JAVASCRIPT

CLASE 13
Material complementario

Efectos y animaciones con jQuery

CODER HOUSE

Efectos y animaciones

JQuery provee métodos para incorporar efectos y animaciones en la interfaz, con el fin de conferir cierto nivel de interactividad a nuestros componentes HTML. A diferencia de otras animaciones o efectos, como los creados con CSS, estos están orientados a ser la parte de la respuesta a un proceso.

Trabajar con animaciones en jQuery nos permite combinar las diferentes variantes entre sí, para darle a una web o app dinamismo y movimiento, así como personalizarlas según nuestras necesidades.

Animación show

El método show() permite forzar la visibilidad de un elemento del DOM que se encuentre oculto mediante CSS (display:none); al seleccionarlo con JQuery y ejecutar show(), el elemento aparecerá en la pantalla:

```
//Agreguemos <h3> con jQuery ocultos con  style="display: none;"
$("body").prepend('<h3  style="display: none" >¡Hola Coder1!</h3>');
$("body").prepend('<h3  style="display: none" >¡Hola Coder2!</h3>');
//Mostramos con show() todos los <h3>
$("h3").show();
```

Animación hide

El método hide() permite ocultar cualquier elemento del DOM. Si el elemento es visible mediante CSS, al seleccionarlo con JQuery y ejecutar hide () el elemento desaparecerá de la pantalla. Tengan en cuenta que en el ejemplo a continuación, al utilizar el método hide() en “h3” voy a ocultar todas las etiquetas h3 creadas:

```
//Agregamos <h3> con jQuery
$("body").prepend('<h3>¡Hola Coder1!</h3>');
```

```
$("body").prepend('<h3>¡Hola Coder2!</h3>');
//Ocultamos con hide() todos los <h3>
$("h3").hide();
```

Si no se combinan con ninguna otra animación, las acciones de mostrar y ocultar (*show()* y *hide()*, respectivamente) se realizan de forma inmediata, sin transición.

Animación Fade In

El método fadeIn() cumple la misma función que el método show(), esto es, permite otorgar visibilidad a un elemento del DOM oculto (display:none), con la diferencia de que el elemento aparece con una transición visual. Cuando utilizamos fadeIn(), el elemento empieza a aparecer de forma gradual, hasta estar completamente visible.

```
//Agreguemos <h3> con jQuery ocultos con
style="display: none;"
$("body").prepend('<h3  style="display: none" >¡Hola
Coder1!</h3>');
$("body").prepend('<h3  style="display: none" >¡Hola
Coder2!</h3>');
//Mostramos con fadeIn() todos los <h3>
$("h3").fadeIn();
```

Animación Fade Out

El método fadeOut() tiene el mismo fin que el método hide(), sólo que en este caso, el elemento se desvanece suavemente en una transición visual. Es decir que al llamar fadeOut(), el elemento empieza a desaparecer de forma gradual, hasta estar completamente invisible.

```
//Agreguemos <h3> con jQuery"
$("body").prepend('<h3>¡Hola Coder1!</h3>');
$("body").prepend('<h3>¡Hola Coder2!</h3>');
```

```
//Ocultamos con fadeOut() todos los <h3>
$("h3").fadeOut();
```

Velocidad del Fade

Tanto para fadeOut() como fadeIn() se puede enviar un parámetro opcional para indicar la velocidad de la transición del fade. Este parámetro puede ser enviado con las palabras “slow" o "fast", o también con un número en milisegundos que indica la duración total de la transición. En caso de que no se defina ningún parámetro, estas animaciones se ejecutarán con su velocidad por defecto (400 ms).

Callback del Fade

Para las animaciones fadeOut() y fadeIn(), existe la posibilidad de configurar que se ejecute una acción al finalizar las mismas. Para lograr esto, se le puede enviar un segundo parámetro opcional para indicar una función que se ejecutará cuando la transición de la animación termine. En el siguiente ejemplo, podemos apreciar que cuando la animación del fadeOut() termine, se ejecutará una nueva función que realiza un fadeIn() en la etiqueta h3, generando un efecto “blink” (parpadeo) de la etiqueta:

```
//Agregamos <h3> con jQuery
$("body").prepend('<h3>¡Hola Coder1!</h3>');
//Ocultamos con fadeOut() todos los <h3>
$("h3").fadeOut("slow", function(){
    //Cuando termina de ocultarse el elemento lo
```

```
mostramos nuevamente
    $("h3").fadeIn(1000);
);
```

Animación Slide Down

El método slideDown() posibilita la visualización de un elemento, haciendo una transición hacia abajo. Se presenta como una alternativa al fadeIn(), ya que también permite enviar los mismos parámetros de velocidad y callback que en fadeIn y fadeOut. Generalmente se utiliza este tipo de animación para destacar una salida después de que se realice un evento. En el ejemplo a continuación, podemos ver que después del evento click se realiza la animación del div1:

```
//Agregamos un botón y un div con jQuery
$("body").prepend('<button id="btn1">Mostrar</button>');
$("body").prepend(`<div id="div1" style="height: 120px">
                        <h3>¡Hola Coder!</h3>
                        <h4>Sorpresa 2</h4>
                    </div>`);
//Usamos slideDown sobre div1 en respuesta al click del boton
btn1
$("#btn1").click(() => {
    $("#div1").slideDown("fast");});
```

Animación Slide Up

El método slideUp() hace desaparecer al elemento, ocultándolo con transicion hacia arriba. Es una alternativa al fadeOut(), ya que permite enviar los mismos parámetros de velocidad y callback que en fadeIn y fadeOut. En el siguiente ejemplo, similar al anterior, podemos ver que luego del evento click, se realiza la animación de slideUp() para ocultar el elemento, desvaneciéndolo hacia arriba:

```
//Agregamos un botón y un div con jQuery
```

```
$("body").prepend('<button id="btn1">Mostrar</button>');
$("body").prepend(`<div id="div1" style="height: 120px">
                        <h3>¡Hola Coder!</h3>
                        <h4>Sorpresa 2</h4>
                    </div>`);
//Usamos slideUp sobre div1 en respuesta al click del boton btn1
$("#btn1").click(() => {
    $("#div1").slideUp("fast");
});
```

Animación Slide Toggle

El método slideToggle() sirve para mostrar u ocultar un elemento mediante slide. Esto lo realiza evaluando el elemento con el que interactúa; si el mismo está visible, ejecuta slideUp() para ocultarlo; en caso contrario, ejecuta slideDown() para desplegarlo. Entre otras cosas, este método se utiliza comúnmente para realizar menús interactivos:

```
//Agregamos un botón y un div con jQuery
$("body").prepend('<button id="btn1">Mostrar</button>');
$("body").prepend(`<div id="div1" style="height: 120px">
                        <h3>¡Hola Coder!</h3>
                        <h4>Sorpresa</h4>
                    </div>`);
//Usamos toggle sobre div1 en respuesta al click del botón btn1
$("#btn1").click(() => {
    $("#div1").toggle("fast");
});
```

Modificar CSS en jQuery

JQuery provee un método para personalizar las reglas de estilo CSS de un elemento del DOM. Para realizar esto, debemos emplear el método css(), con el cual podemos indicar la propiedad a modificar del elemento seleccionado. La sintaxis funciona así: selector.css("propiedad-css","valor");

Como podemos ver en el ejemplo a continuación, por parámetro podemos indicar cada una de las modificaciones de estilo; también podríamos pasarle un objeto con todas las reglas de estilo a modificar:

```
//Agregamos un párrafo con jQuery
$("body").prepend('<p class="titulo">Code House</p>');
//Modificamos las reglas CSS desde jQuery
$("p").css("background-color", "yellow");
$("p").css("width", "50%");
$(".titulo").css({  "color": "#ccc",
                    "font-size": "40px",
                    "borderLeft": "5px solid #ccc" });
```

Método Animate

El método animate() nos permite crear nuestros propios efectos de animación sobre cualquier propiedad CSS numérica (opacidad, alto, ancho, margen, etcétera). Se

requiere como parámetro un objeto de propiedades CSS, similar al que se puede enviar al método css(), con la diferencia de que el rango de propiedades es más restrictivo.

Como vemos en el ejemplo a continuación, para utilizar este método debemos proporcionarle, en primer lugar, el objeto con las propiedades CSS que queremos obtener en el elemento seleccionado, y luego, opcionalmente, podemos agregar la velocidad de duración de la animación, y un callback (de igual manera que en los métodos fadeOut y fadeIn):

```
//Agregamos un párrafo con jQuery
$("body").prepend('<p class="titulo">Code House</p>');
//Animamos sus propiedades CSS con animate
$(".titulo").animate({  left:'250px',
                        opacity:'0.5',
                        height:'150px',
                        width:'150px'   }, //1er parámetro propiedades
                        "slow",            //2do parámetro duración
                        function(){        //3er parámetro callback
                            console.log("final de animación");
                            });
```

Ejemplo aplicado: Scroll Animado

En el siguiente código vemos un ejemplo de Scroll Animado. Un scroll es una acción de desplazamiento que se produce en una página al presionar un enlace o botón, el cual redirige el foco a otra sección de la página automáticamente. Utilizando los métodos de animación de jQuery, podemos replicar este comportamiento agregándole un efecto de transición visual, el cual otorga al usuario una experiencia más interactiva, "fluida", en la página:

```
//Agregamos una estructura con jQuery
$("body").prepend(`</div>
                        <a>Ir a contacto</a>
                        <p style="height: 800px"></p>
                        <section id="seccionContacto">
```

```
                        <h4>¡Somos Coders!</h4>
                    </section>
                </div>`);
// Asociamos la animación al click en un elemento <a>
$('a').click( function(e) {
    e.preventDefault();
    //Animamos sus propiedades CSS con animate
    $('html, body').animate({
        scrollTop: $("#seccionContacto").offset().top
    }, 2000);
} );
```

Encadenar animaciones

En los casos en los exista la necesidad de realizar una sucesión de animaciones, jQuery nos provee la posibilidad de encadenar métodos, lo cual permite definir una animación tras otra empleando en una única declaración.

Es recomendable emplear esta forma si es necesario asociar múltiples animaciones consecutivas a un mismo elemento, lo cual nos permite ahorrarnos consultas al DOM. Este encadenamiento puede ser utilizado con cualquier tipo de método que provea jQuery.

Como vemos en el ejemplo a continuación, nuestro elemento p1 realiza primero una modificación de estilo mediante el método css(), y luego ejecuta las animaciones slideUp() y slideDown():

```
//Agreguemos un párrafo con jQuery
$("body").prepend('<p id="p1">Coder House</p>');
//Declaración de métodos encadenados
$("#p1").css("color", "red")
        .slideUp(2000)
```

```
        .slideDown(2000);
```

Método Delay

El método delay() nos permite retrasar, durante una determinada cantidad de tiempo, la ejecución entre dos métodos encadenados. Suele utilizarse para establecer un tiempo de espera entre una animación y la siguiente. Para usarlo, debemos enviarle como parámetro el tiempo que se va a esperar en milisegundos:

```
//Agregamos un párrafo con jQuery
$("body").prepend('<p id="p1">Coder House</p>');
//Declaración de métodos encadenados
$("#p1").css("color", "red")
        .slideUp(2000)
        .delay(2000)
        .slideDown(2000);
```