Мобильное устройство. Эпохи мобильного Интернета. Архитектура и дизайн сайта. Разметка и стандарты. HTML-редакторы. Структура HTML-документа

Мобильное устройство

Согласно Википедии, **Мобильное интернет-устройство** (англ. Mobile Internet Device, MID) — компактные мобильные компьютеры с размером диагонали экрана 4-7 дюймов (10—17,8 см), предназначенные, в первую очередь, для просмотра веб-страниц и работы с веб-сервисами, развлечения и коммуникации. Кроме этого, **MID** может выступать в качестве спутникового навигатора или игровой консоли. Форм-фактор у **MID** может быть самым различным: раскладушка, слайдер или планшет.

Давайте разберемся, в чем заключается эта всем известная мобильность.

Мобильные устройства – ряд устройств, который включает в себя смартфоны, планшеты, электронные книги, телефоны, КПК и нетбуки, ***главной особенностью которых является размер, а также количество выполняемых ими функций. Смартфоны*** *– устройства, важной особенностью которых является размер и способность к транспортированию, а также большой ряд функциональных возможностей.* ***Интернет-планшеты*** *оснащены большим экраном, и позволяют пользоваться интернетом, книгами, офисными пакетами, а также играми.*

Электронные книги *по характеру напоминают планшеты, однако они узко специализированы. Основной их задачей считается чтение книг и электронных файлов. Эти мобильные устройства основаны на матрице e-ink, которая по своим свойствам имитирует обычную бумагу, т.е. экран не имеет подсветки и на вид глазом воспринимается как обычный лист бумаги.*

Время автономной работы электронных книг составляет от трех до десяти дней. Далее следуют смартфоны, которые работают автономно от одного до трех дней, а планшеты работают не более 10 часов в зависимости от интенсивности нагрузки.

Эпохи мобильного Интернета

В первую очередь скажем главное: ***"G"*** *значит* ***"generation"*** *(поколение), поэтому если вы слышите, как кто-то говорит о "4G сети", то это значит, что они имеют в виду беспроводную сеть, разработанную по технологии четвёртого поколения.*

Спецификации любого поколения связи, как правило, относятся к изменению фундаментального характера обслуживания, несовместимым технологиям передачи, более высоким пиковым битрейтом, новыми полосами частот, более широким каналом полосы пропускания, выражаемой в единицах частоты – герцах, а также большей ёмкостью для множественной одновременной передачи данных (более высокой системой спектральной эффективности, измеряемой в бит/с/Гц/сектор).

*Новые поколения мобильной связи начинали разрабатываться практически через каждые десять лет с момента перехода от разработок первого поколения аналоговых сотовых сетей в 1970-х годах (**1G**) к сетям с цифровой передачей (**2G**) в 1980-х годах. От начала разработок до реального внедрения проходило достаточное количество времени (например, сети 1G были внедрены в 1984 году, сети 2G – в 1991 году). В 1990-х годах начал разрабатываться стандарт **3G**, основанный на методе множественного доступа с кодовым разделением каналов (CDMA); он был внедрен только в 2000-х годах. Сети поколения **4G**, основанные на IP-протоколе, стали разрабатываться в 2000 году и начали внедряться во многих странах с 2010 года.*

***1G** (или 1-G) – первое поколение беспроводных телефонных технологий и мобильных телекоммуникаций. Это аналоговые телекоммуникационные стандарты, которые были введены в 1980-х.*

Первым стандартом 1-G стал NMT (Nordic Mobile Telephone), используемый в странах Северной Европы, Швейцарии, Нидерландах, Восточной Европе и России. Другие стандарты включают в себя AMPS (Advanced Mobile Phone System), используемые в Северной Америке и Австралии, TACS (Total Access Communications System) в Великобритании, C-450 в Западной Германии, Португалии и Южной Африке, Radiocom 2000 во Франции и RtMI в Италии. В Японии было несколько систем. NTT разработала три стандарта (TZ-801, TZ-802 и TZ-803), в то время как конкурирующая система под управлением DDI использовала стандарт JTACS (Japan Total Access Communications System).

AMPS в США и комбинации TACS и NMT в Европе. эти новые стандарты предоставили широкий диапазон частот для достаточно интенсивного использования абонентами, были полностью автоматизированными со стороны провайдеров, не нуждаясь в операторе, и использовали электронику, которую можно было легко уместить в маленький корпус.

В 1G-сетях фактическая скорость загрузки составляла от 2.9 Кбайт/с до 5.6 Кбайт/с. Дело в том, что во времена 1G никто не думал об услугах передачи данных; это были чисто аналоговые системы, разработанные для голосовых звонков.

Предшественником 1G технологии является подвижная радиотелефонная связь (или стандарт «нулевого поколения» 0G).

*В начале 90-х 1-G были вытеснены более совершенной цифровой технологией **2G.** Основным различием между системам 1G и 2G является то, что сети 1G используют аналоговую модуляцию радиосигналов, в то время как сети 2G является цифровыми, что позволяло, среди прочего, **шифровать разговоры и посылать СМС.***

2G обладали большим количеством очевидных преимуществ над аналоговыми сетями, на смену которым они пришли: лучшее качество звука, улучшенный уровень безопасности, а также более высокая пропускная способность, если говорить о наиболее значительных изменениях.

GSM начал распространяться в Европе, а D-AMPS и ранняя версия CDMA от Qualcomm, известная как IS-95, становилась популярной в США. Однако эти возникшие 2G стандарты всё ещё ***плохо поддерживали встроенную в них передачу данных.***

Общий сервис пакетной радиопередачи данных, ***GPRS****, появившись в 1997-м году, явился переломным моментом в истории сотовой связи. Он содержал в себе дополнение для GSM-сетей, делающее услуги связи доступными всегда. Однако он не сделал никакого вклада в своё поколение. Видите ли, к тому времени, как GPRS появился на рынке, Международный союз телекоммуникаций ООН (ITU) уже установил свой стандарт* ***IMT-2000,*** *официальный список спецификаций, которым должна соответствовать "настоящая" технология 3G. Что более важно, в соответствии с IMT-2000, стационарная скорость должна быть 2 Мб/с, а мобильная – 384 кб/с - критерии, которым GPRS не соответствовала даже в свои лучшие времена.*

Вот и вся история о том, как GPRS застрял между двух огней: он был лучше, чем 2G, но недостаточно хорош, чтобы считаться 3G.

3G *(от англ. third generation – третье поколение), технологии мобильной связи 3 поколения – набор услуг, который* ***объединяет как высокоскоростной мобильный доступ с услугами сети Интернет, так и технологию радиосвязи, которая создаёт канал передачи данных****. Официальная спецификация 3G от ITU также требовала от совместимых технологий лёгкого перехода с 2G-сетей. По этому критерию GSM-операторы отдавали предпочтение стандарту UMTS, а CDMA2000 стал последователем IS-95, совместимый с прежними версиями.*

EDGE – Enhanced Data-rates for GSM Evolution (Усовершенствованная передача данных для эволюции GSM) – воспринималась как лёгкий способ для операторов GSM сетей выжать ещё чуть-чуть сока из своих 2.5G устройств без серьёзных капиталовложений на модернизацию оборудования для UMTS и расширения диапазона. EDGE не обладает такой большой скоростью, как UMTS или EV-DO, поэтому нельзя сказать, что это 3G, но он всё же быстрее, чем GPRS, а значит, он должен быть лучше, чем 2.5G, многие люди назвали бы EDGE 2.75G технологией, или 2G стандартом, способным достигать скоростей 3G.

4G *(от англ. fourth generation – четвёртое поколение) – поколение мобильной связи с повышенными требованиями. К четвёртому поколению принято относить перспективные технологии, позволяющие осуществлять передачу данных со скоростью, превышающей 100 Мбит/с – подвижным (с высокой мобильностью) и 1 Гбит/с – стационарным абонентам (с низкой мобильностью), что соответственно в 250 и 500 раз лучше, чем в IMT-2000. Это действительно поразительные скорости, превосходящие обычное DSL либо кабельное широкополосное соединение, и поэтому FCC так настаивали на том, что беспроводные технологии играют главную роль в обеспечении сельских местностей широкополосной связью – гораздо дешевле разместить одну 4G точку, которая может покрыть несколько дюжин миль, чем застилать обрабатываемые земли оптоволокном.*

Технологии LTE Advanced (LTE-A) и WiMAX 2 (WMAN-Advanced, IEEE 802.16m) (SIM-карта не требуется) были официально признаны беспроводными стандартами связи четвёртого поколения 4G (IMT-Advanced) Международным союзом электросвязи на конференции в Женеве в 2012 году.

5G *– новое поколение мобильной связи, способное обеспечить большую пропускную способность для широкополосной мобильной связи. Вопреки распространенному*

заблуждению, 5G на самом деле не является модифицированной версией 4G. Это совершенно новая технология, которая кардинально изменит нашу жизнь.

5G уже заявлена как сеть с возможностью передачи данных на скорости до 20 Гбит/с. Вот пример для сравнения: максимальная скорость 4G сегодня – 150 Мбит/с, что в 133 раза меньше!

Архитектура и дизайн сайта

Два весомых критерия - ***юзабилити*** *(удобство пользователей) и* ***эффективное поисковое продвижение сайта*** *- главные аргументы для разработки и реализации эффективной структуры сайта.*

Архитектура сайта – это расположение его страниц. *Главная цель архитектуры сайта состоит в поиске оптимальных решений для максимально удобного взаимодействия посетителя и ресурса. Архитектура помогает визуально представить все разделы сайта, что очень важно в процессе разработки. Владельцу сайта сразу видно, какие именно разделы включены в веб-сайт, дизайнеру видно, какие страницы надо разработать и сверстать, райтер знает, каким контентом наполнять конкретные разделы.*

Дизайна сайта – это более широкое понятие, которое включает в себя несколько факторов: удобность ресурса, его восприятие и оформление.

Процесс создания любого сайта или его реконструкции начинается с разработки структуры. Необходимо сформулировать: сколько и каких разделов и подразделов целесообразно сделать, сколько страниц должно быть в каждом из разделов (подразделов). Особое внимание целесообразно уделить полноте охвата, логике подачи материала.

Архитектура и дизайн сайта создается по эскизу на листике.

Современные подходы к составлению архитектуры и дизайна сайта:

1. Линейная структура – *это самый распространенный тип структурирования информации на сайте. Информация в линейной структуре не размещается по уровням, она равномерно распределена на всех страницах, независимо от своей ценности.*

Область применения линейной структуры сайта: *Линейная структура размещения информации отлично подходит для сайтов-визиток и онлайновых учебных пособий. Создается такая структура при помощи обычного набора HTML-страниц. Страницы связываются между собой ссылками, ведущими на предыдущую или следующую страницу: пользователь должен просматривать их как презентацию или слайд-шоу. В стороне размещается оглавление или содержание сайта.*

Реализовать такой вид структуры очень легко, так как в большинстве случаев она представляет собой набор html-страниц, с каждой из которых есть ссылка на следующую-предыдущую.

Очень важно, чтобы на каждой странице сайта были название и ссылка на первую страницу, желательно также указывать общее количество страниц и обозначать ту, на которой в данный момент пользователь находится.

Особенности линейной структуры сайта:

1) *акцент на одном продукте;*
2) *возможность выложить большой объем специализированной или маркетинговой информации;*

3) «по дефолту» очевидный процесс покупки продукта, который может быть спроектирован исходя из специфики продукта.

2. Альтернативная линейная структура *практически ничем не отличается от первого типа. Единственное, что следует отметить, это наличие выбора у посетителя, а точнее - выбор между 2-мя ветками. Например, когда на сайте разделяются корпоративные и частные клиенты.*

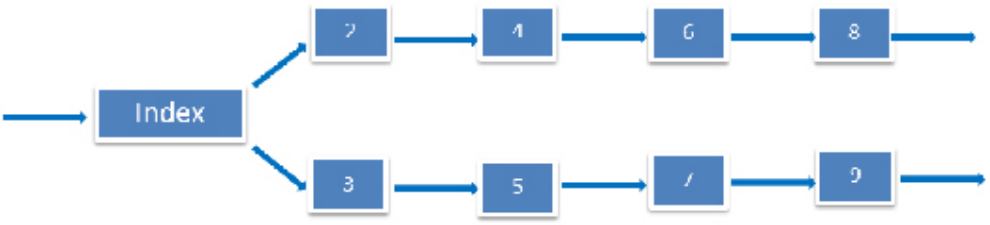


Рис. 2 Линейная структура сайта с альтернативным вариантом А

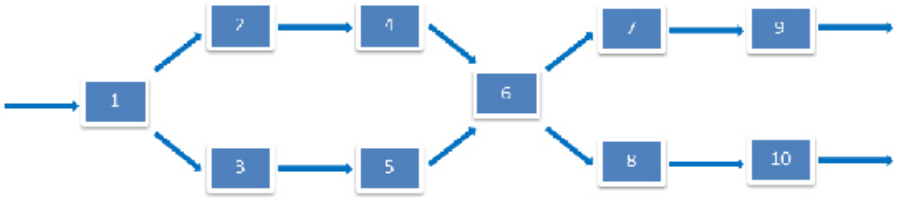


Рис. 3 Линейная структура сайта с альтернативным вариантом Б

Область применения: *Чаще всего подобная структура используется для заполнения форм регистрации посетителей сайта. В этом случае все люди начинают работу со стартовой страницы. Однако потом частным лицам предлагается ввести одну информацию, а представителям коммерческих структур – другую.*

Довольно часто встречается ситуация, когда уже "разделённые по веткам" посетители, тем не менее, встречаются на какой-либо странице – например, оплата или отзыв о работе.

3. Разветвленная линейная структура *сайта по своему принципу организации напоминает небольшие дороги, ведущие к главному шоссе. То есть, пользователь может просмотреть заинтересовавший его блок информации, после чего вернуться в исходное положение. Такая схема организации информации подходит для постоянно развивающихся сайтов.*

Она представляет собой линию со множеством небольших ответвлений, в которых посетитель может подробнее узнать о заинтересовавшей его информации, а позже продолжить изучение контента основной линии.

Основным достоинством линейной структуры с ответвлениями – это относительно несложная возможность web-мастерам перейти на неё с обычной линейной структуры, без глубокой переработки сайта, если требования к нему возросли. При развитии сайта в этом часто возникает необходимость. Контент сильно разрастается и необходимо улучшить

навигацию. И после этих изменений сайт не перестает быть удобным и простым для просмотра посетителями.

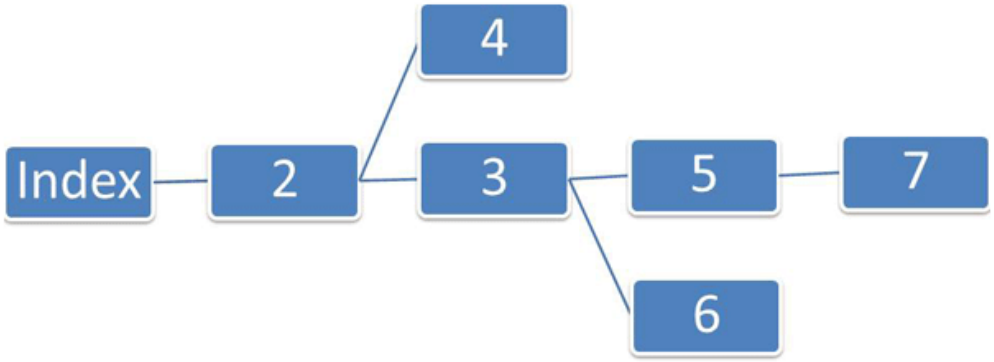


Рис. 4 Линейная структура сайта с ответвлениями

Это чуть более сложная структура, свойственная небольшим корпоративным ресурсам, сайтам-визиткам, некоторым авторским блогам. Как правило, здесь также нет разделов, а есть только отдельные статичные страницы. Но ссылки на все эти страницы (или на большинство из них) размещены на главной. Благодаря этому система навигации здесь очень простая и интуитивно-понятная, а доступ ко всем страницам осуществляется всего лишь за 1 или 2 клика.

Характерный пример сайта с подобной структурой - визитка какой-нибудь фирмы (с главной проставлены ссылки на страницу с каталогом товаров, прайс-листом, контактными данными, вакансиями и т.д.).

Например, это может быть онлайн-библиотека какого-то автора с несколькими книгами. Здесь вес опять же передается от главной к последней странице, правда таких страниц уже несколько.

И снова неудачный для продвижения вариант.

4. Древовидная структура. *Эта схема расположения информации считается самой универсальной. Ее используют для составления дизайна в коммерческих и новостных сайтов также она отлично подходит для информационных и других ресурсов.*

Архитектура и дизайн сайта строиться по принципу «от большого к меньшему». То есть, пользователь на главной странице выбирает тему, откуда его направляют на другую страницу, где размещается несколько подзаголовков к главной теме. Такая схема обеспечивает максимально быстрый доступ к информации любого уровня. Однако, древовидная структура имеет один недостаток – она теряет свои достоинства при своем расширении.

Идея применения подобной структуры заключается в том, что у человека есть выбор и возможность как с главной страницы сайта, так и любой другой, перейти в любой раздел, подраздел и на конкретную страницу (документ) (каждая страница раздела будет ссылаться не только на главную, но и на свой раздел).

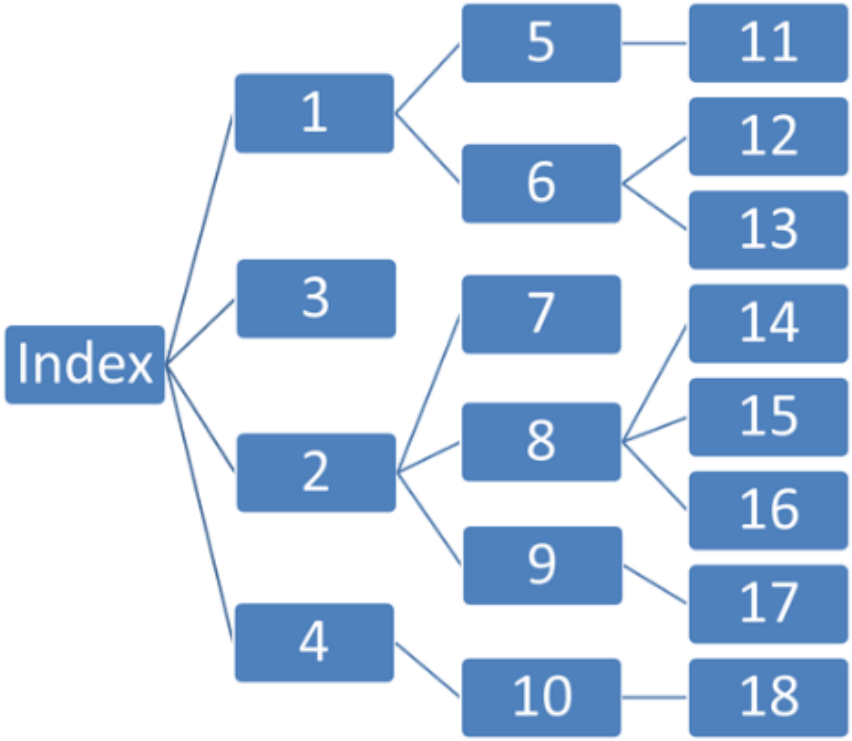


Рис. 6 Древовидная структура сайта

Достоинства древовидной структуры сайта:

- главное достоинство древовидной структуры сайта - универсальность. Древовидная структура прекрасно может подойти для любого вида сайта, будь то домашняя веб-страничка, сайт-визитка, корпоративный сайт, портал или каталог;

- прекрасная навигация. Идея применения подобной структуры заключается в том, что у человека есть выбор и возможность как с главной страницы сайта, так и любой другой, перейти в любой раздел, подраздел и на конкретный страницу (документ);

- большая гибкость. Хотя на обычном HTML такую структуру сайта практически невозможно организовать (да и не нужно), для её создания пишется движок или используется CMS.

Недостатки древовидной структуры сайта:

- при использовании древовидной структуры очень сложно соблюдать баланс между "глубиной и шириной". Если «дерево» сайта будет расти только вглубь, то пользователям, чтобы дойти до какой-то информации, придётся загрузить и просмотреть слишком много страниц, что будет раздражать пользователей. Если создать очень широкую древовидную структуру, то посетители будут вынуждены каждый раз тратить очень много времени для выбора нужной им ветки. А это тоже плохо. Таким образом, при использовании древовидной структуры сайта необходимо постоянно следить за её разрастанием и придерживаться золотой середины.

5. Решетчатая структура (смешанная). *По принципу организации напоминает древовидную.*

Одна из самых сложных структур сайта, где все документы располагаются в разных ветках. Однако посетитель может легко перемещаться по этим веткам как горизонтально (слева направо или между ветками на различных уровнях), так и вертикально (сверху вниз).

Она достаточно сложна в реализации и используется чаще всего в каталогах. Ведь если недостаточно кропотливо подойти к проектированию сайта с данной структурой, он рискует превратиться в тот самый лабиринт хаоса.

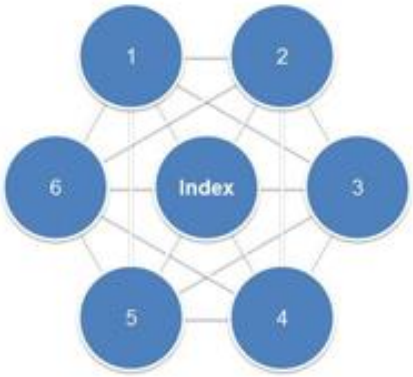


Рис. 5 Решётчатая структура сайта

Данный вид структуры характерен преимущественно для каталогов статей или ссылок.

На первый взгляд она очень удобна для пользователей, но для обычных сайтов её лучше не использоваться.

Недостатки:

- решётчатую структуру сложно создать, так как придётся долго копаться в коде или настраивать под неё CMS;

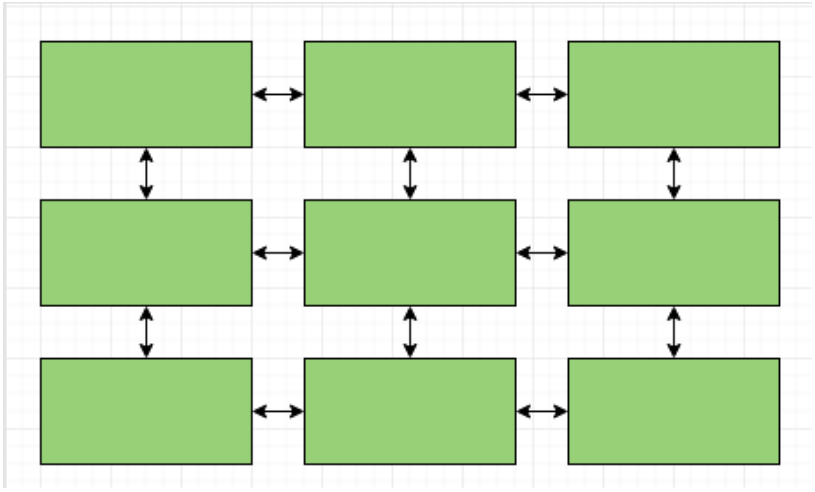
- при организации решётчатой структуры можно легко запутаться не только пользователю во время поиска информации, но и самому веб-мастеру при размещении контента;

-на сайте размещается большое количество гиперссылок, поэтому применение решётчатой структуры ограничено для больших сайтов.

Мы поговорили лишь об основных структурах, существуют еще и разные их вариации.

Такая классификация, конечно, является общей, и структура многих сайтов может представлять собой комбинацию основных видов, а иногда и совсем не подходить ни под одно из описаний.

6. Блочная структура. *Здесь все страницы ссылаются на несколько других, которые равнозначны между собой. Такую структуру неплохо применять для конкретного продукта, когда каждую страницу можно использовать, как описание какого-то отдельного свойства/достоинства и их совокупностей. С распределением веса здесь все вполне неплохо, страницы уже перелинкованы и отдают свой вес на главную, что позволяет продвигать ее более эффективно. Но такая структура весьма специфична и применять ее можно далеко не везде. Пример структуры сайта:*



Главное, что при выборе структуры для будущего сайта, веб-мастеру необходимо учитывать потребности, которые могут возникнуть. Так он сумеет не только сохранить удобство и простоту своего сайта, но и подарить посетителям приятные моменты, тем самым сохранив их для себя, и быть может, приумножив их число.

Грамотно структурированный сайт – не только предмет гордости самого веб-мастера, но и важное условие благосклонности к сайту со стороны посетителей. Ведь важно не только качество информации, но и ее удобное, последовательное представление.

Ведь даже если проект был создан удобным для посетителей, то из-за постоянных расширений и дополнений, которые прикрепляют "куда придется", он превращается в сайт, который не только заставляет нас уйти на сайт конкурента, но и навсегда отбивает желание посещать его вновь. Но избежать этой участи сайту несложно. Для этого необходимо, еще до того как разработчики приступили к дизайну, продумать подходящую структуру будущего проекта и, по мере расширения его, всегда придерживались этой структуры.

Разметка и стандарты

Консорциум Всемирной паутины *(англ. World Wide Web Consortium, W3C) — организация, разрабатывающая и внедряющая технологические принципы и стандарты для Всемирной паутины (часто называемые «Рекомендациями»).*

Как и для любого другого канала передачи информации, для веб-среды постепенно складывается ***набор стандартов,*** *называемые* ***веб-стандартами****, в той или иной степени разделяемый всеми игроками (веб-разработчиками, производителями программного обеспечения, хостинг-провайдерами и т. д.).*

Отсутствие или нежелание следовать подобным стандартам почти всегда приводит к проблемам с совместимостью, например, между программным кодом сайта и особенностями его отображения в разных браузерах.

К числу World Wide Web Consortium, W3C рекомендаций относятся:

- общедоступность. Особый упор при реализации данного принципа делается на обеспечение доступности сайтов для людей с ограниченными возможностями;

- аппаратная независимость. Согласно этому принципу, веб-сайт не должен быть рассчитан на конкретное физическое оборудование, а должен адекватно работать на любом стандартном аппаратном обеспечении;

- интернационализация. Речь идет, прежде всего, о поддержке всего многообразия языков (и шрифтовых систем);

- многоформенное взаимодействие. Сайт должен поддерживать самые разные формы взаимодействия с пользователями;

- мобильная паутина. Этот принцип определяет, что ресурсы Всемирной сети должны быть доступны для мобильных устройств.

На основе стандартов W3C разработано множество спецификаций и технологий создания веб-сайтов. Например, вы можете использовать технологии:

*- **HTML** (Hypertext Markup Language — язык гипертекстовой разметки).*

*- **XML** (ExtensibIe Markup Language — расширенный язык разметки).*

*- **XHTML** (ExtensibIe Hypertext Markup Language — расширенный язык гипертекстовой разметки).*

*- **CSS** (Cascading Style Sheets — каскадные таблицы стилей).*

*- **WSDL** (Web Service Description Language — язык описания веб-сервисов).*

Например, если следовать спецификации HTML, разработанной W3C, то полученная веб-страница будет одинаково отображаться в любом браузере, который также следует этой спецификации. При этом в настоящее время нет ни одного широко используемого браузера, полностью отвечающего спецификации, но усилия разработчиков браузеров направлены как раз на более точное соответствие этой спецификации.

В частности, для браузеров существуют такие понятия, как «режим стандартов» и «режим совместимости» (названия, конечно, несколько условные).

*«**Режим стандартов**» (Standards mode) – режим работы браузера, при котором веб-страница отображается в наиболее полном соответствии со стандартом (спецификацией), разработанным W3C. В этот режим браузер переводится при отображении документов, содержащих определенный (скрытый от пользователя) заголовок. В большинстве браузеров режим стандартов не является режимом отображения по умолчанию, исключение составляет браузер Internet Explorer 8, которым мало кто пользуется. Поэтому большинство разработчиков сайтов используют этот заголовок, что позволяет грамотно отображать страницы почти в любом браузере. Заметьте, что «грамотно» здесь не значит «так, как было задумано дизайнером».*

*«**Режим совместимости**» (Quirks mode) – режим работы браузера, при котором веб-страница отображается максимально близко к отображению, используемому в предыдущих версиях того же браузера. Обычно браузер находится в этом режиме по умолчанию. Если веб-страница работает в режиме совместимости, то в каждом браузере ее отображение может быть разным. При этом разработчики часто используют так называемые «разводки» по браузерам – механизм динамического определения текущего браузера и модификации страницы в зависимости от него. Таким образом, страница в режиме совместимости обычно является оптимизированной под один или несколько конкретных браузеров.*

***Прочие стандарты.** Существует также значительное количество отраслевых и страновых стандартов, законов и рекомендаций. Например, документы общества «Europe initiative» (Европейский союз), «508-я статья» (США), Рекомендации правительства Великобритании и другие. Множество документов регламентирует процессы разработки IT-систем (например, методология Prince2 в Великобритании).*

В России и Беларуси отсутствует даже ГОСТ, формализующий требования к веб-разработкам и их продуктам, слабо развита отраслевая законодательная база. Отдельные рекомендации и спецификации описывают требования к программным компонентам, например, к каскадным таблицам стилей (CSS).

На основе рекомендаций различных стандартов создано несколько онлайн-инструментов, позволяющих проверить корректность разметки и доступность сайта (так называемый «валидатор»). Кроме того, при помощи этих инструментов вы можете проверить любой (в том числе и ваш) сайт на предмет соответствия его веб-стандартам:

- Validate Your Markup (validator.w3.org);

- Cynthia Says (www.cynthiasays.com);

- WebAIM's Wave (www.wave.webaim.org).

Не все сайты в сети следуют стандартам. Случается это по разным причинам: изначально сайт был создан с ошибками (80% случаев), ошибка внесена людьми, наполняющими сайт (20%), ошибка была сделана намеренно (около 10% случаев).

К примеру, популярнейший белорусский портал ***TUT.BY*** *сделан с грубым несоблюдением спецификаций (более 600 ошибок и 300 предупреждений валидатора (данные на момент написания статьи — 2009 г., сейчас количество ошибок уменьшено.), что однако не мешает ему одинаково отображаться в различных браузерах.*

Другими словами — нужно знать, где копать.

HTML-редакторы

Создание web-сайтов, их поддержка и развитие осуществляется с помощью специализированного ПО.

Реда́ктор HTML или HTML-реда́ктор — программа, позволяющая создавать и изменять HTML-страницы.

Несмотря на то, что HTML-код может быть написан в простом текстовом редакторе (например, в Блокноте), специальные редакторы для написания кода HTML предлагают больше удобств и функциональности, могут содержать дополнительные шаблоны и ресурсы для создания и редактирования страниц. Делятся на следующие основные группы:

1) редакторы исходного кода. HTML-документ, можно создавать в любом текстовом редакторе, например, в "блокноте" (MS NotePAD в составе Windows). HTML-документ, можно конвертировать из многих программ, (например, Microsoft Word). После конвертации одна страница текста может превратиться в огромный HTML-файл размером более 100кб.

2) визуальные (WYSIWYG – What You See Is What You Get)-редакторы (что видишь, то и получишь). Кроме визуального редактирования, такие редакторы сохраняют возможность работы с исходным кодом. WYSIWYG-редакторы сами вырабатывают html-код документа, в то время как разработчик лишь выбирает нужные ему опции из меню.

Кроме того, в некоторых источниках выделяется третья группа редакторов - комбинированные редакторы.

Разработчик сайта должен использовать разумное сочетание всех методов создания HTML-документов. При использовании этих методов следует учесть следующее:

- создание различных эффектов в простом текстовом редакторе - громоздкая и сложная задача;

- документ подготовленный с помощью какой-либо программы проще конвертировать, чем создавать заново;

- текстовые редакторы можно использовать для очистки от "мусора" HTML-документов, созданных с помощью специализированных программ;

- при создании эффектов с помощью специальных программ (например, Microsoft FrontPage) следует предусмотреть поддержку этих эффектов на web-сервере.

Одним из лидеров в области разработки программного обеспечения для подготовки web-публикаций является компания Macromedia (http://www.macromedia.com). Очень

популярны пакеты Macromedia: Dreamweaver, HomeSite (до версии 5 этот пакет выходил под названием Allaire HomeSite), а также специализированные пакеты для создания компьютерной графики и анимации.

Создание графики для web заслуживает особого внимания . Следует иметь в виду, что графика бывает двух видов: растровой (когда описывается каждая точка изображения) и векторная (когда задается формула для генерации изображения). Последний способ построения графики более компактен и, в частности, идеально подходит для создания анимаций.

В таблице представлены наиболее популярные программные средства, предназначенные для разработки Web-сайтов:

Microsoft FrontPage	**WYSIWYG** -редактор. Недостаток: автоматически вырабатываемый html-код документа, созданного разработчиком в визуальном режиме, как правило, неоптимален.
Macromedia Flash 5	Технология **Flash** становится очень популярной. Она позволяет создавать очень эффектные web-страницы, содержащие FLASH-объекты или исполняемые файлы, содержащие большое количество векторной графики, анимационные ролики. За счет применения векторной графики **Flash**-страницы быстрее загружаются на компьютеры клиента, чем традиционные (содержащие растровую графику) и одинаково воспринимаются на различных платформах: Windows, Macintosh, Solaris, Unix. Имеется возможность передачи данных из HTML-документа FLASH-объекту и наоборот, что позволяет создавать *управляемые*FLASH-объекты, а также делать более эффектными HTML-страницы (например, формы).
Macromedia Director 8	**Лидер рынка мультимедийных средств**. Объединяет графику, звук, анимацию, текст и видео для интерактивных информационных каналов, которые можно разместить как на web-страницах, так и на CD- или DVD-дисках. От технологии **Flash** отличается *более развитым встроенным языком программирования*.
Macromedia Dreamweaver MX 2004	WYSIWYG-редактор. **Профессиональное решение для web-дизайна и разработки web-сайтов**. • Имеет очень удобный, простой интерфейс (в стиле PageMaker/Illustrator/PhotoShop). • Автоматизирует работу над проектом. Создаваемый код почти не отличается от написанного программистом. • Содержит встроенные средства работы с графикой. • Позволяет *непосредственно внутри пакета* создавать **FLASH-анимации**. • Обеспечивает средства отладки **JavaScript**-сценариев для браузеров **MS Internet Explorer** и **Netscape Navigator**. • Допускает расширение возможностей за счет дополнительных модулей. Библиотека дополнительных компонент (более 150) входят в комплект поставки • при вводе кодов создает список значений тэгов и атрибутов в виде всплывающей подсказки
Macromedia Fireworks 3	Профессиональное приложение для создания графических изображений и их размещения в Интернет.

	• Позволяет обрабатывать изображения, полученные с помощью других графических редакторов, цифровые фотографии, отсканированные изображения • Позволяет создавать эффекты анимации, использовать динамические стили **Macromedia Dreamweaver 3 Fireworks 3 Studio** Совместное использовании Dreamweaver 3 и Fireworks 3 сокращает время разработки за счет взаимной автоматизации повторяющихся действий.
Allaire HomeSite 4.5	Позволяет легко и быстро создавать эффектные web-сайты • Имеет удобный интуитивно понятный интерфейс, богатую палитру инструментов • Содержит средства контроля качества: проверку синтаксиса html-кода, верификацию ссылок **НОВОСТЬ**: Фирма **Allaire** несколько лет назад была поглощена компанией **Macromedia**. Известный продукт **HomeSite** теперь выходит под названием **Macromedia HomeSite 4.5.** (хотя название **Allaire** все еще присутствует на упаковке.)
Macromedia DreamWeaver UltraDev 4	Первая визуальная среда, позволяющая быстро разрабатывать Web-приложения для доступа к серверным базам данных. • БД могут поддерживаться на различных серверных платформах. • Достаточно просто создаются системы электронной коммерции, такие как электронные витрины, системы регистрации клиентов. Продукт уже имеет награды как лучший в своем классе средств разработки.
Macromedia ColdFusion 4.5. UltraDev 4 Studio	Объединение среды разработки **ColdFusion Studio** и среды **DreamWeaver UltraDev**. Содержит мощные инструменты визуальной разработки приложений для размещения на платформе **ColdFusion Server 5**, визуального представления серверного источника данных (набора записей, переменной, директория и пр.), средства отладки сценариев.
Adobe PhotoShop 6	Мировой стандарт обработки изображений как для печати, так и для web

Писать странички в блокноте конечно можно, но я Вам могу рекомендовать воспользоваться полноценным HTML редактором, благо их на рынке программного обеспечения великое множество. Я не буду давать их подробное описание, скажу лишь что они, как правило, несут в себе одинаковые стандартные наборы инструментов и отличаются друг от друга пожалуй только интерфейсом. Они носят исключительно характер привычки пользования веб мастером тем или иным редактором html кода.

Структура HTML-документа

HTML-документ *– это обычный текстовый документ, может быть создан как в обычном текстовом редакторе (Блокнот), так и в специализированном, с подсветкой кода (Notepad++, Visual Studio Code и т.п.). HTML-документ имеет расширение* ***.html***.

Структура HTML-документа. Физическое и логическое форматирование текста

HTML-документ состоит из дерева HTML-элементов и текста. Каждый элемент обозначается в исходном документе начальным (открывающим) и конечным (закрывающим) тегом (за редким исключением).

В основном все теги парные. Начальный тег показывает, где начинается элемент, конечный – где заканчивается. Такая пара тегов называется контейнером. Закрывающий тег образуется путем добавления слэша / перед именем тега: ***<имя тега>…</имя тега>.*** *Между начальным и закрывающим тегами находится содержимое тега — контент. Действия тегов распространяются только на их содержимое.*

Одиночные теги не могут хранить в себе содержимого напрямую, оно прописывается как значение атрибута, например, тег <input type="button" value="Кнопка"> создаст кнопку с текстом ***Кнопка*** *внутри.*

Теги могут вкладываться друг в друга, например, Текст . При вложении следует соблюдать порядок их закрытия (принцип «матрёшки»), например, следующая запись будет неверной: Текст .

HTML-элементы могут иметь атрибуты (глобальные, применяемые для всех HTML-элементов, и собственные). Атрибуты прописываются в открывающем теге элемента и содержат имя и значение, указываемые в формате имя атрибута="значение". Атрибуты позволяют изменять свойства и поведение элемента, для которого они заданы.

Простейшая структура html-документа состоит из трех пар тегов:

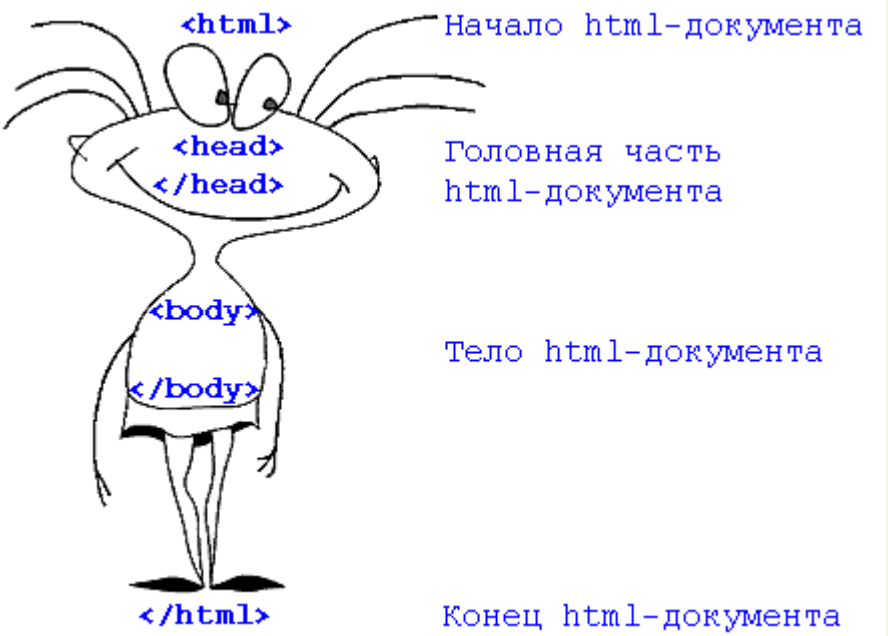
- HTML-документ обрамляется парными тегами <html> … </html>. Теги <html> </html> являются контейнером для всех остальных, т.е в них помещаются все остальные. Таким образом, документ должен начинаться с тега <html>, а заканчиваться тегом </html>;

- сам документ условно разделен на две части – заголовок документа (теги <head> </head>) и тело документа (теги <body> </body>).

```
<html>

   <head>
        Заголовок документа
   </head>

   <body>
        Тело документа
   </body>
</html>
```



Пример:

```
<html>

   <head>
      <b>  Первая страница </b>
   </head>

   <body>
        Ура!!! Я сделал первую html страницу в своей жизни
   </body>
</html>
```

Браузер просматривает (интерпретирует) HTML-документ, выстраивая его структуру (DOM) и отображая ее в соответствии с инструкциями, включенными в этот файл (таблицы стилей, скрипты). Если разметка правильная, то в окне браузера будет отображена HTML-страница, содержащая HTML-элементы — заголовки, таблицы, изображения и т.д.

Процесс интерпретации ***(парсинг)****начинается прежде, чем веб-страница полностью загружена в браузер. Браузеры обрабатывают HTML-документы последовательно, с самого начала, при этом обрабатывая CSS и соотнося таблицы стилей с элементами страницы.*

Кроме того, для обеспечения корректного отображения документа современный стандарт требует использования одиночного тега *(означает, что закрывающий тег не требуется)* <!DOCTYPE>.

Элемент ***<!DOCTYPE>*** *предназначен для указания типа текущего документа — DTD (document type definition, описание типа документа). Это необходимо, чтобы браузер понимал, как следует интерпретировать текущую веб-страницу, поскольку HTML существует в нескольких версиях, кроме того, имеется XHTML (EXtensible HyperText Markup Language, расширенный язык разметки гипертекста), похожий на HTML, но различающийся с ним по синтаксису. Чтобы браузер «не путался» и понимал, согласно какому стандарту отображать веб-страницу и необходимо в первой строке кода задавать<!DOCTYPE>.*

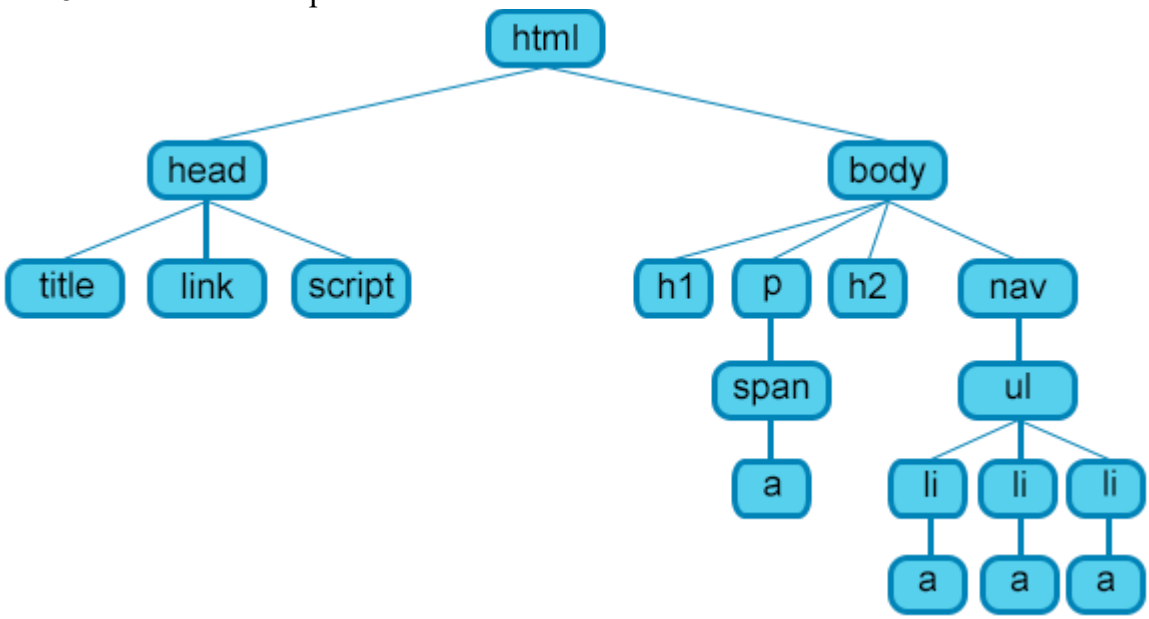
```
<!DOCTYPE html> <!-- Объявление формата документа -->
<html>
<head> <!-- Техническая информация о документе -->
<meta charset="UTF-8"> <!-- Определяем кодировку символов документа -->
<title>...</title> <!-- Задаем заголовок документа -->
<link rel="stylesheet" type="text/css" href="style.css"> <!-- Подключаем внешнюю таблицу стилей -->
<script src="script.js"></script> <!-- Подключаем сценарии -->
</head>
<body> <!-- Основная часть документа -->
</body>
</html>
```

Согласно спецификациям HTML и XHTML тег DOCTYPE (что означает «объявление типа документа») сообщает валидатору, какую именно версию (X)HTML вы используете в своей странице. Этот тег должен всегда находиться в первой строке каждой страницы. Тег DOCTYPE — ключевой компонент web-страниц, претендующих на соответствие стандартам: без него ваш код и CSS не пройдут проверку валидатором.

Таким образом, каркас HTML-документа будет иметь следующую структуру:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
СОДЕРЖАНИЕ ЗАГОЛОВКА
</head>
<body>
СОДЕРЖАНИЕ ТЕЛА ДОКУМЕНТА
</body>
</html>
```

Элементы, находящиеся внутри тега **<html>**, образуют дерево документа, так называемую **объектную модель документа, DOM (document object model)**. При этом элемент **<html>** является корневым элементом.



Для элемента **<html>** доступны глобальные атрибуты.

Заголовок *содержит техническую информацию о странице: заголовок, описание, ключевые слова для поисковых машин, кодировку и т.д. Введенная в нем информация не отображается в окне браузера, однако содержит данные, которые указывают браузеру, как следует обрабатывать страницу.*

Заголовок включает в себя несколько специализированных тегов, основными из которых являются **<title> … </title>** и **<meta> … </meta>**. *Хотя тег <meta> всего один, он имеет несколько атрибутов, поэтому к нему и применяется множественное число.* Теги заголовка напрямую не отображаются в окне браузера, за исключением тега <title>, который определяет название веб-страницы.

Тег **<title>** содержит заголовок документа, который будет выводится в строке заголовка окна браузера. *Длина заголовка должна быть не более 60 символов, чтобы полностью поместиться в заголовке. Текст заголовка должен содержать максимально полное описание содержимого веб-страницы.*

Необязательным тегом раздела ***<head>*** *является одинарный тег* ***<meta>***. Метатеги **<meta>** содержат специальную информацию, такую как тип кодировки, а также используются для хранения информации, предназначенной для браузеров и поисковых систем. Например, механизмы поисковых систем обращаются к метатегам для получения описания сайта, ключевых слов и других данных.

Пример описания содержимого страницы и ключевые слова для поисковых машин:

```
<meta name="description" content="Описание содержимого страницы">
<meta name="keywords" content="Ключевые слова через запятую">
```

Описание содержимого страницы и ключевые слова одновременно можно указывать на нескольких языках, например, на русском и английском:

```
<meta name="description" lang="ru" content="Описание содержимого страницы">
<meta name="description" lang="en" content="Description">
<meta name="keywords" lang="ru" content="Ключевые слова через запятую">
<meta name="keywords" lang="en" content="Keywords">
```

С помощью тега ***<meta>*** *можно запретить или разрешить индексацию веб-страницы поисковыми машинами (процесс добавления сведений (о сайте) роботом поисковой машины в базу данных, впоследствии использующуюся для (полнотекстового) поиска информации на проиндексированных сайтах):*

Индексация и переход по ссылкам разрешены:

```
<meta name="robots" content="index, follow">
```

Индексация разрешена, переход по ссылкам запрещен:

```
<meta name="robots" content="index, nofollow">
```

Индексация и переход по ссылкам запрещены:

```
<meta name="robots" content="noindex, nofollow">
```

Теги ***<noindex>*** *и атрибут* ***<nofollow>*** *закрывают содержимое сайта от роботов Яндекса и Google соответственно (Yahoo использует тег <nofollow>).*

Для автоматической перезагрузки страницы через заданный промежуток времени нужно воспользоваться значением ***refresh:***

```
<meta http-equiv="refresh" content="30">
```

Страница будет перезагружена через 30 секунд.

*Чтобы перебросить посетителя на другую страницу, нужно указать URL-адрес в параметре **url**:*

```
<meta http-equiv="refresh" content="0; url=http://yandex.ru/">
```

*Атрибуты тега **<meta>***

Атрибут	Описание, принимаемое значение
Charset	Указывает кодировку символов для текущего HTML-документа: <meta charset="UTF-8">
Content	Содержит произвольный текст, который определяет значение, ассоциируемое с атрибутом **http-equiv** или **name**, в зависимости от их значения.
http-equiv	Контролирует действия браузера на данной веб-странице (эквивалент HTTP заголовков). При отображении страницы браузер будет следовать инструкциям, заданным в атрибуте: **default-style** указывает предпочтительный стиль для использования на странице; **content** должен содержать идентификатор элемента **<link>**, который ссылается на таблицу стилей CSS, или идентификатор элемента **<style>**, содержащего таблицу стилей; **refresh** указывает время в секундах до перезагрузки страницы или время до перенаправления на другую страницу, если в атрибуте **content** после указания времени идет строка **"url=адрес_страницы"**. Автоматическая перезагрузка страницы через заданный промежуток времени, в данном примере, через 30 секунд: <meta http-equiv="refresh" content="30"> Если необходимо сразу перебросить посетителя на другую страницу, то можно указать URL-адрес в параметре url: <meta http-equiv="refresh" content="0; url=http://mail.ru/">
Name	Ассоциируется со значением, содержащемся в атрибуте **content**. Не должен использоваться в случае, если для элемента уже заданы атрибуты **http-equiv**, **charset** или **itemprop**. **application-name** указывает название веб-приложения, используемого на странице; **author** указывает имя автора документа в свободном формате; **description** определяет краткое описание к содержимому страницы; <meta name="description" content="Описание содержимого страницы"> **generator** указывает один из пакетов программного обеспечения, используемого для создания документа, например: <meta name="generator" content="WordPress 4.0"> **keywords** содержит список ключевых слов, разделенных запятыми, соответствующих содержимому страницы, например: <meta name="keywords" content="Ключевые слова через запятую"> *Также атрибут **name** может принимать следующие значения из расширенной спецификации, такие как **creator**, **googlebot**, **publisher**, **robots**,*

	slurp, ***viewport***, *хотя ни одно из них еще не было официально принято.*

В рассмотренных тегах ***name="keywords"*** *и* ***content="список ключевых слов"*** *являются* ***атрибутами*** *тегов, которые конкретизируют их. Например, атрибуты могут указывать, что текст, заключенный в данном теге, при отображении должен выравниваться по центру. Атрибуты записываются сразу после имени тега, причем значения атрибутов заключаются в кавычки. Атрибутов у тега может быть несколько, но могут и вовсе отсутствовать.*

HTML-атрибуты *сообщают браузеру, каким образом должен отображаться тот или иной элемент страницы. Атрибуты позволяют сделать более разнообразными внешний вид информации, добавляемой с помощью одинаковых тегов.*

Значение атрибута заключается в кавычки "". Названия и значения атрибутов не чувствительны к регистру, но, тем не менее, рекомендуется набирать их в нижнем регистре.

Внутри элемента ***<style>*** *задаются стили, которые используются на странице. Для задания стилей в HTML-документе используется язык CSS. Таких элементов на странице может быть несколько.*

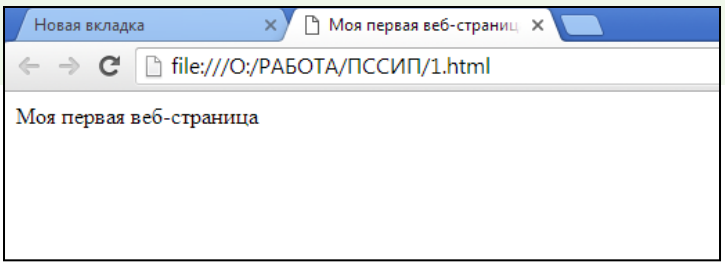
Элемент ***<link>*** *позволяет задать стили для документа, содержащиеся в отдельном файле с расширением* ***.css****, например,* ***style.css****.*

Элемент ***<script>*** *позволяет присоединять к документу различные сценарии.*

Заголовок документа содержит служебную информацию и не влияет на внешний вид документа.

Приведем пример простейшей веб-страницы:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
 <head>
  <title>Моя первая веб-страница</title>
  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
  <meta name="keywords" content="веб-страница первая">
 </head>
 <body>
  Моя первая веб-страница
 </body>
</html>
```



Для элементов ***<head>****,* ***<meta>****,* ***<style>****,* ***<link>****,* ***<script>*** *доступны глобальные атрибуты.*

Глобальные атрибуты

Глобальные атрибуты, приведенные в таблице ниже, могут быть использованы для любого HTML-элемента, хотя некоторые из них могут не оказывать на элементы никакого влияния.

Атрибут	Описание, принимаемое значение
Accesskey	Генерирует сочетания клавиш для доступа к текущему элементу. Состоит из разделенного пробелами списка символов. Браузер в первую очередь

	выбирает те клавиши, которые существуют на раскладке клавиатуры. Применяется к следующим элементам: <a>, <area>, <button>, <input>, <label>, <legend>, <textarea>. Принимаемые значения: перечень названий клавиш.
Class	Определяет имя класса для элемента (используется для определения класса в таблице стилей). Принимаемые значения: имя класса.
Contenteditable	Определяет, может ли пользователь редактировать содержимое (контент). Позволяет преобразовать любое поле HTML в редактируемый элемент. Принимаемые значения: true/false.
Contextmenu	Добавляет к элементу контекстное меню, заданное тегом <menu>. Принимаемые значения: значение атрибута id элемента <menu>.
Dir	Определяет направление текста контента в элементах <bdo> и <bdi>. Принимаемые значения: ltr/rtl/auto.
Draggable	Определяет, может ли пользователь перетащить элемент. Принимаемые значения: true/false/auto.
Dropzone	Определяет область для приема перемещаемых элементов, сообщая браузеру пользователя, какие действия совершить при перемещении. Принимаемые значения: copy — содержимое перемещаемого элемента будет скопировано в область. move — содержимое перемещаемого элемента будет перемещено в новую область. link — при перемещении будет создана ссылка на первоначальные данные элемента.
Hidden	Указывает на то, что элемент должен быть скрыт. Принимаемые значения: hidden.
Id	Определяет уникальный идентификатор элемента. Принимаемые значения: id — идентификатор элемента.
Lang	Определяет код языка содержимого (контента) в элементе. Принимаемые значения: код языка.
Spellcheck	Указывает, подлежит ли содержимое элемента проверке орфографии и грамматики. Принимаемые значения: true/false.
Style	Указывает на код CSS, применяемую для оформления элемента. Принимаемые значения: код CSS.
Tabindex	Определяет порядок перехода к элементу при помощи клавиши TAB. Принимаемые значения: порядковый номер.
Title	Определяет дополнительную информацию об элементе, задавая всплывающую подсказку для страницы. Принимаемые значения: текст.
Translate	Разрешает или запрещает перевод текста внутри элемента. Принимаемые значения: yes/no.

Элемент **<body>** – **тело документа** – определяет видимую часть документа и предназначен для хранения содержания веб-страницы (контента), отображаемого в окне браузера. Информацию, которую следует выводить в документе, следует располагать именно внутри контейнера **<body>**. К такой информации относится текст, изображения, теги, скрипты JavaScript и т.д.

Тег **<body>** также применяется для определения цветов ссылок и текста на веб-странице.

Обязательных параметров у тега **<body>** нет, да и применение необязательных параметров тоже не приветствуется. Тем не менее, большинство параметров до сих пор поддерживается разными браузерами.

*Часто тег **<body>** используется для размещения обработчика событий, например, **onload**, которое выполняется после того, как документ завершил загрузку в текущее окно или фрейм.*

*Атрибуты тега **<body>***

Атрибут	Описание, принимаемое значение
onafterprint	Событие, срабатывающее после отправки страницы на печать или после закрытия окна печати.
onbeforeprint	Событие, срабатывающее перед отправкой страницы на печать.
onbeforeunload	Событие срабатывает, когда посетитель инициировал переход на другую страницу или нажал «закрыть окно». Позволяет отображать сообщение в диалоговом окне подтверждения, чтобы сообщить пользователю, хочет ли он остаться или покинуть текущую страницу.
onhashchange	Событие срабатывает, когда меняется hash-часть URL, например, когда пользователь перейдет с адреса example.domain/test.aspx#page1 на example.domain/test.aspx#page2.
onmessage	Событие происходит, когда сообщение получено через источник события.
onoffline	Событие вызывается браузером в том случае, когда браузер определит, что соединение с интернет пропало.
ononline	Событие вызывается браузером в том случае, когда соединение с интернет возобновилось.
onpagehide	Событие происходит, когда пользователь покидает страницу посредством навигации, например, нажав на ссылку, обновив страницу, заполнив форму и т.д.
onpageshow	Событие происходит, когда пользователь переходит на веб-страницу, после события onload.
onunload	Событие срабатывает если страница не загрузилась по каким-либо причинам, либо при закрытии окна браузера.

*Для элемента **<body>** доступны глобальные атрибуты.*

*До появления стандарта HTML5 вся разметка страниц осуществлялась преимущественно с помощью элементов ** **, которым присваивали классы **class** или идентификаторы **id** для наглядности разметки (например, ** **). С их помощью в HTML-документе размещали верхние и нижние колонтитулы, боковые панели, навигацию и многое другое.*

Стандарт HTML5 предоставил новые элементы для структурирования, группировки контента и разметки текстового содержимого. Новые семантические элементы позволили улучшить структуру веб-страницы, добавив смысловое значение заключенному в них содержимому (было *** ****, стало* ***<header>****). Для отображения внешнего вида элементов не задано никаких правил, поэтому элементы можно стилизовать по своему усмотрению. Для всех элементов доступны глобальные атрибуты.*

Согласно спецификации HTML5 каждый элемент принадлежит к определенной (ноль или более) категории. Каждая из них группирует элементы со схожими характеристиками. Выделяют следующие общие категории:

- мета содержимое;
- потоковое содержимое: <header>, <nav>, <article>, <section>, <aside>, <footer>, <address>, <main>, <figure>, <time>, <mark>, <bdi>, <wbr>, <ruby>;
- секционное содержимое: <nav>, <article>, <section>, <aside>, <figure>;
- заголовочное содержимое;
- текстовое содержимое: <time>, <mark>, <bdi>, <wbr>, <ruby>;
- встроенное содержимое;
- интерактивное содержимое.

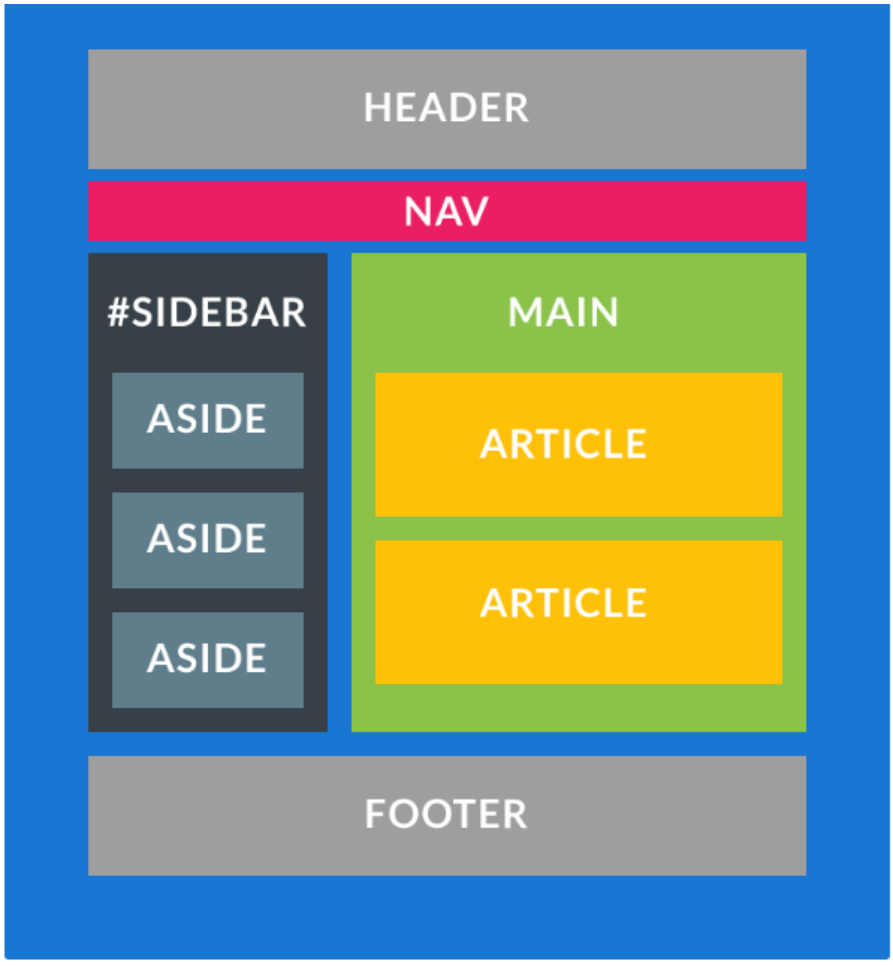


Рисунок — Семантическая структура для HTML5 страницы.

Элемент **<header>** группирует вводные и навигационные элементы, не является обязательным. Может содержать заголовки, оборачивать содержание раздела страницы, форму поиска или логотип. В HTML-документе может содержаться одновременно несколько

элементов **<header>** и они могут располагаться в любой части страницы. Элемент **<header>** нельзя помещать внутрь элементов **<footer>**, **<address>** или другого элемента **<header>**.

```
<header>
  <h1>Site description</h1>
  <nav>
    <ul>
      <li><a href="">About</a>
      <li><a href="">Forum</a>
      <li><a href="">Download</a>
    </ul>
  </nav>
</header>
```

Элемент **<nav>** предназначен для создания блока навигации веб-страницы или всего веб-сайта, при этом не обязательно должен находиться внутри **<header>**. На странице может быть несколько элементов **<nav>**. Не заменяет теги **<ul>** или **<ol>**, он просто их обрамляет. Не все группы ссылок на странице должны быть обёрнуты **<nav>**, этот элемент предназначен в первую очередь для разделов, которые состоят из главных навигационных блоков.

```
<nav>
  <ul>
    <li><a>...</a></li>
    <li><a>...</a></li>
    <li><a>...</a></li>
  </ul>
</nav>
```

В качестве элементов панели навигации можно использовать не только элементы списков:

```
<nav>
  <p><a href="">...</a></p>
  <p><a href="">...</a></p>
</nav>
```

Также можно добавлять заголовки внутрь элемента:

```
<nav>
  <h2>...</h2>
    <ul>
      <li><a>...</a></li>
      <li><a>...</a></li>
      <li><a>...</a></li>
    </ul>
</nav>
```

Элемент **<article>** используется для группировки записей — публикаций, статей, записей блога, комментариев. Представляет собой независимый обособленный блок, предназначенный для многократного использования, как правило, начинается с заголовка. Может дублироваться на других страницах сайта и содержать внутри другие элементы **<article>**, которые по содержанию имеют близкое отношение к содержанию внешней статьи. Если на странице присутствует только одна статья с заголовком и текстовым содержимым, она не нуждается в обёртке элементом **<article>**. Элемент рекомендуется использовать только в том случае, если содержимое элемента будет явно указано в схеме документа.

```
<article>
  <header>
    <h2>...</h2>
```

```
  </header>
  <p>...</p>
  <footer>
  Опубликовано в категории<a href="">Музыка</a>.
    <a href="">0 комментариев</a>
  </footer>
</article>
```

Элемент <**section**> представляет собой универсальный раздел документа. Группирует тематическое содержимое и обычно содержит заголовок. Не является блоком-оберткой, для этих целей уместнее использовать элемент <**div**>. В качестве содержимого может выступать оглавление, разделы научных публикаций, программа мероприятия. Домашняя страница сайта также может быть поделена на секции – блок вводной информации, новости и контакты. Элемент рекомендуется использовать только в том случае, если содержимое элемента будет явно указано в схеме документа. Например, можно сделать на одной странице секцию статей, секцию комментарий, секцию галереи и т.д.

```
<article>
  <h1>...</h1>
    <section>
      <h2>...</h2>
      <p>...</p>
    </section>
    <section>
      <h2>...</h2>
      <p>...</p>
    </section>
    <p>...</p>
</article>
```

Можно создавать родительские элементы <**section**> с вложенными элементами <**article**>, в которых есть один или несколько элементов <**article**>. Не все страницы должны быть устроены именно так, но это допустимый способ вложения элементов. Например, основная область контента страницы содержит два блока со статьями разной тематики. Можно сделать на этом акцент, поместив каждую статью одной тематики внутрь элемента <**section**>.

```
<section>
  <h1>Заметки о природе</h1>
    <article>
      <h2>...</h2>
      <p>...</p>
    </article>
    <article>
      <h2>...</h2>
      <p>...</p>
    </article>
</section>
<section>
  <h1>Исторические заметки</h1>
    <article>
      <h2>...</h2>
      <p>...</p>
    </article>
    <article>
      <h2>...</h2>
      <p>...</p>
    </article>
</section>
```

Элемент **<aside>** группирует содержимое, связанное с окружающим его контентом напрямую, но которое можно счесть отдельным *(т.е., удаление этого блока не повлияет на понимание основного содержимого)*. Чаще всего элемент позиционируется как боковая колонка (как в книгах) и включает в себя группу элементов: **<nav>**, цифровые данные, цитаты, рекламные блоки, архивные записи. Не подходит для блоков, просто позиционированных в стороне.

```
<aside>
  <h2>...</h2>
  <p>...</p>
</aside>
<aside>
  <h2>...</h2>
  <p>...</p>
  <blockquote>
    <p>...<cite>...</cite>...</p>
    <p>...</p>
  </blockquote>
</aside>
```

Элемент **<footer>** представляет собой нижний колонтитул содержащей его секции или корневого элемента. Обычно содержит информацию об авторе статьи, данные о копирайте и т.д. Если используется как колонтитул всей страницы, содержимое дополняется сведениями об авторских правах, ссылками на условия использования, контактную информацию, ссылками на связанное содержимое и т.п.

В одном веб-документе может быть несколько элементов **<footer>**. Как каждая страница, так и каждая статья может иметь свой элемент **<footer>**, более того, **<footer>** можно поместить в элемент **<blockquote>**, чтобы указать источник цитирования.

```
<footer>
  <address>...</address>
  <small>@2014...</small>
</footer>
```

Элемент **<address>** используется для определения контактной информации автора/владельца документа или статьи. Для обозначения автора документа тег размещают внутри элемента **<body>**, для отображения автора статьи – внутри тега **<article>**. В браузере обычно отображается курсивом.

Элемент **<main>** представляет основное содержимое документа (содержимое элемента **<body>**). Контент, находящийся внутри элемента, должен быть уникальным и не повторяться во всех документах сайта, таких как навигационные ссылки, информация о копирайте, логотипы, формы поиска (в случае, если форма поиска является основной функцией документа).

Элемент <main> не может быть потомком таких элементов как **<article>, <aside>, <footer>, <header>** или **<nav>**.

```
<body>
<header>
  <h1>Пудель</h1>
    <nav>
      <ul>
        <li><a href="index.html">Главная</a></li>
        <li><a href="about.html">О породе</a></li>
        <li><a href="health.html">Здоровье</a></li>
      </ul>
```

```
    </nav>
  </header>
  <main>
    <section>
      <header>
        <h2>О породе</h2>
          <nav>
            <ul>
              <li><a href="#basic">Разновидности</a></li>
              <li><a href="#app">Внешний вид</a></li>
              <li><a href="#temp">Характер</a></li>
            </ul>
          </nav>
      </header>
      <section id="basic">
        <h3>Разновидности</h3>
          <p>...</p>
      </section>
      <section id="app">
        <h3>Внешний вид</h3>
          <p>...</p>
      </section>
      <section id="temp">
        <h3>Характер</h3>
          <p>...</p>
      </section>
      <footer>
        <a href="#basic">Разновидности</a>
        <a href="#app">Внешний вид</a>
        <a href="#temp">Характер</a>
      </footer>
      </section>
    </main>
    <footer>
      <small>Copyright © <time datetime="2016">2016</time> Моя собака.ру</small>
    </footer>
</body>
```

Элемент **<figure>** представляет автономный контент (необязательно с заголовком), являющийся самостоятельным элементом основного потока. Элемент может быть перемещен из основного содержимого документа в боковую колонку или приложение, не затрагивая поток документа. С помощью элемента **<figure>** можно добавлять краткие характеристики к иллюстрациям, фотографиям, диаграммам, фрагментам кода и т.д.

```
<figure>
    <img src="picture.jpg" alt="Осень">
    <figcaption>Осенний лес</figcaption>
</figure>
```

Элемент <figure> является блочным, по ширине занимает всю ширину блока-контейнера за минусом внешних отступов **margin**:

```
margin-left: 40px;
margin-right: 40px;
margin-top: 1em;
margin-bottom: 1em;
```

Элемент **<figcaption>** - потомок элемента **<figure>**, не принадлежит ни к одной категории контента. Элемент является блочным, по ширине равен ширине элемента **<figure>**, высота по умолчанию равна 18px.

Элемент **<time>** определяет время (24 часа) или дату по григорианскому календарю с возможным указанием времени и смещения часового пояса. Текст, заключенный в данный тег, не имеет стилевого оформления браузером. Для тега доступен атрибут **datetime**, в качестве содержимого которого указывается то, что будет видеть пользователь на экране своего компьютера:

```
<time datetime="2014-09-25"> 25 Сентября 2014</time>
```

С появлением данного тега, стало возможным задавать дату в стандартных временных форматах.

Чтобы дата могла считываться автоматически, она должна быть в формате YYYY-MM-DD. Время, которое также может указываться, задается в формате HH:MM с добавлением разделяющего префикса **T (time)**:

```
<time datetime="2014-09-25T12:00"> 25 Сентября 2014</time>
```

Элемент **<mark>**. Текст, помещенный внутрь тега <mark>, выделяется по умолчанию желтым цветом (цвет фона и цвет шрифта в выделенном блоке можно изменить, задав определенные css-стили) , как маркером на бумаге. С помощью данного тега можно отмечать важное содержимое, а также ключевые слова.

Элемент **<bdi>** отделяет фрагмент текста, который должен быть изолирован от остального текста для двунаправленного форматирования текста. Используется для текстов, написанных одновременно на языках, читающихся слева направо и справа налево.

Элемент **<wbr>** - одиночный тег, показывает браузеру место, где можно добавить разрыв длинной строки в случае необходимости.

*Элемент **<ruby>** позволяет помечать один и более элементов категории текстовое содержимое с помощью ruby-аннотации. Ruby-аннотация используется в преимущественно в Восточно-Азиатской типографики как руководство по произношению или для включения других характеристик. Элемент может содержать:*

*— один и более текстовых узлов или элементов **<rb>**;*

*— один и более элементов **<rt>, <rtc>**, каждый из которых предшествует или следует непосредственно за элементом **<rp>**.*

*Элементы **<rb>, <rt>, <rtc>** и **<rp>** не относятся ни к одной категории контента.*

*Элемент **<rb>** определяет вложенный в него текст как базовый компонент аннотации.*

*Элемент **<rt>** выводит аннотацию к тексту сверху или снизу над ним.*

*Элемент **<rtc>** отмечает вложенный в него текст как дополнительную аннотацию.*

*Элемент **<rp>** выводит альтернативный текст в случае если браузер не поддерживает элемент **<ruby>**.*

***Важное замечание по использованию HTML5 тегов.** В спецификации не указаны жесткие правила по использованию семантических тегов, указаны лишь рекомендации п их использованию. Если вы не понимаете или не знаете где и какой HTML5 тег использовать, лучше используйте **div** — чтобы не навредить и не нарушить структуру документа.*

Семантика в HTML5 не ограничивается приведенными выше в статье тегами. Но используя приведенные примеры вы можете освежить свою разметку, и сделать сайт более

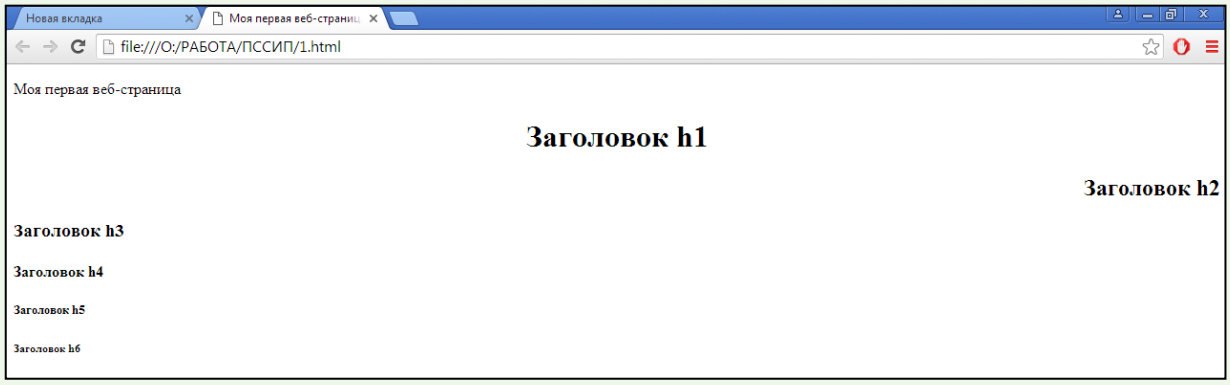
дружелюбным к поисковым системам, что скажется на более высоком рейтинге сайта в поисковой выдаче.

В продолжение темы можно изучить другие новые HTML5 теги. А также микро форматы для оформления и структуризации данных, например такие как schema.org

HTML не задает конкретные и точные атрибуты форматирования документа. Конкретный вид документа окончательно определяет только *браузер* на компьютере пользователя Интернета.

Пример использования тегов:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<title>Моя первая веб-страница</title>
</head>
<body>
<p align="left">Моя первая веб-страница</p>
<h1 align="center">Заголовок h1</h1>
<h2 align="right">Заголовок h2</h2>
<h3 align="justify">Заголовок h3</h3>
<h4>Заголовок h4</h4>
<h5>Заголовок h5</h5>
<h6>Заголовок h6</h6>
</body>
</html>
```



Физическое и логическое форматирование текста на HTML-странице

Форматирование текста — средства его изменения, такие как выбор начертания шрифта и использование эффектов, позволяющих менять вид текста.

Для форматирования текста HTML-документов предусмотрена целая группа тэгов, которую можно условно разделить на 2 группы:

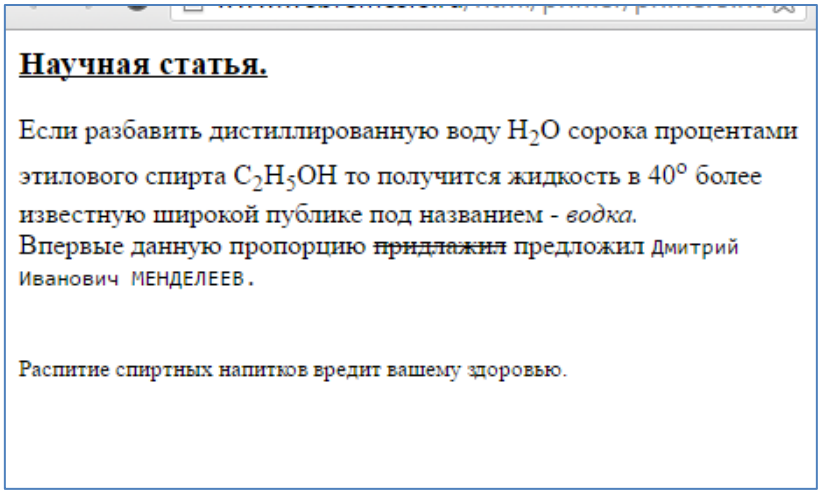
- теги логического форматирования;
- теги физического форматирования.

Тэги физического форматирования определяют формат отображения указанного в них фрагмента текста в окне браузера (согласно предпочтениям автора документа). Например, для отображения фрагмента курсивом можно использовать тег курсива <I>.

Пример:

```
<html>
<head>
<title>Стили текста</title>
</head>
```

```
<body>
<big><b><u>Научная статья.</u></b></big>
<br>
<br>
Если разбавить дистиллированную воду H<sub>2</sub>O сорока процентами этилового спирта C<sub>2</sub>H<sub>5</sub>OH то получится жидкость в 40<sup>o</sup> более известную широкой публике под названием <i>- водка.</i>
<br>
Впервые данную пропорцию <s>придлажил</s> предложил <tt>Дмитрий Иванович МЕНДЕЛЕЕВ.</tt>
<br>
<br>
<br>
<small>Распитие спиртных напитков вредит вашему здоровью.</small>
</body>
</html>
```



К тегам физического форматирования относятся:

<B>полужирный шрифт **</B>**;

<I>выделяет курсивом**</I>;**

<TT>моноширинный шрифт **</TT>**;

<U>подчеркивание текста **</U>;**

<STRIKE>зачеркнутый текст **</STRIKE>;**

<S>зачеркнутый текст **</S>;**

<BIG>Шрифт большего размера **</BIG>;**

<SMALL> Шрифт меньшего размера **</SMALL>;**

_{**шрифт для нижнего индекса **};

^{** шрифт для верхнего индекса **};

<PRE>текст в том виде, в котором набран, т.е. со всеми пробелами и переносами строк**</PRE>;**

<SPAN> является аналогом тега уровня блока **<DIV>;**

<P> для указания границ абзаца (выравнивание задается свойством **ALIGN**) **</P>;**

<FONT> указывает параметры шрифта (свойство **SIZE** определяет размер текста, **FACE** задает имя шрифта, **COLOR** задает цвет текста, **STYLE** позволяет выделить текст цветовым фоном)**</FONT>.**

*Свойство **SIZE** определяет размер текста, значения размера от 1 (мелкий) до 6 (крупный), по умолчанию **SIZE=3**. Свойство **FACE** задает имя шрифта. Можно задавать несколько имен шрифтов через запятую. Если у пользователя не окажется первого шрифта, то браузер будет подставлять второй и т.д. Свойство **COLOR** задает цвет текста. Можно использовать как имена цветов (red, blue, green) так и номер в виде #RRGGBB, где*

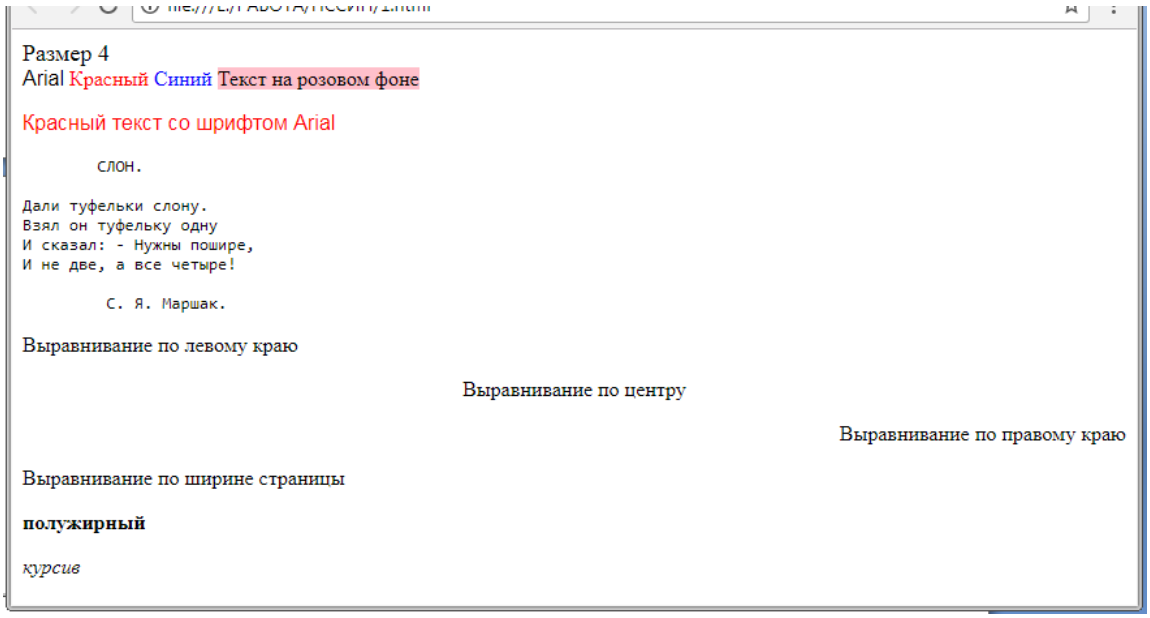
RR - концентрация красного, GG - зеленого, BB - синего. Выделение текста цветовым фоном достигается использованием свойства ***STYLE="backgroud-color: цвет"***.

Пример:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
<title>Теги физического форматирования текста</title>
</head>
<body>
<FONT SIZE="4">Размер 4</FONT>
<br>
<FONT FACE="Arial">Arial</FONT>
<FONT COLOR="red">Красный</FONT>
<FONT COLOR="#0000FF">Синий</FONT>
<FONT STYLE="background-color: pink">Текст на розовом фоне</FONT>
<p><font color="#FF0000" face="Arial">Красный текст со шрифтом Arial</font></p>
<pre>
        СЛОН.

Дали туфельки слону.
Взял он туфельку одну
И сказал: - Нужны пошире,
И не две, а все четыре!

         С. Я. Маршак.
</pre>
<P ALIGN="left"> Выравнивание по левому краю</P>
<P ALIGN="center"> Выравнивание по центру</P>
<P ALIGN="right"> Выравнивание по правому краю</P>
<P ALIGN="justify"> Выравнивание по ширине страницы.
<p><B>полужирный</B></p>
<p><I>курсив</I></p>
</body>
</html>
```



Теги форматирования могут быть вложенными друг в друга. При этом нужно внимательно следить, чтобы один контейнер находился целиком в другом контейнере.

Основная задача тегов физического форматирования текста это выполнение сугубо декоративных функций.

Тэги логического форматирования обозначают (своими именами) структурные типы своих текстовых фрагментов, такие, например, как программный код (тэг <CODE>), цитата (тэг <CITE>), аббревиатура (тэг <ABBR>) и т. д. С помощью тегов <EM> и <STRONG> можно, например, отметить отдельные фрагменты как выделенные, или сильно выделенные. Заметим, что речь идет о структурной разметке, которая не влияет на конкретное экранное представление фрагмента браузером. Поэтому такая разметка и называется логической. Фрагменты с логическим форматированием браузеры отображают на экране определенным образом, заданным по умолчанию. Вид отображения никак не связан со структурным типом фрагмента (т. е. именем тега логического форматирования), но может быть легко переопределен.

К тегам логического форматирования относятся:

<abbr>CSS**</abbr>** - **в**ыделяет в тексте аббревиатуру. Браузерами обычно подчеркивается пунктирной линией;

<acronym>комсомол**</acronym>** – выделяет в тексте акроним. *Акроним это почти то же самое что и аббревиатура, только образованная из начальных букв, слов или словосочетаний, произносимая как единое слово, а не побуквенно*;

<address>Вася Пупкин г. Урюпинск Макаронная фабрика 2010г.**</address>** - указывает автора документа и его адрес. Обычно отображается курсивом. *Предназначен для поисковых систем для сбора информации об авторе его адресе и прочей информации владельца/цев сайта;*

<cite>Лед тронулся!**</cite>** - выделяет в тексте цитату или сноску на другой документ. Обычно браузеры отображают её курсивом;

<code>function()**</code>** - используется для отображения фрагментов программного кода. Обычно отображается моноширинным шрифтом;

~~Старая цена 1000р.**~~**Новая 999 р.!!! - выделяет удалённый текст в новой версии документа. Выделенный текст станет перечёркнутым;

<dfn>Акроним**</dfn>** - аббревиатура, образованная из начальных букв, слов или словосочетаний, произносимая как единое слово, а не побуквенно. Выделяет текст как определение. *Как правило, когда в тексте встречается новый термин, авторы выделяют его курсивом и дают его определение, собственно для этого и нужен тег* ***<dfn>;***

~~Старая цена 1000р.**~~** -**<ins>**Новая 999 р.**</ins>**!!! – выделяет новый текст в новой версии документа. Выделенный текст в большинстве браузеров станет подчёркнутым. *Противоположен по значению тегу* ***<del>*** *с ним же в паре обычно он и используется*;

****Как создать сайт?**** - выделяет особенно важный фрагмент текста. Обычно отображается курсивом;

<kbd>Ctrl + Z**</kbd>** - указывает текст, вводимый с клавиатуры, кроме того, используется для названия клавиш. Обычно отображается моноширинным шрифтом;

<q>Лед тронулся!**</q>** - выделяет в тексте цитату. *В отличие от тега <cite>* цитата, обозначенная тегом **<q>**, автоматически берётся браузерами в кавычки;

<samp>4**</samp>** - обозначает текст, который выводится на экран в результате работы какой-либо программы. Обычно браузерами отображается моноширинным шрифтом;

****Как создать сайт?**** - выделяет особенно важный фрагмент текста. Обычно отображается полужирным;

<var>$count**</var>** - выделяет текст, как переменную в работе какой либо программы. Обычно браузерами отображается курсивом.

Практически ко всем тегам применим атрибут **title** - всплывающая подсказка, поэтому, если в тексте необходимо выделить некую аббревиатуру, желательно давать к ней расшифровку, используя данный атрибут.

Пример:

```
<abbr title="Cascading Style Sheets">CSS</abbr> позволит Вам без труда изменить стиль любого тега логического форматирования текста!
<acronym title="коммунистический союз молодёжи">комсомол</acronym>
```

Под логической разметкой документа, прежде всего, принято понимать разметку текста тегами направленную на удобство работы поисковых систем и других программ, которые работают с данным документом, а так же придавать тексту в документе некий смысл. Спецификация HTML не говорит о том, что теги логического форматирования текста должны определённым образом отображаться браузерами, *что вот, например, тот же тег* ****** *должен быть отображен именно курсивом и не как иначе,* а говорит лишь о том, что такой текст должен отличатся от основного и особым образом выделятся в документе.

Кроме того, теги логического форматирования можно переопределить с использованием CSS.

Поэтому, если вы хотите не просто отформатировать текст, но и передать или подчеркнуть смысл слов, встречающихся в HTML документе, то вам необходимо использовать тэги логического форматирования.

С развитием технологий, как браузеров, так и поисковых систем, возникла потребность в том, чтобы внести какой-то ***смысл в HTML тэги и в слова****, которые отображаются на наших HTML страницах. Во-первых, чтобы поисковые системы «понимали»* ***смысл и важность тех или иных слов, которые встречаются в HTML документе****, а во-вторых, то же самое, но для браузеров, которые используют люди с ограниченными возможностями.*

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="ru">
<head>
<meta content="text/html; charset=windows-1251" http-equiv="content-type">
<title>Физическое и логическое форматирование</title>
</head>
<body>
<p>Обычный текст<br>
<b>Физическое форматирование - полужирное начертание</b><br>
<i>Физическое форматирование - курсив</i><br>
<u>Физическое форматирование - подчёркивание</u><br>
<strong>Логическое форматирование - важный фрагмент выделен полужирным начертанием</strong><br>
<em>Логическое форматирование - важный фрагмент выделен курсивом</em>
</body>
</html>
```

Обычный текст
Физическое форматирование - полужирное начертание
Физическое форматирование - курсив
Физическое форматирование - подчёркивание
Логическое форматирование - важный фрагмент выделен полужирным начертанием
Логическое форматирование - важный фрагмент выделен курсивом

Между разработчиками HTML-документов долгое время шли споры о преимуществах и недостатках того или иного подхода. С выходом спецификации HTML 4.0 эти споры завершились в пользу применения логического форматирования, поскольку был провозглашен принцип отделения структуры документа от его представления.

Действительно, только на базе логического форматирования можно гибко управлять представлением документа, используя современные методы (основанные на таблицах стилей, динамически изменяющихся документах и т. д.).

Форматирование текста оказывает влияние и на продвижение.

Поэтому ***важно научиться не просто выделять слова в HTML****, а использовать правильные HTML тэги, которые* ***правильно передают назначение и смысл слов в HTML документе****.*

Закрепление знаний по изучаемой теме:

Создайте веб-страницу в соответствии с макетом: