|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт информационных технологий |
| Кафедра инструментального и прикладного программного обеспечения |
|  |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ** | |
| **по дисциплине** | |
| «Разработка программных приложений» | |
|  | |
| Выполнил студент группы ИКБО-01-17 | Сухов И.А. |
| Принял старший преподаватель | Перова Ю.П. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №/р. | Страница в отчете. | Студент выполнил:  (подпись) | Преподаватель принял:  (подпись) | Дата |
| 1. |  |  |  |  |
| 2. |  |  |  |  |
| 3. |  |  |  |  |
| 4. |  |  |  |  |

|  |
| --- |
|  |
|  |

Москва 2019

Содержание

[Лабораторная работа №1 3](#_Toc10218520)

[Лабораторная работа №2 9](#_Toc10218582)

[Лабораторная работа №3 16](#_Toc10218646)

[Лабораторная работа №4. 22](#_Toc10218724)

# Лабораторная работа №1

**Цель работы:** ознакомиться с IDE Android Studio, создать первые activity, разобраться с жизненным циклом activity и его элементами. Научиться работе с потоками.

**Задачи:**

1. Реализовать приложение, состоящее из двух активити: Первое активити - это splash screen. Полноэкранное без ActionBar, c одной картинкой (или анимацией по желанию) по центру. Активити показывается 2 секунды, потом запускает второе и первое «умирает». Ожидание необходимо реализовать через Thread.sleep.
2. Второе активити представляет из себя список из 1000000 (один миллион) элементов, где четные элементы имеют серый фон (#CCCCCC), а нечетные — белый (#FFFFFF). Каждый элемент содержит картинку и текст. Картинка может быть любая, и она не меняется для элементов (выбирайте маленькую). Текст каждого элемента — это его индекс текстом (пример «сто двадцать»). Отсчет начинается с единицы. Шрифт большой.
3. Приложение должно нормально обрабатывать поворот экрана. При нажатии на клавишу back во втором активити приложение должно закрываться.

**Ход выполнения работы**

1. Было создано активити, которое после паузы в 2 секунды закрывается и вызывает основное activity, которое содержит основную часть задания.

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

getSupportActionBar().hide();

setContentView(R.layout.activity\_main);

Handler handler = new Handler();

handler.postDelayed(() -> {

try {

Thread.sleep(2000);

} catch (InterruptedException e) {

e.printStackTrace();

}

Intent intent = new Intent(getApplicationContext(), App.class);

startActivity(intent);

finish();

}, 300);

}

}

1. Было создано активити, содержащее в себе прокручиваемый список из 1 миллиона элементов, каждый из которых содержит картинку и свой номер в символьном виде.

public class App extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_app);

RecyclerView recyclerView = findViewById(R.id.my\_recycler\_view);

TIAdapter tiAdapter = new TIAdapter(1\_000\_000);

recyclerView.setAdapter(tiAdapter);

LinearLayoutManager linearLayoutManager = new LinearLayoutManager(this);

recyclerView.setLayoutManager(linearLayoutManager);

}

}

1. Реализован адаптер, который описывает элементы списка, который содержится в главном активити.

public class TIAdapter extends RecyclerView.Adapter<TIAdapter.ItemHolder> {

private final int itemNumber;

private convertIndex converter;

public TIAdapter(int itemNumber) {

this.itemNumber = itemNumber;

converter = new convertIndex();

}

@Override

public ItemHolder onCreateViewHolder(ViewGroup parent, int viewType) {

LayoutInflater inflater = LayoutInflater.from(parent.getContext());

View view = inflater.inflate(R.layout.recycler\_view\_item, parent, false);

return new ItemHolder(view);

}

@Override

public void onBindViewHolder(ItemHolder holder, int position) {

int number = position;

holder.textView.setText(converter.convert(number));

int backgroundColor = ContextCompat.getColor(holder.itemView.getContext(),

number % 2 == 0 ? R.color.gray : R.color.white);

holder.linearLayout.setBackgroundColor(backgroundColor);

}

1. Создан класс ItemHolder, описывающий поля, которые содержатся в элементе списка и используемый для заполнения элементов списка

public class ItemHolder extends RecyclerView.ViewHolder {

private final TextView textView;

private final LinearLayout linearLayout;

ItemHolder(View itemView) {

super(itemView);

textView = itemView.findViewById(R.id.textView1);

linearLayout = itemView.findViewById(R.id.line);

}

}

1. Был реализован метод конвертации индексов элементов в символьную форму. Метод работает триадами и таким образом позволяет конвертировать элементы от 0 до 1 миллиона, что и требовалось по условию задания.

**Пример работы:**



Рисунок 1. Splash Screen приложения

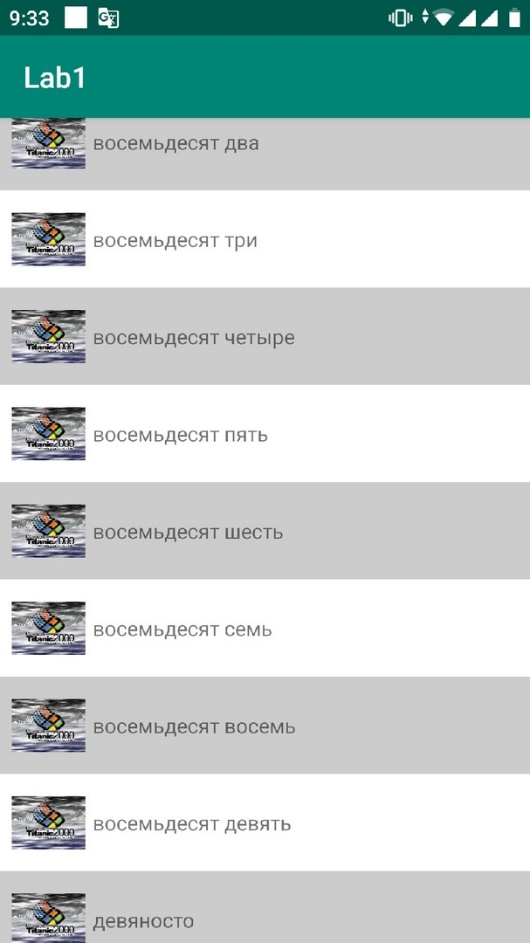


Рисунок 2. Основное окно приложения

**Вывод:**

В ходе выполнения лабораторной работы мы ознакомились с IDE Android Studio, а также научились создавать простые android приложения, состоящие из нескольких Activity.

**Библиографический список**

1. Введение в разработку приложений для ОС Android [Электронный ресурс] : учебное пособие / Ю.В. Березовская [и др.]. — Электрон. дан. — Москва : , 2016. — 433 с. — Режим доступа: https://e.lanbook.com/book/100707. — Загл. с экрана.

2. Ёранссон, А. Эффективное использование потоков в операционной системе Android [Электронный ресурс] / А. Ёранссон ; пер. с англ. Снастина А.В. — Электрон. дан. — Москва : ДМК Пресс, 2015. — 304 с. — Режим доступа: https://e.lanbook.com/book/93268. — Загл. с экрана.

3. Разработка приложений под мобильную платформу Android [Электронный ресурс] : учебное пособие / Д.В. Кравцов [и др.]. — Электрон. дан. — Москва : ФЛИНТА, 2018. — 72 с. — Режим доступа: https://e.lanbook.com/book/113495. — Загл. с экрана.

# Лабораторная работа №2

**Цель работы:** приобретение студентами практических навыков работы с внешними API, JSON-файлами, виджетом ViewPager и передачей данных между двумя Activity.

**Задание:**

1. Реализовать приложение, состоящее из двух активити: Первое активити — сплеш скрин из первой лабораторной, только показывается не больше чем время загрузки. Второе активити представляет собой список и ViewPager сформированный из json файла: https://raw.githubusercontent.com/wesleywerner/ancienttech/02decf875616dd9692b31658d92e64a20d99f816/src/data/techs.ruleset.json Этот JSON файл — массив «технологий» в игре Цивилизация.
2. В каждой технологии вас должны интересовать поля:

* graphic - картинка
* name – название
* helptext - дополнительная информация (может не быть)

1. Второе активити должно начинаться с фрагмента со списком, в элементе списка маленькая картинка 64dp и название. Список кликабельный. При клике на технологии открывается ViewPager c фрагментом, в котором соответствующая картинка большого размера (не больше ширины экрана20dp) и описанием технологии. При свайпе вправо описание должно меняться на описание предмета выше в списке, соответственно влево - на описание ниже в списке.
2. Опционально приложение может предоставлять Content Provider с данными технологий. Загрузка JSON должна происходить в момент показа сплеш скрина. Загрузка картинок, в процессе показа списка (естественно не в UI потоке), имена технологий не должны прыгать в процессе загрузки. Необходимо минимизировать использование трафика.

**Ход выполнения работы:**

* 1. Было реализовано активити, выполняющее роль splash screen. Активити завершается при получении сигнала от главного активити о завершении загрузки JSON файла.

public class SplashActivity extends AppCompatActivity {

BroadcastReceiver broadcastReceiver = new BroadcastReceiver() {

@Override

public void onReceive(Context ctx, Intent intent) {

if (Objects.equals(intent.getAction(), "close\_splash")) {

finish();

overridePendingTransition(android.R.anim.fade\_in, android.R.anim.fade\_out);

unregisterReceiver(broadcastReceiver);

}

}

};

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

registerReceiver(broadcastReceiver, new IntentFilter("close\_splash"));

setContentView(R.layout.activity\_splash);

getWindow().getDecorView().setSystemUiVisibility(

View.SYSTEM\_UI\_FLAG\_IMMERSIVE\_STICKY | View.SYSTEM\_UI\_FLAG\_FULLSCREEN | View.SYSTEM\_UI\_FLAG\_HIDE\_NAVIGATION

);

}

}

* 1. Второе активити состоит из двух фрагментов, которые меняются между собой при нажатии на: элемент в списке в фрагменте с ListView, кнопку «Назад» в фрагменте с ViewPager. В этом активити загружается JSON-файл, доступный по ссылке в задании

public class MainActivity extends AppCompatActivity {

public static int currentPosition;

private static final String KEY\_CURRENT\_POSITION = "KEY\_CURRENT\_POSITION";

private static final String DATA\_URL = "https://raw.githubusercontent.com/wesleywerner/ancient-tech/02decf875616dd9692b31658d92e64a20d99f816/src/data/techs.ruleset.json";

@Override

protected void onCreate(Bundle savedInstanceState) {

if (savedInstanceState == null) {

startActivity(new Intent(this, SplashActivity.class));

}

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

if (savedInstanceState != null) {

currentPosition = savedInstanceState.getInt(KEY\_CURRENT\_POSITION, 0);

loadData();

return;

}

loadData();

}

* 1. Активити содержит в себе метод loadData(), который отвечает за загрузку JSON-файла

private void loadData() {

JsonArrayRequest request = new JsonArrayRequest(Request.Method.GET,

DATA\_URL, null, new Response.Listener<JSONArray>() {

@Override

public void onResponse(JSONArray response) {

response.remove(0); // get rid of metadata on item 0

FragmentManager fragmentManager = MainActivity.this.getSupportFragmentManager();

ListFragment fragment = new ListFragment();

fragment.setData(response);

fragmentManager

.beginTransaction()

.add(R.id.fragment\_container, fragment, fragment.getClass().getSimpleName())

.commitAllowingStateLoss();

MainActivity.this.sendBroadcast(new Intent("close\_splash"));

}

}, new Response.ErrorListener() {

@Override

public void onErrorResponse(VolleyError error) {

error.printStackTrace();

MainActivity.this.sendBroadcast(new Intent("close\_splash"));

Toast.makeText(MainActivity.this.getApplicationContext(), "Failed to fetch data: " + error.getMessage(), Toast.LENGTH\_LONG).show();

}});

VolleyController.getInstance(getApplicationContext()).addToRequestQueue(request);

}

**Пример работы**



Рисунок 3. Splash Screen приложения

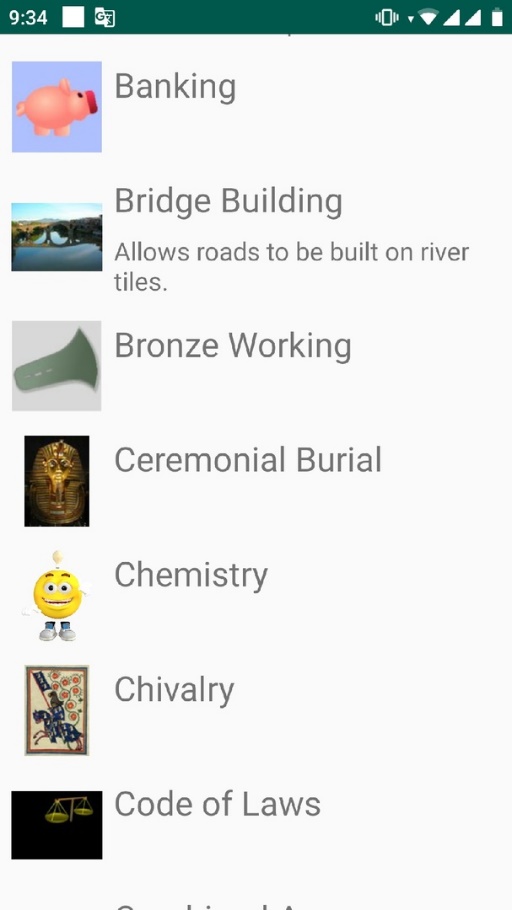


Рисунок 4. ListView приложения



Рисунок 5. ViewPager приложения

**Вывод:** В данной лабораторной работе были приобретены практические навыки работы с внешними API, JSON-файлами, виджетом ViewPager и передачей данных между двумя Activity.

**Библиографический список:**

1. Введение в разработку приложений для ОС Android [Электронный ресурс] : учебное пособие / Ю.В. Березовская [и др.]. — Электрон. дан. — Москва : , 2016. — 433 с. — Режим доступа: https://e.lanbook.com/book/100707. — Загл. с экрана.

2. Ёранссон, А. Эффективное использование потоков в операционной системе Android [Электронный ресурс] / А. Ёранссон ; пер. с англ. Снастина А.В. — Электрон. дан. — Москва : ДМК Пресс, 2015. — 304 с. — Режим доступа: https://e.lanbook.com/book/93268. — Загл. с экрана.

3. Разработка приложений под мобильную платформу Android [Электронный ресурс] : учебное пособие / Д.В. Кравцов [и др.]. — Электрон. дан. — Москва : ФЛИНТА, 2018. — 72 с. — Режим доступа: https://e.lanbook.com/book/113495. — Загл. с экрана.

# Лабораторная работа №3

#### Цель работы

Ознакомиться с возможностями создания и использования баз данных в приложениях для ОС Android. Получить практические навыки в работе с базами данных в рамках приложения.

#### Задачи

1. Создать приложение, взаимодействующее с базой данных. Первое активити должно содержать три кнопки. При нажатии на первую кнопку должно открываться новое активити, выводящее информацию из таблицы Студенты в удобном для восприятия формате.

При запуске приложения необходимо:

1. Создать БД, если ее не существует.

2. Создать таблицу Студенты, содержащую поля:

● ID

● ФИО

● Время добавления записи

3. Удалять все записи из БД, а затем вносить 5 записей об одногруппниках со случайными данными

При нажатии на вторую кнопку необходимо внести еще одну запись в таблицу.

При нажатии на третью кнопку необходимо заменить ФИО в последней внесенной записи на Иванов Иван Иванович.

1. Создать также новое отдельное приложение на основе приложения, созданного в части

Переопределить функцию onUpgrade. При изменении изменить таблицу Студенты следующим образом:

● ID

● Фамилия

● Имя

● Отчество

● Время добавления записи

Данные из поля ФИО необходимо по проблема разделить на три поля: Фамилия, Имя, Отчество. Также, изменить версию базы данных

#### Ход работы

1. В рамках выполнения первой части лабораторной работы было создано приложение, которое создаёт и заполняет базу данных записями о пятью студентах, если её не существует. Реализованы 3 кнопки, которые отвечают требованиям задания. При закрытии приложения база данных удаляется.

В примере кода показан адаптер списка, получающий данные из базы данных.

public class RWAdapter extends RecyclerView.Adapter<RWAdapter.MyViewHolder> { static class MyViewHolder extends RecyclerView.ViewHolder {

TextView t1;

TextView t2;

TextView t3;

MyViewHolder(LinearLayout v) {

super(v);

t1 = v.findViewById(R.id.student\_id);

t2 = v.findViewById(R.id.fullname);

t3 = v.findViewById(R.id.added);

}}

RWAdapter() {}

@NonNull

@Override

public RWAdapter.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

LinearLayout v = (LinearLayout) LayoutInflater.from(parent.getContext())

.inflate(R.layout.item, parent, false);

return new MyViewHolder(v);}

@Override

public void onBindViewHolder(@NonNull MyViewHolder holder, int position) {

Student student;

try {

student = HelperFactory.getHelper().getStudentDAO().queryForAll().get(position);

} catch (SQLException e) {

e.printStackTrace();

return;}

holder.t1.setText(String.valueOf(student.getId()));

holder.t3.setText(student.getFullname());

holder.t2.setText(student.getAdded().toString());

}

@Override

public int getItemCount() {

try {

return (int) HelperFactory.getHelper().getStudentDAO().countOf();

} catch (SQLException e) {

e.printStackTrace();

return 0;}}}}

1. В ходе выполнения второй части лабораторной работы основное приложение было модифицировано для обновления базы данных, которую оно создаёт при первом запуске. При обновлении ФИО разделяется на три колонки, а версия базы данных становится равной 2.

На примере кода показана база данных версии 2.

@DatabaseTable(tableName = "students")

public class Student {

@DatabaseField(generatedId = true)

private int id;

@DatabaseField(canBeNull = false)

private String name;

@DatabaseField(canBeNull = false)

private String lastname;

@DatabaseField(canBeNull = false)

private String middlename;

@DatabaseField(canBeNull = false)

private Date added;

public Student() {};

public int getId() {

return id;

}

public Date getAdded() {

return added;

}

public String getName() {

return name;

}

public String getLastname() {

return lastname;

}

public String getMiddlename() {

return middlename;

}

public void setName(String name) {

this.name = name;

}

public void setLastname(String lastname) {

this.lastname = lastname;

}

public void setMiddlename(String middlename) {

this.middlename = middlename;

}

public void setAdded(Date added) {

this.added = added;

}

**Пример работы:**

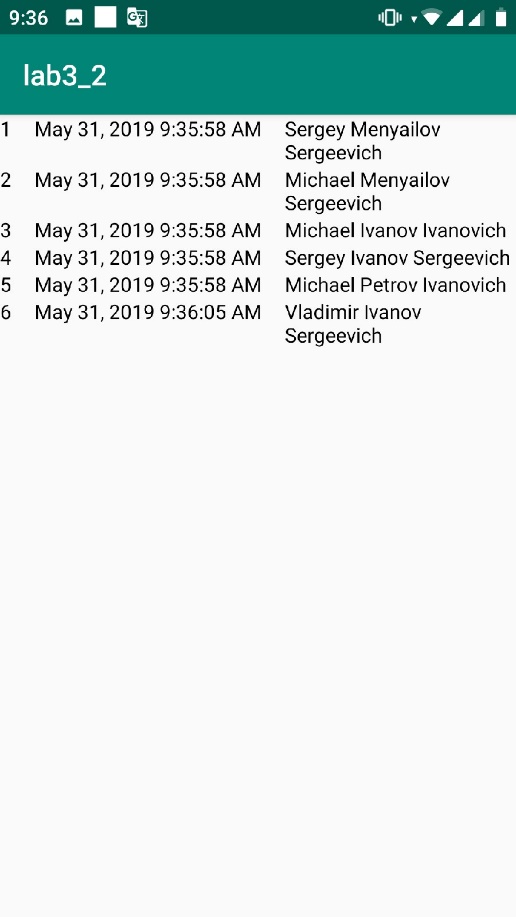


Рисунок 6. Отображение базы данных

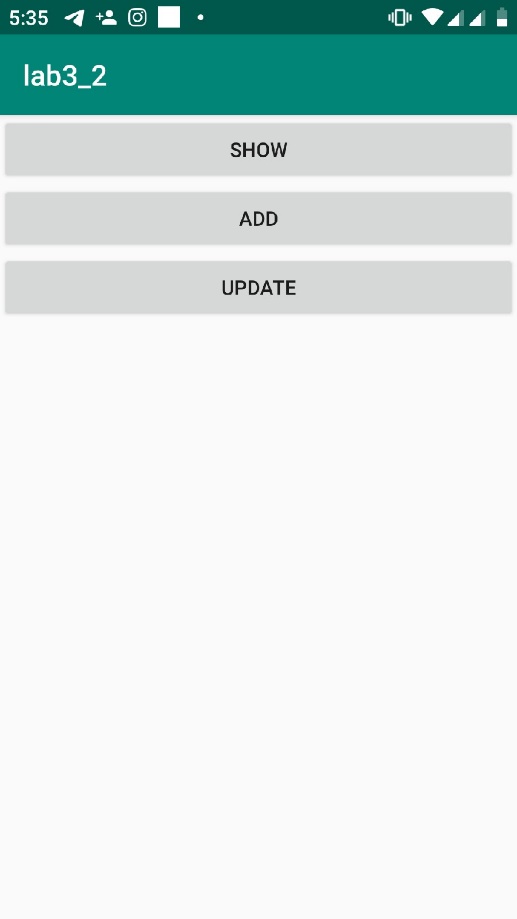


Рисунок 7. Главное активити

# Вывод

В результате выполнения лабораторной работы были получены практические навыки работы с базами данных при разработке приложений под ОС Android.

#### Библиографический список

1. Введение в разработку приложений для ОС Android [Электронный ресурс] : учебное пособие / Ю.В. Березовская [и др.]. — Электрон. дан. — Москва : , 2016. — 433 с. — Режим доступа: https://e.lanbook.com/book/100707. — Загл. с экрана.

2. Ёранссон, А. Эффективное использование потоков в операционной системе Android [Электронный ресурс] / А. Ёранссон ; пер. с англ. Снастина А.В. — Электрон. дан. — Москва : ДМК Пресс, 2015. — 304 с. — Режим доступа: https://e.lanbook.com/book/93268. — Загл. с экрана.

3. Разработка приложений под мобильную платформу Android [Электронный ресурс] : учебное пособие / Д.В. Кравцов [и др.]. — Электрон. дан. — Москва : ФЛИНТА, 2018. — 72 с. — Режим доступа: https://e.lanbook.com/book/113495. — Загл. с экрана.

# Лабораторная работа №4.

#### Цель работы

Получение практических навыков разработки виджетов для ОС Android на примере простого виджета для показа уведомлений.

#### Задачи

Реализовать widget, показывающий количество дней до установленной даты. По нажатию на сам виджет, отображать диалог для выбора даты. В указанный день, в 9:00, показывать уведомление о том, что событие наступило.

#### Ход работы

1. Был создан виджет 2x1, при нажатии на который открывается выбор даты через DatePickerDialog. При выборе даты в виджет устанавливается значение, равное количеству дней, оставшемуся до наступления до наступления выбранной даты. По истечении срока, в 9 часов утра будет получено уведомление, что подтверждает успешную работу программы

На примере кода представлен BroadcastReceiver, передающий данные между виджетом и activity, обрабатывающем события

public class Receiver extends BroadcastReceiver {

@Override

public void onReceive(Context ctx, Intent intent) {

if (intent.getAction() == null

|| intent.getExtras() == null

|| !intent.getAction().startsWith("updateDaysCounter")) {

return;

}

SharedPreferences sp = ctx.getSharedPreferences(ConfigActivity.WIDGET\_PREF, Context.MODE\_PRIVATE);

int widgetID = intent.getExtras().getInt("widgetID");

Calendar calendar = Utils.getCalendarFromPreferences(sp, widgetID);

if (calendar != null && Utils.getDaysUntil(calendar) < 0) {

Utils.createChannel(ctx);

Utils.showNotification(ctx, "It's now", widgetID);

}

AppWidgetManager appWidgetManager = AppWidgetManager.getInstance(ctx);

DaysUntilWidget.updateWidget(ctx, appWidgetManager, sp, widgetID);

}

}

На примере кода показан метод, обрабатывающий изменения в состоянии виджета

@SuppressLint("DefaultLocale")

static void updateWidget(Context context, AppWidgetManager appWidgetManager,

SharedPreferences sp, int widgetID) {

RemoteViews widgetView = new RemoteViews(context.getPackageName(), R.layout.widget);

Calendar calendar = Utils.getCalendarFromPreferences(sp, widgetID);

long daysBetween = calendar != null ? Utils.getDaysUntil(calendar) : -1;

if (daysBetween >= 0) {

widgetView.setTextViewText(R.id.tv, String.format("%d", daysBetween));

} else {

widgetView.setTextViewText(R.id.tv, "-");

}

Intent configIntent = new Intent(context, ConfigActivity.class);

configIntent.putExtra(AppWidgetManager.EXTRA\_APPWIDGET\_ID, widgetID);

PendingIntent configPendingIntent = PendingIntent.getActivity(context, widgetID, configIntent, 0);

widgetView.setOnClickPendingIntent(R.id.linear\_layout, configPendingIntent);

appWidgetManager.updateAppWidget(widgetID, widgetView);

Calendar alarmCalendar = Calendar.getInstance();

if (daysBetween > 0) {

alarmCalendar.add(Calendar.DATE, 1);

alarmCalendar.set(Calendar.HOUR\_OF\_DAY, 0);

alarmCalendar.set(Calendar.MINUTE, 0);

alarmCalendar.set(Calendar.SECOND, 0);

} else if (daysBetween == 0){

alarmCalendar.set(Calendar.HOUR\_OF\_DAY, ConfigActivity.HOUR\_TO\_TRIGGER);

alarmCalendar.set(Calendar.MINUTE, ConfigActivity.MINUTE\_TO\_TRIGGER);

alarmCalendar.set(Calendar.SECOND, 0);

} else {

return;

}

Utils.scheduleAlarmForWidget(context, widgetID, alarmCalendar);

}

#### Пример работы:

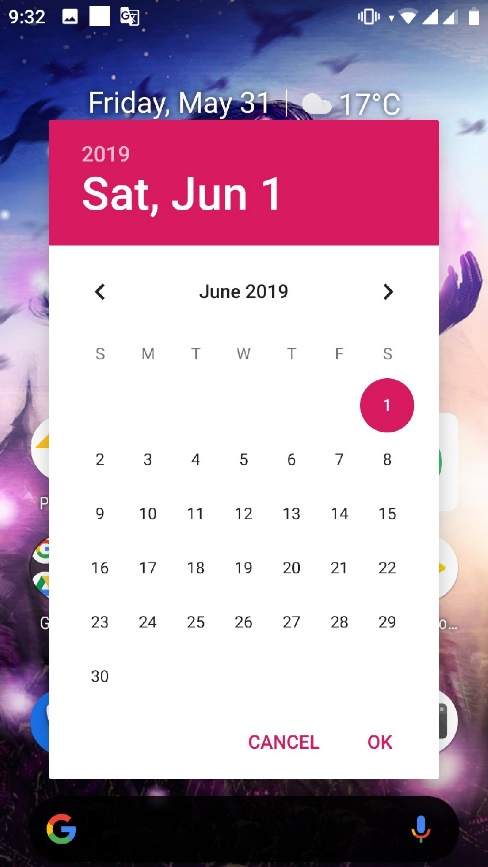


Рисунок 8. Выбор даты в виджете

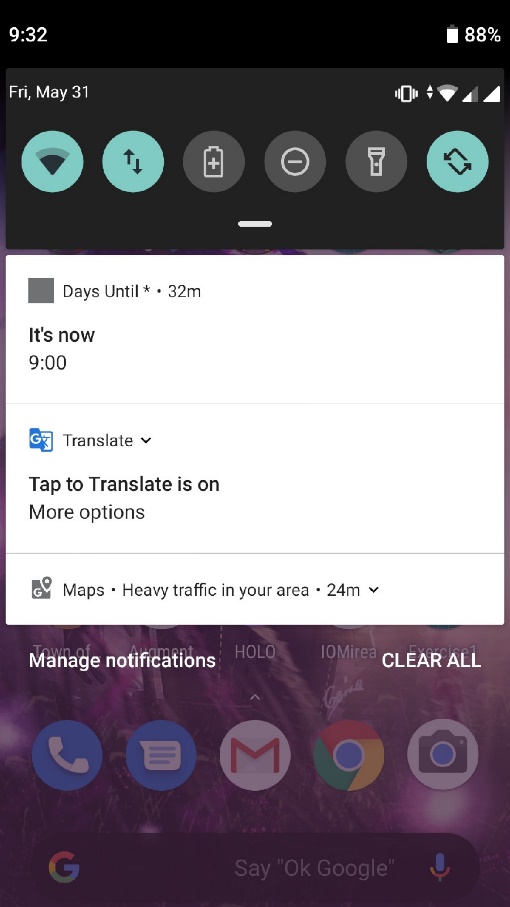


Рисунок 9. Уведомление, отправляемое приложением

#### Вывод

В результате выполнения данной лабораторной работы были получены практические навыки в работе с виджетами на примере разработки простого виджета для ОС Android.

#### Библиографический список

1. Введение в разработку приложений для ОС Android [Электронный ресурс] : учебное пособие / Ю.В. Березовская [и др.]. — Электрон. дан. — Москва : , 2016. — 433 с. — Режим доступа: https://e.lanbook.com/book/100707. — Загл. с экрана.

2. Ёранссон, А. Эффективное использование потоков в операционной системе Android [Электронный ресурс] / А. Ёранссон ; пер. с англ. Снастина А.В. — Электрон. дан. — Москва : ДМК Пресс, 2015. — 304 с. — Режим доступа: https://e.lanbook.com/book/93268. — Загл. с экрана.

3. Разработка приложений под мобильную платформу Android [Электронный ресурс] : учебное пособие / Д.В. Кравцов [и др.]. — Электрон. дан. — Москва : ФЛИНТА, 2018. — 72 с. — Режим доступа: https://e.lanbook.com/book/113495. — Загл. с экрана.