HTML

&CSS

Содержание

HyperText Markup Language - язык гипертекстовой разметки

HTML не является языком программирования; это *язык разметки*, используемый для определения структуры веб-страниц, посещаемых пользователями.

Они могут иметь сложную или простую структуру, всё зависит от замысла и желания веб-разработчика.

HTML состоит из ряда элементов, которые вы используете для того, чтобы охватить, обернуть или *разметить* различные части содержимого, чтобы оно имело определённый вид или срабатывало определённым способом.

Встроенные тэги могут преобразовать часть содержимого в гиперссылку, по которой можно перейти на другую веб-страницу, выделить курсивом слова и так далее.

Элемент:

- открывающий тег + закрывающий тег + содержимое = элемент.

Вы также можете вкладывать элементы внутрь других элементов — это называется **вложенностью**.
Если мы хотим подчеркнуть, что наш кот **очень** сердитый, мы можем заключить слово "очень" в элемент , который означает, что это слово крайне важно в данном контексте:
Мой кот очень сердитый.

Элементы блочного уровня и Строчные элементы.

- Существует две важных категории элементов в HTML,— элементы блочного уровня и строчные элементы.
- Элементы блочного уровня формируют видимый блок на странице — они окажутся на новой строке после любого контента, который шёл до них, и любой контент после них также окажется на новой строке.
- Чаще всего элементами блочного уровня бывают структурные элементы страницы, представляющие собой, например, параграфы (абзацы), списки, меню навигации, футеры, или подвалы, и т. п.
- Элементы блочного уровня не вкладываются в строчные элементы, но иногда могут вкладываться в другие элементы блочного уровня.

Пример

```
<p>четвёртый</p><p>пятый</p><p>шестой</p>
```

четвёртый

пятый

шестой

Строчные элементы

— это те, которые содержатся в элементах блочного уровня и окружают только малые части содержимого документа, не целые абзацы и группировки контента.

Строчные элементы не приводят к появлению новой строки в документе: они обычно встречаются внутри абзаца текста, например, элемент <a> (ссылка) или акцентирующие элементы вроде или .

Пустые элементы

- Не все элементы соответствуют вышеупомянутому шаблону: открывающий тег, контент, закрывающий тег.
- Некоторые элементы состоят из одного тега и обычно используются для вставки чего-либо в то место документа, где размещены.
- Например, элемент <img> вставляет картинку на страницу в том самом месте, где он расположен:

```
<img
  src="https://raw.githubusercontent.com/mdn/beginner-html-site/gh-pages/images/firefox-icon.png">
```

- Пустые элементы иногда называют *void-элементами*.

Атрибуты

- У элементов также могут быть атрибуты, которые выглядят так:
- Атрибуты содержат дополнительную информацию об элементе, которая, по вашему мнению, не должна отображаться в содержимом элемента.

- В данном случае атрибут class позволяет вам дать элементу идентификационное имя, которое в дальнейшем может быть использовано для обращения к элементу с информацией о стиле и прочими вещами.

Атрибут должен иметь:

- **Пробел между атрибутом и именем элемента (или предыдущим атрибутом, если у элемента уже есть один или несколько атрибутов).**
- **Имя атрибута и следующий за ним знак равенства.**
- **Значение атрибута, заключённое в кавычки**.

Элемент HTML <a> определяет гиперссылку для перехода на определённое место на странице или на другую страницу в Интернете.

Также он может быть использован (в устаревшем варианте) для создания якоря — это место назначения для гиперссылок внутри страницы: так ссылки не ограничены только в перемещении между страницами.

MDN

href: В значении этого атрибута прописывается веб-адрес, на который, по вашей задумке, должна указывать ссылка, куда браузер переходит, когда вы по ней кликаете.

title: Атрибут title описывает дополнительную информацию о ссылке, такую как: на какую страницу она ведёт. Например, title="The Mozilla homepage". Она появится в виде всплывающей подсказки, когда вы наведёте курсор на ссылку.

target: Атрибут target определяет контекст просмотра, который будет использоваться для отображения ссылки. Например, target="_blank" отобразит ссылку на новой вкладке. Если вы хотите отобразить ссылку на текущей вкладке, просто опустите этот атрибут.

Булевые атрибуты

Иногда вы будете видеть атрибуты, написанные без значения — это совершенно допустимо.

Такие атрибуты называются булевые, и они могут иметь только одно значение, которое в основном совпадает с его именем.

В качестве примера возьмём атрибут disabled, который можно назначить для формирования элементов ввода, если вы хотите, чтобы они были отключены (неактивны), так что пользователь не может вводить какие-либо данные в них.

```
<input type="text" disabled="disabled">
```

Для краткости совершенно допустимо записывать их следующим образом:

```
<input type="text" disabled>
```

```
<input type="text">
```

Опускание кавычек вокруг значений атрибутов

Это допустимо при определённых условиях, но разрушит вашу разметку при других.

Например, с гиперссылкой, мы можем написать основной вариант только с атрибутом href так:

любимый веб-сайт

Однако, как только мы добавим атрибут title в таком же стиле, мы поступим неверно:

favorite website

В этом месте браузер неверно истолкует вашу разметку, думая, что атрибут title — это на самом деле три разных атрибута — атрибут title со значением "The" и два булевых атрибута: Mozilla и homepage.

Это, очевидно, не то, что имелось в виду, и приведёт к ошибке или неожиданному поведению кода.

Одинарные или двойные кавычки?

Это исключительно дело вкуса, и вы можете свободно выбирать, какие из них предпочитаете. Обе следующие строки эквивалентны:

Ссылка к моему примеру.

Ссылка к моему примеру.

Однако вы должны убедиться, что не смешиваете их вместе. Следующее будет неверным!

Ссылка к моему примеру.

Если вы используете один тип кавычек в своём HTML, то вы можете поместить внутрь их кавычки другого типа, не вызывая никаких проблем:

A link to my example.

Структура HTML документа

Ниже дан пример оборачивания основных, самостоятельных HTML-элементов, которые сами по себе не очень полезны.

Давайте посмотрим, как самостоятельные элементы объединяются для формирования всей HTML страницы:

```
<!DOCTYPE html>
 <html>
 <head>
<meta charset="utf-8">
 <title>Тестовая страница</title>
 </head>
<body>
<p>Это — моя страница</p>
</body>
 </html>
```

- <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Пробелы в HTML

Следующие два примера эквивалентны:

Некто говорит, что лжет.

Некто говорит,
что лжет.

Не важно, сколько пустого места вы используете в разметке (что может включать пробелы и сдвиги строк): браузер при анализе кода сократит всё пустое место до одного пробела.

Зачем использовать много пробелов? Ответ: это доступность для понимания — гораздо легче разобраться, что происходит в вашем коде, если он удобно отформатирован, а не просто собран вместе в одном большом беспорядке.

В нашем коде каждый вложенный элемент сдвинут на два пробела относительно элемента, в котором он находится. Вы можете использовать любое форматирование (в частности, количество пробелов для отступа), но лучше придерживаться одного стиля.

Ссылки на сущности: Включение специальных символов в HTML

В HTML символы <, >, ", ' и & являются специальными. Они являются частью самого синтаксиса HTML.

Так как же включить в текст один из этих специальных символов?

Например, если вы хотите использовать амперсанд или знак «меньше» и не интерпретировать его как код.

Мы должны использовать ссылки-мнемоники — специальные коды, которые отображают спецсимволы, и могут быть использованы в необходимых позициях.

Каждая ссылка-мнемоник начинается с амперсанда (&) и завершается точкой с запятой (;).

ссылки-мнемоники

Буквенный символ	Символьный эквивалент
<	&lt;
>	&gt;
"	&quot;
'	&apos;
&	&amp;

HTML комментарии

В HTML, как и в большинстве языков программирования, есть возможность писать комментарии в коде.

Комментарии игнорируются обозревателем и не видны пользователю, их добавляют для того, чтобы пояснить, как работает написанный код, что делают отдельные его части и т. д.

Такая практика полезна, если вы возвращаетесь к коду, который давно не видели или когда хотите передать его кому-то другому.

Чтобы превратить часть содержимого HTML-файла в комментарий, нужно поместить её в специальные маркеры

<!-- и -->,

например:

```
<p> Меня нет в комментариях </p>
<!-- <p>А теперь есть!</p> -->
```

Применение цвета к HTML-элементам с помощью CSS

С помощью CSS, существует множество способов присвоить цвет HTML элементам, чтобы придать им желаемый вид.

К счастью, присвоить цвет к HTML-элементу очень просто, и это можно сделать практически со всеми элементами.

На уровне элементов HTML, всему можно присвоить цвет. С точки зрения отдельных составляющих элементов, таких как текст, границы и т.д., существует ряд свойств CSS, с помощью которых можно присвоить цвет.

На фундаментальном уровне, свойство color (en-US) определяет цвет текста HTML-элемента, а свойство background-color - цвет фона элемента.

Они работают практически для всех элементов.

Текст

Эти свойства используются для определения цвета текста, его фона и любого оформления текста:

- color (en-US)Свойство color применяется к тексту и любому оформлению текста, например: подчёркивание, линии на текстом, перечёркивание и т.д.
- background-colorЦвет фона текста.
- text-shadowДобавляет и устанавливает параметры тени для текста. Один из параметров тени - это основной цвет, который размывается и смешивается с цветом фона на основе других параметров.
- text-decoration-color (en-US)По умолчанию, элементы оформление текста (подчёркивание, перечёркивание) используют цвет свойства color. Но вы можете присвоить другой цвет с помощью свойства text-decoration-color.
- text-emphasis-color (en-US)Цвет, который используется для выделения диакритических знаков, прилегающих к каждому текстовому символу. Это свойство используется преимущественно для восточноазиатских языков.
- caret-color (en-US)Цвет, который используется для каретки (caret (en-US)) (курсора ввода текста). Применимо только к редактируемым элементам, таким как <input> и <textarea> (en-US) или элементам , для которых установлен атрибут contenteditable.

CSS

CSS (каскадные таблицы стилей) используется для стилизации и компоновки веб-страниц - например, для изменения шрифта, цвета, размера и интервала содержимого, разделения его на несколько столбцов или добавления анимации и других декоративных элементов.

- **CSS** (Cascading Style Sheets, или каскадные таблицы стилей) - это декларативный язык, который отвечает за то, как страницы выглядят в веб браузере. CSS стили содержат свойства и их значения, которые и определяют, как будет выглядеть сайт.
- CSS одна из ключевых Web технологий, наряду с HTML и JavaScript. Как правило CSS используется для определения стилей HTML-элементов, но также может быть применён совместно с другими языками разметки, такими как SVG или XML.
- CSS-правило состоит из селектора и набора свойств (en-US) с их значениями. :

В этом примере все HTML параграфы будут иметь текст жёлтого цвета на чёрном фоне

```
/* Селектор "p" означает, что данное правило будет применено ко
    всем параграфам в документе */
p {
/* Свойство "color" определяет цвет текста, в данном случае
    желтый. */
color: yellow;
 /* Свойство "background-color" определяет цвет фона элемента, в
    данном
 случае черный. */
 background-color: black;
 }
```

"Каскадность" CSS - это правила, которые регулируют приоритет селекторов при отображении внешнего вида элементов страницы. Это очень важная особенность, поскольку сложный веб-сайт может иметь тысячи CSS-селекторов.

Добавление CSS в наш документ

Самое первое, что нам нужно сделать, — это сообщить HTML-документу, что у нас есть некоторые правила CSS, которые мы хотим использовать.

Существует три различных способа применения CSS к документу HTML, с которым вы обычно сталкиваетесь, однако сейчас мы рассмотрим наиболее обычный и полезный способ сделать это — связать CSS с заголовком вашего документа.

Создайте файл в той же папке, что и документ HTML, и сохраните его как styles.css.

Расширение .css показывает, что это файл CSS.

Чтобы связать styles.css с index.html, добавьте следующую строку где-то внутри<head> HTML документа:

```
<link rel="stylesheet" href="styles.css">
```

Элемент <link> сообщает браузеру, что у нас есть таблица стилей, используя атрибут *rel*, и местоположение этой таблицы стилей в качестве значения атрибута *href.* вы можете проверить, работает ли CSS, добавив правило в *styles.css*.

Стилизация HTML-элементов

Делая наш заголовок красным, мы уже продемонстрировали, что можем нацеливать и стилизовать элемент HTML.

Мы делаем это путём нацеливания на элемент *selector* — это селектор, который напрямую соответствует имени элемента HTML.

Чтобы нацелиться на все абзацы в документе, вы должны использовать селектор p. Чтобы сделать все абзацы зелёными, вы должны использовать:

```
p { color: green; }
```

Вы можете выбрать несколько селекторов одновременно, разделив их запятыми. Если я хочу, чтобы все параграфы и все элементы списка были зелёными, моё правило выглядит так:

```
p, li { color: green; }
```

Изменение поведения элементов по умолчанию

- Когда мы смотрим на хорошо размеченный HTML-документ, мы можем увидеть, как браузер делает HTML читаемым, добавив некоторые стили по умолчанию.
- Это происходит потому, что в браузерах есть внутренние таблицы стилей, содержащие стили по умолчанию, которые по умолчанию применяются ко всем страницам; без них весь текст работал бы вместе, и мы должны были бы стилизовать всё с нуля. Все современные браузеры по умолчанию отображают HTML-контент практически одинаково.
- Однако вам часто захочется что-то другое, кроме выбора, сделанного браузером.
- Это можно сделать, просто выбрав элемент HTML, который вы хотите изменить, и используя правило CSS, чтобы изменить его внешний вид. Хорошим примером является наш <li> — упорядоченный список. Он добавляет маркеры, и если я решу, что я не хочу эти маркеры, я могу удалить их вот так:
- li { list-style-type: none; }

Добавление класса

Пока у нас есть стилизованные элементы, основанные на их именах HTML-элементов. Это работает до тех пор, пока вы хотите, чтобы все элементы этого типа в вашем документе выглядели одинаково.

В большинстве случаев это не так, и вам нужно будет найти способ выбрать подмножество элементов, не меняя остальные.

Самый распространённый способ сделать это — добавить класс к вашему HTML-элементу и нацелиться на этот класс.

В своём HTML-документе добавьте Атрибут class ко второму пункту списка.

Ваш список теперь будет выглядеть так:

```
<ul>
 <li>Элемент один</li>
<li class="special">Элемент два</li>
<li>Элемент <em>три</em></li>
</ul>
```

Использование класса

Иногда вы увидите правила с селектором, который перечисляет селектор HTML-элемента вместе с классом:

```
li.special
{ color: orange; font-weight: bold; }
```

Этот синтаксис означает «предназначаться для любого элемента li, который имеет класс special».

Если бы вы сделали это, вы бы больше не смогли применить класс к или другому элементу, просто добавив к нему класс; вы должны добавить этот элемент в список селекторов:

```
li.special, span.special { color: orange; font-weight: bold; }
```

Поэтому иногда лучше обойти элемент и просто обратиться к классу, если только вы не знаете, что хотите создать некоторые специальные правила для одного элемента и, возможно, хотите убедиться, что они не применяются к другим элементам.

Стилизация элементов на основе их расположения в документе

Есть моменты, когда вы хотите, чтобы что-то выглядело иначе, в зависимости от того, где оно находится в документе.

Здесь есть несколько селекторов, которые могут вам помочь, но сейчас мы рассмотрим только пару.

В нашем документе два элемента — один внутри абзаца, а другой внутри элемента списка. Чтобы выбрать только который вложен в элемент <li>, я могу использовать селектор под названием **descendant combinator (комбинатор-потомок)**, который просто принимает форму пробела между двумя другими селекторами.

Добавьте следующее правило в таблицу стилей.

```
li em { color: rebeccapurple; }
```

Этот селектор выберет любой элемент , который находится внутри (потомка) .

Итак, в вашем примере документа вы должны найти, что в третьем элементе списка теперь фиолетовый, но тот, который находится внутри абзаца, не изменился.

Стилизация элементов на основе их расположения в документе

- Ещё можно попробовать стилизовать абзац, когда он идёт сразу после заголовка на том же уровне иерархии в HTML.
- Для этого поместите + (**соседний братский комбинатор**) между селекторами.
- Попробуйте также добавить это правило в таблицу стилей:
- h1 + p { font-size: 400%; }

Стилизация элементов на основе состояния

Прямым примером этого является стиль ссылок.

Когда мы создаём ссылку, мы должны нацелить элемент <a> (якорь). Он имеет различные состояния в зависимости от того, посещается ли он, посещается, находится над ним, фокусируется с помощью клавиатуры или в процессе нажатия (активации).

Вы можете использовать CSS для нацеливания на эти разные состояния — CSS-код ниже отображает невидимые ссылки розового цвета и посещённые ссылки зелёного цвета.

```
a:link { color: pink; }
 a:visited { color: green; }
```

Вы можете изменить внешний вид ссылки, когда пользователь наводит на неё курсор, например, удалив подчёркивание, что достигается с помощью следующего правила:

```
a:hover { text-decoration: none; }
```

Сочетание селекторов и комбинаторов

Стоит отметить, что вы можете комбинировать несколько селекторов и комбинаторов вместе. Вот пример:

/* выбирает любой внутри <p>, который находится внутри <article> */

```
article p span { ... }
```

/* выбирает любой <p>, который идёт сразу после <ul>, который идёт сразу после <h1> */

```
h1 + ul + p { ... }
```

Вы также можете комбинировать несколько типов вместе. Попробуйте добавить следующее в ваш код:

```
body h1 + p .special { color: yellow; background-color: black; padding: 5px; }
```

Это будет стиль любого элемента с классом special, который находится внутри <p>, который приходит сразу после <h1>, который находится внутри <body>.

Что внутри "head"? Метаданные в HTML

Элемент head HTML-документа не отображается на странице в веб-браузере. Он содержит такую информацию, как:

заголовок (title) страницы

ссылки на файлы CSS (если вы хотите применить к вашему HTML стили CSS)

ссылки на иконки

другие метаданные (данные о HTML: автор и важные ключевые слова, описывающие документ.)

Пример:

```
<head>
 <meta charset="utf-8">
<title>Моя тестовая страница</title>
 </head>
```

- Однако на больших страницах блок <head> может быть довольно объёмным. Попробуйте зайти на какие-нибудь из ваших любимых сайтов и посмотреть содержимое <head> с помощью инструментов разработчика.

Название страницы (title)

Элемент <title>: используют для добавления заголовка (названия страницы) в документ.

Элемент <h1> (en-US) тоже иногда называют заголовком страницы.

Но это разные вещи!

Элемент <h1> (en-US) виден на странице, открытой в браузере, — его используют **один раз на странице**, чтобы выделить название содержимого.

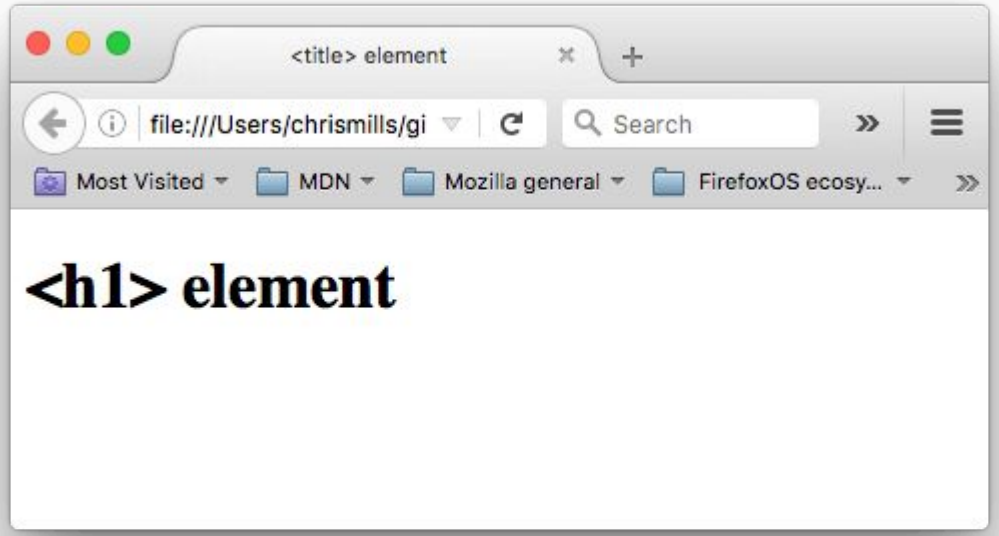
Это может быть название истории, заголовок новости или что-то в этом роде.

Элемент <title> — метаданные, название всего HTML-документа, а не заголовок внутри его содержимого.

Заголовки

Код:

```
<!DOCTYPE html>
<html>
 <head>
 <meta charset="utf-8">
 <title>&lt;title&gt;
    element</title>
</head>
 <body>
<h1>&lt;h1&gt;
    element</h1>
 </body>
</html>
```



Содержимое элемента `<title>` используется и в других местах. Например, при добавлении страницы в избранное (*Bookmarks* > *Bookmark This Page* в Firefox), текст из `<title>` предлагается в качестве названия закладки.

Метаданные: Элемент <meta>

Метаданные — данные, которые описывают данные.

У HTML есть «официальное» место для метаданных документа — элемент <meta>. Существует множество разновидностей <meta>

<meta charset="utf-8" >

В этом элементе указана кодировка документа — набор символов, которые в нём можно использовать . utf-8 — универсальный набор символов, который включает почти все символы со всех языков человечества.

Такая веб-страница сможет работать с любым языком. Установить эту кодировку на всех веб-странице, которые вы создаёте — отличная идея!

Страница в такой кодировке прекрасно отображает как английские, так и японские символы:

Указываем автора и описание

У элементов <meta> часто есть атрибуты name и content:

name — тип элемента, то есть какие именно метаданные он содержит.

content — сами метаданные.

Два полезных элемента метаданных — указание автора страницы и краткое описание её содержимого.

Рассмотрим эти элементы на примере:

<meta name="author" content="Крис Миллс">

<meta name="description" content="Задача MDN — в том, чтобы обучить новичков всему тому, что нужно им для разработки веб-сайтов и приложений.”>

По указанному имени автора (author) можно найти человека, который написал страницу, и связаться с ним.

Некоторые системы управления содержимым (CMS) автоматически обрабатывают эту информацию и делают её доступной для таких целей.

Краткое описание (description) содержимого страницы учитывается поисковыми системами при совпадении ключевых слов. Такое называют поисковой оптимизацией, или SEO.

Другие виды метаданных

В сети вы найдёте также другие типы метаданных. Многие из них — это собственные форматы, созданные для предоставления определённым сайтам (например, социальных сетей) специальной информации, которую они могут использовать.

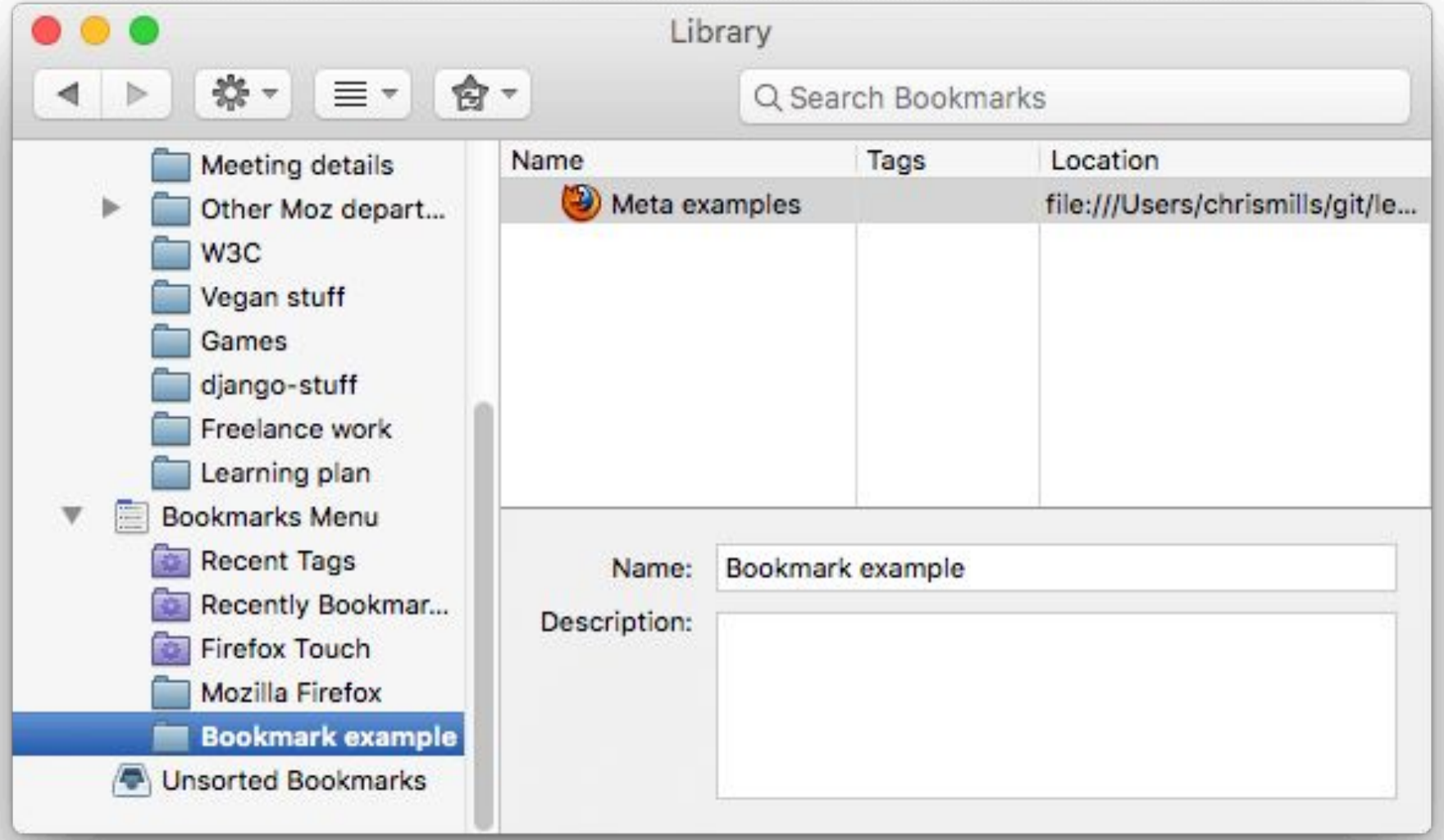
Например, Протокол Open Graph создан Facebook чтобы предоставить сайтам дополнительные возможности использования метаданных.

У Twitter также есть собственный формат метаданных, с помощью которого создаётся аналогичный эффект, при отображении URL сайта на twitter.com:

```
<meta name="twitter:title" content="MDN Web Docs">
```

Добавление иконок

- Чтобы добавить своему сайту узнаваемости, можно указать в метаданных разные иконки.
- Favicon, один из старожилов интернета, стал первой из таких иконок.
- Браузеры показывают её в заголовке вкладки и в списке избранных страниц.
- Чтобы добавить на страницу favicon:
- Сохраните изображение в формате .ico (многие браузеры поддерживают и в более привычных форматах, таких как .gif или .png) в папку со своим документом.
- Старые браузеры, например, Internet Explorer 6, поддерживают только формат .ico
- Добавьте ссылку на иконку в <head> документа:
- <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">


Library
Search Bookmarks
Meeting details
Other Moz depart...
W3C
Vegan stuff
Games
django-stuff
Freelance work
Learning plan
Bookmarks Menu
Recent Tags
Recently Bookmar...
Firefox Touch
Mozilla Firefox
Bookmark example
Unsorted Bookmarks
Name
Tags
Location
Meta examples
file:///Users/chrismills/git/le...
Name:
Bookmark example
Description:

link element

Content attributes:
Global attributes
`href` — Address of the hyperlink
`crossorigin` — How the element handles crossorigin requests
`rel` — Relationship between the document containing the hyperlink and the destination resource
`media` — Applicable media
`integrity` — Integrity metadata used in *Subresource Integrity* checks [SRI]
`hreflang` — Language of the linked resource
`type` — Hint for the type of the referenced resource
`referrerpolicy` — Referrer policy for fetches initiated by the element
`sizes` — Sizes of the icons (for `rel`="`icon`")
`imagesrcset` — Images to use in different situations, e.g., high-resolution displays, small monitors, etc. (for `rel`="`preload`")
`imagesizes` — Image sizes for different page layouts (for `rel`="`preload`")
`as` — Potential destination for a preload request (for `rel`="`preload`" and `rel`="`modulepreload`")
`color` — Color to use when customizing a site's icon (for `rel`="`mask-icon`")
`disabled` — Whether the link is disabled
Also, the `title` attribute has special semantics on this element: Title of the link; CSS style sheet set name.

Глобальные атрибуты

- href — Адрес кроссоригина гиперссылки — Как элемент обрабатывает запросы кроссоригина
- rel — Связь между документом, содержащим гиперссылку, и носителем целевого ресурса — Применимая целостность носителя — Метаданные целостности, используемые при проверке целостности подресурсов [SRI]
- href lang — Язык типа связанного ресурса— Подсказка для типа ссылочной политики ссылок на ресурсы— Политика ссылок для выборок, инициируемых размерами элементов —Размеры значков (для rel=" значок")imagesrcset — Изображения для использования в различных ситуациях, например, на дисплеях с высоким разрешением, небольших мониторах и т. Д. (для rel="предварительная загрузка")размеры изображений — Размеры изображений для разных макетов страниц (для rel="предварительная загрузка")как потенциальное место назначения для запроса предварительной загрузки (для rel="предварительная загрузка" и rel="модульная загрузка")цвет — Цвет, используемый при настройке значка сайта (для rel="маска-значок")отключено — Отключена ли ссылкаКроме того, атрибут title имеет особую семантику для этого элемента: Заголовок ссылки; Имя набора таблиц стилей CSS.

Литература

- https://developer.mozilla.org/ru/docs/Learn/HTML/Introduction_to_HTML
- https://html.spec.whatwg.org/multipage/
- http://www.itmathrepetitor.ru/zadachi-po-html-i-css/