TypeScript lets you write JavaScript the way you really want to. TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. TypeScript is pure object oriented with classes, interfaces and statically typed like C# or Java. The popular JavaScript framework **Angular 2.0** is written in TypeScript. Mastering TypeScript can help programmers to write object-oriented programs and have them compiled to JavaScript, both on server side and client side.

Audience

Programmers coming from Object Oriented world will find it easy to use TypeScript. With the knowledge of TypeScript, they can build web applications much faster, as TypeScript has good tooling support.

Prerequisites

As a reader of this tutorial, you should have a good understanding of OOP concepts and basic JavaScript, to make the most of this tutorial.

Compile/Execute TypeScript Programs

For most of the examples given in this tutorial you will find **Try it** option, so just make use of it and enjoy your learning.

Try following example using **Try it** option available at the top right corner of the below sample code box:

var message:string = "Hello World"

console.log(message)

On compiling, it will generate following JavaScript code.

//Generated by typescript 1.8.10

var message = "Hello World";

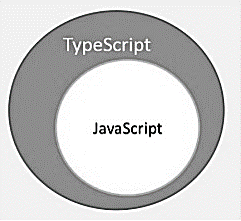
console.log(message);

JavaScript was introduced as a language for the client side. The development of Node.js has marked JavaScript as an emerging server-side technology too. However, as JavaScript code grows, it tends to get messier, making it difficult to maintain and reuse the code. Moreover, its failure to embrace the features of Object Orientation, strong type checking and compile-time error checks prevents JavaScript from succeeding at the enterprise level as a full-fledged server-side technology. **TypeScript** was presented to bridge this gap.

## What is TypeScript?

By definition, “TypeScript is JavaScript for application-scale development.”

TypeScript is a strongly typed, object oriented, compiled language. It was designed by **Anders Hejlsberg** (designer of C#) at Microsoft. TypeScript is both a language and a set of tools. TypeScript is a typed superset of JavaScript compiled to JavaScript. In other words, TypeScript is JavaScript plus some additional features.



## Features of TypeScript

**TypeScript is just JavaScript**. TypeScript starts with JavaScript and ends with JavaScript. Typescript adopts the basic building blocks of your program from JavaScript. Hence, you only need to know JavaScript to use TypeScript. All TypeScript code is converted into its JavaScript equivalent for the purpose of execution.

**TypeScript supports other JS libraries**. Compiled TypeScript can be consumed from any JavaScript code. TypeScript-generated JavaScript can reuse all of the existing JavaScript frameworks, tools, and libraries.

**JavaScript is TypeScript**. This means that any valid **.js** file can be renamed to **.ts** and compiled with other TypeScript files.

**TypeScript is portable**. TypeScript is portable across browsers, devices, and operating systems. It can run on any environment that JavaScript runs on. Unlike its counterparts, TypeScript doesn’t need a dedicated VM or a specific runtime environment to execute.

### TypeScript and ECMAScript

The ECMAScript specification is a standardized specification of a scripting language. There are six editions of ECMA-262 published. Version 6 of the standard is codenamed "Harmony". TypeScript is aligned with the ECMAScript6 specification.



TypeScript adopts its basic language features from the ECMAScript5 specification, i.e., the official specification for JavaScript. TypeScript language features like Modules and class-based orientation are in line with the EcmaScript 6 specification. Additionally, TypeScript also embraces features like generics and type annotations that aren’t a part of the EcmaScript6 specification.

## Why Use TypeScript?

TypeScript is superior to its other counterparts like CoffeeScript and Dart programming languages in a way that TypeScript is extended JavaScript. In contrast, languages like Dart, CoffeeScript are new languages in themselves and require language-specific execution environment.

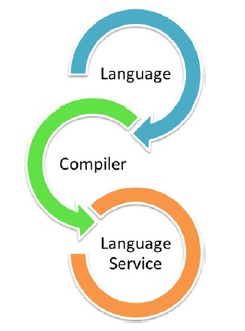
The benefits of TypeScript include −

* **Compilation** − JavaScript is an interpreted language. Hence, it needs to be run to test that it is valid. It means you write all the codes just to find no output, in case there is an error. Hence, you have to spend hours trying to find bugs in the code. The TypeScript transpiler provides the error-checking feature. TypeScript will compile the code and generate compilation errors, if it finds some sort of syntax errors. This helps to highlight errors before the script is run.
* **Strong Static Typing** − JavaScript is not strongly typed. TypeScript comes with an optional static typing and type inference system through the TLS (TypeScript Language Service). The type of a variable, declared with no type, may be inferred by the TLS based on its value.
* TypeScript **supports type definitions** for existing JavaScript libraries. TypeScript Definition file (with **.d.ts** extension) provides definition for external JavaScript libraries. Hence, TypeScript code can contain these libraries.
* TypeScript **supports Object Oriented Programming** concepts like classes, interfaces, inheritance, etc.

## Components of TypeScript

At its heart, TypeScript has the following three components −

* **Language** − It comprises of the syntax, keywords, and type annotations.
* **The TypeScript Compiler** − The TypeScript compiler (tsc) converts the instructions written in TypeScript to its JavaScript equivalent.
* **The TypeScript Language Service** − The "Language Service" exposes an additional layer around the core compiler pipeline that are editor-like applications. The language service supports the common set of a typical editor operations like statement completions, signature help, code formatting and outlining, colorization, etc.



### Declaration Files

When a TypeScript script gets compiled, there is an option to generate a **declaration file** (with the extension **.d.ts**) that functions as an interface to the components in the compiled JavaScript. The concept of declaration files is analogous to the concept of header files found in C/C++. The declaration files (files with **.d.ts** extension) provide intellisense for types, function calls, and variable support for JavaScript libraries like jQuery, MooTools, etc.

## Try it Option Online

We already have set up TypeScript programming online, so that you can execute all the available examples online at the same time when you are doing your theory work. This gives you confidence in what you are reading and to check the result with different options. Feel free to modify any example and execute it online.

Try the following example using our online compiler option available at [CodingGround](https://www.tutorialspoint.com/codingground.htm" \t "_blank)

var message:string = "Hello World"

console.log(message)

On compiling, it will generate following JavaScript code.

//Generated by typescript 1.8.10

var message = "Hello World";

console.log(message);

For most of the examples given in this tutorial, you will find a **Try it** option in our website code sections at the top right corner, that will take you to the online compiler. So just make use of it and enjoy your learning.

In this chapter, we will discuss how to install TypeScript on Windows platform. We will also explain how to install the Brackets IDE.

## TypeScript ─ Try it Option Online

You may test your scripts online by using The TypeScript at [www.typescriptlang.org/Playground](https://www.typescriptlang.org/play/). The online editor shows the corresponding JavaScript emitted by the compiler.



You may try the following example using **Playground**.

var num:number = 12

console.log(num)

On compiling , it will generate following JavaScript code.

//Generated by typescript 1.8.10

var num = 12;

console.log(num);

The output of the above program is given below −

12

## Local Environment Setup

Typescript is an Open Source technology. It can run on any browser, any host, and any OS. You will need the following tools to write and test a Typescript program −

### A Text Editor

The text editor helps you to write your source code. Examples of a few editors include Windows Notepad, Notepad++, Emacs, vim or vi, etc. Editors used may vary with Operating Systems.

The source files are typically named with the extension **.ts**

### The TypeScript Compiler

The TypeScript compiler is itself a **.ts** file compiled down to JavaScript (.js) file. The TSC (TypeScript Compiler) is a source-to-source compiler (transcompiler / transpiler).



The TSC generates a JavaScript version of the **.ts** file passed to it. In other words, the TSC produces an equivalent JavaScript source code from the Typescript file given as an input to it. This process is termed as transpilation.

However, the compiler rejects any raw JavaScript file passed to it. The compiler deals with only **.ts** or **.d.ts** files.

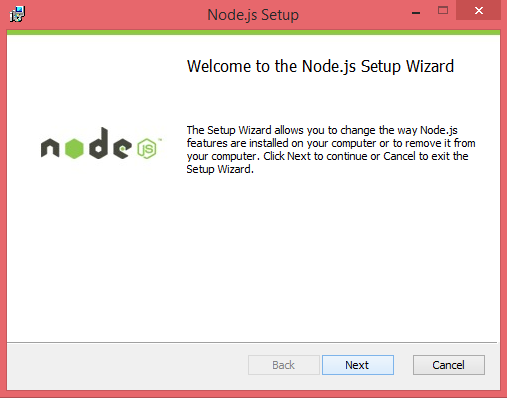
## Installing Node.js

Node.js is an open source, cross-platform runtime environment for server-side JavaScript. Node.js is required to run JavaScript without a browser support. It uses Google V8 JavaScript engine to execute code. You may download Node.js source code or a pre-built installer for your platform. Node is available here − [*https://nodejs.org/en/download*](https://nodejs.org/en/download/)

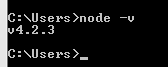
### Installation on Windows

Follow the steps given below to install Node.js in Windows environment.

**Step 1** − Download and run the .msi installer for Node.

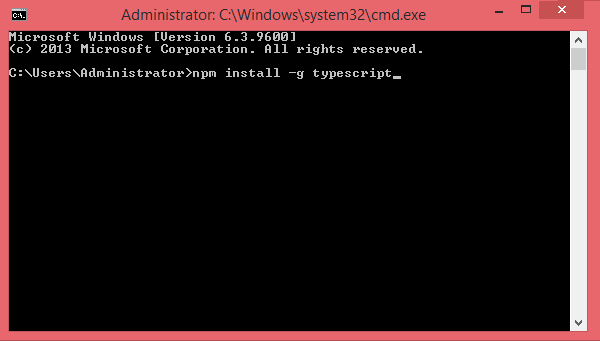


**Step 2** − To verify if the installation was successful, enter the command ***node –v*** in the terminal window.



**Step 3** − Type the following command in the terminal window to install TypeScript.

npm install -g typescript

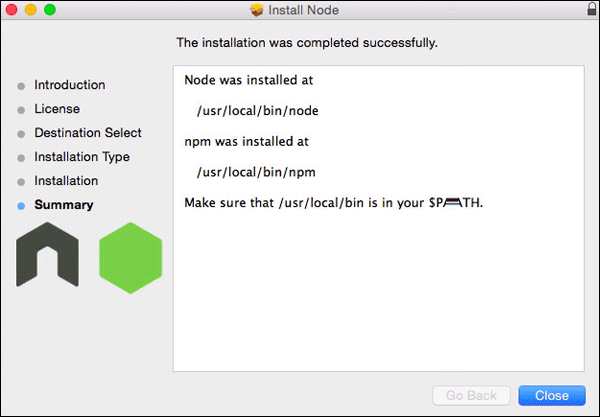


### Installation on Mac OS X

To install node.js on Mac OS X, you can download a pre-compiled binary package which makes a nice and easy installation. Head over to [http://nodejs.org/](https://nodejs.org/en/) and click the install button to download the latest package.



Install the package from the **.dmg** by following the install wizard which will install both node and **npm**. npm is Node Package Manager which facilitates installation of additional packages for node.js.



### Installation on Linux

You need to install a number of dependencies before you can install Node.js and NPM.

* **Ruby** and **GCC**. You’ll need Ruby 1.8.6 or newer and GCC 4.2 or newer.
* **Homebrew**. Homebrew is a package manager originally designed for Mac, but it’s been ported to Linux as Linuxbrew. You can learn more about Homebrew at <http://brew.sh/> and Linuxbrew at [http://brew.sh/linuxbrew](http://linuxbrew.sh/)

Once these dependencies are installed, you may install Node.js by using the following command on the terminal −

brew install node.

## IDE Support

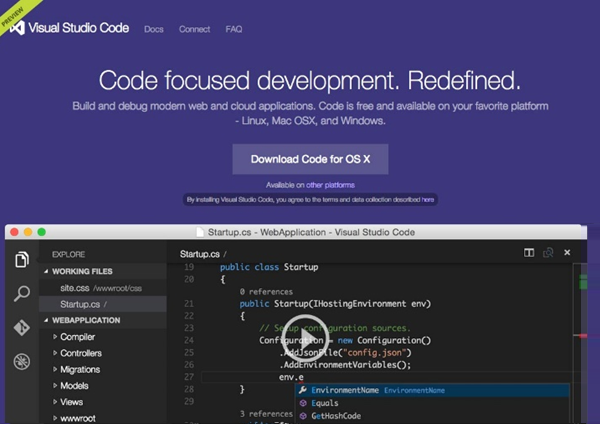
Typescript can be built on a plethora of development environments like Visual Studio, Sublime Text 2, WebStorm/PHPStorm, Eclipse, Brackets, etc. Visual Studio Code and Brackets IDEs are discussed here. The development environment used here is Visual Studio Code (Windows platform).

## Visual Studio Code

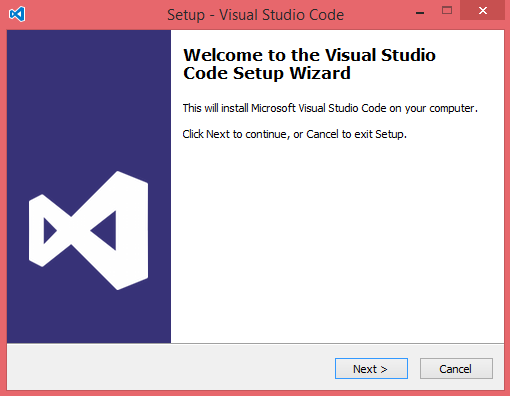
This is an open source IDE from Visual Studio. It is available for Mac OS X, Linux and Windows platforms. VScode is available at − [https://code.visualstudio.com/](https://code.visualstudio.com/?utm_expid=101350005-25.TcgI322oRoCwQD7KJ5t8zQ.0)

### Installation on Windows

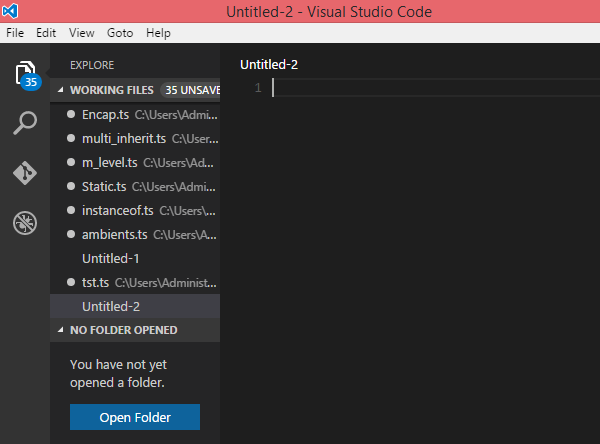
**Step 1** − [Download Visual Studio Code](https://code.visualstudio.com/docs?dv=win) for Windows.



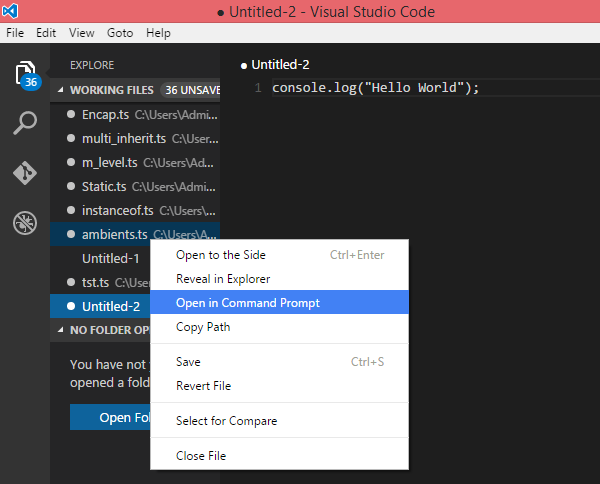
**Step 2** − Double-click on VSCodeSetup.exe Launch Setup Process to launch the setup process. This will only take a minute.



**Step 3** − A screenshot of the IDE is given below.



**Step 4** − You may directly traverse to the file’s path by right clicking on the file → open in command prompt. Similarly, the Reveal in Explorer option shows the file in the File Explorer.



### Installation on Mac OS X

Visual Studio Code’s Mac OS X specific installation guide can be found at

[https://code.visualstudio.com/Docs/editor/setup](https://code.visualstudio.com/docs/setup/setup-overview)

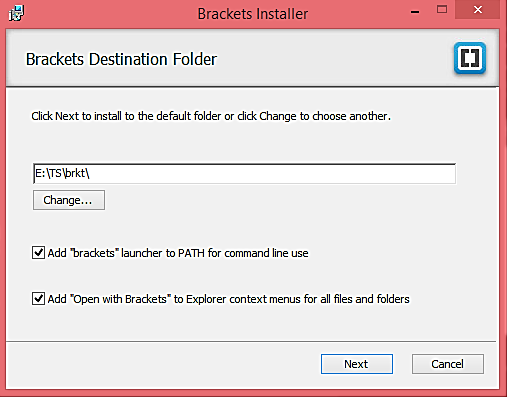
### Installation on Linux

Linux specific installation guide for Visual Studio Code can be found at

[https://code.visualstudio.com/Docs/editor/setup](https://code.visualstudio.com/docs/setup/setup-overview)

## Brackets

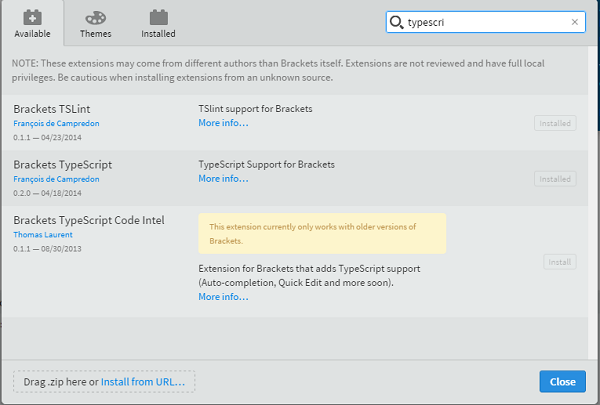
Brackets is a free open-source editor for web development, created by Adobe Systems. It is available for Linux, Windows and Mac OS X. Brackets is available at <http://brackets.io/>



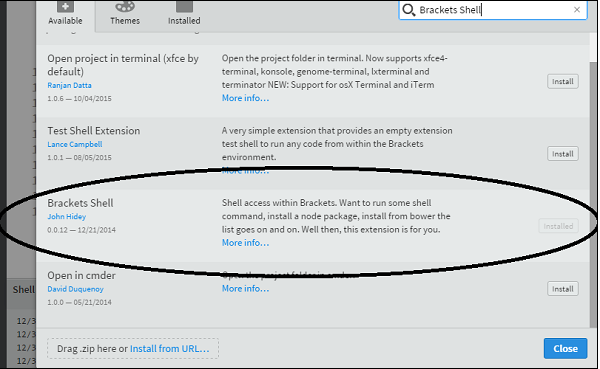
### TypeScript Extensions for Brackets

Brackets supports extensions for adding extra functionality via the Extension Manager. The following steps explain installing TypeScript extensions using the same.

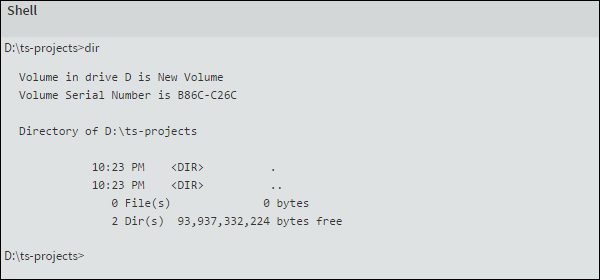
* Post installation, click on the extension manager icon Extension Manager on the right-hand side of the editor. Enter typescript in the search box.
* Install the Brackets TSLint and Brackets TypeScript plugins.



You can run DOS prompt / shell within Brackets itself by adding one more extension Brackets Shell.



Upon installation, you will find an icon of shell on the right-hand side of the editor Shell. Once you click on the icon, you will see the shell window as shown below −



**Note** − Typescript is also available as a plugin for Visual Studio 2012 and 2013 environments [(https://www.typescriptlang.org/#Download).VS](https://www.typescriptlang.org/#Download) 2015 and above includes Typescript plugin by default.

Now, you are all set!!!