ЛАБОРАТОРНАЯ РАБОТА. РАБОТА С LINUX из ANDROID.

**Цель**. Изучить доступ к Linux из программы Java Android.

Введение.

Использование команд Shell позволяет глубоко «проникнуть» в ядро Linux. В данной работе мы именно ставим такую цель.

Создадим следующий файл разметки

<LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"

    xmlns:tools="<http://schemas.android.com/tools>"

    android:layout\_width="match\_parent"

    android:layout\_height="match\_parent"

    android:orientation="vertical"

    android:layout\_gravity="center"

    tools:context=".MainActivity" >

    <TextView

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content"

        android:gravity="center"

        android:textAlignment="center"

        android:text="Shell Executer" />

    <EditText

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content"

        android:id="@+id/txt"

        android:hint="Example - ls" />

    <Button

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content"

        android:id="@+id/btn"

        android:text="Execute" />

     <TextView

        android:layout\_width="fill\_parent"

        android:layout\_height="wrap\_content"

        android:id="@+id/out"

        android:textAlignment="textStart" />

</LinearLayout>

В работе будем использовать два класса. Наряду с главной активностью добавим следующий класс ShellExecuter (для исполнения команд Linux).

package com.learn2crack.androidshell;

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class ShellExecuter {

     public ShellExecuter() {

        }

public String Executer(String command) {

            StringBuffer output = new StringBuffer();

            Process p;

            try {

                p = Runtime.getRuntime().exec(command);

                p.waitFor();

                BufferedReader reader = new BufferedReader(new InputStreamReader(p.getInputStream()));

                String line = "";

                while ((line = reader.readLine())!= null) {

                    output.append(line + "n");

                }

            } catch (Exception e) {

                e.printStackTrace();

            }

            String response = output.toString();

            return response;

        }

}

Команда передается как параметр command. Результат выполнения команды записывается в переменную response.

Класс главной активности имеет такой вид:

package com.learn2crack.androidshell;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.app.Activity;

import com.learn2crack.androidshell.ShellExecuter;

public class MainActivity extends Activity {

    EditText input;

    Button btn;

    TextView out;

    String command;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        input = (EditText)findViewById(R.id.txt);

        btn = (Button)findViewById(R.id.btn);

        out = (TextView)findViewById(R.id.out);

        btn.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View arg0) {

                // TODO Auto-generated method stub

                ShellExecuter exe = new ShellExecuter();

                command = input.getText().toString();

                String outp = exe.Executer(command);

                out.setText(outp);

                Log.d("Output", outp);

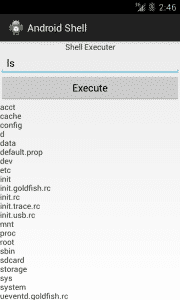
            }

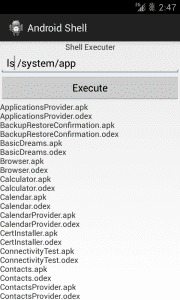
        });

    }

}

Манифест файл не требует дополнительных изменений. Примеры скриншотов, получаемых для разных команд Linux, приведены ниже:

[](https://www.learn2crack.com/wp-content/uploads/2014/03/device-2014-03-30-201606.png)

[](https://www.learn2crack.com/wp-content/uploads/2014/03/device-2014-03-30-201741.png)

Пример рабочего экрана для команд *ls* и *date* приведен на рис. 1.

|  |  |
| --- | --- |
| [Описание: Android Executing Shell Commands - Example](https://www.learn2crack.com/wp-content/uploads/2014/03/device-2014-03-30-201606.png) |  |

*а* *б*

Рис. *1*. Результат работы команды Linux: *а* − *ls*; *б* − *date*

Заметим, что многие команды в этом варианте не срабатывают либо требуют прав суперпользователя (*su*).Срабатывают команды: *ps* (информация о процессах), *du* (информация о размере занимаемой памяти системными папками), *df* (информация о системных файлах), *uptime* (информация по времени работы системы и простоя), *cat* /*proc/version* (вывести версию ядра), *cat /proc/mounts* (отобразить смонтированные файловые системы), *cat /proc/cpuinfo* (выдать информацию о процессоре), *cat /proc/meminfo* (выдать информацию об использовании памяти), *ls /system/app* (вывести установленные приложения) и некоторые др.

Задания для выполнения

* 1. Выведите все установленные приложения в список *ListView* (на консоль, если работаете в *С*).
  2. Из даты / времени возвратите только время.
  3. Выведите модель процессора.
  4. Выведите число работающих в системе процессов.
  5. Определите размер свободной памяти системы.

## 

Вопросы для самоконтроля

1. Как можно обратиться к ядру *Linux*?
2. Объясните использование ядра *Linux* из программы на *Java Android*.
3. Для чего нужна команда *chmod*?
4. Что делает команда *cat*?
5. Какая команда используется для создания файла?