ANGULARJS

DATA RECAP

Data from servers: $http and AJAX

Managing data on the front-end: MVC, Client-side Templating, Data Binding

JavaScript Frameworks

MOTIVATION

call-back style programming associated with AJAX apps is hard to maintain/test

non-trivial to get the data into the correct state, both in the View and in the Model

Frameworks do the heavy lifting!

Bring structure and organization

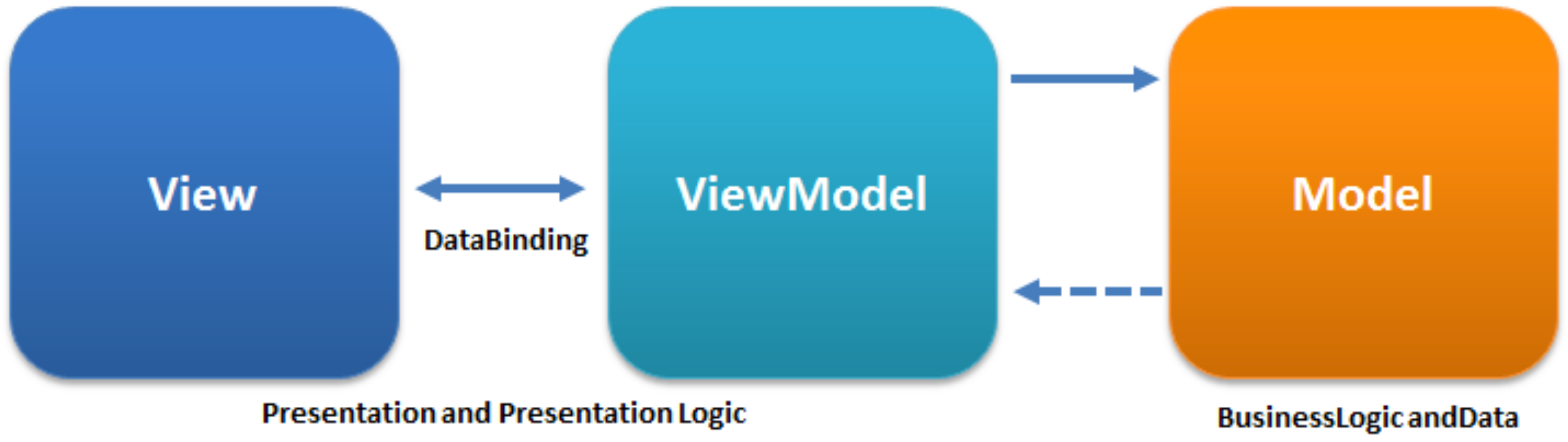
Scaffolding (*e.g.*, data binding) means fewer lines of code

MODEL VIEW CONTROLLER (MVC)

Separation between

Model	managing data
Controller	application logic
View	presenting the data

MV*



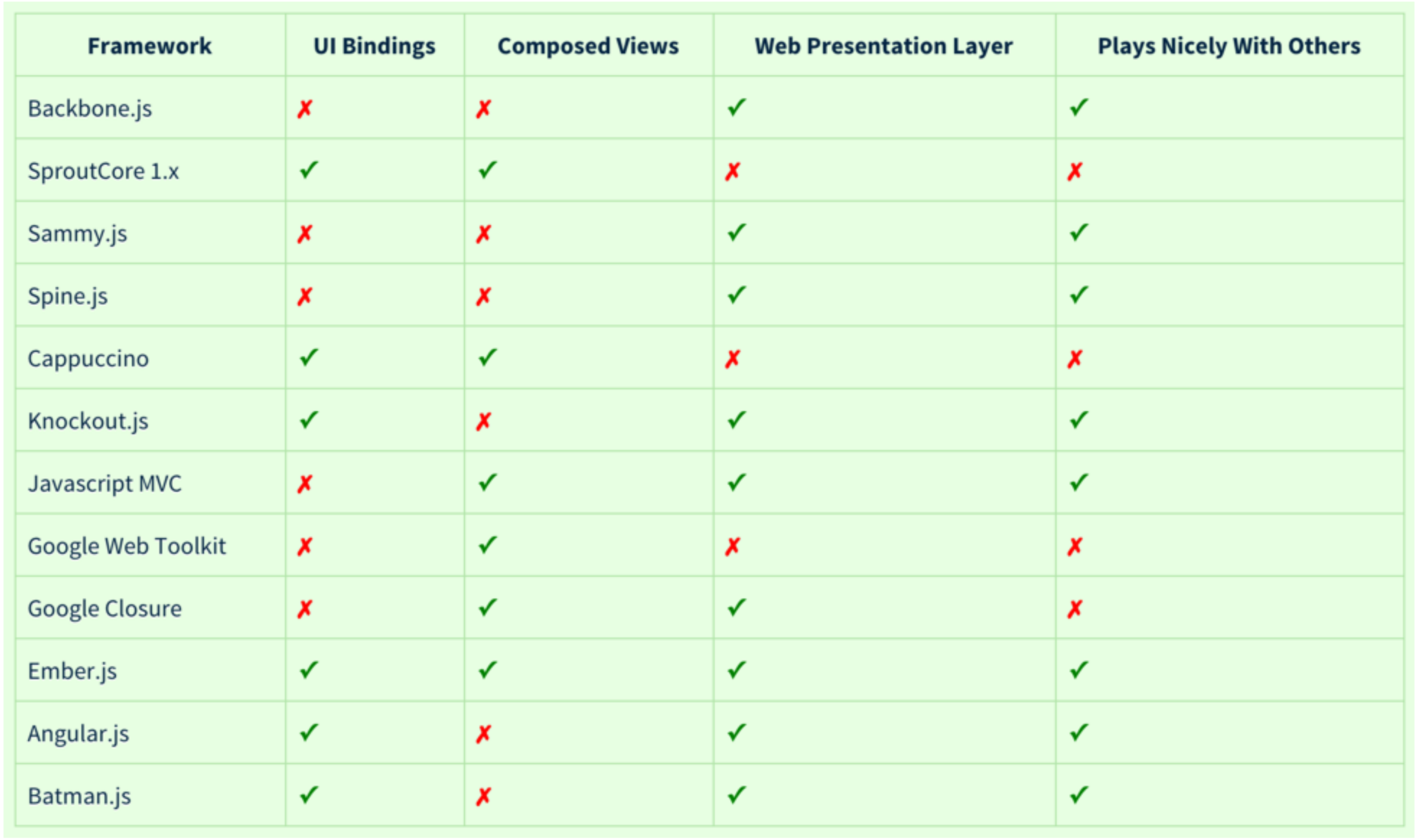
MVC vs MVVM vs MVP?

Model View Whatever: model and a view and something that's managing how they interact

en.wikipedia.org/wiki/Model_View_ViewModel

So many frameworks, so little time…

www.smashingmagazine.com/2012/07/27/journey-through-the-javascript-mvc-jungle/



Framework	UI Bindings	Composed Views	Web Presentation Layer	Plays Nicely With Others
Backbone.js	✗	✗	✓	✓
SproutCore 1.x	✓	✓	✗	✗
Sammy.js	✗	✗	✓	✓
Spine.js	✗	✗	✓	✓
Cappuccino	✓	✓	✗	✗
Knockout.js	✓	✗	✓	✓
Javascript MVC	✗	✓	✓	✓
Google Web Toolkit	✗	✓	✗	✗
Google Closure	✗	✓	✓	✗
Ember.js	✓	✓	✓	✓
Angular.js	✓	✗	✓	✓
Batman.js	✓	✗	✓	✓

codebrief.com/2012/01/the-top-10-javascript-mvc-frameworks-reviewed/

Angular JS

ANGULAR CONCEPTS

1.3

Controllers

`$scope` object

Directives

MVC
Client-side Templating
Data Binding

Templates and Data Binding

CLIENT-SIDE TEMPLATING

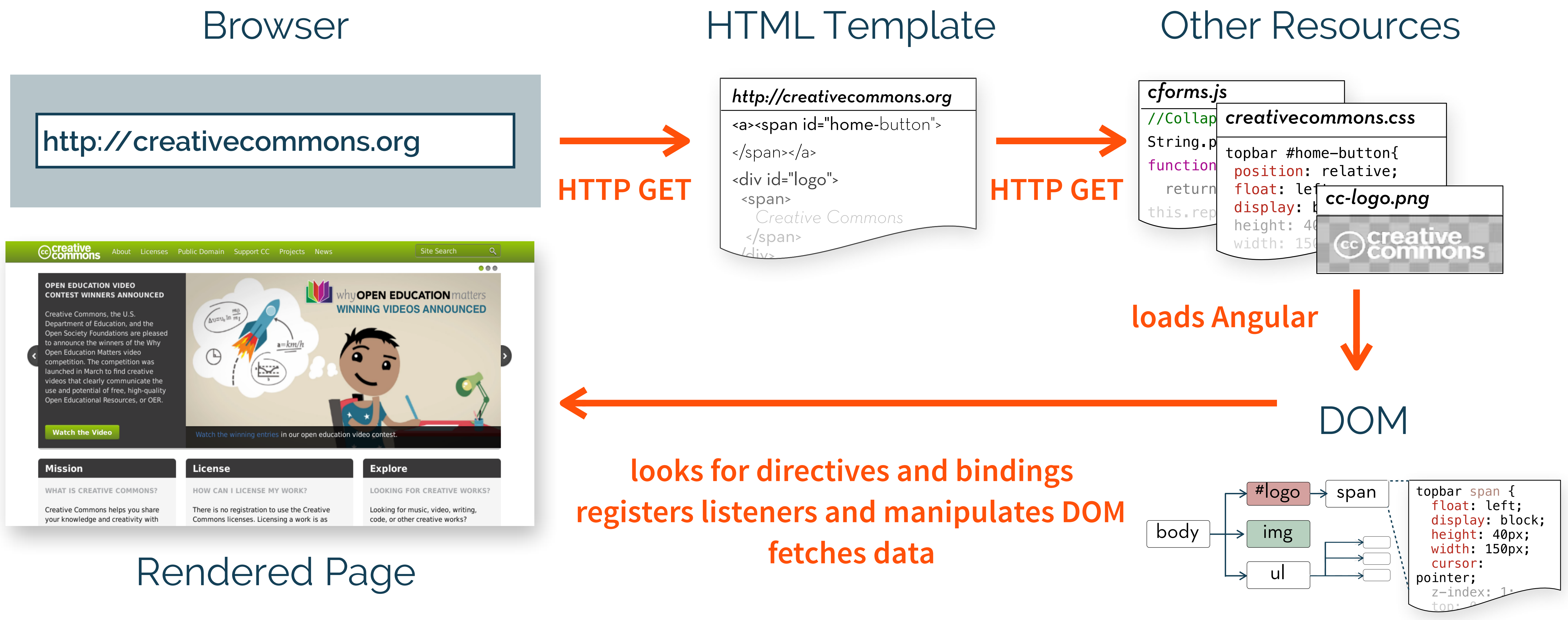
Cacheable

Templates are HTML documents which define the UI

Loaded from server like any other static resource

Once in the browser, Angular merges template with data, defining a **view**

ANGULAR AT WORK



```
http://creativecommons.org
<a><span id="home-button">
</span></a>
<div id="logo">
  <span>
    Creative Commons
  </span>
</div>
```

```
cforms.js
//Collap
String.p
function
  return
this.rep
```

```
creativecommons.css
topbar #home-button{
 position: relative;
 float: lef
 display: b
 height: 40
 width: 150
```

```
cc-logo.png
```

```
topbar span {
  float: left;
  display: block;
  height: 40px;
  width: 150px;
  cursor:
pointer;
  z-index: 1;
  top: 0
```

creative commons   About   Licenses   Public Domain   Support CC   Projects   News   Site Search

OPEN EDUCATION VIDEO CONTEST WINNERS ANNOUNCED

Creative Commons, the U.S. Department of Education, and the Open Society Foundations are pleased to announce the winners of the Why Open Education Matters video competition. The competition was launched in March to find creative videos that clearly communicate the use and potential of free, high-quality Open Educational Resources, or OER.

Watch the Video

why OPEN EDUCATION matters
WINNING VIDEOS ANNOUNCED

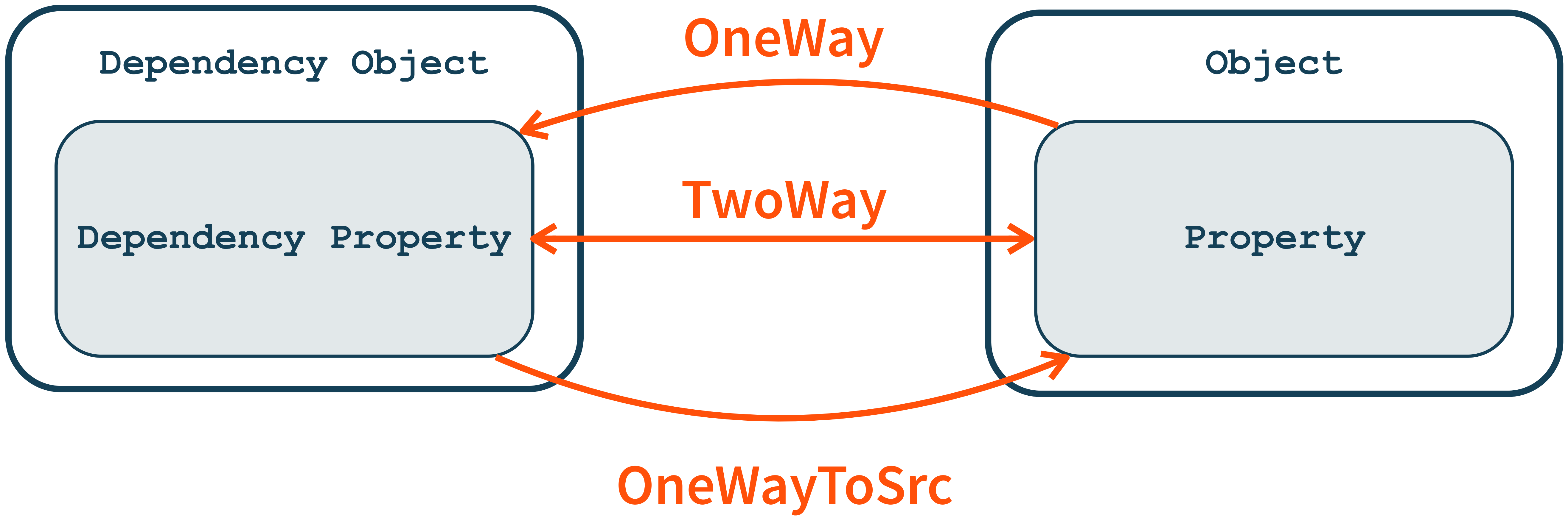
Watch the winning entries in our open education video contest.

Mission
WHAT IS CREATIVE COMMONS?
Creative Commons helps you share your knowledge and creativity with

License
HOW CAN I LICENSE MY WORK?
There is no registration to use the Creative Commons licenses. Licensing a work is as

Explore
LOOKING FOR CREATIVE WORKS?
Looking for music, video, writing, code, or other creative works?

WAYS OF DATA BINDING



TEMPLATE DATA BINDING

`{{ }}` interpolation syntax; one-way binding

`ng-bind` one-way binding for displaying text

`ng-model` two-way binding for form elements

MADLIBS TEMPLATE

```
<div ng-app>
  <div ng-controller='MadlibsController'>
    <div>Hola
      <span class="madlib">{{madlibs.animal}}</span>,
    </div>
    <div>Se llama
      <span class="madlib" ng-bind="madlibs.name"></span>!
    </div>
    <form>
      <input ng-model="madlibs.name">
    </form>
   </div>
</div>
```

CODEPEN

MADLIBS VIEW

Hola Llama,
Se llama Francesca!

Francesca

CODEPEN

Controllers and $scope

CONTROLLERS

Javascript classes which do the following:

Initialize state in your application’s model

Expose model and functions to the view through **`$scope`**

Watch parts of the model for changes and take action

`$scope`

mechanism used to expose **model** data to **view**

any data that you want exposed in the view should be assigned to the `$scope` object

MADLIBS JS

```
function MadlibsController($scope) {

  $scope.madlibs = {animal: ‘Llama',
  name:'Francesca'};

}
```

CODEPEN

MANAGING CONTROLLERS

One controller per functional area

Tied to a specific piece of DOM:

`ng-controller`

views — dynamically loaded template fragment via **routes**

Nesting: child controller has access to parent controller's **`$scope`**

`$scope.$watch`

```
$watch(watchFn, watchAction, deepWatch)
```

watches parts of the model and takes action when model changes

`$scope.watch`

```
$watch(watchFn, watchAction, deepWatch)
```

Angular expression or a function that returns the current value of the model to watch

evaluated multiple times

shouldn’t have side effects or be an expensive computation

`$scope.watch`

```
$watch(watchFn, watchAction, deepWatch)
```

function or expression to be called when **`watchFn`** changes

`$scope.watch`

```
$watch(watchFn, watchAction, deepWatch)
```

optional boolean parameter

examine each property within a watched object or each element within a watched array for changes

AVG HEIGHT JS

```
function AnimalController($scope) {
  $scope.camelids = {};
  $scope.camelids.data = [{"name": "llama", "height": 1.8}, {"name": "alpaca",
"height": 0.9},{"name": "vicuna", "height": 0.8}];

  $scope.computeAvgHeight = function() {
    var height = 0;
    for (var i=0; i<$scope.camelids.data.length;i++) {
          height += parseFloat($scope.camelids.data[i].height);}
    height = height/$scope.camelids.data.length;
    $scope.camelids.avgHeight = Number((height).toFixed(1));
  }

  $scope.$watch('camelids.data',$scope.computeAvgHeight,true);
}
```

CODEPEN

Directives

DIRECTIVES

DOM transformation engine that lets you extend HTML's syntax to create your own custom tags, attributes, etc.

`{{}}, ng-*`: predefined directives that come with Angular

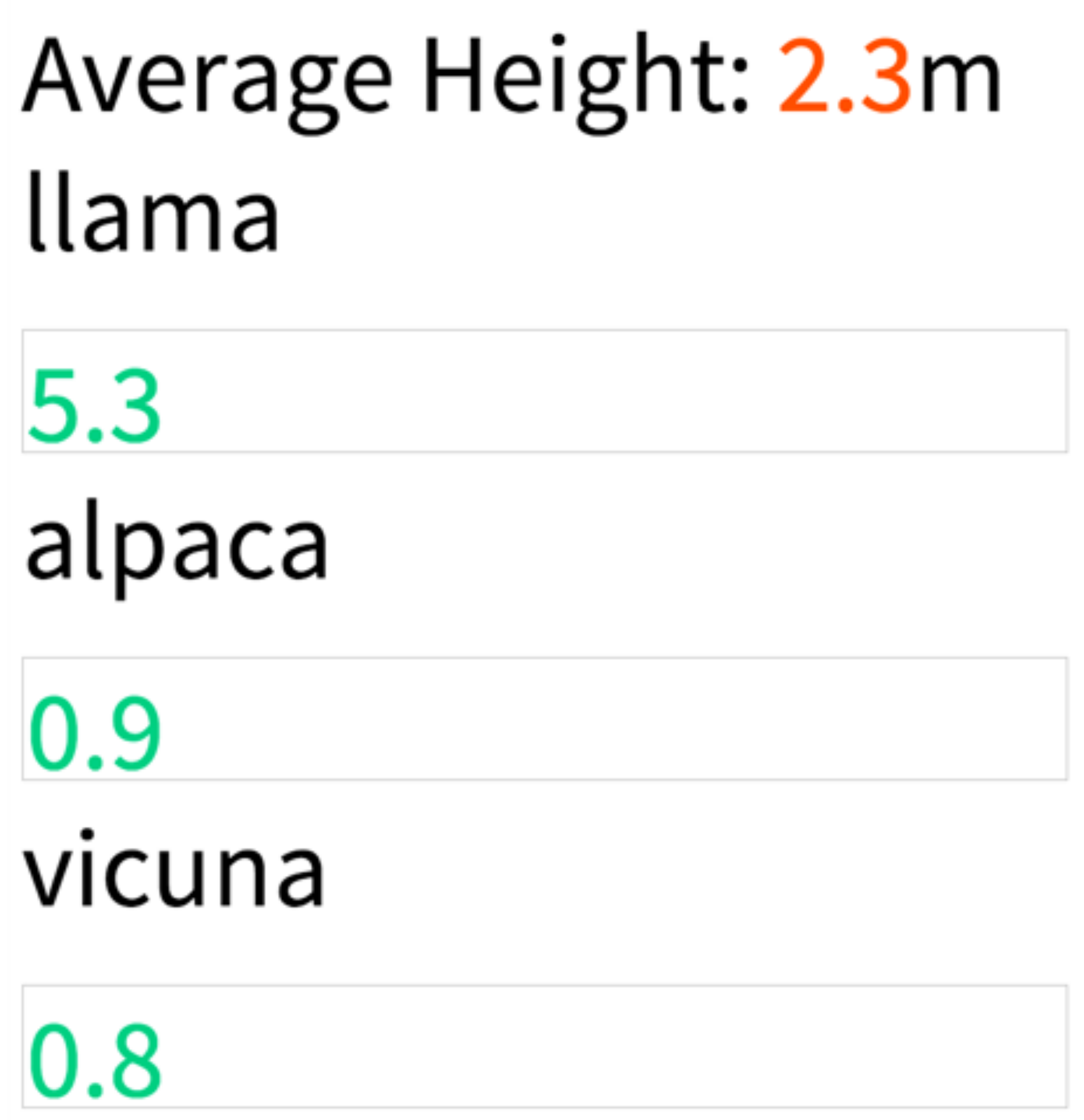
`ng-repeat`

AVG HEIGHT TEMPLATE

```
<div ng-app>
  <div ng-controller='AnimalController'>
    <div>Average Height:
      <span class="avgheight" ng-bind="camelids.avgHeight"></span>m
    </div>
    <div ng-repeat="camelid in camelids.data">
      <span ng-bind="camelid.name"></span>
      <form><input ng-model="camelid.height"><form>
    </div>
   </div>
</div>
```

CODEPEN

AVG HEIGHT VIEW



Average Height: 2.3m

llama

5.3

alpaca

0.9

vicuna

0.8

CODEPEN

EVENT-HANDLING WITH DIRECTIVES

```
ng-eventhandler = ‘expression’
```

```
<div ng-click = “doSomething()”>
```

behave in the same way in every browser

not in global namespace — $scope defined by element’s controller

`ng-hide/ng-show`

for displaying and hiding based on expression on the right

toggles between default **`display`** and **`display:none`**

`ng-class/ng-style`

expression on the right can be one of the following:

A string representing space-delimited class names

An array of class names

A map of class names to boolean values

`ng-src/ng-href`

use with `<img>/<a>` when dynamically loading paths

otherwise, you might see a broken link: browser tries to display before the data binding has happened

CODEPEN

ANGULAR EXPRESSIONS

```
ng-* = 'angularExpression'
```

Supports simple math, comparisons, boolean logic

No loops, conditionals

undefined and null don't throw errors

use model state that hasn't been initialized

Angular 2.0

www.youtube.com/watch?v=9IBljKq4XrQ

ANGULAR CONCEPTS

$\text{2.0} \rightarrow$

~~Controllers~~

~~$scope object~~

~~Directives~~

no backwards compatibility!

Angular 1.3

```
<div ng-controller=“SantaTodoController”>
  <input type="text" ng-model="newTodoTitle">
  <button ng-click="addTodo()">+</button>

  <tab-container>
    <tab-pane title="Tobias">
      <div ng-repeat="todo in todosOf('tobias')">
        <input type=“checkbox" ng-model="todo.done">
        {{todo.title}}
        <button ng-click="deleteTodo(todo)">
          X
        </button>
       </div>
```

Angular 2.0

```
<div>
  <input type="text" [value]="newTodoTitle">
  <button (click)="addTodo()">+</button>
  <tab-container>
    <tab-pane title="Good kids">
        <div [ng-repeat|todo]="todosOf('good')">
            <input type="checkbox" [checked]="todo.done">
            {{todo.title}}
            <button (click)="deleteTodo(todo)">
            X
            </button>
        </div>
```

what has changed?

Why?!

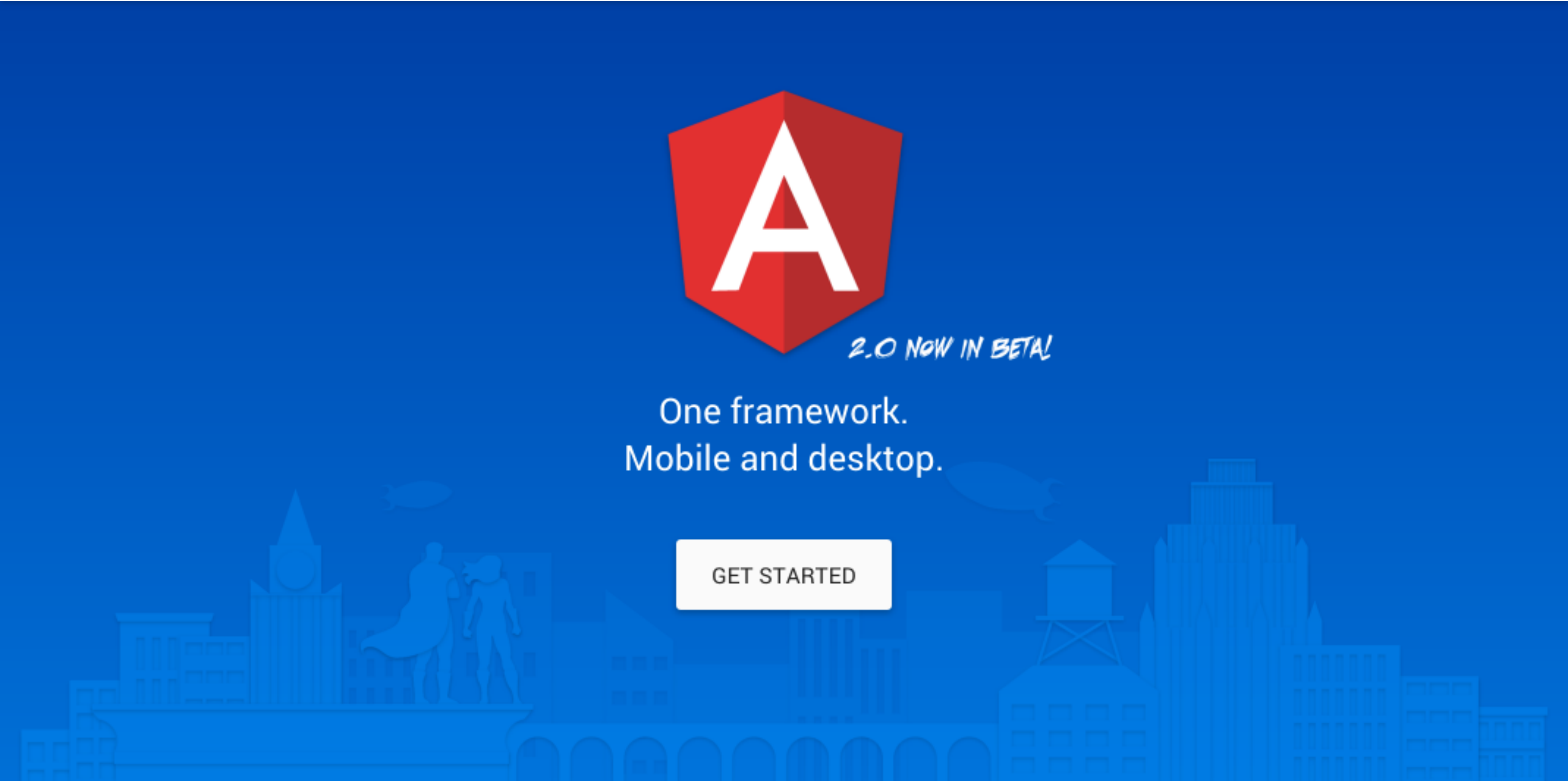
make Angular faster and more logical

work directly with new versions/features of web technologies:

work with DOM directly

get rid of controllers and directives in favor of HTML5 web components

adoption of the very latest JavaScript and browsers



angular.io

Bring your laptops!

NEXT CLASS: LAB

courses.engr.illinois.edu/cs498rk1/