НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП’ЮТЕРНИХ СИСТЕМ

КАФЕДРА «ІНФОРМАЦІЙНИХ СИСТЕМ»

Лабораторна робота №7

з дисципліни «Програмування мобільних пристроїв»

Тема: «Робота з широкомовними повідомленнями та повідомленнями»

Виконано:

Ст. групи АІ-195

Урбанович М. К.

Перевірили:

Годовиченко М. А.

Смик С. Ю.

Одеса 2022

Завдання:

1. Розробте програму, яка показуватиме діалогове вікно з інформаційним повідомленням за наступних подій:

- низький заряд батареї;
- натискання кнопки камери;
- увімкнення та вимкнення режиму "в літаку" (у приймачі дізнайтеся, чи був режим увімкнено або вимкнено і видайте різне повідомлення в діалоговому вікні).

Приймач реєструйте у методах контексту.

2. Модифікуйте програму з першого завдання таким чином, щоб замість діалогових вікон вона видавала повідомлення.

Широкомовні повідомлення та повідомлення.

Широкомовні повідомлення – механізм оповіщення програм про настання певних подій. Програми користувача можуть надсилати та отримувати повідомлення від інших програм, а також можуть отримувати повідомлення від системних програм Android. Також, широкомовні повідомлення можуть призначатися іншим компонентам цієї програми (наприклад, один із способів взаємодії Service і Activity це обмін широкомовними повідомленнями).

Механізм широкомовних повідомлень реалізується та підтримується засобами операційної системи (операційна система виступає у ролі «брокера повідомлень»). Всі повідомлення надсилаються через контекст "в операційну систему", а всі бажаючі отримувати повідомлення повинні бути зареєстровані в операційній системі.

Основні сценарії використання механізму широкомовних повідомлень:

- реакція на системні події та на події пристрою (основний сценарій використання широкомовних повідомлень);
- Обмін повідомленнями між програмами (як правило, обидва додатки повинні бути написані вами, тому що щоб прийняти повідомлення, необхідно точно знати його ідентифікатор);
- Обмін повідомленнями між компонентами однієї програми (обмін повідомленнями між service та activity або між service).

Прикладів використання широкомовних повідомлень багато - ОС розсилає повідомлення при настанні різних системних подій (кількість таких подій дуже велика), програми можуть розсилати повідомлення, якщо відбулася деяка подія, яка може бути цікава іншим програмам (наприклад, поява нових даних, завантаження файлу) і так далі. Робота механізму широкомовних повідомлень заснована на моделі publisher-subscriber («передплатник-видавець»), тому нам необхідно окремо розглянути питання надсилання та прийому широкомовного повідомлення.

Хід роботи

Завдання 1.

Код програми:

dialog/first/MainActivity.kt

```
package com.example.dialog.first

import android.content.Intent
import android.content.IntentFilter
import android.net.ConnectivityManager
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.example.dialog.R

class MainActivity : AppCompatActivity() {

    private val airplaneModeReceiver = Receiver.AirplaneModeReceiver { mes-
sage ->
        val dialog = Dialog.newInstance("AirplaneMode", message)
        dialog.show(supportFragmentManager, "tag")
        }

    private val batteryReceiver = Receiver.BatteryReceiver { message ->
        val dialog = Dialog.newInstance("BatteryLow", message)
        dialog.show(supportFragmentManager, "tag")
        }

    private val cameraReceiver = Receiver.CameraReceiver { message ->
        val dialog = Dialog.newInstance("Camera", message)
        dialog.show(supportFragmentManager, "tag")
        }


    override fun onCreate(savedInstanceState: Bundle?) {

        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    override fun onStart() {
        super.onStart()

        registerReceiver(batteryReceiver, IntentFilter(ConnectivityManager.-
CONNECTIVITY_ACTION)
            .apply {
            addAction(Intent.ACTION_BATTERY_LOW)
        })

        registerReceiver(cameraReceiver, IntentFilter(ConnectivityManager.-
CONNECTIVITY_ACTION)
            .apply {
            addAction(Intent.ACTION_CAMERA_BUTTON)
        })

        registerReceiver(airplaneModeReceiver, IntentFilter(ConnectivityMan-
ager.CONNECTIVITY_ACTION)
            .apply {
            addAction(Intent.ACTION_AIRPLANE_MODE_CHANGED)
        })
    }
```

```
    override fun onStop() {
        super.onStop()
        unregisterReceiver(batteryReceiver)
        unregisterReceiver(cameraReceiver)
        unregisterReceiver(airplaneModeReceiver)
    }
}
```

dialog/first/Dialog.kt

```
package com.example.dialog.first

import android.app.Dialog
import android.os.Bundle
import androidx.fragment.app.DialogFragment

// контейнер диалоговых окон
class Dialog : DialogFragment() {

    private var title: String? = null
    private var message: String? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        arguments?.let {
            title = it.getString(ARG_TITLE)
            message = it.getString(ARG_MESSAGE)
        }
    }

    // создание объекта диалога
    override fun onCreateDialog(savedInstanceState: Bundle?): Dialog {
        return activity.let {
            val builder = it?.let { thisdlg -> androidx.appcompat.app.Alert-
Dialog.Builder(thisdlg) }
            builder
                ?.setTitle(title)
                ?.setMessage(message)
                ?.setPositiveButton("OK") { _, _ -> }
            builder!!.create()
        }
    }

    // структура текста в диалоге
    companion object {
        private const val ARG_TITLE = "title"
        private const val ARG_MESSAGE = "message"

        fun newInstance(title: String, message: String) = Dialog().apply {
            arguments = Bundle().apply {
                putString(ARG_TITLE, title)
                putString(ARG_MESSAGE, message)
            }
        }
    }
}
```

dialog/first/Reciever.kt

```
package com.example.dialog.first

import android.content.BroadcastReceiver
import android.content.Context
```

```
import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.provider.Settings

class Receiver : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
    }

    class AirplaneModeReceiver(val callback: (String) -> Unit) : Broadcas-
tReceiver() {

        override fun onReceive(context: Context?, intent: Intent?) {
            context.let {
                intent.let { actionIntent ->
                    if (actionIntent?.action == In-
tent.ACTION_AIRPLANE_MODE_CHANGED) {
                        if (getAirplaneMode(context)) {
                            callback("Airplane Mode is on")
                        } else {
                            callback("Airplane Mode is off")
                        }
                    }
                }
            }
        }

        private fun getAirplaneMode(context: Context?) : Boolean {
            return Settings.Global.getInt(context
                ?.contentResolver,
                Settings.Global.AIRPLANE_MODE_ON, 0) != 0
        }
    }

    class BatteryReceiver(val callback: (String) -> Unit) : BroadcastRe-
ceiver() {

        override fun onReceive(context: Context?, intent: Intent?) {
            context.let {
                intent.let { actionIntent ->
                    if (actionIntent?.action == Intent.ACTION_BATTERY_LOW) {
                        callback("Low battery percentage")
                    }
                }
            }
        }
    }

    class CameraReceiver(private val callback: (String) -> Unit) : Broadcas-
tReceiver(){

        override fun onReceive(context: Context?, intent: Intent?) {
            context.let {
                intent.let { actionIntent ->
                    if (actionIntent?.action == Intent.ACTION_CAMERA_BUTTON)
{

                        callback("Camera is on")
                    }
                }
            }
        }
    }
}
```

Завдання 2.

Код програми:

dialog/second/MainActivity.kt

```
package com.example.dialog.second

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.example.dialog.R

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

dialog/second/App.kt

```
package com.example.dialog.second

import android.app.Application
import android.app.NotificationChannel
import android.app.NotificationManager
import android.content.Intent
import android.content.IntentFilter
import android.net.ConnectivityManager
import android.os.Build
import com.example.dialog.second.NorifyRecievers.AirplaneModeReceiver
import com.example.dialog.second.NorifyRecievers.BatteryReceiver
import com.example.dialog.second.NorifyRecievers.CameraReceiver


class App : Application() {

    private val batteryReceiver = BatteryReceiver()
    private val cameraReceiver = CameraReceiver()
    private val airplaneModeReceiver = AirplaneModeReceiver()

    override fun onCreate() {
        super.onCreate()

        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val descText = "Important notifications"
            val importance = NotificationManager.IMPORTANCE_HIGH
            val mChannel = NotificationChannel("1", descText, importance)
            mChannel.description = descText

            val notificationManager = getSystemService(NOTIFICATION_SERVICE)
as NotificationManager
            notificationManager.createNotificationChannel(mChannel)
        }

        val batteryFilter =
IntentFilter(ConnectivityManager.CONNECTIVITY_ACTION).apply {
            addAction(Intent.ACTION_BATTERY_LOW)
        }
        registerReceiver(batteryReceiver, batteryFilter)

        val cameraFilter = IntentFilter(ConnectivityManager.CONNECTIVITY_AC-
TION).apply {
            addAction(Intent.ACTION_CAMERA_BUTTON)
```

```
        }
        registerReceiver(cameraReceiver, cameraFilter)

        val flyModeFilter =
IntentFilter(ConnectivityManager.CONNECTIVITY_ACTION).apply {
            addAction(Intent.ACTION_AIRPLANE_MODE_CHANGED)
        }
        registerReceiver(airplaneModeReceiver, flyModeFilter)
    }

    override fun onTerminate() {
        super.onTerminate()

        unregisterReceiver(batteryReceiver)
        unregisterReceiver(cameraReceiver)
        unregisterReceiver(airplaneModeReceiver)
    }
}
```

dialog/second/NorifyRecievers/AirplaneModeReceiver.kt

```
package com.example.dialog.second.NorifyRecievers

import android.content.BroadcastReceiver
import android.content.Context
import android.content.Intent
import android.provider.Settings
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat
import com.example.dialog.R

class AirplaneModeReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context?, intent: Intent?) {
        context?.let {
            intent?.let { actionIntent ->
                if (actionIntent.action == In-
tent.ACTION_AIRPLANE_MODE_CHANGED) {
                    if (getAirplaneMode(context)) {
                        notify(context, "Airplane mode is on")
                    } else {
                        notify(context, "Airplane mode is off")
                    }
                }
            }
        }
    }

    private fun getAirplaneMode(context: Context?) : Boolean {
        return Settings.Global.getInt(context?.contentResolver,
            Settings.Global.AIRPLANE_MODE_ON, 0) != 0;
    }

    private fun notify(context: Context, message: String) {
        val builder = NotificationCompat.Builder(context, "1")
            .setSmallIcon(R.drawable.ic_launcher_foreground)
            .setContentTitle("Incoming broadcast!")
            .setContentText(message)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)

        NotificationManagerCompat.from(context).apply {
            this.notify(0,  builder.build())
        }
```

```
    }
}
```

dialog/second/NorifyRecievers/BatteryReceiver.kt

```
package com.example.dialog.second.NorifyRecievers

import android.content.BroadcastReceiver
import android.content.Context
import android.content.Intent
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat
import com.example.dialog.R

class BatteryReceiver : BroadcastReceiver() {

    override fun onReceive(context: Context?, intent: Intent?) {
        context?.let {
            intent?.let { actionIntent ->
                if (actionIntent.action == Intent.ACTION_BATTERY_LOW) {
                    notify(context, "Low battery percentage")
                }
            }
        }
    }

    private fun notify(context: Context, message: String) {
        val builder = NotificationCompat.Builder(context, "1")
            .setSmallIcon(R.drawable.ic_launcher_foreground)
            .setContentTitle("Incoming broadcast!")
            .setContentText(message)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)

        NotificationManagerCompat.from(context).apply {
            this.notify(0,  builder.build())
        }
    }
}
```

dialog/second/NorifyRecievers/CameraReceiver.kt

```
package com.example.dialog.second.NorifyRecievers

import android.content.BroadcastReceiver
import android.content.Context
import android.content.Intent
import androidx.core.app.NotificationCompat
import androidx.core.app.NotificationManagerCompat
import com.example.dialog.R

class CameraReceiver : BroadcastReceiver(){

    override fun onReceive(context: Context?, intent: Intent?) {
        context?.let {
            intent?.let { actionIntent ->
                if (actionIntent.action == Intent.ACTION_CAMERA_BUTTON) {
                    notify(context, "Camera is on")
                }
            }
        }
    }

    private fun notify(context: Context, message: String) {
        val builder = NotificationCompat.Builder(context, "1")
```

```
            .setSmallIcon(R.drawable.ic_launcher_foreground)
            .setContentTitle("Incoming broadcast!")
            .setContentText(message)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT)

        NotificationManagerCompat.from(context).apply {
            this.notify(0,  builder.build())
        }
    }

}
```

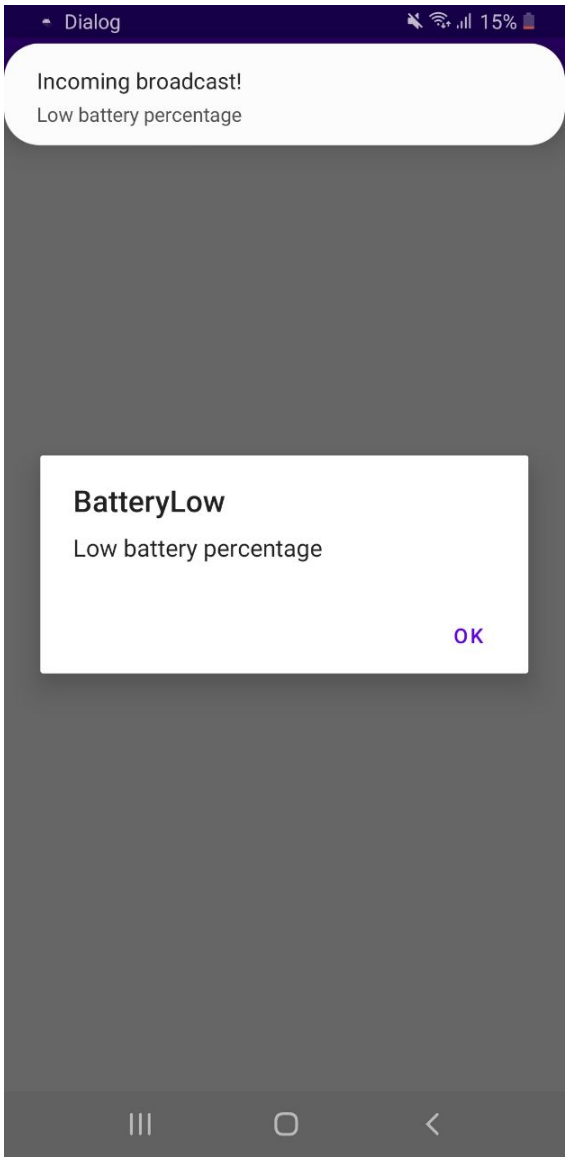
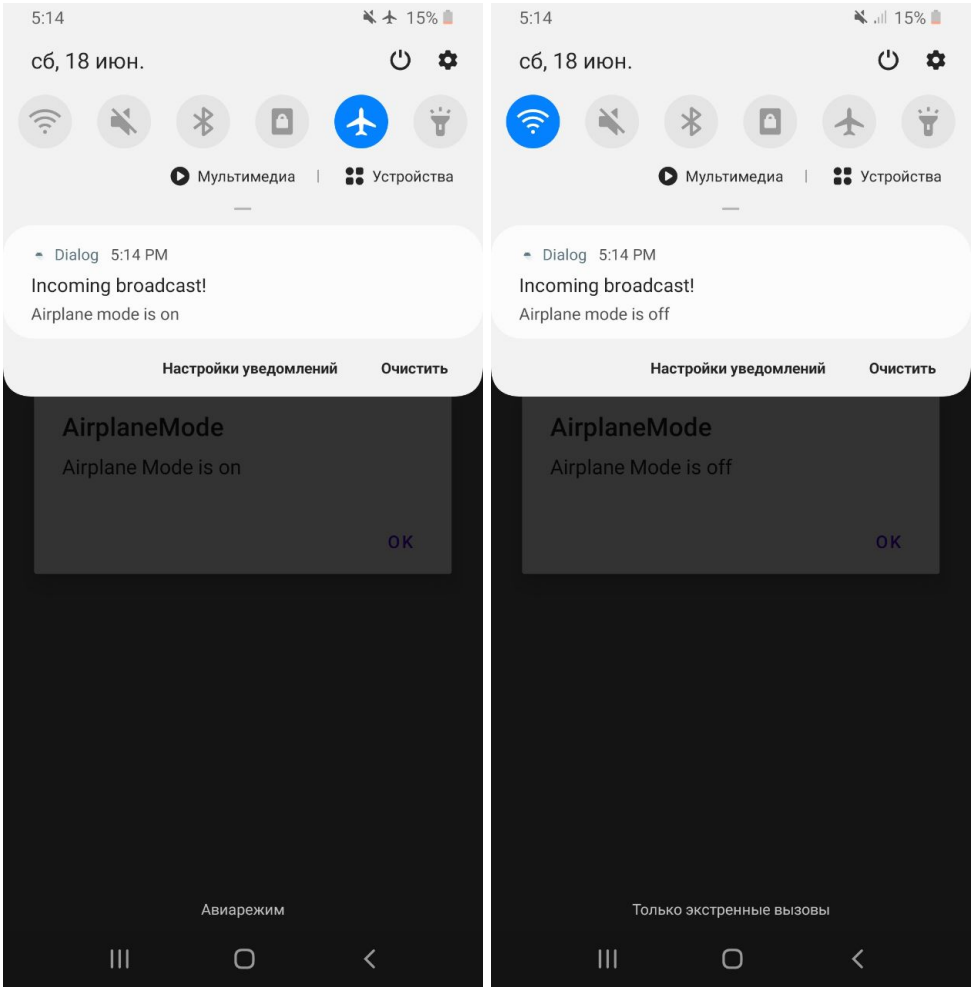


Рисунок 1 – Вікно додатку з діалогом та повідомленням при низькому відсотку заряду батареї



а) б)

Рисунок 2 – – Вікно додатку з діалогом та повідомленням при а) увімкнені режиму польоту; б) вимкнення режиму польоту

Висновки: В ході виконання лабораторної роботи було отримано навички пов’язані з роботою з діалогами та широкомовними повідомленнями.

ДОДАТОК А

Посилання на GitHub з кодом додатку:

Ayashma/Dialog _AI195_Urbanovich (github.com)
https://github.com/Ayashma/Dialog_AI195_Urbanovich