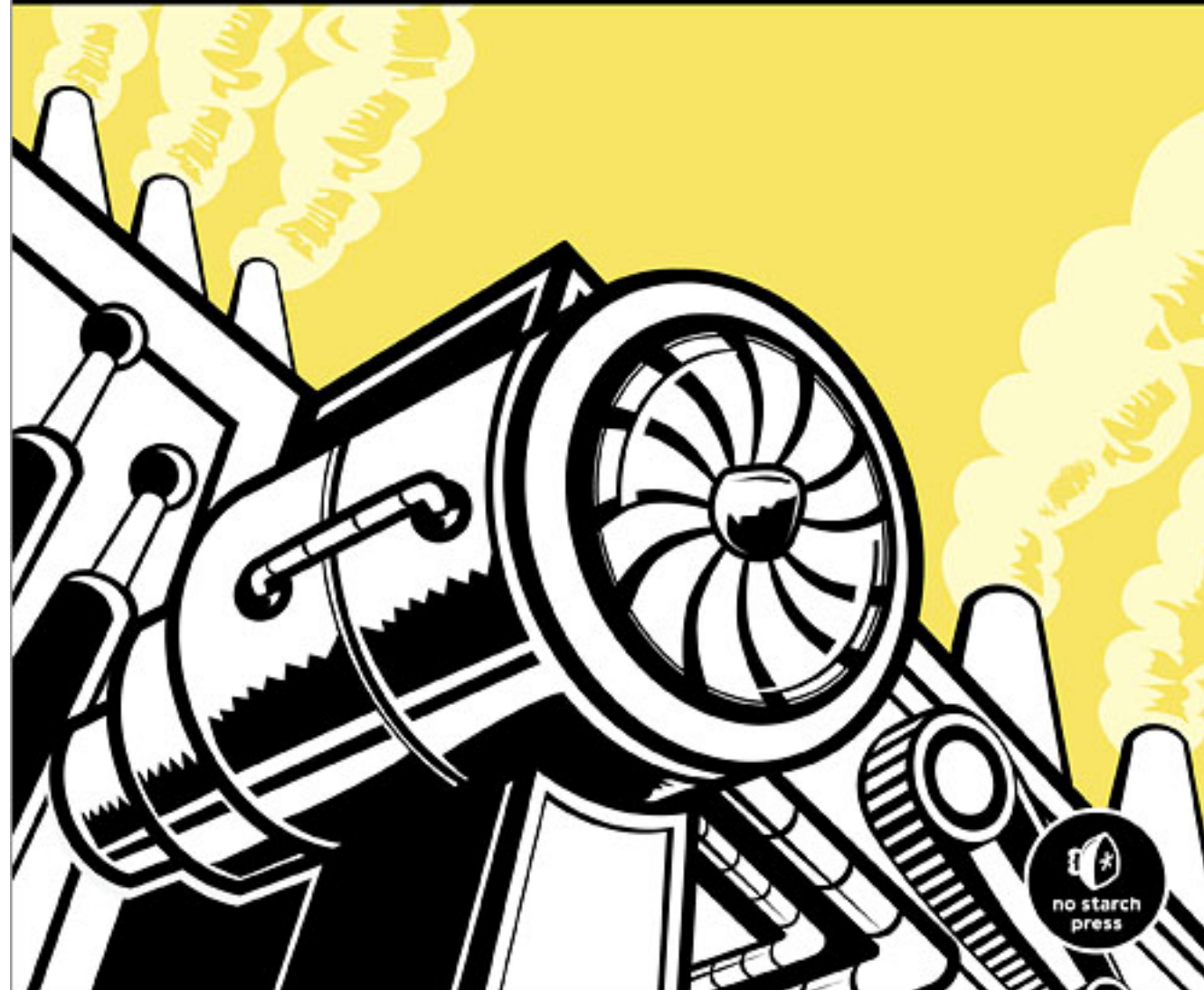


serious **#js**
to **infinity** and beyond!



THE PRINCIPLES OF **OBJECT-ORIENTED** **JAVASCRIPT**

NICHOLAS C. ZAKAS



views are wrappers around
HTML elements. 1 view should be
responsible for 1 selector

```
//Create some logic structure within your project
<script src="js/init.js" type="text/javascript"></script>
<script src="js/views/View1.js" type="text/javascript"></script>
<script src="js/views/View2.js" type="text/javascript"></script>
<script src="js/main.js" type="text/javascript"></script>
```

```
//Base structure to create a global scope variable (init.js)
(function () {
    window.site = {};
    site.views = {};
})();

//Now you're able to create views within the site.views scope:
//Every view should be in it's own file, just like any other object
site.views.View1 = function(){
    //Logic
}
```

assignment

Maak 2 views, **BoxBlock** & **ClickA**. Beide zijn verantwoordelijk voor 1 element.

BoxBlock is verantwoordelijk voor een `<div>` met het id 'box', en **ClickA** is verantwoordelijk voor een `<a>` met het id 'clicker'.

Doe dit met native Javascript & initialiseer de Views binnen je main.js

```
//main.js where we initiate the 2 views.  
(function () {  
    site.init = function () {  
        new site.views.ClickA(document.getElementById("clicker"));  
        new site.views.BoxBlock(document.getElementById("box"));  
    };  
  
    window.addEventListener('load', site.init);  
})();
```



```
site.views.BoxBlock = function (el) { //Same goes for ClickA
  this.el = el;

  this.init = function () {
    //From now on we have this.el to use our element for anything
    console.log(this.el);
  };

  this.init();
};
```

assignment

Zorg er nu eens voor dat wanneer je op de link klikt, de box een andere kleur krijgt. Je moet hiervoor met een **CustomEvent** gaan werken.

```
site.views.ClickA = function (el) {
  this.el = el;
  //Custom event initialisation
  this.event = new CustomEvent("boxChange");

  this.init = function () {
    //Normal click event on this.el
    this.el.addEventListener('click', this.clickHandler.bind(this));
  };

  this.clickHandler = function (e) {
    //Prevent default behaviour & dispatch event for others to listen
    e.preventDefault();
    document.dispatchEvent(this.event);
  };

  this.init();
};
```

```
site.views.BoxBlock = function (el) {  
    this.el = el;  
  
    this.init = function () {  
        //Listen to event & bind custom function, just like normal events  
        document.addEventListener("boxChange", this.changeColor.bind(this));  
    };  
  
    this.changeColor = function () {  
        this.el.classList.add('blue');  
    };  
  
    this.init();  
};
```

assignment

Maak nu eens de vertaalslag van native Javascript naar jQuery. Voeg de jQuery library toe en bouw de code om naar **jQuery syntax**.

```
(function () {  
  window.site = {};  
  //We can use wrappers for anything we will use more than once.  
  //Think about $('body'), $(window), etc.  
  site.$ = jQuery;  
  site.$document = site.$(document);  
  site.views = {};  
})();
```

```
(function () {  
    site.init = function () {  
        new site.views.ClickA($("#clicker")); //jQuery selectors  
        new site.views.BoxBlock($("#box"));  
    };  
  
    site.$(site.init); //jQuery ready event  
})();
```

```
site.views.ClickA = function ($el) {  
    this.$el = $el;  
  
    this.init = function () {  
        this.$el.on('click', this.clickHandler); //on handler  
    };  
  
    this.clickHandler = function (e) {  
        e.preventDefault();  
        //Trigger can be used, it expects custom events by default  
        site.$document.trigger("boxChange");  
    };  
  
    this.init();  
};
```



```
site.views.BoxBlock = function ($el) {  
    this.$el = $el;  
  
    this.init = function () {  
        // .on & using $.proxy instead of .bind  
        site.$document.on("boxChange", $.proxy(this.changeColor, this));  
    };  
  
    this.changeColor = function () {  
        this.$el.addClass("blue");  
    };  
  
    this.init();  
};
```


underscore.js is a library with a lot of utility functions. If you only use 1 function, it's always better to use custom Javascript for performance reasons

underscore is also the Backbone of the backbone.js framework, developed by the same development team

```

/** Some source code of underscore */

// The cornerstone, an `each` implementation, aka `forEach`.
// Handles objects with the built-in `forEach`, arrays, and raw
objects.
// Delegates to **ECMAScript 5**'s native `forEach` if available.
var each = _.each = _.forEach = function(obj, iterator, context) {
  if (obj == null) return obj;
  if (nativeForEach && obj.forEach === nativeForEach) {
    obj.forEach(iterator, context);
  } else if (obj.length === +obj.length) {
    for (var i = 0, length = obj.length; i < length; i++) {
      if (iterator.call(context, obj[i], i, obj) === breaker) return;
    }
  } else {
    var keys = _.keys(obj);
    for (var i = 0, length = keys.length; i < length; i++) {
      if (iterator.call(context, obj[keys[i]], keys[i], obj) ===
breaker) return;
    }
  }
  return obj;
};

```

```
var data = [  
  {  
    title: "Article 1",  
    order: 3,  
    status: 1  
  },  
  {  
    title: "Article 2",  
    order: 1,  
    status: 0  
  },  
  {  
    title: "Article 3",  
    order: 2,  
    status: 1,  
    awesome: true  
  }  
];
```

```
//Alter data with the _.each method, using _.random to create different
numbers for our images
_.each(data, function (item, index, list) {
  item.imageUrl = "http://lorempixel.com/" + _.random(200, 600) + "/" +
_.random(200, 600) + "/";
});

//Filter data by property with _.filter
var publishedData = _.filter(data, function (item, index, list) {
  return item.status == 1;
});
```

```
//Native
document.getElementById("title").addEventListener('click',
this.click.bind(this));

//Native with _.bind
document.getElementById("title").addEventListener('click',
_.bind(this.click, this));

//jQuery
$("#title").on('click', $.proxy(this.click, this));

//jQuery with _.bind
$("#title").on('click', _.bind(this.click, this));
```

```
//Extending the _ functionality with your own Utils
_.mixin({
  isIE8OrLower: function () {
    return navigator.userAgent.match(/MSIE\s/) !== null ? document.all
    && !document.addEventListener : false;
  }
});

//Now available within the _ namespace
console.log("_mixin:_.isIE8OrLower", _.isIE8OrLower());
```


assignment

Probeer eens de `_.template` method te gebruiken. Check <http://underscorejs.org/> voor een voorbeeld implementatie.

Gegeven is:

`$.get('templates/articles.html', articlesHtmlLoaded);`

Creeër een bovenstaande map en bestand, en zorg dat de `articlesHtmlLoaded` de artikelen data (van de eerder gegeven dummy data) op je scherm toont (in een `<div>` met id 'articles'). Je zult hiervoor de underscore template syntax moeten gebruiken.

```
var data = [  
  {  
    title: "Article 1",  
    order: 3,  
    status: 1  
  },  
  {  
    title: "Article 2",  
    order: 1,  
    status: 0  
  },  
  {  
    title: "Article 3",  
    order: 2,  
    status: 1,  
    awesome: true  
  }  
];  
  
//Filter data by property with _.filter  
var publishedData = _.filter(data, function (item, index, list) {  
  return item.status == 1;  
});
```

```
<% _.each(articles, function(article, index, list){ %>
<div class="article">
  <h2><%= article.title %></h2>
  "/>
</div>
<% }) %>
```

```
//Using _.template to generate HTML without any HTML in JS,  
//HTML needs to be loaded first from external template file  
function articlesHtmlLoaded(html) {  
    var articlesHtml = _.template(html, {articles: publishedData});  
    $('#articles').append(articlesHtml);  
}  
  
$.get('templates/articles.html', articlesHtmlLoaded);
```

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 waar nodig je ToDo list code om zodat het concept van Views er goed in verwerkt zit
- Pas templating toe zodat nieuwe TodoList items netjes worden toegevoegd, geen HTML meer in je Javascript code dus!