<P> абзац

<br> перенос на новую строку

<h1>-<h6> Заголовки. H1 может быть только один

<span> выделяет область, которую можно потом редактировать

<div> - это контейнер, в который можно запаковать другие теги и менять их местополежение в самом контейере, плюс так же можно менять характеристики этого контейнера.

В <div> можно так же запаковать другие дивы.

Стили можно прописать как в любом теге, так в html-файле, так и в css-файле. Применяется они в порядке очередности(каскадом), заменяя себя в порядке очереди.

Вес селекторов: <тег> < class < id < селектор прописан в самом теге

Если надо перекрыть CSS другим, то используется меньший из больших вариантов, что бы было пространство для коррекций в будущем! Например:

Div < div.main ~~~~~ main < #main2.main

**Можно:** прописывать одинаковые свойства сразу для нескольких классов .main, .main2 {}

прописать сразу 2 селектора тегу через пробел: class=”main main2”

обратиться к тегу через 2 селектора через точку: .main.main2

CSS подключается через тег:

<link rel="stylesheet" href="static/styles/style.css">

<a> гиперссылка, вставляется в href=””. Если дописать target=”\_blank” откроется с новой вкладки

Если дописать display-block, то ссылка запонит всю область, станет кнопкой.

<ol> упорядоченный список,<ul> неупорядоченный. Внутри этого тега элементы оборачивают в<li>

В <ul>: list-style: none; убирает точки

<img> ссылка на изображение(url или локально)

Margin~ширина края(отступа) блока от другого блока

Padding~отступ текста блока, от его края, внутрь блока

Display = inline~делает из блока, заполняющего всю ширину(block)-инлайн блок(по размеру текста)

<к инлайновым блокам не применимы высота и ширина, они их игнорируют, для учета этих параметров добавляют свойство inline-block, это позволяет задать 1 параметр, а 2й будет инлайн>

Display-flex одномерный контейнер, распологает элементы рядом\столбцом и тд

Flex-direction куда направлено расположение

justify-content центровка, работает только во flex и грид(сетка) и элементы стоят в ряд

флекс контейнер не переносит содержимое внутри себя как другие

display: none; скрывает объект

float – позволяет вести себя блоку как инлайновый, другие блоки будут его обтекать.(не работает во flex)

border-style: solid делает рамку вокруг контейнера.

border-width: 0.5px; ширина рамки

font-weight: bold; текст-жирный

В первой части тега можно дописать class=”имя” и css файле обращаться к этому классу через точку: <div class="conteiner"> ~~~ .conteiner {}

text-align: center расположение контента

Можно писать при создании тега div.container и создасться тег div класса container

.container>img обращение к тегу img который находится только в этом классе

<input type="text"> поле ввода текста

placeholder="Поиск" добавляется в конец тега, надпись в поле.

@media (max-width: 900px) {} в скобках прописывается какие параметры будут меняться при выходе за указанные параметры(>900px в данном случае)

.header>div {

    width: 100%;

}

.header>div>img {

    width: 100%;

Позволяет отформатировать картинку по ширине контейнера

.card>img {

    width: 100%;

}

Масштабируется строго в контейнере ,заполняя его, без него выйдет за рамки контейнера!

a {text-decoration: none;} убирает подчеркивание.

<div class=”” id =””> Обращение к классу через точку, к id через #

border-style: solid !important;

эта приписка ставит условие в абсолютный приоритет, его нельзя никак теперь изменить

.main>p обращение к тегу <p> в классе main

.main p обращение ко всем вложенным тегам p в классе main

1em=стандартному шрифту(16px), меняя этот параметр меняется размер по отношению к РОДИТЕЛЬСКОМУ шрифту(если есть вложенность, но шрифт меняется каскадно)

1rem меняет шрифт по отношению к ГЛАВНОМУ шрифту(во вложенности шрифт меняется 1 раз)

Margin отталкивается от шрифта (общего размера текста), соответственно при каскадном EM блоки будут смещаться асинхронно

display: grid; многомерная сетка

grid-template-columns: 1fr 1fr 1fr; размер(ширина) каждой колонки и сколько их будет(3)

grid-template-rows: repeat(3); сколько будет столбцов. Рипит-сколько раз повторим(можно указать ширину в скобках после запятой)

Для позиционирования в GRID контейнере его разбивают на количество колонок, а потом каждой части страницы(нав, поле текста, картинки и тд) приписывают ширину в колонках, в отрезке не учитывается последняя цифра(как в питоне).

.main-container {

display: grid;

grid-template-columns:repeat(12, 1fr);

grid-template-rows: 1}

.left-bar {

grid-column: 1/3;}

.container {

grid-column: 3/11

.navlist a:hover Псевдокласс hover говорит что происходит с полем при наведении на него(для мобильной версии сработает не при наведении, а при нажатии)

cursor: pointer; подсвечивает, что эту область можно нажать.

Для реализации “бургера” надо :

display: block;

        position: absolute;

        right: 1px;

прописать свойство position: absolute; и тогда контейнер, к которому он будет применен будет вырван из своего местоположения и прикреплен к другому,который есть в этом контейнере. При блоке, он выстроится в столбец, при флексе-в ряд. Если Бургерный блок перенести в правую часть страницы, то для корректного отображения надо указать позицию-right 1px.

Для разных устройств можно через none убирать отображение ёмких блоков и засовывать их функционал в один контейнер, указывая как он будет отображаться.

В Bootstrap экран разделён на 12 долей.