

[HTML Multimedia \(w3schools.com\)](https://www.w3schools.com/html/html5_multimedia.asp)

[Media type and format guide: image, audio, and video content - Web media technologies | MDN \(mozilla.org\)](#)

Audio element

Audio and video are new HTML 5 elements that were highly anticipated. With HTML5 support for multimedia, this has become much easier, than previous methods.

Audio tag

You can use the <audio> tag to embed audio in your page.

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

Your browser does not support the audio file.

```
</audio>
```

Any text within the <audio> tags will be displayed if the browser does not support the audio element. **You should add such a message to provide better user experience** for your page as it will be viewed in all types of devices and browsers.

The audio element has several attributes that can be used to configure audio playback. The following table lists the audio element's attributes:

Examples:

```
<!--cargamos un audio en formato MP3, pero indicamos que no
precargue nada. Empezará a descargarse sólo cuando el usuario pulse en lo
s controles de reproducción. Este escenario puede ser interesante para ev
itar consumo de ancho de banda de archivos que es probable que el usuario
no escuche o en dispositivos móviles donde las tarifas de datos son cost
osas.-->
```

```
<audio src="audio/hindi.mp3"preload="none" controls>
```

Your browser does not support the audio file.

```
</audio>
```

```
carga un archivo de audio en formato mp3 y lo reproduce automáticamente,
en bucle, de modo que vuelve a empezar cuando termina. -->
```


```
<audio src="audio/hindi.mp3"autoplay loop>
```

Your browser does not support the audio file.

```
</audio>
```

```
importante: Los navegadores han cambiado la política de autoreproducción
con autoplay. Para evitar el uso abusivo de audio en una página sin permi
so del usuario, los navegadores exigen que el usuario haya interactuado c
on la página con anterioridad-->
```

```
Lo aconsejable sería utilizar botones o areas pulsables para activar el s
onido mediante JS.-->
```

Attribute	Description	Usage
src	<p>Used to specify the URL of the audio file to embed.</p> <p>Values:</p> <ul style="list-style-type: none"> absolute URL (file residing somewhere on the Web) relative URL (within your Web site) 	<pre><audio src="sounds/flute.mp3"></audio></pre>
controls	<p>Boolean attribute when specified provides controls for the user like play, pause, seek bar and volume</p> 	<pre><audio src="sounds/flute.mp3" controls></audio></pre>
loop	<p>Boolean attribute when specified loops media content</p>	<pre><audio src="sounds/flute.mp3" controls loop></audio></pre>

-muted	Boolean attribute when specified mutes media when playback begins	<audio src="sounds/flute.mp3" controls muted></audio>
preload	Allows author to communicate to the browser which settings will work best - audio should not be preloaded (none), only audio metadata is fetched (metadata), audio file can be downloaded when page loads (auto) values: none, metadata, auto	<audio src="sounds/flute.mp3" controls preload="auto"></audio>
autoplay	Boolean attribute when specified will automatically begin playing the source file as soon as it can without waiting for the entire audio file to finish downloading	<audio src="sounds/flute.mp3" controls autoplay></audio>

Example

```
<audio src="http://audio.ibeat.org/content/p1rj1s/p1rj1s_-_rockGuitar.mp3" controls loop muted preload="none">
  Your browser does not support the audio file.
</audio>
```

Audio file format

Just like image file formats, not all audio file formats are supported by all browsers. You will want to use common audio file formats for browser compatibility ensuring the highest probability that your audio file will play.

The most common ones are MP3, WAV (microsoft) and Ogg (free alternative to mp3).

[This page](#) under 'Browser Compatibility' lists the audio formats supported by the audio element and their browser support.

Here is some information regarding different types of audio formats and their compression techniques that can help you decide which audio format to choose apart from audio element and browser compatibility:

- There are three major groups of audio file formats - uncompressed (eg: WAV), lossless compressed (eg: MPEG-4, WMA Lossless) and lossy compressed (eg: Opus, MPC, AAC, WMA Lossy).
- In uncompressed audio file formats, no compression is applied to the audio file. The memory used for both sound and silence is the same though silence contains less information/data.
- In lossless compression, no data is lost but the file is compressed as silence is designed to take up very little space. Compared to uncompressed, lossless compression's compression ratio is approximately 2:1.
- Lossy compression provides the greatest compression by simplifying the data and removing some audio information resulting in some loss of quality. It is also the most popular. There are techniques in place to ensure that the parts of sound that is lost has little effect on quality. You can also select a range of compression rates. The larger the rate of compression, the bigger the loss in quality and smaller the file size.
- The .ogg files are those that use the Ogg Vorbis compressed audio format, a free and open container that is developed and maintained by the Xiph.Org Foundation. Opus is a lossy audio encoding format. It is considered as an alternative to mp3. OGG compression is lower than MP3, which means that the audio quality will be better.
- If you have an audio file in one format and wish to convert it to another, there are a lot of software applications available to help you do that.

Source element for multiple source files

The source element, also new in HTML5, serves the same purpose as the src attribute in an audio element. It is used to specify source files for the audio and video elements. Using the source element, you can specify multiple source files. The <source> tag is self-closing, therefore, it does not require a closing tag.

<audio controls>

<source src="./audio/splash.wav" type="audio/wav">

<source src="./audio/splash.mp3" type="audio/mpeg">

Your browser does not support the audio element.

</audio>

The advantage of providing multiple source files in different formats is that if the browser doesn't support the first format, it will try the second source file. The browser can select from the list based on its file format or codec support. In the code snippet example above, Internet Explorer does not support .wav files. So if you tried to play the file above in Internet Explorer, the browser would have tried to play .wav, failed and played the .mp3 version instead.

The following table lists the <source> element's attributes:

Attribute	Description	Usage
src	Specifies the URL of the media file	<source src="sounds/flute.mp3">
type	Specifies the internet media type, also known as the MIME type for the audio resource. .	<source src="sounds/flute.mp3" type="audio/mpeg">

MIME:

A media type is an identifier for file formats and format contents transmitted over the internet like text and audio files.

It consists of a type and a sub-type. Eg: "audio/mpeg" - **audio** is the type and **mpeg** is the subtype. It can also take optional parameters that can be specified after a semicolon - "audio/ogg; codecs=opus" means the audio is in the ogg format and uses the opus

codec. If the browser supports the Ogg format but not the Opus codec, the audio file will not load.

If the type attribute is not specified, the media type is retrieved from the server.

Video element.

Video is fairly new in HTML5. Previously, the most reliable way to add video was the Adobe Flash Player for example.

You can use the video element to embed video in your page. You can specify the location of your video file using the src attribute or source element (for multiple source files).


```
<video src="multimedia/running.mp4">
```

Your browser does not support the HTML5 video element.

```
</video>
```

Any text within the <video> tags will be displayed if the browser does not support the video element. You should add such a message to provide better user experience for your page as it will be viewed in all types of devices and browsers.

Similar to the audio element, the video element has several attributes that can be used to configure playback. The following table lists the video element's attributes:

Attribute	Description	Usage
src	specifies the URL or location of the media file	<code><video src="multimedia/running.mp4"></video></code>
autoplay	Boolean attribute when specified will automatically begin playing source file as soon as it can without waiting for the entire video file to finish downloading. Note: Some versions of chrome support autostart instead of autoplay	<code><video src="multimedia/running.mp4" autoplay></video></code>
controls	Boolean attribute when specified provides controls for the user like play, pause, seek bar and volume 	<code><video src="multimedia/running.mp4" controls></video></code>

loop	Boolean attribute when specified loops media content	<video src="multimedia/running.mp4" controls loop></video>
muted	Boolean attribute when specified mutes media when playback begins	<video src="multimedia/running.mp4" controls muted></video>
preload	Allows author to communicate to the browser which settings will work best - video should not be preloaded (none), only video metadata is fetched (metadata), video file can be downloaded (auto) values: none, metadata, auto	<video> src="multimedia/running.mp4" controls preload="auto"></video>
poster	Specifies the URL of the frame you want to display as the video cover until the user starts or seeks the video. By default, the first frame is considered the poster frame. The poster can also be an arbitrary image, not necessarily in any frame of the video.	<video src="multimedia/running.mp4" poster="/images/video-screenshot.png" controls>

height, width	height and width of the video's play area in pixels. Always set height and width for a video so the browser can allocate the specified space for it when it loads the page.	<code><video src="multimedia/running.mp4" controls width="320" height="240"></video></code>
---------------	---	---

Example:

```
<!--esto mostrará el primer fotograma del video-->
  <video src="video/artesano-1920x1080.mp4" class="video">
    Your browser does not support the HTML5 video element.
  </video>
<!--se carga un vídeo, se muestra una imagen de presentación
el vídeo no se reproduce hasta que el usuario pulse el botón de
reproducir vídeo-->
<video src="video/artesano-1920x1080.mp4" poster="images/artesano.png" controls class="video">

</video>

<!--se reproduce un vídeo automáticamente al cargar la página, indefinido y en silencio-->
<video src="video/artesano-1920x1080.mp4" autoplay muted loop class="video">

</video>
```

Poster attribute

The <video> tag has an important attribute that you don't find on the audio element. The poster attribute is used to specify what picture is shown before the video starts playing. By default, the poster shown is simply the first frame of the video, but the poster attribute can be used to specify a different image. It can specify a particular frame of the video or, like a real movie poster, it can be an image that doesn't actually appear in the video.

Video file formats

We must know that a video file has two main parts: the **container format**, which is the format of the video itself, while inside **it can have multiple components** encoded with different **codecs**.

In other words, a basic video has at least one video component and one audio component, but it can have many more (subtitles, images, etc ...). These details are very important, since depending on the format and / or codec of a video, it may or may not be possible to use it for the web.



Just like audio file formats, not all video file formats are supported by all browsers. You should use common video file formats for browser compatibility ensuring the highest probability that your video file will play.

The most common ones are MP4, WebM and Ogg.

[This page](#) under 'Browser Compatibility' lists the video formats supported by the video element and their browser support for both desktop and mobile. Using the source element, you will have to provide a combination of video formats to target most browsers.

Here is some information regarding different types of video formats and their compression techniques that can help you decide which video format to choose apart from video element and browser compatibility:

- Most videos go through some form of compression to reduce redundancy in video files and make them smaller, allowing them to download faster. Most also use audio compression techniques to compress sound in video files.
- Like audio, there are three major groups of video file formats - uncompressed, lossless compressed ([list of lossless video compression formats](#)) and lossy compressed ([list of lossy video compression formats](#)).
- In uncompressed video file formats, no compression is applied to the video file.

- In lossless compression, no data is lost. If you were to uncompress a file compressed using the lossless technique, you will get back the exact same data you started with.
- Most videos use lossy compression as it results in significantly smaller video files. Lossy compression provides compression by simplifying the data and removing video information resulting in some loss of quality. There are techniques in place to ensure that the parts of the video that are lost have little effect on quality. You can also select a range of compression rates. The larger the rate of compression, the bigger the loss in quality and smaller the file size. However, if you uncompress a video file that was compressed using the lossy technique, you will not be able to retrieve the same data you put in. With text or spreadsheets, loss of data might be a significant problem. However, with images and video losing a bit of quality is not going to affect the file because you can still make out what the video is about.
- The video format 'H.264 and MP3 in MP4' has three parts to it. H.264 is a video compression standard. MP3 is an audio coding format that uses lossy compression for sound in the video. MP4, like Ogg, is a digital container format. It stores audio and video data rather than code the information. A program that opens a container file like MP4 might not know how to decode it. So it requires other standards like H.264 and MP3 to dictate how the video will be coded and possibly compressed.
- If you have a video file in one format and wish to convert it to another, there are a lot of software applications available to help you do that.

Source element for multiple source files

The source element that we saw in the previous unit is also used to specify multiple source files for the video element. The <source> tag is self-closing and so does not require a closing tag.

```
<!--Video como etiqueta contenedora
      videos alternativos -->

<video class="video" autoplay loop>
  <source src="video/artesano-1920x1080.mp4" type="video/mp4" />
  <source src="video/artesano-1920x1080.webm" type="video/webm" />
  
  Su navegador no soporta contenido multimedia.
</video>
<!--
```

Intenta mostrar el primer formato (MP4). Si el navegador no soporta este formato, salta al siguiente.

Intenta mostrar el segundo formato (WEBM). Si el navegador no soporta este formato, salta al siguiente.
 Si se trata de un navegador que no soporta HTML5, intentará mostrar la imagen.
 Si se trata de un navegador de terminal de texto (o sin capacidades gráficas),
 mostrará el texto "Su navegador no soporta contenido multimedia".
 -->

The advantage of providing multiple source files in different formats is that if the browser doesn't support the first format, it will try the second source file. The browser can select from the list based on its file format or codec support. There is no one format that is supported by all browsers. So you will have to use the source element to list a combination of formats.

The following table lists the source element's attributes:

Attribute	Description	Usage
src	Specifies the URL or location of the media file	<code><source src="multimedia/small.mp4"></source></code>
type	Specifies the internet media type, If the type attribute is not specified, the media type is retrieved from the server.	<code><source src="multimedia/small.mp4" type="video/mp4"></source></code>

Type: It consists of a type and a sub-type. Eg: "video/mp4" - video is the type and mp4 is the subtype. It can also take optional parameters that can be specified after a semicolon - "video/mp4; codecs="avc1.42E01E, mp4a.40.2"" means the video is in the mp4 format and uses the codecs - avc1.42E01E, mp4a.40.2. If the browser supports the mp4 format but none of the avc1.42E01E, mp4a.40.2 codecs, the video file will not load.

Track element for caption and subtitles

The <video> element is very similar to the HTML5 <audio> element except for one addition - the <track> element. The <track> element is used to add timed text like subtitles, captions or any text you would like to display to the user when the video is playing.

- Web Video Text Tracks (WebVTT) files are the standard to include subtitles or captions. You can learn [how to create them here](#).
- Captions and subtitles are not the same. Subtitles are meant to translate the language (for those who do not understand the language being spoken in the video). Captions are meant for the deaf or people who have difficulty hearing. It includes sound effects and other significant audio like music and lyrics and is usually in the same language as the audio.
- Like the <source> tag, you can add multiple <track> tags in your video element to add multiple subtitle/caption tracks. This is commonly done when providing them in different languages.

The <track> tag is self-closing and so does not require a closing tag. You specify the <track> element as a child element of your <video> tag like this:

```
<video width="320" height="240" controls>
  <source src="module.mp4" type="video/mp4">
  <track src="module-captions.vtt" kind="captions" srclang="en" label="English" default>
  Your browser does not support the HTML5 video element.
</video>
```

The following table lists the <track> element's attributes:

Attribute	Description	Usage
default	It is a boolean attribute. If you have multiple tracks for the same video file, you can specify which one is the default using this attribute. It can be used on one track element in a video. If you only have one track element, default should still be added to deliver the video with	<pre><video src="multimedia/small.mp4" controls> <track src="captions/small- en.vtt" label="english" default></pre>

	captions turned on in most browsers.	<pre><track src="captions/small-fr.vtt" label="French"> </video></pre>
kind	Specifies the kind of the source file. Values: subtitles (default value), captions, descriptions (textual description of the video best suited for the blind who cannot be seen), chapters (meant for chapter titles), metadata (kind of track that is used by scripts and is not visible to the user).	<pre><track src="captions/small-en.vtt" kind="captions"></pre>
label	Label of the track. Browser uses the label value to display track options for user to select.	<pre><track src="captions/small-en.vtt" label="English"> <track src="captions/small-fr.vtt" label="French"></pre>
src	URL of track. The file must be on a Web server . The .vtt file cannot be loaded from a file (file://) protocol.	<pre><track src="http://www.xyz.org/small-en.vtt"></pre>
srclang	Language of text track. Eg: en, fr. If kind is 'subtitles', then the srclang attribute must be specified.	<pre><track src="captions/small-en.vtt" kind="subtitles" srclang="en"></pre>

Access to specific seconds

Using the [multimedia fragments](#) you can achieve some interesting actions, such as specifying the start or end time of the video (or audio)

```
<!--se reproduce un vídeo automáticamente a partir del segundo 15-->
  <video src="video/artesano-
1920x1080.mp4#t=15" autoplay class="video" >

  </video>
<!--
se reproduce un vídeo automáticamente a partir del segundo 5 y termina en
el 15-->
  <video src="video/artesano-
1920x1080.mp4#t=5,15" autoplay class="video" >
  </video>
```

Ejemplo vídeo fondo

<https://css-tricks.com/full-page-background-video-styles/>

[How To Create a Fullscreen Video Background \(w3schools.com\)](#)

Sobre vídeo

<https://pixabay.com/es/videos/>

<http://techslides.com/?s=video>

The iframes tag

There are tags for all kinds of content in your Web page, text, images, videos, animations. There's even a tag that allows you to put another Web page in your Web page - the `<iframe>` tag (HTML Inline Frame Element).

Iframes are generally used in Web pages to show external content/resources. The type of content is not limited to other Web pages. You can add YouTube videos or display a PDF file (some browsers will display the file inline while some older browsers will try to download it instead).

```
<p>This is a parent page that will host the iframe.</p>
  <iframe src="https://www.w3.org/">
    <p>Your browser does not support iframes.</p>
  </iframe>
```

Because iframes are HTML elements, they can be styled just like other elements, with borders, margins, sizes specified with CSS rules.

Example:

Here, we've embedded a YouTube video with an iframe like this:

```
<iframe src="https://www.youtube.com/embed/WjoDEQqyTig "></iframe>
```

And we've added styling like this to get the border and drop-shadow:

```
iframe {
  border: 10px solid red;
  padding: .5rem;
  margin: 1rem;
  box-shadow: 20px 20px 10px #888888;
  width: 355;
  height: 200;
}
```

There is one significant problem with iframes. Suppose you create your Web page, containing only an iframe with `src="http://foo.com"`, with no borders, padding or margin. By all appearances, you would seem to be on the Web site `foo.com`. If you don't look at the URL, it might be difficult to tell. For reasons like this, some Web sites disallow their inclusion, so if you create an iframe with `src="https://google.com"`, you'll get a blank frame and an error message in the console. This isn't a bug, it's a feature. There are a number of important attributes for an `<iframe>` tag, but for now we'll just look at a few of them:

Description	Value	Example
Specifies the address of the page you want to display in your frame.	URL	<code><iframe src="https://www.w3.org/"></iframe></code>
This will allow the iframe to open "Full screen mode", often used with videos. Without this attribute, full screen mode is disabled for the iframe.	text, HTML code, etc	<code><iframe src="https://www.w3.org/" allowfullscreen></iframe></code>
Specifies a name for the iframe. Using the name attribute, the iframe can act as a target for a link. Just as the 'self' target will replace the current window with the site at the href URL, and "_blank" will open a new window at that URL, if you set the name attribute, that name can be used as a target so that when you click on it, the new page will open up in that iframe.	text	<code><iframe name="frame-one" src="https://www.w3.org/"></iframe></code> <code></code>
This can apply a number of restrictions on the iframe, preventing the site in the iframe from using pop-ups, running scripts, automatically running videos and numerous other things. This helps avoid some of the potential security issues that iframes may be prone to.	no value (applies all restrictions) allow-forms allow-modals allow-orientation-lock allow-pointer-lock allow-popups allow-same-origin allow-scripts allow-top-navigation	<code><iframe src="https://www.w3.org/" sandbox></iframe></code> OR <code><iframe src="https://www.w3.org/" sandbox="allow-popups"></iframe></code>
While width and height are valid attributes for an iframe, they should be avoided in favor of CSS properties.	pixels	<code><iframe src="https://www.w3.org/" width="500"></iframe></code>

Notes:

Certain Web sites like Google and Yahoo disallow embedding their Web pages in iframes. So you will not be able to use these pages in an iframe.

Not all attributes are supported in all browsers. You are encouraged to explore their browser support before adding to your HTML.

You can find more details about iframes from the [W3C Specification](#).

Iframes can be very useful:

Iframes load separately from the main page. However, they do block the main page's load command until its content finishes loading. You can avoid this by applying some Javascript. This allows them to load independently. Then, if the embedded page you are displaying loads slower, you can use your parent page to keep the reader occupied.

Sandboxing provides security.

Great for third party content like ads.

It is convenient to use if you need to have one part of your page static while the other is changed

However, there can be some disadvantages:

It is easy to misuse them. It should be considered a piece of content in the webpage and not as an integral part of it.

Accessibility of iframes is poor. Screenreaders do not process them well but you can proceed to use iframes with a notice for the reader.

You have no control over the content in an iframe if you display external content. That content can change anytime or can upload malicious content without your permission.

Search engines have trouble accessing and in turn indexing the content in your iframes. This doesn't help your search ranking.