|  |  |  |  |
| --- | --- | --- | --- |
|  | | для прик эмбл |  |
| МИНОБРНАУКИ РОССИИ | | | |
| **Федеральное государственное бюджетное образовательное учреждение****высшего профессионального образования**"Московский государственный технический университет радиотехники,электроники и автоматики"МГТУМИРЭА | | | |
| Факультет информационных технологий (ИТ) | | | |
| Кафедра информатики и информационных систем (ИИС) | | | |

|  |  |
| --- | --- |
| **КУРСОВАЯ РАБОТА** | |
| **по дисциплине** | |
| **«ОБЪЕКТНО-ОРИЕНТИРОВАНОЕ ПРОГРАММИРОВАНИЕ»** | |
| **Тема курсовой работы «Компьютерная модель сбора статистики музыкальных предпочтений на базе открытого интерфейса социальной сети «Вконтакте»»**    **Вариант № 131237** | |
| Студент группы ИСБО-1-13 | *Жигайков Р.О.* |
| Руководитель курсовой работы | *Хлебников А.А.* |
| Рецензент | *Хлебников А.А.* |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работа представлена к защите | «\_\_»\_\_\_\_\_\_\_201\_\_\_ г. |  |
|  |  |  |
| «Допущен к защите» | «\_\_»\_\_\_\_\_\_\_201\_\_\_ г. |  |

Москва 2015

|  |  |
| --- | --- |
|  | |
| МИНОБРНАУКИ РОССИИ | |
| **Федеральное государственное бюджетное образовательное учреждение****высшего профессионального образования**"Московский государственный университет информационных технологий,радиотехники и электроники"МГТУМИРЭА | |
| Факультет информационных технологий (ИТ) | |
| Кафедра информатики и информационных систем (ИИС) | |
|  | **Утверждаю** |
|  | Заведующий  кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_201\_\_\_ г. |
| **ЗАДАНИЕ** | |
| **на выполнение курсовой работы** | |
| **по** **дисциплине** «**ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ**» | |

1. **Тема «Компьютерная модель сбора статистики музыкальных предпочтений на базе открытого интерфейса социальной сети «Вконтакте»»**
2. **Исходные данные:**

JDK 1.8;

Сторонние библиотеки: google-gson, retrofit;

Целевая-файловая система (log файл);

Описание структуры сторонних ресурсов;

* 1. **Перечень вопросов, подлежащих разработке:**
* Разработать модуль расширения на базе Java Extensible Applications.
* Разработать окружение тестирования.
* Разработать методы для доступа к API музыкальных исполнителей.
* Разработать классы для хранения полученной информации;
* Реализовать логику построения статистики музыкальных предпочтений по полученным данным.

|  |  |  |  |
| --- | --- | --- | --- |
| **4. Срок представления к защите курсовой работы:** **до** « » \_\_\_\_\_\_\_\_ 2015 г. | | | |
|  | | | |
| Задание на курсовую  работу выдал | «\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2015г. |  | ***Хлебников А.А.*** |
| Задание на курсовую  работу получил | «\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2015г. |  | *Жигайков Р.О.* |

**ОТЗЫВ**

на курсовой проект студента группы ИСБ-1-13 *Жигайкова Р.О.*

по дисциплине «*Объектно-ориентированное программирование*» на тему

*«***Создание компьютерной модели сбора статистики музыкальных предпочтений на базе открытого интерфейса социальной сети «Вконтакте»»**

(вариант № 131237)

УДК 004

ББК 32.97

Ж 68

**Рецензент: Хлебников А.А.**

**Жигайков Р.О.** **Курсовая работа по дисциплине «Объектно-ориентированное программирование» на тему: «Создание компьютерной модели сбора статистики музыкальных предпочтений на базе открытого интерфейса социальной сети «Вконтакте»»**

**/**руководитель Хлебников А.А. / М.: МИРЭА, 2015. -

Курсовая работа включает в себя описание разрабатываемой программы, задачей которой является сбор статистки музыкальных предпочтений, описание теоретической части и перечень вопросов к разработке.

Course work includes a description of a program under development, whose task is to collect the musical preferences of Statistics, the description of a theoretical part and a list of questions to develop.

# **Список основных сокращений**

**Id** – идентификационный номер пользователя

**API** - набор готовых [классов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [процедур](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [функций](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [структур](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и [констант](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D0%B0%D0%BD%D1%82%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), предоставляемых приложением (библиотекой, сервисом) для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений.

**HTML** (от англ. *HyperText Markup Language* — «язык гипертекстовой разметки») — стандартный язык разметки документов во Всемирной паутине.

**TCP** - один из основных [протоколов передачи данных](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) интернета, предназначенный для управления [передачей данных](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Сети и подсети, в которых совместно используются протоколы TCP и [IP](https://ru.wikipedia.org/wiki/IP) называются сетями [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP).

**IP**- низкоуровневый протокол, который направляет пакеты данных по отдельным сетям, связанным вместе с помощью маршрутизаторов для формирования Internet или интрасети. Данные "путешествуют" в форме пакетов, называемых IP-дейтаграммами.

**URL** (*Universal Resource Locator)* - универсальный указатель ресурса. URL служит стандартизированным способом записи адреса ресурса в сети [Интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82)

**Реферат**

В курсовой работе выполнено поэтапное проектирование программы по предложенному заданию, разработан интерфейс для общения с пользователем.

Курсовая работа имеет стандартный вид и включает в себя введение, основную часть, заключение, список источников и приложение.

Во Введении сформулирована цель и задачи работы на основе анализа исходных данных.

В основной части пояснительной записки приведены теоретические сведения о функциональных средствах, используемых в разработке, а также перечень вопросов к разработке и пример работы программы.

В заключительной части пояснительной записки дана оценка результатам разработки.

**Article summary**

In the course work done step-by-step design of the program on the proposed assignment, the interface is designed to communicate with the user.

Course work is standard and includes an introduction, main part, conclusion, list of sources and application.

In the Introduction the purpose and objectives based on analysis of the original data.

The main part of the explanatory notes are given theoretical information about functional tools used in the design, as well as a list of questions to the development and an example of the program.

In the final part of the explanatory note assesses the results of development.

**Оглавление**

**Оглавление** 8

**Введение** 9

**Теоретические основы** 10-15

Протоколы 10-12

API 12-13

Логирование 13-14

Сторонние библиотеки 14-15

**Реализация** 16-18

**Пример работы программы** 19

**Заключение** 20

Список использованных источников 21

# **Введение**

На основе анализа исходных данных сформулирована цель и поставлены задачи проектирования.

Цель работы — создание программного обеспечения для сбора статистки сбора музыкальных предпочтений пользователей на базе открытого интерфейса социальной сети. Для достижения заданной цели в работе поставлены следующие задачи:

1. Изучить исходные библиотеки.

2. Изучить API социальной сети «Вконтакте», выявить его недостатки для нашего проекта и провести анализ возможных решений.

3. Реализовать выбранное решение.

**Теоретические основы**

Бот - специальная [программа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), выполняющая [автоматически](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) по заданному алгоритму какие-либо действия через те же [интерфейсы](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81), что и обычный пользователь.

Основной задачей ботов является выполнения работы, однообразной и повторяемой, с максимально возможной скоростью, скоростью, которая очевидно намного превышает возможности человека.

В работе происходит создание бота, который производит сбор статистики музыкальных предпочтений пользователей в социальной сети. Из-за колоссального количества пользователей, человек неспособен быстро и качественно обработать нужную информацию. Потому, использование интернет-бота для этой задачи выглядит разумным решением.

Для работы бота используется пространство всемирной сети интернет, это означает что в этом задействованы сетевые протоколы TCP, IP, HTTP, FTP. Все эти протоколы различаются, и используются для различных целей. К тому же, все они имеют отличные друг от друга уровни сетевой модели OSI.

**Протоколы**

**HTTP**  (*HyperText Transfer Protocol)* - [протокол](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) [прикладного уровня](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D1%8B_%D0%BF%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D1%8F) передачи данных (изначально — в виде гипертекстовых документов в формате [HTML](https://ru.wikipedia.org/wiki/HTML), в настоящий момент используется для передачи произвольных данных). Основой HTTP является [технология «клиент-сервер»](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), то есть предполагается существование потребителей ([клиентов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9))), которые инициируют соединение и посылают [запрос](https://ru.wikipedia.org/w/index.php?title=HTTP-%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81&action=edit&redlink=1), и поставщиков ([серверов](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5))), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTP в настоящее время повсеместно используется во [Всемирной паутине](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D0%BC%D0%B8%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%83%D1%82%D0%B8%D0%BD%D0%B0) для получения информации с [веб-сайтов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82)

Основным объектом манипуляции в HTTP является ресурс, на который указывает [URI](https://ru.wikipedia.org/wiki/URI) в запросе клиента. Обычно такими ресурсами являются хранящиеся на сервере [файлы](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB), но ими могут быть логические объекты или что-то абстрактное. Особенностью протокола HTTP является возможность указать в запросе и [ответе](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%BE%D1%82%D0%B2%D0%B5%D1%82%D0%B0#.D0.92.D0.B7.D0.B0.D0.B8.D0.BC.D0.BE.D0.B4.D0.B5.D0.B9.D1.81.D1.82.D0.B2.D0.B8.D0.B5_.D1.81_.D0.B2.D0.B5.D0.B1-.D1.81.D0.B5.D1.80.D0.B2.D0.B5.D1.80.D0.BE.D0.BC_.28HTTP.29) способ представления одного и того же ресурса по различным параметрам: формату, [кодировке](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D0%B1%D0%BE%D1%80_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D0%BE%D0%B2), языку и т. д. (В частности для этого используется [HTTP-заголовок](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8_HTTP).) Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

**FTP** - или **File Transfer Protocol** — это один из базовых протоколов передачи файлов, предназначенный для передачи файлов в сети между компьютерами. С его помощью можно подключаться к FTP-серверам, просматривать содержимое их каталогов и загружать файлы с сервера или на сервер.

Проткол Http не предназначен для передачи больших файлов, поэтому для этого исользуется протокол FTP.

Протокол FTP построен на архитектуре «[клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80)» и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. Пользователи FTP могут пройти аутентификацию, передавая логин и пароль [открытым текстом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%82%D0%B5%D0%BA%D1%81%D1%82), или же, если это разрешено на сервере, они могут подключиться анонимно. Можно использовать протокол [SSH](https://ru.wikipedia.org/wiki/SSH) для безопасной передачи, скрывающей (шифрующей) логин и пароль, а также шифрующей содержимое.

FTP является одним из старейших прикладных протоколов, появившимся задолго до [HTTP](https://ru.wikipedia.org/wiki/HTTP), и даже до [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP), в 1971 году.

**IP -** маршрутизируемый протокол сетевого уровня. C помощью IP-адресов удалось соединить отдельные компьютерные сети в сеть Интернет.

Протокол IP классифицируется как протокол третьего уровня по сетевой модели OSI, является ненадежным протоколом без установления соединения. Это означает, что протокол IP не подтверждает доставку данных, пакеты могут прийти не в том порядке, в котором были отправлены, продублироваться (приходят две копии одного пакета), оказаться повреждёнными (обычно повреждённые пакеты уничтожаются) или не прийти вовсе. Гарантировать безошибочную доставку пакетов могут протоколы более высокго уровня – например, TCP.

**TCP** (transmission control protocol) - один из основных [протоколов передачи данных](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) интернета, предназначенный для управления [передачей данных](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)

Механизм TCP предоставляет [поток данных](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым, целостность передаваемых данных и уведомление отправителя о результатах передачи. Что можно считать явным преимуществом по сравнению с IP.

Реализации TCP, обычно, встроены в [ядра](https://ru.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B) [ОС](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0). Однако, существуют реализации TCP, работающие в [пространстве пользователя](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F).

Когда осуществляется передача от компьютера к компьютеру через Интернет, TCP работает на верхнем уровне между двумя конечными системами, например, [браузером](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) и веб-сервером. TCP осуществляет надежную передачу потока байтов от одной программы на некотором компьютере к другой программе на другом компьютере (например, программы для электронной почты, для обмена файлами). TCP контролирует длину сообщения, скорость обмена сообщениями, сетевой трафик.

**API**

API (application programming interface) - это интерфейс программирования, интерфейс создания приложений. Если говорить более понятным языком, то API — это готовый код для упрощения жизни программисту. API создавался для того, чтобы программист реально мог облегчить задачу написания того или иного приложения благодаря использованию готового кода (например, функций).

API принимает запросы по методам (GET & POST) и отдаёт не обычный HTML для браузеров, а результат запроса в определённом формате (XML, JSON и др.) Предназначен он скрипту со стороннего сайта/сервиса/программы, который посылает эти GET/POST запросы, получает результат и как-то использует данные. Посылает он запросы естественно не просто так, а чтобы выполнить определённое действие.

API «ВКонтакте» позволяет пользоваться почти всеми функциями данной социальной сети, API необходим для возможного интегрирования в какие-либо программные продукты, многие почтовые ящики, интерент магазины и т.д. интегрируются в друг друга, что создает удобсвто пользователям.

POST – запрос на создание , GET – запрос на получение , DELETE – запрос на удаления – в основном будут использвоаться следуюшие вопросы.

Логика посторения запросов строиться исходя из заданных параметров (или полей), к примеру метод users.get позволяет получить нам расширенную информацию о пользователях. На основе полей происходит конкретезация запросов, то есть мы сможем получить или только номер телефона пользователя, его пол, город, или же всё сразу. Функции и поля позволяют очень гибко создавать запросы и не запрашивать у сервера, то что нам не нужно в данной ситуации.

Например, нужно узнать конкретно дату рождения и город проживания человека с известным нам ID.

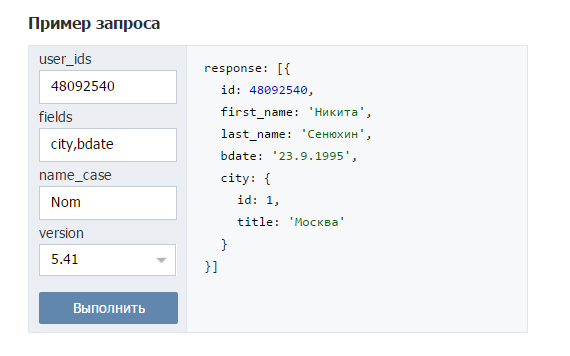


Рисунок 1. Users get

Также стоит упомянуть о LongPoll. Если бы все пользователи постоянно использовали прямые запросы, то сервера не могли бы их корректно обрабатывать и из-за этого создавалась бы уйма лишних проблем, поэтому сейчас используется такое понятие как LongPoll.

LongPoll используется для уменьшения нагрузки на основные сервера, его суть заключается в том, что пользователи не подключены к основному серверу, они находятся как бы на удержании и только если происходит какое-либо событие они посылают запрос уже на основной сервер.

**Логирование**

Файл регистрации или лог (англ. log) — файл с записями о событиях в хронологическом порядке. Различают регистрацию внешних событий и протоколирование работы самой программы - источника записей (хотя часто всё записывается в единый файл).

Логгирование используется для учета всех действий, что происходят в программе, логирование позволяет находить ошибки, вести отчет о простых измененниях в программе и ее работе.

Лог-файлы сервера — специальные файлы, в которых протоколируются определённые действия пользователя или программы на сервере.[[1]](https://ru.m.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB_%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D0%B8#cite_note-1) Например, в лог-файлы веб-сервера записывается информация, откуда пришёл тот либо иной посетитель, когда и сколько времени он провел на сайте, что там смотрел и скачивал, какой у него [браузер](https://ru.m.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) и какой IP-адрес у его компьютера. Каждая запись в лог-файле соответствует определенному [хиту](https://ru.m.wikipedia.org/wiki/%D0%A5%D0%B8%D1%82_(%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82)), так как сервер может фиксировать именно запрос к одному из элементов сайта.

протоколирование, — хронологическая запись с различной (настраиваемой) степенью детализации сведений о происходящих в системе событиях (ошибки, предупреждения, сообщения), обычно в файл. В абсолютном большинстве современных программ используются текстовые файлы протоколов (одно событие — одна строка), они легко генерируются программой и анализируются человеком.

**Сторонние библиотеки**

Оставить программиста без набора готовых дополнительных модулей - получится математик который при решении задачи по стереометрии доказывает каждую теорему заново, или физик, который в простейшей задаче на 3 действия вынужден выводить все формулы.

Обычно все профессиональные системы разработки приложений на языке программирования Java содержат в своем составе набор стандартных библиотечных функций или библиотеки классов. В комплекте со всеми средствами разработки Java поставляются достаточно развитые библиотеки классов, значительно упрощающие программирование.

В Java JDK 1.8 имеется большое количество изначально заданных классов, однако случается, что начальных классов недостаточно для реализации поставленной задачи или решение будет слишком громоздким, поэтому используются сторонние библиотеки, которые позволят увеличить инструментарий разработчика, упростить разработку, сократить её время. Мы использовали определенное количество библиотек, к примеру, такие как Retrofit:

**Retrofit** это библиотека для создания динамических запросов на сервер. Ее преимуществами являются:

1) Больше не нужно выводить запросы к API в отдельный поток в коде – Retrofit это сделает за нас.  
2) Значительно сокращает длину кода и, соответственно, ускоряет разработку  
3) Динамически строит запросы  
4) Автоматически конвертирует JSON в объекты (используется библиотека Gson)  
5) Обрабатывает ошибки  
6) Умеет передавать файлы



Рисунок 2. Пример описания get-запроса



Рисунок 3. Пример описания post-запроса

**google-gson –** библиотека, предназначенная для преобразования Java-объектов в текстовый формат JSON (сериализация) и обратного преобразования (десереализация).

Преимущества **Google-Gson:**

1) Предоставляет простой в использовании механизм для преобразования Java в JSON и обратно

2) Поддержка даже очень сложных объектов  
3) Создает компактный и читаемый JSON вывод

**Реализация**

В работе описанное ранее API «ВКонтакте» использовано будет, но не смотря на все своим преимущества и удобства одного его будет не достаточно. Основной проблемой конкретно для бота тематики «Компьютерная модель сбора статистики музыкальных предпочтений» является то, что большинство неопытных пользователей при добавлении музыки в свои аудиозаписи не заполняют важнейшее для нас поле – жанр. Выглядит это примерно следующим образом:

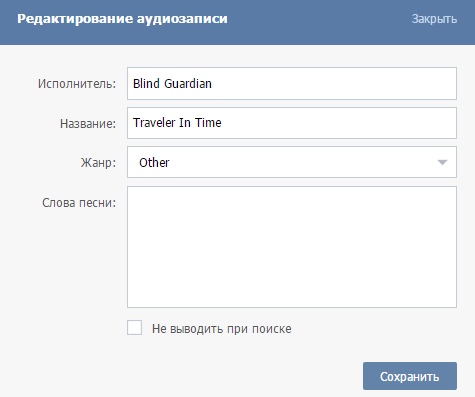


Рисунок 4. Добавление аудиозаписи

Если специально не указывать жанр при добавлении, автоматически выбирается жанр «Other», т.е. «другое». Если изучить аудиозаписи среднестатистического пользователя, можно увидеть, что около 70% всех композиций имеют именно жанр «Other». Учитывая, что бот имеет задачу выдать точную статистику музыкальных предпочтений пользователя, увидеть у каждого «подопытного» 70% жанра «другое» будет весьма неловко, потому, для получения более точной статистики конкретно по жанрам будем использовать <http://developer.echonest.com/>. Но посмотрим по порядку:

Для начала необходимо получить список пользователей, аудиозаписи которых мы хотим проанализировать.

После того, как мы получили список пользователей, необходимо пройти по списку, получить запрашиваемое количество треков и далее получить из треков их исполнителей, для отправки запроса о жанрах на <http://developer.echonest.com/>.

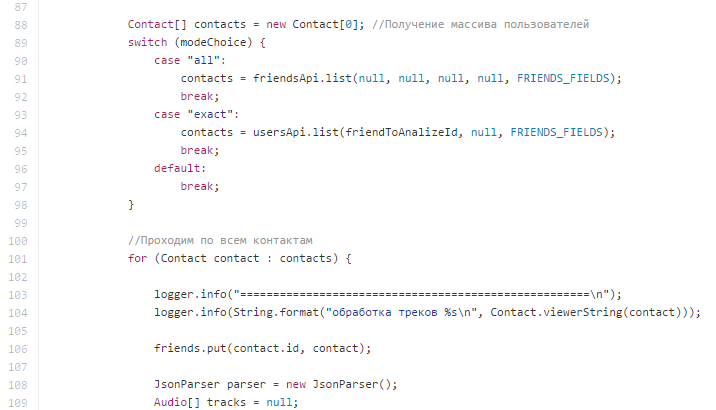


Рисунок 5. Получаем массив пользователей и проходим по их трекам

После того, как получили список жанров с <https://developer.echonest.com/>, присваиваем полученным жанрам «вес». Как правило, творчество каждого исполнителя можно отнести к нескольким жанрам. Внешний API присылает их нам в порядке понижения значимости, поэтому поступаем следующим образом: первый жанр получает коэффициент 0,7; второй и третий 0,4; четвертый и пятый 0,2; 6й и последующие 0,1. Таким образом мы разнообразим статистику и сделаем её более глубокой.

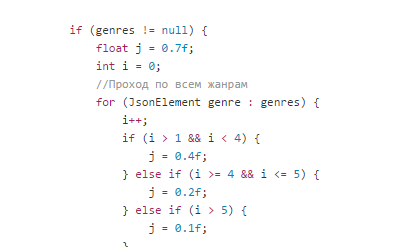


Рисунок 6. Интересный фрагмент кода

После каждого последующего трека, «вес» некоторых жанров увеличивается и так до тех пор, пока запрашиваемое количество треков не будет обработано. Когда программа прошла все песни нужно вычислить % каждого жанра, для получения статистических данных.

# C:\Users\Seton\Desktop\12311331.png

Рисунок 7. Вычисление % жанра от общей суммы

После того как % каждого жанра вычислен, происходит сортировка от большего по нисходящей к меньшему и последующий вывод в файл.

В случае, если нет возможности получить данные о аудиозаписях пользователя, программа напишет нам «Аудиозаписи пользователя недоступны».

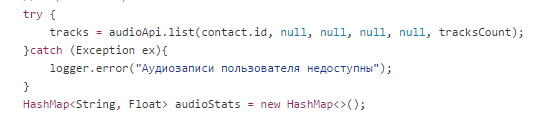


Рисунок 8. Результат ошибки получения списка треков

# **Пример работы программы**

# **C:\Users\Seton\Desktop\обработка конкретного друга.png**

Рисунок 9. Статистика пользователя Адушкин Антон(id38707691)

# **Заключение**

В ходе выполнения курсовой работы была достигнута цель, суть которой заключается в создании программного обеспечения, соответствующим исходным требованиям. В ходе разработки была проанализирована достаточность API Вконтакте для цели нашей курсовой работы. Найдены альтернативные возможности получения жанров музыки.

В результате работы, созданное приложение выполняет поставленную перед ней задачу в создании статистики музыкальных предпочтений пользователей.

# **Список литературы**

1. <http://www.xnets.ru/plugins/content/content.php?content.34>

2.<https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B1%D0%BE%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)>

3. <https://ru.wikipedia.org/wiki/HTTP>

4. <http://www.jino.ru/help/faq/ftp/what-is-ftp/>

5. <http://habrahabr.ru/sandbox/52599/>

6.<https://ru.m.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB_%D1%80%D0%B5%D0%B3%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%86%D0%B8%D0%B8>

7. <http://java-help.ru/retrofit-library/>