actibyti] a netmind social project

BACKBONE

Autor: Ricardo Ahumada

GOBIERNO DE ESPAÑA
MINISTERIO DE ENERGÍA, TURISMO Y AGENDA DIGITAL
red.es
ESTRATEGIA DE EMPRENDIMIENTO Y EMPLEO JOVEN garantía juvenil
Agenda Digital para España
UNIÓN EUROPEA
Fondo Social Europeo
"El FSE invierte en tu futuro"

Backbone

- BackboneJS es una librería ligera de JavaScript que permite desarrollar y estructurar aplicaciones cliente que se ejecutan en un navegador web. Ofrece un marco MVC que abstrae los datos en modelos, DOM (Document Object Model) en vistas y enlaza estos dos eventos de uso.

- Observe que BackboneJS no es un framework sino una librería. La diferencia es quién tiene el control. Usando una librería USTED tiene el control, pero usando un framework hay una inversión de control. Las librerías le dan mucha flexibilidad, mientras que los framework tienen opiniones sobre las maneras de hacer las cosas.

- BackboneJS fue desarrollado por Jeremy Ashkenas y fue lanzado inicialmente el 13 de octubre de 2010.

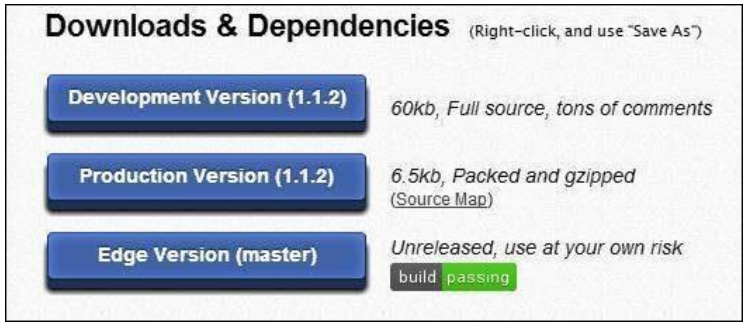
Backbone

- Cuándo utilizar Backbone:
  - Considere que está creando una aplicación con numerosas líneas de código usando JavaScript o jQuery. En esta aplicación, si usted desea añadir o reemplazar elementos DOM a la aplicación o hacer algunas peticiones o mostrar animación en la aplicación o agregar más número de líneas a su código, Entonces su aplicación puede ser complicada trabajarla con backbone.
  - Si desea un diseño con menos código, es mejor utilizar la librería BackboneJS que proporciona una buena funcionalidad, está bien organizada y en forma estructurada para desarrollar su aplicación.
  - BackboneJS se comunica a través de eventos. Esto asegura que no estropeara la aplicación. Su código será más limpio, más agradable y fácil de mantener.
- Características:
  - BackboneJS permite el desarrollo de aplicaciones y el frontend de una manera mucho más fácil mediante el uso de funciones JavaScript.
  - BackboneJS proporciona varios bloques de construcción tales como modelos, vistas, eventos, routers y colecciones para ensamblar las aplicaciones web del lado del cliente.
  - Cuando un modelo cambia, automáticamente actualiza el HTML de su aplicación.
  - BackboneJS es una biblioteca sencilla que ayuda a separar la lógica de la interfaz de usuario y de negocio.
  - Es una biblioteca libre y de código abierto y contiene más de 100 extensiones disponibles.

Configuración de Backbone

- BackboneJS es muy fácil de configurar y trabajar. BackboneJS se puede utilizar de las dos maneras:
  - Descargando la librería de la UI desde su sitio web oficial.
  - Descarga de la librería de la UI desde CDN.

- Descargando la biblioteca de la UI desde su página oficial
  - Se abre el enlace http://backbonejs.org/, se verá una captura de pantalla como esta:



  - Como se puede ver, hay tres opciones para descargar de esta librería:
    - Versión de desarrollo : se obtendrá la librería completa de JavaScript.
    - Versión de producción: se obtendrá el archivo de la librería Backbone-min.js que está comprimido.
    - Versión edge: se obtendrá una versión inédita, es decir, que esta en desarrollo; Por lo tanto, necesita usarlo bajo su propio riesgo.

Configuración de Backbone

- Descarga de la librería desde de la UI desde CND
  - Un CDN o Content Delivery Network es una red de servidores diseñados para servir archivos a los usuarios. Si utiliza un enlace CDN en su página web, se traslada la responsabilidad de alojar archivos de sus propios servidores a una serie de externos. Esto también ofrece una ventaja de que si el visitante de su página web ya ha descargado una copia de BackboneJS del mismo CDN, no tendrá que volver a descargarlo.

```
<script type = "text/javascript" src = "https://ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.min.js">
</script>
<script type = "text/javascript" src = "https://ajax.cdnjs.com/ajax/libs/underscore.js/1.1.4/underscore-
min.js"></script>
<script type = "text/javascript" src = "https://ajax.cdnjs.com/ajax/libs/backbone.js/0.3.3/backbone-
min.js"></script>
```

- BackboneJS depende de los siguientes archivos de JavaScript:
  - Underscore.js
  - JQuery.js
  - Json2.js

Primera app en Backbone

- Hola mundo en BackboneJs, crearemos un ejemplo sencillo.

```
<!DOCTYPE html>
  <html>
    <head>
      <meta charset = "UTF-8">
      <meta http-equiv = "X-UA-Compatible" content = "IE = edge,chrome = 1">
      <title>Hello World using Backbone.js</title>
    </head>
    <body>
      <!-- Your HTML -->
        <div id = "container">Loading...</div>
      <!-- Libraries -->
        <script src = "https://code.jquery.com/jquery-2.1.3.min.js" type = "text/javascript"></script>
        <script src = "https://cdnjs.cloudflare.com/ajax/libs/underscore.js/1.3.3/underscoremin.js" type =
"text/javascript"></script>
        <script src = "https://cdnjs.cloudflare.com/ajax/libs/backbone.js/0.9.2/backbone-min.js" type =
"text/javascript"></script>
      <!-- Javascript code -->
        <script type = "text/javascript"> var AppView = Backbone.View.extend ({
// el - Cada vista tiene un elemento asociado con contenido HTML, que se mostrará.
el: '#container',
// Es la primera función llamada cuando esta vista es instanciada.
initialize: function() { this.render(); },
// $el - Es un objeto jQuery en caché (el), en el que puede utilizar las funciones jQuery para empujar el contenido.
render: function() { this.$el.html("Hola Mundo!!!"); } });
var appView = new AppView(); </script>
    </body>
  </html>
```

Primera app en Backbone

- Hay un código html al inicio de la etiqueta <body> que imprime “Loading…:

```
<div id = "container">Loading...</div>
```

- A continuación, hemos añadido las siguientes CDN

```
<script src = "https://code.jquery.com/jquery-2.1.3.min.js" type = "text/javascript"></script>
<script src = "https://cdnjs.cloudflare.com/ajax/libs/underscore.js/1.3.3/underscore-min.js" type = "text/javascript"></script>
<script src = "https://cdnjs.cloudflare.com/ajax/libs/backbone.js/0.9.2/backbone-min.js" type = "text/javascript"></script>
```

- Y seguido del siguiente script:

```
<script type = "text/javascript"> var AppView = Backbone.View.extend ({
// el - Cada vista tiene un elemento asociado con contenido HTML, que se mostrará.
el: '#container',
// Es la primera función llamada cuando esta vista es instanciada.
initialize: function() { this.render(); },
// $el - Es un objeto jQuery en caché (el), en el que puede utilizar las funciones jQuery para empujar el contenido.
render: function() { this.$el.html(“Hola Mundo!!!"); } });
var appView = new AppView(); </script>
```

Primera app en Backbone

- Al guardar en ejemplo y ejecutarlo nos dará por consiguiente:

```
Hola Mundo!!!
```

- Como se pudo observar, el script introdujo el “hola mundo!!!” en la etiqueta “ ” mencionada en el “<body>” del HTML, aun así cuando ya esta etiquita “ ” tenia para mostrarnos “loading…”

Backbone.Model

- La tabla siguiente enumera todos los métodos que puede utilizar para manipular el BackboneJS-Model -

Módulo	Descripción
extend	Extiende la clase backbone.Model mientras crea su propio modelo de backbone.
initialize	Cuando se crea una instancia de modelo, se llama al constructor de la clase y se invoca definiendo la función initialize.
get	Obtiene el valor de un atributo en el modelo.
set	Establece el valor de un atributo en el modelo.
escape	Devuelve la versión HTML del atributo de un modelo.
has	Devuelve true si el valor de atributo se define con valor no nulo o no definido.
unset	Elimina un atributo de un modelo de columna.
clear	Elimina todos los atributos, incluido el atributo id de un modelo de backbone.
attributes	Define la propiedad de un modelo.
toJSON	Devuelve una copia de los atributos como un objeto para la cadena JSON.
url	Devuelve la url donde se encuentra el recurso del modelo

Backbone.Model

- Hay seis métodos de Underscore.js que proporcionan su funcionalidad para ser utilizados en el Backbone.Model.

Módulo	Descripción
_.keys(object)	Se utiliza para acceder a las propiedades enumerables del objeto.
_.values(object)	Se utiliza para obtener valores de las propiedades del objeto.
_.pairs(object)	Describe las propiedades del objeto en términos de pares de valores clave.
_.invert(object)	Devuelve la copia del objeto, en la que las claves se han convertido en valores y viceversa.
_.pick(object, *keys)	Devuelve la copia del objeto e indica qué teclas recoger.
_.omit(object, *keys)	Devuelve la copia del objeto e indica qué teclas se deben omitir.

Backbone.Model

- Los modelos son el corazón de cada aplicación. Contiene los datos interactivos y la lógica que lo rodea, como validación de datos, getters y setters, valores por defecto, inicialización de datos, conversiones, etc. Para nuestro ejemplo, vamos a crear un modelo llamado Todo, que almacenará una cadena de texto (título) y si la tarea se ha completado o no.

```
var app = {}; // Crear nombre para nuestra aplicación
app.Todo = Backbone.Model.extend({
defaults: {
title: '',
completed: false
}
});
```

- Tenga en cuenta que para las clases de convención los nombres son mayúsculas, mientras que las variables de instancia y los objetos no lo son. Otro aspecto importante de los modelos es que sus propiedades son dinámicas; Se pueden crear sobre la marcha y no tiene ningún tipo específico asociado.

Backbone.Model

- Después de completar el fragmento de código anterior, puede abrir la consola del navegador (consola de chrome: ctrl + shift + i) y probar esto para familiarizarse con los modelos:

```
var todo = new app.Todo({title: 'Learn Backbone.js', completed: false}); // Crear objeto con los atributos especificados.
todo.get('title'); // " Aprenda Backbone.js "
todo.get('completed'); // falso
todo.get('created_at'); // indefinido
todo.set('created_at', Date());
todo.get('created_at'); // "Wed Sep 12 2012 12:51:17 GMT-0400 (EDT)"
```

Backbone.Collection

- Como su nombre lo indica, las colecciones se ordenan con conjuntos de modelos, donde puede obtener y establecer modelos en la colección, escuchar eventos cuando cambia cualquier elemento de la colección y buscar datos del modelo desde el servidor.

- Las colecciones permiten guardar datos (en base de datos, archivo, memoria), y requiere una referencia a él. Por lo tanto, debe especificar el parámetro url con una url relativa, donde el recurso del modelo se ubicaría en el servidor. De lo contrario, obtendrá errores como:

```
A "url" property or function must be specified
```

Backbone.Collection

- No vamos a utilizar un servidor por simplicidad; En lugar de eso vamos a utilizar el almacenamiento local de HTML5 para la persistencia a través del complemento de Backbone. Por lo tanto, tenemos que definir la propiedad localStorage en lugar de URL. Debe incluir el backbone-localstorage.js con el resto de sus librerías como se muestra en el código com

```
<script src="http://cdnjs.cloudflare.com/ajax/libs/backbone-
localstorage.js/1.0/backbone.localStorage-min.js" type="text/javascript“>
app.TodoList = Backbone.Collection.extend({
model: app.Todo,
localStorage: new Store("backbone-todo")
});
// instance of the Collection
app.todoList = new app.TodoList();
```

```
var todoList = new app.TodoList()
todoList.create({title: 'Learn Backbone\'s Collection'}); //Aviso: `completado` se establecerá en falso por defecto.
var lmodel = new app.Todo({title: 'Learn Models', completed: true});
todoList.add(lmodel);
todoList.pluck('title'); // ["Learn Backbone's Collection", "Learn Models"]
todoList.pluck('completed'); // [false, true]
JSON.stringify(todoList);
```

Backbone.View

- Views no tiene las marcas HTML para nuestra aplicación, sino que en su lugar (es como el controlador en frameworks MVC) procesa datos y los vincula a plantillas y finalmente procesar HTML basado en eventos o cambios de datos.

- Propiedades básicas: Hay 4 propiedades básicas en una vista:
  - El
  - Initialize
  - Render
  - Events

Backbone.View

- Views no tiene las marcas HTML para nuestra aplicación, sino que en su lugar (es como el controlador en frameworks MVC) procesa datos y los vincula a plantillas y finalmente procesar HTML basado en eventos o cambios de datos.
- Propiedades básicas: Hay 4 propiedades básicas en una view:
  - El
  - Initialize
  - Render
  - Events
- ¿Recuerdas el Hola Mundo!!!?:

```
<script type = "text/javascript"> var AppView = Backbone.View.extend ({
// el - Cada vista tiene un elemento asociado con contenido HTML, que se mostrará.
el: '#container',
// Es la primera función llamada cuando esta vista es instanciada.
initialize: function() { this.render(); },
// $el - Es un objeto jQuery en caché (el), en el que puede utilizar las funciones jQuery para empujar el
contenido.
render: function() { this.$el.html("Hola Mundo!!!"); } });
var appView = new AppView(); </script>
```

Backbone.View

- View.el
  - Cada vista tiene que hacer referencia a un DOM en todo momento. Por lo tanto, la vista inyectará contenido en este elemento. Esta es la propiedad el This.el se crea a partir de las propiedades el, tagName, className, id o attributes. Si ninguno de estos se especifica, entonces this.el es una vacía. La view.$el es un objeto jQuery en caché del elemento de la vista (view.el).
- Initialize/constructor
  - Aquí tienes la opción de pasar parámetros que se adjuntarán a un modelo, colección o view.el.
- Render
  - Esta función inyecta el marcado en los elementos. No todas las view requieren tener una función de render, como se va a ver en el código de ejemplo, pueden llamar a las funciones de render de otra vista.

Backbone.View

- Events
  - Los eventos se escriben en el siguiente formato:

```
{"<EVENT_TYPE> <ELEMENT_ID>": "<CALLBACK_FUNTION>"}
```

Ejemplo:

```
events: {'keypress #new-todo': 'createTodoOnEnter'}
```

  - En jQuery sería:

```
$('#new-todo').keypress(createTodoOnEnter);
```

Backbone.Events

- Este módulo se puede mezclar con cualquier objeto y darle el comportamiento pub / sub (observador patter). Los eventos además proporcionan un par de métodos: on, off y trigger.
- Para hacer el llamado a un evento, debemos seguir la siguiente sintaxis:

```
object.on(event, callback, [context])
```

Ejemplo:

```
todoList.on('add', this.addAll, this);
```

BananaTube: Decisión en equipo

- Discute en equipo la conveniencia de usar Backbone para modelar BananaTube
- Sería interesante reemplazar Angular o React por las opciones que ofrece Backbone?
- Tomad nota de las fortalezas y debilidades que hayas observado en el framework, así como, la dificultad de incorporarlo en el proyecto.

netmind

WeKnowIT

Barcelona

C. Almogàvers, 123
08018 Barcelona
Tel. 93 304.17.20
Fax. 93 304.17.22

Madrid

Plaza Carlos Trías Bertrán, 7
28020 Madrid
Tel. 91 442.77.03
Fax. 91 442.77.07

www.netmind.es

GOBIERNO DE ESPAÑA
MINISTERIO DE ENERGÍA, TURISMO Y AGENDA DIGITAL
red.es
ESTRATEGIA DE EMPRENDIMIENTO Y EMPLEO JOVEN garantía juvenil
Agenda Digital para España
UNIÓN EUROPEA
Fondo Social Europeo
"El FSE invierte en tu futuro"