ECMA Script

<https://kangax.github.io/compat-table/es6/> : To check javascript version and feature changes.

Typescript (after ES6)

* Strongly typed: data needed to mention.
* Need to be transpiled (transformed + compiled).
* Transpiled can be done :

1. At browser 2) At server before shipping to browser(more recommended).

Node.js

* Should be there to install typescript.
* Npm(Node Package Manager) tool will come with this. > helps to download and manage javascript libraries/dependencies.

Installing typescript

* npm install -g typescript / npm i -g typescript > -g install globally.

Compile

* tsc <filename.ts>
* node <filename.js> to run ts program //node programming , say in node space we are running but not ideal way.

Variable

* var name: “niki”; name will be undefined till we assign value

For loop

* for-in > to get index
* for-of > to get value

Tuples

* heterogeneous collection of values
* eg: var tuple = [10,”Hi”];

Add element to array:

* push – at end of array
* unshift – at start

Delete :

* delete – will delete only value and it’ll be null then.
* splice – entire element
* pop
* shift

Destructing of array/tuple :

* get the value of elements by diff ways i.e a[1][2]
* eg: var a=[10,’hi’]

var [b,c] = a . o/p : b= 10,c = hi

Object/Array Destructuring

* var {city,name} = {“name”:”niki”, “city” : “pune”}

Rest Parameter

* user when no of params are not known.
* Place ellipsis … before the parameter.
* It’ll generate an array, work nd access like array.
* E.g : funct name(firstname, …othernames) : string {}

NOTE: Rest and optional cannot be used together.

Anonymous function

* Use > can be used as an argument/parameter for calling another function.

**Callback Function : read**

Lambda Function

* Also called Fat arrow function.
* 3 parts : parameters, fat arrow, statement.
* E.g : var sum = (x: number,y: number) => x+y;

Console.log(sum(5,7));

Function Overloading

Constructor Function

* Syntax: var obj = new Function(parameter\_list, statement\_list);
* Eg: var myFunct = new Function(“a”, “b”, “return a+b”);

OBJECT ORIENTED CONCEPTS

* Object
* Class
* Variable cab be declared as public, private, protected but not in JavaScript.
* Typescript defines constructor using keyword ‘constructor’.
* ‘this’ object

Variable Scope

* Global > declared outside programming constructs. Can be accessed from anywhere within code.
* Class > fields or class variables declared within class but outside methods.
* Local > are declared within constructs like : methods, loops etc.

NOTE : Default constructor is not supported in Typescript.

Static field and method

Data Hiding

Inheritance

* Syntax: class chil\_cls extends parent\_cls {

//code

}

* Types supported : > single > multi-level

Interface

* Syntax: interface interface\_name{ scope;}
* To achieve multiple inheritance.
* Use implements keyword > class cls\_name implements interface\_name {}

Typescript Objects

* Set of key-value pairs.
* Value can be scalar, function or array.
* We can pass objects as parameter.

Module

* Specify and load dependencies between diff js files.
* How do we manage JS dependency ? > using <script> tag.
* In case of external module, its done using import , export and the file itself becomes module.

**ANGULAR 8**

Angular App’s Building Blocks

* Modules
* Components
* Directives – DOM manipulation
* Services – Encapsulation UI logic
* Routers – Handles navigation requests.

NOTE: ng serve -o > -o for build nd open the app.

Project Folder Structure

* E2e > end to end testing ,protractor is used.
* Command > npm run e2e.
* All dependencies are placed in node\_modules folder.
* Npm install command will check package.json and download required dependencies.
* Package.json is application related configuration.
* ~ : allow any version to be downloaded
* ^ : allow minor changes.
* Decorator provides metadata.
* Main.ts tells which first module to executed. Bydefault it is app.module.ts
* Component is nothing but a view or screen.
* BrowserModule: gives angular support
* AppROutingModule: for navigation support
* These in-built library are called as barrels.

View/Screen of App comes from component (app.component.ts)

* @Components : contains metadata
* Templateurl: which UI is attached. i.e html file. (not mandatory)
* Template option can be used for internal template if html is small.
* styleUrls : to specify from where to take styling.
* Selector : whenever u want to refer html we’ll use selector i.e app-root.
* Whenever u want to refer class , we import in module.ts.

Assets

* Can be mp3, jpeg, styling files etc placed over here needed for app.

FavIcon – read on google

Angular.json

* Related to angular-cli.json configuration.

Tsconfig.json

* Related to transpilation.
* Transformation of TS to JS.