mitchell

mitchell

Intro To AngularJS

5/3/2016

- Patrick Traeger, SW Dev Eng II

We (m)power better outcomes.

WIFI Access and Project Site

User: Guest

Password: Mitchell

Project Site:
github.com/angularjs-gdit

Agenda

- Intro
- Quick Review of Prerequisites
- What is AngularJS
- MVC / MV*
- Getting Started with AngularJS
- Built-in Directives
- Modules
- Controllers and Scope
- Dependency Injection
- Angular 2.0

Intro

Tell us about you:

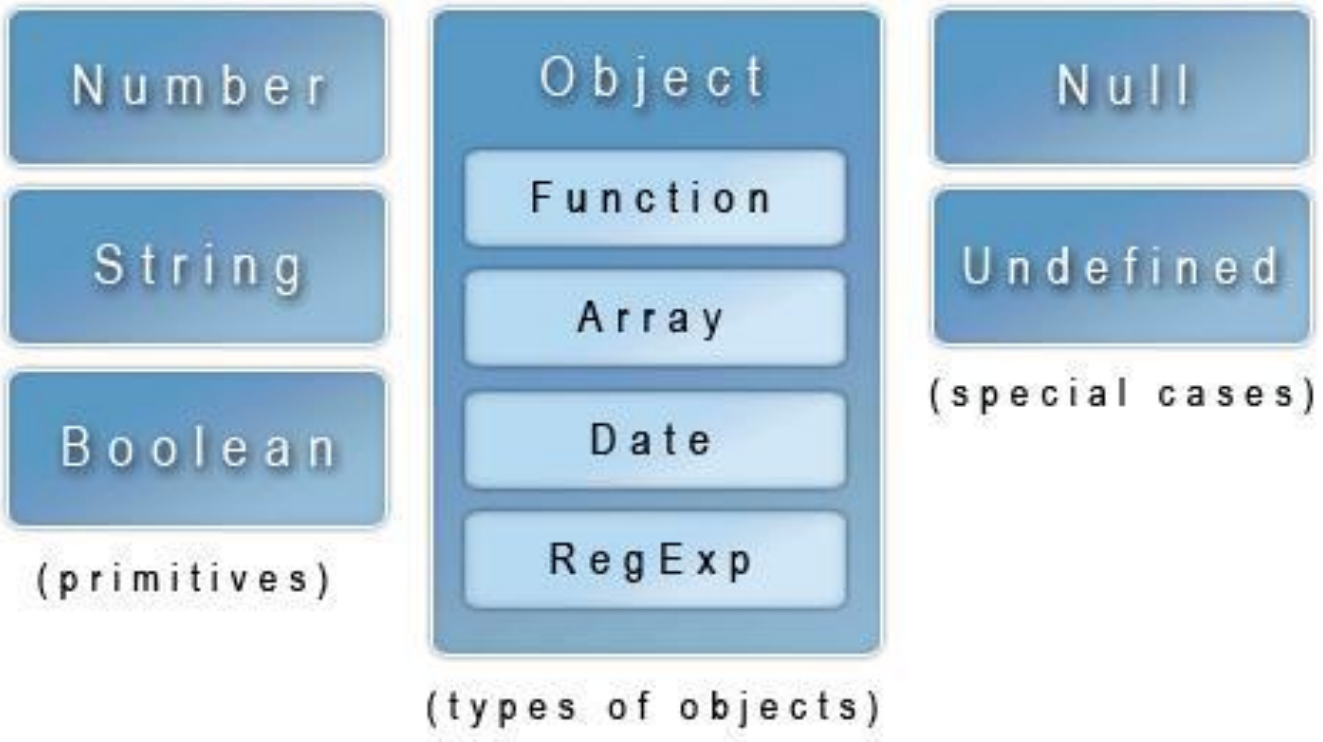
- Name
- Experience with Angular
- What are you hoping to gain f

Prerequisites

Quick Review of JS and HTML

mitchell

JavaScript in 90 seconds: Data Types



JavaScript in 90 seconds: Declaring Variables

```
var variable = 3;
```

JavaScript in 90 seconds: Declaring Variables

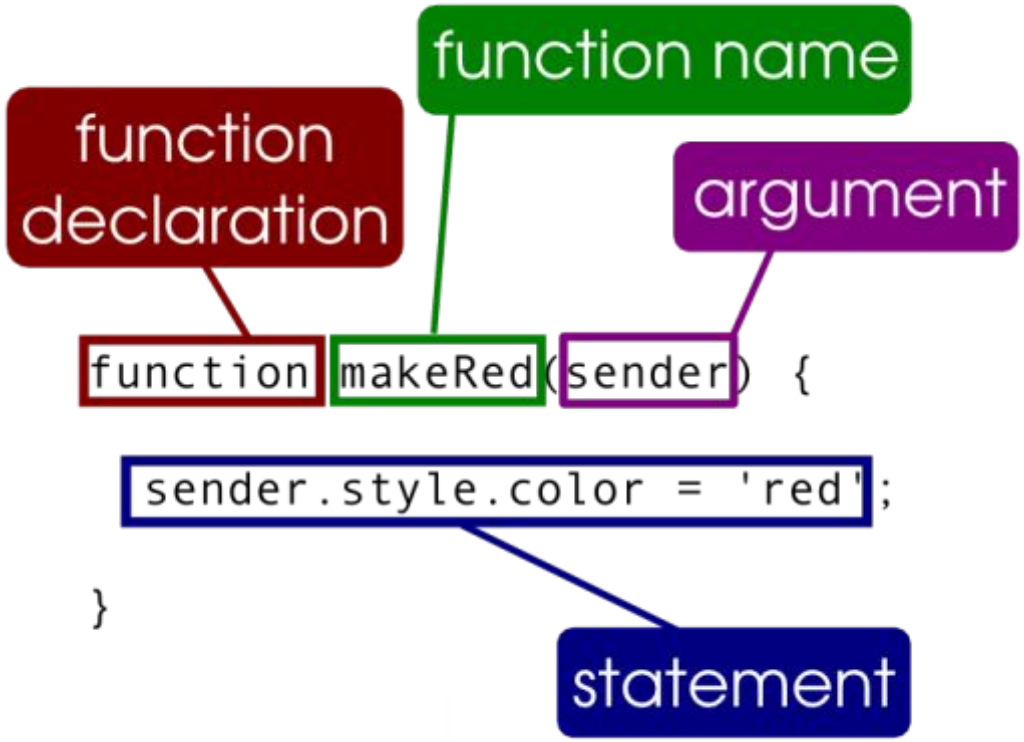
```
var object = {};
object.property = 'my value!';
```

JavaScript in 90 seconds: Declaring Variables

```
var object = {
    property : 'my value!'
};
```

JavaScript in 90 seconds: Functions

Functions are a block of code that can execute a task. They can be called elsewhere in your program to perform its code.



JavaScript in 90 seconds: Functions as args

```
var myFunction = function(func){
    func();
};

var callback = function(){
    alert('hello');
};

myFunction(callback);
```

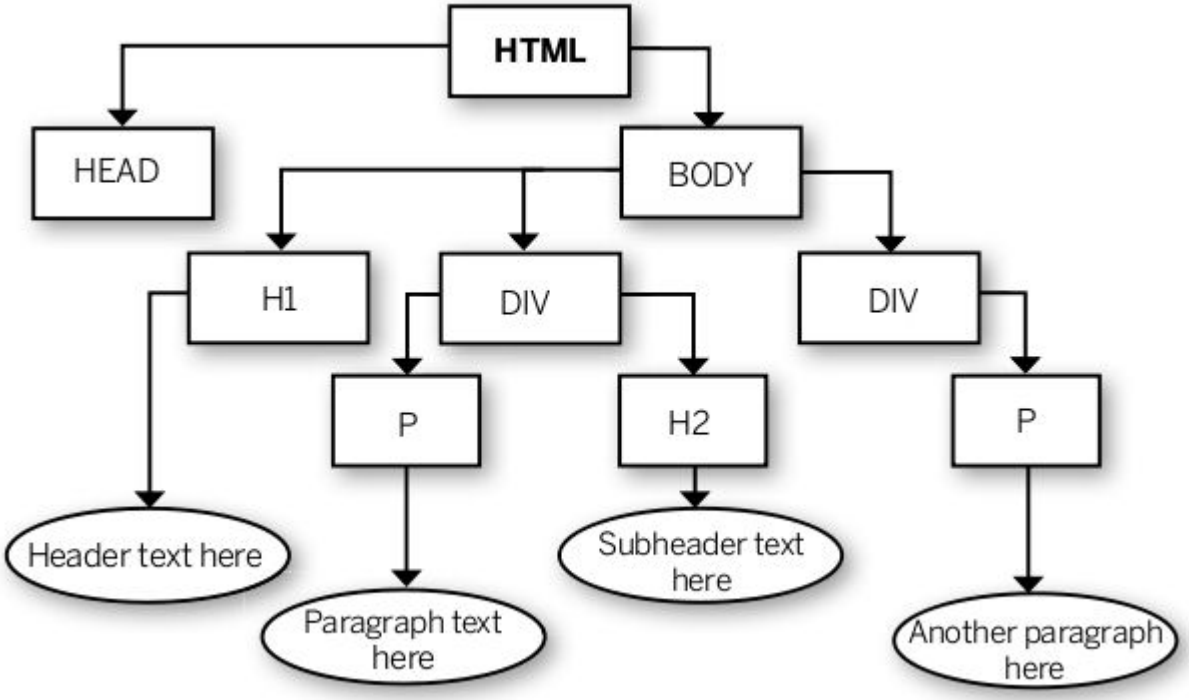
JavaScript in 90 seconds: JSON

Key Value Pair data type wrapped in curly braces

```
{
  "arguments" : { "number" : 10 },
  "url" : "http://localhost:8080/restty-tester/collection",
  "method" : "POST",
  "header" : {
    "Content-Type" : "application/json"
  },
  "body" : [
    {
      "id" : 0,
      "name" : "name 0",
      "description" : "description 0"
    },
    {
      "id" : 1,
      "name" : "name 1",
      "description" : "description 1"
    }
  ],
  "output" : "json"
}
```

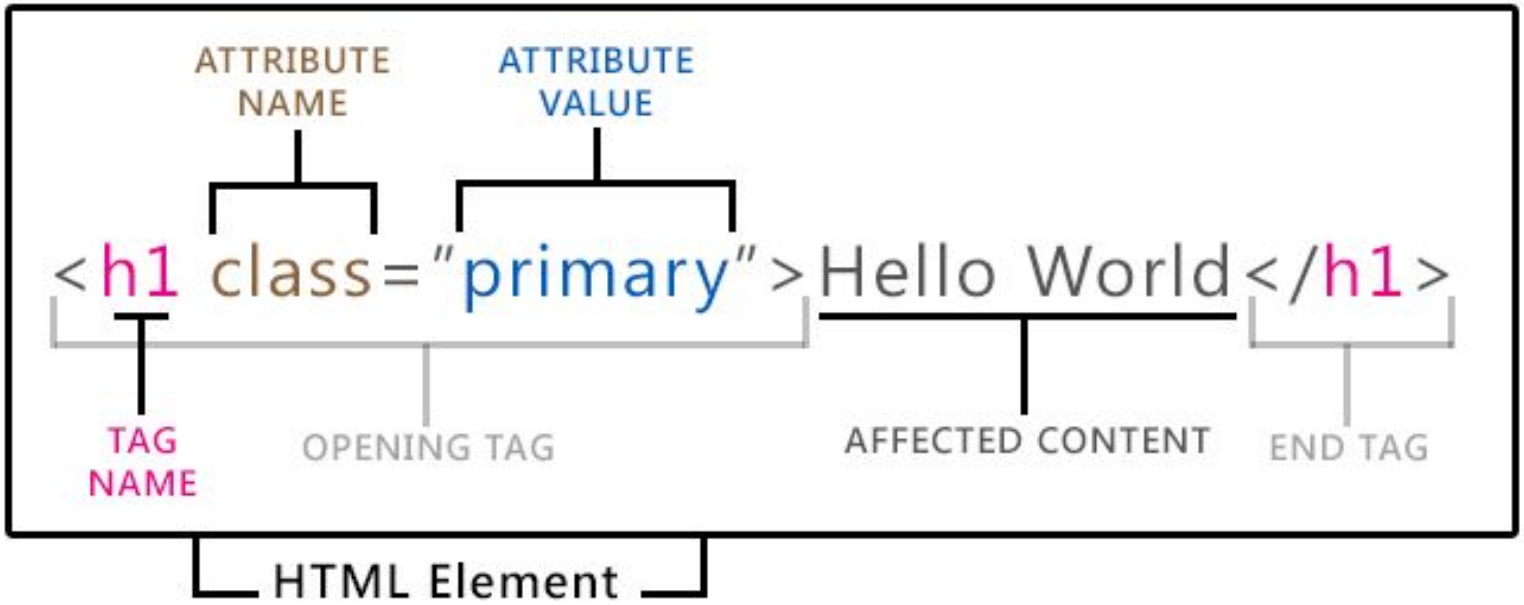
mitchell

HTML in 90 seconds: the DOM



HTML in 90 seconds: Syntax

HTML in 90 seconds: Syntax



HTML 5 Cheat Sheet [TAGS]

New [tags added in HTML5]

<article>	self-contained composition that is independently distributable
<aside>	section of page that consists of content tangentially related to content around it
<audio>	sound content
<bdi>	span of text to be isolated from surroundings for bidirectional formatting purposes
<canvas>	area that can be used to draw graphics via JavaScript
<command>	user invokable command
<datalist>	dropdown list
<datatemplate>	data template
<details>	details of an element
<embed>	embedded content
<figcaption>	caption of figure element
<figure>	group of media content
<footer>	footer for section or page
<header>	header for section or page
<hgroup>	group of headings for section
<keygen>	generated key in a form
<mark>	marked text
<meter>	measurement in defined range
<nav>	navigation links
<output>	represents results of calculation
<progress>	progress of any kind of task
<rp>	parenthesized ruby text
<rt>	ruby text
<ruby>	ruby annotations
<section>	section in a document
<source>	media resources
<summary>	header of a detail element
<time>	date/time
<video>	video
<wbr>	possible line break

Old [unsupported tags]

<acronym>	acronym
<applet>	applet
<basefont>	base font
<bgsound>	background sound
<big>	big text
	centered text
<fn>	footnotes
<font>	text font, size, and color
<frame>	sub window
<frameset>	set of frames
<isindex>	provides searchable index related to current document
<dir>	directory list
<noembed>	no embed section
<noframes>	no frame section
<s>	strikethrough text
<strike>	strikethrough text
<tt>	teletype text
<u>	underlined text
<xmp>	preformatted text

Existing [tags in HTML4 & 5]

<!--...-->	comment
<!doctype>	document type
<a>	hyperlink
<abbr>	abbreviation
<address>	address element
<area>	image map area
<b>	bold text
<base>	base URL for all links in page relative to document root
<bdo>	text direction
<blockquote>	long quotation
<body>	body element
 	single line break
<button>	push button
<caption>	table caption
<cite>	citation
<code>	code text
<col>	attributes for columns
<colgroup>	groups of columns
<dd>	definition description
<del>	deleted text
	generic block-level element
<dfn>	defining instance of a term
<dl>	definition list
<dt>	definition term
	emphasized text
<fieldset>	logically group items in a form
<form>	defines a form
<h1> to <h6>	header 1 to header 6
<head>	document information
<hr>	horizontal rule
<html>	html document
<i>	italic text
<iframe>	inline sub window
<img>	image
<input>	input field
<ins>	inserted text
<kbd>	keyboard text
<label>	label for a form control
<legend>	title in a fieldset
	list item
<link>	resource reference
<map>	image map
<menu>	menu list
<meta>	meta information
<noscript>	no script section
<object>	embedded object
	ordered list
<optgroup>	option group
<option>	option in a drop-down list
<p>	paragraph
<param>	parameter for an object
<pre>	preformatted object
<q>	short quotation
<samp>	sample computer code
<script>	script
<select>	selectable list
<small>	small text
	inline generic container
	strong text
<style>	style definition
<sub>	subscripted text
<sup>	superscripted text
<table>	table
<tbody>	table body
<td>	table cell
<textarea>	text area
<tfoot>	table footer
<th>	table header
<thead>	wraps row containing table headers
<title>	document title
<tr>	table row
	unordered list
<var>	variable

Enabling HTML 5

```
<!DOCTYPE HTML>
```

HTML in 90 seconds: Data Attributes

What the syntax looks like

in HTML tag

data-something = “anything”

```
<article
  id="electriccars"
  data-columns="3"
  data-indexnumber="12314"
  data-parent="cars">
...
</article>
```

How to access attributes with JavaScript

```
1
2       var attribute       = '[data-myAttribute]';
3       var elementNodeList = document.querySelectorAll(attribute);
4
5       for(var i = 0; i < elementNodeList.length; i++){
6           var element = elementNodeList[i];
7           var value = element.getAttribute(attribute.replace('[', '').replace(']', ''));
8           console.log(value);
9       }
```

AngularJS

The Story of AngularJS

AngularJS was created, as a side project, in 2009 by two developers, Misko Hevery and Adam Abrons.

Hevery eventually began working on a project at Google called Google Feedback. Hevery and 2 other developers wrote 17,000 lines of code over the period of 6 months for Google Feedback. However, as the code size increased, Hevery began to grow frustrated with how difficult it was to test and modify the code the team had written.

So Hevery made the bet with his manager that he could rewrite the entire application using his side project in two weeks. Hevery lost the bet. Instead of 2 weeks it took him 3 weeks to rewrite the entire application, but he was able to cut the application from 17,000 lines to 1,500 lines.

What is AngularJS?

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate’s much of the code you would otherwise have to write

What is AngularJS?

1. DOM has markup
2. Data is POJO
3. DI for modules

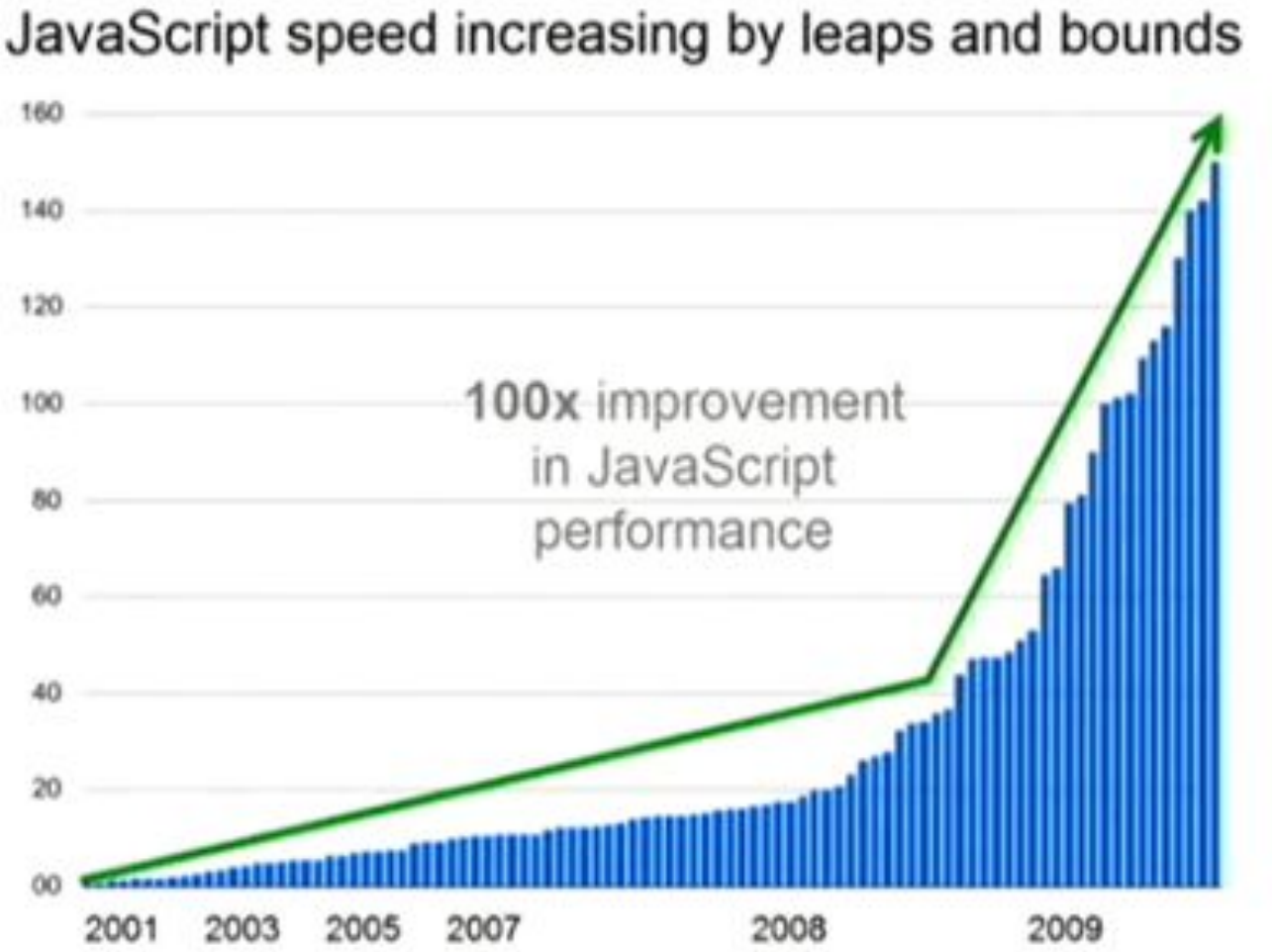
What it’s not

- It is not a JavaScript library. There are no functions to call directly like underscore.js
- It is not a DOM manipulation library, it actually uses a subset of JQuery called jqlite
- It is not just another tool to use, it is THE tool to use when one is using AngularJs
  - There is an angular way of doing things
  - Don’t mix and match this with other frameworks / DOM manipulation strategies
- It is not strictly a SPA (single page application) framework – in fact – you don’t need to use SPA concepts at all with it

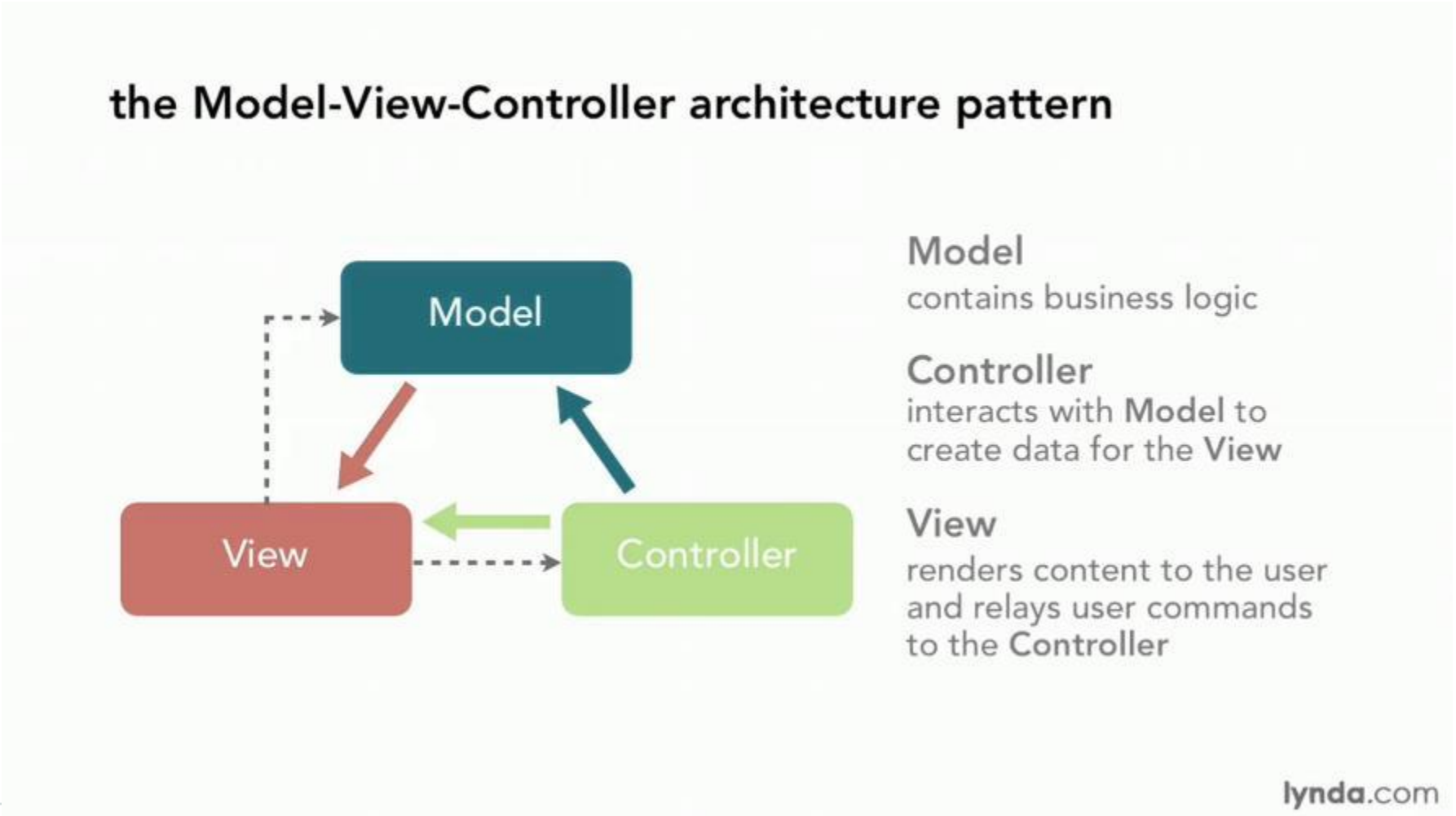
What AngularJS is trying to do

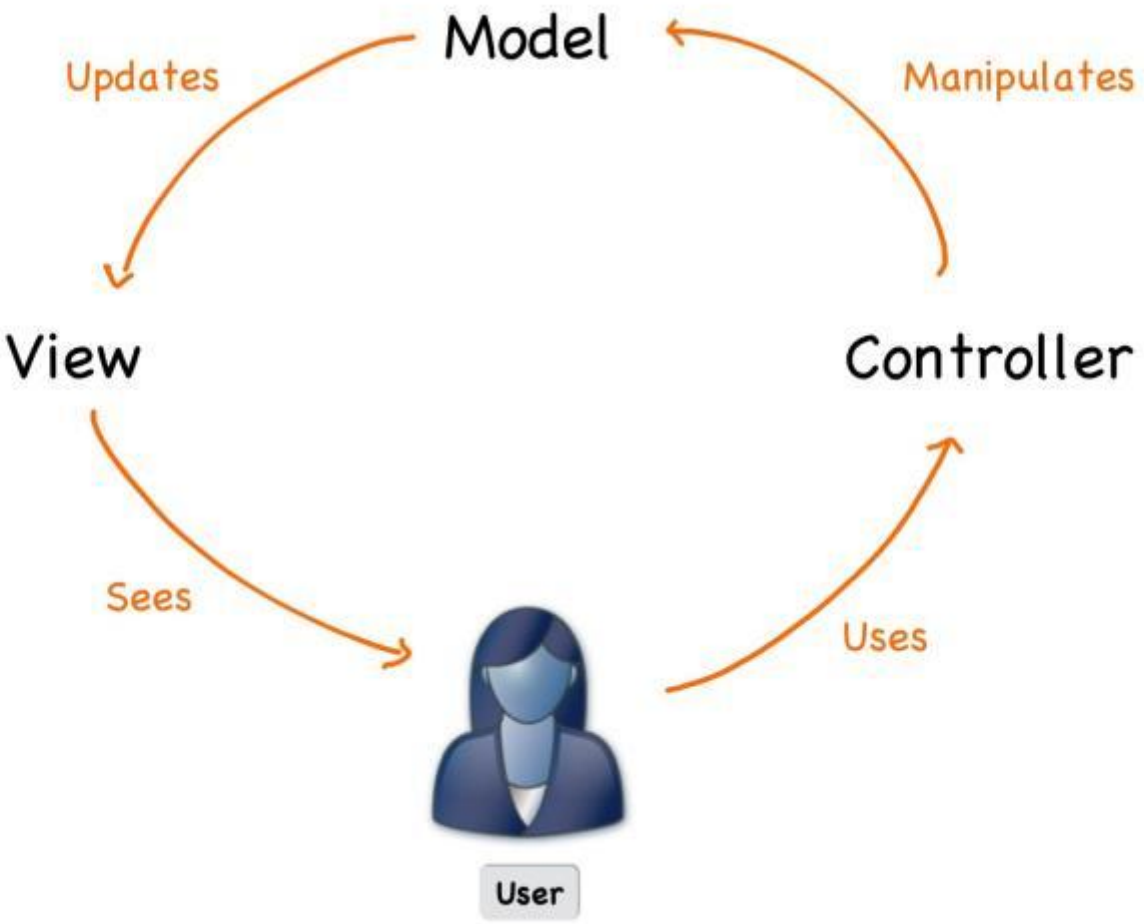
- Ease DOM complexity
- Reduce code bloat
- Framework for doing things
- Use client side processing
- MVC paradigm

Do more things on the client side

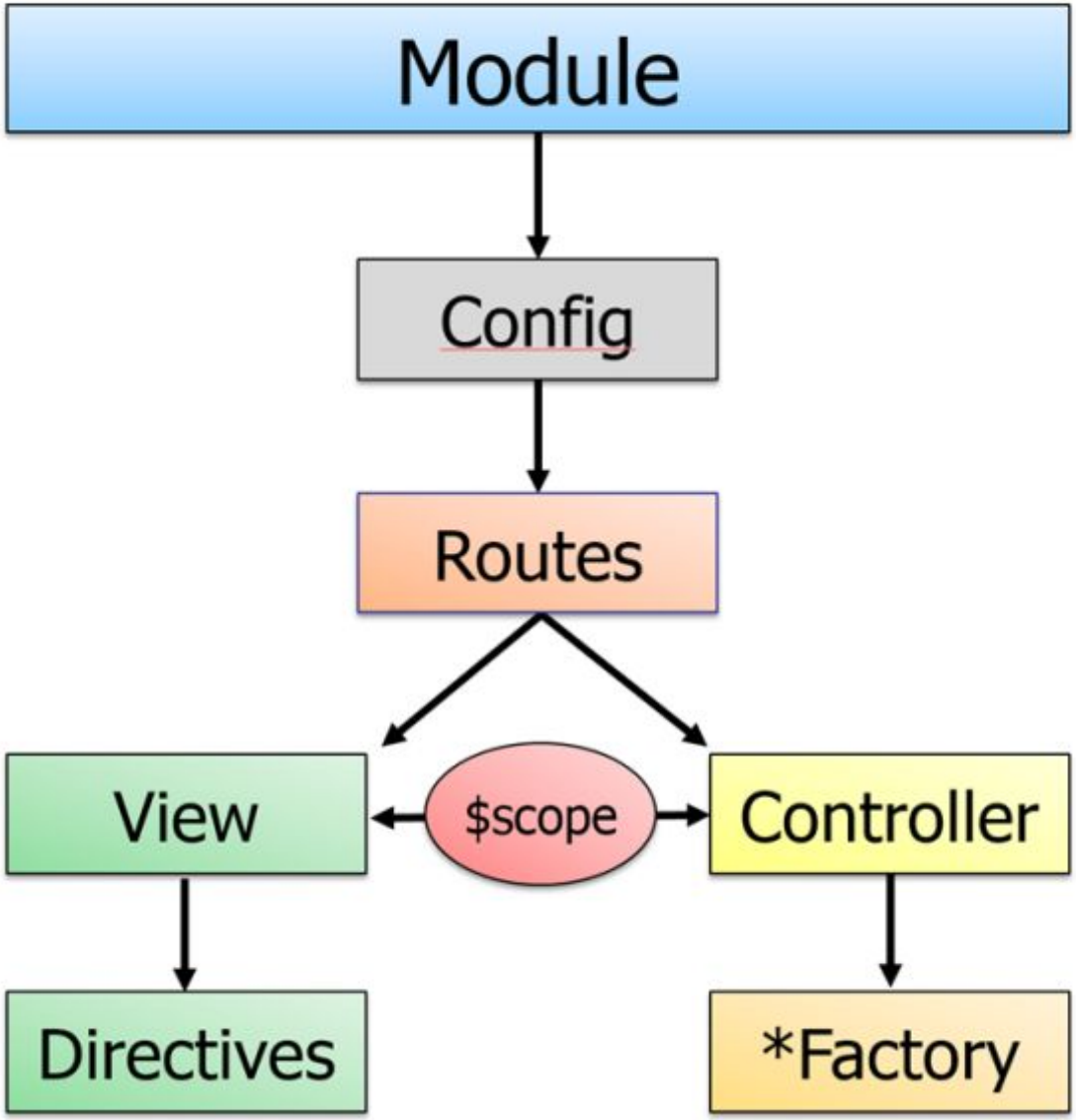


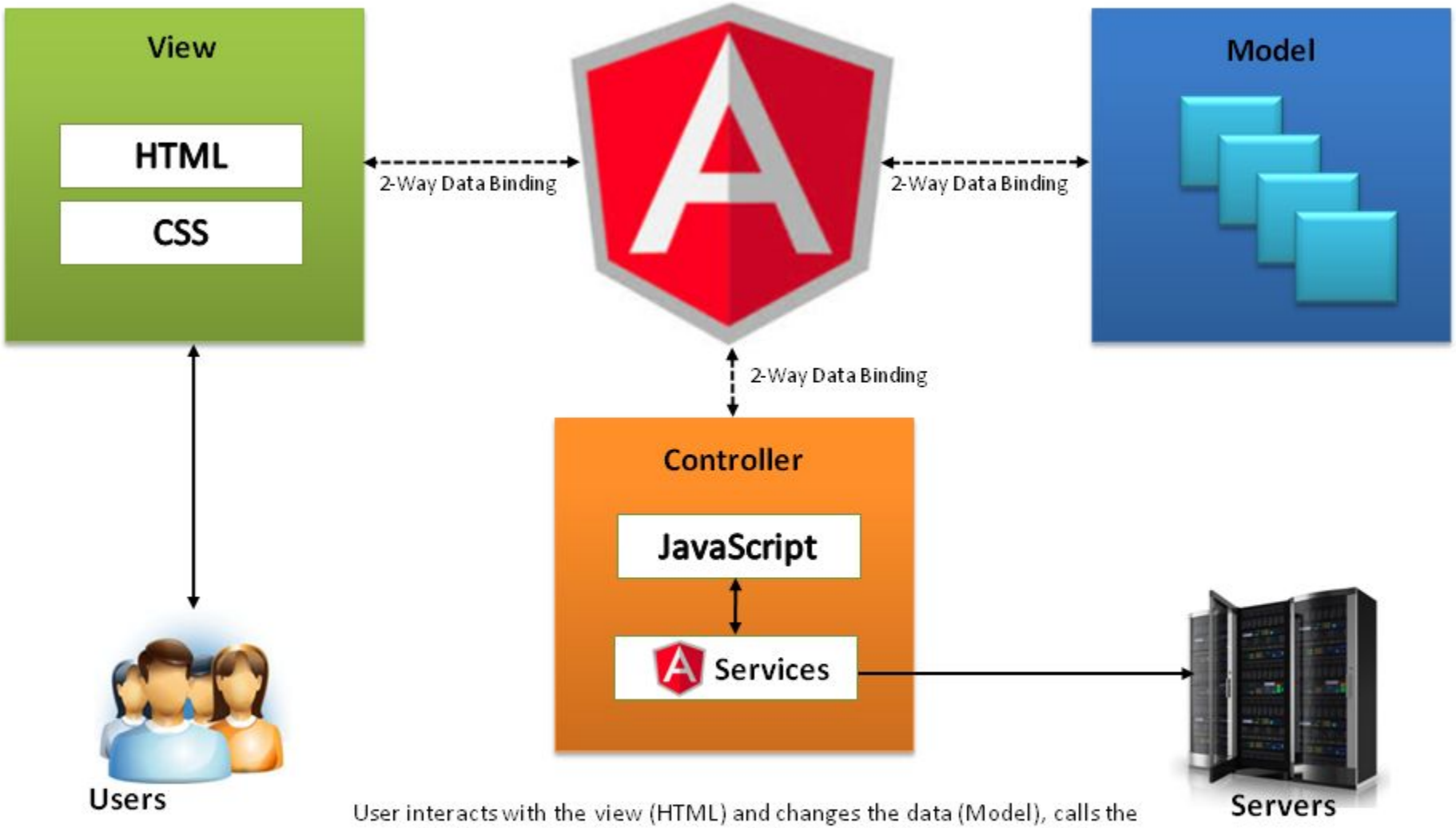
MVC





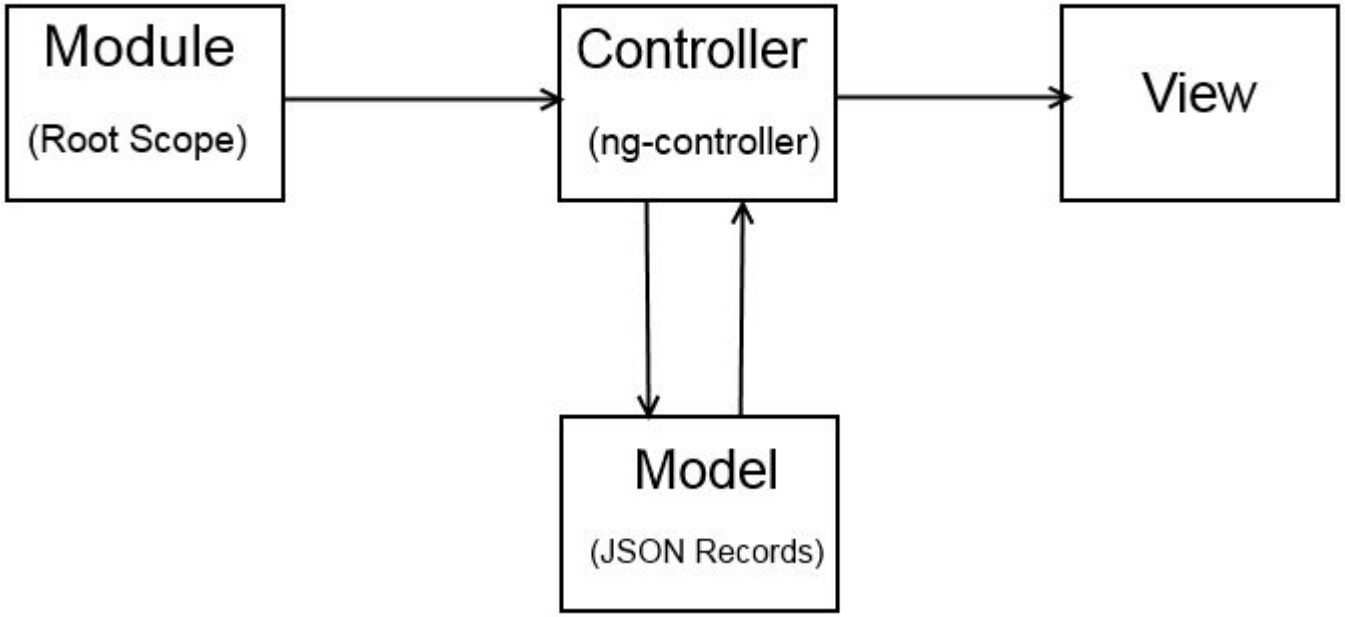
The Big Picture





User interacts with the view (HTML) and changes the data (Model), calls the controller (interaction), Controller modifies the Model, interacts with Servers via services and performs CRUD/Lookup operations on the data. AngularJS detects any model changes and updates the View via 2-way data binding.

The simpler picture



ENOUGH ALREADY… show me some angular

AngularJS via CDN

https://cdnjs.com/libraries/angular.js/

EXAMPLE

Exercise 1: hello world

1. Ng-app to start application on html tag

```
<html ng-app = "helloWorld">
```

2. Use CDN to get AngularJS

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.5.5/angular.min.js

3. Initialize your greeting

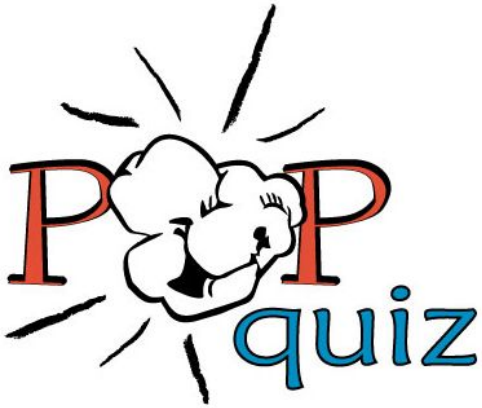
```
<body ng-init = "greeting = 'Hello World'">
```

4. Create your template string

```
<h1>{{greeting}}</h1>
```

5. Start your app

```
var app = angular.module('helloWorld', []);
```



- What happens if I don’t declare ng-app?
- What happens if ng-app is placed on the body instead of the html tag?
- What happens if I don’t include the script tag linking to angular.min.js?

What are directives?

Directives, for simplicity sake, are HTML 5 attributes that AngularJS attaches to DOM elements. These directives all start with “ng”:

```
<html ng-app = "MyApp">
```

Angular Directive

Key AngularJS Directives

Templates

- ng-disabled
- ng-cloak
- ng-hide
- ng-if
- ng-repeat
- ng-show
- ng-switch
- ng-view

Application

- ng-app
- ng-controller

Data Binding

- ng-bind
- ng-href
- ng-init
- ng-model
- ng-src
- ng-style

Forms

- ng-maxlength
- ng-minlength
- ng-pattern
- ng-required
- ng-submit

Behavior

- ng-blur
- ng-change
- ng-checked
- ng-click
- ng-key*
- ng-mouse*

Directives we'll talk about and use today:

ng-app
ng-init
ng-controller
ng-click
ng-model
ng-repeat

Controller Directive

HTML

```
<div ng-controller = "myController"></div>
```

JavaScript

```
var app = angular.module('myApplication')

app.controller('myController', function($scope){
    //my controller
});
```

View, Controllers & Scope

$scope is the "glue" (ViewModel) between a controller and a view

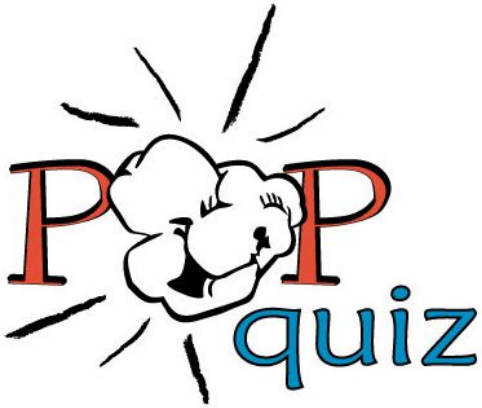
$scope Explained

```
$scope -> functions, variables, JSON, etc
```

$scope is really just an object that AngularJS uses to update the view and the model. It can store functions, variables, JSON, etc. We can place whatever we want on the $scope.

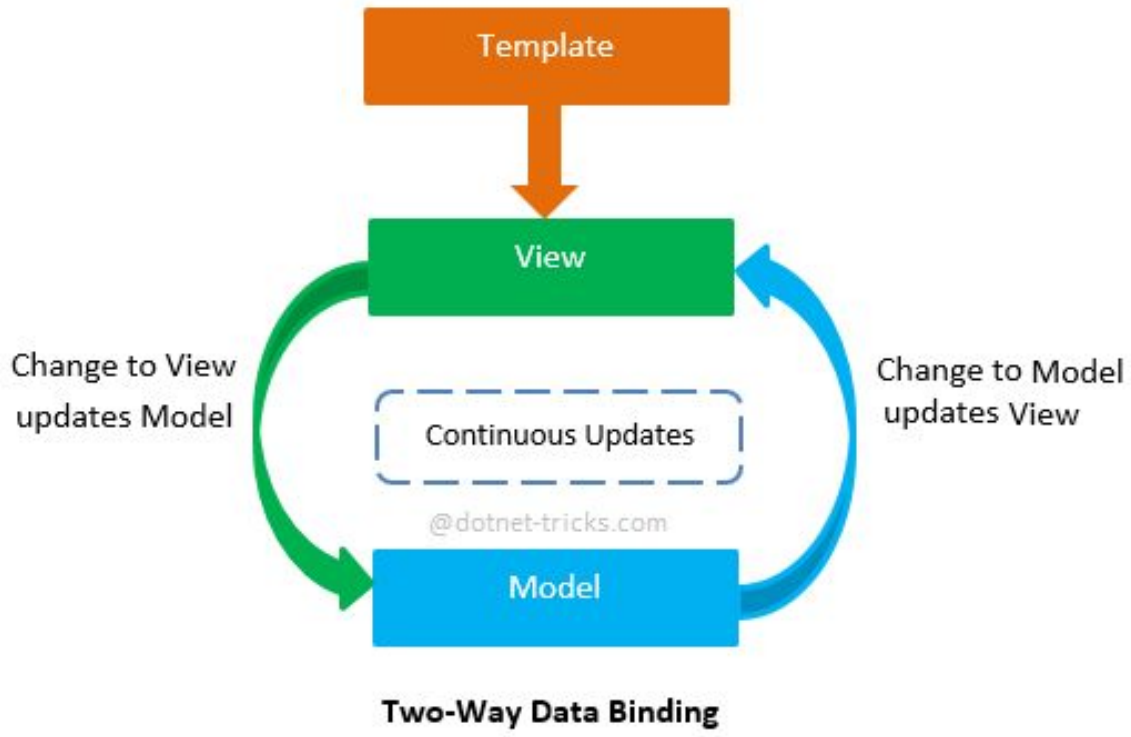
EXAMPLE

$scope and console.log



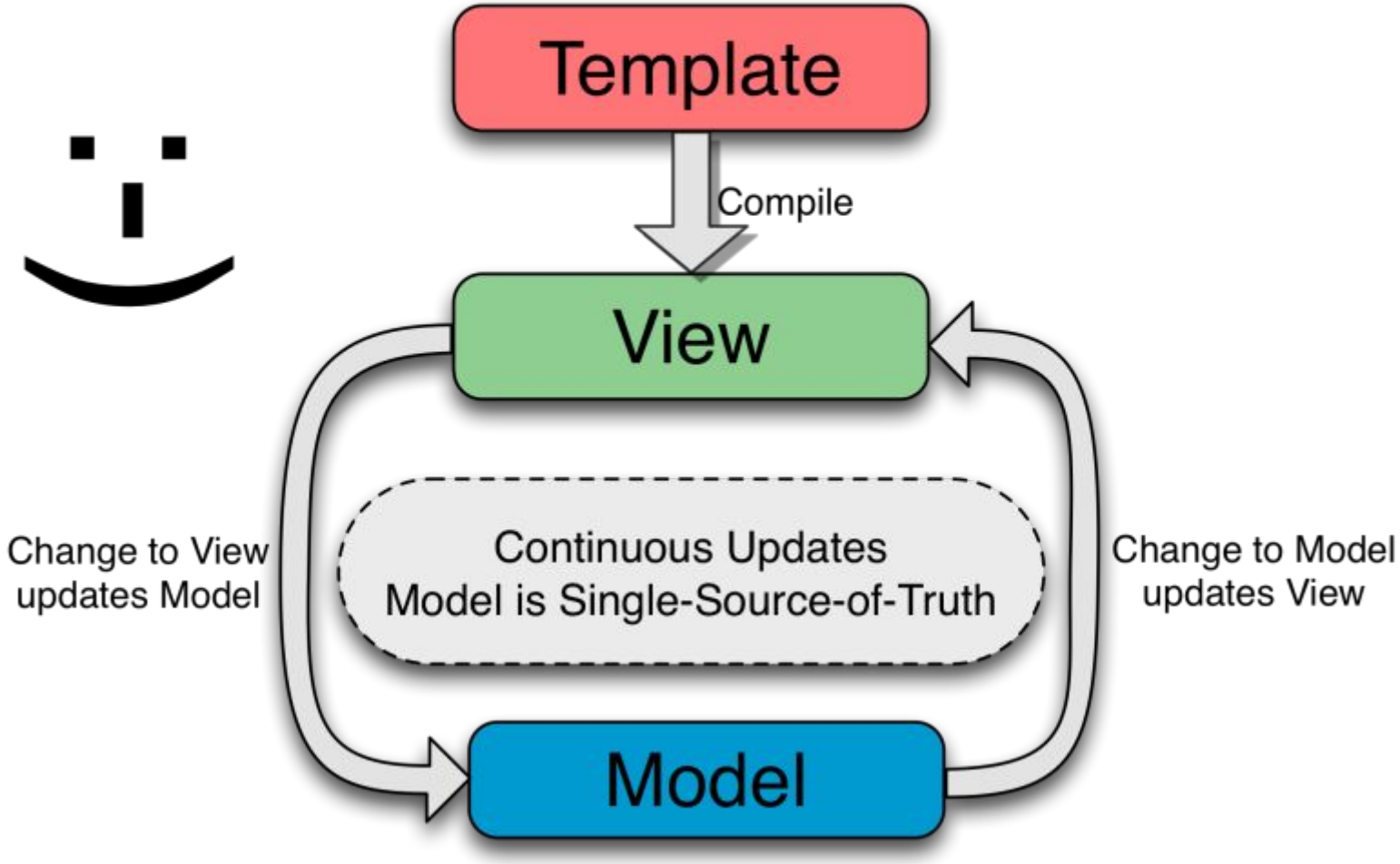
- What is the difference between the $scope and a directive?

Ng-model



Creates two way data-binding between the model and the view

Two-Way Data Binding

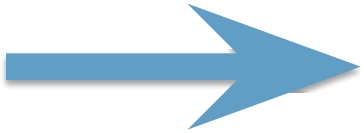


Ng-model

```
<input type = "text" ng-model = "myData"/>

{{myData}}
```

Ng-model



EXAMPLE

ng-model and input text

Can you provide an example of how to create a click event in HTML/JavaScript?

```
<button onclick = "myFunction()"></button>

<script type="text/javascript">
    var myFunction = function(){
        //my action
    };
</script>
```

Ng-Click

```
<button ng-click = "myFunction()">Click Me</button>

$scope.myFunction = function(){
    //my action here
};
```

EXAMPLE

Button with alert

EXAMPLE

buttonNumberExample.html

Exercise 2: Button

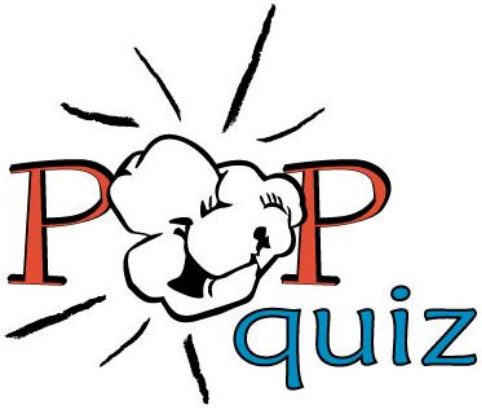
buttonExampleForm-exercise.html

1. Add ng-model to input tags

2. Add functionality to button's ng-click function to display text to screen

3. Store your results into $scope.displayText and watch AngularJS automatically update the view after clicking the button (data-binding in action)

4. Run the application



- What happens if I use onclick instead of ng-click?
- What happens if I don’t use ng-model to obtain the value in the input text box?

Ng-Repeat

Allows us to use a template of html and repeat it for every member in an array

Ng-Repeat

```
<ul ng-repeat = "item in list">
    <li>{{item}}</li>
</ul>
```

EXAMPLE

List-example.html

Exercise 3: List

Create list with array of objects

1. Create UL element with ng-repeat

```
<ul ng-repeat = "animal in animals">
```

2. Create an array of objects on $scope.animals

```
[{animal : 'Lion', sound : 'roar'}]
```

3. Create an <li> element with template for object

```
<li>{{animal.type}} {{animal.sound}}</li>
```

4. Start your app

Exercise 4: Tweeter

tweeter-exercise.html

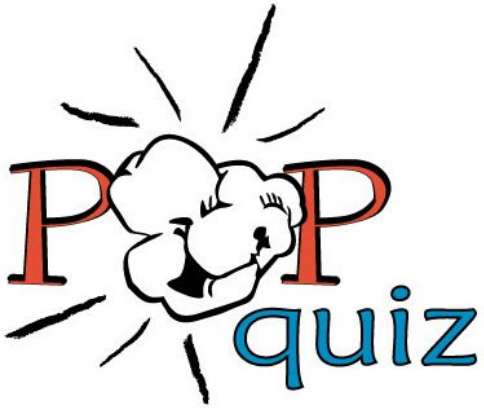
1. Add ng-repeat

```
<div id = "tweetTemplate" class="panel-body">
```

2. Ng-repeat for tweet array

Properties of object are text, retweets and likes

3. Start your app



- Which value is relative and which value is static?

```
<ul ng-repeat = "item in list">
    <li>{{item}}</li>
</ul>
```

- What happens if we provide ng-repeat a non-iterable element such as null?

Filters

Filters are added to AngularJS expressions or directives in order to transform the displayed data.

They are added using the pipe "|" character.

Example

```
<div ng-app="" ng-controller="appCtrl">
      <input type="number" ng-model="quantity">
      <input type="number" ng-model="itemPrice">
      <p>Total = {{ (quantity * itemPrice) | currency }}</p>
</div>
```

Result

Quantity: 2 Price: 12

Total = $24.00

EXAMPLE

filter-example.html

Exercise 5: Filters

filter-exercise.html

1. Add ng-repeat
2. Attach data from service to $scope
3. Add your filter using the ng-model
4. Run your application

EXAMPLE

report.html

Dependency Injection

Dependency Injection

Two notations to inject

- From parameter names in functions:

```
app.controller('SuperheroicController', function($scope){
    $scope.hello = 'world';
});
```

- Inline array notation:

```
app.controller('SuperheroicController', ['$scope', function(whatever)
{
    whatever.hello = 'world';
}]);
```

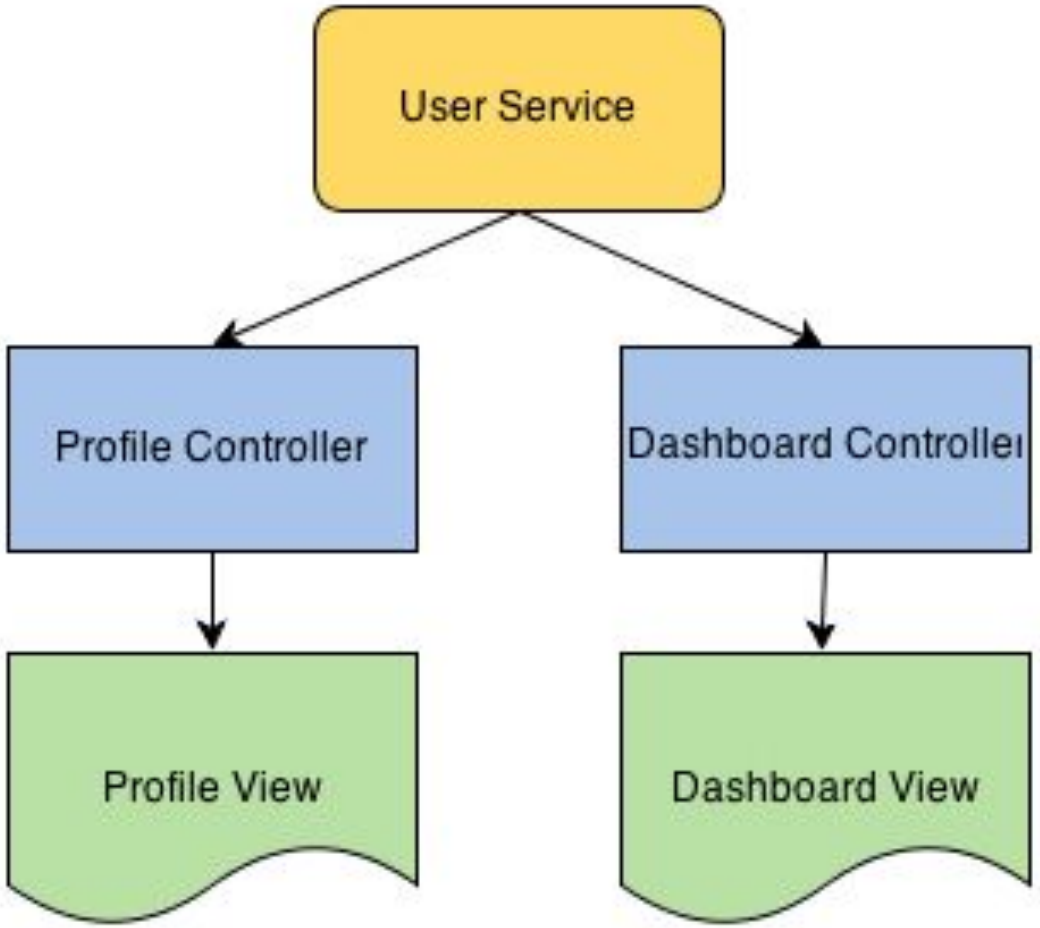
T3DD14

sunzinet

mitchell

Angular Services

- Angular services are singletons that carry out specific tasks.
- All services in Angular are instantiated lazily.
- There are three functions for creating generic services, with different levels of complexity and ability:
  - Service
  - Factory
  - Provider



Exercise 6: Services

Calculator.html

1. Dependency inject service to controller in controller.js

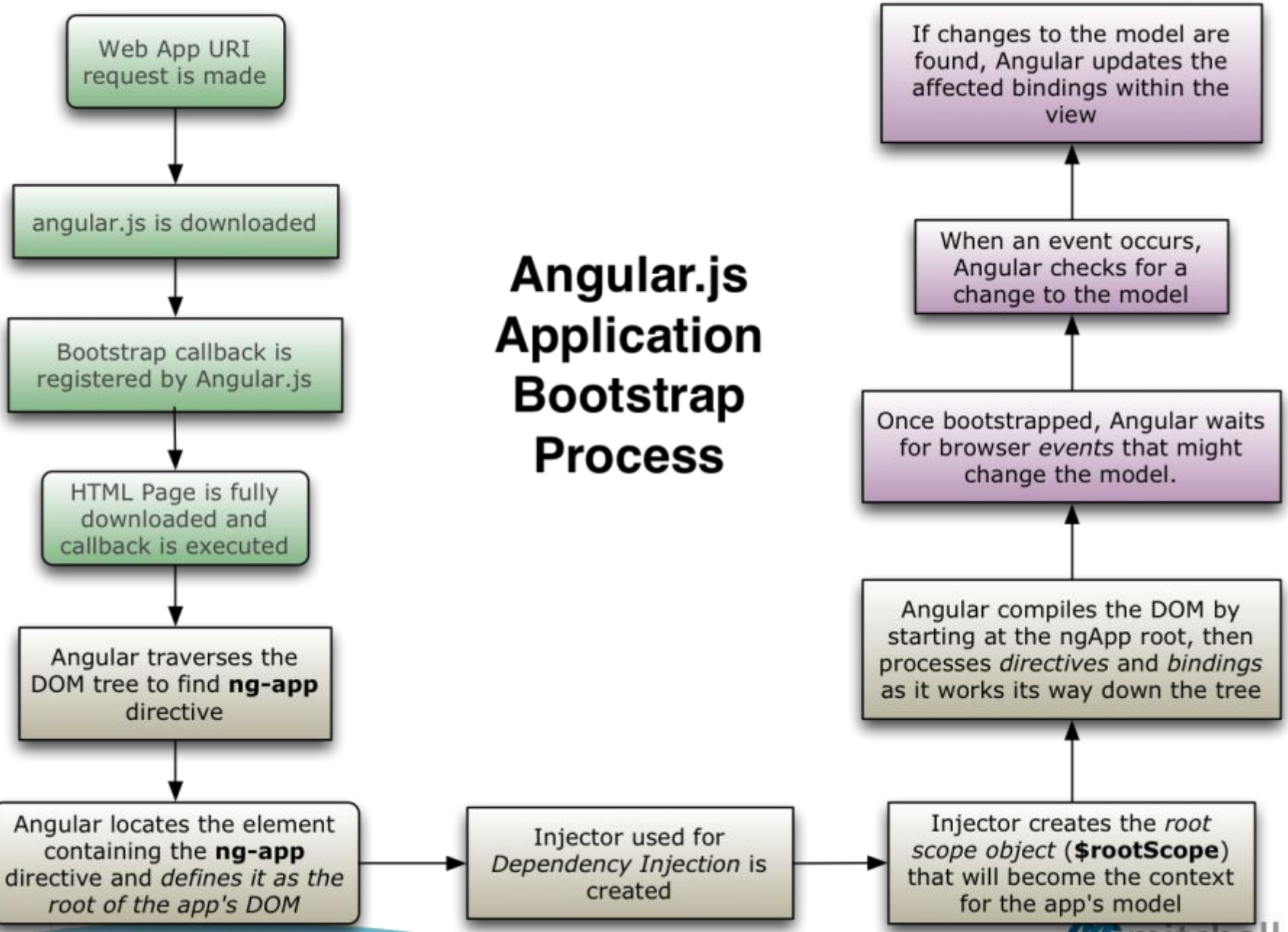
2. Create methods in math.js in service for:

   multiplication, addition, subtraction, division

3. Start your app

4. Test your controller

Angular.js Application Bootstrap Process



Pros and Cons

AngularJS 1.x advanced topics

- $http, $q and promises
- Factory vs providers vs services
- Angular routing and SPA
- Angular custom directives
- components
- Unit Testing

Angular 2.0

- Angular 2.0 != Angular 1.x
- New syntax
- Creates components with ES6 classes

AngularJS 1.x	AngularJS 2.0
• Create Module • Declare ng-app • Create Controller • Attach items to $scope • Declare Controller • Create template	• Create component • Create template • Bootstrap • Transpilation

AngularJS Documentation

https://docs.angularjs.org/api

Final Project: Telephone Book

Create a simple phone directory that you can search for your friends and it will find their phone #’s

Must use ng-repeat
Must use ng-model
Must use filter

Bonus: Create a service that you call to store and get the phone numbers

(m)powering better outcomes