Distilled JavaScript

Session 4 - Purifying for real

We're Osper.

We believe in financial freedom for young people.

We believe in building good habits early.

We believe in simplicity and safety.

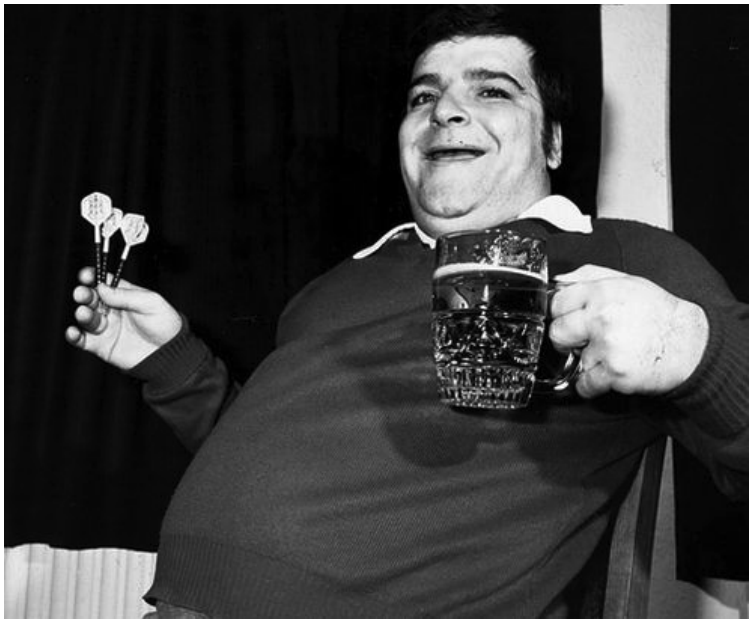
We believe it's time for something new.

About me - Grahame 'Frank' Oakland

Programmer - recovering Java-holic

JavaScripter - {

```
  1:'Closure',
  2:'AngularJS',
  3:'ExtJS',
  4:'YUI',
  5:'jQuery',
  6:'Vanilla DOM'
}
```



Now searching for the **MEAN**-ing of life?

JS

Exercise 1

Grab the relevant `purifying` branch from github and fire it up

```
git clone -b $BRANCH https://github.com/distillers/purifying.git
```

where `$BRANCH` in `[bruce, ice, lionel, mr_t]`

{{ review }}

MEAN teastack - handy mongo bits for local and remote

mongodb connect string for remote db:

```
mongodb://tea:mean@ds033669.mongolab.com:33669/tea
```

mongoimport command for loading usages into local db:

```
mongoimport --db tea --collection usages --type json --file usages.json
```

Code Exercise - Connect up to some real tea

1. Modify the `teastack` app to use the remote mongo db

2. Remember not to scrub the data.

```
mongodb://tea:mean@ds033669.mongolab.com:33669/tea
```

AngularJS - distilled

Module - The starting point for your app

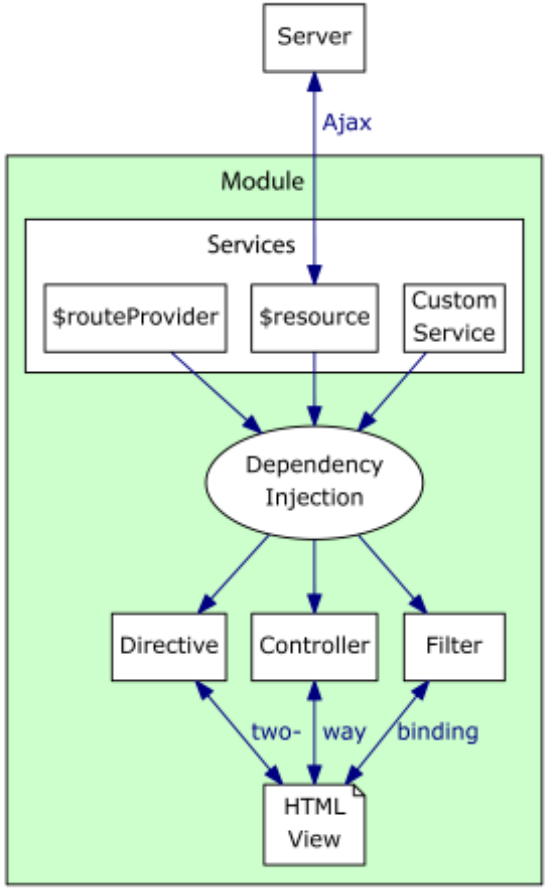
Dependency Injection - IOC Angular style

Services - `$routeProvider, $resource`

Controllers - connect Model and View

Directives - custom HTML tags

Two way data binding - Observe and learn



JS

Custom Directive - Displaying the postcodes data

```
angular.module('components', [])
  .directive('postalArea', function () {
    return {
      restrict: 'E',
      transclude: true,
      scope: {
        postalArea: '=area'
      },
      controller: function ($scope, $element) {
      },
      template: '<div class="area-dir" +
        'style="background-color: rgba(0, 102, 0, {{ postalArea.energy }});" +
      ' </div>',
      replace: true
    };
  })
```

Purifying – connecting the front end MVC to our RESTful service

Method 1. RESTful resource

```
// Define the service in app/services.js
var teaServices = angular.module('teaServices',
['ngResource']);

teaServices.factory('Usage', ['$resource',
  function($resource){
    return $resource('usages/:postcode.json', {}, {
      query: {method:'GET', params:{postcode:'postcode'},
isArray:true}
    });
  }]);


// Controller definition in app/controllers.js for example
teaAppControllers.controller('WheelOfFortuneCtrl',
['$scope', 'Usages', function($scope, Usages) {
  $scope.Usages = Usages.query();
  $scope.orderProp = 'age';
}]);
```

Method 2: $http service

```
var teaApp = angular.module('teaApp', []);

teaApp.controller('WheeloFFortuneCtrl', ['$scope', '$http',
  function ($scope, $http) {
    $http.get('api/usages.json').success(function(data) {
      $scope.usages = data;
    });

    $scope.orderProp = 'day';
  }]);
```

Code Exercise - Spin up the wheel of fortune

1. Connect `/usages` route up to the `/usages/api` method

2. Calculate the saving for each postcode (square)

3. Display the saving by reducing the opacity

Bonus: Can you display this on an hour by hour basis

tea.js – Average consumption per household

Utility	Annual	Hourly
Gas	16,500,000 Wh	1884 Wh
Electricity	3,300,000 Wh	377 Wh
Water	54,750 litres	6.25 litres

**Assumption: There are exactly 100 houses per postcode*

{{ have.a.brew }}

WebSockets – using **socket.io** to broadcast monitor data changes

Server

```
var app = require('express').createServer(),
    io = require('socket.io').listen(app);

app.listen(80);

app.get('/', function (req, res) {
  res.sendfile(__dirname + '/index.html');
});

var activeClients = 0;
io.sockets.on('connection', function (socket) {
  activeClients++;
  io.sockets.emit('clients', {clients: activeClients});
  socket.on('my other event', function (data) {
    console.log(data);
  });
});
```

Client

```
<script src="/socket.io/socket.io.js"></script>
<script>
  var socket = io.connect('http://localhost');
  socket.on('clients', function (data) {
    console.log(data);
    socket.emit('my other event', { my: 'data' });
  });
</script>
```

Code Exercise - Use socket.io to broadcast data changes

1. Hook up socket.io into express

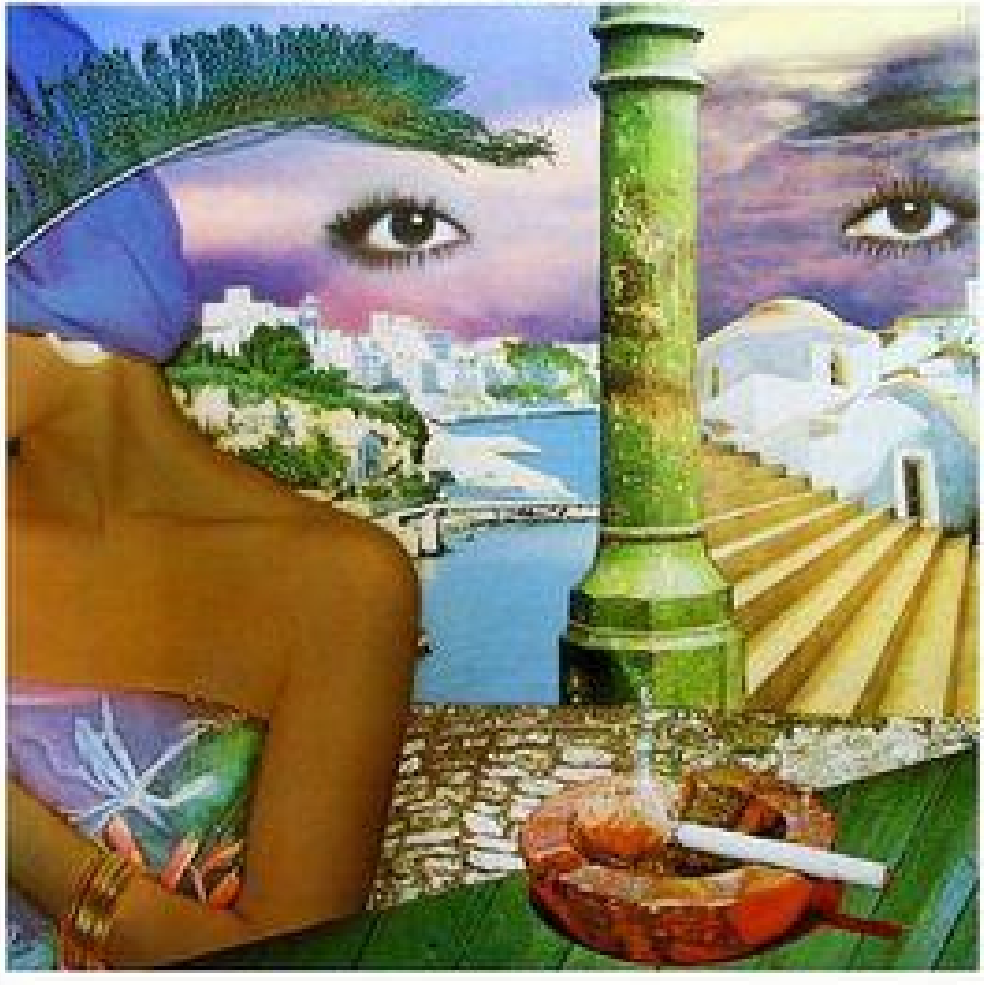
2. broadcast a `usage` message to connected clients

3. Recalculate the savings and update the view

*Check out Brian Fords **angular-socket-io-seed** project for inspiration*

https://github.com/btford/angular-socket-io-seed.git

WEATHER REPORT MR. GONE



```dart
library todo;

import 'package:angular/angular.dart';

class Item {
  String text;
  bool done;

  Item([String this.text = '', bool this.done = false]);

  bool get isEmpty => text.isEmpty;

  clone() => new Item(text, done);

  clear() {
    text = '';
    done = false;
  }
}

// ServerController interface. Logic in main.dart determines which
// implementation we should use.
abstract class ServerController {
  init(TodoController todo);
}

// An implementation of ServerController that does nothing.
class NoServerController implements ServerController {
  init(TodoController todo) { }
}
```

```
(define (square x) (* x x))

(define (tree-map f tree)
  (cond ((null? tree) null)
        ((pair? tree)
         (cons (tree-map f (car tree))
               (tree-map f  cdr tree))))
        (else (f tree))))

(define (map-tree-map f tree)
  (define (rec subtree)
    (map (lambda (subtree)
           ((if (pair? subtree) rec f) subtree))
         subtree))
  (rec tree))

(define (square-tree tree) (tree-map square tree))
(define (map-square-tree tree) (map-tree-map square tree))
```

```
(sel1 :body) ; => document.body

(sel1 :#my-id) ; => document.getElementById
("my-id")

(sel parent :.child) ; => [].slice.call
(parent.getElementsByClassName("child"))

(sel ".c1, .c2") ; => [].slice.call(document.
querySelector(".c1, .c2"))
```


CHEESUS
CHRIST
CHEESEGRATER
OUR GRATE

There are two schools of thought about teaching computer science.

- ***The conservative view***: *Computer programs have become too large and complex to encompass in a human mind. Therefore, the job is to teach people how to discipline their work in such a way that 500 mediocre programmers can join together and produce a program that correctly meets its specification.*

- ***The radical view***: *Computer programs have become too large and complex to encompass in a human mind. Therefore, the job is to teach people how to expand their minds so that the programs can fit, by learning to think in a vocabulary of larger, more powerful, more flexible ideas than the obvious ones. Each unit of programming thought must have a big payoff in the capabilities of the program.*

-Brian Harvey, Simply Scheme

SONY

{{ have.a.beer }}