

Aqua Reminder		
Projektowanie aplikacji mobilnych	Prowadzący: dr. Tomasz Wesołowski	
Wykonał: Patryk Biegun	PAW3 Rok III	Data: 1.06.2022

## Wstęp:

### 1.1 Treść zadania

Zadaniem do wykonania jest aplikacja mobilna posiadająca typowe dla niej cechy, takie jak ( powiadomienia, wibracje, wykorzystywanie modułu GPS, akcelerometru itp.) Aplikacja ta ma być funkcjonalna oraz ma zostać zrealizowana w technologii mobilnej np. Android lub dowolnej innej.

### 1.2

Celem pracy było zaprojektowanie i implementacja kompleksowej aplikacji mobilnej odpowiadającej za automatyczne dobieranie ilości wody oraz przypominanie o jej spożywaniu dostosowanej do danego użytkownika w oparciu o uzyskane dane.

### 1.3

Wymagania funkcjonalne to liczenie ilości wypitej wody, wyliczanie ile wody dany użytkownik powinien wypić oraz regularne przypominanie o piciu wody za pomocą powiadomień, język dopasowany do języka systemowego.

#### Aplikacja powinna:

- Mieć intuicyjny interface
- Wysyłać powiadomienia/przypomnienia
- Pokazywać Ile wody zostało do wypicia

### 1.4

Wymagania нефункционалне pełna mobilność, zapisywanie danych między sesjami

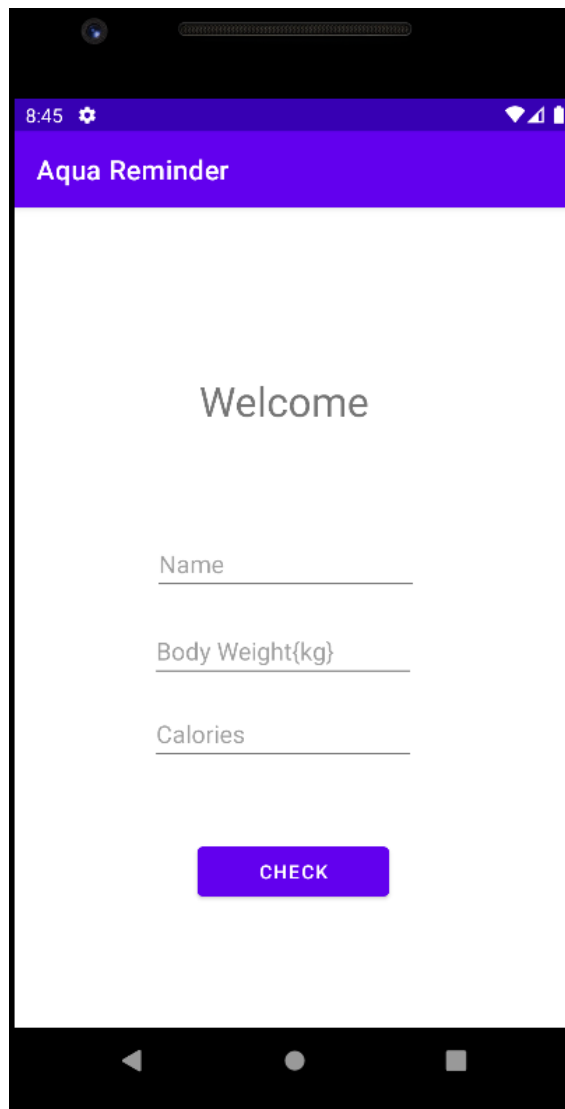
- Aplikacja ma być dostępne w systemie całą dobę
- Aplikacja jest dla każdego bez ograniczeń wieku
- Maksymalnie jedna osoba może korzystać z aplikacji

## Specyfikacja zewnętrzna

1.0 Aplikacja jest bardzo prosta i intuicyjna, zaczynamy na Ekranie gdzie mamy uzupełnić 3 podstawowe dane

Imię – tak aplikacja będzie się do nas zwracać

Masa Ciała i Dziennie spożywane kalorie - na ich podstawie wyliczam ilość wody



1.1 Panel Edycji, tutaj wpisujemy swoje dane

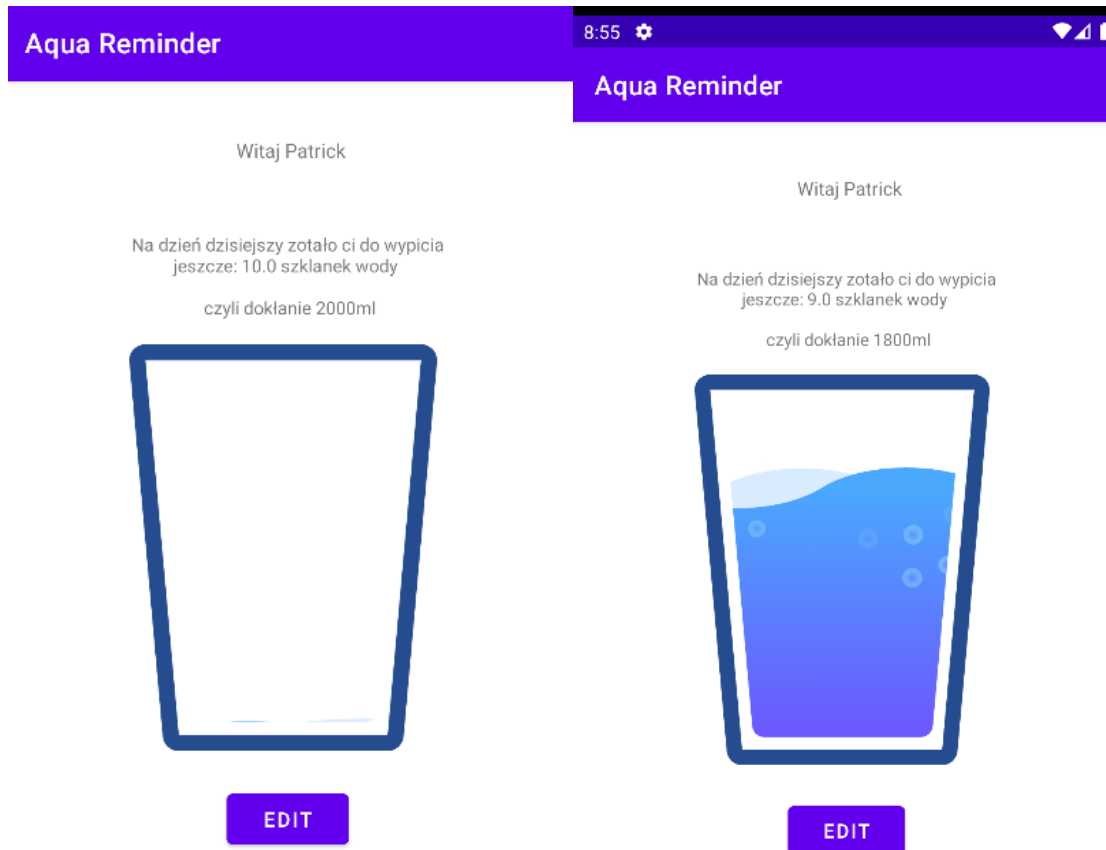
Musimy uzupełnić każdą ze zmiennych

Potem za pomocą przycisku zapisujemy swoje dane i przechodzimy dalej

W ekranie głównym mamy informację ile Wody nam zostało na dziś do wypicia i przeliczenie tej ilości na szklanki wody (200ml)

Kliknięcie w szklankę powoduje usunięcie jednej szklanki oraz odtworzenie animacji napełniania się wodą

Niżej jest przycisk powrotu na stronę Edycji



1.2 Ekran główny (szklanka pusta)

1.3 Ekran główny (szklanka pełna)

Kliknięcie w szklankę ustawia licznik który przypomni nam o piciu wody za 30 minut, ekranu nie da się obrócić

Aplikacja będzie nam wysyłać codziennie powiadomienie o 8:30

## 2.0 Specyfikacja wewnętrzna

Aby wszystkie funkcje były aktywne kod trzeba zsynchronizować z java 1.8

```
compileOptions {  
    sourceCompatibility JavaVersion.VERSION_1_8  
    targetCompatibility JavaVersion.VERSION_1_8  
}
```

Oraz

```
defaultConfig {  
    applicationId "com.example.aquareminder"  
    minSdk 26  
    targetSdk 32  
    versionCode 1  
    versionName "1.0"  
}
```

Activity\_main.xml

Warstwa graficzna panelu edytowania danych

Na starcie używam RelativeLayout oraz ustawiam całość grawitacji na środek

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res/app"   
    xmlns:tools="http://schemas.android.com/tools"   
    android:layout_width="wrap_content"   
    android:layout_height="match_parent"   
    android:gravity="center"   
    tools:context="com.example.aquareminder.MainActivity"   
>  
      
</RelativeLayout>
```

2.1 Layout (ustawienia)

Do Wpisywania tekstów używam standardowego okienka

EditText

Jako tekst wpisywany ustawiam definiowany odgórnie String żeby potem móc łatwo tłumaczyć aplikację na inne języki

Pozycję tego tekstu ustawiam manualnie za pomocą Panelu z wizualizacją aplikacji

Ustawiam też maksymalne wartości

Imię - max12 znaków

```
<EditText  
    android:id="@+id/inputWeight"  
    android:layout_width="194dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentStart="true"  
    android:layout_alignParentTop="true"  
    android:layout_centerHorizontal="true"  
    android:layout_marginStart="100dp"  
    android:layout_marginTop="304dp"  
    android:autoFillHints="70"  
    android:hint="@string/masa_ciala"  
    android:inputType="number"  
    android:maxLength="12"  
    tools:ignore="TextFields">  
      
</EditText>
```

2.2 EditText

Ustawiam tutaj też przyciski

Standardowy przycisk z ID

```
<Button
    android:id="@+id/check"
    android:layout_width="141dp"
    android:layout_height="wrap_content"
    android:layout_alignParentStart="true"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginStart="134dp"
    android:layout_marginBottom="91dp"
    android:text="Zatwierdź">

</Button>
```

2.3 przycisk

MainActivity.java

Tworząc klasę implementuję i pobieram zmienne, następnie wyszukuje je za pomocą ID i przypisuje im metody

```
public class MainActivity extends AppCompatActivity {

    private TextView welcome;
    private EditText inputName;
    private EditText inputWeight;
    private String Weight, Calories, Name;
    private EditText inputCalories;
    SharedPreferences sp;
    private Button check;
    private int waterquantity = 1600, i = 1600, waterconst ;
```

3.1 tworzenie zmiennych w pliku

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    waterquantity = 1600;

    welcome = findViewById(R.id.welcome);
    inputName = findViewById(R.id.inputName);
    inputWeight = findViewById(R.id.inputWeight);
    inputCalories = findViewById(R.id.inputCalories);
```

3.2 importowanie zmiennych

Teraz opiszę w jaki sposób dbam o to żeby dane z aplikacji się nie resetowały kiedy ją zamykamy

Używam do tego funkcji androida o nazwie „Shared Preferences”

Na początek używam tego aby pominąć ekran Edytowania, za pomocą warunku dla zmiennej „reminder” która jest zmienną typu boolean

```
sp = getSharedPreferences( name: "MyUserPrefs", Context.MODE_PRIVATE);

SharedPreferences preferences = getSharedPreferences( name: "MyUserPrefs", MODE_PRIVATE);
String remember_value = preferences.getString( s: "remember", s1: "");
if(remember_value.equals("true")){
    openWaterpanel();
}else if(remember_value.equals("false")){
    Toast.makeText( context: MainActivity.this, text: "Edytuj swoje dane", Toast.LENGTH_SHORT).show();
}
```

3.3 Funkcja sprawdzająca czy mam już zapisane dane użytkownika

```
public void openWaterpanel(){
    Intent intent = new Intent( packageContext: this, Waterpanel.class);
    startActivity(intent);
}
```

3.3.1 funkcja otwierająca następny panel

```
SharedPreferences.Editor editor = sp.edit();
```

3.4 inicjowanie SharedPreferences

Za pomocą If sprawdzam czy wszystkie zmienne są wypełnione, jeżeli tak to zapisuje je za pomocą Shared Preferences oraz ustawiam „remember” na „true” aby przy kolejnym włączeniu aplikacji pominąć etap wpisywania danych

```
if(!Weight.isEmpty() && !Calories.isEmpty() && !Name.isEmpty()){
    editor.putString( s: "Name", Name);
    editor.putString( s: "Calories", Calories);
    editor.putString( s: "Weight", Weight);
    editor.putInt ( s: "watercraft", waterquantity);
    editor.putInt ( s: "waterconst", waterconst);
    editor.putString( s: "remember", s1: "true");

    editor.commit();
    Toast.makeText( context: MainActivity.this, text: "Informacje zapisane", Toast.LENGTH_LONG).show();
    openWaterpanel();
}else{
    Toast.makeText( context: MainActivity.this, text: "Musisz uzupełnić swoje dane!", Toast.LENGTH_SHORT).show();
}
```

3.4.1 Funkcja zapisująca dane za pomocą SharedPreferences

Funkcja przypisana do przycisku, która włącza się w momencie jego naciśnięcia. Algorytm wylicza ilość wody na podstawie wagi lub kalorii, i przypisuje wartość z zakresu 1600-3000

```
@Override
public void onClick(View view) {

    Weight = inputWeight.getText().toString();
    Calories = inputCalories.getText().toString();
    Name = inputName.getText().toString();
    SharedPreferences.Editor editor = sp.edit();

    if(Integer.parseInt ( Calories ) > 3000 || Integer.parseInt ( Weight ) > 100 ){
        waterquantity = 3000;
    }else
    {
        if(Integer.parseInt ( Calories ) > Integer.parseInt ( Weight ) * 30 && Integer.parseInt ( Calories ) > waterquantity){
            {waterquantity = Integer.parseInt ( Calories );}
        }else
        {
            if(Integer.parseInt ( Weight ) * 30 > waterquantity )
            {waterquantity = Integer.parseInt ( Weight ) * 30;}

            for (i = waterquantity; i % 200 != 0; i=i+1){
                waterquantity = i+1;
            }

        }

        waterconst = waterquantity;
    }
}
```

### 3.5 funkcja zapisująca oraz licząca

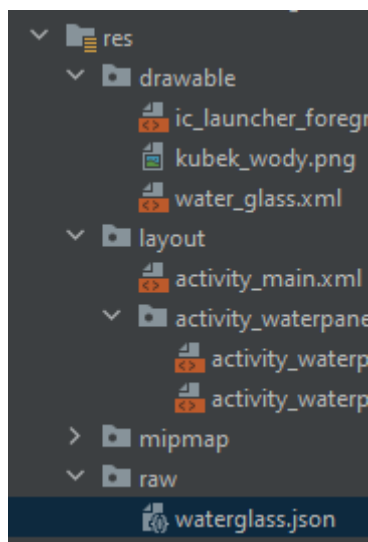
Opiszę również użycie zewnętrznej biblioteki w tym wypadku „Lottie” która pozwala na implementowanie animacji w formie grafik, oraz odtwarzanie ich przy konkretnych warunkach.

```
def lottieVersion = "5.0.3"
implementation "com.airbnb.android:lottie:$lottieVersion"
```

### 4.1 Implementacja lottie w build.grade

Odpowiednie pliki można pobrać na ich stronie, następnie umieścić je w folderze „raw”

Z rozszerzeniem „json”



### 4.2 implementacja obrazków

Następnie implementuje je w pliku xml, konkretnie w activity\_waterpanel.xml

Przydzielam jej Id, ustawiam relatywnie przeciągając na podglądzie, oraz za pomocą „Lottie” wybieram który plik ma się pojawić

```
<com.airbnb.lottie.LottieAnimationView
    android:id="@+id/lottieGlass"
    android:layout_width="wrap_content"
    android:layout_height="412dp"
    android:layout_alignParentStart="true"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_centerInParent="true"
    android:layout_marginStart="17dp"
    android:layout_marginEnd="0dp"
    android:layout_marginBottom="62dp"
    app:lottie_rawRes="@raw/waterglass" />
```

#### 4.3 wstawienie obrazka na stronę

W kodzie Waterpanel.java ustalam funkcjonalność tego przycisku

```
SharedPreferences sp;
LottieAnimationView water_glass;
```

```
water_glass = findViewById (R.id.lottieGlass);
```

Tutaj za pomocą komendy „water.glass.playanimation()” odtwarzam animację szklanki oraz naliczam wypitą wodę

Jeżeli wody już nie ma to wyświetlam tylko komunikat o tym jak się skończyła

Zmienna „watercount” pozwala na liczenie ile wody nam zostało do wypicia i zmniejsza się z każdym odtworzeniem animacji, może być też ponownie ustawiona na maksymalną ilość wody dla danego użytkownika jeżeli na dany dzień woda się skończy

```
water_glass.setOnClickListener (v -> {
    if(watercount > 0)watercount = watercount - 200;
    SharedPreferences.Editor editor = sp.edit();
    editor.putInt ( "watercraft", watercount);
    editor.commit();
    if(watercount!=0) {
        t2.setText ( "Na dzień dzisiejszy zostało ci do wypicia jeszcze: " + (float) watercount / 200 + " szklanek wody \n \n czyli dokładnie " + watercount + "ml" );
    }else {
        t2.setText ( "Gratuluję! Cel na dzisiaj spełniony, jesteś świetnie nawodniony \n Kliknij dalej aby zresetować licznik" );
        again.setVisibility ( View.VISIBLE );
    }

    water_glass.playAnimation ();
    if(watercount > 0) {
        Toast.makeText ( context, this, text: "Woda wypita!", Toast.LENGTH_LONG).show ();
        sendNote ();
    }
});
```

#### 4.4 przypisanie animacji oraz algorytmu picia wody do obrazka



Opis każdego Pliku aplikacji

## App/java

**MainActivity** – znajduje się tutaj panel do wpisywania danych użytkownika

**ReminderBroadcast** – tutaj tworzony jest kanał powiadomień

**Waterpanel** – tutaj są przystki funkcjonalności aplikacji

**res/drawable** - Katalog zawierający pliki obrazów, czyli zdjęcie główne aplikacji

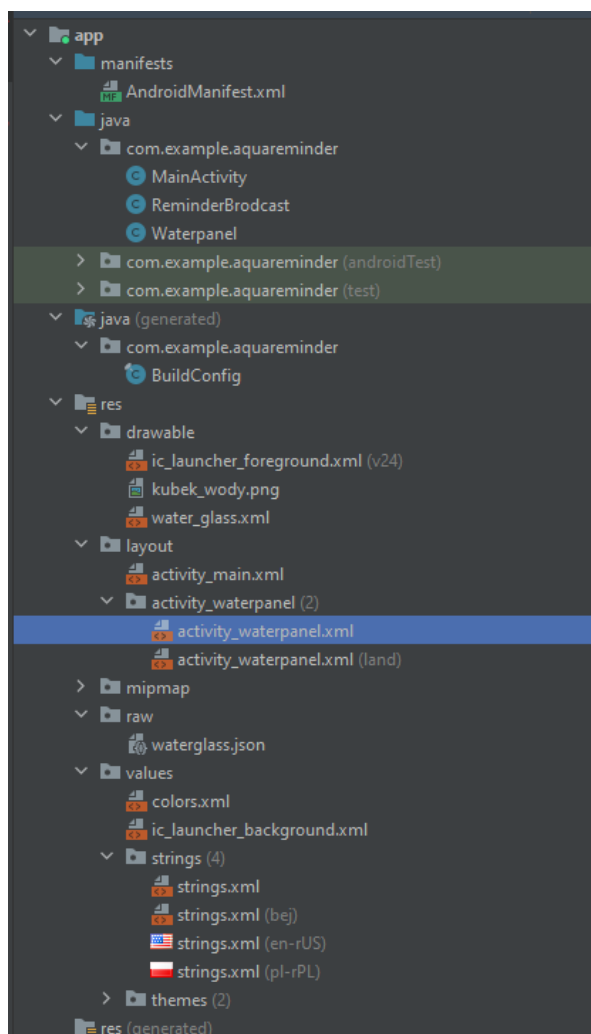
**res/raw** – zawiera animacje potrzebe do lottie

**values/string** – zawiera zmienne tekstów oraz ich tłumaczenia

**AndroidManifest.xml** - Plik manifestu aplikacji. Znajdują się tutaj informacje o wersji aplikacji

**res/layout** - Katalog zawierający widoki aplikacji. Jest tutaj plik „activity\_main.xml” oraz „waterpanel.xml”

Widok całej aplikacji



5.1 Struktura całej aplikacji

## Wnioski

Najważniejsze przy projektowaniu aplikacji okazało się zrozumienie jak działa cykl życia aplikacji oraz to jak za pomocą Shared Preferences mogę przesyłać zmienne między aktywnościami. Całość projektu zdecydowanie przyszła mi łatwo za pomocą mnóstwa dostępnych kursów oraz poradników w Internecie. Język jest bardzo intuicyjny i ciekawy.

Ważne było również ustawienie powiadomień, szczególnie funkcja „calendar” która pozwala mi na wysyłanie wiadomości o danej godzinie. Pozwala mi to na przypominanie użytkownikowi o aplikacji nawet kiedy użytkownik z niej nie korzysta