Desarrollo de aplicaciones multiplataforma

Elaborado por: José Antonio Sánchez

Módulo 3

Programación HTML5

Innova4B

SESIÓN 3

Innova4B

5.4. Ejercicio práctico

Crear un formulario que contenga lo siguiente:

- Los 12 nuevos tipos de elementos input.
- El nuevo elemento datalist, que contenga algunos nombres de provincia y un campo de texto que se relacione con él.
- Una caja de texto (<input type="text">), a la cual aplicar los atributos autofocus, placeholder, required y autocomplete.
- Una caja de texto que sólo pueda contener números (<input type="number">), cuyos valores tienen que estar comprendidos entre 0 y 10.
- Un campo de selección de ficheros (<input type="file">), al que aplicar el atributo multiple.
- Un campo de introducción de password (<input type="password">), donde el valor introducido debe cumplir lo siguiente: debe tener una longitud mínima de 8 caracteres, comenzar por un número y terminar en una letra mayúscula.
- Un nuevo elemento progress que represente el avance de completado de campos del formulario.
- Utilizar un elemento meter, para simular la ocupación de un disco duro de 1TB, suponiendo que un grado alto de ocupación sea 800GB

5.4. Ejercicio práctico

Acceder al formulario desde al menos 4 navegadores (2 de escritorio y 2 de dispositivos móviles), y comprobad el comportamiento y funcionamiento en cada elemento del formulario

Anotar dichos resultados en una hoja de cálculo para futuras referencias

Probar las funcionalidades con un IE7 e IE8:

http://utilu.com/IECollection/

Innova4B

6. Modernizr

Es una librería JavaScript que nos permite conocer la compatibilidad del navegador con tecnologías HTML5 y CSS3

Esto nos permitirá desarrollar sitios web que se adaptados a las capacidades de los distintos navegadores, o los más utilizados

Sabiendo que nuestro navegador soporta ciertas capacidades de CSS3 o de HTML5, podremos utilizarlas con libertad

Si sabemos que un navegador no es compatible con determinada funcionalidad, podremos implementar variantes o “hacks”

6.1. Cómo utilizar Modernizr

Descargar el archivo con el código fuente de Modernizr

http://modernizr.com/download/

Se trata de un archivo con código JavaScript que podemos encontrar en dos variantes:

- Development: contiene el código fuente completo, sin comprimir y con comentarios.
  Se utiliza en desarrollo o si queremos acceder a su código para comprenderlo o ampliarlo
- Production: es OBLIGATORIO utilizar esta variante cuando pasamos a un entornos de producción

6.1. Cómo utilizar Modernizr

Al descargar Modernizr, tenemos la posibilidad de generar una librería únicamente con las funcionalidades que queremos detectar, esto nos permite ahorrar la carga de KB innecesarios

Se incluirá como cualquier otro fichero JavaScript

<script src="/js/lib/vendor/modernizr-custom.min.js"></script>

Se aconseja colocar el script dentro del HEAD, porque debe cargarse antes del BODY de la página, debido a un componente que quizás utilicemos, para permitir HTML5 en Internet Explorer, llamado HTML5 Shiv

6.1. Cómo utilizar Modernizr

Además, se recomienda colocarlo después de los estilos CSS para evitar un comportamiento poco deseable llamado FOUC, por el cual puede mostrarse, por un pequeño espacio de tiempo, la página sin los estilos CSS aplicados

Una vez incluida la librería, tendremos disponibles los scripts de detección de funcionalidades además de una serie de clases CSS que nos ayudarán a aplicar estilos solo cuando los navegadores los soporten

6.2. El objeto Modernizr

Una vez tenemos Modernizr cargado, se crea automáticamente un objeto JavaScript que tiene una serie de propiedades (booleanas) que nos indican si están o no disponibles cada una de las funcionalidades presentes en CSS3 y HTML5

```
if (Modernizr.boxshadow) {
  // Podemos aplicar sombras de CSS3
} else {
  // La propiedad box-shadow no está disponible
}

if (Modernizr.canvas) {
  // Podemos utilizar canvas de HTML5
} else {
  // El elemento canvas no está disponible
}
```

Innova4B

6.2. El objeto Modernizr

El listado completo de propiedades del objeto para la detección de funcionalidades HTML5 y CSS3 se puede encontrar en la propia documentación de Modernizr:

http://modernizr.com/docs/

6.3. Clases CSS de Modernizr

Cuando tenemos Modernizr cargado en nuestra página, éste crea automáticamente una serie de clases que asigna al elemento html del documento

Cada una de estas clases hace referencia a las características que soporta el navegador, permitiendo desde CSS adaptar la interfaz según las funcionalidades:

<html lang="en" class=" js flexbox canvas canvastext webgl no-touch geolocation postmessage websqldatabase indexeddb hashchange history draganddrop websockets rgba hsla multiplebgs backgroundsize borderimage borderradius boxshadow textshadow opacity cssanimations csscolumns cssgradients cssreflections csstransforms csstransforms3d csstransitions fontface generatedcontent video audio localstorage sessionstorage webworkers applicationcache svg inlinesvg smil svgclippaths">

....

</html>

Innova4B

6.3. Clases CSS de Modernizr

La creación de cada una de esas clases se realizará únicamente en caso de que el navegador sea compatible, de forma que luego podamos utilizar las clases de manera sencilla.

Por ejemplo, para aplicar sombra a un elemento con CSS3 en los navegadores que lo permitan y emular ese sombreado en los navegadores que no soporten el atributo box-shadow:

```
.elemento{
   border-left: 1px solid #ccc;
   border-top: 1px solid #ccc;
   border-bottom: 1px solid #666;
   border-right: 1px solid #666;
}
```

6.3. Clases CSS de Modernizr

Si nuestro navegador es compatible con el atributo box-shadow de CSS3, Modernizr habrá incluido la clase "boxshadow" en el elemento html

Nosotros podemos utilizar dicha clase CSS para aplicar estilos que sabemos que solo acabarán afectando a los navegadores que soporten el atributo box-shadow

```
.boxshadow .elemento{
   border: 1px solid #ccc;
   box-shadow: #999 3px 3px 3px;
}
```

6.4. Ejercicio práctico

Identificar las siguientes características de al menos 4 navegadores (2 de escritorio y 2 de dispositivos móviles):

- Cuáles de los 12 nuevos tipos de input soporta el navegador
- Qué codecs de reproducción de vídeo soporta cada navegador
- Qué sistema(s) de almacenamiento local soporta cada navegador

Innova4B

6.5. Modernizr.load()

El método Modernizr.load() es una sencilla manera para cargar librerías sólo cuando los usuarios las necesitan, normalmente, cuando una funcionalidad en concreto está (o no) disponible

Es una buena manera de ahorrar ancho de banda y mejorar un poco más el rendimiento de la aplicación

Por ejemplo, podemos saber si el navegador ofrece soporte al API de geolocalizacion (Modernizr.geolocation)

Podemos cargar unos recursos u otros dependiendo de la disponibilidad de esta funcionalidad con el método Modernizr.load(), que permite indicar a los navegadores que no soporten ese API que carguen el polyfill correspondiente

6.5. Modernizr.load()

Un “polyfill” es un trozo de código o un plugin que permite tener las nuevas funcionalidades de HTML5 en aquellos navegadores que nativamente no lo soportan

La sintaxis de Modernizr.load() es bastante sencilla de comprender:

```
Modernizr.load({
  test: Modernizr.geolocation,
  yep : 'geo.js',
  nope: 'geo-polyfill.js'
});
```

Innova4B

6.6. Ejercicio práctico

Identificar si el navegador soporta el atributo placeholder y la etiqueta progress. En caso de no soportar dichas funcionalidades, cargar los “polyfill” correspondiente para añadir dicha funcionalidad al navegador

Probar que está funcionando el ejercicio con navegadores que no soportan placeholder y progress

7. Dataset

HTML5 permite incorporar atributos de datos personalizados, custom data (data-*), en todos los elementos HTML

Prefijar los atributos personalizados con data- asegura que van a ser completamente ignorados por el agente de usuario. Para el navegador y el usuario final no existe esta información

Cada elemento HTML puede tener un número indefinido de atributos de datos personalizados, con cualquier valor

7. Dataset

Hasta ahora, la manera de lograr un comportamiento similar era incluir estos datos como clases CSS:

```
<input class="spaceship shields-5 lives-3 energy-75">
```

Una vez definidos estos valores en class era necesario realizar un trabajo extra para extraer su nombre y su valor (convertir energy-75 en energy = 75)

7. Dataset

Gracias a los atributos dataset esto ya no es necesario, y se crean de la siguiente forma:

Nombre del atributo: debe ser de al menos un carácter de largo y debe tener el prefijo data-. No debe contener letras mayúsculas

Valor del atributo: el valor del atributo puede ser cualquier cadena

```
<ul id="vegetable-seeds">
  <li data-spacing="10cm" data-sowing-time="March to June">Carrots</li>
  <li data-spacing="30cm" data-sowing-time="February to March">Celery</li>
  <li data-spacing="3cm" data-sowing-time="March to September">Radishes</li>
</ul>
```

Innova4B

7. Dataset

Con estos datos almacenados podemos crear una experiencia de usuario más rica y atractiva. Por ejemplo, al hacer clic en un "vegetable", una nueva capa se abre en el explorador que muestra la separación de semillas e instrucciones de siembra

Podemos ahorrar llamadas AJAX al tener los datos directamente en el HTML

7.1. Utilización de los data attributes

Algunos casos en los que PUEDEN ser utilizados:

- Para almacenar la altura inicial o la opacidad de un elemento que pudiera ser necesaria en los cálculos de animación JavaScript posterior
- Para almacenar los parámetros para una película de Flash que se carga a través de JavaScript
- Para almacenar los datos estadísticos de una web
- Para almacenar los datos acerca de la salud, munición o vida de un elemento en un juego JavaScript
- Para poder añadir subtítulos a un <video>
- ...

7.1. Utilización de los data attributes

Algunos casos en los que NO DEBEN ser utilizados:

- Si hay un atributo o elemento existente que es más adecuado: por ejemplo un “input required”
- No están pensados para ser usados públicamente, es decir, el software externo no debe interactuar con ellos (ni siquiera el navegador)
- La presencia/ausencia de un atributo de datos personalizado no se deben utilizar como una referencia para los estilos de CSS, esto se deberían marcar de una manera más accesible

Innova4B

7.1. Data attributes y JavaScript

Si quisiéramos recuperar o actualizar estos “data attributes” podríamos hacerlo a través de JavaScript utilizando los métodos getAttribute y setAttribute

```
<div id="strawberry-plant" data-fruit="12"></div>


<script>
	// "Getting" data-attributes using getAttribute
	var plant = document.getElementById("strawberry-plant");
	var fruitCount = plant.getAttribute("data-fruit"); // fruitCount = "12"

	// "Setting" data-attributes using setAttribute
	plant.setAttribute("data-fruit","7"); // Pesky birds
</script>
```

Innova4B

7.1. Data attributes y JavaScript

Este método funcionará en los navegadores modernos, pero no es la manera en la que los data attributes deben ser utilizados

Para lograr lo mismo debe accederse a la propiedad dataset de un elemento

```
<div id="sunflower" data-leaves="47" data-plant-height="2.4m"></div>

<script>
    var plant = document.getElementById("sunflower");
    var leaves = plant.dataset.leaves; // leaves = 47;

    var tallness = plant.dataset.plantHeight;
    plant.dataset.plantHeight = "3.6m";
</script>
```

Innova4B

7.1. Data attributes y JavaScript

Si en algún momento un atributo data- específico ya no es necesario, es posible eliminarlo por completo del elemento DOM

```
plant.dataset.leaves = null;
```

Los data attributes personalizados son una buena manera de simplificar el almacenamiento de datos de la aplicación en las páginas web

7.2. Data attributes y jQuery

jQuery permite acceder a los data attributes de una forma más eficiente en memoria que el método anterior (eso dicen), a través de la función “data()”

```
<section id="content" data-role="page" data-page-num="42">
    <!-- Imagine a bunch of page-type content here... -->
</section>

<script type="text/javascript">
    console.log($('content').data('role')); // Expect string "page"
    console.log($('content').data('pageNum'));    // Expect 42, an integer...!
    $('#content').data('newAttribute',34); // New attributte is hidden
</script>
```

Cómo utilizar data(): http://api.jquery.com/data/

Innova4B

7.3. Ejercicio práctico

A partir del siguiente HTML, realizar los siguiente JavaScript y jQuery:

```
<ul>
   <li class="user" data-name="Arkaitz Garro" data-city="Donostia"
      data-lang="es" data-food="Txuleta">Arkaitz Garro</li>
   <li class="user" data-name="John Doe" data-city="Boston"
      data-lang="en" data-food="Bacon">John Doe</li>
   <li class="user" data-name="Divya Resig" data-city="Tokyo"
      data-lang="jp" data-food="Sushi" data-delete="true">Divya Resig</li>
</ul>
```

- Obtener cada uno de los atributos data- de los elementos de la lista, y mostrarlos por consola.
- Modificar el idioma es por es_ES.
- Eliminar los elementos de la lista cuyo atributo data-delete sea true

Innova4B

8. Multimedia

Hasta hace no mucho tiempo, la tecnología Flash era el dominador indiscutible en el campo multimedia de la web

Gracias a esta tecnología es relativamente sencillo transmitir audio y vídeo a través de la red, y realizar animaciones que de otra manera sería imposible

Incluso se diseñaban páginas completamente en Flash

Prácticamente todos los navegadores tienen incorporado un plugin para la reproducción de archivos flash

Entonces, ¿por qué una necesidad de cambio?

8. Multimedia

Para incluir un elemento multimedia en un documento, se hacía uso del elemento <object>

Debido a la incompatibilidad entre navegadores, se hacía también necesario el uso del elemento <embed> y duplicar una serie de parámetros:

```
<object width="425" height="344">
   <param name="movie"
      value="http://www.youtube.com/v/9sEI1AUFJKw&hl=en_GB&fs=1&"></param>
   <param name="allowFullScreen" value="true"></param>
   <param name="allowscriptaccess" value="always"></param>
   <embed src="http://www.youtube.com/v/9sEI1AUFJKw&hl=en_GB&fs=1&"
      type="application/x-shockwave-flash" allowscriptaccess="always"
      allowfullscreen="true" width="425" height="344"></embed>
</object>
```

Innova4B

8. Multimedia

Tenemos varios inconvenientes:

- El navegador tiene que transmitir el vídeo a un plugin instalado en el navegador, y el usuario deberá tener instalada la versión correcta: ¿puede instalar? ¿tiene permisos para hacerlo? Deberá instalar el plugin antes de ver el vídeo... :(
- Los plugins pueden causar que el navegador o el sistema se comporte de manera inestable, algunos navegadores desactivan los plugins desactualizados (por inseguros) y usuarios sin conocimientos técnicos pueden percibir esa inseguridad... :(
- Y si mi dispositivo no dispone de plugin para ese contenido... :(

Innova4B

8.1. Video

Una de las mayores ventajas de HTML5 son los elementos multimedia <video> y <audio>

Están totalmente integrados en la web, no es necesario depender de software de terceros

Además, el elemento <video> puede personalizarse a través de estilos CSS: se puede cambiar su tamaño, animarlo con transiciones CSS...

Podemos manipularlos con total libertad, ya que pertenecen al estándar. Ya no se ejecutan en una caja negra, tenemos disponible un API:

http://www.w3schools.com/tags/ref_av_dom.asp

Innova4B

8.1. Video

<video></video>

Para hacer funcionar el vídeo en HTML, es suficiente con incluir la etiqueta:

```
<video src="movie.webm"> </video>
```

Sin embargo, lo único que se muestra es el primer fotograma de la película

No hemos dicho al navegador que inicie el vídeo, ni le hemos mostrado al usuario ningún tipo de control para reproducir o pausar el vídeo

Innova4B

8.1. Video

autoplay

Podemos indicar al navegador que reproduzca el vídeo de manera automática una vez se haya cargado la página

```
<video src="movie.webm" autoplay>
    <!-- Your fallback content here -->
</video>
```

No es una buena práctica, a muchos usuarios les parecerá una práctica muy intrusiva, sobre todo los usuarios de dispositivos móviles

Innova4B

8.1. Video

controls

Proporcionar controles es mejor que reproducir el vídeo de manera automática en la mayoría de los casos

```
<video src="movie.webm" controls>
<!-- Your fallback content here -->
</video>
```

Los navegadores muestran la interfaz de los controles de manera diferente, ya que la especificación no indica qué aspecto deben tener

Pero todos ellos muestran: reproducir/pausa, una barra de progreso y un control de volumen

Innova4B

8.1. Video

poster

Indica la imagen que el navegador debe mostrar mientras el vídeo se está descargando, o hasta que el usuario reproduce el vídeo

Si no se indica este atributo, el navegador muestra el primer fotograma del vídeo, que puede no ser representativo del vídeo que se va a reproducir

8.1. Video

muted

Permite que el elemento multimedia se reproduzca inicialmente sin sonido, lo que requiere una acción por parte del usuario para recuperar el volumen

loop

Indica que el vídeo se reproduce de nuevo una vez que ha finalizado su reproducción

8.1. Video

dimensiones

Los atributos height y width indican al navegador el tamaño del vídeo en pixels

Si no se indican estas medidas, el navegador utiliza las medidas definidas en el vídeo de origen, si están disponibles

Si no lo están, utiliza las medidas definidas en el fotograma poster, si están disponibles

Si ninguna de estas medidas está disponible, el ancho por defecto es de 300 pixels

8.1. Video

dimensiones

Si se especifica una de las dos medidas, el navegador ajusta la medida de la dimensión no proporcionada, conservando la proporción del vídeo

Si se especifican las dos medidas, pero no coinciden con la proporción del vídeo original, el vídeo no se deforma a estas nuevas dimensiones

8.1. Video

preload

Es posible indicar al navegador que comience la descarga del vídeo antes de que el usuario inicie su reproducción.

```
<video src="movie.webm" controls preload>
   <!-- Your fallback content here -->
</video>
```

Existen tres valores definidos para preload, pero preload es un consejo, no un comando

Innova4B

8.1. Video

preload

El navegador tomará una decisión en función del dispositivo, las condiciones de la red y otros factores:

- preload=auto: se sugiere al navegador que comience la descarga
- preload=none: se sugiere al navegador que no comience la descarga hasta que lo indique el usuario
- preload=metadata: se sugiere al navegador que cargue los metadatos (dimensiones, fotogramas, duración...), pero nada más

Si no indicamos uno en concreto, es el propio navegador el que decide qué hacer

Innova4B

8.1. Video

src

El atributo src indica la localización del recurso, que el navegador debe reproducir si el navegador soporta el codec o formato específico

Utilizar un único atributo src es únicamente útil y viable en entornos totalmente controlados: disponibilidad asegurada y codec controlado

Sin embargo, como no todos los navegadores pueden reproducir los mismos formatos, en entornos de producción debemos especificar más de una fuente de vídeo

8.2. Y ahora los codecs...

En los primeros borradores de HTML5, se indicaba que se debía de ofrecer soporte para al menos dos codecs multimedia: Ogg Vorbis para audio y Ogg Theora para vídeo

Sin embargo, estos requisitos fueron eliminados después de que Apple y Nokia se opusieran, de modo que la especificación no recomendase ningún codec en concreto

Esto ha creado una situación de fragmentación, con diferentes navegadores optando por diferentes formatos, basándose en sus ideologías o convicciones comerciales

Innova4B

8.2. Y ahora los codecs...

Actualmente, hay dos codecs principales: el nuevo formato WebM, construido sobre el formato VP8 que Google compró y ofrece de manera libre, y el formato MP4, que contiene el codec propietario H.264

	WEBM	MP4	OGV
Opera	Sí	No	Sí
Firefox	Sí	Sí	Sí
Chrome	Sí	No	Sí
IE9+	No	Sí	No
Safari	No	Sí	No

WebM funciona en IE9+ y Safari sólo si el usuario ha instalado los codec de manera manual.

La mejor solución en estos momentos es ofrecer tanto el formato libre WebM, como el propietario H.264

Innova4B

8.2. Y ahora los codecs...

<source>

Para poder ofrecer ambos formatos, primeramente debemos codificarlos por separado

Existen diversas herramientas y servicios on-line para realizar esta tarea

Uno de los software más conocidos sea Miro Video Converter que permite convertir los vídeos a muchos formatos y optimizar para diferentes tipos de dispositivos como iPhone, Android, PS2, etc.

Innova4B

8.2. Y ahora los codecs...

<source>

Después, es necesario indicar todas las localizaciones de estos formatos, para que sea el navegador el que decida que formato reproducir

Para ello, no podemos especificarlos todos dentro del atributo scr, por lo que tendremos que hacerlo de manera separada

```
<video controls>
   <source src="leverage-a-synergy.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
   <source src="leverage-a-synergy.webm" type='video/webm; codecs="vp8, vorbis"'>
   <p>Your browser doesn’t support video.
      Please download the video in <a href="leverage-a-synergy.webm">webM</a>
      or <a href="leverage-a-synergy.mp4">MP4</a> format. </p>
</video>
```

Innova4B

8.2. Y ahora los codecs...

media queries

Los ficheros de vídeo tienden a ser pesados, y enviar un vídeo en alta calidad a un dispositivo con un tamaño de pantalla reducido es algo totalmente ineficiente

HTML5 permite utilizar el atributo media en el elemento <source>, ofreciendo la misma funcionalidad que los Media Queries en CSS3. Por lo tanto, podemos consultar al navegador por el ancho de la pantalla, la relación de aspecto, colores, etc

```
<video controls>
    <source src="hi-res.mp4" media="(min-device-width: 800px)">
    <source src="lo-res.mp4">
</video>
```

Innova4B

8.3. El API multimedia

Los elementos multimedia <video> y <audio> ofrecen un API muy completo y fácil de utilizar

Los eventos y métodos de los elementos de audio y vídeo son exactamente los mismos, su única diferencia se da en los atributos

Atributos	Métodos	Eventos
error state	load()	loadstart
error	canPlayType(type)	progress
network state	play()	suspend
src	pause()	abort
currentSrc	addTrack(label, kind, language)	error
networkState		emptied
preload		stalled
buffered		play
ready state		pause

Innova4B

8.3. El API multimedia

Atributos	Métodos	Eventos
readyState		loadedmetadata
seeking		loadeddata
controls		waiting
controls		playing
volume		canplay
muted		canplaythrough
tracks		seeking
tracks		seeked
playback state		timeupdate
currentTime		ended
startTime		ratechange

Innova4B

8.3. El API multimedia

Atributos	Métodos	Eventos
muted		
paused		
defaultPlaybackRate		
playbackRate		
played		
seekable		
ended		
autoplay		
loop		
width [video only]		
height [video only]		
videoWidth [video only]		
videoHeight [video only]		
poster [video only]		

Innova4B

8.3. El API multimedia

Con este nuevo API, tenemos el control completo sobre los elementos multimedia

http://www.w3schools.com/tags/ref_av_dom.asp

Por ejemplo:

```
video.addEventListener('canplay', function(e) {
    this.volume = 0.4;
    this.currentTime = 10;
    this.play();
}, false);
```

Innova4B

8.4. Video Fullscreen

HTML5 establece un único elemento full-screen, que está pensado para imágenes, vídeo y juegos que utilizan el elemento canvas

Una vez que un elemento pasa a pantalla completa, aparece un mensaje de forma temporal para informar al usuario, así no genera confusión

Las principales propiedades, métodos y estilos son:

- element.requestFullScreen(): hace que un elemento individual pase a pantalla completa: document.getElementById("myvideo").requestFullScreen()
- document.cancelFullScreen(): sale del modo pantalla completa y vuelve a la vista del documento
- document.fullScreen: devuelve true si el navegador está en pantalla completa
- :full-screen: se trata de una pseudo-clase CSS que se aplica a un elemento cuando está en modo pantalla completa.

8.4. Video Fullscreen

Además, podemos modificar los estilos del elemento utilizando CSS:

```
#myelement {
   width: 500px;
}
#myelement:full-screen {
   width: 100%;
}
#myelement:full-screen img {
   width: 100%;
}
```

Innova4B

8.5. Audio

El elemento multimedia <audio> es muy similar en cuanto a funcionalidad al elemento video, de hecho el API es el mismo

La principal diferencia existe al indicar el atributo controls:

- Si lo especificamos, el elemento se mostrará en la página juntamente con los controles
- Si no lo hacemos, el audio se reproducirá, pero no existirá ningún elemento visual en el documento. El elemento existirá en el DOM y tendremos acceso completo a su API desde JavaScript

Para hacer funcionar el audio en HTML:

```
<audio src="audio.mp3">
</audio>
```

Innova4B

8.5. Audio

Pero con audio seguimos con los problemas de codecs...

	MP3	MP4	WAV	OGG
Opera	No	No	Sí	Sí
Firefox	No	No	Sí	Sí
Chrome	Sí	Sí	Sí	Sí
IE9+	Sí	Sí	No	No
Safari	Sí	Sí	Sí	No

La mejor solución en estos momentos es ofrecer tanto el formato libre OGG, como el propietario MP3:

```
<audio controls>
    <source src="audio.ogg" type="audio/ogg">
    <source src="audio.mp3" type="audio/mpeg">
</audio>
```

Innova4B

8.6. Ejercicio práctico

Crear un reproductor de vídeo que cumpla las siguientes características:

- Reproducir los vídeos independientemente del codec soportado por el navegador.
- Incluir controles de reproducción, pausa, parar, avanzar y retroceder 10 segundos, inicio y fin.
- Control de volumen y paso a pantalla completa.
- Un indicador de progreso de la reproducción.

Añadir una lista de reproducción que permita seleccionar un nuevo vídeo, y éste se reproduzca sin recargar la página.

http://www.arkaitzgarro.com/html5/capitulo-18.html#ej06

Innova4B