Type script:

TS is a superset of JS with ES6 features

Angular 2 is not angular *js* 2 but Angular TS 2

Transpilation - types

Type support- other than std types, any, custom types

Helps debugging - one of major drawback of js - as types are determined before running - is it editor based - where is <??>[transpiler ?] - will TS be compiled [as opp to js-interpreted]

????????

Tooling support: like auto complete,peek[like f3 in eclipse, ctrl shift g etc]

ECMA INternational

ECMA Script - ES

ES 2015-ES6

New feautures:  
OOP features - Classes, arrow functions, Constants and block scoped variables, MODULES

<https://kangax.github.io/compat-table/es6/>

Compatibility tables - desktop/web/mobile etc

<http://www.ecma-international.org/ecma-262/6.0/>

<http://stackoverflow.com/questions/12178429/whats-the-difference-between-using-link-and-script-tag-to-reference-javascript>

* Also script is a container tag - so <script options.. /> would not work,instead <script options.. ></script> will work

Babel[is one of available transpiler] ?? es6 to es5

Constants:

Const keyword to declare constants [like final in java]

Blockscoped:

Var [ES5] is global scoped variable,

Let [ES6] is block scoped variable.

Functions:

Anonymous fn - without name

IIFE - immediate fn execn

Fn without name with inside braces with empty braces at end- () will invoke the fn defn immediately

Arrow functions[new in ES6]:

Without using name[keyword] function, return

Fat arrow[=>]: avoid fn,return

When multiple stmts in arrow fn defn - use curly braces {}

Transpiler - converts es6 to es5

[Babel is 1 popular transpiler]

**Other is type script**

[www.typescriptlang.org](http://www.typescriptlang.org)

This type script is supposed to be installed in the machine for it to work.

* Developed by MS[Microsoft]
* supports ES6, ES5.
* Strongly typed lang [determine errors at code writing time]
* class /object based
* Use .ts extension
* Transpiler[type script engine] converts .ts to .js file

Install typescript globally to be available for other projects too : npm install -g typescript

Tsc <ts\_file>

This command creates ES5 js file

Tsc -w <ts\_file>

This -w[watch] option watches and updates js file automatically

This creates a loop/batch which can be terminated

Better management through - tsc options through tsconfig.json configuration

After creating tsconfig.json in prj root,

use command tsc or tsc -w in prj root from cmd line

@@ In Visual studio 2015, we have gui support for ts commands @@

We can use options of ts as per need to set the [comp]transpilation process

<https://www.typescriptlang.org/docs/handbook/tsconfig-json.html>

**Strong typing** is only available in TS not in ES6

Var a:<data\_type> [number, boolean, object, any]

Automatic[Implicit] type inference:

Use : for explicit type inference-

Once implicit data type is assigned/inferred, we cannot change the type of var - in Type Script

Data types:

Primitive [Number, boolean,string]

Special types

Arrays

Functions

object

Special: any, null, undefined, void

Arrays: Type assertions[generics] - Array<string> or string[]

Functions:

Function type is Function [with capital F]

Optional params [?] eg: fn(a,b?, c)

Default parameters [=] eg: fn(a, b, c=0)

Rest parameters: like var args in java

* Each fn can only have 1 rest param
* notation/syntax - prefix with … eg. fn(a, ...b) [here b is an array]
  + Eg2: function getSum(...numbers:number[]):number{//}

Function overloading:define methods at the beginning ?

*// arrow types always use anony fns - while method overloading anony fn can’t be used to provide defn*

Objects:

{} vs constructor

No concept of Class in ES5

This = prototype

Custom types:

* Interface
* Class

Q: What do variables starting with \_ or \_\_ mean in JS ?

A:

Npm:

package .json is not automatically created- create pakcage.json and mention dependencies list and then npm install will install all the dependencies - else use npm install <pacakge\_name> to install manually

Use npm install <pacakge\_name> **--save** [to update the package.json file]

Day 2:

<http://dmitrysoshnikov.com/ecmascript/javascript-the-core/>

Remember: when you declare a variable in JavaScript (using "var"), that variable declaration is "hoisted" to the top of the current scope—meaning the top of the current function or the top of the script if the variable isn't in a function.

Hoisting can cause unexpected behavior, so a good way to keep things clear is to always declare your variables at the top of the scope.

Interview cake

Interface impl examples

Modules:

Export / import

Acc. to es6, each file is a module

When we transpile TS code to JS, no compile issues but we would have runtime issues,

THis is BECAUSE, js does not have module loaders by default

SO we need libraries like CommonJS, RequireJS or SystemJS[SystemJS is recommended by Angular]

Note: angular 1 had its own module loader ?

SystemJS:  
System.import(package/script file) - command to import based on System.config()

System.config() - options

* Packages
* Map
  + Key in the above two eg.app

Working with other JS libraries:

When we try to use other libraries like jQuery/backbone/knockout which are in JS, we need to have some kind of marshalling for TS to understand JS libraries.

For this we need TS definition files

* Typically defined with prefix.d - .d.ts

→> THese are called **Typings**

Decorators:

Used to describe / give more info about a class or method on which this decorator would be used.

Decorators add metadata, are prefixed with @

Web app develop

Diff in modern web apps - ajax

Due to ajax, there is also data in UI[view]

So we need MVC on client side [frame works] also

Eg. backbone,knockout, angular

Single page applications: SPA

ANGULAR 2:

For all types of apps: web apps, desktop, mobile apps[Hybrid]  
All single page apps

Angular 2 uses es6 features which are not supported by all[most] browsers yet,

So we use additioanl dependecies like : transpilers, loaders, polyfills, shims

Core-js shim: tries to provide es6 support to browsers

, zone.js: A Zone is an execution context that persists across async tasks. You can think of it as [thread-local storage](http://en.wikipedia.org/wiki/Thread-local_storage) for JavaScript VMs.

, rxjs:

, systemjs:

, reflect-metadata:

Again for all of these js libraries, we need typescript definitions

Eg. typings for core-js, we need to: npm install **@types/core-js** --save

Min basic libs:

@angular/core, @a/common, compiler, platform-browser, platform-browser-dynamic

Ts typings config:

moduleResoluition

emitDecoratorData,

experimentalDecorators

Structure of angular 2 apps:

Constructing ang 2 components:

* Import external modules
* Add meta data [how to show this component, how to config etc]
* Create class

Convention while creating a component

* app**.component.js**

<http://dmitrysoshnikov.com/ecmascript/javascript-the-core/>

Dividing the component into modules, services, components, models - modularization

MVC - separation of concerns

Built in directives:

Structural directives - help add remove or replace parts of DOM

Eg. \*ngFor

Day 3:

ngIf directive

Passign values between components

nested-parent/child

Sending parent to child - through - INPUT

* @Input decorator

From child to parent - through - OUTPUT - Through **events**

* @Output() decorator
* [child has to] Raise an event
* EventEmitter is used to raise an event
* Emit method in child, onNotify in parent
* In html use (notify)=”method(**$event**)”
  + Here $event is a keyword
  + Notify is also a variable name, it is the variable we define in child
  + So the same event (xyz) should be used in parent also

**Event handlers -** (event\_name) : eg:(click)=”method\_call”

Built in security to avoid xss

**Sanitization** allows provision to make untrusted dom elements as trusted

DomSanitizer byPassSecurityTrustResourceUrl()

Data Binding:

Two way data binding:

When changed in view, model is updated and vice versa

Angular supports both 1 way and 2 way bindings

[..] - used for property binding [1 way]

(..) - used for event binding [1 way]

[(..)] - both prop and event bind [2 way]

Working with External models:

FillText.com

Angular HTTP service:

Observable pattern - RxJS

@angular/http

Server communication:

Creating REST api:

Using **Deployd** [npm package] npm install -g deployd

cmd> dpd create <prj-name>;

Observable is a stream of events.

**SPA:**

SINGLE PAGE APPLICATIONS  
Single html page with some of its contents changing per need/request.

Use routing to move from view in one component to view[partials] in other component

Use layout component - whcih would be common for all components

Before deifinign routes in app.module[module loader],

Error: no provider found for <each> component

Cannot find primary outlet to load 'HomeComponent'

Error: Cannot find primary outlet to load 'HomeComponent'

at getOutlet

For this add <router-list><> int he base coponent at end of router links

ROuteMopdule - route list, useHash property[to reload page]

Unique compoent styules

Styles key in component defn

Css for local - not for all other components

Use style or styleUrl for path of css

Format or transform data - using pipes

Eg. currency

Pipe chaining- to left to Right

Custom pipe

@Pipe

Should imnplement PipeTransform interface

Should add impl for transform method in PipeTransform interface

**Unit testing:**

Jasmine, karma, protractor

Karma is a test runner NOT A test framework  
Karma built for angular - subset of testacular

Karma uses karma.conf.js

**ANgular 2 prj setup:**

1. Manual setup
2. Use quick start files from angular.io
3. ANgular CLI : pref approach
   1. Install using npm i **-g** angular-cli
   2. Once angular-cli is installed, ng command works in cmd>
   3. Test >ng
   4. <https://cli.angular.io/>
   5. Use the link for reference

Hola Amigo

**ES6 OO Concepts:**