Устройство сайта.

Index.html в данном примере в нем содержится весь код. Его принято называть именно так, хотя назвать можно, как угодно. Код можно посмотреть в нем, но можно и иначе. Если на странице сайта нажать **f12**, то откроется код страницы. Нажав на эту кнопку можно нажимать на элементы, чтобы сразу найти их в коде.

Работой сайтов управляют **протоколы,** назовем это пока что просто программными инструкциями. Кстати, а вот этой кнопкой можно преобразовать сайт в мобильный.

Протокол **FTP** для передачи файлов. Чтобы посылать и получать **SMTP, POP3, IMAP.** Чтобы давать доступ к веб сайтам **HTTP.**

Сайт — это объединение нескольких веб-страниц. Раньше на страницах только текст. Но теперь и картинки, и видео, и аудио.

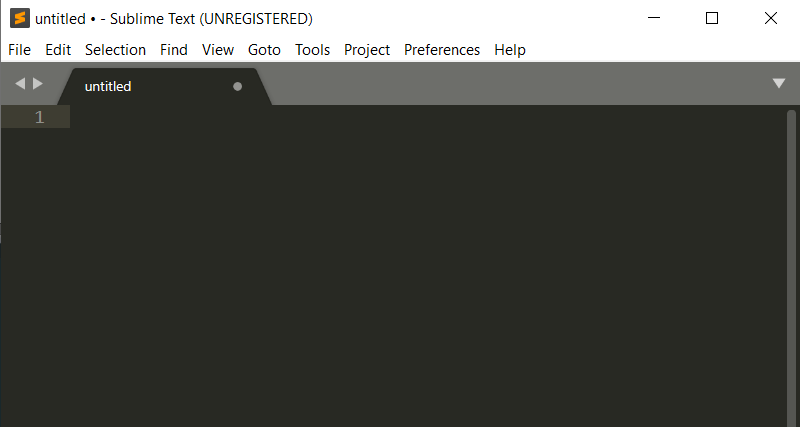
**HTML –** язык гипертекстовой разметки. Гипертекст – текст с ссылками, которые ведут на какое-то место в документе, другой документ или сайт. С помощью него мы определяем структуру расположения контента на сайте.

Когда браузер открывает правильный html код, он превращает его в результат, то есть в сайт или веб страницу. В отличии от других языков программирования в html можно допускать ошибки и сайт даже откроется, просто он может отображаться неправильно.

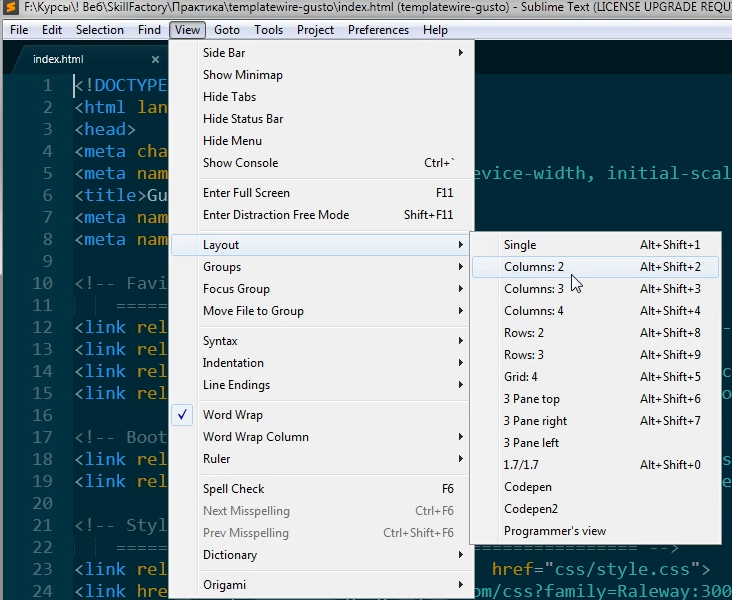
Работать будем в Google Chrome, хотя правильнее будет проводить тестирование сразу в нескольких браузерах.

Существует несколько стандартов написания html кода, но мы будем рассматривать последний **HTML5.**  Я хотел писать в VSCode, но в этом курсе предлагают **Sublime.** Ладно.

Вот, как он выглядит.



Если нажать дважды на серую панель, то добавится новая вкладка. А так обычный редактор. Ничего такого. Можно одновременно отображать несколько вкладок, это делается в этом меню.



Также есть разные сочетания клавиш. **Ctrl S – save. Ctrl O – open**  и т.д.

Сайт где много вопросов и ответов по программированию. <https://stackoverflow.com/>

**HTML** состоит из элементов, которые обозначаются тегами: **<>**

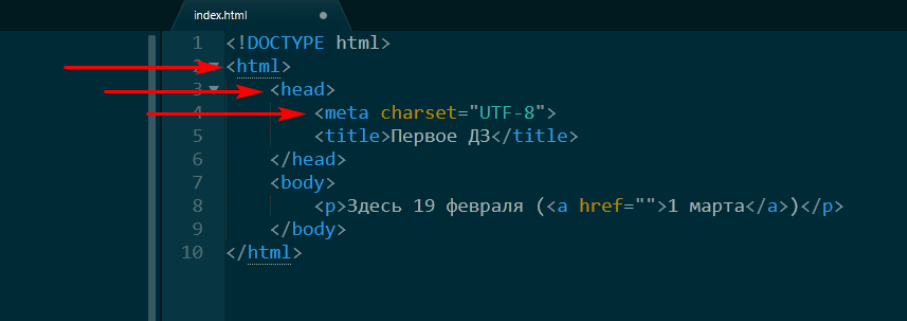
У любого элемента их минимум 1, но чаще 2. Открывающий **<>** и закрывающий **</>**, а между ними содержимое. Важно закрывать **тег** там, где нужно.

**Пример:** <p>Текст параграфа, который увидит пользователь</p>

У элементов могут быть **атрибуты** – дополнительные настройки, которые пишутся внутри открывающегося тега.

Элементы могут быть вложены друг в друга, это отображается **отступами**. Те, что лежат глубже **дочерние** или **потомки,** а те, что выше **родители.** Это нужно, чтобы позже внедрить **CSS** или для \***семантики?** Или для использования одного элемента в другом. С отступами легче читать код. **Но сами по себе на код они не влияют!**

**Пример:**



**Главные элементы**

**<!DOCTYPE html>** - тег сообщает браузеру, что документ написан на современной версии HTML5

<**html**> - элемент содержит остальные элементы html, обычно их **2**, **head и body.** Через атрибут **lang** ему можно указать язык **<html lang="en-US">** .

**<head>** здесь расположены элементы, отвечающие за работу и настройку страницы. Большую часть этого пользователь не увидит.

**<body>** - Здесь располагается то, что увидит пользователь, **контент** страницы.

**Важные элементы**

**<meta>** - универсальный элемент, в атрибутах <**meta charset=”кодировка”> .** Здесь мы указываем кодировку страницы, **UTF-8** обычно. Кстати, этот **тег не закрывают.**

**<title> -** заголовок страницы. Именно это название будет видно в браузере и в закладках.

**<h1> … </h1> -** заголовок первого уровня. У любой страницы следует сделать главный заголовок.

**<p> -** параграф, блок с информацией, абзац.

**Элементы для форматирования**

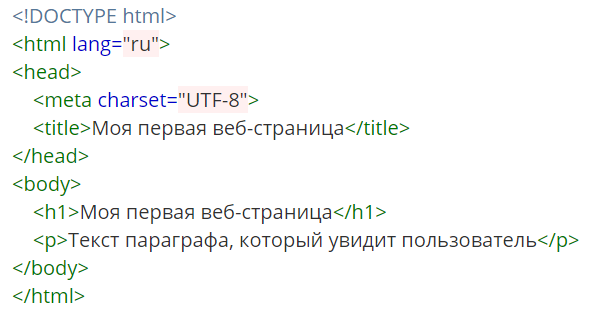
**<strong>** выделяет слово или предложение **полужирным**. Пример: <p>Меня зовут Вася. Я работаю <strong>маркетологом</strong>. </p>

**<em> - по умолчанию** **курсивное** написание слова или фразы.

Семантика – смысл.

**<title>** можно назвать также как заголовок первого уровня **<h1>.** При этом заголовков может быть несколько, в таком случае используем заголовки уровнем ниже. Существует шесть уровней заголовков. Чем меньше цифры, тем больше размер.

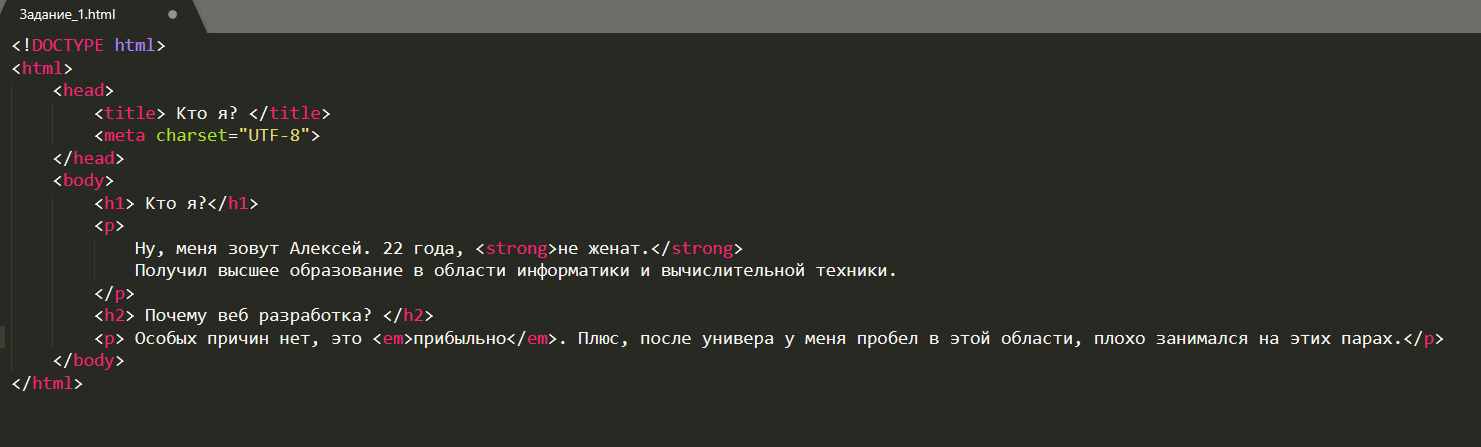
Вот пример необходимого минимума, для работающей страницы:



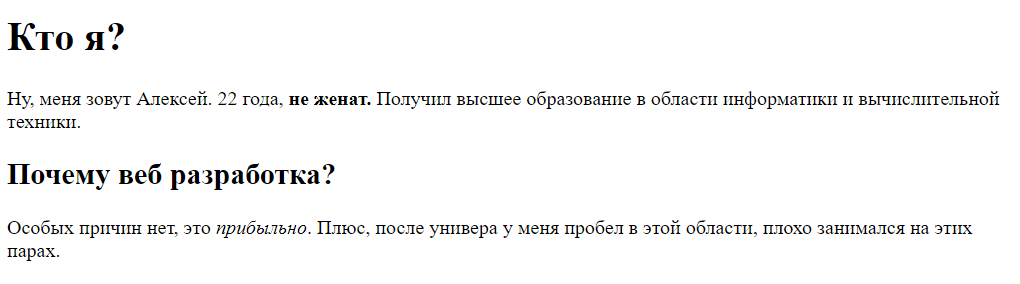
**P.S. Специально вставляю картинку, чтобы писать, а не копировать. Так развивается память и легче запоминается.**

Ладно первое задание, создать страницу. О себе и чему хочу научиться. В заголовках вопрос. В элементе ответ.

Наверно когда-нибудь я пожалею, что не вставляю сюда код.



Результат.



**Специальные символы**

**<del> -** зачеркивает текст. **<del>9 999</del>**

**<sup> -** делает символу верхний регистр: Пример:

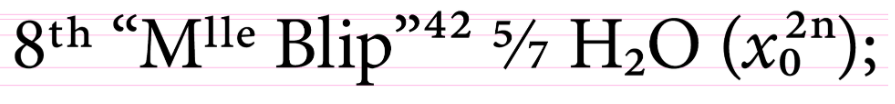
a<sup>2</sup> + b<sup>2</sup> = c<sup>2</sup>



**<sub> -** делает символу нижний регистр. Пример:

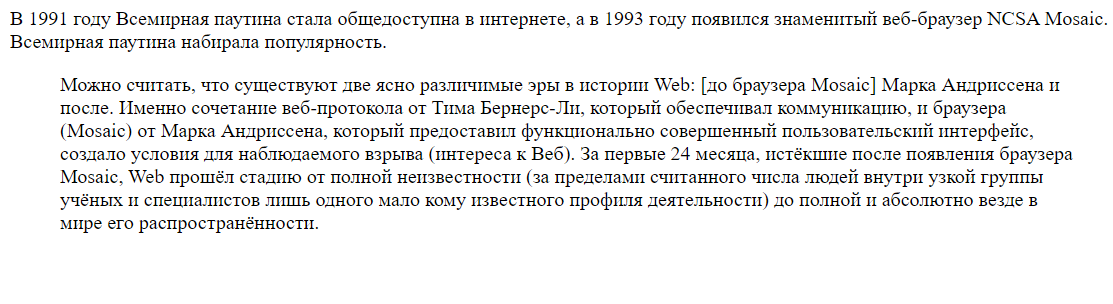
C<sub>8</sub>H<sub>10</sub>N<sub>4</sub>O<sub>2</sub>,

Совмещая их удобно делать такие конструкции:



**<q> -** для цитат. По умолчанию просто оборачивает текст в кавычки.

**<blockquote>** - для цитат, но для целых блоков, выделенная область отодвигается от левого края, формируя своеобразный блок. Пример:



И хотя по умолчанию у этих тегов такие функции, эти **стили** оформления можно будет менять.

Зачем ставить кавычки таким образом? Дело в том, что они относятся к специальным символам и если использовать обычные кавычки, что-то может пойти не так. Вот пример этих символов и как их добавить в текст:

**&lt** — левая угловая скобка, начало тега (**<**),

**&gt**— угловая скобка, конец тега (**>**),

**&quot** — двойная кавычка (**“**),

**&mdash** — длинное тире (**—**),

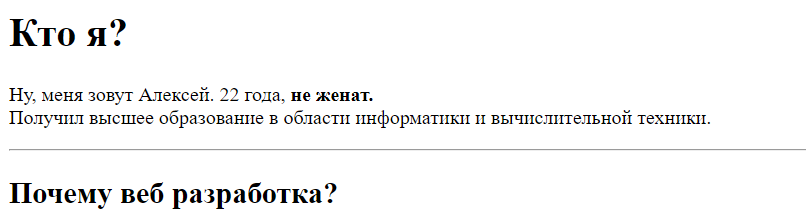
**&nbsp** — неразрывный **пробел**.

И их вообще-то много. Можно глянуть здесь <https://unicode-table.com/ru/#basic-latin>

**Другие элементы для форматирования.**

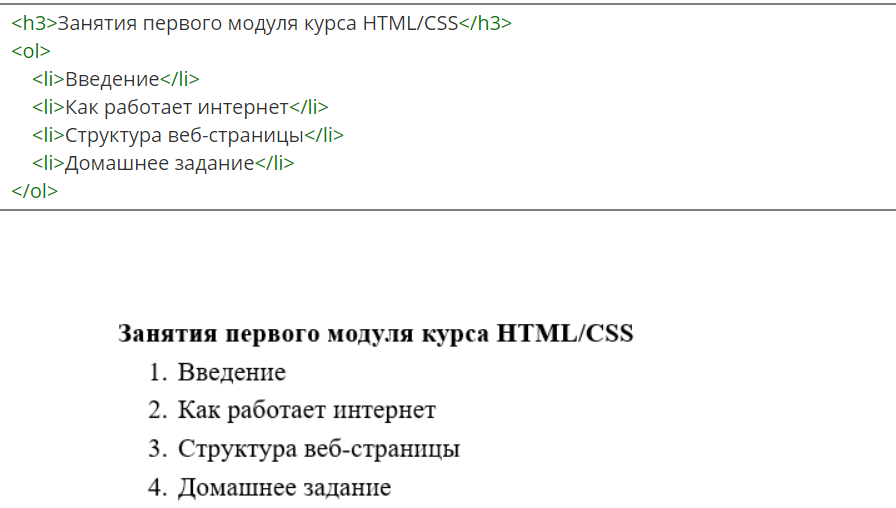
**<br> -** Перенос строки, положено использовать внутри параграфов <p>.

**<hr> -** Разрыв между параграфами, создает длинную горизонтальную линию. Пример:

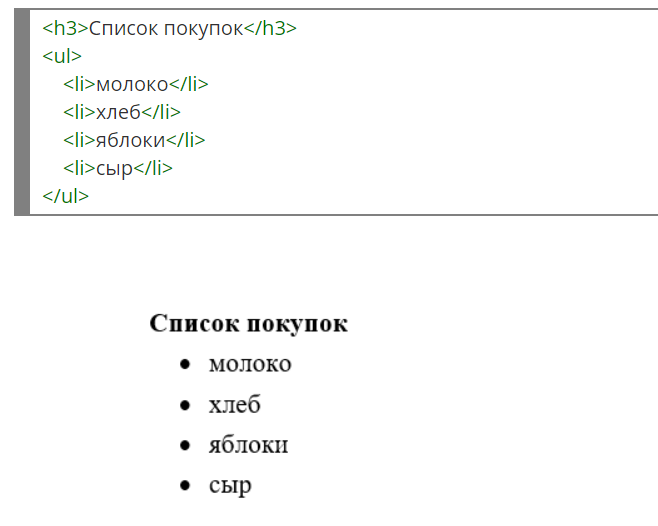


**<ol> - Упорядоченный список**. Используя теги **<li>** можно создать несколько пунктов в таком списке и они про нумеруются.

**<li> -** Элемент любого списка. Пример:

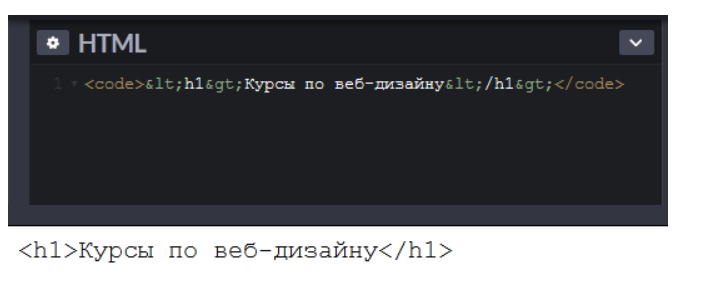


**<ul> - неупорядоченный** список. Пример:



**<pre> -** удобный тег, в нем не учитываются специальные символы, а значит, как текст выглядит в коде, так он и на странице отобразится.

**<code> -** тег для кода, но в нем придется ставить спец. символы ставить вручную. Не удобно.



**Изображения на сайте.**

Еще пару важных элементов сайта. Ссылки и изображения.

**<img>** - картинки, но не те, что являются частью фона. Фон научимся делать через **CSS.** Чтобы вставить картинку нужно еще дополнительно указать атрибут **src** и адрес и название картинки. Бывает два пути, **относительный** и **абсолютный**.

Если адрес картинки глубже, что адрес html файл, то писать так: <img src="images/photo.jpeg"> А если наоборот, картинка выше, то:

<img src="../images/photo.jpeg">

А кроме того, можно указывать и абсолютный путь. Причем слешы могут быть в обе стороны.

**<img src="C:/Users/Booklove/Downloads/12.png">**

**<img src="C:\Users\Booklove\Downloads\12.png">**

Еще один обязательный атрибут **alt=””** В нем можно указать текст и этот текст будет отображаться, если картинка не загрузилась.





И два последних атрибута ширина и высота:

width="300" height="200"

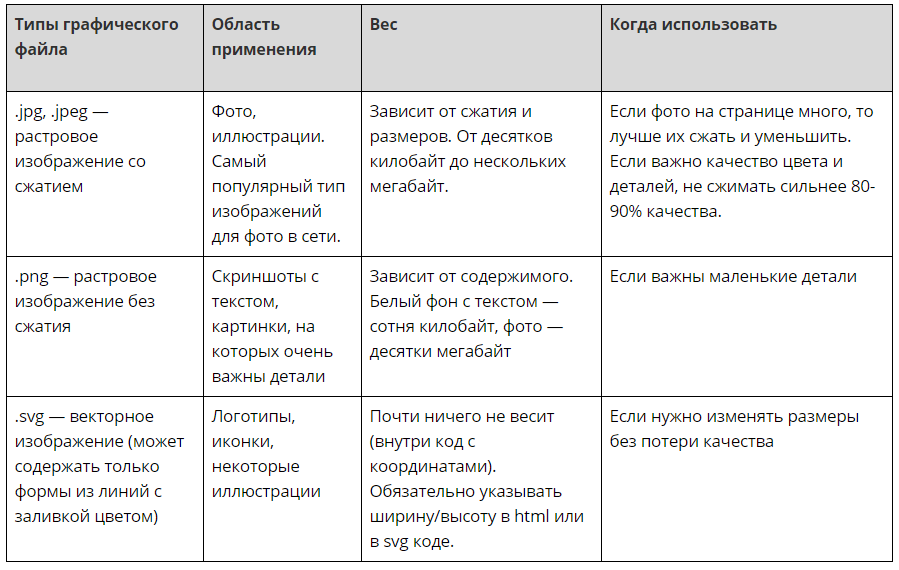
Вообще, размеры и внешний вид задаются в **CSS,** но лучше перестраховаться на случай если стили не загрузятся.

**Важно:**

Между атрибутами должен стоять **пробел**.

Синтаксис атрибута: **атрибут=””**

Чем больше весит картинка, тем дольше она грузится. На вес влияют: тип файла, сжатие, размеры, содержимое.



**Ссылки.**

**<a> -** ссылка — это интерактивный текст, который переносит на другую страницу, сайт или часть страницы. Имеет атрибут **href** в котором мы указываем адрес куда перенестись. **Важно:** чтобы этот тег работал, нужно добавить само **текстовое содержимое** и **закрыть тег.**

**Пример: <a href="https://vk.com/id76376728"> Буклов </a>**

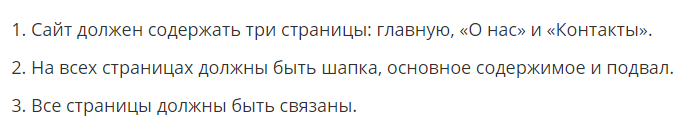


Выше указан пример **абсолютного пути**. На другой сайт. А бывает также **относительный путь**, когда переходим ну другой **html** этого же сайта.

**Пример: <a href="pages/catalog.html">Каталог товаров</a>**

Следующим заданием стало создание еще одной странице с ссылкой на основную и добавлением на основную страницу ссылку на эту.

А теперь задание **по сложнее:**



Но пока просто по шаблону.

Во-первых, на основной странице внутри **<body>** нужно разместить такие теги:

**<header>** - тег шапка сайта, там обычно размещают название, логотип, регистрационную панель и тд. Обычно не сильно отличается на разных сайтах.

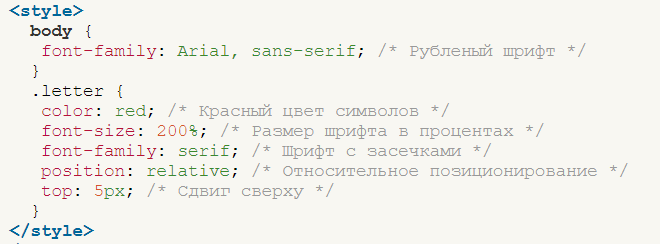
**<main> -** в этом теге хранится весь основной код с контентом сайта.

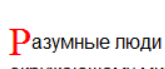
**<footer> -** подвал сайта, там обычно контактная информация сайта, ссылки и т.д.

**Интересная находка,** если в атрибут **href** поместить символ **#**. То, созданная ссылка будет возвращать на эту же страницу.

**Пример: <a href="#">Главная </a>**

**<span>** - тег для работы со строчными, а не блоковыми элементами, можно вставлять в другие теги, например, в **<p>**. Им можно скажем выделить конкретную букву предложении в другой цвет позже применив стиль. **Пример:**





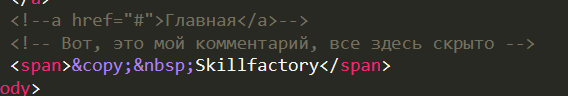
В коде можно оставлять **комментарии** или отключать теги или целые блоки.

Чтобы добавить комментарий нужно использовать такую конструкцию:

**<!-- Ваш комментарий и весь текст -->**

Если же поставить **!--** внутри какого-то тега, то, все, что ниже его и он сам будет закоменчено, до тех пор пока вы не поставите **-->**

**Примеры:**



**Или лучше так:** 

**Ctrl + /** - быстро комментировать или разкомментировать.

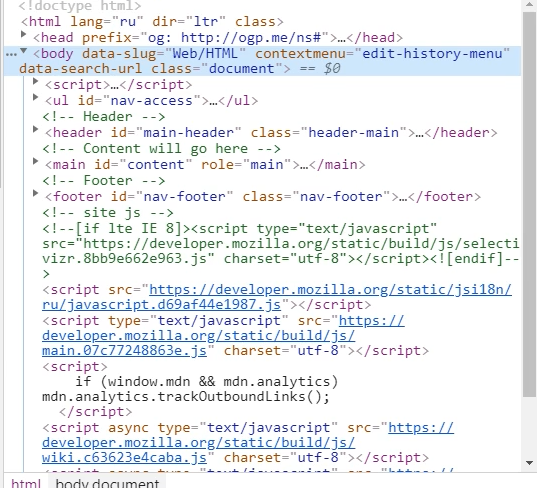
**А теперь выполнить задание выше!**

Если в sublime начать код с **<html>,** то все основная структура сформируется сама:

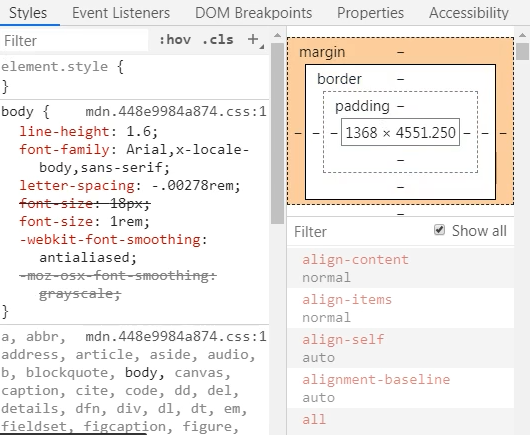


**Как манипулировать страницей с помощью инструментов разработчика.**

**F12 и откроются инструменты.** Эта часть с **html** кодом



А эта часть с **CSS**



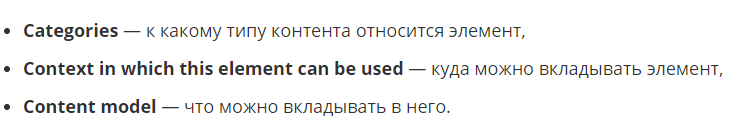
И если выбрать какой-то элемент на странице через значок со стрелочкой, а затем нажать на него два раза в коде, то его можно легко изменить. Только эти изменения не влияют на настоящий сайт. Картинки менять тоже возможно, достаточно копировать и заменять адреса. Некоторые картинки могут быть добавлены с помощью стилей.

Пару атрибут тега **<meta>:**

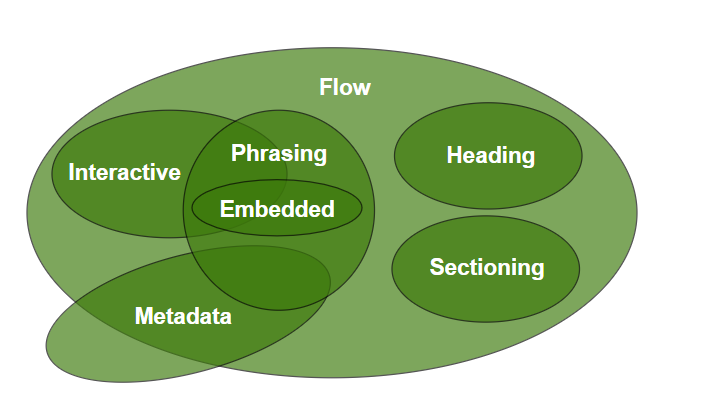
**<meta> — charset, name="viewport", name="description"**

Сайт с **html** стандартами. <https://html.spec.whatwg.org/multipage/>

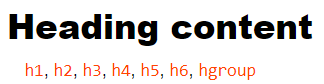
И по каждому элементу можно найти полезную информацию, к примеру:



Чуть подробнее о типах контента. Вот общая картина:



На ней видно, что куда входит. А если подробнее:

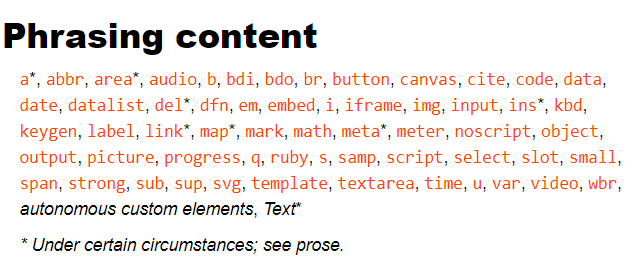


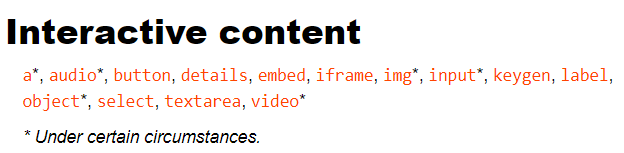
Заголовки



Пока не знаю.

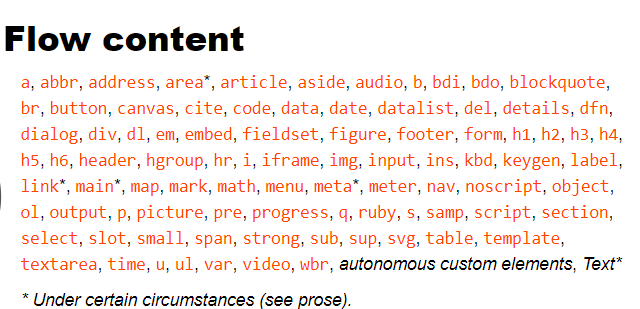








А **Flow** это все это.



Например, параграф **<p>** можно вставить где угодно. Но в него можно вставить только элементы из категории **phrasing.**

Или документация по браузеру.

Важно понимать, что не стоит в слепую копировать код без тестирования. Например, разные браузеры могут по-разному отображать **CSS** элементы. Вообще с помощью **CSS** на странице появится полноценный дизайн.

**Вебинар типичные ошибки новичков.**

Это часовой вебинар. По ошибкам.

Сохраняйте через ctrl s. В VS code есть расширение, чтобы обновление страницы происходило автоматически, называется live server. Не путать регистры букв, а лучше копируйте пути, чем писать их путь. В названиях не использовать русские буквы и пробелы, Do\_like\_that.

Есть такие штуки, как валидаторы. В них можно вставить код, чтобы обнаружить ошибки и опечатки.

**Важно:** нельзя копировать локальный путь ( E:\Files\Works\SkillFactory\A1\Сайт). Использовать только относительный, потому что локальный путь будет работать, только на вашем компьютере.

Часто делать копии, затем научиться бэкапить, а потом и до **git** дойдет.

Еще бывают ошибки **вложенности**, когда один элемент вкладывают туда, куда нельзя. Помнить, что куда не обязательно, есть удобный сайт под названием: <https://caninclude.glitch.me/>

Если в инструменте разработчика, что-то зачеркнуто, значит это не подключено или там ошибка.

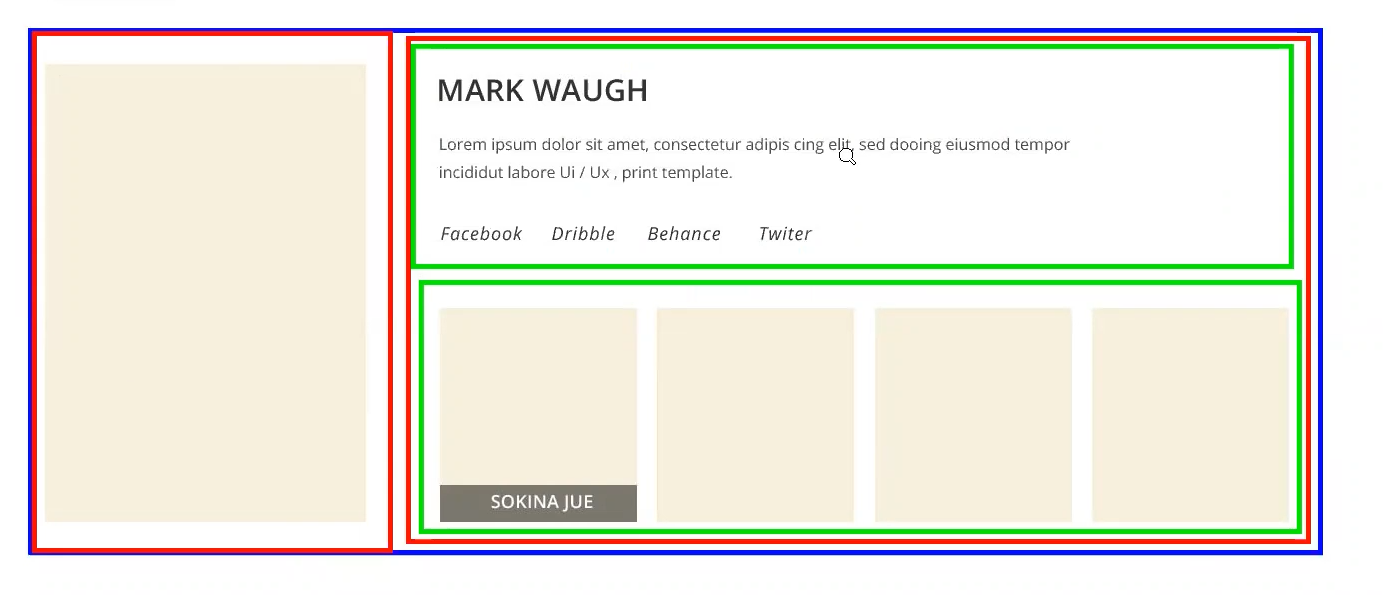
**Margin** отступ, его лучше настраивать не в ручную, а автоматическим отступом. Если точнее:

Макет-фиксированная картинка, а экрана у всех разные. Поэтому и выравнивают через авто марджин или флексбокс/гриды. Простой пример: **margin: 0 auto;**

Если верстка фиксированная, то не должно быть горизонтального скрола. Стараться использовать **абсолютное позиционирование** только по делу.

Не ограничивать высоту элементов, так где без этого не обойтись можно задать минимальную высоту. Проверять элементы на переполнение.

Перед версткой глядя на макет нужно узнать сколько прямоугольников нам потребуется, можно даже так нарисовать на макете.



Иногда отдыхать от кода. Просто чтобы запомнить, стили подключаем в head, через **link.**

16:00

**Практика 1.**

Задание 1.

В файле index.html заменить три картинки на свои и добавить им в html размер. Заменил. Добавил.

Задание 2.

Добавить в шапке еще одно меню, т.е. элемент списка. Добавил пункт галерея, который ссылается на этот объект.

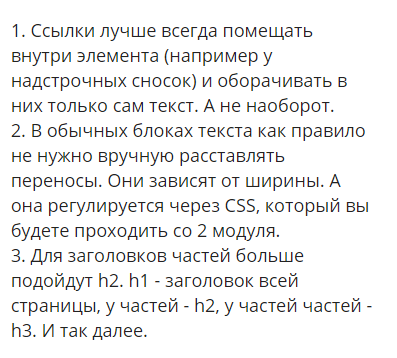
Задание 3.

В футере значки соц. Сетей сделаны через шрифты, но есть недостатки, т.к. загружаются целые шрифты. Отключить иконочный шрифт заменить их на картинки. Заменил, но я не нашел как отключить иконочный шрифт, или он прямо в этих строчках меняется?

Отправил задание. Задание проверили и один недочет. Картинки мессенджеров я выгрузил в формате jpg, а требовалось в .svg

**Практика 2.**

Это была верстка. Ошибок не было, но были такие замечания:



Иногда сайт может не обновится после изменений. В таком случае страницу нужно обновить по-другому, не просто **F5,** а **Ctrl F5** или **Ctrl R.** Это обновления без кэша.

**Вебинар основы HTML.**

Есть абсолютный путь - это на другой сайт. Относительный – это на самом сайте, пути на css и тд.

А есть локальны путь, путь на самом компе. Локальный путь использовать **нельзя.** Даже косые черты в пути на **windows,** направлены в другую сторону.

http – протокол доступа.

https – протокол с шифрованием.

**../** директория выше и таким образом подняться можно несколько раз, **пример:**

<img src="../../m1-7-1.png">

Я в практике забыл использовать в картинках векторный формат, так вот, даже если файл имеет формат **.svg** еще не факт, что он векторный. Можно открыть эту картинку через инструменты разработчика, и если там кучу всего, то это действительно **svg**. Иначе нет.

**Сайт htmlbook.ru устарел.**

Не делать внешний вид сайта через **html** для этого есть стили.

Все же делать ***валидацию*** кода.

**Cms –** это системы где с помощью графического интерфейса можно собрать сайт. Конструкторы это штуки попроще и ограничены.

**Span** и **div** по факту теги без смысла, но как правило используются для **стилизации.**

**Span** для текстового содержимого или чаще.

Конструкторы — это больше не разработка, а так, дизайн на коленке.

**Вебинар веб верстка.**

В sublime можно включить меню с отображением файловой структурой: **view>side bar**

24 min

../ выше

./ корень

Если речь о реальном сайте, то надо ведь оставлять ссылки на ресурсы, которые мы брали. Лицензии. Есть те, на которые надо оставлять ссылки. Есть те, которые платные, а есть те, которые использовать вообще нельзя.

MDN web docks норм сайт.

**Конец Модуля\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**