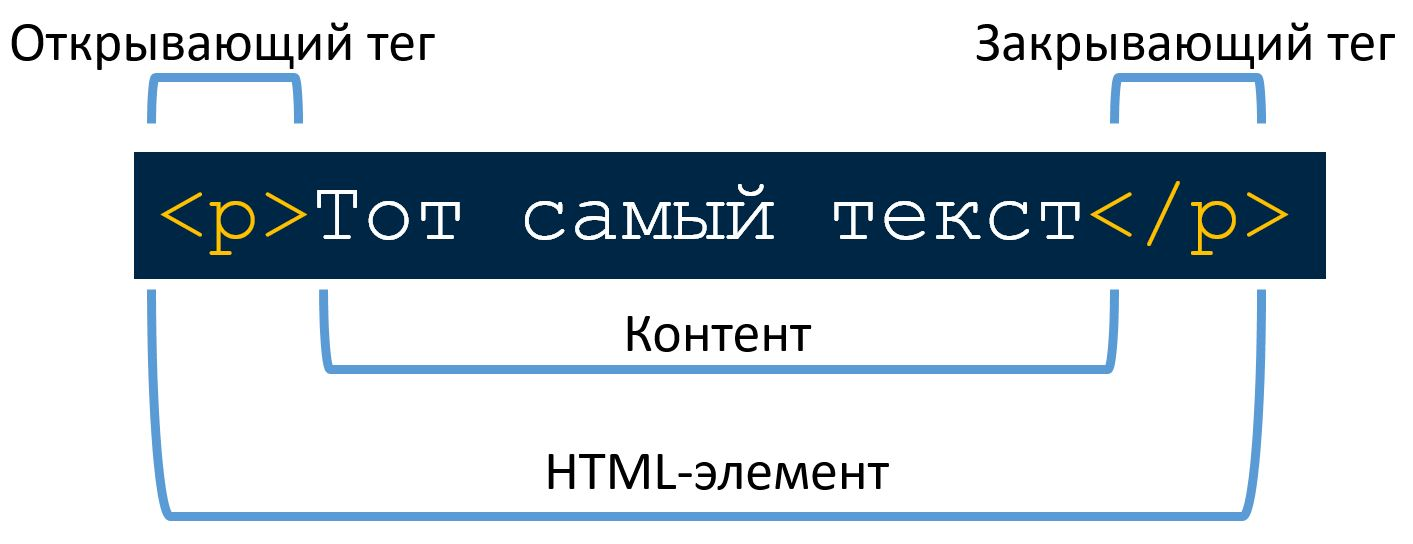
## HTML (HyperText Markup Language)

HTML (HyperText Markup Language) — это «язык гипертекстовой разметки», который определяет содержание и структуру веб-страниц. Это не язык программирования, а язык разметки, с помощью которого браузер отображает веб-страницы. HTML состоит из элементов — тегов, позволяющих организовывать и управлять контентом страницы: создавать ссылки, форматировать текст, вставлять изображения, видео и прочие мультимедийные элементы.

## Состав HTML-элемента

Тег — это HTML-элемент, состоящий из имени элемента окруженного < и >



* Открывающий тег: задает начало HTML-элемента
* Закрывающий тег: служит концом HTML-элемента. Частая ошибка новичков - потеря закрывающего тега, однако современные редакторы автоматически умеют закрывать теги.
* Контент: это содержимое между открывающим и закрывающим тегом. В данном случае это текст. Контента может и не быть - т.е. тег открылся и закрылся <div></div>.

Основные теги:

* <p>, <br> — для абзаца и переноса на другую строку.
* <h1>, <p>, <div>, <span> и другие — для организации текста и мультимедийных элементов.
* <b>, <i>, <u>, <strong>, <em> и так далее — для форматирования текста.
* <ul>, <ol> и <li> — для создания маркированных и нумерованных списков.
* <a> — для создания ссылки на определённый раздел страницы или другой веб-документ.
* <img>, <video>, <audio> и так далее — для вставки изображений, видео, аудио и других мультимедийных элементов.
* <button>, <form>, <input>, <textarea> — для создания кнопок и форм для ввода данных и так далее.

Многие теги имеют парный закрывающий тег, например:

<p>Текст абзаца</p>

Есть одиночные теги, которые не требуют закрытия, например <br> для переноса строки или <img src=”image.jpg”> для вставки изображения.

Семантический HTML

Семантические теги чётко описывают смысл содержимого, облегчая понимание кода как для людей, так и для поисковых систем. В отличие от несемантических тегов, таких как <div> и <span>, семантические элементы уточняют назначение контента.

Вот основные примеры семантических HTML-тегов:

* <header> — заголовок страницы или раздела.
* <nav> — навигационное меню.
* <main> — основное содержание документа.
* <article> — независимый, автономный контент, который может быть отдельно распространён или использован.
* <section> — раздел, группирующий связанный контент.
* <aside> — контент, косвенно связанный с основным содержанием (например, боковая панель).
* <footer> — нижний колонтитул страницы или раздела.

Вложенные элементы

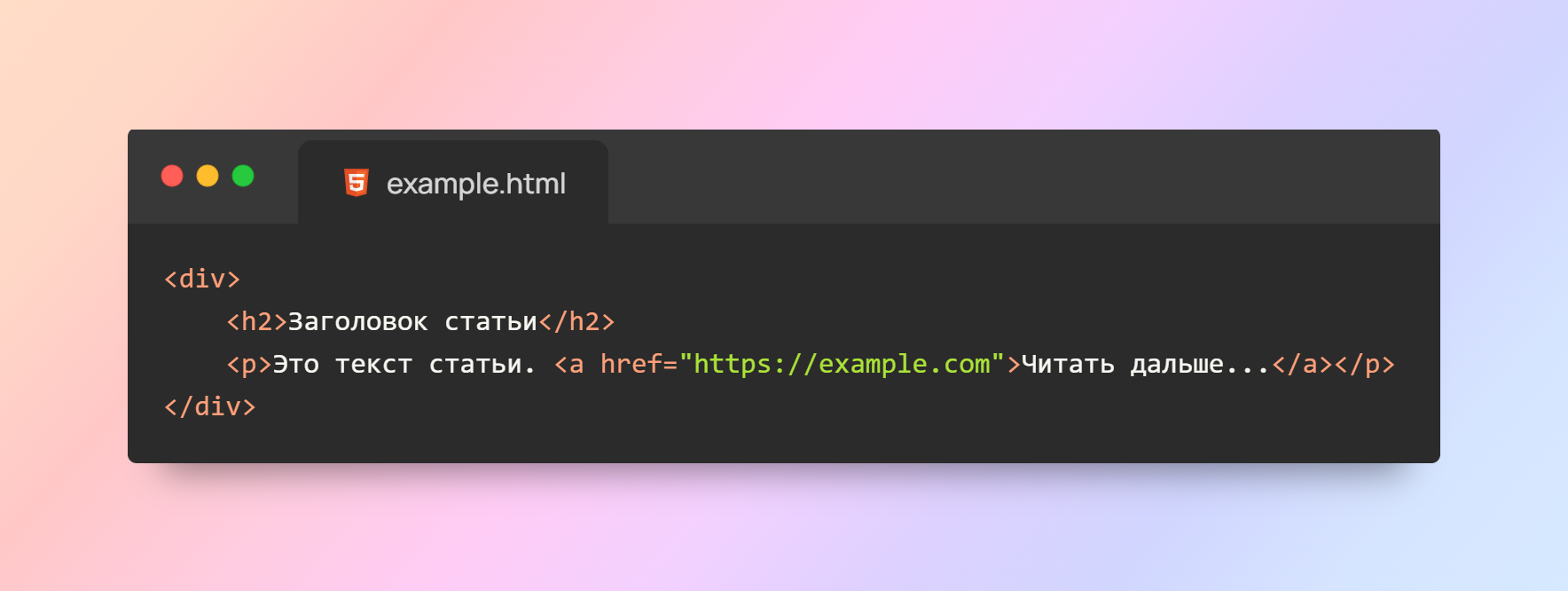
HTML-элементы могут содержать другие элементы внутри себя. Например, внутри блока <div> можно разместить заголовок и абзац с гиперссылкой:

<div>

<h2>Заголовок статьи</h2>

<p>Это текст статьи. <a href="https://example.com">Читать дальше...</a></p>

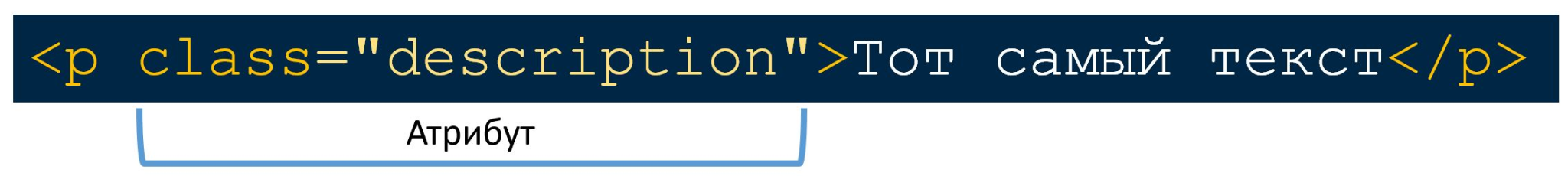
</div>



В этом примере:

* Элемент <h2> и элемент <p> вложены в элемент <div>.
* Внутри <p> также есть элемент <a>.

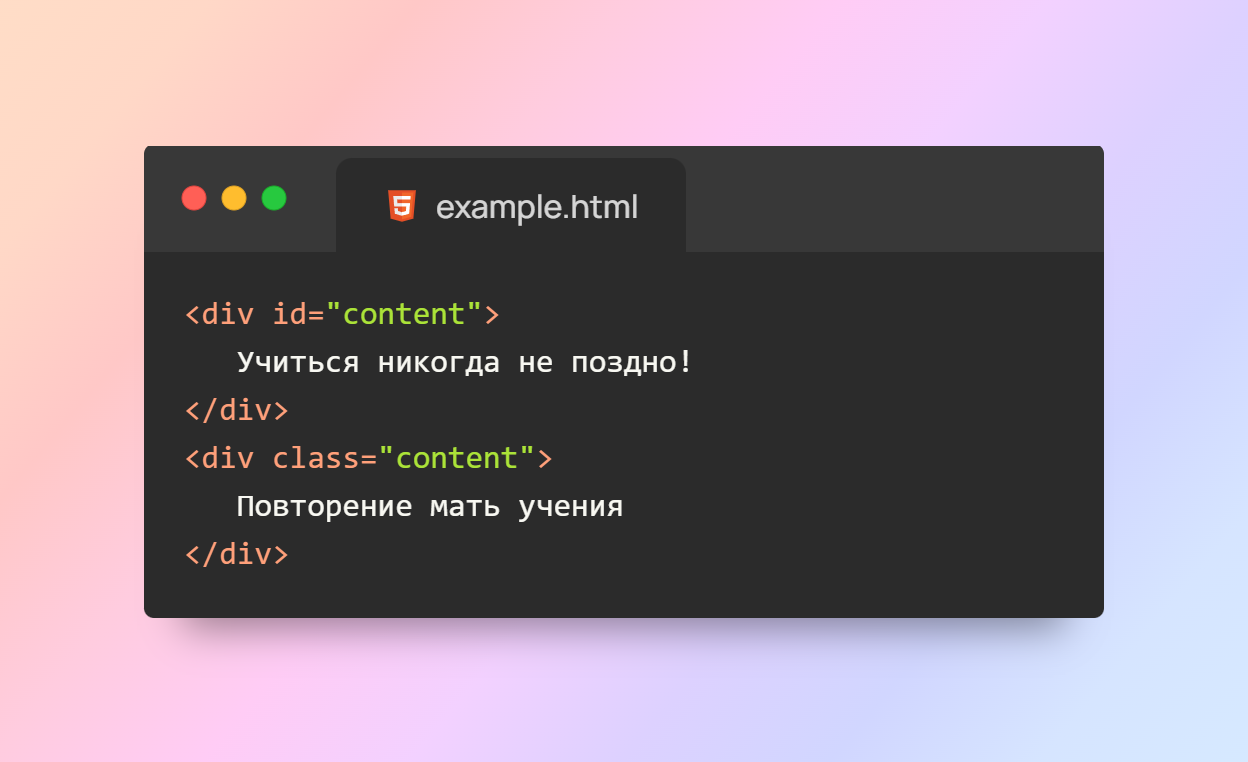
Такая структура часто используется для создания семантически корректного HTML-кода, где один элемент может содержать несколько подэлементов.

Атрибуты 

Атрибуты используются для предоставления дополнительной информации о HTML-элементах. Они обычно содержат пару “имя-значение” и располагаются внутри открывающего тега. Вот несколько часто используемых атрибутов:

* href для <a>: указывает URL ссылки.  
  <a href="https://example.com">Ссылка</a>
* src и alt для <img>: src указывает на путь к изображению, а alt — текстовое описание изображения.  
  <img src="image.jpg" alt="Изображение">
* type для <input>: указывает тип поля ввода.  
  <input type="text" placeholder="Введите имя">

Некоторые атрибуты могут быть универсальными, например, class и id, которые применяются ко многим элементам для их стилизации и управления.



## Структура HTML-документа

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Моя страница</title>

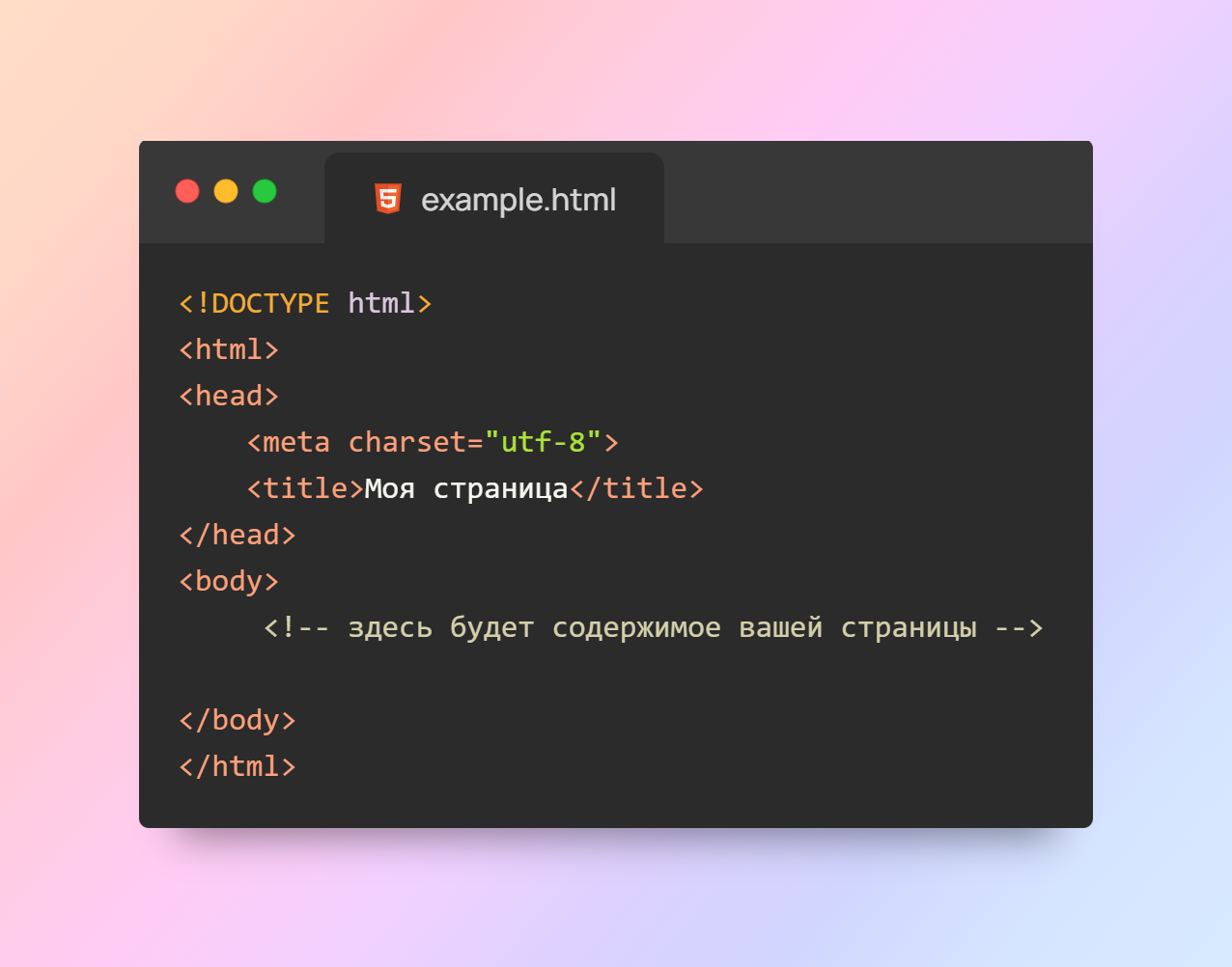
</head>

<body>

<!-- здесь будет содержимое вашей страницы -->

</body>

</html>



<!DOCTYPE html>: Объявление типа документа. Возможно вы можете увидеть более старые записи, которые относятся к предыдущим версиям HTML. В HTML5 данный формат объявления считается действующим, поэтому нужно использовать его.

<html></html>: Содержит в себе все, что есть на странице.

<head></head>: Содержит в себе содержимое, которое должно присутствовать на странице, но должно быть скрыто от пользователей. Включает в себя ключевые слова, описание страницы, которое будет показано в поисковых запросах. Здесь же происходит подключение стилей и многое другое.

<meta charset="utf-8">: Устанавливает в качестве символьной кодировки для вашего документа кодировку utf-8 , которая включает большинство символов из всех известных человечеству языков. Стоит включать в свой документ.

<title></title>: Это заголовок страницы, который появляется во вкладке браузера, также это заглавие используется при cохранении страницы в закладки или избранное.

<body></body>: Содержит весь контент, который будет виден пользователю сайта. Текст, изображения, видео, музыка, формы обратной связи и многое другое.

Чтобы было проще ориентироваться в коде, можно добавлять комментарии: они обозначаются таким синтаксисом:  
<!– комментарий →

Элементы формы и валидация

HTML предоставляет множество элементов формы, таких как <input>, <textarea>, <select>, <button>, которые помогают создавать формы для ввода данных. В HTML5 добавлены новые типы полей ввода, например, email, number, date, и встроенные средства валидации, что облегчает создание удобных и корректных форм:

<form>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<button type="submit">Отправить</button>

</form>

Встроенные медиаэлементы

В HTML5 добавлена поддержка встроенного воспроизведения медиафайлов без использования плагинов, таких как Flash. Это делается с помощью элементов <video> и <audio>:

<video controls>

<source src="movie.mp4" type="video/mp4">

Тег video не поддерживается вашим браузером.

</video>

Canvas и SVG

HTML5 предоставляет элемент <canvas>, который позволяет рисовать на веб-странице с помощью JavaScript, а также встроенные элементы для работы с векторной графикой — <svg>. Это полезно для динамической графики, анимаций, диаграмм и интерактивных элементов:

<canvas id="myCanvas"></canvas>

<script>

var canvas = document.getElementById("myCanvas");

var ctx = canvas.getContext("2d");

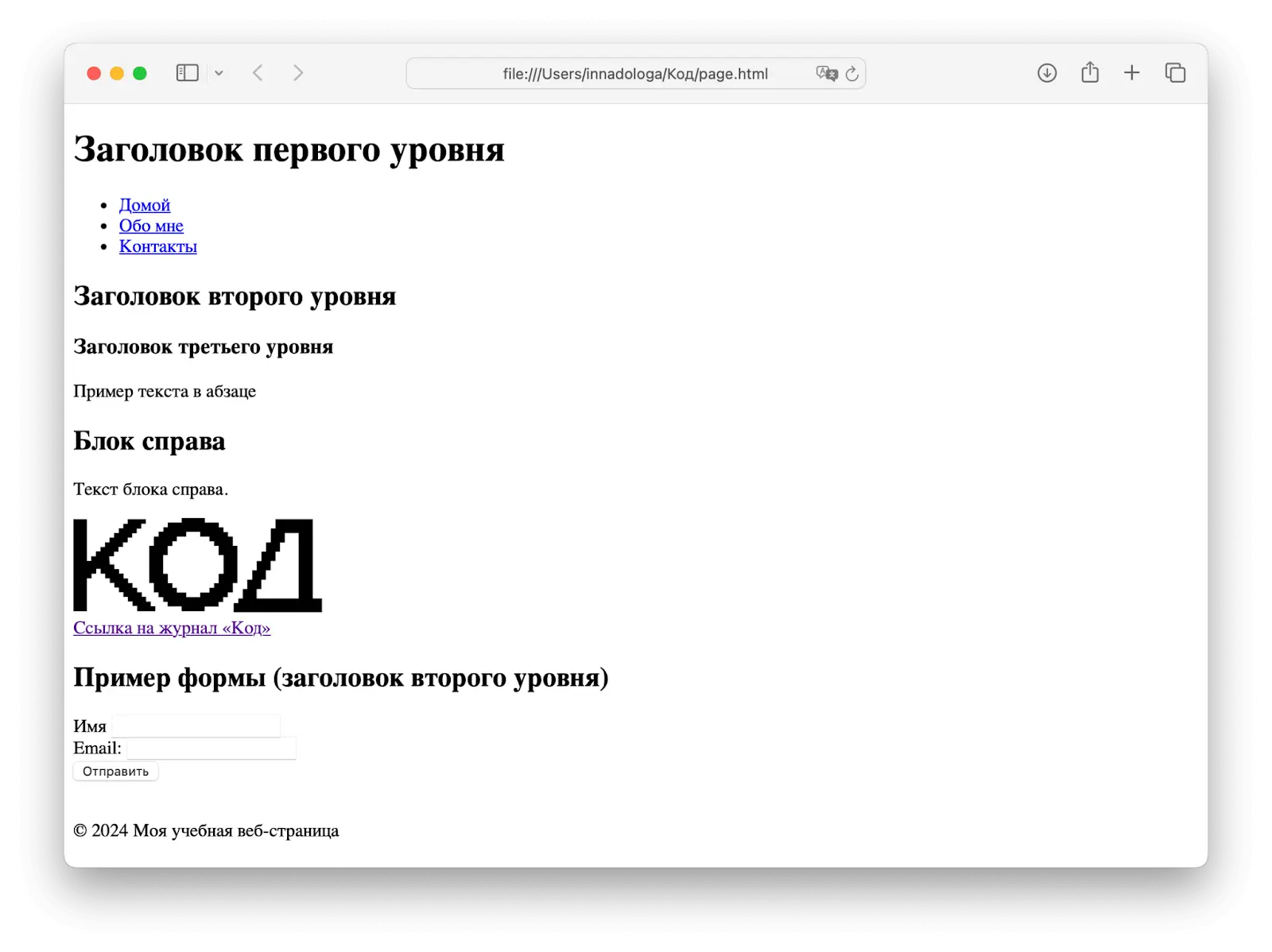
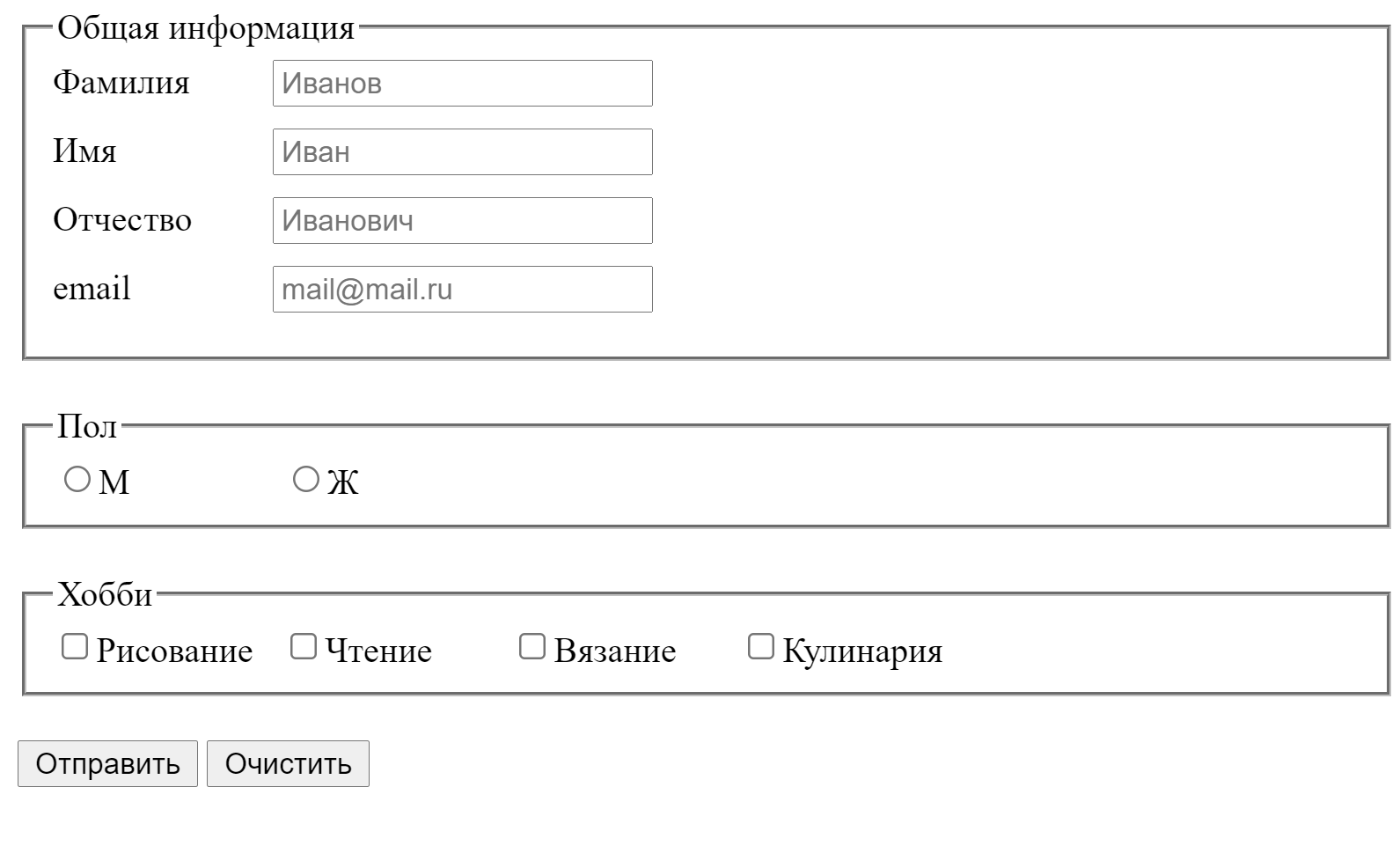
ctx.fillStyle = "#FF0000";

ctx.fillRect(0, 0, 150, 75);

</script>

Пример кода:

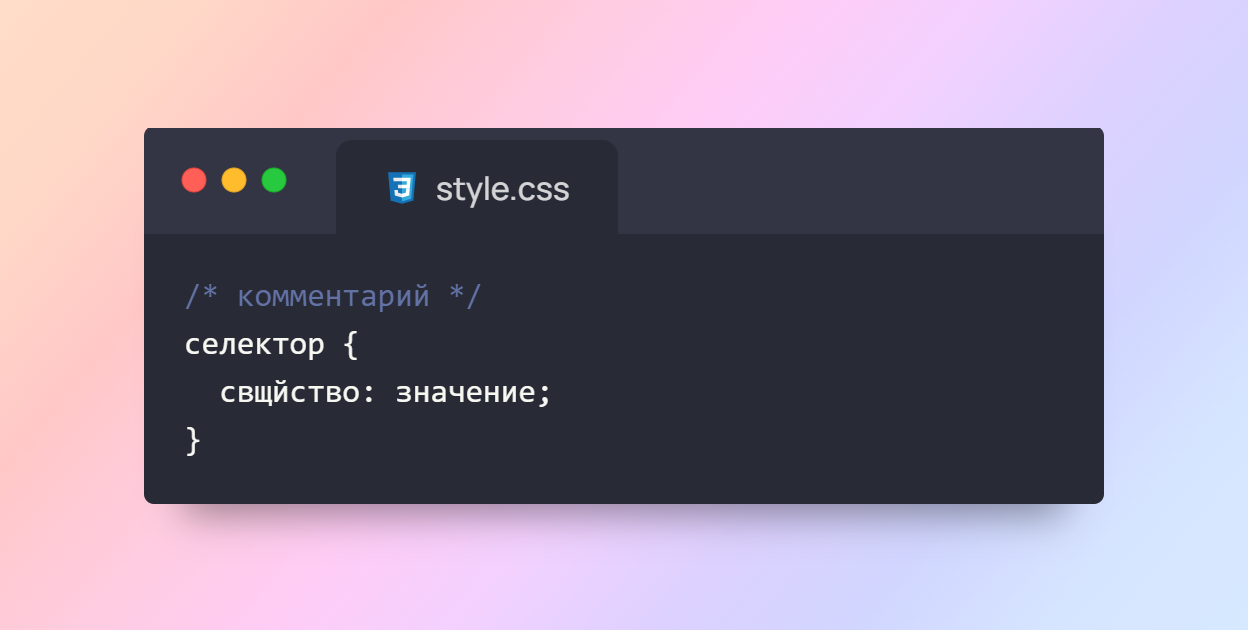


## 

## CSS - Cascading Style Sheets

CSS (Cascading Style Sheets) — это язык стилей, который используется для оформления веб-страниц. CSS позволяет изменять внешний вид элементов, такие как цвет текста, размеры, фон, выравнивание и многое другое.

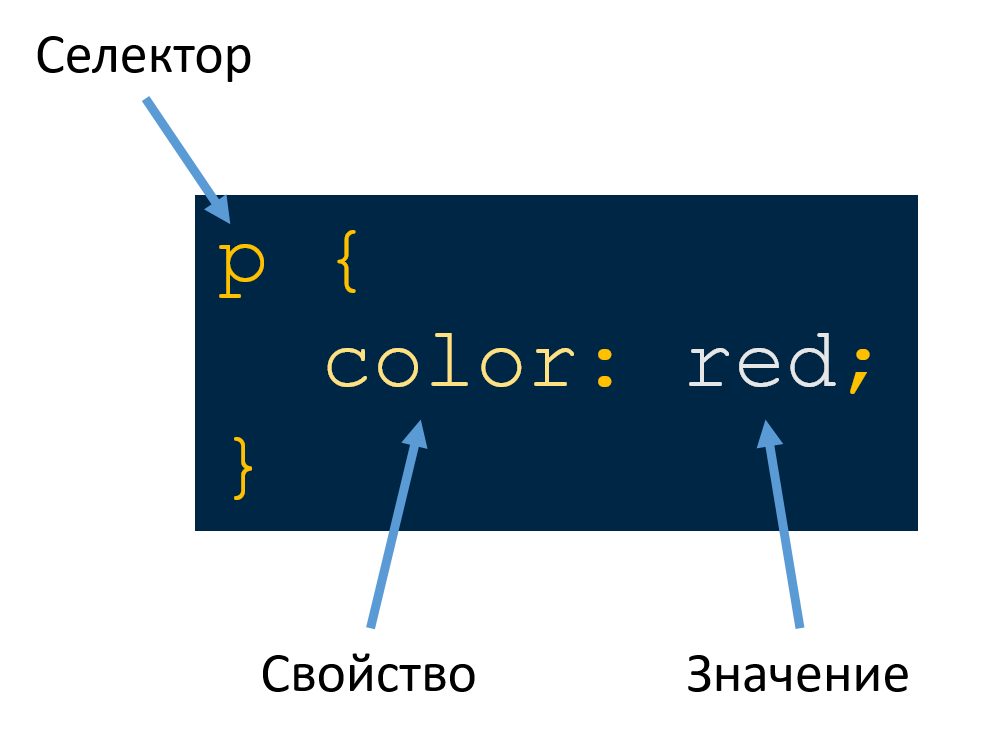


Селектор (Selector): имя HTML-элемента в начале набора правил. Он выбирает элемент(ы) для применения стиля (в данном случае, элементы p ). Для стилизации другого элемента, просто измените селектор. Можно выбирать сразу несколько селекторов, достаточно перечислить их через запятую (например, h1, h2, p, li { ... })

Свойства (Properties): cпособы, которыми вы можете стилизовать определённый HTML-элемент (в данном случае, color является свойством для элементов <p>). В CSS вы выбираете, какие свойства вы хотите затронуть в вашем правиле.

* font-family, font-size, font-weight, font-style — свойства шрифта;
* color, background-color, border-color — свойства цвета;
* text-align, text-decoration, text-transform — свойства текста;
* display, float, position, margin, padding — свойства макета;
* background-image, background-repeat, background-position — свойства фона.

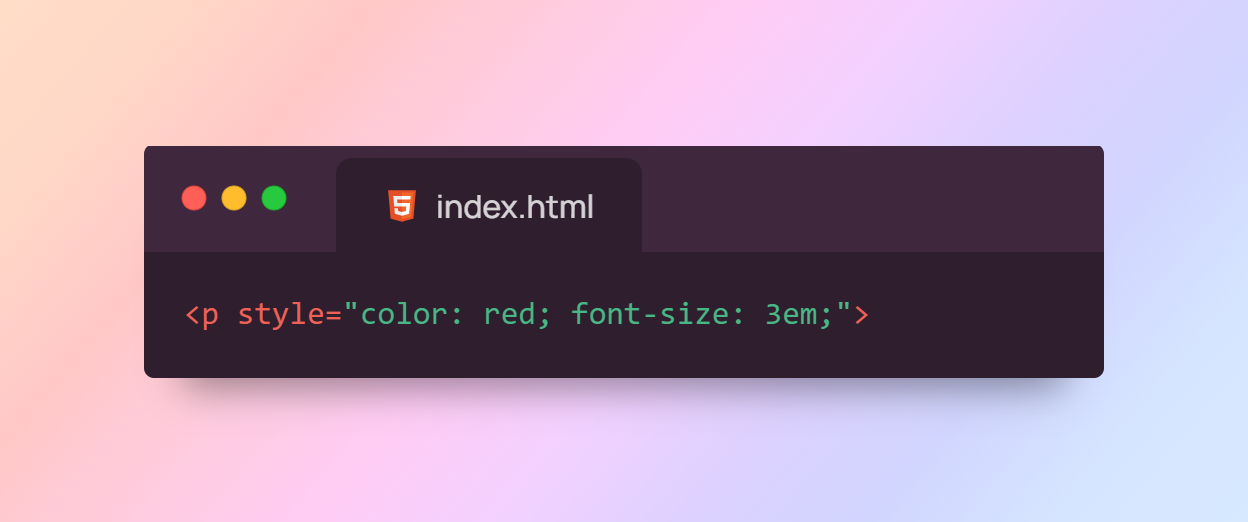
Значение свойства (Property value): справа от свойства, после двоеточия, у нас есть значение свойства, которое выбирает одно из множества возможных признаков для данного свойства (существует множество значений color, помимо red).



## **Глобальные стили** — это стили, которые применяются ко всему документу, а не к отдельным элементам или классам. Они могут задавать базовые настройки визуального оформления для всего веб-приложения или сайта, обеспечивая единообразие и легкость в управлении стилями. Глобальные стили позволяют устанавливать шрифты, цвета, отступы, размеры и другие параметры, которые будут влиять на все элементы страницы. Глобальные стили чаще всего задаются с помощью селекторов, которые охватывают весь документ, таких как: - body — устанавливает стили для всего документа. - \* (универсальный селектор) — применяется ко всем элементам на странице. - Селекторы элементов, например, h1, p, a и т.д. - Селекторы классов и идентификаторов, которые могут использоваться повсеместно.

**Как импортировать стили из файла .css**

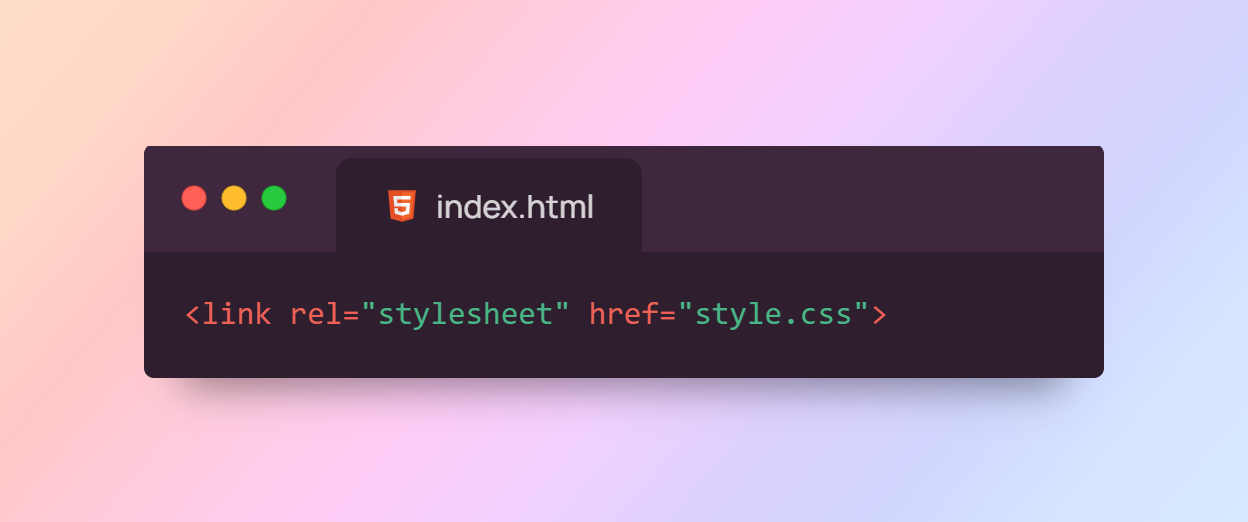
Самый простой способ передать информацию о стилях браузеру, чтобы он отобразил страницу так, как мы хотим, — это прописать стили прямо в HTML-документе.



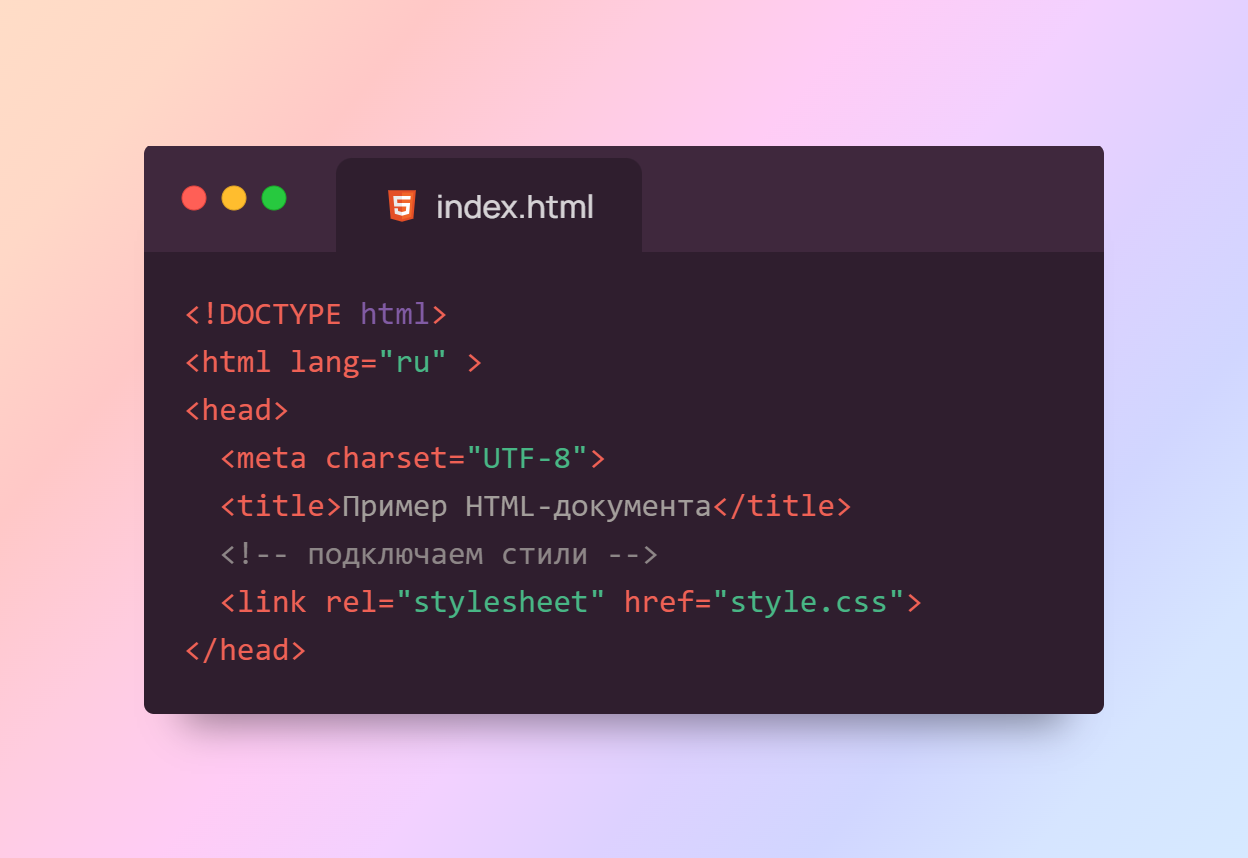
Этот код также можно внедрить HTML-страницу, обрамив тегами <style>...</style>. Все это вставляется в код страницы сразу после заголовка (тег <title>). Таким образом, страница будет теперь выглядеть следующим образом:



Но в реальном мире веб-разработки необходимо хранить код HTML, CSS и JavaScript в отдельных файлах — так проще и удобнее поддерживать проекты. Сохраним CSS-код в файл styles.css в ту же папку, в которой лежит HTML-документ, и подключим стили в него:



Эта строка сообщает браузеру, к какому файлу ему обращаться, чтобы знать, как оформлять страницу. Стили всегда подключаются в блоке «головы» страницы:



**Псевдокласс в CSS** — *это ключевое слово, добавленное к селектору,* которое определяет его особое состояние. Псевдоклассы предназначены для изменения стиля существующих элементов страницы в зависимости от их динамического состояния, например, при работе со ссылками (:link, :visited, :hover, :active, :focus). Например, [:hover](https://developer.mozilla.org/ru/docs/Web/CSS/:hover) может быть использован для изменения цвета кнопки при наведении курсора на неё.

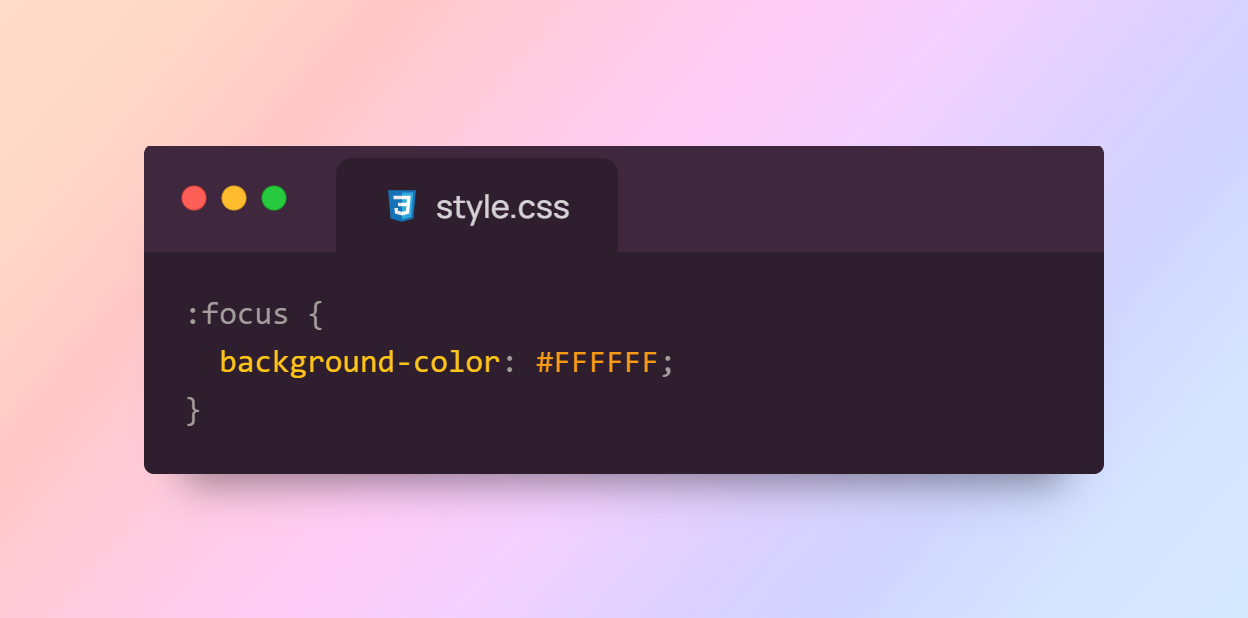
Рассмотрим CSS для подсветки строки таблицы при наведении курсора.  
В обычном состоянии цвет фона — тёмный:



При наведении курсора цвет фона изменится на голубой:



Селектор может и отсутствовать. Тогда правило применится ко всем элементам, которые могут иметь признак этого псевдокласса. Например, CSS-правило :focus {} применится к любому элементу, который будет в фокусе.

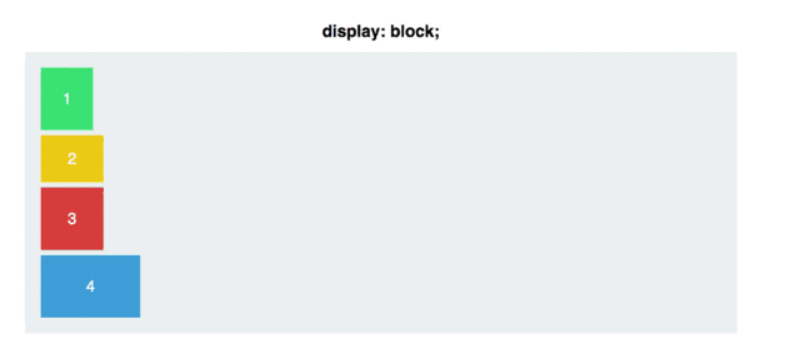


**CSS-селектор** — это часть CSS-правила, которая указывает, к каким элементам HTML будут применяться заданные стили. Если проще, они идентифицируют элементы: «такой элемент должен отображаться вот так». Простые селекторы указывают на элементы на основе их имён:

* h1, p, ul, a, img — селекторы тегов;
* .highlight, .button, .image-container — селекторы классов;
* #main-content, #header, #footer — селекторы идентификаторов;
* \* — универсальный селектор, который выбирает все элементы;
* type=”text” — селектор атрибута (в данном случае текста).

**Блочная модель и макет (подробнее про flexbox)**

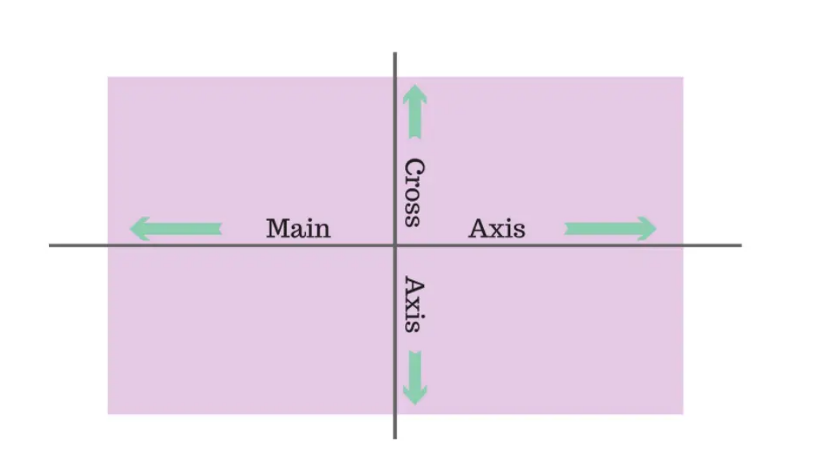
CSS Flexbox — это технология для создания сложных гибких макетов за счёт правильного размещения элементов на странице. У каждого div по умолчанию есть свойство display: block. Поэтому каждый блок занимает всю ширину строки.



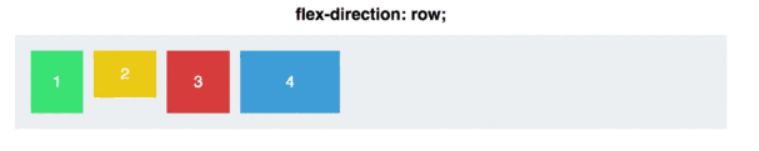
Чтобы начать работать с CSS Flexbox, нужно сделать контейнер flex контейнером. Делается это так: #container { display: flex;}.



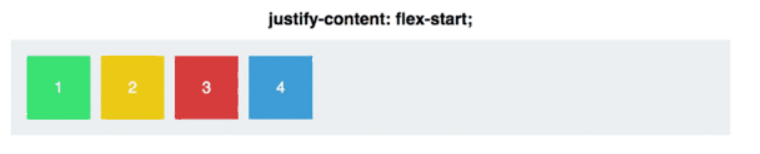
У flex-контейнера есть две оси: главная и перпендикулярная ей. По умолчанию все предметы располагаются вдоль главной оси — слева направо.



Свойство flex-direction позволяет вращать главную ось.  
#container { display: flex; flex-direction: column;}



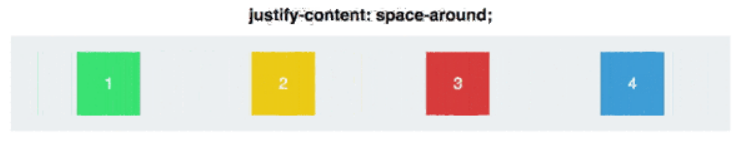
Свойство justify-content отвечает за выравнивание элементов по главной оси. Justify-content может принимать 5 значений: flex-start, flex-end, center, space-between, space-around.







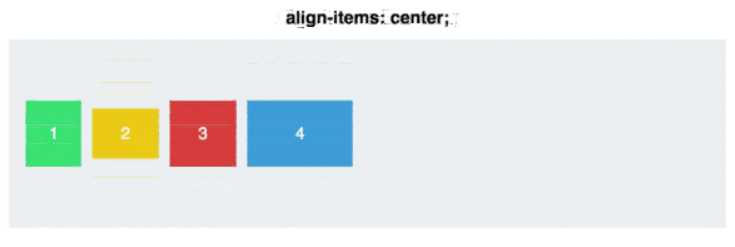




Если justify-content работает с главной осью, то align-items работает с осью, перпендикулярной главной оси. Оно также может принимать 5 значений: flex-start, flex-end, center, stretch, baseline.





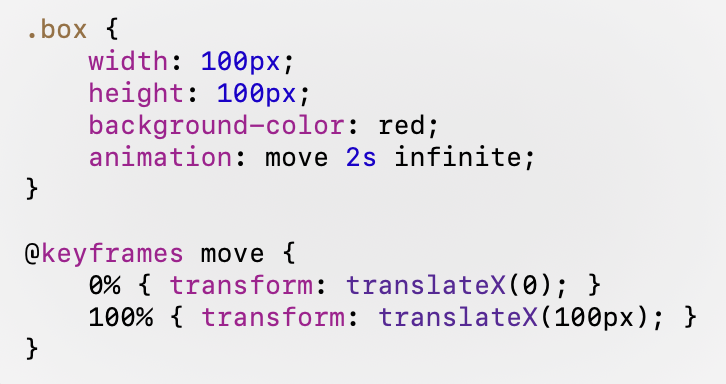






Анимации и переходы (Transitions and Animations)

CSS позволяет создавать анимации без использования JavaScript. Используя ключевые кадры с помощью @keyframes и свойства анимации, можно анимировать практически любые CSS-свойства:



Переходы позволяют создать плавные изменения свойств:



**Grid Layout**

CSS Grid Layout — это мощная система для создания сеток и сложных макетов страниц. В отличие от Flexbox, Grid позволяет работать с двумя осями — как вертикальной, так и горизонтальной:

Лабораторная работа №1.

Цель работы: создание персонального сайта-визитки (портфолио) с использованием языка гипертекстовой разметки HTML и языка стилей CSS.

Требования к работе:

1. Для HTML-файла: наличие тегов header, main и footer, блоки о себе и своих проектах, ссылки на свои социальные сети, hard и soft skills;
2. Для файла со стилями: добавление стилей через классы и глобальных стилей, использование псевдоклассов, создание блочной модели (flex-box).

Примеры работ для вдохновения:

* <https://elforastero.dev>
* <https://nikolovlazar.com/>
* <https://dribbble.com/tags/portfolio>

Полезные материалы для изучения HTML и CSS:

* <https://www.notion.so/ningaro/e4ca9669f0d2472db6fafd92dd2f68c4?v=107dadcbcbd54f60ae00107870a46aa0> (все про Frontend)
* <https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics> (гайд по HTML)
* <https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics> (гайд по CSS)
* <https://css-tricks.com/snippets/css/a-guide-to-flexbox> (гайд по flex-box)

Видео:

* <https://youtube.com/playlist?list=PLd3XpjKSrDevD7IQKoKG8nMGLl539UsVj>
* <https://youtube.com/playlist?list=PLM6XATa8CAG4uCli-pMvuvwj46UaQoqIc>
* <https://www.youtube.com/watch?v=HihYQVuH64U> (методология бэм)
* <https://www.youtube.com/watch?v=EO8hH_2OwCU> (flex-box)
* <https://tpverstak.ru/flex-cheatsheet/> (шпаргалка по флексам)