МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Кафедра автоматизированных систем управления

Отчёт по расчётно-графической работе

Дисциплина: Системы управления веб-контентом

Задание выполнил:

Нуриева Ольга Михайловна

АВТФ, АПМ-24



#### Преподаватель:

Хворостов Владимир Александрович

Новосибирск, 2024

**Введение**

В рамках курса веб-разработки были выполнены пять лабораторных работ, которые позволили изучить основы HTML, CSS и принципы создания веб-страниц. Эти работы дали понимание того, как устроена архитектура веб-разработки, какие инструменты используются для создания сайтов и как применять полученные знания на практике. Эти работы позволили мне углубить свои знания в области frontend-разработки, изучить современные библиотеки и подходы к созданию интерактивных веб-приложений.

В данном работе собрана вся информация и выводы, которые были получены мной в процессе выполнения лабораторных работ, а также примеры того, как эти знания могут быть полезны в дальнейшем, включая написание магистерской диссертации.

**Лабораторная работа № 1: Основы HTML**

Первая лабораторная работа была посвящена изучению базовых принципов HTML. В ходе работы я научился создавать структурированные HTML-документы, используя такие теги, как doctype, html, head, title, meta и body. Также я освоил использование тегов для разметки текста (p, ul, ol, li, h1-h6, strong, b, em, i, hr, pre, br), вставки изображений (img) и создания форм и таблиц (form, input, select, textarea, table, tr, td).

#### **Базовые теги для структуры документа**

Эти теги необходимы для создания корректной структуры HTML-документа:

**<!DOCTYPE html>**:  
 Указывает браузеру, что документ соответствует стандарту HTML5. Это обязательный тег, который должен быть первым в документе.

**<html>**:  
 Корневой элемент документа. Внутри него находятся все остальные элементы.

**<head>**:  
 Содержит метаинформацию о документе, такую как заголовок, кодировка, ссылки на стили и скрипты. Эта информация не отображается на странице, но важна для браузеров и поисковых систем.

**<title>**:  
 Заголовок страницы, который отображается во вкладке браузера. Также используется поисковыми системами для отображения в результатах поиска.

**<meta>**:  
 Используется для указания метаданных, таких как кодировка (<meta charset="UTF-8">), описание страницы и ключевые слова.

**<body>**:  
 Основная часть документа, где размещается контент, который отображается на странице.

#### **Таблица 1 - Теги для разметки текста**

|  |  |
| --- | --- |
| **Тег** | **Значение** |
| <p> | Параграф |
| <ul> | Маркированный список |
| <ol> | Нумированный список |
| <li> | Элемент списка |
| <h1>**-**<h6> | Заголовки разного уровня |
| <strong> | Важность текста |
| <b> | Жирный текст |
| <em> | Указывает на акцент |
| <i> | Курсив |
| <hr> | Горизонтальная линия |
| <pre> | Форматирование текста, включая пробелы и переносы строк |
| <br> | Перенос строки |
| <img> | Вставка изображений: атрибуты src (путь к изображению) и alt (альтернативный текст) |
| <form> | Контейнер для формы |
| <input> | Поле ввода |
| <select>**и**<option> | Выпадающий список |
| <table>  <tr>  <td>  <th> | Создания таблиц  Строки  Заголовок  Данные |
| <a> | Гиперссылка. Атрибут href указывает URL-адрес |

**Выводы:**

HTML был разработан как язык разметки, который позволяет структурировать информацию и делать её доступной для браузеров. Его простота и гибкость сделали его стандартом для создания веб-страниц.

Каждый тег выполняет свою роль: одни отвечают за текстовое содержание, другие — за изображения, формы или таблицы. Правильное использование тегов делает страницу семантически корректной. Эти знания являются основой для дальнейшего изучения веб-разработки, включая CSS для стилизации и JavaScript для добавления интерактивности.Гиперссылки (<a>) связывают страницы между собой, что является основой навигации в вебе.

**Лабораторная работа № 2: Основы CSS**

Вторая лабораторная работа была посвящена изучению CSS. В ней изучается три способа подключения стилей: через внешний файл (<link>), внутри HTML-документа (<style>) и внутри тега (атрибут style), а также синтаксис CSS, привязка стилей к тегам, классам и идентификаторам.

### Каскадированием и наследованием в CSS

Каскадирование и наследование — это два ключевых механизма в CSS, которые определяют, как стили применяются к элементам на веб-странице. Хотя они оба влияют на внешний вид элементов, их принципы работы и назначение различаются. Давайте разберем их подробнее.

Каскадирование — это механизм, который определяет, какие стили будут применены к элементу, если к нему одновременно применяются несколько правил CSS.

Специфичность определяет, какой селектор "сильнее". Она рассчитывается по формуле:

* + !important — наивысший приоритет.
  + Inline-стили — 1000.
  + ID — 100.
  + Классы, псевдоклассы, атрибуты — 10.
  + Теги и псевдоэлементы — 1.

Если специфичность одинакова, применяется последнее объявленное правило.

Наследование — это механизм, при котором дочерние элементы наследуют стили от своих родительских элементов. Однако не все свойства наследуются. Например:

* Наследуемые свойства: color, font-family, font-size, line-height.
* Ненаследуемые свойства: margin, padding, border, background.

### Таблица 2 - Различия между каскадированием и наследованием

|  |  |  |
| --- | --- | --- |
| **Аспект** | **Каскадирование** | **Наследование** |
| **Что определяет** | Какие стили применяются к элементу | Какие стили наследуются дочерними элементами |
| **Зависит от** | Источника стилей, специфичности, порядка | Родительских элементов |
| **Пример** | Приоритет inline-стилей над внешними | Наследование цвета текста от родителя |
| **Применение** | Ко всем элементам | Только к наследуемым свойствам |

**Выводы:**

CSS был создан для управления внешним видом веб-страниц. Его архитектура основана на принципах каскадирования и наследования, что позволяет разработчикам эффективно управлять стилями и избегать дублирования кода. Разделение HTML и CSS способствует поддержанию чистоты кода и упрощает процесс разработки.

**Лабораторная работа № 3: Блочная модель и позиционирование**

Третья лабораторная работа была посвящена изучению блочной модели CSS и свойств, таких как width, height, position, display, float, clear, margin, padding и border. Также были изучены различия между absolute, relative и статическим позиционированием, а также между inline, block и inline-block элементами. Эти свойства являются основой для создания макетов и управления внешним видом элементов.

### Таблица 3 - CSS-свойства

|  |  |
| --- | --- |
| Свойство | Применение |
| width | Ширина элемента |
| height | Высота элемента |
| position | Static (элемент располагается в нормальном потоке документа)Relative (позиционируется относительно своего обычного положения) - можно использовать top, right, bottom, left для смещенияAbsolute (позиционируется относительно ближайшего позиционированного родителя или относительно <body>)Fixed (фиксируется относительно окна браузера)Sticky(ведет себя как relative, пока не достигнет определенной точки, после чего становится fixed) |
| display | Block - Элемент занимает всю доступную ширину и начинается с новой строкиInline - Элемент занимает только необходимое пространство и не начинает новую строкуinline-block - Элемент ведет себя как inline, но может иметь ширину, высоту и отступы, как block |
| float | Элемент "плавает" влево или вправо, а текст или другие элементы обтекают его |
| clear | Останавливает обтекание элемента. |
| margin | Внешние отступы. Определяют расстояние между элементами |
| padding | Внутренние отступы. Определяют расстояние между содержимым элемента и его границей |
| border | Граница элемента. Можно задать толщину, стиль и цвет. |

**Выводы:**

Блочная модель и позиционирование были разработаны для управления расположением элементов на странице. Эти механизмы позволяют создавать сложные макеты, адаптивные интерфейсы и обеспечивают кросс-браузерную совместимость. Современные технологии, такие как Flexbox и Grid, упрощают процесс верстки, но основы блочной модели остаются актуальными.

**Лабораторная работа № 4: Фоны и Flexbox**

Четвертая лабораторная работа включала изучение CSS-свойств для работы с фонами (background-color, background-image, background-position, background) и создание интерактивных элементов, таких как ссылки с изменением цвета при наведении и блоки с изменением фона. Также изучена работа Flexbox, который позволяет создавать гибкие и адаптивные макеты.

### CSS-свойств для работы с фоном

* background-color- задает цвет фона элемента.
* background-image - устанавливает фоновое изображение.
* background-position - определяет начальное положение фонового изображения.
* Псевдокласс :hover используется для создания интерактивных элементов, таких как ссылки и блоки, которые меняют цвет при наведении.
* Использование transition делает изменения плавными и визуально привлекательными.

**Выводы:**

Flexbox — это мощный инструмент для создания адаптивных макетов, который упрощает выравнивание и распределение элементов. Использование фоновых свойств позволяет создавать визуально привлекательные интерфейсы. Псевдоклассы, такие как :hover, добавляют интерактивность веб-страницам.

Flexbox был разработан для решения проблем, связанных с созданием адаптивных макетов. Его гибкость и простота сделали его стандартом для верстки. Архитектура веб-разработки постоянно развивается, и такие инструменты, как Flexbox и Grid, упрощают процесс создания современных интерфейсов.

**Лабораторная работа № 5. Основы JS и git**

В рамках пятой лабораторной были выполнены две ключевые лабораторные работы: создание веб-страницы со слайдером изображений и разработка калькулятора вкладов для банка.

В задании по созданию слайдера изображений была использована библиотека **Swiper**. Эта работа позволяет изучить основы работы с библиотеками, а также понять, как современные инструменты упрощают процесс разработки веб-приложений.

В процессе выполнения работы была подключена библиотека Swiper, создана структура слайдера с использованием HTML и CSS и настроен слайдер с помощью JavaScript с добавлением навигационные кнопки, пагинации и поддержки клавиатуры и колеса мыши.

#### **Выводы**

1. **Использование библиотек:** Библиотеки, такие как Swiper, значительно упрощают процесс создания сложных компонентов, таких как слайдеры. Они предоставляют готовые решения, которые можно легко настроить под свои нужды.
2. **Адаптивность:** Swiper автоматически адаптируется под разные устройства, что делает его удобным для создания современных веб-приложений.
3. **Интерактивность:** Добавление навигационных кнопок и пагинации улучшает пользовательский опыт, делая слайдер более удобным в использовании.
4. **Экономия времени:** Использование готовых решений позволяет сосредоточиться на логике приложения, не тратя время на написание кода с нуля.

Целью работы было создание калькулятора вкладов, который позволяет пользователю выбрать вид вклада, срок и сумму, а затем рассчитать итоговую сумму с учетом процентной ставки. В процессе выполнения была создана структура формы с использованием HTML, стилизована форма с помощью CSS, реализована логика расчета с использованием JavaScript и добавлена валидация данных и вывод результата.

#### **Выводы**

1. **Использование JavaScript**: JavaScript позволяет создавать динамические и интерактивные веб-приложения. В этой работе он используется для обработки выбора пользователя, расчета итоговой суммы и вывода результата.
2. **Валидация данных**: Валидация данных на стороне клиента улучшает пользовательский опыт и предотвращает ошибки. Была добавлена проверка на заполнение всех полей и корректность введенной суммы.
3. **Динамическое изменение контента**: При выборе вида вклада динамически изменяется список доступных сроков. Это делает интерфейс более удобным и интуитивно понятным.
4. **Разделение ответственности**: HTML отвечает за структуру, CSS — за стили, а JavaScript — за поведение.

**Заключение**

Курс веб-разработки дает фундаментальные знания о том, как устроены веб-страницы и как создавать их с нуля. В процессе выполнения задания студент учится рабоать с HTML, CSS, JavaScript и современными библиотеками, такими как Swiper. Эти знания не только полезны для создания сайтов, но и могут быть применены в научной работе, включая написание магистерской диссертации.

Один из ключевых выводов – это понимание важности использования готовых решений, таких как библиотеки и фреймворки. Они значительно ускоряют процесс разработки и позволяют сосредоточиться на решении более сложных задач.

Знания, полученные в ходе курса, могут быть полезны для написания магистерской диссертации, особенно если тема связана с веб-разработкой, пользовательскими интерфейсами или frontend-технологиями. Например:

* Создание прототипов интерфейсов для исследований.
* Разработка веб-приложений для сбора и анализа данных.
* Изучение современных подходов к верстке и их применение в научных проектах.