UT4. Integración de contenido interactivo

01 de Septiembre de 2017

I.E.S. Virgen de la Paz

Introducción

- En esta unidad añadiremos interactividad a los diseños web y utilizaremos una biblioteca para facilitar la creación de elementos interactivos.
- Trabajaremos con:
  - Elementos interactivos básicos y avanzados.
  - Cambio de las propiedades de un elemento.
  - Ejecución de secuencias de comandos.
  - Comportamiento de los elementos de efectos visuales.
  - Comportamientos interactivos.
  - Reproducción de sonido, vídeo y animación.

Animaciones

- Las animaciones ocupan un lugar muy importante a la hora de lograr que las aplicaciones y los sitios web resulten atractivos
- Los usuarios desean usar interfaces de usuario altamente receptivas e interactivas
- Sin embargo, animar tu interfaz no es necesariamente una tarea simple
  - ¿Qué se debe animar?
  - ¿En qué momento?
  - ¿Qué tipo de apariencia debe tener la animación?

Animaciones

- El objetivo es conocer la manera de incluir y generar animaciones para un sitio web, mostrando la tecnología y las tendencias actuales en este campo, pero sin profundizar en el uso de una herramienta concreta
- Las animaciones web se usan para muchos fines:
  - Para publicidad, como banners
  - Objetos animados (botones, imágenes, opciones de menús...)
  - Slides de imágenes
  - Presentar nuevos contenidos según avanza la página
  - etc

Animaciones

Ejemplos

- https://www.beoplay.com/products/beoplayh7#buy
- http://www.unoknokke.be/
- https://www.truedigital.co.uk/
- https://yearinreview.fb.com/2015/syrian-civil-war-refugee-crisis/
- https://seasoncreates.com/
- https://www.gobag.co/
- https://aventus.io/
- https://cardconnect.com/

Animaciones

- Actualmente la web ofrece muchas alternativas para la creación de estas animaciones
  - De pago y libres
  - Implementadas por todos los navegadores, otros son exclusivas de unos pocos
  - Siguen el estándar W3C y otras no.
- La variedad es mucha, y el desarrollador tiene que elegir en cada momento la solución que mejor se adapte a sus necesidades, a la tecnología empleada en el desarrollo del sitio web y a la tecnología que permiten los navegadores

Animaciones

- El desarrollador y diseñador de un sitio web debe conocer todas las alternativas actuales para la creación de animaciones para sitios web
- Un desarrollador web no tiene que ser necesariamente creativo, ni un artista del diseño digital
- Muchas veces ocurre que un desarrollador sabe manejar una herramienta del tipo Google Web Designer, pero eso no quiere decir que lo que diseña con ella sea atractivo.

Animaciones con CSS

- Las animaciones CSS3 permiten animar la transición entre un estilo CSS y otro
- Las animaciones constan de dos componentes:
  - Un **estilo** que describe la animación
  - **Conjunto de fotogramas** que indican su estado inicial y final, así como posibles puntos intermedios en la misma.
- Las animaciones CSS tienen tres ventajas principales sobre las técnicas tradicionales de animación basada en scripts
  - Es muy fácil crear animaciones sencillas
  - No se necesitan equipos potentes para ejecutarlas
  - El navegador controla la secuencia de la animación, permitiendo optimizar el rendimiento y la eficiencia

Animaciones con CSS

Configurando la animación

- Para crear una secuencia de animación CSS usaremos la propiedad animation y sus sub-propiedades
- Con ellas podemos no solo configurar el ritmo y la duración de la animación sino otros detalles sobre la secuencia de la animación
- Con ellas no configuramos la apariencia actual de la animación, para ello disponemos de @keyframes

Animaciones con CSS

Configurando la animación

- **animation-delay**: Tiempo de retardo entre el momento en que el elemento se carga y el comienzo de la secuencia de la animación.
- **animation-direction**: Indica si la animación debe retroceder hasta el fotograma de inicio al finalizar la secuencia o si debe comenzar desde el principio al llegar al final.
- **animation-duration**: tiempo que dura la animación
- **animation-iteration-count** número de veces que se repite. Con el valor infinite se repite la animación indefinidamente.
- **animation-name** Nombre de la regla @keyframes que describe los fotogramas de la animación.
- **animation-play-state** Pausar y reanudar la animación
- **animation-timing-function** Indica el ritmo de la animación, es decir, como se muestran los fotogramas de la animación, estableciendo curvas de aceleración.
- **animation-fill-mode** Especifica qué valores tendrán las propiedades después de finalizar la animación

Animaciones con CSS

Configurando la animación

- Una vez configurado el tiempo de la animación, necesitamos definir su apariencia
- Se establecen dos fotogramas más con @keyframes
- Cada uno describe cómo se muestra cada elemento animado en un momento dado durante la secuencia
- Desde que se define el tiempo y el ritmo de la animación, el fotograma usa percentage para indicar en qué momento de la secuencia la animación tiene lugar
  - 0% es el principio, alias from
  - 100% es el estado final de la animación. alias to
- Hay que especificar estos dos momentos para que el navegador sepa dónde debe comenzar y finalizar
- Opcionalmente se pueden añadir fotogramas que describan pasos intermedios de la animación.

Animaciones con CSS

Ejemplo animación de rojo a amarillo

```
/* The animation code */
@keyframes example {
    from {background-color: red;}
    to {background-color: yellow;}
}

/* The element to apply the animation to */
div {
    width: 100px;
    height: 100px;
    background-color: red;
    animation-name: example;
    animation-duration: 4s;
}
```

Animaciones con CSS

Ejemplo animación de rojo a amarillo a azul y finalmente a verde

```
/* The animation code */
@keyframes example {
    0%   {background-color: red;}
    25%  {background-color: yellow;}
    50%  {background-color: blue;}
    100% {background-color: green;}
}
/* The element to apply the animation to */
div {
    width: 100px;
    height: 100px;
    background-color: red;
    animation-name: example;
    animation-duration: 4s;
}
```

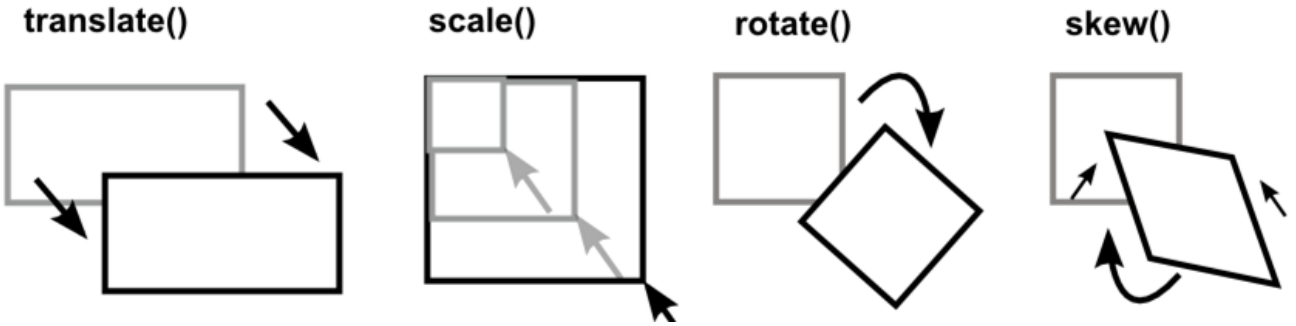
Transformaciones con CSS

- Las transformaciones en CSS3 permiten realizar efectos visuales, incluido 2D y 3D
- Las propiedades principales para realizar transformaciones son las siguientes:
  - **transform**: (funcion1, función2,...) Aplica una o varias funciones de transformación sobre un elemento.
  - **transform-origin**: (pos-x pos-y pos-z) Cambia el punto de origen del elemento en una transformación.
  - **transform-style**: (flat | preserve-3d) Modifica el tratamiento de los elementos hijos.

Transformaciones con CSS

Funciones de transformación 2D

- Existen múltiples propiedades CSS que ofrecen diferentes funcionalidades de transformación en dos dimensiones
  - **translate** mueven un elemento de un lugar a otro
  - **scale**: aumentan o reducen el tamaño de un elemento
  - **rotate**: gira el elemento el número de grados indicado:
  - **skew**: establecen un ángulo para torcer o inclinar un elemento en 2D



https://css-tricks.com/almanac/properties/t/transform/

Transformaciones con CSS

Funciones de transformación 3D

- **translateZ(z)** Traslada el elemento una distancia de z en el eje de profundidad.
- **translate3d(x, y, z)** Establece una translación 3D, donde aplica los parámetros a cada eje.
- **scaleZ(fz)** Reescala el elemento a un nuevo tamaño con factor fz de profundidad.
- **scale3d(fx, fy, fz)** Establece un escalado 3D, donde aplica los factores a cada eje.
- **rotateZ(zdeg)** Establece una rotación 2D de zdeg grados sólo para el eje de profundidad Z.
- **rotate3d(x, y, z, deg)** Establece una rotación 3D, donde aplica a cada eje el número de grados deg.
- **perspective(n)** Establece una perspectiva 3D
- **matrix3d(n, n, ...)** Establece una matriz de transformación 3D (16 valores)
- https://desandro.github.io/3dtransforms/

Transiciones con CSS

- Permiten cambiar los valores de una propiedad suavemente con una duración determinada
  - **transition-property** Especifica el nombre/s de las propiedades CSS a las que deberían aplicarse las transiciones
  - **transition-duration** duración de la transiciones. Puedes especificar una única duración o valores múltiples que permitan a cada propiedad de transición un período de tiempo diferente.
  - **transition-timing-function** Especifica la curva cúbica bézier que se usa para definir cómo se computan los valores intermedios para las propiedades.
  - **transition-delay** tiempo de espera entre el momento en que se cambia una propiedad y el inicio de la transición.
- https://robots.thoughtbot.com/transitions-and-transforms

Animaciones con JavaScript

- Crear animaciones con JavaScript mucho más complejo que escribir transiciones o animaciones de CSS,
- Generalmente le proporciona a los programadores mucho más control
- Puedes usar la Web Animations API,para animar propiedades de CSS específicas o para crear objetos de efecto componible.
- Las animaciones de JavaScript son imperativas, cuando las escribes de manera integrada como parte de un código
- También puedes encapsularlas dentro de otros objetos.

Animaciones con JavaScript

Ejemplo animación en JavaScript

```
train.onclick = function() {
      let start = Date.now();
      let timer = setInterval(function() {
      let timePassed = Date.now() - start;

        train.style.left = timePassed / 5 + 'px';

        if (timePassed > 2000) clearInterval(timer);

      }, 20);
    }
}
```

http://plnkr.co/edit/LabvtJ7jitdbsN6BXuQ1?p=preview

Animaciones con JavaScript

- La Web Animations API es un estándar nuevo de W3C.
- Es compatible de forma nativa con **Chrome** y **Opera** y se encuentra en proceso activo de desarrollo para **Firefox**.
- Para otros navegadores modernos, se encuentra disponible mediante **polyfill**.
- Con las animaciones de JavaScript, puedes controlar totalmente los estilos de un elemento en cada paso.
- Esto significa que es posible reducir la velocidad de las animaciones, pausarlas, detenerlas, invertirlas y manipular elementos según lo creas conveniente
- Esto resulta especialmente útil si creas apps complejas orientadas a objetos, ya que puedes encapsular de forma adecuada tu comportamiento.

Animaciones con JavaScript

Propiedades de Animation

- **currentTime** El valor del tiempo de la animación actual en milisegundos
- **effect** Obtiene y establece el AnimationEffectReadOnly asociado con esta animación. Normalmente, este será un objeto KeyframeEffect.
- **finished** Devuelve la Promesa actual terminada para esta animación.
- **id** Obtiene y establece el identificador de la animación.
- **playState** Devuelve un valor enumerado que describe el estado de la reproducción de una animación.
- **playbackRate** Obtiene o establece la velocidad de reproducción de la animación.
- **ready** Devuelve la Promesa actual lista para esta animación.
- **startTime** Obtiene o establece la hora programada en la que debe comenzar la reproducción de la animación.
- **timeline** Obtiene o establece el timeline asociado con esta animación.

Métodos de Animation

- **cancel()** Borra todos los keyframeEffects causados por la animación y aborta la reproducción.
- **finish()** Seeks either end of an animation, depending on whether the animation is playing or reversing.
- **pause()** Pone la reproducción en pausa.
- **play()** Busca el final de una animación, dependiendo de si la animación se está reproduciendo o si se está invirtiendo.
- **reverse()** Invierte la dirección de reproducción, deteniéndose al comienzo de la animación. Si la animación finaliza o no se activa, volverá a reproducirse desde el final al principio.

Animaciones con JavaScript

Ejemplo de Animation

```
    var target = document.querySelector('.box');
        var player = target.animate([
  {transform: 'translate(0)'},
  {transform: 'translate(100px, 100px)'}
], 500);
player.addEventListener('finish', function() {
  target.style.transform = 'translate(100px, 100px)';
});
```

https://googlesamples.github.io/web-fundamentals/fundamentals/design-and-ux/animations/box-move-wa.html

https://developer.mozilla.org/en-US/docs/Web/API/Web_Animations_API/Using_the_Web_Animations_API

- https://developer.mozilla.org/en-US/docs/Web/API/Web_Animations_API/Using_the_Web_Animations_API
- http://danielcwilson.com/blog/2015/07/animations-part-1/

jQuery

¿Qué es jQuery?

- Es una librería de funciones javascript
- Su objetivo es hacer más fácil las tareas a los desarrolladores
  - Manipulando la página web
  - responder a eventos del usuario
  - crear efectos y animaciones
  - y mucho más
- jQuery está encima del framework que nos ofrecen los navegadores con Javascript y DOM permitiendo hacer lo mismo con menos líneas de código
- La última versión disponible es la 3.2.1

JavaScript vs jQuery

- Imagina que queremos cambiar el tamaño de todas las imágenes de nuestra página
- Podemos hacerlo mediante Vainilla javaScript con unas pocas líneas de código

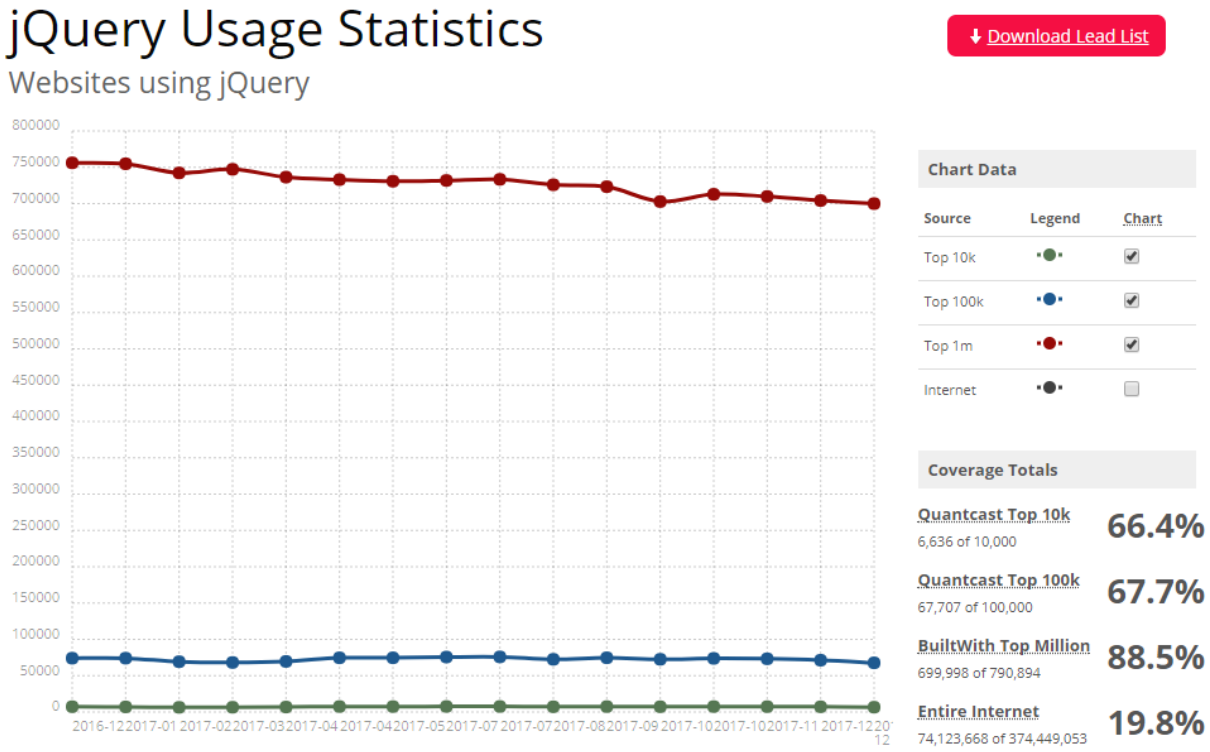
```
document.getElementsByTagName("img")
        .forEach(function(el);
             el.style.width = "50px";
});
```

- O hacer lo mismo con una sola línea de código con jQuery

```
$("img").width(50);
```

jQuery

- jQuery se encarga que nuestro código sea compatible con todos los navegadores
- En 2017, el 60% de las páginas web incluían jQuery



jQuery

añadiendo la librería

- El primer paso para utilizar jQuery en tu sitio web es incluir la librería de jQuery
- Añadimos una etiqueta <script> al final, justo antes del cierre de la etiqueta body: </body>

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
```

- Esta url coge la última versión de la librería de jquery del servidor CDN (Content Delivery Network) googleapis.com

Empezando con jQuery

Core jQuery

- En javaScript tenemos muchos nombres de funciones y tenemos que consultar la documentación para saber que nombres usar
- En jQuery existe una única función principal, jQuery()

```
jQuery();
```

- La podemos llamar también con el símbolo del dólar $

```
$();
```

- http://api.jquery.com/jQuery/

Empezando con jQuery

Llamando a funciones de jQuery

- Dependiendo lo que le pasemos, realizará una acción
  - Ejemplo: Obtener todos los elementos h1 en una colección

```
$("h1");
```

- Podemos concatenar distintas funciones en la función principal para manipular nuestra selección
  - Ejemplo: Cambiar el contenido de todos los elementos h1 por “hola mundo”

```
$("h1").text("hola mundo");
```

Acceso a DOM con jQuery

Encontrar elementos con jQuery

- jQuery permite seleccionar elementos por **selectores CSS** como la sentencia **querySelectorAll** de JavaScript
  - `Selectores de etiqueta: $("p");`
  - `Selectores de id $("#main-heading");`
  - `Selectores de clase $(".note");`
- Esto incluye selectores más complejos, como los de relaciones y atributos
  - `$("ul li ");`
  - `$("article : h1 ");`
  - `$("p > a ");`
  - `$("h1 + h2");`
  - `$("a[href*=madrid.org]");`
- http://api.jquery.com/category/selectors/

Acceso a DOM con jQuery

Obteniendo información de elementos con jQuery

- jQuery permite obtener información de elementos seleccionados previamente
- La misma función que utilizamos para cambiar el contenido, nos sirve para obtener su valor

```
var headingText = $("#main-heading").text();
```

- Las funciones en jQuery se comportan diferente dependiendo de si le pasamos o no argumentos

Modificando DOM con jQuery

Métodos de jQuery para modificar elementos existentes

- El método html() de jQuery nos permite añadir nuevas etiquetas

```
$(#menu).html("<li>Nueva opción</li>");
```

- El método css() permite cambiar el css

```
$(#menu).css("color","red");
```

- El método addClass() añade una clase CSS existente

```
$("h2").addClass("title");
```

- El método attr() añade o modifica un atributo

```
$("a[href*='madrid.org']").attr("href","http://ctif.madridnorte.educa.madrid.org/");
```

- http://api.jquery.com/category/attributes/

Modificando DOM con jQuery

crear elementos con jQuery

- Indicamos a la función principal la etiqueta que queremos crear

```
var menuItem = $("<li>");
```

- Añadimos el contenido a nuestro nuevo elemento

```
menuItem.text("Nueva opción");
```

- Podemos añadirle también una clase

```
menuItem.addClass("seleccionado");
```

- Y por último lo añadimos a nuestra página con el método append

```
$("#menu").append(menuItem);
```

jQuery collections vs DOM nodes

- Fijate en estas dos sentencias:

```
var headingFromD = document.getElementById("header");
var headingFromJ = $("#header");
```

- La primera devuelve un **nodo DOM** en el que se pueden ejecutar funciones como innerHTML
- La segunda devuelve una **colección jQuery** en el que se puede ejecutar el método html
- Solo podemos ejecutar métodos de jQuery en jQuery Collections, para recordarlo, podemos nombrar las variables que guarden colecciones jQuery para que comiencen por dólar: `var $heading = $("#header");`

jQuery collections vs DOM nodes

- En ocasiones nos puede interesar acceder al nodo DOM de una colección jQuery para ello debemos tratarlo como un array y obtener el primer elemento

```
headingD = $heading[0];
```

- En otras ocasiones nos puede interesar convertir un nodo DOM en una colección jQuery

```
$headingJ = $(headingD);
```

- Esto muchas veces no es necesario, pero es interesante saberlo y ayuda a entender la diferencia entre nodos DOM y colección jQuery

Técnicas jQuery

Recorrer jQuery collections

- Hay dos formas de recorrer colecciones jQuery
- Utilizando un bucle for tratando la colección jQuery como un array

```
for (var i = 0; i < $coleccion.length; i++) {
    var elemento = $coleccion[i];
    var $elemento = $(elemento);
    // realizar modificaciones en el elemento

    }
```

Recorrer jQuery collections

- Con el método **each** de jQuery similar al método **forEach** de JavaScript

```
$coleccion.each(function(index, element) {
    var $elemento = $(elemento);
   // realizar modificaciones en el elemento
 });
```

- Podemos utilizar this para simplificarlo

```
$coleccion.each(function() {
        var $elemento = $(this);
    // realizar modificaciones en el
elemento
 });
```

Concatenando métodos en jQuery

- Podemos acortar las líneas de código en jQuery concatenando métodos:

```
var menuItem = $("<li>");
menuItem.text("Nueva opción");
menuItem.addClass("seleccionado");
$("#menu").append(menuItem);
```

- Podemos acortarlo en :

```
$("<li>").text("Nueva opción")
 .addClass("seleccionado").appendTo("#menu");
```

- Esto es posible porque la mayoría de los métodos jQuery devuelven una colección jQuery

Concatenando métodos en jQuery

- Para mejorar la visibilidad podemos añadir retornos de carro y tabulaciones/espacios en cada llamada

```
$("<li>")
        .text("Nueva opción")
        .addClass("seleccionado")
    .appendTo("#menu");
```

Eventos

- Para añadir eventos en una colección jQuery utilizamos el método **on(event;function())**
  - El primer parámetro es el evento, por ejemplo “click”
  - El segundo la función que se va a ejecutar, por ejemplo una función anónima

```
$("#menu").on("click", function() {
        // hacer cosas
});
```

Eventos DOM con jQuery

propiedades de los eventos

- Los eventos tienen propiedades que pueden sernos muy útiles, para ello introduciremos el parámetro event
- Por ejemplo cada vez pulsamos en un elemento jQuery nos ayuda a obtener las coordenadas

```
$("#logo").on("click", function(event) {
        console.log(event.pageY);
         console.log(event.pageX);
 });
```

- http://api.jquery.com/category/events/event-object/

Eventos DOM con jQuery

Cargar javaScript cuando esté DOM cargado

- Por lo general el código javascript se carga al final de la página para que el usuario vea el contenido de la página rápidamente
- Pero a veces puede interesarte añadir código en la etiqueta head
- El problema es que el DOM todavía no se ha cargado, y puedes tener resultados indeseados
- Para evitarlo podemos llamar al método ready, que se llamará cuando haya terminado de cargar todos los elementos DOM de la página.

```
$(document).ready(function() {
    // hacer cosas
});
```

Procesando formularios con jQuery

- Igual que podemos hacer con JavaScript, jQuery permite procesar formularios
- Podemos realizar comprobaciones con los eventos del formulario o al realizar el submit
- Para procesar un submit solo tenemos que añadir el método on con la acción submit en la colección jQuery del formulario

```
$("#form").on("submit", function() {

}
```

- http://api.jquery.com/category/forms/

Procesando formularios con jQuery

- Los formularios normalmente después de enviar los datos al servidor por defecto recargan la página
- Para evitar este efecto pasaremos el argumento event y llamaremos al método preventDefault

```
$("#form").on("submit", function(event) {

    event.preventDefault();

}
```

Procesando formularios con jQuery

- Dentro de la función callback podemos hacer referencia al formulario con la palabra reservada this
- Un patrón muy utilizado es llamar al método **find()** en el elemento del formulario para buscar inputs dentro de el

```
$("form").on("submit", function() {
  // guardar el valor del input edad
  var edad= $(this).find('[name=edad]').val();
   });
```

Animaciones y efectos con jQuery

- Una animación típica es hacer aparecer elementos en nuestra página web
- jQuery tiene los métodos **hide()** y **show(tiempo_ms)** para ocultar y mostrar elementos poco a poco
- Podemos utilizar otros métodos más vistosos como
  - slideDown
  - slideUp
  - fadeIn
  - slideToggle
  - etc
- http://api.jquery.com/category/effects/

- A los métodos se les puede pasar un callback para que se ejecute cuando finalice la animación

```
$("#pic").toggle(1000, function() {
    $("body").append("It's here!");
});
```

- También puedes concatenar múltiples efectos y llamar a delay() si quieres añadir un retardo entre ellos

```
$("#pic").slideUp(300).delay().fadeIn();
```

- Para crear animaciones personalizadas tenemos el método **animate()**

```
$("#pic").animate({
        width: "70%",
        opacity: 0.7,
        padding: 20
   }, 1000);
```

- Solo se puede animar propiedades CSS con valores numéricos, no podrías usar por ejemplo color

- Las animaciones deben mejorar la experiencia del usuario no empeorarla haciendo la carga más lenta y tediosa
- Puedes pedir a los usuarios que te den retroalimentación sobre la experiencia o trabajar con un diseñador que decida que animar y que no
- También puedes dar la opción al usuario de utilizar animaciones o no, para ello solo tienes que cambiar la variable **$.fx.off** a verdadero

jQuery UI

- jQuery User Interface es una biblioteca de componentes para el framework jQuery que le añaden un conjunto de plug-ins, widgets y efectos visuales para la creación de aplicaciones web
- Cada componente o módulo se desarrolla de acuerdo a la filosofía de jQuery5 (find something, manipulate it: encuentra algo, manipúlalo).

jQuery UI

Comportamientos complejos

- Draggable: Hace al elemento arrastrable.
- Droppable: Permite que el elemento responda a elementos arrastrables.
- Resizable: Permite redimensionar el elemento.
- Selectable: Permite seleccionar entre una lista de elementos.
- Sortable: Ordena una lista de elementos.

jQuery UI

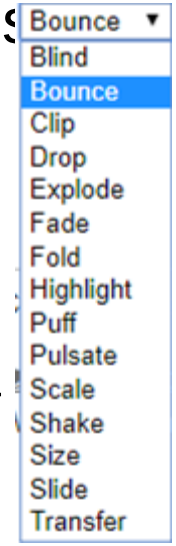
Widgets

- Accordion: Menú con efecto acordeón.
- Autocomplete: Caja con autocompletado.
- Button: Botón.
- Dialog: Ventanas con contenido.
- Slider: Elemento para elegir en un rango de valores.
- Tabs: Pestañas.
- Datepicker: Calendario gráfico.
- Progressbar: Barra de progreso.

jQuery UI

Efectos

- addClass, RemoveClass, switchClass y ToggleClass: Añade, quita, cambia o intercambia una clase por otra animando mientras se aplican los estilos
- colorAnimation: Anima las propiedades de elementos entre colores
- easing: Aplica una animación de tipo s
- Effect: Aplica un efecto de animación a un elemento
- Hide, Show y Toggle: Oculta, muestra o intercambia elementos aplicando un efecto



jQuery

plugins

- Existen cientos de plugins que permiten utilizar animaciones basadas en jQuery
- Algunos ejemplos:
  - https://speckyboy.com/free-jquery-plugins/
  - http://www.creativebloq.com/jquery/top-jquery-plugins-6133175
  - http://cumboandrea.me/26-jquery-plugins-muy-utiles/

Otras librerías para animaciones

- Animate.css
- Bounce.js
- AnimeJS
- Magic Animations
- DynCSS
- CSShake
- Hover.CSS
- Velocity.js
- AniJS
- GreenSock

Animaciones

Recursos

- https://developers.google.com/web/fundamentals/design-and-ux/animations/?hl=es
- https://www.w3schools.com/css/css3_animations.asp
- https://javascript.info/css-animations
- https://developer.mozilla.org/es/docs/Web/CSS/CSS_Animations/Usando_animaciones_CSS
- https://codepen.io/tag/css%20animation/

Animaciones

Ejemplos

- https://tympanus.net/codrops/2014/12/15/elastic-svg-elements/
- https://medium.com/net-magazine/create-a-set-of-micro-animations-6bb42a292f8b