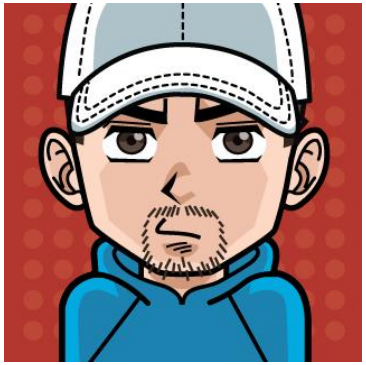
JS Battle

Le combat le plus attendu de l'histoire du Web

@k33g_org



@sebmade

@tchak13

Backbone.js

@k33g_org / Lyon / Steria

1995 : TECHCO-DEV-CP-DI-CP-DEV-CP-DP-DT-AVV

MVC

Jeremy Ashkenas

jQuery + Underscore

tous navigateurs

Backbone : 19 kb

jQuery : 83 kb

Underscore : 13 kb

Backbone, c'est Play!> dans votre navigateur

Front Stack Constructor Helper

RESThub Backbone.js Stack
Marionette.js
Chaplin

mais nu, c'est bien meilleur ;)

MVC ?

Backbone.Model
Backbone.Collection

MVC

Templates

_.template()

MVC

Backbone.View

?!*#!???

Mais aussi ...

Backbone.Sync
Backbone.Router
...

Modèle Objet

Un modèle "interne" ?

Pompage

```
var Kind = function() {};
Kind.extend =
Backbone.Model.extend;
```

klass

```
var TinyToon = Kind.extend({
    firstName : "John",
    lastName : "Doe",
     constructor : function (firstName, lastName){
        this.firstName = firstName;
        this.lastName = lastName;
    },
    toString : function() {
        return this.firstName + " " +this.lastName
    }
});

var babs = new TinyToon("Babs", "Bunny")
```

héritage

```
var Elmyra = TinyToon.extend({
    firstName : "Elmyra",
    lastName : "Duff",
    constructor : function (){
        Elmyra.__super__.constructor.call(this);
        console.log("Oh, it's so cute ♥ ♥ ♥");
    },
    isChasing : function(toon) {
      //foo
    }
});
```

static

```
var Elmyra = TinyToon.extend({
    firstName : "Elmyra",
    lastName : "Duff",
    /**/
},{ //statics members
    theOnlyOne : null,
    getInstance : function() {
        if(Elmyra.theOnlyOne == null) {
            Elmyra.theOnlyOne = new Elmyra();
        }
        return Elmyra.theOnlyOne;
    }
});
var elmyra = Elmyra.getInstance();
```

Backbone.Model
Backbone.Collection
Backbone.synk

Model

```
var Player = Backbone.Model.extend({
    urlRoot:"/players"
});

var bob = new Player({
 firstName : "Bob",
 lastName : "Morane"
})
```

get / set

```
bob.get("firstName")
bob.set("firstName","Bob")
bob.set({
   lastName:"Morane", twitter:"@bobby"
})

bob.on(
"change:lastName change:firstName",
...);
```

mais aussi

save
fetch
destroy

Backbone.sync

Appelé à chaque lecture / écriture des modèles

Backbone.sync

Par défaut :

REST

$.ajax()

save

```
bob.save()
```

```
POST : http://domain/players
```

fetch

```
var bob = new Player({
  id:"001"
})

bob.fetch()
```

GET : http://domain/players/001

save/update

```
bob.set("twitter","@bobby")

bob.save()
```

PUT : http://domain/players/001

destroy

```
bob.destroy()
```

```
DELETE : http://domain/players/001
```

Collection

```
var Players = Backbone.Collection.
extend({
    url :"/players",
    model : Player
});

var players = new Players()
```

Underscore methods

```
players.each(function(player) {
    console.log(player.get("firstName"));
})
```

Underscore methods

```
players.filter(function(player) {
    return player.get("firstName")=="Bob";
})
```

Underscore methods

```
players.find(function(player) {
    return player.get("framework")==null;
})
```

fetch : getAll

```
players.fetch()
```

GET : http://domain/**players/**

_.template()

Oh p... du Scala ?

```
<ol>
<% _.each(players, function(player) { %>
    <li id="<%= player.id %>">
            <%= player.firstName %>
            <%= player.lastName %>
    </li>
<% }); %>
</ol>
```

d'autres moteurs

Mustache
Handelbars
TempoJS

Backbone.View

... Controller

```
var PlayersView = Backbone.View.extend({
    el : "#players",
    initialize : function() {
        this.template =
            _.template($("#template").html());
    },
    render : function() {
        this.$el.html(
            this.template({
                players:this.collection.toJSON()
            })
        );
    }
});
```

```
var PlayersView = Backbone.View.extend({
    /*...*/
    events:{
        "click :checkbox" : function(event) {},
        "click .btn-success" : function() {}
    }
});

var playersView =
    new PlayersView({collection:players})
```

Backbone.Router

```
var router = Backbone.Router.extend({

    routes: {
        "home":                     "index",
        "products/:computer":       "display",
        "products/:computer/:model": "display"
    },

    index: function() { },
    display: function(computer, model) { }

});
```

Conclusion

Bad news !

Pour faire du Backbone, il faut connaître Javascript ... et HTML

Pas d'inquiétude

Enorme communauté
Livres
Tutos
...

Troll ignition ;)

Si vous n'y connaissez rien en Javascript ...

Ember.js

Paul Chavart
@tchak13

Sébastien Letélié

@sebmade

"Ce que le web aurait du être si il avait été pensé pour les applications"

Misko Hevery

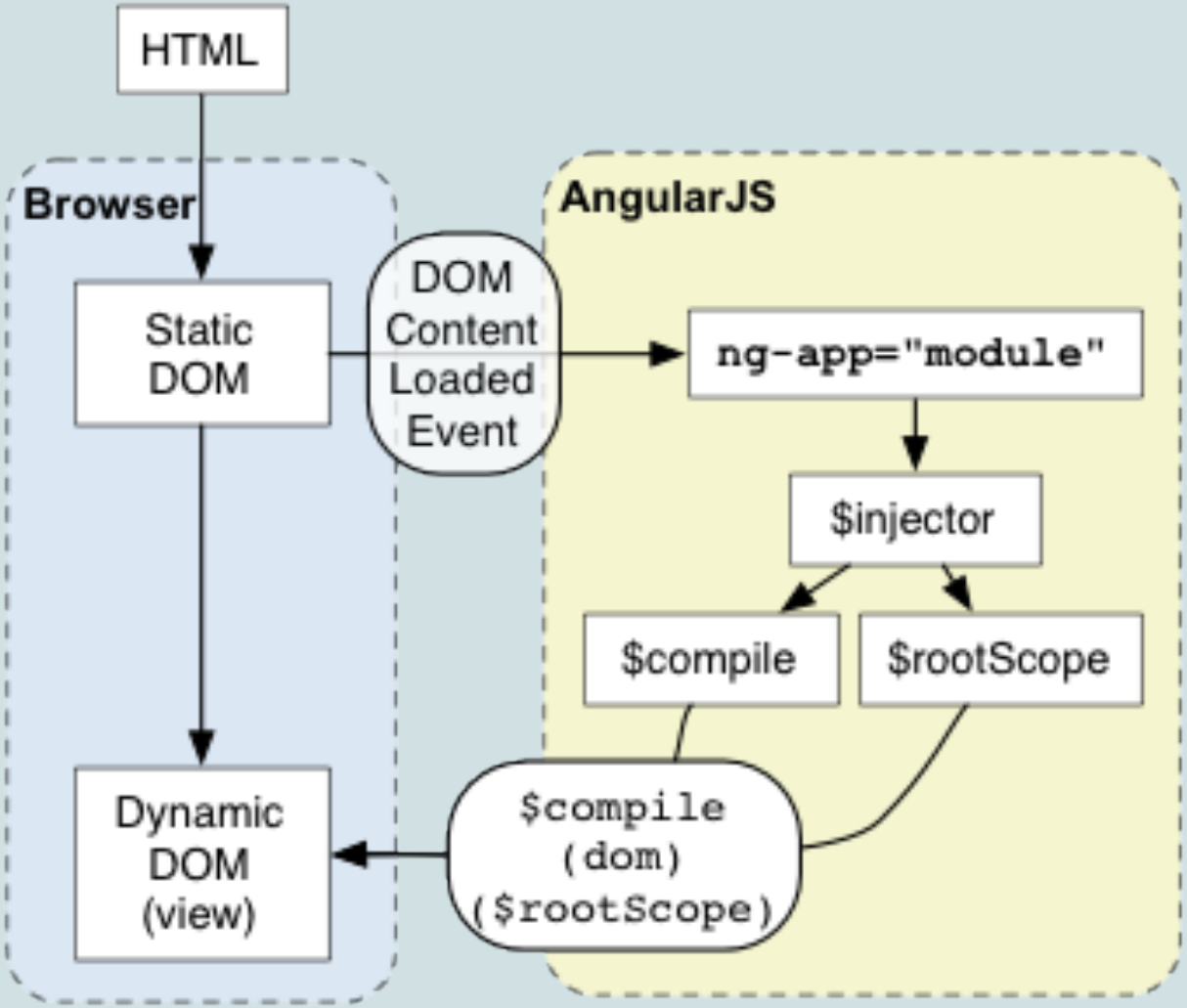
JavaScript

```
angular.module('public.sebmadeApp')
        .controller('MainCtrl', function ($scope, $http, $log) {
  $scope.players = [
    {firstName: 'Sébastien', lastName: 'Letélié', id: 1}
  ];
  $scope.addPlayer = function() {
    ...
  };
};
```

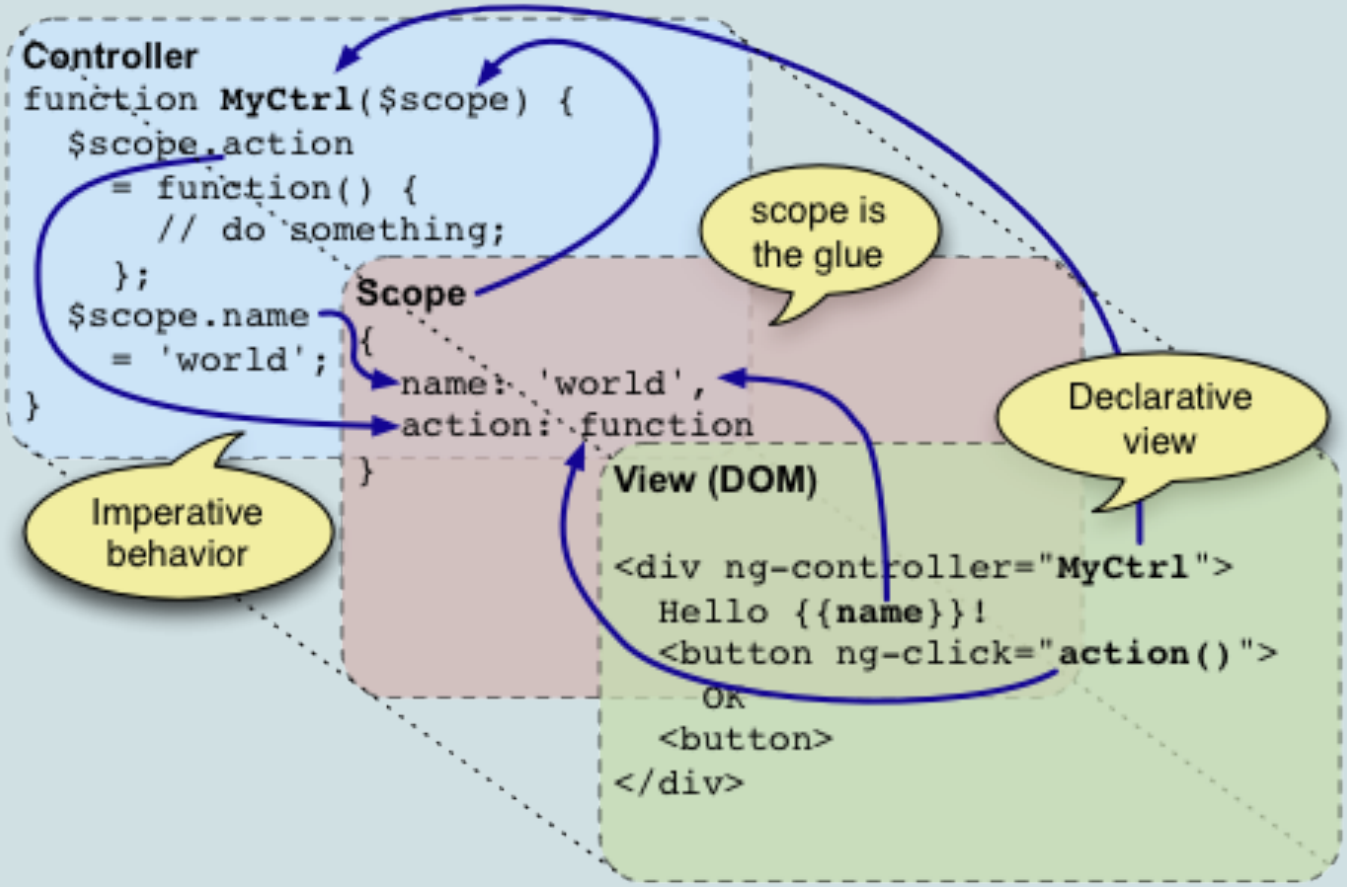
HTML

```
<div ng-controller="MainCtrl" ng-repeat="p in players">
  <a href="#/player/{{p.id}}">
    {{p.firstName}} {{p.lastName}}
  </a>
  <button ng-click="addPlayer()">Ajouter un joueur</button>
</div>
```

Bootstrap

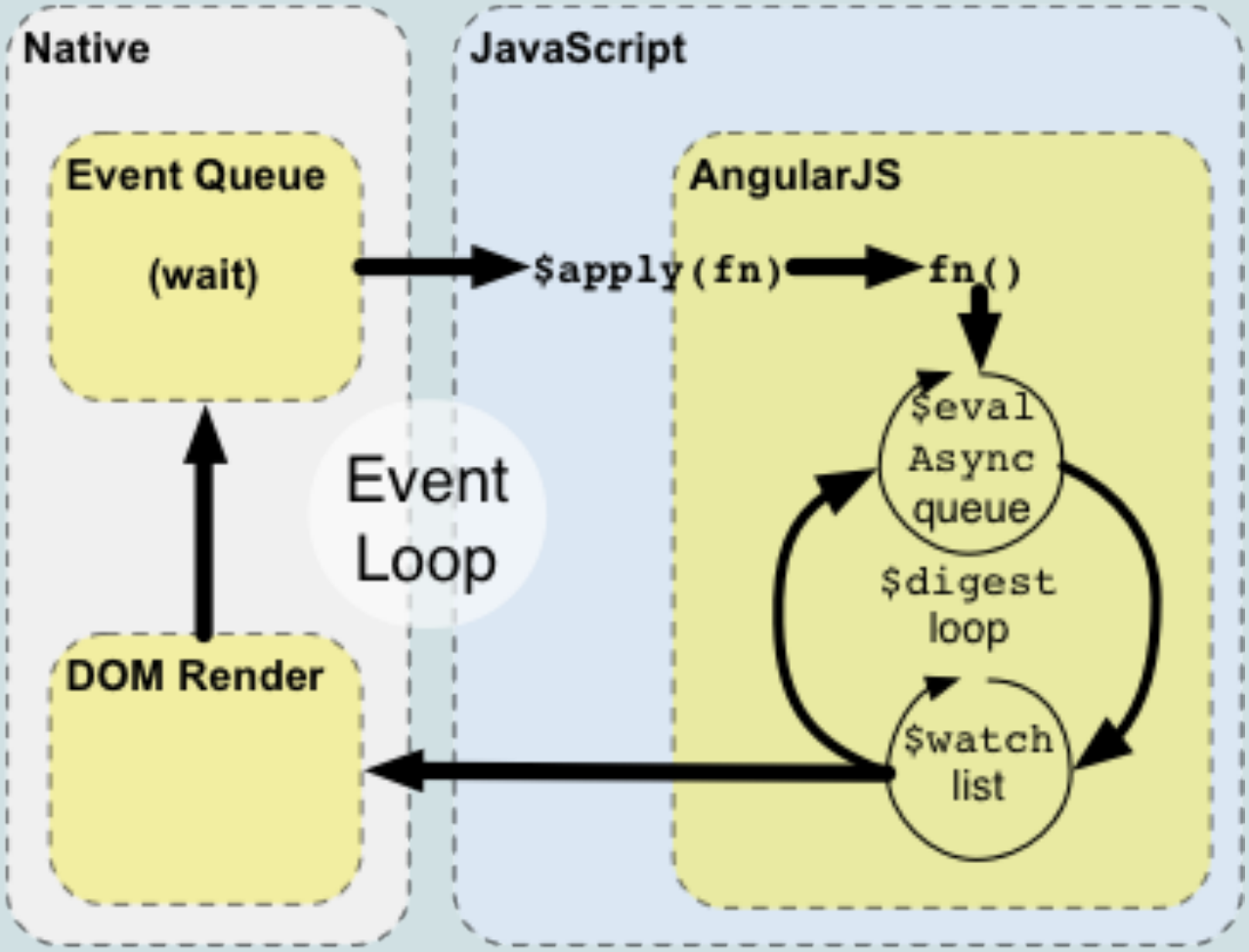

HTML
Browser
Static DOM
DOM Content Loaded Event
AngularJS
ng-app="module"
$injector
$compile
$rootScope
$compile (dom) ($rootScope)
Dynamic DOM (view)

MV* et scope

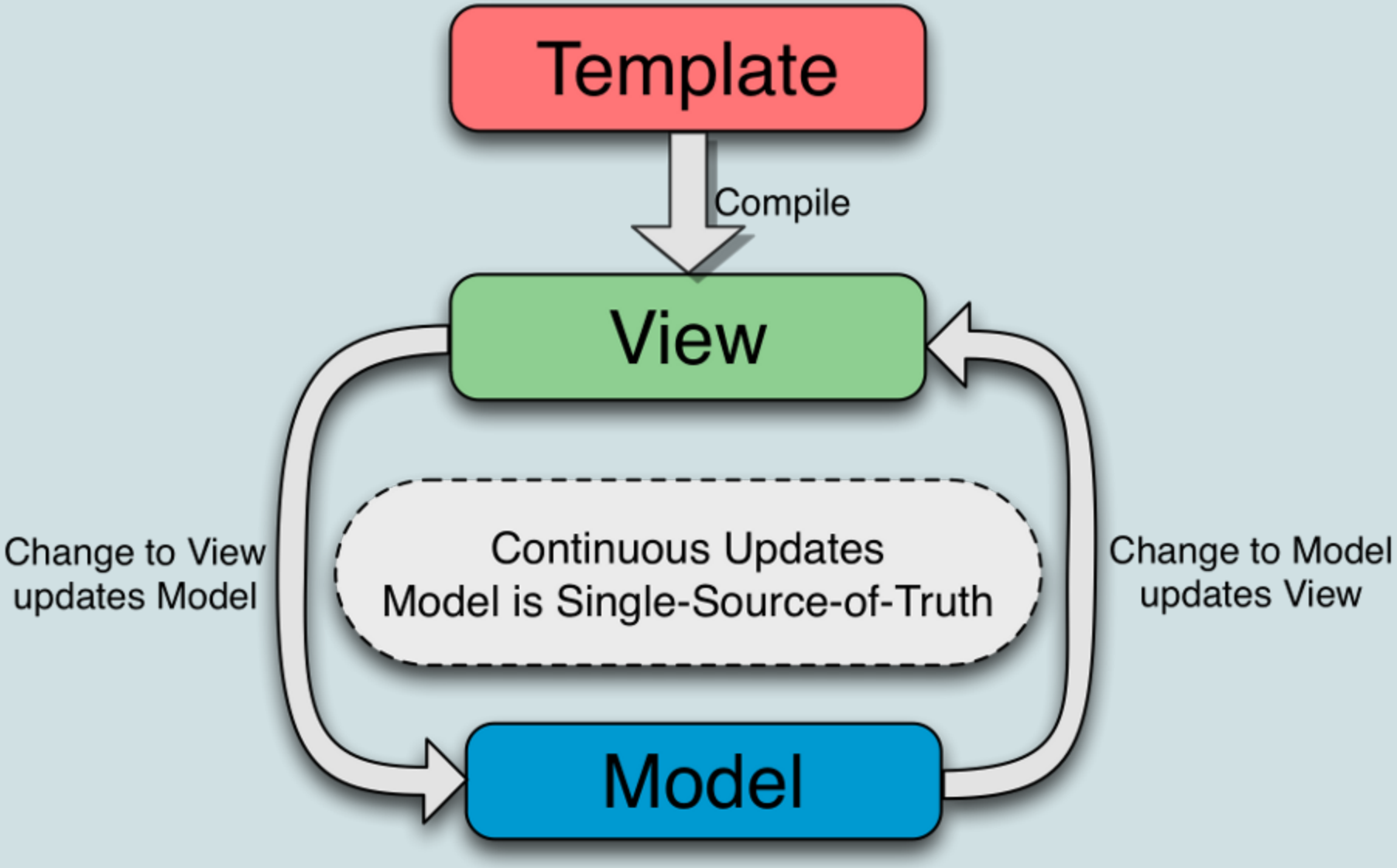

Controller
function MyCtrl($scope) {
$scope.action
= function() {
// do something;
};
$scope.name
= 'world';
}
Scope
{
name: 'world',
action: function
}
scope is the glue
Declarative view
Imperative behavior
View (DOM)

Hello {{name}}!
<button ng-click="action()">
OK
<button>

Runtime


Native
JavaScript
Event Queue
(wait)
$apply(fn)
fn()
AngularJS
$eval
Async
queue
$digest
loop
$watch
list
Event
Loop
DOM Render

Two-Way Data Binding



Directives

De base

ng-click, ng-repeat, ng-include, ng-bind, ng-mouse*, ng-show, ng-hide, ng-switch, ng-form, input[ng-model], ...

Notation

element, attribut ou class

Compatibilité

ng-click = data-ng-click = x-data-ng-click

Directives

<ma-balise/>

```
angular.module('public.sebmadeApp')
       .directive('maBalise', function () {
  return {
    template: '<div></div>',
    restrict: 'E',
    replace: true,
    link: function(scope, element, attrs) {
      ...
      // manipulation du DOM
      // encapsulation de composants externes
      // (ex: jQueryUI, Bootstrap Components)
      ...
    };
  };
};
```

Services

De base

$http, $httpBackend, $log, $location, $route, $routeParams, $rootScope, $timeout, $locale, $q, ...

```
angular.module('public.sebmadeApp')
       .factory('monService', function () {
    var monObjet = ...;
    return monObjet;
  };
};
angular.module('public.sebmadeApp')
       .controller('monCtrl', function (monService) {
    ...
};
```

Filtres

```
<span>Le {{aDate | date:'dd/MM/yyyy'}}</span>
```

```
Le 15/05/2013
```

De base

date, number, currency, filter, orderBy, ...

```
angular.module('public.sebmadeApp')
       .filter(monFiltre, function () {
    return function(input, params...) {
      ...
    };
  };
};
```

Injection = easy to test

```
describe('Controller: MainCtrl', function () {

  // chargement du module
  beforeEach(module('public.sebmadeApp'));

  var MainCtrl, scope;

  // init controleur et mock scope
  beforeEach(inject(function ($controller) {
    scope = {};
    MainCtrl = $controller('MainCtrl', {$scope: scope});
  }));

  it('should init players', function () {
    expect(scope.players.length).toBe(1);
  });
});
```

Routes > ng-view

```
angular.module('public.sebmadeApp', [])
        .config(function ($routeProvider) {
         $routeProvider
              .when('/', {
                  templateUrl: 'views/main.html',
                  controller: 'MainCtrl'
              })
              .when('/player/:id', {
                  templateUrl: 'views/player.html',
                  controller: 'PlayerCtrl'
              })
              .otherwise({
                  redirectTo: '/'
              });
    });
```

REST

$http

```
$http.get('/players')
      .success(function(data) {
        ...
      })
      .error(function(data) {
        ...
      });
```

$resource

```
var Player = $resource('/players/:playerId',
                       {playerId:'@id'});

var player = Player.get({playerId:123}, function() {
                         player.abc = true;
                         player.$save();
                       });
```

+ *déclaratif vs impératif*
+ *bonnes pratiques*
+ *design patterns*
+ *testable*

Je vais briser les 2 autres

Déroulement du battle

Objectif : créer une appli de combat de boxe

1er round

1. créer, modifier, supprimer des joueurs
2. lister les joueurs

2eme round

1. créer des combats en sélectionnant 2 joueurs
2. lister les combats
3. supprimer un combat
4. augmenter ou diminuer le score d'un player dans un combat

Model

Player

- **firstName**:String
- **lastName**:String
- **twitter**:String
- **framework**:String

Fight

- **name**:String
- **opponentOne**:Player
- **opponentTwo**:Player
- **opponentOneScore**:int
- **opponentTwoScore**:int

API REST

GET /players
GET|POST|PUT|DELETE /players/<id>

GET /fights
GET|POST|PUT|DELETE /fights/<id>

A la fin on compte les points