МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Лабораторная работа № 13

По дисциплине: «Разработка программного обеспечения для мобильных платформ»

Тема: «Хранение данных. Настройки и внешние файлы»

Выполнила: Велесевич В. А.

Группа: ПО-10

Проверил: Козинский А. А.

Брест 2025

**Цель:** реализовать приложение работающее с файлами внешнего хранилища.

**Постановка задачи:**

Задание 1. Реализуйте пример подключения к сети.

Задание 2. Реализуйте коды приложений в примерах из источника (запросы, взаимодействие с сервером через сокеты).

Задание 3. Разработайте мобильное приложение согласно заданию 3 источника, позволяющее пользователю асинхронно скачивать файлы журнала Научно-технический вестник (возможно взять другой источник файлов подобной структуры).

Задание 4. Хранение и чтение настроек. При запуске приложения пользователю должно выводиться всплывающее полупрозрачное уведомление (popupWindow), с краткой инструкцией по использованию приложения (можете написать случайный текст), чекбоксом «Больше не показывать» и кнопкой «ОК».

**Графическое изображение интерфейса:**

|  |  |
| --- | --- |
| Рис. 1 - Отображение списка элементов с помощью собственного адаптера | Рис. 2 - Отображение инструкции приложения |

**Код программы:**

**MainActivity**

package com.example.lab4;  
  
import android.content.ContentUris;  
import android.content.ContentValues;  
import android.content.Context;  
import android.content.Intent;  
import android.database.Cursor;  
import android.net.Uri;  
import android.os.Build;  
import android.os.Bundle;  
import android.os.Environment;  
import android.provider.MediaStore;  
import android.util.Log;  
import android.view.KeyEvent;  
import android.view.LayoutInflater;  
import android.view.Menu;  
import android.view.MenuItem;  
import android.view.View;  
import android.view.inputmethod.EditorInfo;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.appcompat.widget.Toolbar;  
import androidx.core.content.FileProvider;  
import androidx.lifecycle.ViewModelProvider;  
import java.io.File;  
  
public class MainActivity extends AppCompatActivity {  
 private Button downloadButton, viewButton, deleteButton, helpButton;  
 private EditText idEdit;  
 private ProgressBar progressBar;  
 private FileViewModel model;  
  
 public boolean fileExists(String fileId) {  
 Uri uri = MediaStore.Files.getContentUri("external");  
 String[] projection = {MediaStore.MediaColumns.\_ID};  
 String selection = MediaStore.MediaColumns.DISPLAY\_NAME + " = ? AND " +  
 MediaStore.MediaColumns.RELATIVE\_PATH + " = ?";  
 String[] selectionArgs = {fileId, "Documents/my\_pdfs/"};  
  
 Log.d("FILEEXISTS", selection);  
 boolean fileExists = false;  
 try (Cursor cursor = getContentResolver().query(uri, projection, selection, selectionArgs, null)) {  
 if (cursor != null && cursor.moveToFirst()) {  
 fileExists = true;  
 }  
 } catch (Exception e) {  
 Log.e("FileCheck", "Error accessing MediaStore", e);  
 } finally {  
 return fileExists;  
 }  
 }  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 progressBar = findViewById(R.id.progressBar);  
 idEdit = findViewById(R.id.idEdit);  
 downloadButton = findViewById(R.id.downloadBtn);  
 viewButton = findViewById(R.id.openBtn);  
 deleteButton = findViewById(R.id.deleteBtn);  
 helpButton = findViewById(R.id.buttonHelp);  
  
 Toolbar toolbar = findViewById(R.id.toolbar);  
 setSupportActionBar(toolbar);  
 getSupportActionBar().setDisplayShowTitleEnabled(false);  
 View customView = getLayoutInflater().inflate(R.layout.custom\_actionbar, null);  
 getSupportActionBar().setCustomView(customView);  
 getSupportActionBar().setDisplayShowCustomEnabled(true);  
 model = new ViewModelProvider(this).get(FileViewModel.class);  
 model.getFileExists().observe(this, isExists -> {  
 viewButton.setEnabled(isExists);  
 deleteButton.setEnabled(isExists);  
 });  
  
 model.getProgress().observe(this, prValue -> {  
 progressBar.setProgress(prValue);  
 });  
 helpButton.setOnClickListener(v -> {  
 Intent intent = new Intent(MainActivity.this, HelpActivity.class);  
 startActivity(intent);  
 });  
 model.getIsDownloading().observe(this, isDnld -> {  
 if (isDnld) {  
 progressBar.setVisibility(View.VISIBLE);  
 } else {  
 progressBar.setVisibility(View.GONE);  
 }  
 });  
  
 idEdit.setOnEditorActionListener(new TextView.OnEditorActionListener() {  
 @Override  
 public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {  
 // Проверка, что была нажата клавиша Enter  
 if (actionId == EditorInfo.IME\_ACTION\_DONE || actionId == EditorInfo.IME\_ACTION\_NEXT ||  
 (event != null && event.getKeyCode() == KeyEvent.KEYCODE\_ENTER)) {  
  
 String fileId = idEdit.getText().toString() + ".pdf";  
 if (!fileId.isEmpty()) {  
  
 Log.d("FILEEXISTS", fileId);  
 boolean ind = fileExists(fileId);  
 Log.d("FILEEXISTS", String.valueOf(ind));  
 // Включаем/выключаем кнопки в зависимости от того, найден ли файл  
 model.setFileExists(ind);  
  
 } else {  
 Toast.makeText(MainActivity.this, "Введите ID файла", Toast.LENGTH\_SHORT).show();  
 }  
 return true; // Возвращаем true, чтобы предотвратить обработку события по умолчанию  
 }  
 return false; // Если не Enter, то возвращаем false, чтобы другие действия могли быть выполнены  
 }  
 });  
  
 if (!NetworkUtils.isNetworkAvailable(this)) {  
 Toast.makeText(this, "Нет подключения", Toast.LENGTH\_LONG).show();  
 }  
 downloadButton.setOnClickListener(v -> {  
 String idFile = idEdit.getText().toString() + ".pdf";  
 progressBar.setVisibility(View.VISIBLE);  
 new FileDownloader(this).execute(idFile);  
 });  
 viewButton.setOnClickListener(v -> {  
 String idFile = idEdit.getText().toString() + ".pdf";  
  
 Uri uri = MediaStore.Files.getContentUri("external");  
  
 String[] projection = {MediaStore.MediaColumns.\_ID};  
 String selection = MediaStore.MediaColumns.DISPLAY\_NAME + " = ?";  
 String[] selectionArgs = {idFile};  
  
 try (Cursor cursor = getContentResolver().query(uri, projection, selection, selectionArgs, null)) {  
 if (cursor != null && cursor.moveToFirst()) {  
 long fileId = cursor.getLong(cursor.getColumnIndexOrThrow(MediaStore.MediaColumns.\_ID));  
 uri = ContentUris.withAppendedId(uri, fileId);  
 }  
 } catch (Exception e) {  
 Log.e("FilePicker", "Error accessing MediaStore", e); }  
 if (uri != null) {  
 Intent intent = new Intent(Intent.ACTION\_VIEW);  
 intent.setDataAndType(uri, "application/pdf");  
 intent.setFlags(Intent.FLAG\_GRANT\_READ\_URI\_PERMISSION);  
  
 startActivity(Intent.createChooser(intent, "Open PDF with"));  
 } else {  
 Log.e("FilePicker", "File not found"); } });  
 deleteButton.setOnClickListener(v -> {  
 String idFile = idEdit.getText().toString() + ".pdf";  
 Uri uri = MediaStore.Files.getContentUri("external");  
 String[] projection = {MediaStore.MediaColumns.\_ID};  
 String selection = MediaStore.MediaColumns.DISPLAY\_NAME + " = ?";  
 String[] selectionArgs = {idFile};  
 try (Cursor cursor = getContentResolver().query(uri, projection, selection, selectionArgs, null)) {  
 if (cursor != null && cursor.moveToFirst()) {  
 long fileId = cursor.getLong(cursor.getColumnIndexOrThrow(MediaStore.MediaColumns.\_ID));  
 Uri fileUri = ContentUris.withAppendedId(uri, fileId);  
 int rowsDeleted = getContentResolver().delete(fileUri, null, null);  
 if (rowsDeleted > 0) {  
 viewButton.setEnabled(false);  
 deleteButton.setEnabled(false);  
 idEdit.setText(null);  
 Toast.makeText(this, "Файл удален", Toast.LENGTH\_SHORT).show();  
 } else {  
 Toast.makeText(this, "Не удалось удалить файл", Toast.LENGTH\_SHORT).show();  
 }  
 } else {  
 Toast.makeText(this, "Файл не найден", Toast.LENGTH\_SHORT).show();  
 }  
 } catch (Exception e) {  
 Log.e("FileDelete", "Error accessing MediaStore", e);  
 }  
 });  
 PopupHelper.showPopup(this); }}

**PopupHelper**

package com.example.lab4;  
import android.app.AlertDialog;  
import android.content.Context;  
import android.graphics.Color;  
import android.graphics.drawable.ColorDrawable;  
import android.view.ViewGroup;  
import android.widget.CheckBox;  
import android.widget.LinearLayout;  
public class PopupHelper {  
 public static void showPopup(Context context) {  
 LinearLayout layout = new LinearLayout(context);  
 layout.setOrientation(LinearLayout.VERTICAL);  
 layout.setPadding(50, 50, 50, 50); // (left, top, right, bottom)  
 LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(  
 ViewGroup.LayoutParams.MATCH\_PARENT, // Ширина  
 ViewGroup.LayoutParams.WRAP\_CONTENT // Высота );  
 params.setMargins(50, 50, 50, 50); // (left, top, right, bottom)  
 layout.setLayoutParams(params);  
 CheckBox checkBox = new CheckBox(context);  
 checkBox.setText("Больше не показывать");  
 layout.addView(checkBox);  
 if (SettingsManager.shouldShowPopup(context)) {  
 AlertDialog dialog = new AlertDialog.Builder(context)  
 .setTitle("Инструкция")  
 .setMessage("Загрузка статей из журнала Научно-технический вестник.")  
 .setView(layout)  
 .setPositiveButton("ОК", (dlg, which) -> {  
 if (checkBox.isChecked()) {  
 SettingsManager.setShowPopup(context, false); } })  
 .create();  
 dialog.show(); } } }  
  
 **Вывод:** реализовала интерфейс приложения для загрузки файлов