Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

Отчет

по лабораторной работе №4

«*DHTML*»

Вариант №10

|  |  |
| --- | --- |
| Выполнил:  ст. гр. 820603  Дрозд В.А. | Проверил:  А. Л. Гончаревич |

Минск 2021

**СОДЕРЖАНИЕ**

Введение 3

1 Постановка задачи 4

2 Аналитический обзор литературы 5

3 Ход работы 8

Заключение 13

## **ВВЕДЕНИЕ**

*DHTML* (динамический *HTML*) – это набор средств, которые позволяют создавать более интерактивные *web*-страницы без увеличения загрузки сервера. Другими словами, определенные действия посетителя ведут к изменениям внешнего вида и содержания страницы без обращения к серверу.

*DHTML* построен на объектной модели документа (*Document* *Object* *Model*, *DOM*), которая расширяет традиционный статический *HTML*-документ. *DOM* обеспечивает динамический доступ к содержимому документа, его структуре и стилям. В *DOM* каждый элемент *web*-страницы является объектом, который можно изменять. *DOM* не определяет новых тегов и атрибутов, а просто обеспечивает возможность программного управления всеми тегами, атрибутами и каскадными листами стилей (*CSS*).

На *web*-страницах можно обеспечить реакцию на определенные действия посетителя или изменения состояния документа или окна, которые вызывают определенные события. Модель событий *DHTML* связана с определенной иерархией *HTML*-контейнеров и основана на всплывании событий и действии по умолчанию. Всплывание событий заключается в том, что на событие может быть получена реакция не только от элемента-источника события, но также и от всех его родительских элементов вплоть до тела документа и самого документа.

# ПОСТАНОВКА ЗАДАЧИ

В данной лабораторной работе необходимо изучить базовые возможности языка *DHTML*. Необходимо создать несколько страниц, на которых продемонстрировать работу *DHTML*. Научиться использовать фильтры, продемонстрировать работу с файлами.

# АНАЛИТИЧЕСКИЙ ОБЗОР ЛИТЕРАТУРЫ

*Dynamic* *HTML* или *DHTML* – это набор средств, которые позволяют создавать более интерактивные *web*-страницы без увеличения загрузки сервера. Другими словами, определенные действия посетителя ведут к изменениям внешнего вида и содержания страницы без обращения к серверу.

*DHTML* построен на объектной модели документа (*Document* *Object* *Model*, *DOM*), которая расширяет традиционный статический *HTML*-документ. *DOM* обеспечивает динамический доступ к содержимому документа, его структуре и стилям. В *DOM* каждый элемент *web*-страницы является объектом, который можно изменять. *DOM* не определяет новых тегов и атрибутов, а просто обеспечивает возможность программного управления всеми тегами, атрибутами и каскадными таблицами стилей (*CSS*).

На *web*-страницах можно обеспечить реакцию на определенные действия посетителя или изменения состояния документа и окна, которые вызывают определенные события.

Посетитель генерирует события при передвижении мыши, нажатии кнопок мыши и клавиатуры. Изменения состояния документа генерируют события при загрузке документа, изображений или объектов, при появлении ошибки на странице или переходе фокуса от одного элемента к другому.

Модель событий *DHTML* связана с определенной иерархией *HTML*-контейнеров и основана на всплывании событий и действии по умолчанию.

Всплывание событий заключается в том, что на событие может быть получена реакция не только от элемента-источника события, но также и от всех его родительских элементов вплоть до тела документа и самого документа. Событие может быть обработано на любом уровне. Действие по умолчанию обеспечивается встроенной в браузер обработкой событий. Например, действием по умолчанию на нажатие ссылки является переход по указанному адресу и загрузка страницы. Многие события позволяют заменить встроенные действия по умолчанию на индивидуальную обработку.

Установление связи между определенным событием и сценарием называется связыванием событий. События можно связать с помощью специальных атрибутов любого элемента и с помощью тега *script*.

*DHTML* является кроссплатформенным – независимым от платформы и языка программирования. То есть, обращение к свойствам и методам объектов в рамках *DOM* должно обеспечиваться различными операционными системами, браузерами и языками программирования.

Благодаря *DHTML* страницы начинают функционировать подобно любой компьютерной программе. Динамический *HTML* объединяет *HTML*, *CSS* и *JavaScript*. Связующим звеном для перечисленных средств служит объектная модель документа (*DOM*).

Согласно *DOM*, документ может быть представлен в виде дерева объектов. Каждому такому объекту можно присвоить имя и атрибуты и затем обратиться по этому имени к объекту и его атрибутам в сценарии, содержащемся в коде данной страницы. Элементы страницы затем можно будет изменять в результате определенного события. Изменяться может не только стиль или цвет текста, но и весь объект, включая код *HTML*, текст или рисунок. Кроме этого, имеется возможность динамически управлять стилевым оформлением страницы.

С помощью *CSS* описания признаков форматирования и свойств компоновки страницы можно сосредоточить в одном месте и избежать их повторного ввода в каждом отдельном дескрипторе. Еще одно преимущество *CSS* состоит в возможности обращаться к его правилам форматирования на языке *JavaScript* и манипулировать этими правилами, позволяя автоматизировать создание дизайна страниц и оснащать страницу разнообразными динамическими эффектами.

Динамический *HTML* имеет достаточный набор средств для манипулирования окнами путём программирования. Окна можно разделить на модальные, немодальные и наборы фреймов. Приведем методы объекта *windows*, предназначенные для выполнения операций перемещения, изменения размера и прокручивания немодальных окон и фреймов.

*MoveBy* – смещает окно на определённое расстояние, измеряемое в пикселах.

*MoveTo* – перемещает окно таким образом, что верхний левый угол оказывается в указанном месте, координаты которого указаны в пикселах.

*ResizeBy* – изменяет размер окна на определённую величину, измеряемую в пикселах.

*ResizeTo* – изменяет размер окна до указанного значения, измеряемого в пикселах.

*ScrollBy* – прокручивает документ на определённую длину, указанную в пикселах.

*ScrollTo* – прокручивает документ до определённой позиции, указанной в пикселах.

Начиная с версии 4.0 *Internet* *Explorer* поддерживает набор графических (визуальных) фильтров, которые могут быть связаны с *HTML*-содержанием. Фильтры разделяют на статические и динамические (преобразования или переходы). Фильтры могут быть непосредственно применены к тексту в документе без программирования. Приведем основные визуальные фильтры.

*Alpha* – различные эффекты прозрачности объекта.

*Blur* – размывает и смазывает объект.

*Chroma* – делает определенный цвет прозрачным.

*FlipV* – вертикальное зеркальное изображение объекта.

*Light* – осветляет объект.

*Invert* – инвертирует цвета, оттенки и яркость.

Приведем основные динамические фильтры.

*RevealTrans* – показывает или прячет объект. Использует различные типы перехода и продолжительности.

*BlendTrans* – «проявляет» объект через определенный период времени.

Фильтры перехода имеют свойства *status* и *duration*. Свойство *duration* отражает текущую продолжительность фильтра, а *status* определяет значение в зависимости от текущего состояния перехода.

Фильтры перехода лучше применять со следующими методами.

***Apply*** –используется, чтобы фактически применить фильтр.

***Play*** –используется, чтобы фактически запустить фильтр.

***Stop*** –используется, чтобы остановить применение фильтра, и может быть вызван в любое время, когда фильтр запущен.

# ХОД РАБОТЫ

В задании 4.5.1 требовалось создать набор из пяти фреймов. Для каждого фрейма был создан документ с заголовком и кратким описанием теоретического материала на основе освоенных подразделов. Страница с фреймами показана на рисунке 1.

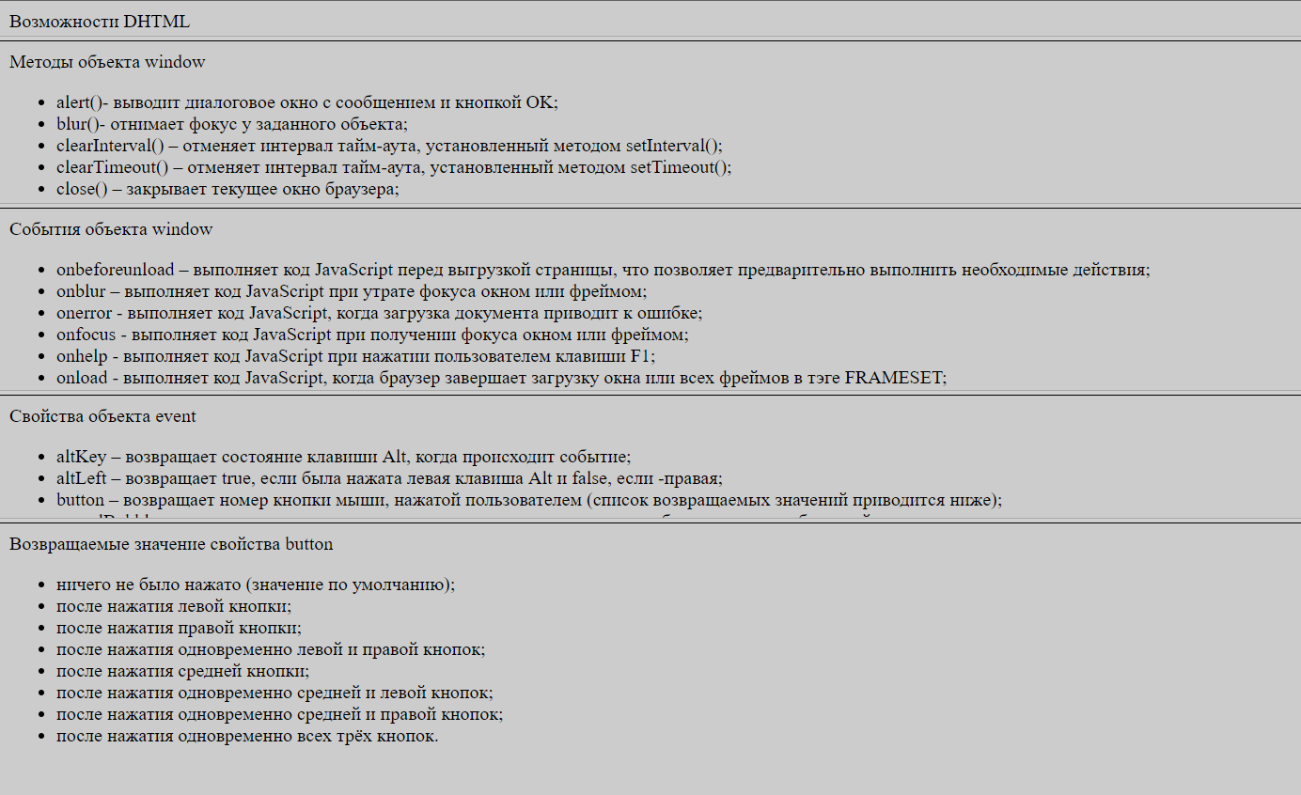


Рисунок 1 – Фреймы

В задании 4.5.2 необходимо использовать фильтры *flipv*, *light* и *invert*. Первая картинка не использует фильтров. Картинка правее отражена по вертикали при помощи фильтра *flipv*. Для следующей картинки применялся фильтр *light*. Для последней картинки применен фильтр *invert*. Результат выполнения задания продемонстрирован на рисунке 2.

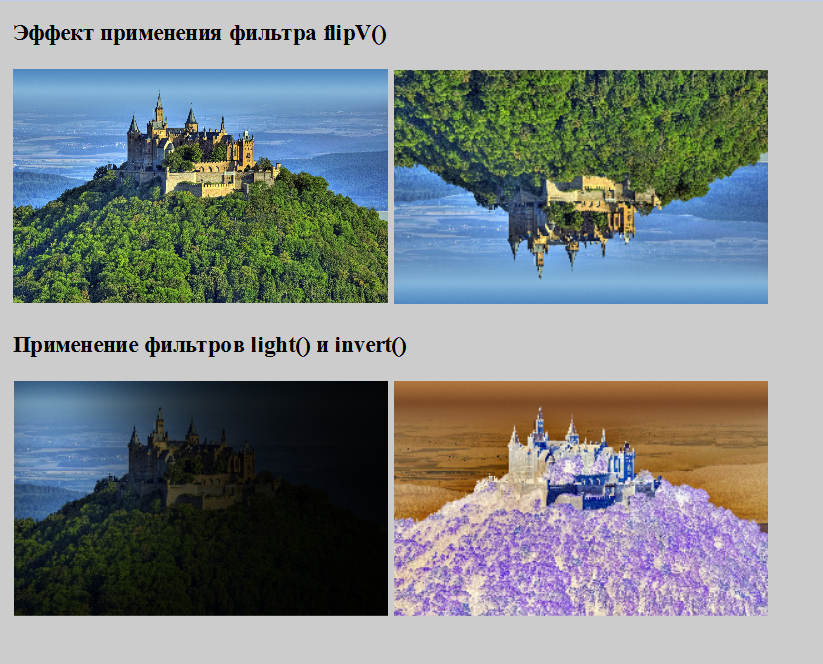
****

Рисунок 2 – Применение фильтров

В задании 4.5.4 необходимо было изменить код примера из лекции 19 так, чтобы можно было задавать также продолжительность перехода, и использовать доработанный пример для изучения визуальных эффектов всех типов перехода *revealtrans*() с возможностью введения в форме двух параметров: типа и времени перехода.

Код обрабатывающий переход, приведен ниже.

*<script>*

*function light\_logo(ii, duration) {*

*ii = Number(ii);*

*ImageLogo.filters.revealtrans.Apply();*

*ImageLogo.filters.revealtrans.transition = ii;*

*ImageLogo.filters.revealtrans.duration = duration;*

*ImageLogo.style.visibility = "visible";*

*ImageLogo.filters.revealtrans.play();*

*}*

*</script>*

Результат выполнения задания показан на рисунке 3.

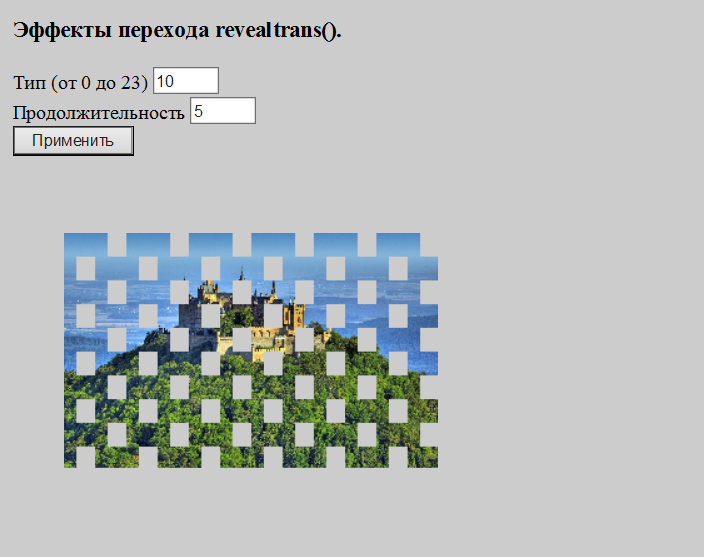
**

Рисунок 3 – Эффект перехода

В задании 4.5.4 требовалось создать часы, чтобы на них отображалось время с точностью до 10 микросекунд. На рисунке 4 показан результат работы часов.

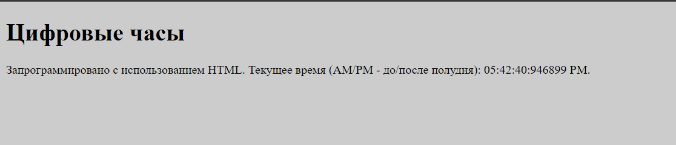


Рисунок 4 – Часы

В задании 4.5.5, по условию, есть текстовый файл, в котором каждая строка содержит информацию о товаре. Формат строки имеет следующий вид: Код\_товара#Наименование\_товара#цена#имя\_файла\_с\_описанием\_товара. Необходимо разработать два документа. В одном будет отображаться список товаров, по нажатию на товар загружается следующий документ, отображающий содержимое файла с подробным описанием товара.

Код программы приведен ниже.

*function showFile(input) {*

*let file = input.files[0];*

*let reader = new FileReader();*

*let productCodes = new Array();*

*let productNames = new Array();*

*let productPrices = new Array();*

*let productFiles = new Array();*

*reader.readAsText(file);*

*reader.onload = function () {*

*let rows = reader.result.split("\n");*

*const oTable = document.getElementById("goodsTable");*

*rows.forEach(sRow => {*

*let aData = sRow.split('#');*

*let oTableRow = oTable.insertRow(1);*

*let oNumberCell = oTableRow.insertCell(0);*

*let oNameCell = oTableRow.insertCell(1);*

*let oPriceCell = oTableRow.insertCell(2);*

*let oLinkCell = oTableRow.insertCell(3);*

*let oLink = document.createElement("a");*

*var oLinkText = document.createTextNode("подробнее...");*

*oLink.appendChild(oLinkText);*

*oLink.setAttribute("href", "../data/" + aData[3]);*

*oNumberCell.innerHTML = aData[0];*

*oNameCell.innerHTML = aData[1];*

*oPriceCell.innerHTML = aData[2];*

*oLinkCell.appendChild(oLink);*

*});*

*};*

*}*

Для начала работы необходимо выбрать файл с информацией о товарах. После выбора файла на странице появится таблица с товарами. На рисунке 5 показан выбор файла.

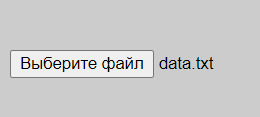


Рисунок 5 – Выбор файла

Затем пользователь выбирает интересующий его товар из таблицы. Таблица показана на рисунке 6.

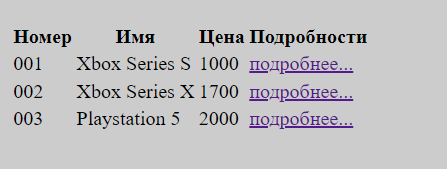


Рисунок 6 – Отображение таблицы с товарами

После выбора товара на другой странице появляется подробная информация о товаре. Вывод описания продемонстрирован на рисунке 7.

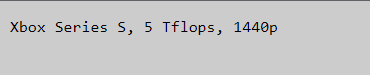
****

Рисунок 7 – Подробное описание товара

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной лабораторной работы мною были изучены основы работы с языком *DHTML*. Были реализованы примеры использования фреймов и фильтров. В ходе работы были также применены методы работы с данными.