serious #**js** to **infinity** and beyond!

plaatje: http://www.disneypictures.tk/disney-pictures/buzz-lightyear-pictures.html backbonejs

Backbone.js gives structure to web applications by providing models with key-value binding and custom events, collections with a rich API of enumerable functions, views with declarative event handling, and connects it all to your existing API over a RESTful JSON interface.

To wrap it up: backbone provides a rich framework which you can implement to build a professional javascript based application

backbone views

Backbone views are almost more convention than they are code — they don't determine anything about your HTML or CSS for you, and can be used with any JavaScript templating library

```html
<!DOCTYPE html>
<html>
<head>
    <title>Backbone Views</title>
    <link rel="stylesheet" type="text/css" href="css/style.css"/>
</head>
<body>
<a href="#" id="clicker">KLIK</a>

<div id="box"></div>
<script type="text/javascript" src="//ajax.googleapis.com/ajax/libs/
jquery/2.1.0/jquery.min.js"></script>
<script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/
underscore.js/1.6.0/underscore-min.js"></script>
<script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/
backbone.js/1.1.2/backbone-min.js"></script>
<script type="text/javascript" src="js/init.js"></script>
<script type="text/javascript" src="js/views/ClickA.js"></script>
<script type="text/javascript" src="js/views/BoxBlock.js"></script>
<script type="text/javascript" src="js/main.js"></script>
</body>
</html>
```

#JS - STRUCTURE

code

```
//init.js
(function () {
    window.site = {};
    site.$document = $(document);
    site.views = {};
    site.events = _.clone(Backbone.Events); //For global events
})();
```

#JS - STRUCTURE

code

```
//main.js
(function () {
    site.init = function () {
        new site.views.ClickA({el: "#clicker"});
        new site.views.BoxBlock({el: "#box"});
    };

    site.$document.on('ready', site.init);
})();
```

#JS - STRUCTURE

code

assignment

Bekijk hoe **views** werken in Backbone op de website:
http://backbonejs.org/#View

Zorg ervoor dat je de **BoxBlock** & **ClickA** van de vorige keer ombouwd naar een Backbone View. Let hierbij ook goed op het gebruik van je **events property** én je global **site.events** object.

```
site.views.ClickA = Backbone.View.extend({
    events: {
        'click': 'clickHandler'
    },

    initialize: function () {
         //Automaticly called after initialisation of object
    },

    clickHandler: function (e) {
        e.preventDefault();
        site.events.trigger("boxChange");
    }
});
```

#JS - BACKBONE VIEWS

code

```
site.views.BoxBlock = Backbone.View.extend({
    initialize: function () {
        site.events.on("boxChange", this.changeColor, this);
    },

    changeColor: function () {
        this.$el.toggleClass(“blue"); //this.$el has been created for you
    }
});
```

#JS - BACKBONE VIEWS

code

backbone models

Models are the heart of any JavaScript application, containing the interactive data as well as a large part of the logic surrounding it: conversions, validations, computed properties, and access control.

```
<script type="text/javascript" src="js/init.js"></script>
<script type="text/javascript" src="js/models/Settings.js"></script>
<script type="text/javascript" src="js/views/ClickA.js"></script>
```

#JS - STRUCTURE

code

```
(function () {
    window.site = {};
    site.$document = $(document);
    site.views = {};
    site.models = {}; //Place to store our models
    site.events = _.clone(Backbone.Events);
})();
```

#JS - STRUCTURE

code

```
(function () {
    site.init = function () {
        //Create a Model and pass it to our views
        var settings = new site.models.Settings();
        new site.views.ClickA({el: "#clicker", model: settings});
        new site.views.BoxBlock({el: "#box", model: settings});
    };

    site.$document.on('ready', site.init);
})();
```

#JS - STRUCTURE

code

assignment

Bekijk hoe **models** werken in Backbone op de website:
http://backbonejs.org/#Model

Zorg ervoor dat je de **2 views** beide gebruik laat maken van het meegeven model. ClickA **past een property aan**, en BoxBlock **luistert** naar een aanpassing van deze property.

Noem de property '**clickToggle**' en zorg dat hij per click verandert in **true/false** en de kleur van de box daarmee om en om aanpast (rood/blauw bijvoorbeeld)

```
site.models.Settings = Backbone.Model.extend({
   //Default there is no need for any property
});
```

#JS - BACKBONE MODELS

code

```
site.views.ClickA = Backbone.View.extend({
    clickToggle: false, //State variable

    events: {
        'click': 'clickHandler'
    },

    initialize: function () {

    },

    /**
     * @param e
     * @see site.views.ClickA.events
     */
    clickHandler: function (e) {
        e.preventDefault();

        this.clickToggle = !this.clickToggle; //Change the state
        this.model.set({clickToggle: this.clickToggle}); //Change property
    }
});
```

#JS - BACKBONE MODELS

code

```
site.views.BoxBlock = Backbone.View.extend({
    initialize: function () {
        //Listen to a change of a property
        this.model.on("change:clickToggle", this.changeColor, this);
    },

    /**
     * @see site.views.BoxBlock.initialize
     */
    changeColor: function (model, clickToggle) {
        //Properties are passed automatically, use property for magic
        if (clickToggle) {
            this.$el.addClass("blue");
        } else {
            this.$el.removeClass("blue");
        }
    }
});
```

#JS - BACKBONE MODELS

code

backbone models data

```
site.models.Matches = Backbone.Model.extend({
    url: 'http://docent.cmi.hr.nl/moora/imp03/api/wedstrijden'
});
```

#JS - BACKBONE MODELS

code

assignment

Verwijder eerst je vorige Model, en maak een nieuwe aan genaamd ‘**Matches.js**’. Geef deze mee aan BoxBlock binnen je main.js.

Probeer na een klik op ClickA, binnen BoxBlock data van je model in te laden via de **fetch** method. Geef ook data properties mee genaamd ‘**league**’ en ‘**team**’ om je set aan data te beperken.

console.log de output om te zien dat je de data successvol hebt ingeladen.

```
site.views.ClickA = Backbone.View.extend({
    events: {
        'click': 'clickHandler'
    },

    initialize: function () {

    },

    /**
     * @param e
     * @see site.views.ClickA.events
     */
    clickHandler: function (e) {
        e.preventDefault();
        site.events.trigger("boxChange");
    }
});
```

#JS - BACKBONE MODELS

code

```
site.views.BoxBlock = Backbone.View.extend({
    initialize: function () {
        site.events.on("boxChange", this.changeColor, this);
    },

    /**
     * @see site.views.BoxBlock.initialize
     */
    changeColor: function () {
        this.$el.addClass("blue");
        this.loadMatches();
    },

    /**
     * Wrapper function to load the matches through the model
     */
    loadMatches: function () {
        this.model.fetch({
            success: _.bind(this.loadMatchesSuccessHandler, this),
            error: _.bind(this.loadMatchesErrorHandler, this),
            data: {
                league: 'PrimeraDivision',
                club: 'Getafe'
            }
        });
    },
```

#JS - BACKBONE MODELS

code

```
    /**
     * @param model
     * @param response
     * @param options
     */
    loadMatchesSuccessHandler: function (model, response, options) {
        console.log("SUCCESS");
        console.dir(model);
        console.dir(response);
        console.dir(options);
    },

    /**
     * @param model
     * @param response
     * @param options
     */
    loadMatchesErrorHandler: function (model, response, options) {
        console.log("ERROR");
        console.dir(model);
        console.dir(response);
        console.dir(options);
    }
});
```

#JS - BACKBONE MODELS

code

huiswerk

Zie github:

- Backbone.js website & voorbeelden (http://backbonejs.org)
- From jQuery to Backbone, beetje verouderd maar goed voor je inzicht! (https://github.com/kjbekkelund/writings/blob/master/published/understanding-backbone.md)

- Bouw je View code van je ToDo list om naar Backbone Views.
- Probeer eens een PHP Backend op te zetten die todo items in je database opslaat via Backbone Models