**Использование переменных в CSS**

**11.1 Стилизация с помощью переменных**

Как и в языках программирования, в CSS можно определять переменные. Переменные в CSS могут хранить стандартные значения, которые можно присвоить, обычным свойствам CSS, например, цвет фона, цвет шрифта, высоту шрифта, ширину и высоту элементов и так далее. Затем их можно многократно использовать в различных частях определения стилей [7].

Хотя используется термин “переменные” (css variables), официально они называются custom properties (кастомные или настраиваемые свойства).

Определение переменных CSS начинается с префикса --, например, --my-color. Они могут быть определены для любого элемента. Далее приведен пример определения переменных для задания цвета текста, начертания семейства шрифтов, а также размера текста.

**<head> <style> div {**

**--text-color1:MediumSeaGreen;**

**--text-color2: LightSkyBlue; --text-color3: HotPink;**

**--text-size1: 30px; --text-size2: 20px;**

**--text-size3: 15px; --text-font-family-verdana: Verdana;**

**--text-font-family-fantasy: fantasy;**

**--text-font-family-courier-new: 'Courier New';}**

**#div1 {font-size: var(--text-size1); var(--text-color1);**

**font-family: var(--text-font-family-verdana), serif;}**

**#div2 {font-size: var(--text-size2); color: var(--text-color2);**

**font-family: var(--text-font-family-fantasy), serif;}**

**#p1 {font-size: var(--text-size3); var(--text-color3);**

**font-family: var(--text-font-family-courier-new);} </style></head>**

**<body> <div id="div1">Волшебник изумрудного города</div>**

**<div id="div2">Сказка о приключениях маленькой девочки Элли и её собачки Тотошки. Однажды во время грозы, Элли попала в чудесную страну, а чтобы вернуться назад, ей нужно было помочь трем существам исполнить заветные желания. </div>**

**<div><p id="p1">Сказка была написана Александра Волковым в 1939 году на основе сюжета сказки американского писателя Лаймена Фрэнка Баума “Удивительный волшебник из страны Оз”.</p></div></body>**

Здесь в коде CSS для элемента div определены несколько переменных:

**--text-color1:MediumSeaGreen;**

**--text-color2: LightSkyBlue;**

**--text-color3: HotPink;**

**--text-size1: 30px;**

**--text-size2: 20px;**

**--text-size3: 15px;**

**--text-font-family-verdana: Verdana;**

**--text-font-family-fantasy: fantasy;**

**--text-font-family-courier-new: 'Courier New';}**

С помощью выражения var() можно ссылаться на эти переменные в любой части кода CSS:

**#div1 {font-size: var(--text-size1);**

**var(--text-color1); font-family:**

**var(--text-font-family-verdana), serif;}**

**#div2 {font-size: var(--text-size2); color: var(--text-color2);**

**font-family: var(--text-font-family-fantasy), serif;}**

**#p1 {font-size: var(--text-size3); var(--text-color3);**

**font-family: var(--text-font-family-courier-new);}**

Реализация данного кода представлена на рисунке 11.1.

Изображение выглядит как текст, снимок экрана, мультимедиа, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11.1 - Использование переменных в CSS3

Если потребуется изменить какие-то параметры текста (цвет, высоту, гарнитуру), достаточно будет изменить значение переменной.

Подобные переменные можно определить для любого элемента. При этом они наследуется дочерними элементами.

Если же необходимо, чтобы переменные могли бы использоваться глобально для всех элементов, тогда их определяют для элемента: root

**11.2 Создание тем CSS с помощью переменных**

Использование переменных в CSS открывает возможность создания и применения тем на веб-странице. Например, определяется следующая веб-страница [14]:

**<head> <style>**

**:root { --panel-bg-color: #ebebeb;**

**--container-bg-color: white;**

**--text-color: black;}**

**:root[theme="dark"] {**

**--panel-bg-color: #646464;**

**--container-bg-color: black;**

**--text-color: white;}**

**\* {margin: 0;}**

**html { height: 100%;}**

**body {height: 100%; font-family: Verdana; display: flex; flex-direction: column; color: var(--text-color); }**

**.navbar { align-items: center;**

**display: flex;**

**justify-content: space-between;**

**padding: 10px;**

**background: var(--panel-bg-color); }**

**.navbar a {padding: 10px; cursor: pointer;}**

**.container {flex: 1; padding: 10px;**

**background: var(--container-bg-color); }**

**footer { padding: 10px;**

**background: var(--panel-bg-color); }input[type="button"] {**

**color: var(--text-color);**

**background: var(--panel-bg-color);cursor: pointer;**

**padding: 0.3 rem; margin: 0.2 rem;**

**font-size: 1rem; }**

**</style> </head>**

**<body> <nav class="navbar">**

**<div><a>Введение в Kotlin</a>**

**<a>IDE Inteliigi Idea</a>**

**<a>IDE Android Studio</a></div>**

**<input type="button" value="Сменить тему" id="toggle-theme" />**

**</nav> <div class="container">**

**<h2 class="title">Разработка мобильных приложений с помощью языка программирования Kotlin</h2>**

**<p class="content">Kotlin - это язык разработки общего назначения, используемый в основном для разработки мобильных приложений Android. Помимо приложений Android, Kotlin также полезен для следующих целей: разработка на стороне сервера. Разработка веб-приложений на стороне сервера традиционно использует Java. </p> </div><footer>**

**<p><p>&copy; Copyright 2024</p>**

**</footer>**

**<script>**

**const toggleBtn = document.querySelector("#toggle-theme");**

**toggleBtn.addEventListener("click", function () {**

**if (document.documentElement.hasAttribute("theme"))**

**{**

**document.documentElement.removeAttribute("theme");}**

**else {**

**document.documentElement.setAttribute("theme", "dark"); }**

**});**

**</script>**

**</body>**

В стилях страницы определены две темы, которые содержат три переменных:

**:root { --panel-bg-color: #ebebeb;**

**--container-bg-color: white;**

**--text-color: black;}**

**:root[theme="dark"]**

**{**

**--panel-bg-color: #646464;**

**--container-bg-color: black;**

**--text-color: white;}**

Первая тема - условно светлая, вторая - условно темная. При темной теме корневой элемент, то есть элемент <html> будет иметь атрибут theme=“dark”. Затем эти переменные используются для установки стилевых свойств отдельных элементов.

Для переключения тем у кнопки через несложный код javascript установлен обработчик нажатия, который проверяет наличие атрибута “theme” (что будет означать, что установлена темная схема). И при наличии атрибута убирает его, а при его отсутствии, наоборот, устанавливает.

**const toggleBtn = document.querySelector("#toggle-theme");**

**toggleBtn.addEventListener("click", function() {**

**if(document.documentElement.hasAttribute("theme"))**

**{**

**document.documentElement.removeAttribute("theme");}**

**else {document.documentElement.setAttribute("theme", "dark"); }**

**});**

В результате при нажатии на кнопку произойдет глобальное переключение стилей веб-страницы. Реализация данного кода представлена на рисунках 11.2-11.3.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11.2 - Переключение тем в CSS и HTML (светлая тема)

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11.3 - Переключение тем в CSS и HTML (темная тема)

**11.3 Стили CSS как хранилище данных**

Применение переменных в CSS не ограничивается настройкой стилизации. Их предназначение более широко - а именно хранение состояния. Благодаря этому даже можно определить с помощью стилей CSS и переменных своеобразную базу данных или хранилище данных, которое может в определенных сценариях применяться для хранения данных веб-страницы.

Например, определяется файл стилей, который называется user.css со следующим содержимым:

**.user { --name: "Дмитрий";**

**--age: "25";**

**--email: "DmitrySergeev@mail.com";**

**--address: "г. Оренбург, улица Просторная, д.10, кв. 15";}**

Здесь для класса user определены четыре переменных, которые хранят некоторые значения. В данном случае все значения представляют строки.

Теперь определяется следующая html-страница, на которой будет подключаться вышеопределенный файл стилей.

**<head> <link rel="stylesheet" type="text/css" href="user.css"/>**

**<style> .user-name:after { content: var(--name); }**

**.user-age:after { content: var(--age); }**

**.user-email:after { content: var(--email); }**

**.user-address:after { content: var(--address); }   </style> </head> <body>**

**<div class="user">**

**<h2 class="user-name">User </h2>**

**<p class="user-age">Age: </p>**

**<p class="user-email">Email: </p>**

**<p class="user-address">Address: </p> </div> </body>**

Для вывода данных на страницу применяется элемент <div class="user">, в котором определены четыре html-элемента: один заголовок h2 и три параграфа. И для каждого элемента определен свой класс.

В элементе <style> определяются стили для этих элементов, которые применяют переменные из подключаемого файла user.css. Все эти стили однотипны, они устанавливают текст элемента, который добавляется после уже имеющегося текстового содержимого. Например:

**.user-name:after { content: var(--name); }**

::after создает псевдоэлемент в конце html-элемента. И у этого псевдоэлемента в качестве содержимого устанавливается текст из переменной --name.

Стили для остальных элементов идентичны. Таким образом, ожидается, что значения, которые храняться в файле user.css, будут выводиться на веб-страницу. Кинем странице в браузер и на ней будут данные из подключённого файла css.

Реализация данного кода представлена на рисунке 11.6.

Изображение выглядит как текст, снимок экрана, мультимедиа, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 11.6 – Использование CSS как хранилище данных

Хранение текстовых данных и вывод их в текстовые блоки на веб-страницу, естественно, это только частный случай. В более широком смысле переменные CSS позволяют хранить некоторое состояние, которое применяется к html-странице.