

Интерфейс HTMLMediaElement

Урок 11





Кадочников Алексей

Frontend-разработчик

- ☀ Веб-разработчик со стажем более 9 лет
- ☀ Преподаватель GeekBrains с 2015 года
- ☀ Автор курсов по Frontend на портале Geekbrains
- ☀ Работал в таких компаниях, как VK и Wizard-C



План курса

1

Знакомство с JavaScript

2

Основы JavaScript

3

Функции в JavaScript

4

Циклы и массивы

5

Работа с объектами в JavaScript

6

Введение в DOM

7

Работа с DOM

8

Основы событий в JavaScript

9

Работа с событиями в JavaScript

10

Основы шаблонизации, работа с JSON

11




Работа с медиафайлами

12

Основы API, итоги курса



Что будет на уроке сегодня

-  Интерфейс HTMLMediaElement
-  Интерфейс MediaStream
-  Использование MediaStream в качестве источника Audio



Интерфейс HTMLMediaElement

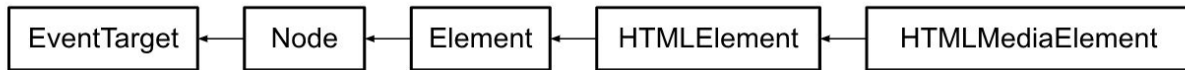




Интерфейс HTMLMediaElement

Этот интерфейс добавляет к HTMLElement свойства и методы, необходимые для поддержки базовых мультимедийных возможностей, общих для аудио и видео. Элементы HTMLVideoElement и HTMLAudioElement наследуют этот интерфейс.

HTMLMediaElement также наследует свойства от своих предков: HTMLElement, Element, Node и EventTarget.





События HTMLMediaElement

В плане прослушивания событий медиаэлементы ничем не отличаются от остальных DOM-элементов. Мы также можем использовать `addEventListener()` или присваивать обработчики `<on + имя события>` свойствам медиаэлемента.





События HTMLMediaElement



События HTMLMediaElement: когда они срабатывают

1. **abort** — когда ресурс не был полностью загружен, однако не в результате ошибки. Например, в процессе загрузки ресурса вызван метода `load`.
2. **canplay** — когда браузер может начать воспроизведение, но загружено недостаточно данных для воспроизведения без прерываний на буферизацию ещё не загруженного, исходя из скорости соединения.
3. **canplaythrough** — когда браузер может начать воспроизведение и загружено достаточно данных для воспроизведения без прерываний на буферизацию.
4. **durationchange** — в момент обновления атрибута `duration`.
5. **emptied** — когда медиасодержимое удаляется. Например, медиаресурс полностью или частично загружен, а метод `HTMLMediaElement.load()` вызван для его перезагрузки.



События HTMLMediaElement: когда они срабатывают

6. **ended** — в момент окончания воспроизведения (<audio> или <video>) при достижении конца файла или по причине недоступности данных.
7. **error** — когда медиаресурс не может быть загружен из-за ошибки.
8. **loadeddata** — когда первый блок (фрейм) загружен.
9. **loadedmetadata** — в момент загрузки метаданных.
10. **loadstart** — в момент начала загрузки медиаресурса.
11. **pause** — когда запрос на остановку обработан, а воспроизведение остановлено. Как правило, при вызове метода `pause`.



События HTMLMediaElement: когда они срабатывают

- 12. **play** — в момент изменения свойства `paused` из значения `true` в `false`, как результат вызова метода `play` или изменения значения свойства `autoplay`.
- 13. **playing** — когда воспроизведение готово начаться после остановки на паузу или из-за задержки при получении недостающих данных.
- 14. **progress** — срабатывает периодически в процессе загрузки браузером данных медиаресурса.
- 15. **ratechange** — при изменении скорости воспроизведения.
- 16. **seeked** — в момент завершения операции поиска, когда пользователь отпускает ползунок на шкале прогресса воспроизведения трека.
- 17. **seeking** — в момент начала операции поиска.



События HTMLMediaElement: когда они срабатывают

- 18. **stalled** — когда браузер пытается получить данные медиаресурса, но данные не поступают.
- 19. **suspend** — когда загрузка данных медиаресурса приостановлена.
- 20. **timeupdate** — при обновлении текущего времени воспроизведения, представленного в атрибуте `currentTime`.
- 21. **volumechange** — при изменении громкости.
- 22. **waiting** — когда воспроизведение приостановлено из-за временной нехватки данных медиаресурса.



Свойства и методы HTMLMediaElement

Рассмотрим, какие именно свойства добавляет интерфейс HTMLMediaElement.
Часть свойств представляют DOM-атрибуты элемента audio



События HTMLMediaElement: когда они срабатывают

1. **autoplay (Boolean)** — отражает значение HTML-атрибута `autoplay`, указывающего, должно ли воспроизведение начинаться автоматически, как только будет доступно достаточно медиафайлов, чтобы сделать это без прерывания.
2. **buffered (TimeRanges)**, только для чтения — указатель на объект временных интервалов медиаресурса, который браузер буферизировал в момент обращения к свойству.
3. **controls (Boolean)** — отражает значение HTML-атрибута `controls`, указывающего, должны ли отображаться элементы пользовательского интерфейса управления воспроизведением медиаресурса.
4. **controlsList (DOMTokenList)**, только для чтения — отражает значение HTML-атрибута. Свойство `controlslist` позволяет выбирать, какие элементы управления будут отображаться в интерфейсе медиапроигрывателя. `DOMTokenList` принимает одно или несколько из трёх возможных значений: `nodownload`, `nofullscreen` (только `video`) и `noremoteplayback`.



События HTMLMediaElement: когда они срабатывают

- 5. **crossOrigin (DOMString)** — отражает значение HTML-атрибута crossorigin, указывающего настройку CORS для этого медиаэлемента.
- 6. **currentSrc (DOMString) только для чтения** — возвращает абсолютный URL-адрес выбранного браузером медиаресурса.
- 7. **currentTime (Number)** — время текущей позиции воспроизведения в секундах. Установка этого значения переместит точку воспроизведения в новую позицию.
- 8. **duration (Number) только для чтения** — общая продолжительность медиаресурса в секундах. Если данные о медиаресурсе недоступны, поле получит значение NaN. А если медиаресурс неопределённой длины (в случае MediaStream), будет содержать значение +Infinity.



События HTMLMediaElement: когда они срабатывают

- 9. **ended (Boolean)**, только для чтения — возвращает индикатор того, что воспроизведение медиаресурса завершено.
- 10. **error (MediaError)**, только для чтения — содержит указатель на объект ошибки или null, если ошибки не было.
- 11. **loop (Boolean)** — отражает значение HTML-атрибута loop, указывающего, что воспроизведение медиа должно быть зациклено.
- 12. **muted (Boolean)** — отражает значение HTML-атрибута muted, указывающего, выключен ли звук у медиаресурса (true, если выключен, false — включён).



События HTMLMediaElement: когда они срабатывают

- 13. networkState (Number)**, только для чтения — содержит целочисленную константу (enumeration), которая отражает текущее состояние получения медиаресурса по сети и может принимать следующие значения.

Имя константы	Значение	Описание
NETWORK_EMPTY	0	Пока нет данных. При этом поле readyState содержит значение HTMLMediaElement.HAVE_NOTHING
NETWORK_IDLE	1	Браузер выбрал источник медиаресурса, но загрузка ещё не начиналась
NETWORK_LOADING	2	Идёт загрузка данных медиаресурса
NETWORK_NO_SOURCE	3	Браузер не нашёл источников медиаресурса



События HTMLMediaElement: когда они срабатывают

- 14. **paused (Boolean)**, только для чтения — указывает, что воспроизведение медиаресурса остановлено.
- 15. **playbackRate (Number)** — указывает скорость, с которой воспроизводится медиаресурс.
- 16. **played (TimeRanges)**, только для чтения — указатель на объект временных интервалов медиаресурса, который браузер воспроизвёл (если есть).
- 17. **preload (DOMString)** — отражает значение HTML-атрибута preload, определяющего способ загрузки браузером данных медиаресурса. Возможные значения: none, metadata, auto.



События HTMLMediaElement: когда они срабатывают

- 18. readyState (Number)**, только для чтения — содержит целочисленную константу (enumeration), которая отражает состояние готовности медиаресурса и может принимать следующие значения.

Имя константы	Значение	Описание
HAVE_NOTHING	0	Информация о медиаресурсе недоступна
HAVE_METADATA	1	Браузер загрузил достаточно информации о медиаресурсе для инициализации метаданных. Вызов поиска (seeking) больше не вызовет ошибки
HAVE_CURRENT_DATA	2	Доступны данные для текущей позиции воспроизведения, но недостаточно, чтобы воспроизвести больше одного кадра
HAVE_FUTURE_DATA	3	Доступны данные для текущей позиции воспроизведения, а также для как минимум двух кадров видео
HAVE_FUTURE_DATA	4	Загружено достаточно данных для воспроизведения, и скорость загрузки стабильно высокая для воспроизведения медиапотока без прерываний



События HTMLMediaElement: когда они срабатывают

- 19. **seekable (TimeRanges)**, только для чтения — указатель на объект временных интервалов (если есть) медиаресурса, которые может выбрать пользователь.
- 20. **seeking (Boolean)**, только для чтения — индикатор того, что медиаресурс находится в состоянии выбора новой позиции воспроизведения.
- 21. **src (DOMString)** — отражает значение HTML-атрибута src, определяющего URL-адрес медиаресурса для использования.
- 22. **textTracks (TextTrackList)**, только для чтения — указатель на динамический список объектов TextTrack.
- 23. **volume (Number)** — значение громкости звука, от 0.0 (звук выключен) до 1.0 (максимальная громкость).



События HTMLMediaElement

Помимо методов, наследуемых от базовых классов
HTMLElement, Element, Node и EventTarget,
HTMLMediaElement поддерживает следующие методы:





События HTMLMediaElement: какие элементы поддерживает

1. **HTMLMediaElement.addTextTrack()** — добавляет объект типа TextTrack, например, как трек для субтитров.
2. **HTMLMediaElement.captureStream()** — возвращает MediaStream, захватывает media-поток медиасодержимого.
3. **HTMLMediaElement.canPlayType()** — принимает строку MIME-типа медиаресурса, может также включать параметры кодека. Вызов canPlayType() возвращает строковые значения:
 - *“probably”* — если браузер способен воспроизвести данный формат,
 - *“maybe”* — если недостаточно информации для определения возможности воспроизведения,
 - пустую строку — если формат не поддерживается.



События HTMLMediaElement: какие элементы поддерживает

4. **HTMLMediaElement.load()** — перезапускает процесс выбора оптимального источника медиаресурса из вариантов, указанных в элементах source или переданного в атрибуте src или единственном source-элементе.
5. **HTMLMediaElement.pause()** — останавливает воспроизведение медиаресурса.
6. **HTMLMediaElement.play()** — начинает воспроизведение медиаресурса.
7. **HTMLMediaElement.setMediaKeys()** — возвращает Promise. Устанавливает ключи MediaKeys для декодирования медиаресурса в процессе воспроизведения.
8. **HTMLMediaElement.setSinkId()** — устанавливает ID медиаустройства для воспроизведения медиа и возвращает Promise. Требуется получения разрешения на доступ к соответствующему устройству.



Медиабуферизация, поиск и временные интервалы



Медиабуферизация, поиск и временные интервалы

Иногда полезно знать, какая именно часть аудио или видео загружена или воспроизводится без задержки. Например, для отображения в буферном индикаторе загрузки медиапроигрывателя.





Медиабуферизация, поиск и временные интервалы

Атрибут `HTMLMediaElement.buffered` содержит временные интервалы загруженных частей медиаресурса — объект `TimeRanges`.

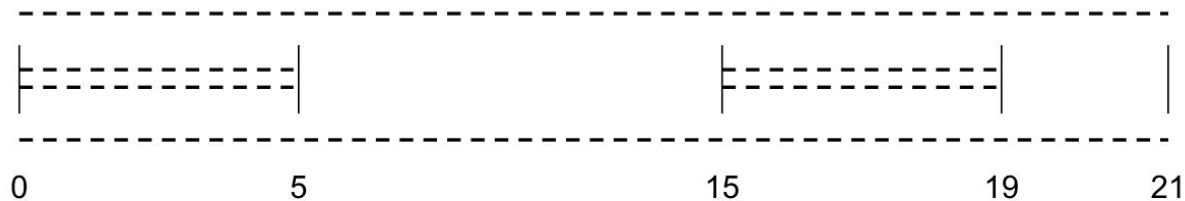
Будет одинаково работать как для **<audio>**, так и для **<video>**. Рассмотрим простой пример для audio:

```
1 <audio src="file.mp3" controls></audio>
2
3 <script>
4   const audio = document.querySelector('audio')
5   const buffered = audio.buffered
6 </script>
7
```



Медиабуферизация, поиск и временные интервалы

Если в процессе буферизации пользователь не перемещал ползунок по временной шкале трека, обычно существует только один временной интервал. В противном случае могут появиться новые интервалы, как показано на рисунке ниже:



Может получиться, к примеру, два буферизированных интервала по времени: один с нулевой по пятую секунду, а второй — с пятнадцатой по девятнадцатую секунду.



Пара слов о Played

Свойство `played` содержит указатель на объект `TimeRanges` временных интервалов медиаресурса, полностью воспроизведённых браузером. Если суммировать все интервалы `played`, получим долю прослушанного аудио. Это может быть полезно для сбора метрик, например.

```
1 <audio src="file.mp3" controls></audio>
2
3 <script>
4   const audio = document.querySelector('audio')
5   const {played} = audio // объект TimeRanges
6
7   const handlePlayed = () => {
8     let totalPlayedSeconds = 0
9     for(i = 0; played.length; i++) {
10       totalPlayedSeconds += played.end(i) - played.start(i)
11     }
12     console.log(totalPlayedSeconds)
13   }
14
15   audio.addEventListener('pause', handlePlayed, false)
16   audio.addEventListener('ended', handlePlayed, false)
17 </script>
18
19
```



Интерфейс MediaStream

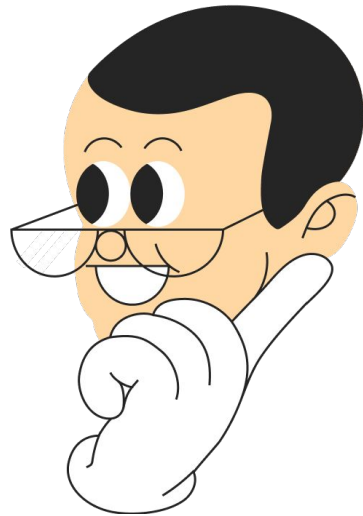




Интерфейс MediaStream

Интерфейс MediaStream представляет поток медиаданных и может использоваться как источник медиасодержимого в HTMLMediaElement. Поток состоит из нескольких треков, таких как видео- и аудиотреки.

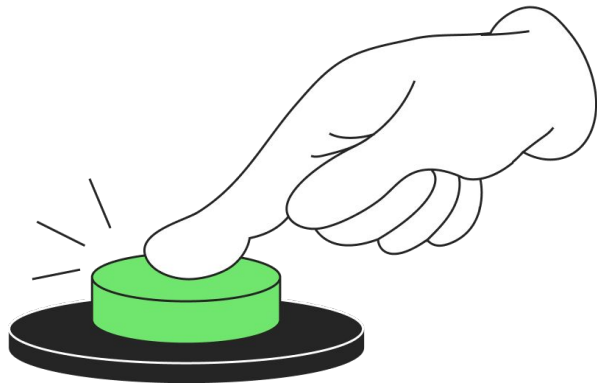
Каждый трек — экземпляр MediaStreamTrack. Получить MediaStream можно либо посредством конструктора, либо вызовом MediaDevices.getUserMedia().





События MediaStream

1. **addtrack** — срабатывает при добавлении нового объекта `MediaStreamTrack`. Доступно как свойство `onaddtrack`.
2. **removetrack** — срабатывает при удалении объекта `MediaStreamTrack`. Доступно как свойство `onremovetrack`.





Свойства и методы `MediaStream`

Интерфейс `MediaStream` наследует свойства своего родителя `EventTarget`.

1. **`active (boolean)`**, только для чтения — возвращает `true`, если `MediaStream` активен, иначе — `false`.
2. **`readyState (String)`**, только для чтения — может принимать значение `live`, которое указывает, что поток подключён и делает всё возможное для предоставления данных в реальном времени. В этом случае получение данных можно включить или выключить посредством свойства `enabled`. Значение `ended` указывает, что поток завершён и больше не предоставляет новых данных.
3. **`id (DOMString)`**, только для чтения — строка, содержащая 36 символов универсального уникального идентификатора (UUID) потока.



Использование MediaStream в качестве источника Audio





Использование MediaStream в качестве источника Audio

Самый простой способ — попросить пользователя предоставить предварительно записанный файл. Сделать это можно, создав простой элемент ввода файла и добавив фильтр, позволяющий выбирать только аудиофайлы, и атрибут `capture`, который указывает, что мы хотим получить его прямо с микрофона.

```
1 <input type="file" accept="audio/*" capture>  
2
```



Использование MediaStream в качестве источника Audio

Как только пользователь закончит запись и вернётся на веб-сайт, вам надо каким-то образом получить данные файла. Вы можете получить к ним доступ в обработчике события onchange элемента ввода, а затем прочитав свойство files объекта события.

```
1 <input type="file" accept="audio/*" capture id="recorder">
2
3 <audio id="player" controls></audio>
4
5 <script>
6   const recorder = document.getElementById('recorder')
7   const player = document.getElementById('player')
8
9   recorder.addEventListener('change', (event) => {
10     const [file] = event.target.files
11     const url = URL.createObjectURL(file)
12     // Добавление потока в качестве источника элементу
13     player.src = url
14   })
15 </script>
16
17
```



Какие возможности открывает доступ к файлу

Вы можете делать с доступным файлом практически всё что хотите.




Например:


1. Добавить его элементу audio для воспроизведения, как в примере выше.
2. Загрузить его на устройство пользователя.
3. Загрузить его на сервер с помощью XMLHttpRequest / fetch / FormData.
4. Передать его через Web Audio API и применить к нему фильтры.





Итоги урока

-  Интерфейс HTMLMediaElement
-  Интерфейс MediaStream
-  Использование MediaStream в качестве источника Audio

Спасибо 
за внимание

