Angular 1.6

Front end interfaces: a new day has dawned

Objectives

- **Explain** what frontend frameworks are.
- **Explain** what single page applications are.
- **Demonstrate** how a frontend framework integrates with a backend
- **Show** where Angular 1.6 fits into the ecosystem of other web frameworks.

Separation of Concerns

Database

Persists the data

PostgreSQL, Oracle, etc

Backend

Creates an API, Auth+Auth

JS/Node, Java/Spring, etc

Frontend

UI on the browser. Rewrites DOM in place.

Angular 1, Angular 2, React

And a lot of JSON thrown into the mix!

Single Page Apps

- Move the UI code to the front end into its own app.
- Can distribute front end code on a fast CDN.
- Running all in the browser makes a more responsive experience for the UI.
- Communicates with the backend(s) using JSON APIs.

Frameworks

- **Angular < 1.6:** Used in many applications. Currently the most popular.
- **Angular 1.6:** Best of both worlds. Component based architecture similar to…
- **Angular 2, Angular 4:** Different than the Angulars above
- **React, Vue, etc:** Other competitors.

Data Binding Basics

Objectives

- **Enable** Angular 1.6 on your web page.
- **Set** state on a controller via form fields
- **Access** state on a controller via expressions

Turn it on!

```
<script
type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.0/angular.js">
</script>
```

Grab Angular from your local filesystem or a CDN.

```
<html ng-app=”app”>
```

And now the magic happens.

What’s that ng-thing?

`<html ng-app=”app”>`

- **ng-** is the name for attributes that do Angular things.
- **ng-app:** tells Angular to take over the DOM at that point.
- **“app”** tells angular to use the module “app” as the app

```
<html ng-app=”app”>
```

Allows Angular to take over your DOM and do its magic.

Angular two-way binding

- The **controller** in the component holds variables (here one is included behind the scenes by default)

- The **view** has the input and output components.

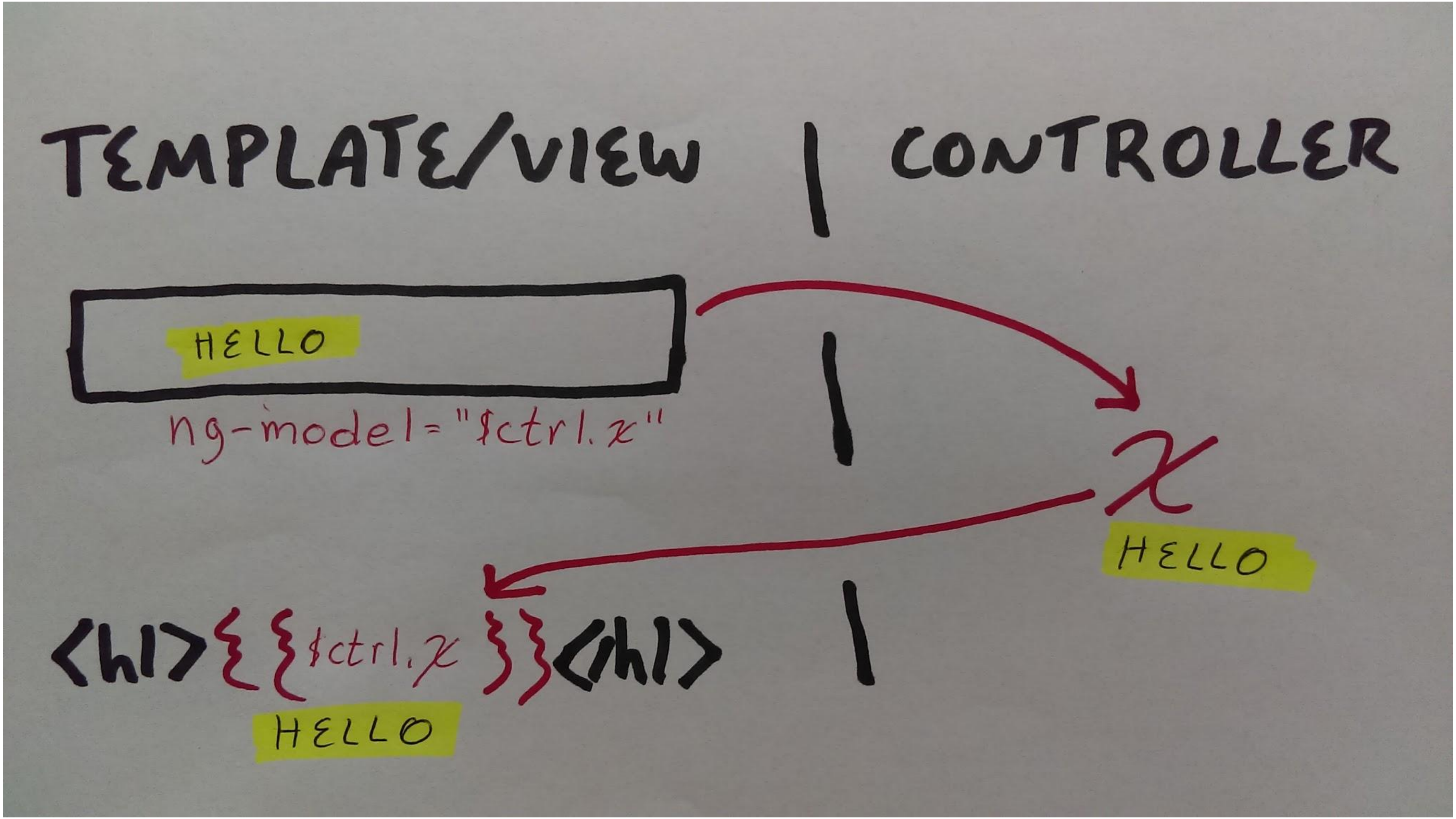
- When the UI components are bound to the

```
{{ $ctrl.✗}}
```

Accesses state on controller to display on **OUTPUT** elements of the view as specified by the template. ✗ is whatever your variable name is.

```
ng-model=”$ctrl.✗”
```

Sets state on controller from **INPUT** elements of the view as specified by the template. ✗ is whatever your variable name is.



TEMPLATE/VIEW
CONTROLLER
HELLO
ng-model="$ctrl.x"
x
HELLO
<h1>{{$ctrl.x}}</h1>
HELLO

Coding demo