AKADEMIA NAUK STOSOWANYCH
W NOWYM SĄCZU

Wydział Nauk Inżynieryjnych
Katedra Informatyki

DOKUMENTACJA PROJEKTOWA

PROGRAMOWANIE URZĄDZEŃ MOBILNYCH

Aplikacja Survivalowa

Autor:
Bogusław Gruca
Karol Kowalik

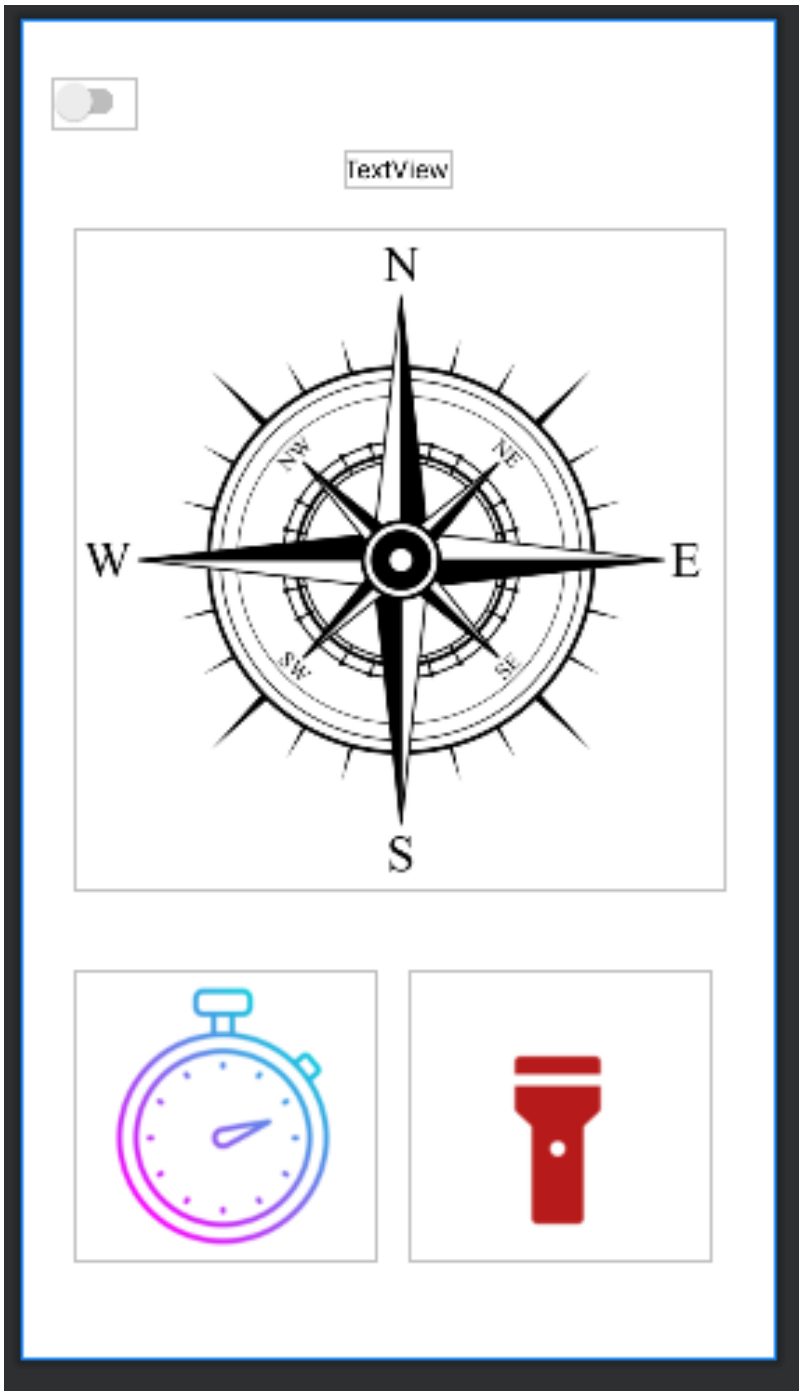
Prowadzący:
mgr inż. Dawid Kotlarski

Nowy Sącz 2022

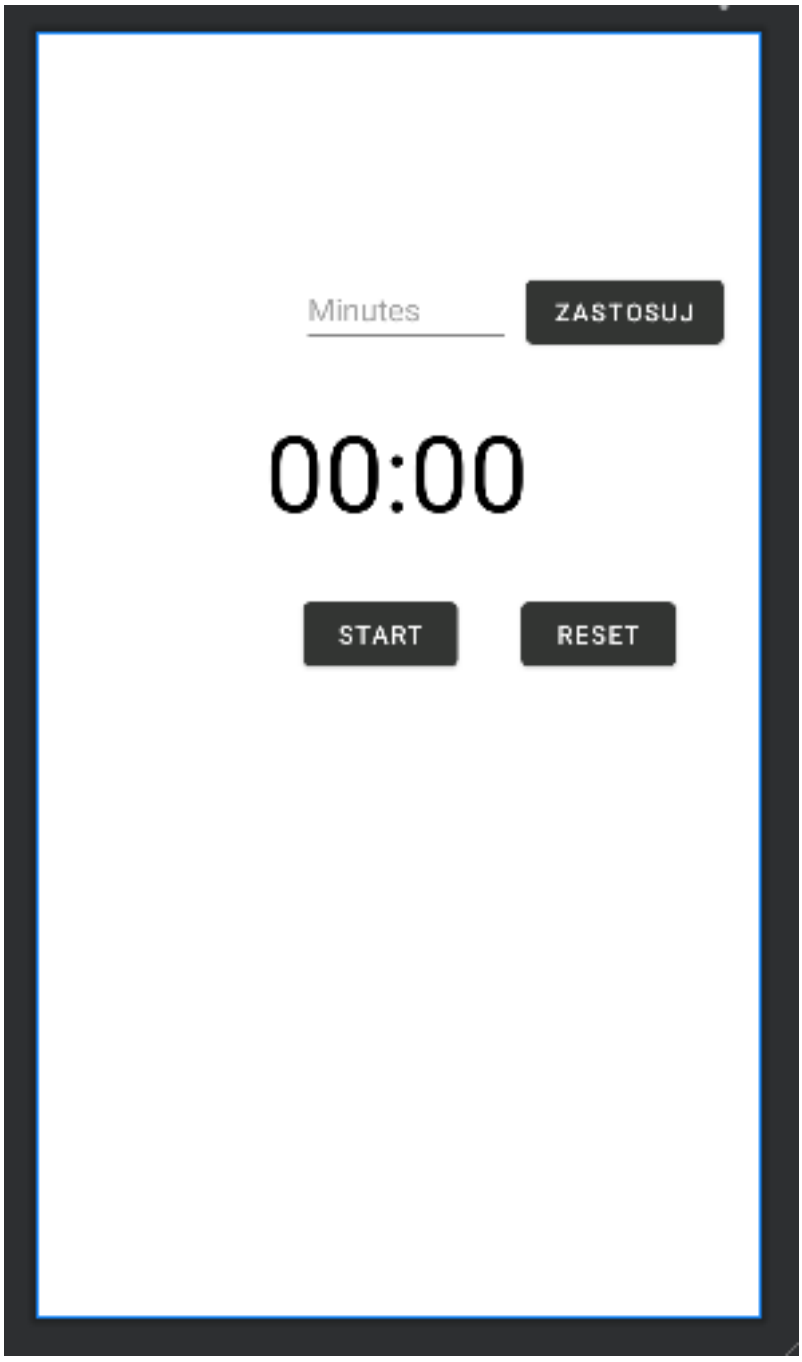
Spis treści

1. Ogólne określenie wymagań

Celem programu jest utworzenie prostej w obsłudze oraz uniwersalnej w każdej sytuacji aplikacji, dzięki której będzie możliwe bezpieczne poruszanie się w terenie. Aplikacja powinna zawierać wszystkie najważniejsze narzędzia do posługiwania się poza domem. Odbiorcą ma być każdy, niezależnie od wieku liczy się prostota w obsłudze, aby nikt nie miał problemu z użytkowaniem. Aplikacja ma być obsługiwana przez system Android. Program musi posiadać dostęp do mapy, która będzie umożliwiała orientowanie się w terenie górzystym, zabudowanym, leśnym.



Rys. 1.1



Rys. 1.2

2. Określenie wymagań szczegółowych

Naszym celem w budowaniu aplikacji jest zaprogramowanie tak,aby posiadała następujące funkcje:
-mape
-latarke
-kompas
-stoper który będzie odliczał czas do przerwy
-lokalizacja
Aplikacja będzie miała prosty interfejs. Brak internetu będzie skutkował utrudnieniami w korzystaniu z aplikacji. Zaczniemy budowe programu od zaprojektowania graficznego aplikacji i ułożenia danych narzędzi,przycisków. Po ustaleniu grafiki i przycisków funkcyjnych, przejdziemy do rozpoczęnia kodowania aplikacji i podłączać mape z lokalizacją.Po zaprogramowaniu aplikacji będziemy sprawdzali jej prawidłowe działanie i poprawiali błedy. Program będziemy rozwijali w przyszłości o kolejne funkcje.

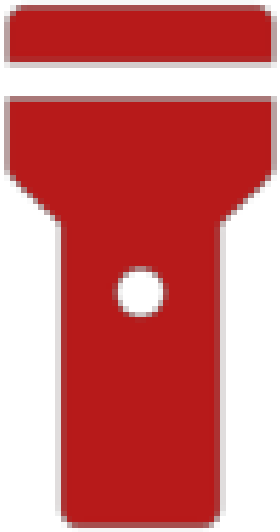
Przycisk latarki będącej w stanie off.(Rys.2.2)

Przycisk latarki będącej w stanie on.(Rys.2.3)

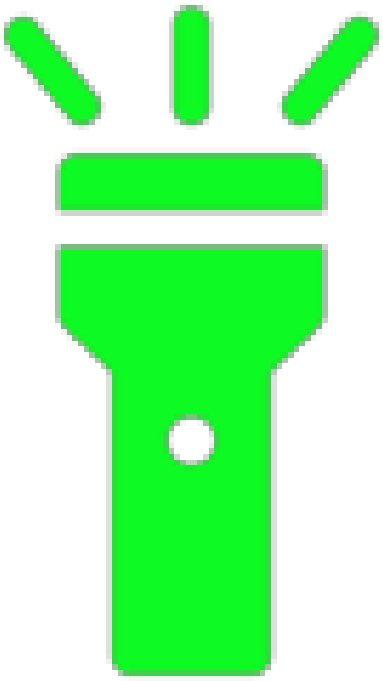
Wygląd kompasu.(Rys.2.4)

Wygląd przycisku timera.(Rys.2.5)

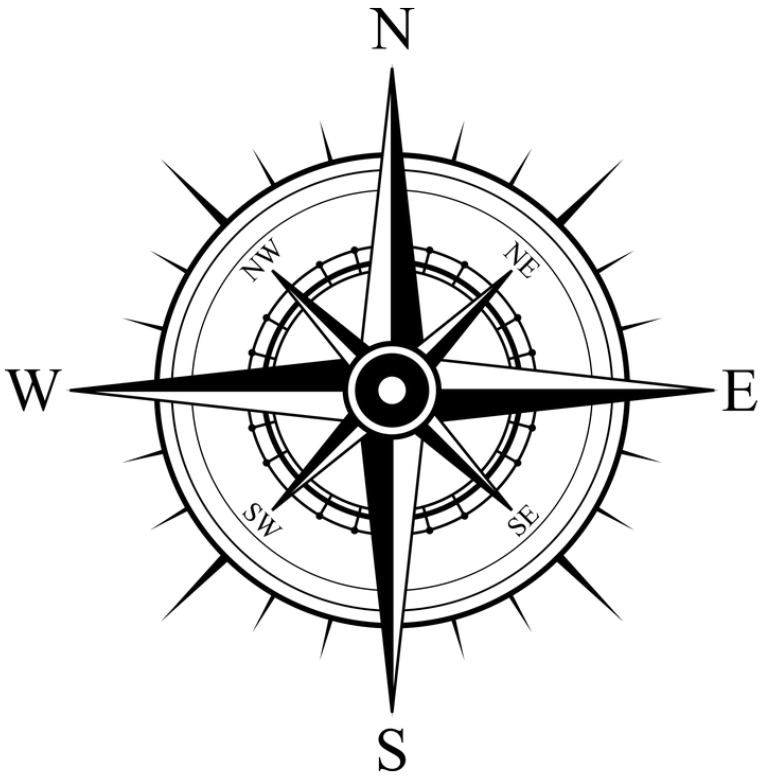
Rys. 2.1. Ekran głowny aplikacji.



Rys. 2.2. Przycisk latarki będącej w stanie off.



Rys. 2.3. Przycisk latarki będącej w stanie on.



Rys. 2.4. Wygląd kompasu.

Rys. 2.5

3. Projektowanie

Do projektu wykorzystaliśmy narzędzia takie jak GIT czyli rozproszony system kontroli wersji, oprogramowanie służące do śledzenia zmian głównie w kodzie źródłowym oraz pomocy programistom w łączeniu zmian dokonanych w plikach przez wiele osób w różnym czasie.

GitHub– hostingowy serwis internetowy przeznaczony do projektów programistycznych wykorzystujących system kontroli wersji Git. Stworzony został przy wykorzystaniu frameworka Ruby on Rails i języka Erlang. Serwis działa od kwietnia 2008 roku[1]. GitHub udostępnia darmowy hosting programów open source i prywatnych repozytoriów (część funkcji w ramach prywatnych repozytoriów jest płatna)(Rys.3.1).

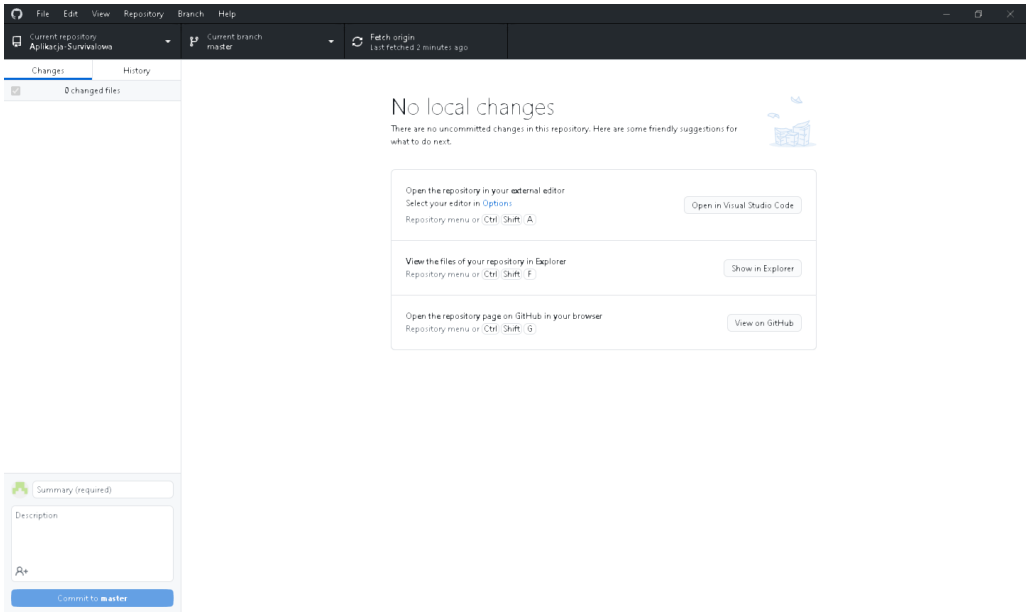
Aby efektowanie pracować w grupie, utworzyliśmy repozytorium Aplikacja-Survivalowa, gdzie umieściliśmy kod do naszej aplikacji wraz z dokumentacją, dzięki temu możemy w prosty sposób wprowadzać zmiany oraz ulepszać program.Do zapisywania oraz wysyłania zmian używamy programu GitHub Desktop (Rys.3.2).

Do wykonania dokumentacji oraz raportów wykorzystaliśmy TeXstudio to wieloplatformowy edytor LaTeX typu open source. Jego funkcje obejmują interaktywne sprawdzanie pisowni, składanie kodu i podświetlanie składni. Nie zapewnia samego LaTeX-a – użytkownik musi najpierw wybrać dystrybucję TeX-a i ją zainstalować.

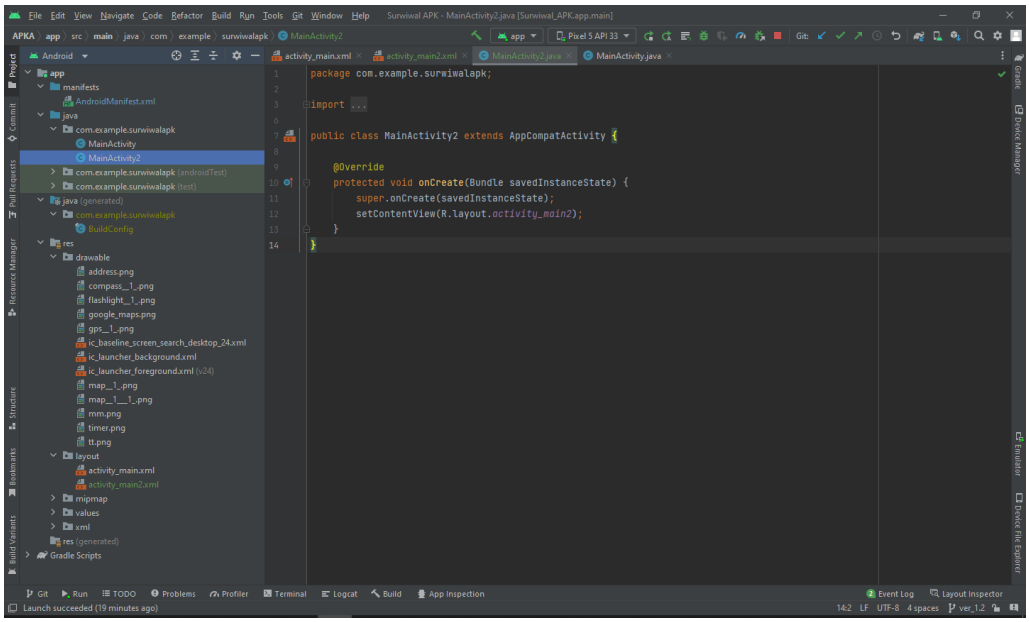
Pierwotnie nazywany TexMakerX, TeXstudio został uruchomiony jako rozwidlenie Texmakera, który próbował rozszerzyć go o dodatkowe funkcje, zachowując jednocześnie jego wygląd i styl

Środowiskiem programistycznym gdzie piszemy naszą aplikacje jest Android Studio (Rys.3.3) czyli środowisko programistyczne (IDE) stworzone przez Google na bazie IntelliJ, które kierowane jest do developerów aplikacji na Androida. Pozwala ono wygodnie projektować, tworzyć i debugować własne programy na najpopularniejszą obecnie platformę systemową dla urządzeń mobilnych.

Rys. 3.1. GitHub



Rys. 3.2. GitHub Setup



Rys. 3.3. Android Studio

4. Implementacja

Na listingu nr. 1 przedstawiony jest kod utworzenia przyciku latarki oraz pozycjonowanie na ekranie. Przykładowo w linijce 7 kodu marginBottom - oznacza zachować odległość od dołu widoku

```
1  <ImageButton
2    android:id="@+id/imageButton14"
3    android:layout_width="wrap_content"
4    android:layout_height="wrap_content"
5    android:layout_marginStart="203dp"
6    android:layout_marginEnd="16dp"
7    android:layout_marginBottom="260dp"
8    android:backgroundTint="#00FFFFFF"
9    app:layout_constraintBottom_toBottomOf="parent"
10   app:layout_constraintEnd_toEndOf="parent"
11   app:layout_constraintStart_toStartOf="@+id/imageButton12"
12   app:srcCompat="@drawable/flashlight__1_" />
13
```

Listing 1. Przykładowy kod Przycisku latarki

Na listingu nr. 2 przedstawiony jest kod przycisku wyłączonej latarki.

```
1  <ImageButton
2    android:id="@+id/imageButton"
3    android:layout_width="wrap_content"
4    android:layout_height="wrap_content"
5    android:layout_marginBottom="24dp"
```

```
  android:backgroundTint="#00FFFFFF"
  app:layout_constraintBottom_toTopOf="@+id/imageButton19"
  app:layout_constraintEnd_toEndOf="parent"
  app:layout_constraintHorizontal_bias="0.428"
  app:layout_constraintStart_toEndOf="@+id/imageButton12"
  app:srcCompat="@drawable/off" />
```

Listing 2. Przykładowy kod przycisku wyłączonej latarki

Na listingu nr. 3 przedstawiony jest kod definiowaniowania zmiennych.

```
private ImageButton toggleButton;
boolean hasCameraFlash = false;
boolean flashOn = false;
```

Listing 3. Przykładowy kod definiowania zmiennych

Na listingu nr. 4 przedstawiony jest kod pozwalający na wyłączenie latarki.(kod poniżej)

```
    public void onClick(View view) {
  if (hasCameraFlash){
    if (flashOn){
      flashOn = false;
      toggleButton.setImageResource(R.drawable.off);
      try {
        flashLightOff();
      } catch (CameraAccessException e) {
        e.printStackTrace();
      }
    }


    private void flashLightOn() throws CameraAccessException {
      CameraManager cameraManager = (CameraManager)
   getSystemService(Context.CAMERA_SERVICE);
    assert cameraManager != null;
    String cameraId = cameraManager.getCameraIdList()[0];
      cameraManager.setTorchMode(cameraId, true);
      Toast.makeText(MainActivity.this, "FlashLight is ON",
   Toast.LENGTH_SHORT).show();
}
```

Listing 4. Przykładowy kod ukazujący mechanizm wyłączenia latarki.

Na listingu nr.5 przedstawiony jest kod prezentujący mechanizm przechodzenia do nowej strony poprzez kliknięcie przycisku na stronie głownej.

```
    toggleButton2 = findViewById(R.id.imageButton2);
  toggleButton2.setOnClickListener(new View.OnClickListener() {
  @Override//nowa strona tajmer
  public void onClick(View view) {
    openTajmer();
  }
});
```

Listing 5. Przechodzenie do nowej strony.

5. Testowanie

6. Podręcznik użytkownika

Spis rysunków

Spis tabel

Spis listingów