AJAX

El framework jQuery

Índice

- Frameworks en JavaScript.
- Introducción a jQuery y como instalarlo.
- El selector $ y los selectores CSS.
- Efectos graficos en jQuery.
- La caracteristica chainability.
- Otros metodos y atajos en jQuery.
- AJAX en jQuery.

- ¿Que es un framework?
  - En el desarrollo del software, un framework es una estructura de soporte definida en la cual otro proyecto puede ser organizado y desarrollado. Tipicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

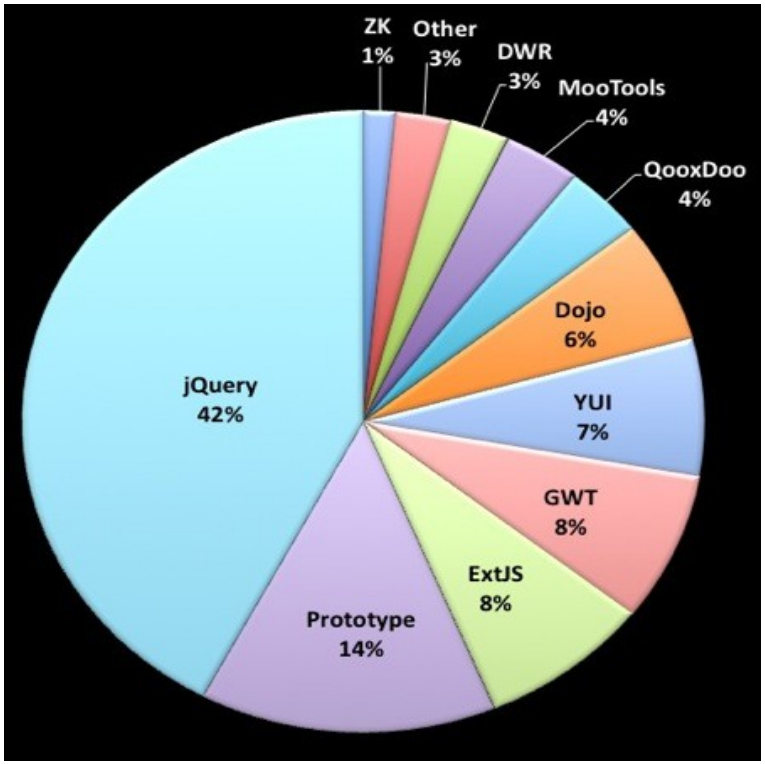
- ¿Porque usar un framework para JavaScript y Ajax en nuestras aplicaciones web?
  - Facilita y agiliza el proceso de desarrollo de aplicaciones Web.
  - Permite reutilizar código ya existente y promueve buenas prácticas de desarrollo. Evita reinventar la rueda.
  - Debido a la existencia de múltiples navegadores que se comportan de diferente forma, es necesario escribir códigos diferentes para cada uno. Un framework nos permite “abstraernos” del navegador para solo centrarnos en la funcionalidad del código.

- Inconvenientes de un framework
  - Existe una curva de aprendizaje para cada framework. Hacernos con el control de uno requiere tiempo.
  - Saber utilizar un framework no implica saber como funciona ni conocer las tecnologías que soporta.
  - La relevancia de un framework esta sujeto a modas, es fácil que pase de ser el más el usado a un reducto del pasado en solo un año.

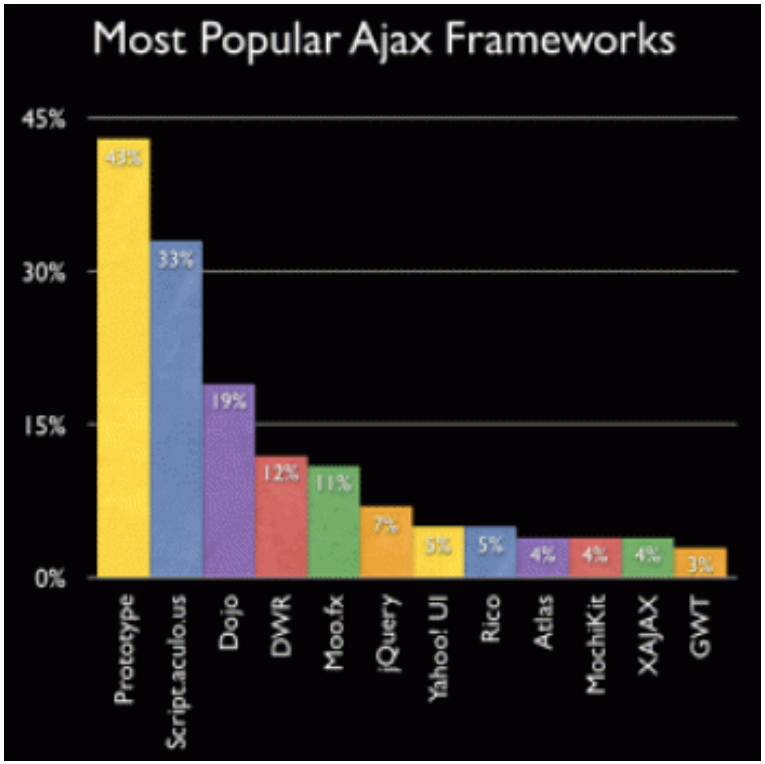
- Consecuencias de utilizar un framework en nuestras aplicaciones web:
  - Normalmente el código es mas claro y reutilizable por otros programadores.
  - Un framework suele estar altamente testeado, con lo que nuestras páginas suelen ser más fiables.
  - El hecho de que los frameworks esten pensados para funcionar en multitud de entornos distintos en algunos casos peude ser innecesario y conlleva una lentitud y peso añadido a nuestro código.
  - No siempre vamos a utilizar todas las funciones de un framework, pero normalmente las cargaremos todas.

- Uso de frameworks de JavaScript:

2009



2007



- Introducción a JQuery
  - Query es un una biblioteca o framework de Javascript que permite simplificar la manera de interactuar con los documentos HTML, permitiendo manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web.
  - jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en Javascript que de otra manera requerirían de mucho más código.

- Ventajas de JQuery
  - Tiene el apoyo de Google y Microsoft, de hecho esta integrado en el entorno de desarrollo Visual Studio de Microsoft desde el 2008.
  - Es el framework basado en JavaScript con una mayor tasa de crecimiento.
  - Incorpora tanto herramientas para la simplificación de código como efectos gráficos y de control (autocomplete, drag&drop etc...).

- Como descargar e instalar JQuery
  - QJuery lo podemos encontrar en su propia pagina web en dos versiones, una de desarrollo (con código legible) y otra comprimida para producción. Nosotros nos barajemos la versión de desarrollo:
    - http://docs.jquery.com/Downloading_jQuery
  - Para integrar JQuery en nuestra aplicación web basta con escribir esta linea:
    - <script type="text/javascript" src="jquery.js"></script>

- El selector $ en Jquery:
  - Podemos acceder a los elementos del DOM mediante atajos. El atajo mas útil es la función $().
  - A la función $() se le pasa como parámetro un selector CSS. Por ejemplo para seleccionar un div con id=elemento1 escribiriamos:
    - *$('#elemento1')*
  - Cuando utilizamos un selector que ataca a múltiples elementos (pj: 'div') las acciones sobre el repercutirán en cada uno de los elementos

- Repaso de selectores CSS
  - .nombreClase = hara referencia a todos lo elementos con la clase “nombreClase”.
  - E F (pj: div a) = referenciara a todos los elemento a dentro de elementos div.
  - E > F (pj: div > a) = Muy parecido al caso anterior, con la salvedad que solo referenciara los hijos de primer grado (no hara referencia a nietos, bisnietos, etc...).
  - E[atributo] (pj: img[width=100]) = En este ejemplo referencia los elementos img con el atributo width=100.

- Modificar CSS con Jquery:
  - Función addClass(“nombreClase”) = Añade la clase “nombreClase” al elemento elegido

    pj: $(“#elemento1”).addClass(“clase1”).
  - Función removeClass(“nombreClase”) = Elimina la clase del Elemento.
  - Función css(“atributo1”,”valor1”) = Establece el valor “valor1” al atributo “atributo1”.

    pj: $(“#elemento1”).css(“color”,”#FF0000”).

- Efectos gráficos con jQuery:
  - La sintaxis común a todos los ejectos gráficos en Jquery es:
    - nombreEfecto(velocidad,callback)
      - Velocidad: puede tomar los valores [“slow”,”normal”,”fast”] o un entero que representa milisengundos
      - Callback: sera la función que se lanzara al terminar el efecto.
  - Las efectos mas comunes en jQuery son:
    - show(velocidad,callback) = muestra el elemento.
    - hide(velocidad, callback) = oculta el elemento.
    - toggle(velocidad,callback) = cambia la visibilidad.

- Otro efectos gráficos:
  - slideDown(velocidad,callback) = efecto persiana mostrando el contenido.
  - slideUp(velocidad,callback) = efecto persiana ocultando el contenido.
  - fadeIn(velocidad,callback) = efecto que va cambiando la opacidad hasta mostrar el elemento.
  - fadeOut(velocidad,callback) = mismo efecto que el anterior, pero termina ocultando el elemento.

- Animaciones en jQuery:
  - Jquery permite la personalización de efectos mediante la función animate. Esta función nos permite pasar de unas propiedades css iniciales en un elemento, a otra finales mediante una animación.
  - La sintaxis es:
    - Elemento.animate({propiedades finales css},velocidad)

- Ejemplo animate:
  - En este ejemplo la palabra HOLA cambiara de tamaño agrandandose, al mismo tiempo se cambiara su opacidad al 50%

```
<button id=”iniciar”></button>
<div>HOLA</div>

<script>

$("#iniciar").click(function(){
                  $("#block").animate({ opacity: 0.5,
                  fontSize: "48px",}, 1500 );
                  });

<script>
```

- Chainability (concatenacion de acciones)
  - Es una de las herramientas mas poderosas de Jquery. Permite al concatenación sucesiva de metodos.
  - Su sintaxis es Elemento.metodo1.metodo2.metodo3 (etc...)
  - En este ejemplo el elemento1 desaparecera y volvera a aparecer:
    - *$("#elemento1").fadeOut().fadeIn();*

- Otros metodos Jquery:
  - Elemento.ready(callback) = Esta función sustituye al metodo onLoad en la etiqueta body. Se lanzara la función callback cuando el Elemento este disponible (se puede usar en cualquier elemento).
  - Elemento.filter(“selector CSS”) = todos los elementos que se incluyan dentro de “Elemento” se filtran por el “selector CSS”.
    - Ejemplo: $('div').filter(“.nombreClase”).hide() = ocultara todos los elementos contenidos en divs que tengan la clase “nombreClase”.

- Otros metodos Jquery II:
  - Elemento.next() = devuelve el nodo siguiente.
  - Elemento.prev() = devuelve el nodo anterior.
  - Elemento.partent() = devuelve el nodo padre.
  - Elemento.childer() = devuelve los hijos de Elemento.
  - Elemento.is("selector CSS") = devuelve false si el elemento no cumple la condición especificada en "selector CSS".

- AJAX en jQuery
  - jQuery dispone de múltiples funciones para las diferentes conexiones AJAX. Entre ellas podemos destacar:
    - load(url,data,callback) //carga una pagina
    - jQuery.get(url,data,callback,type) //peticion get
    - jQuery.post(url,data,callback,type) //peticion post

- Funcion load(url,data,callback)
  - La función load carga una pagina externa a traves de una petición Ajax.
    - Url = sera la dirección a la que haremos las petición AJAX
    - Data = seran los parametros que se pasaran por GET
    - Callback = sera la función que se ejecutara tras la petición AJAX
  - La función load se ejecuta sobre un elemento, sustituyendo su valor por la respuesta.

- Ejemplo Load

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script>
// cuando este cargado el documento haremos la peticion
$(document).ready(inicio);
function inicio(){
$
("#divRespuesta").load("ajaxRequest.php","opcion=load&parametro1=valor1",v
uelta);
}
function vuelta()
{
alert("se realizo la conexion");
}
</script>
</head>
<body>
<div id="divRespuesta">AQUI IRAN LOS DATOS DE VUELTA</div>
</body>
</html>
```

- Funcion get(url,data,callback)
  - La funcion get, realiza una peticion AJAX de tipo GET.
  - Su sintaxis es $.get(url,data,callback) donde url es la página objetivo, data son los parámetros que enviaremos por get y el callback es la función que se lanzara cuando se complete la petición.
  - A la funcion callback se le pasara por parámetro el objeto response.

- Ejemplo GET

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script>
// cuando este cargado el documento haremos la peticion
$(document).ready(inicio);
function inicio(){
$.get("ajaxRequest.php","opcion=get",vuelta);
}
function vuelta(response)
{
$("#divRespuesta").append(response);
}
</script>
</head>
<body>
<div id="divRespuesta">El libro de vuelta es: </div>
</body>
</html>
```

- Funcion post(url,data,callback)
  - La funcion post, realiza una peticion AJAX de tipo POST.
  - Su sintaxis es $.post(url,data,callback) donde url es la página objetivo, data son los parámetros que enviaremos por post y el callback es la función que se lanzara cuando se complete la petición.
  - A la función callback se le pasara por parámetro el objeto response.

- Ejemplo Post:

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script>
// cuando este cargado el documento haremos la peticion
$(document).ready(inicio);
function inicio(){
$.post("ajaxRequest.php?opcion=post","usuario=33",vuelta);
}
function vuelta(response)
{
$("#divRespuesta").append(response);
}
</script>
</head>
<body>
<div id="divRespuesta">El nombre del usuario es: </div>
</body>
</html>
```