# цель работы

Получить навыки разработки фоновых приложений под ОС Android. Научиться работать со сторонними API для организации сетевого приложения в ОС Android.

Разработать приложение для ОС Google Android, представляющее собой фоновый сервис без графического интерфейса. Данный сервис должен осуществлять периодический сбор и выгрузку информации в соответствии с выданным вариантом.

* Выгрузка информации должна осуществляться при наличие любого подключения к сети Интернет и при любом состоянии телефона (включен/выключен дисплей);
* Фоновая работа должна происходить в отдельном потоке ( не в UI thread’е).

Вариант функционала:

Сбор системной информации (версия ОС, версия SDK, свободное место, список установленных приложений, список запущенных процессов, синхронизированные с ОС аккаунты).

Вариант выгрузки:

Выгрузка по протоколу HTTP через GET/POST-запрос с написанием серверной части;

# ход работы

## Добавление разрешений

Для возможности собирать информацию необходимую информацию первым делом были добавлены новые разрешения в файле манифеста (Рисунок 1)

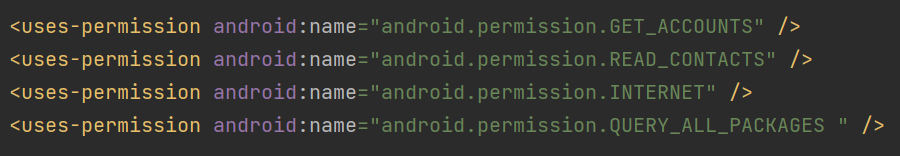


Рисунок 1 – Новые разрешения

* GET\_ACCOUNTS – Разрешает доступ к списку учетных записей в службе учетных записей.
* READ\_CONTACTS – Позволяет приложению считывать контактные данные пользователя.
* INTERNET – Позволяет приложению доступ к выполнению сетевых операций.
* QUERY\_ALL\_PACKAGES – Позволяет получать полный список установленных приложений

## Сбор информации

### OS, SDK

Для получения версии ОС, версии SDK и информации о модели типе и прочих данных об устройстве был использован класс “Build” (Рисунок 2)

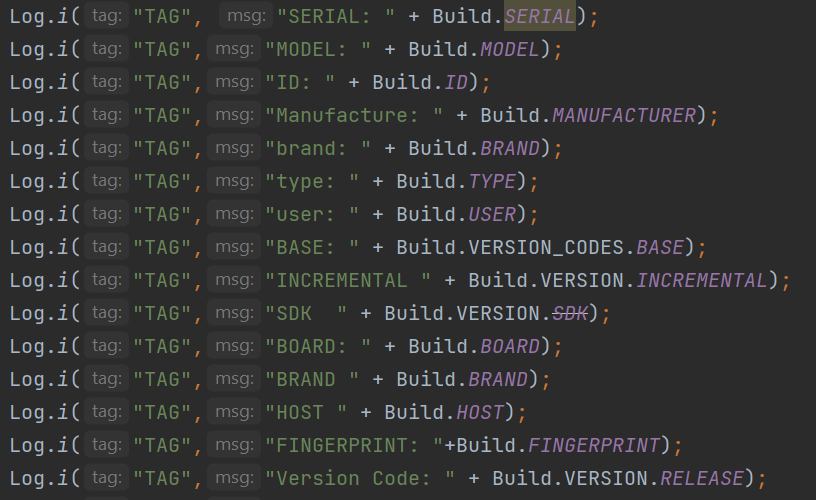


Рисунок 2 – Получение данных

### Количество свободного места

Для получения количества свободного места на устройстве был использован класс “ StatFs” и его методы “ stats.getAvailableBlocksLong()” и “stats.getBlockSizeLong()”. Первый метод возвращает количество свободных блоков, а вторая их размер (Рисунок 3).



Рисунок 3 – Получение количества свободного места

### Установленные приложения

Для получение установленных приложений был использован класс “ApplicationInfo” и его метод “ getInstalledApplications”. Данный метод возвращает список всех установленных приложений (Рисунок 4)

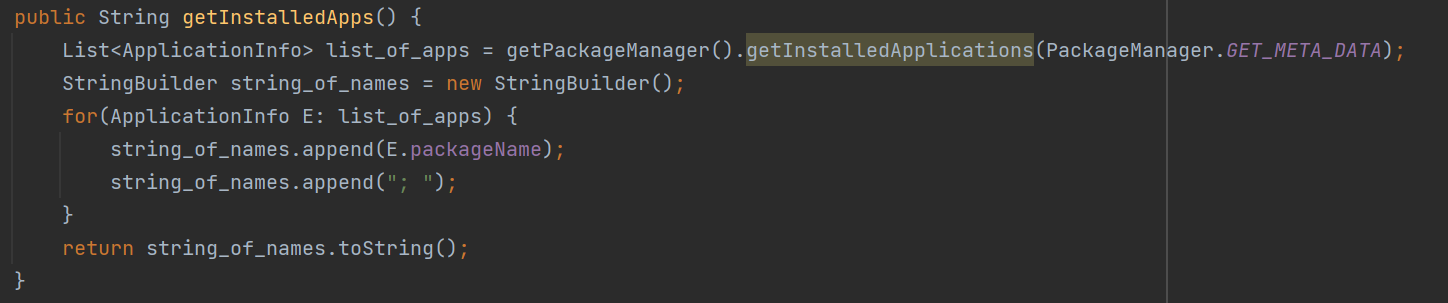


Рисунок 4 – Получение установленных приложений

### Запущенные приложения

Для получение запущенных приложений был использован класс “ ActivityManager” и его метод “ getRunningAppProcesses()”. Данный метод возвращает список всех запущенных приложений (Рисунок 5)



Рисунок 5 – Получение списка запущенных приложений

## Отправка данных на сервер

Для отправки данных на сервер был использован класс “ HttpURLConnection”, и его метод “ openConnection()”. Данный метод создает HTTP соединение с сервером. Для отправки был использован метод “write()”, класса “OutputStream” (Рисунок 6).



Рисунок 6 – Отправка данных на сервер

## Реализация сервера

Для получения данных был написан сервер на языке “Python”. Первым делом был создан класс “ MyServer”, который наследуется от класса “ BaseHTTPRequestHandler”. После чего был реализован метод “do\_POST”. Реализованный метод при получении “post” запроса отвечает кодом 200 (“HTTP\_ok”) после чего обрабатывает полученные данные (Рисунок 7)

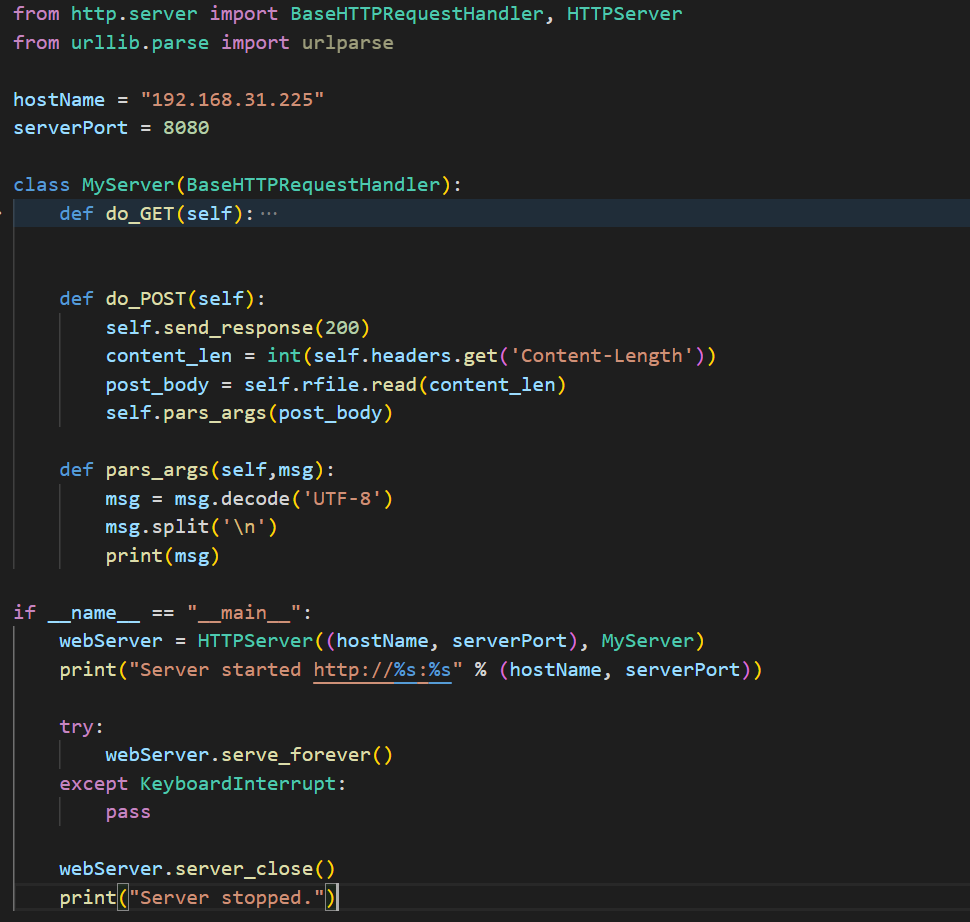


Рисунок 7 – Реализованный сервер

## Полученные результаты

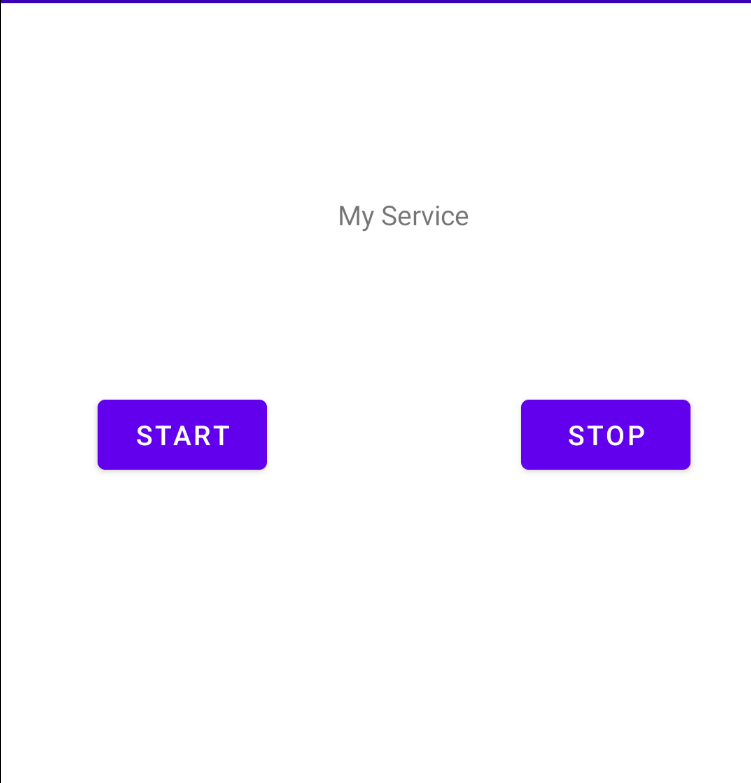


Рисунок 8 – Запущенное приложение

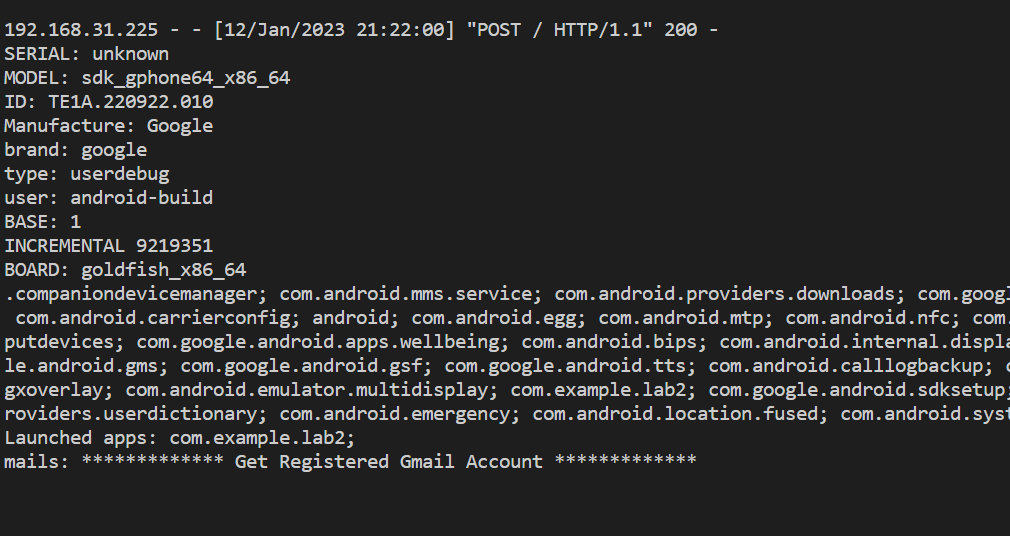


Рисунок 9 – Полученные данные на сервере