AngularJS by Google

Cambell Prince

Cambell Prince 6th December 2013

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

Registration

- Please sign-in and pay (100 Baht)
- Make sure you put in your lunch order
  - This needs to be in by 9:30

Installation and Setup

- This course is hands on, your setup needs to work. Make sure you have:
  - Eclipse (our setup, not your own)
  - Virtualbox (debian7 import)
  - Chrome
  - Gitextensions and clone:
    - https://github.com/cambell-prince/ng-training-101.git
    - To c:\src\sil\ng-training-101
- Ask Chris, Robin, Ira, Tim for help

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

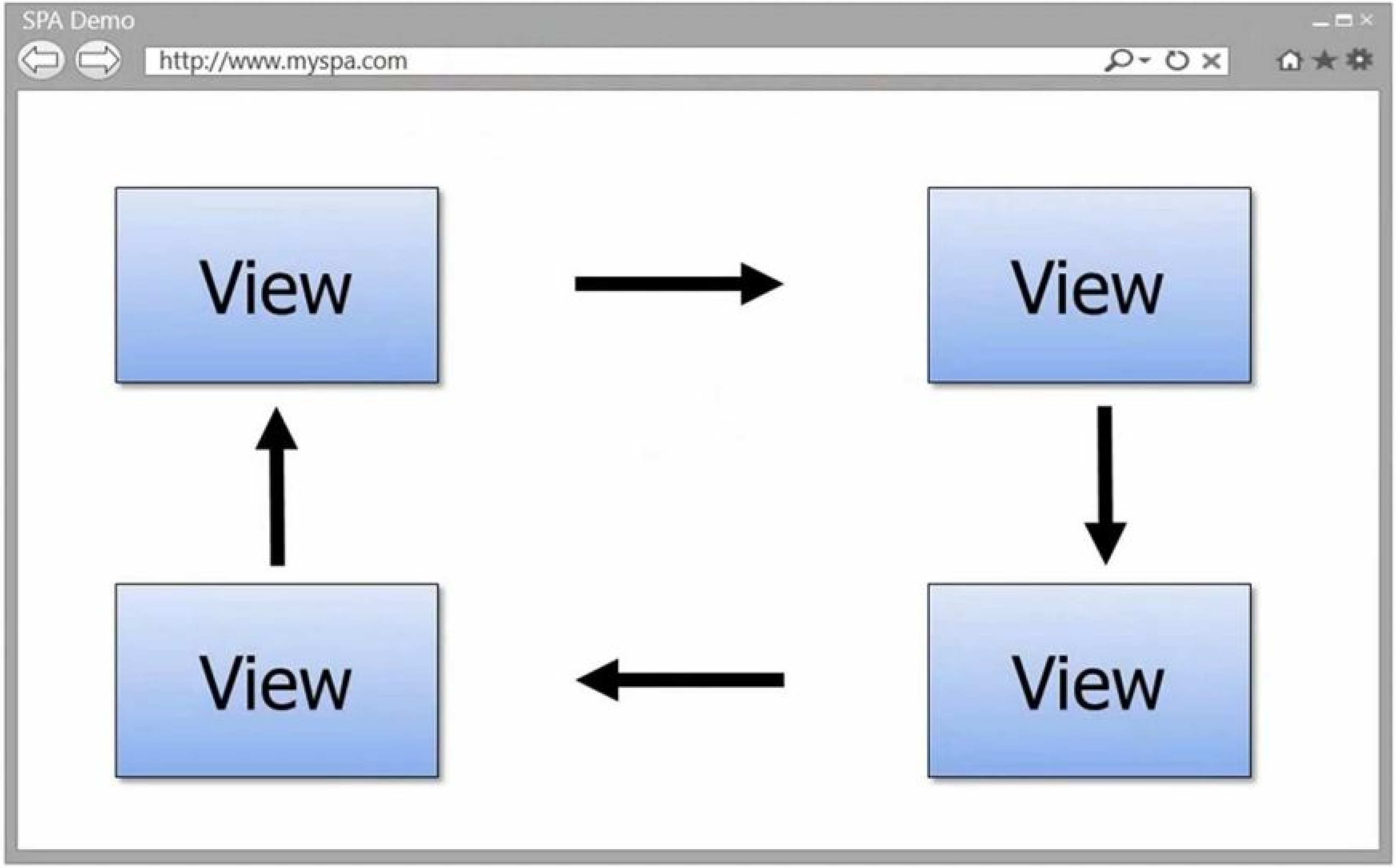
Introduction

- Hands on
- Building a real app
  - Well, almost real :-)
- We're not hiding the complexity
- Mentors are in the room

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

Introduction: What is Angular?



Introduction: SPA Challenges

- DOM Manipulation
- Routing
- Data binding
- View loading
- Module dependencies
- Object modeling

Introduction: SPA Challenges

Data Binding | MVC | Routing | Testing

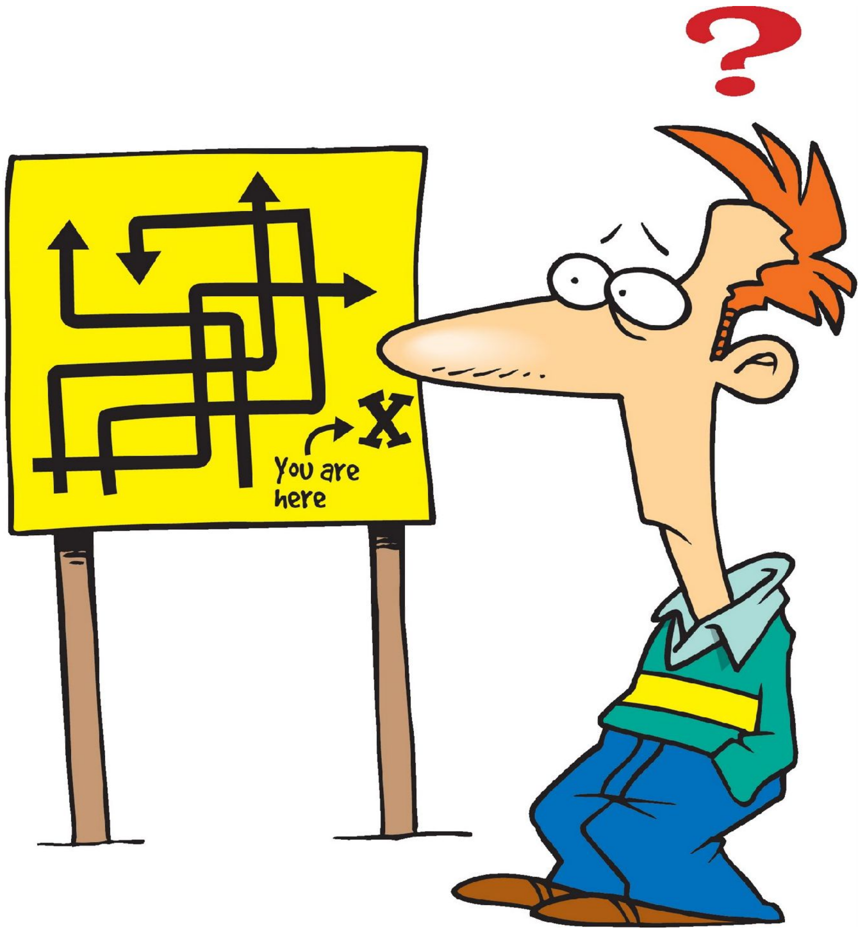
jqLite | Templates | History | Factories

AngularJS is a full-featured SPA framework

ViewModel | Controllers | Views | Directives

Services | Dependency Injection | Validation

Angular



- Scope
- Controller
- Directives
- Routes
- Transclusion
- Compile / Link
- ??? ??? ???

Angular

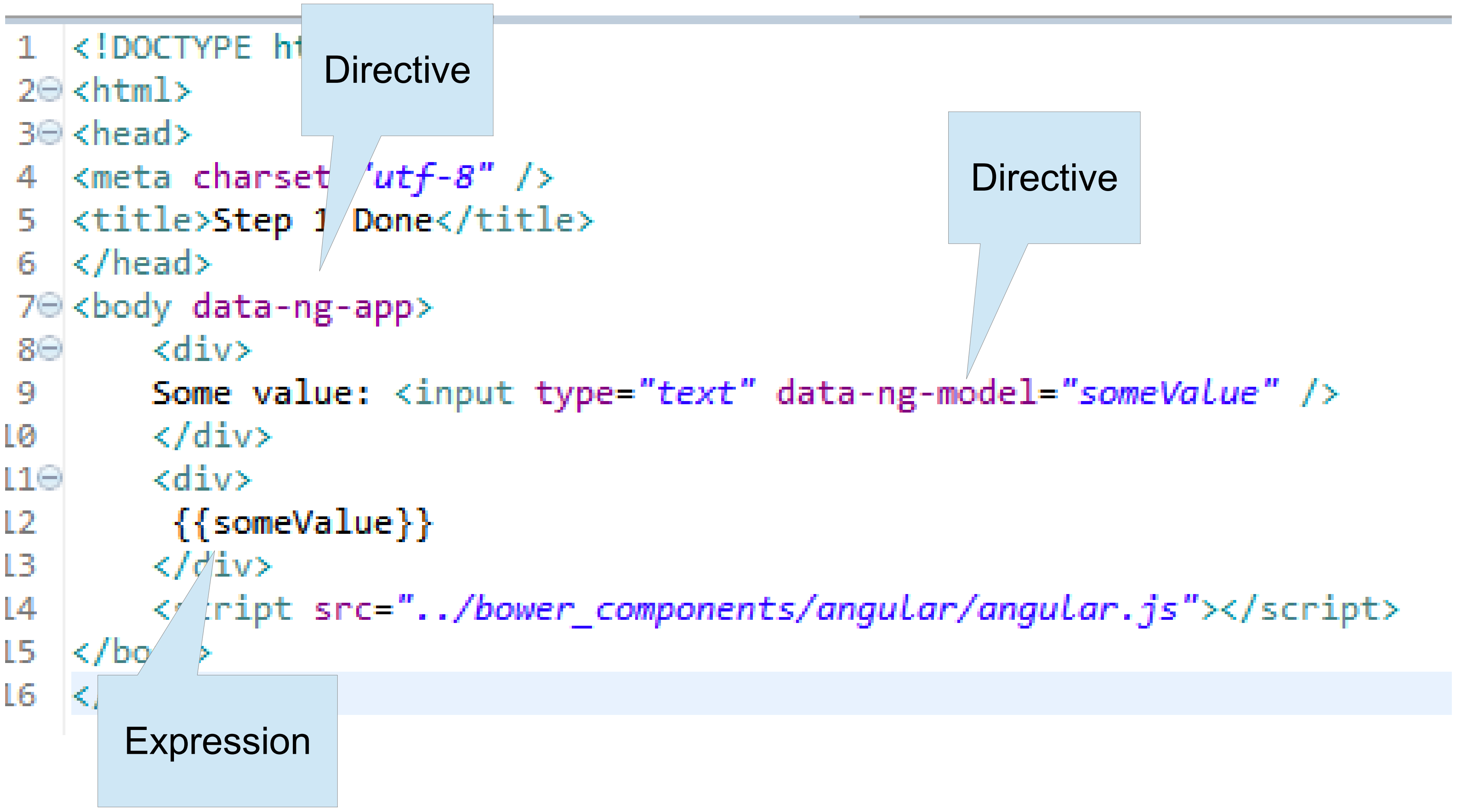
Teaches HTML new tricks

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

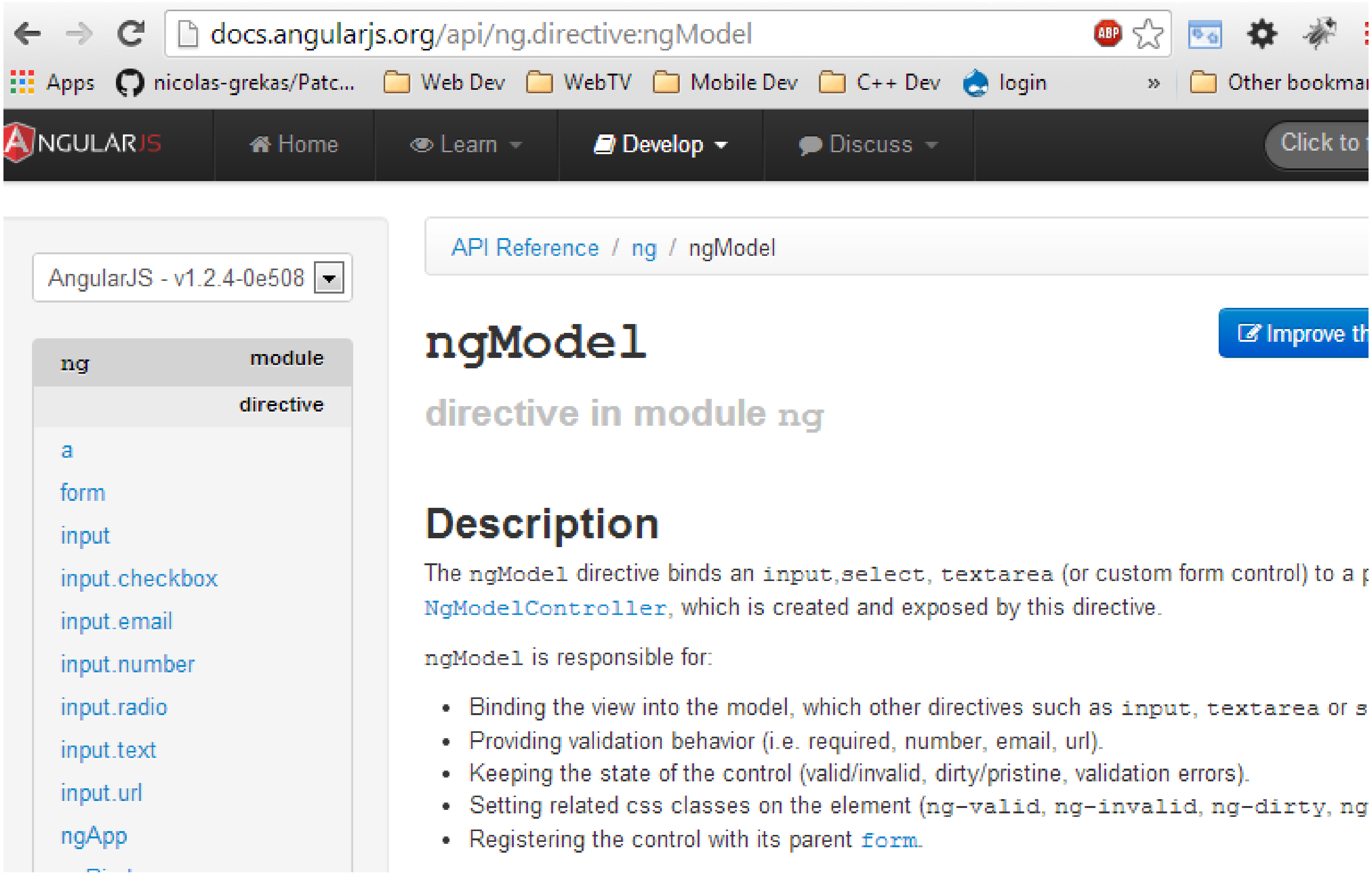
Coding Step 1
Two way data binding with ng-model

Syntax



```
<!DOCTYPE ht
<html>
<head>
<meta charset  'utf-8" />
<title>Step 1 Done</title>
</head>
<body data-ng-app>
    <div>
    Some value: <input type="text" data-ng-model="someValue" />
    </div>
    <div>
     {{someValue}}
    </div>
    <script src="../bower_components/angular/angular.js"></script>
</bo  >
<
```

Documentation



docs.angularjs.org/api/ng.directive:ngModel

API Reference / ng / ngModel

AngularJS - v1.2.4-0e508

ng module

directive

a
form
input
input.checkbox
input.email
input.number
input.radio
input.text
input.url
ngApp

ngModel

directive in module ng

Description

The `ngModel` directive binds an `input`,`select`, `textarea` (or custom form control) to a p
`NgModelController`, which is created and exposed by this directive.

`ngModel` is responsible for:

- Binding the view into the model, which other directives such as `input`, `textarea` or s
- Providing validation behavior (i.e. required, number, email, url).
- Keeping the state of the control (valid/invalid, dirty/pristine, validation errors).
- Setting related css classes on the element (`ng-valid`, `ng-invalid`, `ng-dirty`, ng
- Registering the control with its parent `form`.

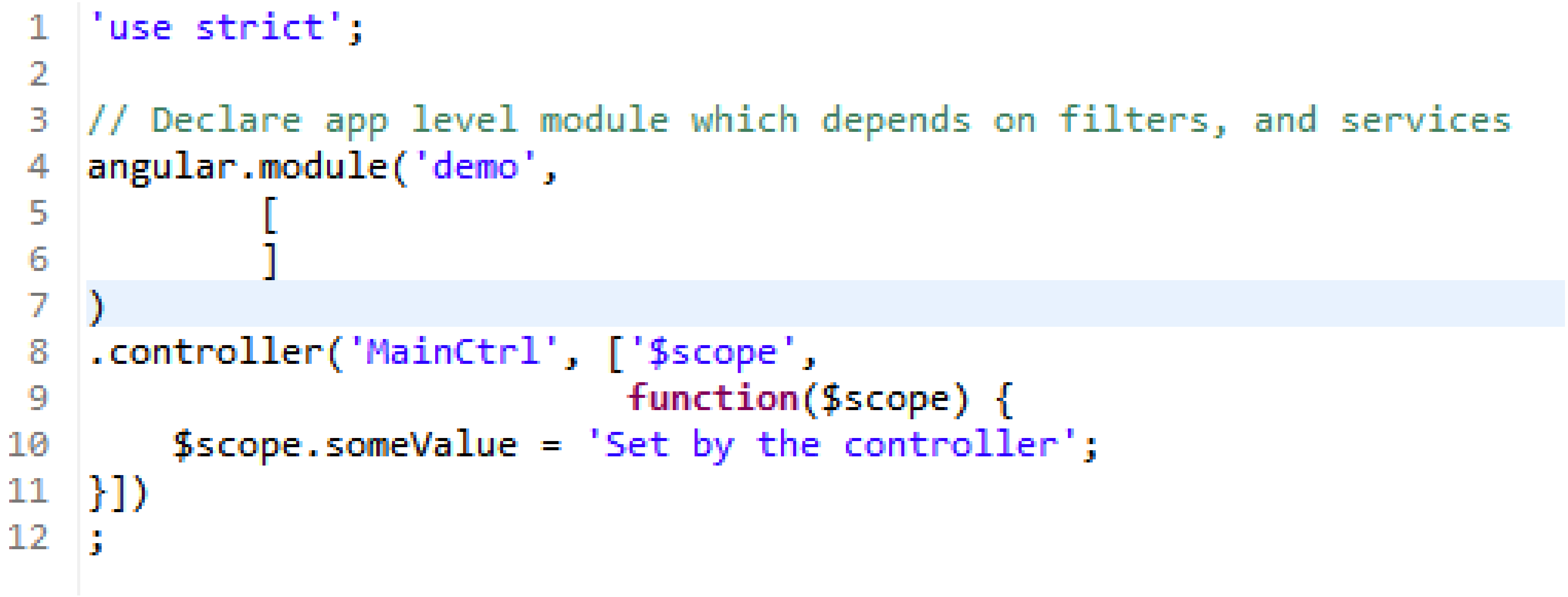
BREAK

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

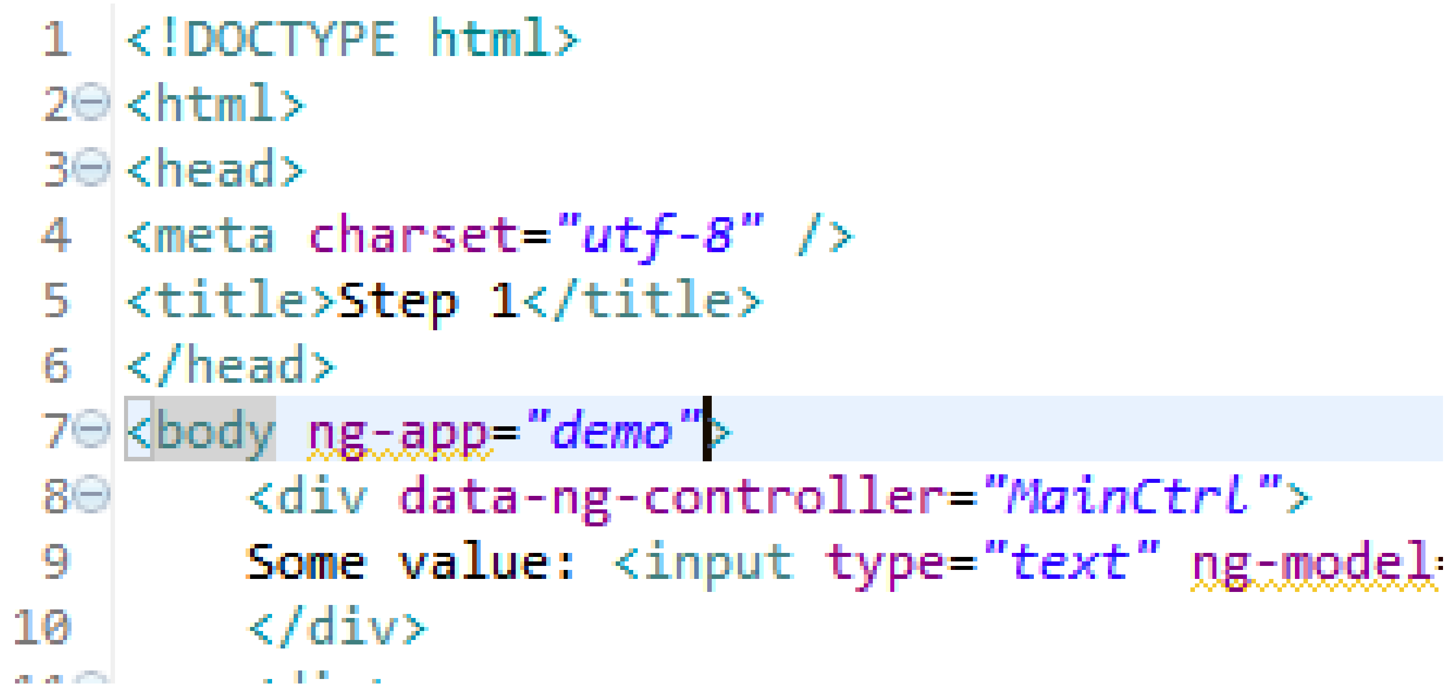
Coding Step 2
Adding a Controller in Javascript

Controller



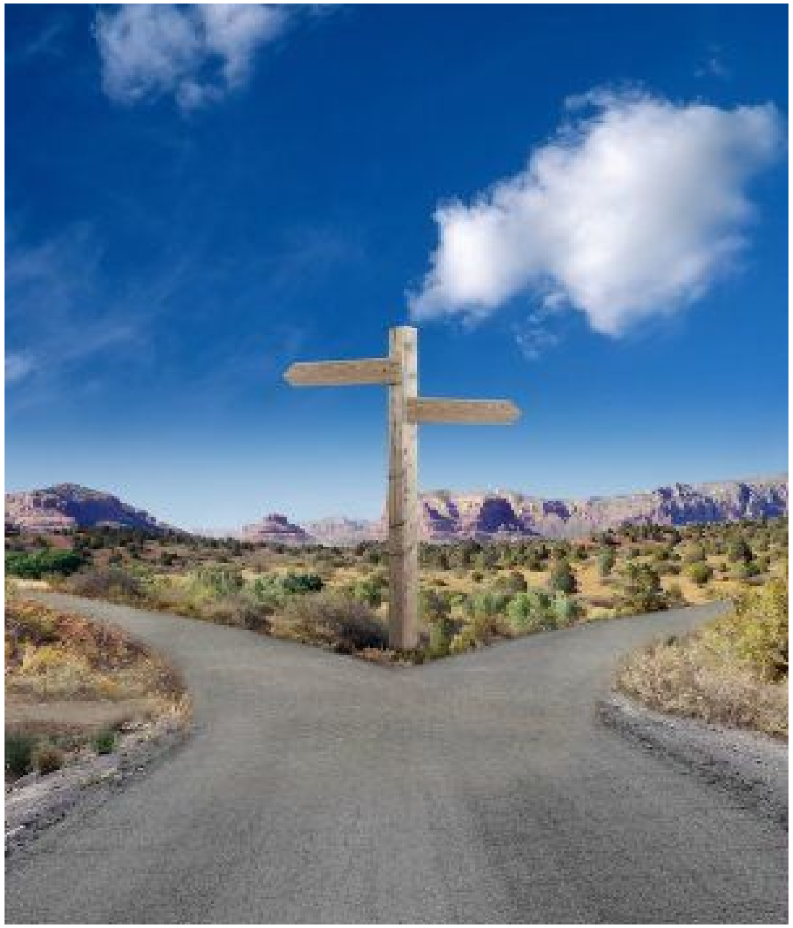
```
'use strict';

// Declare app level module which depends on filters, and services
angular.module('demo',
        [
        ]
)
.controller('MainCtrl', ['$scope',
                         function($scope) {
    $scope.someValue = 'Set by the controller';
}])
;
```

Controller in the View



```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Step 1</title>
</head>
<body ng-app="demo">
    <div data-ng-controller="MainCtrl">
    Some value: <input type="text" ng-model
    </div>
```

Debugging Javascript



- Chrome dev tools
  - Ctrl+shift+i
- Demo Step 2

What are we doing?

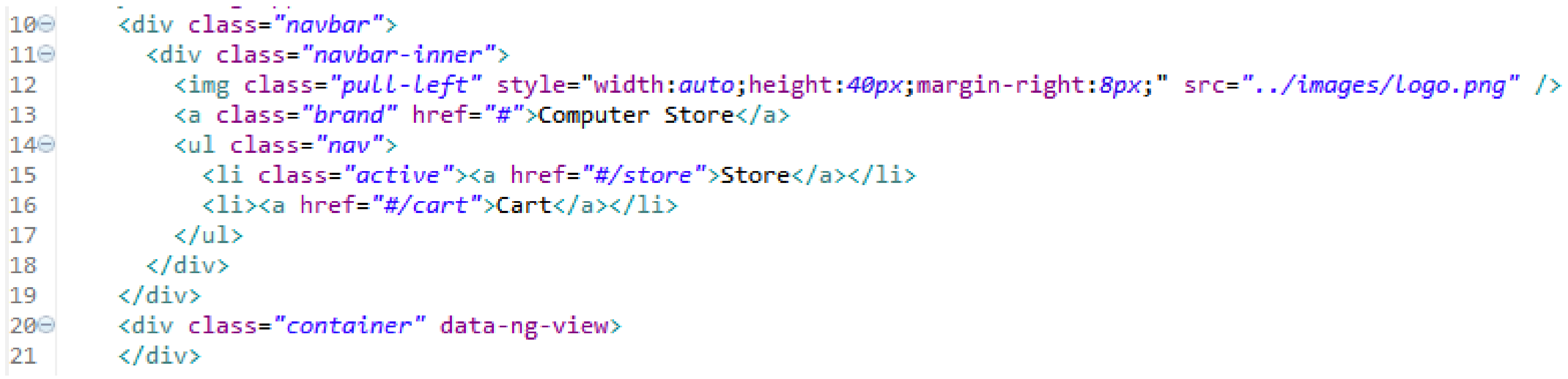
- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

First: About Bootstrap

- Provides css that looks good by default
  - http://getbootstrap.com/2.3.2/
- Scaffolding; the layout grid
- Base CSS; text, buttons, labels, inputs, forms
- Don't use the components with angular
  - Use the angular-ui bootstrap components
  - http://angular-ui.github.io/bootstrap/

Coding Step 3
Adding Routes and Views and Bootstrap

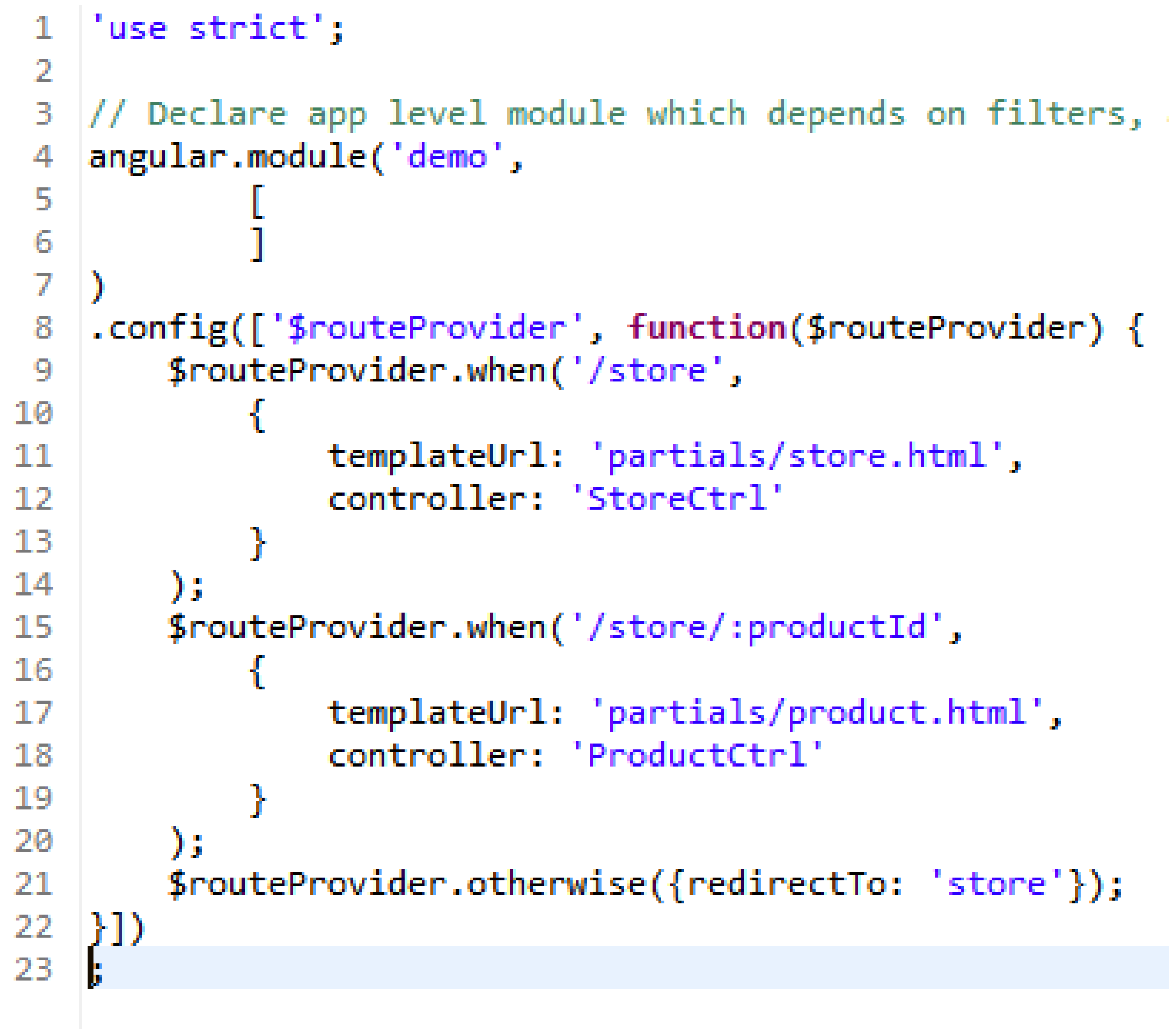
Route: Main View



```
<div class="navbar">
  <div class="navbar-inner">
    <img class="pull-left" style="width:auto;height:40px;margin-right:8px;" src="../images/logo.png" />
    <a class="brand" href="#">Computer Store</a>
    <ul class="nav">
      <li class="active"><a href="#/store">Store</a></li>
      <li><a href="#/cart">Cart</a></li>
    </ul>
  </div>
</div>
<div class="container" data-ng-view>
</div>
```

- From bootstrap navbar
  - http://getbootstrap.com/2.3.2/components.html#navbar
  - Add img for logo with pull-left class
- Ng-view to contain the views as determined by the route

Route: Setting the Routes



```
'use strict';

// Declare app level module which depends on filters,
angular.module('demo',
        [
        ]
)
.config(['$routeProvider', function($routeProvider) {
    $routeProvider.when('/store',
        {
            templateUrl: 'partials/store.html',
            controller: 'StoreCtrl'
        }
    );
    $routeProvider.when('/store/:productId',
        {
            templateUrl: 'partials/product.html',
            controller: 'ProductCtrl'
        }
    );
    $routeProvider.otherwise({redirectTo: 'store'});
}])
;
```

Bootstrap

- Keep your site looking good enough by default
  - Use bootstrap markup, then override the styles to suit the graphics design you want.
- Do this as you go
  - You will **not** come back and fix it up later.
  - Really, it just doesn't happen.

Dependencies

- Modules are defined with their dependencies
- Controllers and Services are defined with their dependencies
- Angular can automatically determine dependencies from function arguments
  - But don't do that. Causes problems with minification.
  - Best practice. Declare your dependencies.

LUNCH

What are we doing?

- Register and Setup
- Introduction
- AngularJS: What is it?
- Models and Scope (MV VM)
- Controllers and Directives
- Routes and Views
- Services

Coding Step 4

Adding Services and Views
The ng-repeat directive

Services: Let's get some data!

Services provide an abstraction of 'data provider', hiding the implementation details from the application.

Code from the application should always presume 'async'.

Services: Let's get some data!

- 'services.js' provides a fake data store. Normally you would implement with $http or $resource angular services for REST data sources.
- Json-rpc is also an option.
  - Data Transfer Object (DTO) for a composite of models.

Services: Let's get some data!

- The controller calls the service to get the model data.
- The model data is then set into the $scope (the VM)
- The view renders
- Excercise:
  - store.html and store.js
  - product.html and product.js

Coding Step 5

Adding the Shopping Cart
The ng-repeat directive

Adding the Shopping Cart

?

Future Features

- Fix the navbar, it doesn't set the active property on the current route.
- Add a row for total to the cart and a checkout button.
- Add 'items in cart count' into navbar.
- Add 'reset cart' button.

Real Word Workflow

- Package Managers:
  - **Bower** (javascript)
  - **Composer** (php)
  - **NPM** (node; server side javascript)
- DVCS
  - **Git**
  - Mercurial
- Tests
  - Karma; unit tests and end to end tests

Real Word Workflow

- Build
  - **Grunt** current best practice
  - **Ant** what we currently use
- Continuous Integration (CI)
  - **Team City**
  - Jenkins

END

Evaluation Please

http://goo.gl/TGW7Ge