Аудио и видео в интернете

Веб-разработчики хотели использовать видео и аудио в Интернете в течение длительного времени, начиная с начала 2000-х годов, когда пропускная способность сети стала достаточной, чтобы поддерживать любое видео (видеофайлы намного больше, чем текст, или даже изображения). На раннем этапе базовые веб-технологии, такие как HTML, не имели возможности вставлять видео и аудио в Интернет, поэтому запатентованные технологии (или плагины), такие как Flash (а затем и Silverlight), стали популярными для обработки такого контента. Такая технология работала нормально, но у нее было много недостатков, включавших плохую работу с функционалом HTML и CSS, проблемы безопасности и проблемы доступности.

Самостоятельное решение решило бы большую часть этого, если сделано правильно. К счастью, несколько лет спустя в спецификации HTML5 были добавлены такие функции, с элементами <video> и<audio>, и некоторые новые JavaScript API для их управления. Мы не будем рассматривать JavaScript здесь - только необходимые основы, которые могут быть достигнуты с помощью HTML.

Элемент <video>

Элемент <video> позволяет вам вставлять видео достаточно легко. Очень простой пример выглядит так:

```
<video src="rabbit320.webm" controls>
  <p>Ваш браузер не поддерживает HTML5 видео. Используйте <a
href="rabbit320.webm">ссылку на видео</a> для доступа.</p>
</video>
```

Описание параметров:

src

Точно так же, как для элемента <img>, атрибут src (source — источник) содержит путь к видео, которое вы хотите внедрить. Он работает точно так же.

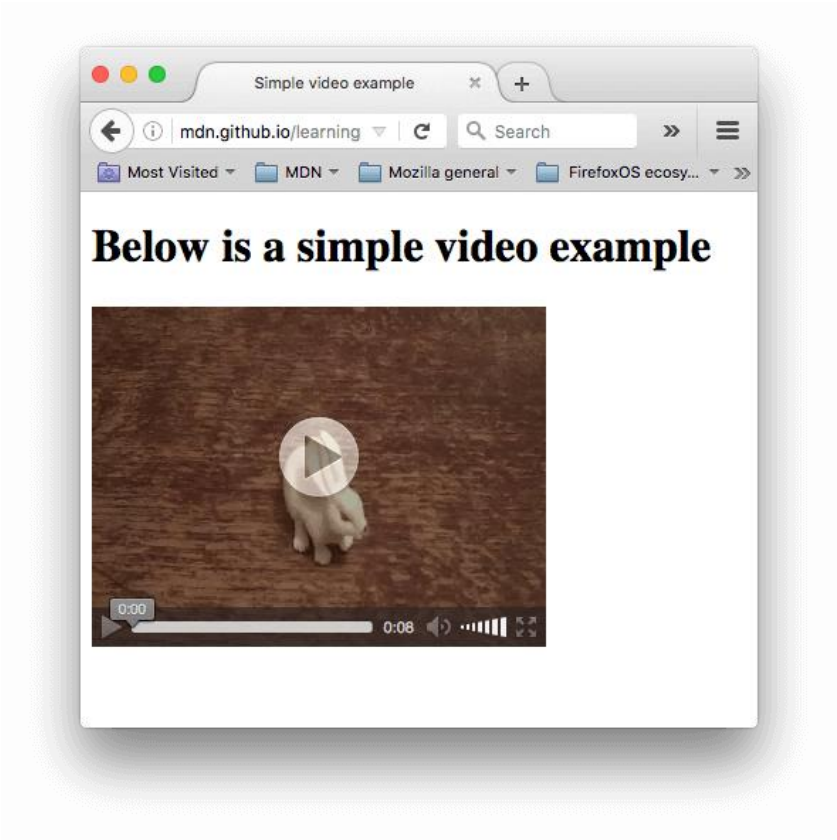
controls

Пользователи должны иметь возможность контролировать воспроизведение видео и аудио (особенно это важно для людей, которые больны эпилепсией). Вы должны либо использовать атрибут controls, чтобы использовать встроенный в браузер интерфейс управления или создать собственный интерфейс, используя соответствующие JavaScript API. Как минимум, интерфейс должен включать способ запуска и остановки медиа-носителя и регулировки громкости.

Параграф внутри тегов <video>

Это называют **резервный контент** — он будет отображаться, если браузер, показывающий страницу, не поддерживает элемент <video>, позволяя нам обеспечить поддержку для старых версий браузеров. Это может быть все, что вы захотите; в нашем примере мы предоставили прямую ссылку на видеофайл, поэтому пользователь может хотя бы получить к нему доступ, независимо от того, какой браузер он используют.

Встроенное видео будет выглядеть примерно так:



Живой пример: https://vk.cc/aubgeK

Код примера: https://vk.cc/aubgjb

Поддержка нескольких форматов

Присутствует одна проблема с приведенным выше примером, которую вы, возможно, уже заметили, если пытались получить доступ к прямой ссылке выше с помощью браузера, такого как Safari или Internet Explorer. Видео не будет играть!

Давайте кратко рассмотрим терминологию. Форматы, такие как MP3, MP4 и WebM, называются **форматами контейнеров**. Они содержат различные части, которые составляют всю песню или видео — например, звуковую дорожку, видеодорожку (в случае видео) и метаданные для описания представленного носителя.

Аудио и видео треки также находятся в разных форматах, например:

- Контейнер WebM обычно загружает звук Ogg Vorbis с видео VP8 / VP9. Поддерживается в основном в Firefox и Chrome.

- Контейнер MP4 часто включает аудио AAC или MP3 с видео H.264. Поддерживается в основном в Internet Explorer и Safari.
- Более старый контейнер Ogg имеет тенденцию идти с аудио Ogg Vorbis и видео Ogg Theora. Поддерживалось главным образом в Firefox и Chrome, но было вытеснено более качественным форматом WebM.

Звуковой проигрыватель обычно воспроизводит звуковую дорожку напрямую, например, файл MP3 или Ogg.

Вышеупомянутые форматы существуют для сжатия видео и аудио в управляемые файлы (необработанные видео и аудио очень большие). Браузеры содержат разные Codecs, вроде Vorbis или H.264, которые используются для преобразования сжатого звука и видео в двоичные цифры и обратно. Как указано выше, браузеры, к сожалению, не поддерживают одни и те же кодеки, поэтому вам придется предоставить несколько файлов для каждого медиа-продукта. Если вам не хватает правильного кодека для декодирования носителя, он просто не сможет играть.

Примечание: Возможно, вам интересно, как сложилась такая ситуация. **MP3** (для аудио) и **MP4/H.264** (для видео) широко поддерживаются и имеют высокое качество. В то же время, они защищены патентами — американские патенты охватывают MP3 по крайней мере до 2017 года, и H.264 самое меньшее до 2027 года, так что браузеры, которые не являются держетелями этих патентов, должны платить огромные суммы денег для поддержки этих форматов. Кроме того, многие люди избегают несвободного программного обеспечения в принципе, предпочитая открытые форматы. Вот почему мы должны предоставить несколько форматов для разных браузеров.

Так как мы это сделаем? Взгляните на следующий обновленный пример:

```
<video controls>
  <source src="rabbit320.mp4" type="video/mp4">
  <source src="rabbit320.webm" type="video/webm">
  <p>Ваш браузер не подддерживает HTML5 видео. Вот <a
href="rabbit320.mp4">ссылка на видео</a> взамен.</p>
</video>
```

Здесь мы изъяли атрибут src из нашего тега <video>, и вместо этого включали отдельные элементы <source>, каждый из которых ссылается на собственный источник. В этом случае браузер пройдется по элементам <source> и начнёт воспроизводить первый из них, который имеет поддерживаемый кодек. Включение источников WebM и MP4 должно быть достаточно для воспроизведения вашего видео на большинстве платформ и браузеров в наши дни.

Каждый элемент <source> также имеет атрибут type. Он не обязательный, но рекомендуется его включать — он содержит MIME types видеофайла, браузеры могут прочитать их и сразу же пропустить видео, которые они не понимают. Если type не включен, браузеры загружают и пытаются воспроизвести каждый файл до тех пор, пока не найдут тот, который будет работать, затрачивая больше времени и ресурсов.

Другие параметры <video>

Есть ряд других параметры, которые вы можете включить в HTML5 элемент video. Взгляните на наш третий пример:

```
<video controls width="400" height="400"
       autoplay loop muted
       poster="poster.png">
  <source src="rabbit320.mp4" type="video/mp4">
  <source src="rabbit320.webm" type="video/webm">
  <p>Your browser doesn't support HTML5 video. Here is a <a
href="rabbit320.mp4">link to the video</a> instead.</p>
</video>
```

На выходе получим нечто, подобное этому:

Новые параметры:

width and height

Вы можете контролировать размер видео либо с помощью этих атрибутов, либо с помощью CSS. В обоих случаях видео поддерживают собственное соотношение ширины и высоты — известное как **соотношение сторон**. Если соотношение сторон не поддерживается установленными вами размерами, видео будет увеличиваться, чтобы заполнить пространство по горизонтали, а заполненному пространству по умолчанию будет задан сплошной цвет фона.

autoplay

Этот атрибут позволяет сразу начать воспроизведение звука или видео, пока остальная часть страницы загружается. Вам не рекомендуется использовать автовоспроизведение видео (или аудио) на ваших сайтах, потому что пользователи могут найти это действительно раздражающим.

loop

Этот атрибут позволяет воспроизводить видео (или аудио) снова, когда он заканчивается. Это также может раздражать, поэтому используйте тогда, когда это действительно необходимо.

muted

Этот атрибут заставляет проигрыватель воспроизводить звук, отключенный по умолчанию.

poster

Этот атрибут принимает в качестве значения URL-адрес изображения, который будет отображаться до воспроизведения видео. Он предназначен для заставки к видео или рекламы.

preload

этот атрибут используется в элементе для буферизации больших файлов. Он может принимать одно из трех значений:

- "none" не буферизирует файл
- "auto" буферизирует медиафайл
- "metadata" буферирует только метаданные файла

Пример, приведенный выше: https://vk.cc/aubiCV

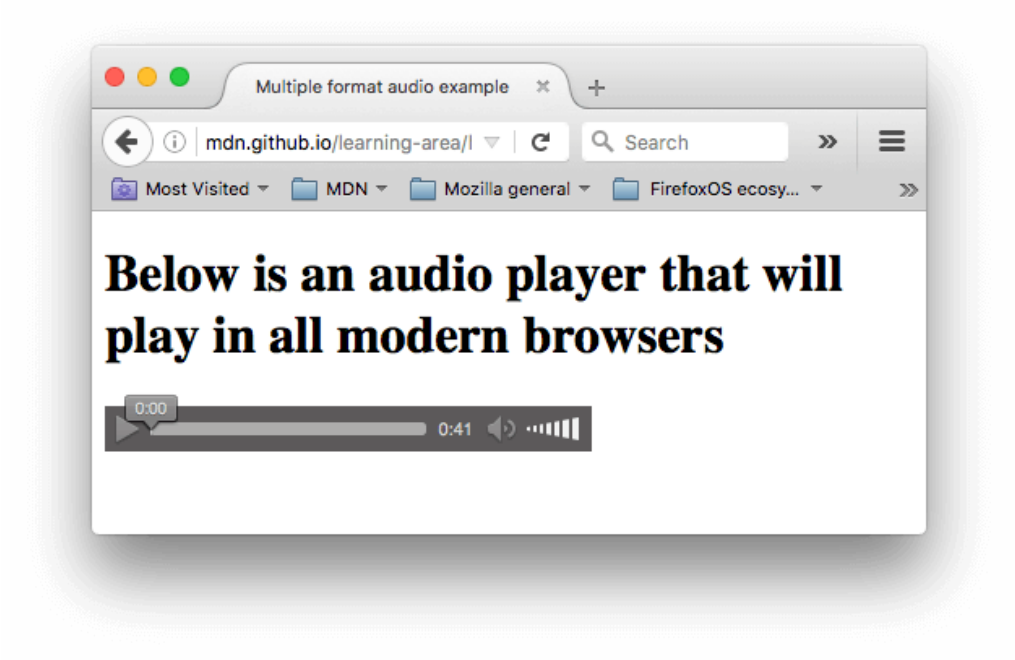
Исходный код: https://vk.cc/aubiI1

Элемент <audio>

Элемент <audio> работает точно так же, как элемент <video>, с несколькими небольшими отличиями, которые описаны ниже. Типичный пример может выглядеть так:

```
<audio controls>
  <source src="viper.mp3" type="audio/mp3">
  <source src="viper.ogg" type="audio/ogg">
  <p>Your browser doesn't support HTML5 audio. Here is a <a
href="viper.mp3">link to the audio</a> instead.</p>
</audio>
```

В браузере это вызывает следующее:



Он занимает меньше места, чем видеоплеер, поскольку нет визуального компонента - вам просто нужно отображать элементы управления для воспроизведения звука. Другие отличия от видео HTML5 заключаются в следующем:

- Элемент <audio> не поддерживает атрибуты width / height - опять же, нет визуального компонента, поэтому присваивать ширину или высоту не к чему.
- Он также не поддерживает атрибут poster- опять же, из-за отсутствия визуального компонента.

Помимо этого, <audio> поддерживает все те же функции, что и <video>.

Отображение текстовых дорожек к видео

Теперь обсудим немного более продвинутую концепцию, о которой очень полезно знать. Многие люди не могут или не хотят слышать аудио / видео контент, который они находят в Интернете, по крайней мере, в определенное время. Например:

- У многих людей есть слуховые нарушения (более известные как слабослышащие или глухие).
- Другие могут не слышать звук, потому что они находятся в шумной обстановке (например, в переполненном баре при показе спортивной игры) или, возможно, не хотят беспокоить других, если они находятся в тихом месте (например, в библиотеке).
- Люди, которые не говорят на языке из видео, могут захотеть увидеть текстовую расшифровку или даже перевод, чтобы помочь им понять медиа-контент.

Разве было бы неплохо иметь возможность предоставить этим людям транскрипцию слов, произносимых в аудио / видео? Благодаря HTML5 видео, вы можете, с форматом WebVTT и элементом <track>.

Замечание: "Транскрибировать" значит записывать устную речь в форме письменной.Получившийся в результате текст есть 'расшифровка'.

WebVTT - это формат для записи текстовых файлов, содержащих несколько строк текста, а также метаданные, такие как время, в которое вы хотите отображать каждую текстовую строку, и даже ограниченную информацию о стиле / позиционировании. Эти текстовые строки называются **репликами**, и вы можете отображать разные типы для разных целей, наиболее распространенными являются:

субтитры

Переводы иностранного материала, для людей, которые не понимают слов, произнесенных в аудио.

титры

Синхронизированные транскрипции диалога или описания значимых звуков, чтобы люди, которые не могут слышать звук, поняли что происходит.

рассчитанные описания

Текст для преобразования в аудио, чтобы обслуживать людей с нарушениями зрения.

Типичный файл WebVTT будет выглядеть примерно так:

```
WEBVTT

1
00:00:22.230 --> 00:00:24.606
Это первый субтитр.

2
00:00:30.739 --> 00:00:34.074
Это второй.

  ...
```

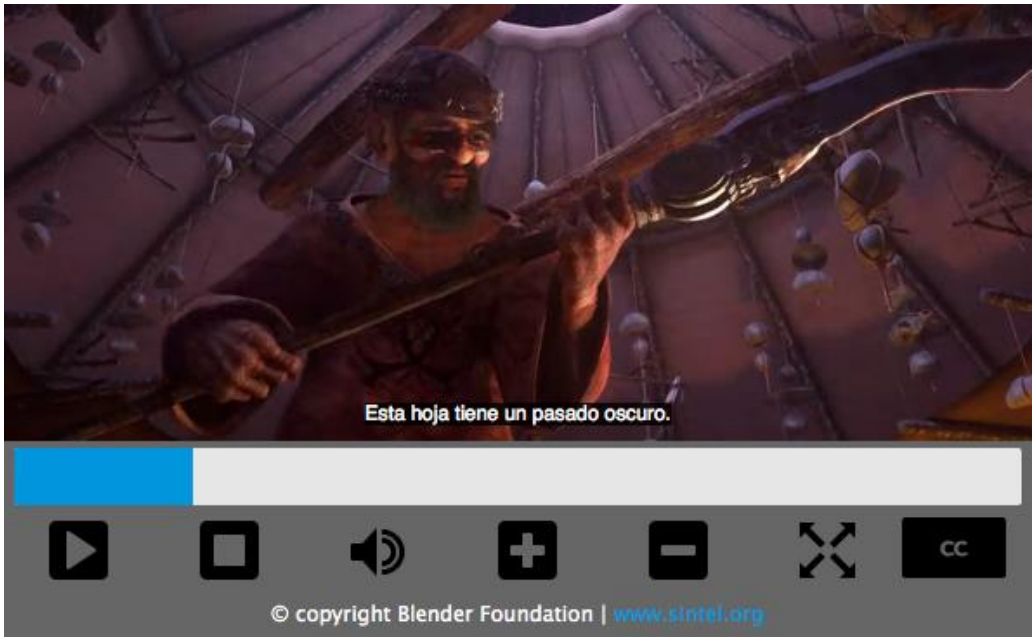
Чтобы отобразить это вместе с воспроизведением мультимедиа HTML, вам необходимо:

1. Сохраните его как .vtt- файл, в разумном месте.
2. Ссылка на файл .vtt с элементом <track>. <track> должен быть помещен в <audio> или <video>, но после элементов <source>. Используйте атрибут kind, чтобы указать, являются ли реплики субтитрами, титрами или описаниями. Кроме того, используйте srclang, чтобы сообщить браузеру, на каком языке вы записывали субтитры.

Вот пример:

```
<video controls>
    <source src="example.mp4" type="video/mp4">
    <source src="example.webm" type="video/webm">
    <track kind="subtitles" src="subtitles_en.vtt" srclang="en">
</video>
```

Это приведет к просмотру видео с субтитрами, таким как:



Задание:

1. Запишите несколько собственных видео и аудио.
   Возможно, вам придется сделать некоторое преобразование, чтобы в конечном итоге получить WebM и MP4 в случае видео, а также MP3 и Ogg в случае аудио, но есть достаточно программ, чтобы вы могли сделать это без особых проблем, таких как Miro Video Converter и Audacity.
2. Сохраните аудио и видео файлы в новом каталоге на вашем компьютере.
3. Создайте новый HTML файл в том же каталоге, и назовите его index.html.
4. Добавьте элементы <audio> и <video> на страницу; чтобы они отображали элементы управления браузером по умолчанию.
5. Введите оба варианта элемента <source>, чтобы браузеры находили оптимальный формат звука, который он поддерживает и загружает. Они должны включать type атрибуты.
6. Дайте элементу <video> заставку, которая будет отображаться до начала воспроизведения видео. Получайте удовольствие, создавая свою собственную заставку к видео.

Больше информации можно найти здесь:
https://developer.mozilla.org/ru/docs/Web/HTML