МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

«Разработка и реализация системного приложения “Менеджер загрузок”»

Текстовый документ курсовой работы

по дисциплине «Операционные системы и системное ПО»

КР. 180009479.ТД

Выполнил студент Селезнёв Н.В.

Факультет, группа ФКФН, ПО(аб) – 81

Руководитель работы Федосеев А.А.

Виза: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(доработать, к защите и т.д.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Хабаровск – 2021г.

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc73947434)

[2 Диаграмма классов 4](#_Toc73947435)

[2.1 Описание классов 4](#_Toc73947436)

[Заключение 10](#_Toc73947438)

[Список использованных источников 11](#_Toc73947439)

[Приложение А 12](#_Toc73947440)

# ПОСТАНОВКА ЗАДАЧИ

Требуется реализовать интерактивное приложение “Менеджер загрузок”, ­ обладающее следующими свойствами: добавление загрузки, возможность поставить на паузу, возобновить, отменить и удалить загрузку. Для разработки приложения будет использован язык программирования Java, библиотека JavaFx и среда разработки IntelliJ IDEA.

# ДИАГРАММА КЛАССОВ

На рисунке 1 рассматривается диаграмма классов.

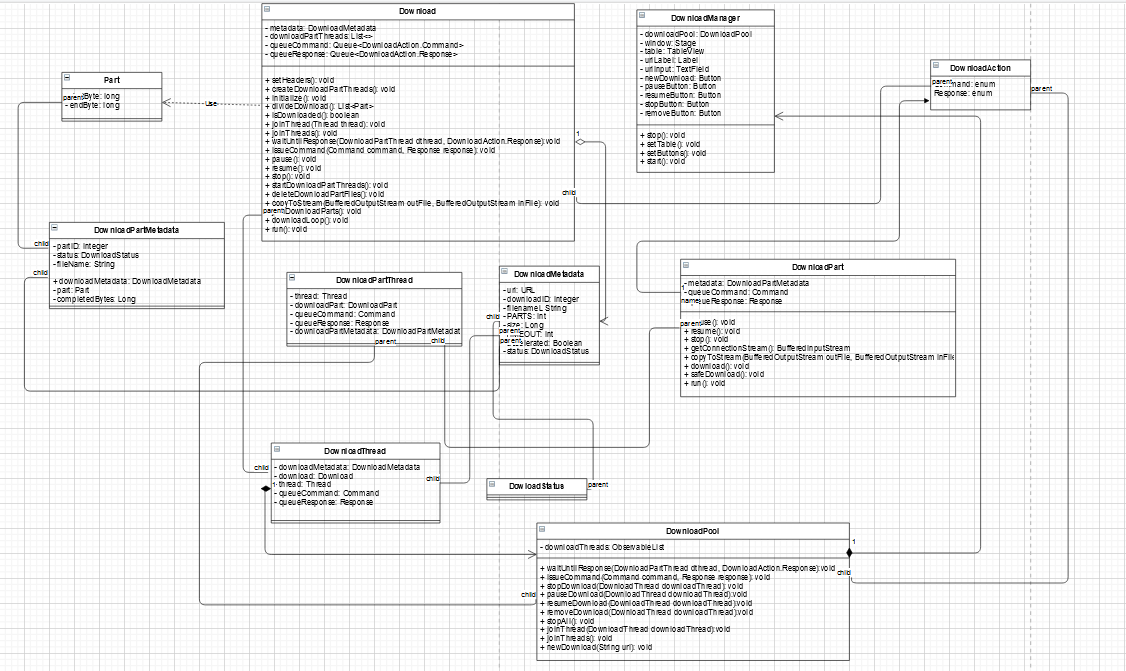


Рисунок 1 - Диаграмма классов

* 1. Описание классов
     1. Download – класс, представляющий объект загрузки. Отвечает за поддержку всех действий над загрузкой и его частями

Методы:

* void setHeaders() – устанавливает заголовок с HTTP запроса. Инициализирует заголовок Accept-Headers;
* void createDownloadPartThreads() – создает потоки для загрузки по частям;
* List<Part> divideDownload() – возвращает список, разделенных на равные части загрузочные файлы;
* boolean isDownloaded() – проверяет все ли части загрузок завершили скачивание;
* void joinThread(Thread thread) – присоединяет поток;
* void joinThreads() – присоединяет потоки;
* void waitUntilResponse(DownloadPartThread dthread, DownloadAction.Response response) – получает response от dthread;
* void issueCommand(DownloadAction.Command command, DownloadAction.Response response) – выдает command и ожидает response;
* void pause() – останавливает загрузку;
* void resume() – возобновляет загрузку;
* void stop() – останавливает загрузку;
* void startDownloadPartThreads() – запускает потоки для раздельной загрузки;
* void deleteDownloadPartFiles() – удаляет файлы частичных загрузок;
* void copyToStream(BufferedOutputStream outFile, BufferedInputStream inFile) – копирует данные из outFile в inFile;
* void joinDownloadParts() – присоединяет все части загрузок после того, как загрузка завершилась;
* void downloadLoop() – цикл, запущенный, пока загрузка не завершена;
* void run() – переопределение метода для запуска класса в отдельном потоке.
  + 1. DownloadAction – класс, представляющий все действия и ответы, которые команда может иметь

Включает два перечисляемых типа:

* Command – перечисление возможных команд.
* Response – перечисление возможных ответов.

2.1.3 DownloadManager – главный класс, запускающий приложение

Поля:

* DownloadPool downloadPool – поле, отвечающее за загрузку файла;
* Stage window – контейнер, используемый для вывода главного окна;
* TableView<DownloadThread> table – таблица, выводящая основную информацию о загружаемом файле.

Методы:

* static void main(String[] args) – метод, запускающий приложение;
* void stop() – переопределение метода, отвечающее за действия после закрытия приложения;
* void setTable() – создание таблицы;
* void setButtons() – настройка кнопок;
* void start(Stage stage) – переопределение метода, отвечающий за обработку главного окна при открытие.

2.1.4 DownloadMetadata – класс, содержащий все метаданные о загрузке

Поля:

* SimpleObjectProperty<URL> url – ссылка на файл;
* SimpleObjectProperty<Integer> downloadID – id загрузки;
* SimpleObjectProperty<String> filename – имя файла;
* SimpleObjectProperty<Long> size = new SimpleObjectProperty<>() – размер файла;
* SimpleObjectProperty<Boolean> accelerated – поле, показывающее поддерживает ли сервер разбиение файла на части;
* SimpleObjectProperty<DownloadStatus> status – статус загрузки.

2.1.5 DownloadPart – класс, представляющий часть загрузки

Поля:

* SimpleObjectProperty<DownloadPartMetadata> metadata – метаданные о части загружаемого объекта;
* ConcurrentLinkedQueue<DownloadAction.Command> queueCommand – очередь команд;
* ConcurrentLinkedQueue<DownloadAction.Response> queueResponse – очередь ответов.

Методы:

* void pause() – приостановление загрузки;
* void resume() – возобновление загрузки;
* void stop() – остановка загрузки;
* BufferedInputStream getConnectionStream() – настройка соединения с загружаемым файлом;
* boolean copyToStream(BufferedInputStream inputStream, BufferedOutputStream fileStream) – копирует данные из inputStream в fileStream;
* void download() – начинает загрузку файла;
* void run() – переопределение метода для запуска класса в отдельном потоке.
  + 1. DownloadPartMetadata – класс, представляющий метаданные о каждой части загрузки

Поля:

* SimpleObjectProperty<Integer> partID – id части загрузки;
* SimpleObjectProperty<DownloadStatus> status – статус загрузки;
* String filename – имя файла;
* DownloadMetadata downloadMetadata – метаданные;
* SimpleObjectProperty<Part> part – размер загружаемой части;
* SimpleObjectProperty<Long> completedBytes – количество загруженных байтов.
  + 1. DownloadPartThread – класс, представляющий поток для загружаемой части файла

Поля:

* Thread thread – поток для загрузки;
* SimpleObjectProperty<DownloadPart> downloadPart – загружаемая часть;
* ConcurrentLinkedQueue<DownloadAction.Command> queueCommand – очередь команд;
* ConcurrentLinkedQueue<DownloadAction.Response> queueResponse – очередь ответов;
* SimpleObjectProperty<DownloadPartMetadata> downloadPartMetadata – метаданные о загружаемой части файла.
  + 1. DownloadPool – класс, отвечающий за загрузку файла

Поля:

* ObservableList<DownloadThread> downloadThreads – список потоков для загрузки.

Методы:

* void waitUntilCommand(DownloadThread downloadThread, DownloadAction.Response response) – метод, получающий response для downloadThread;
* void issueCommand(DownloadThread downloadThread, DownloadAction.Command command) – метод, выдающий command для downloadThread;
* void stopDownload(DownloadThread downloadThread) – останавливает загрузку для конкретного DownloadThread;
* void pauseDownload(DownloadThread downloadThread) – приостанавливает загрузку для конкретного DownloadThread;
* void resumeDownload(DownloadThread downloadThread) - возобновляет загрузку для конкретного DownloadThread;
* void removeDownload(DownloadThread downloadThread) – останавливает и удаляет конкретный DownloadThread;
* void stopAll() – останавливает все загрузки;
* void joinThread(DownloadThread downloadThread) – присоединяет downloadThread;
* void joinThreads() – присоединяет все downloadThread;
* void newDownload(String url) – старт загрузки нового файла по указанному в адресе url.
  + 1. DownloadStatus – класс, представляющий различные состояния загрузки файла
    2. DownloadThread – класс, представляющий поток для загрузки. Включает объект Download и очереди для взаимодействия с потоками

Поля:

* SimpleObjectProperty<DownloadMetadata> downloadMetadata – метаданные о загружаемом файле;
* SimpleObjectProperty<Download> download – объект загрузки файла;
* Thread thread – поток;
* ConcurrentLinkedQueue<DownloadAction.Command> queueCommand – очередь команд;
* ConcurrentLinkedQueue<DownloadAction.Response> queueResponse – очередь ответов.
  + 1. Part – класс, представляющий размер загружаемой части

Поля:

* long startByte – начальный байт;
* long endByte – конечный байт.

# ЗАКЛЮЧЕНИЕ

Выполнение курсовой работы завершилось созданием программного продукта, который обладает функциональными возможностями, определенными заданием.

В приложении А приведены скриншоты, полученные в результате тестирования программы.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Шилдт, Герберт Java. Полное руководство. / Герберт Шилдт. — 11-е изд. — Вильямс, 2018. — 1488 с.

2. Буч, Гради Введение в UML от создателей языка. / Гради Буч, Джеймс Рамбо, Ивар Якобсон — ДМК Пресс, 2015. — 498 с.

3. Schildt, Herbert Introducing JavaFX 8 Programming. / Herbert Schildt. — McGraw-Hill Education, 2015. — 280 с.

4. Брайан, Гетц Java Concurrency in Practice. / Гетц Брайан, Пайерлс Тим, Блох Джошуа, Боубер Джозеф — СПб.: Питер, 2020. — 464 с.

5. Harold, Elliotte Rusty Java Network Programming. / Elliotte Rusty Harold — O’Reilly Media, Inc. 2014. — 506 c.

# ПРИЛОЖЕНИЕ А

(обязательное)

Функционирование приложения

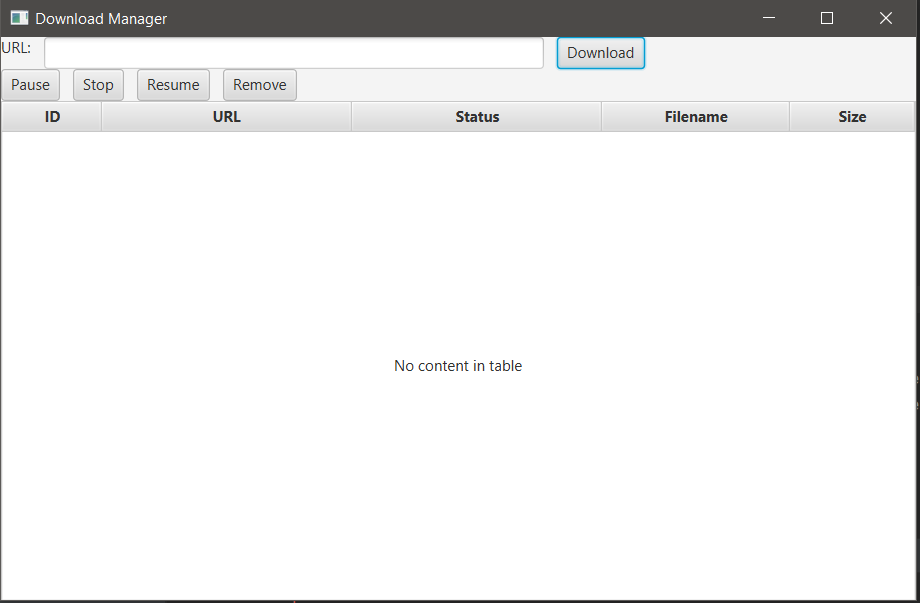


Рисунок А.1 – Основной вид приложения

Продолжение приложения А

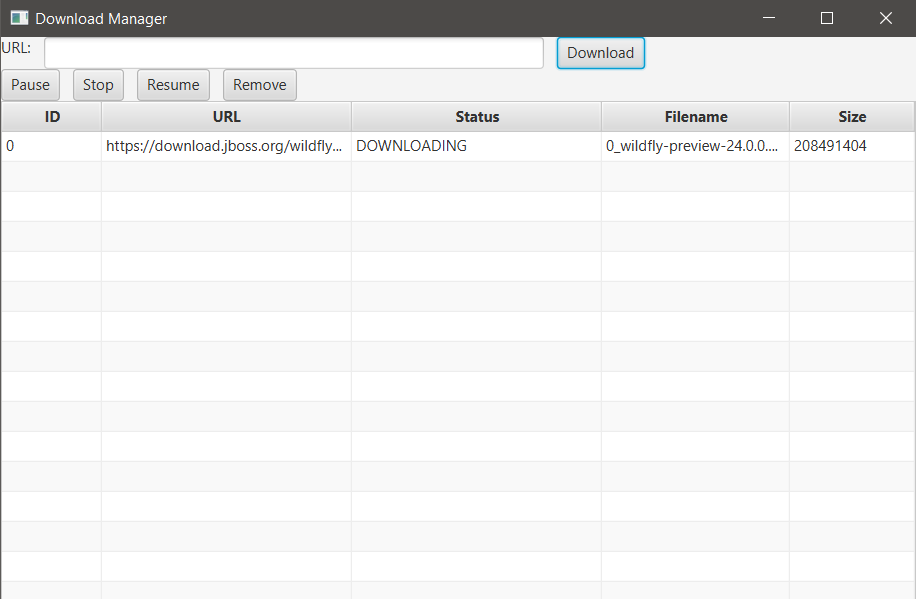


Рисунок А.2 – Демонстрация загрузки файла

Продолжение приложения А

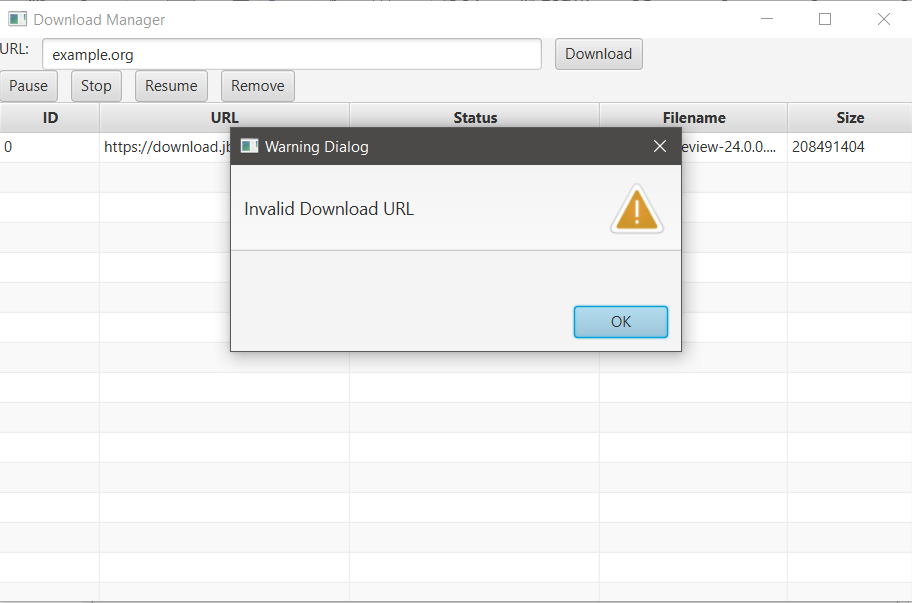


Рисунок А.3 – Демонстрация ошибочной ситуации, в данном случае, если ввести неправильную ссылку .

Продолжение приложения А

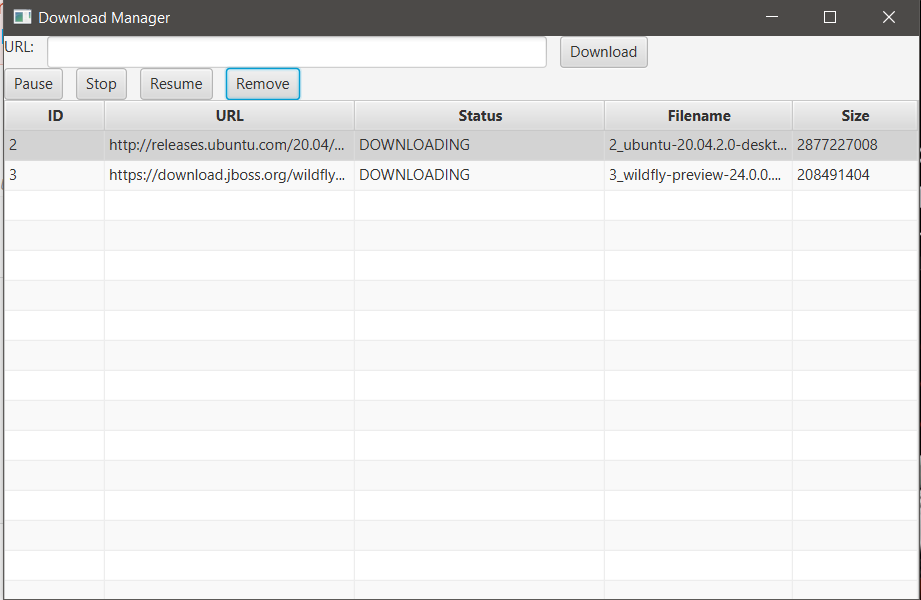


Рисунок А.4 – Демонстрация загрузки нескольких файлов