

Московский государственный технический университет  
имени Н. Э. Баумана

---

В. И. Кузнецов

## **Создание простейших HTML-страниц, Каскадные таблицы стилей CSS**

*Методические указания к выполнению  
лабораторной работы №2 по дисциплине  
«Языки интернет программирования»*

Редакция от 1.09.2025

Факультет «Информатика и системы управления»  
Кафедра «Компьютерные системы и сети»

**Кузнецов В.И.**

Создание простейших HTML-страниц, устаревшие элементы разметки. Валидаторы кода. Каскадные таблицы стилей CSS : методические указания к выполнению практикума № 1 и лабораторной работы № 1 по дисциплине «Языки интернет- программирования» / В. И. Кузнецов.

Приведены основные теоретические сведения о языке разметки HTML и о таблицах стилей CSS, необходимые для выполнения лабораторной работы и практикума по созданию HTML-страниц с помощью стилей CSS. Рассмотрены примеры. Также приведена литература по курсу и даны ссылки на интернет- источники.

Для студентов МГТУ им. Н.Э. Баумана, обучающихся по направлению «Информатика и вычислительная техника».

## **Лабораторная работа №2**

### **Создание простейших HTML-СТРАНИЦ,, Каскадные таблицы стилей CSS , блочная верстка, страница с формами, BOOTSTRAP**

*Цель работы* — знакомство с языками HTML и CSS, а также получение практических навыков применения каскадных таблиц стилей для формирования отображения страниц HTML.

### **ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

HTML (HyperText Markup Language) — язык гипертекстовой разметки, являющийся основой веб-документов. Он описывает структуру веб-страниц с использованием тегов, которые указывает браузеру, как отображать текст, изображения, ссылки и другие элементы.

CSS (Cascading Style Sheets) — каскадные таблицы стилей, которые определяют внешний вид HTML-документа. Вместе они позволяют разделять структуру и оформление, что делает разработку сайтов удобнее и эффективнее.

Первая версия HTML была предложена Тимом Бернерсом-Ли в 1991 году. Она содержала всего несколько тегов для форматирования текста и ссылок. С развитием интернета язык разметки постепенно расширялся, и в настоящее время актуальной является версия HTML5.

CSS появился в середине 1990-х годов. До его внедрения внешний вид веб-страниц определялся в основном с помощью тегов HTML. Это делало код перегруженным и неудобным для поддержки. CSS позволил отделить содержание от оформления, что стало стандартом современной веб-разработки.

# Подробный разбор основных тегов HTML

## Различия между типами тегов HTML

Все теги HTML можно разделить на несколько категорий по их назначению и особенностям использования:

### 1. Парные и одинарные теги

Парные теги имеют открывающий и закрывающий элемент, например: `<p>...</p>`, `<h1>...</h1>`. Они обрамляют содержимое. Одинарные теги не имеют закрывающего элемента, например: `<img>`, `<br>`, `<hr>`

### 2. Блочные и строчные теги

Блочные теги формируют отдельный блок на странице, занимают всю доступную ширину и начинаются с новой строки. Примеры: `<div>`, `<p>`, `<h1>`. Строчные теги не создают нового блока и располагаются в строке вместе с текстом. Примеры: `<span>`, `<a>`, `<strong>`, `<em>`.

### 3. Семантические и несемантические теги

Семантические теги описывают смысловое назначение содержимого. Например, `<header>`, `<footer>`, `<section>`, `<article>`, `<nav>`.

Несемантические теги не несут смысловой нагрузки, а лишь задают контейнер для элементов. Примеры: `<div>`, `<span>`.

### 4. Глобальные и специализированные теги

Глобальные теги используются во множестве случаев и могут содержать любые элементы: `<div>`, `<p>`, `<span>`. Специализированные теги выполняют конкретные задачи: `<form>` для форм, `<table>` для таблиц, `<img>` для изображений, `<video>` для видео.

Понимание различий между типами тегов помогает правильно строить структуру веб-страниц, улучшать читаемость кода и создавать более семантические и доступные сайты.

## Состав HTML-тега

### 1. Открывающий тег

Это часть, с которой начинается элемент. Он заключён в угловые скобки:

`<p>` Здесь: - «<» и «>» — границы тега - p — имя тега (в данном случае параграф)

### 2. Атрибуты (дополнительные параметры)

Атрибуты уточняют поведение или внешний вид элемента. Они указываются внутри открывающего тега:

`<a href="https://example.com" target="_blank">`

Здесь:

- href="<https://example.com>

HTML состоит из множества тегов, которые можно разделить на классов. Ниже приведены основные группы тегов с примерами и пояснениями.

Тег	Назначение	Пример
<code>&lt;h1&gt;</code> – <code>&lt;h6&gt;</code>	Заголовки различных уровней	<code>&lt;h1&gt;Заголовок&lt;/h1&gt;</code>
<code>&lt;p&gt;</code>	Абзац текста	<code>&lt;p&gt;Пример текста&lt;/p&gt;</code>
<code>&lt;a&gt;</code>	Гиперссылка	<code>&lt;a href='https://example.com '&gt;Сайт&lt;/a&gt;</code>
<code>&lt;img&gt;</code>	Изображение	<code>&lt;img src='image.jpg' alt='Описание'&gt;</code>

<code>&lt;ul&gt;, &lt;ol&gt;, &lt;li&gt;</code>	Списки (маркированный, нумерованный)	<code>&lt;ul&gt;&lt;li&gt;Элемент&lt;/li&gt;&lt;/ul&gt;</code>
<code>&lt;table&gt;</code>	Таблица	<code>&lt;table&gt;&lt;tr&gt;&lt;td&gt;Ячейка&lt;/td&gt;&lt;/tr&gt;&lt;/table&gt;</code>
<code>&lt;form&gt;</code>	Форма ввода	<code>&lt;form&gt;&lt;input type='text'&gt;&lt;/form&gt;</code>
<code>&lt;header&gt; , &lt;footer&gt;</code>	Семантические блоки HTML5	<code>&lt;header&gt;Шапка сайта&lt;/header&gt;</code>
<code>&lt;div&gt;</code>	Элемент разделения контента, контейнер блочный	<code>&lt;div&gt;      &lt;p&gt;Любой тип контента.     &lt;/p&gt;  &lt;/div&gt;</code>
<code>&lt;span&gt;</code>	Элемент разделения контента, контейнер строчный	<code>&lt;span class="ingredient"&gt; Базилик &lt;/span&gt;</code>

## Пример HTML-кода

```
<!DOCTYPE html>
<html>
  <head>
    <title>Пример страницы</title>
  </head>
  <body>
    <header>
      <h1>Добро пожаловать!</h1>
    </header>
    <section>
      <p>Это пример параграфа.</p>
      <a href='https://example.com'>Перейти на сайт</a>
    </section>
    <footer>
      <p>Все права защищены.</p>
    </footer>
  </body>
</html>
```

## Виды верстки веб-страниц

Верстка веб-страниц это процесс создание и верного расположения элементов страницы при её отображении. Существует несколько подходов к верстке веб-страниц. Каждый из них имеет свои особенности, преимущества и недостатки.

Вид верстки	Описание	Особенности
Табличная верстка	Использование таблиц (<table>) для расположения элементов на странице.	Устаревший метод; сложность адаптации под мобильные устройства; плохая семантика.

Блочная верстка	Построение страницы с помощью тегов <div>, <section>, <header>, <footer> и CSS.	Гибкий и современный подход; позволяет легко изменять дизайн.
Резиновая верстка	Размеры элементов задаются в относительных единицах (%), а не в пикселях.	Страница подстраивается под ширину окна браузера; может ломать дизайн.
Адаптивная верстка	Использование медиа-запросов (media queries) в CSS для изменения стилей в зависимости от устройства.	Хорошо работает на мобильных устройствах; требует дополнительной проработки дизайна.
Отзывчивая (responsive) верстка	Комбинация адаптивной и резиновой верстки, когда страница подстраивается под разные размеры экрана.	Современный стандарт; оптимальна для всех устройств; требует грамотной реализации.
Мобильная верстка (mobile first)	Сначала создается дизайн для мобильных устройств, а затем для больших	Удобно для пользователей смартфонов; соответствует современным



	экранов.	трендам.
--	----------	----------

Понимание различных видов верстки позволяет выбирать оптимальный подход при разработке сайтов, ориентируясь на целевую аудиторию и современные стандарты веб-дизайна.

## **CSS и его возможности**

CSS определяет оформление элементов: цвета, шрифты, размеры, расположение на странице. Основные свойства включают:

- color — цвет текста
- background-color — цвет фона
- font-size — размер шрифта
- margin и padding — отступы
- border — границы элементов
- display, flex, grid — способы расположения блоков

## **Способы подключения CSS**

1. Встроенные стили (атрибут style внутри тега):

```
<p style='color:red'>Текст</p>
```

2. Внутренние стили (раздел <style> в документе):

```
<style> p { color: blue; } </style>
```

3. Внешние стили (отдельный .css файл):

```
<link rel='stylesheet' href='styles.css'>
```

Пример CSS-кода:

```
body {
  font-family: Arial, sans-serif;
  background-color: #f0f0f0;
  color: #333;
}
```

```
h1 {
  color: darkblue;
  text-align: center;
}
```

```
}
```

```
p {  
  line-height: 1.5;  
}
```

## **Способы подключения CSS стилей**

В веб-разработке существует несколько способов подключения CSS стилей к HTML-документу. Рассмотрим основные:

### **Встроенные стили (Inline CSS)**

Стили прописываются непосредственно внутри HTML-элемента с помощью атрибута style.

Пример:

```
<p style="color: red; font-size: 20px;">Привет, мир!</p>
```

### **Внутренние стили (Internal CSS)**

Стили размещаются в блоке <style> внутри секции <head> документа.

Пример:

```
<head>  
  <style>  
    p {  
      color: blue;  
      font-size: 18px;  
    }  
  </style>  
</head>
```

### **Внешние стили (External CSS)**

Стили подключаются отдельным CSS-файлом через тег <link>. Могут использоваться как локальные файлы так и файлы с удаленных серверов.

Пример:

```
<head>  
  <link rel="stylesheet" href="styles.css">  
</head>
```

Наиболее предпочтительным способом является использование внешнего CSS-файла, так как это упрощает поддержку и масштабируемость кода.

### **Современные возможности CSS**

Современный CSS позволяет использовать адаптивную верстку (media queries), Flexbox и CSS Grid для построения сложных макетов, а также анимации и переходы. Это делает страницы не только красивыми, но и удобными для пользователей на разных устройствах.

```
.container {  
  display: flex;  
  justify-content: space-between;  
}  
  
.grid {  
  display: grid;  
  grid-template-columns: 1fr 1fr 1fr;  
  gap: 10px;  
}
```

### **Адаптивная верстка**

Адаптивная верстка позволяет сайту подстраиваться под размеры экрана. Для этого используются media queries. Например:

```
@media (max-width: 600px) {  
  body {  
    background-color: lightgray;  
  }  
}
```

### **Анимации в CSS**

CSS также позволяет создавать анимации и плавные переходы. Это улучшает пользовательский опыт и делает сайты более современными.

Пример:

```
button {  
    transition: background-color 0.3s ease;  
}  
  
button:hover {  
    background-color: orange;  
}
```

## **Bootstrap CSS**

Bootstrap — это один из самых популярных CSS-фреймворков, используемых для создания адаптивных и современных веб-сайтов. Он предоставляет готовые стили, сетку и компоненты, что значительно ускоряет процесс разработки.

### **Основные особенности Bootstrap CSS**

#### **1. Сетка (Grid system)**

Bootstrap использует систему из 12 колонок для построения адаптивных макетов.

Пример:

```
<div class="container">  
    <div class="row">  
        <div class="col-6">Левая колонка</div>  
        <div class="col-6">Правая колонка</div>  
    </div>  
</div>
```

#### **2. Готовые стили**

Кнопки, формы, заголовки и другие элементы уже имеют встроенные стили.

Пример:

```
<button class="btn btn-primary">Кнопка</button>  
<button class="btn btn-success">Успех</button>
```

#### **3. Адаптивность**

Сайты автоматически подстраиваются под экраны разных размеров. Можно скрывать или показывать элементы на определённых устройствах.

Пример:

```
<div class="d-none d-md-block">Видно только на больших  
экранах</div>
```

#### 4. Компоненты

Bootstrap предоставляет десятки готовых UI-элементов: навигационные панели, модальные окна, карусели, алерты.

Пример:

```
<div class="alert alert-warning" role="alert">  
  Это предупреждение!  
</div>
```

### Подключение Bootstrap

Подключить Bootstrap можно через CDN(удаленный сервер) или локально.

Пример (через CDN):

```
<head>  
  <link  
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstr  
    trap.min.css" rel="stylesheet">  
  <script  
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstr  
    ap.bundle.min.js"></script>  
</head>
```

### Заключение

HTML и CSS — фундаментальные технологии веб-разработки. HTML задает структуру документа, а CSS отвечает за внешний вид. Вместе они позволяют создавать удобные и современные сайты. Понимание этих технологий открывает путь к изучению JavaScript и разработке полноценных веб-приложений. Владение HTML и CSS — первый шаг в мир программирования для интернета.

## Практическая часть

Существующие редакторы для веб-страниц можно разделить на несколько типов. Они помогают разработчикам писать HTML, CSS и JavaScript-код, а также сразу видеть результат.

Текстовые редакторы общего назначения  
Это простые программы для редактирования текста, которые можно использовать для написания кода:

- Блокнот (Notepad) — стандартный редактор Windows, подходит для самых простых задач.
- Notepad++ — популярный бесплатный редактор с подсветкой синтаксиса и плагинами.
- Sublime Text — лёгкий, быстрый редактор, поддерживает плагины, подсветку кода.
- Vim или Emacs — мощные консольные редакторы, часто используются опытными программистами.

Специализированные редакторы для веб-разработки  
Эти редакторы созданы специально для работы с HTML, CSS, JavaScript:

- Visual Studio Code (VS Code) — один из самых популярных редакторов. Поддерживает расширения, Git, отладку, встроенный терминал.
- Atom — редактор с открытым исходным кодом от GitHub, поддерживает плагины и настройку интерфейса. Более не поддерживается но до сих пор используется большим количеством разработчиков
- Brackets — создан для веб-разработчиков, есть функция «живого просмотра» (live preview).

Интегрированные среды разработки (IDE) Это более мощные инструменты, которые подходят не только для HTML, но и для крупных проектов:

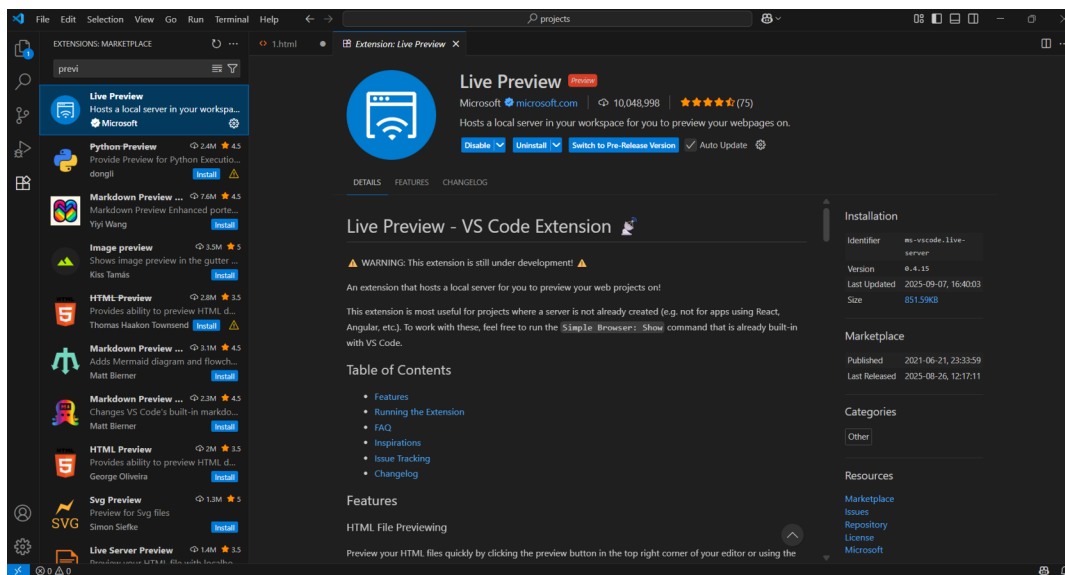
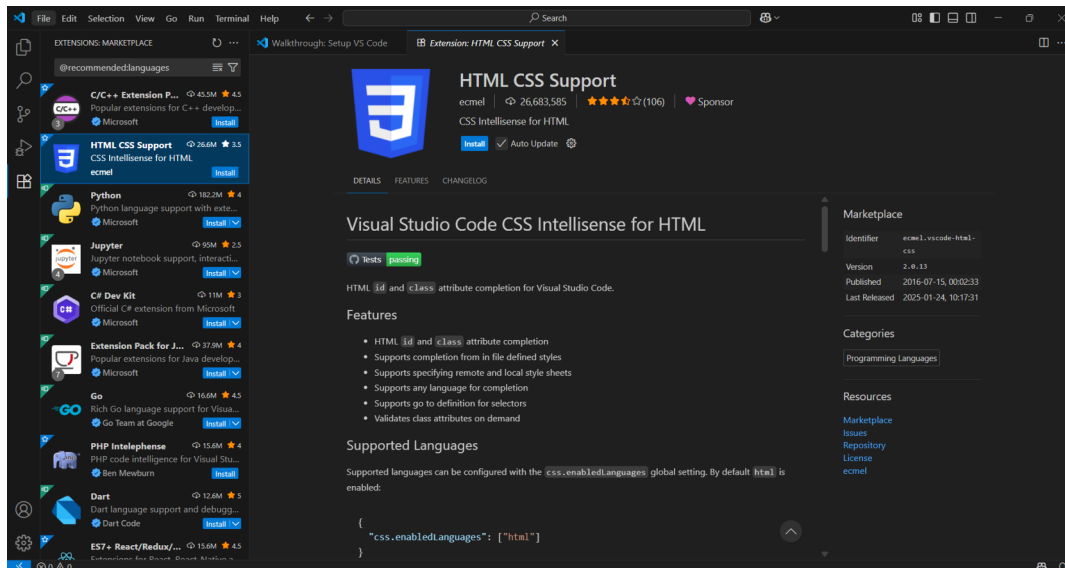
- WebStorm — IDE от JetBrains, создана специально для JavaScript и веб-разработки.
- Eclipse — универсальная IDE, поддерживает плагины для веб-технологий.

- Adobe Dreamweaver — редактор с визуальным интерфейсом (WYSIWYG), можно редактировать код и сразу видеть результат.

Для выполнения рекомендуется использовать Visual Studio Code. Это доступный открытый инструмент с большим количеством полезных плагинов.

### [Ссылка на скачивание](#)

Для дальнейшей работы необходимо установить расширение HTML CSS Support и Live Preview для работы.



## ПОРЯДОК ВЫПОЛНЕНИЯ ЗАДАНИЯ

1. Создайте пустую HTML страницы из примера html кода. Откройте пример с помощью браузера и в панели разработчика продемонстрируйте код.
2. Используя личный кабинет студента получите информацию о расписании любой группы кроме своей и с помощью тегов для создания таблиц сверстайте расписание по числителям выбранной группы. Полученный результат так же проинспектируйте в браузере. Пример:

### ИУ6-32Б

	Понедельник	Вторник	Среда	Четверг	Пятница	Суббота
1 пара			Физика	Теория вероятностей и математическая статистика		
2 пара	Теория вероятностей и математическая статистика		Электротехника		Теория вероятностей и математическая статистика	Языки интернет-программирования
3 пара			Иностранный язык		Элективный курс по физической культуре и спорту	
4 пара	Правоведение	Физика	Элективный курс по физической культуре и спорту	Электротехника		
5 пара					Базы данных	
6 пара						

3. К полученному результату из предыдущего пункта примените различные стили CSS как в строчном режиме так и внутренние. Обязательно необходимо применить выравнивание текста, изменение цвета текста, толщины границ таблицы и отступов от края контента и границ.
4. Создайте на полученной странице форму изменения расписания содержащую поле ввода текста, радиокнопку, чекбокс и селект вида:



Название предмета:

День недели  
☒ Понедельник ☐ Вторник ☐ Среда ☐ Четверг ☐ Пятница ☐ Суббота

Пара  
День недели ☐ 1 пара ☐ 2 пара ☐ 3 пара ☐ 4 пара ☐ 5 пара ☐ 6 пара

Тип пары -- тип пары --

Добавьте в полученный из предыдущего пункта результат после Выберите и добавьте любую готовую тему bootstrap с ресурса [Bootstrap themes](#) как внешний источник через скачивание файла или внешний ресурс и примените стили к созданной форме.

## СОДЕРЖАНИЕ ОТЧЕТА

Отчет должен включать:

1. ФИО студента и номер группы;
2. наименование лабораторной работы;
3. названия выполненных пунктов и тексты реализованных HTML и CSS-разметки с указанием имен файлов.

Отчет предоставляется в электронном виде в формате pdf.

Зачет ставится при условии выполнения всех пунктов задания, демонстрации работы программы и при наличии отчета и устных ответов на контрольные вопросы.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Каков основной принцип формирования разметки в HTML?
2. Какие виды верстки существуют и наиболее часто используются?
3. Каково назначение элементов div и span?
4. Приведите примеры элементов семантической разметки.
5. Приведите способы подключения CSS стилей