B

Vincent Farah

Abstract

<https://app.pluralsight.com/library/courses/angularjs-line-of-business-applications/table-of-contents>

Angular front to back with web api

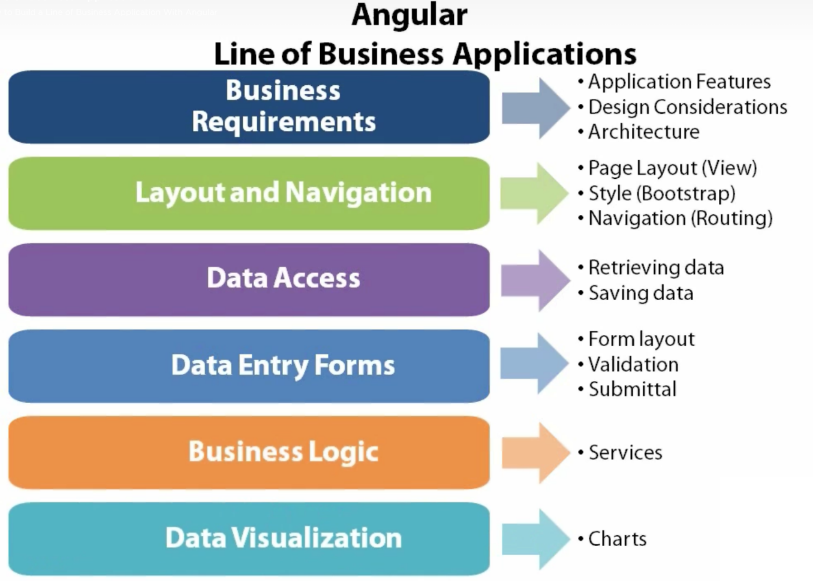
Help keep track of basics and git sample to fork and play with and to use tracking stuff done and keeping a nice summary of important information

# Angular line of business

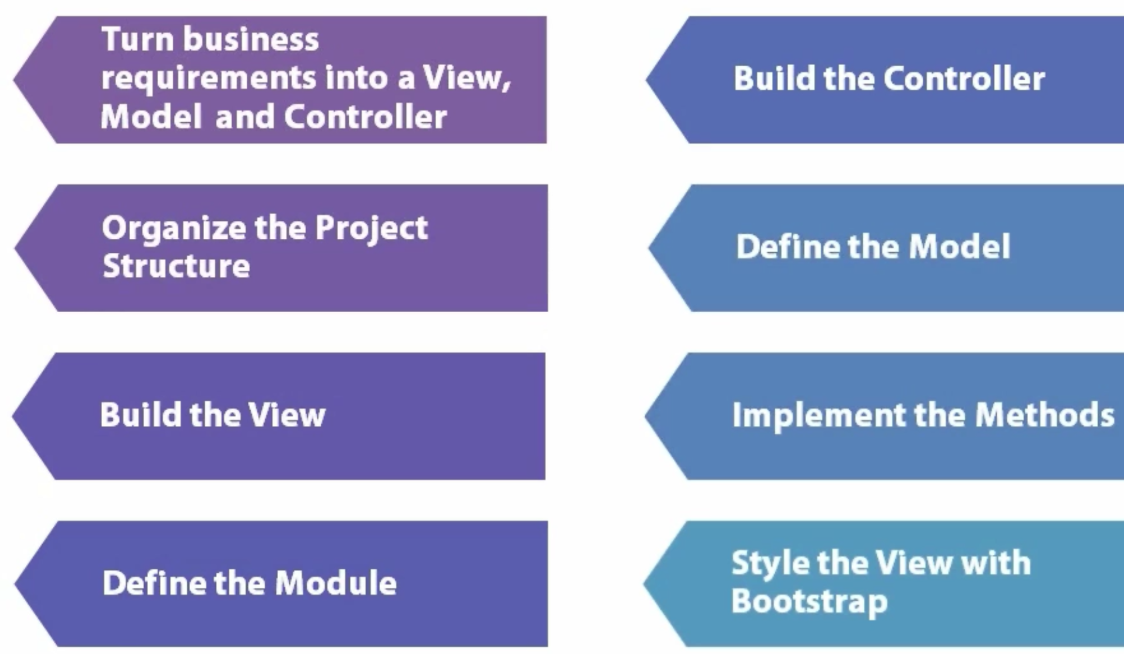
<https://app.pluralsight.com/library/courses/angularjs-line-of-business-applications/table-of-contents>

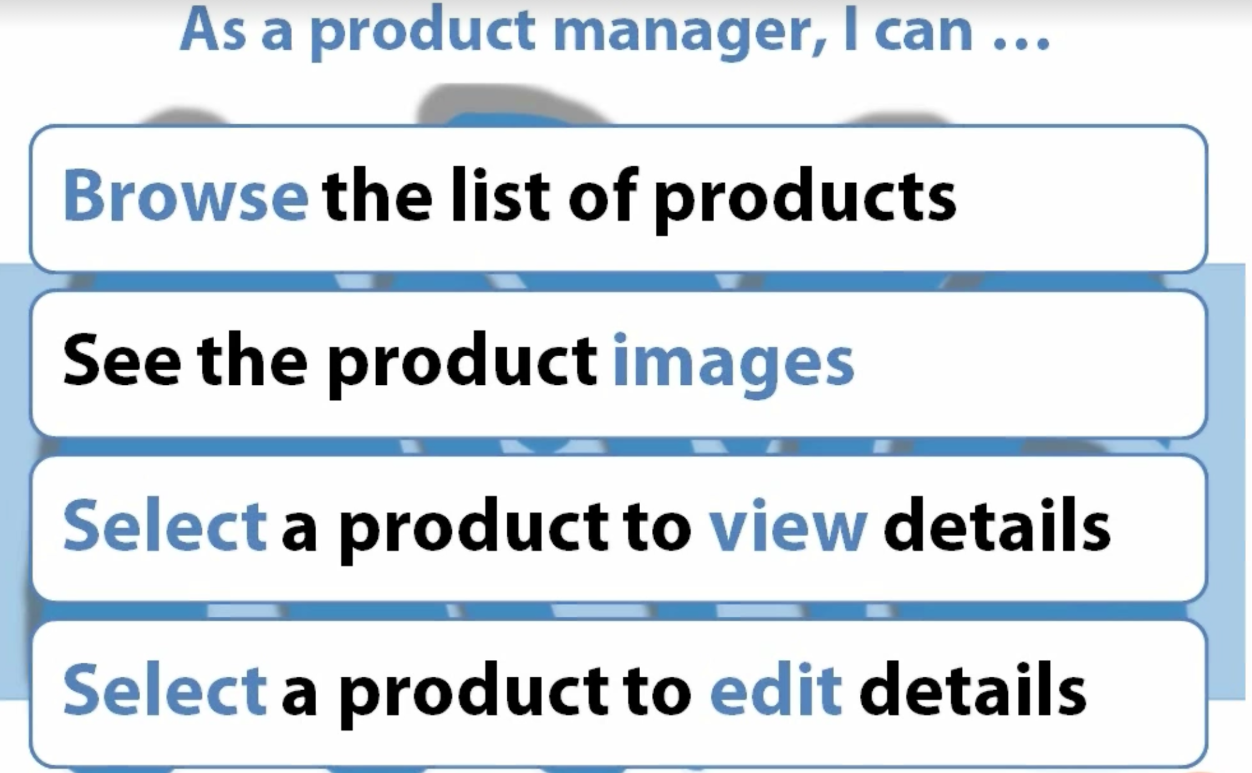
### Why use Angular

* A client side js for building interactive web sites
* Brings simple and clean back to the web site
* Originally developed by google but now open source
* Html is more expressive, easier to read
* Angular is modular, broken up into small units
* Rule based Navigation
* Powerful databinding
* Angular is testable and a focus on separation of concerns means it makes it easier to testing it
* Because it is so popular, it is easy to find help or any resource to help doing this



### Building the first page





### Organising Project Structure

* Create an empty solution
* Install
  + Install-package AngularJS.Core, AngularJS.Resource, Bootstrap, font-awesome, JQuery\*, bootswatch

**<?xml version="1.0" encoding="utf-8"?>**

**<packages>**

**<package id="AngularJS.Core" version="1.5.7" targetFramework="net452" />**

**<package id="AngularJS.Resource" version="1.5.7" targetFramework="net452" />**

**<package id="bootstrap" version="3.3.6" targetFramework="net452" />**

**<package id="Bootswatch" version="3.3.6" targetFramework="net452" />**

**<package id="font-awesome" version="4.6.1" targetFramework="net452" />**

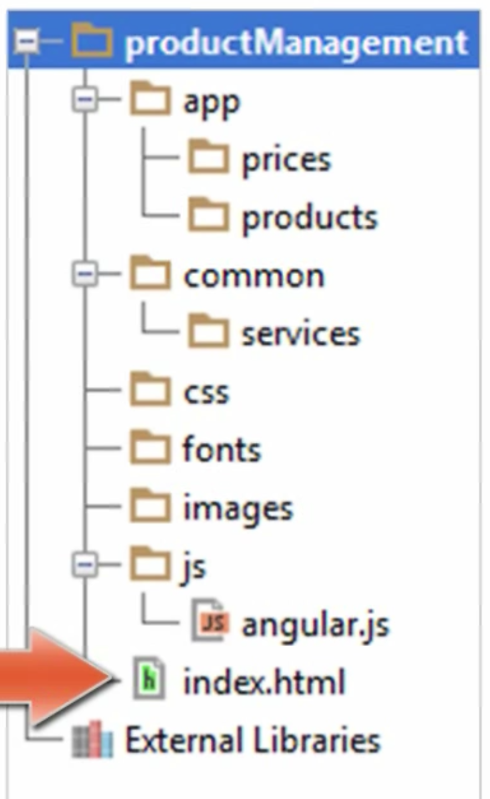
**<package id="jQuery" version="2.2.4" targetFramework="net452" />**

**<package id="jQuery.Validation" version="1.15.0" targetFramework="net452" />**

**<package id="Microsoft.jQuery.Unobtrusive.Validation" version="3.2.3" targetFramework="net452" />**

**<package id="Microsoft.Net.Compilers" version="1.3.0" targetFramework="net452" developmentDependency="true" />**

**</packages>**

* This will be designed by feature
* The app will have two areas, price and products
* Common services like data access will be setup in a common area to be shared
* The rest is self explanatory
* **ng-app** is the most important directive for defining what Angular utilises

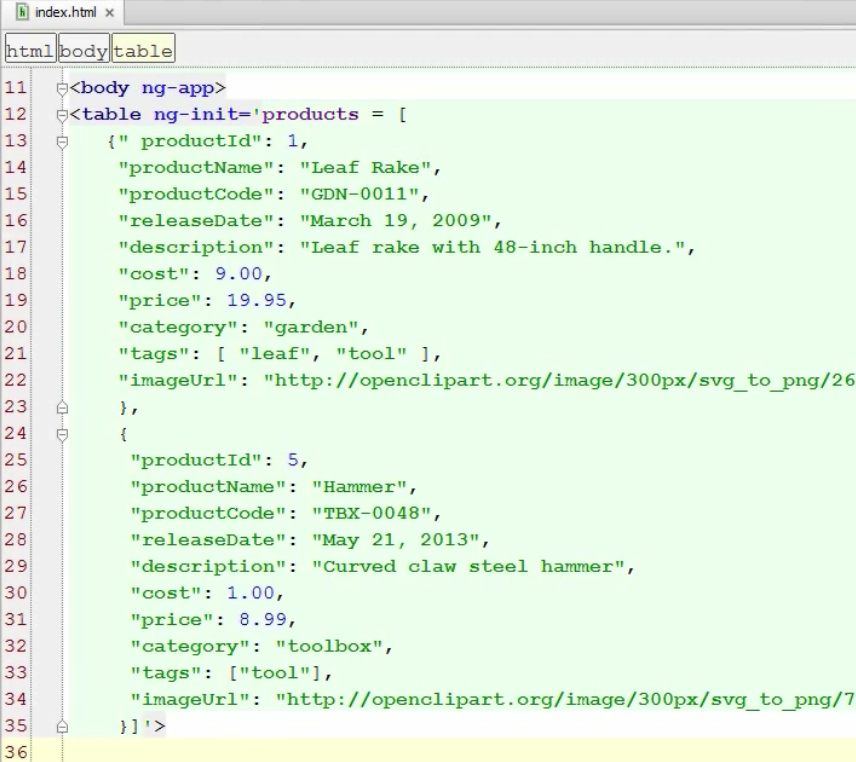
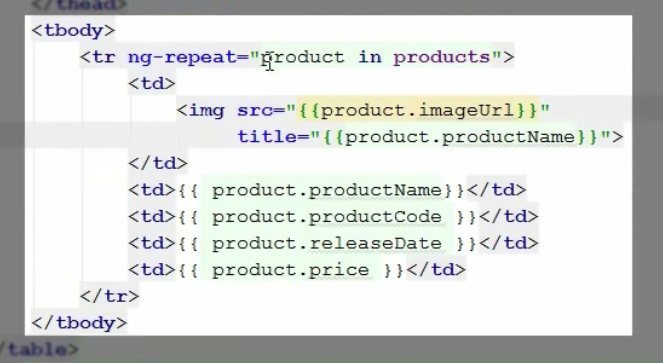
<!DOCTYPE html>

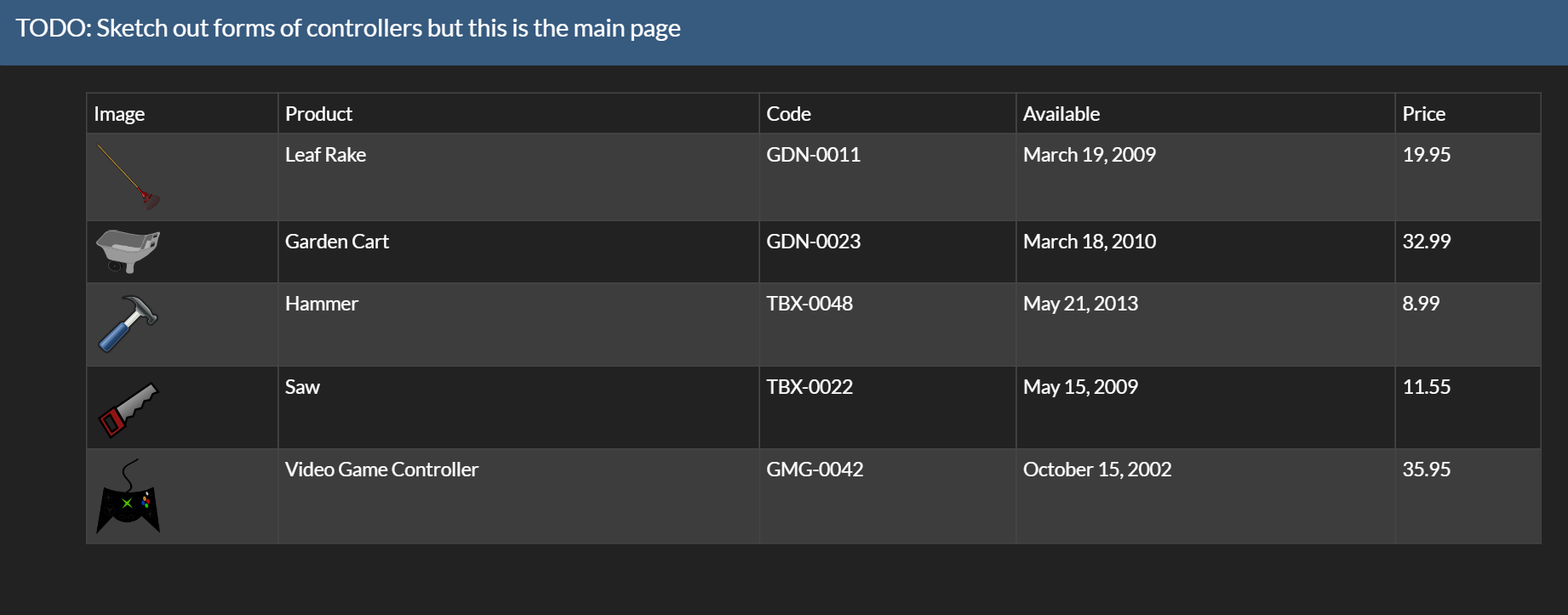
<html>

<head>…</head>

<body **ng-app="productManagement"**

* The **view** is the HTML that defines the visual elements and directives and is called the live template
* When building the view, it is easier to create the mark-up in the html direct and include an **ng-init** with test data

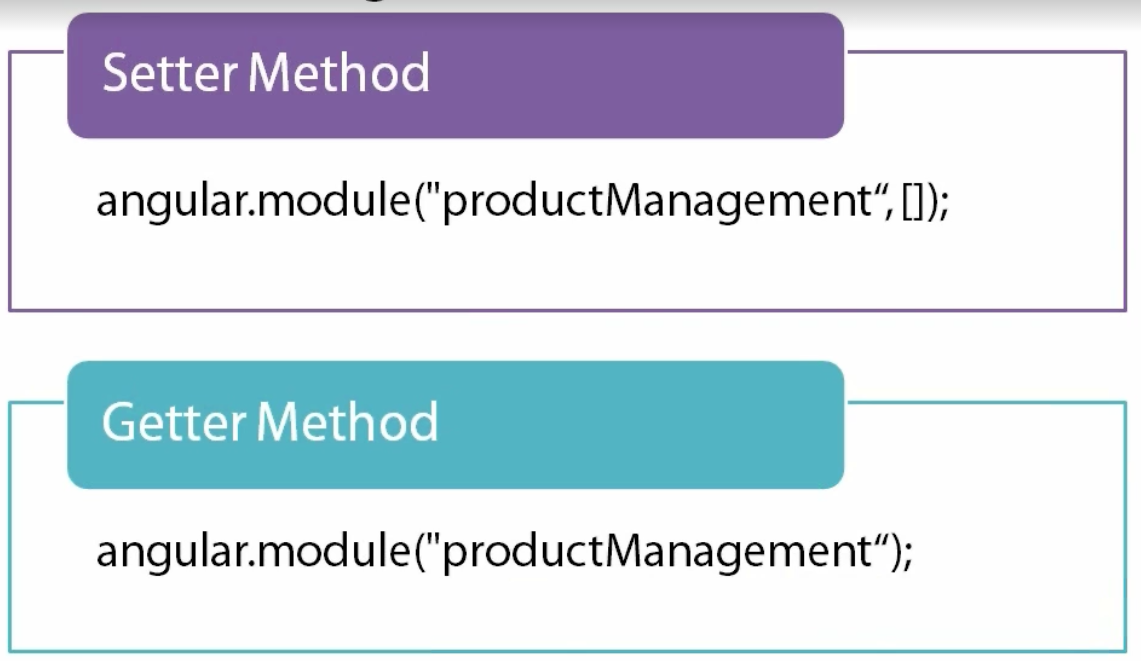




* So the table data can iterate through the data to test the data binding
* Each td is bound to the product data

# Defining the Module

* The Angular Module Method helps to setup the reference to a module



* Assigning this in an IIFE creates this as a self-executing anonymous function and all declarations will not be placed in the global space.

**(function () {**

**'use strict';**

**var app = angular.module('productManagement', [**

**// Angular modules**

**// Custom modules**

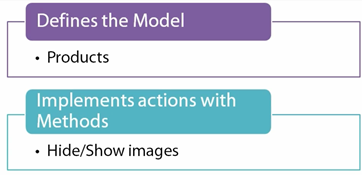
**// 3rd Party Modules**

**]);**

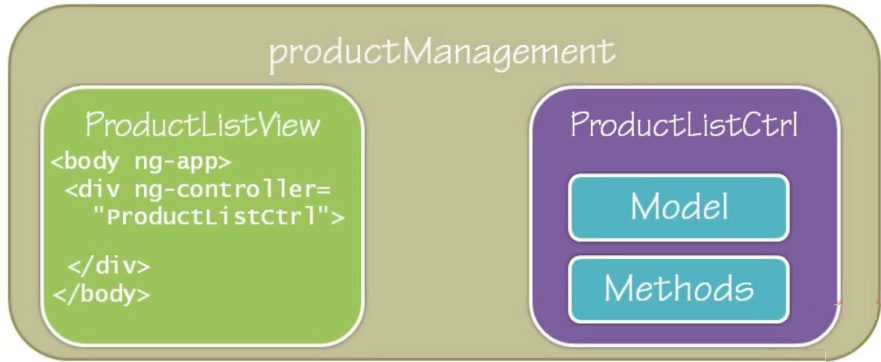
**})();**

# Defining the Controller

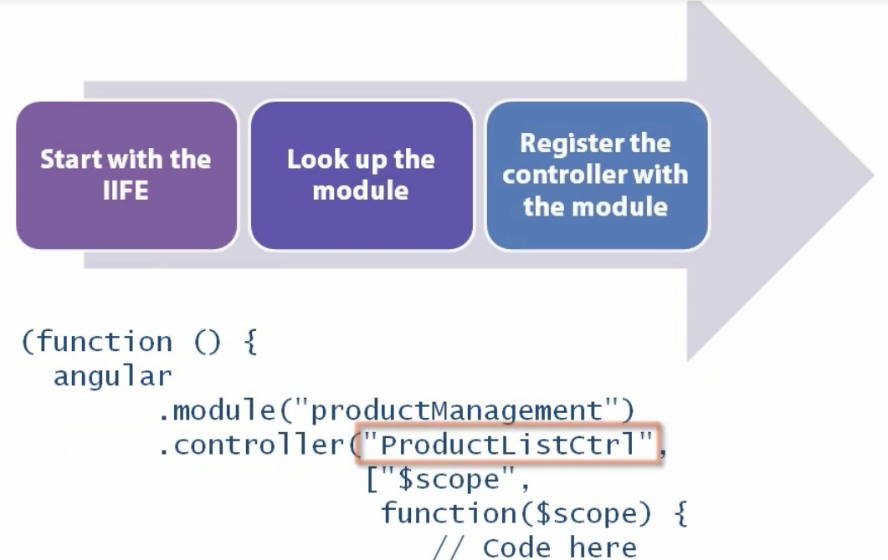
* Defines the model
* Implements any actions



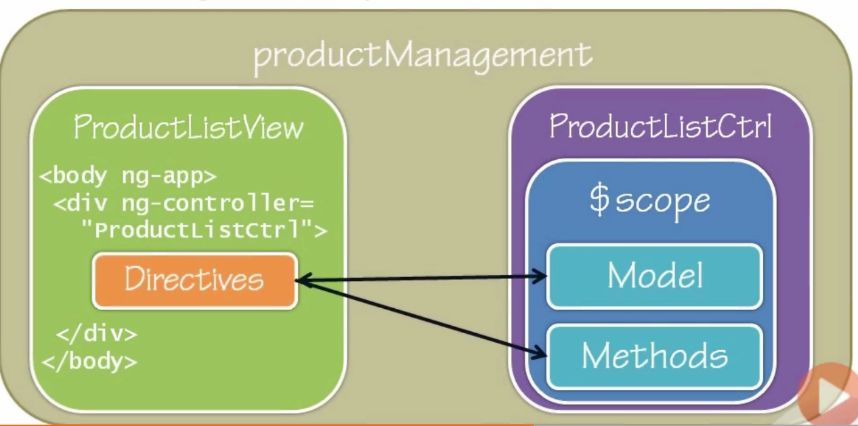
* Controller functions are registered with a module using the **ng-controller** directive



* So the logical order is



* **$scope** is a communication mechanism between the View and the Controller
* Angular injects the scope adding the scope to the list ctrl



* “Controller As” syntax simplified the $scope handling making it not required as a parameter e.g. ProductListCtrl As vm/product
* The model methods are defined in the controller itself
* The views and the model methods are referenced using an alias defined in the ng-controller
* $scope still exists but it lies behind the scenes
* Controller best practises

