Angular Modules

Magnus Gudmundsson CyberDaDa,
Christoffer Noring, Softhouse

UI-Grid
Angular NVD3
Angular Fire (Firebase)
Angular + browserify
REST, ngResource, angular-restify
Angular translate

UI-GRID

To grid or not to grid?

The simple nogrid solution
‘Filtering + Sorting Asc/Desc

```
<th ng-click="reverse = predicate == 'loB' && !reverse;
predicate = 'loB'">LoB </th>

<tr ng-repeat="itm in collection |
               orderBy:predicate:reverse |
               filter:myFilter()">
```

Demo:
http://localhost:64551/#/weeklyperproject

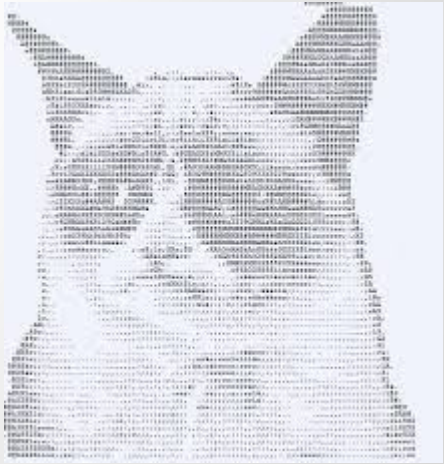
But sometimes you need more...

Render large numbers of rows (10K plus)
In place editing.
5000 Columns
Export to PDF
etc etc etc etc

http://ui-grid.info/

http://ui-grid.info/docs/#/tutorial/

CHARTING



ALTERNATIVES

Server Side Render

Canvas

SVG

Raphaël - http://raphaeljs.com/

D3 - http://d3js.org/

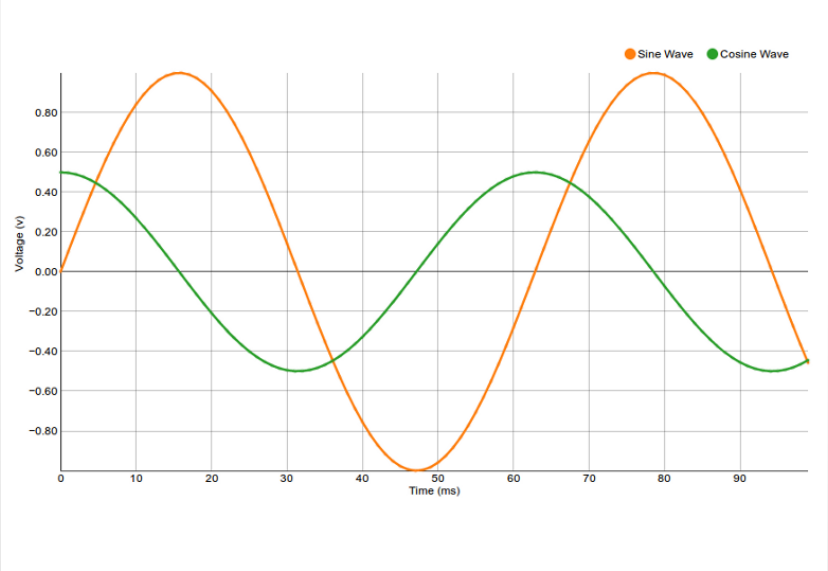
But..

I just want to draw a pie and some bars based on some questionable statistics…

+ they have to look good in order to keep managent happy.

NVD3 to the rescue

http://nvd3.org/ - A reusable chart library



```
nv.addGraph(function() {
  var chart = nv.models.discreteBarChart()
    .x(function(d) { return d.label })
    .y(function(d) { return d.value })
    .staggerLabels(true)
    .tooltips(false)
    .showValues(true)

  d3.select('#chart svg')
    .datum(data)
    .transition().duration(500)
    .call(chart)
    ;

  nv.utils.windowResize(chart.update);
  return chart;
});
```

What about Angular ???

Angular-nvD3 (Beta 1)

http://krispo.github.io/angular-nvd3/

Angularjs-nvd3-directives

https://cmaurer.github.io/angularjs-nvd3-directives/

Angularjs-nvd3-directives - code example

```
<nvd3-discrete-bar-chart
    data="exampleData"
        id="exampleId"
        showXAxis="true"
        showYAxis="true"
        xAxisTickFormat="xAxisTickFormatFunction()"
        width="550"
        height="400">
         <svg></svg>
    </nvd3-discrete-bar-chart>
```

Caveats

Long dependency chain

D3--nvd3--angular-nvd3 + angular x

When getting errors it is not easy to deduct where the cause is (usually in your code :)

Other alternatives

AngularJS Directives for charts
https://github.com/carlcraig/tc-angular-chartjs

Angular-charts

https://github.com/chinmaymk/angular-charts

Highcharts.js (Directive embrio exists on github). Licence needed when doing commercial work.

Firebase, www.firebase.com

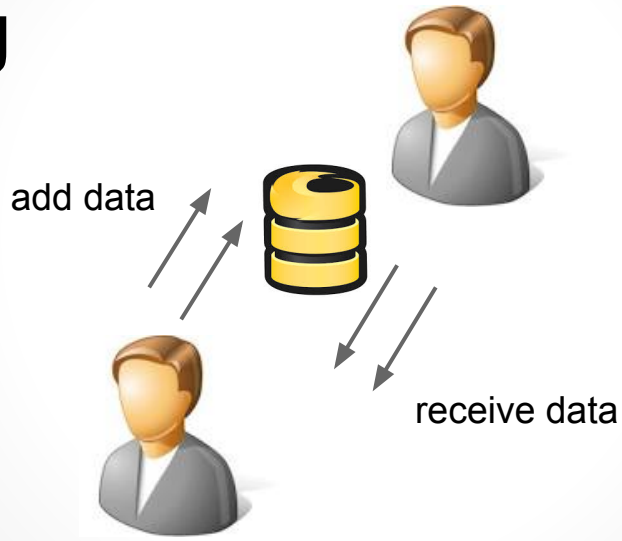
What is it?, cloud service provider and “backend as a service”

Callable API where data is synchronized in real time. For creating “collaboration apps”

chat, booking apps for movie, laundry room, train anything you can imagine

Firebase + AngularFire

3- way binding




```
{
  test : “”,
  cars : {
    id : {
      name : ‘car1’
    },
    id2 : {
      name : ‘car2’
    }
  }
}
```

DB is json tree

no real array, arrays are bad!

Firebase - prerequisits

Scripts

```
<!-- AngularJs -->
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.3.15/angular.min.js"></script>

<!-- Firebase -->
<script src="https://cdn.firebase.com/js/client/2.2.4/firebase.js"></script>

<!-- AngularFire -->
<script src="https://cdn.firebase.com/libs/angularfire/1.1.1/angularfire.min.js"></script>
```

Firebase - code

```
var app = angular.module("sampleApp", ["firebase"]);

app.controller("CarCtrl", function($scope, $firebaseArray, $firebaseObject){

    var ref = new Firebase("https://<userdb>.firebaseio.com/cars");
    var refObject = new Firebase("https://<userdb>.firebaseio.com/test");

    $scope.cars = $firebaseArray(ref);
    $scope.cars.$add(item);

    $scope.aProperty = $firebaseObject(refObject);
}
```

endpoint reference

convert to a bindable array

Firebase, CRUD

Replace content

```
ref.set({
        ferrari : { name : 'Ferrari', year : 1990 },
        ferrariF40 : { name : 'FerrariF40', year : 1989 },
        lamborghini : { name : 'Lamborghi Diablo', year : 2001 }
});
```

Remove content

```
ref.child('ferrariF40')
remove();
```

Update content

```
ref.child('ferrariF40')
.update
({ color : "Red", year : 1991 });
```

Array add

```
ref.child('cars')
.push({ color : "Red", year : 1991, name :’Porsche’ });
```

Firebase, event + queries

```
ref.on(‘child_added’, function(snapshot){ var item = snapshot.val() })
```

called once per item but also for new added items

```
ref.on(‘value’, function(snapshot){ var item = snapshot.val() })
```

gives all the data, triggered on init + data change

```
var queryRef = ref.orderByChild('year').limitToLast(2);

queryRef.on('value', function (snapshot) { snapshot.value();  }
```

ordering + limit

REST ngResource

ngResource is from angular core team

Covers calling with GET, PUT, POST, DELETE

ngResource - starting out

```
$resource(url,params,actions);
```

First thing is to wrap it in a factory like so

```
app.factory('Book', function($resource){
        return $resource(baseUrl + '/Books',params,actions);
})
```

And use it like

```
Book.get({ id : 2 }, function(myBook){

})
```

ngResource CRUD

GET http://domain/books

```
Book.query(function(allBooks){

})
```

GET http://domain/books/1

```
Book.get({ id : 1 }, function(oneBook){

})
```

POST http://domain/books/

```
Book.save({ name : 'Farewell to arms', author : 'Ernest Hemingway' }, function(savedBook){

});
```

PUT http://domain/books

```
Book.update({ name : 'Farewell to arms', author : 'Ernest Hemingway' }, function(updatedBook){

});
```

DELETE http://domain/books/11

```
Book.remove({ id : 11 }, function(removedBook){

})
```

ngResouce override

GOTCHA : transformResponse behaves very different
1.2 transforms the way you expect
higher versions copies DomainObject properties but forgets protptype methods

$resource(url,params,**actions**);

```
actions = {
  get: {
    url : 'someUrl',
    method : 'GET',
    isArray : true
  // assume what comes back is an array
  }
}
```

override the method **get** :
aka replace key

```
actions = {
  get: {
    url : 'someUrl',
    method : 'GET',
    isArray : true
// assume what comes back is an array, else false,
    transformResponse : function(response){
      return new DomainObject(response);
    }
  }
}
```

defined method
transformResponse

ngResource - custom action

```
Book
    .booksWithAuthor(
      { authorId : 124 },
      function(books){
});
```

```
booksWithAuthor : {
        url : baseUrl + '/Books/Author/:authorId',
        method : 'GET',
        isArray : true,
        transformResponse : function(result, headersGetter){
                var books = angular.fromJson(result);
                var domainBooks = [];
                books.forEach(function(book){
                        domainBooks.push(new DomainBook(book));
                });

                return domainBooks;
        }
    }
```

```
$resource(url,params,actions);
```

```
var parameters = { id: '@_id', authorId : '@_authorId'}
```

Internationalization and Localization

i18n, i**nternationalizatio**n, funny fact : 18 characters between i and n

Internationalization is the process of designing a software application so that it can potentially be adapted to various languages and regions without engineering changes

Localization is the process of adapting internationalized software for a specific region or language by adding **locale-specific components** and translating text

Angular-translate - installation & concepts

current version 2.7.0

bower install **angular-translate**

{{ 'KEY' | translate }}

$translate

$translateProvider

Static files

JSON

Angular translation setup

Config:

define lang dictionary

```
var app = angular.module('app',['pascalprecht.translate']);

app.config(function($translateProvider){
        $translateProvider.translations('en', {
            TEST: 'testing testing, one two',
            APPLICATION_HEADER: 'My awesome app',
            SAVE : 'Save',
            CHANGE : 'change language to english',
            CHANGE_SV : 'change language to swedish'
        })
        .translations('sv', {
             TEST : 'testar 1 2',
             APPLICATION_HEADER: 'Min grymma app',
             SAVE : 'Spara',
             CHANGE : 'byt språk till engelska',
             CHANGE_SV : 'byt språk till svenska'
        });
```

set start language

```
        $translateProvider.preferredLanguage('en');
});
```

Angular translation usage

```
<div ng-controller="appCtrl">
        <h1>
            {{ 'APPLICATION_HEADER' | translate }}
        </h1>
</div>
```

OR

```
<button translate="SAVE"></button>
```

Angular translation - change language

```
app.controller('appCtrl', function($scope,$translate){
        $scope.switchLanguage = function(lang){
                $translate.use(lang);
        }
}
```

```
<button ng-click="switchLanguage('en')">change</button>
```

Angular translate - static files

```
app.config(function($translateProvider){
        $translateProvider.useStaticFilesLoader({
                prefix: '/langs/',
                suffix: '.json'
        });
});
```

directory

file ending

example : langs/en.json

```
{
    "TEST": "testing testing, one two",
    "APPLICATION_HEADER": "My awesome app",
    "SAVE" : "Save",
    "CHANGE" : "change language to english",
    "CHANGE_SV" : "change language to swedish"
}
```

Angular & Browserify

use node (CommonJs) and node modules in the browser. **True fullstack**

```
var userService = require('./userService');
```

how can it help angular?

- more nicely separated code
- easier to test
- no more <script> * n tags in index.html
- you can use node modules in your angular app

Browersify bundling

browserify app.js -o bundle.js

won’t this create a big bundled mess?

browserify app.js -o bundle.js **-d**

generates a source map, so you can debug your files

2* the size

Browserify / nodify your app

```
npm install angular
require(‘./angular/angular’)

angular.module(‘app’,[])
.controller(‘appCtrl’, require(‘infrastructure/appCtrl’))

--- appCtrl.js -----
function AppCtrl($scope,UserModel){
        $scope.test = 'test';
        var user = new UserModel({});
}

module.exports = AppCtrl;
```

Browerify testing

test with “for example” NodeUnit

```
exports.testUserModel = function(test){
        var productSrvMock = {
                get : function(){
                        console.log('typing mock from productSrv');
        }};

        var UserModel = require('../../features/user/user')(productSrvMock);
        var user = new UserModel({ name : 'pelle' });

       test.ok(user.title === 'pelle', "should set title to Pelle");
       test.done();
};
```

easier to mock!

Questions?

Feel free to contact / connect with us on either linked in or twitter