Angular Lynda.com

Component architecture
↳ write with angular to easy re-use them

Basics of typescript
↳ ES 2015 class
↳ modules
↳ Variables
↳ Function signatures

Minimal learning curve to get where you need to be. Main focus, writing classes, using decorator's and a bit of function parameter typing.

2 benefits of using typescript to write your angular code.

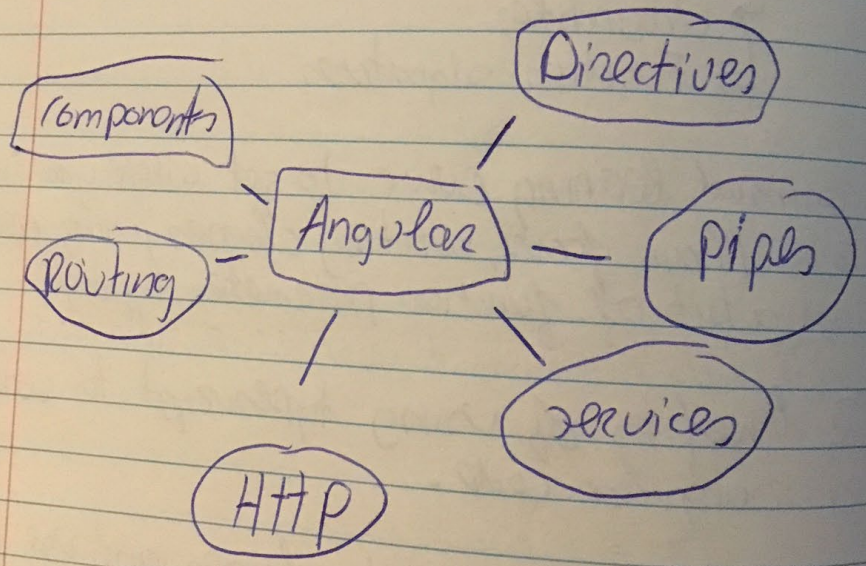
1: Simplicity → write classes and use decorators you end up writing less code.

2: Angular sare code. If you're writing your code in the same fashion as the angular code. It'll be easy for you to look through your sarce code as you grow your angular skill

import statement → typescript syntax that will handle module loading

@Component block - typescript decorator.

Class app component export keyword is a typescript thing for turning the class into a module.



1. Architecture Overview.

Angular is build upon components.

Angular runs on a component tree model.

↳ After angular loads the first component with the bootstrap call, it looks within that components html view and sees if it has any nested components.

↳ ~~Angular~~ Matches? → Runs the appropriate code component

A component in Angular is used to render a portion of html and provide functionality to that portion.

Example → MediaItem Component that can have a property named mediaItem that represents the data for a media item

↳ With each component in Angular you can specify a Html template.

Directives and pipes

In Angular, a component is actually a directive with a template

Directives provide functionality and can transform the DOM (Document Object Model)

two types of directives

1: Structural - modify layout by altering elements
2: Attribute - change the behavior or appearance of an existing DOM element.

~~you can write an element~~

You can write an attribute on an element that matches your selector. Or you can use the template syntax to add a directive in an assignment statement.

A pipe takes end data, like a string or array, and runs some logic to transform it to a new output.

Data binding

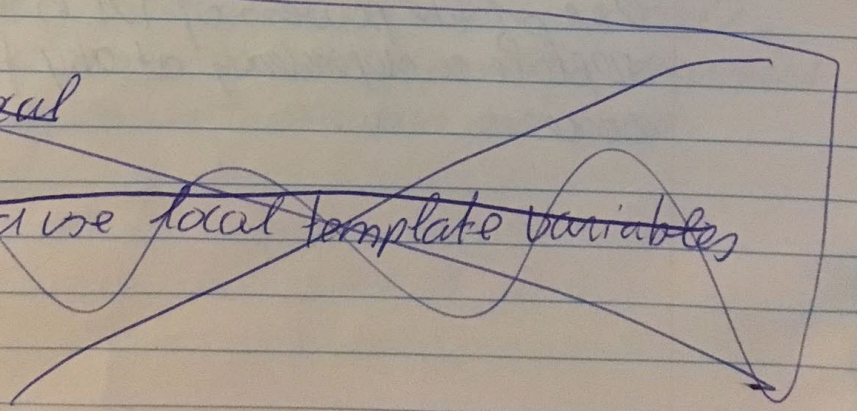
the most common way of displaying data in a view template is via interpolation, where you use curly brackets around a component property to tell Angular to render the content of that property.

In addition to interpolation and build in directives, the syntax has constructs and patterns such as template expressions and statements, a binding syntax for property, attribute, class and style bindings, event bindings, and template expression operators.

~~Create a~~

~~Create local~~

Create and use local ~~template variables~~



Dependency injection

DI in short, is the concept of inversion of control a IOC for short, where you architect code in a way that you provide modules with other modules

DI allows you to write decoupled code that is faster to unit test and to work with

Decouple = Seperate
Unit testen = een methode om software module of stukjes broncode afzonderlijk te testen.

the most common place you use DI is in your class constructor.

one of the powers of DI is the ability to replace a dependency at any phase of application code.

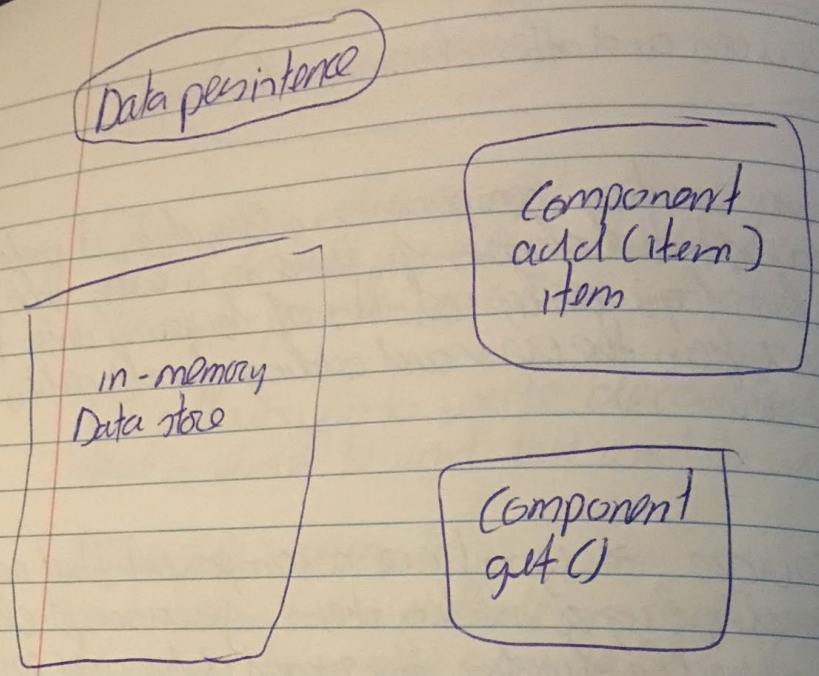
Services and other business logic

When writing components in Angular, it preferred to craft the classes for these in a way that the class logic only consists of brokering data to and from the view and adding functionality to the view

Services → you have a component that needs to display a media item. Javascript class can handle finding the record data and returning it as an object.

One of the nice advantages of angular. the framework is designed to use your code. this allows you to write modular decoupled code that is easier to maintain and reuse.

Data persistence



When you build client-side apps, your intent is typically to work with some kind of data in a read-write fashion

You can create a javascript class to store data, provide it to your app as something that can be injected in and then do constructor injection where needed to bring in the instance of that object.

Web Components. Lynda.com Javascript

modern web page

three layers
↳ Html layer. this is where the content lives. Without Html there is no web page

↳ CSS layer, instructs the browser how to display the Html. Html and CSS work together

↳ javascript, interactive layer. Runs in the Browser and interacts with the Html mark up and CSS rules

in short, javascript is a scripting language that allows you to write small programs that run in the Browser, and change the Html and CSS of the current document

jQuery, library of javascript functions. CSS like syntax and several visual and UI enhancements.
↳ Simplifies the use of javascript in websites.

Javascript frameworks.
↳ AngularJS, React, Vue.js.

Node.js. platform build on browser runtime environments. Used to run advanced operation and application on servers and computers using javascript

the console is a command line for the browser where you can write, manage, and monitor Javascript

1. javascript is case sensitive everywhere instead of the text inside quote

2 Use camelcase
Variables start with lowercase letters
Objects and classes start with uppercase letters
Constants are all-caps

3 Whitespace matters (to humans) consistently throughout.

4 End each statement with a semicolon

~~Anytime you declare~~

Everytime you create a variable, declare it using the var prefix to avoid issues in your code

Data types in javascript

- Numeric - stores regular numbers and integers
- String - stores strings of letters and symbols
- Boolean - stores one or the other of the literals, pair true/false
- null
- undefined
- symbol