Пловдивски университет „Паисий Хилендарски”



Факултет по математика и информатика

Курсова работа

По дисциплина „Програмиране на приложения за мобилни устройства“

На тема: „Метеорологично приложение“

|  |  |
| --- | --- |
| Изготвил: Костадин Спиридонов Спиридонов  Специалност: СТД задочно  Факултетен номер: 1701682012 | Проверил:  / доц. д‐р Н. Касъклиев/ |

СЪДЪРЖАНИЕ

1. Увод
2. Oсновни функционалности
3. Използвани технологии и библиотеки
4. Потребителски инструктаж
5. Архитектура на приложението
6. Имплементация (Извличане и Съхранение)
7. Заключение
8. Линкове и Контакти

**Увод**

Целта на курсовия проект е да се изгради мобилно приложение показващо текущото време в избран град, записвайки история на всички търсения.

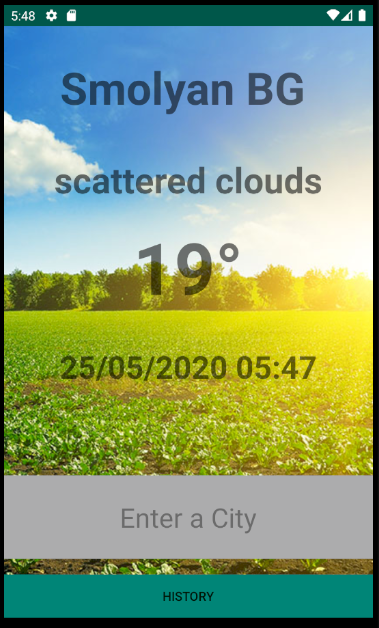
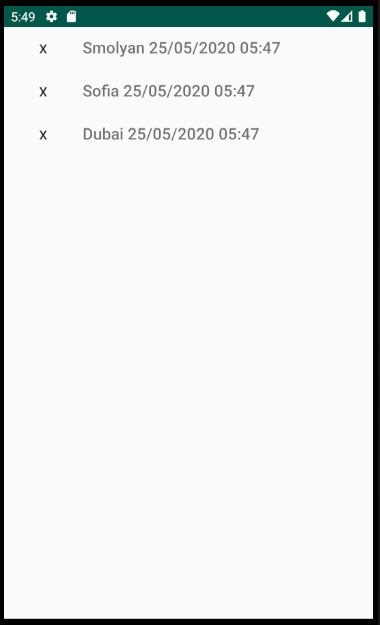
**Основни функционалности**

* Намиране и извеждане на времето на град по име
* Показване на времето от последния търсен град
* Записването на история от търсенията в база данни за свеждане
* Изтриване на записи от историята

**Използвани технологии и библиотеки**

* Android Studio
* Java
* Android SDK
* GSON – Десериализация на обекти
* SQLite - Използван за работа с база данни - взимането на данните за търсения, запазването им, извеждането им за четене и изтриване при заявка на потребителя.

**Потребителски инструктаж**

При стартиране, приложението ще трябва да въведете град в текстово поле. След успешно въвеждане на екрана ще се заредят метеорологични данни за въведения град.

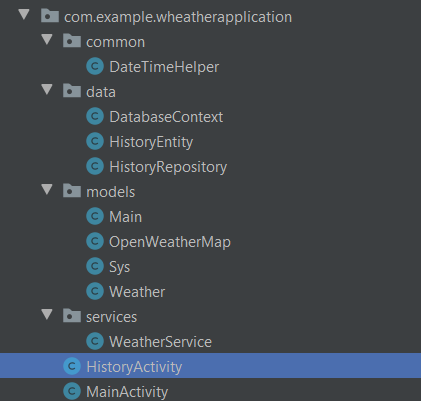
При всяко следващо въвеждане на нов град, на екрана се виждат данните за последно въведения такъв.

При всяко избиране на нов град, приложението запазва история. От бутона History потребителят може да разгледа всички предишни търсения и да изтрие ненужните.

Данните, които се визуализират на екрана са:

* Име на град и съкращение на държава
* Статус на времето
* Температура
* Последен запис

**Архитектура на приложението**



* Common – пакет в който се позиционират всички помощни класове
  + DateTimeHelper – клас предоставят функции за работа с дати
* Data – абстрактно ниво за работа с дани
  + DatabaseContext – клас отворен за работа с SQLLite база, поддържане и създаването й
  + HistoryEntity – домейн клас репрезентиращ една History таблица
  + HistoryRepository – специфично репозитори предоставящо функции за работа с данните в History таблицата
* Models – пакетът в който се държат основните модели с които работи мобилното приложение
* Services – пакет съдържащ всички сървиси работещи с външни източници
  + WeatherService – клас отговорен за извличане на метеорологични данни от външен HTTP източник
* HistoryActivity – активити репрезентиращо History екран
* MainActivty – активити репрезентиращо главния екран

**Имплементация (Извличане и Съхранение)**

1. При въведен град се пуска асинхронна задача за извличане на данни от сървъра, визуализация и записване в базата

String city = editCity.getText().toString();  
if(city.isEmpty() || city == null){  
 return false;  
}  
new GetWeatherTask().execute(city);

1. С помощта на WeatherService-а се извличат данните от сървъра

@Override

protected OpenWeatherMap doInBackground(String... params) {

String city = params[0];

WeatherService http = new WeatherService();

return http.getData(city);

}

1. С помощта на метода по-долу данните се визуализират и запазват

@Override

protected void onPostExecute(OpenWeatherMap openWeatherMap) {

super.onPostExecute(openWeatherMap);

if(openWeatherMap == null){

pd.dismiss();

displayError("Could Not Find City");

return;

}

if(save){

save(openWeatherMap);

}

show(openWeatherMap);

pd.dismiss();

}

1. Репозиторито е отговорно да ги съхрани в базата

public long insert(HistoryEntity entity){

ContentValues contentValues = new ContentValues();

contentValues.put(HistoryEntity.CityColumn, entity.getCity());

contentValues.put(HistoryEntity.UpdateDateColumn, entity.getUpdateDate());

return db.insert(HistoryEntity.TableName, null, contentValues);

}

1. След като вече са съхранение отново с репозиторито може да бъдат извлечени

public ArrayList<HistoryEntity> getAll(){

ArrayList<HistoryEntity> data = new ArrayList<HistoryEntity>();

Cursor cursor = this.getSelectQueryCursor();

while(cursor.moveToNext()){

data.add(this.parse(cursor));

}

return data;

}

1. Визуализиране в графичния интерфейс

private void refresh() {

List<HistoryEntity> data = historyRepository.getAll();

historyLayout.removeAllViews();

for(HistoryEntity entity : data){

final LinearLayout layout = new LinearLayout(this);

layout.setOrientation(LinearLayout.HORIZONTAL);

final TextView tv = new TextView(this);

tv.setText(entity.getCity() + " " + entity.getUpdateDate());

tv.setTextAppearance(this, android.R.style.TextAppearance\_DeviceDefault\_Medium);

Button btnDelete = new Button(this);

btnDelete.setText("X");

btnDelete.setBackgroundColor(Color.TRANSPARENT);

btnDelete.setTag(entity.getId());

btnDelete.setOnClickListener(onDeleteClicked);

layout.addView(btnDelete);

layout.addView(tv);

historyLayout.addView(layout);

}

}

**Заключение**

Курсовият проект даде добра основа за разработване на мобилни приложения с помощта на Java и Android Studio, полезни за бъдещо развитие.

Приложението предоставя интуитивен и лесен за интерфейс, с помощта на който потребителите лесно могат да получат необходимата за тях метеорологична информация.

**Линкове и Контакти**

* Github Repo - <https://github.com/kostadinSpiridonov/WeatherApplication>
* OpenWeatherMap (Използван API) - <https://openweathermap.org/>
* Email – [kostadin.spiridonov@gmail.com](kostadin.spiridonov@gmail.com%20%20)