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 №2

«Работа с *JavaScript*»

Вариант № 18

|  |  |
| --- | --- |
| Выполнил:  Жуков А.А. | Проверил:  А. Л. Гончаревич |

Минск 2021

СОДЕРЖАНИЕ

[1 Цель работы 3](#_Toc26004)

[2 Теоретическая часть 4](#_Toc4500)

[3 Ход работы 8](#_Toc30027)

[Заключение 15](#_Toc30066)

# ЦЕЛЬ РАБОТЫ

Познакомиться с синтаксисом языка *JavaScript,* изучить методы *alert*(), *confirm*() и *prompt*(). Научиться инициализировать переменные и создавать массивы, написать калькулятор с поддержкой нескольких операций, в том числе префиксных и постфиксных. Добавить выполненное на сайт, созданный в лабораторной работе 1.

# ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

*JavaScript* является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, *JavaScript* имеет ряд свойств, присущих функциональным языкам — функции как объекты 1-го класса, объекты как списки, карринг, анонимные функции, замыкания — что придаёт языку дополнительную гибкость.

Несмотря на схожий с *C* синтаксис, *JavaScript* по сравнению с языком Си имеет отличия, к примеру объекты с возможностью интроспекции, функции как объекты первого класса, автоматическое приведение типов, автоматическая сборка мусора, анонимные функции.

В языке отсутствуют такие полезные вещи, как модульная система, стандартная библиотека, стандартные интерфейсы к веб-серверам и базам данных, система управления пакетами, которая бы отслеживала зависимости и автоматически устанавливала их.

Синтаксис языка *JavaScript* во многом напоминает синтаксис С и *Java*, семантически же язык гораздо ближе к *Self*, *Smalltalk* или даже Лиспу.

Простейшие элементы, которыми оперируют остальные, более сложные элементы, называют литералами. В языке *JavaScript* предусмотрены следующие типы литералов: целого типа, вещественные, логического типа, строковые литералы.

К объектам относятся данные основных системных типов, например, числовые значения и массивы. Свойствами являются значения, принадлежащие объектам, например, *Math*.*PI* – значение математической константы π объекта *Math*.

Объект – это какой-либо предмет, в том числе и составляющие компьютерного виртуального мира. Что касается *JavaScript*, его объекты (готовые к использованию программы) находятся внутри браузера. Кроме этого, непосредственно в языке *JavaScript* имеется возможность создавать новые объекты и потом их использовать в каждом подходящем случае. При объектно-ориентированном программировании первоочередное внимание уделяется объектам, которые могут производить некоторые манипуляции (действия), а не логическим правилам и вычислительным процедурам, необходимым для таких манипуляций. Примером объекта может служить объект *Window* – окно при запуске браузера.

Метод – это действия, которые может выполнять объект. В реальном мире каждый объект тоже обладает некоторым набором методов (действий): машины ездят, собаки лают, доллар покупается и т.д. Примерами методов объекта являются open(), *close*() и *click*(): открытие окон, закрытие окон и нажатие кнопок.

У каждого объекта могут иметься какие-то свойства (*properties*): машины можно отличить по колесам, а у собаки имеется шерсть. Что касается *JavaScript*, то, например, у такого объекта языка, как браузер, имеется название и номер версии.

Также, как и в любом языке программирования, в *JavaScript* есть возможность использовать однострочные и многострочные комментарии, которые используются разработчиками программ для того, чтобы облегчить чтение собственного кода при повторном обращении к нему. Вставив два подряд идущих прямых слэша // в начале строки, можно скрыть от показа в окне браузера следующий за этими символами текст. А чтобы сделать комментарием многострочный текст нужно ограничить его символами /\* … \*/.

В *JavaScript* используется три основных типа данных (литералов) и соответствующих им переменных: строка (*string*), число (*number*), булево значение (*boolean*).

Строковые литералы (строки), вставляемые в сценарий, заключаются в двойные или одинарные кавычки. Строка – это связанный набор символов, таких как буквы, знаки препинания и цифры. Чаще всего строки представляют собой какой-либо текст.

Числа, воспринимаемые *JavaScript,* могут быть двух типов: целые числа и числа с плавающей точкой (вещественные). Хотя чаще всего используются числа, записанные в десятичной системе счисления, могут применяться и числа в восьмеричной и шестнадцатеричной системе. Вещественные числа могут представляться с дробной десятичной частью, либо в экспоненциальном виде (3.14*e*2 или 3.14*E*2).

Булевы значения могут принимать лишь два значения: *true* (истина) и *false* (ложь). Используется для проверки какого-либо условия.

Кроме рассмотренных типов данных существуют еще некоторые другие, которые также используются. Это, к примеру *NaN* (несуществующее число), которое свидетельствует об обращении к несуществующему числу; *null* (неопределенный тип), который уведомляет о полном отсутствии каких-либо данных; *undefined* (неопределяемый тип), который говорит об ошибке в вашей программе.

Для преобразования строк в числа в *JavaScript* предусмотрены встроенные функции *parseInt*() и *parseFloat*(). Можно также использовать встроенную функцию *Number*(). Функция parseInt(строка, основание) преобразует строку в целое число в системе счисления по указанному основанию (8, 10, 16). Основание можно не указывать, то предполагается десятеричная система счисления (основание 10). Функция *parseFloat*(строка) преобразует строку в число с плавающей точкой.

Обратная задача преобразования строк в числа осуществляется с помощью встроенной функции *String*(*n*), которая преобразует число *n* в строку. Такое преобразование также выполняется простым прибавлением числа к строке с помощью оператора сложения для строк (оператора конкатенации). Понятие «оператор» будет рассмотрено ниже.

Встроенная функция *IsNaN*(значение) может использоваться для определения того, является ли значение числом. Если указанное значение не является числом, функция возвращает логическое значение *true*, а иначе – *false*. При этом эта функция считает числом и строку, содержащую только числа, а также логические значения.

Переменные имеют огромное значение в любом языке программирования. Переменная – это имя (или идентификатор), присваиваемое ячейке памяти компьютера, которая хранит определенные литералы во время исполнения сценария *JavaScript*. Вам следует помнить, что имена переменных не могут начинаться с цифры и содержать пробелы.

Переменные в *JavaScript* создаются очень легко двумя способами: объявив ее заранее или создав «на лету». Чтобы объявить (создать) переменную в языке *JavaScript* используется оператор *var*, вслед за которым указывается имя, которое присваивается переменной.

Область действия переменных - то, где их можно использовать в скрипте. В *JavaScript*, существует две области действия, которые переменные могут иметь. Это глобальная, когда переменную можно использовать внутри программы, в том числе и внутри всех функций, вызываемых в программе после создания глобальной переменной, и локальная, когда переменную можно использовать только внутри текущей функции. Чтобы объявить локальную переменную внутри функции, необходимо использовать ключевое слово *var*.

Существует несколько способов ввода кода *JavaScript* в *HTML*-документ, например с помощью использования тега <*SCRIPT*>. Блок <*SCRIPT*> и </*SCRIPT*> – это блок языка *HTML*, специально предназначенный для указания браузеру, что содержимое блока, заключенное между открывающим и закрывающим тэгами, является сценарием, для которого требуется соответствующая интерпретация. Этот блок может быть вставлен в любом месте внутри блока <*head*> … </*head*>, но обычно добавляется в конце, перед закрывающим тэгом </*head*>.

Однако такой способ не подходит для серьёзных проектов, так как *JavaScript* код при необходимости внесения изменений придётся искать и редактировать во всех *HTML* документах сайта, что увеличивает объём *HTML* документов, делает поддержку и развитие проекта затратным по времени и ресурсам. Поэтому *JavaScript* код лучше выносить в отдельные файлы с расширением *.js*, которые можно подключать в любой *HTML* документ с помощью команды *<script src="myscripts.js"></script>*, такой подход значительно упрощает разработку и поддержку сайта, поэтому считается предпочтительным и конвенциональным.

Escape-последовательности используются для управления выводом информации:

– \b возврат на одну позицию;

– \t табуляция;

– \f переход на следующую страницу;

– \n переход на следующую строку;

– \r возврат каретки;

– \” двойная кавычка;

– \’ одинарная кавычка - апостроф;

– \\ обратная косая черта - обратный слэш.

# 3 ХОД РАБОТЫ

Рассмотрим реализацию регистрации на сайте, которая доступна при нажатии кнопки *Sign In*, показанной на рисунке 3.1.

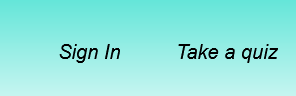


Рисунок 3.1 – Регистрация

Процесс состоит из трёх частей: ввод информации, подтверждение ввода и вывод информации. Ввод показан на рисунках 3.2–3.4, а подтверждение и вывод на рисунках 3.5 - 3.6.

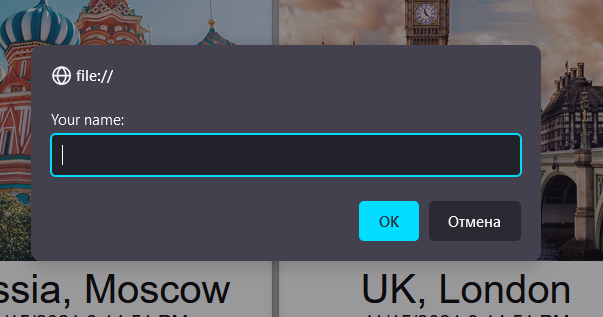


Рисунок 3.2 – Ввод имени

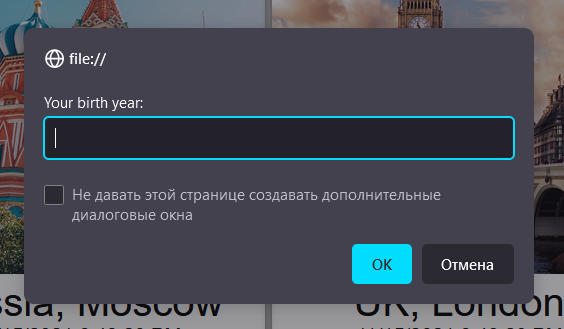


Рисунок 3.3 – Ввод года рождения

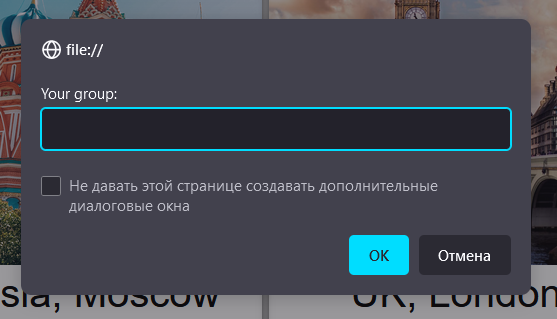


Рисунок 3.4 – Ввод номера группы

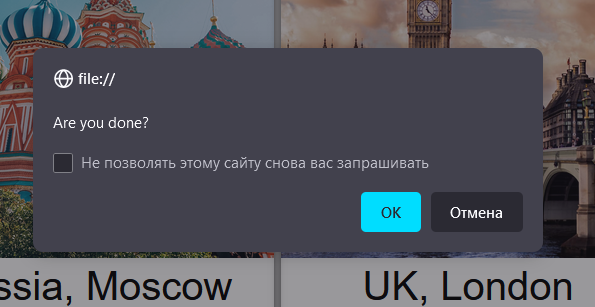


Рисунок 3.5 – Подтверждение ввода

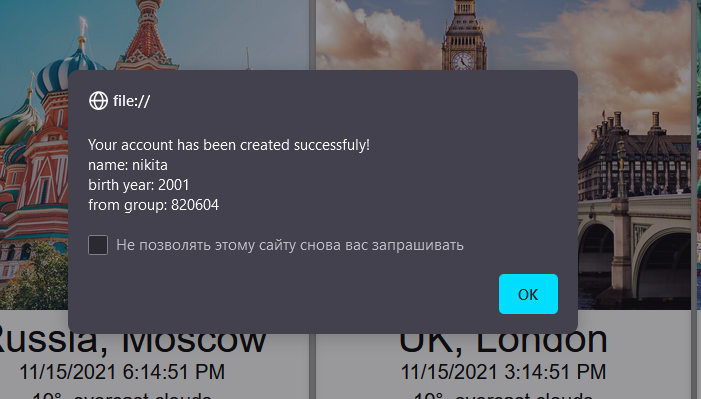


Рисунок 3.6 – Вывод информации

Воспользоваться календарем или калькулятором можно, перейдя на вкладку *Instruments*, расположенной на главной странице. Внешний вид вкладки показан на рисунке 3.7.

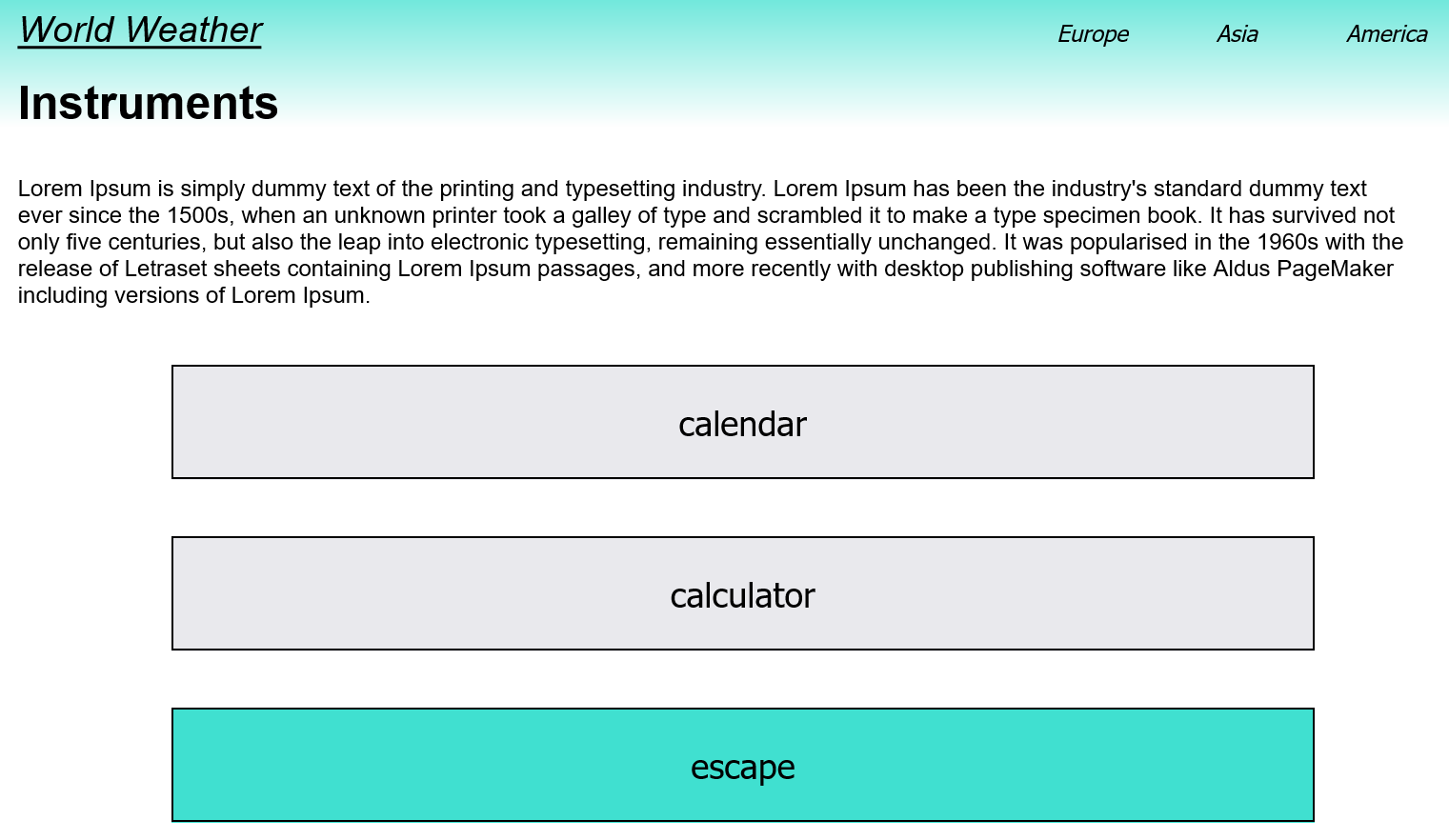


Рисунок 3.7 – Инструменты

При нажатии на кнопку *calendar* есть возможность выбрать язык, после чего появляется список месяцев на этом языке, реализованный с помощью массивов на языке *Java Script*. Процесс показан на рисунках 3.8 - 3.9.

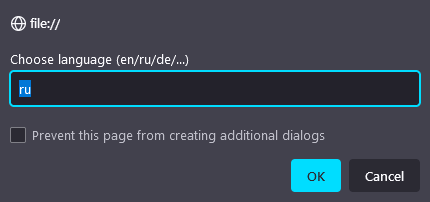


Рисунок 3.8 – Выбор языка

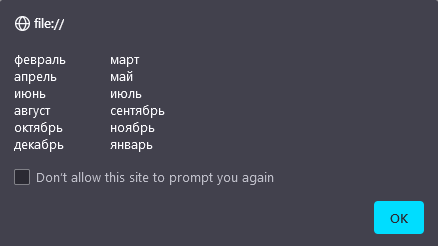


Рисунок 3.9 – Вывод месяцев на выбранном языке

При нажатии на кнопку *calculator* открывается окно для ввода математического выражения, после ввода появляется окно с результатом. Процесс показан на рисунках 3.10, 3.11.

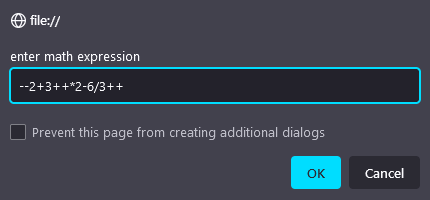


Рисунок 3.10 – Ввод выражения

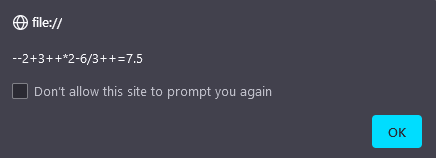


Рисунок 3.11 – Вывод результата

После нажатия на кнопку escape во всплывающем окне отобразиться текст-образец, где использованы все основные экранирующие последовательности, результат приведён на рисунке 3.12.

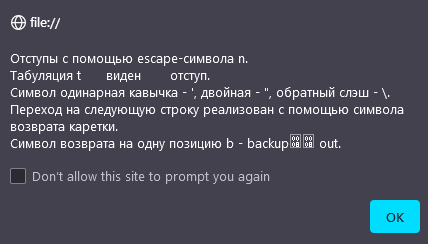


Рисунок 3.12 – Текст с экранирующими последовательностями

# ЗАКЛЮЧЕНИЕ

*JavaScript* – это язык программирования, который даёт возможность реализовывать сложное поведение веб-страницы. В ходе лабораторной работыизучены методы *alert*(), *confirm*() и *prompt*(), инициализированы переменные и созданы массивы, написан калькулятор с поддержкой нескольких операций, в том числе префиксных и постфиксных. Весь новый функционал адаптирован к лабораторной работе 1.