**2.2 Scope-a-dope 250 pts**

When using the controllerAs syntax, the controller's context (this) attaches things to the current scope behind the scenes.

Sol: true

# 2.3 Inline Controller & $scope 250 pts

We have already separated out the duplicate note code into nw‑card.html. Clean up the template using $scope syntax instead of controllerAs.

1. Inject the $scope service into the controller.

2. Delete the unnecessary alias for our controller.

3. Set the header property on the controller's $scope instead of this. Note: We do not want to modify the string.

4. Set description property on the controller's $scope instead of this. Note: We do not want to modify the string.

5. Display the header and description on the nw-card.html page.

Nw-card.js:

angular.module('NoteWrangler').directive('nwCard', [function() {

return {

restrict: 'E',

templateUrl: 'templates/directives/nw-card.html',

controller: function($scope) {

$scope.header = 'Note Title';

$scope.description = 'A lovely note description.';

},

};

}]);

Nw-card.html:

<div class="card" title="{{header}}">

<h2 class="h3">{{header}}</h2>

<p>{{description}}</p>

</div>

# 2.4 Outside Controller & $scope 240 pts

Now that you are familiar with $scope, let's practice using it with an outside controller.

1. Attach notes to $scope instead of the controller (this).

Use explicit dependency injection to make $scope available to the NotesIndexController function.

angular.module('NoteWrangler')

.controller('NotesIndexController', ['$http', '$scope', function($http, $scope) {

$http({method: 'GET', url: '/notes'}).success(function(data) {

$scope.notes = data;

});

}]);

2. Remove the controllerAs controller alias from routes.js.

3. Update the notes/index.html template to work with $scope.

#### notes-index-controller.js:

angular.module('NoteWrangler')

.controller('NotesIndexController', ['$http','$scope', function($http, $scope) {

$http({method: 'GET', url: '/notes'}).success(function(data) {

$scope.notes=data;

});

}]);

Routes.js:

angular.module('NoteWrangler', ['ngRoute'])

.config(['$routeProvider', function($routeProvider) {

$routeProvider

.when('/', {

redirectTo: '/users'

})

.when('/notes', {

templateUrl: 'templates/pages/notes/index.html',

controller: 'NotesIndexController',

})

.when('/users', {

templateUrl: 'templates/pages/users/index.html'

})

.when('/notes/new', {

templateUrl: 'templates/pages/notes/edit.html',

controller: 'NotesCreateController'

})

.when('/notes/:id', {

templateUrl: 'templates/pages/notes/show.html',

controller: 'NotesShowController'

})

.otherwise({

redirectTo: '/notes'

});

}]);

Index.html:

<div class="note-wrapper">

<div class="note-content">

<div class="notes-header">

<h1 title="Notes">Notes</h1>

</div>

<div class="note-wrapper">

<a class="card-notes" ng-repeat="note in notes">

<nw-card></nw-card>

</a>

</div>

</div>

</div>

# 2.6 Default Scope 250 pts

By default, directives use their parents' scope.

Sol: true

# 2.7 Setting on Scope 250 pts

When setting $scope.title inside a directive using the default scope, we are actually setting a variable named title on the parent $scope.

angular.module('NoteWrangler', [])

.directive("nwCard", function() {

return {

restrict: "E",

templateUrl: "templates/directives/nw-card.html",

controller: function($scope) {

$scope.title = 'Space Llamas';

}

}

});

Sol: true

# 2.8 A Flexible Card Directive 250 pts

In order to make the nw-card directive reusable, it needs to accept a few attributes.

1. Give the nw-card directive an isolate scope, and enable it to accept a header and description attribute with two-way bindings.

2. Now make nw-card.html work for both Notes and Users by accessing the header and description directly from the scope.

Nw-card.js:

angular.module('NoteWrangler')

.directive('nwCard', function() {

return {

restrict: 'E',

templateUrl: 'templates/directives/nw-card.html',

scope: {

header: "=",

description: "="}

};

});

Nw-card.html:

<div class="card" title="{{header}}">

<h2 class="h3">{{header}}</h2>

<p>{{description}}</p>

</div>

# 2.9 Making Isolation Work for You 250 pts

Now that the nw-card directive has an isolate scope, the note cards no longer have access to the variables in their expressions. Let's fix that and finish making this Directive reusable for both Notes and Users.

1. Add attributes for header and description for the nw-card directive in the notes/index.html template.

The note title should be used for the header, and the note description should be used for the description.

2. Add attributes for header and description for the nw-card directive in the users/index.html template.

The user name should be used for the header, and the user bio should be used for the description.

#### notes/index.html:

<div class="note-wrapper">

<div class="note-content">

<div class="notes-header">

<h1 title="Notes">Notes</h1>

</div>

<div class="note-wrapper">

<a class="card-notes" ng-repeat="note in notes" ng-href="#/notes/{{note.id}}">

<nw-card header="note.title" description="note.description"></nw-card>

</a>

</div>

</div>

</div>

#### users/index.html:

<div class="users-wrapper">

<h1>Users</h1>

<div class="users-wrapper">

<a class="card-users" ng-repeat="user in users">

<nw-card header="user.name" description="user.bio"></nw-card>

</a>

</div>

</div>

# 2.11 DOM Manipulation 250 pts

Where is the proper place to access the HTML element within a directive?

Sol: Inside a directive's link function

# 2.12 Style the Tweeted Note 240 pts

Some notes have been altered to have a tweeted property set to a timestamp of when that note was tweeted.

{

"tweeted": "603590400"

}

We want to be able to edit how the card looks if it has been tweeted. Let's use the link function to access the directive's element.

1. Create a link function in the directive and include the parameters we will need to grab element.

2. Add a class of tweeted to the nw-card element for any card that has a tweeted property on its scope.

Add logic to apply a 'tweeted' class when our scope has a tweeted property.

{

// ...

link: function(scope, element) {

if(scope.tweeted) {

element.addClass('tweeted');

}

}

}

#### js/directives/nw-card.js:

angular.module('NoteWrangler')

.directive('nwCard', function() {

return {

restrict: 'E',

templateUrl: './templates/directives/nw-card.html',

scope: {

header: '=',

description: '=',

tweeted: '='

},

link: function(scope,element){

if(scope.tweeted){

element.addClass('tweeted');}

}

};

});