Librerías

Grupo 13 — Integrantes

Ballestero, Letizia (alias: Leti)

Gaitan, Catalina (alias: Cath)

Pereyra, Lucia (alias: Lulu)

JQuery

📌 JQuery es una librería de JavaScript, libre y de código abierto.

Pero, desde el inicio *¿Qué es una librería?* Una librería ofrece una serie de funcionalidades básicas basadas en un lenguaje (en este caso, JavaScript) que nos ahorrarán código, ya que las librerías poseen funciones propias de las cuáles vamos a hacer uso.

Introducción

Con el uso de JQuery simplificamos (debido a una sintaxis sencilla y clara) diversas cuestiones, entre ellas:

- El modo en que interactuamos con los documentos HTML, ya que no vamos a trabajar directamente sobre el documento HTML.
- La manipulación del árbol DOM y la selección de sus elementos.
- El manejo de eventos.

Estos tres puntos mencionados se modifican con el uso de JQuery, ya que gracias a esta librería se puede manipular el DOM parar separar el documento JavaScript del HTML. Se debe a que agregamos *eventos manipuladores al DOM* (usando el archivo JavaScript) en lugar de *eventos atributos al HTML* para llamar funciones de JavaScript.

- El desarrollo de animaciones.
- Interacción con páginas web mediante la técnica AJAX.
- Se genera un enfoque **modular** a la hora de trabajar.
- Eliminación de incompatibilidad entre navegadores. Los motores de JavaScript difieren entre los distintos navegadores y el código puede funcionar diferentes y tener problemas de inconsistencia. JQuery permite una consistencia interna para que nuestro aplicación web tenga la misma funcionalidad en los diferentes navegadores.
- Permite la extensibilidad, se agrega nuevos eventos y luego pueden reutilizarse fácilmente.

Ejemplos de su uso

Se puede buscar en un documento a todos los elementos con alguna propiedad (por ejemplo la etiqueta H1) y cambiar uno o más de sus atributos como respuesta ante cierto evento.

Ventajas

Algunas ya mencionadas pero que queremos dejar en claro.

- Tiene una API fácil de usar y minimalista.
- Utiliza selectores *CSS3* para manipular las propiedades de estilo y encontrar elementos.
- jQuery es ligero, sólo necesita 30 kb para gzip y minifica, y soporta un módulo AMD.
- Como su sintaxis es bastante similar a la de *CSS*, es fácil de aprender para los principiantes.
- Ampliable con plugins.
- Versatilidad con una API que soporta múltiples navegadores, incluyendo Chrome y Firefox. Relacionada con la eliminación de compatibilidad antes mencionada.

Instalación

Hay dos versiones de JQuery disponibles:

- *Production version* - para sitios webs. Está comprimida.
- *Development version* - versión de desarrolladores, para pruebas y, justamente, desarrollar. Está sin comprimir y el código es leíble.

Ambas versiones puede ser descargadas desde el sitio de JQuery

La librería es JQuery es sólo un archivo de JavaScript que puede ser referenciado desde el HTML en su sección correspondiente:

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

El archivo tiene que estar en el mismo directorio que la página en la cuál se desea usar.

Otra forma es mediante CDN (*content delivery network*), de esta manera no tenemos que incluirlo desde el host.

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/j
</head>
```

La ventaja es que la mayoría de los usuarios han descargado jQuery desde Google visitando otros sitios. De esta manera, ya estará cargado en la caché cuándo visite el nuestro y cargará más rápido

Sintaxis de JQuery

Como ya se mencionó la sintaxis de JQuery selecciona un elemento y ejecuta una acción sobre ese elemento (o elementos). La sintaxis básica es:

```
$(selector).action(function)
```

- **$** para acceder a jQuery
- (*selector*) para encontrar los elementos del HTML, gracias a su nombre, id, class, type, atributos, valores de estos atributos etcs. Basado en los selectores que usa el CSS.
- *.action(function)* para cuándo se da un evento (action) ocurra la función que se llama. Ejemplos de eventos pueden leerse acá.

Los métodos jQuery están dentro de un documento con el evento *ready,* de esta forma:

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

  //
```

```
});
```

De esta manera evitamos correr el código antes que el documento termine de cargarse (esté listo), ya que de lo contrario podrían fallar varias acciones, como por ejemplo ocultar algo que todavía no se creo u obtener el tamaño de una imagen que todavía no se cargó.

Pero como somos eficientes y vagos (o eficiente porque somos vagos) claramente hay una versión incluso más corta de esto:

```
$(function(){

  // métodos jQuery

});
```

Efectos de jQuery — Animación

Esconder, mostrar, toggle, desvanecer, animar y deslizar.

Ejemplos: acá.

Algo muy curioso, es que puede ejecutarse la siguiente línea en un código sin que la animación o el efecto haya terminado de ejecutarse. Consecuentemente esto puede ocasionar errores. Por eso mismo, en estos casos, se utiliza funciones *callbacks* para asegurarse la correcta ejecución.

En este ejemplo aparecerá una alerta **una vez** se haya escondido el párrafo.

```
$("button").click(function(){
  $("p").hide("slow", function(){
    alert("The paragraph is now hidden");
  });
});
```

Encadenamiento

El encadenamiento permite ejecutar múltiples comandos de jQuery uno tras otro en el mismo elemento (o elementos). Esto favorece la eficiencia en cuánto al tiempo, ya que de esta manera el buscador no tiene que buscar el mismo elemento más de una vez.

Para encadenar acciones, simplemente se *appendea* (verbo derivado de append, creado una madrugada charlando sobre el trabajo práctico) la acción nueva a la anterior. Por ejemplo:

```
$("#p1").css("color", "red").slideUp(2000).slideDown(2000);
```

Al clickear sobre el elemento con la etiqueta ***#p1*** el texto se vuelve rojo y se desliza un elemento del tipo botón sobre el elemento ***#p1*** y después vuelve a deslizarse pero hacía abajo.

La manipulación del DOM

JQuery viene con un montón de métodos relacionados a manipular el DOM y hacer fácil el acceso a los elementos del mismo mediante los atributos.

Get Content - text(), html(), and val()

- `text()` - Settea o devuelve el contenido de texto del elemento seleccionado
- `html()` - Settea o devuelve el contenido del elemento seleccionado
- `val()` - Settea o devuelve el valor de determinados campos.

Por ejemplo:

```
$("#btn1").click(function(){
  alert("Text: " + $("#test").text());
});
$("#btn2").click(function(){
  alert("HTML: " + $("#test").html());
});
```

Dónde la primer alerta es un: "Text: This is some bold text in a paragraph."

Y la segunda: " HTML: This is some bold text in a paragraph."

Más ejemplos de manipulación del DOM mediante jQuery se puede ver acá

Añadir o Quitar elementos con jQuery

Se puede **añadir** elementos al HTML mediante:

- `append()` - Inserta contenido al final del elemento seleccionado
- `prepend()` - Inserta contenido al principio del elemento seleccionado
- `after()` - Inserta contenido antes del elemento seleccionado
- `before()` - Inserta contenido después del elemento seleccionado

Acá se puede encontrar ejemplos de estos métodos.

Los elementos se pueden **quitar** del HTML mediante:

- `remove()` - Remueve el elemento hijo seleccionado.
- `empty()` - Remueve *los* elementos hijos de un elemento seleccionado.

Ejemplos.

Manipulación del CSS

Métodos Get y Set para clases CSS

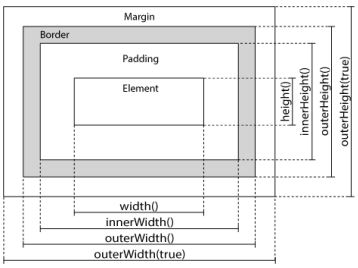
Podemos obtener el estilo de determinado atributo, podemos remover o añadir un estilo de clase al elemento seleccionado,

Ejemplos

Dimensiones en jQuery

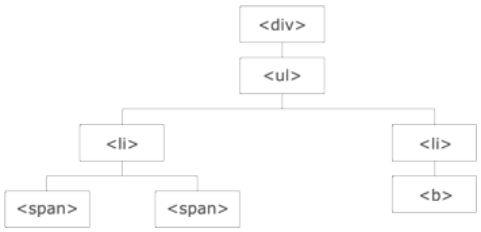
Algo bastante interesante son los métodos que funcionan con las dimensiones del CSS. Se puede leer más acá.

- `width()` el ancho del elemento
- `height()` el algo del elemento



- `innerWidth()` el ancho del elemento incluyendo el padding
- `innerHeight()` el alto del elemento incluyendo el padding
- `outerWidth()` el ancho del elemento incluyendo el borde
- `outerHeight()` el alto del elemento incluyendo el borde
- `outerWidth(true)` el ancho del elemento incluyendo el margen
- `outerHeight(true)` el alto del elemento incluyendo el margen

Traversing



Significa *"moverse a través de"* y se usa para encontrar (o seleccionar) elementos del HTML basado en la relación que tienen con otros elementos. Se empieza desde un elemento hasta encontrar el elemento deseado.

Está bastante relacionado con el DOM, ya que se usa la abstracción del mismo para buscar entre los *ancestros* y los elementos *descendientes* y *hermanos*

Ejemplos

- Métodos con los *ancestros*: acá. Son métodos que devuelven el elemento ancestro o hasta un ancestro en particular.
- Métodos con los *descendientes* : acá. Dónde tenemos métodos que devuelven el hijo directo o buscan un descendiente en particular.
- Métodos de *hermanos*: acá. Hay métodos que devuelven todos los hermanos, el siguiente, todos los que lo rodean, el siguiente hasta cierto criterio. Y así también con los anteriores, cambiando la sintaxis.
- Filtrado: acá. Los métodos básicos de filtrado, tales como primero, último, "not" , un índice o característica específica.

jQuery y AJAX

JQuery provee varios métodos para implementar AJAX.

- `load()` - carga información desde un servidor y la coloca en el elemento seleccionado.

más información y ejemplos: acá

- `$.get(url, callback)` y `$.post(URL,data,callback)` - para obtener y colocar información desde un elemento seleccionado.

más información y ejemplos: acá

Otros métodos

- `noConflict()` - ¿Qué pasa cuándo algún framework utiliza **$** como *shortcut*? En ese caso, alguno tendrá que detenerse. Por eso mismo existe este método, que mientras haya conflicto cede a otro framework o librería el uso de **$**

más información: acá