Formatear Números Con Accounting.Js

Anteriormente habíamos hablado de un plugin de Javascript, el cual nos facilita la transición de números mediante animaciones. Pero si necesitamos algo mas formal, como presentar números de cierta manera para el sector financiero, necesitaremos una solución para esto.

Si usted necesita que un número aparezca en determinado formato de moneda o separado con comas ó puntos decimales, entonces te sera de gran ayuda usar **Accounting.js**, una biblioteca de **JavaScript** para dar el formato deseado.

En este artículo, mostraremos algunas de sus funciones básicas, usaremos un ejemplo real para mostrar cómo funciona. Empecemos!

Primeros pasos

Accounting.js es una biblioteca de **JavaScript** sin dependencias. Usted no necesita tener **jQuery** para usarlo; puede funcionar por sí mismo. Descarga el código fuente desde el repositorio de Github, pongalo en un directorio apropiado, y vincule el archivo en el documento HTML.

[js] <script src="js/accounting.js"> </script> [/js]

Formateado Básico

Accounting.js ofrece algunos métodos para **dar formato a números**. Y el primero que vamos a echar un vistazo es a **formatMoney()**. Este método es la función básica para

convertir los números en moneda. Para usarlo, cada método es inicializado por **accounting** y luego seguido por el nombre del método. Por ejemplo:

[js] accounting.formatMoney(2000000); [/js]

En la configuración predeterminada, **Accounting.js** mostrará el ejemplo anterior con el símbolo del dólar, separacion de cada tres dígitos con una coma, y el uso de un punto decimal para separar dólares de centavos.

[js] $2,000,000.00 [/js]

Algunos países utilizan diferentes separadores por cada tres dígitos (miles) y decimal. **Accounting.js** es totalmente configurable en este sentido. Si la salida por defecto no es la aparece en su moneda local, puede hacer cambios con **Options**.

A continuación, tomamos la geolocalizacion de alemánia como el ejemplo, que utiliza separadores de puntos para miles y la coma para los decimales:

[js]

accounting.formatMoney(2000000, {
symbol : "€",
thousand : ".",
decimal : ",",
});

[/js]

Esto es lo que veremos:

[js]€ 2.000.000,00 [/js]

Si desea dar formato al número sin el símbolo de la moneda, puede utilizar el método **formatNumber().**

Redondeo De Números

Por lo general redondeamos las cantidades hacia arriba o hacia abajo al valor más cercano segun sea la necesidad. Con **Accounting.js**, podemos utilizar **.toFixed()** para hacerlo. Este ejemplo muestra cómo eliminamos los dígitos decimales, así como redondear la cantidad a la décima más cercana:

[js]accounting.toFixed(102.58, 0); [/js]

Veriamos algo de la siguiente manera:

[js]103[/js]

Construyendo Un Convertidor De Divisas

En esta sección, vamos a utilizar las funciones mencionadas anteriormente para construir un **convertidor de divisas**. No vamos a estar construyendo una herramienta extensa, sólo un sencillo ejemplo para mostrar como funciona **Accounting.js**.

En el ejercicio, vamos a convertir **USD** a 2 monedas, **KRW (Won de Corea) y JPY (Yen japonés)**.

Vamos a diseñar la estructura del documento de la siguiente manera:

[html]

<div class="currency-option">
<div class="row">
<h4 class="heading">From</h4>
<select id="input-currency" disabled>
<option value="USD" data-symbol="$" selected>US Dollar</option>
</select>
<span id="input-

```
symbol&amp;quot;&amp;gt;$&amp;lt;/span&amp;gt;  &amp;lt;input  id=&amp;quot;input-number&amp;quot; class=&amp;quot;input&amp;quot; type=&amp;quot;number&amp;quot; min=&amp;quot;0&amp;quot;&amp;gt;
&amp;lt;/div&amp;gt;

&amp;lt;div  class=&amp;quot;row&amp;quot;&amp;gt;
&amp;lt;h4 class=&amp;quot;heading&amp;quot;&amp;gt;To&amp;lt;/h4&amp;gt;
&amp;lt;select id=&amp;quot;output-currency&amp;quot;&amp;gt;
&amp;lt;option value=&amp;quot;krw&amp;quot; data-symbol=&amp;quot;₩&amp;quot; selected&amp;gt;Korean Won&amp;lt;/option&amp;gt;
&amp;lt;option value=&amp;quot;jpy&amp;quot; data-symbol=&amp;quot;¥&amp;quot;&amp;gt;Japanese Yen&amp;lt;/option&amp;gt;
&amp;lt;/select&amp;gt;
&amp;lt;span id=&amp;quot;output-number&amp;quot;&amp;gt;₩ 0&amp;lt;/span&amp;gt;
&amp;lt;/div&amp;gt;
&amp;lt;/div&amp;gt;

[/html]
```

Como se puede ver arriba, tenemos dos **div «principales»**. La primera capa contiene una opción de menú desplegable que se establece en **USD**, y desactivamos la selección para que el usuario no pueda elegir mas nada.También contiene un campo de entrada del tipo número, donde ingresamos la cantidad de **USD** para convertir.

En la segunda capa, tenemos también una opción desplegable, la cual contiene dos opciones de Moneda: **Won de Corea y el yen japonés**. Cada opción tiene un atributo **value**, y un atributo de **data-symbol** para almacenar el símbolo de moneda. Usamos un elemento **span** para mostrar el resultado convertido.

Tipo De Cambio

En el momento de convertir **1 USD** es igual a KRW 1077.80 y JPY 102.24. Por ahora, simplemente ponemos el valor en una variable con el método **.toFixed()** para redondear el número.:

[js]

```
var jpy = accounting.toFixed(102.24, 0),
krw = accounting.toFixed(1077.80, 0),
```

[/js]

Obtener La Opción

A continuación, vamos a crear una nueva función para obtener el valor dependiendo del atributo **value** y el atributo **data-symbol** de la opción desplegable. Los valores luego se almacenan en una matriz.

[js]

```
var getCurrency = function(elem) {
var $curAbbr = elem.find(':selected').val(),
$curSign = elem.find(':selected').data('symbol');
return {
'symbol' : $curSign,
'value' : $curAbbr,
};
};
```

[/js]

La Función De Conversión

Queremos que la conversión se produzca en tiempo real. Esto significa que va a suceder cuando el usuario está escribiendo en el campo de entrada o el cambio entre las monedas .

Para lograr esta idea , le asignaremos **#output-currency** , así como **#input-number** con tres eventos de **JavaScript: change, keyup, and keydown:**

[js]

```
$('#output-currency,  #input-number').on('change  keyup keydown', function() {
// Demás código
}
```

[/js]

Entonces, vamos a obtener el valor de la opción desplegable, con **#output-currency**, mediante la función **getCurrency** que hemos creado anteriormente. Los valores se separan dentro de dos variables diferentes, **$symbol y $val**, de la siguiente forma.

[js]

```
var $currency = getCurrency($('#output-currency')),
$symbol = $currency['symbol'],
$val = $currency['value'];
```

[/js]

También tenemos que obtener el número del campo de entrada , y el valor de cambio, el cual hemos fijado en la variable **JPY y KRW**; utilizando una condicional podemos decidir qué tipo de conversión (**krw o jpy**) vamos a utilizar.

[js]

```
// get number
var mulitplyNum = ($val == 'jpy') ? jpy : krw;
var $getInput = $('#input-number').val();
```

[/js]

Con los números anteriores, podemos calcular el resultado:

[js] var $getTotal   = ($getInput * mulitplyNum); [/js]

Pero, antes de mostrar el resultado, vamos a envolver en un formato adecuado utilizando el método **.formatMoney()**:

[js]

var number = accounting.formatMoney($getTotal, {
symbol : $symbol,
precision : 0,
thousand : ‘,’
});

[/js]

Y por último, mostramos el numero con el formato dado en el **span**:

[js]$(‘#output-number’).text(number); [/js]

Y ya hemos terminado. Usted puede ver a continuación la **demo** en acción:

No podemos negar la utilidad que nos da este tipo de plugin, solo debemos conocer su funcionamiento para dar un bueno uso de estos en nuestro proyecto. Como vemos, la implementación de es **Accounting.js** para dar formato a los números es bastante sencillo.