ОТДЕЛ ПО ОБРАЗОВАНИЮ ПЕРВОМАЙСКОГО

РАЙОНА ГОРОДА ВИТЕБСКА

ГУО «Гимназия № 5 г. Витебска имени И. И. Людникова»

Секция «Информатика»

Создание оконного приложения с помощью веб-интерфесов

Автор:

Иванов Александр Игоревич

Руководитель:

Овчинникова Оксана Викторовна

Адрес учебного заведения:210027, Витебск, ул. Чкалова 19-а, гимназия №5, тел.8-0212-642774, gymn5@pervroo-vitebsk.gov.by

**Содержание**

1. [Введение и информация об инструментах, использовавшихся в работе](#Вступление)
2. [Основная часть](#ОснованаяЧасть)
3. [Заключение](#Заключение)
4. [Источники информации](#ИсточникиИнформации)
5. [Приложения](#Приложения)

**Введение**

Задачи: создать оконное приложение (калькулятор) для компьютерных операционных систем с использованием веб-интерфейсов, изучить язык гипертекстовой разметки – HTML, язык стилей – CSS, языки программирования: Python, JavaScript.

Главная цель: доказать, что приложения на веб-интерфейсах – это легко и полезно.

Актуальность: веб-интерфейсы появились относительно давно, но сейчас, с появлением HTML5 и CSS3 разработка веб-страниц вышла на новый уровень. Сегодня появляется всё больше приложений, созданных на веб-интерфейсах, потому что это просто и быстро. Ярким примером приложения на веб-интерфейсах служит Discord, что имеет значительный успех на рынке мессенджеров.

**Основная часть**

Немного информации о том, с чем мы будем иметь дело:

**Веб-интерфейс** — веб-страница или совокупность веб-страниц, предоставляющая пользовательский интерфейс для взаимодействия с сервисом или устройством посредством протокола HTTP и веб-браузера. Веб-интерфейсы получили широкое распространение в связи с ростом популярности всемирной паутины[1] и соответственно — повсеместного распространения веб-браузеров.

Одним из основных требований к веб-интерфейсам является их одинаковый внешний вид и одинаковая функциональность при работе в различных браузерах.

Классическим и наиболее популярным методом создания веб-интерфейсов является использование HTML с применением CSS и JavaScript'a.

Основным преимуществом веб-интерфейсов является отсутствие необходимости установки дополнительного программного обеспечения, так как популярные операционные системы поставляются с уже установленным браузером.

Для отображения веб-интерфейсов нужен браузер. Чтобы открыть такую страницу, как отдельно оконное приложение, мы воспользуемся Eel.

**Eel**— это небольшая библиотека Python для создания простых автономных приложений HTML/JS с графическим интерфейсом, подобных Electron, с полным доступом к возможностям и библиотекам Python.

С помощью eel мы можем открыть нашу веб-страницу, как отдельное от браузера приложение, и самое главное – связать всю логику с Python, что позволяет нам воспользоваться всеми его библиотеками, которых на сегодняшний день огромное множество.

**HTML** (от англ. HyperText Markup Language — «язык гипертекстовой разметки») — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее интерпретируют код в интерфейс, который будет отображаться на экране монитора.

**CSS** (англ. CascadingStyleSheets «каскадные таблицы стилей») — формальный язык описания внешнего вида документа (веб-страницы), написанного с использованием языка разметки (чаще всего HTML или XHTML). Также может применяться к любым XML-документам, например, к SVG или XUL.

*То есть HTML – каркас, CSS – стиль сайта.*

**JavaScript** — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Не путать JavaScript с Java!

**Python** — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

*То есть JavaScript – язык программирования для интерактивности интерфейса, Python – для основной программы.*

Основные среды разработки, которые были использованы:

VisualStudioCode – удобный редактор кода, поддерживающий иерархию проекта, подсветку синтаксиса и компиляцию кода.VSC

Работа началась с создания интерфейса для домашней страницы – страницы, которая встречает пользователя при первом запуске программы. Тогда и началось изучение HTML и CSS. На самом деле, найти информацию об основах создания сайта не составило труда, ввиду популярности и актуальности веб-интерфесов.

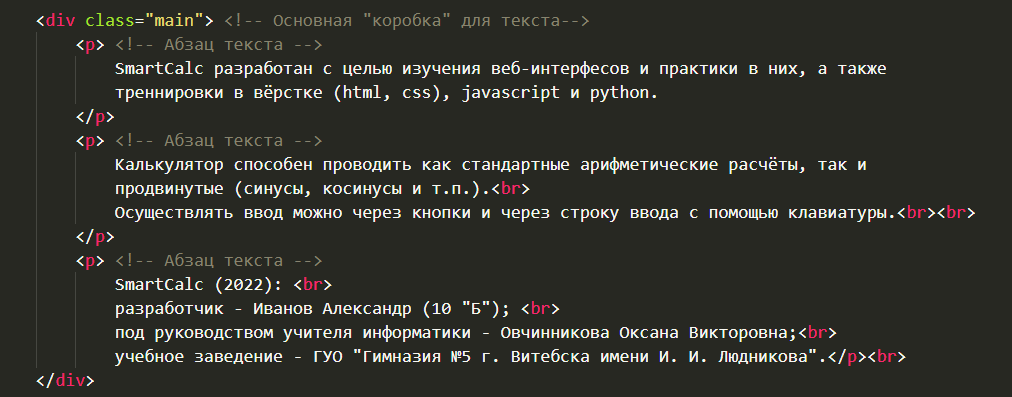
В работе с HTML используются тэги. Основные из них:<html> , <head>, <body>, нужны обязательно в каждой странице. В теге <body> прописывается вся видимая часть интерфейса типа заднего фона, текста, картинок, кнопок и т.п. (<p>, <div> и др.). В теге <head> располагаются служебные теги, которые не видны пользователю: подключение библиотек, например, CSS,JS (<script>, <meta>, <link>);видимые пользователю: заголовок, иконка страницы (<title>).

Здесь можно сразу выделить главные преимущества такого способа создания интерфейса:

* Код понятен своему создателю;
* С помощью CSS можно создать **абсолютно любой** дизайн из-за гибкости языка и множества различных параметров, которые можно изменить у элемента страницы;
* В этом нет ничего сложного;
* Изменения можно легко отслеживать в окне браузера или программы, достаточно лишь сохранить документ и перезагрузить страницу, например, с помощью F5, ctrl+R и т.п.

Так выглядит текст, отображаемый на начальной странице, в HTML коде.

Рисунок 1 Пример текста на домашней странице



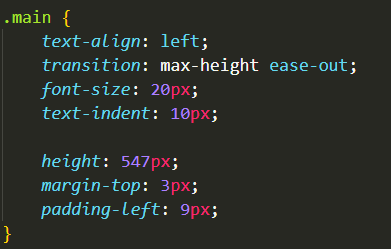
<div> - контейнер для других тегов. С помощью <div> можно связать между собой элементы, чтобы управлять ими как отдельное целое. Это полезно, когда нужно применить какой-нибудь стиль(например, шрифт) для всех элементов контейнера.

<p> создаёт отдельный абзац текста и переводит курсор на новую строку после него.

<br> - искусственный переход на новую строку.

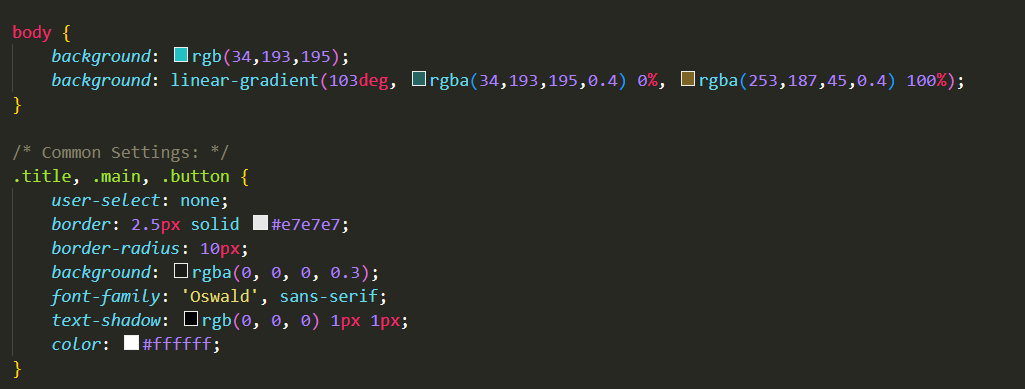
Class=”main” – присвоение контейнеру отдельного класса, к которому можно обратиться в CSS коде и прописать стили:

Рисунок 2



Остальные стили на домашней странице, где прописан шрифт, размер шрифта, фон, тени текста:

Рисунок 3 Общие настройки



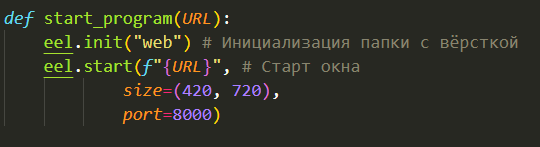
Параметры имеют говорящие названия, поэтому проблем с пониманием кода не будет.

После того, как базовые знания о верстке были получены, нужно было создать основную программу на python, которая бы создавала окно и запускала страницу.

С помощью библиотеки Eel это оказалось несложно сделать.

Так выглядит функция, которая вызывает окно по входящей ссылке:

Рисунок 4



Eel.init инициализирует папку, где лежат HTML документы.

Eel.startзапускает страницу по полученной ссылке к ней.

*f*в *f”{URL}”* означает, что в фигурных скобках используется переменная и она не является частью строки.

Size задаёт размеры выводимого окна, port – порт, который используется для открытия страницы.

**!** Главное не забыть добавить на каждую страницу проекта в теге <head> следующую строку - <scriptsrc="eel.js"></script>, иначе ничего не будет работать.

На странице так же есть интерактивный элемент – кнопка, которая переключает активную страницу намануал к использованию калькулятора.

Код на HTML и CSS для кнопки:

Рисунок 5

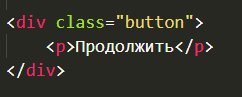
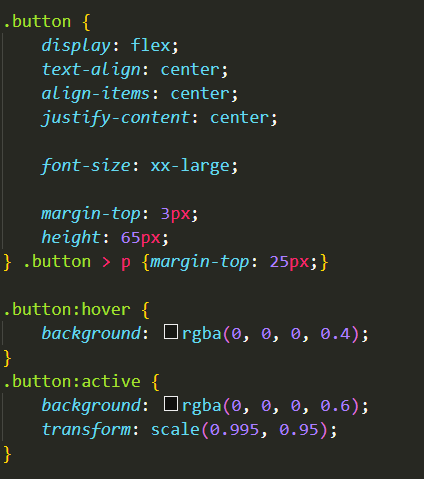


Рисунок 6

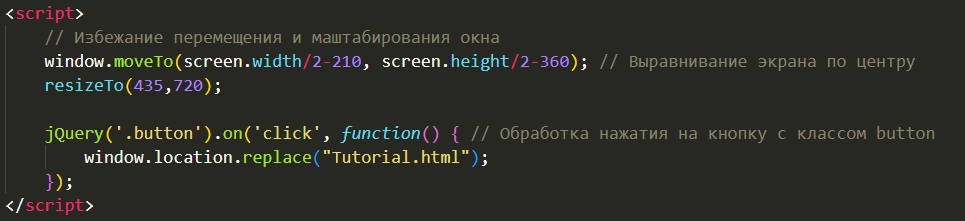


Измение стандартных параметров было видно на Рисунок 3.

Из новыхинтересностей языка css: состоянияhover (когда пользователь наводит курсор на элемент) и active (когда пользователь нажимает на элемент). Это нужно, для создания плавности и интерактивности интерфейса.

Чтобы конопка работала, подключим javascript:

Рисунок 7



Window.location.replace() переключает активную страницу на ту, что указана в скобках.

На самом деле этот обработчик нажатий принадлежит jQuery – библиотеке функция js, созданных, чтобы облегчить работу программиста.

Таким образом была получена готовая домашняя страница с интерактивным элементом.

Далее следует страница с мануалом. Здесь всё схоже с предыдущей: добавить заголовок, минимальный интерфейс, задний фон, несколько картинок и кнопки.

Остановимся на кнопках и на том, как они работают. Дизайн создантак же на html и css:

Рисунок 8

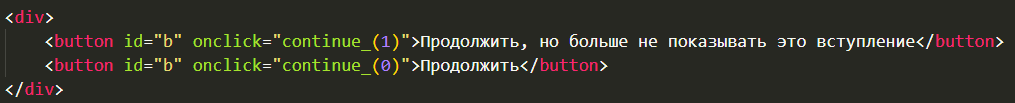
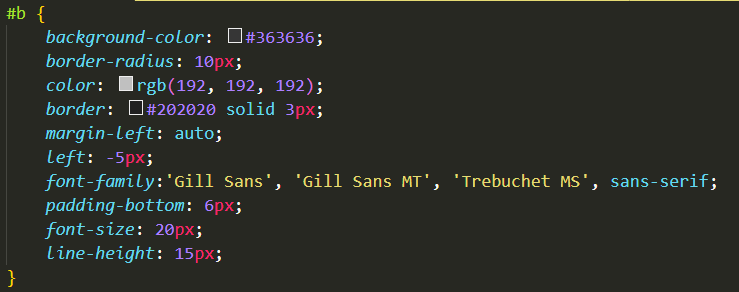
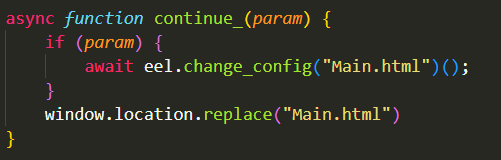


Рисунок 9



Ничего особенного в графической части здесь нет, а принцип взаимодействия с кнопками немного отличается. У тега <button> есть атрибут – onclick, который выполняет какое-то действие при нажатии на кнопку. Здесь же запускается функцияcontinue\_(), которая принимает значения 0 или 1.

Рисунок 10



Суть функции в том, что если пользователь не хочет более видеть вступление, то она получает от кнопки на вход 1 и обращается к программе на python, где меняется в конфиге (config.txt) значение URL, и теперь там окажется ссылка не на домашнюю страницу (“Home.html”), а ссылка на страницу калькулятора (“Main.html”).

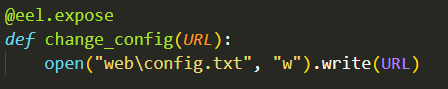
Чтобы обратиться к функциям в python программе, нужно создать в асинхронную функцию (функция, которая может работать параллельно с программой и не приостанавливает её) в коде js. Ключевое слово – async. После обращаемся к функции в python программе:

eel.<имя функции>**(**<параметры, которые принимает функция>**)(***здесь должно быть пусто***)**;

Ключевое слово – await.

Далее просто переключаемся на страницу калькулятора.

Рисунок 11 Изменение конфига в python



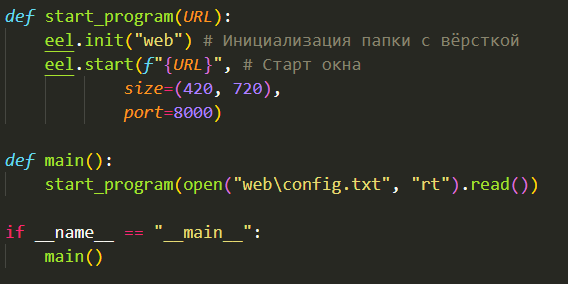
@eel.exposeнужен, чтобы к функции можно было обратиться из js кода.

Немного о том, как происходит взамодействие с конфигом:

На Рисунке 11 изображено изменение конфига. Сначала он открывается, а после перезаписывается значение на полученное от js.

А так выглядит старт программы, если учесть всё, что происходит после запуска:

Рисунок 12



Если программа запущена → открываем конфиг → читаем ссылку на html страницу → передаём полученный результат на старт окна.

Перейдём к основной программе – калькулятору.

Здесь размер кода растянулся даже до 250 строк только в html документе. Большая его часть – код на js, который было решено писать именно в самом документе html (так показалось удобнее).

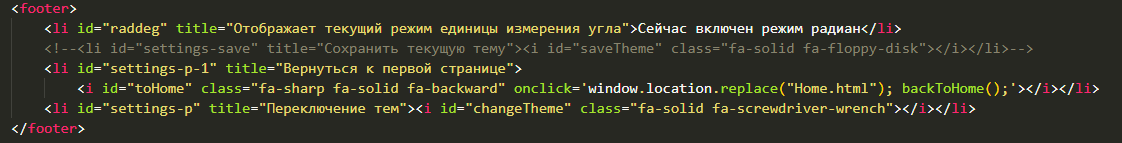
Новые элементы страницы:

1. <input> - поле, где вводится запрос. Было использовано несколько его атрибутов: autocomplete=“off” (чтобы не выводились подсказки от браузера), type=“text” (чтобы обозначить вводимое строковым типом), onblur= “this.focus()” (чтобы поле ввода было всегда активным).
2. <footer> - контейнер для элементов, которые находятся в низу страницы.
3. Тег <li> определяет отдельный элемент списка. Здесь использовался для отображения иконок переключения тем и возвращения в главное меню, а так же для отображения текста с текущим состоянием режима градусов. Иконки этих элементов были взяты с FontAwesome. Их подключение и использование в программе:

Рисунок 13

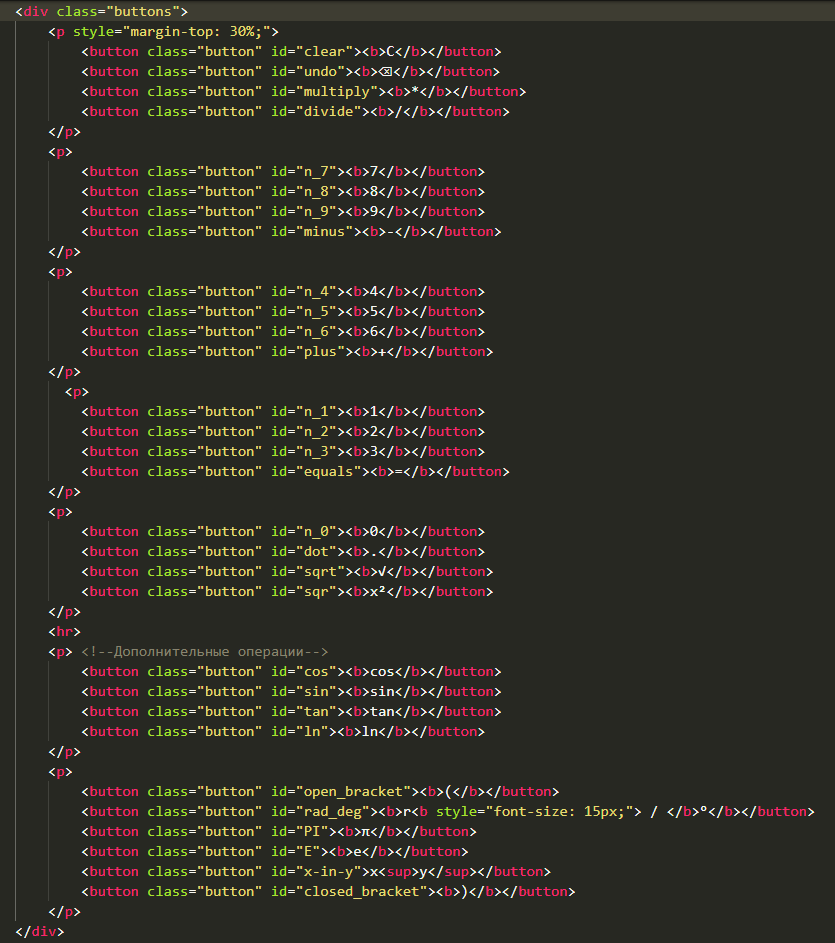


Рисунок 14



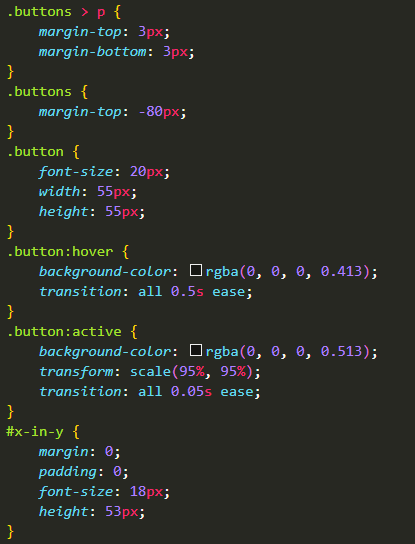
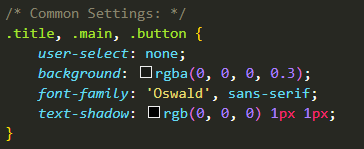
Массив кнопок представлен таким образом:

Рисунок 15



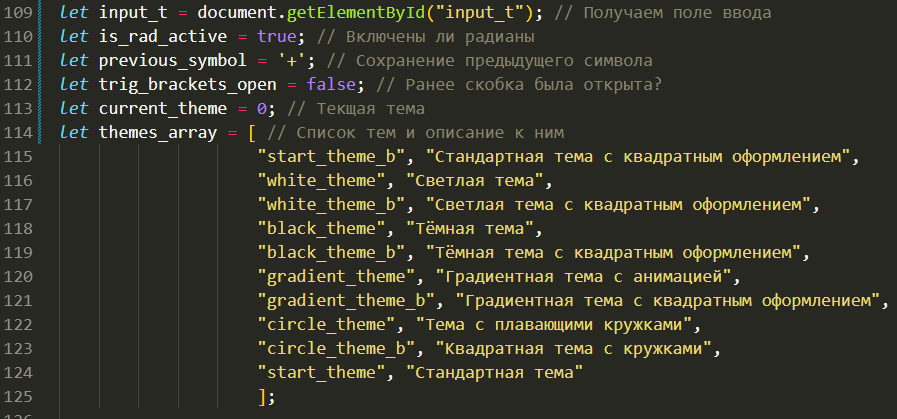
Стили для них:

Рисунок 16



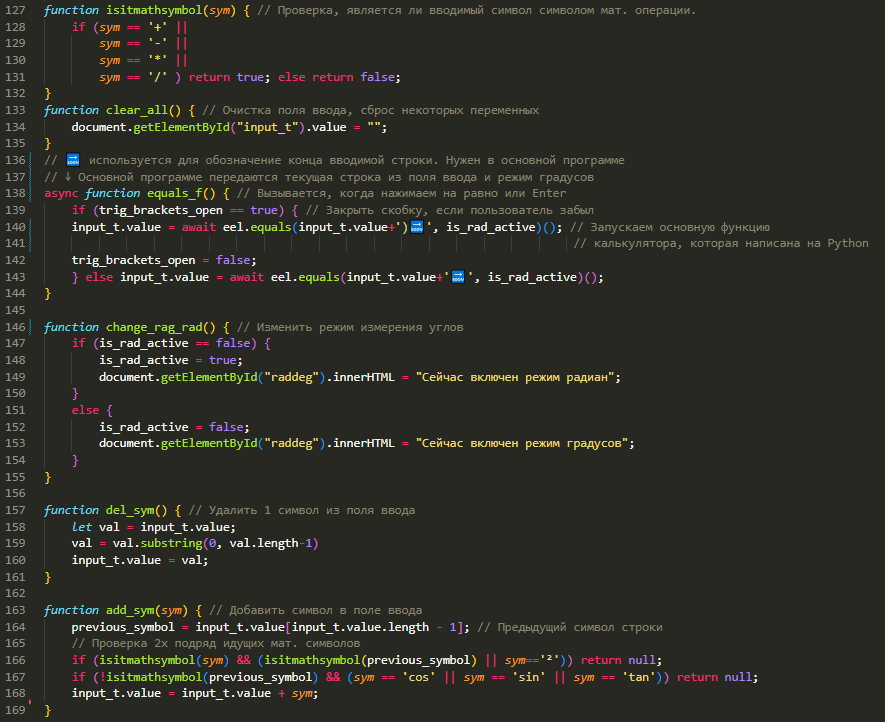
Переменные, которые использовались в программе:

Рисунок 17



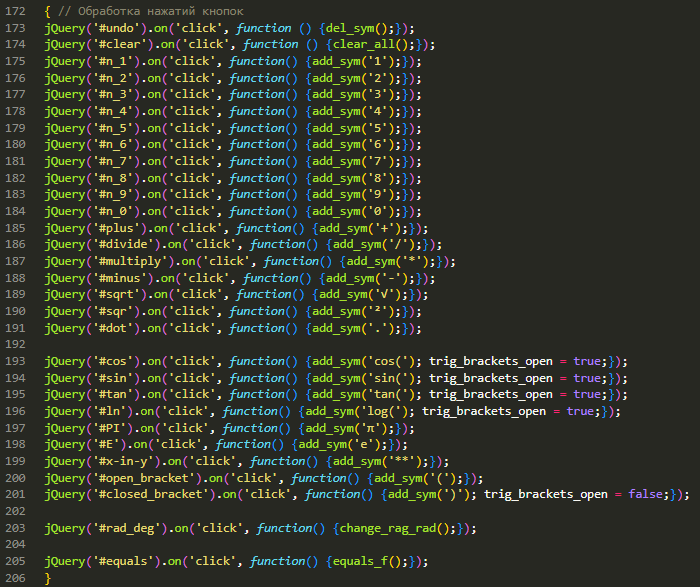
Функции:

Рисунок 18



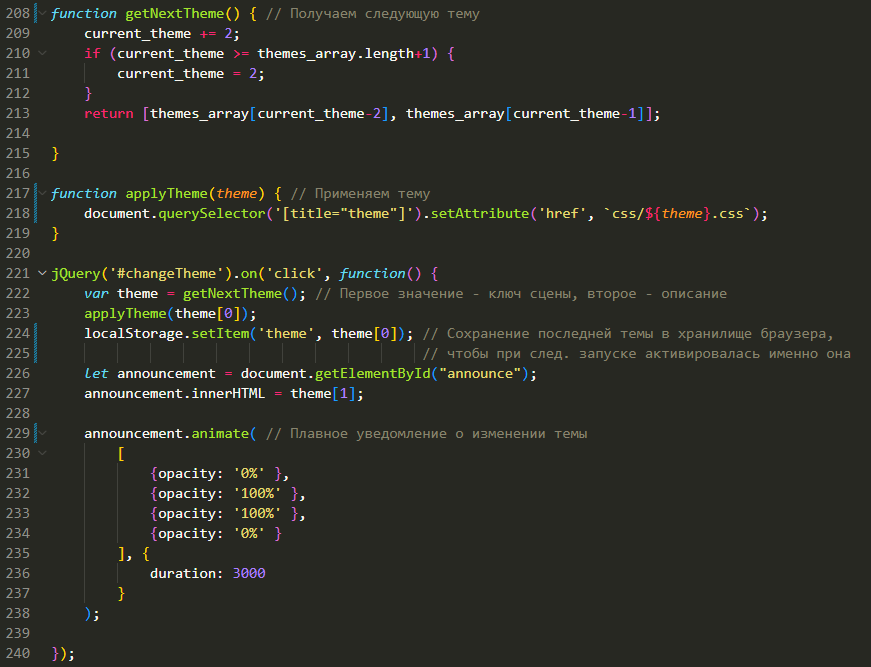
Отслеживание нажатий кнопок:

Рисунок 19



Изменение темы:

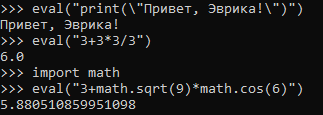
Рисунок 20



Для тем было создано ещё несколько файлов CSS, которые имели изменения для элементов, которые меняются с переключением тем: задний фон, оформление плиток и др.

Перейдём к основному коду на Python. Ему не уделялось особного внимания, т.к. основной задачей было показать, что создать приложение на веб-интерфейсах вообще возможно, это просто и это будет хорошо работать. Основной функцией была eval – встроенная функция python, которая выполняет код языка, полученный из строки. С математическими операциями это так же работает:

Рисунок 21



Но, несмотря на использование eval, нужно было решить несколько проблем: избежать деление на 0 и мусорного символа на конце строки; перевести символы квадрата, корня, Пи, Е в читаемый для программы язык; перевести градусы в радианы, потому что тригонометрические функции в python работают только с ними; расставить знаки умножения между выражениями, которые подразумевают это.

Рисунок 22

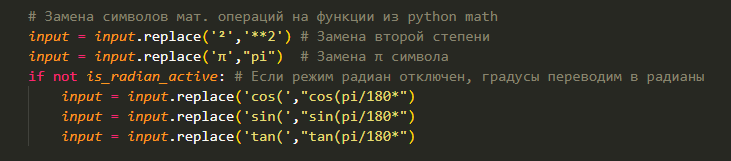


Рисунок 23



Рисунок 24

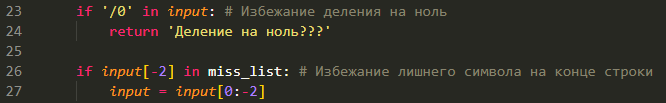


Рисунок 25

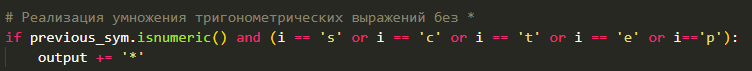
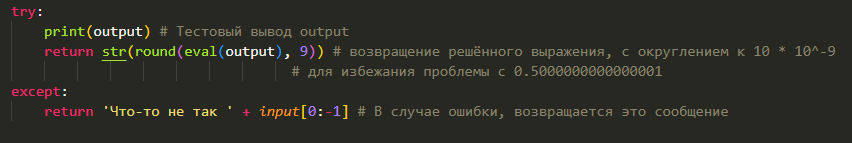


Рисунок 26



Таким образом у нас входная строка проходит некоторые изменения и на выходе получается математическое выражение, которое решает функция eval. Если решить это не получается: из-за неверного ввода пользователя или из-за несовершенного алгоритма растовления скобок, - выводится сообщение об ошибке.

Когда вся программа была готова, осталось превратить проект в самостоятельное приложение, то есть перевести python файл в exe. Для этого была использована библиотека pyinstaller, а точнее auto-py-to-exe (тот же pyinstaller, но с графической оболочкой). Теперь имеем готовое приложение, правда с кучей зависимых папок. Можно распространять его таким образом, а можно создать установочный файл, который самостоятельно перенесёт всё необходимое куда надо пользователю. Для создания единого файла установки, был использован InnoSetup.

**Заключение**

Таким образом мы создали рабочий калькулятор с интересным интерфейсом, которым вполне можно пользоваться в обычной жизни.

Со стороны может показаться, что код на js и python выглядит страшно или непонятно, но на самом деле, разобраться в этом было достаточно легко, поэтому личное мнение – программа на веб-интерфейсах – это легко и элегантно.

На проект было потрачено около 2 месяцев и 70 часов сплошной работы и теперь можно сделать выводы о проделанной работе.

Программа была опробована на 5 компьютерах. И на 1 были найдены неполадки, поэтому предлагаю 3 способа для установки программы. На последующих слайдах они отображены. Самый удобный – 3. Все они были размещены на github. Ссылка на репозиторий - <https://github.com/Obektev/SmartCalc->  
Можете скачать, опробовать приложение, посмотреть на код проекта.

**Итоги:**

1. Изучили HTML, CSS, JS, Python.
2. Написали интерфейс для калькулятора.
3. Сделали его интерактивным.
4. Сделали его полностью рабочим, причём с поддержкой тригонометрических функций.

**Выводы:**

1. Веб-интерфейсы – это легко и красиво.
2. Создание приложения на веб-интерфейсах - быстро и просто.

Так SmartCalc выглядит на момент написания этой теории:

Рисунок 27 Конечный результат



**Использованные ресурсы**

<http://htmlbook.ru/>

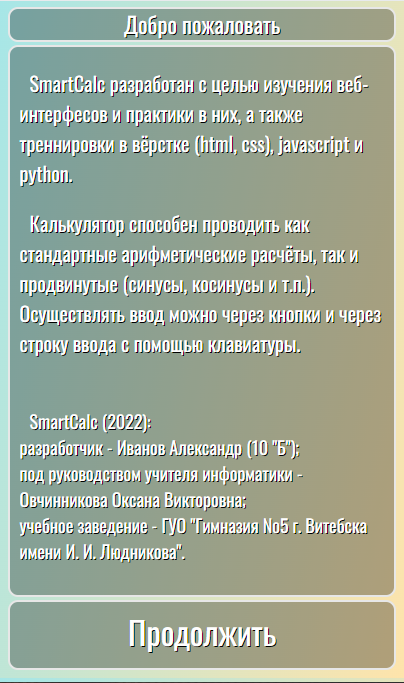
<https://ru.stackoverflow.com/>

https://github.com/python-eel/Eel

**Приложения**

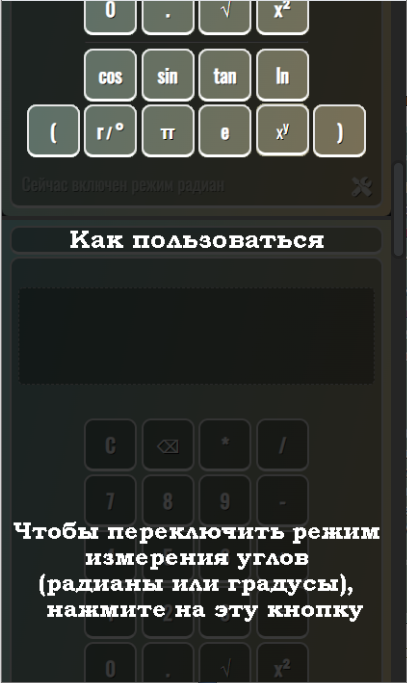
Домашняя страница SmartCalc:

Рисунок Домашняя страница



Страница мануала:

Рисунок Страница с мануалом



Так выглядит сам калькулятор:

