AngularJS ControllerAs

Learning Objectives

Students understand and explain what controllerAs is and why it's best practice.
Students demonstrate the effective use of controllerAs.
Students understand and demonstrate how bindToController works and its benefits when used along controllerAs.

Resources

https://toddmotto.com/digging-into-angulars-controller-as-syntax/
http://blog.thoughtram.io/angularjs/2015/01/02/exploring-angular-1.3-bindToController.html
http://campus.codeschool.com/courses/shaping-up-with-angular-js/intro
Monthly Meetup: http://www.meetup.com/AngularJS-Utah/events/226023776/
http://www.freecodecamp.com/

ControllerAs Keypoints

What:

- Syntax to use controllers throughout our apps the smart way

Why:

- Makes scoping easier to use and less error prone
- Makes the application more testable
- It is best practice

Syntax

```
// we declare as usual, just using the `this` Object instead of `$scope`
myModule.controller('MainCtrl', function() {
  this.title = 'Some title';
});

// and in the DOM
<div ng-controller="MainCtrl as main">
  {{ main.title }}
</div>
```

Why is it best practice?

It gets rid of stuff like this:

```
// confusing to use and one scope will take precedence over another
<div ng-controller="MainCtrl">
  {{ title }}
  <div ng-controller="AnotherCtrl">
    {{ title }}
    <div ng-controller="YetAnotherCtrl">
      {{ title }}
    </div>
  </div>
</div>
```

And like this:

```
// works but it is bad practice to use $parent
<div ng-controller="MainCtrl">
  {{ title }}
  <div ng-controller="AnotherCtrl">
    {{ $parent.title }}
    <div ng-controller="YetAnotherCtrl">
      {{ $parent.$parent.title }}
    </div>
  </div>
</div>
```

By replacing it with:

```
// far clearer and variables can be accessed properly across scopes
<div ng-controller="MainCtrl as main">
  {{ main.title }}
  <div ng-controller="AnotherCtrl as another">
    {{ another.title }}
    <div ng-controller="YetAnotherCtrl as yet">
      {{ yet.title }}
    </div>
  </div>
</div>
```

Can be used wherever a controller is used

```
$routeProvider.when('/', {
  templateUrl: 'stateTemplate.html',
  controllerAs: 'ctrl',
  controller: 'StateController'
});
```

```
app.directive('someDirective', function () {
  return {
    restrict: 'A',
    controller: 'SomeController',
    controllerAs: 'ctrl',
    template: '{{ctrl.foo}}'
  };
});
```

BindToController

Does exactly what it says. When set to true in a directive with isolated scope that uses controllerAs, the component’s properties are bound to the controller rather than to the scope.

Syntax

```
app.directive('someDirective', function () {
  return {
    scope: {},
    bindToController: {
      someObject: '=',
      someString: '@',
      someExpr: '&'
    }
    controller: function () {
      this.name = 'mario';
    },
    controllerAs: 'ctrl',
    template: '<div>{{ctrl.name}}</div>'
  };
```

```
    });
```