

Bloque 4

Programación con JavaScript



Anexo: Handlebars

Programación con JavaScript
Cefire 2017/2018
Autores: Alejandro Amat Reina
Arturo Bernal Mayordomo

Índice

Handlebars.js.....	4
Sustitución de valores.....	4
Condicionales (if, unless).....	5
El bloque with.....	5
Iteradores (each).....	5
Generar HTML (Webpack).....	6

Handlebars.js

[Handlebars.js](#) es una librería de JavaScript (que está basada en [Mustache](#)) pensada para generar HTML de forma automática a partir de plantillas predefinidas y datos almacenados en JSON (recibido, por ejemplo, de un servicio web).

Una plantilla no es más que código HTML con ciertas expresiones encerradas entre llaves dobles `{{expresión}}` que se evalúan en base a, o se sustituyen por valores almacenados en las propiedades de un objeto JSON.

Instalaremos Handlebars con NPM en nuestro proyecto como dependencia de producción (si se quiere, también de forma global):

npm i handlebars -S

A continuación vamos a ver como crear plantillas básicas y precompilarlas usando un script de NPM (lo que mejora mucho el rendimiento frente a compilarlas directamente desde código JavaScript), además de usarlas.

Sustitución de valores

La plantilla Handlebars es en realidad una estructura HTML que vamos a rellenar. Lo más simple que tienen esta plantilla es la sustitución de valores. En este caso, vamos a poner el nombre de una propiedad del objeto JSON que contendrá los datos entre llaves dobles.

Si esta es mi plantilla:

```
<p>{{nombre}}</p>
<p>{{edad}}</p>
```

Y este un objeto JSON:

```
{
  nombre: "Pepe",
  edad   : 43
}
```

Al pasarle ese objeto a la plantilla de Handlebars, el HTML resultante sería:

```
<p>Pepe</p>
<p>43</p>
```

También podemos usar propiedades anidadas:

```
<p>{{nombre}}</p>
<p>{{edad}}</p>
<p>{{direccion.calle}}, {{direccion.ciudad}}</p>
```

Objeto JSON

```
{
```

```

    "nombre": "Pepe",
    "edad" : 43,
    "direccion": {
        "calle": "Calle Ancora 15",
        "ciudad": "Madrid"
    }
}

```

Resultado:

```

<p>Pepe</p>
<p>43</p>
<p>Calle Ancora 15, Madrid</p>

```

Condicionales (if, unless)

Dentro de una plantilla podremos usar el condicional **if..else**. En este caso funciona de manera limitada ya que se limita a verificar si un valor no existe o es equivalente a falso (null, 0, "", []). Sólo imprimirá el HTML que contenga si la propiedad no se evalúa a falso (en ese caso si hay un else, se imprime ese trozo).

```

<p>{{nombre}}</p>
<p>{{edad}}</p>
{{#if direccion}}
    <p>{{direccion.calle}}, {{direccion.ciudad}}</p>
{{else}}
    <p>No tiene dirección!</p>
{{/if}}

```

En lugar de **if** podemos usar **unless** que funciona al revés (se imprime si la condición es falsa).

El bloque with

Usando **with propiedad**, accedemos a las propiedades anidadas (dentro del bloque with) sin necesidad de usar dicha propiedad como prefijo.

```

<p>{{nombre}}</p>
<p>{{edad}}</p>
{{#with direccion}}
    <p>{{calle}}, {{ciudad}}</p>
{{/with}}

```

Iteradores (each)

Si nos encontramos ante un array y queremos recorrerlo generando HTML, utilizamos el bloque **each**, que iterará sobre el array y generará tantos bloques HTML como elementos tenga dicho array (dentro del bloque accedemos a las propiedades de cada objeto del array directamente).

```

{
    "nombre": "Pepe",
    "edad" : 43,
    "direcciones": [
        {
            "calle": "Calle Ancora 15",
            "ciudad": "Madrid"
        }, {
            "calle": "Calle Tudela 12",

```

```

        "ciudad": "Valladolid"
    }
]
}

```

```

<p>{{nombre}}</p>
<p>{{edad}}</p>
{{#each direcciones}}
    <p>{{calle}}, {{ciudad}}</p>
{{/each}}

```

Resultado:

```

<p>Pepe</p>
<p>43</p>
    <p>Calle Ancora 15, Madrid</p>
    <p>Calle Tudela 12, Valladolid</p>

```

Si el array contiene directamente valores y no objeto, podemos referenciar cada valor con la propiedad **this**, e incluso usar propiedades como **@index** para referirnos al índice actual:

```

{
    "nombre": "Pepe",
    "edad": 43,
    "telefonos": [92485325, 4353246, 69496334]
}

```

```

<p>{{nombre}}</p>
<p>{{edad}}</p>
{{#each telefonos}}
    <p>Teléfono {{@index}}: {{this}}</p>
{{/each}}

```

Resultado:

```

<p>Pepe</p>
<p>43</p>
    <p>Teléfono 0: 92485325</p>
    <p>Teléfono 1: 4353246</p>
    <p>Teléfono 2: 69496334</p>

```

Generar HTML (Webpack)

Integrar Handlebars en un proyecto con Webpack es sencillo usando el Handlebars Loader. En el archivo JavaScript donde necesitemos generar el HTML a partir de una plantilla, cargamos el archivo correspondiente:

```
import productsTemplate from '../templates/products.handlebars';
```

Y generamos el HTML pasándole el objeto (o array de objetos) JSON así:

```
let htmlProds = productsTemplate(prodsJSON);
```

Como generará el HTML en formato string y no como objetos del DOM, debemos usar la propiedad **innerHTML** para asignárselo al elemento contenedor (padre):

```
var container = document.getElementById("container");
container.innerHTML = htmlProds;
```