ANGULAR.JS FOR THE GRAILS ENTHUSIAST

@wbucksoft

willbuck @ github

INTENDED AUDIENCE

- Pretty new to Angular
- Want to know more about SPAs and Grails
- Feeling like there's a better way

WHO AM I?

I'm Will.

I LIKE MN

I LIKE TO PLAY

I LIKE TO CODE

ANGULAR AT VIRTUWELL

- Started in 2012 building a Backbone app
- Ended up with a lot of code & inconsistent 2-way binding
- Chose AngularJS for our next big project
- Team fell in love
- Re-wrote much of Backbone app in Angular in short matter of months

WHAT WE BUILT

- Nurse practitioners treat VW cases
- Old Grails app was eyesore & spaghetti code
- Re-write in Grails 2.1 as light-weight JSON endpoints
- Majority of work as SPA in Angular
- Re-design to support efficient workflow

WHAT WE USED

Besides Angular + Grails...

- Angular-UI
- Restangular
- Coffeescript
- SASS
- Grunt
- And more!

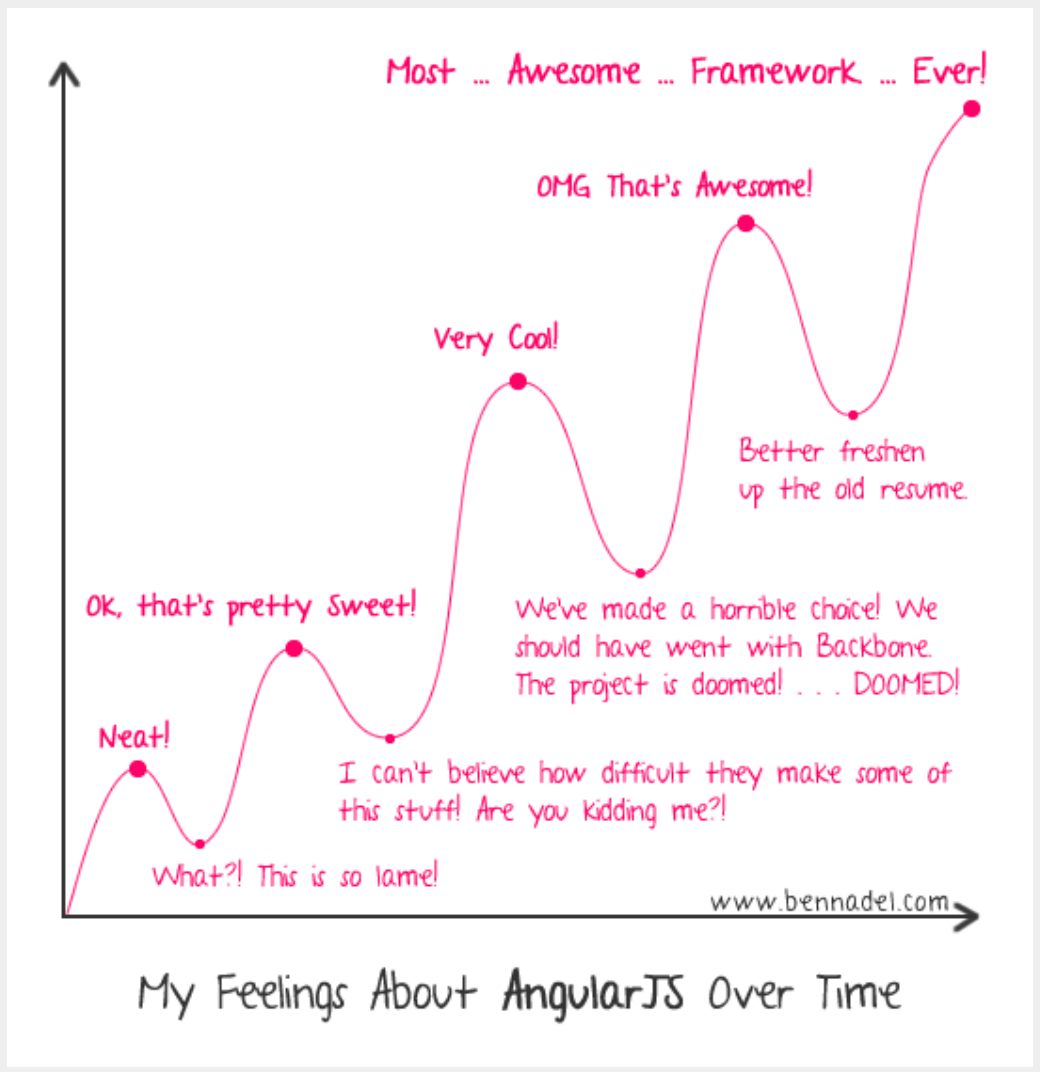
WHY ANGULAR?

- Easy 2-way binding
- Grails-like MVC(S)
- Light, fast, productive
- Lots of activity & adoption
- TESTABILITY

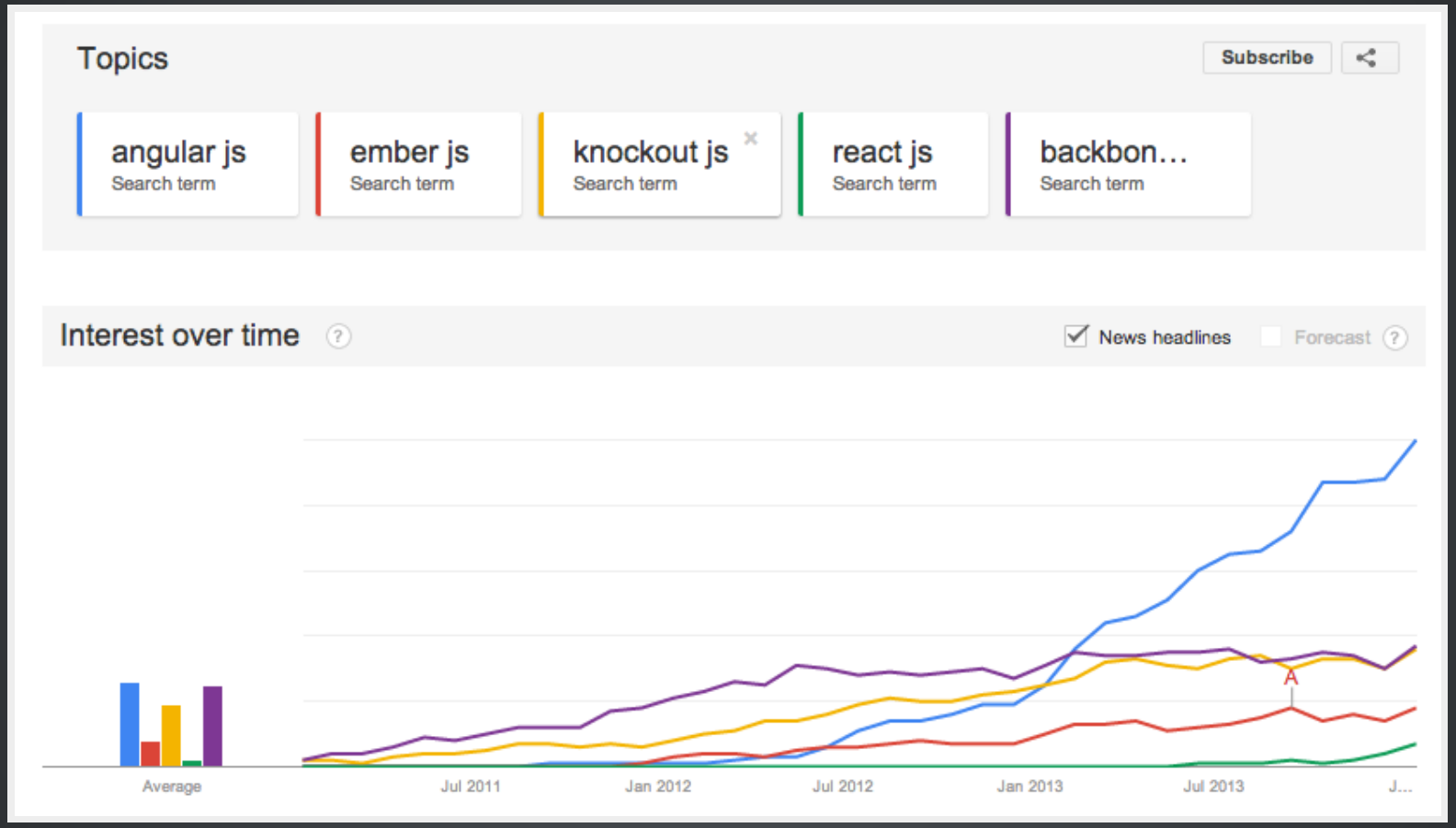
WHY GRAILS?

- Familiar tool chains for development and deployment
- Dead simple database integrations, SQL & NoSQL
- Recognition of rapidly growing front-end ecosystem
- Powerful new REST support

THE LEARNING CURVE



CONVINCING YOUR BOSS



ANGULAR KEY COMPONENTS

a.k.a. Show us some code please

ANGULAR KEY COMPONENTS: APP

Where it all begins

Central place for defining & configuring the app

```
<body ng-app="angNewsApp">
```

app.js

ANGULAR KEY COMPONENTS: MODULES

Like Packages (kind of)

../auth/authModule.js

```
1 angular.module(
2     'auth',
3     ['auth.view', 'auth.api', 'auth.user.api', 'auth.profile.view']
4 );
```

../auth/authController.js

```
1  'use strict';
2
3  angular.module('auth.view', ['auth.api', 'auth.user.api'])
4      .controller('AuthController', function($scope, $location, AuthService, UserService) {
5          if(AuthService.signedIn()) {
6              $location.path('/');
7          }
8
9          $scope.$on('$firebaseSimpleLogin:login', function () {
10             $location.path('/');
11         });
12
13         $scope.login = function () {
14           AuthService.login($scope.user).then(function () {
15             $location.path('/');
16           }, function (error) {
17             $scope.error = error.toString();
18           });
19         };
20
21         $scope.register = function() {
22             AuthService.register($scope.user).then(function(authUser) {
23                 UserService.create(authUser, $scope.user.username);
24                 console.log(authUser);
25                 $location.path('/');
26             }, function (error) {
27                 $scope.error = error.toString();
28             });
29         };
30     });
```

app.js

ANGULAR KEY COMPONENTS: CONTROLLERS

Similar to Grails, keep this light, just "glue code"

../auth/authController.js

```
'use strict';

angular.module('auth.view', ['auth.api', 'auth.user.api'])
    .controller('AuthController', function($scope, $location, AuthService, UserService) {
        if(AuthService.signedIn()) {
            $location.path('/');
        }

        $scope.$on('$firebaseSimpleLogin:login', function () {
            $location.path('/');
        });

        $scope.login = function () {
          AuthService.login($scope.user).then(function () {
            $location.path('/');
          }, function (error) {
            $scope.error = error.toString();
          });
        };

        $scope.register = function() {
            AuthService.register($scope.user).then(function(authUser) {
                UserService.create(authUser, $scope.user.username);
                console.log(authUser);
                $location.path('/');
            }, function (error) {
                $scope.error = error.toString();
            });
        };
    });
```

../auth/authController.js (min-proof)

```
angular.module('auth.view', ['auth.api', 'auth.user.api'])
    .controller('AuthController', ['$scope', '$location', 'AuthService', 'UserService',
        function($scope, $location, AuthService, UserService) {
            if(AuthService.signedIn()) {
                $location.path('/');
            }

            $scope.$on('$firebaseSimpleLogin:login', function () {
                $location.path('/');
```

ANGULAR KEY COMPONENTS: SCOPE

`$scope` ties controllers and views together

Try not to clutter it too bad

Be aware of pass by value (primatives) vs pass by ref (objects)

```
$scope.myInt = MyService.myInt
```

```
MyService.myInt = 5
```

vs

```
MyService.myInt = {value: 5}
```

../posts/postController.js

```
$scope.post = {url: 'http://gr8conf.us', title: 'Grrrr8!'};

$scope.deletePost = function (postId) {
    PostService.delete(postId);
};
```

../posts/postView.html

```
<div class="info">
  <a href="{{ post.url }}">
    {{ post.title }}
    <span class="url">({{ post.url | hostnameFromUrl}})</span>
  </a>
</div>
 <div>
  <span>{{ post.score || 0 }} votes</span>
  &mdash;
  <span>submitted by <a href="#/users/{{ post.owner }}">{{ post.owner }}</a></span>
  -
  <a href="#/post/{{ postId }}">comments</a>
  <a ng-show="signedIn() && post.owner === currentUser.username" ng-click="deletePost(postId)">delete</a>
</div>
```

ANGULAR KEY COMPONENTS: VIEWS

Templates are auto-bound, update automatically

../auth/register.html

```
  <h2>Register</h2>

<form ng-submit="register()" name="form">
  <div ng-show="error || form.username.$error">
    <p ng-show="error" class="text-danger">{{ error }}</p>
    <p ng-show="form.username.$error.taken" class="text-danger">Username is already taken.</p>
    <p ng-show="form.username.$error.invalid" class="text-danger">Username contains invalid characters.</p>
  </div>
  <input type="email" ng-model="user.email" placeholder="Email" class="form-control"><br>
  <input type="text" ng-model="user.username" placeholder="Username" name="username" check-username class="form-
  control"><br>
  <input type="password" ng-model="user.password" placeholder="Password" class="form-control"><br>
  <input type="submit" value="Register" class="btn btn-primary" />
</form>

</div>
```

ANGULAR KEY COMPONENTS: SERVICES

Like Grails, where you want to:

- interact with your storage (via REST, LocalStorage, w/e)
- keep core business logic (data manipulation)
- hold data to share

StackOverflow - Service vs Factory vs Provider

```
myApp.service('helloWorldFromService', function() {
    this.sayHello = function() {
        return "Hello, World!"
    };
});

//factory style, more involved but more sophisticated
myApp.factory('helloWorldFromFactory', function() {
    return {
        sayHello: function() {
            return "Hello, World!"
        }
    };
});

//provider style, full blown, configurable version
myApp.provider('helloWorld', function() {
    // In the provider function, you cannot inject any
    // service or factory. This can only be done at the
    // "$get" method.

    this.name = 'Default';

    this.$get = function() {
        var name = this.name;
        return {
            sayHello: function() {
                return "Hello, " + name + "!"
            }
        }
    };

    this.setName = function(name) {
        this.name = name;
    };
});
```

../auth/authService.js

```
1  'use strict';
2
3  angular.module('auth.api', [])
4      .factory('AuthService',
5        function ($http) {
6          var currentUser = {};
7
8          var publicApi = {
9            register: function (user) {
10             return $http.post('registerUser', {user: user});
11           },
12           signedIn: function () {
13             return currentUser !== null;
14           },
15           login: function (user) {
16             return $http.post('login', user);
17           },
18           logout: function () {
19             return $http.get('logout');
20           },
21           currentUser: currentUser
22         };
23
24         return publicApi;
25       });
```

ANGULAR KEY COMPONENTS: FILTERS

Value transformers, useful for some 'display logic'

../common/hostnameFromUrlFilter.js

```
1  'use strict';
2
3  angular.module('common.filter.hostname', [])
4      .filter('hostnameFromUrl', function () {
5        return function (str) {
6          var url = document.createElement('a');
7
8          url.href = str;
9
10         return url.hostname;
11       };
12     });
```

../posts/postView.js

```
<a href="{{ post.url }}">
  {{ post.title }}
  <span class="url">({{ post.url | hostnameFromUrl}})</span>
</a>
```

ANGULAR KEY COMPONENTS: DIRECTIVES

Like web-components

"Secret sauce" of angular

Enables incredibly expressive views via tags & attributes

../common/checkUsernameDirective.js

../auth/register.html

```
  <h2>Register</h2>

<form ng-submit="register()" name="form">
  <div ng-show="error || form.username.$error">
    <p ng-show="error" class="text-danger">{{ error }}</p>
    <p ng-show="form.username.$error.taken" class="text-danger">Username is already taken.</p>
    <p ng-show="form.username.$error.invalid" class="text-danger">Username contains invalid characters.</p>
  </div>
  <input type="email" ng-model="user.email" placeholder="Email" class="form-control"><br>
  <input type="text" ng-model="user.username" placeholder="Username" name="username" check-username class="form-
  control"><br>
  <input type="password" ng-model="user.password" placeholder="Password" class="form-control"><br>
  <input type="submit" value="Register" class="btn btn-primary" />
</form>

</div>
```

REPLACING YOUR FAVORITE GRAILS TAGS

- `<g:each>` becomes `ng-repeat`
- `<g:if>` becomes `ng-if` or `ng-show`
- `<g:formatDate>` & co. become number/date filters
- `<g:link>` ? don't need it! UI-Router has great stateful navigation
- `<g:message>` has third party options `angular-gettext` and `angular-translate`

REPLACING YOUR FAVORITE JQUERY-ISMS

- `$.ajax` becomes `$http`, `$resource`, or maybe `Restangular`
- `$.click` becomes `ng-click`
- `$.animate` becomes `ng-animate` (and some css)
- Little / no need for selectors (2-way binding) or adding/removing classes (`ng-class`)!
- jQLite included with framework

SECURITY

I like Spring Security, can I still use it?

(yes)

SECURITY: ROLE-BASED CONTROLS

- Front-end code is beatable
- Deliver only what user is authorized for server-side

A NOTE ON REST AND APIS

- Restangular is really nice
- Build your grails urls w/ versioning in mind
- Think beyond domain-models to minimize requests

A NOTE ON TOOLING

We used Grunt and Bower, seemed more features / support than Resources

No experience w/ Asset Pipeline, sounds promising though

BUT HOW DO I START?

FIRST STEPS

- Egghead bite-sized video chunks
- Plunker small examples

DEEPER DIVES

- Thinkster Cool tutorial app
- NG-Book Comprehensive & constantly up to date
- NG-Conf Videos, esp DoubleClick Team

KEEPING UP TO DATE

- If you have freedom to stay up to date, may as well
- However don't let shiny new hotness keep you from shipping

THANK YOU FOR LISTENING!

Questions? Comments? Please give me feedback!
https://www.surveymonkey.com/s/DDG73GD