TestPing

Software Documentation

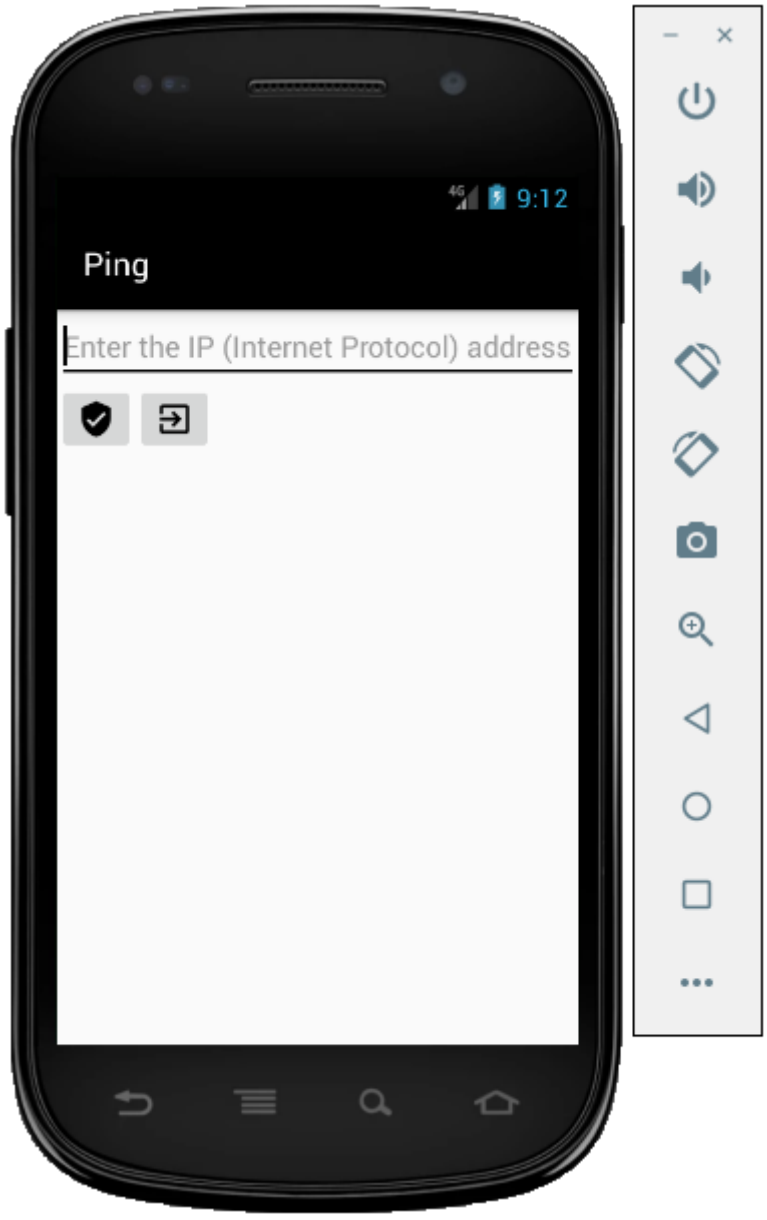
Author: matiwa

Table of contents

Introduction

This software documentation includes: description of the application's operation, what is needed for use, algorithms used, interface description and source code description. This application is used to diagnose network connections with the Ping command.

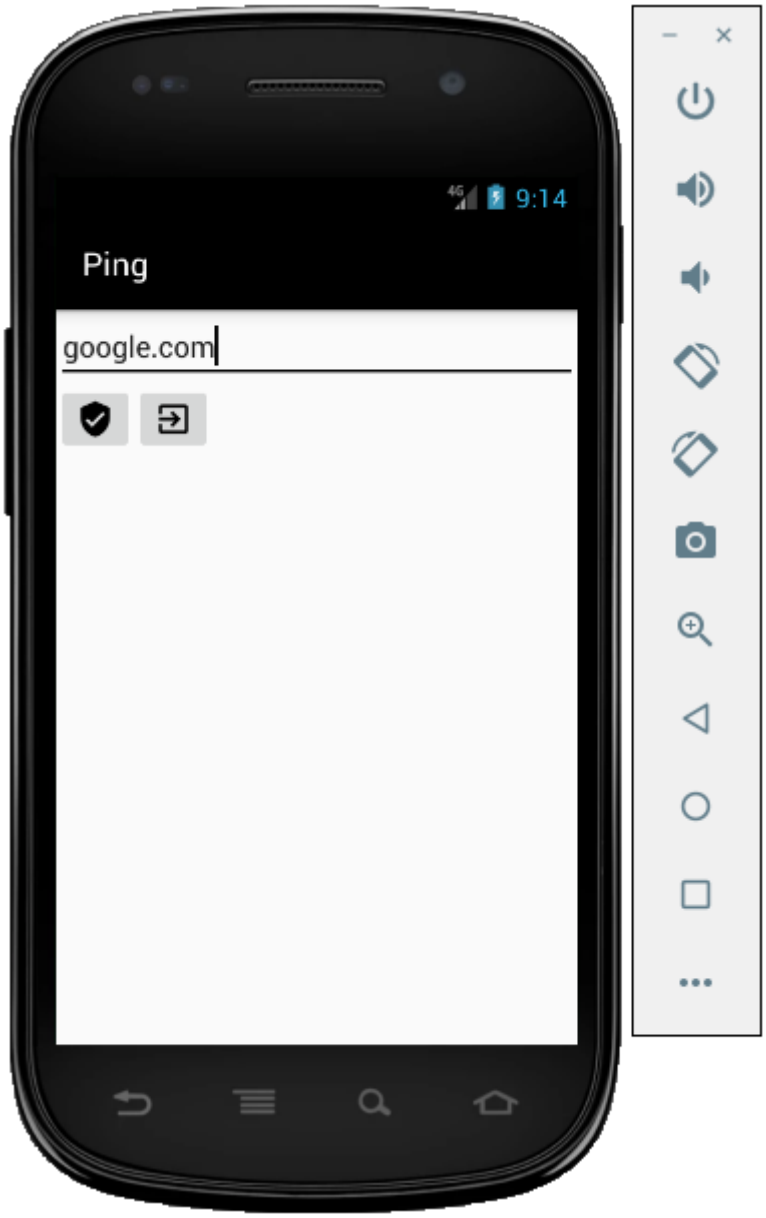
Describing of the application’s operation



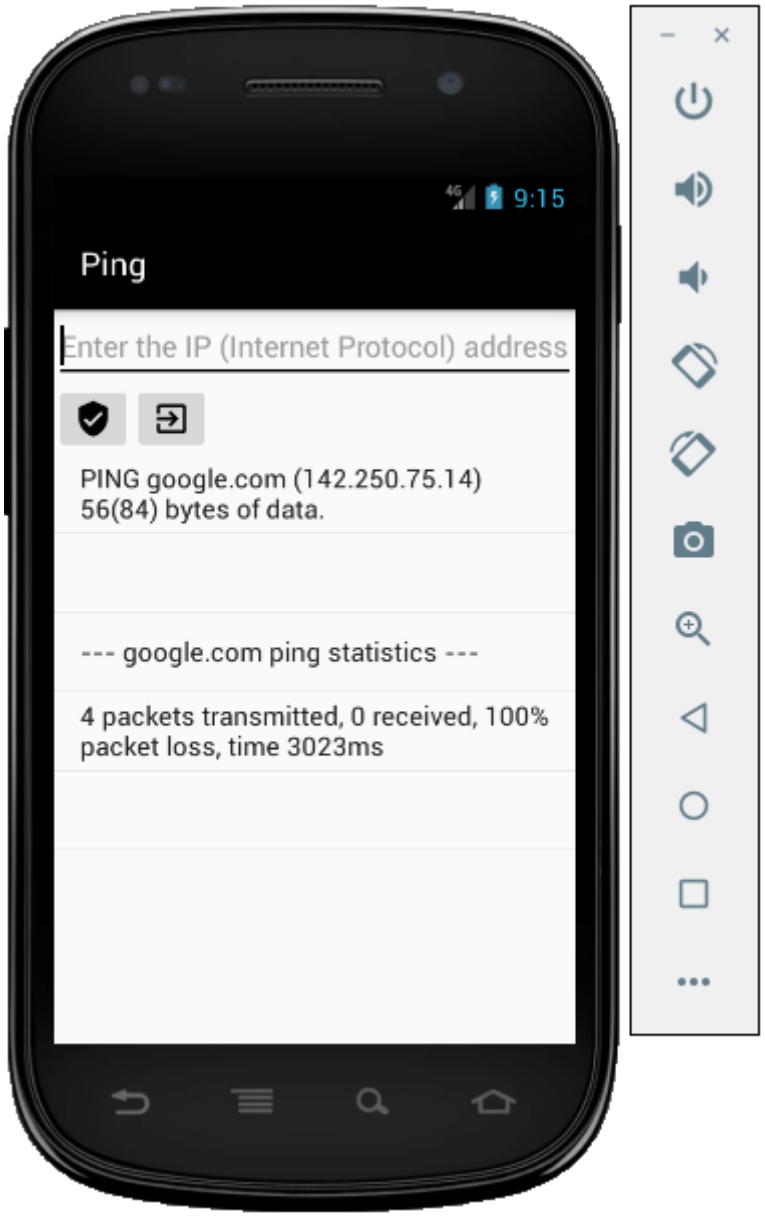
Drawing 1: The beginning of the application's operation [own study]

The user has to enter the website address in the text box. Then he has to click the picture button with the shield icon. After a while, a message about the successful completion of the process appears. It is important to enter the address without https: // or http: // because then

the message is the same, but you cannot see the effects of the program on the screen. This is the first mistake. The second error is that the url is not typed, which has the same effect. The application has no security against these errors and treats them as correct.



Drawing 2: Entering an url address [own study]



Drawing 3: Displaying the results [own study]

What is needed for use?

The application requires installation on the Android operating system with a minimum API level Android 4.0 (IceCreamSandwich), i.e. on all mobile devices with this system.

Algorithm used

Ping - A command used in TCP / IP computer networks (such as the Internet) to diagnose network connections. It allows you to check if there is a connection between the test and test hosts. It allows you to measure the number of lost packets and delays in their transmission, called lags.

Ping uses the ICMP protocol, sends ICMP Echo Request packets, and receives ICMP Echo Reply.

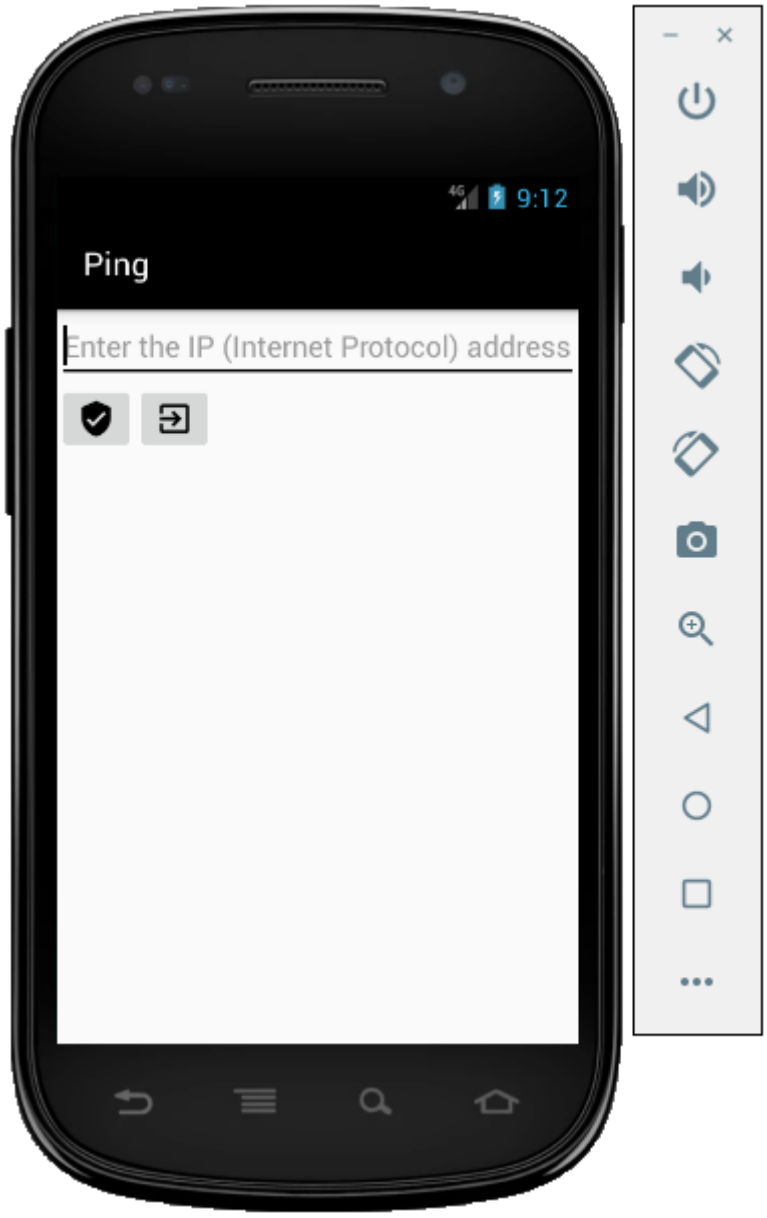
Most publicly available servers support these packages. In this case, the ping application allows you to check the correct operation of the computer network, however, most users of the ping program mistakenly believe that if the remote host does not respond to pings sent, it is only when it is not turned on or there is a problem with the network between the two machines.

It is therefore worth remembering that blocking the sending of ICMP Echo Reply-reply packets (firewalls or filters in routers are used for this purpose) is one of the common methods of protection against an attack from the network. This is because analyzing the ICMP Echo Reply packets not only confirms the existence of a host at a given IP address, but often also allows you to accurately determine the operating system, which makes it easier for a computer to attack.

Also, the answer to the ping does not mean that the service (e.g. a website) is working on this system, or even that the operating system is still working.

Sometimes the ping response (ICMP Echo Reply) is also called pong.[1]

Interface description



Drawing 4: Graphical interface [own study]

The interface has the basic components available in the Android Studio development environment: ListView, ImageButton and EditText.

Source code description

The project was made in the Java programming language, in the Android Studio programming environment. All work was done on the Windows 10 operating system. The application's source code looks like this.

```
package com.example.testping;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.text.Editable;
import android.view.View;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.EditText;
import android.widget.ArrayAdapter;
import android.widget.Toast;
import java.io.InputStreamReader;
import java.io.BufferedReader;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    ImageButton imgbtnip,imgbtne;
    ListView listviewping;
    EditText editip;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        initializecomponents();
    }

    private void initializecomponents(){
        listviewping=findViewById(R.id.listviewping);
        editip=findViewById(R.id.editip);
        imgbtnip=findViewById(R.id.imgbtnip);
        imgbtnip.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                fExcecutarPing();
            }
        });
        imgbtne=findViewById(R.id.imgbtne);
        imgbtne.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
```

```
    private void fExcecutarPing(){
        Editable host=editip.getText();
        List<String>listaResponstaPing=new ArrayList<String>();
        ArrayAdapter<String> adapterLista=new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,
                listaResponstaPing);
        try{
            String cmdPing="ping -c 4 "+host;
            Runtime r=Runtime.getRuntime();
            Process p=r.exec(cmdPing);
            BufferedReader in =new BufferedReader(new
InputStreamReader(p.getInputStream()));
            String inputLinhe;
            while((inputLinhe=in.readLine())!=null){
                listaResponstaPing.add(inputLinhe);
                listviewping.setAdapter(adapterLista);
            }
            Toast.makeText(this,"The command was successfully
completed!",Toast.LENGTH_LONG).show();
        }catch(Exception e){
            Toast.makeText(this,"Error:
"+e.getMessage(),Toast.LENGTH_LONG).show();
        }finally {
            editip.setText("");
        }
    }
}
```

Listing 1: The source code of MainActivity.java file [own study]

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical" >
```

```
            <EditText
                android:id="@+id/editip"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:ems="10"
                android:hint="Enter the IP (Internet Protocol)
address"
                android:inputType="textPersonName" />

            <LinearLayout
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:orientation="horizontal">

                <ImageButton
                    android:id="@+id/imgbtnip"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"

app:srcCompat="@drawable/ic_verified_user_black_24dp" />

                <ImageButton
                    android:id="@+id/imgbtne"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"

app:srcCompat="@drawable/ic_exit_to_app_black_24dp" />

            </LinearLayout>

            <ListView
                android:id="@+id/listviewping"
                android:layout_width="match_parent"
                android:layout_height="600dp" >

            </ListView>
        </LinearLayout>
    </ScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Listing 2: The source code of activity_main.xml file [own study]

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.testping">
```

```
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Listing 3: The source code of AndroidMainfest.xml file [own study]

The code snippet from the AndroidMainfest.xml file below allows you to check the network connection. For this reason, its content has been included in the documentation.

```
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"
/>
<uses-permission android:name="android.permission.INTERNET" />
```

Listing 4: Source code snippet from AndroidMainfest.xml to check network connection [own study]

List of drawings

List of listings

Bibliography

[1] https://pl.wikipedia.org/wiki/Ping