TypeScript Basics for Automation Testers – Day 2

Topic: TypeScript Prerequisites

What Do We Need Before Starting TypeScript?

To start writing and running TypeScript programs, we need three things:

1. **Node.js**
2. **TypeScript Compiler (tsc)**
3. **Visual Studio Code (VS Code) Editor**

Why and What is Node.js?

- Node.js allows us to **execute TypeScript and JavaScript programs** outside a browser.
- TypeScript cannot run directly — it needs Node.js to execute the JavaScript file generated after compilation.
- Once TypeScript is compiled to JavaScript, Node.js runs the `.js` file.

In simple terms:
Think of Node.js as an **engine** that helps run your TypeScript code on your computer.

Why and What is TypeScript Compiler (tsc)?

- TypeScript code is not directly understood by browsers or Node.js.
- The **TypeScript Compiler (tsc)** converts `.ts` files into `.js` files.
- After compilation, the `.js` file can be executed using Node.js.

Example:

```
tsc FirstDemo.ts    → generates →  FirstDemo.js
node FirstDemo.js   → runs the code
```

So the flow is:
Write TypeScript → Compile using tsc → Run using Node.js

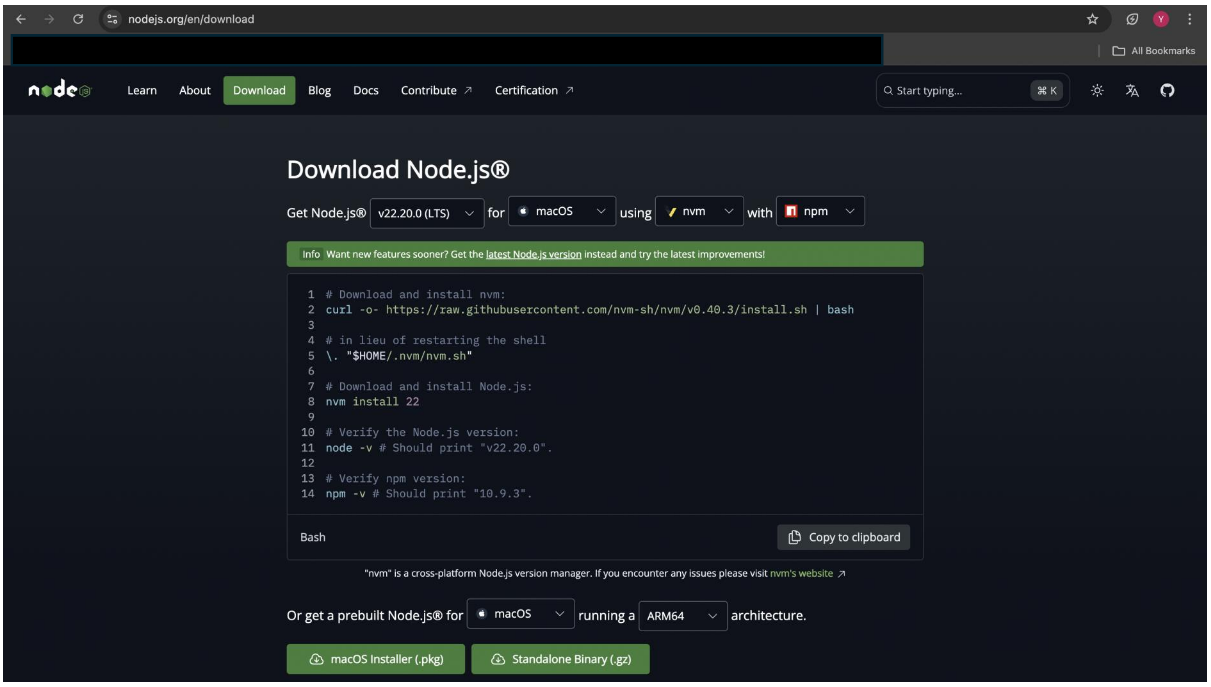
Why VS Code Editor?

- Visual Studio Code is one of the best IDEs from Microsoft.
- It provides great support for TypeScript and JavaScript with helpful extensions, syntax highlighting, and auto-completion.
- It is lightweight, fast, and widely used for web and automation projects.

How to Install Node.js

Step 1: Download Node.js

- Go to the official Node.js website:
  https://nodejs.org
- You'll see two download options:
  - **LTS (Long-Term Support)** → Recommended for most users (includes npm).
  - **Current** → For developers who want the latest features.
- Choose **LTS** and download the version for your operating system.



[redacted]

Download Node.js®

Get Node.js® v22.20.0 (LTS) for macOS using nvm with npm

Info Want new features sooner? Get the latest Node.js version instead and try the latest improvements!

```bash
# Download and install nvm:
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.3/install.sh | bash

# in lieu of restarting the shell
\. "$HOME/.nvm/nvm.sh"

# Download and install Node.js:
nvm install 22

# Verify the Node.js version:
node -v # Should print "v22.20.0".

# Verify npm version:
npm -v # Should print "10.9.3".
```

Bash

"nvm" is a cross-platform Node.js version manager. If you encounter any issues please visit nvm's website

Or get a prebuilt Node.js® for macOS running a ARM64 architecture.

Step 2: Install Node.js

For Windows:

1. Run the downloaded `.msi` file.
2. Follow the setup wizard and use all default options.

3. Default install path:

   ```
   C:\Program Files\nodejs
   ```

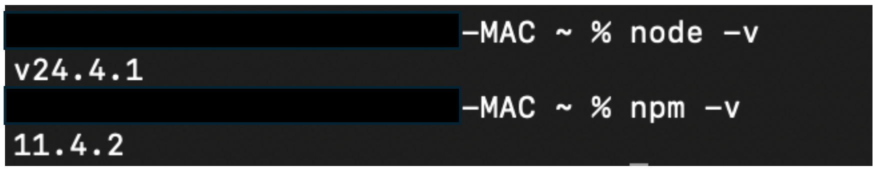
4. Once installation is done, open a new Command Prompt or Terminal.

For macOS:

1. Run the downloaded `.pkg` installer file.
2. Follow the installation steps — it will automatically install both **Node.js** and **npm**.
3. On Mac, Node.js is installed **globally**, and its path is automatically added to your environment variables.
   You don't need to set it manually.
4. To verify, open **Terminal** and run:

   ```
   node -v
   npm -v
   ```

   You should see version numbers for both Node and npm.

   ```
   [redacted]-MAC ~ % node -v
   v24.4.1
   [redacted]-MAC ~ % npm -v
   11.4.2
   ```

Verify Node.js Installation

Open a terminal and type:

```
node -v
```

or

```
node --version
```

Example Output:
```
v22.14.0
```

If you get an error like "node not recognized," it means the **path is not set properly**.

To fix it (Windows):

- Go to **C:\Program Files\nodejs**
- Copy the path
- Open **Environment Variables → System Variables → Path → New → Paste the Node.js path**
- Click **OK**

Installing TypeScript Compiler

When Node.js is installed, it also installs **npm (Node Package Manager)**.
You can use npm to install other tools — including the TypeScript compiler.

Command:

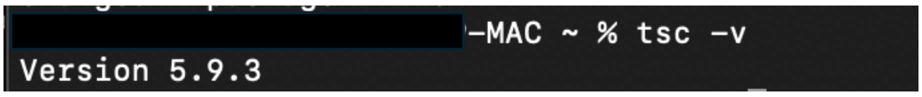
```
npm install -g typescript
```

```
[redacted]-MAC ~ % npm install -g typescript
```

This installs the TypeScript compiler globally, so you can use the `tsc` command anywhere.

Verify installation:

```
tsc -v
```



```
[redacted]-MAC ~ % tsc -v
Version 5.9.3
```

Installing VS Code Editor

1. Download Visual Studio Code from the official website.

   https://code.visualstudio.com/

2. Install it (default settings are fine).
3. Open it and create your first TypeScript project.

Create and Run Your First TypeScript Program

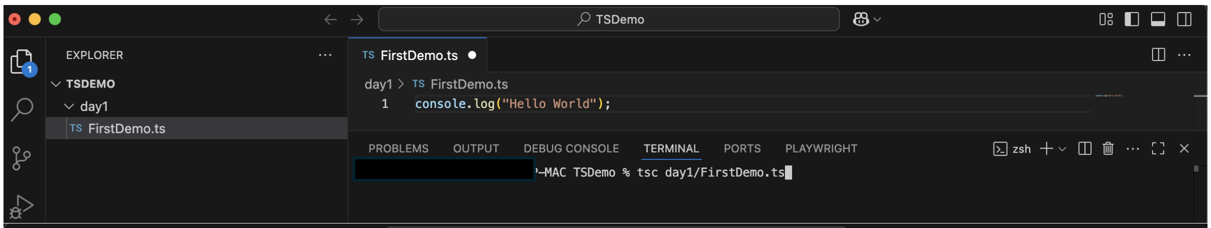
Steps:

1. Open **VS Code**.
2. Create a folder — for example: `TSDemo`
3. Inside `TSDemo`, create a new folder called `day1`.
4. Inside `day1`, create a file named **FirstDemo.ts**
5. Write this code:

   ```
   console.log("Hello World");
   ```

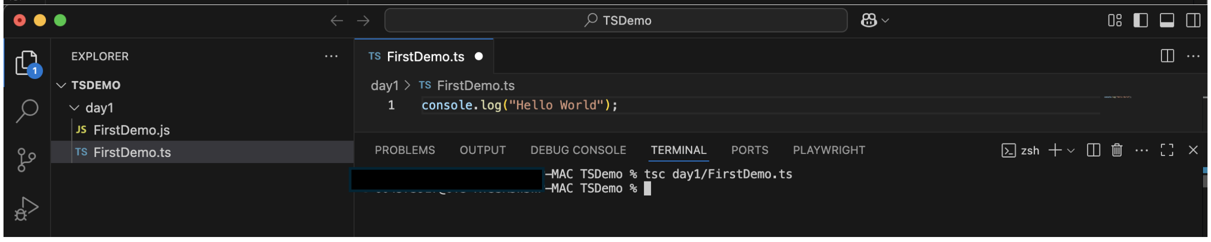
6. Open the terminal in VS Code (**Ctrl + J**)
7. Run the following command to compile:

   ```
   tsc day1/FirstDemo.ts
   ```



```
day1 > FirstDemo.ts
console.log("Hello World");
```

```
[redacted]-MAC TSDemo % tsc day1/FirstDemo.ts
```



```
day1 > FirstDemo.ts
console.log("Hello World");
```

```
[redacted]-MAC TSDemo % tsc day1/FirstDemo.ts
-MAC TSDemo %
```

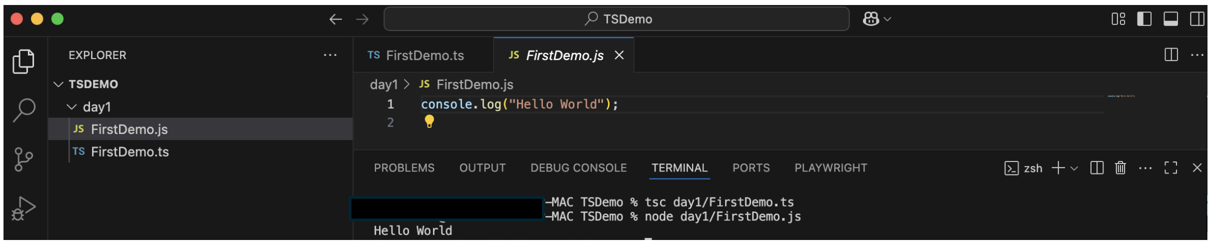
This will generate a new file: **FirstDemo.js**

8. Now run the JavaScript file:

```
node day1/FirstDemo.js
```

Output:

```
Hello World
```



```
day1 > FirstDemo.js
console.log("Hello World");
```

```
[redacted]-MAC TSDemo % tsc day1/FirstDemo.ts
-MAC TSDemo % node day1/FirstDemo.js
Hello World
```

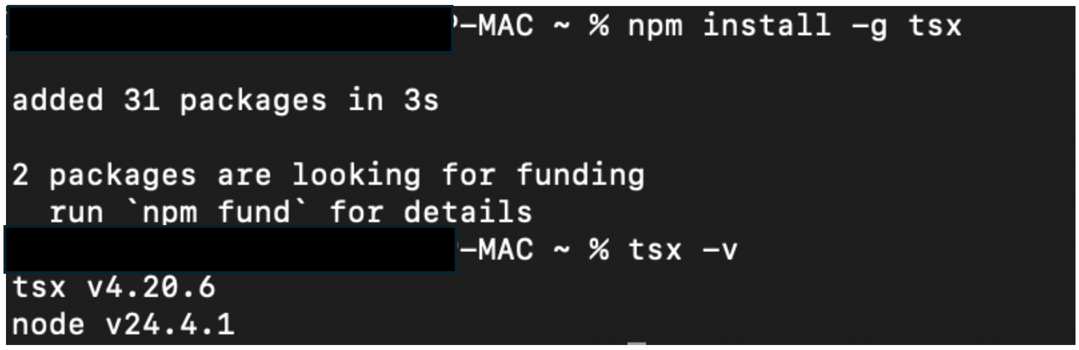
If tsc Command is Not Recognized

Sometimes you may see this error:

```
tsc is not recognized as an internal or external command
```

To fix:

```
npm uninstall -g tsc
npm uninstall -g typescript
npm install -g typescript
npm install -g tsx
```



```
[redacted]-MAC ~ % npm install -g tsx

added 31 packages in 3s

2 packages are looking for funding
  run `npm fund` for details
[redacted]-MAC ~ % tsx -v
tsx v4.20.6
node v24.4.1
```

This reinstalls both the compiler and the TypeScript executor.

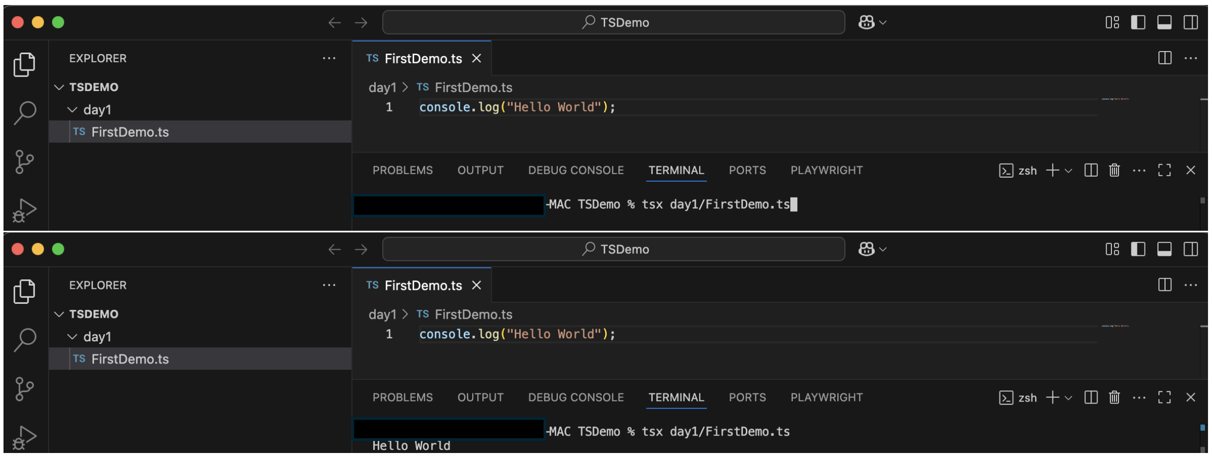
Using tsx for Direct Execution

Normally, you need two steps to run TypeScript:

```
tsc FirstDemo.ts    → generates JS
node FirstDemo.js   → executes JS
```

But with **tsx**, you can run TypeScript directly without compiling separately:

```
tsx FirstDemo.ts
```



```
console.log("Hello World");
```

```
[redacted]-MAC TSDemo % tsx day1/FirstDemo.ts
```

```
console.log("Hello World");
```

```
[redacted]-MAC TSDemo % tsx day1/FirstDemo.ts
Hello World
```

Summary:

Command	Purpose
`npm install -g typescript`	Install TypeScript compiler
`npm install -g tsx`	Install TypeScript executor
`tsc FirstDemo.ts`	Compile `.ts` file to `.js`
`node FirstDemo.js`	Run compiled JavaScript
`tsx FirstDemo.ts`	Directly execute TypeScript

Practice

1. Install Node.js, TypeScript, and VS Code on your system.
2. Create a simple `FirstDemo.ts` file that prints `"Hello from TypeScript"`.
3. Try running it first using `tsc` + `node`, then directly using `tsx`.
4. Verify your Node and TypeScript versions using `node -v` and `tsc -v`.

Common Question: Why do I see a browser icon for `.js` files?

Why do `.js` files show a browser icon?

When you create or compile a TypeScript file (`.ts`), it generates a JavaScript file (`.js`).
You might see a **browser icon** (like Chrome, Safari, or Edge) beside the `.js` file — here's **why:**

Simple explanation:

- Your **computer thinks `.js` files belong to the browser**, because JavaScript was first created for browsers (to make web pages interactive).
- So, macOS or Windows automatically sets the browser (like Chrome or Safari) as the **default app** to open `.js` files.
- That's why the file shows a **browser icon** — it just means, *"If you double-click me, I'll open in the browser."*

But in reality (for developers):

- You should **not** open `.js` files in the browser this way.
- Instead, we **run them using Node.js** through the terminal.

Example:

```
node firstDemo.js
```

When you run this command:

- Node.js executes the JavaScript file.
- You'll see the output (like "Hello World") in the **terminal**, not in the browser.

In short:

What it means	What you should do
Browser icon	Just a default system icon (no impact)
How to open	Always use Node.js in terminal
Example	`node firstDemo.js`

Questions

1. Why do we need Node.js for TypeScript?
2. What is the purpose of the TypeScript compiler (tsc)?
3. Can we run a `.ts` file directly in the browser?
4. What is npm?
5. How do we verify if Node.js and TypeScript are installed?
6. What is the difference between `tsc` and `tsx`?
7. What IDE is recommended for TypeScript development and why?
8. What happens when you install Node.js on your system?

Answers

1. Node.js helps execute JavaScript files generated from TypeScript.
2. The TypeScript compiler converts `.ts` code into `.js` code that can run using Node.js.
3. No, browsers do not understand TypeScript directly — it must first be compiled to JavaScript.
4. npm (Node Package Manager) is used to install and manage Node.js packages like TypeScript.
5. We can check versions using `node -v`, `tsc -v`, and `tsx -v`.
6. `tsc` compiles TypeScript into JavaScript, while `tsx` directly executes TypeScript code without separate compilation.
7. Visual Studio Code is recommended because it provides built-in support for TypeScript, syntax highlighting, and extensions.
8. When Node.js is installed, it automatically installs **npm (Node Package Manager)**, which allows you to install and manage JavaScript/TypeScript packages globally or locally.
