Librerías JavaScript: JQuery

Desarrollo de Aplicaciones en Entornos Web

Curso 2016/2017

Contenido

- Que es JQuery
- Carácterísiticas JQuery
- Funciones básicas y selectores
- Funciones para eventos
- Funciones para efectos visuales
- Funciones para AJAX
- Funciones para CSS
- Funciones para nodos DOM

Que es JQuery

https://**jquery.com/**

- Librería de JavaScript
  - autor: John Resig (14/01/2006 en el BarCamp NYC)
  - múltiples contribuciones de desarrolladores
  - bajo licencia GPL

- Facilita la programación de JavaScript:
  - Código más reducido y compatible con casi todos los navegadores.

- Programada eficientemente: comprimida solo ocupa unas decenas de KBs.

Características JQuery

- Selección de elementos **DOM**.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para **CSS** 1-3 y un plugin básico de **XPath**.
- Eventos.
- Manipulación de la hoja de estilos **CSS**.
- Efectos y **animaciones**. Animaciones personalizadas.
- **AJAX**.
- Utilidades varias: obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- **Compatible** con la mayoría de los navegadores

Características JQuery

- Sencillo de escribir. El código es comprensible y fácil de leer.

- Fácil de escalar y mantener
  - jQuery no es un *framework, es solamente una librería para facilitarnos el manejo del DOM, los eventos y las peticiones AJAX*
  - *Framework MVC en JavaScript: Angular.js, Backbone.js, React.js*

- Otros proyectos JQuery: jQuery User Interface, QUnit JS Unit Testing, **jQuery Mobile**

Ejemplo JavaScript vs JQuery

- **JavaScript**

```
function cambiaFondo(color) {
      document.body.style.background = color;
   }
onload="cambiaFondo('#ccc');”
```

- **JQuery**

```
$('body').css('background', '#ccc');
```

Conceptos Básicos de JQuery

- `$(document).ready()` : indicar si el documento se encuentra preparado para su manipulación.

```
$(document).ready(function() {
    console.log('el documento está preparado'); });

function readyFn() {
    console.log('el documento está preparado'); }
$(document).ready(readyFn);

$(function() {
    console.log('el documento está preparado'); });
```

- JQuery ofrece selectores basados en atributos donde el valor de los atributos emplea expresiones regulares.

```
// encontrar todos los <a> cuyo atributo rel
   terminan en "thinger”
$("a[rel$='thinger']");
```

- Usando `.match()`

```
$("#texto").attr("value").match("expresión
   regular");
```

Funciones y métodos básicos

- **$(*cadena*)**
  - document.getElementByXXX(*cadena*)
  - la cadena de texto: usa **XPath** o *selector CSS*
  - usando el separador "," : seleccionar muchos elementos.

```
$('a')          // Todos los enlaces de la página
$('#primero')// Elemento cuyo id sea "primero"
$('h1.titular') // Elementos h1 con class
   "titular"
$('a, #primero, h1.titular') // Selecciona
   todo lo anterior
```

- Métodos que se llaman desde el objeto JQuery:
  - `$(...)`
  - métodos utilizados en selecciones
- Métodos en el espacio de nombres `$` son generalmente métodos para diferentes utilidades
  - no trabajan con selecciones.
- Ejemplo: `$.each` y `$('a').each`

```
$.trim('    quitar espacios en blanco   ');
$.each([ 'foo', 'bar', 'baz' ], function(idx, val) {
            console.log('elemento ' + idx + 'es ' + val);} );
var myArray = [ 1, 2, 3, 5 ];
if ($.inArray(4, myArray) !== -1) {
            console.log('valor encontrado'); }

var myValue = [1, 2, 3];
// Utilizar el operador typeof de JavaScript para comprobar tipos primitivos
typeof myValue == 'string'; // falso (false)
// Utilizar los métodos jQuery para comprobar tipos no primitivos
jQuery.isFunction(myValue); // falso (false)
jQuery.isArray(myValue); // verdadero (true)
```

Selectores CSS3 y otros

api.jquery.com/category/selectors

- Permite seleccionar algunos elementos y realizar acciones con ellos.
- La elección de buenos selectores es importante para mejorar el rendimiento del código.

- elementos en base a su id : `$('#myId');`
- elementos en base al nombre de clase: `$('div.myClass');`
- elementos por su atributo: `$('input[name=first_name]');`
- elementos en forma de selector CSS:

```
$('#contents ul.people li');
```

Otros selectores:

```
// selecciona el primer elemento <a> con la clase 'external'
$('a.external:first');
// selecciona todos los elementos <tr> impares de una tabla
$('tr:odd');
// selecciona todos los elementos del tipo input dentro del
    formulario #myForm
$('#myForm:input');
// selecciona todos los divs visibles
$('div:visible');
// selecciona los divs excepto los 3 primeros (greater tan)
$('div:gt(2)');
// selecciona todos los divs actualmente animados
$('div:animated');
```

Mas ejemplos:

```
// Selecciona todos los párrafos de la página con al menos un enlace
$('p[a]')
// Selecciona todos los radiobutton de los formularios de la página
$('input:radio')
// Selecciona todos los enlaces que contengan la palabra "Imprimir"
$('a:contains("Imprimir")');
// Selecciona todos los elementos pares de una lista
$("ul#menuPrincipal li:even")
// Selecciona todos los elementos impares de una lista
$("ul#menuPrincipal li:odd")
// Selecciona los 5 primeros párrafos de la página
$("p:lt(5)")
// Selecciona los 4 párrafo de la página
$("p:eq(4)")
```

- Evaluar si una selección posee elementos

```
if ($('div.foo').length) { ... }
```

- Guardar selecciones en una variable

```
var $divs = $('div');
```

- JQuery ofrece pseudo-selectores que ayudan a encontrar elementos dentro de los formularios
  - `:button, :checkbox, :checked, :file, :disabled, :enabled, :image, :input, :password, :radio, :reset, :selected, :submit, :text`

```
// obtener todos los inputs dentro del formulario #myForm
$('#myForm :input');
```

Acceso a elementos HTML

- $.fn.**attr** : Valor de un determinado atributo.
- $.fn.**html** : Contenido HTML de un elemento.
- $.fn.**text** : Contenido en texto del elemento.
- $.fn.**val** : Valor (*value) en elementos de formularios.*

```
//Modificación atributos
$('a').attr('href', 'www.um.es');
$('a').attr({
    'title' : 'Universidad de Murcia',
    'href' : 'www.um.es'});
//Consulta atributos
$('a').attr('href');
```

Crear elementos HTML

- A través del mismo método **`$()`** que se utiliza para realizar selecciones.

```
var $myNewElement = $('<p>Nuevo elemento</p>');
$myNewElement.appendTo('#content');
$('<a/>', {
    html : 'Un <strong>nuevo</strong> enlace',
    'class' : 'new',
    href : 'index.html'} );
```

- Los nombres de propiedades no deben estar entre comillas
  - Excepto que se utilice como propiedad una palabra reservada

Funciones para eventos

- Al cargar la página, el navegador construye el árbol DOM

```
window.onload = function() {  ...});
```

  - Problema: página cargada completamente (incluyendo todas las imágenes y archivos externos)

- JQuery propone:

```
$(document).ready(function() {  ...});
```

  - no espera a que se carguen todos los elementos de la página
  - árbol DOM disponible para ser manipulado al descargar el contenido HTML

Funciones para eventos

- Si se utiliza la función sin argumentos, se ejecuta el evento

```
$(document).ready(function() … );
$('p').click();
// Ejecuta el evento 'onclick' en todos los párrafos de
   la página
$('div#menu').mouseover();
// Ejecuta el evento 'mouseover' sobre un 'div' con id
   'menu'
```

- **keypress()**: evento de presión de tecla.
- **toggle()**: alterna la ejecución de dos funciones.

Funciones para eventos

- Uso habitual de las funciones de eventos: **establecer** la función **manejador** que ejecuta cuando se produzca el evento

```
$('p').click(
   function() {
      alert($(this).text());
   });
```

Vincular funciones a eventos con $().bind

```
$('p').bind('click', function() {
    console.log('click');});


$('input').bind(
    'click change',
    // es posible vincular múltiples eventos al elemento
    { foo : 'bar' }, // se debe pasar la información asociada como argumento
    function(eventObject) {
        console.log(eventObject.type, eventObject.data);
      // registra el tipo de evento y la información asociada { foo : 'bar' }
} );
$('p').unbind('click');
```

Funciones para efectos visuales

```
// Oculta todos los enlaces de la página
$('a').hide();
// Muestra todos los 'div' que estaban ocultos
$('div:hidden').show();
// Muestra los 'div' que estaba ocultos y oculta los 'div' que eran
    visibles
$('div').toggle();
// Muestra, espera y oculta los 'h1' en tiempos determinados
$('h1').show(300).delay(1000).hide(300);
```

- **fadeIn(), fadeOut()** : muestra/oculta los elementos con un fundido
- **slideDown(), slideUp(), slideToggle()** : muestra/oculta/alterna los elementos con sentido descendente/ascendente

Funciones para AJAX

```
$.ajax(opciones);


$.ajax({
   url: '/ruta/hasta/pagina.php',
   type: 'POST',
   async: true,
   data: 'parametro1=valor1&parametro2=valor2',
   success: procesaRespuesta,
   error: muestraError}
);
```

Funciones para AJAX

async	petición asíncrona	ifModified	petición correcta si respuesta modificada
beforeSend	unción modificar XMLHttpRequest antes de petición	processData	indica si transforma los datos *data* en una cadena
complete	función tras petición completada	success	función cuando una petición correcta
contentType	cabecera Content-Type petición	timeout	tiempo espera en milisegundos
data	datos/parámetros enviados al servidor	type	GET, POST
dataType	tipo de dato respuesta	url	url de la petición
error	función tras error		

Funciones para AJAX

- **`dataType`**:
  - *xml* : responseXML
  - *html*: responseText
  - *script*: evalúa JavaScript y devuelve el resultado
  - *json*: objeto JavaScript generado

- funciones simplificadas y especializadas de **`$.ajax()` : `$.get()`, `$.post()`**

```
$.get('/ruta/hasta/pagina.php');
$.get(url, datos, funcionManejadora); //tambien $.post()
```

- `$.load()` : inserta el contenido de la respuesta del servidor en el elemento de la página que se indica

```
$('#info').load('/ruta/hasta/pagina.php');
```

- `$.getJSON()` : carga respuesta de tipo JSON

- `$.getScript()` : evalúa/ejecuta una respuesta JavaScript

Ajax y formularios

- Transformar información de un formulario a una cadena de datos

```
$('#myForm').serialize();
```

- Crear un array de objetos conteniendo información de un formulario

```
$('#myForm').serializeArray();
// crea una estructura como esta:
[
   { name : 'field1', value : 123 },
   { name : 'field2', value : 'hello world' }
]
```

Funciones para CSS

- `$().css()`

```
// Establece varias propiedades CSS en este caso,
   para todos los 'div' de la página
$('div').css({ padding: '3px', color: '#CC0000' });
```

- Otras funciones específicas (leer/modificar):

```
$('div').height();
$('div').height('150px');
$('div').width();
$('div').width('300px');
```

Funciones para CSS

- Trabajar con clases CSS es mas apropiado que trabajar con los atributos visuales directamente:

- Añadir/quitar/consultsar una clase CSS

```
$h1.addClass('big');
$h1.removeClass('big');
if ($h1.hasClass('big')) { ... }
```

Funciones para nodos DOM

- funciones de filtrado y funciones de búsqueda
- filtros son funciones que modifican y limitan la selección con la función `$()`

```
// Sólo obtiene los párrafos que contengan la palabra
    'importante'
$('p').contains('importante');


// Selecciona todos los enlaces de la página, salvo el que
    tiene una 'class' igual a 'especial'
$('a').not('.especial');


// Selecciona todas las listas de elementos de la página y
    quedate sólo con las que tengan una 'class' igual a 'menu'
$('ul').filter('.menu');
```

- **end()**: permite volver a la selección original de elementos después de filtrar.

```
$('a')
    .filter('.pinchame')
    .click(function(){
        alert('Estás abandonando este siti web');
    })
    .end()
    .filter('ocultame')
        .click(function(){
            $(this).hide();
                return false;
        })
    .end();
```

- **end()** permite encadenar varias selecciones.

Navegación entre nodos DOM

- **`children()`** : obtener todos los *nodos hijo o descendientes del nodo actual*
- **`parent()`** *: obtener el nodo padre o nodo ascendente del nodo actual*
- **`parents()`** *: obtener todos los ascendentes del nodo hasta la raíz*
- **`siblings()`***: obtener todos los nodos hermano del nodo actual*
- **`next()`** *y* **`prev()`** *: avanzar o retroceder a través de la lista de nodos hermanos (siblings)*

Manipulación de nodos DOM

- **`append`**`()` y **`prepend`**`()` : añadir el contenido indicado como parámetro al principio o al final del contenido original del nodo
- **`after`**`()` y **`before`**`()` : añadir el contenido indicado como parámetro antes de cada uno de los elementos seleccionados
- **`wrap`**`()` : *"envolver" un elemento con el contenido indicado (se añade parte del contenido por delante y el resto por detrás)*
- **`empty`**`()` *: vaciar de contenido a un elemento*
- **`remove`**`()` *: eliminar los elementos seleccionados*
- **`clone`**`()` *: copiar los nodos seleccionados.*

Otras funciones

- **`$.browser`** : JQuery detecta automáticamente el tipo de navegador.

```
$.browser.msie; // 'true' familia Internet Explorer
$.browser.mozilla; // 'true' familia Firefox
$.browser.safari; // 'true' familia Safari
```

- **`$.each()`** : recorrer arrays y objetos

```
var vocales = ['a', 'e', 'i', 'o', 'u'];
$.each( vocales, function(i, n){
    alert('Vocal número ' + i + " = " + n);}
);
var producto = { id: '12DW2', precio: 12.34, cantidad: 5 };
$.each( producto, function(i, n){
    alert(i + ' : ' + n);}
);
```