

# Angular Introduction



# Agenda

- What is a SPA?
- Angular.JS Overview and Version Comparison
- Node.js, npm, WebPack, Yarn
- Angular Command Line Interface – Angular CLI
- Bootstrapping Angular / Project Configuration using Node.js

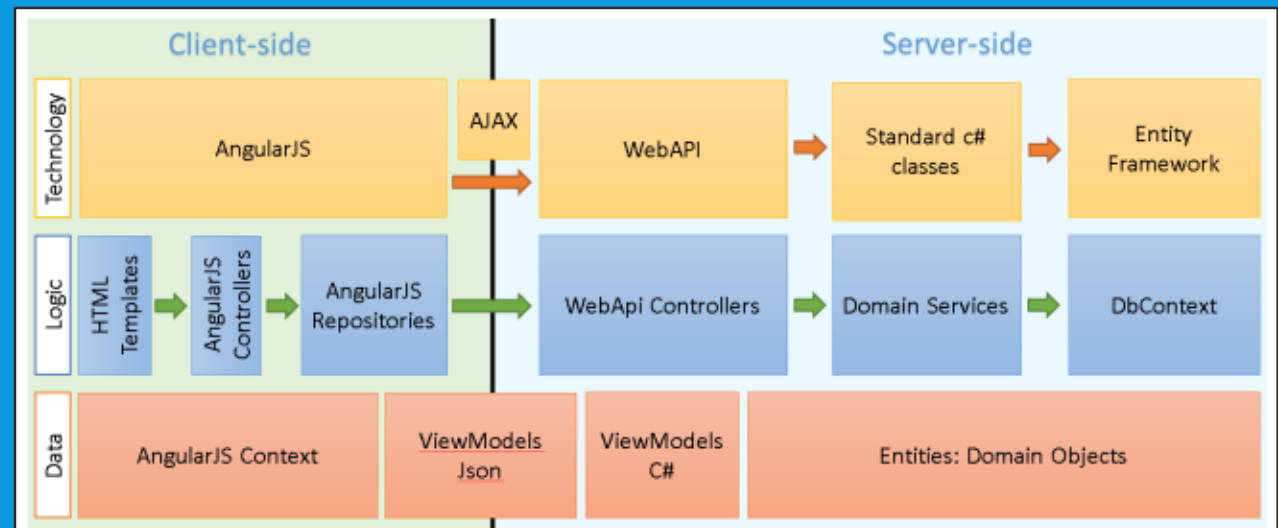
# What is a SPA

# What is a Single Page Application (SPA)

- A single-page application (SPA) is a web application that fits on a single page with the goal of providing a more fluent user experience similar to a desktop application
- In a SPA, either all necessary assets like HTML, JavaScript, and CSS –
  - is retrieved with a single page load, or
  - resources are dynamically loaded and added to the page, usually in response to user actions
- SPA's are often implemented using JavaScript Frameworks that use async XMLHttpRequests like
  - Ember.js / Knockout.js
  - Angular JS
  - React

# SPA Architecture

- SPA's can be implemented using a combination of client- and server side technologies
- Microsoft server side technologies often used together with SPA's are
  - WebAPI
  - ASP MVC
  - Entity Framework (or Linq to SQL)



# Comparing SPA / Multi Page Application (MPA)

- SPA advantages over MPA:
  - Faster page loading times
  - Improved user experience because the data is loading in the background from server
  - No need to write the code to render pages on the server
  - Decoupling of front-end and back-end development
  - Simplified mobile development; you can reuse the same backend for web application and native mobile application
- SPA disadvantages to MPA:
  - Heavy client frameworks which are required to be loaded to the client
  - UI code is not compiled, so it's harder to debug and it's exposed to potential malicious user
  - SEO (search engine optimization) implications; since your pages are built in the browser, the search engine crawler will see a different version of the page than that of your users

# Angular Introduction

# What is ANGULAR?

- Defines numerous ways to organize web application at client side.
- Enhances HTML by attaching directives, custom tags, attributes, expressions, templates within HTML.
- Encourage TDD & Client Side MVC/MVVM design pattern
- Good for Single Page Apps (SPA)
- Current version 5 released Nov 01 2017
- Mainly maintained by Google

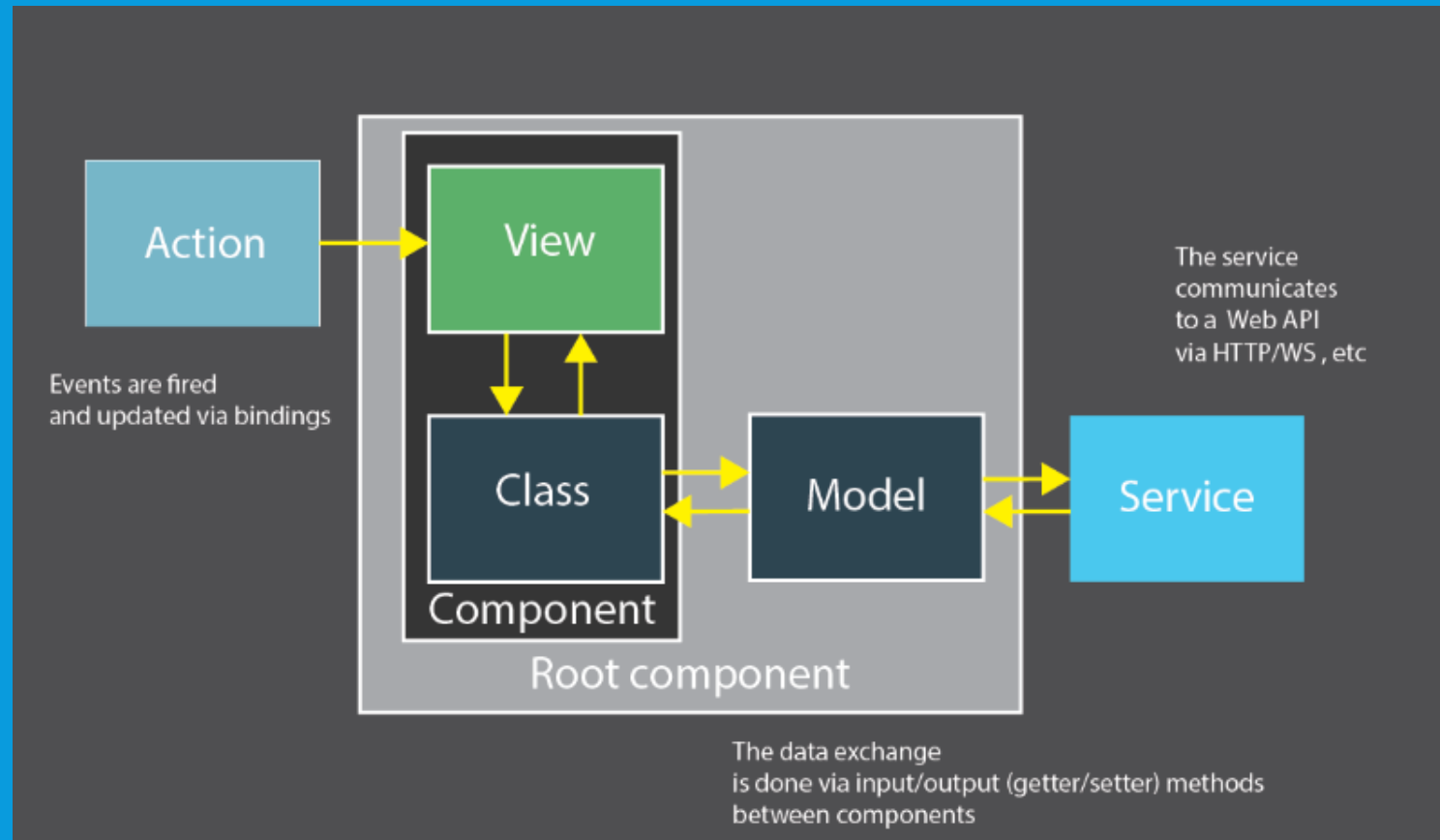




# Versions

- Angular JS (1.0)
  - Implemented mostly in pure JavaScript
- Angular (2+)
  - Complete Rewrite
  - Drastic change between 1.x and 2.x
  - Mostly Implemented in TypeScript
  - Current Version 5.x

# Components - Angulars MVC



# Routing

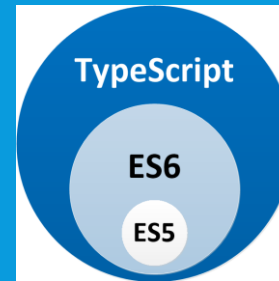
- Routing is the process to switch from one view to another
- Routing is achieved using Angular Router
- Routing is configured in app.module.ts
- Router Links are used for navigation

```
<nav>
<a routerLink="/" routerLinkActive="active">Home</a>
<a routerLink="/vouchers" routerLinkActive="active">Vouchers</a>
<a routerLink="/accounts" routerLinkActive="active">Accounts</a>
</nav>
<div>
<router-outlet></router-outlet>
</div>
```

```
const appRoutes: Routes = [
  { path: '',
    component: HomeComponent,
    children: [
      { path: 'inline', component: InlineStyleComponent },
      { path: 'stylebinding', component: StyleBindingComponent }
    ]
  },
  { path: 'vouchers',
    component: VouchersComponent
  },
  { path: 'vouchers/:id',
    component: VoucherComponent
  },
  { path: 'accounts',
    component: AccountsComponent,
    data: { title: 'Accounts' }
  }
];
```

# Angular Technology Stack

- Runtime / Package Management: Node.js
- Language: TypeScript (ES 6, Dart)
- Templating / Dependencies:  
Angular CLI, Yeoman, NPM, Yarn
- Bundling: Webpack



webpack  
MODULE BUNDLER



# What is Node.js

- Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications
- Uses an event-driven, non-blocking I/O model that makes it lightweight and efficient
- Used to build:
  - I/O bound Applications
  - Data Streaming Applications
  - JSON APIs based Applications
  - Single Page Applications
- Doku @ <https://nodejs.org/en/>



# Angular CLI

- Command Line Interface for managing Angular Applications
- Install using `npm install -g @angular/cli`
- Basic commands
  - Create new projects:
    - `ng new`
  - Create new components, services, ...
    - `ng generate component` – shortcut: `ng g c`
  - Serving
    - `ng serve`
- Documentation @ <https://cli.angular.io/>



Scaffold	Usage
Component	<code>ng g component my-new-component</code>
Directive	<code>ng g directive my-new-directive</code>
Pipe	<code>ng g pipe my-new-pipe</code>
Service	<code>ng g service my-new-service</code>
Class	<code>ng g class my-new-class</code>
Guard	<code>ng g guard my-new-guard</code>
Interface	<code>ng g interface my-new-interface</code>
Enum	<code>ng g enum my-new-enum</code>
Module	<code>ng g module my-module</code>

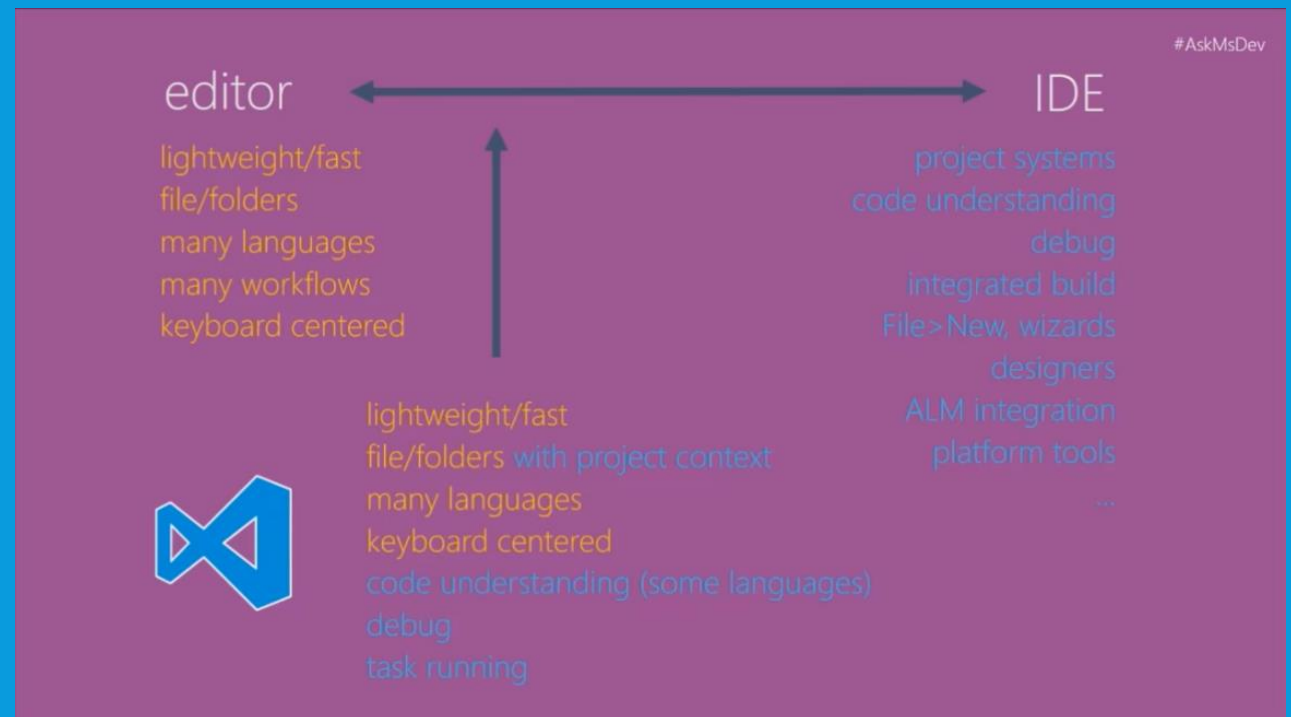
# Common Editors

- Any Editor that has integrated Support for Node.js
- Editor is many times result of Plugins available
  - Visual Studio Code
  - Atom, Sublime
  - WebStorm
  - Visual Studio Professional



# Visual Studio Code

- Free, Open Source, lightweight cross platform editor built on top up GitHubs Electron platform
- Optimized to build HTML, JS, TS based applications
- Out-of-box integration of GitHub & Node.js
- Get from <https://code.visualstudio.com/>
- Source @ <https://github.com/microsoft/vscode>
- 2nd Choice Editor would be Webstorm  
<https://www.jetbrains.com/webstorm>





# VS Code Shortcuts & Settings

- Ctrl + P Quick open
- Ctrl + Shift + F Find in Files
- Ctrl + K, S Save All
- Ctrl + ö Open Terminal Window
- Shift + # Toggle Comment
- Alt + Shift + A Toggle Block Comment

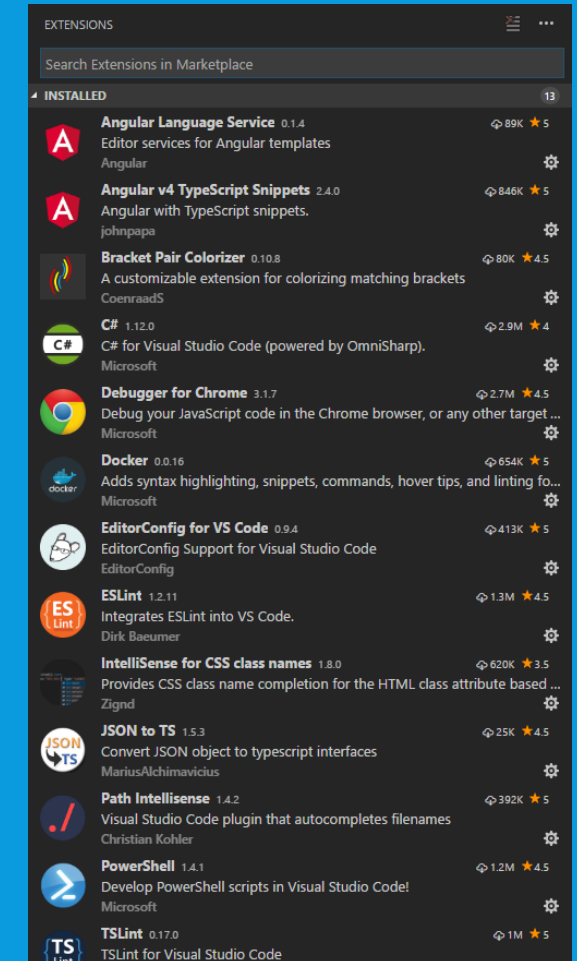
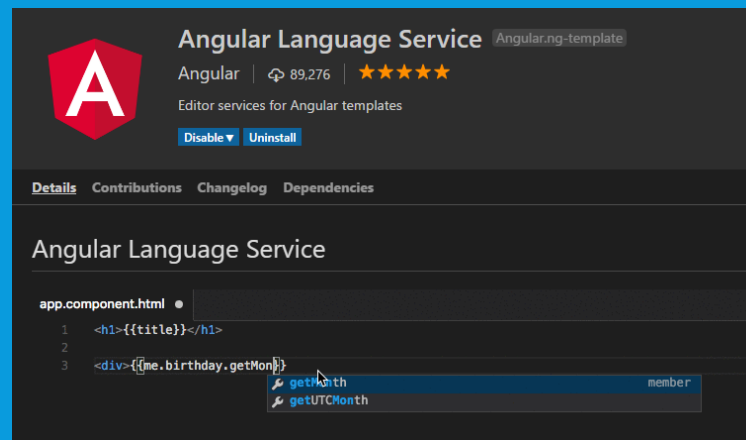
## Useful Workspace Settings

```
"editor.wordWrap": "on"  
"workbench.editor.enablePreview": false
```

- Complete VS Code Shortcute Guide:
- <https://github.com/Microsoft/vscode-tips-and-tricks>

# Usefull VS Code Extensions

- Extensions make VS Code development easier
  - Angular v5 Snippets
  - Angular Language Service
  - Debugger for Chrome
  - IntelliSense for CSS class names
  - Bracket Pair Colorizer
  - Path Intellisense
  - JSON 2 TS



# Extension Management

- List
  - `code --list-extensions`
- Install (VSIX install possible)
  - `code --install-extension NAME` ie: `Angular.ng-template`
- Uninstall
  - `code --uninstall-extension NAME` ie: `ms-vscode.csharp`

```
Angular.ng-template
christian-kohler.path-intellisense
clinyong.vscode-css-modules
CoenraadS.bracket-pair-colorizer
DougFinke.vscode-PSSstackoverflow
eg2.vscode-npm-script
formulahendry.auto-close-tag
johnpapa.Angular2
KnisterPeter.vscode-github
MariusAlchimavicius.json-to-ts
Mikael.Angular-BeastCode
ms-vscode.csharp
ms-vscode.PowerShell
msjsdiag.debugger-for-chrome
rafaelsalguero.csharp2ts
rbbittypescript-hero
xabikos.JavaScriptSnippets
```

# Node.js, Webpack & Gulp

# What is Node.js

- Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications
- Uses an event-driven, non-blocking I/O model that makes it lightweight and efficient
- Used to build:
  - I/O bound, Data Streaming Applications
  - Data Streaming Applications
  - JSON APIs based Applications
  - Single Page Applications
- Current version 8.0
- Doku @ <https://nodejs.org/en/>



# package.json

- package.json is the configuration file for node.js
- packages are saved to node\_modules – name cannot be changed
- npm install xxx [-g] -save | npm install xxx [-g] -save-dev

```
1  {
2    "version": "1.0.0",
3    "name": "asp.net",
4    "private": true,
5    "dependencies": {
6      "gulp-sourcemaps": "^2.4.1",
7      "lodash": "^4.17.4",
8      "webpack": "^2.6.1",
9      "babel-core": "^6.2.1",
10     "babel-loader": "^6.2.0",
11     "babel-preset-es2015": "^6.1.18"
12   },
13   "devDependencies": {
14     "babel-preset-es2015": "^6.24.1",
15     "gulp": "3.9.1",
16     "gulp-babel": "^6.1.2",
17     "gulp-concat": "2.6.0",
18     "gulp-cssmin": "0.1.7",
19     "gulp-sass": "^3.1.0",
20     "gulp-sourcemaps": "^2.6.0",
21     "gulp-uglify": "1.5.3"
22   }
23 }
```

# Node.js Basic Setup

- Install Node.js from <https://nodejs.org/>
- `npm install -g xxx -->` installs globally
- Go to cmd of your project
- Get Angular Seeder Project or create using Angular CLI: `ng new xxx`
- `npm install -->` installs all dependencies listed in `project.json`
- `npm install -g typescript`

# npm – Node Packet Manager



# npm

- npm makes it easy for JavaScript developers to share the code
- Essentially it acts as the package manager for node.js using the “install” cmd
- Lots of other cli commands
  - version
  - build
  - bundle
  - start
- Documented @ <https://docs.npmjs.com/cli/install>

# npmjs.com

- npmjs.com is a centralized repository for packages
- Allows to have private packages called @scope (@microsoft/xx, @angular/xx)
- Used in vendor packages like Angular, SharePoint Framework (SPFx)

```
"dependencies": {  
  "@microsoft/sp-client-base": "~1.0.0",  
  "@microsoft/sp-core-library": "~1.0.0",  
  "@microsoft/sp-webpart-base": "~1.0.0",  
  "@types/webpack-env": ">=1.12.1 <1.14.0"  
},
```

```
"dependencies": {  
  "@angular/common": "2.1.0",  
  "@angular/compiler": "2.1.0",  
  "@angular/core": "2.1.0",  
  "@angular/forms": "2.1.0",  
  "@angular/http": "2.1.0",  
  "@angular/platform-browser": "2.1.0",  
  "@angular/platform-browser-dynamic": "2.1.0",
```

# npm scripts

- npm allows scripting to automate tasks – eg. Starting dotnet cli
- Scripted cmds may be stated
  - Inline
  - External file

```
1 {  
2   "name": "first-angular",  
3   "version": "0.0.0",  
4   "license": "MIT",  
5   "scripts": {  
6     "ng": "ng",  
7     "start": "ng serve",  
8     "build": "ng build",  
9     "test": "ng test",  
10    "lint": "ng lint",  
11    "e2e": "ng e2e"  
12  },
```

# Semantic versioning - Semver

- Major release, increment the first number, e.g. **2**.0.0
- Minor release, increment the middle number, e.g. 1.**1**.0
- Patch release, increment the last number, e.g. 1.0.**1**

- >1.2.3 means greater than a specific version.
- <1.2.3 means less than a specific version.
- 1.2.3 - 2.3.4 means  $\geq 1.2.3 \leq 2.3.4$ .
- ~1.2.3 means  $\geq 1.2.3 < 1.3.0$ .
- ~1.2 means  $\geq 1.2.0 < 2.0.0$ .
- ~1 means  $\geq 1.0.0 < 2.0.0$ .
- 1.2.x means  $\geq 1.2.0 < 1.3.0$ .
- 1.x means  $\geq 1.0.0 < 2.0.0$ .

# Yarn

# Yarn

- Replacement for npm that Speeds up Node based dev by downloading and caching packages more efficient than npm
- Installation: `npm install -g yarn`
- `Npm install -> yarn`
- `Npm install xxx --save -> yarn add xxx --save`
- Yarn can be set as global Package Manager for Angular
  - `ng set --global packageManager=yarn`
- Documentation @ <https://yarnpkg.com/en/docs>



# Managing Yarn Cache

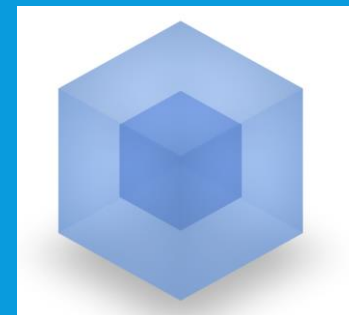
- Yarn stores every package in a global cache in your user directory on the file system.
- Speeds up creation / loading of Angular projects
- print out every cached package
  - `yarn cache ls`
- print out the path where yarn's global cache is currently stored
  - `yarn cache dir`
- will clear the global cache
  - `yarn cache clean`
- Set cache-folder – will be created if it does not exist
  - `yarn config set cache-folder <path>`

# Understanding Webpack



# Webpack

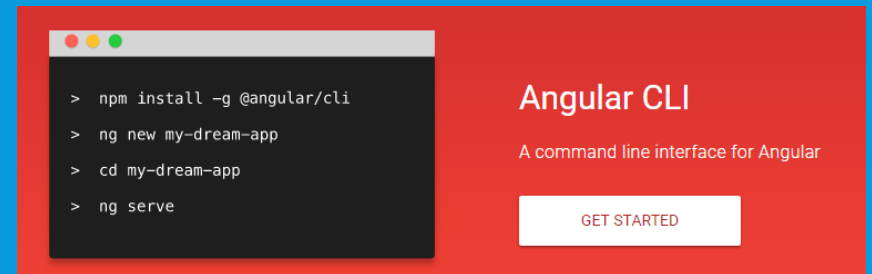
- A module bundler for the static assets in your Angular application
- Transforms & creates bundles provided to the browser
- Works on:
  - JavaScript,
  - Typescript,
  - CSS,
  - ...



# Angular CLI

# What is Angular CLI

- Command Line Interface used to manage Angular projects
- Installation
  - `npm install -g @angular/cli`
- Common Commands
  - `ng new`
  - `ng generate`
  - `ng serve`
- Documentations @ <https://cli.angular.io/>



# .angular-cli.json

- Configuration file for Angular CLI
- Contains settings for
  - root
  - outDir
  - testing
  - linting
  - assets

# ng generate

- Used to create Angular artifacts like
  - class
  - component
  - directive
  - enum
  - guard
  - interface
  - module
  - pipe
  - service
- Syntax:
  - `ng generate directive [name]`
  - `ng g directive [name]`

# Use Angular CLI with Yarn

- By default Angular CLI uses NPM
- To speed up DL time use Yarn instead
- Can be achieved using ng set cmd
  - `ng set [key]=[value]`
  - `ng set --global packageManager=yarn`



# ng serve / ng build

- ng serve
  - Uses webpack-dev-server
  - Compiled output is served from memory, not from disk
  - Does NOT include all project files
  - Runs on `http://localhost:4200` by default
  - Runs in watch mode
- ng build
  - Writes output to `dist/` folder

# Update Angular

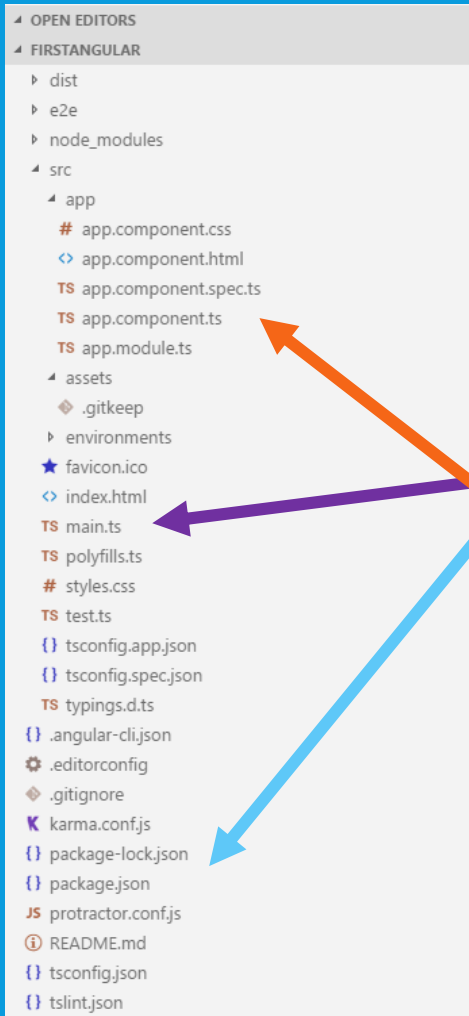
- Customize to your needs!

```
npm install @angular/common@latest
@angular/compiler@latest
@angular/compiler-cli@latest
@angular/core@latest
@angular/forms@latest
@angular/http@latest
@angular/platform-browser@latest
@angular/platform-browser-dynamic@latest
@angular/platform-server@latest
@angular/router@latest
@angular/animations@latest
"@angular/cli"@latest
typescript@latest
--save
```



# Bootstrapping Angular / Project Configuration

# Project Structure



- Configuration:

- package.json, tsconfig.json, typings.json

- Bootstrapping:

- main.ts

- App:

- app.module.ts
- app.module.ts

# package.json

- package.json is the configuration file for node.js
- Packages are downloaded automatically, when file is saved
- Consists of:
  - Metadata
  - Dependencies
  - DevDependencies
    - ... not installed using: npm install --production
  - Scripts

```
package.json x
1 {
2   "name": "angular2-seed",
3   "version": "1.0.0",
4   "description": "A simple starter Angular2 project",
5   "scripts": {
6     "build": "webpack --progress",
7     "watch": "npm run build -- --watch",
8     "server": "webpack-dev-server --inline --progress --port 3000",
9     "start": "npm run server"
10  },
11  "contributors": [
12    "Rob Wormald <robwormald@gmail.com>",
13    "PatrickJS <github@gdi2290.com>"
14  ],
15  "license": "MIT",
16  "dependencies": {
17    "@angular/common": "~2.2.1",
18    "@angular/compiler": "~2.2.1",
19    "@angular/compiler-cli": "~2.2.1",
20    "@angular/core": "~2.2.1",
21    "@angular/forms": "~2.2.1",
22    "@angular/http": "~2.2.1",
23    "@angular/platform-browser": "~2.2.1",
24    "@angular/platform-browser-dynamic": "~2.2.1",
25    "@angular/platform-server": "~2.2.1",
26    "@angular/router": "~3.2.1",
27    "@angular/upgrade": "~2.2.1",
28    "angular2-in-memory-web-api": "0.0.21",
29    "bootstrap": "^3.3.7",
30    "core-js": "^2.4.1",
31    "ie-shim": "^0.1.0",
32    "reflect-metadata": "^0.1.3",
33    "rxjs": "5.0.0-beta.12",
34    "zone.js": "~0.6.26"
35  },
36  "devDependencies": {
37    "webpack": "^1.12.2",
38    "webpack-dev-server": "^1.16.2",
39    "webpack-cli": "^1.2.0",
40    "typescript": "^2.0.3",
41    "tslint": "^3.5.0",
42    "tslint-loader": "^3.0.1",
43    "tslint-config-airbnb": "^5.11.1",
44    "tslint-config-angular": "^0.10.0",
45    "tslint-plugin-interface": "^0.1.0",
46    "tslint-plugin-jsdoc": "^0.2.0",
47    "tslint-plugin-mocha": "^0.2.0",
48  },
49  "keywords": [
50    "Angular2",
51    "angular2-seed",
52    "official angular 2 seed",
53    "official angular2 seed"
```

# Angular Dependencies

- Angular has the following dependencies
  - systemjs – Module Loading
  - es6-promise – Promises to ES5
  - es6-shim – Other missing ES6 functionality for ES5
  - reflect-metadata – ES7 Decorators to ES5/ES6
  - rxjs – Reactive extensions – Use observables instead of Promises
  - zone.js – Event handling outside Angular

# tsconfig.json

- Indicates the root of a typescript project
- Specifies the compiler options
- Most common used settings
  - sourceMap
  - moduleResolution
  - target

```
tsconfig.json x
1 {
2   "compileOnSave": false,
3   "compilerOptions": {
4     "outDir": "./dist/out-tsc",
5     "baseUrl": "src",
6     "sourceMap": true,
7     "declaration": false,
8     "moduleResolution": "node",
9     "emitDecoratorMetadata": true,
10    "experimentalDecorators": true,
11    "target": "es5",
12    "typeRoots": [
13      "node_modules/@types"
14    ],
15    "lib": [
16      "es2016",
17      "dom"
18    ]
19  }
20 }
21
```

# typings.json

- References Typings used in project
- Using TypeScript Definition Manager
  - `npm install -g typings`
  - `typings install dt~jquery --save --global ...` dt corresponds to source – definitely typed
  - Docu @ <https://github.com/typings/typings>
- TypeScript 2+ pattern
  - `npm install @types/moment --save`

# main.ts

- Bootstraps the Angular Application
- Load the AppModule

```
main.ts  x
1  import { enableProdMode } from '@angular/core';
2  import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';
3
4  import { AppModule } from './app/app.module';
5  import { environment } from './environments/environment';
6
7  if (environment.production) {
8    enableProdMode();
9  }
10
11 platformBrowserDynamic().bootstrapModule(AppModule);
12
```

# app.module.ts

- The @NgModule decorator identifies AppModule as an Angular module class which is the root of an Angular app
- All artifacts must be registered in the module so that they can be used
- Consists of sections:
  - Imports
  - Declarations
  - Bootstrap
  - Providers

src/app/app.module.ts

```
import { NgModule }      from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent }  from './app.component';

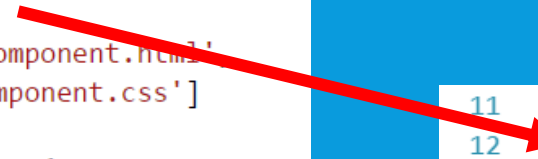
@NgModule({
  imports:      [ BrowserModule ],
  declarations: [ AppComponent ],
  bootstrap:   [ AppComponent ]
})
export class AppModule { }
```



# app.component.ts

- Is the root component of an angular app used in your start page (index.html)
- All other components are nested in this one

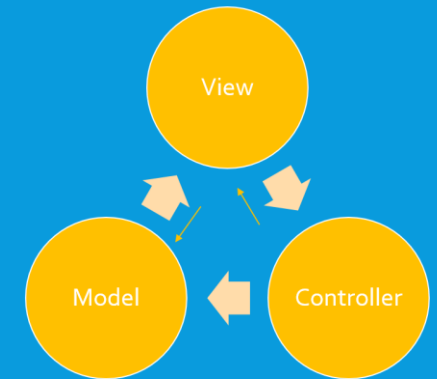
```
app.component.ts x
1  import { Component } from '@angular/core';
2
3  @Component({
4    selector: 'app-root',
5    templateUrl: './app.component.html',
6    styleUrls: ['./app.component.css']
7  })
8  export class AppComponent {
9    title = 'app works!';
10 }
```



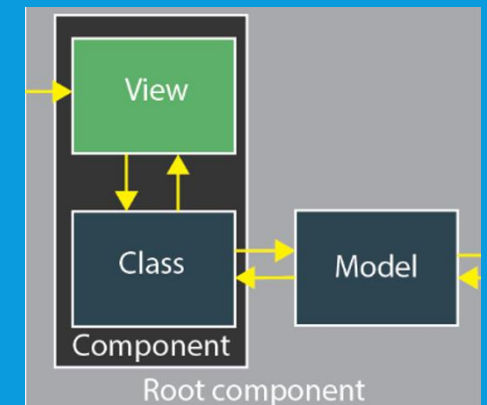
```
11 <body>
12 <app-root>Loading...</app-root>
13 </body>
14 </html>
```

# Components

- An Angular App consists of a set of one or more [nested] component
- @Input | @Output are used to exchange data
- It defines:
  - A selector
  - View: HTML | Inline
  - Directives
  - CSS
  - ...

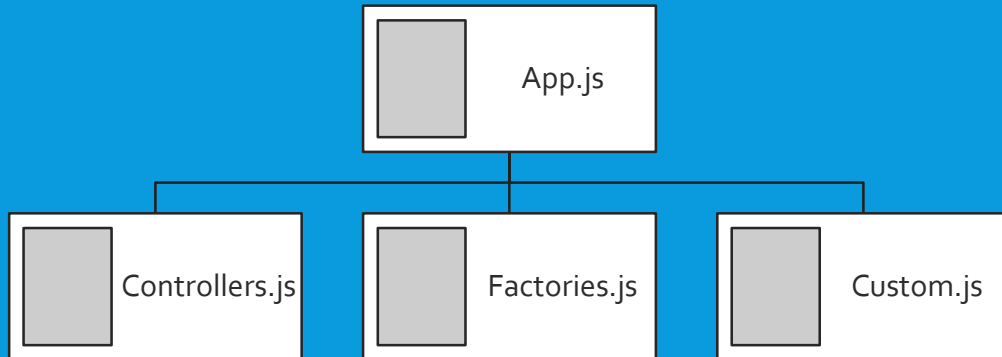


```
home.component.ts x
1  import {Component} from '@angular/core';
2
3  @Component({
4    selector: 'home',
5    styleUrls: ['./home.component.css'],
6    templateUrl: './home.component.html'
7  })
8  export class HomeComponent {
9  }
10
```



# Dependency Graphs

- require keyword is used to build dependency graphs
- Example: `require(' ./wwwroot/app.js ');`



# Installation

- Webpack is installed by default in Angular CLI projects
- Core elements installed using Node.js

- WebPack:

```
npm install webpack -g
```

- WebPack Dev Server:

```
npm install webpack-dev-server -g
```

# webpack.config.js

- Used to automate build
  - `__dirname` refers to the directory where this `webpack.config.js` lives
  - Entry: entry file of the app
  - Output: file to generate
  - Watch: use watcher

```
var path = require('path');
const webpack = require('webpack');

module.exports = {
  entry: {
    app: './wwwroot/app.js'
  },
  resolve: {
    extensions: ['.js']
  },
  output: {
    path: './wwwroot',
    filename: 'js/bundle.js'
  }
};
```

# Angular Debugging into bundles

- Use „Ctrl + o“ in Chrome Dev tools to directly navigate to a file you want to debug

