "print()" method definition.
- Terminal:** The terminal shows the command "Foo > print()".
- Debug View:** The "Debug" view is active, showing a list of threads. The "Main Thread" is selected, and the expression "this.field1 = 123" is highlighted in the "Local" scope.
- Status Bar:** The status bar at the bottom right indicates "TypeScript 5.2.2" and other system information.

```
1 console.log('Hello NodeJS - Typescript')
2
3 export class Foo {
4     field2: number = 1;    field2: 1
5
6     constructor(public field1: number) {}  this.field1: 123
7
8     print(): void {
9         console.log('Foo field1:', this.field1);  this.field1: 123
10        console.log('Foo field2:', this.field2);  field2: 1
11    }
12
13
14 let foo : Foo  = new Foo( field1: 123);
15 foo.print();
```

\$ tsc
(or equivalent : \$ tsc -p tsconfig.json)

Case when error...



arnaud@DesktopArnaud /cygdrive/c/test-node
\$ tsc -p tsconfig.json
a.ts:6:17 - error TS1003: Identifier expected.
6 ERROR! !./.?
 ~

a.ts:6:18 - error TS1109: Expression expected.
6 ERROR! !./.?
 ~

a.ts:6:19 - error TS1003: Identifier expected.
6 ERROR! !./.?
 ~

a.ts:7:5 - error TS1109: Expression expected.
7 }
 ~

Found 4 errors in the same file, starting at: a.ts:6

arnaud@DesktopArnaud /cygdrive/c/test-node
\$ tsc -p tsconfig.json

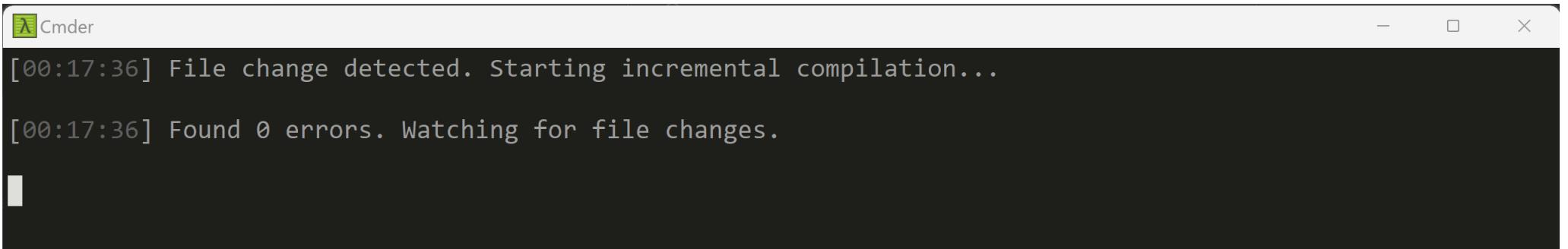
arnaud@DesktopArnaud /cygdrive/c/test-node
\$

Case when OK...

NO log !!

\$ tsc -W

Case when OK...
NO log



The screenshot shows a Cmder terminal window with the title bar "Cmder". The window contains the following text:

```
[00:17:36] File change detected. Starting incremental compilation...
[00:17:36] Found 0 errors. Watching for file changes.
```

Case when error... cf next slide

Edit file to get Compile Error with -w (watch)

The image shows a terminal window titled "Cmder" running on Windows. It displays the output of a TypeScript compilation process. The terminal shows the following text:

```
[00:18:53] File change detected. Starting incremental compilation...
a.ts:6:9 - error TS1434: Unexpected keyword or identifier.
6       edit file to get error
      ~~~~
a.ts:6:14 - error TS1434: Unexpected keyword or identifier.
6       edit file to get error
      ~~~~
a.ts:6:19 - error TS1434: Unexpected keyword or identifier.
6       edit file to get error
      ~~
a.ts:6:22 - error TS1434: Unexpected keyword or identifier.
6       edit file to get error
      ~~~
[00:18:53] Found 4 errors. Watching for file changes.
```

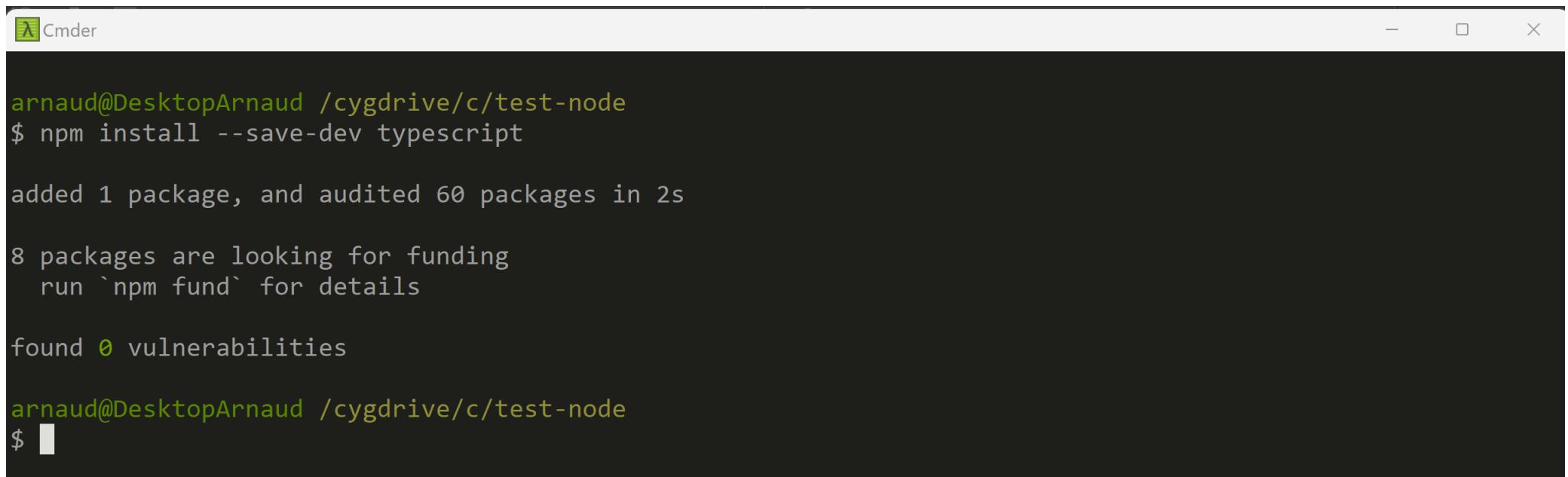
On the left, there is a code editor window showing a file named "a.ts". The code contains the following:

```
1 no usages
2 export class A {
3
4     no usages
5     constructor() {
6         console.log('Hey');
7         edit file to get error
8     }
}
```

The line "edit file to get error" is underlined with a red squiggle, indicating it is a syntax error. The terminal output shows four separate errors, each pointing to this line at different character positions (9, 14, 19, and 22).

Install « typescript » in package.json
devDependencies
(devDependencies != dependencies, != global)

\$ npm install –save-dev typescript



```
arnaud@DesktopArnaud /cygdrive/c/test-node
$ npm install --save-dev typescript

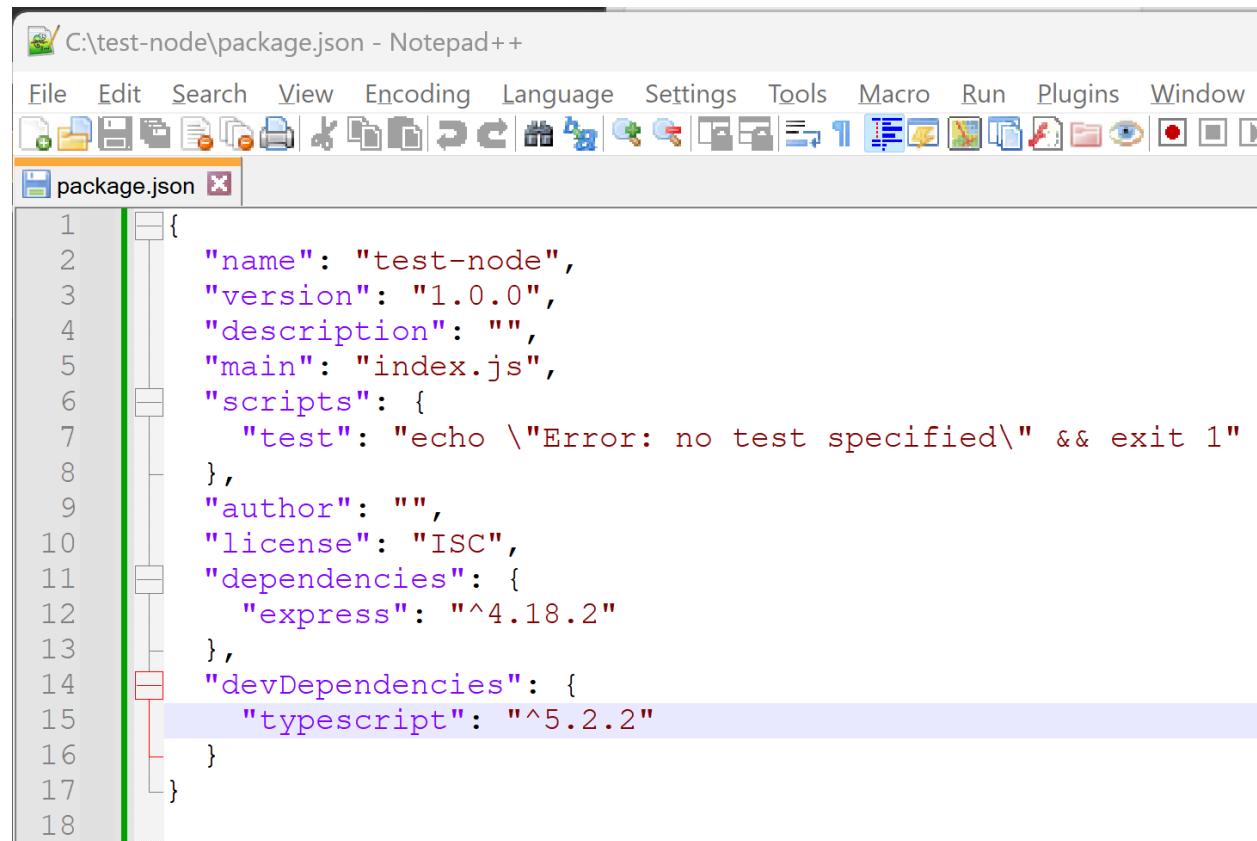
added 1 package, and audited 60 packages in 2s

8 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

arnaud@DesktopArnaud /cygdrive/c/test-node
$ █
```

devDependencies...



The screenshot shows a Notepad++ window displaying a `package.json` file. The file content is as follows:

```
1 {  
2   "name": "test-node",  
3   "version": "1.0.0",  
4   "description": "",  
5   "main": "index.js",  
6   "scripts": {  
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"  
8   },  
9   "author": "",  
10  "license": "ISC",  
11  "dependencies": {  
12    "express": "^4.18.2"  
13  },  
14  "devDependencies": {  
15    "typescript": "^5.2.2"  
16  }  
17}  
18
```

The line `"typescript": "^5.2.2"` under the `devDependencies` section is highlighted with a light purple background.

Edit package.json « scripts »

```
package.json x

1  {
2      "name": "test-node",
3      "version": "1.0.0",
4      "description": "",
5      "main": "index.js",
6      "scripts": {  
7          "test": "echo \\\"Error: no test specified\\\" & exit 1",  
8          "compile-typescript": "tsc",  
9          "build": "tsc -w",  
10         "run": "node a.js"  
11     },  
12     "author": "",  
13     "license": "ISC",  
14     "dependencies": {  
15         "express": "^4.18.2"  
16     },  
17     "devDependencies": {  
18         "typescript": "^5.2.2"  
19     }  
20 }
```

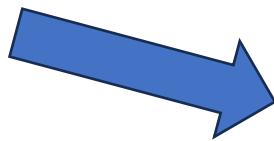
« **\$ npm run compile-typescript** »
... is equivalent to « **\$ tsc** »

« **\$ npm run build** »
... is equivalent to « **\$ tsc -w** »

Notice:
There are Run icons  **for each script line**

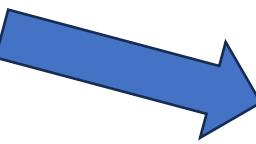
Understanding Angular « ng » client internals

« npm run start »



Internally Call

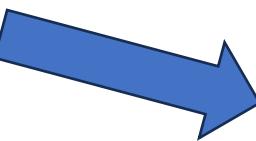
« **ng serve** »



Internally Call

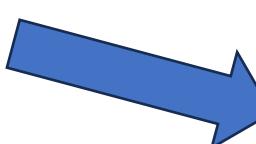
« **tsc -W** »

Compile all *.ts to *.js
(maybe webpack all into a single app.js)



Internall call « **node** » with **express server**

... http server running on port « http://localhost:4200 »



Also Use « **live-reload.js** » ...

so Web browser automatically refresh page on change

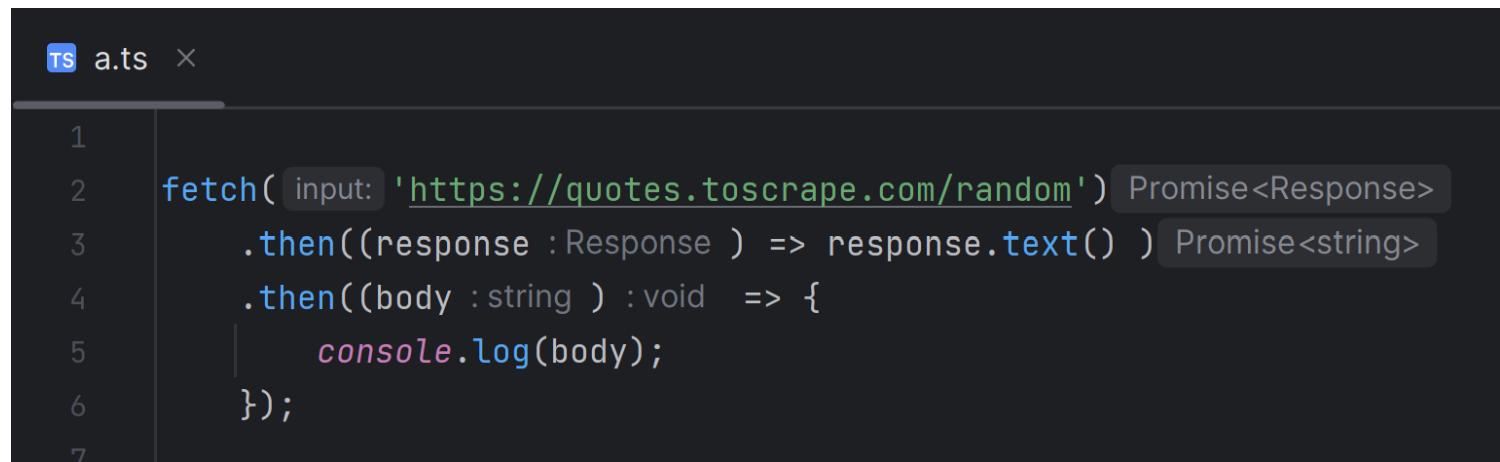
All commands « npm », « ng », « tsc »

Run internally « **node** » !!!

Coffe Break (?)

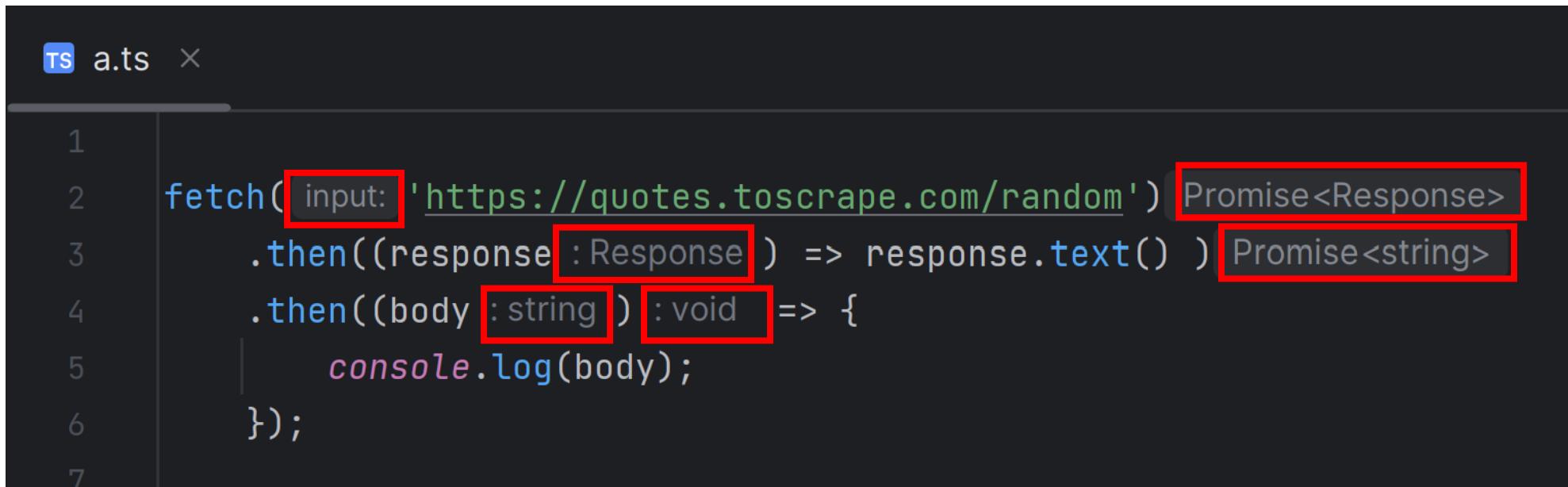
Introducing difficulties with Async Code: Callbacks, Promises, async/await, ... Threads in a mono-threaded-user NodeJs model

A simple http GET fetch()
.. The Hello world on http !



```
TS a.ts ×
1
2  fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
3    .then((response : Response ) => response.text() ) Promise<string>
4    .then((body : string ) : void   => {
5      console.log(body);
6    );
7
```

Noticed WebStorm automatically
show « :type » and parameter « name: » in grey



The screenshot shows a code editor window for a file named 'a.ts'. The code uses the `fetch` API to retrieve a random quote from a specific URL. The code is annotated with type information, which is displayed in grey text. Red boxes highlight several instances of this grey text:

- A red box surrounds the word "input" in the first argument of the `fetch` call.
- A red box surrounds the type annotation "`: Response`" in the `.then` callback.
- A red box surrounds the type annotation "`: string`" in the inner `.then` callback.
- A red box surrounds the type annotation "`: void`" in the inner `.then` callback.
- A red box surrounds the type annotation "`: Promise<Response>`" in the return value of the outer `.then` callback.
- A red box surrounds the type annotation "`: Promise<string>`" in the return value of the inner `.then` callback.

```
1
2 fetch(input: 'https://quotes.toscrape.com/random'): Promise<Response>
3   .then(response: Response) => response.text(): Promise<string>
4   .then(body: string): void => {
5     console.log(body);
6   };
7 }
```

Adding console.log(...)

```
TS a.ts ×

8
9  console.log("1: calling (async) http GET ..");
10
11 fetch( input: 'https://quotes.toscrape.com/random' ) Promise<Response>
12     .then((response :Response ) => {
13         console.log('2: .. got response  http Status code: ' + response.status);
14         return response.text() // => cascade (+return) another async reading for consuming http response body ...
15     }) Promise<string>
16     .then((body :string ) :void  => {
17         console.log('3: .. got response text body');
18         console.log(body);
19     });
20
21 console.log('4: .. done launched async fetch() + attaching .then() callback ');
22
```

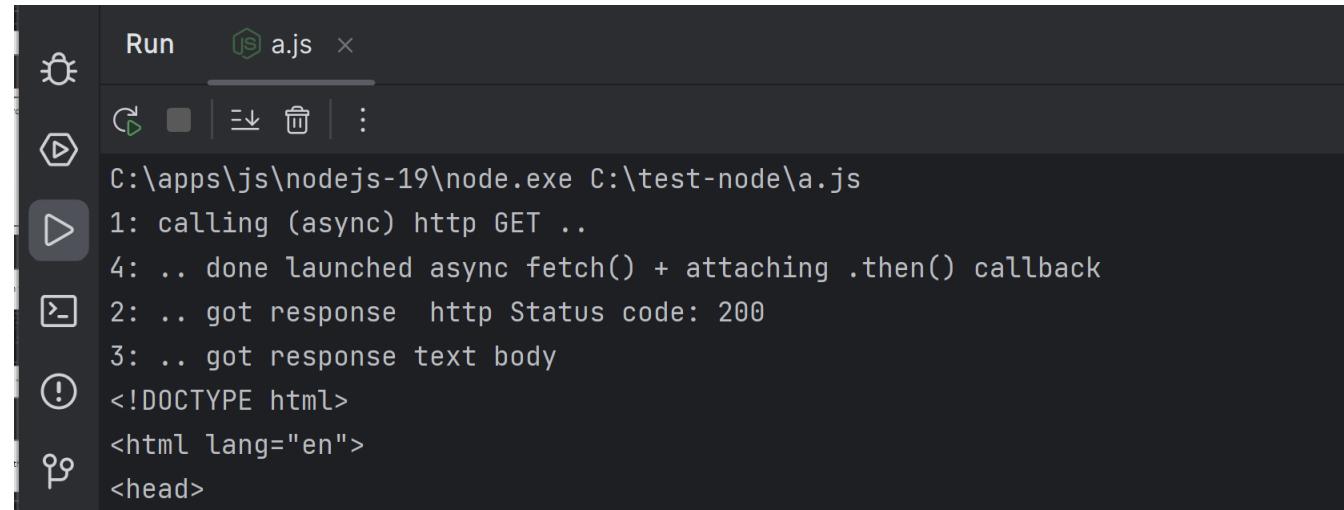
In which order are the console.log()**s** printed?

1, 2, 3, 4 ? ...

other? which order?



Response: 1, 4, 2, 3 !!



The screenshot shows a terminal window with the following details:

- Tab Bar:** Run, a.js (highlighted), and a close button.
- Icons:** A vertical toolbar on the left with icons for file operations (copy, paste, delete, etc.), a play button, and a warning symbol.
- Output Area:** The terminal output is displayed below the tabs.
 - Path: C:\apps\js\nodejs-19\node.exe C:\test-node\a.js
 - Log entries:
 - 1: calling (async) http GET ..
 - 4: .. done launched async fetch() + attaching .then() callback
 - 2: .. got response http Status code: 200
 - 3: .. got response text body
 - <!DOCTYPE html>
 - <html lang="en">
 - <head>

Put Breakpoint and Debug

The screenshot shows a development environment in VS Code with the following details:

- Project Explorer:** Shows a project structure with files like `a.ts`, `c.ts`, `b.ts`, and `i-prefer-typescript-than-js.ts`.
- Editor:** The file `a.ts` is open, displaying code that uses `fetch` to make an HTTP request. A red circular breakpoint icon is visible on line 17.
- Terminal:** The terminal tab shows the command `a.js`.
- Debug View:** The "Debug" tab is active, showing the "Threads & Variables" tab. It lists the "Main Thread" and shows a local variable `body` with its value expanded to show HTML content and a length of 2,691 characters.
- Status Bar:** The status bar at the bottom indicates the TypeScript version is 5.2.2, and the file encoding is UTF-8.

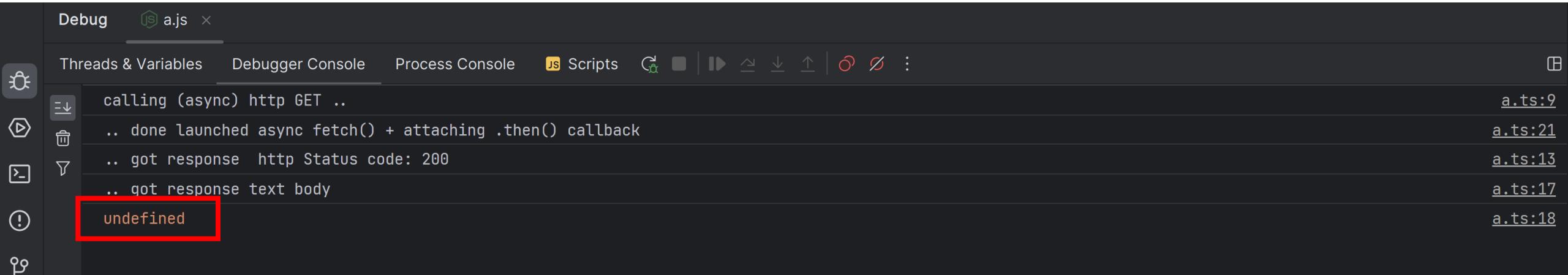
Can you spot the bug here ?

```
11  fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
12    .then((response :Response ) :void  => {
13      console.log(.. got response  http Status code: ' + response.status);
14      response.text() // => cascade (missing return) another async reading for consuming http response body ...
15    }) Promise<void>
16    .then((body :void ) :void  => {
17      console.log(.. got response text body');
18      console.log(body);
19    });
20
21  console.log(.. done launched async fetch() + attaching .then() callback ');
```

Spot the Bug ... 4 Clues

- 1/ No Compile ERROR ... but Running => see in console logs
- 2/ Executing in Debug mode => check variables values
- 3/ WebStorm underline « .text() » as a possible warning
- 4/ Can you Spot the inconsistent Type infered by WebStorm
(not a bug in WebStorm)

Clue 1/ Run, see in console log
Understand the « undefined » instead of <html>... ?



A screenshot of the VS Code debugger interface, specifically the 'Debugger Console' tab. The console shows the following log entries:

- calling (async) http GET .. a.ts:9
- .. done launched async fetch() + attaching .then() callback a.ts:21
- .. got response http Status code: 200 a.ts:13
- .. got response text body a.ts:17
- undefined** a.ts:18

The last entry, "undefined", is highlighted with a red rectangular box.

Clue 2 / Debug with Breakpoint ... indeed « body » is undefined

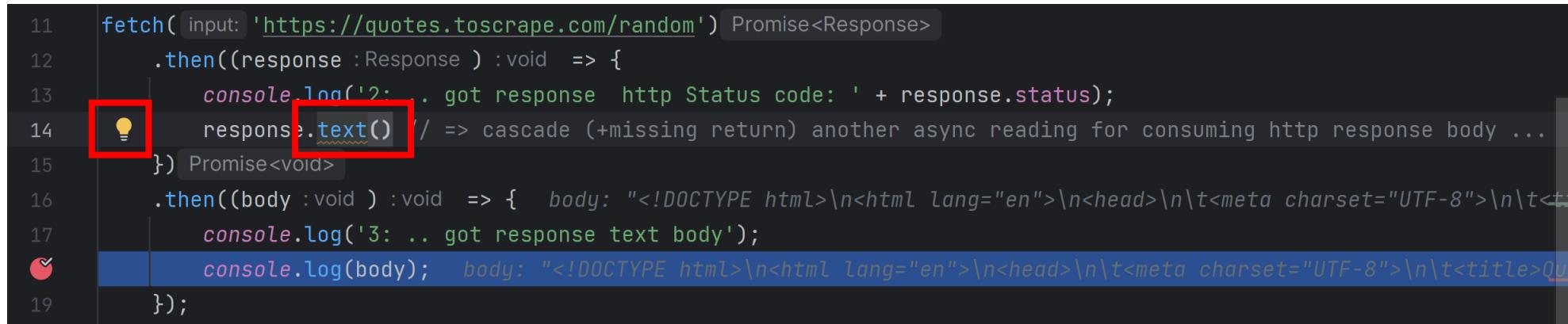
The screenshot shows a Visual Studio Code (VS Code) interface with the following details:

- Project Explorer:** Shows a project structure with files like `a.ts`, `c.ts`, `b.ts`, and `i-prefer-typescript-than-js.ts`.
- Editor:** The `a.ts` file is open, containing the following code:

```
8
9 console.log("1: calling (async) http GET ..");
10
11 fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
12     .then((response :Response ) :void  => {
13         console.log('2: .. got response  http Status code: ' + response.status);
14         response.text() // => cascade (+missing return) another async reading for consuming http response body ...
15     }) Promise<void>
16     .then((body :void ) :void  => { body: undefined
17         console.log('3: .. got response text body');
18         console.log(body); body: undefined
19     });
20
21 console.log('4: .. done launched async fetch() + attaching .then() callback ');
22
23
24
```

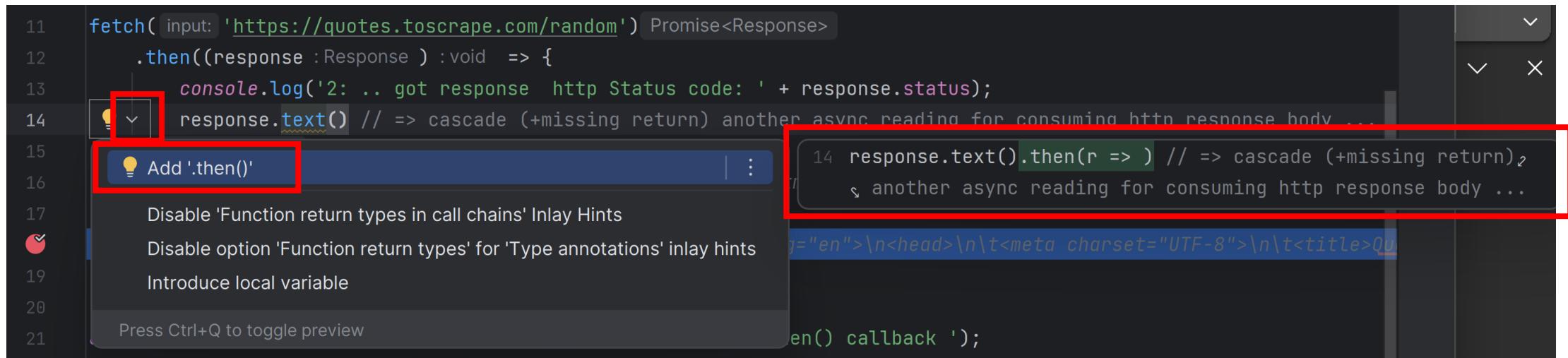
A red box highlights the line `body: undefined` at line 16.
- Debug Bar:** The "Debug" tab is selected, showing the `a.js` file. It lists the "Main Thread" and other frames.
- Debugger Console:** Shows the expression `body` being evaluated. A red box highlights the value `body = undefined` under the "Local" scope.
- Status Bar:** Shows the version `TypeScript 5.2.2`, time `18:9`, and encoding `UTF-8`.

Clue 3/ .text() in underlined in WebStorm ?



```
11 fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
12     .then((response : Response) : void => {
13         console.log('2: .. got response http Status code: ' + response.status);
14         response.text() // => cascade (+missing return) another async reading for consuming http response body ...
15     ) Promise<void>
16     .then((body : void) : void => {   body: "<!DOCTYPE html>\n<html lang="en">\n<head>\n\t<meta charset="UTF-8">\n\t<t
17         console.log('3: .. got response text body');
18         console.log(body);   body: "<!DOCTYPE html>\n<html lang="en">\n<head>\n\t<meta charset="UTF-8">\n\t<title>Qu
19     });

```



```
11 fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
12     .then((response : Response) : void => {
13         console.log('2: .. got response http Status code: ' + response.status);
14         response.text() // => cascade (+missing return) another async reading for consuming http response body ...
15         |: Add '.then()' | ...
16             Disable 'Function return types in call chains' Inlay Hints
17             Disable option 'Function return types' for 'Type annotations' inlay hints
18             Introduce local variable
19
20             Press Ctrl+Q to toggle preview
21

```

The screenshot shows a tooltip for the `response.text()` call. The tooltip contains the following text:

- 14 response.text().then(r =>) // => cascade (+missing return),
↳ another async reading for consuming http response body ...

Check « .text() » : Browsing code -> definition !

The screenshot shows a code editor interface with two tabs at the top: "a.ts" and "lib.dom.d.ts". The "lib.dom.d.ts" tab is active and has a red box around it. The code in the editor is:

```
3151     formData(): Promise<FormData>;
3152
3153     json(): Promise<any>;
3154
3155     text(): Promise<string>;
3156 }
3157
3158 interface BroadcastChannelEventMap {
3159     "message": MessageEvent;
3160     "messageerror": MessageEvent;
3161 }
```

Below the code, there is a detailed description of the `BroadcastChannel` interface:

The `BroadcastChannel` interface represents a named channel that any browsing context of a given origin can subscribe to. It allows communication between different documents (in different windows, tabs, frames or iframes) of the same origin. Messages are broadcasted via a `message` event fired at all `BroadcastChannel` objects listening to the channel, except the object that sent the message.

Note: This feature is available in Web Workers

Supported by Chrome 54, Chrome Android 54, Edge 70, Firefox 38, Opera 41, Safari 15.4, Safari iOS 15.4

Body > text()

Clue 4/ Inconsistent types

... was expecting « string », got « void »

```
TS a.ts × ⋮
8
9  console.log("1: calling (async) http GET ..");
10
11 fetch( input: 'https://quotes.toscrape.com/random') Promise<Response>
12     .then((response : Response) : void => {
13         console.log('2: .. got response http Status code: ' + response.status);
14         response.text() // => cascade (+missing return) another async reading for consuming http response body ...
15     }) Promise<void>
16     .then((body : void) : void => {
17         console.log('3: .. got response text body');
18         console.log(body);
19     });
20
21 console.log('4: .. done launched async fetch() + attaching .then() callback ');
22
```

Promise and nested Callbacks => Cascading Hell

async/await Solution

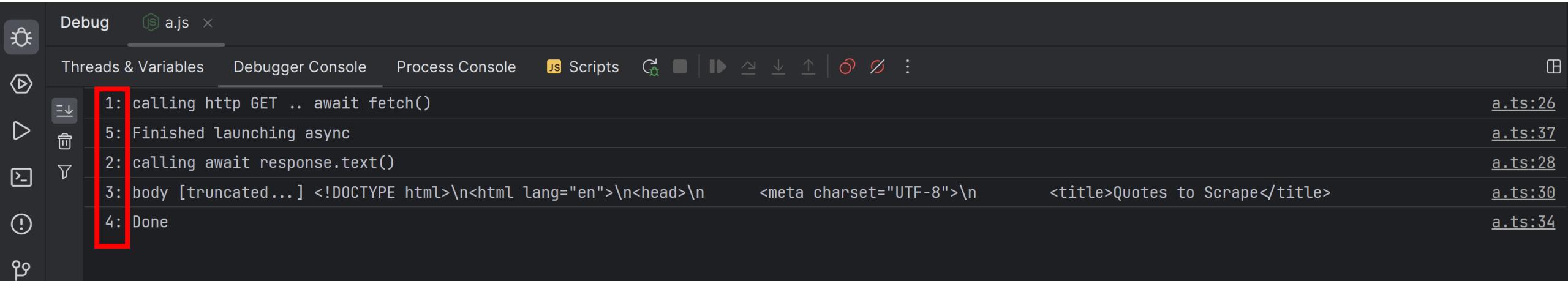
Cf also RxJS (out of scope here... But used in Angular)

Transforming fetch().then() ... to await fetch()

```
ts a.ts ×
⚠ 1 ⚠ 1 ✅ 1 ^

24 // @ts-ignore
  1 usage
25 async function asyncFetchAndLog() :Promise<void> {
26   console.log("1: calling http GET .. await fetch()");
27   let response :Response = await fetch(input: 'https://quotes.toscrape.com/random');
28   console.log("2: calling await response.text()");
29   let body :string = await response.text();
30   console.log("3: body [truncated...] " + body.replace(searchValue: /\n/g, replaceValue: '\n').substring(0, 100));
31 }
32
33 asyncFetchAndLog().then(resp :void => {
34   console.log('4: Done');
35 });
36
37 console.log('5: Finished launching async');
```

Run async/await ...



The screenshot shows the VS Code interface with the 'Debug' tab selected. The title bar indicates the file being debugged is 'a.js'. The left sidebar has several icons: a sun-like icon, a play/pause icon, a stop icon, a close icon, and a refresh icon. The main area is the 'Threads & Variables' tab of the Debug Console, which displays the following log output:

```
1: calling http GET .. await fetch() a.ts:26
5: Finished launching async a.ts:37
2: calling await response.text() a.ts:28
3: body [truncated...] <!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Quotes to Scrape</title> a.ts:30
4: Done a.ts:34
```

The first line (1: calling ...) is highlighted with a red rectangle.

Coffe Break / Lunch Break (?)

Next Steps...