Typescript:

Typescript is superset of Javascript: ES6:

Open source: Object oriented Language: class based:

Developed by Microsoft: but not directly: **Browser do not have the inbuild support TS**: inbuild:

abc.ts tsc abc.ts Compiler🡪abc.js :

TS is strictly typed and static typed language: JS: var a=100;

var a:number=100; Data types:

Function you must specify there return types:

Compiling at the same time variable there binding happens at compile time:

TS is not a new language:

Why TS:

1. Strict typed and static typed language:\
2. TS supports all the features of OOP: class, interface, generic, inheritance
3. Simple , fast and easy to learn:
4. Compiled: error checking: at the time of compilation we can see if any errors are there:
5. TS supports all the libraries which are there in JS and additionally it has it own.
6. 2015: ES-6: let , const
7. Save lot of development:
8. Initially 1st Oct 2012:
9. MS:

Code management is very easy

Supports all the feature ES-6

IDE support/VS, notepad++ , eclipse

Prerequisite required for TS:

1. VS code
2. Nodejs and npm:
3. Basic idea abt the JS:

First Program in TS:

Data types in TS: strictly and statically typed:

Types:

1. Static:
   1. Built in
      1. Number: floating: 100, 100.10:
      2. Void: DT for variable declaration: but for specifying return type of the function
      3. String: name, city… “”, ‘’
      4. Null: var a=null
      5. Boolean: var: boolean=true/false
      6. **any: =====var a:any=true**
   2. User defined DT:
2. Generic
3. Decorators:

**Static**: memory binding supposed to happen at compilation: JVM:

Number:

String: split()

* 1. User defined DT:
     1. Array in TS:
        1. var list:number[]=[10,20,30,40]//of same type
        2. var name:string[]=[“Deepak”, “Nikhil”….]
        3. var list:Array<Number>=[10,20,30,40]

Types of array in TS:

1. Single Dimensional Array: [10, 20, 30, 40…….100]
2. Multi dimensional array:

|  |  |  |
| --- | --- | --- |
| 10 list**[0][0]** | 20 list[0][1] | 30 list[0][2] |
| 40 list[1][0] | 50 | 60 |
| 70 | 80 | 90 |

let list:number[][]=[[10,20,30],[40,50,60]]

**Tuple** in TS:

let emp=[101,”Deepak”, 85000];// array in JS ;

Union: var a:number=100

Var b:string=”string”

let **a:**number | string | Boolean

a=100;

a=”ABC”;

a=true;

Enum in TS;

1. Numeric
2. String
3. Heterogeneous :

Functions:

function fname(arglist):return type(number, string, ….)

{

}

Functions can be named or unnamed :

Arrow function In TS: lambda expression:

=>

let add=()=>

Class concept in TS:

Class: all such instances are having similar kind of Attribute: id: name : designation:

Every class:

Fields/variables

Methods

Constructors;

class Employee{

fields;

methods;

constructor;

}

3 different ways to initialized our instance variables;

1. By reference variables
2. By methods
3. Constructor; Special kind of method: its main task to initialize variables:
   1. TS: constructor name is not same as that of your class name:
      1. constructor(){}
4. no return type: does not mean your constructor do not return any thing: it return an object of respective class:

var **e1**=new Employee();

Inheritance in TS:

Concept in which object of one can acquire the property of another :

Parent:Hair color Brown: car, flat… flat

We : hair color: eyes: ur property : flat:

Class Shape{

area():number{

x=a+b

}}

class Circle extends Shape{

area():number{//method

x=a/b;

}

}

Types: Single , Multi level, Multiple inheritance, Hier

Hybrid:

TS supports which inheritances:

Single:

Multi level,

Hie: