**Глава 4. ЭЛЕМЕНТ VIDEO**

**4.1. Назначение элемента VIDEO и принципы его применения**

Элемент <video> – новый элемент языка HTML5, предназначенный для воспроизведения видеозаписей. До появления HTML5 и <video> для этого в большинстве случаев применялся плагин мультимедийной платформы Adobe Flash Player [9], который может быть встроен в html-страницу с помощью элемента <object>.

Помимо собственно html-элемента <video> HTML5 предоставляет программный интерфейс позволяющий разработчику на JavaScript управлять воспроизведением, создавать пользовательские интерфейсы управления и интегрировать видео с остальными html-элементами и программными интерфейсами (например, с Canvas API).

В простейшем случае элемент <video> в исходном коде html-страницы выглядит так, как это показано на рис. 4.1. Элемент содержит только два атрибута: идентификатор (id) и путь к видеофайлу (src).



Рис. 4.1. Простейший вид элемента <video>

Обратите внимание: в элементе <video> (рис. 4.1) даже не указывается размеры окна для воспроизведения видеофайла – в этом случае, автоматически создается окно, размеры которого определяются шириной и высотой кадра записанного видео. Получить эти характеристики в Windows можно, просмотрев вкладку «Подробно», свойств видеофайла, указанного атрибутом src (рис. 4.2). Кроме размеров окна, в свойствах видеофайла можно найти еще много полезной информации: продолжительность видео, частоту кадров, скорость передачи потоков видео и аудиоданных и пр. Вся эта информация используется элементом <video> при воспроизведении по умолчанию.

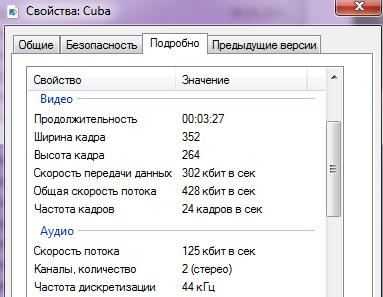


Рис. 4.2. Фрагмент скриншота окна свойств видеофайла

В окне браузера элемент <video> (рис. 4.1) отобразится примерно так, как это продемонстрировано на рис. 4.3. Внешний вид видеоплеера зависит от браузера

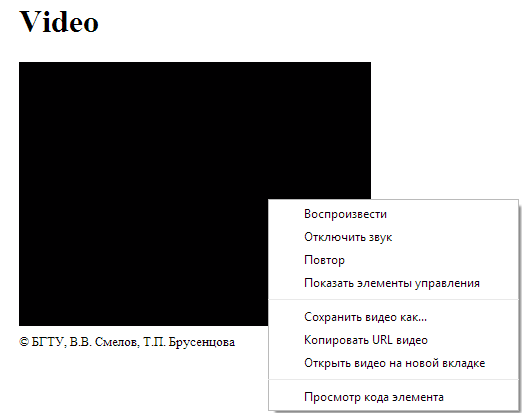


Рис. 4.3. Отображение браузером html-файла на рис. 4.1

Обратите внимание на следующее: браузер не отображает никаких элементов управления воспроизведением (пуск, стоп, пауза и т. д.); размеры окна воспроизведения, скорость воспроизведения и пр. определяются метаинформацией, записанной в видеофайле; для управления воспроизведением доступно контекстное меню.

**4.2. Форматы видеофайлов**

Видеофайл в любого формата следует воспринимать как сжатый файл, содержащий видео-поток и аудио-поток. В табл. 4.1 приведены три наиболее распространенных видео-формата.

Таблица 4.1. Наиболее распространённые форматы

видеофайлов

|  |  |  |
| --- | --- | --- |
| Видео-формат | Стандарт видео | Стандарт аудио |
| mp4 | H.264 | AAC |
| ogg/ogv | Theora | Vorbis |
| webm | VP8 | Vorbis |

Каждая часть видеофайла (видео-поток и аудио-поток), представляет собой закодированные с целью уменьшения размера данные. Стандарты видео и аудио данных, приведенные в табл. 4.1, по сути, определяют алгоритм их сжатия. Устройства или программы, способные кодировать исходный сигнал в последовательность бит и, наоборот, декодировать последовательность бит в вид готовый для воспроизведения, называется кодеком. Файл, содержащий закодированные с помощью определенных кодеков видео- и аудио-данные называется контейнером. Поэтому, когда говорят о видео-формате, по сути, имеют ввиду формат контейнера.

СпецификацияHTML5 допускает применение любого видео-формата. Все зависит от реализации браузера, которая определяет перечень поддерживаемых им форматов (табл. 4.2).

Таблица 4.2. Поддержка форматов браузерами

|  |  |  |  |
| --- | --- | --- | --- |
| Браузер | mp4 | webm | ogg |
| Internet Explorer | + | - | + |
| Chrome | + | + | + |
| Firefox | + | + | + |
| Safari | + | - | - |
| Opera |  | + | + |

Отчасти такое положение с форматами видеофайлов скрашивается возможностью указать несколько источников в одном элементе <video> с помощью вложенных элементов <source> (рис. 4.4). В этом случае, заданные источники будут проверяться на возможность использования в порядке их перечисления в теле тега <video>. Если, ни один из перечисленных источников не может быть применен или элемент <video> вообще не поддерживается, в окно браузера выведется текстовое сообщение, расположенное ниже тегов <source>.

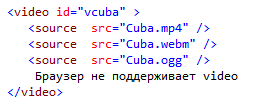


Рис. 4.4. Элемент <video> может указывать на

несколько видеофайлов

Для более полного знакомства с форматами видеофайлов рекомендуется источник [10].

**4.3. Потоковое видео**

В приведенных ранее примерах (рис. 4.1, 4.3) значение атрибута src (элементов <video> или <source>) представляет собой путь к локальному видеофайлу. Однако, если html-страница является компонентой web-приложения, видеофайл расположен на сервере и путь к нему должен быть представлен в формате URL (рис. 4.5). В этом случае доставка содержимого видеофайла будет осуществляться через сеть по протоколу HTTP.

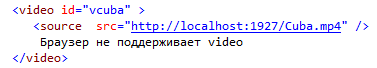


Рис. 4.5. В элементе <video> путь к исходному файлу может быть указан в формате URL

Если к html-странице, содержащий фрагмент подобный представленному на рис. 4.5, будет выполнено несколько http-запросов, в ответ на каждый будет осуществляться доставка содержимого одного и того же видеофайла. Учитывая, что общая скорость потока (битрейт) для каждого запроса должна составлять 428 килобит в секунду (рис. 4.2) несложно подсчитать, что при пятидесяти одновременно работающих подключениях к серверу, суммарный поток сервера, доставляющий видео должен составлять более 21 мегабайт в секунду. Кроме того, на стороне сервера не обязательно может быть видеофайл. Сервер может транслировать содержимое видеокамеры или другого устройства генерирующего видеопоток в реальном режиме времени. Очевидно, что количество возможных подключений к такому серверу будет ограничено и следует применять другие способы передачи видеопотоков.

Технологии потокового видео не рассматриваются в данном пособии, но отметим, что на сегодняшний день наибольшее распространение получили три технологии: HTTP Live Streaming компании Apple [11], Smooth Streaming от Microsoft [12] и HTTP Dynamic Streaming от Adobe.

**4.4. Применение элемента VIDEO**

Возьмем за основу html-страницу, представленную на рис. 4.1и 4.3, и поясним принципы применения элемента <video> и Video API, развивая это простейшее одностраничное web-приложение.

**4.4.1. Атрибут poster**

На рис. 4.6 представлен фрагмент html-страницы с элементом <video> , использующим атрибут poster.



Рис. 4.6. Применение атрибута poster элемента <video>

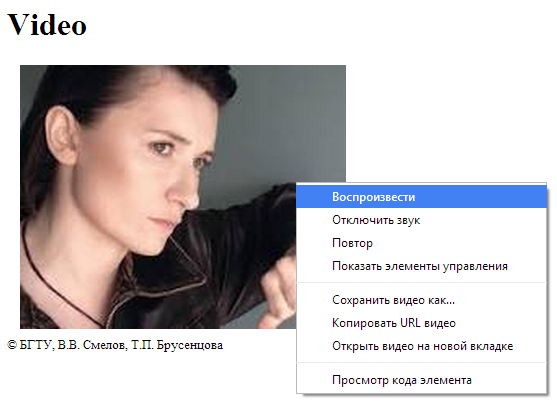


Рис. 4.7. Отображение браузером элемента <video> c атрибутом poster

Атрибут poster позволяет указать файл или URL картинки, которая будет отображаться в окне элемента <video> в начальный момент времени. Формат картинки, указываемой атрибутом poster, определяется применяемым браузером.

**4.4.2. Атрибут controls**

С помощью атрибута controls (рис. 4.8) в окно, отображаемое тегом <video> можно добавить элементы управления, позволяющие запускать, останавливать и позиционировать воспроизведение, регулировать звук и отображать текущую позицию видеофайла (рис. 4.9).



Рис. 4.8 Применение атрибута poster элемента <video>

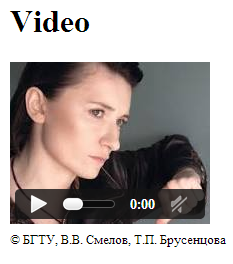


Рис. 4.7. Отображение браузером элемента <video> c атрибутом controls

**4.4.3. Атрибуты width и height**

Если в элементе <video> не используется атрибуты, задающие размеры окна для отображения видео, то по умолчанию будут установлены размеры, совпадающие с размерами кадра. Ширина и высота кадра видео является атрибутами видеофайла (рис. 4.2).

Размеры окна элемента <video> могут быть заданы атрибутами width (ширина) и height (высота). Какие бы значения для этих атрибутов не устанавливались, пропорции изображения при воспроизведении видео нарушены не будут.

Однако, может потребоваться согласовать размеры окна <video> с размерами картинки-постера, чтобы панель элементов управления не «вылезла» за пределы стартовой картинки.

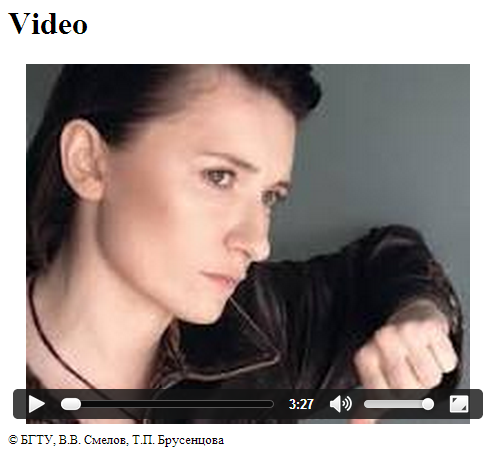


Рис. 4.8. Панель элементов управления тега <video> может «вылезти» за пределы картинки-постера

Кроме того, следует иметь ввиду, что если значения атрибутов width и height отличается от размера кадра, то при воспроизведении браузер будет постоянно (кадр за кадром) выполняет преобразование выводимого в окно изображения.

На рис. 4.3 представлен Обратите внимание: в элементе <video> (рис. 4.1) даже не указывается размеры окна для воспроизведения видеофайла – в этом случае, автоматически создается окно, размеры которого определяются шириной и высотой кадра записанного