getting started

Žiga Špindler, June 2015

WHY ANGULAR?

- extends HTML
- quick development
- extensible
- open source

FEATURES

(some of them)

DATA BINDING

- model and view are automaticaly updated
- no DOM manipulation

DATA BINDING

```
<html>
<head>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.15
</head>
<body ng-app style="background-color: white;">
  Type your name:
  <input type="text" ng-model="name">

  <h1>Hello {{ name }}</h1>
</body>
</html>
```

DEPENDENCY INJECTION

- we require it and DI provides it
- works great with unit tests

```
function SomeController($location, $routeParams) {
        // Our code
}
```

DIRECTIVES

- many provided with Angular (ng-show, ng-repeat, ...)
- create custom tags or attributes

```
<my-component ng-model="message"></my-component>
```

LET'S BUILD AN APP!

"HELPERS"

- Swiip generator-gulp-angular https://github.com/Swiip/generator-gulp-angular
- Yeoman http://yeoman.io/
- Node.js & npm https://nodejs.org/
- Gulp http://gulpjs.com/
- Bower http://bower.io/

MyLibrary

- for organising personal book library
- modules
- components

```
yo gulp-angular mylibrary
```

FIRST ROUTE

books-routes.js

```
angular.module('mylibrary.books.routes', [
  'ui.router'
  ])
  .config(function ($stateProvider, $urlRouterProvider) {
    $stateProvider
      .state('home', {
        url: '/',
        templateUrl: 'app/components/books/views/books-home.html',
        controller: 'BooksHomeController',
        controllerAs: 'ctrl'
      });

      $urlRouterProvider.otherwise('/');
  });
```

BooksHomeController

books-home-controller.js

```
angular.module('mylibrary.books.controllers.home', [])
  .controller('BooksHomeController', [function() {
    this.books = [
      {
        'id': 1,
        'title': 'Moby-Dick',
        'author': 'Herman Melville',
        'read': 0,
        'pages': 927
      },...
    ];
  }]);
```

booksList directive

books-list-directive.js

```
angular.module('mylibrary.books.directives.list', [])
  .directive('booksList', function() {
    return {
      scope: {
        books: '='
      },
      restrict: 'E',
      templateUrl: 'app/components/books/views/books-list.html',
      controller: 'BooksListController',
      controllerAs: 'ctrl',
      bindToController: true
    };
  });
```

```
<books-list books="ctrl.books"></books-list>
```

booksList directive

books-list-controller.js

```
angular.module('mylibrary.books.controllers.list', [])
  .controller('BooksListController', function() {
    this.bookReadClass = function(book) {
      return book.read ? 'glyphicon-ok text-success' : 'glyphicon-rem
    };
  });
```

books-list.html

```
<tr ng-repeat="book in ctrl.books">
  <td>{{book.id}}</td>
  <td>{{book.title}}</td>
  <td>{{book.author}}</td>
  <td>{{book.pages}}</td>
  <td><span class="glyphicon" ng-class="ctrl.bookReadClass(book)"
        aria-hidden="true"></span></td>
</tr>
```

RESULT

UNIT TESTS

books-list-controller.spec.js

```
describe('BooksListController', function() {
  var ctrl;
  beforeEach(function() {
    module('mylibrary.books.controllers.list');
    inject(function($controller) {
      ctrl = $controller('BooksListController');
    });
  });
  it('should return appropriate class', function() {
    var book = { read: false };
    expect(ctrl.bookReadClass(book))
                .toEqual('glyphicon-remove text-danger');
    book.read = true;
    expect(ctrl.bookReadClass(book))
                .toEqual('glyphicon-ok text-success');
  });
```

ADDING BOOKS

books-new-controller.js

```
angular.module('mylibrary.books.controllers.new', [])
  .controller('BooksNewController', [function() {
    var self = this;

    self.reset = function() {
      self.book = {};
    };

    self.save = function() {
      self.book.id = self.books.length + 1;
      self.books.push(self.book);
      self.reset();
    };

    self.reset();
  }]);
```

ADDING BOOKS

books-new.html

```
<form>
  <div class="form-group">
    <label for="title">Title</label>
    <input type="text" class="form-control" id="title" placeholder="Title" ng-model="ctrl.book.title">
  </div>
  <div class="form-group">
    <label for="author">Author</label>
    <input type="text" class="form-control" id="author" placeholder="Author" ng-model="ctrl.book.author">
  </div>
  <div class="form-group">
    <label>
      Read? <input type="checkbox" ng-model="ctrl.book.read" ng-true-value="1" ng-false-value="0">
    </label>
  </div>
  <div class="form-group">
    <label for="pages">No. of pages</label>
    <input type="number" class="form-control" id="pages" ng-model="ctrl.book.pages">
  </div>
  <a href="#" ng-click="ctrl.reset()">Reset</a>
  <button type="submit" class="btn btn-success" ng-click="ctrl.save()">Save</button>
</form>
```

RESULT

UNIT TESTS

books-new-controller.spec.js

```
describe('BooksNewController', function() {
  var ctrl;
  beforeEach(function() {
    module('mylibrary.books.controllers.new');
    inject(function($controller) {
      ctrl = $controller('BooksNewController');
    });
  });
  it('should add new book', function() {
    ctrl.books = [];
    ctrl.book = {title: 't1',author: 'a1',read: true,pages: 100};
    ctrl.save();
    expect(ctrl.books.length).toBe(1);
    expect(ctrl.books).toEqual([{id: 1,title: 't1',author: 'a1',read:
    expect(ctrl.book).toEqual({});
  });
```

USING FILTERS

- formating data
- many built-in filters (orderBy)
- easy to add custom filters

SORTING DATA

books-list.html

```
<select class="form-control" id="orderby" ng-model="ctrl.orderBy">
  <option value='title'>Title</option>
  <option value='author'>Author</option>
  <option value='read'>Read?</option>
  <option value='id'>ID</option>
  <option value='pages'>Pages</option>
</select>
```

```
<tr ng-repeat="book in ctrl.books | orderBy:ctrl.orderBy">
```

RESULT

USING REST API

- ngResource
- simple API:
  - GET /api/books
  - GET /api/books/{:id}
  - POST /api/books
  - PUT /api/books

CREATING SERVICE

books-services.js

```
angular.module('mylibrary.books.services', [])
  .factory('Book', ['$resource', function($resource) {
    var Book = $resource('/api/books/:bookId', {}, {
      update: { method: 'PUT' }
    });

    Book.createNew = function(bookData) {
      var newBook = new Book(bookData);
      return newBook.$save();
    };

    return Book;
  }]);
```

PASSING DATA TO CONTROLLER

books-routes.js

```
  ...
  controller: 'BooksHomeController',
  controllerAs: 'ctrl',
  resolve: {
    books: function(Book) {
      return Book.query().$promise;
    }
  }
});
```

books-home-controller.js

```
angular.module('mylibrary.books.controllers.home', [])
  .controller('BooksHomeController', ['books', function(books) {
    this.books = books;
  }]);
```

books-new-controller.js

```
angular.module('mylibrary.books.controllers.new', [])
  .controller('BooksNewController', ['Book', function(Book) {
    var self = this;
    self.reset = function() {
      self.book = {};
    };
    self.save = function() {
      Book.createNew(self.book).then(function(book) {
        self.books.push(book);
        self.reset();
      });
    };
    self.reset();
  }]);
```

EDITING BOOKS

books-routes.js

```
.state('edit', {
  url: '/books/:bookId/edit',
  templateUrl: 'app/components/books/views/books-edit.html',
  controller: 'BooksEditController',
  controllerAs: 'ctrl',
  resolve: {
    book: function($stateParams, Book) {
      return Book.get({bookId: $stateParams.bookId}).$promise;
    }
  }
});
```

books-list.html

```
<td>
  <a ui-sref="edit({ bookId: book.id })">edit</a>
</td>
```

RESULT

UNIT TESTS

books-new-controller.spec.js

```
describe('BooksNewController', function() {
  var ctrl, mockBackend;
  beforeEach(function() {
    module('mylibrary.books.controllers.new');
    inject(function($controller, $httpBackend) {
      ctrl = $controller('BooksNewController');
      mockBackend = $httpBackend;
      mockBackend.expectPOST('/api/books').respond({id: 1,title: 't1'
    });
  });
  it('should add new book', function() {
    ctrl.books = [];
    ctrl.book = {title: 't1',author: 'a1',read: true,pages: 100};
    ctrl.save();
    mockBackend.flush();
    expect(ctrl.books[0].id).toEqual(1);
```

E2E TESTING

- Protractor https://angular.github.io/protractor/
- build for AngularJS
- supports Angular locators

---

main.po.js

```
var MainPage = function() {
  this.jumbotron = element(by.css('.jumbotron'));
  this.h1 = this.jumbotron.element(by.css('h1'));
  this.books = element(by.css('body'))
        .all(by.repeater('book in ctrl.books'));
};
```

EXAMPLE

main.spec.js

```
describe('The main view', function () {
  var page;
  beforeEach(function () {
    browser.get('http://localhost:3001/');
    page = require('./main.po');
  });
  it('should include jumbotron with correct data', function() {
    expect(page.h1.getText()).toBe('MyLibrary');
  });
  it('list more than 4 books', function () {
    expect(page.books.count()).toBeGreaterThan(4);
  });
});
```

QUESTIONS?

Code: https://github.com/springfield/mylibrary