



# DEBUGGING JAVA SCRIPT BASED WEB APPLICATIONS LOCALLY USING VSCODE

---

By Meshal Al Yami from Confidential Team

@Mesh3l\_911 | @\_Conflab

06 of Jan, 2023

## Table of Contents

What's Debugging? .....	3
What's Debugger? .....	4
Debugger Extensions for Multiple Languages in VSCode (PHP, C#, PYTHON and C/C++).....	5
Debugging JS Based Apps / Debugging Features in VSCode .....	6
Launch Vs Attach .....	8
Launch: .....	8
Attach: .....	8
Breakpoints, Continue, Step Over, Step Into, Step Out, Restart and Stop.....	9
References:.....	12



## What's Debugging?

بكل إختصار وبشكل سطحي ال Debugging هو عملية تتبع الكود البرمجي خلال فترة عمل التطبيق (Run-Time) لعدة أسباب كإكتشاف الأخطاء أو التأكد من أن التطبيق يعمل بالشكل الصحيح أو فهم وتحليل أجزاء من التطبيق عند حالات محددة.

وفي حالتنا كمختبرين إختراق ال Debugging عملية مهمة لتحليل وفهم عمل التطبيق وإكتشاف الأخطاء والسلوكيات المسببة للثغرات (Security Flaws).

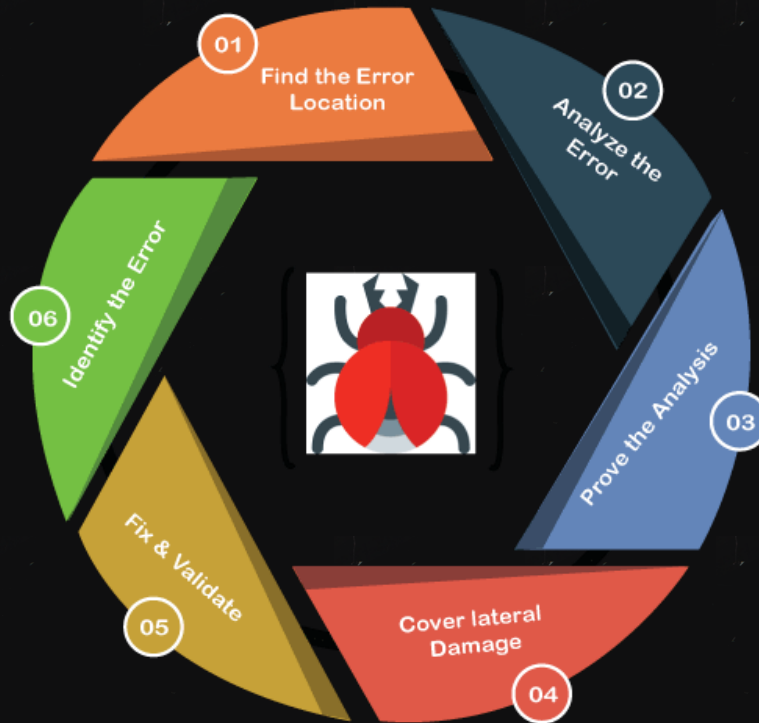


Figure1 - Common Debugging Process



## What's Debugger?

عشان نسوي عملية Debugging فيه عدة طرق، على سبيل المثال لا الحصر:

- مراقبة الـ Logs
- استخدام عبارات الطباعة عند حالات (Distributed Print Statements) محددة

ولكن فيه طريقة أفضل غالبا وهي عن طريق استخدام الـ Debuggers، وهي عبارة عن برامج تملك عدد من الخصائص تمكننا من عمل Debugging بطريقة أكثر كفاءة كإيقاف عمل البرنامج عند حالات محددة أو إظهار الـ Call stack و الـ Application's memory .



# Debugger Extensions for Multiple Languages in VSCode (PHP, C#, PYTHON and C/C++)

VSCode فيه: built-in debugging support for the Node.js runtime

لكن باقي اللغات تحتاج Debugger Extensions

PHP:

<https://marketplace.visualstudio.com/items?itemName=xdebug.php-debug>

C#:

<https://marketplace.visualstudio.com/items?itemName=ms-python.python>

Python:

<https://marketplace.visualstudio.com/items?itemName=ms-python.python>

C/C++

<https://marketplace.visualstudio.com/items?itemName=ms-vscode.cpptools>



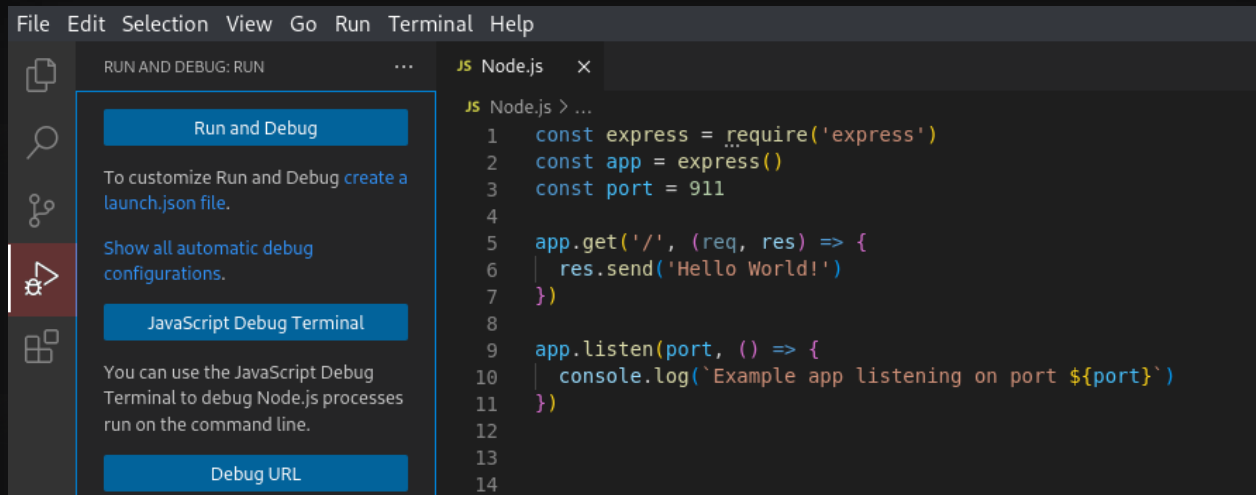
# Debugging JS Based Apps / Debugging Features in VSCode

في هذا السكشن بإذن الله بنشرح بشكل مبسط كيف راح نسوي Debugging ل NodeJS Web App ونتعرف على بعض ميزات ال VSCode فال Debugging

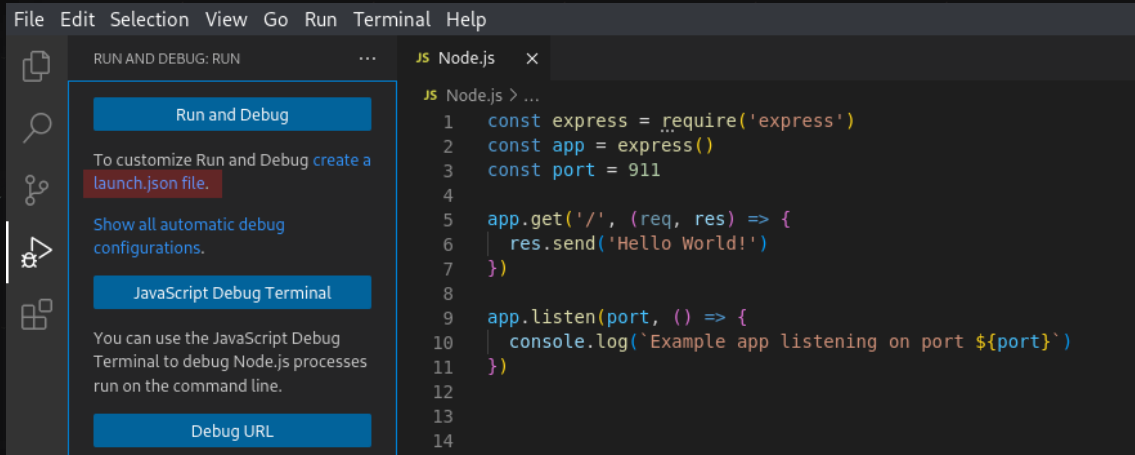
فالبداية من الخيارات اللي باليسار راح نختار

## 1. Run and Debug (Ctrl+Shift+D)

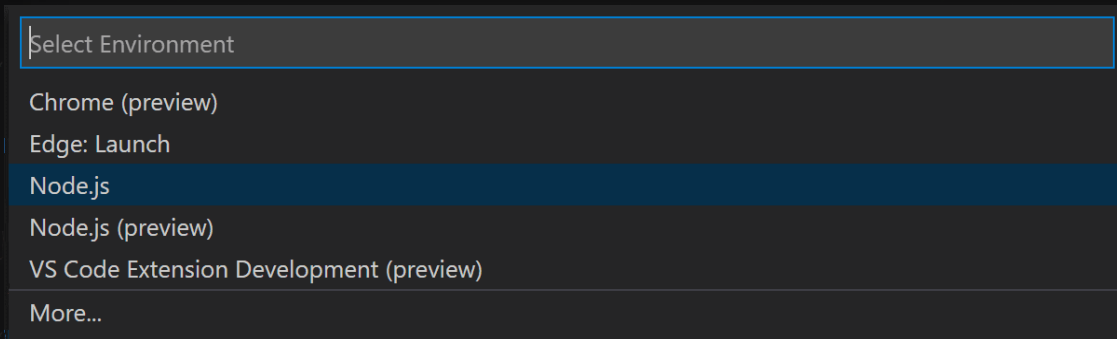
# للتسهيل حددت الخطوات بخلفية حمراء



## 2. Create a launch.json file



## 3. Node.JS



راح ينتج عندنا ملف ال Configuration بالشكل هذا

```
File Edit Selection View Go Run Terminal Help
RUN AND D... Launch
VS Node.js launch.json
VARIABLES
.vscode > {} launch.json > ...
1 {
2   // Use IntelliSense to learn about possible attributes.
3   // Hover to view descriptions of existing attributes.
4   // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
5   "version": "0.2.0",
6   "configurations": [
7     {
8       "type": "node",
9       "request": "launch",
10      "name": "Launch Program",
11      "skipFiles": [
12        "<node_internals>/**"
13      ],
14      "program": "${workspaceFolder}/Node.js"
15    }
16  ]
17 }
```

قبل نكمل ودي أشرح الفرق بين Launch و Attach

## Launch Vs Attach

### :Launch

هنا ال VSCode هو اللي راح يشغل التطبيق بعدين راح يسوي Attach لل Debugger بالبروسيس اللي اشتغلت بشكل تلقائي

### :Attach

هنا راح نحتاج نسوي Attach لل Debugger بالبروسيس اللي شغالة مسبقا

الآن نكمل طريقنا , بعد ماسوينا خطوة رقم 3 الآن باقي علينا نحت Breakpoints ونشغل التطبيق ونشوف ....

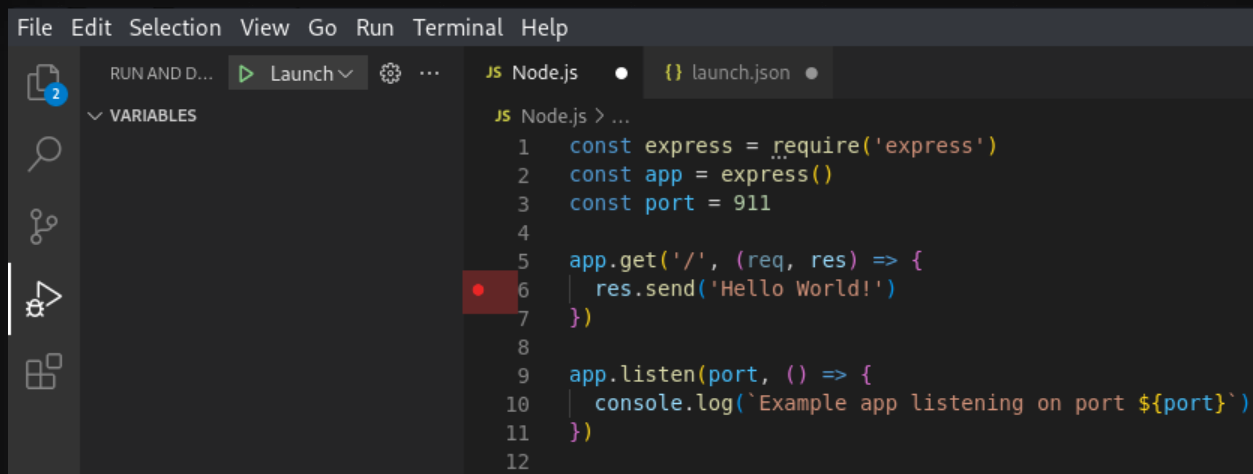




# Breakpoints, Continue, Step Over, Step Into, Step Out, Restart and Stop

ال Breakpoint من اسمها هي المكان الي ودنا يوقف البرنامج عنده بوقت التشغيل.

كيف نسويها؟ بكل بساطة نروح للسطر الي نحتاج نحط عنده Breakpoint ونضغط F9



```
File Edit Selection View Go Run Terminal Help
RUN AND D... Launch
VARIABLES
JS Node.js > ...
1 const express = require('express')
2 const app = express()
3 const port = 911
4
5 app.get('/', (req, res) => {
6   res.send('Hello World!')
7 })
8
9 app.listen(port, () => {
10  console.log(`Example app listening on port ${port}`)
11 })
12
```

والآن نشغل التطبيق ونكمل باقي الخصائص

Run > Start Debugging

بعدين نروح للمتصفح ونسوي ريفريش او اي اكشن



Node.js - Debugging - Visual Studio Code [Superuser]

127.0.0.1:911/

File Edit Selection View Go Run Terminal Help

VARIABLES

- Local
  - req: IncomingMessage {\_read...
  - res: ServerResponse {\_event...  
this: undefined
  - Closure
  - Global

WATCH

CALL STACK

- Launch ... PAUSED ON BREAKPOINT
  - <anonymous> Node.js 6:7
    - handle /root/Desktop/node...
    - next /root/Desktop/node\_mo...
    - dispatch /root/Desktop/hod...
    - handle /root/Desktop/node...
    - <anonymous> /root/Desktop/...
    - process\_params /root/Desk...
    - next /root/Desktop/node\_mo...
    - expressInit /root/Desktop/...
    - handle /root/Desktop/node...

LOADED SCRIPTS

BREAKPOINTS

- Caught Exceptions
- Uncaught Exceptions
- Node.js

```
Node.js > ...
1 const express = require('express')
2 const app = express()
3 const port = 911
4
5 app.get('/', (req, res) => {
6   res.send('Hello World!')
7 })
8
9 app.listen(port, () => {
10  console.log(`Example app listening on port ${port}`)
11 })
```

DEBUG CONSOLE

```
/usr/bin/node ./Node.js
Example app listening on port 911
Node.js:10
```

Hello World!

Ln 11, Col 3 Spaces: 2 UTF-8 LF JavaScript



وأخيرا نشوف باقي الخصائص بالترتيب من اليسار لليمين:



### :Continuo

بكل إختصار نقول للبرنامج كمل طريقك للBreakpoint اللي بعدها إذا فيه

### :Step Over

هنا نقوله نفذ الMethod ولا تدخل الBlock الخاص فيها

### :Step Into

نفذ الMethod وادخل الBlock الخاص فيها وكمل سطر سطر لحد الإنتهاء منه

### :Step Out

في حال على سبيل المثال دخلنا داخل الMethod Block هنا نقدر نتراجع بالخطوات لحد ما نخرج منه

### :Restart/Stop

لإعادة وإغلاق التنفيذ الحالي



## References:

{1} <https://code.visualstudio.com/>

{2} <https://expressjs.com/en/starter/hello-world.html>

{3} <https://code.visualstudio.com/docs/editor/debugging>

