

# TypeScript Crash-course

## Installing TypeScript

You can install the TypeScript compiler with node:

```
1 | npm i typescript -g
```

Then to verify that it is installed, run `tsc -v` to see the version of the compiler. You will get an output like this:

```
message TS6029: Version 1.7.5
```

In addition to the compiler, we also need to install the TypeScript Definition manager for DefinitelyTyped (tsd). You can install tsd with:

```
1 | npm i tsd -g
```

Using TSD, you can search and install TypeScript definition files directly from the community driven DefinitelyTyped repository. To verify that tsd is installed, run tsd with the `version` flag:

```
1 | tsd --version
```

You should get an output like this:

```
>> tsd 0.6.5
```

After `tsd` and `tsc` are installed, we can compile a hello world program:

- make a file called `hello.ts` on your desktop:

```
1 touch ~/Desktop/hello.ts
```

- Then, put some TypeScript code in the file:

```
1 echo "const adder = (a: number, b: number): number => a + b;" > .
```

- Then you can compile the file to JavaScript:

```
1 tsc ~/Desktop/hello.ts
```

- It should output a file in `Desktop/hello.js`:

```
1 var adder = function (a, b) { return a + b; };
```

Now that your TypeScript compiler setup, we can move on to configuring Visual Studio Code.

## Setting up TypeScript for VSCode

You can set up Visual Studio Code to compile your TypeScript code as your work.

- First, open Visual Studio Code
- Make a new window: `File > New Window`
- Then, make a folder on your desktop for a new project:  
`mkdir ~/Desktop/vscode-demo`
- The, open the folder in VSCode: `File > open` and select the `vscode-demo` folder on your desktop.
- Now we need to make three configuration files:
  1. `tsconfig.json` : configuration for the TypeScript compiler
  2. `tasks.json` : Task configuration for VSCode to watch and compile files
  3. `launch.json` : Configuration for the debugger

- The `tsconfig.json` file should be in the root of the project. Let's make the file and put the following in it:

```
{
  "compilerOptions": {
    "experimentalDecorators": true,
    "emitDecoratorMetadata": true,
    "module": "commonjs",
    "target": "es5",
    "sourceMap": true,
    "outDir": "output",
    "watch": true
  }
}
```

- Now to make the `tasks.json` file, open the prompt with `command + shift + p` and type:

```
> configure task runner
```

- Then put the following in the file and save the file:

```
{
  "version": "0.1.0",
  "command": "tsc",
  "showOutput": "silent",
  "isShellCommand": true,
  "problemMatcher": "$tsc"
}
```

- The last thing that we need to set up is the debugger, i.e. `launch.json` file. Right click on the `.vscode` folder in the file navigator and make a new file called `launch.json` and put in the following:

```
{
  "version": "0.1.0",
  "configurations": [
    {
```

```

    "name": "TS Debugger",
    "type": "node",
    "program": "main.ts",
    "stopOnEntry": false,
    "sourceMaps": true,
    "outDir": "output"
  }
]
}

```

- After you save the file, you should be able to see the debugger in the debugger dropdown options.
- Now, we are ready to make the `main.ts` file in the root of the project:

`main.ts`

```

1 | console.log('hello');

```

- Now you can start the task to watch the files and compile as you work. Open the prompt with `command + shift + p` and type:

`> run build tasks`

you can also use the `command + shift + b` keyboard shortcut instead. This will start the debugger and watch the files. After making a change to `main.ts`, you should be able to see the output in the `output` folder.

- Now that the build task is running, we can put a breakpoint anywhere in our typescript code. Lets add some more code to the main file and use the debugger:

```

1 | let a = 2;
2 | let b = 3;
3 | let c = 4;

```

- Then click on the margin of line two for example to add a breakpoint. Then open the debugger tab to run the debugger and you should see that the program will stop at the breakpoint and you can step over or into the line.
- To stop the task you can terminate it. Open the prompt and type:

```
> terminate running task
```