
Debugging Javascript with VSCode

1 message

Michael Tadyshak <michael.tadyshak@nfm.com>

To: Michael Tadyshak <mike.tadyshak@gmail.com>

1. Enable Remote Debugging

To allow remote debugging, you need to start Chrome with the `--remote-debugging-port` flag. This flag specifies the port on which Chrome will listen for debugging connections.

```
chrome --remote-debugging-port=9222
```

Enables remote debug over HTTP on the specified port.

2. Disable Extensions (Optional)

Sometimes, extensions can interfere with debugging. To start Chrome without extensions, use

```
chrome --disable-extensions
```

3. Open a Specific URL (Optional)

If you want Chrome to open a specific URL when starting, you can include it in the command

```
chrome --remote-debugging-port=9222 http://localhost:8000
```

4. Incognito mode

Chrome `--incognito`

How to Use These Options

On Windows:

Locate Chrome's Executable Path

The path is usually something like C:\Program Files\Google\Chrome\Application\chrome.exe.
Create a Shortcut or Use Command Prompt

To create a shortcut with these options:

Right-click on the desktop, select New -> Shortcut.

Enter the path to Chrome's executable followed by the desired command-line options

Setting Up Debugging in VS Code

Make sure your launch.json configuration in VS Code matches the port you specified:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "type": "chrome",
      "request": "launch",
      "name": "Launch Chrome against localhost",
      "url": http://localhost:8000,
      "webRoot": "${workspaceFolder}",
      "port": 9222 // Make sure this matches the port specified above
    }
  ]
}
```

Using these steps, you can start Chrome with debugging enabled and set up Visual Studio C for a smooth debugging experience.

[Useful Chrome Command-Line Switches and What to Do with Them - Make Tech Easier](#)

New ChatGPT Starts

Debugging JavaScript applications using the Attach mode in Visual Studio Code (VS C connect to an already running instance of a Node.js or web application. This is useful | server running or a browser already open and you want to attach the debugger to it wi

Here's how you can set up debugging in Attach mode for both Node.js and browser-ba applications:

1. Debugging Node.js in Attach Mode

1.1 Modify Your Node.js Application

Start your Node.js application with the `--inspect` or `--inspect-brk` flag to enable debugg


- `--inspect` allows you to attach the debugger without pausing at the start.
- `--inspect-brk` pauses execution at the first line, which is useful for debugging startu

For example:

```
node --inspect app.js
```

```
node --inspect-brk app.js
```

1.2 Configure launch.json in VS Code

1. **Open Your Project in VS Code** and navigate to the Debug view by clicking the play icon pressing `Ctrl+Shift+D` (`Cmd+Shift+D` on macOS).
2. **Create or Modify launch.json:**
 - Click the gear icon () to open the configuration file. Select "Node.js" when prompted
 - Add an attach configuration to launch.json:

```
{  
  "version": "0.2.0",  
  "configurations": [  
    {  
      "type": "node",
```

```

    "request": "attach",
    "name": "Attach to Node.js",
    "address": "localhost",
    "port": 9229, // Default port for Node.js --inspect
    "restart": true,
    "localRoot": "${workspaceFolder}",
    "remoteRoot": null // Set this if the remote code is in a different location
  }
]
}

```

1.
 - Ensure the "port" matches the port specified when you started your Node.js application
2. **Start Debugging:**
 - Start your Node.js application with the --inspect flag.
 - In VS Code, go to the Debug view, select "Attach to Node.js" from the dropdown, and button or press F5.

2. Debugging Browser-Based JavaScript in Attach Mode

2.1 Enable Remote Debugging in Chrome

You need to start Chrome with remote debugging enabled if it's not already running:

```
chrome --remote-debugging-port=9222
```

Or, if you're using a different browser or a specific setup, adjust the command accordingly.

2.2 Configure launch.json in VS Code

1. **Open Your Project in VS Code** and navigate to the Debug view.
2. **Create or Modify launch.json:**
 - Click the gear icon (⚙️) to open the configuration file. Select "Chrome" if prompted.
 - Add an attach configuration to launch.json:

```

{
  "version": "0.2.0",
  "configurations": [
    {
      "type": "chrome",
      "request": "attach",

```

```

"name": "Attach to Chrome",
"port": 9222, // Default port for Chrome remote debugging
"webRoot": "${workspaceFolder}",
"sourceMapPathOverrides": {
  "webpack:///*": "${webRoot}/*",
  "webpack:///./~/*": "${webRoot}/node_modules/*"
}
}
]
}

```

1. Ensure the "port" matches the port specified when you started Chrome.

2. **Start Debugging:**

- Make sure Chrome is running with the remote debugging port enabled.
- In VS Code, go to the Debug view, select "Attach to Chrome" from the dropdown, and click the button or press F5.
-

Summary

- **For Node.js:** Use `--inspect` or `--inspect-brk` to start your Node.js application and configure attach to the specified port.
- **For Browser-based JavaScript:** Start Chrome with `--remote-debugging-port`, and configure VS Code to attach to that port.

By following these steps, you can effectively debug your JavaScript applications using attach



Michael Tadyshak
 Customer Service Specialist
 P: +19726683000 Ext. 67265
michael.tadyshak@nfm.com
 Omaha | Des Moines | Kansas City | Dallas/Fort Worth | Austin - Opening 2026



Think before you print

