Марк Пилгрим

bhv

ПОГРУЖЕНИЕ В HTML5

- Новые семантические элементы <header>, <footer>, <section> и др.
- Новый элемент для работы с двумерной графикой <Canvas>
- Новые типы полей ввода для Web-форм
- Возможности встраивания видео в Web-страницы без использования дополнительных плагинов
- Возможности построения Web-приложений, способных работать в автономном режиме при отсутствии подключения к Интернету
- Возможности геопозиционирования
- Преимущества использования локального хранилища по сравнению с обычными cookie-файлами
- Создание собственных словарей микроданных для настройки разметки, используемой поисковиками при выводе результатов поиска

HTML5: Up and Running

Mark Pilgrim

Марк Пилгрим

ПОГРУЖЕНИЕ В HTML5

Санкт-Петербург

«БХВ-Петербург»

2011

УДК 681.3.06
ББК 32.973.26-018.2
П32

Пилгрим М.

П32 Погружение в HTML5: перев. с англ. — СПб.: БХВ-Петербург, 2011. — 304 с.: ил.

ISBN 978-5-9775-0688-5

Подробное руководство по всем новшествам стандарта HTML5. Показано, как использовать в Web-разработках новые функциональные возможности, открывающиеся при применении HTML5. Представлено множество простых и понятных практических примеров, позволяющих использовать разметку HTML5 для добавления графики, аудио, видео, автономных возможностей и много другого.

Для Web-разработчиков

УДК 681.3.06
ББК 32.973.26-018.2

Эта книга написана на основе первоисточника, находящегося по адресу http://diveintohtml5.org и поддерживаемого автором

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Перевод с английского и редактирование	*Ольги Кокоревой*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Виктория Пиотровская*
Дизайн обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 20.04.11.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 24,51.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12.

ISBN 978-5-9775-0688-5

Оглавление

Благодарности

Автор приносит глубокую благодарность издательству O'Reilly Media, опубликовавшего версию этой книги, защищенную их авторскими правами, но любезно предоставившего авторую разрешение опубликовать ее на своем сайте на условиях лицензии Creative Commons Attribution 3.0 Unported (см. **http://diveintohtml5.org/about.html**).

Кроме того, в книге использованы иллюстрации из Open Clip Art Library (**http://openclipart.org/**) и других источников, владельцам которых я признателен за разрешение использовать в своей книге результаты их труда.

Далее, в книге использован код `jQuery` (MIT), `Modernizr` (MIT), `geo_location_javascript.js` (MIT), `gears_init.js` (BSD), `highlighter.js` (ASL 2.0), `excanvas.js` (ASL 2.0), `Bespin` (MPL) и `canvas.text.js` (MIT). Полный список всех правовых положений и замечаний об авторских правах можно найти по адресу: **http://diveintohtml5.org/legal.html**. Всем коллегам, давшим свое разрешение на использование в книге своих разработок и ценные советы, я весьма признателен. По ходу изложения автор приносит благодарности и всем, кто помог ему ценным советом в процессе работы над этой книгой.

Марк Пилгрим

Введение

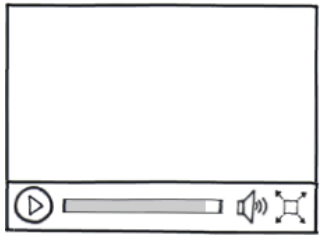
Что представляет собой HTML5? Это — следующее поколение стандартов HTML 4.01[1], XHTML 1.0[2] и XHTML 1.1[3]. HTML5 предоставляет новые функциональные возможности, необходимые современным Web-приложениям. Кроме того, этот стандарт регламентирует множество функций Web-платформы, которые Web-разработчики использовали годами, но которые никогда не проверялись и не документировались комитетом по стандартизации. Скажем, удивит ли вас тот факт, что объект `Window` никогда не был формально документирован? Помимо того что HTML5 вводит новые функции, спецификация HTML5 представляет собой первую попытку формально документировать все стандарты "де-факто", которые Web-браузеры использовали годами.

Пять фактов, которые необходимо знать о HTML5

Прежде чем углубляться в детали, приведем пять ключевых фактов о HTML5, которые должен знать каждый:

1. HTML5 — это не есть некая большая, единая, "монолитная" сущность.

   У вас может возникнуть вопрос: "Как мне начать работать с HTML5, если старые браузеры его не поддерживают?" Но этот вопрос поставлен некорректно, потому что он уводит вас в сторону от обсуждаемой темы. Дело в том, что HTML5 — это не есть нечто



[1] Подробнее см. **http://www.w3.org/TR/html401/**, **http://www.w3.org/TR/html401/struct/global.html**, **http://www.umade.ru/resources/specifications/html401/index.htm**. — *Прим. перев*.

[2] XHTML (Extensible Hypertext Markup Language — Расширяемый язык разметки гипертекста) — язык разметки Web-страниц, по возможностям сопоставимый с HTML, созданный на базе XML. Подробнее см. **http://www.w3.org/TR/xhtml1/**, **http://www.opennet.ru/docs/RUS/XHTML1/**. — *Прим. перев*.

[3] Подробнее см. **http://www.w3.org/TR/xhtml11/**, **http://www.opennet.ru/docs/RUS/XHTML11/**. — *Прим. перев*.

единое и неделимое. Напротив, данная спецификация представляет собой набор индивидуальных функций и возможностей. Поэтому нет никакого смысла пытаться дать общее определение термину "поддержка HTML5". Однако, несмотря на это, вполне возможно дать определения поддержке отдельных функциональных возможностей, например, таких как элементы `<canvas>`, `<video>` или `<geolocation>`.

Возможно, вы думаете о HTML как о наборе тегов и угловых скобок. Это отчасти правильно, потому что теги и угловые скобки действительно представляют собой важную часть HTML. Важную, но не единственную, и это — еще не все. Спецификация HTML5 также определяет, каким образом все эти теги и угловые скобки используют объектную модель документов (Document Object Model, DOM[1]) для взаимодействия с JavaScript. Так, спецификация HTML5 не просто определяет тег `<video>`; в ней определяется и соответствующий интерфейс прикладного программирования DOM (DOM API) для обработки объектов видео в DOM. Вы можете использовать этот API, чтобы обнаруживать поддержку различных форматов видео, воспроизводить видео, прерывать воспроизведение, воспроизводить аудио в немом режиме (mute), отслеживать, какой объем видео был загружен, а также делать все, что вам необходимо для построения пользовательского интерфейса приложений, богатого функциональными возможностями, базирующимися на теге `<video>`.

В *главе 2* и *приложении 1* будет показано, как правильно определять наличие поддержки браузером каждой из новых возможностей HTML5.

2. Вам ничего не придется "выбрасывать".

Нравится вам это или нет, но вы не можете отрицать бесспорного факта: HTML 4 — это самый успешный из всех форматов разметки, существовавших до сих пор. Формат HTML5 строится на базе HTML 4, на волне этого успеха. Вам нет никакой необходимости отказываться от имеющихся у вас "наработок" — вашей уже существующей разметки. Вам не требуется заново учить или переучивать то, что вы уже знаете. Если ваше Web-приложение работает сегодня, используя HTML 4, то оно будет работать и завтра, используя HTML5. На этом можно поставить точку.

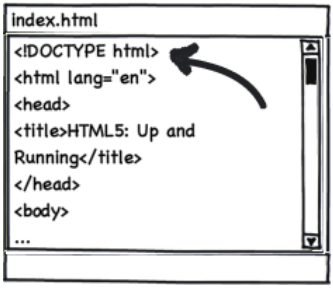
Зато, если вам хочется *усовершенствовать* свои Web-приложения, созданные ранее, то данная книга — это как раз то, что вам и требуется. Вот реальный пример: HTML5 поддерживает все элементы управления формами, которые имелись и в HTML 4, но, кроме того, добавляет и новые элементы, в том числе — новые элементы управления вводом. Некоторые из них представляют со-

[1] DOM (Document Object Model — "объектная модель документа") — это не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получать доступ к содержимому документов на языках HTML, XHTML и XML, а также изменять содержимое, структуру и оформление таких документов. Подробнее см. **http://www.w3.org/DOM/**, **http://javascript.ru/tutorial/dom**. — *Прим. перев.*

бой долгожданные дополнения, наподобие "ползунковых регуляторов" (sliders) и элементов выбора даты (date pickers); другие же не так просты. Например, элемент ввода данных типа `email` выглядит как обычное текстовое поле, но мобильные браузеры могут настраивать свою экранную клавиатуру так, чтобы пользователю было удобнее вводить почтовые адреса. Более старые браузеры, которые не поддерживают элемента ввода данных типа `email`, будут обращаться с этим полем точно так же, как они обычно обращаются с текстовыми полями, и форма все равно будет работать, без всякой необходимости вносить модификации в ее код или в скрипты. Это означает, что вы уже сегодня можете начать работать над усовершенствованием Web-форм, несмотря даже на то, что некоторые из посетителей вашего сайта могут продолжать пользоваться браузером Internet Explorer 6.

Подробная и детальная информация о формах HTML5 будет приведена в *главе 9*.

3. Начать работу с HTML5 очень просто.

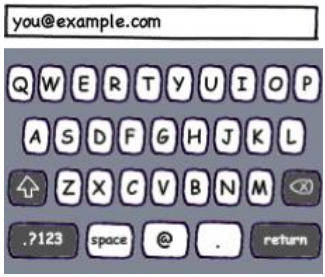


"Обновление" до HTML5 может оказаться не сложнее, чем изменение типа документа (*doctype*). Тип документа должен быть указан уже в первой строке любой страницы HTML. Предшествующие версии HTML определяли большое количество типов документов (doctypes), и выбор правильного типа документа из числа доступных мог представлять проблему. В HTML5, напротив, существует только один тип документа:

```
<!DOCTYPE html>
```

Обновление до типа документа HTML5 не разрушит вашу существующую разметку, потому что теги, определенные в HTML 4, по-прежнему поддерживаются и в HTML5. Но теперь вы сможете определять, использовать и новые семантические элементы наподобие `<article>`, `<section>`, `<header>` и `<footer>`, а также проверять их правильность. Об использовании этих элементов подробно рассказывается в *главе 3*.

4. Все работает уже сейчас.



Не имеет значения, что вы хотите делать с элементом `<canvas>`: рисовать на нем, воспроизводить видео, конструировать новые, усовершенствованные формы или строить Web-приложения, работающие автономно (offline), — вы быстро убедитесь в том, что спецификация HTML5 уже сегодня обеспечивается мощной поддержкой. Большинство популярных Web-браузеров, в том числе — Mozilla Firefox, Safari, Google Chrome, Opera, а также мобильные браузеры уже поддерживают такие элементы новой спецификации, как `<canvas>` (*глава 4*), `<video>` (*глава 5*), `<geolocation>` (*глава 6*), локальное хранилище `<localstorage>` (*глава 7*) и многое другое. Google уже поддерживает аннотации микроданных (microdata

annotations), о чем будет рассказано в *главе 10*. Даже корпорация Microsoft, которая редко бывает первопроходцем в пунктуальном следовании новым стандартам (особенно тем, которые находятся на стадии разработки и еще не утверждены), будет поддерживать большинство функций HTML5 в ожидаемом релизе своего браузера Internet Explorer 9[1].

Каждая глава этой книги содержит диаграммы совместимости для всех широко известных браузеров. Что еще важнее, в каждой главе приводится открытая дискуссия о том, какие опции вам нужны для поддержки старых браузеров. Такие элементы HTML5, как `<geolocation>` (*глава 6*) и `<video>` (*глава 5*) оказались первыми функциями, предоставляемыми плагинами браузеров, наподобие Gears[2] и Flash[3].

Некоторые другие функции, например, `<canvas>` (*глава 4*), могут полностью эмулироваться с помощью JavaScript. В данной книге будет показано, как обращаться к этим функциям, встроенная поддержка которых обеспечивается новейшими версиями браузеров, не отказываясь при этом от старых версий.

5. HTML5 уже здесь, и это — всерьез и надолго.

Как известно всем, Тим Бернерс-Ли (Tim Berners-Lee) изобрел Всемирную паутину (World Wide Web) в начале 1990-х годов. Впоследствии он основал Консорциум Всемирной паутины (World Wide Web Consortium, W3C), призванный координировать работу по созданию стандартов Web, чем эта организация и занималась в течение более чем 15 лет. Вот какое сообщение было сделано W3C относительно будущего Web-стандартов в июле 2009 года:

"На сегодня Директор объявляет о том, что, когда истечет срок действия текущего рабочего соглашения на право ведения деятельности для рабочей группы XHTML 2, намеченный на конец 2009 года, это соглашение продлено не будет. Поступая таким образом и выделяя основные ресурсы рабочей группе HTML, W3C

---

[1] Фактически, к моменту подготовки русского издания книги, этот браузер уже вышел (релиз-кандидат был представлен 10 февраля 2011 года). Подробнее о поддержке HTML5 в Internet Explorer 9 можно прочесть по следующим адресам: **http://www.coolwebmasters.com/browsers/1011-html5-support-internet-explorer-9.html**, **http://habrahabr.ru/blogs/ie/94968/**, **http://internet.cnews.ru/reviews/index.shtml?2010/12/17/420552**, **http://internet.cnews.ru/reviews/index.shtml?2010/12/17/420552**. — *Прим. перев.*

[2] Gears (ранее Google Gears) — открытое ПО от Google, позволяющее использование Web-приложений в режиме оффлайн. Еще 30 ноября 2009 года компания Google объявила: "Мы продолжаем поддержку Gears, так что ничего не сломается на сайтах, которые его используют. Но мы ожидаем, что разработчики будут использовать HTML5 для получения нужных функциональных возможностей, переходя на ориентированный на стандарты подход, который будет применим во всех браузерах". Подробнее см. **http://latimesblogs.latimes.com/technology/2009/11/google-gears.html**. — *Прим. перев.*

[3] Adobe Flash (ранее Macromedia Flash), или просто Flash (по-русски часто пишут флеш или флэш) — мультимедийная платформа компании Adobe для создания Web-приложений или мультимедийных презентаций. Широко используется для создания рекламных баннеров, анимации, игр, а также воспроизведения на Web-страницах видео- и аудиозаписей. Подробнее см. **http://www.adobe.com/ru/products/flash/?promoid=BPCEP**. — *Прим. перев.*

выражает надежду на ускорение развития HTML5 и проясняет свою позицию относительно будущего HTML"[1].

Так что HTML5 — это всерьез и надолго. А теперь давайте приступим к детальному изучению этой новой спецификации.

В данной книге будут рассмотрены следующие основные темы:

- Новые семантические элементы наподобие `<header>`, `<footer>` и `<section>` (*глава 3*);
- Элемент `<canvas>`, двумерная поверхность, содержимое которой можно программировать на JavaScript (*глава 4*);
- Элемент `<video>`, который можно встраивать в Web-страницы без необходимости прибегать к сторонним плагинам (*глава 5*);
- Элемент `<geolocation>`, который посетители вашего Web-сайта смогут выбирать для публикации своих географических координат в вашем Web-приложении;
- Постоянное локальное хранилище, использование которого не будет требовать использования сторонних плагинов (*глава 7*);
- Возможности автономной работы Web-приложений даже в случае разрыва сетевых соединений (*глава 8*);
- Усовершенствованные Web-формы HTML (*глава 9*);
- Микроданные (microdata), которые позволяют вам создавать собственные словари и встраивать семантическую разметку в документы HTML5 (*глава 10*).

Разработка HTML5 ведется таким образом, чтобы, по мере возможности, обеспечивать обратную совместимость с существующими Web-браузерами. Новые функции строятся на основе уже существующих и позволяют вам создавать информационное содержимое, которое (с игнорированием функций HTML5) может отображаться и более старыми версиями браузеров. Если вам требуется еще более высокий уровень контроля, вы можете выявлять поддержку браузером отдельных функций HTML5, используя всего лишь несколько строк на языке JavaScript. Не следует полагаться в этом на слабые возможности самих браузеров. Вместо этого вы можете протестировать поддержку нужной вам функции HTML5 самостоятельно. О том, как это делается, рассказывается в *главе 2*.

Соглашения, используемые в данной книге

В оформлении этой книги используются следующие соглашения о типографском оформлении:

- Текстом *курсивного начертания* выделяются новые термины;
- Текстом **полужирного начертания** выделяются почтовые адреса, URL и интерфейсные элементы;
- `Моноширинный шрифт` используется для листингов программ, имен переменных и функций, баз данных, типов данных, переменных окружения, утверждений и ключевых слов, имен файлов и расширений имен файлов;

[1] См. **http://www.w3.org/MarkUp/**. — *Прим. перев.*

- **`Моноширинный шрифт полужирного начертания`** используется для выделения команд или иного текста, который пользователь должен вводить вручную;
- *`Моноширинный шрифт курсивного начертания`* используется для выделения значений, которые должны быть заменены на пользовательский ввод, а также для контекстно-зависимых значений;

Использование примеров кода

Эта книга написана с тем, чтобы помочь вам справляться с вашей работой. В общем случае, вы можете использовать код, приведенный в книге, в ваших программах и написанной вами документации. Вам не требуется обращаться к нам за разрешением на использование материалов, за тем исключением, когда вы заимствуете значительные фрагменты кода. К примеру, написание программы с использованием нескольких небольших фрагментов кода из этой книги не требует обращения за разрешением. Но тиражирование и продажа носителей CD-ROM с примерами из этой или других книг O'Reilly разрешения требует. Если вы отвечаете на чей-то вопрос и приводите цитату со ссылкой на данную книгу, то такое использование разрешения не требует. Но для включения масштабного фрагмента, заимствованного из книги O'Reilly, в документацию к вашему продукту разрешение требуется.

Мы приветствуем библиографические ссылки, хотя и не настаиваем на точном соблюдении формы, в которой они приводятся. Такие ссылки обычно включают заголовок, имя автора, издательство и ISBN (International Standard Book Number, стандартный международный номер книги). Например: "*HTML5: Up and Running*, by Mark Pilgrim. Copyright 2010 O'Reilly Media, Inc., 978-0-596-80606-6."

Если вам кажется, что использование кода примеров выходит за рамки законного применения или только что приведенных условий получения разрешения, вы можете связаться с нами по адресу **permissions@oreilly.com**.

Safari® Books Online

Safari Books Online — это электронная библиотека "по требованию", позволяющая легко найти более 7 500 оригинальных справочных пособий и видео о передовых технологиях для получения быстрых ответов на интересующие вас вопросы.

Оформив подписку на сервис Safari Books Online, вы сможете прочесть любую страницу и просмотреть любой видеофильм из нашей интерактивной библиотеки. Читать книги можно на мобильном телефоне или других мобильных устройствах. Вы сможете просматривать заголовки новых книг до того, как они выйдут из печати, получать эксклюзивный доступ к рукописям, готовящимся к печати, а также возможность связаться по почте с их авторами. Более того, вы сможете копировать и вставлять код примеров, формировать свои списки "Избранного", скачивать отдельные главы, создавать закладки в интересующих вас разделах, создавать примечания, печатать страницы и пользоваться множеством экономящих время функций.

Издательство O'Reilly Media включило данную книгу в библиотеку Safari Books Online. Для получения полного доступа к этой книге и другим книгам сходной тематики издательства O'Reilly и других издательств подключайтесь бесплатно на Web-сайте **http://my.safaribooksonline.com**.

Как связаться с издательством O'Reilly

Пожалуйста, посылайте замечания и вопросы, относящиеся к этой книге, издателю:

O'Reilly Media Inc.,
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (в США и Канаде)
707-829-0515 (международный или местный)
707-829-0104 (факс)

У нас есть посвященная этой книге Web-страница, на которой приведен список опечаток, примеры и другая дополнительная информация. К этой странице можно обратиться по адресу:

http://oreilly.com/catalog/9780596806066

У этой книги есть и свой Web-сайт по адресу:

http://diveintohtml5.org/

Для отправки замечаний и вопросов, касающихся этой книги, отправляйте письма с указанием ISBN-номера (**9780596806066**) по адресу:

bookquestions@oreilly.com

Для получения дополнительной информации о книгах, конференциях, центрах распространения (Resource Centers) и сервисе O'Reilly Network посетите Web-сайт издательства:

http://www.oreilly.com

Замечания об изданиях этой книги

Эта книга создана на основе исходного текста HTML5, размещенного по адресу **http://diveintohtml5.org/** и поддерживаемого автором. Электронные версии книги (**http://oreilly.com/catalog/9780596806066**) в форматах ePub, Mobi и PDF, свободных от ограничений DRM, как и версия Safari Books Online (**http://my.safaribooksonline.com/9781449392154**), содержат все гиперссылки, присутствующие в оригинале, в то время как в печатное издание включено лишь подмножество гиперссылок. Если вы читаете печатное издание, обращайтесь к одному из электронных изданий или к исходному варианту, где вы найдете гораздо большее количество ссылок. Поскольку автор поддерживает сайт **http://diveintohtml5.org/** в HTML5, там вы найдете живые примеры кода, описанные в этой книге, многие из которых при подготовке печатного издания были модифицированы. При просмотре примеров на сайте **http://diveintohtml5.org/** следует иметь в виду, что их визуализация зависит от используемого вами браузера.

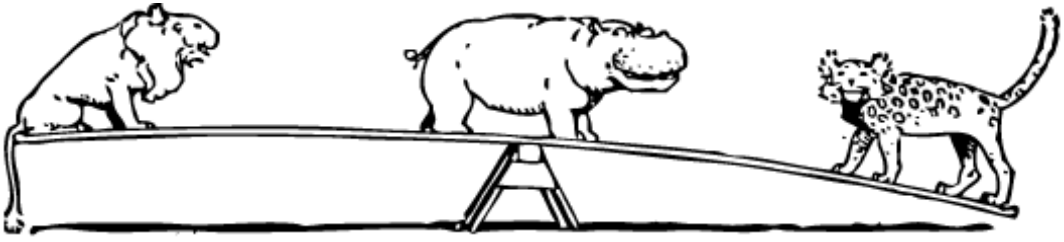
Глава 1

Как мы пришли туда, где находимся?

Недавно я натолкнулся на цитату, взятую из текста, написанного одним из разработчиков Mozilla о трениях, которые внутренне присущи командам, занятым разработкой стандартов (**http://lists.w3.org/Archives/Public/public-html/2010Jan/0107.html**):

"Реализации и спецификации должны взаимодействовать между собой крайне осторожно. Никто не хочет, чтобы реализации появлялись до того, как будет завершена работа над спецификацией, потому что тогда пользователи будут зависеть и от деталей реализации, и от ограничений спецификации. Но при этом никто не хочет и такого развития событий, когда спецификация оказывается уже законченной, принятой и утвержденной, но ни один из авторов и разработчиков ПО еще не получил никакого опыта реализации проектов, основанных на этой спецификации. Дело в том, что разработчикам спецификаций тоже нужна обратная связь. Это — неизбежный источник напряженности и трений, но нам просто необходимо преодолеть и эту напряженность, и эти трения".

Запомните эту цитату, зарубите ее себе на носу, запишите ее в свой блокнот. А теперь давайте разбираться с тем, что представляет собой стандарт HTML5 и что вызвало его к жизни.



Типы MIME

Эта книга посвящена HTML5, а не предшествующим версиям HTML, и не более ранним версиям XHTML. Но для понимания истории появления и развития HTML5, как и мотивов, послуживших движущими силами развития этого стандар-

та, вам необходимо сначала разобраться с некоторыми техническими деталями, с частности, с *типами MIME*[1].

Каждый раз, когда ваш Web-браузер запрашивает страницу, Web-сервер сначала отсылает ему *заголовки* (headers), и только затем — фактическую разметку страницы (page markup). Обычно эти заголовки невидимы, несмотря на то, что на самом деле они представляют собой средства Web-разработки, которые станут видимыми, если вы в этом заинтересованы. Заголовки играют важную роль, потому что они сообщают вашему браузеру, как следует интерпретировать разметку страницы, которая будет получена браузером впоследствии. Наиболее важный заголовок называется `Content-Type` и выглядит он так:

```
Content-Type: text/html
```

Строка `text/html` называется "типом содержимого" (или "типом контента", или "MIME-типом") страницы. Этот заголовок единолично определяет тип ресурса, который представляет собой страница, и, следовательно, указывает, как эта страница должна быть визуализирована. Изображения имеют собственные типы MIME (например, `image/jpeg` — для изображений формата JPEG, `image/png` — для изображений формата PNG и т. д.). Файлы JavaScript имеют собственный тип MIME. Каскадные страницы стилей (CSS) тоже имеют свой собственный тип MIME. Все имеет свой тип MIME. Вся Всемирная Паутина работает благодаря типам MIME.

Естественно, окружающая нас реальность гораздо сложнее. Первое поколение Web-серверов (а я рассматриваю Web-серверы, появившиеся, начиная с 1993 года) не отправляло заголовков `Content-Type`, потому что таких заголовков в те времена еще не существовало (они были изобретены после 1994 года). В целях обеспечения обратной совместимости с продуктами, датируемыми вплоть до 1993 года, некоторые популярные браузеры при определенных обстоятельствах игнорируют заголовок `Content-Type`. Этот прием называется "сниффингом контента" (content sniffing) или "распознаванием браузера" (browser sniffing). Но, как общее эмпирическое правило, все, что вы когда-либо просматривали через Web, включая страницы HTML, изображения, скрипты, видео, PDF-файлы, и все, что имеет URL, предоставлялось вам с указанием конкретного типа MIME в заголовке `Content-Type`.

Запомните этот факт. Впоследствии мы к этому еще вернемся.

Длинный экскурс в историю разработки стандартов

Почему существует элемент `<img>`? Естественно, это не тот вопрос, который вы слышите каждый день. Очевидно, что *кто-то* его создал. Такие вещи не появ-

[1] Multipurpose Internet Mail Extensions (MIME) — многоцелевые расширения почты Интернета, произносится как "майм"). Это стандарт, описывающий передачу различных типов данных по электронной почте, а также, в более широком понимании — спецификация для кодирования информации и форматирования сообщений таким образом, чтобы их можно было пересылать по Интернету. Подробнее см. **http://ru.wikipedia.org/wiki/MIME**, **http://www.relcom.ru/Internet/Services/Email/MIME/**, **http://www.mhonarc.org/~ehood/MIME/**. — *Прим. перев.*

ляются из ниоткуда. Каждый элемент, каждый атрибут, каждая функция HTML, которой вы когда-либо пользовались, определенно, кем-то были созданы. Тот, кто их создал, продумал, как они должны работать, и написал все это. Все эти люди — не боги, и они далеко не безупречны. Они просто люди. Умные люди, конечно, но просто люди.

Одной из замечательных особенностей стандартов, которые разрабатываются открыто и сразу же становятся достоянием общественности, является то, что вы можете совершить экскурс в прошлое и ответить на некоторые из этих вопросов. Дискуссии происходят в списках рассылки, которые обычно архивируются и доступны для публичного поиска. Поэтому я решил немного заняться "почтовыми раскопками", чтобы попытаться ответить на вопрос о том, откуда взялся элемент `<img>`. Мне пришлось сделать экскурс в те времена, когда еще не существовало такой организации, как World Wide Web Consortium (W3C). Я вернулся к самым ранним дням существования Web, когда количество Web-серверов по всему миру можно было сосчитать по пальцам рук (ну, возможно, задействовав дополнительно еще парочку пальцев и на ногах).

25 февраля 1993 года Марк Андреессен (Mark Andreessen) написал (**http://1997.webhistory.org/www.lists/www-talk.1993q1/0182.html**)[1]:

"Я хотел бы предложить новый, необязательный тег HTML:

```
IMG
```

Обязательный аргумент выглядит так: `SRC="url"`.

Этот тег указывает браузеру растровый или пиксельный файл, предназначенные для извлечения через локальную сеть или Интернет в виде изображения, которое должно быть встроено в текст в той точке, где указан тег.

Рассмотрим пример:

```
<IMG SRC="file://foobar.com/foo/bar/blargh.xbm">
```

(Закрывающего тега нет, это просто самостоятельный тег.)

Этот тег может встраиваться в качестве якоря как и любой другой; когда это происходит, он становится значком, чувствительным к активации, как и любой текстовый якорь.

Браузерам следует дать возможность выбирать, какие форматы изображений они будут поддерживать. Например, Xbm и Xpm вполне подходят для поддержки. Если браузер не может интерпретировать данный формат, он вместо отображения может выбрать любое действие в соответствии со своими предпочтениями (например, X Mosaic выведет в качестве метки-заполнителя стандартное растровое изображение).

[1] Ветвь дискуссии, обсуждаемую на протяжении последующих нескольких страниц, можно отслеживать, выполняя щелчки мышью по ссылкам **Next message** и **Previous message**.

Это — необходимая функциональность для X Mosaic; мы добились того, что это работает, и теперь собираемся применять эту функциональность хотя бы для внутреннего пользования. Я открыт для предложений по использованию этой функции в HTML; если у вас есть лучшие идеи, чем та, которую я представил здесь, сообщите мне об этом, пожалуйста. Я знаю, что здесь есть неопределенность относительно формата изображения, но я пока не вижу никакого альтернативного варианта, отличного от того, чтобы просто сказать "дайте браузеру возможность сделать то, что он может" и подождать, пока со временем не появится лучшее решение (возможно, со временем, MIME)".

Приведенная цитата нуждается в некоторых пояснениях:

Xbm[1] и Xpm[2] представляли собой популярные графические форматы в системах UNIX.

"Mosaic" представлял собой один из самых ранних Web-браузеров, а "X Mosaic" — версию, которая работала в UNIX-системах. На момент написания этого сообщения в начале 1993 года, Марк Андреессен (Mark Andreessen) еще не основал компанию, которая сделала его знаменитым, Mosaic Communications Corporation, и еще не начинал работать над продуктом, который стал "флагманом" этой компании, "Mosaic Netscape" (хотя возможно, вы лучше знаете и компанию, и ее продукт по именам, которые они получили впоследствии: "Netscape Corporation" и "Netscape Navigator").

Упоминание "возможно, со временем, MIME" представляет собой ссылку на согласование содержимого (content negotiation), функцию HTTP, при которой клиент (например, Web-браузер) сообщает серверу (например, Web-серверу) о том, какие типы ресурсов он поддерживает (например, `image/jpeg`), чтобы сервер мог возвращать данные в формате, предпочитаемом клиентом. Оригинальный вариант HTTP был определен в 1991 году, и единственная его реализация, существовавшая на февраль 1993, не предоставляла клиентам способа сообщить серверу о том, какие типы изображений они поддерживают, отсюда и дилемма, с которой столкнулся Марк.

Через несколько часов был получен ответ от Тони Джонсона (Tony Johnson):

"У меня в Midas 2.0 тоже есть очень похожая вещь (и мы пользуемся ею в SLAC и выпустим ее в ближайшем публичном релизе, который должен произойти с недели на неделю), за тем исключением, что все имена отличаются, и функция имеет дополнительный аргумент `NAME="name"`. Возможности функции примерно такие же, как все то, что обеспечивается предложенным вами тегом `IMG`, например:

```
<ICON name="NoEntry" href="http://note/foo/bar/NoEntry.xbm">
```

[1] В компьютерной графике, текстовый формат для монохромных изображений, применяется в X Windows System для хранения изображений курсоров и значков (icons), используемых в X GUI. Подробнее см. **http://en.wikipedia.org/wiki/X_BitMap**. — *Прим. перев.*

[2] X Pixmap (XPM) — текстовый формат графических файлов. Используется серверной частью системы X Windows. Главной особенностью формата является то, что он использует семантику языка C (Си) и может быть включен в код. Подробнее см. **http://en.wikipedia.org/wiki/X_PixMap**, **http://openfile.ru/types/xpm**. — *Прим. перев.*

Смысл параметра `name` в том, чтобы дать браузеру возможность иметь набор "встроенных" изображений. Если значение `name` совпадет с именем "встроенного" изображения, браузер сможет использовать его вместо того, чтобы искать и извлекать изображение. Кроме того, имя может служить и как подсказка для "линейных" браузеров, сообщающая им, какой символ нужно подставить вместо изображения.

Я не придаю слишком важного значения именам параметров и тегов, но считаю, что было бы благоразумно, если бы мы использовали одно и то же. Я не слишком озабочен сокращениями, так что почему бы не использовать `IMAGE=` и `SOURCE=`. Я склоняюсь к слову `ICON`, потому что оно подразумевает, что изображение (`IMAGE`) должно быть небольшим, но не будет ли `ICON` слишком "тяжеловесным" словом?"

Midas являлся еще одним ранним Web-браузером, "современником" X Mosaic. Это был кросс-платформенный браузер; он работал как на UNIX, так и на VMS[1]. Аббревиатура SLAC означала "Стэнфордский центр линейного ускорителя" (Stanford Linear Accelerator Center), ныне — Национальная ускорительная лаборатория SLAC (SLAC National Accelerator Laboratory), исследовательскую лабораторию, в которой располагался первый Web-сервер в США (и первый Web-сервер за пределами Европы[2]). На тот момент, когда Тони (**http://www.slac.stanford.edu/history/earlyweb/wizards.shtml#Tony%20Johnson**) написал это сообщение, SLAC уже был "ветераном" WWW, который содержал на своем сервере целых пять Web-страниц в течение 441 дня (и на то время этот срок был гигантским).

Затем Тони продолжил:

"Хотя мы обсуждаем новые теги, у меня есть еще один, в чем-то похожий тег, который я хотел бы поддерживать в Midas 2.0. В принципе, он выглядит так:

```
<INCLUDE HREF="...">
```

Здесь мои намерения заключаются в том, что я хотел бы, чтобы в первый документ был включен второй, как раз в месте вхождения тега. В принципе, документ, на который дается ссылка, может представлять собой что угодно, но основной целью является обеспечение возможности встраивать в документы изображения (в данном случае размер изображения может быть произвольным). Опять же, намерения таковы, чтобы после появления HTTP2 формат включенного документа стал предметом для отдельного согласования".

Здесь "HTTP2" относится к протоколу Basic HTTP в том виде, в котором он был определен в 1992 году. На тот момент, в начале 1993 года, он в значительной степени был еще нереализованным. Черновик стандарта, на тот момент известного

---

[1] Серверная ОС, разработанная во второй половине 1970-х компанией Digital Equipment Corporation для компьютеров VAX (см. **http://ru.wikipedia.org/wiki/VAX**). Впоследствии была портирована на платформы DEC Alpha и Intel Itanium. Сейчас принадлежит Hewlett-Packard. Подробнее см. **http://wikiadmin.net/OpenVMS**, **http://h71000.www7.hp.com/index.html?jumpid=/go/openvms**. — *Прим. перев.*

[2] См. **http://www.slac.stanford.edu/history/earlyweb/history.shtml**, **http://www.slac.stanford.edu/history/earlyweb/firstpages.shtml**. — *Прим. перев.*

как "HTTP2", продолжал развиваться, и со временем был утвержден как стандарт "HTTP 1.0" (хоть и произошло это через целых три года). Стандарт HTTP 1.0[1] включал заголовки для согласования контента (**http://www.w3.org/Protocols/HTTP/HTRQ_Headers.html#z3**), иначе говоря, так было реализовано предположение "возможно, со временем, MIME".

Далее Тони продолжил развивать свою идею:

"Альтернатива, которую я рассматривал, выглядит так:

```
<A HREF="..." INCLUDE>See photo</A>
```

Я не слишком стремлюсь добавлять функциональные возможности тегу `<A>`, но основная идея заключается в том, чтобы поддерживать совместимость с браузерами, которые не принимают во внимание параметр `INCLUDE`. Мое намерение заключается в том, чтобы браузеры, которые интерпретируют `INCLUDE`, замещали текст привязки (в данном случае "`See photo`") встраиваемым документом (картинкой), а старые или менее продвинутые браузеры полностью игнорировали тег `INCLUDE`".

Это предложение никогда не было воплощено в жизнь, хотя сама идея предоставления текста в случае отсутствия изображения представляла собой важный прием (**http://diveintoaccessibility.org/day_23_providing_text_equivalents_for_images.html**), которого не было в изначальном предложении тега `<IMG>`, сделанного Марком. Год спустя эта функция была искусственно "прикручена" как атрибут `<img alt>`, который Netscape сразу же "сломал" (**http://www.cs.tut.fi/~jkorpela/html/alt.html#tooltip**), по ошибке интерпретируя его как всплывающую подсказку (tooltip).

Через несколько часов после того, как Тони отправил свое сообщение, на это сообщение ответил Тим Бернерс-Ли (Tim Berners-Lee):

Я предполагал следующее представление:

```
<a name=fig1 href="fghjkdfghj" REL="EMBED, PRESENT">Figure </a>
```

где соотношение между значениями обозначало бы следующее:

`EMBED`	Встроить сюда при представлении
`PRESENT`	Отобразить, если представлен исходный документ

Обратите внимание, что вы можете использовать различные комбинации, и, если браузер не поддерживает ни одной из этих комбинаций, он не "падает".

[Я] вижу, что использование этого метода для выбираемых значков означает вставку "якорей" (anchors). Хм. Но я не хотел вводить специальный тег".

Это предложение принято не было, но атрибут `rel` по-прежнему существует.

Затем Джим Дэвис (Jim Davis) добавил:

"Было бы неплохо, если бы существовал способ указывать тип содержимого, например:

```
<IMG HREF="http://nsa.gov/pub/sounds/gorby.au" CONTENT-TYPE=
audio/basic>
```

Но меня вполне удовлетворит и жизнь с требованием того, чтобы я самостоятельно указывал тип по расширению имени файла".

---

[1] См. **http://www.w3.org/Protocols/HTTP/1.0/draft-ietf-http-spec.html**, **http://www8.org/w8-papers/5c-protocols/key/key.html**. — *Прим. перев.*

Это предложение тоже не было реализовано, но впоследствии Netscape добавил поддержку для мультимедийных объектов с помощью элемента `<embed>`.

Джей Си Вебер (Jay C. Weber) спросил:

"Хотя изображения имеют приоритет в моем списке пожеланий предпочитаемых типов информации для WWW-браузера, я не думаю, что мы должны добавлять специфические обработчики для характерных типов информации по одному за раз. Куда подевался весь энтузиазм по поводу использования механизма типов MIME?"

Марк Андреессен ответил на это сообщение так:

"Это не замена ожидаемому применению MIME как стандартного механизма для работы с документами; это всего лишь простая и необходимая реализация функциональной возможности, которая нужна независимо от MIME".

Джей Си Вебер (Jay C. Weber) на это отвечал:

"Давайте на время забудем о MIME, если это затемняет суть вопроса. Я возразил против того, чтобы начинать дискуссию о том, "как мы собираемся поддерживать встроенные изображения", а не о том, "как мы собираемся поддерживать встроенные изображения в различных средах распространения информации".

В противном случае, на следующей неделе кто-нибудь предложит новый тег "`<AUD SRC="file://foobar.com/foo/bar/blargh.snd">`" для звуковой информации.

Не нужно тратить слишком много усилий на то, что можно обобщить".

С учетом ретроспективного взгляда, складывается впечатление, что доводы Джея были хорошо обоснованы. Конечно, времени понадобилось больше недели, но, в конце концов, в HTML5 элементы `<video>` и `<audio>` действительно были все-таки добавлены.

Отвечая на изначальное сообщение Джея, Дэйв Рэггетт (Dave Raggett) заметил:

"Естественно, вы правы! Я хотел рассмотреть весь диапазон всевозможных типов изображений/штриховой графики, вместе с возможностью согласования форматов. Важное значение имеет и замечание Тима о возможности поддержки областей с возможностью щелчков мышью в пределах изображений".

Впоследствии, в 1993 году, Дэйв Рэггетт (Dave Raggett) предложил HTML+ (**http://www.w3.org/MarkUp/HTMLPlus/htmlplus_1.html**) как дальнейшее развитие стандарта HTML. Это предложение реализовано не было, и впоследствии было заменено спецификацией HTML 2.0 (**http://www.w3.org/MarkUp/html-spec/html-spec_toc.html**). HTML 2.0 представляла собой "ретроспецификацию", в том смысле, что она просто формализовала те функции, которые использовались уже повсеместно **(http://www.w3.org/MarkUp/html-spec/html-spec_1.html#SEC1.1**). Эта спецификация сводила воедино, проясняла и формализовала набор функциональных возможностей, который приблизительно соответствовал возможностям спецификации HTML, в том виде, в котором она широко использовалась до начала июня 1994 года.

Впоследствии Дэйв написал спецификацию HTML 3.0 **(http://www.w3.org/MarkUp/html3/CoverPage.html**), которая основывалась на его раннем черновике

HTML+. За пределами собственной реализации W3C, Arena (**http://www.w3.org/Arena/**), спецификация HTML 3.0 так и не была никогда воплощена в жизнь, а затем была вытеснена HTML 3.2 (**http://www.w3.org/MarkUp/Wilbur/**), еще одной "ретро" спецификацией. Спецификация HTML 3.2 добавляла такие широко распространенные функции, как таблицы (tables), апплеты (applets) и обтекание текста вокруг изображений, при одновременном обеспечении обратной совместимости с существующим стандартом HTML 2.0 (**http://www.w3.org/TR/REC-html32.html#intro**).

Впоследствии Дэйв выступил одним из соавторов HTML 4.0 (**http://www.w3.org/TR/html4/**), разработал HTML Tidy (**http://tidy.sourceforge.net/**), а также принял участие в работе над XHTML, XForms, MathML и множеством других современных спецификаций W3C.

Если вернуться к нашей ретроспективе, то в 1993 Марк ответил Дэйву:

"На самом деле, возможно, нам следовало бы задуматься об общецелевом процедурном языке обработки графики, с помощью которого мы могли бы встраивать произвольные гиперссылки, указывающие на значки (icons), изображения (images), текст или что угодно другое. Кто-нибудь видел возможности Intermedia, демонстрируемые данным проектом в этом отношении?"

Intermedia (**http://en.wikipedia.org/wiki/Intermedia_%28hypertext%29**) представляла собой гипертекстовый проект, разработанный в Брауновском университете (**http://tinyurl.com/2g89g6q**). Разработка проекта велась с 1985 по 1991 г., приложение работало на A/UX (**http://en.wikipedia.org/wiki/A/UX**, **http://ru.wikipedia.org/wiki/A/UX**)[1], UNIX-подобной операционной системе, предназначенной для ранних компьютеров Macintosh.

Идея "процедурного языка обработки графики общего назначения" со временем набрала популярность. Современные браузеры поддерживают как SVG (Scalable Vector Graphics, декларативный язык разметки со встроенными возможностями поддержки скриптов, см. **http://www.w3.org/Graphics/SVG/**), так и элемент `<canvas>` (процедурно-ориентированный графический API прямого режима, см. **http://www.whatwg.org/specs/web-apps/current-work/multipage/the-canvas-element.html#the-canvas-element**), хотя последний начал свое существование как патентованное расширение (см. **http://ln.hixie.ch/?start=1089635050&count=1**), и только потом был включен в "ретроспецификацию" рабочей группой WHATWG (**http://www.whatwg.org/**).

На это замечание ответил Билл Янссен (Bill Janssen):

"Другие системы, на которые следовало бы посмотреть и которые имеют эту ценную возможность, это Andrew и Slate. Система Andrew построена на основе типов `_insets_`, каждый из которых имеет некоторый интересный тип, например, текст (text), растровое изображение (bitmap), рисунок (drawing), анимация (animation), сообщение (message), электронная таблица (spreadsheet) и т. д. Присутствует понятие произвольного рекурсивного встраивания, так что встроить

[1] Интересующимся читателям можно дополнительно порекомендовать следующие ссылки: **http://www.aux-penelope.com/**, **http://www.floodgap.com/retrotech/os/aux/**, **http://christtrek.dyndns.org:8000/doc/aux/faq.html**. — *Прим. перев.*

в документ можно вставку любого другого вида, который поддерживает встраивание. Например, вставку можно добавить в любом месте текста или текстового виджета (widget), или в любую прямоугольную область графического виджета, или же в ячейку электронной таблицы".

В данном случае под "Andrew" понимается система пользовательского интерфейса Andrew (Andrew User Interface System, **http://www-2.cs.cmu.edu/~AUIS/**), хотя в те времена она была широко известна под названием "проект Эндрю" (Andrew Project, см. **http://en.wikipedia.org/wiki/Andrew_Project**).

Тем временем, Томас Файн (Thomas Fine) высказал другую идею:

"Вот мое мнение. Наилучшим способом встраивания изображений в документы WWW является использование MIME. Я уверен, что PostScript уже является одним из поддерживаемых типов MIME, и этот формат очень хорошо справляется со смешанной обработкой текста и графики.

Но вы возражаете, что он не кликабелен? Да, вы правы. Я подозреваю, что ответ на этот вопрос уже дан Display PostScript. Даже если дополнение к стандартному PostScript и не тривиально. Определите "якорную" команду, которая указывала бы URL и использовала текущий путь как замкнутый регион для кнопки. Поскольку postscript отлично работает с путями, это делает тривиальной задачу создания кнопок произвольной формы".

Display PostScript (**http://en.wikipedia.org/wiki/Display_PostScript**) представлял собой экранную технологию визуализации, совместно разработанную Adobe и NeXT (**http://en.wikipedia.org/wiki/NeXT**, **http://ru.wikipedia.org/wiki/NeXT**).

Это предложение никогда не было реализовано, но идея о том, что лучший способ исправить проблемы с HTML заключается в том, чтобы полностью заменить его чем-то другим, по-прежнему время от времени всплывает (см., например: **http://dbaron.org/log/20090707-ex-html**).

Второго марта 1993 года Тим Бернерс-Ли выступил со следующим комментарием:

"HTTP2 позволяет документу содержать любой тип, о котором пользователь заявляет, что его можно обрабатывать, а не только зарегистрированные типы MIME. Поэтому можно экспериментировать. Да, я считаю, что есть повод рассмотреть PostScript с гипертекстом. Я не знаю, достаточно ли возможностей имеется у Display PostScript. Я знаю, что в Adobe пытаются создать собственный "PDF" на базе PostScript, который будет иметь ссылки и будет читаться их собственными патентованными средствами просмотра.

Я думаю, что общий язык вышележащего уровня (может быть, основанный на HyTime?) мог бы помочь стандартам гипертекста и графики/видео развиваться отдельно, и это помогло бы и тем, и другим.

Пусть тег `IMG` будет играть роль `INCLUDE`, и пусть он ссылается на документ произвольного типа. Или пусть это будет `EMBED`, если `INCLUDE` звучит слишком похоже на директиву `include` из cpp, вследствие чего у людей будет складываться впечатление, что исходный код SGML будет разбираться встраиваемо (inline) — так как это не то, что подразумевается разработчиками".

HyTime (**http://www.hytime.org/**) представлял собой раннюю гипертекстовую систему документов, основанную на базе SGML (Standard Generalized Markup

Language, см. **http://en.wikipedia.org/wiki/Standard_Generalized_Markup_Language**, **http://www.w3.org/MarkUp/SGML/**). Она часто маячила в ранних дискуссиях, посвященных HTML и, впоследствии — XML.

Предложение Тима о теге `<INCLUDE>` никогда не было реализовано, хотя его отголоски можно обнаружить в элементах `<object>`, `<embed>` и `<iframe>`.

Наконец, 12 марта 1993 года Марк Андреессен вновь вернулся к этой ветви дискуссии:

> "Возвращаясь к обсуждению встраиваемого изображения — я склоняюсь к выпуску Mosaic v. 0.10, где будут поддерживаться инлайновые изображения/растры в форматах GIF и XBM, как отмечалось ранее.
>
> […]
>
> На данном этапе мы не готовы поддерживать `INCLUDE`/`EMBED`. … Поэтому, вероятно, мы удовлетворимся `<IMG SRC="url">` (не `ICON`, потому что не все встраиваемые изображения можно назвать "значками"). На текущий момент, встроенные изображения не будут иметь явных типов содержимого; впоследствии мы планируем обеспечить такую поддержку (хотя и с общей адаптацией MIME). Фактически, процедуры чтения изображений, которые мы используем сейчас, определяют формат изображения "на лету", так что даже расширение имени файла не будет иметь значения".

Непрерывная линия

Я не перестаю восхищаться всеми аспектами данной дискуссии, продолжающейся уже более 17 лет и приведшей к созданию элемента HTML, который применяется практически на каждой Web-странице, которая когда-либо была опубликована. Вот посмотрите:

- HTTP до сих пор существует. HTTP успешно развивался с версии 0.9 до версии 1.0 и, наконец, до версии 1.1. И в настоящее время он все еще продолжает развиваться[1].
- HTML до сих пор существует. Этот рудиментарный формат данных, который изначально даже не поддерживал встроенных изображений, успешно развивался, со временем вышли его версии 2.0, 3.2, 4.0. HTML развивается по непрерывной линии. Да, это запутанная и извилистая линия, но она непрерывна. Существует множество ответвлений от этой общей линии, множество "тупиковых" ветвей эволюционного дерева, множество направлений, которые люди, поставившие себе целью разработку стандартов, обошли, опережая сами себя (а также авторов реализаций). И тем не менее, линия развития HTML не пресекалась никогда. Эта книга была написана в 2010 году, но современные браузеры

[1] См. **http://datatracker.ietf.org/wg/httpbis/charter/**.

по-прежнему отображают Web-страницы, датирующиеся 1990 годом[1]! Например, я только что загрузил одну из таких страниц на свой мобильный телефон Android, и мне даже не было выведено сообщения, предлагающего подождать, пока завершится обработка старого формата…

- Язык HTML всегда развивался в ходе дискуссии между разработчиками браузеров, авторами Web-страниц, разработчиками стандартов и другими людьми, которые сначала заинтересовались этим "разговором об угловых скобках", а потом и включились в него сами. Большинство успешных версий HTML представляют собой "ретроспецификации", стремящиеся одновременно актуализировать все то, что к моменту их выхода стало фактическим стандартом, и одновременно задать правильное направление дальнейшего развития. Любой, кто скажет вам о том, что необходимо "блюсти чистоту" HTML (предположительно, игнорируя интересы разработчиков браузеров, авторов Web-страниц или и тех, и других), на самом деле просто недостаточно хорошо или даже неверно информирован. HTML никогда не был "чистым", и все попытки его "очистить" неизменно заканчивались грандиозными провалами, масштабы которых сравнимы только с масштабами фиаско, которое неизменно терпели все попытки вообще заменить HTML чем-нибудь другим.
- Ни один из браузеров, появившихся с момента 1993 года, не существует в своем первозданном виде. Работа над проектом Netscape Navigator была прекращена в 1998 году (**http://en.wikipedia.org/wiki/History_of_Mozilla_Application_Suite#Open_sourcing_of_Communicator**), после чего исходный код этого браузера был полностью переработан и фактически "переписан с нуля", в результате чего появился новый проект — Mozilla Suite, от которого впоследствии отпочковался проект Firefox (**http://en.wikipedia.org/wiki/History_of_Mozilla_Firefox)**. Internet Explorer начинал свой жизненный путь достаточно скромно под названием "Microsoft Plus! for Windows 95", где он был упакован в единый пакет с рядом тем оформления рабочего стола и игрой Pinball. Но естественно, что развитие этого браузера тоже можно проследить вплоть до самых истоков (см., например: **http://en.wikipedia.org/wiki/Spyglass_Mosaic**).
- Некоторые операционные системы, датируемые 1993 годом, все еще существуют, но уже ни одна из них не идет в ногу со временем с современной WWW. Большинство пользователей современной "всемирной паутины" подключаются к Интернету с современных ПК, работающих под управлением Windows 2000 или более новой версии Windows, с Macintosh, работающего под управлением Mac OS X, или с ПК, на котором установлен один из современных дистрибутивов Linux, или даже с наладонного устройства наподобие iPhone. В 1993 году доминирующей версией Windows была Windows 3.1 (которая тогда конкурировала с OS/2), компьютеры Macintosh работали под управлением System 7, а дистрибутивы Linux распространялись через Usenet. Хотите получить об этом представление? Найдите какого-нибудь убеленного сединами ветерана и заведите с ним разговор о "Trumpet Winsock" (**http://en.wikipedia.org/wiki/Winsock**,

[1] См. **http://www.w3.org/People/Berners-Lee/FAQ.html#Examples**.

http://en.wikipedia.org/wiki/Berkeley_sockets, **http://www.trumpet.com.au/index.php/downloads.html**) или "MacPPP" (**http://www.index-site.com/macppp.html**).

- Некоторые или даже *одни и те же* люди до сих пор занимаются тем, что мы сейчас называем "стандартами Web". И это после 20 лет развития! Причем некоторые из них принимали участие в разработке предшественников HTML, датируемых 1980 годом, и даже более ранними датами.
- Если же говорить о предшественниках… С учетом роста популярности HTML и Web, несложно просто забыть о тех форматах и системах, которые стояли у истоков. Andrew? Intermedia? HyTime? Между прочим, HyTime был стандартом ISO (International Organization for Standardization, **http://www.iso.org/iso/home.html**), а не просто одним из экспериментальных проектов, вышедших из недр исследовательской лаборатории (см. **http://xml.coverpages.org/hytime.html**). Этот стандарт был одобрен для применения в военных целях. Его разработка спонсировалась Большим Бизнесом. И прочесть об этом вы можете сами, вот здесь: **http://www.sgmlsource.com/history/hthist.htm**. Между прочим, страница превосходно откроется любым современным браузером!

Но ничто из только что сказанного не дает ответа на исходный вопрос, которым мы задались в начале этой главы: почему мы пользуемся элементом `<img>`? Почему не элементом `<icon>`? Или элементом `<include>`? Почему не использовать вместо этого атрибут `include` или некоторую комбинацию значений `rel`? Почему ни одно, ни другое, ни третье, а все-таки элемент `<img>`? Очень просто: потому что Марк Андреессен (Marc Andreessen) реализовал его, а реализованный код побеждает.

Впрочем, нельзя сказать, что *весь* реализованный код побеждает; в конце концов, Andrew, Intermedia и HyTime тоже представляли собой реализованный код. Код — это необходимое, но недостаточное условие успеха. И я *действительно* не хочу сказать, что поставки кода еще до утверждения стандарта — это наилучшее решение. Введенный Марком элемент `<img>` не требовал общего графического формата; он не определял, каким образом этот элемент будет обтекаться текстом; он не поддерживал текстовых альтернатив или резервного, замещающего контента для более старых браузеров. И теперь, 17 лет спустя, мы продолжаем бороться с согласованием контента (content sniffing)[1], и данный элемент до сих пор продолжает оставаться источником безумных уязвимостей в системе безопасности (**http://code.google.com/p/doctype/wiki/ArticleContentSniffing**). И вы можете отследить всю эволюцию и развитие событий в исторической ретроспективе, вернувшись на 17 лет назад, в эпоху Великих браузерных войн (**http://en.wikipedia.org/wiki/Browser_wars**, **http://evolt.org/node/60181/**), до того момента, когда 25 февраля 1993 года Марк Андреессен (Marc Andreessen) небрежно заметил "возможно, когда-нибудь, MIME", а после этого все равно выпустил свой код.

Но для того, чтобы что-то победило, оно обязательно должно быть реализовано.

[1] См. **http://tools.ietf.org/html/draft-abarth-mime-sniff-06**.

Шкала развития HTML с 1997 по 2004 год

В декабре 1997 года консорциум WWW (World Wide Web Consortium, W3C) опубликовал стандарт HTML 4.0 (**http://www.w3.org/TR/REC-html40-971218/**), а вскоре после этого закрыл рабочую группу HTML (HTML Working Group). Менее чем через два месяца, отдельная рабочая группа W3C опубликовала спецификацию XML 1.0 **(http://www.w3.org/TR/1998/REC-xml-19980210**). А еще через три месяца руководство W3C провело семинар под названием "формирование будущего HTML" ("Shaping the Future of HTML", см. **http://www.w3.org/MarkUp/future/**) с тем, чтобы дать общественности ответ на вопрос "Не прекратил ли консорциум W3C работу над HTML?" Ответ на данный вопрос был таким:

В ходе дискуссий было решено, что дальнейшее расширение спецификации HTML 4.0 усложнится, как усложнится и преобразование приложений HTML 4.0 в приложения XML[1]. Предлагаемый выход из этой ситуации и освобождения от этих ограничений заключается в том, чтобы начать разработку следующего поколения HTML "с чистого листа", основываясь на наборах тегов XML.

Консорциум W3C принял решение переориентировать рабочую группу HTML на выработку "семейства наборов тегов XML". В качестве первого шага, предпринятого в декабре 1998, рассматривается черновик промежуточной спецификации, которая просто переформулирует HTML в термины XML без добавления новых элементов или атрибутов (**http://www.w3.org/TR/1998/WD-html-in-xml-19981205/**). Впоследствии эта спецификация получила известность под именем "XHTML 1.0" (**http://www.w3.org/TR/xhtml1/**). Она определила новый тип MIME для документов XHTML, `application/xhtml+xml`. Однако, чтобы упростить миграцию существующих страниц HTML 4, в состав этой спецификации было включено "Приложение C" (**http://www.w3.org/TR/xhtml1/#guidelines**), которое "обобщило руководящие указания по разработке для авторов, которые желали, чтобы их документы XHTML визуализировались существующими пользовательскими агентами HTML". В "Приложении C" говорилось о том, что авторам разрешается создавать так называемые "страницы XHTML", и при этом по-прежнему обслуживать их с помощью типа MIME `text/html`.

Следующей целью разработчиков спецификации явились Web-формы. В августе 1999 года та же рабочая группа HTML опубликовала первый черновик будущего стандарта XHTML Extended Forms (**http://www.w3.org/TR/1999/WD-xhtml-forms-**

---

[1] XML (eXtensible Markup Language) — расширяемый язык разметки, рекомендованный Консорциумом Всемирной паутины, и фактически представляющий собой свод общих синтаксических правил. Подробнее см. **http://en.wikipedia.org/wiki/XML**, **http://en.wikipedia.org/wiki/List_of_XML_markup_languages**, **http://www.codenet.ru/webmast/xml/**. XML — текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки, например, XHTML (**http://en.wikipedia.org/wiki/XHTML**). XML является упрощенным подмножеством языка SGML. — *Прим. перев.*

req-19990830). Свои ожидания они сформулировали в первом же абзаце (см. **http://www.w3.org/TR/1999/WD-xhtml-forms-req-19990830#intro**):

> "После всестороннего тщательного обсуждения, рабочая группа HTML приняла решение о том, что цели форм следующего поколения несовместимы с поддержанием обратной совместимости с браузерами, разрабатывавшимися для более ранних версий HTML. Наша цель заключается в предоставлении новой модели форм ("XHTML Extended Forms") на основе набора четко определенных требований. Требования, описанные в этом документе, основываются на опыте работы с широким спектром приложений, использующих формы".

Через несколько месяцев спецификация "XHTML Extended Forms" была переименована в "XForms", и для продолжения работы над ней была образована отдельная рабочая группа (**http://www.w3.org/MarkUp/Forms/2000/Charter.html**). Эта группа работала параллельно с рабочей группой HTML и, наконец, в октябре 2003 года опубликовала первую редакцию XForms 1.0 (**http://www.w3.org/TR/2003/REC-xforms-20031014/**).

Тем временем, когда завершился переход к XML, рабочая группа HTML сосредоточила свои усилия на создании "стандарта HTML нового поколения". В мае 2001 года они опубликовали первую редакцию стандарта XHTML 1.1 (**http://www.w3.org/TR/2001/REC-xhtml11-20010531/**), которая добавляла к XHTML 1.0 лишь небольшое количество дополнительных функций, имевших второстепенное значение, а также ликвидировала "Приложение C". Начиная с версии 1.1, все документы XHTML должны были обслуживаться с помощью типа MIME `application/xhtml+xml`.

Все, что вы знаете о XHTML — неправильно

Почему типы MIME настолько важны? Почему я все время к ним возвращаюсь? В трех словах: драконовская обработка ошибок (Draconian error handling). Браузеры всегда относились к HTML в высшей степени "терпимо". Если вы, создав страницу HTML, забудете использовать тег `</head>`, браузер все равно отобразит эту страницу. (Некоторые теги неявно подразумевают конец тега `<head>` и начало тега `<body>`.) Предполагается, что вы используете иерархическое вложение тегов — закрывая их по принципу "матрешки" (сначала наружный, затем внутренний). Но, если вы создадите разметку наподобие `<b><i></b></i>`, браузеры с ней уж как-нибудь да разберутся и продолжат работу без отображения сообщения об ошибке.

Как и следовало ожидать, тот факт, что "битая" разметка HTML все-таки работала в Web-браузерах, привел к тому, что авторы массово создавали "битые" страницы HTML. Таких страниц стало очень много! По некоторым оценкам, более 99% всех страниц HTML, существующих на сегодняшний день, содержат, как минимум, одну ошибку. Но, поскольку