Медиавозможности, встроенные в HTML5

Многие услышали об HTML5, когда Apple отказалась добавлять поддержку Flash на iOS-устройства. Технология Flash доминировала на рынке (некоторые утверждают, что она его и вовсе захватила) в качестве дополнительного модуля для обслуживания видео в браузере. Но вместо использования технологии от Adobe Apple в вопросах вывода на экран сложного медиасодержимого решила положиться на HTML5. HTML5 неплохо справился с задачей, а публичная поддержка со стороны Apple дала этой версии могучий толчок и сделала медиаинструменты HTML5 привлекательными для широкой аудитории.

Мы уже обсуждали, что сегодня люди склонны использовать термин «HTML» вместо «HTML5», но в отношении медиа различие имело значение. До HTML5 добавление видео и аудио в разметку было не таким удобным. Теперь эта задача упростилась.

Добавление видео и аудио в разметку HTML

Работать с видео и аудио в HTML довольно просто. Вот пример ссылки на видеофайл из разряда «проще некуда»:

```
<video src="myVideo.mp4"></video>
```

В HTML со всей трудной работой справляется один-единственный элемент `<video></video>` (или `<audio></audio>` для аудио). Между открывающим и закрывающим тегами можно вставить текст: он будет отображаться, если у пользователя возникнут проблемы с загрузкой. Можно добавить дополнительные атрибуты, например задать высоту и ширину. Сделаем это:

```
<video src="myVideo.mp4" width="640" height="480">If you're reading this
either the video didn't load or your browser is waaaayyyyyy old!</video>
```

Теперь, если добавить предыдущий фрагмент кода на страницу и посмотреть на его работу в любом браузере, видео появится, но без элементов управления его воспроизведением. Чтобы получить возможность управлять видео, нужно добавить атрибут `controls`. В качестве примера можно добавить атрибут `autoplay`, но вообще этого лучше не делать: автовоспроизведение выбесит кого угодно! Вот как выглядит код с этими атрибутами:

```
<video src="myVideo.mp4" width="640" height="480" controls autoplay>If
you're reading this either the video didn't load or your browser is
waaaayyyyyy old!</video>
```

Результат выполнения этого фрагмента показан на скриншоте.

Рис. 2.3. Мы вставили видео на страницу, добавив минимум кода

В число прочих атрибутов входят `preload` для управления предварительной загрузкой медиа, `loop` для повторного воспроизведения и `poster` для определения кадра заставки для видео (изображение, отображающееся во время загрузки видео). Для применения атрибута достаточно вставить его в тег. Вот как выглядит пример, включающий все эти атрибуты:

```
<video src="myVideo.mp4" width="640" height="480" controls autoplay
preload="auto" loop poster="myVideoPoster.png"> If you're reading
this either the video didn't load or your browser is waaaayyyyyy
old!</video>
```

Резервные возможности

Элемент `<source>` позволяет при необходимости добавить резервные возможности. Например, наряду с видео в формате MP4 можно обеспечить поддержку другого формата. Кроме того, если у пользователя в браузере нет подходящей технологии проигрывания, можно предоставить ссылки на скачивание видеофайлов. Рассмотрим такой пример:

```
<video width="640" height="480" controls preload="auto" loop
poster="myVideoPoster.png">
    <source src="myVideo.sp8" type="video/super8" />
```

```
    <source src="myVideo.mp4" type="video/mp4" />
    <p><b>Download Video:</b> MP4 Format: <a href="myVideo.mp4">"MP4"</a></p>
</video>
```

Здесь мы сначала указали выдуманный видеоформат `super8`. Браузер считывает код сверху вниз, решая, какой файл воспроизводить. Если он не поддерживает формат `super8`, то переходит к следующему — в нашем случае `mp4`. В итоге браузер переходит по ссылкам для скачивания, если не может согласовать какой-либо из перечисленных форматов. Атрибут `type` сообщает браузеру MIME-тип файла. Без этого атрибута браузер получит контент и все равно попытается воспроизвести его. Но если вы знаете MIME-тип, лучше добавьте его. Предыдущий пример кода и образец видеофайла в формате MP4 (по чистой случайности это фрагмент сериала Coronation Stree, в котором я снимался; тогда у меня еще были волосы и надежды сыграть главную роль вместе с Де Ниро) находятся в разделе example2 кода для этой главы.

Работа audio и video практически ничем не различается

Элемент `<audio>` работает по таким же принципам и с такими же атрибутами (исключая `width`, `height` и `poster`), но у `<audio>` отсутствует область проигрывания визуального содержимого.

Отзывчивое HTML5-видео и iFrames

Единственная проблема со столь понравившейся нам реализацией видео в HTML5 — в ней нет отзывчивости. И действительно, в книге приведен пример отзывчивой HTML5- и CSS-разметки, которая не реагирует на изменение условий просмотра. К счастью, для встроенного HTML-видео это можно легко исправить. Просто уберите из разметки все атрибуты высоты и ширины (например, удалите `width="640" height="480"`) и добавьте в CSS следующий код:

```
video {
  max-width: 100%;
  height: auto;
}
```

Да, с файлами, которые могут храниться на локальном устройстве, этот прием работает вполне успешно. Но он не решает проблемы видео, встроенного в iFrame (низкий поклон YouTube, Vimeo и другим сайтам). Следующий код позволяет добавить трейлер фильма «Успеть до полуночи» из YouTube:

```
<iframe width="960" height="720" src="https://www.youtube.com/embed/B1_
N28DA3gY" frameborder="0" allowfullscreen></iframe>
```

Но если добавлять этот код к странице в неизменном виде, то даже при использовании CSS-правила произойдет обрезка, если ширина окна просмотра будет менее 960 пикселей.

Проще всего решить эту проблему с помощью небольшого CSS-трюка, впервые примененного французским CSS-специалистом Тьерри Кобленцем (Thierry Ko-

blentz), который создал, по сути, блок с правильным соотношением сторон для содержащегося в нем видео. Собственные объяснения этого мага можно найти по ссылке http://alistapart.com/article/creating-intrinsic-ratios-for-video.

Если лень самим вычислять и подставлять соотношение сторон, можно не заморачиваться: один онлайн-сервис способен сделать это за вас. Просто откройте http://embedresponsively.com/ и поместите в адресную строку этого сайта URL-адрес вашего iFrame. В результате вы получите простой фрагмент кода, который можно вставить в собственную разметку.

К примеру, для трейлера фильма «Успеть до полуночи» ресурс выдает следующее (обратите внимание на значение `padding-bottom` для определения соотношения сторон):

```
<style>
  .embed-container {
    position: relative;
    padding-bottom: 56.25%;
    height: 0;
    overflow: hidden;
    max-width: 100%;
    height: auto;
  }
  .embed-container iframe,
  .embed-container object,
  .embed-container embed {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
  }
</style>
<div class="embed-container">
  <iframe
    src="http://www.youtube.com/embed/B1_N28DA3gY"
    frameborder="0"
    allowfullscreen
  ></iframe>
</div>
```

Вот и все! Просто добавьте этот фрагмент в код и получите полностью адаптируемое YouTube-видео (примечание: дети, не берите пример с мистера Де Ниро — курить вредно!).

Итоги

В этой главе мы рассмотрели много всего, начиная с основ создания страницы, проходящей проверку на соответствие стандартам HTML5, и заканчивая встраи-

ваемым в разметку сложным медиасодержимым (видео) и вопросами обеспечения его отзывчивого поведения. Все это и не имеет отношения к отзывчивому веб-дизайну, но зато мы познакомились с семантикой в коде, смысловым наполнением страниц и обеспечением их пригодности для пользователей, применяющих вспомогательные технологии.

Упражнение

Мы рассмотрели множество HTML-элементов. Это лишь малая часть всех существующих элементов, но мы, безусловно, рассмотрели те из них, которые могут вам понадобиться в ежедневной работе. Предлагаю сделать небольшое упражнение, чтобы увидеть, как много вы поняли. Вот скриншот с веб-дизайном сайта для этой книги:

Рис. 2.4. Веб-дизайн сайта, разработанного для этой книги

Если у вас macOS, вы можете получить исходный файл Sketch: он включен в загружаемый код под названием RWD3e_design.sketch.

Взгляните на дизайн и попробуйте создать для него HTML-страницу. Рассмотрите часть заголовка. Подумайте о языке контента и о метатегах, которые могут вам понадобиться. Затем подумайте о визуальных эффектах. Какие элементы лучше использовать для создания раздела навигации? Как быть с каждым из этих маленьких разделов под изображением книги? А как насчет блока DOWNLOAD CODE? Есть идеи, как его разметить?

Сайт https://rwd.education покажет, что выбрал я. Но постарайтесь не заглядывайть туда, пока не попробуете сами!