Министерство науки и высшего образования Российской Федерации

федеральное государственное бюджетное образовательное учреждение высшего образования

“Российский экономический университет имени Г.В. Плеханова”

**МОСКОВСКИЙ ПРИБОРОСТРОИТЕЛЬНЫЙ ТЕХНИКУМ**

Специальность: 09.02.07 Информационные системы и программирование

Квалификация: программист

ПРАКТИЧЕСКИЕ РАБОТЫ

ПО «Разработка программных модулей»

Листов:

|  |  |
| --- | --- |
| Выполнил студент группы  П50-4-21  /А.В. Плахова  «»2022 года | Проверил преподаватель  /Д. В. Серяк  «» 2023 года |

Москва 2023

*Оглавление*

[ПРАКТИЧЕСКАЯ РАБОТА №1 3](#_Toc132374816)

[Тема: Повторение дизайна из PDF 3](#_Toc132374817)

[ПРАКТИЧЕСКАЯ РАБОТА №2 10](#_Toc132374818)

[Тема: Работа с <img> и <map>. 10](#_Toc132374819)

[ПРАКТИЧЕСКАЯ РАБОТА №3 12](#_Toc132374820)

[Тема: Создание таблиц. 12](#_Toc132374821)

[ПРАКТИЧЕСКАЯ РАБОТА №4 15](#_Toc132374822)

[Тема: Создание многоуровневых списков. 15](#_Toc132374823)

[ПРАКТИЧЕСКАЯ РАБОТА №5 20](#_Toc132374824)

[Тема: Работа с <div>. 20](#_Toc132374825)

[ПРАКТИЧЕСКАЯ РАБОТА №6 23](#_Toc132374826)

[Тема: Создание класса css. 23](#_Toc132374827)

[ПРАКТИЧЕСКАЯ РАБОТА №7 25](#_Toc132374828)

[Тема: FlexBox. 25](#_Toc132374829)

[ПРАКТИЧЕСКАЯ РАБОТА №8 28](#_Toc132374830)

[Тема: Grid. 28](#_Toc132374831)

[ПРАКТИЧЕСКАЯ РАБОТА №9 30](#_Toc132374832)

[Тема: Медиа запросы. 30](#_Toc132374833)

[ПРАКТИЧЕСКАЯ РАБОТА №10 33](#_Toc132374834)

[Тема: Итоговый сайт. 33](#_Toc132374835)

# ПРАКТИЧЕСКАЯ РАБОТА №1

Тема: Повторение дизайна из PDF

Цель практической работы:научиться переносить информацию из PDF файлов, сохраняя положение объектов, шрифт, размер шрифта, изменять цвет с чёрного н другой, понять, как подчёркивать, делать жирный и курсивный стиль и многое другое.

Ход работы:

Нам дан PDF файл в котором содержится определённая информация, наша задача повторить дизайн PDF файла в программе html.

Сначала мы создаём файл и пишем восклицательный знак и выбираем где он стоит один, тогда у нас создаётся автоматические параметры.

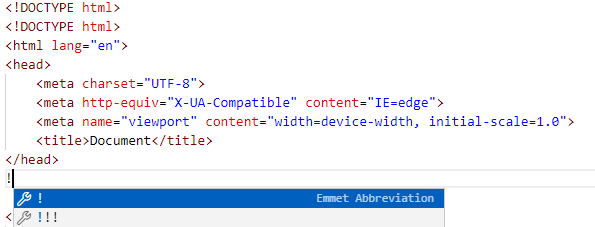


Рисунок 1 создание

Сначала мы должны создать отступ слева от страницы, для этого нам понадобится команда margin-left и именно эту команду мы вписываем в body и указываем определённый размер отступа.

Далее мы используем pre, это для того, чтобы каждая строчка начиналась с новой строки, а не подряд. Так же в этом теге мы пишем style и в скобках мы можем редактировать наш текст. Каждая функция делает свою роль, сейчас будем перечислять:

Font-size: это команда которая позволяет нам настроить размер текста.

Font-famili: позволяет нам менять шрифт.

Line-height: эта команда позволяет устанавливать интервал между строками.

Margin-bottom:эта команда позволяет устанавливать расстояние снизу текста.

Margin-top: эта команда позволяет устанавливать расстояние сверху текста.

Margin-left:эта команда позволяет устанавливать отступ слева от текста.

Margin-right: эта команда позволяет устанавливать отступ справа от текста.

Далее мы видим тэг a, этот тэг используется для создания ссылок, hrefустанавливает URL страницы или документа и в скобках мы указываем id заголовка (примечание: заголовки не могут повторяться).

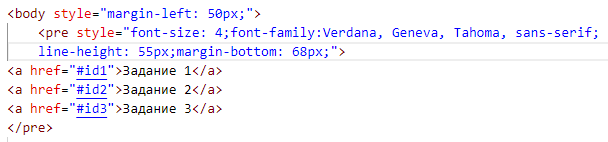


Рисунок 2 первая часть

Спускаемся ниже по коду и видим, тэг заголовка 1, и что id присвоено 1, и то есть при нажатия заголовок, то нас будет перебрасывать там, где он находится.

Далее мы используем тэг span, его в основном используют как контейнер для хранения чего-то, например, его использует для стилизации текста. И далее мы видим неизвестный атрибут style и в его скобках находится color, это позволяет нам изменить цвет самого текста на какой-нибудь другой. Так же есть и другие варианты изменения текста.

Cursive: делает текст наклонным.

Strong: делает текст жирным.

I: делает текст курсивным.

Sup:переводит текст в надстрочный вид.

Sub: переводит текст в подстрочный вид.

S: делает текст зачёркнутым.

Для того что бы закрыть тэг, нам надо поставить слэш в начале скобки. Например:<span> какой-то текст</span>.

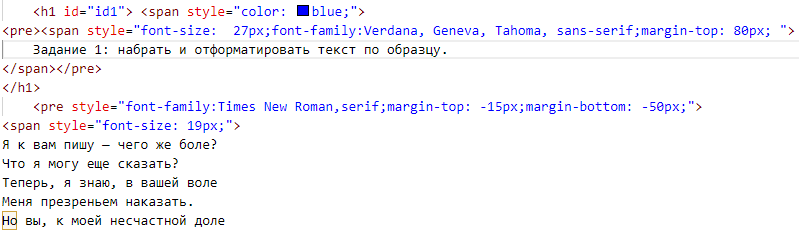


Рисунок 3 задание 1

Теперь пойдём дальше по коду. Разбираем код задания 2. Здесь нужно повторить начало задания один, но только мы меняем заголовки и idс 1 на 2. Далее мы используем снова то, что использовали для первого задания. Но тут есть и новые команды.

Text-align: это атрибут которые отвечает за расположение текста в пространстве. End– располагает в конце, center – в центре, left - выравнивание по левому краю, right - выравнивание по правому краю, justify - выравнивание по ширине.



Третье задание никак не отличается от первого и второго, поэтому мы разберём только, то что написано там нового.

Во-первых, это атрибут sup: переводит текст в надстрочный вид, аатрибут sub: переводит текст в подстрочный вид. И всё, более здесь не используется ничего нового.

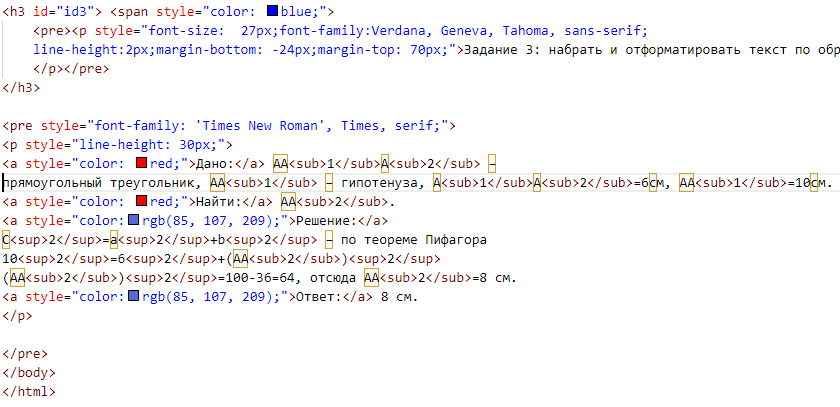


Рисунок 4 задание 3

И вот так выглядит преобразованный PDF файл, но только в виде страницы.

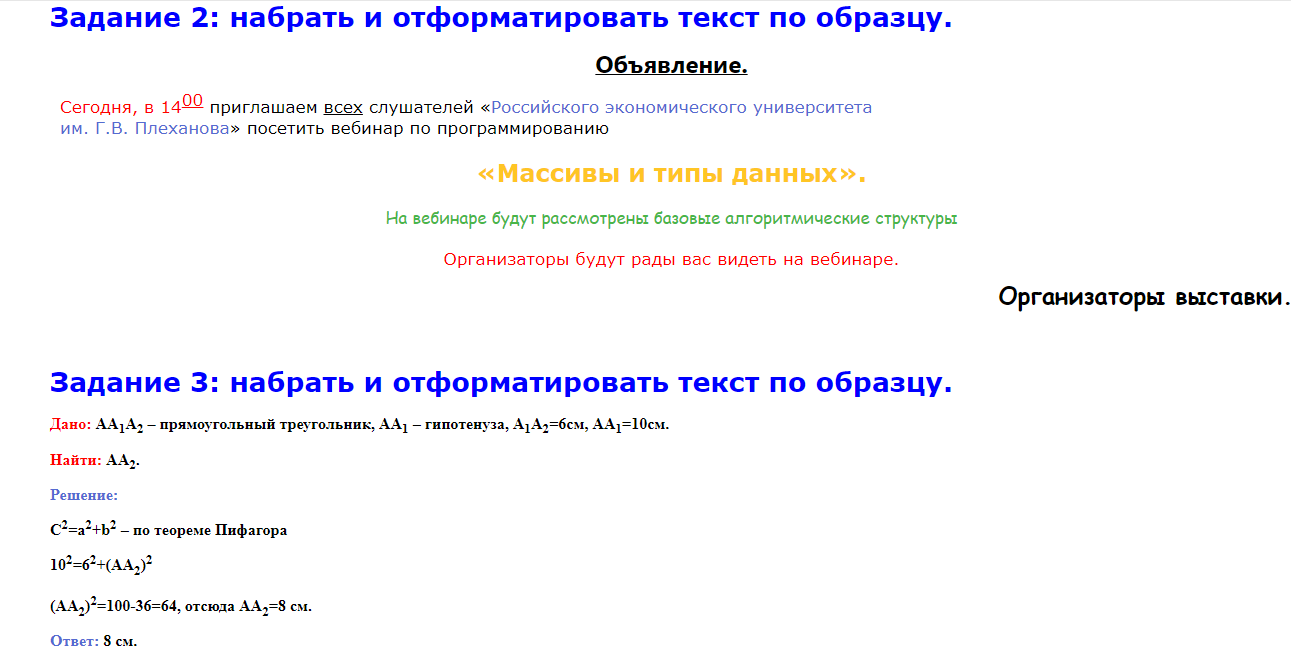


Рисунок 5 итог

Но это не все команды, давайте продолжим разработку сайта, но теперь по своему хотению. Например, я хочу создать отдельное по японской теме. Для этого мне понадобится вот такой тэг <hrcolor="red" size="10">, он позволяет создать полоску, которая будет «разделять»нашу страничку, так же здесь можно указать цвет и размер этой полоски, и ещё с помощью widhtмы можем сами определить длину этой линии.

Так же здесь используются тэги, которые, не были использованы в выше напечатанной. Также здесь мы поставили фон картинкой и сделали подложку текста.

Давайте разберём как сделать фон. Сначала, нам надо сохранить картинку и потом написать следующее: background-image:url(sakur.png). В скобках мы пишем название сохранённой картинки.

Также здесь мы использовали mark, этот тэг используется для того что бы выделить текст, если он будет на тёмном фоне.

<blockquote>эта команда выносит текст в виде цитаты, обращая на неё внимание.

Для того, чтобы вставить изображение нам надо использовать тэг imgscr=”…” в скобках надо указать ссылку на изображение, далее мы можем настроить размер изображения для этого надо написать hightили whight.

Они отвечают за высоту и ширину изображения, но если написать какой-либо один параметр, то тогда второй параметр сам подогнётся под написанный параметр.

Но, можно ещё сделать переход по ссылке по изображения. Например, мы хотим, чтобы, когда мы нажимали на картинку нас перебрасывало на какой-то сайт. Мы сделаем, чтобы переходила на сайт со стихотворения про Японию.

Для этого нам надо использовать href, его используют для установления ссылки на определённый сайт.

Target – является атрибутом, который определяет, где открыть ссылку.

И уже в кавычках мы можем написать следующие значения:

\_blank - загружает страницу в новое окно браузера.

\_self - Загружает страницу в текущее окно.

\_parent - Загружает страницу во фрейм-родитель, если фреймов нет, то это значение работает как \_self.

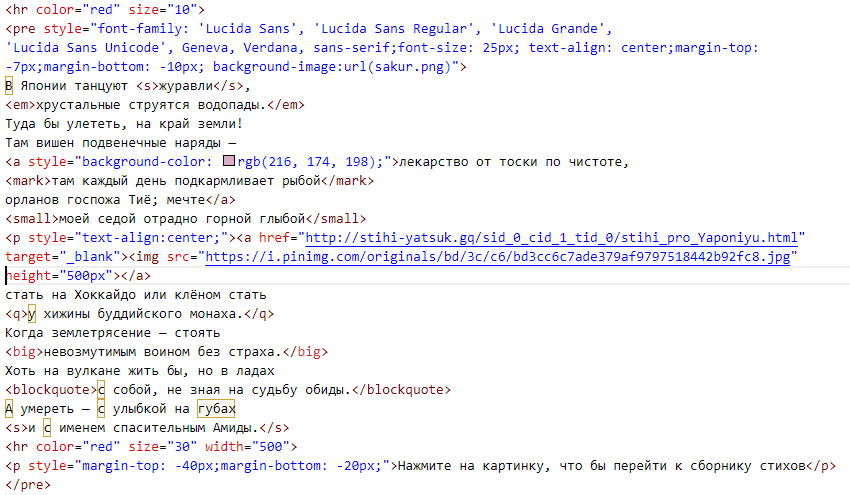


Рисунок 6 Япония

Ну и следующий стих, мы можем повторяем рисование линии, но теперь мы закрашиваем голубым и ставим свой размер линии. И сделаем фоном этого текста, градиентом.

Для это нам понадобится атрибут style и внутри его кавычек написать следующие: background: выбираем вид градиента. С градиентом можно экспериментировать и выбирать любой. Снизу представлен выбор, типа градиента.

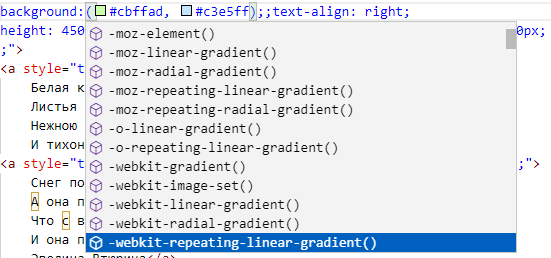


Рисунок 7 виды градиента

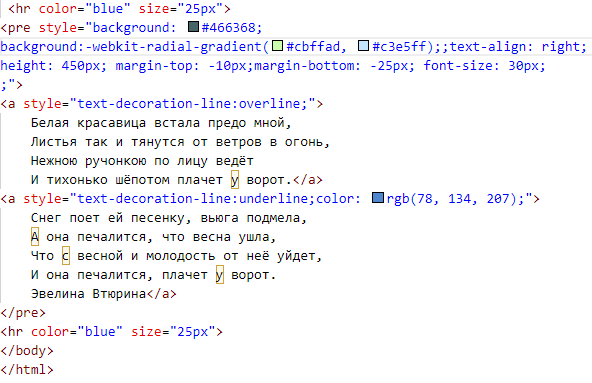


Рисунок 8 Россия

Вывод:научиласьпереносить информацию из PDF файлов, сохранять положение объектов, шрифт, размер шрифта, изменять цвет с чёрного н другой, поняла, как подчёркивать, делать жирный и курсивный стиль, как вставлять изображения и многое другое.

# ПРАКТИЧЕСКАЯ РАБОТА №2

Тема: Работа с <img> и <map>.

Цель: научиться выделять нестандартные фигуры, и чтобы когда наводилось на них курсором, то должно выписываться названия этого места.

Ход работы:

Для этого нам нужно выбрать изображение с этой нестандартной фигуры : scr=”картинка”. И далее мы пишем имя этой картинки и так же как делали с заголовками, мы ставим решётку. И далее создаём тэг map и пишем name и приравниваем это к username.

И далее мы пишем новый тэг area, который позволит задать форму области, ее размеры, устанавливает адрес документа, на который следует сделать ссылку, а также имя окна или фрейма, куда браузер будет загружать документ.

Далее пишем shape и указываем poly, который позволит нам создать нестандартную фигуру. И далее, самое долгое, мы пишем cords и указываем координаты.



Рисунок 9 начало

Для того чтобы найти координаты мы можем использовать обычный Paint или же какое-нибудь спецелезированое приложение.



Рисунок 10 координаты в Paint

И после того как мы указали все координаты, мы используем title он позволит когда наведешь на какую-то область появляется название этой области.

И функция href позволяет перейти на какой-то сайт по щелчку на эту область.



Рисунок 11 конец



Рисунок 12 код

Вывод: научилась выделять нестандартные фигуры, и чтобы когда наводилось на них курсор, то выписывается название этого места.

# ПРАКТИЧЕСКАЯ РАБОТА №3

Тема: Создание таблиц.

Цель: научится создавать таблицы в html, понять, как можно объединять столбцы и строки, как делать границу таблицы.

Ход работы:

Для того что бы создать таблицу, мы должны использовать определённый тэг table и внутри него мы будем использовать следующие тэги:

Элемент <tr> определяет строку таблицы.

Элемент <th> определяет верхний колонтитул/заголовок таблицы, а элемент <td> определяет ячейку таблицы.

Примечание: Тег <th>, отличается от <td> визуальным представлением данных внутри таблицы: по умолчанию, текст, размещенный в теге <th>, отображается жирным начертанием с выравниванием текста по центру, а текст тега <td> ‒ обычным начертанием с выравниванием по левому краю.

Так же в коде мы используем тэг tbody, он используется для группирования информации в таблице.

Далее мы заполняем внутренности таблицы и смотрим результат. По ТЗ последние две колонки должны быть объедены. Для этого мы используем colspan и пишем сколько нужно объединить колонок.

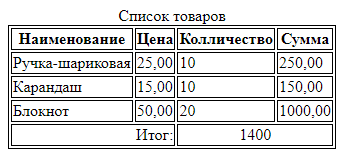


Рисунок 13 итог



Рисунок 14 код

Но теперь давайте сделаем свою таблицу. Например, с какой-нибудь мангой. В таблице есть не так много тэгов и атрибутов, поэтому рассмотрим некоторые и попробуем их применить.

Например мы хотим изменить стиль границы с двойных полосок на точки, к примеру. Для этого нам понадобится использовать атрибут border: и после двоеточий выбрать один из видов границ. И там уже из предложенных можно выбрать стиль.

Далее мы использовали цвет границ, для этого мы должны использовать border-color:” Здесь пишем какой-нибудь цвет”.

"background-color: aqua;" – используется что бы задать цветной задний фон.

"height: 10cm;" – используется что бы задать высоту ячейки.

Width – используется что бы растя

colspan="4" – используется для объединения несколько строк в единое целое.

Rowspan =”2” – используется для объединения нескольких строк в единое целое.



Вывод: научилась создавать таблицы в html, поняла, как можно объединять столбцы и строки, как делать границу таблицы.

# ПРАКТИЧЕСКАЯ РАБОТА №4

Тема: Создание многоуровневых списков.

Цель: научиться работать со списками. Понять, как делать разные нумерации списков. Изучить какие именно надо использовать тэги, узнать маркировки и нумерации списков.

Сначала давайте разберём какие именно нужны тэги для создания нумерации, вот некоторые из них:

Элемент <li> используется для создания элементов списка

Элемент <ol> используется для создания элементов номерованного упорядоченного списка

Элемент <dl> используется для создания списка терминов

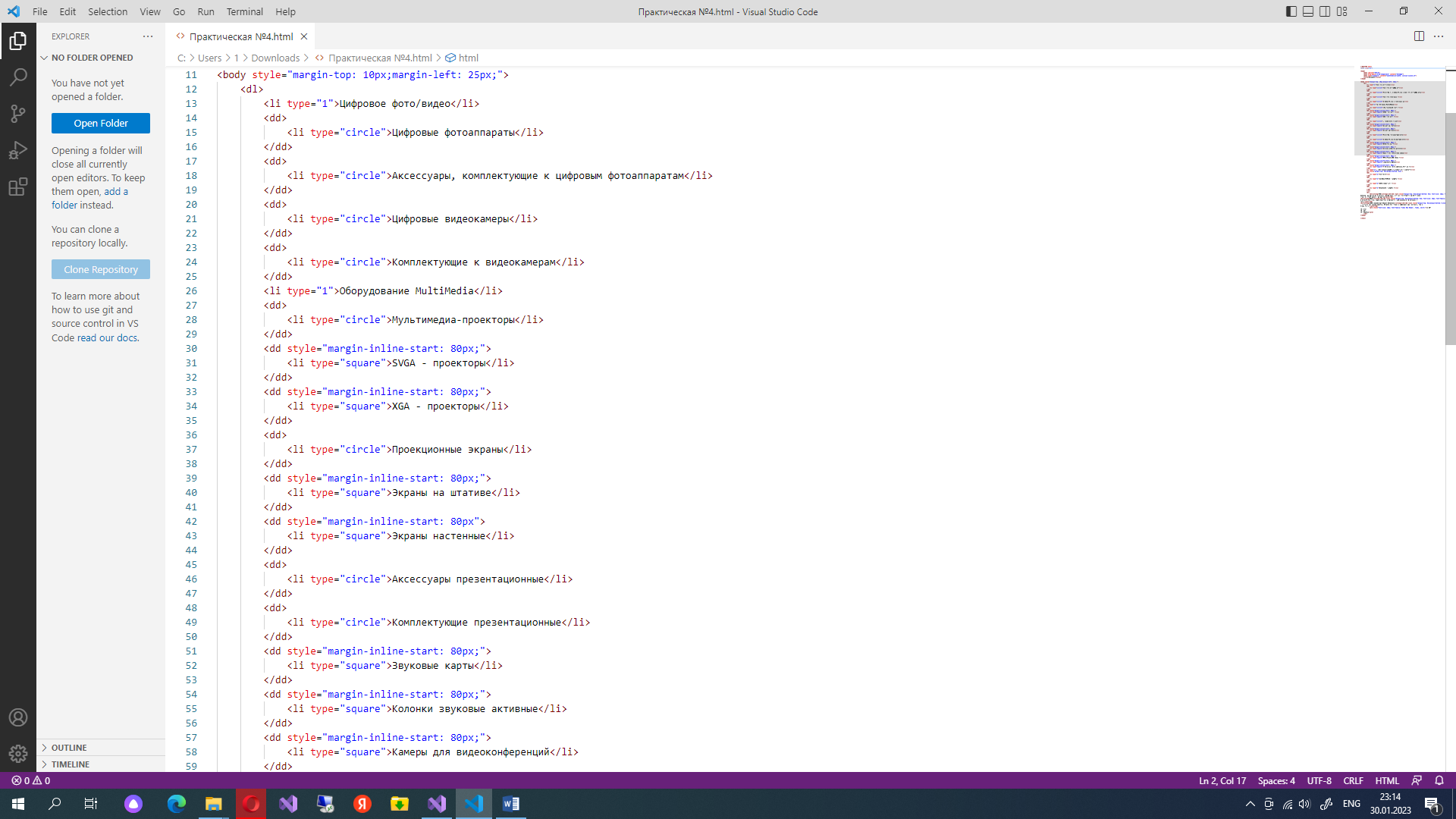


Рисунок 15 пример

Для создания маркированных списков используется тег UL, а для создания нумерованных – OL. Эти теги являются парными и блочными, точно так же, как и элемент LI.

Так же если в тэг li поставить type=”…”, можно выбрать какого именно вида будет идти нумерация, то есть цифры, буквы, символы и так далее.

Тег <dd> входит в тройку элементов <dl>, <dt>, <dd>, предназначенных для создания списка определений. Каждый такой список начинается с контейнера <dl>, куда входит тег <dt> создающий термин и тег <dd> задающий определение этого термина. Закрывающий тег </dd> не обязателен, поскольку следующий тег сообщает о завершении предыдущего элемента. Тем не менее, хорошим стилем является закрывать все теги.

Также для того что бы был отступ то мы используем следующую:

<dd style="margin-inline-start: 80px;">

Далее мы используем одно и тоже просто меняем тип списка.

Для этого мы пишем type=”….” И в скобках пишем какого вида хотим. Например:

* <li type="disc">
* <li type="circle">
* <li type="square">
* <ol> <li type="A"> </ol>
* <ol> <li type="a"> </ol>
* <ol> <li type="I"> </ol>
* <ol> <li type="i"> </ol>
* <ol> <li type="1"> </ol>

Так же можно использовать list-style-type, например:

* style="list-style-type: trad-chinese-informal;"
* style="list-style-type: trad-chinese-informal;"
* style="list-style-type: trad-chinese-informal;"
* style="list-style-type: lower-greek;"
* style="list-style-type: disc;"
* style="list-style-type: georgian,"

И многие другие, я привела некоторые из них.

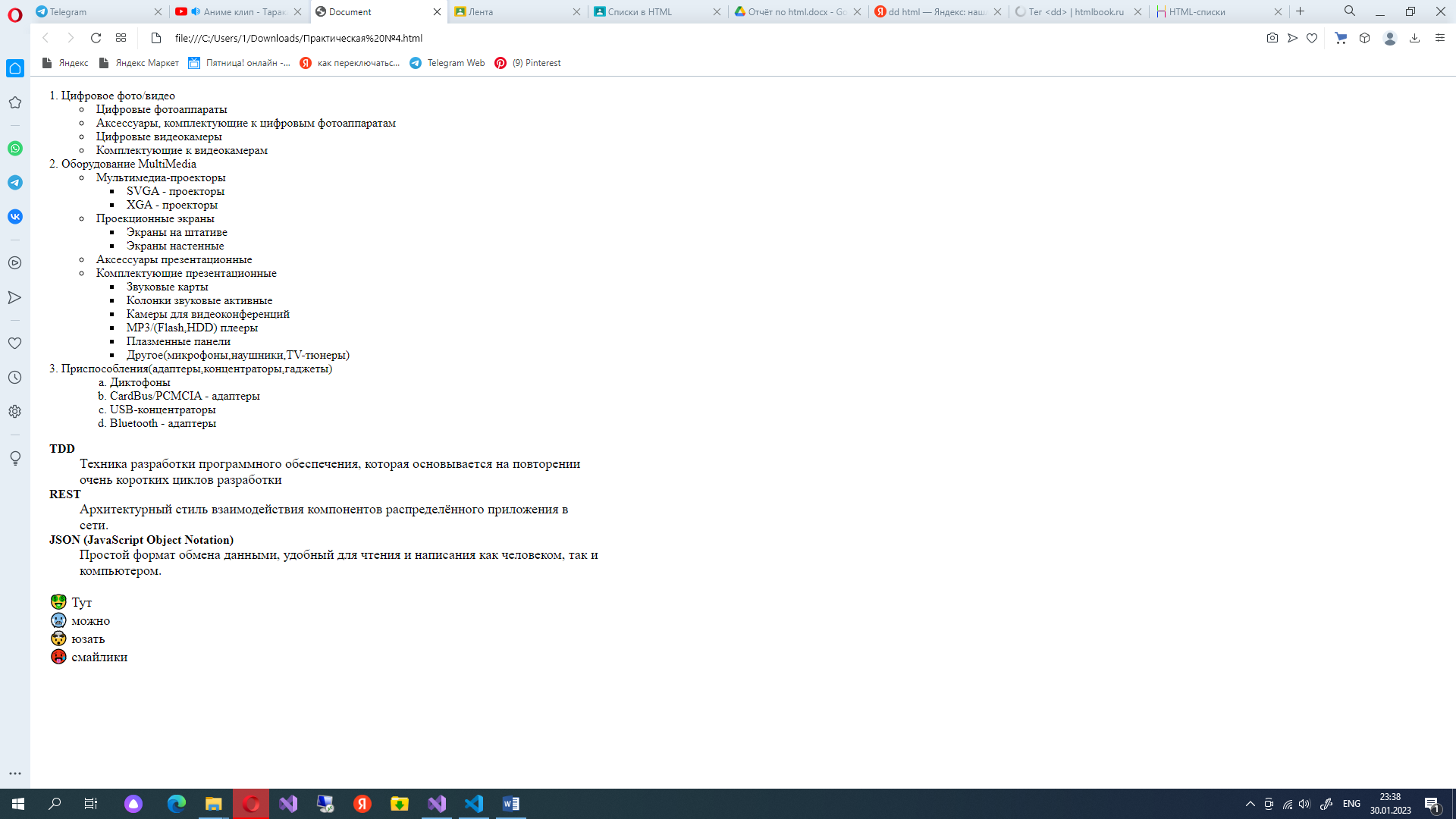


Рисунок 16 итог

Сверху вы можете увидеть итоговый вид переноса из ворда на html сайт. Но теперь мы можем его чуточку изменить под свой вкус.

Например, мне не нравится сами виды нумерации, я хочу изменить их на другие и сами смайлики не нравятся.

Поэтому я воспользуюсь другими вилами, которые я привела выше. Так же мне не нравится сами смайлики и текст, поэтому их я тоже заменяю, на заметку, не все смайлики сюда подходят, поэтому лучше использовать смайлики с этого сайта: <https://snipp.ru/handbk/emoji> .

Также можно создать вложенный список, для этого нам надо всего два тэга: li и ol. И в ol нам надо поставить какого вида у нас будет идти нумерация, они могут быть совершенно разными.

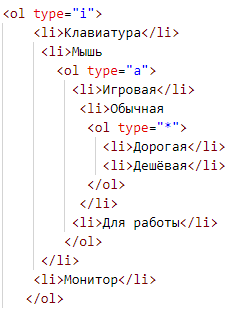


Рисунок 17 вложенный список

Так же можно вставить смайлики разными способами, это напрямую вставить и через код смайлика вставить. И можно это сделать несколькими методами. Первый вставить сам смайлик в текс. Второй вставить его CCS или Html код. Или же вставить через style. Как показано ниже.

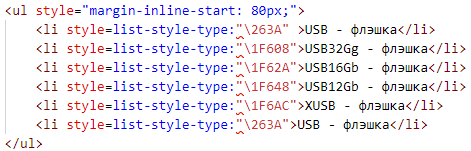


Рисунок 18 вставка смайликов

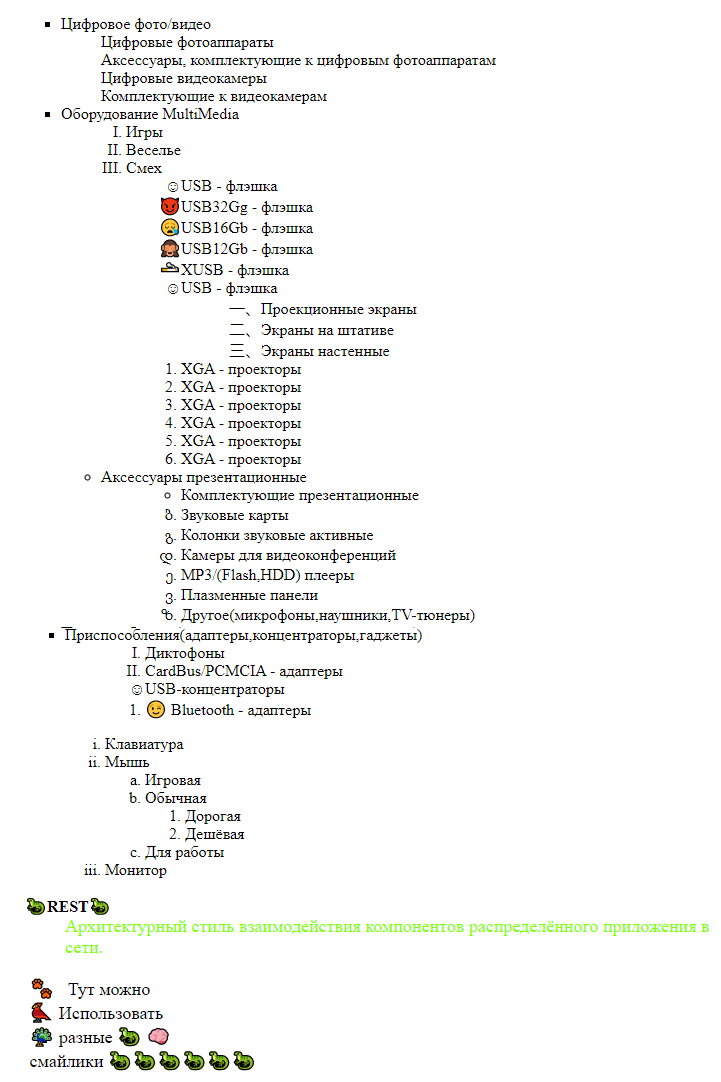


Рисунок 19 переделанная страница

Вывод: научилась работать со списками. Поняла, как делать разные нумерации списков. Изучила какие именно надо использовать тэги, узнала маркировки и нумерации списков.

# ПРАКТИЧЕСКАЯ РАБОТА №5

Тема: Работа с <div>.

Цель: Научиться использовать div в html. Понять зачем он нужен и когда используется. Как вызвать его и установить позицию окна.

Тег Div — тег, определяющий логическое разделение содержимого веб-страницы. Теги div можно использовать для центрирования блоков содержимого, создания эффектов столбцов, создания различных областей цвета и многого другого.

Так же есть ещё другие свойства, которое включённые значения, такие как размеры, положение и позиция.

Позиции можно разделить на следующие:

Позиционируемый элемент — это элемент, у которого вычисленное значение position является relative, absolute, fixed либо sticky. (Другими словами, это все, кроме static.)

Относительно позиционируемый элемент является элементом, вычисленное значение position которого является relative. Свойства top и bottom определяют смещение по вертикали от его нормального положения; свойства left (en-US) и right задают горизонтальное смещение.

Абсолютно позиционируемый элемент — это элемент, чьё вычисленное значение position является absolute или fixed. top, right, bottom и left (en-US) задают смещения от краёв содержащего блок элемента.



Рисунок 20 примеры

Для того что бы заполнить это окошко нужно написать div class=””, и в скобках нужно указать, какой класс вызывается. И внутрь можно записывать содержание определённого окна.

Так же если окно находится за другим, но при этом текст вылезет за то пространство, то текст не будет виден из-за этого окна.

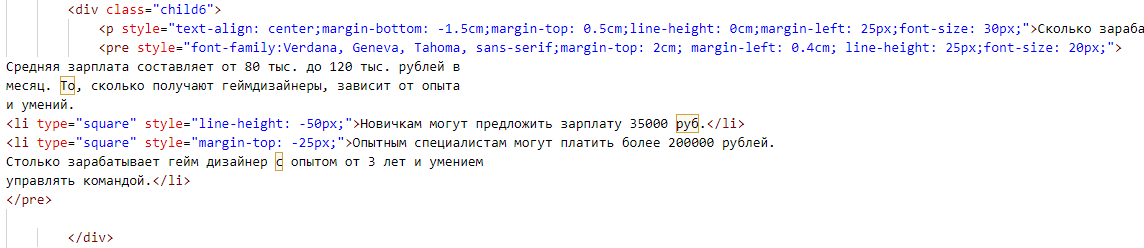


Рисунок 21 пример вызова

Вывод: Научилась использовать div в html. Поняла зачем он нужен и когда используется. Поняла как вызвать его и установить позицию окна.

# ПРАКТИЧЕСКАЯ РАБОТА №6

Тема: Создание класса css.

Цель: Научиться использовать класс css и отредактировать не используя, то есть тэг и атрибут style. Сделать 2 страницы сайта, под свой вкус.

Для того что бы создать, нам надо изменить расширение на css, и далее пы пишем классы, как делали в предыдущей работе.

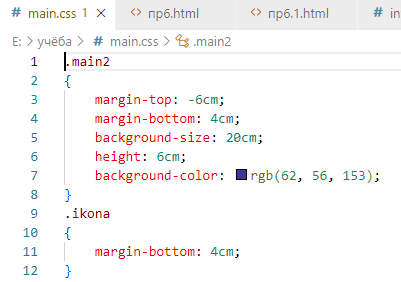


Рисунок 22 пример

Для того что бы использовать класс, нам для этого надо написать href и указать какой именно файл css надо использовать в противном случае файл не найдётся.

Для вызова класса нам надо написать class=”..” и в кавычках написать класс. В этих классах мы можем использовать атрибут style, которые мы изучили в прошлых практических. Мы не будем на них заострять внимания.

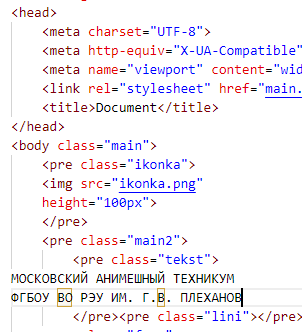


Рисунок 23 вызов класса

Так же для того что бы использовать css нам надо в head написать <link rel href>. Указывается путь к таблице стилей внутри атрибута href и атрибут rel со значением stylesheet. rel означает "отношения (relationship)", и, вероятно, является одной из ключевых особенностей элемента <link> — значение сообщает как указанный элемент связан с содержащим его документом.

Так же в этом работе из нового я использовала:

Letter-spacing – задаёт интервал между символами,

Linear-gradient – создаёт градиента на фоне, можно определить угол направления градиента, так же можно делать несколько цветов,

Border-radius – позволяет закруглять объекты, можно указать на сколько будет закругление, поэтому лучше использовать px, для более детального изменения.

Box-shadow - создание тени для блока. text-shadow: rgb(238, 235, 255) 3px 4px 7px;

Вывод: Научилась использовать класс css и отредактировала, не используя атрибут style. Сделала 2 страницы сайта, под свой вкус.

# ПРАКТИЧЕСКАЯ РАБОТА №7

Тема: FlexBox.

Цель работы: Научиться использовать FlexBox, переделать практическую работу 6 без использования position: absolute, научиться делать адаптивную вёрстку сайтов, что бы блоки не теряли свой первоначальный вид.

Сначала поймём, что такое FlexBox. Flexbox — это технология, созданная именно для раскладки элементов. С помощью. Flexbox можно легко выравнивать элементы по горизонтали и по вертикали, менять направление и порядок отображение элементов, растягивать блоки на всю высоту родителя или прибивать их к нижнему краю.

Для того что бы включить его надо написать следующее: display: flex. И далее мы уже пишем свойства, мы можем определить, как будет располагаться элементы на странице как будет выглядеть при уменьшении или увеличения экрана.



Рисунок 24 flexbox

Рассмотрим несколько свойств:

Justify-content – отвечает за расположение элементов на странице, то есть можно расположить их по центру или сделать одинаковый интервал между элементами.

Flex-wrap – отвечает за расположение элементов на одной строчке.

* + - nowrap (по умолчанию): все flex-элементы будут в одной строке
    - wrap: flex-элементы будут перенесены на несколько строк сверху вниз.
    - wrap-reverse: flex-элементы будут перенесены на несколько строк снизу вверх.

Flex-grow – это свойство определяет способность гибкого элемента расти, если это необходимо.

flex-shrink – это свойство определяет способность гибкого элемента сжиматься при необходимости.

align-items – это определяет поведение по умолчанию для того, как гибкие элементы располагаются вдоль поперечной оси на текущей линии.

flex-flow – это сокращение для flex-direction и flex-wrap свойств, которые вместе определяют основные и поперечные оси гибкого контейнера.



Рисунок 25 flexbox 2

Так же существует такая конструкция, она позволяет определить сколько места будет занимать определённый элемент на странице.

Сначала создаётся класс и там задаётся значение, далее указывается через умножение её значение для всех экранах, сколько всего должен занимать места на странице, если не указать flex, то вид может немного измениться и следующий элемент будет идти после этого на одной строке.

Далее с помощью media мы указываем сколько места какие элементы должны занимать, относительно друг друга. В скобках мы указываем значение для разных экранов 600 для маленьких, 800 для больших.

Так же есть другие свойства, они имеют другие предназначенья, но это основные по моему мнению.

Вывод: Научилась использовать FlexBox, переделала практическую работу 6 без использования position: absolute, научилась делать адаптивную вёрстку сайтов, что бы блоки не теряли свой первоначальный вид.

# ПРАКТИЧЕСКАЯ РАБОТА №8

Тема: Grid.

Цель работы: Научиться использовать Grid, переделать практическую работу 7 без использования position: absolute, FlexBox, научится делать адаптивную вёрстку сайтов, что бы блоки не теряли свой первоначальный вид, понять, что такое сетка и как с ней работать.

Сначала давайте поймём, что такое Grid и зачем он используется.

Grid — это набор горизонтальных и вертикальных линий, пересекающихся между собой. Поддерживается всеми основными веб-браузерами. Эта сетка — некая система координат для контента, который в ней размещается. Она выравнивает элементы по строкам и столбцам.

Чтобы создать макет на основе сетки, необходимо определить grid-контейнер — область с контекстом форматирования, где все дочерние элементы будут размещаться в соответствии с правилами сетки.

То есть grid чем-то похож на FlexBox, но здесь каждый элемент находится в отдельном месте.

Для того чтобы вызвать Grid, нужно написать display: grid.

Внутри нашего грид-контейнера у нас есть четыре дочерних элемента. Если мы не размещаем их явным образом, эти элементы будут расположены в гриде в соответствии с правилами авторазмещения, то есть, по одному элементу - в каждой из четырёх первых ячеек.

grid-template-columns: repeat(3, 1fr); - первое значение указывает сколько будет по горизонтали.

grid-template-rows: repeat(3, 100px); - здесь опять указываем кол-во по горизонтали и какой будет отступ от друг друга.

Grid container - выравнивает элементы сетки по оси блока (column) (в отличие от justify-items, который выравнивает по оси inline (row)). Это значение применяется ко всем элементам сетки внутри контейнера.

stretch - заполняет всю высоту ячейки (по умолчанию)

start - выравнивает элементы по начальному краю их ячейки.

end - выравнивает элементы по конечным краем их ячейки.

center - выравнивает элементы по центру ячейки.

baseline - выравнивание элементов по базовой линии текста.

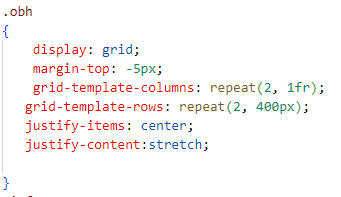


Рисунок 26 grid

justify-content - иногда общий размер вашей сетки может быть меньше размера ее контейнера. Это может произойти, если все элементы сетки имеют размеры в негибких единицах, например, px. В этом случае вы можете задать выравнивание сетки в контейнере сетки.

justify-content:stretch - изменить размер элементов сетки, чтобы сетка заполнила всю ширину контейнера сетки

justify-content:space-around - размещает равномерное количество пространства между каждым элементом сетки, с промежутками половинного размера на дальних концах.

Grid container:place-content - это объединение align-content и justify-content в виде одной записи

Вывод: Научилась использовать Grid, переделала практическую работу 7 без использования position: absolute, FlexBox, научилась делать адаптивную вёрстку сайтов, что бы блоки не теряли свой первоначальный вид, поняла, что такое сетка и как с ней работать.

# ПРАКТИЧЕСКАЯ РАБОТА №9

Тема: Медиа запросы.

Цель работы: научиться работать с медиа запросами, понять, что это такое и зачем нужно их использовать. Научится делать медиа запросы, посмотреть, как и зачем они делаются.

Медиа-запросы (media queries) – это правила CSS, которые позволяют управлять стилями элементов в зависимости от значений технических параметров устройств.

Иными словами, это конструкции, которые позволяют определять на основании некоторых условий какие стили необходимо использовать на веб-странице, а какие нет.

Они объявляются через тэг div и ему даётся название какого-то класса.

Далее в файле css мы пишем следующие, сам запрос @media и в скобках мы указываем максимальный и минимально размер экрана, то есть если размер экрана будет больше значения или равно, то применяем для большего, и аналогично с маленьким размером экрана.



Рисунок 27 media

Так будет выглядеть наш сайт для маленьких экранов.

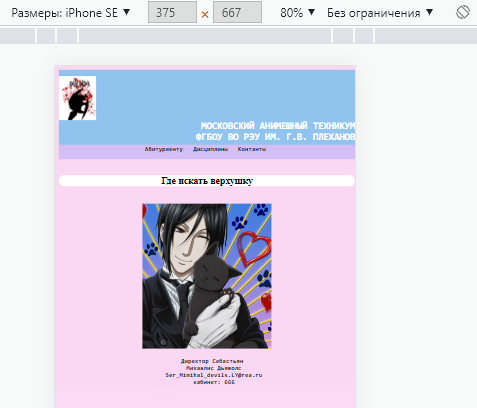


Рисунок 28 small пример

Так будет выглядеть для больших экранов наш сайт.

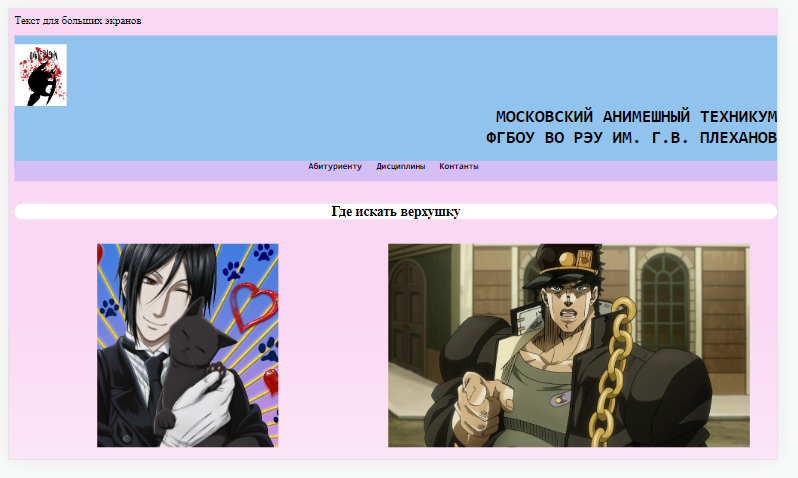


Рисунок 29 big пример

Если бы мы не использовали наш медиа запрос, тогда бы у нас просто изменилось размер объектов и всё, а так с помощью медиа мы можем скрыть один элемента если он не подходит условию запроса.

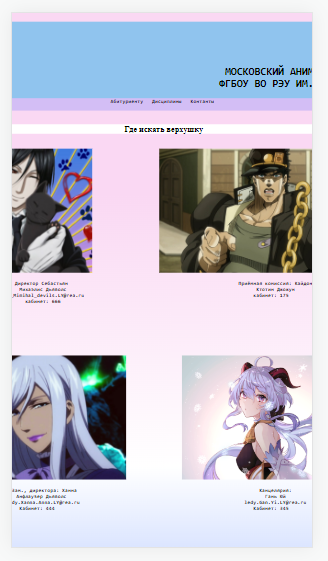


Рисунок 30 пример без media

Вывод: научилась работать с медиа запросами, поняла, что это такое и зачем нужно их использовать. Научилась делать медиа запросы, посмотрела, как и зачем они делаются.

# ПРАКТИЧЕСКАЯ РАБОТА №10

Тема: Итоговый сайт.

Цель работы: Научиться использовать все знания, которые приобрели в процессе обучения html. И создать свой сайт используя полученные знания, сделать сайт для нескольких устройств.

Для начала нам надо определится, что именно мы будем создавать, в моём случае – это аниме магазин, где будут продаваться разные товары. То есть мы должны создать главную страницу, страницу с товарами, страницу с ответами на часто задаваемые вопросы, страницу с сроками и отправкой посылок, а также страница с более подробным описанием товара.

Сначала мы определимся с размерами для экрана, в моём случае мы будем делать для 3 девайсов – это: телефон, айпад и ноутбук.

Для компьютеров мы сделаем экран, который больше 960 пикселей. Далее мы сделаем для телефонов, здесь нам нужна только по ширине экрана мы пишем для телефонов, которые будут больше 299 и меньше 500 пикселей и аналогично для айпадов. В первых фигурных скобках мы как-бы выключаем дисплей, а в последней мы мы указываем, что здесь может быть только один блок, для этого мы используем display:inline, то есть это свойство будет отвечать за то, что бы у нас не накладывался контент друг на друга.



Рисунок 31 медиа запросы

Далее мы будем рассматривать наш код на примере ноутбука. Ведь для всех медиа запросов содержание контента будет одинаковым, только они изменяются размеры контента и их местоположения.

Сначала мы разберём нашу приветственную страничку, где будет рассказывать о нашем магазине. Он выглядит так:



Рисунок 32 главная страничка

Сначала давайте разберём текст, где он будет расположен. Так как у нас слева фото, то мы должны сделать отступы, они делаются с помощью margin, далее мы видим задний фон для того что бы он находился позади текста мы должны указать position: relative – это относительное позиционирование. В данном случае положение элементов устанавливается относительно его исходного места, при этом если мы добавим такие свойства как: left, top, right и bottom, то мы изменим позицию элемента и сдвинем его в ту или иную сторону в зависимости от значений.

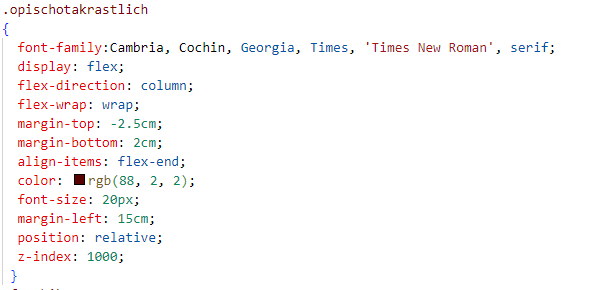


Рисунок 33 текст

Далее мы разберём задний фон у текста, мы размещаем его позади текста и используем для этого margin. Дальше мы используем фильтр и указываем на сколько нам размыть фон, также мы можем определить какой длины будет градиент.

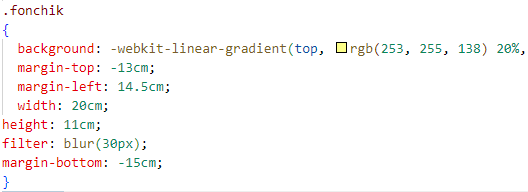


Рисунок 34 градиент

Далее давайте перейдём к ссылкам, которые находятся за красной линией, их можно создать по следующему коду, blank указывает что открывается новая страница. В div у нас находится красная линия.

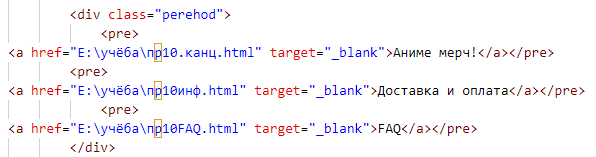


Рисунок 35 ссылка

Тут мы можем настроить цвет, длину, ширину и многое другое. border – это универсальное свойство для стилизации границ (рамки) вокруг элемента. Оно позволяет установить сразу три характеристики: толщину границ (border-width), стиль линий (border-style) и цвет (border-color). Мы указываем, что его нет.

Для того, чтобы сделать тень для линии мы пишем text-shadow – это добавление тени к тексту. Свойство задаётся разделённым запятыми списком теней, которые будут применены к тексту и к любым его свойствам decorations. Любая тень описывается комбинацией смещений по осям X и Y относительно элемента, радиусом размытия и цветом.

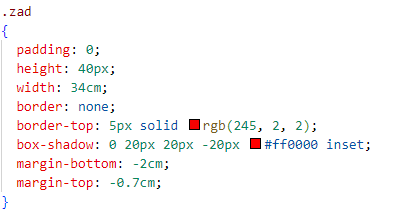


Рисунок 36 красная линия

Теперь давайте перейдём на страницу с товарами. У нас они будут располагаться столбцами, для того, чтобы они настраивались самостоятельно по размеру экрана с помощью flex. И его располагается с помощью flex-wrap и указываем значение, что если элементы не влезают на строку перенести на строки ниже.

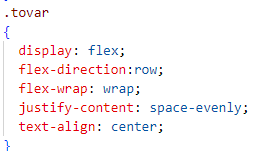


Рисунок 37 расположение

Теперь же перейдём на страницу для подробного описания товара, здесь мы создаём две картинки, его описание и возвращение обратно на выбор товара.

Здесь мы вставляем две картинки и более подробное описание нашего товара. Для текста мы делаем определённый размер шрифта, сделаем отступ что бы выглядело красиво и сделаем, что бы текст располагался посередине экрана.

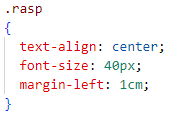


Рисунок 38 текст

Для перехода по страницам сделаем чуток другой стиль текста. Здесь отличается только размер шрифта, и отступом слева.

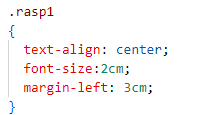


Рисунок 39 текст ссылки

Вот такой получился результат для телефона и ноутбука:



Рисунок 40 вариант для ноутбука



Рисунок 41 вариант для телефона

Сейчас рассмотрим страницы, которые предназначены для часто задаваемых вопросов и ответов про доставку.

Вот такой вариант получился для ноутбука, здесь довольно просто. Текст был изменен с помощью font-size, а линию я взяла с приветственной страницы. Для других же устройств я просто изменила размер и расположение текста.



Рисунок 42 вариант для ноутбука вопросы

И похожий вариант получился для странички с часто задаваемыми вопросами. Только здесь отсутствует переход на страничку о доставке.

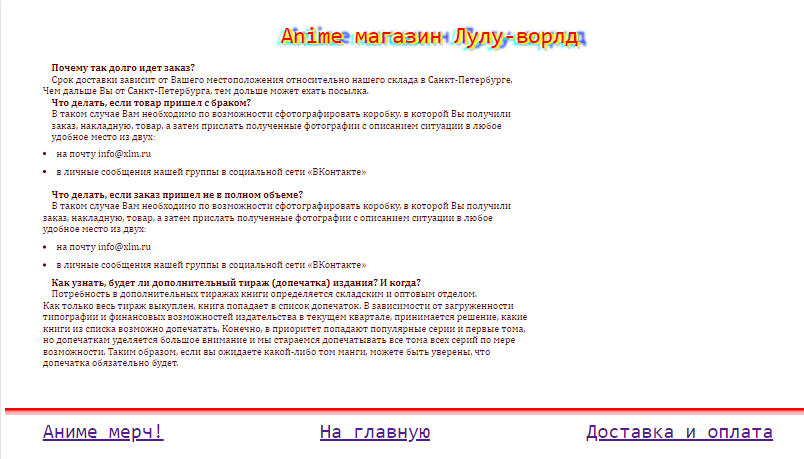


Рисунок 43 вариант для ноутбука FQA

Для того что бы сделать нумерацию я воспользовалась <li> благодаря ей у меня получилась простая нумерация с помощью кружков.

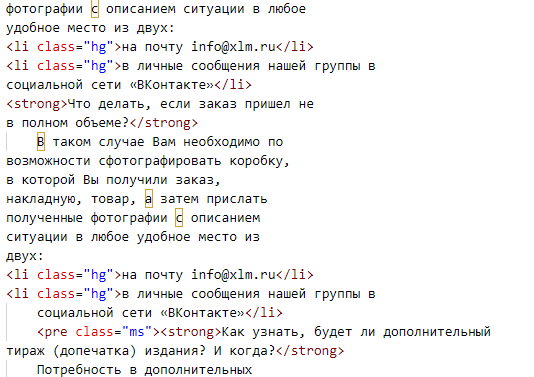


Рисунок 44 отрывок кода для текста

Так же здесь видно, что я использовала класс “hg”, этот класс я создала что бы подправить местоположение отступов.



Рисунок 45 отступы

Вывод: Научилась использовать все знания, которые приобрела в процессе обучения html. И создала свой сайт используя полученные знания, сделала сайт для нескольких устройств.