Государственное бюджетное общеобразовательное учреждение города   
Москвы «Школа №185 имени Героя Советского Союза,   
Героя Социалистического Труда В.С Гризодубовой»

**Проектная работа  
«Организатор тегов»**

Подготовил:  
Опарин Григорий,  
ученик 10 И класса.

Руководитель:  
Галганова Д.А.  
учитель информатики.

Допущен к защите: \_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /  
 подпись ФИО руководителя проекта

Москва   
2020

**Содержание**

Введение2

Проблема2

Идея проекта2

Актуальность проекта2

Цель проекта2

Задачи2

Ход работы3

Этапы работы3

Анализ существующих решений3

Язык программирования4

Создание интерфейса4

Работа с базой данных5

Устройство продукта5

Интерфейс программы8

Продукт9

Тестирование продукта9

Заключение10

Приложения11

Приложение 111

Приложение 213

Приложение 314

Список источников15

**Введение**

**Проблема**

В период дистанционного обучения я с моими одноклассниками столкнулся с проблемой: скриншоты, сделанные во время занятия, скапливались в одной папке, и уже через неделю невозможно было найти нужный файл. Проанализировав ситуацию, я разделил эту проблему на три основных части:

1. Папки быстро заполняются изображениями без удобных названий;
2. В обычных файловых менеджерах нет возможности искать изображения по их содержимому;
3. Отсутствие возможности поиска изображений с похожим содержимым.

**Идея проекта**

В Интернете есть довольно много сайтов-галерей с системой тегов, по которой очень удобно находить нужные изображения. Однако иногда такая функция не помешала бы и в режиме офлайн, когда в папке с парой тысяч изображений нужно срочно найти нужное.

**Актуальность проекта**

Продукт данного проекта может быть применим во многих сферах деятельности. Он будет полезен как для использования фотографами, дизайнерами, фоторедакторами и другими специальностями, требующими работы с изображениями, так и для кратковременного использования, например, для сортировки папки со скриншотами.

**Цель проекта**

Создать программу, которая позволяет пользователю присваивать теги изображениям и находить нужные при помощи быстрого поиска.

**Задачи**

1. Ознакомиться с инструментами фильтрации на поисковых сайтах;
2. Освоить необходимые методы создания графических интерфейсов приложений;
3. Научиться создавать и редактировать различные базы данных;
4. Создать программу с изученными инструментами;
5. Протестировать программу на реальных пользователях.

**Ход работы**

**Этапы работы**

Для удобства и экономии времени было решено заранее разбить создание программы на этапы:

1. Изучить наиболее удобные инструменты для быстрого поиска и фильтрации (программы-аналоги);
2. Узнать о необходимых для программы средствах, предоставленных языком программирования Python и дополнительными модулями для работы с графическими интерфейсами;
3. Используя язык программирования Python, создать основу для офлайн-поисковика;
4. Подключить поисковик к базе данных.
5. Добавить изученные инструменты в программу и создать справку по использованию.

**Анализ существующих решений**

В Интернете я нашел несколько программ с похожими функциями, среди них были такие программы, как:

* Adobe Bridge[1];
* JetPhoto Studio[2];
* TagSpaces[3].

Первые две программы довольно большие, и предназначены для более общих задач, вследствие чего ими сложнее пользоваться, и поэтому для решения проблем они не подходят. TagSpaces же, как и другие программы подобного рода, не предоставляет все свои функции бесплатно, из-за чего тоже не подходит

**Язык программирования**

Для написания программы был выбран язык программирования Python[4]. Этот язык не самый подходящий для создания полноценных приложений с пользовательским интерфейсом (изначально и вовсе планировалось использовать C#). Python – это язык программирования, который известен в первую очередь простотой и читаемостью по сравнению с другими языками, что делает его полезным для небольших консольных программ. Кроме того, в официальном репозитории Python находится более двухсот тысяч библиотек, которые позволяют решить любую проблему. К счастью, продукт этого проекта изначально планировался как простое приложение с простым интерфейсом, поэтому известные проблемы низкой производительности интерпретаторов Python не повлияют на результат моего проекта.

**Создание интерфейса**

Для создания дизайна использовалась программа, разработанная специально для программ на базе библиотек Qt и PyQt[5,6]: Qt Designer. Она позволила быстро добавлять недостающие виджеты по ходу создания программы, и в целом ускорила процесс во много раз.

В программу не планировалось добавлять много функций, не относящихся к проекту напрямую, поэтому ее интерфейс можно разделить на три части: основную (в ней находится таблица с информацией о изображениях), боковую панель (в ней находятся поисковая строка, а также инструменты для создания/изменения тегов) и строку меню (в ней находятся элементы управления базами данных, их создание/открытие, а также глобальное изменение тегов осуществляется здесь).

Решено было придерживаться традиционного и простого стиля: светлый фон, стандартные кнопки и шрифт.

**Работа с базой данных[[1]](#footnote-1)**

База данных — совместно используемый набор логически связанных данных (и описание этих данных), предназначенный для удовлетворения информационных потребностей организации[7].

Базы данных (далее – БД) отличаются от других наборов информации следующими признаками[8]:

1. БД хранится и обрабатывается в вычислительной системе.
2. Данные в БД логически структурированы (систематизированы) с целью обеспечения возможности их эффективного поиска и обработки в вычислительной системе.
3. БД включает схему, или метаданные, описывающие логическую структуру БД в формальном виде.

Для работы с базами данных была использована встроенная в Python библиотека sqlite3.

sqlite3 предоставляла широкие возможности для создания и редактирования баз данных на уровне программы, но с ее помощью нельзя было вручную проверить работу кода. Для ручной проверки пригодилась программа SQLiteStudio[9], которая специализируется на удобном просмотре и редактировании баз данных.

**Устройство продукта**

Основной принцип работы программы таков: пользователь выбирает папку с изображениями, после чего программа создает в ней каталог тегов: специальный файл-базу данных с расширением .db. В этом файле хранятся названия, даты создания и размеры всех графических файлов, а также, в дальнейшем, теги, которые пользователь присваивает изображениям.



Данная блок-схема показывает взаимодействие частей программы друг с другом и с внешними источниками информации. Пользователь посылает запрос на файлы с какими-либо фильтрами, либо запрос на открытие изображения. Код посылает запрос с указанными фильтрами в базу данных, принимает в ответ часть данных в виде таблицы и отображает эту таблицу в интерфейсе. Также код может послать запрос изображения прямо в папку, и показать это изображение в новом окне.

Программа[[2]](#footnote-2) состоит из единственного класса TagOrganizer, а также небольшого кода для отображения окна.

Класс TagOrganizer содержит в общей сложности 25 методов. Ниже приведен список всех методов и их описание.

* **\_\_init\_\_():** конструктор класса и инициализатор программы. В нем определяются параметры приложения по умолчанию, такие как способ поиска файлов. Также в нем загружается дизайн приложения и запускается метод connect().
* **connect():** метод для подключения всех виджетов к предназначенным для них методам. Например, кнопка «Открыть…» подключается к методу, который открывает диалоговое окно с выбором папки.
* **create\_file():** метод для создания каталога тегов. Создает диалоговое окно с выбором папки с изображениями, в которой создает каталог тегов и открывает его.
* **open():** метод для открытия существующих каталогов тегов. Также создает диалоговое окно с выбором папки.
* **create\_table():** создает таблицу, отображающую все графические файлы в папке.
* **update\_table():** обновляет таблицу в соответствии с фильтрами и ключевыми словами в поиске.
* **close():** закрывает каталог и очищает все виджеты от набранного текста, тегов и информации об изображениях.
* **quick\_search():** ищет изображения по заданным ключевым словам и настройкам, затем обновляет таблицу.
* **change\_flags():** изменяет настройки поиска в соответствии с поставленными галочками.
* **update\_db():** проверяет папку на наличие новых или измененных файлов, и обновляет базу данных, если таковые имеются. ВНИМАНИЕ: все измененные файлы потеряют свои теги.
* **toggle\_tag\_editor():** при выделении одного или нескольких файлов включает редактор тегов.
* **add\_tags\_to\_image():** добавляет в базу данных теги, добавленные пользователем к изображению.
* **add\_tags\_from\_existing():** выбирает из существующих в базе тегов нужный и присваивает его к изображению.
* **add\_tag():** добавляет тег в базу данных, не присваивая его к какому-либо изображению.
* **edit\_tag():** изменяет имя тега и обновляет его для всех изображений, содержащих его.
* **get\_err():** открывает окно с сообщением об ошибке. Используется во многих выше- и нижеописанных методах.
* **dir\_is\_empty():** проверяет, пустая ли папка, выбранная пользователем, и если да, то сообщает об этом программе и посылает пользователю сообщение об ошибке.
* **open\_image():** при двойном нажатии на строку с информацией об изображении открывает это изображение.
* **update\_add\_combobox():** метод для обновления виджета с выбором тегов из существующих.
* **confirm\_delete():** посылает сообщение с подтверждением удаления чего-либо. Также используется в некоторых других методах.
* **delete\_tag():** удаляет выбранный тег.
* **clear\_db():** очищает все изображения от тегов (оставляя при этом сами теги в базе данных).
* **delete\_all\_tags():** очищает как изображения, так и всю базу данных от тегов.
* **delete\_db():** удаляет всю базу данных целиком.
* **open\_about():** открывает файл about.txt, содержащий полезную информацию о программе.

**Интерфейс программы[[3]](#footnote-3)**

При открытии программы пользователь встречает пустое окно с боковой и панелями. Чтобы начать использование, нужно в меню «файл» нажать на кнопку «Создать…», затем выбрать нужную папку с изображениями. После этого в окне появится таблица с названиями всех изображений в папке.

Чтобы присвоить фотографиям теги, нужно выделить нужные, затем через запятую и пробел написать теги или выбрать теги из уже существующих в этой папке.

В программе присутствует функция быстрого поиска, позволяющая находить изображения по их названию, дате создания или тегам. По умолчанию поиск производится по вхождению слов – то есть если разделять слова пробелами, поиск будет производиться вне зависимости от их порядка. Также поиск производится для файлов, содержащих все включенные в поиск слова. Специальные режимы (включающиеся галочками рядом с поисковой строкой) позволяют изменить способ поиска, например, на поиск точных совпадений с набранной строкой.

Подробнее со всеми функциями можно ознакомиться в файле about.txt, или нажав кнопку «Помощь» в самой программе.

**Продукт**

Продукт представляет собой папку с тремя файлами: main.pyw, proj.ui, about.txt.

В течение месяца (текст написан 23.02.2021) эти файлы также появятся на GitHub по ссылке: <https://github.com/GroX24/school-project/>.

**Тестирование продукта**

Тестирование альфа-версии продукта происходило на занятиях Яндекс.Лицея[10]. Программа получила единогласную положительную оценку пользователей. Большинство опрошенных пользователей быстро разобрались с функционалом программы, пользовательский интерфейс также не вызвал затруднений.

По итогам тестирования потенциальные пользователи дали множество советов по дизайну и устройству программы, некоторые из которых планируется реализовать в ближайшее время. Ниже представлены советы, которые будут реализованы в первую очередь или уже частично реализованы:

* Добавить возможность поиска похожих изображений;
* Добавить различные сортировки файлов;
* Добавить описание всем виджетам и расширить меню «Помощь»;
* Добавить отображение изображений в самой программе.

**Заключение**

В результате работы над проектом была создана программа, которая позволяет пользователю присваивать теги изображениям и находить нужные при помощи быстрого поиска. К сожалению, она решает не все поставленные проблемы: так, нахождение похожих изображений в данный момент находится в разработке.

Тестирование показало, что продукт будет востребован пользователями.

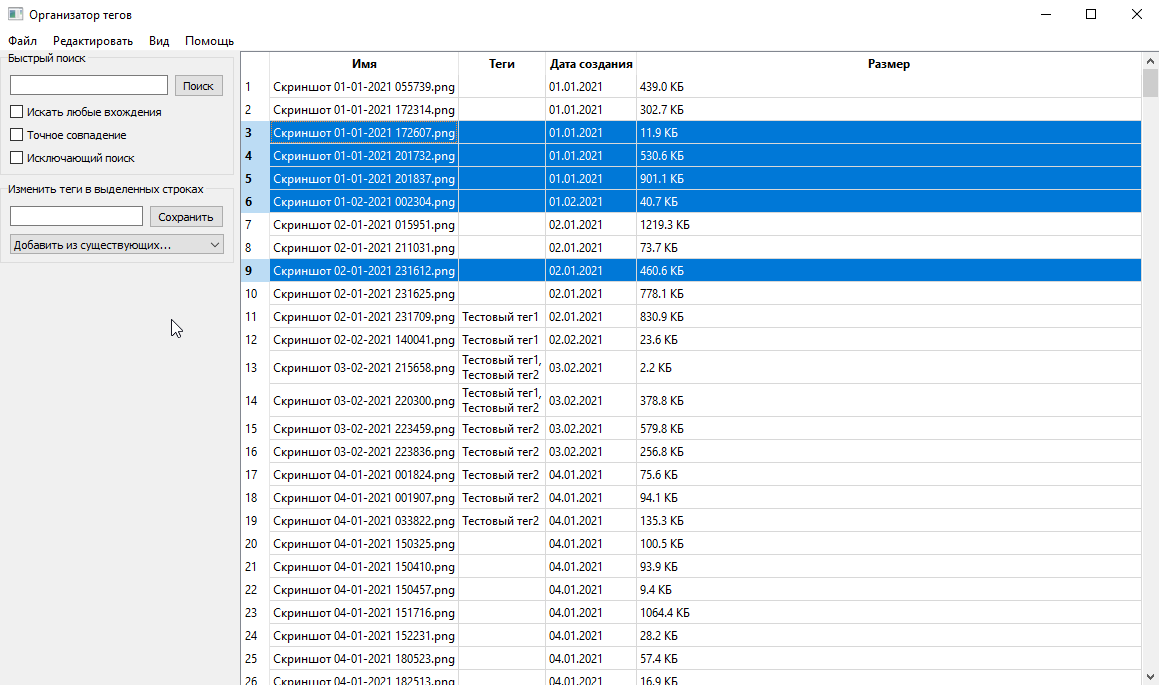
Несмотря на то, что продукт сам по себе является полноценной программой, его все еще можно и нужно улучшать. В очень скором времени планируется выложить продукт на сервис GitHub, добавить в программу функцию автоматического обновления, что позволит продолжить работу над улучшением программы, а также добавит возможность обратной связи.

**Приложения**

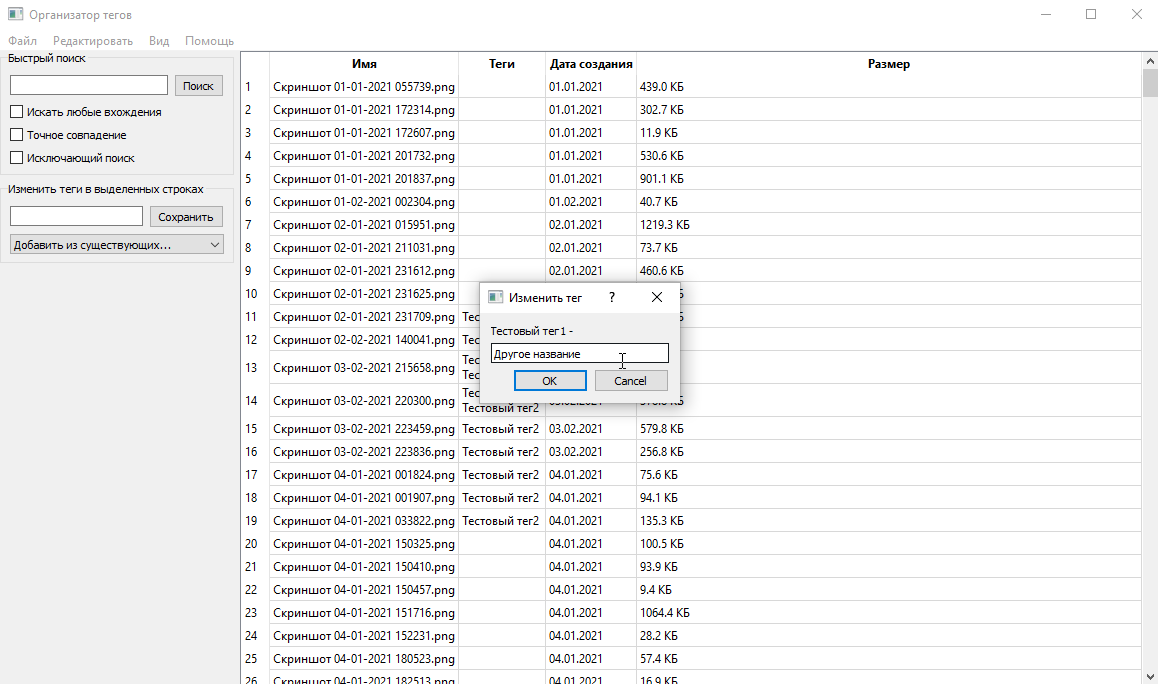
**Приложение 1**



Программа при запуске

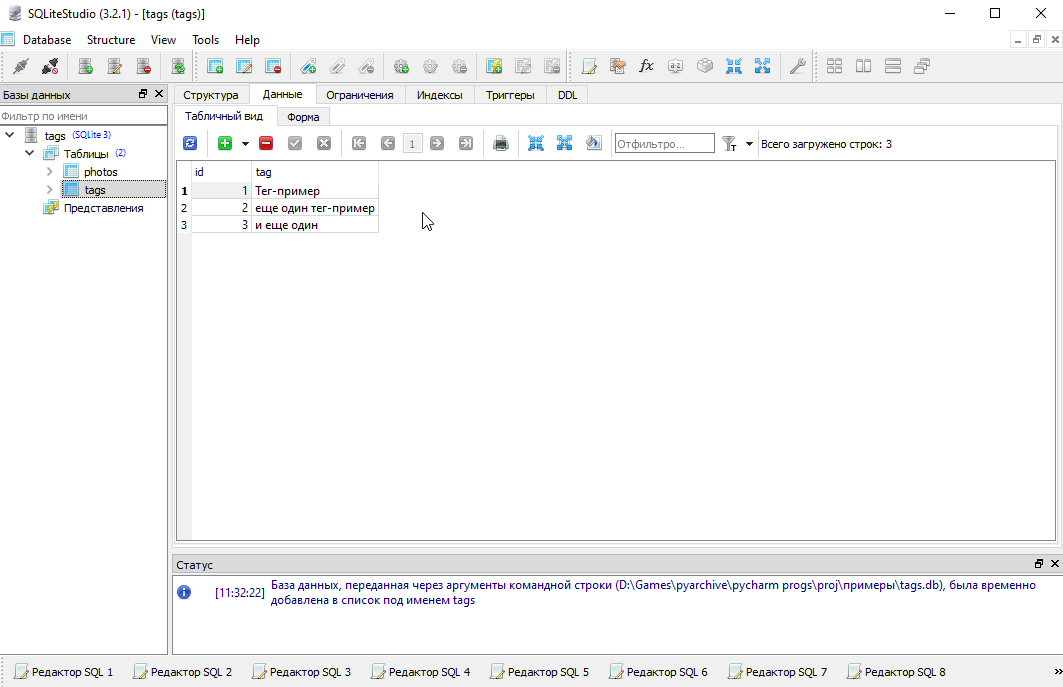


Работа с файлами

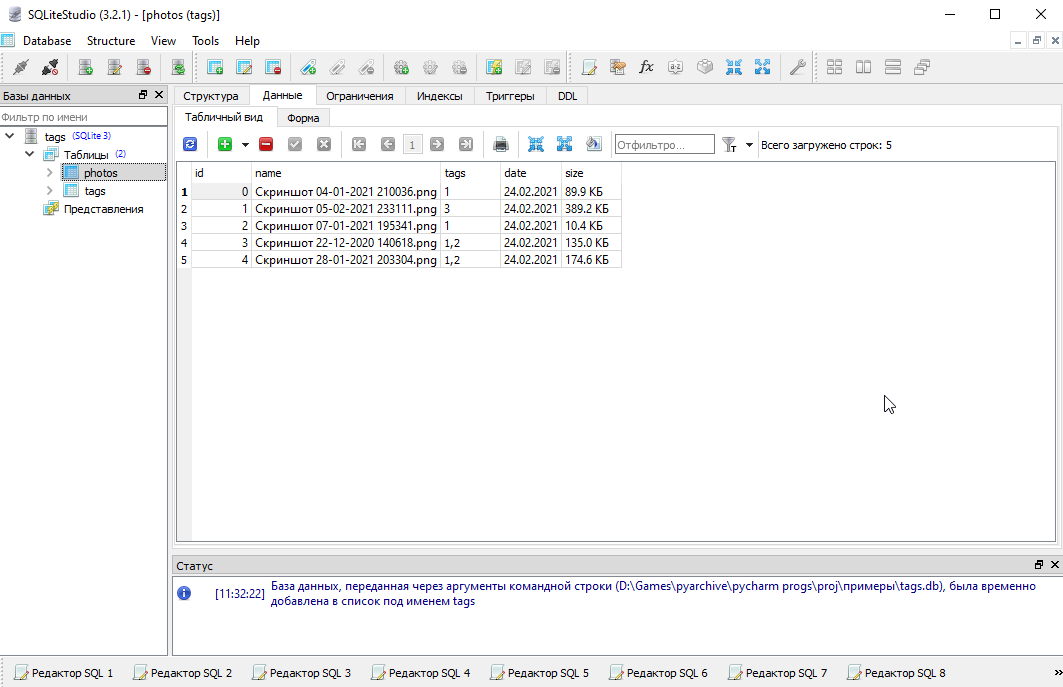


Глобальное изменение тега

**Приложение 2**

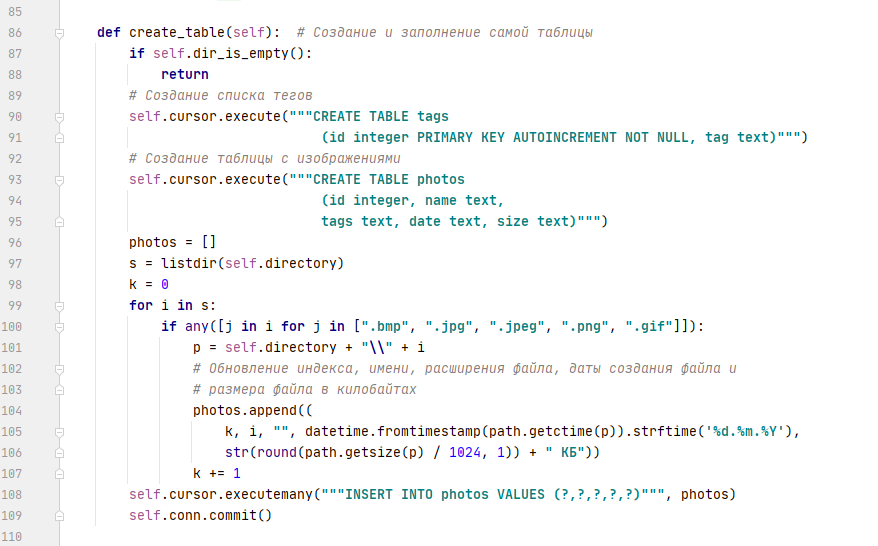


Хранение тегов



Информация об изображениях и тегах

**Приложение 3**



Создание базы данных



Обновление интерфейса. Переменная table - сам виджет

**Список источников**

Официальный сайт Adobe Bridge: <https://www.adobe.com/ru/products/bridge.html/>

Официальный сайт JetPhoto Studio: <http://jetphotosoft.com/>

Официальный сайт TagSpaces: <https://www.tagspaces.org/>

Официальный сайт языка программирования Python: <https://www.python.org/>

Репозиторий Python с библиотекой PyQt5: <https://pypi.org/project/PyQt5/>

Официальный сайт Qt Company: <https://www.qt.io/>

Томас Коннолли, К. Бегг «Базы данных. Проектирование, реализация и сопровождение. Теория и практика», 3-е издание 2003 г. с. 55

Мирошниченко Е. А. «К формальному определению понятия “база даных”», журнал «Проблемы информатики». 2011. № 2. С. 83-87.

Официальный сайт SQLiteStudio: <https://sqlitestudio.pl/>

Образовательный проект Яндекс.Лицей: <https://yandexlyceum.ru/>

1. Скриншоты базы данных - см. приложение [↑](#footnote-ref-1)
2. Скриншоты текста программы – см. приложение 3. [↑](#footnote-ref-2)
3. Скриншот интерфейса – см. приложение 1. [↑](#footnote-ref-3)