Министерство образования Республики Беларусь  
Учреждение образования  
«Белорусский государственный университет информатики и радиоэлектроники» Филиал  
«Минский радиотехнический колледж»

Учебная дисциплина «Программные средства создания Internet-приложений»

**Отчет**

по выполнению лабораторной работы

«Управление CSS-свойствами HTML-элементов при помощи языка JavaScript»

Выполнила Гончаров П.В.

Проверила Терешко О.И.

Минск

2020

Лабораторная работа № 26

Номер учебной группы: 7K2492.

Фамилия, инициалы учащегося: Гончаров П.В.

Дата выполнения работы: 09.02.2021

Тема работы: «Управление CSS-свойствами HTML-элементов при помощи языка JavaScript»

Цель работы: Формирование умений динамического изменения CSS-свойств HTML-элементов средствами JavaScript.

Результат выполнения работы:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Lab 26</title>

  <script src="script.js"></script>

  <link rel="stylesheet" href="style.css">

  <link rel="stylesheet" href="lignt.css" title="light">

  <link rel="stylesheet" href="./dark.css" title="dark" disabled>

</head>

<body>

  <!-- задание 1 -->

  <div id="div1" onclick="changeRadio(event)">

    <input type="radio" name="radio" value="yes" id="yes">

    <label for="yes">Да</label>

    <input type="radio" name="radio" value="no" id="no">

    <label for="no">Нет</label>

  </div>

  <!-- задание 2 -->

  <select name="new-list" id="new-list">

    <option value="ordered">Упорядоченный</option>

    <option value="unordered">Неупорядоченный</option>

  </select>

  <button onclick="createList()">Создать список</button>

  <div id="listDiv"></div>

  <!-- задание 3 -->

  <button onclick="changeStyle()">Изменить стиль</button>

  <div id="div3">

    <div id="textDiv">

      <p>

        Вокруг людей существует целая куча объединений людей по интересам.

        По-моему иметь хобби – это правильно. Нет хобби – есть пустота.

        А пустота всегда стремиться быть заполненной. Причем заполненной всякими ненужными,

        а иногда и вредными вещами.

      </p>

      <p>

        Хобби бывают разные. Есть хобби, в основе которых лежит коллекционирование.

        Собирать можно все: марки, редкие монеты , зажигалки, фигурки каких-нибудь персонажей.

        Коллекционирование очень полезное занятие. Оно помогает развивать людям память, нестандартное мышление.

      </p>

      <p>

        Есть хобби, смысл которых сводится к изготовлению чего либо.

        Изготовлять можно все что угодно. Важно иметь под рукой необходимые материалы.

        Можно изготовлять предметы быта и украшения из дерева или из других материалов.

        Также можно изготовлять модели, например: модель самолета или машины.

      </p>

      <p>

        Ещё одна разновидность хобби связана с разнообразной физической и умственной активностью.

        Рыбалка, собирание грибов, охота, гонки на воздушных шарах – занятия такого вида хобби.

      </p>

      <p>

        Я думаю, что хобби помогает людям многому научиться. Терпению, усердию, умению мастерить.

        Умению общаться с другими людьми. Поэтому я думаю, что хобби должен иметь каждый.

      </p>

      <p>

        Но у меня нет хобби, я ленивец)

      </p>

    </div>

    <div id="divImg">

      <img src="img/img2.jpg" alt="img2">

      <img src="img/img3.jpg" alt="img3">

      <img src="img/img4.jpg" alt="img4">

      <img src="img/img5.jpg" alt="img5">

    </div>

  </div>

</body>

</html>

//Задание 1

function changeRadio(event) {

  const { target } = event;

  if (target?.parentElement.id === "div1") {

    let activeLabel = null;

    switch (target.tagName.toLowerCase()) {

      case "input":

        activeLabel = document.querySelector(`label[for='${target.id}']`);

        break;

      default:

        activeLabel = target;

        break;

    }

    const labels = document.querySelectorAll("#div1 > label");

    [...labels].forEach(label => {

      label.style.color = "";

    })

    activeLabel.style.color = "red";

  }

}

//Задание 2

function createList() {

  const listsContainer = document.querySelector("#listDiv");

  const listContainer = document.createElement("div");

  const list = document.createElement(`${checkIsNewListOrdered() ? "ol": "ul"}`);

  listContainer.appendChild(list);

  const actions = document.createElement("div");

  const addToStartButton = document.createElement("button");

  addToStartButton.addEventListener("click", () => addNewItemToList(list, true));

  addToStartButton.innerHTML = "Добавить в начало списка";

  actions.appendChild(addToStartButton);

  const addToEndButton = document.createElement("button");

  addToEndButton.addEventListener("click", () => addNewItemToList(list));

  addToEndButton.innerHTML = "Добавить в конец списка";

  actions.appendChild(addToEndButton);

  const changeColorButton = document.createElement("button");

  changeColorButton.addEventListener("click", () => changeColor(list))

  changeColorButton.innerHTML = "Изменить цвет списка";

  actions.appendChild(changeColorButton);

  listContainer.appendChild(actions);

  listsContainer.appendChild(listContainer);

}

function checkIsNewListOrdered() {

  const newList = document.querySelector("#new-list");

  const checkedElement = newList.querySelector(":checked");

  return checkedElement.value === "ordered"

}

function addNewItemToList(list, shouldAddToStart) {

  const newItemText = prompt("Новый пункт списка:");

  const newItem = document.createElement("li");

  newItem.innerText = newItemText;

  if (shouldAddToStart) {

    list.prepend(newItem);

  } else {

    list.appendChild(newItem);

  }

}

function changeColor(list) {

  [...list.children].forEach((child, index) => {

    if (!checkIsNumberEven(index)) {

      child.style.color = "lime"

    } else {

      child.style.color = "black";

    }

  })

}

function checkIsNumberEven(number) {

  return number % 2;

}

//Задание 3

let flag = false;

function changeStyle() {

  const styleSheets = [...document.querySelectorAll("head > link[rel=stylesheet]")];

  const lightStylesheet = styleSheets.find(stylesheet => {

    return stylesheet.title === "light"

  });

  const darkStylesheet = styleSheets.find(stylesheet => {

    return stylesheet.title === "dark"

  });

  if(flag === true) {

    lightStylesheet.disabled = false;

    darkStylesheet.disabled = true;

    flag = false;

  }

  else{

    lightStylesheet.disabled = true;

    darkStylesheet.disabled = false;

    flag = true;

  }

}

#div1 label {

  color: blue

}

#divImg{

  display: flex;

  align-items: center;

  justify-content: center;

}

#divImg img{

  width: 300px;

  height: 250px;

  margin: 20px 20px;

}

#textDiv{

  border-radius: 5px;

  padding: 10px 10px;

  margin: 20px 90px;

  border: 2px solid purple;

}

body {

  background-color: white;

}

#textDiv {

  color: #070930;

  background-color: #f5f5f5;

}

body{

  background-color: black;

}

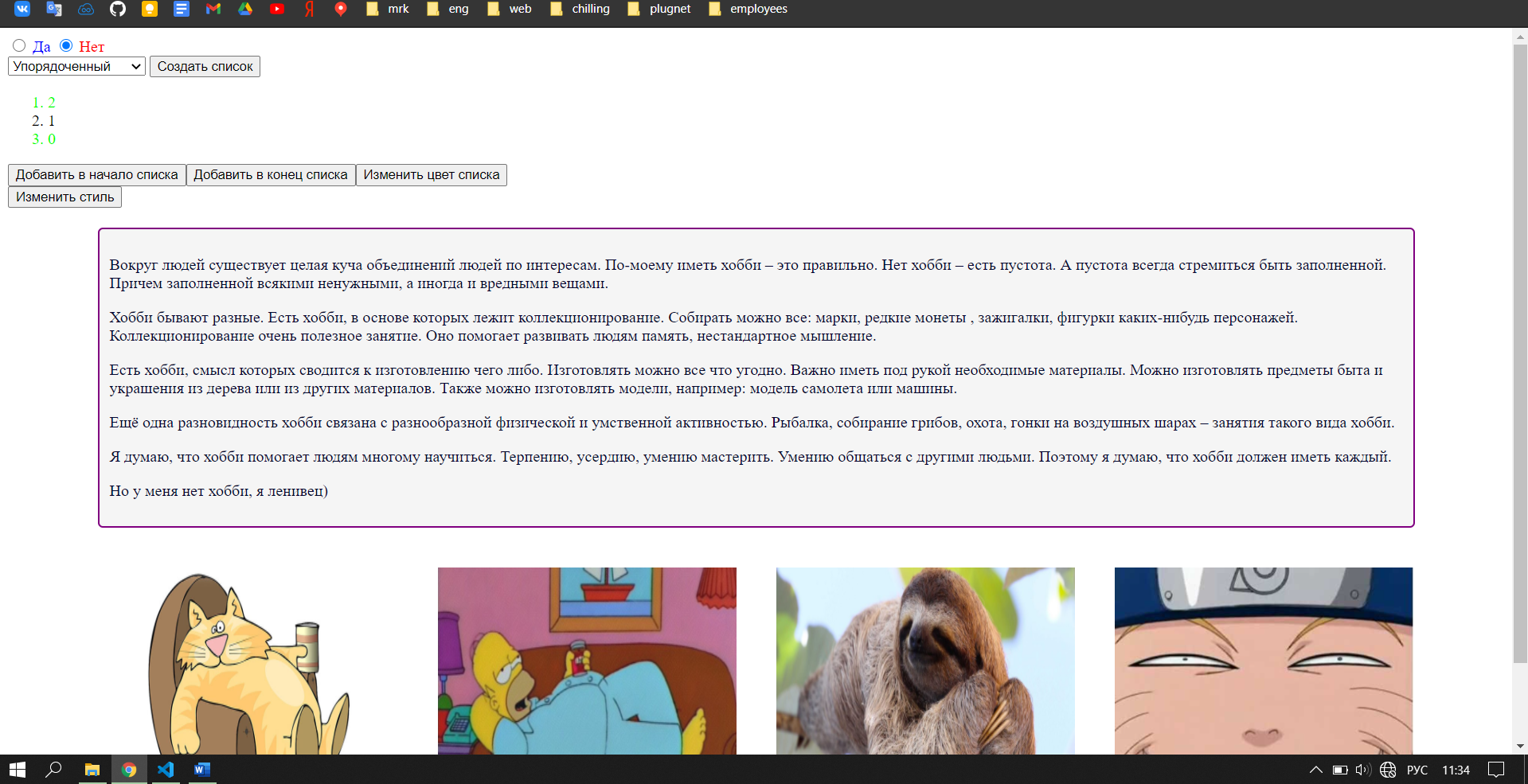
#textDiv {

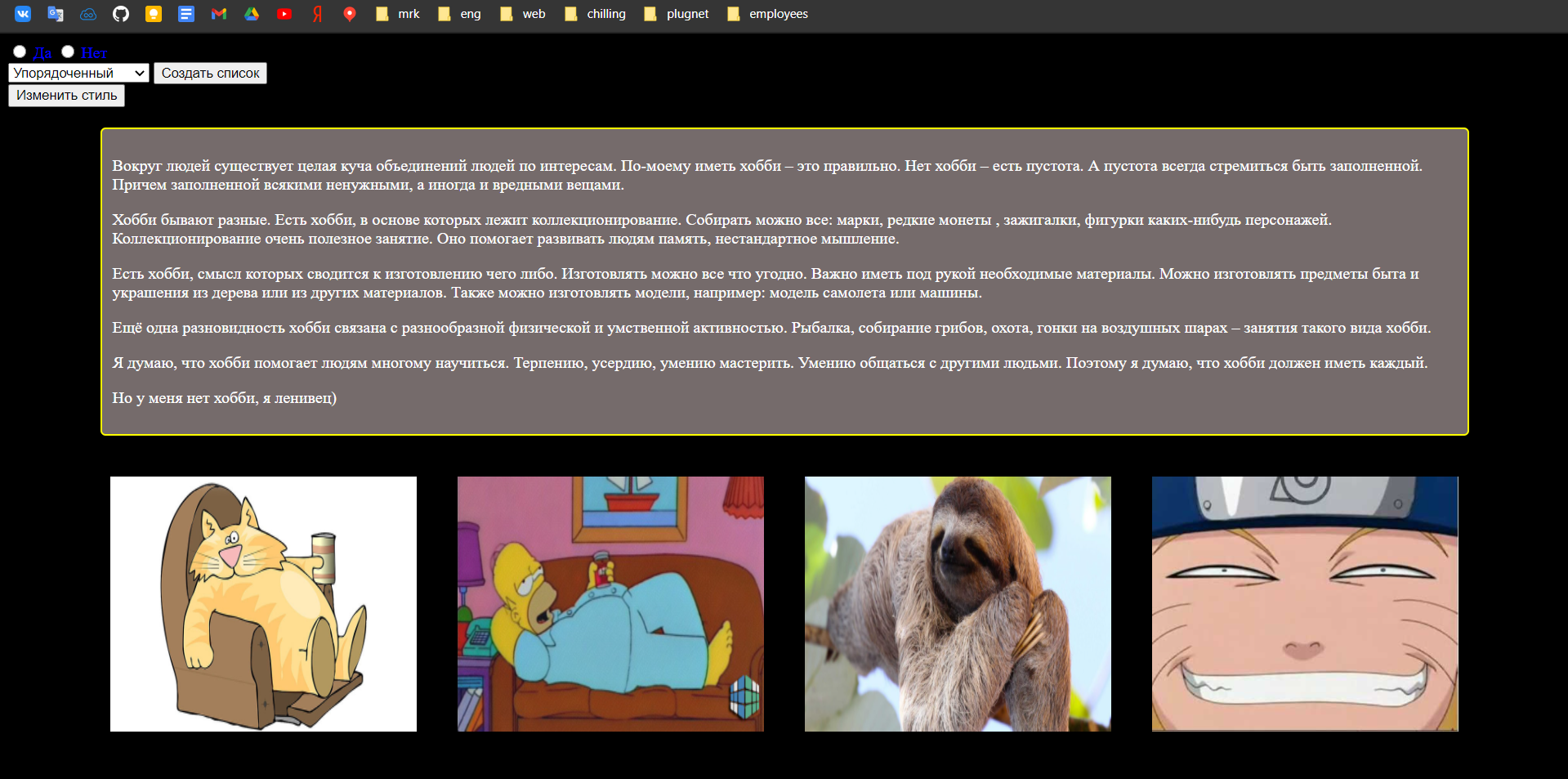
  background-color: #756b6b;

  color: white;

  border-color: yellow;

}





Контрольные вопросы и задания

1. Какие подходы используются для работы со стилевыми свойствами элементов в JavaScript?

Изменение свойства style или атрибута class.

2. Как можно получить значения всех стилей, которые в данный момент применены к элементу (с учётом внешних файлов стилей и тега <style>)?

Element.style

3. Какие операции могут выполняться с элементами <style>, <link> и объектами таблиц стилей?

* Disabled - свойство имеет тип Boolean, принимающий одно из двух значений и определяющий, применяется или нет текущая таблица стилей.
* Href - свойство возвращает строку DOMString, указывающую расположение таблицы стилей. Свойство не изменяется.
* Media - свойство возвращает ссылку на объект (узел)Node, ссылающийся на данную таблицу стилей текущего документа. Свойство не изменяется.
* parentStyleSheet - свойство возвращает ссылку на родительскую таблицу стилей StyleSheet, включающую данную таблицу стилей; если таковой нет, возвращает значение null. Свойство не изменяется
* title - свойство возвращает строку DOMString, указывающую на заголовок текущей таблицы стилей. Свойство не изменяется
* type - свойство возвращает строку DOMString, указывающую на язык, используемый таблицей стилей. Свойство не изменяется