**3. Установка и настройка редактора кода**

Visual Studio Code <https://code.visualstudio.com/>

Node.js <https://nodejs.org/en/>

Список необходимых плагинов:

* All Autocomplete
* Auto Close Tag
* Auto Complete Tag
* Auto Rename Tag
* Beauty
* Code Runner
* Import Cost
* JavaScript (ES6) code snippets
* jshint
* Live Server
* Live Sass Compiler (by Ritwick Dey)
* Multiple clipboards for VSCode
* Reactjs code snippets
* Sass
* Easy Less
* Theme - Oceanic Next (опционально)
* vscode-icons

Сокращенные команды Emmet <https://docs.emmet.io/cheat-sheet/>

**5. Работа с современными редакторами: Avocode, Zeplin, Figma...**

Avocode <https://avocode.com/>

Для того чтобы пользоваться авокодом бесплатно , нужно открывать в хроме вкладку инкогнито . А для того чтобы не создавать каждый раз новый email нужно при регистрации писать так username+1655165@gmail.com. username - ваш login от gmail , +1655165 - любое число через + . Письмо с подтверждением придет на ваш постояный gmail аккаунт .

Проверено не один десяток раз .(Входить в аккаунт авокод только в окне инкогнито)

Регистрация с username+1655165@gmail.com работает не только в авокод , а и на других сайтах .

Zeplin <https://zeplin.io/>

Figma <https://www.figma.com/files/recent?fuid=1080840094836041855>

Конвертация макетов <https://avocode.com/convert-psd-to-sketch>

Marsy это онлайн инспектор файлов PSD для верстальщиков <https://www.markupeasy.ru/>

PSDETCH <https://psdetch.com/>

**8. Основные теги HTML на практике**

Справочник HTML

<https://webref.ru/html>

Ещё справочник HTML

<https://htmlreference.io/>

Универсальные атрибуты

<https://webref.ru/html/attr/common>

Сокращенные команды Emmet

<https://docs.emmet.io/cheat-sheet/>

У тегов есть атрибуты. Например,

<ol type="I"> </ol>

**9. Семантические теги HTML5**

|  |  |
| --- | --- |
| <header> </header> | задает «шапку» сайта или раздела, в которой обычно располагается заголовок |
| <nav> </nav> | Навигационное меню |
| <article> </article> | Запись в блоге |
| <section> </section> | Секция на странице |
| <aside> </aside> | Боковая колонка |
| <main> </main> | Основное содержимое документа. Содержимое должно быть уникальным и не включать типовые блоки вроде шапки сайта, подвала, навигации, боковой панели, формы поиска и т. п. |
| <footer> </footer> | задаёт «подвал» сайта или раздела, в нём может располагаться имя автора, дата документа, контактная и правовая информация. |
| <address> </address> | предназначен для хранения информации об авторе |
| <mark> </mark> | помечает текст как выделенный. Такой текст ничем не отличается от обычного, но его вид может быть изменен с помощью стилей |
| <wbr> | указывает браузеру место, где допускается делать перенос строки в тексте, если этого требует ширина родительского элемента |
| <time> </time> | Помечает текст внутри тега <time> как дата, время или оба значения |

**10. Основы CSS на практике**

**Подключение стилей**

1. <link rel="stylesheet" href="css/style.css">
2. <style>

css-текст

</style>

1. @import 'css/buttons.css'
2. <h1 style="color: red">First</h1> (<https://webref.ru/html/attr/style>)

**Задание стилей**

селектор {

свойство : значение;

свойство : значение;

свойство : значение;

}

Справочник CSS <https://webref.ru/css>

Ещё <https://cssreference.io/>

CSS3-селекторы <https://learn.javascript.ru/css-selectors>

CSS-цвета https://www.rapidtables.com/web/css/css-color.html

Конвертер цветов <https://colorscheme.ru/color-converter.html>

Много всего про CSS <https://sitehere.ru/tablica-stilej-css-v-udobnom-vide>

**Стили для работы с текстом**

font-family: comic sans ms, times new roman, verdana;

font-size: 16px;

font-style: normal|italic|oblique|inherit;

font-weight: bold|normal|100|200|300|400|500|600|700|800|900;

color: red;

text-decoration: line-through|blink|overline|underline|none|inherit;

text-align: left|center|right|inherit;

text-indent: 20px; (красная строка)

text-transform: lowercase|uppercase|capitalize|none|inherit;

line-height; (определяет интервал)

**11. Блочная модель CSS**



**13. Блочная модель CSS. Часть 2**

Справочник <https://htmlreference.io/>

1. **Блочные элементы**

Display: block;

1. составляют каркас страницы
2. «жадные»
3. имеют св-ва ширины, высоты, полей, границ и отступов.

Примеры: div, p,

Св-во **box-sizing: content-box | border-box**

определяет включают width и height поля и границы или нет.

Обычно используют

**\* {**

**box-sizing:border-box;**

**}**

1. **Строчные элементы**

Display: inline;

1. составляют наполнение страницы
2. «нежадные»
3. имеют св-ва отступов margin только по вертикали, не имеют ширины width.

Примеры: a, img,

Свойство display можно изменить вручную в css.

Пример: выравнивание картинки по центру

**img {**

**display: block;**

**width: 500px;**

**margin: 0 auto;**

**}**

1. **Строчно-блочные элементы**

Display: inline-block;

1. Объединяют св-ва тех и других
2. «нежадные»
3. ~~имеют св-ва отступов margin только по вертикали, не имеют ширины width.~~

**14. Позиционирование элементов в CSS. Принцип карточной колоды**

**Свойство position**

|  |  |
| --- | --- |
| position: static | Отображение как задумано браузером(?) (по умолчанию) |
| position: relative;  left: 130px;  top: 50px; | свойства left, top, right и bottom изменяют позицию элемента и сдвигают его в ту или иную сторону от первоначального расположения (относительно положения static?). Заметьте, что первоначальное место, где был элемент, резервируется. |
| position: absolute;  left: 130px;  top: 50px; | Указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и нет (выводит элемент из потока). Положение элемента задается свойствами left, top, right и bottom:  - если у родителя значение position установлено как static или родителя нет, то отсчет координат ведется *от края окна браузера*.  - если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется *от края родительского элемента*. |
| position: fixed; | По своему действию это значение близко к absolute, но в отличие от него привязывается к указанной свойствами left, top, right и bottom точке на экране и *не меняет своего положения при прокрутке* веб-страницы  (выводит элемент из потока и привязывает к определённой точке окна) |
| position: inherit; | Наследует от родительского блока |

**Свойство z-index**

|  |  |
| --- | --- |
| z-index: (>0); | Элемент с большим z-index будет на переднем плане.  Элемент с position: absolute;  будет всегда на переднем плане по сравнению с элементами с position: relative; |
| z-index: (<0); | Элемент с большим z-index будет на переднем плане.  Элемент с position: absolute;  будет всегда на заднем плане по сравнению с элементами с position: relative; |

**15. Выравнивание элементов по вертикали. Верстка таблицами и float'ами**

1. **Свойство float**

Задает обтекание элемента слева:

float: right;

или справа:

float: left;

1. **Свойство vertical-align**

vertical-align: top|bottom|middle|…;

Применимо только для элементов display: inline-block.

**16. Бонус. Единицы измерения CSS**

Единицы измерения <http://htmlbook.ru/content/edinitsy-izmereniya>

<https://www.w3.org/Style/Examples/007/units.ru.html>

|  |  |
| --- | --- |
| px | пиксели |
| em | высота шрифта текущего элемента (px=em\*font-size[px]) |
| ex | x-высота символа. Не используется |
| % | проценты от ширины/высоты **родительского** элемента |
| in |  |
| cm |  |
| mm | миллиметры |
| pt | Пункты |
| pc | Пики |

Новые единицы:

|  |  |
| --- | --- |
| rem | root em. В отличие от em, который может быть для каждого элемента свой, rem — это размер шрифта корневого элемента в документе |
| vw | 1/100 ширины окна пользователя |
| vh | 1/100 высоты окна пользователя |

**17. Специфичность CSS селекторов**

Знаете ли вы селекторы <https://learn.javascript.ru/css-selectors>

Каскадность CSS. Приоритеты стилей <https://idg.net.ua/blog/uchebnik-css/azy-css/kaskadnost>

По возрастанию приоритетности (специфичности)

|  |
| --- |
| Тег |
| Класс |
| Класс, расположенный ниже в коде css |
| Вложенный класс. Пример  .parent\_elem\_class .descendant\_elem\_class { } |
| Идентификатор |
| Идентификатор в сочетании с классом  #id.class |
| Атрибут style=”…” |
| Объявление !important  Пример:  .class {color: red !important;} |

**18. Практика. Создаем сайт на чистом HTML и CSS**

Про background <https://cssreference.io/backgrounds/>

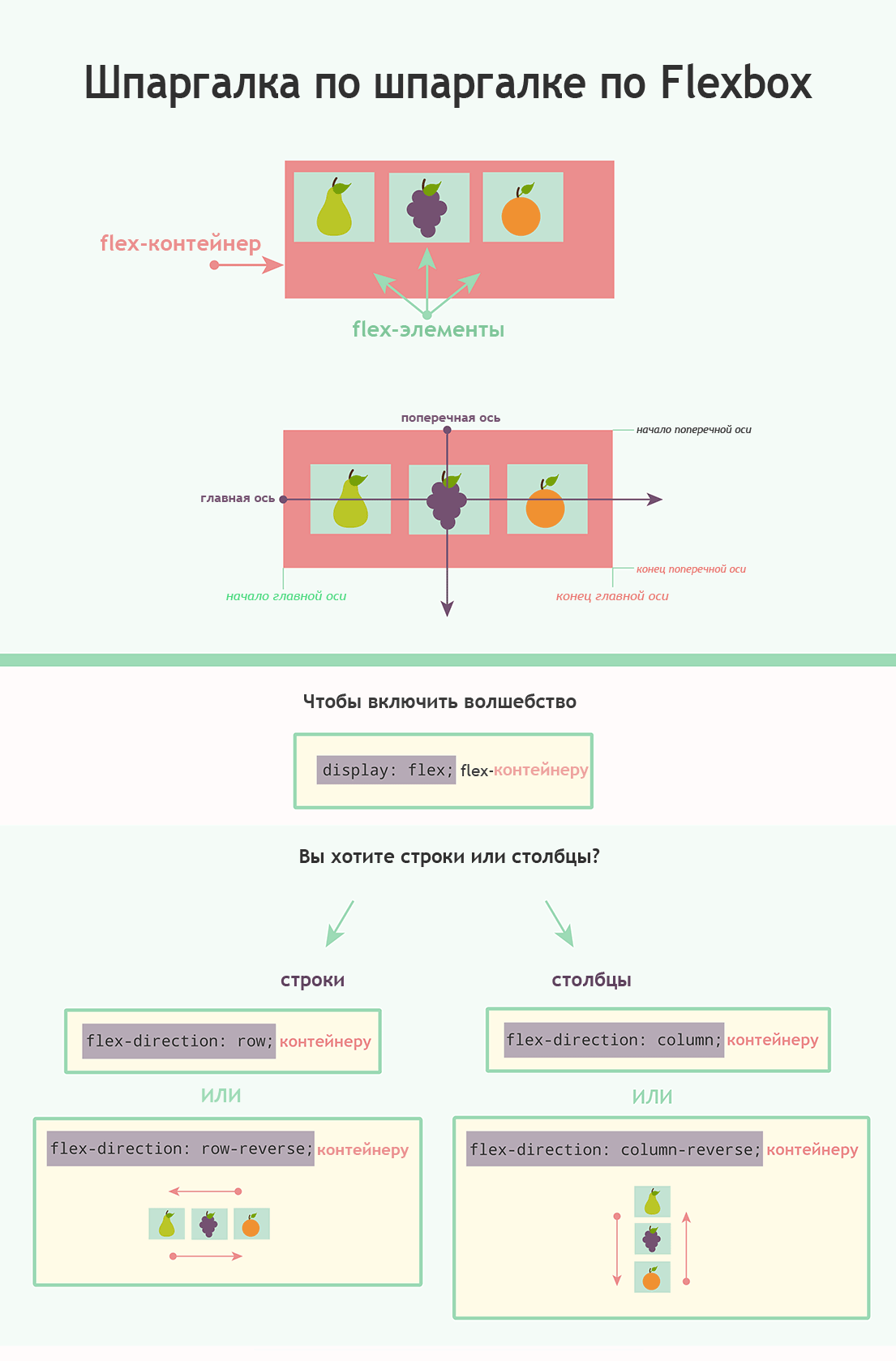
Макет в Фигма <https://www.figma.com/file/dmDxKhqtcdB2tQagsgFz8U/WordPress2>

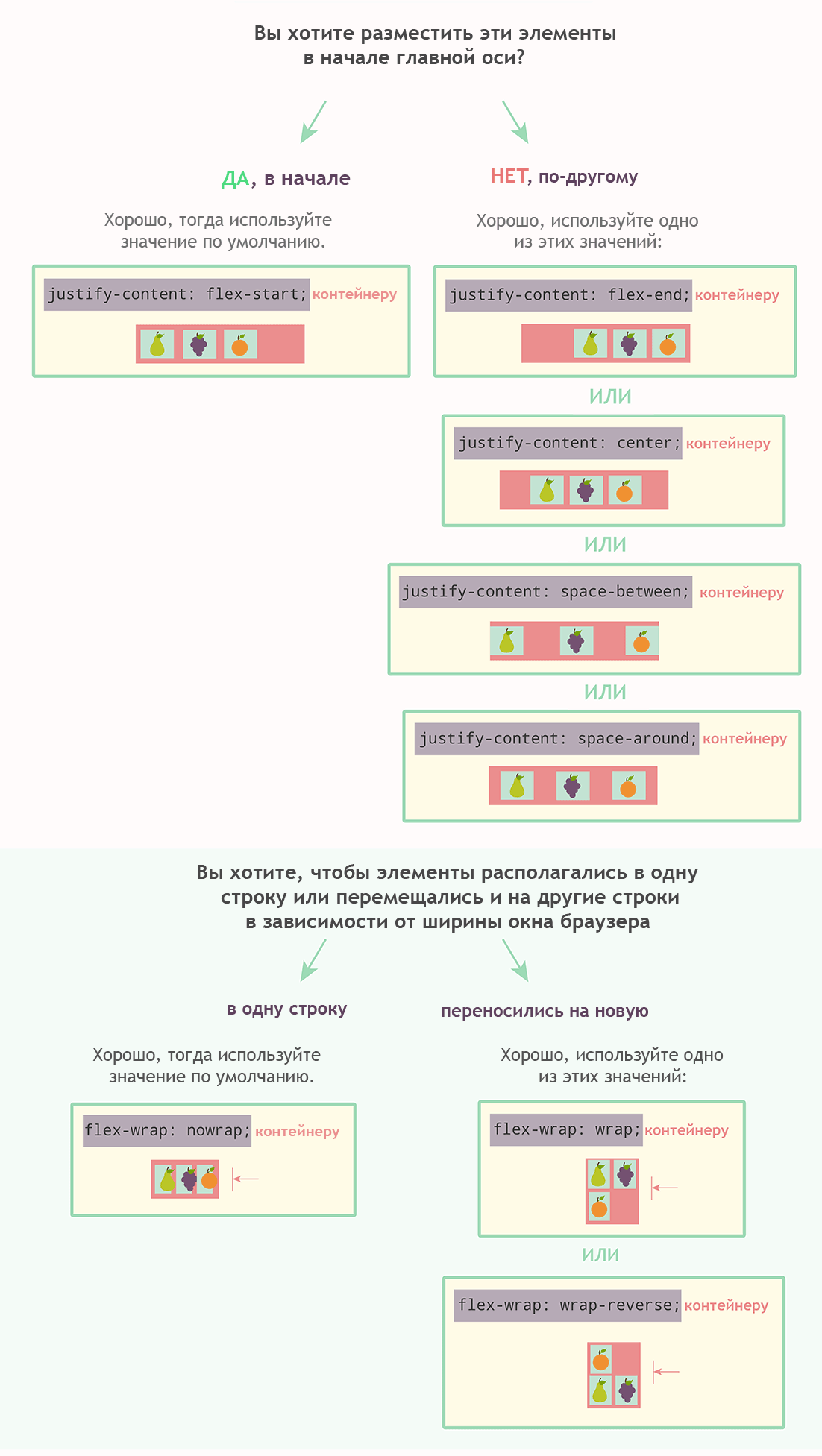
(аккаунт rv-csi@yandex.ru/Fi123454321)

**19. Технология Flexbox и применение её на макете**

Шпаргалка по Flexbox <https://habr.com/ru/post/313938/>

Игра для изучения https://flexboxfroggy.com/#ru





**20. Практика. Заканчиваем сайт на чистом HTML и CSS + домашнее задание**

background-position <https://cssreference.io/backgrounds/#background-position>

Эталонный вариант верстки (и все другие домашние задания) <https://github.com/yankovalenko94/Webdev2019>

**21. Сброс стилей reset.css/normalize.css**

Normalize.css <https://necolas.github.io/normalize.css/>

Статья про reset.css <https://habr.com/ru/post/45296/>

Статья про normalize.css <https://htmlacademy.ru/blog/boost/frontend/about-normalize-css>

|  |  |
| --- | --- |
| Локальное подключение | <link rel="stylesheet" href="css/normalize.css"> |
| Удаленное подключение | <link rel="stylesheet" href="*url*">  Адреса в облаке можно взять на ресурсе <https://cdnjs.com/libraries/normalize> |

**22. Подключение шрифтов на сайт**

Про font-family <https://developer.mozilla.org/ru/docs/Web/CSS/font-family>

Здесь собраны лучшие шрифты <https://webfonts.pro/>

Сервис для конвертации шрифтов <https://transfonter.org/>

Про форматы шрифтов <https://webformyself.com/ponyatie-formatov-shriftov-i-kak-vyzhat-iz-nix-maksimum-dlya-veb/>

Шрифты можно подключать тремя способами

|  |  |
| --- | --- |
| Подключение стандартных шрифтов, которые установлены в компьютере  <https://developer.mozilla.org/ru/docs/Web/CSS/font-family#%3Cfamily-name%3E> | font-family: <family-name>, <generic-name>;  где  <family-name> - имя семейства шрифтов (сложные имена д/б в кавычках).  Напр., Lucida Palatino Trebuchet,  <generic-name> - общие семейства шрифтов (как резервный механизм для сохранения некоторого авторского стиля в момент, когда ни один из желаемых шрифтов недоступен). Напр.,  font-family: serif;  font-family: sans-serif;  font-family: monospace;  font-family: cursive;  font-family: fantasy;  font-family: system-ui;  font-family: emoji;  font-family: math;  font-family: fangsong; |
| Удаленное подключение (напр., Google fonts  <https://fonts.google.com/>) | Подключение шрифтов осуществляется в html через тег <link rel=” ” href=” ”>  На Google fonts надо выбрать шрифт, а в нём **все необходимые варианты жирности** (Thin 100, Bold 700 и т.п.). Скопировать код html:  **<link rel="preconnect" href="https://fonts.googleapis.com">**  **<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>**  **<link href="https://fonts.googleapis.com/css2?family=Roboto:**  **wght@100;700&display=swap" rel="stylesheet">**  И css-правило будет иметь вид:  **body {**  **font-family: 'Roboto', sans-serif;**  **}** |
| Локальное подключение | Подключение шрифтов осуществляется в CSS через директиву @font-face:  **@font-face {**  **font-family: FontName;**  **src: url('path/filename.eot');**  **src: url('path/filename.eot?#iefix') format('embedded-opentype'),**  **url('path/filename.woff2') format('woff2'),**  **url('path/filename.woff') format('woff'),**  **url('path/filename.ttf') format('truetype');**  **}**  Использование в CSS:  **body {**  **font-family: 'FontName', Helvetica, Arial, sans-serif;**  **}**  Шрифты можно взять на <https://webfonts.pro/>  Подключения через **@font-face** обычно делают в отдельном файле font.css, который нужно подключить в html.  **ЗАМЕЧАНИЕ**. При локальном подключении вместо подключения каждого начертания одного шрифта по отдельности:  **@font-face {**  **font-family: 'Roboto Black';**  **src: url('../fonts/Roboto-Black.eot');**  **src: url('../fonts/Roboto-Black.eot?#iefix')**  **format('embedded-opentype'),**  **url('../fonts/Roboto-Black.woff') format('woff'),**  **url('../fonts/Roboto-Black.ttf') format('truetype');**  **font-weight: normal;**  **font-style: normal;**  **}**  **@font-face {**  **font-family: 'Roboto Light';**  **src: url('Roboto-Light.eot');**  **src: url('Roboto-Light.eot?#iefix') format('embedded-opentype'),**  **url('Roboto-Light.woff') format('woff'),**  **url('Roboto-Light.ttf') format('truetype');**  **font-weight: normal;**  **font-style: normal;**  **}**  и такого использования каждого начертания:  **.title\_big {**  **font-family: 'Roboto Black', sans-serif;**  **}**  **.title\_small {**  **font-family: 'Roboto Light', sans-serif;**  **}**  можно подключать все начертания в один шрифт так:  **@font-face {**  **font-family: 'Roboto';**  **src: url('../fonts/Roboto-Black.eot');**  **src: url('../fonts/Roboto-Black.eot?#iefix') format('embedded-opentype'),**  **url('../fonts/Roboto-Black.woff') format('woff'),**  **url('../fonts/Roboto-Black.ttf') format('truetype');**  **font-weight: 900;**  **font-style: normal;**  **}**  **@font-face {**  **font-family: 'Roboto';**  **src: url('Roboto-Light.eot');**  **src: url('Roboto-Light.eot?#iefix') format('embedded-opentype'),**  **url('Roboto-Light.woff') format('woff'),**  **url('Roboto-Light.ttf') format('truetype');**  **font-weight: 300;**  **font-style: normal;**  **}**  и использовать как один шрифт с разными весами:  **.body {**  **font-family: 'Roboto', sans-serif;**  **}**  **title\_big {**  **font-weight: 900;**  **}**  **.title\_small {**  **font-weight: 300;**  **}** |

**24. Библиотека Bootstrap 4**

<https://getbootstrap.com/>

Варианты:

1. Скачать архив и подключить нужные файлы css из состава архива.
2. Облачное подключение (cdn).

Колоночная верстка с помощью Bootstrap через использование классов:

container > row > col

Задание количества колонок и смещения от предыдущего блока:

<div class="col-md-2 offset-md-1">

…

<div>

**26. Практика. Новый проект с использованием сетки Bootstrap**

Макет в Figma

<https://www.figma.com/file/ivgb9OfADjPvRowi32CSgY/UBER_course>

**28. Препроцессоры. SASS/SCSS/LESS**

Документация по SASS

<https://sass-lang.com/>

<https://sass-scss.ru/documentation/>

SASS/SCSS компилятор Scout <https://scout-app.io/>

Компилятор Koala <http://koala-app.com/>

Настройка плагина для VSCode Live SASS Compiler (by Ritwick Dey) <https://github.com/ritwickdey/vscode-live-sass-compiler/blob/master/docs/settings.md>

Статья «LESS: программируемый язык стилей» <https://habr.com/ru/post/136525/>

1. **Plug-in для VSCode Live Sass Compiler**

В настройки добавить (<https://github.com/ritwickdey/vscode-live-sass-compiler/blob/master/docs/settings.md>) (поставить запятую после существующих настроек):

**,**

**"liveSassCompile.settings.formats":[**

// This is Default.

// {

// "format": "expanded",

// "extensionName": ".css",

// "savePath": null

// },

// You can add more

// {

// "format": "compressed",

// "extensionName": ".min.css",

// "savePath": "/dist/css"

// },

// More Complex

**{**

**"format": "compressed",**

**"extensionName": ".min.css",**

**"savePath": "~/../css/"**

**}**

**],**

**"liveSassCompile.settings.autoprefix": [**

**"> 1%",**

**"last 2 versions"**

**]**

1. **Переменные SASS**

Объявление

$text\_color: blue

Использование

.block\_color

color: $text\_color

1. **Миксины**

Объявление

@mixin box

width: 250px

height: 200px

Использование

button

@include box

Или 2-й вариант:

Объявление

=box

width: 250px

height: 200px

Использование

button

+box

*SASS (или SCCS) предпочтительнее, чем LESS.*

1. **Организация файлов**
   1. Разбиение стилей на отдельные файлы по «темам»

\_variables.sass

\_mixins.sass

\_general.sass

\_media.sass

(знак \_ нужен, чтобы компилятор пропустил эти файлы, а не создавал соответствующие им css-файлы)

* 1. Импортирование отдельных файлов в Style.sass

\*

font-family: 'Roboto', sans-serif

@import "variables"

@import "mixins"

@import "general"

@import "media"

(файлы указываются без расширений)

**30. Вендорные префиксы в CSS**

Зачем нужны вендорные префиксы <https://vaden-pro.ru/blog/css/vendornye-prefiksy>

**31. Практика. Продолжаем работу и используем препроцессор SASS**

Если уже есть CSS, то для перехода на SASS используем конвертер .css в .sass <https://css2sass.herokuapp.com/> или аналог.

В VSCode используем плагин Live Sass Compiler (by Ritwick Dey) – см. выше.

CSS3-трансформации <https://html5book.ru/css3-transform/>

**32. Псевдоклассы и псевдоэлементы в CSS**

**33. Практика. Используем псевдоэлементы и псевдоклассы в проекте**

Документация по псевдоклассам и псевдоэлементам

<https://webref.ru/css#pseudo-class>

Классификация и описание <https://html5book.ru/psevdoklassy/>

1. **Псевдоклассы**

|  |  |
| --- | --- |
| :nth-child(…) | используется для добавления стиля к элементам на основе нумерации в дереве элементов |
| :link | непосещенная ссылка |
| :visited | посещенная ссылка |
| :focus | ссылки, а также элементы форм, которые активированы посредством курсора мыши или на которые перешли с помощью клавиатуры (кнопка TAB) |
| :hover | ссылки, а также другие элементы, стили применяются при наведении пользователем на элемент |
| :active | выбирает элемент, активированный пользователем с помощью клика мышки. Обычно применяется для ссылок, но может отбирать и другие элементы на странице |

1. **Псевдоэлементы**

|  |  |
| --- | --- |
| :after | (обязательно наличие св-ва content: ‘ ‘) |
| :before | (обязательно наличие св-ва content: ‘ ‘) |
| :first-letter |  |

Для установки эффекта перехода между двумя состояниями элемента используется свойство **transition** <https://webref.ru/css/transition>.

**1-й способ выравнивания элемента по центру**

element {

position: absolute

top: 50% /\* смещаем наполовину родительского элемента по вертикали\*/

left: 50% /\* смещаем наполовину родительского элемента по горизонтали\*/

transform: translate(-50%, -50%) /\*и возвращаем на половину ширины

} элемента\*/

**2-й способ выравнивания элемента по центру**

Для выравнивания дочернего элемента в родительском (например, иконки в круглом родительском блоке:



parent-element {

display: inline-flex

justify-content: center

align-items: center

}

**Повторяющиеся картинки (иконки)**

Размещение на странице повторяющихся картинок (например, маркеров ненумерованного списка или кнопок «+» на к-л карточках и т.д.) с помощью <img src=”…”> неоптимально, т.к. приводит к многократному обращению к серверу и загрузке картинок с него.

Правильно делать это с помощью псевдоэлементов :before или :after.

.warning

&\_list

li:before

background: url("../icons/check\_red.png") center center/cover no-repeat

**34. Как работать с иконками. Иконочные шрифты**

1. **Использовать иконки с к.-н. ресурса.**

Например, <https://seeklogo.com/>.

1. **Иконочные шрифты из облака**

Font Awesome <https://fontawesome.com>

Подключается через таблицу стилей с помощью тега <link>:

<link rel="stylesheet" href="https://use. fontawesome.com/releases/.../all.css ...>

\*На момент обучения на этом сайте предлагается подключение через скрипт:

<script src=<https://kit.fontawesome.com/38684c248f.js> crossorigin="anonymous">

</script>

Но можно подключить и через тег <link>. Для этого:

1. на сайте <https://cdnjs.com/> через поиск найти font-awesome:

<https://cdnjs.com/libraries/font-awesome>

1. выбрать нужную ссылку на таблицу стилей css и скопировать её (в строке с url нажать на иконку </> чтобы скопировать весь тег link);
2. вставить тег link в html перед style.min.css:

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css" integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />

1. На сайте fontawesome.com через поиск найти нужную иконку. Открыть ее и скопировать html-код с тегом <i>.
2. Вставить код в html.
3. В таблице стилей задать дополнительные css стили, например, цвет иконки (цвет шрифта через свойство color).

Недостаток способа: подключенный css содержит массу ненужных для конкретной задачи иконок. Проблема решается двумя способами:

- «почистить» этот css-файл;

- собрать иконочный шрифт вручную из нужных иконок.

1. **Сборка своего иконочного шрифта (только с нужными иконками)**
2. Скачать иконки. Например, с fontawesome.com или
3. с <https://icomoon.io/app/#/select>
   1. Кнопка «Import icons» для импорта имеющихся иконок или кнопка «Icon library» для выбора готовых шрифтов. Таким образом формируется библиотека.
   2. Для создания шрифта надо выделить нужные иконки на экране.
   3. Внизу справа нажать кнопку «Generate font». Откроется страница с выбранными иконками. Внизу справа кнопка «Download». Скачивается архив, в котором нужны только:

* файл style.css – переименовать в font.css, скопировать в папку css проекта и подключить в проект (index.html):

<link rel="stylesheet" href="css/font.css">

* папка fonts – скопировать в проект.
  1. В новом файле css/font.css откорректировать пути к папке fonts: ../fonts/
  2. Для использования шрифта возвращаемся на страницу

<https://icomoon.io/app/#/select/font>, на которой была кнопка «Download» (см. пункт c). Наводя мышь на нужную иконку, нажимаем <>Get code. Копируем тег span в html-код и стили в css.

* 1. В стилях добавить свои стили, например, цвет

color: white;

**35. Адаптация проектов под различные устройства**

Статья про медиа-запросы <https://html5book.ru/css3-mediazaprosy/>

Медиа-запросы используются для изменения css-свойств в зависимости от проверяемых характеристик устройства. Содержит критерий для вступления в силу такого изменения. Медиа-запрос является логическим выражением, которое возвращает истину или ложь.

Четыре вида верстки.

1. **Фиксированная**

HTML

<div class="fixed"></div>

<div class="fixed"></div>

SASS

.fixed

display: inline-block

vertical-align: middle

width: 600px

height: 300px

background-color: blue

margin: 50px

1. **Резиновая**

HTML

<div class="rubber"></div>

<div class="rubber"></div>

SASS

.rubber

display: inline-block

vertical-align: middle

width: 45%

height: 300px

background-color: yellow

margin: 20px

1. **Адаптивная**

HTML

<div class="adaptive"></div>

<div class="adaptive"></div>

SASS

.adaptive

display: inline-block

vertical-align: middle

width: 600px

height: 300px

background-color: red

margin: 20px

@media (max-width: 992px)

.adaptive

width: 300px

1. **Отзывчивая**

HTML

<div class="responsive"></div>

<div class="responsive"></div>

SASS

.responsive

display: inline-block

vertical-align: middle

width: 45%

height: 300px

background-color: pink

margin: 20px

@media (max-width: 992px)

.responsive

width: 95%

**36. Pixel Perfect верстка**

Плагин PerfectPixel by WellDoneCode для проверки верстки <https://chrome.google.com/webstore/detail/perfectpixel-by-welldonec/dkaagdgjmgdmbnecmcefdhjekcoceebi>

1. Установить плагин
2. Сделать и сохранить на диске скриншот ***макета*** с помощью инструмента Slice (в Avocode или Figma (на панели Frame)).
3. Включить плагин
4. В окне плагина загружаем сохраненный скрин макета.
5. Регулируя прозрачность скрина, сопоставляем его с версткой.

**37. Адаптация при помощи Bootstrap**

Документация <https://getbootstrap.com/docs/4.3/layout/grid/>

1. **Классы Grid**

По умолчанию шесть брейкпойнтов размера экрана:

|  | **xs <576px** | **sm ≥576px** | **md ≥768px** | **lg ≥992px** | **xl ≥1200px** | **xxl ≥1400px** |
| --- | --- | --- | --- | --- | --- | --- |
| **max-width**  контейнера | None (auto) | 540px | 720px | 960px | 1140px | 1320px |
| Префикс класса | .col- | .col-sm- | .col-md- | .col-lg- | .col-xl- | .col-xxl- |

После префикса указываем сколько колонок должен занимать блок, например,

<div class="box col-12 col-md-6 col-xl-4">.

В этом примере блок будет занимать:

12 колонок при ширине окна <768px,

6 колонок - при ширине окна от ≥768px до 1200px,

4 колонки - при ширине окна от ≥1200px.

То есть адаптация будет происходить в брейкпойнтах, соответствующих каждому использованному классу и до следующего указанного брейкпойнта.

Класс без указания ширины => блоки с неуказанной шириной будут одинаковой ширины.

1. **Использование Columns для выравнивания**

<https://getbootstrap.com/docs/5.1/layout/columns/>

Вертикальное выравнивание

<div class="row align-items-start">

<div class="row align-items-center">

<div class="row align-items-end">

Горизонтальное

<div class="row justify-content-start">

<div class="row justify-content-center">

<div class="row justify-content-end">

<div class="row justify-content-around">

<div class="row justify-content-between">

<div class="row justify-content-evenly">

1. **Использование Columns для переупорядочивания**

Классы .order-\*

Например,

<div class="col order-5">

**38. Практика. Адаптация проекта. Часть 1**

Нежелательно уменьшать шрифт до значений менее 10px.

В качестве максимальных значений ширины в медиа-запросах нужно использовать значения Bootstrap-брейкпойнтов минус 1 (так как колонки boostrap изменяются как раз с величины брейкпойнта).

Один из подходов к адаптации – исчезновение одних элементов и появление других. Это достигается применением к элементам классов, наподобие:

class=”xs-hidden” и

class=”xs-visible”

(xs – обозначение бутстраповского брейкпойнта, то есть в этом примере для экрана с шириной менее 576px),

которые в соответствующем медиа-запросе (в данном примере к экрану xs) будут использованы в качестве селекторов свойств видимости:

. xs-hidden

display: none

. xs-visible

display: block