Департамент образования Воронежской области

Государственное бюджетное профессиональное образовательное учреждение

Воронежской области «Борисоглебский техникум промышленных

и информационных технологий»

**ОТЧЕТ**

**по учебной практике**

**УП.05.01 Проектирование и разработка информационных систем**

|  |  |
| --- | --- |
| Выполнил студент \_\_3\_\_ курса  спец. 09.02.07 «Информационные системы и программирование»  Группа 3.2ИСИП\_\_  Дейнекин Р.К.  (ФИО)  Дата сдачи \_\_31.05.2023 г.\_\_\_\_ | Проверил  преподаватель  Пеньков А.С.  (ФИО)  Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Борисоглебск

**2022 –2023 учебный год**

Тема 1.

Выбор средств разработки проекта

Отвечаем на вопросы

1. Определение ОС Android

Android-это мобильная операционная система, основанная на модифицированной версии ядра Linux и другого программного обеспечения с открытым исходным кодом, предназначенная в первую очередь для сенсорных мобильных устройств, таких как смартфоны и планшеты.

1. Что входит в инструментарий Android разработчика

Java Development Kit (JDK)

Android Software Development Kit (SDK)

Приложения

Библиотеки

Платформа приложений

Выполняемая среда Android

Ядро Linux

1. Перечислите обязательные элементы в структуре приложения

Gen – Файлы сгенерированные самой java

AndroidManifest.xml – файл манифеста предоставляет системе основную информацию о программе.

Src – Каталог в котором содержится исходный код приложения

Res – каталог содержащий ресурсы приложения

1. Перечислите опциальные элементы в структуре приложения

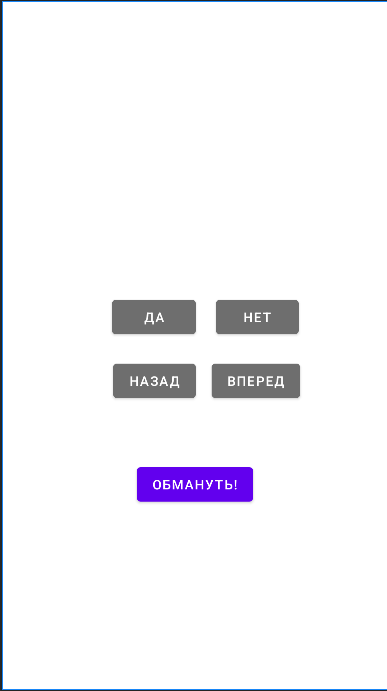
Assets Произвольное собрание каталогов и файлов

1. Изобразите жизненный цикл android приложения



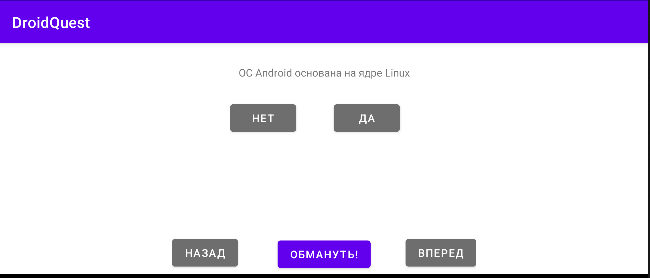
Создаю проект с название QuestActivty

Добавляем разметку в activity\_quest.xml



<?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"  
 android:backgroundTint="#202020"  
 tools:context=".MainActivity">  
  
 <Button  
 android:id="@+id/true\_button"  
 android:layout\_width="89dp"  
 android:layout\_height="48dp"  
 android:layout\_marginTop="312dp"  
 android:backgroundTint="#6E6E6E"  
 android:text="Да"  
 app:layout\_constraintEnd\_toStartOf="@+id/false\_button"  
 app:layout\_constraintHorizontal\_bias="0.847"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <Button  
 android:id="@+id/false\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="312dp"  
 android:layout\_marginEnd="96dp"  
 android:backgroundTint="#6E6E6E"  
 android:text="Нет"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <Button  
 android:id="@+id/deceit\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/deceit\_button"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.498"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/previous\_button"  
 app:layout\_constraintVertical\_bias="0.243" />  
  
 <Button  
 android:id="@+id/next\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:backgroundTint="#6E6E6E"  
 android:text="вперед"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.702"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/true\_button"  
 app:layout\_constraintVertical\_bias="0.061" />  
  
 <Button  
 android:id="@+id/previous\_button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:backgroundTint="#6E6E6E"  
 android:text="назад"  
 android:visibility="visible"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.365"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/true\_button"  
 app:layout\_constraintVertical\_bias="0.061" />  
  
 <TextView  
 android:id="@+id/question\_text\_view"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAlignment="center"  
 app:layout\_constraintBottom\_toTopOf="@+id/next\_button"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <TextView  
 android:id="@+id/question\_index"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintBottom\_toTopOf="@+id/true\_button"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/question\_text\_view" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

И создаем альбомный макет



Тема 2.

Разработка технического задания.

Добавляем код в класс QuestActivity.java

package com.example.questactivity;

import androidx.appcompat.app.AppCompatActivity;  
  
import android.app.Activity;  
import android.content.Intent;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
import android.widget.Toast;  
  
public class MainActivity extends AppCompatActivity {  
 private static final String *TAG* = "MainActivity";  
 private static final String *KEY\_INDEX* = "index";  
 private static final int *REQUEST\_CODE\_DECEIT* = 0;  
 private Button mTrueButton;  
 private Button mFalseButton;  
 private Button mNextButton;  
 private Button mPreviousButton;  
 private Button mDeceitButton;  
 private TextView mQuestionTextView;  
 private boolean mIsDeceiter;  
  
 @Override  
 public void onStart() {  
 super.onStart();  
 Log.*d*(*TAG*, "onStart() вызван");  
 }  
  
 @Override  
 public void onPause() {  
 super.onPause();  
 Log.*d*(*TAG*, "onPause() вызван");  
 }  
  
 @Override  
 public void onStop() {  
 super.onStop();  
 Log.*d*(*TAG*, "onStop() вызван");  
 }  
  
 @Override  
 public void onDestroy() {  
 super.onDestroy();  
 Log.*d*(*TAG*, "onDestroy() вызван");  
 }  
  
 private void checkAnswer(boolean userPressedTrue) {  
 boolean answerIsTrue = mQuestionBank[mCurrentIndex].ismAnswerTrue();  
 int messageResId = 0;  
 if (mIsDeceiter) {  
 messageResId = R.string.*judgement\_toast*;  
 } else {  
 messageResId = R.string.*incorrect*;  
 }  
 Toast.*makeText*(this, messageResId, Toast.*LENGTH\_SHORT*).show();  
 }  
  
 private void updateQuestion() {  
 int question = mQuestionBank[mCurrentIndex].getmTextResId();  
 mQuestionTextView.setText(question);  
 }  
  
 private int mCurrentIndex = 0;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 Log.*d*(*TAG*, "onCreate(Bundle) вызван");  
 setContentView(R.layout.*activity\_main*);  
  
 mQuestionTextView = (TextView) findViewById(R.id.*question\_text\_view*);  
 mTrueButton = (Button) findViewById(R.id.*true\_button*);  
 mTrueButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 checkAnswer(true);  
 }  
 });  
  
 mFalseButton = (Button) findViewById(R.id.*false\_button*);  
 mFalseButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 checkAnswer(false);  
 }  
 });  
  
 mNextButton = (Button) findViewById(R.id.*next\_button*);  
 mNextButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;  
 mIsDeceiter = false;  
 updateQuestion();  
 }  
 });  
  
 mPreviousButton = (Button) findViewById(R.id.*previous\_button*);  
 if (mCurrentIndex == 0) {  
 mPreviousButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 mCurrentIndex = (mCurrentIndex - 1) % mQuestionBank.length;  
 updateQuestion();  
 }  
 });  
 }  
 mDeceitButton = (Button) findViewById(R.id.*deceit\_button*);  
 mDeceitButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
// Intent i = new Intent(MainActivity.this, DeceitActivity.class);  
// startActivity(i);  
 boolean answerIsTrue = mQuestionBank[mCurrentIndex].ismAnswerTrue();  
 Intent i = DeceitActivity.*newIntent*(MainActivity.this, answerIsTrue);  
 startActivityForResult(i, *REQUEST\_CODE\_DECEIT*);  
 }  
 });  
 if(savedInstanceState != null){  
 mCurrentIndex = savedInstanceState.getInt(*KEY\_INDEX*, 0);  
 }  
  
 updateQuestion();  
  
 }  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
 if (resultCode != Activity.*RESULT\_OK*) {  
 return;  
 }  
 if (requestCode == *REQUEST\_CODE\_DECEIT*) {  
 if (data == null) {  
 return;  
 }  
 mIsDeceiter = DeceitActivity.*wasAnswerShown*(data);  
 }  
 }  
 @Override  
 public void onSaveInstanceState(Bundle savedInstanceState){  
 super.onSaveInstanceState(savedInstanceState);  
 Log.*i*(*TAG*, "onSaveInstanceState");  
 savedInstanceState.putInt(*KEY\_INDEX*,mCurrentIndex);  
 }  
  
 private Question[] mQuestionBank = new Question[]{  
 new Question(R.string.*question\_android*, true),  
 new Question(R.string.*question\_linear*, false),  
 new Question(R.string.*question\_service*, false),  
 new Question(R.string.*question\_res*, true),  
 new Question(R.string.*question\_manifest*, true),  
 new Question(R.string.*question\_multitask*, true),  
 new Question(R.string.*question\_first\_version*, false),  
 new Question(R.string.*question\_security*, false),  
 new Question(R.string.*quest\_php*, false),  
 new Question(R.string.*question\_last\_version*, true),  
 };  
}

Создаем класс Question.java

package com.example.questactivity;  
  
public class Question {  
 private int mTextResId;  
 private boolean mAnswerTrue;  
 public Question(int textResId, boolean answerTrue){  
 mTextResId = textResId;  
 mAnswerTrue = answerTrue;  
 }  
  
 public int getmTextResId() {  
 return mTextResId;  
 }  
  
 public void setmTextResId(int mTextResId) {  
 this.mTextResId = mTextResId;  
 }  
  
 public boolean ismAnswerTrue() {  
 return mAnswerTrue;  
 }  
  
 public void setmAnswerTrue(boolean mAnswerTrue) {  
 this.mAnswerTrue = mAnswerTrue;  
 }

Создаем DeceitActivity.java

package com.example.questactivity;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.app.Activity;  
import android.content.Context;  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
  
public class DeceitActivity extends AppCompatActivity {  
 private boolean mAnswerIsTrue;  
 private TextView mAnswerTextView;  
 private Button mShowAnswer;  
 public static final String *EXTRA\_ANSWER\_IS\_TRUE* = "com.example.questactivity.answer\_is\_true";  
 public static final String *EXTRA\_ANSWER\_SHOWN* = "com.example.questactivity.answer\_shown";  
  
 public static Intent newIntent(Context packageContext, boolean answerIsTrue){  
 Intent i = new Intent(packageContext, DeceitActivity.class);  
 i.putExtra(*EXTRA\_ANSWER\_IS\_TRUE*, answerIsTrue);  
 return i;  
 }  
 public static boolean wasAnswerShown(Intent result){  
 return result.getBooleanExtra(*EXTRA\_ANSWER\_SHOWN*, false);  
 }  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_deceit*);  
 mAnswerIsTrue = getIntent().getBooleanExtra(*EXTRA\_ANSWER\_IS\_TRUE*,false);  
 mAnswerTextView = (TextView)findViewById(R.id.*answer\_text\_view*);  
 mShowAnswer = (Button) findViewById(R.id.*show\_answer\_button*);  
 mShowAnswer.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 if(mAnswerIsTrue){  
 mAnswerTextView.setText(R.string.*true\_button*);  
 }else{  
 mAnswerTextView.setText(R.string.*false\_button*);  
 }  
 setAnswerShownResult(true);  
 }  
 });  
 }  
  
 private void setAnswerShownResult(boolean isAnswerShown){  
 Intent data = new Intent();  
 data.putExtra(*EXTRA\_ANSWER\_SHOWN*,isAnswerShown);  
 setResult(*RESULT\_OK*, data);  
 }  
  
}

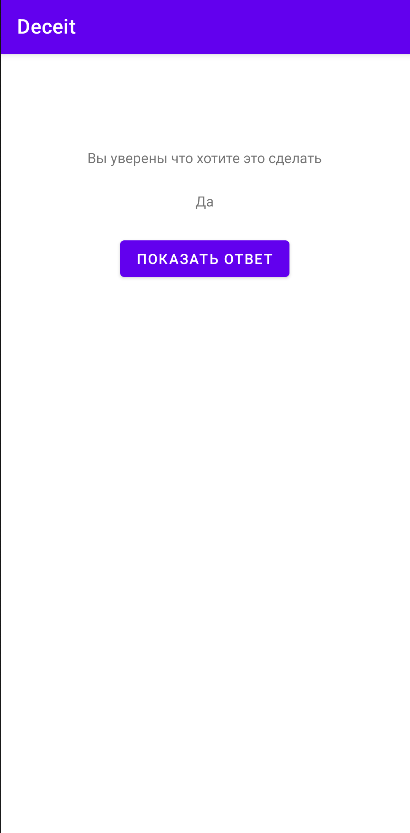
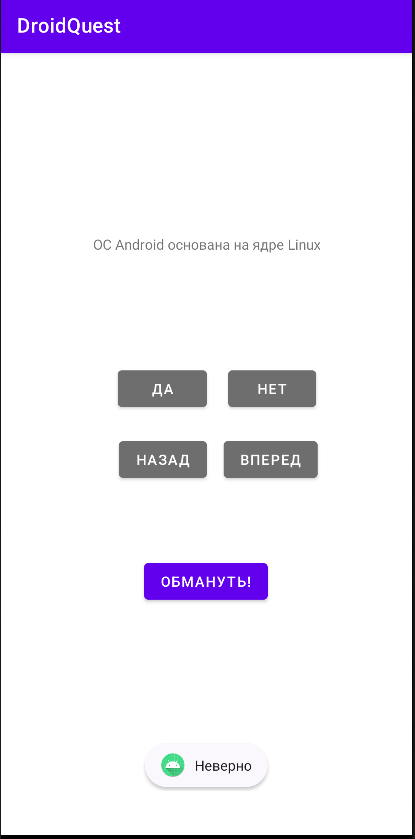
Также дополняем строковые ресурсы

<resources>  
 <string name="app\_name">DroidQuest</string>  
 <string name="question\_linear">Разметка LinearLayout позиционирует свои дочерние элементы в строки и столбцы</string>  
 <string name="question\_res">Ресурсы приложения содержатся в отдельном файле XML</string>  
 <string name="question\_manifest">AndroidManifest.xml предоставляет основную информацию о программе</string>  
 <string name="question\_service">Компонент Android-приложения Service предоставляют доступ к данным</string>  
 <string name="action\_settings">Settings</string>  
 <string name="question\_android">ОС Android основана на ядре Linux</string>  
   
 <string name="question\_last\_version">Последняя версия Android на 2023 год 13?</string>  
 <string name="question\_first\_version">Первая версия Android вышла в 2003 году</string>  
 <string name="question\_security">Android безопаснее IOS?</string>  
 <string name="question\_multitask">Android поддерживает многозадачность?</string>  
 <string name="quest\_php">Android приложения можно написать на php?</string>  
  
 <string name="warning\_text">Вы уверены что хотите это сделать</string>  
 <string name="show\_answer\_button">Показать ответ</string>  
 <string name="deceit\_button">Обмануть!</string>  
 <string name="judgement\_toast">Обманывать нельзя!</string>  
  
 <string name="Deceit">Deceit</string>  
   
 <string name="correct">Верно</string>  
 <string name="incorrect">Неверно</string>  
  
 <string name="true\_button">Да</string>  
 <string name="false\_button">Нет</string>  
</resources>

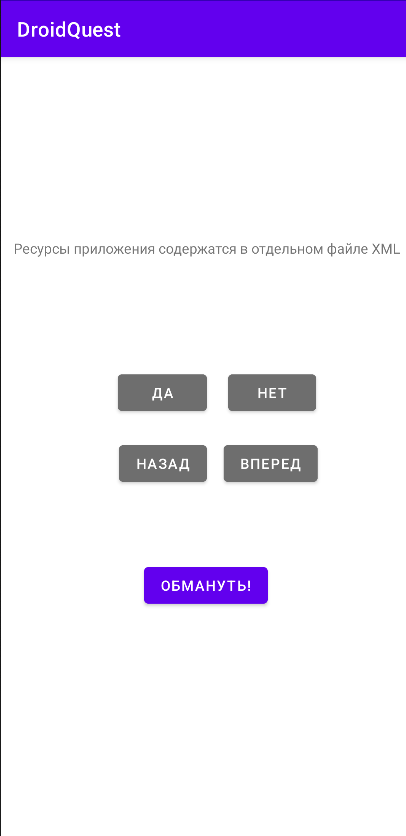
Тема 3.

Система контроля версий.

Тестируем в эмуляторе



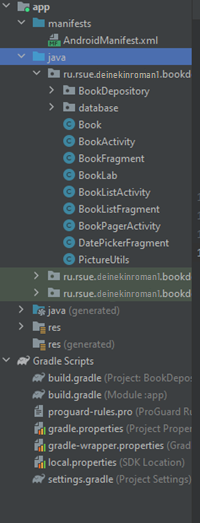
Кнопки вперед и назад функционируют



Тема 4.

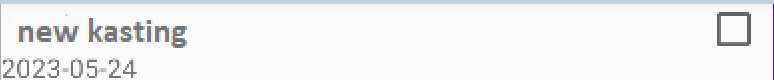
Сбор данных для создания информационной системы

Для начала создали проект под название BookDeopsitory и класс Book для создания экземпляра класса кастингов.



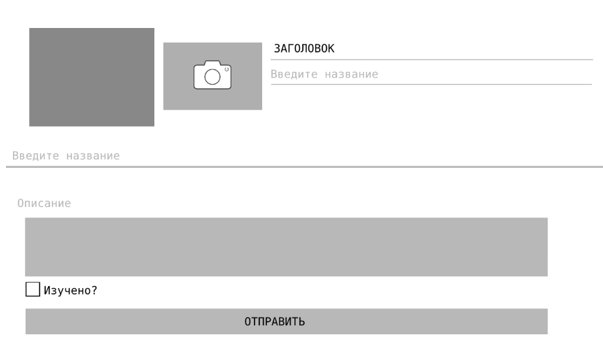
Book.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import java.util.Date;  
import java.util.UUID;  
  
public class Book {  
 private Date mDate;  
 private boolean mReaded;  
 private UUID mId;  
 private String mTitle;  
  
 public Book(UUID id) {  
 mId = id;  
 mDate = new Date();  
 }  
  
 public Date getDate() {  
 return mDate;  
 }  
  
 public void setDate(Date date) {  
 mDate = date;  
 }  
  
 public boolean isReaded() {  
 return mReaded;  
 }  
  
 public void setReaded(boolean readed) {  
 mReaded = readed;  
 }  
  
 public UUID getId() {  
 return mId;  
 }  
  
 public String getTitle() {  
 return mTitle;  
 }  
  
 public void setTitle(String title) {  
 mTitle = title;  
 }  
 public String getPhotoFilename() {  
 return "IMG\_" + getId().toString() + ".jpg";  
 }  
}



У экземпляра книги есть поля название, прочтена ли она, id, и дата запланированного изучения.

Также создали интерфейс book\_fragment.xml



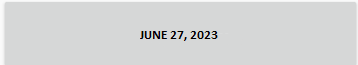
Тема 5.

Создание макета информационной системы.

На данном этапе создаем обработчик для сохранения даты прочтения

Добавили новые виджеты

Кнопка для установки даты прочтения



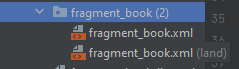
<Button  
 android:id="@+id/book\_date"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginRight="16dp"  
 android:padding="30dp" />

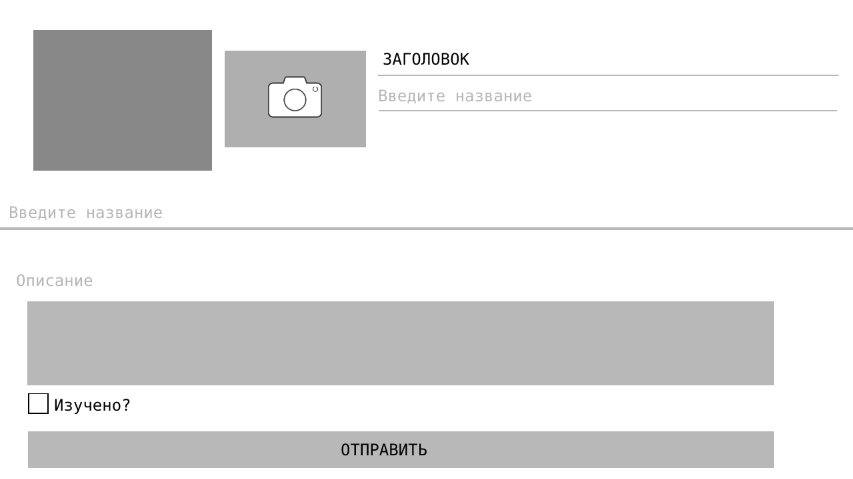
Поля ввода названия и заголовка



<TextView  
 style="?android:listSeparatorTextViewStyle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/book\_title\_lable" />  
  
<EditText  
 android:id="@+id/book\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginRight="16dp"  
 android:hint="@string/book\_title\_hint"  
 android:minHeight="48dp" />

Создали элементы макета в альбомной ориентации





Обработчики чекбокса, кнопки установки даты и заголовка

mTitleField = (EditText) v.findViewById(R.id.*book\_title*);  
mTitleField.setText(mBook.getTitle());  
mTitleField.addTextChangedListener(new TextWatcher() {  
 @Override  
 public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {  
  
 }  
  
 @Override  
 public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {  
 mBook.setTitle(charSequence.toString());  
 }  
  
 @Override  
 public void afterTextChanged(Editable editable) {  
  
 }  
});

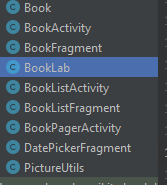
mDateButton = (Button) v.findViewById(R.id.*book\_date*);  
mDateButton.setText(DateFormat.*getDateInstance*().format(mBook.getDate()));  
  
mDateButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 FragmentManager manager = getFragmentManager();  
 DatePickerFragment dialog = DatePickerFragment  
 .*newInstance*(mBook.getDate());  
 dialog.setTargetFragment(BookFragment.this,  
 *REQUEST\_DATE*);  
 dialog.show(manager, *DIALOG\_DATE*);  
 }  
});  
  
mReadedCheckBox = (CheckBox) v.findViewById(R.id.*book\_readed*);  
mReadedCheckBox.setChecked(mBook.isReaded());  
mReadedCheckBox.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
 @Override  
 public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
 mBook.setReaded(isChecked);  
 }  
});

Тема 6.

Создание элементов пользовательского интерфейса

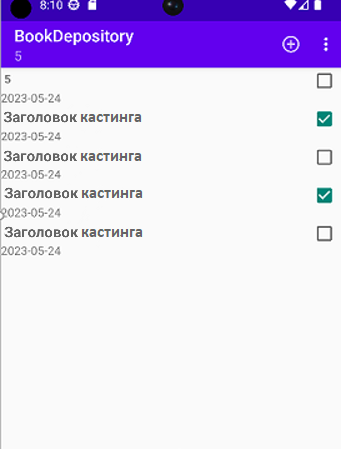
На данном этапе реализовали вывод списка кастингов

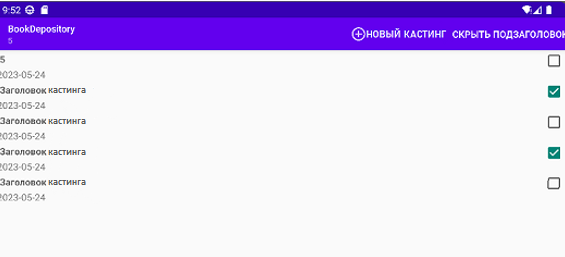
Создали класс BookLab.java



package ru.rsue.deinekinroman1.bookdepository;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.os.Environment;  
import android.view.View;  
import android.widget.Button;  
  
import java.io.File;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.UUID;  
  
import ru.rsue. deinekinroman1.bookdepository.database.BookBaseHelper;  
import ru.rsue. deinekinroman1.bookdepository.database.BookCursorWrapper;  
import ru.rsue. deinekinroman1.bookdepository.database.BookDbSchema;  
  
  
public class BookLab {  
 private static BookLab *sBookLab*;  
 private List<Book> mBooks;  
 private Context mContext;  
 private SQLiteDatabase mDatabase;  
 private Button mDeleteButton;  
  
 public static BookLab get(Context context) {  
 if (*sBookLab* == null) {  
 *sBookLab* = new BookLab(context);  
 }  
 return *sBookLab*;  
 }  
 private BookLab(Context context) {  
 mContext = context.getApplicationContext();  
 mDatabase = new BookBaseHelper(mContext).getWritableDatabase();  
 mBooks = new ArrayList<>();  
 }  
  
 public void addBook(Book b) {  
  
 ContentValues values = *getContentValues*(b);  
 mDatabase.insert(BookDbSchema.BookTable.*NAME*, null, values);  
 }  
  
 public void deleteBook(Book b){  
 mDeleteButton.findViewById(R.id.*delete\_button*);  
 mDeleteButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 mDatabase.delete(BookDbSchema.BookTable.*NAME*,b.getTitle(),null);  
 }  
 });  
  
 }  
 public List<Book> getBooks() {  
 List<Book> books = new ArrayList<>();  
 BookCursorWrapper cursor = queryBooks(null, null);  
 try {  
 cursor.moveToFirst();  
 while (!cursor.isAfterLast()) {  
 books.add(cursor.getBook());  
 cursor.moveToNext();  
 }  
 } finally {  
 cursor.close();  
 }  
 return books;  
 }  
 private static ContentValues getContentValues(Book book) {  
 ContentValues values = new ContentValues();  
 values.put(BookDbSchema.BookTable.Cols.*UUID*, book.getId().toString());  
 values.put(BookDbSchema.BookTable.Cols.*TITLE*, book.getTitle());  
 values.put(BookDbSchema.BookTable.Cols.*DATE*, book.getDate().getTime());  
 values.put(BookDbSchema.BookTable.Cols.*READED*, book.isReaded() ? 1 : 0);  
 return values;  
 }  
 private BookCursorWrapper queryBooks(String whereClause, String[] whereArgs) {  
 Cursor cursor = mDatabase.query(  
 BookDbSchema.BookTable.*NAME*,  
 null, // Columns - null выбирает все столбцы  
 whereClause,  
 whereArgs,  
 null, // groupBy  
 null, // having  
 null // orderBy  
 );  
 return new BookCursorWrapper(cursor);  
 }  
 public Book getBook(UUID id) {  
 for (Book book : mBooks) {  
 if (book.getId().equals(id)) {  
 return book;  
 }  
 }  
 BookCursorWrapper cursor = queryBooks(  
 BookDbSchema.BookTable.Cols.*UUID* + " = ?",  
 new String[] { id.toString() }  
 );  
 try {  
 if (cursor.getCount() == 0) {  
 return null;  
 }  
 cursor.moveToFirst();  
 return cursor.getBook();  
 } finally {  
 cursor.close();  
 }  
 }  
 public File getPhotoFile(Book book) {  
 File externalFilesDir = mContext  
 .getExternalFilesDir(Environment.*DIRECTORY\_PICTURES*);  
 if (externalFilesDir == null) {  
 return null;  
 }  
 return new File(externalFilesDir, book.getPhotoFilename());  
 }  
 public void updateBook(Book book) {  
 String uuidString = book.getId().toString();  
 ContentValues values = *getContentValues*(book);  
 mDatabase.update(BookDbSchema.BookTable.*NAME*, values,  
 BookDbSchema.BookTable.Cols.*UUID* + " = ?",  
 new String[] {  
 uuidString  
 });  
 }

Заполнил интерфейс кастингами.

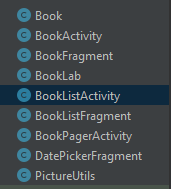




Тема 7.

Material Design

Создали класс BookListActivity.java



BooListActivity.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import androidx.fragment.app.Fragment;  
  
import ru.rsue deinekinroman1.bookdepository.BookDepository.SingleFragmentActivity;  
  
  
public class BookListActivity extends SingleFragmentActivity {  
 @Override  
 protected Fragment createFragment() {  
 return new BookListFragment();  
 }  
}

Тема 8.

Работа со списочными элементами интерфейса

Добавили изменения в манифест приложения для запуска BookListActivty

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools">  
  
 <uses-feature android:name="android.hardware.camera"  
 android:required="false"  
 />  
  
 <application  
 android:allowBackup="true"  
 android:dataExtractionRules="@xml/data\_extraction\_rules"  
 android:fullBackupContent="@xml/backup\_rules"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.BookDepository"  
 tools:targetApi="31">  
 <activity android:name=".BookListActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 <activity  
 android:name=".BookPagerActivity"  
 android:exported="true"  
 android:parentActivityName=".BookListActivity">  
 </activity>  
  
 </application>  
  
</manifest>

Тема 9.

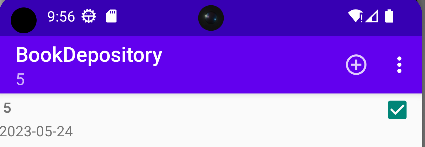
Объектно-ориентированный подход в разработке кода информационной сиcтемы

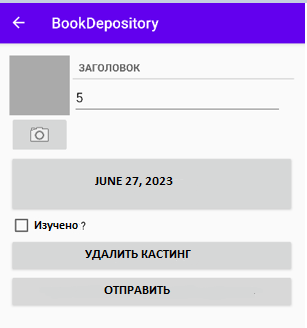
Приложение разрабатывается на языке java который поддерживает только объектно-ориентированное программирование.

Создали класс BookListFragment.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import android.content.Intent;  
import android.os.Bundle;  
  
  
import android.view.LayoutInflater;  
import android.view.Menu;  
import android.view.MenuInflater;  
import android.view.MenuItem;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.CheckBox;  
import android.widget.TextView;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.fragment.app.Fragment;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.List;  
import java.util.UUID;  
  
public class BookListFragment extends Fragment {  
  
 private RecyclerView mBookRecyclerView;  
 private BookAdapter mAdapter;  
 private TextView mTitleTextView;  
 private TextView mDateTextView;  
 private CheckBox mReadedCheckBox;  
 private Book mBook;  
 private boolean mSubtitleVisible;  
 private static final String *SAVED\_SUBTITLE\_VISIBLE* = "subtitle";  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setHasOptionsMenu(true);  
 }  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 View view = inflater.inflate(R.layout.*fragment\_book\_list*, container, false);  
 mBookRecyclerView = (RecyclerView) view.findViewById(R.id.*book\_recycler\_view*);  
 mBookRecyclerView.setLayoutManager(new LinearLayoutManager(getActivity()));  
 if (savedInstanceState != null) {  
 mSubtitleVisible =  
 savedInstanceState.getBoolean(*SAVED\_SUBTITLE\_VISIBLE*);  
 }  
 updateUI();  
 return view;  
 }  
  
  
  
 @Override  
 public void onResume() {  
 super.onResume();  
 updateUI();  
 }  
  
 @Override  
 public void onSaveInstanceState(Bundle outState) {  
 super.onSaveInstanceState(outState);  
 outState.putBoolean(*SAVED\_SUBTITLE\_VISIBLE*, mSubtitleVisible);  
 }  
  
 @Override  
 public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {  
 super.onCreateOptionsMenu(menu, inflater);  
 inflater.inflate(R.menu.*fragment\_book\_list*, menu);  
  
 MenuItem subtitleItem = menu.findItem(R.id.*menu\_item\_show\_subtitle*);  
 if (mSubtitleVisible) {  
 subtitleItem.setTitle(R.string.*hide\_subtitle*);  
 } else {  
 subtitleItem.setTitle(R.string.*show\_subtitle*);  
 }  
 }  
  
 @Override  
 public boolean onOptionsItemSelected(MenuItem item) {  
 switch (item.getItemId()) {  
 case R.id.*menu\_item\_new\_book*:  
 Book book = new Book(UUID.*randomUUID*());  
 BookLab.*get*(getActivity()).addBook(book);  
 Intent intent = BookPagerActivity  
 .*newIntent*(getActivity(), book.getId());  
 startActivity(intent);  
 case R.id.*menu\_item\_show\_subtitle*:  
 mSubtitleVisible = !mSubtitleVisible;  
 getActivity().invalidateOptionsMenu();  
 updateSubtitle();  
 return true;  
 default:  
 return super.onOptionsItemSelected(item);  
 }  
 }  
  
 private void updateSubtitle() {  
 BookLab bookLab = BookLab.*get*(getActivity());  
 int bookCount = bookLab.getBooks().size();  
 String subtitle = getString(R.string.*subtitle\_format*, bookCount);  
 if (!mSubtitleVisible) {  
 subtitle = null;  
 }  
 AppCompatActivity activity = (AppCompatActivity) getActivity();  
 activity.getSupportActionBar().setSubtitle(subtitle);  
 }  
  
 private void updateUI() {  
 BookLab bookLab = BookLab.*get*(getActivity());  
 List<Book> books = bookLab.getBooks();  
 if (mAdapter == null) {  
 mAdapter = new BookAdapter(books);  
 mBookRecyclerView.setAdapter(mAdapter);  
 } else {  
 mAdapter.setBooks(books);  
 mAdapter.notifyDataSetChanged();  
 }  
 updateSubtitle();  
 }  
  
 private class BookHolder extends RecyclerView.ViewHolder implements View.OnClickListener {  
  
 public TextView mTitleTextView;  
  
 public BookHolder(View itemView) {  
 super(itemView);  
 itemView.setOnClickListener(this);  
 mTitleTextView = (TextView)  
 itemView.findViewById(R.id.*list\_item\_book\_title\_text\_view*);  
 mDateTextView = (TextView)  
 itemView.findViewById(R.id.*list\_item\_book\_date\_text\_view*);  
 mReadedCheckBox = (CheckBox)  
 itemView.findViewById(R.id.*list\_item\_book\_readed\_check\_box*);  
 }  
  
 public void bindBook(Book book) {  
 mBook = book;  
 mTitleTextView.setText(mBook.getTitle());  
 mDateTextView.setText(mBook.getDate().toString());  
 mReadedCheckBox.setChecked(mBook.isReaded());  
 }  
  
 @Override  
 public void onClick(View view) {  
 Intent intent = BookPagerActivity.*newIntent*(getActivity(),  
 mBook.getId());  
 startActivity(intent);  
 }  
 }  
  
 private class BookAdapter extends RecyclerView.Adapter<BookHolder> {  
 private List<Book> mBooks;  
  
 public BookAdapter(List<Book> books) {  
 mBooks = books;  
 }  
  
 @Override  
 public BookHolder onCreateViewHolder(ViewGroup parent, int viewType) {  
 LayoutInflater layoutInflater = LayoutInflater.*from*(getActivity());  
 View view = layoutInflater.inflate(R.layout.*list\_item\_book*, parent, false);  
 return new BookHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(BookHolder holder, int position) {  
 Book book = mBooks.get(position);  
 holder.bindBook(book);  
 }  
  
 @Override  
 public int getItemCount() {  
 return mBooks.size();  
 }  
  
 public void setBooks(List<Book> books) {  
 mBooks = books;  
 }  
 }  
  
  
}

с помощью этого класса реализовано отслеживание нажатий и последующее открытие конкретного кастинга

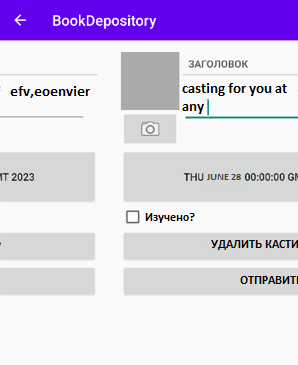




Тема 10.

Создание процедур обработки информации.

Создание viewpager – возможность листать кастинги свайпом вправо или влево.



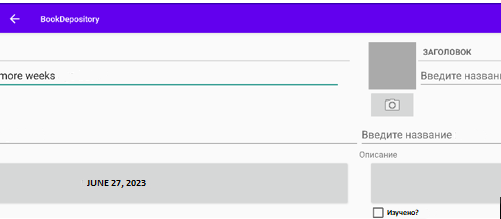
Для этого создали класс BookPagerActivity.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import android.content.Context;  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.MenuItem;  
  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentManager;  
import androidx.fragment.app.FragmentStatePagerAdapter;  
import androidx.viewpager.widget.ViewPager;  
  
import java.util.List;  
import java.util.UUID;  
  
public class BookPagerActivity extends AppCompatActivity {  
 private static final String *EXTRA\_BOOK\_ID* = "ru.rsue.deinekinroman1.bookdepository.book\_id";  
 private ViewPager mViewPager;  
 private List<Book> mBooks;  
  
 public static Intent newIntent(Context packageContext, UUID bookId) {  
 Intent intent = new Intent(packageContext,  
 BookPagerActivity.class);  
 intent.putExtra(*EXTRA\_BOOK\_ID*, bookId);  
 return intent;  
 }  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_book\_pager*);  
  
 UUID bookId = (UUID) getIntent()  
 .getSerializableExtra(*EXTRA\_BOOK\_ID*);  
  
 mViewPager = (ViewPager) findViewById(R.id.*activity\_book\_pager\_view\_pager*);  
 mBooks = BookLab.*get*(this).getBooks();  
 FragmentManager fragmentManager = getSupportFragmentManager();  
 mViewPager.setAdapter(new  
 FragmentStatePagerAdapter(fragmentManager) {  
 @Override  
 public Fragment getItem(int position) {  
 Book book = mBooks.get(position);  
 return BookFragment.*newInstance*(book.getId());  
 }  
  
 @Override  
 public int getCount() {  
 return mBooks.size();  
 }  
 });  
 }  
 @Override  
 public boolean onOptionsItemSelected(MenuItem item) {  
 finish();  
 return true;  
 }  
}

А также создали новый XML файл activity-book-pager.xml и добавили в него специальную разметку реализующую viewpager

<?xml version="1.0" encoding="utf-8"?>  
<androidx.viewpager.widget.ViewPager 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:id="@+id/activity\_book\_pager\_view\_pager"  
 />

Так выглядит интерфейс в альбомной ориентации



Тема 11.

Обеспечение дополнительной функциональности системы.

Ещё я сделал календарь выбора даты прочтения

Для этого создается класс DatePickerFragment.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import android.app.Activity;  
import android.app.AlertDialog;  
import android.app.Dialog;  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.widget.DatePicker;  
  
import androidx.fragment.app.DialogFragment;  
  
import java.util.Calendar;  
import java.util.Date;  
import java.util.GregorianCalendar;  
  
  
  
public class DatePickerFragment extends DialogFragment {  
  
 private static final String *ARG\_DATE* = "date";  
 private DatePicker mDatePicker;  
 public static final String *EXTRA\_DATE* =  
 "ru.rsue.deinekinroman1.bookdepository.date";  
 public static DatePickerFragment newInstance(Date date) {  
 Bundle args = new Bundle();  
 args.putSerializable(*ARG\_DATE*, date);  
  
 DatePickerFragment fragment = new DatePickerFragment();  
 fragment.setArguments(args);  
 return fragment;  
 }  
  
  
 @Override  
 public Dialog onCreateDialog(Bundle savedInstanceState) {  
 Date date = (Date) getArguments().getSerializable(*ARG\_DATE*);  
 Calendar calendar = Calendar.*getInstance*();  
 calendar.setTime(date);  
 View v = LayoutInflater.*from*(getActivity())  
 .inflate(R.layout.*dialog\_date*, null);  
 int year = calendar.get(Calendar.*YEAR*);  
 int month = calendar.get(Calendar.*MONTH*);  
 int day = calendar.get(Calendar.*DAY\_OF\_MONTH*);  
 mDatePicker = (DatePicker)  
 v.findViewById(R.id.*dialog\_date\_date\_picker*);  
 mDatePicker.init(year, month, day, null);  
 return new AlertDialog.Builder(getActivity())  
 .setView(v)  
 .setTitle(R.string.*date\_picker\_title*)  
 .setPositiveButton(android.R.string.*ok*,new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
 int year = mDatePicker.getYear();  
 int month = mDatePicker.getMonth();  
 int day = mDatePicker.getDayOfMonth();  
 Date date = new GregorianCalendar(year, month, day).  
 getTime();  
 sendResult(Activity.*RESULT\_OK*, date);  
 }  
 } ).create();  
  
  
 }  
 private void sendResult(int resultCode, Date date) {  
 if (getTargetFragment() == null) {  
 return;  
 }  
 Intent intent = new Intent();  
 intent.putExtra(*EXTRA\_DATE*, date);  
 getTargetFragment()  
 .onActivityResult(getTargetRequestCode(), resultCode,  
 intent);  
 }  
  
}

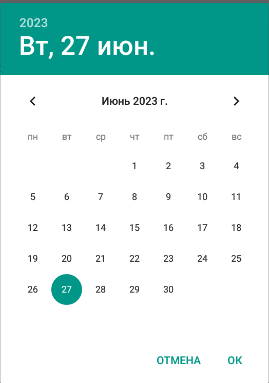
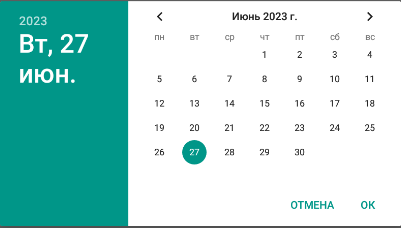
после этого надо добавить новые строковые ресурсы

<string name="book\_readed\_label">Изучено?</string>  
<string name="date\_picker\_title">Дата начала прочтения:</string>

Создал новый XML файл dialog\_date.xml

<?xml version="1.0" encoding="utf-8"?>  
<android.widget.DatePicker xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:id="@+id/dialog\_date\_date\_picker">  
</android.widget.DatePicker>

Результат

Тема 12.

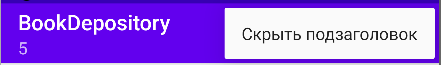
Intents

В этом задании я создал меню инструментов: добавление кастингов и скрытие подзаголовков, и кнопка назад

+ это добавление нового кастинга,



По нажатию на три точки открывается кнопка скрыть подзаголовок



Кнопка назад



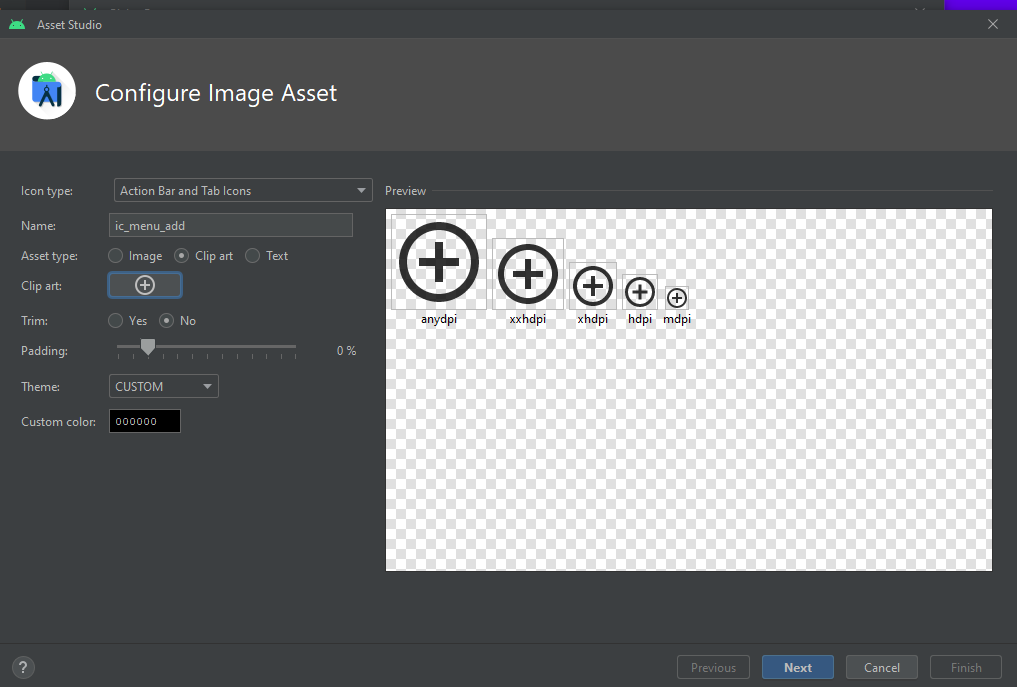
Добавляем строковые ресурсы

<string name="new\_book">Новый кастинг</string>  
<string name="show\_subtitle">Показать подзаголовок</string>  
<string name="hide\_subtitle">Скрыть подзаголовок</string>

Создаем новый XML файл fragment\_book\_list.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.recyclerview.widget.RecyclerView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/book\_recycler\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>

Находим подходящее изображение с помощью встроенного приложения ImageAsset

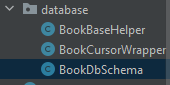


Тема 13.

Тестирование приложения

Создание бд SQLite

Создаем классы в пакете database



BookBaseHelper.java

package ru.rsue.deinekinroman1.bookdepository.database;  
  
import android.content.Context;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
  
public class BookBaseHelper extends SQLiteOpenHelper {  
  
 private static final int *VERSION* = 1;  
 private static final String *DATABASE\_NAME* = "bookBase.db";  
  
 public BookBaseHelper(Context context) {  
 super(context, *DATABASE\_NAME*, null, *VERSION*);  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 db.execSQL("create table " + BookDbSchema.BookTable.*NAME* + "(" +  
 " \_id integer primary key autoincrement, " +  
 BookDbSchema.BookTable.Cols.*UUID* + ", " +  
 BookDbSchema.BookTable.Cols.*TITLE* + ", " +  
 BookDbSchema.BookTable.Cols.*DATE* + ", " +  
 BookDbSchema.BookTable.Cols.*READED* + ")"  
 );  
 }  
 @Override  
 public void onUpgrade (SQLiteDatabase db,int i, int i1){

BookCursorWrapper.java

package ru.rsue.deinekinroman1.bookdepository.database;  
  
import android.database.Cursor;  
import android.database.CursorWrapper;  
  
import java.sql.Date;  
import java.util.UUID;  
  
import ru.rsue.deinekinroman1.bookdepository.Book;  
  
  
public class BookCursorWrapper extends CursorWrapper {  
 public BookCursorWrapper(Cursor cursor) {  
 super(cursor);  
 }  
 public Book getBook() {  
 String uuidString =  
 getString(getColumnIndex(BookDbSchema.BookTable.Cols.*UUID*));  
 String title = getString(getColumnIndex(BookDbSchema.BookTable.Cols.*TITLE*));  
 long date = getLong(getColumnIndex(BookDbSchema.BookTable.Cols.*DATE*));  
 int isReaded = getInt(getColumnIndex(BookDbSchema.BookTable.Cols.*READED*));  
 Book book = new Book(UUID.*fromString*(uuidString));  
 book.setTitle(title);  
 book.setDate(new Date(date));  
 book.setReaded(isReaded != 0);  
 return book;  
 }  
}

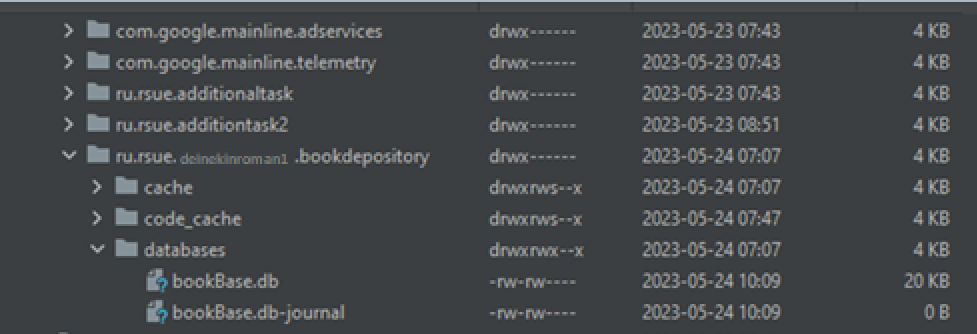
BookDbSchema.java

package ru.rsue.deinekinroman1.bookdepository.database;  
  
public class BookDbSchema {  
 public static final class BookTable {  
 public static final String *NAME* = "books";  
 public static final class Cols {  
 public static final String *UUID* = "uuid";  
 public static final String *TITLE* = "title";  
 public static final String *DATE* = "date";  
 public static final String *READED* = "readed";  
 }  
 }  
}

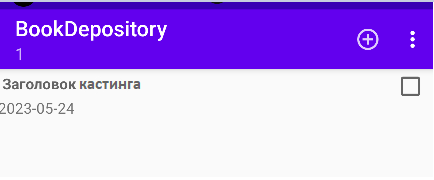
Тема 14.

Отладка приложения.

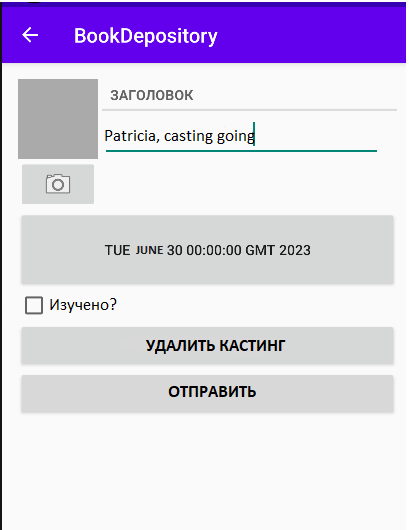
Проверяем создалась ли БД



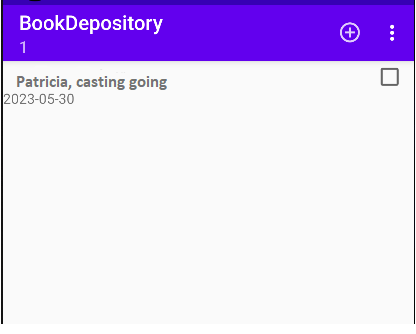
Проверяем добавляются ли кастинги



Добавляем название и дату прочтения и перезапускаем приложение



Все работает



Тема 14.

Отладка приложения.

Создание неявного интента

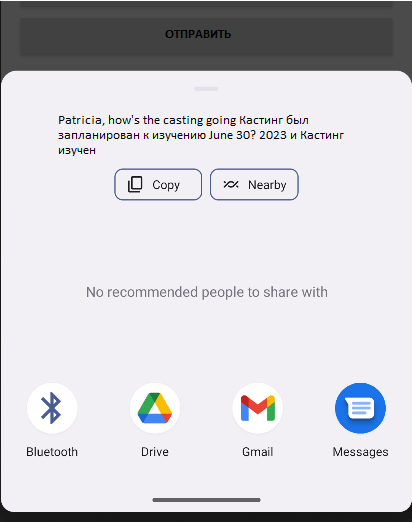
Добавляем строковые ресурсы

<string name="book\_report">%1$s  
 Книга была запланирована к прочтению %2$s и %3$s  
</string>  
 <string name="book\_report\_readed">Книга прочитана</string>  
 <string name="book\_report\_unreaded">Книга не прочитана</string>  
 <string name="book\_report\_subject">BookDepository отчет о прочтении  
 </string>  
 <string name="send\_report">Отправить отчет о прочтении через</string>

Добавляем код в BookFragment.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import android.app.Activity;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.graphics.Bitmap;  
import android.net.Uri;  
import android.os.Build;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.text.Editable;  
import android.text.TextWatcher;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.CheckBox;  
import android.widget.CompoundButton;  
import android.widget.EditText;  
import android.widget.ImageButton;  
import android.widget.ImageView;  
  
import androidx.core.content.FileProvider;  
import androidx.fragment.app.Fragment;  
import androidx.fragment.app.FragmentManager;  
  
import java.io.File;  
import java.text.DateFormat;  
import java.util.Date;  
import java.util.UUID;  
  
import ru.rsue.deinekinrom1.bookdepository.database.BookDbSchema;  
  
public class BookFragment extends Fragment {  
 private Book mBook;  
 private EditText mTitleField;  
 private Button mDateButton;  
 private CheckBox mReadedCheckBox;  
 private static final String *ARG\_BOOK\_ID* = "book\_id";  
 private static final String *DIALOG\_DATE* = "DialogDate";  
 private static final int *REQUEST\_DATE* = 0;  
 private Button mReportButton;  
 private File mPhotoFile;  
 private ImageButton mPhotoButton;  
 private ImageView mPhotoView;  
 private static final int *REQUEST\_PHOTO* = 1;  
  
  
 public static BookFragment newInstance(UUID bookId) {  
 Bundle args = new Bundle();  
 args.putSerializable(*ARG\_BOOK\_ID*, bookId);  
 BookFragment fragment = new BookFragment();  
 fragment.setArguments(args);  
 return fragment;  
 }  
  
 @Override  
 public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 UUID bookId = (UUID) getArguments().getSerializable(*ARG\_BOOK\_ID*);  
 mBook = BookLab.*get*(getActivity()).getBook(bookId);  
 mPhotoFile = BookLab.*get*(getActivity()).getPhotoFile(mBook);  
 }  
  
 @Override  
 public void onPause() {  
 super.onPause();  
 BookLab.*get*(getActivity()).updateBook(mBook);  
 }  
  
 private void updateDate() {  
 mDateButton.setText(mBook.getDate().toString());  
 }  
  
 private String getBookReport() {  
 String readedString = null;  
 if (mBook.isReaded()) {  
 readedString = getString(R.string.*book\_report\_readed*);  
 } else {  
 readedString = getString(R.string.*book\_report\_unreaded*);  
 }  
 String dateFormat = "EEE, MMM dd";  
 String dateString = DateFormat  
 .*getDateInstance*(DateFormat.*MEDIUM*).format(mBook.getDate());  
 String report = getString(R.string.*book\_report*,  
 mBook.getTitle(), dateString, readedString);  
 return report;  
 }  
  
 private void updatePhotoView() {  
 if (mPhotoFile == null || !mPhotoFile.exists()) {  
 mPhotoView.setImageDrawable(null);  
 } else {  
 Bitmap bitmap = PictureUtils.*getScaledBitmap*(  
 mPhotoFile.getPath(), getActivity());  
 mPhotoView.setImageBitmap(bitmap);  
 }  
 }  
  
 public void onActivityResult(int requestCode, int resultCode,  
 Intent data) {  
 if (resultCode != Activity.*RESULT\_OK*) {  
 return;  
 }  
 if (requestCode == *REQUEST\_DATE*) {  
 Date date = (Date) data  
 .getSerializableExtra(DatePickerFragment.*EXTRA\_DATE* );  
 mBook.setDate(date);  
 updateDate();  
 }else if (requestCode == *REQUEST\_PHOTO*) {  
 updatePhotoView();  
 }  
 }  
  
  
  
 @Override  
 public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
  
 View v = inflater.inflate(R.layout.*fragment\_book*, container, false);  
 mTitleField = (EditText) v.findViewById(R.id.*book\_title*);  
 mTitleField.setText(mBook.getTitle());  
 mTitleField.addTextChangedListener(new TextWatcher() {  
 @Override  
 public void beforeTextChanged(CharSequence charSequence, int i, int i1, int i2) {  
  
 }  
  
 @Override  
 public void onTextChanged(CharSequence charSequence, int i, int i1, int i2) {  
 mBook.setTitle(charSequence.toString());  
 }  
  
 @Override  
 public void afterTextChanged(Editable editable) {  
  
 }  
 });  
 mDateButton = (Button) v.findViewById(R.id.*book\_date*);  
 mDateButton.setText(DateFormat.*getDateInstance*().format(mBook.getDate()));  
  
 mDateButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 FragmentManager manager = getFragmentManager();  
 DatePickerFragment dialog = DatePickerFragment  
 .*newInstance*(mBook.getDate());  
 dialog.setTargetFragment(BookFragment.this,  
 *REQUEST\_DATE*);  
 dialog.show(manager, *DIALOG\_DATE*);  
 }  
 });  
  
 mReadedCheckBox = (CheckBox) v.findViewById(R.id.*book\_readed*);  
 mReadedCheckBox.setChecked(mBook.isReaded());  
 mReadedCheckBox.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
 @Override  
 public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
 mBook.setReaded(isChecked);  
 }  
 });  
  
 mReportButton = (Button) v.findViewById(R.id.*book\_report*);  
 mReportButton.setOnClickListener(new View.OnClickListener() {  
 public void onClick(View v) {  
 Intent i = new Intent(Intent.*ACTION\_SEND*);  
 i.setType("text/plain");  
 i.putExtra(Intent.*EXTRA\_TEXT*, getBookReport());  
 i.putExtra(Intent.*EXTRA\_SUBJECT*,  
 getString(R.string.*book\_report\_subject*));  
 i = Intent.*createChooser*(i, getString(R.string.*send\_report*));  
 startActivity(i);  
 }  
 });  
 mPhotoButton = (ImageButton) v.findViewById(R.id.*book\_camera*);  
 final Intent captureImage =  
 new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
 PackageManager packageManager = getActivity().getPackageManager();  
 boolean canTakePhoto = mPhotoFile != null ||  
 captureImage.resolveActivity(packageManager) != null;  
 if (canTakePhoto) {  
 Uri uri;  
 if (Build.VERSION.*SDK\_INT* < 24)  
 uri = Uri.*fromFile*(mPhotoFile);  
 else  
 uri = FileProvider.*getUriForFile*(getActivity(),  
 BuildConfig.*APPLICATION\_ID* + ".provider", mPhotoFile);  
 captureImage.putExtra(MediaStore.*EXTRA\_OUTPUT*, uri);  
 mPhotoButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 startActivityForResult(captureImage, *REQUEST\_PHOTO*);  
 }  
 });  
 mPhotoView = (ImageView) v.findViewById(R.id.*book\_photo*);  
 updatePhotoView();  
  
 }  
 return v;  
 }  
}

Тестируем



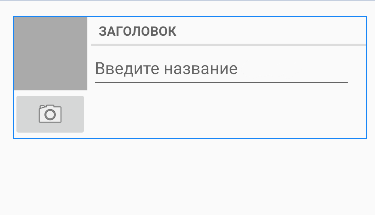
Тема 15.

Защита приложения. Оценка качества и надежности приложения.

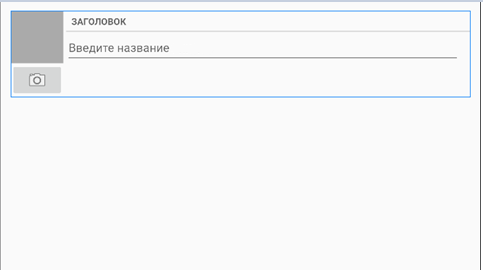
Создание для работы с камерой

Первым делом создаем XMl файл разметки который будет интегрирован в файл fragment\_book.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.appcompat.widget.LinearLayoutCompat xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginTop="16dp"  
 android:layout\_marginRight="16dp">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginRight="4dp"  
 android:orientation="vertical">  
  
 <ImageView  
 android:id="@+id/book\_photo"  
 android:layout\_width="80dp"  
 android:layout\_height="80dp"  
 android:background="@android:color/darker\_gray"  
 android:cropToPadding="true"  
 android:scaleType="centerInside" />  
  
 <ImageButton  
 android:id="@+id/book\_camera"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:src="@android:drawable/ic\_menu\_camera" />  
  
  
 </LinearLayout>  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:orientation="vertical">  
  
 <TextView  
 style="?android:listSeparatorTextViewStyle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/book\_title\_lable" />  
  
 <EditText  
 android:id="@+id/book\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginRight="16dp"  
 android:hint="@string/book\_title\_hint"  
 android:minHeight="48dp" />  
 </LinearLayout>  
</androidx.appcompat.widget.LinearLayoutCompat>



А также создаем аналогичный макет, но в альбомном формате



Тема 16.

Сопровождение программ

Добавляем код в BookLab.java

public File getPhotoFile(Book book) {  
 File externalFilesDir = mContext  
 .getExternalFilesDir(Environment.*DIRECTORY\_PICTURES*);  
 if (externalFilesDir == null) {  
 return null;  
 }  
 return new File(externalFilesDir, book.getPhotoFilename());  
}

Добавляем код в BookFragment.java

private File mPhotoFile;

@Override  
public void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 UUID bookId = (UUID) getArguments().getSerializable(*ARG\_BOOK\_ID*);  
 mBook = BookLab.*get*(getActivity()).getBook(bookId);  
 mPhotoFile = BookLab.*get*(getActivity()).getPhotoFile(mBook);  
}

Объявляем FileProvider в манифесте

<provider  
 android:name="androidx.core.content.FileProvider"  
 android:authorities="${applicationId}.provider"  
 android:exported="false"  
 android:grantUriPermissions="true">  
 <meta-data  
 android:name="android.support.FILE\_PROVIDER\_PATHS"  
 android:resource="@xml/provider\_paths"/>  
</provider>

Тема 17.

Сопровождение программ

Предоставляем доступ к внешнему хранилищу provider\_paths.xml

<?xml version="1.0" encoding="utf-8"?>  
<paths xmlns:android="http://schemas.android.com/apk/res/android">  
 <external-path name="external\_files" path="."/>  
</paths>

Добавляем код в BookFragment.java

private ImageButton mPhotoButton;  
private ImageView mPhotoView;

mPhotoButton = (ImageButton) v.findViewById(R.id.*book\_camera*);  
final Intent captureImage =  
 new Intent(MediaStore.*ACTION\_IMAGE\_CAPTURE*);  
PackageManager packageManager = getActivity().getPackageManager();  
boolean canTakePhoto = mPhotoFile != null ||  
 captureImage.resolveActivity(packageManager) != null;  
if (canTakePhoto) {  
 Uri uri;  
 if (Build.VERSION.*SDK\_INT* < 24)  
 uri = Uri.*fromFile*(mPhotoFile);  
 else  
 uri = FileProvider.*getUriForFile*(getActivity(),  
 BuildConfig.*APPLICATION\_ID* + ".provider", mPhotoFile);  
 captureImage.putExtra(MediaStore.*EXTRA\_OUTPUT*, uri);  
 mPhotoButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 startActivityForResult(captureImage, *REQUEST\_PHOTO*);  
 }  
 });  
 mPhotoView = (ImageView) v.findViewById(R.id.*book\_photo*);  
 updatePhotoView();

Создаем класс PictureUtils.java

package ru.rsue.deinekinroman1.bookdepository;  
  
import android.app.Activity;  
import android.graphics.Bitmap;  
import android.graphics.BitmapFactory;  
import android.graphics.Point;  
  
public class PictureUtils {  
 public static Bitmap getScaledBitmap(String path, Activity activity)  
 {  
 Point size = new Point();  
 activity.getWindowManager().getDefaultDisplay()  
 .getSize(size);  
 return *getScaledBitmap*(path, size.x, size.y);  
 }  
 public static Bitmap getScaledBitmap(String path, int destWidth,  
 int destHeight) {  
 // Чтение размеров изображения на диске  
 BitmapFactory.Options options = new BitmapFactory.Options();  
 options.inJustDecodeBounds = true;  
 BitmapFactory.*decodeFile*(path, options);  
 float srcWidth = options.outWidth;  
 float srcHeight = options.outHeight;  
  
 // Вычисление степени масштабирования  
 int inSampleSize = 1;  
 if (srcHeight > destHeight || srcWidth > destWidth) {  
 if (srcWidth > srcHeight) {  
 inSampleSize = Math.*round*(srcHeight / destHeight);  
 } else {  
 inSampleSize = Math.*round*(srcWidth / destWidth);  
 }  
 }  
 options = new BitmapFactory.Options();  
 options.inSampleSize = inSampleSize;  
  
 // Чтение данных и создание итогового изображения  
 return BitmapFactory.*decodeFile*(path, options);  
 }  
}

Тема 18.

Итоговое занятие

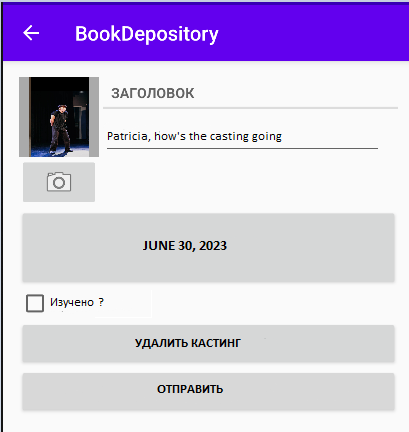
Даем разрешение приложению работать с камерой

<uses-feature android:name="android.hardware.camera"  
 android:required="false"  
 />

Тестируем приложение



Установил его для кастинга



Ссылка на проект BookDepository:

https://github.com/Romandeinekin/BookDepository\_Final\_Version.git