Debugging using VS Code

Launching the nodeJS process

if you use `npm run` then you can see a list of commands.

We essentially add `--inspect` or `--inspect-brk` onto the `node` executable to force the debugger to become active.

For Jest we also need to add `--runInBand` and `--maxParallel=1` in order to ensure only a single process is started.

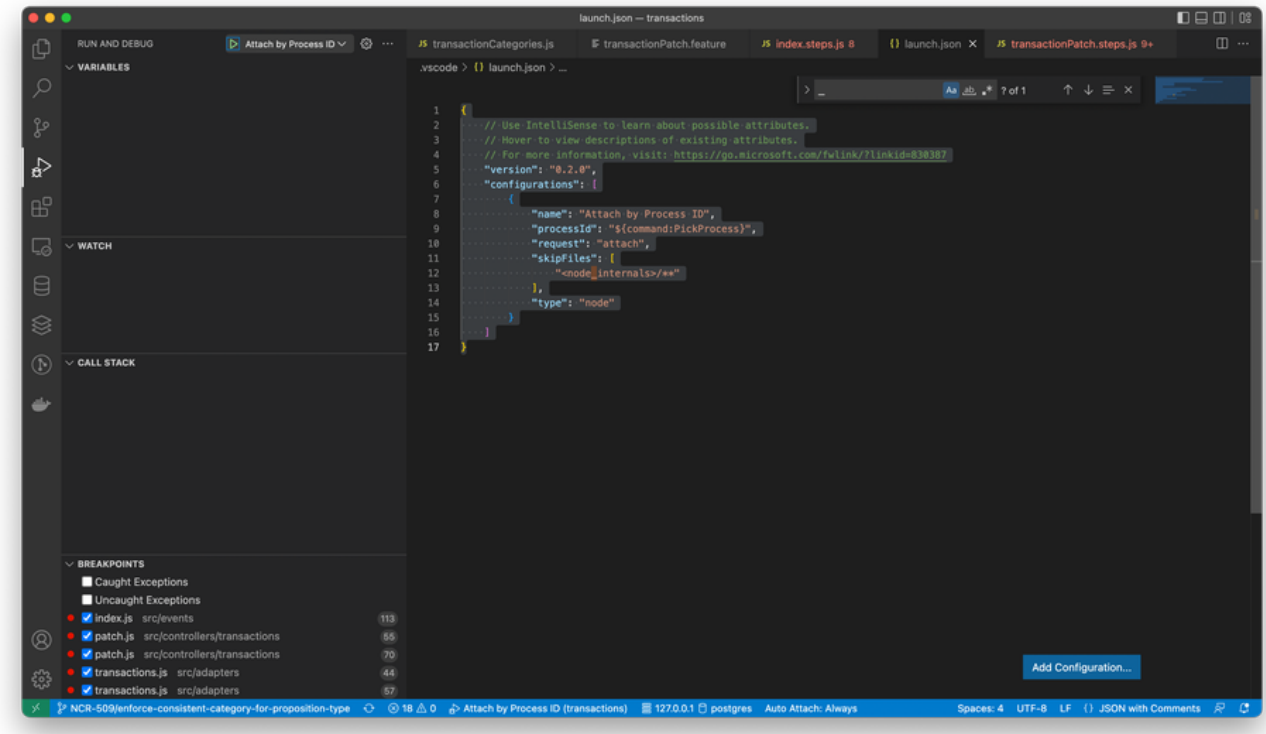
VSCode setup

For now the easiest to setup for local debugging, and most consistent seems to be attaching to a running process.

For unit-tests it's advised not to debug, but for feature tests, I'm using this.

```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
    "version": "0.2.0",
    "configurations": [
        {
            "name": "Attach by Process ID",
            "processId": "${command:PickProcess}",
            "request": "attach",
            "skipFiles": [
                "<node_internals>/**"
            ],
            // the following line is only required for TypeScript projects
            "outFiles": ["${workspaceFolder}/dist/**/*.js"],
            "type": "node"
        }
    ]
}
```

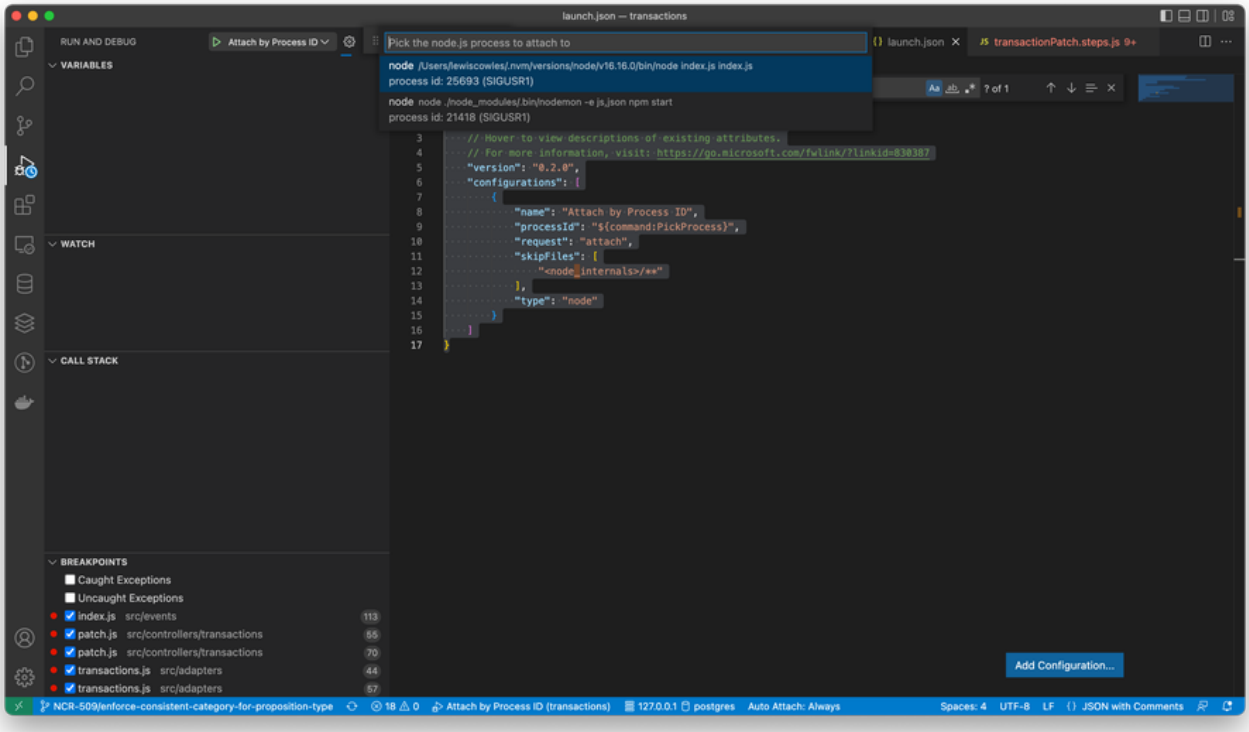
Screenshot of VSCode debugger tab with the change in launch.json



Screenshot of the launch.json for `Attach by Process ID` debug feature of VS Code

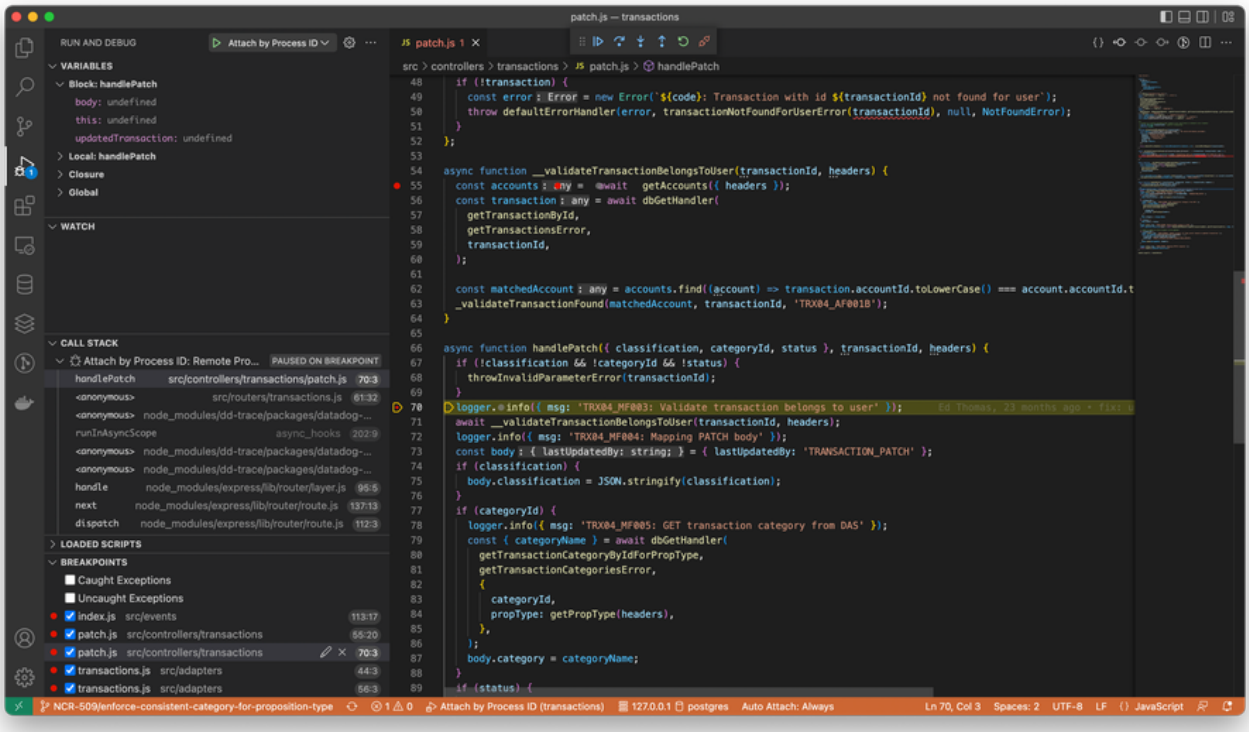
If you click on the play button next to “Attach by Process ID”, you should see a drop-down menu to select a process.

> In this case we are running the project via `npm start:dev` which launches two nodejs processes. You should be aware that nodemon is just a process monitor, so we can ignore the bottom process 21418, and instead use 25693 which has a familiar command `node index.js` (which for some reason is repeated twice 😃.



Screenshot of Pick Process dropdown UI in VS Code, for selecting the process for the NodeJS debugger to attach to.

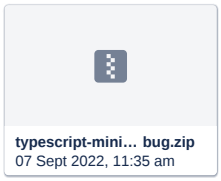
You will now be in a debug window, where you can set breakpoints in code, step, inspect and watch variables.

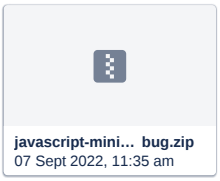


Screenshot of code with an active breakpoint using the debugger.

Sample repositories

The below attachments should be fairly minimal, yet complete examples of using TypeScript and JavaScript with a debugger (outside of a test-suite). They are single-process, simple express single-file repositories with node-modules.





Sample settings.json lines

```
{
    "debug.javascript.autoAttachFilter": "smart",
    "debug.javascript.autoAttachSmartPattern": [
        "${workspaceFolder}/**",
        "!**/node_modules/**",
        "**/$KNOWN_TOOLS$/**"
    ],
    "debug.focusWindowOnBreak": true,
}
```