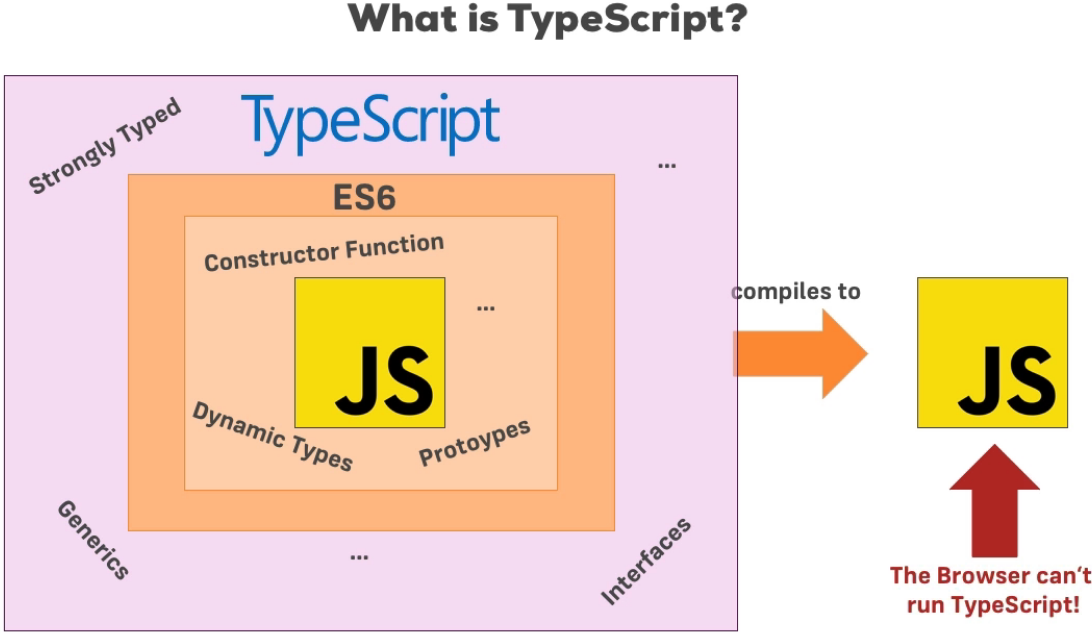
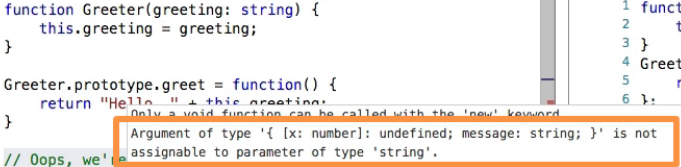
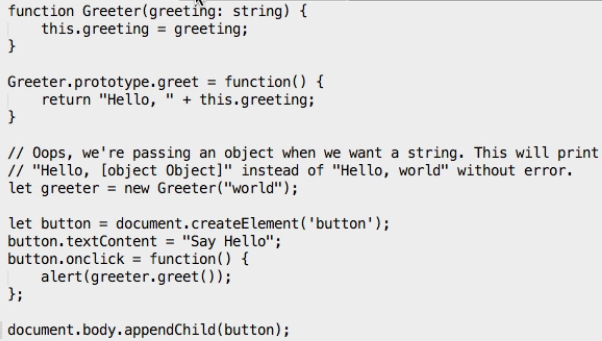
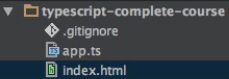
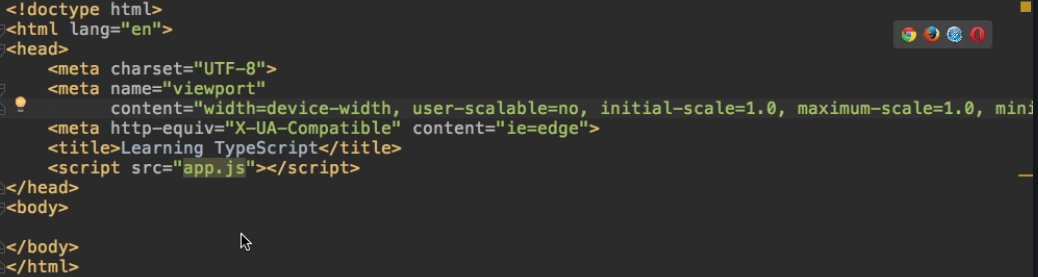
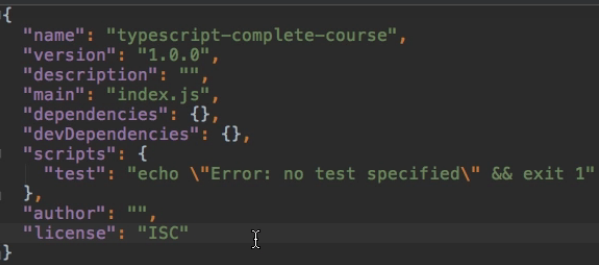
**Course Introduction**  
\* **Types**, **ES6**, **Classes**, **Compiler**, **Namespaces**, **Modules**.  
\* **Interfaces**, **Generics**, **Decorators**.  
\* **How to integrate TypeScript into your Webpack / Gulp workflow**.  
\* Build a brand new workflow using TypeScript only.  
\* How to use JavaScript Libraries.  
\* **How to use React with TypeScript - it’s possible and it’s awesome**.  
\* You can really use TypeScript in any project of your choice.

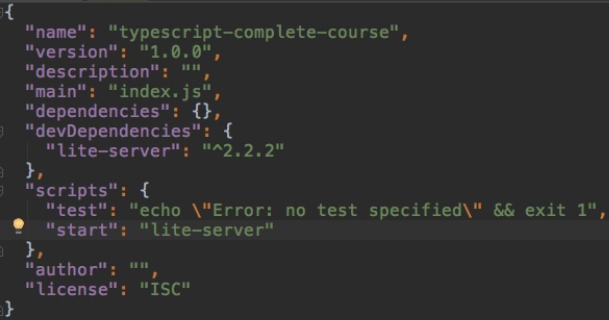
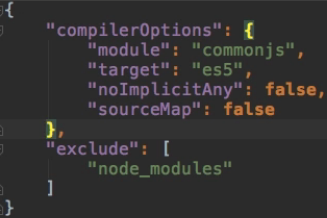
**What is TypeScript?**  
\* JavaScript is the scripting language you’re using in the browser to make your websites reactive, to run any code in the browser.  
\* JavaScript is continually getting extended.  
\* It offers a lot of features like Constructors, Functions, Prototypes and it uses Dynamic Types.  
=> **That’s ES5**.  
\* ES6 introduces new features to JavaScript - Class keyword and so on.  
\* The problem is that ES6 is not supported by all browsers.  
\* The other issue is that while it does enhance JavaScript and introduce new features, there are still a lot of things where you could say: that could be better - that is a bit better in other languages.  
\* And there are some features we might want to use in JavaScript which aren’t available.  
\* This is where TypeScript comes in.  
\* TypeScript is a wrapped, it compiles down to JavaScript in the end and introduces new features which we can use in our code and which will then run in the browser as the TypeScript compiler will make sure to find ways to compile all these great new features into ES5 code to then run in any browser.  
\* The most important feature is **Types**.  
\* **TypeScript is a strongly typed language**.  
=> You have to be specific which type a variable is.  
\* TypeScript is a wrapped around JavaScript so to say.  
\* **TypeScript doesn’t run in the browser, we need to compile it to JavaScript to run there**.  
\* So TypeScript is only an extra language we can use during development.  


**Why TypeScript and How to use it?**  
\* **The TypeScript Official Web Page PLAYGROUND is a great resource to really play around with TypeScript and actually a lot of the examples shown in this course may be recreated there.**  
<https://www.typescriptlang.org/>  
  
\* We mix our TypeScript code with JavaScript code, there is any degree of mixture possible and then we get the benefits from using TypeScript.

**Installing TypeScript**  
\* Open up a command line on your computer.  
\* You need to have npm - Node’s Package Manager installed.  
\* If you don’t have it, go to <https://nodejs.org/en> and download the latest version.  
\* Let’s now globally install TypeScript.  
**npm -g install typescript**  
=> **This will fetch the latest version and install it on your machine**.  
\* On Linux and Mac you may need to throw a sudo in front of it.

**Using TypeScript**  
\* **The file extension is .ts**  
\* I just create **index.html** and **scripts.ts**.  
   
\* Let’s use the Playground code.  
  
  
=> **We want to import JavaScript code which runs in the browser, not TypeScript**.  
=> **It will get compiled to JavaScript code and this .js file will get created by TypeScript**.  
\* **I’m making it a relative import by adding ./ at the beginning**.  
\* Let’s navigate to the folder which holds the script.ts file and run tsc - which is a command available once we install TypeScript, which runs the TypeScript Compiler and then passing the file we want to compile.  
**tsc**  
**tsc script.ts**  
  
\* **The script.js holds our compiled JavaScript code**.

**Setting Up the Course Workspace**\* For the rest of this course, I will have a specific folder structure which I will always keep and a little helper which makes the TypeScript development a little more convenient and makes these examples a little bit more enjoyable and it’s also required for some of these modules.  
=> I’ll install a very lightweight little server which will run automatically, a NodeJS server that is, which will host our index.html file.  
  
  
  
\* I want to put this project (folder) here under control of npm.  
\* The reason for that is that throughout this course I’m going to add a few packages to this project.  
**npm init**  
\* This **creates package.json** file.  
  
**npm install lite-server --save-dev**  
=> By adding --save-dev, I’m making sure that this gets added as a development-only dependency.  
\* **It hosts the index.html and will also automatically update it whenever we change our code**.

\* I can write a convenience script to start the server.  
  
**npm start**  
=> When we run `npm start`, it will run the server.  
**tsc --init**  
=> To not only put this folder under control of npm but **to also put it under control of TypeScript**.  
=> **This tells TypeScript: hey, this folder here is a TypeScript project, by running tsc, please compile all TypeScript files**.  
  
  
=> You can see this **holds some options for the TypeScript compiler**.  
**tsc**  
=> **We can now just open a new terminal and run tsc and this will search for all TypeScript files and compile them to JavaScript**.

**Asking Questions & Course Materials**

**Resources**  
TypeScript Official Web Page  
<https://www.typescriptlang.org/>