Diseño Web — Tecnologías Involucradas

JQuery

info@covetel.com.ve [1]

[1] Cooperativa Venezolana de Tecnologías Libres R.S.

Métodos de manipulación DOM

- Atributos Generales
- Propiedades de Estilo
- Atributos de Clase
- Reemplazo DOM
- Inserciones DOM, Dentro
- Inserciones DOM, Fuera
- DOM Copia, DOM Eliminación

Métodos de Evento

- Controlador adjunto de eventos
- Cargar de Documentos
- Eventos del Mouse
- Eventos del Formulario
- Eventos del Teclado
- Eventos del Navegador

.attr(attribute)

- .attr(attribute) *attribute= el nombre del atributo a obtener.*
- .attr(attribute, value) *value= un valor establecido para el atributo.*
- .attr(map) *map= un mapa de pares atributo-valor para establecer.*
- .attr(attribute, function) *function= una función que devuelve el valor a establecer.*

```
1  <img id="greatphoto" src="brush-seller.jpg" alt="brush seller" />
```

```
1  $('#greatphoto').attr('alt', 'Beijing Brush Seller');
```

Cambia el atributo alt señalando el atributo alt y estableciendo el nuevo valor después de la coma dentro del paréntesis.

.attr()

Mediante el uso de una función para establecer los atributos, podemos concatenar un nuevo valor con un valor existente.

```
1 $('#greatphoto').attr({alt: function() {return 'Beijing' + this.alt}, title:
2 function() {return 'Beijing' + this.alt + ' - photo by Kelly Clarck'}});
```

El uso de una función puede ser aún mas útil cuando se aplican los atributos de elementos multiples.

.removeAttr()

- .removeAttr(attribute) *attribute= un atributo.*

El método .removeAttr utiliza la función JavaScript removeAttribute, pero tiene la ventaja de poder ser encadenado a una expresión de selección JQuery.

.css()

- .css(property, value) *property= un nombre de propiedad CSS. value= un valor establecido para la propiedad.*
- .css(map) *map= un mapa de pares propiedad-valor para establecer.*
- .css(property, function) *function= una función que devuelve el valor a establecer.*

```
1  $('div.example').css('width', function(index) {
2    return index * 50;
3  });
```

Este ejemplo establece la anchura de los elementos que coinciden con lo valores progresivamente más grandes.

Dimensiones

- .height(value) *value= un entero representando el número de pixeles o un entero junto con una unidad opcional de medida.*
- .width(value) *value= un entero representado el número de o un entero junto con una unidad opcional de medida.*

Atributos de Clase

- .addClass(class) *class= uno o mas nombres de clase que se agrega al atributo de clase o a cada elemento emparejado.*
- .removeClass(class) *class= uno o mas nombres de clase que se eliminan del atributo de clase o de cada elemento emparejado.*

```
1 $('p').removeClass('myclass noclass').addClass('yourclass')
```

Aquí las clases `myclass` y `noclass` se eliminan de los párrafos, mientras `yourclass` es añadida.

- .toggleClass(class) *function= un nombre de clase para ser activado en el atributo de clase de cada elemento en el conjunto combinado.*

```
1 <div class="tumble">Some text</div>
```

```
1 $('div.tumble').toggleClass('bounce')
```

```
1 <div class="tumble bounce">Some text</div>
```

Reemplazo DOM

- .html(HTML) *HTML= una cadena HTML para establecer el contenido de cada elemento encontrado.*

```
1    <div class="demo-container">
2      <div class"demo-box">Demostration Box</div>
3    </div>
```

```
1    $('div.demo-container').html('<p>All new content. <em>You bet!</em></p>
         '));
```

```
1    <div class="demo-container">
2      <p>All new content. <em>You Bet!</em></p>
3      <div class"demo-box">Demostration Box</div>
4    </div>
```

Métodos

- .text(text) *text= una cadena de texto para establecer el contenido de cada elemento encontrado.*

```
<div class="demo-container">
  <div class"demo-box">Demostration Box</div>
  <ul>
    <li>list item1</li>
    <li>list <strong>item</strong>2</li>
  </ul>
</div>
```

```
$('div.demo-container').text('<p>This is a Test.</p>')
```

```
<div class="demo-container">
 <p>This is a test</p>
</div>
```

- .val(value) *value= una cadena de texto para establecer el valor de la propiedad de cada elemento encontrado.*

Inserciones DOM, Dentro

- .prepend(content) *content= un elemento, cadena HTML u objeto jQuery para insertar al principio del elemento establecido encontrado.*

```
<div class="demo-container">
  <div class"demo-box">Demostration Box</div>
</div>
```

```
$('div.demo-box').prepend('<div class="insertion">This text
was<strong>inserted</strong></div>');
```

- .prependTo(target) *target= un selector, elemento, cadena HTML, cadena u objeto JQuery; el conjunto combinado de elementos se insertan al principio del elemento especificado por este parámetro.*

```
<div class="demo-container">
  <div class"demo-box">Demostration Box</div>
</div>
```

```
$('<div class="insertion">This text was<strong>inserted</strong></div>'
    ).prependTo('div.demo-box');
```

Inserciones DOM, Dentro

- .append(content) *content= un elemento, cadena HTML u objeto jQuery para insertar al final del elemento establecido encontrado.*

```
1    <div class="demo-container">
2      <div class"demo-box">Demostration Box</div>
3    </div>
```

```
1    $('div.demo-box').append('<div class="insertion">This text
2    was<strong>inserted</strong></div>');
```

- .appendTo(target) *target= un selector, elemento, cadena HTML, cadena u objeto JQuery; el conjunto combinado de elementos se insertan al final del elemento especificado por este parámetro.*

```
1    <div class="demo-container">
2      <div class"demo-box">Demostration Box</div>
3    </div>
```

```
1    $('<div class="insertion">This text was<strong>inserted</strong></div>'
         ).appendTo('div.demo-box');
```

Inserciones DOM, Fuera

- .insertBefore(content) *content= un selector o elemento que se insertara antes del conjunto encontrado.*

```
1    <div class="demo-container">
2      <div class"demo-box">Demostration Box</div>
3    </div>
```

```
1    $('<div class="insertion">This text was<strong>inserted</strong></div>'
         ).insertBefore('div.demo-box');
```

- .insertAfter(content) *content= un selector o elemento que se insertara después del conjunto encontrado.*

```
1    <div class="demo-container">
2      <div class"demo-box">Demostration Box</div>
3    </div>
```

```
1    $('<div class="insertion">This text was<strong>inserted</strong></div>'
         ).insertAfter('div.demo-box');
```

DOM Copia, DOM Eliminación

- .clone([deep]) *deep= Un booleano. Predeterminado verdadero. Si se establece falso el método copia sólo los elementos coincidentes con si mismo con exclusión de cualquier hijo o descendiente.*

```
<div class="demo-container">
  <div class"demo-box">Demostration Box</div>
</div>
```

```
$('div.demo-box:last').clone().insertAfter('div.demo-box:last');
```

- .remove([selector]) *selector= Un selector que filtra los elementos encontrado a ser removidos.*

```
<div class="demo-container">
  <div class"demo-box">Demostration Box</div>
</div>
```

```
$('div.demo-box').remove('#temporary-demo-box')
```

Controlador adjunto de eventos

- .bind(evenType[, eventData, handler]) *evenType= Una cadena que contiene un tipo de evento JavaScript, sea click o submit.*
  eventData= Un mapa de datos que pueden ser pasados por un manejador de eventos.
  handler= Una función que se ejecutará cada vez que se dispare el evento.

```
1    $('#foo').bind('click', function() {
2      alert('User clicked on foo');
3    });
```

- .trigger(evenType [, extraParameters]) *extraParameters= Un array de parámetros adicionales pasados a través de un manejador de eventos.*

```
1    $('#foo').bind('click' function() {
2      alert($(this).text());
3    });
4    $('#foo').trigger('click');
```

Carga de Documentos

- .load(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1       <img class="target" src="hat.gif" width="80" height="54" />
```

```
1   $('.target').load(function() {
2     $(this).log('Load event was triggered.');
3   });
```

- .error(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1       <img class="target" src="missing.gif" width="80" height="54" />
```

```
1   $('.target').error(function() {
2     $(this).log('Error event was triggered.');
3   });
```

Eventos del Mouse

- .click(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1     $('.target').click(function() {
2       $(this).log('Click event was triggered.');
3     });
```

- .mouseover(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1     $('.target').mouseover(function() {
2       $(this).log('Mouseover event was triggered.');
3     });
```

- .hover(handlerIn, handlerOut) *handlerIn= Una función que se ejecutará cuando el puntero del mouse entra en el elemento.*
  handlerOut= Una función que se ejecutará cuando el puntero del mouse sale del elemento.

```
1     $('.target').hover(function() {
2       $(this).log('Hover event was triggered entering');
3     }function() {
4       $(this).log('Hover event was triggered leaving');
5     });
```

Eventos del Mouse

- .mousemove().
- .dblclick().
- .mousemove().
- .toggle().
- .mouseup(.)
- .mousedown().

Eventos del Formulario

- .focus(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
$('.target').focus(function() {
  $(this).log('Focus event was triggered.');
});
```

- .blur().
- .change().
- .select().
- .submit(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
<form class="target" action="foo.html">
  <input type="text" /><input type="submit" value="Go" />
</form>
<div class="trigger button">Button</div>
```

```
$('.trigger').click(function() {
  $('.target').submit();
});
```

Eventos del Teclado

- .keypress(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1     $('.target').keypress(function() {
2       $(this).log('Keypress event was triggered.');
3     });
```

- .keydown().
- .keyup().

Eventos del Navegador

- .resize(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1    $(window).resize(function() {
2      $(this).log('Resize event was triggered.');
3    });
```

- .scroll(handler) *handler= Una función que se ejecutará cada vez que se dispare el evento.*

```
1    $('.trigger').click(function() {
2      $('.target').scroll();
3    });
```