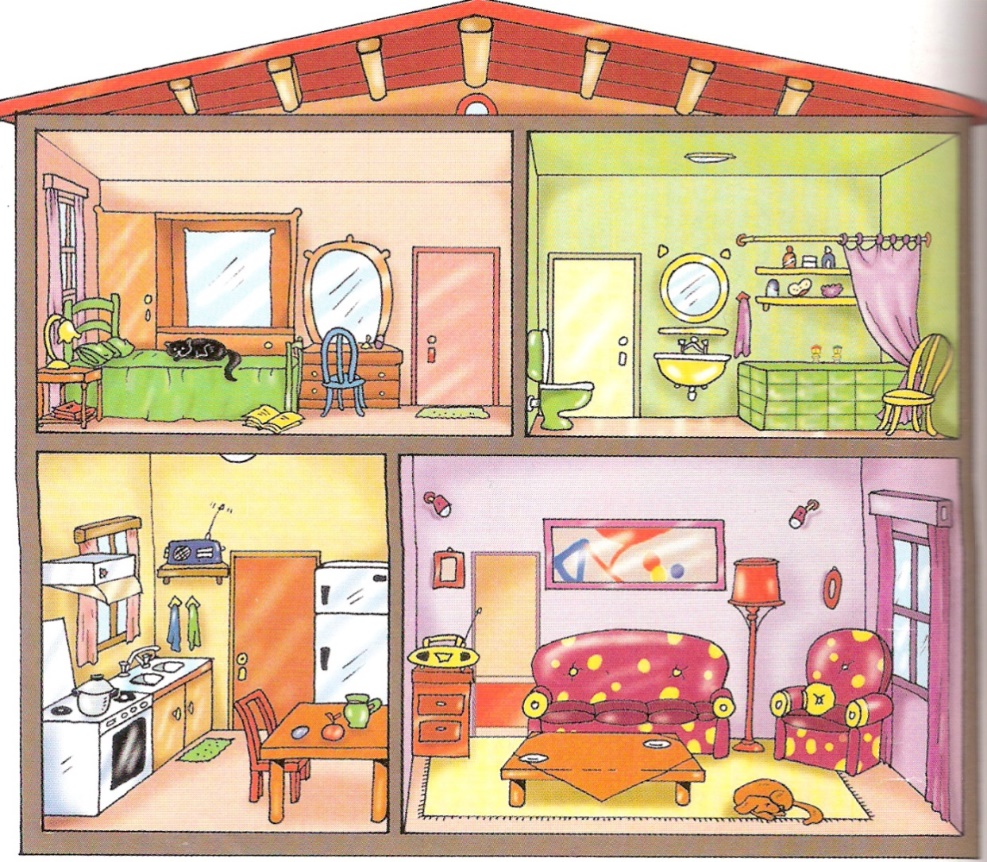
**HTML**

HTML не является языком программирования, это язык разметки. Он используется для расположения элементов на странице в браузере. HTML имеет блочную структуру, то есть страница буквально выстраивается по кирпичикам. Объекты наслаиваются друг на друга или оборачивают дочерние элементы.

Допустим, у нас есть дом:



Давайте посмотрим на рисунок дома и попробуем разбить его на блоки:

1. Дом
   1. Крыша
      1. Тут напишем, что наш дом состоит из дерева и кирпичей.
      2. Название адреса – ул. Кошачья д. 10
   2. Тело дома
      1. Спальня
         1. Кровать
            1. Котик
         2. Стул
         3. Трюмо с зеркалом
         4. Шкаф
         5. Тумбочка
            1. Лампа
      2. Ванна
         1. Зеркало
         2. Ванна
         3. Стул
         4. Занавеска
         5. Раковина
         6. Унитаз
      3. Кухня
         1. Холодильник
            1. Продукты
         2. Стол
            1. Столовые приборы
         3. Кухонный гарнитур
      4. Гостиная
         1. Диван
         2. Кресло
         3. Стол
            1. Блюдца
         4. Картина
         5. Собака

Перепишем наш дом на псевдо-Html:

<дом>

<крыша>

<состав>Дерево и кирпичи</состав>

<адрес> ул. Кошачья д. 10 </адрес>

</крыша>

<тело дома>

<спальня>

<кровать> Котик </кровать>

<стул></стул>

<зеркало></зеркало>

<шкаф></шкаф>

<тумбочка> Лампа </ тумбочка>

</спальня>

<ванная комната> … </ванная комната>

<кухня> … </кухня>

<гостиная> … </гостиная>

</тело дома>

</дом>

Основой HTML-документа являются парные теги <> </> или непарные <>.

В соответствии с объектной моделью документа («Document Object Model», коротко DOM), каждый HTML-тег является объектом. Вложенные теги являются «детьми» родительского элемента. Текст, который находится внутри тега, также является объектом.

Разберем простой шаблон для создания страницы:

<! DOCTYPE *HTML*>  
 <html *lang*="ru">  
 <head>  
 <title>Название сайта</title>  
 <link *rel*="stylesheet" *href*="style.css">  
 </head>  
 <body>  
 Содержимое  
 <script *src*="index.js"></script>  
 </body>  
 </html>

, где

1. <!DOCTYPE *HTML*> - обозначим, что эта страница Html
2. <html *lang*="ru"></html> - самый главный тег (обёртка), с указанием языка страницы.
3. <head></head> - “шапка” сайта. Используется для хранения информации о странице и подключения внешних файлов.
   1. <title> </title> - название сайта. Отображается в браузере на вкладке.
   2. <link *rel*="stylesheet" *href*="style.css"> - это не парный тег “соединение”. Должен находится в “Шапке”, у него есть атрибуты rel и href. Rel отвечает за вид документа, href за ссылку на файл.
4. <body></body> - тело сайта. Всё содержимое страницы должно находиться внутри этого тега.
   1. <script *src*="index.js"></script> - файл со скриптом подключается в конце body, чтобы у JavaScript уже было все нужное для работы с DOMом. Атрибут src принимает ссылку на файл.

Основные теги для отображения содержимого, которые мы будем использовать:

* div – классический блок, который не несет специального смысла, воспринимается как коробка с содержимым.
* h1…h6 – заголовки от самого большого h1 до более мелкого h6.
* p – абзац, в котором текст обособляется переносами строки.
* span – текст без определенного назначения. Используется для вторичного текста.
* a – ссылка.
* ul – список (как этот список тегов).
* li – элемент списка (строка в списке).
* input – поле ввода.
* button – кнопка.
* label – обертка для элементов интерфейса, кнопок, полей ввода (каждый в свой).
* form – обертка для label’ов.

Каждому тегу можно добавить атрибуты, для уточнения специфики и стилизации объекта. Если у вас есть один особенный объект и вы хотите к нему обращаться в поиске или с помощью JavaScript, создайте ему атрибут id, если у вас есть несколько таких объектов, добавьте им атрибут class.

Например так:

<div>

<span id = ”first\_cat” class = ”cats”>Первый котеночек</span>

</div>

**CSS**

CSS (Cascading Style Sheets) — это код, который вы используете для стилизации вашей веб-страницы. Он может находиться прямо на странице между тега <style></style> или в отдельном файле. Метод подключения стилей из отдельного сss файла практикуется намного чаще.

Допустим, у нас есть изображение и нам нужно описать на псевдо-CSS ее стиль:

Изображение выглядит как вычерчивание линий

Автоматически созданное описание

Котенок {

Эмоция: подмигивание;

Цвет банта и ушек: розовый;

Цвет глаз: голубой;

Цвет ленточки: розовый;

Добавить содержимое: сердечко;

Положение: повернуться боком и поднять лапку;

}

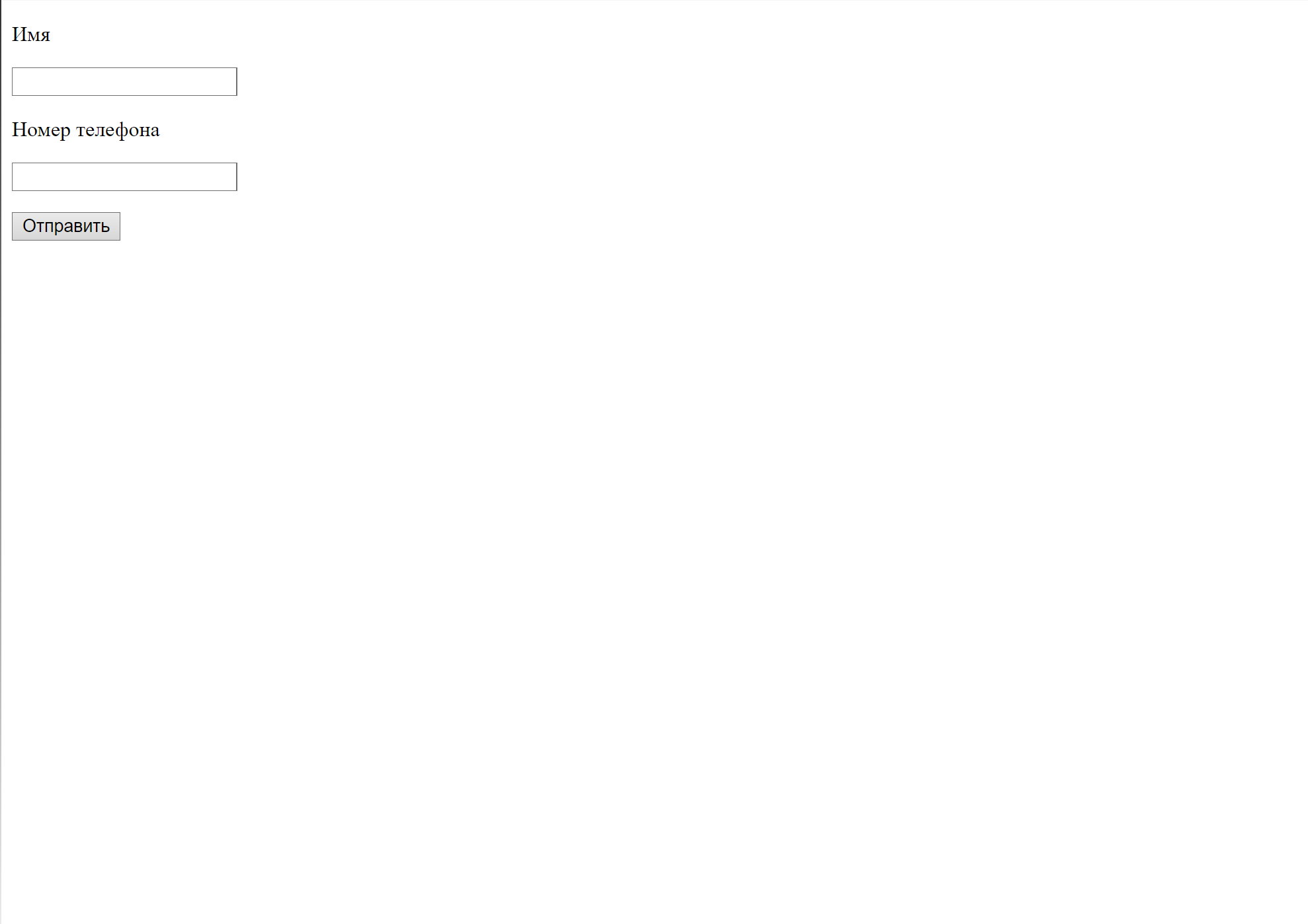
Результат:

Изображение выглядит как текст

Автоматически созданное описание

Теперь мы готовы разобрать пример на настоящем CSS. Допустим, у нас есть небольшая форма для отправки номера телефона и имени:

<!DOCTYPE *HTML*>  
<html *lang*="ru">  
<head>  
 <title>Форма</title>  
</head>  
<body>  
<div *id*="div\_form">  
 <form *id*="form">  
 <div *class*="text\_name">  
 <p>Имя</p>  
 </div>  
 <label>  
 <input *class*="for\_input">  
 </label>  
 <div *class*="text\_name">  
 <p>Номер телефона</p>  
 </div>  
 <label>  
 <input *class*="for\_input">  
 </label>  
 <p></p>  
 <label>  
 <button *id*="button">  
 Отправить  
 </button>  
 </label>  
 </form>  
</div>  
</body>  
</html>



Выглядит это не очень современно. Поэтому мы воспользуемся CSS и улучшим внешний вид нашей формы.

<style>  
 body {  
 font-size: 20px; /\*Размер шрифта \*/  
 }  
  
 #div\_form {  
 position: fixed; /\*Позиция зафиксирована, при прокрутке страницы блок будет всегда на одном месте\*/  
 width: 100vw; /\*Ширина блока равна 100% ширине окна браузера, ее можно задавать в px \*/  
 height: 100vh; /\*Высота блока равна 100% высоте окна браузера , ее можно задавать в px \*/  
 display: flex; /\* Добавим возможность растягиваться\*/  
 align-items: center; /\*Положение по вертикали - центр\*/  
 justify-content: center; /\*Положение по горизонтали - центр\*/  
 }  
  
 #form {  
 width: 200px;/\*Ширина блока в px \*/  
 }  
  
 .text\_name {  
 text-align: center; /\*Положение текста посередине \*/  
 }  
  
 .for\_input {  
 padding: 10px; /\*отступ внутрь у объекта\*/  
 width: 180px; /\*Ширина поля в px \*/

border-style: double; /\*стиль границ\*/  
 border-radius: 10px; /\*радиус закругления границ\*/

}  
  
 #button {  
 background-color: black;/\* Цвет заднего фона \*/  
 color: white;/\* Цвет текста \*/  
 padding: 10px;/\*Отступ внутрь у объекта\*/  
 width: 180px;/\*Ширина кнопки в px \*/  
 margin-left: 10px;/\* Отступ снаружи слева \*/

border-radius: 10px; /\*радиус закругления границ\*/  
 }  
</style>

Результат:

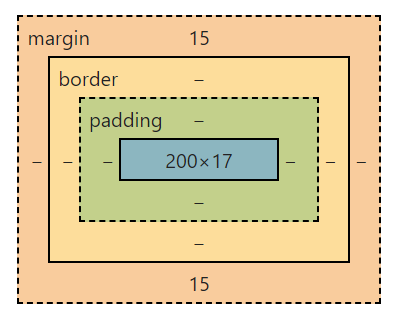


Работа со стилями:

Обращаться к элементам и блокам можно тремя способами, через id, class и teg:

* #cat{} - для обращения к id = “cat”
* .cat{} - для обращения к class = “cat”
* div{} - для обращения к <div></div>

Основные свойства элементов:

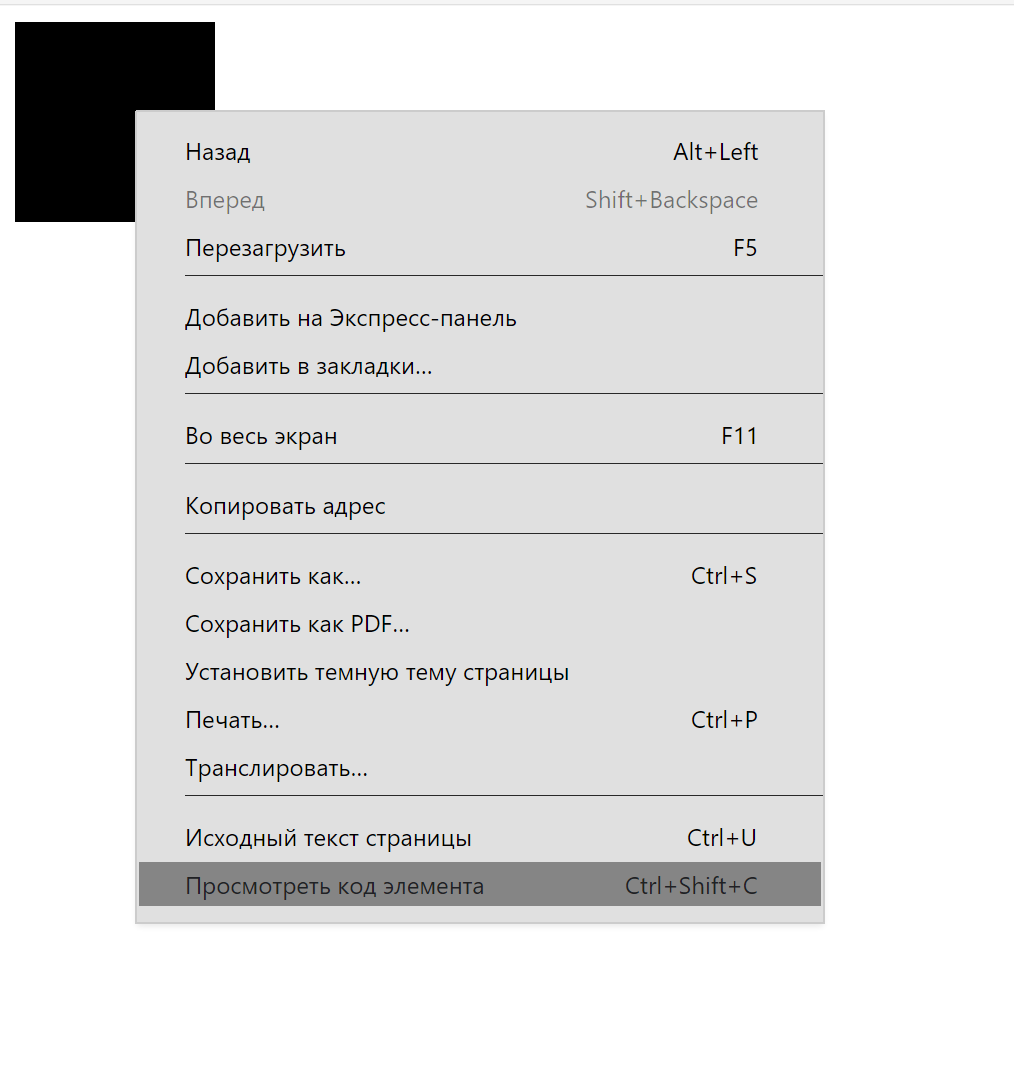
1. Отступы
   1. padding: - внутренний отступ
      1. Сверху | Справа | Снизу | Слева. Например: padding: 1px 2px 3px 4px;
      2. Вокруг. Например: padding: 20px;
      3. По вертикали | по горизонтали. Например: padding: 5px 10px;
   2. border – отступ в виде обводки
      1. толщина || стиль || цвет Например: border: 1px dashed black;
      2. -width – толщина границы. Например: border-width: 0.5px;
      3. -style – стиль границы. Например: border-style: dashed;
      4. -color – цвет границы. Например: border-width: black;
      5. -radius – радиус закругления границ. Например: border-radius: 5px;
   3. margin: - внешний отступ
      1. Сверху | Справа | Снизу | Слева. Например: margin: 1px 2px 3px 4px;
      2. Вокруг. Например: margin: 20px;
      3. По вертикали | по горизонтали. Например: margin: 5px 10px;
      4. 
2. Размеры
   1. width – ширина Например: width: 200px;
   2. height – высота Например: height: 200px;
   3. max-width/ min-width – максимальная/минимальная ширина Например: max-width: 200px;
   4. max- height / min- height – максимальная/минимальная высота Например: max-height: 200px;
   5. Меры исчисления:
      1. px – пиксели – статичная мера.
      2. % - проценты размера экрана – динамичная мера.
      3. vw - % от ширины, даже если относится к высоте.
      4. vh - % от высоты, даже если относится к ширине.
      5. em — относительно размера шрифта в элементе.
3. Положение
   1. position - указывает, как элемент позиционируется в документе.
   2. Виды значения:
      1. static – значение по умолчанию.
      2. absolute – относительно границ документа, перемещается с документом при скролле.
      3. fixed – относительно границ окна браузера, не перемещается.
      4. relative – относительно своего нормального положения.
   3. left/top/right/bottom – позиция позиционируемого элемента по определенной стороне. Например: top: 100px;
4. Выравнивание (стили родителя)
   1. оболочка flex-box – это метод адаптивного позиционирования дочерних элементов внутри родителя со свойством display:flex;
   2. flex-direction – расположение дочерних элементов в строку(raw) или в столбец(column).
   3. flex-wrap - возможность переносить элементы на новую строку wrap/nowrap.
   4. align-content – распределение пространства между и вокруг элементов контента вдоль поперечной оси строки или столбца.
   5. justify-content – распределение пространства между и вокруг элементами контента перпендикулярно поперечной оси строки или столбца.
   6. align-items – выравнивает flex-элементы текущей flex-линии.

Чтобы лучше разобраться во flex-box, можно воспользоваться инструментом разработчика в браузере.

Запустите файл index.html, добавив в него код, представленный ниже:

<!DOCTYPE *HTML*>  
<html *lang*="ru">  
<head>  
</head>  
<style>  
 #parent {  
 position: fixed;  
 width: 100vw;  
 height: 100vh;  
 display: flex;  
 }  
  
 #children {  
 width: 100px;  
 height: 100px;  
 background-color: black;  
 }  
</style>  
<body>  
<div *id*="parent">  
 <div *id*="children"></div>  
</div>  
</body>  
</html>

Страница состоит из “родителя” (страница во всё окно) и “дочернего элемента” (черный квадрат). Нажмите на квадрат правой кнопкой мыши и выберите пункт “Посмотреть код элемента”.



Откроется инструмент разработчика. В нем нужно нажать на блок “родитель” c id = “parent”.

Изображение выглядит как текст

Автоматически созданное описаниеИзображение выглядит как текст

Автоматически созданное описание

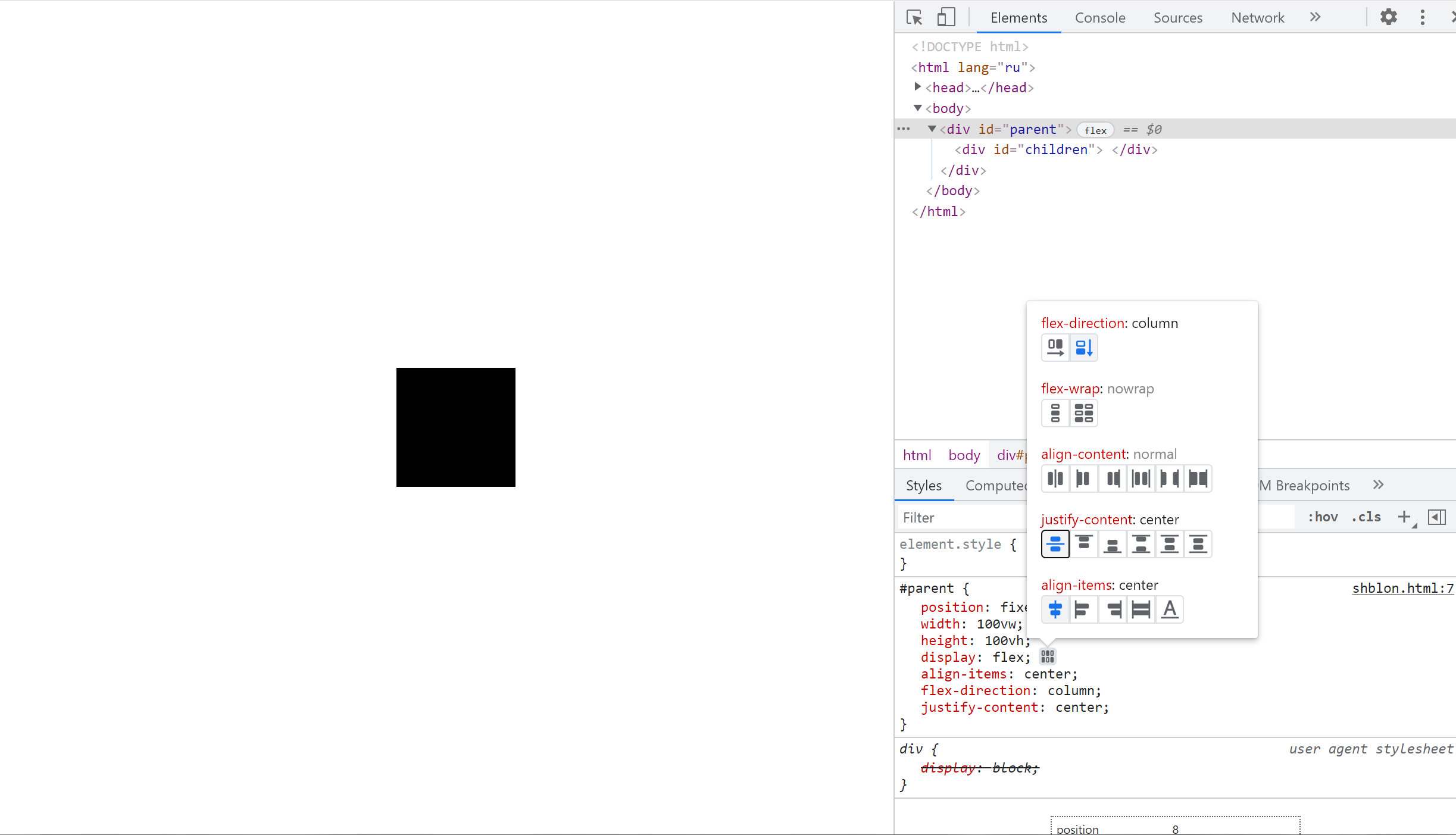
Внизу будут представлены стили выбранного элемента. Около свойства display появится значок ручной настройки flex. При нажатии на него откроется меню для экспериментов с flex. При нажатии на любую кнопку из этого меню вы заметите изменение положения квадрата и смена стилей этого элемента.

Наблюдая за изменениями в стилях, анализируйте и запоминайте. Используя этот способ, понять flex намного проще.

Изображение выглядит как текст

Автоматически созданное описание

Пример одного из способов центрирования элемента:



**Библиотека стилей Bootstrap**

Современные разработчики только в уникальных случаях используют рукописные стили. Существует множество библиотек, которые подключаются тегом <link> в “шапке” сайта. Разберем их работу на примере Bootstrap.

Суть работы заключается в том, что элементы на странице должны использовать уже прописанные в библиотеке классы. Все стили и принципы их работы уже описаны в документации к Bootstrap.

Разберем наш пример с формой отправки имени и номера телефона.

1. Подключим файл bootstrap.min.css, используя link с официального сайта   
   <https://getbootstrap.com/docs/5.1/getting-started/introduction/>:  
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">
2. Используем классы из библиотеки:
   1. form-label – для названий полей ввода.
   2. form-control – для полей ввода.
   3. btn btn-dark – для кнопки (да, есть возможность использовать несколько классов)

Они находятся в пунктах Forms и Components:

Изображение выглядит как текст

Автоматически созданное описание  
Изображение выглядит как текст

Автоматически созданное описание

При использовании классов bootstrap, нам достаточно настроить размер и расположение. Такой код у нас получится в результате:

<!DOCTYPE *HTML*>  
<html *lang*="ru">  
<head>  
 <title>Форма</title>  
 <link *href*="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" *rel*="stylesheet"  
 *integrity*="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" *crossorigin*="anonymous">  
</head>  
<style>  
 p {  
 font-size: 15px;  
 padding-top: 10px;  
 margin-bottom: 0;  
 }  
  
 #div\_form {  
 position: fixed;  
 width: 100vw;  
 height: 100vh;  
 display: flex;  
 align-items: center;  
 justify-content: center;  
 }  
  
 #form, .btn {  
 width: 200px;  
 }  
  
</style>  
<body>  
<div *id*="div\_form">  
 <form *id*="form">  
 <div *class*="form-label">  
 <p>Имя</p>  
 </div>  
 <label>  
 <input *class*="form-control">  
 </label>  
 <div *class*="form-label">  
 <p>Номер телефона</p>  
 </div>  
 <label>  
 <input *class*="form-control">  
 </label>  
 <p></p>  
 <label>  
 <button *class*="btn btn-dark">  
 Отправить  
 </button>  
 </label>  
 </form>  
</div>  
</body>  
</html>

Результат:

