Государственное бюджетное общеобразовательное учреждение города Москвы «Школа № 1532»

**Приложение на Python для объединения нескольких экранов компьютеров**

10 класс, ГБОУ Школа №1532,

Воронов Никита Рустамович

Руководитель: учитель информатики, ГБОУ Школа №1532,

Сергиенко Антон Борисович

Москва, 2023

**Оглавление** Введение………………………………………………………………………….3

Цели и задачи работы……………………………………………………………3

Результат………………………………………………………………………….3

Описание………………………………………………………………………….4

Выводы…………………………………………………………………………...12

Список используемой литературы и статей……………………………………14

**Введение**

В современном мире проблема обмена информацией и данными уже давно была решена, мы можем хранить свои личные файлы в облаке, общаться на расстоянии, решать проблемы удаленно. Но можете ли вы быть уверены в безопасности своих файлов?

Цель работы, поставленная с самого начала ─ создать безопасное приложение для обмена данными без вмешательства крупных кампаний, таких как Google, Яндекс, и другие. Конечно, можно использовать их сервисы для наиболее эффективного управления своей информацией, ведь у каждого есть один – два, а может даже больше, аккаунтов на этих платформах. Однако, пока пользователь хранит свои данные у этих корпораций, они могут беспрепятственно просмотреть или скопировать вашу личную информацию, не говоря уже о том, что доступ к вашему аккаунту могут получить посторонние люди. Конечно, вы можете перемещать свою личную и наиболее важную информацию на внешних носителях, но они так же не обеспечивают наибольшую безопасность, к тому же некоторые «похитители информации» уже способны собирать информацию и с них.

Основная задача моего приложения – трансляция экрана пользователя (клиента) на другой компьютер (сервер). Получение данных осуществляется таким образом, что может быть только одно подключение, что защищает от дополнительных ─ внешних подключений.

**Цели и задачи работы**

Создать полностью рабочую версию приложения: реализовать серверную и клиентскую часть. Изучить типы подключения, понять основной принцип обмена данными между двумя устройствами, путем отправки необходимых данных на клиентско-серверную часть.

Не используя готовых изображений, создать уникальный интерфейс для программы. Для этой задачи можно использовать программу Adobe Illustrator.

**Результат**

Было исследовано: каким образом компьютеры могут обмениваться информацией по беспроводной сети с использованием языка программирования Python; каким образом происходит создание приложения на Python, структуры приложения, параметры общей оптимизации приложений.

 Был получен готовый продукт – приложение для объединения двух мониторов. Иконка была нарисована, с использованием программы Adobe Illustrator, которая создана для реализации векторных изображений, но никто не мешает преобразовать нарисованное в формат, который поддерживаться всеми остальными программами.

**Описание**

Вся программа написана на языке программирования Python. Выбор языка программирования обусловлен тем, что python, на момент написания, находится на первых местах по популярности среди самостоятельных разработчиков. Python прост в освоении, это очень может помочь тем, кто захочет усовершенствовать код или убедиться в безопасности работы приложения. У данного языка огромное количество готовых решений различного рода задач, некоторые из них даже реализованы на других языках программирования. Более 137 тысяч библиотек для python позволяют не задумываться о необходимости отдельно прописывать все методы по работе как с данными (json для работы с одноименными данными, NumPy для работы с массивами, SQLite для работы с базами данных, так и с другими программами (Excel для сортировки под удобные таблицы, веб-браузер для работы со внешними источниками, и другие).

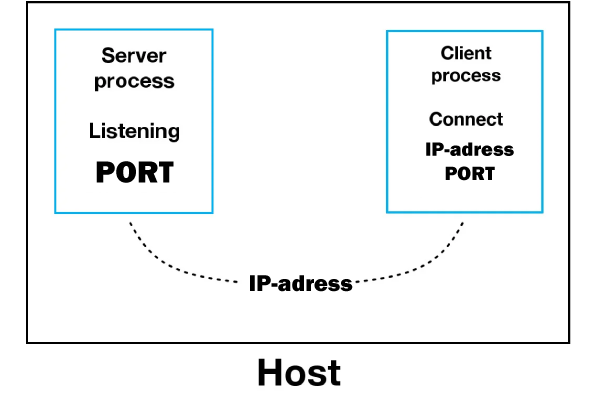
 Для работы в сети python может использовать разные библиотеки, однако если нам требуется работать исключительно с IP-адресами пользователей, то лучший вариант – *socket.* Python socket – это модуль обеспечивает доступ к интерфейсу сокетов BSD (Berkeley Software Distribution). Он доступен на всех современных системах Unix, Windows, MacOS, Linux, и другие. Socket использует IP-адрес и программный порт, принимающий целое число от 0 до 65535, для обмена информацией и данными с системой. Наглядный пример, можно посмотреть на рисунке 1.

Рисунок 1 – наглядный пример, как работает python socket

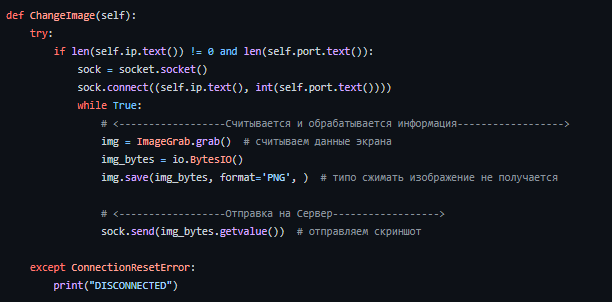
Для визуализации всего программного интерфейса используется одна из популярнейших библиотек по созданию приложений – PyQt5. Это лучший вариант, подходящий для задач, поставленных в проекте. Используя PyQt, можно делать кроссплатформенные приложения, а это может стать огромным преимуществом в разработке приложений для разных пользователей.

Клиентский интерфейс представлен в виде двух строк, в которые можно вписать информацию об IP-адресе и порту подключения, информацию о которых должен выдать *сервер.* Размеры окна, куда требуется вписать значения вычисляются по формуле:

Изображение выглядит как стол

Автоматически созданное описаниеГде S - размер окна в x, y пропорциях, X – ширина экрана, Y – высота экрана.

Рисунок 2 – окно из клиентской части

В окне представлены две полосы, куда нужно вписать основные данные: IP-адрес пользователя и PORT для подключения (по стандарту, в приложении используется порт 9999, так как в большинстве случаев, он не используется основными процессами системы). Так же в окне есть кнопка «Connected», которая служит оповещением программе, что пора начинать обработку данных.

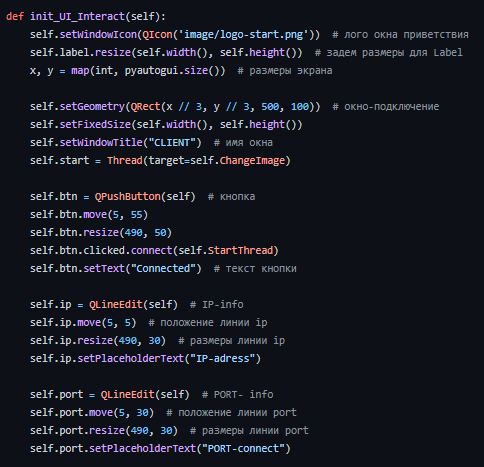
Рисунок 3 – скриншот функции отправки нового потока байт, для смены изображения

Рисунок 4 – функция создания окна для ввода данных

Изображение выглядит как текст

Автоматически созданное описаниеСерверная часть реализована таким образом, чтобы при подключении, сразу же открывалось новое окно, в котором отображается экран *клиента*, использующий “Клиентский интерфейс”. Обмен данными, а конкретно изображение экрана клиента, осуществляется путем создания соединения между клиентом и сервером по протоколу *AF\_INET*. Клиент, используя библиотеку Pillow для работы с изображениями, делает скрин своего экрана в автоматическом режиме и, не сохраняя его на устройстве, отправляет дальше по коду. После, используя библиотеку io, клиент преобразует полученное изображение в поток байтов. Это сделано для того, чтобы можно было отправить изображение любого качества, размера и формата (PNG, JPG, JPEG, и другие). Далее, благодаря беспроводному подключению, программа отправляет исходный поток байтов на сервер. На сервере используется тот же протокол подключения, что и клиент, отличается только то, что вместо отправки информации, он принимает её. После получения определенной команды о то, что подключение установлено, сервер начинает ждать данные. Когда он получил определенную информацию, то начинается процесс по *“досыланию данных”*. Из-за того, что поток байтов может быть очень большим, он может не вместиться в *“один вагон отправки”*, той информации, которой может отправить клиент за один раз, в таком случае нужно разделить исходный поток на несколько поменьше и отправлять уже их. Проверить полноценность потока можно с помощью обычного условия по проверки строки (рисунок 5).

Рисунок 5 – функция приема данных, проверки их на полноценность, заполнение “экрана” полученной картинкой

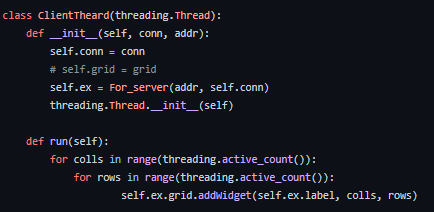
После того как поток байт был полностью доставлен, требуется преобразовать его обратно в изображение, для этого вновь используем библиотеку io, но в этот раз уже сервер делает то же самое, но наоборот – преобразует поток байтов в изображение. Изображение получено, но необходимо его показать. Когда изображение уже полностью преобразовано, сервер запускает новый *поток* для одновременной работы с несколькими подключениями. При этом используется уже встроенная библиотека python – *threading*.

Рисунок 6 – класс для создания нового потока

Создав новый поток, сервер создает окно, размеры которого определяются по формуле:

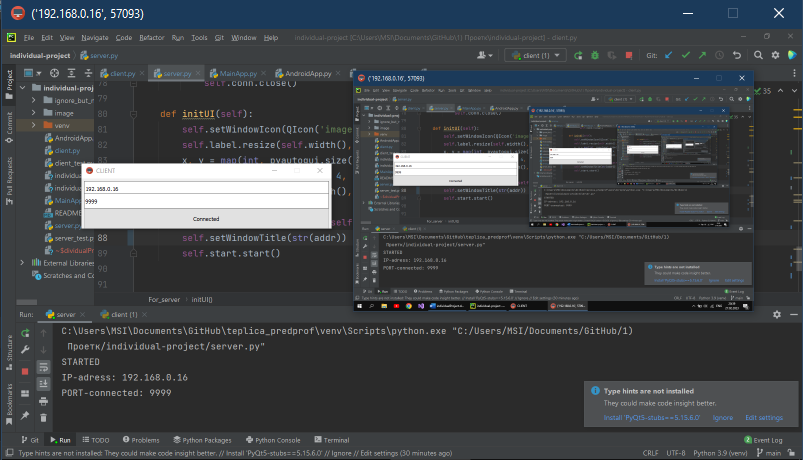
 Где S - размер окна в x, y пропорциях, X – ширина экрана, Y – высота экрана. Данные величина высчитываются с помощью функции *size()* из библиотеки pyautogui, которая создана для работы, как раз, с экранами.

Рисунок 7 – окно серверной части

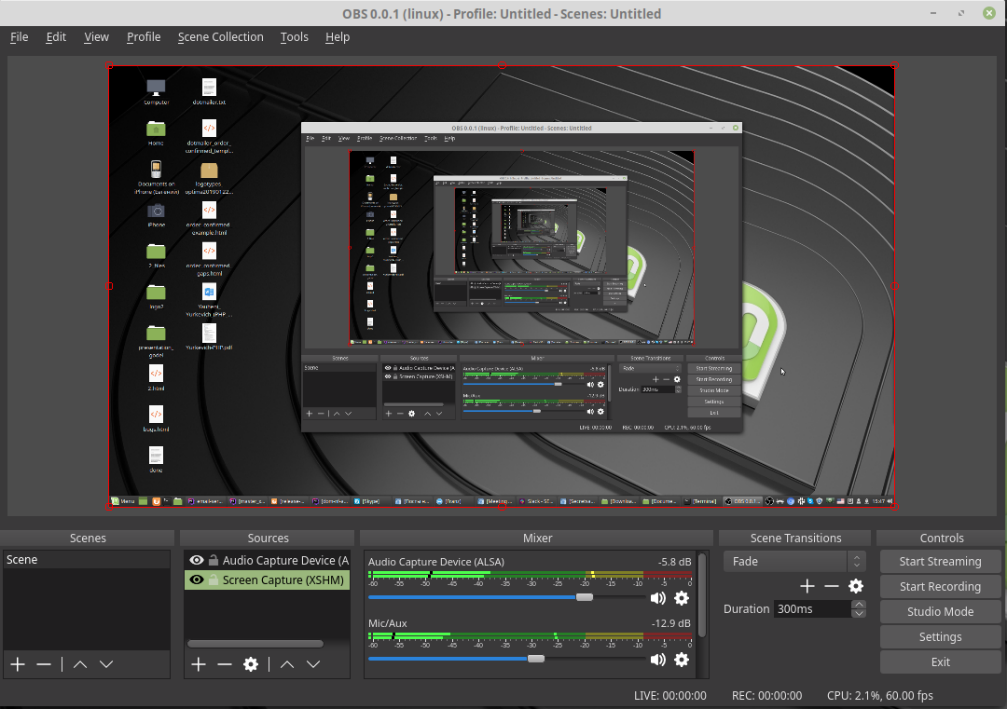
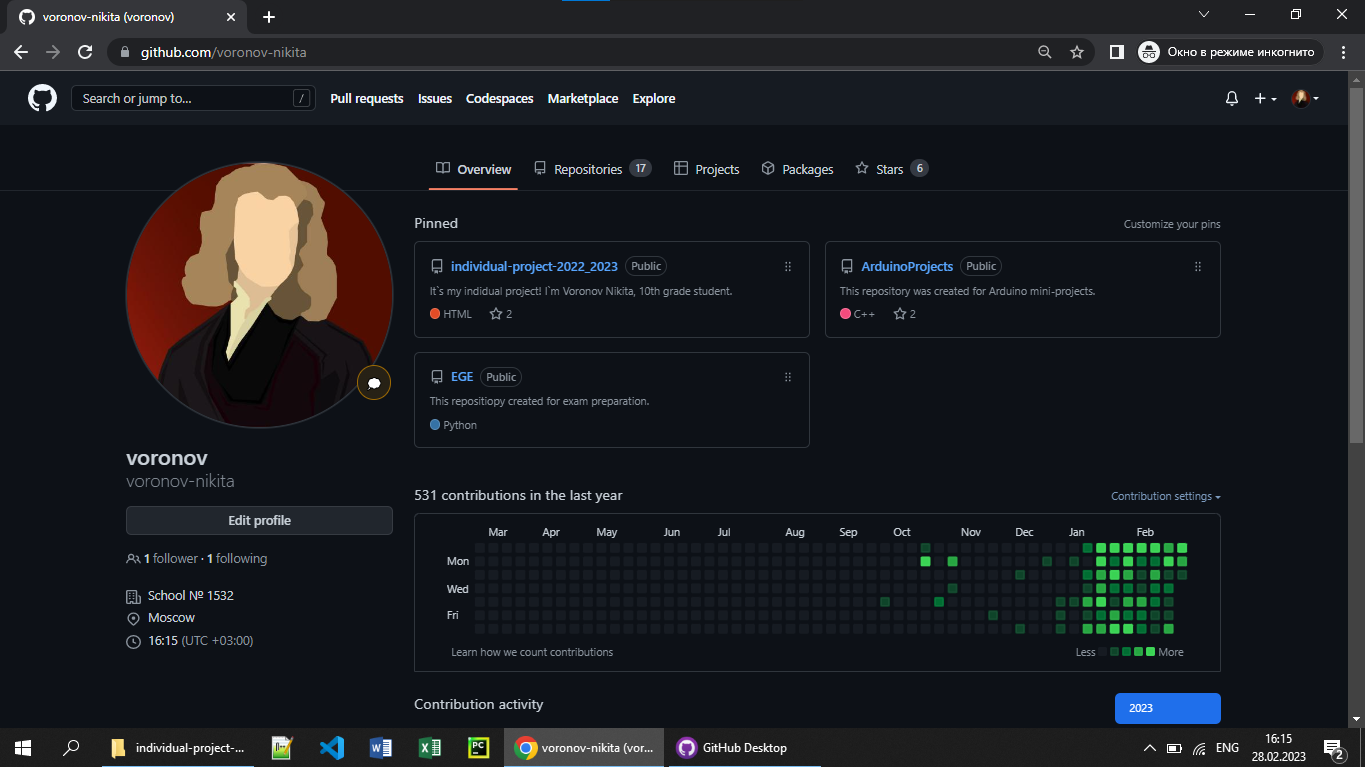
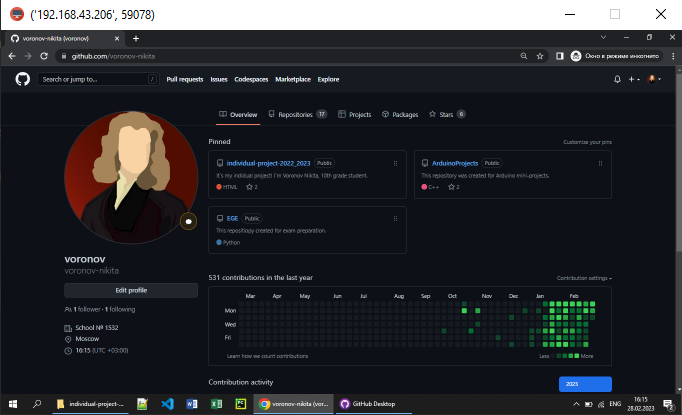
Имя окна – это IP-адрес пользователя и его личный порт обмена информацией (порт присоединения). Если запускать оба приложения на одном устройстве, можно получить похожий эффект, как на рисунке 4. Этот эффект можно назвать рекурсивным. Похожее получается, если запустить OBS-studio, как на рисунке 8.

Рисунок 8 – пример с OBS-studio.

Разумеется, если использовать программы на разных компьютерах, результат будет совершенно другим, не таким эффектным, но тоже, весьма интересным (см. рисунки 5 и 6).



Рисунки 9 и 10 – экран на одном компьютере и активное окно на другом компьютере

На этом долгий путь преобразований и переноса окончен. Но это всего один кадр! Все это должно происходить очень быстро, но процесс будет зависеть от нескольких факторов: скорость подключения беспроводного (или проводного) интернета; суммарная мощность ваших компьютерных комплектующих; используемая операционная система; разрешения от вашего компьютера; и прочее.

**Вывод**

В итоге, путем исследования способов передачи информации по беспроводной сети с использованием языка программирования Python, можно прийти к выводу о том, что особых альтернатив у данного языка нет. Socket, возможно, единственный вариант для наибольшего эффекта и простейшего алгоритма на Python. Компьютеры способны обмениваться данными разного типа ( текст, изображения, и прочее) с помощью потока байт с информацией. При необходимости можно реализовать новую функцию, которая будет переводить исходный файл в поток байт и, по такому же принципу, что описан выше, отправлять его на другой компьютер.

В теории, если двум программам подключиться к единому веб-серверу (один и тот же IP-адрес), размещенному в интернете, то можно увеличить расстояние обмена информацией, но это уже выходит за рамки поставленных задач в данном проекте, так как основная задача – безопасный обмен информацией без вмешательства сторонних источников.

**Список используемой литературы и статей:**

1. Хабр статья - “Сокеты в Python для начинающих” [Статья]

https://habr.com/ru/post/149077/

1. Задорожный Сергей Сергеевич, “Объектно-ориентированное программирование на языке Python” [Текст], М, 2022. -50 стр.

https://clck.ru/33e2VT

1. Тукневич Евгения Иванова, Самоучитель Adobe Illustrator CC [Текст], СПБ, 2014. – 30 стр.
2. Adobe – “Основы рисования в Illustrator” [Статья]

https://helpx.adobe.com/ru/illustrator/using/drawing-basics.html

1. PyQt документация [Статья]

https://doc.qt.io/qtforpython/

1. Python socket документация [Статья]

https://docs.python.org/3/library/socket.html

1. Изучение PyQt5(Python GUI) – серия видео уроков [Видео]

https://youtu.be/HuszbTeETYo