TypeScript

Typescript is a typed superset of JavaScript that compiles to plain JavaScript.

Can run any browser, any host, Any OS, Open Source

Typescript Compilation

Type Script ----🡪 compiled ----🡪 JavaScript

JavaScript -> scripting language but typescript -> OOO language

Why Typescript instead of javascript?

1. More maintainable code
2. More message JavaScript code
3. Javascript has dynamic type system(var a=1,var a=”test”). But that cuases confusion
4. Migrating from server side to client side apps can be challenging. (java to javascript is tough)
5. There are alternatives for javascript like CofeeScript or Dart, or you can write pure javascript or follow all javascript patterns

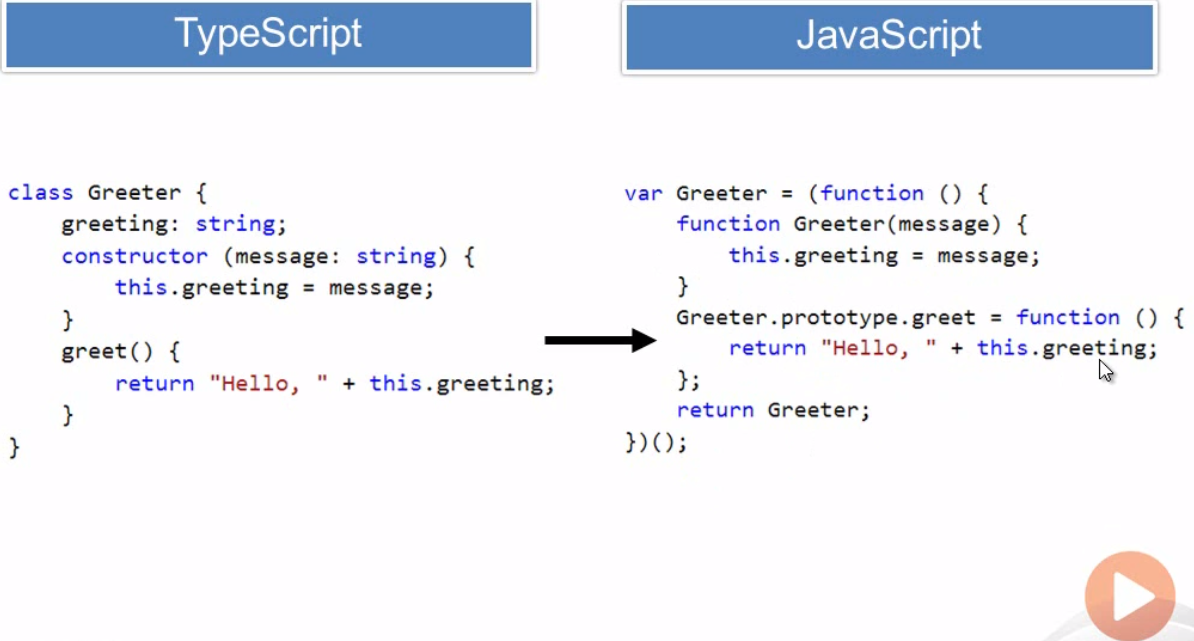
Typescript Features

1. Typescript will work in any browser
2. Work on any host.
3. Any OS, Open source, Good tool support
4. Typescript supports standard typescript code(typescript and javascript both )
5. Static typing (data typing)
6. Encapsulation through class modules(classes)
7. Support constructor, properties and functions
8. Supports Interfaces
9. => lamda functions support (anonymous functiosn)
10. Intellisense and syntax checking

Typescript Installation

1. Installation -> Npm install g typescript
2. To compile any typescript => tsc Customer.ts

Typescript vs Javascript



Typescript Keywords and Operators

Class – class

Constructor – initialize

Exports- exports a member from module

Imports – imports from module

Implements – implement interface

Extends – extends a class or interface

Public/private

Module/namespace – container for class and other code

… - rest parameter syntax

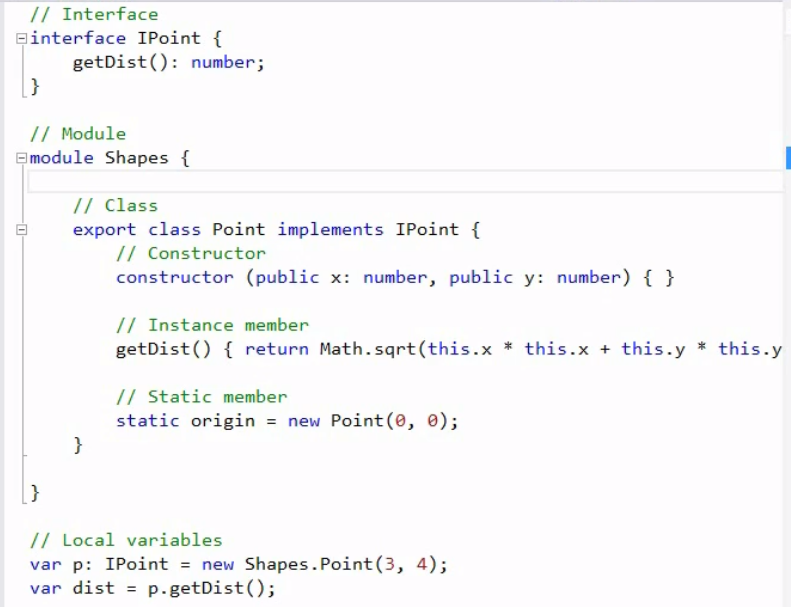
* - used with defintions and functions

<typename> - character used to cast/convert between types

: seperateor between variable and parameter and types

Code Hierarchy

Module/Namespace -> class -> fields, consturcot,r properties,functions



Basics of Typescript

1. Variables & datatypes
2. Inheritance: extends

class NewCustomer extends Customer {

1. Methods

validate(input:number): boolean {

alert(input +'i am in validate function');

return true;

}

}

4) get/set identifier

Private name:string

set CustomerName(ame:string){

this.name = ame;

}

get CustomerName():string{

return this.name;

}

This will work only ECMA 5 or higher

5) Exception handling

Throw in typescript

if (input >10) {

throw "exeption nae is required";

}

Catch in html

try{

var customernew = new NewCustomer();

customer.validate(33);

}

catch (ex ){

alert(ex);

}

6) interfaces

Class customer impelments Icustomer

7) class modules and reference them using export

export class Address {

Street1: string;

Street2: string;

}

import { Address} from './Address';

class Customer {

name: string;

public age: number;

address: Address;

All javascript code are valid inside typescript

Data Types, Variables and Functions

1. Type inference

Automatic type detection works

This should be in the module level outside the class

var name;

var name1:string;

var num ="hello";

var num1:string ="e";

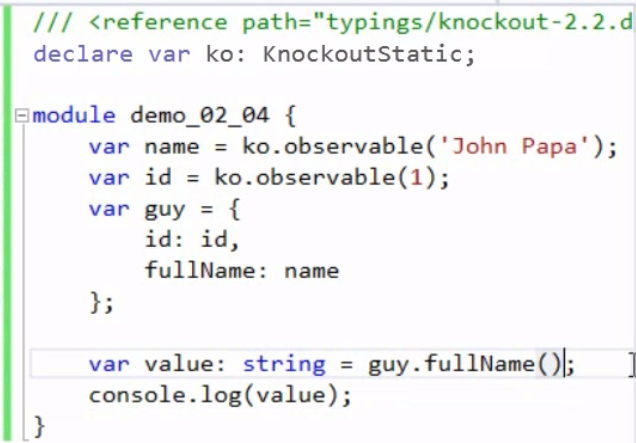
1. Statically typed language

If any variable is not declared or method is improperly used with different parameter or return type. Typescript will capture at compile time rather than like JavaScript will throw some weird error message at runtime.

1. Ambient decerlation

Using knockoutjs

https://github.com/DefinitelyTyped/DefinitelyTyped



1. Any and primitive types

Var data:any;

No static type checking on this “any”

Var name:string =”bala”;

Var age:Boolean =true

Var race:number = 2;

Var race1:number =2.2;

Var names: string[] = [‘hoh’,’d’]

Names[0]

Var name:number = null;

Var nam1:string = null;

Var happy:Boolean = null;

Var customer:{} = null;

If we didn’t declare any then undefined or you can assign It undefined

Var age;

Var age = undefined

When there is no datatype deinfed “any” will be inferred

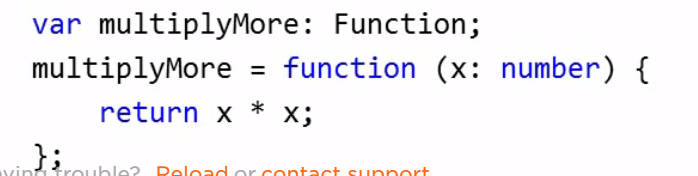
Var age;

Age =10

Objects

var name:object ={h:10,r:20}

var points = {x:10,y:20}



Var points ={};

Points ={x:2,y:2}

Var rectangle = {

H:10,

W:20,

Calcarean: function(){

Return h\*w

}

Example 1

module Cognizant{

export class Employee{

a: number = 10;

write() {

alert(this.a);

}

newobject = {

a: 1, b: 2, c: "test"

};

squre = function (x: number, y?: number) {

if (y != undefined) {

return x \* y

}

}

}

Example 2 – Optional functional parameter

//func representaton

name = function (x?: string) {

return x;

}

Example 3 – Arrow function expression

name1 = (x: string) => { return x };

Example 4 - no return void

name2 = function (x: string) {

console.log("hello",x);

}

name3 = (x: string) => void;

Example 4 – Object as function

sququre: (rect: { a: number, b: number }) => number;

function(rect) {

return rect.a + rect.b;

}

Objects and Classes

Class -> members, proeprties, constucor, function

export class Home{

private entry:string;

location:number;

constructor(entry:string,location:number){

this.entry = entry;

this.location = location;

}

get Entry():string{

return this.entry;

}

set Entry(entry:string){

this.entry = entry;

}

Enter(){

console.log("I am enter",this.entry);

}

Exit(){

console.log("I am exit",this.location);

}

}

Can have class inside class

export class Apartment{

home:Home;

constructor(home:Home){

this.home = home;

}

gethome():Home{

return this.home;

}

}

Thing before using get/set it will work with ECMA script 3. (yous should compile like

**tsc --target es5 home.ts**

casting types in Class/Obhect