

Разработка программного обеспечения «QDDClient»

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГРУППЫ ИВТ-32 РЗАЕВ А. Э.

Цель

Получение навыков в разработке программного обеспечения с использованием различных технологий программирования.

Разрабатываемое ПО – «QDDClient», настольный клиент для облачного хранилища Dropbox.

Обзор существующих решений

Наиболее популярные:

- Google Диск
- Яндекс.Диск
- Microsoft OneDrive
- Dropbox

Общие черты:

- Наличие веб-интерфейса
- Функции общего доступа к файлам
- Настольные клиенты для Windows и macOS
- API для разработчиков сторонних приложений

Актуальность разработки

Проблема:

Для пользователей Dropbox нет легковесного настольного клиента без синхронизации файлов с бесплатным доступом к хранилищу


Преимущества «QDDClient»:

- Небольшое потребление памяти
- Повышенное быстродействие (нет синхронизации)
- Работает с хранилищем напрямую (Dropbox API)
- Не взимает плату за доступ

Функционал приложения

- Изменение файлов, папок:
 - создание
 - копирование
 - перемещение
 - удаление
- Загрузка файлов в Dropbox
- Скачивание файлов из Dropbox
- Аутентификация с помощью учетной записи Dropbox

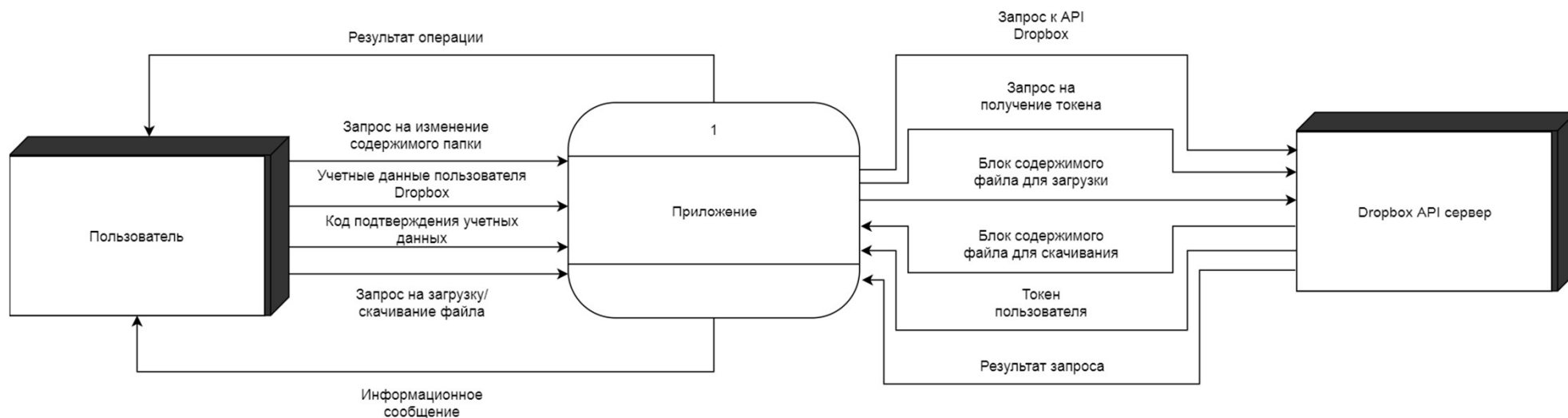
Требования к ПК

- x86-совместимый процессор
 - Не менее 50 МБ ОЗУ
 - Не менее 100 МБ свободного места на диске
 - Доступ к облачному хранилищу Dropbox
- 

Проектирование

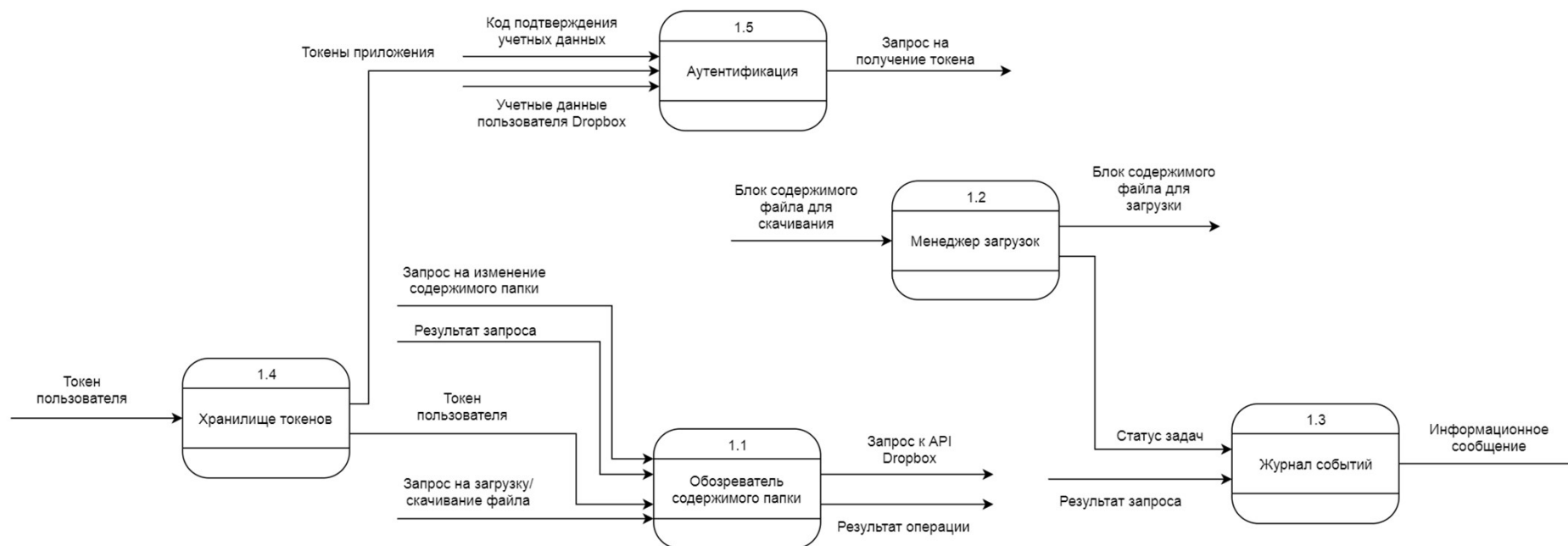
Функциональная структура

Контекстная диаграмма потоков данных



Функциональная структура

Детализация сущностей



Модульная структура



Алгоритмы функционирования

Алгоритм	Функционал
Алгоритм создания папки	Изменение файлов и папок
Алгоритм копирования/вырезки файла/папки	
Алгоритм вставки файла/папки	
Алгоритм удаления файла/папки	
Алгоритм загрузки содержимого папки	
Алгоритм загрузки файла в хранилище	Загрузка файлов
Алгоритм скачивания файла на ПК	Скачивание файлов
Алгоритм аутентификации пользователя	Аутентификация пользователя

Программная реализация

- Язык программирования Python:
 - Поддержка большинства операционных систем
 - Богатая стандартная библиотека
 - Наличие фреймворков для построения GUI
- Фреймворк PyQt:
 - Построение ГПИ
 - Многопоточность
 - Механизм сигналов и слотов

AsyncTask

- Выполнение задачи (`_routine`) в отдельном потоке
- Обработка результата в отдельной функции (`_on_result`)
- Обработка ошибки в отдельной функции (`_on_error`)

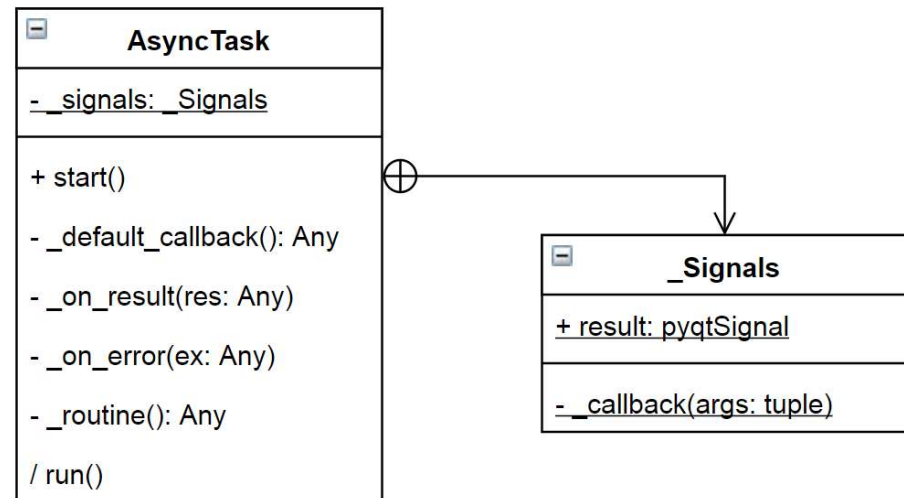
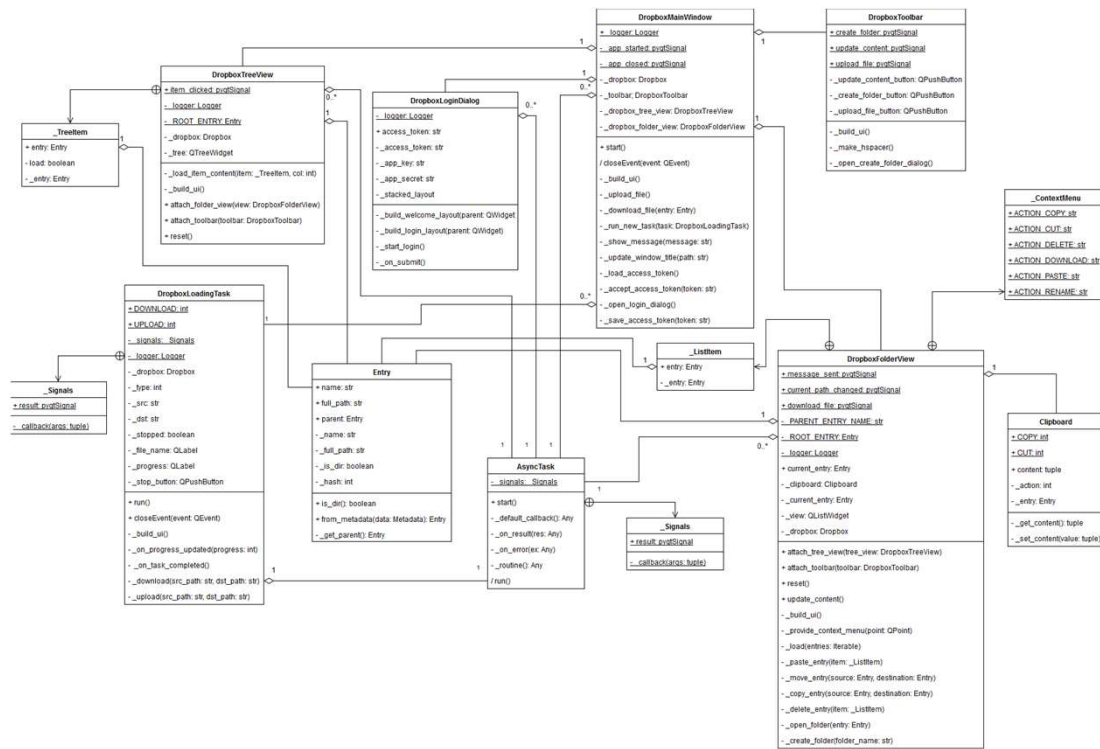


Диаграмма классов



Разработка ГПИ

Общие черты существующих решений:

- Основное пространство страницы – содержимое папки
- Слева – область переходов или дерево папок
- Сверху – путь к папке, различные кнопки

Финальный прототип

Основной экран

<div>"Обновить"</div> <div>"Новая папка"</div> <div>"Загрузить"</div>	
Дерево папок	Содержимое папки

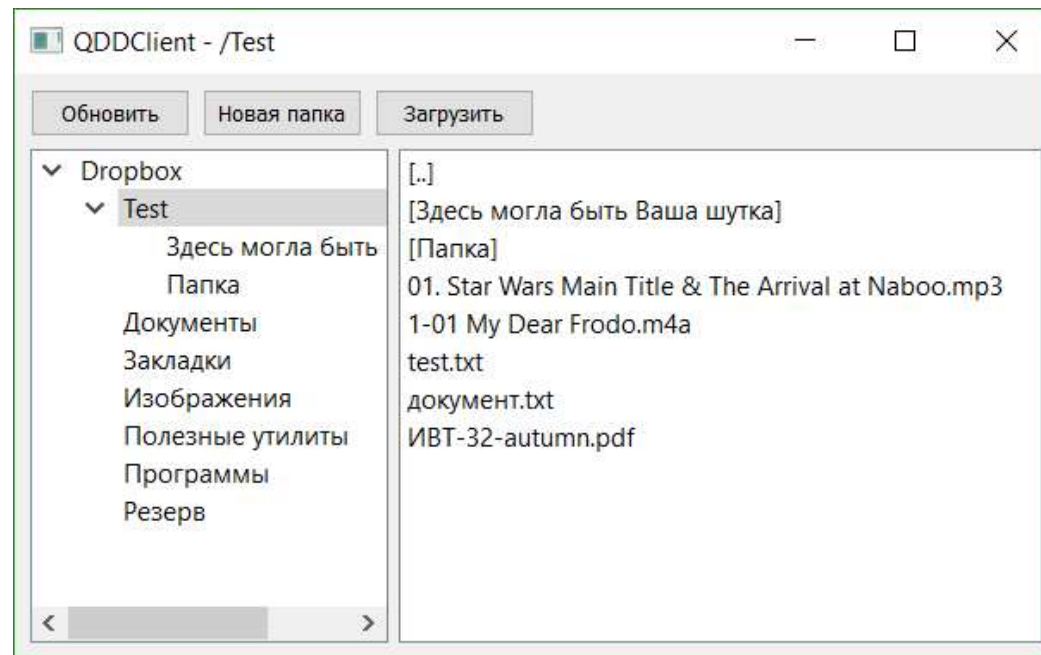
Экран аутентификации

Пояснения, ссылка для перехода
<div>Поле ввода</div>
<div>"Подтвердить"</div>

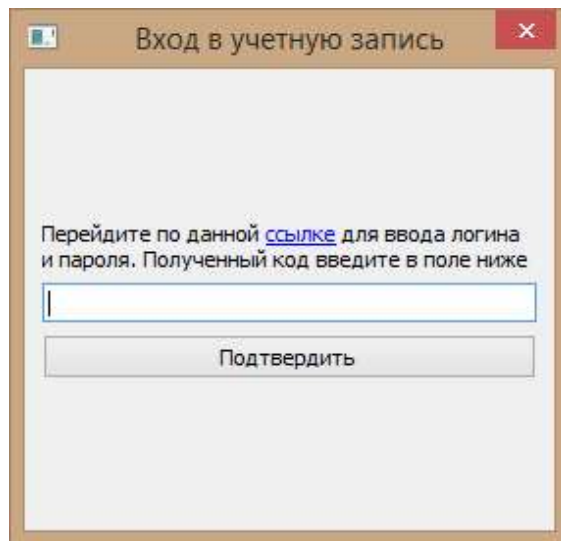
Экран процесса выполнения задачи

Имя файла	Индикатор прогресса	<div>"Стоп"</div>
-----------	---------------------	-------------------

Экранные формы



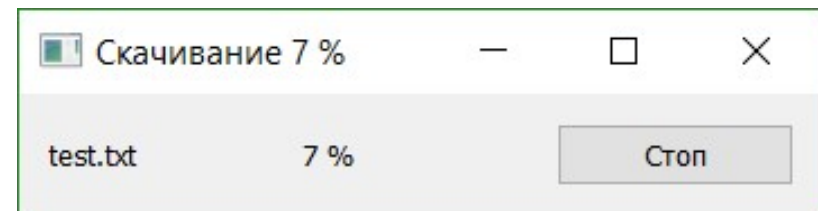
Экранные формы



Вход в учетную запись

Перейдите по данной [ссылке](#) для ввода логина и пароля. Полученный код введите в поле ниже

Подтвердить



Скачивание 7 %

test.txt	7 %	Стоп
----------	-----	------