<u>Zadanie 1</u> Projektowanie interfejsu użytkownika.

Twoim zadaniem jest stworzenie interfejsu użytkownika dla aplikacji mobilnej na platformę Android. Interfejs powinien zawierać:

1. **TextView:**
   - Identyfikator: textView
   - Szerokość: Dopasowana do rodzica
   - Wysokość: Dopasowanie do zawartości
   - Tekst: Zaszyty zasób tekstowy o nazwie Hello
   - Rozmiar tekstu: 30sp
   - Wyrównanie tekstu: Wycentrowane
   - Wypełnienie: 15dp
   - Tło: Kolor niebieski (@color/blue)
   - Kolor tekstu: Biały (@color/white)
   - Styl tekstu: Pogrubiony

2. **Button:**
   - Identyfikator: btn
   - Szerokość: Dopasowana do rodzica
   - Wysokość: Dopasowanie do zawartości
   - Położenie pod textView
   - Wycentrowany na ekranie
   - Margines górny: 20dp
   - Tło: Kolor niebieski (@color/blue) // zdefiniuj kolor
   - Tekst: Zaszyty zasób tekstowy o nazwie kliknij_mnie
   - Kolor tekstu: Biały (@color/white)
   - Obsługa zdarzenia onClick: Metoda sayHello

3. **TextView:**
   - Identyfikator: txt
   - Szerokość: Dopasowana do rodzica
   - Wysokość: Dopasowanie do zawartości

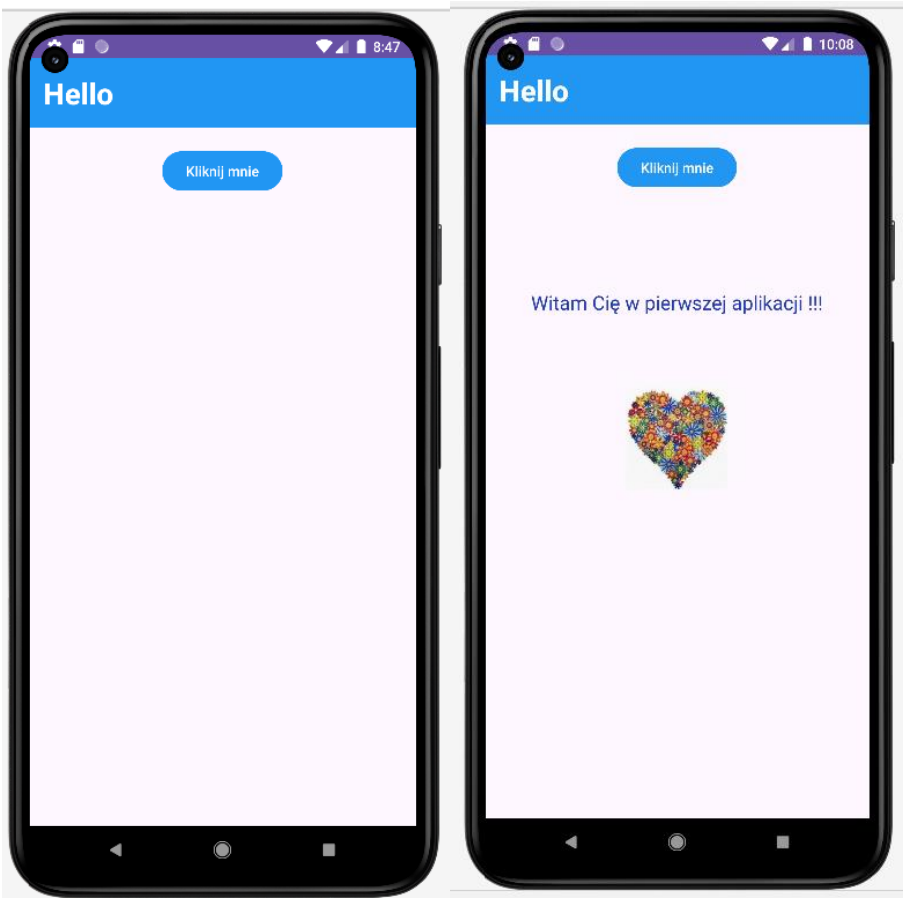
- Położenie pod btn
- Wycentrowany na ekranie
- Margines górny: 100dp
- Rozmiar tekstu: 20sp
- Kolor tekstu: Ciemnoniebieski (@color/darkBlue)
- Początkowo pusty tekst

Upewnij się, że zachowujesz spójność estetyczną i przestrzegasz wytycznych dotyczących położenia i stylu elementów interfejsu. Po zakończeniu zadania, dodaj odpowiednią logikę w kodzie źródłowym, aby obsługiwać zdarzenie kliknięcia przycisku (sayHello).

Zadanie 2.

Napisz prosta aplikacje moblina, która obsługuje kliknięcie przycisku wyświetlając powitanie użytkownika, wyświetlając powitanie na TextView.

Poniżej znajduje się kod XML definiujący widok oparty na RelativeLayout, zawierający elementy TextView, Button i kolejne TextView.



Wymagania :

1. **Wielkość tekstu i styl:**
    - Dopasuj wielkość tekstu dla elementu TextView **textView** do **30sp.**
    - Dodaj pogrubienie do tekstu w tym elemencie.
2. **Tło i kolor tekstu:**
    - Zmień kolor tła TextView **textView** na odcień niebieskiego (**@color/blue**).
    - Ustaw kolor tekstu w tym elemencie na biały (**@color/white**).
3. **Przycisk:**
    - Zmodyfikuj tło przycisku (**btn**) na odcień niebieskiego (**@color/blue**).
    - Skonfiguruj funkcję obsługi kliknięcia przycisku (**sayHello**).
4. **Tekst po kliknięciu:**
    - Ustaw element TextView o identyfikatorze **txt** tak, aby wyświetlał tekst poniżej przycisku (**btn**).
    - Zmień kolor tekstu w elemencie **txt** na ciemny odcień niebieskiego (**@color/darkBlue**).
5. **Marginesy i rozmieszczenie:**
    - Dostosuj marginesy i rozmieszczenie elementów w RelativeLayout, aby uzyskać atrakcyjny układ.

Twoim zadaniem jest wprowadzenie pewnych modyfikacji w celu poprawy funkcjonalności i dodania nowych elementów. Pamiętaj o uwzględnieniu następujących punktów:

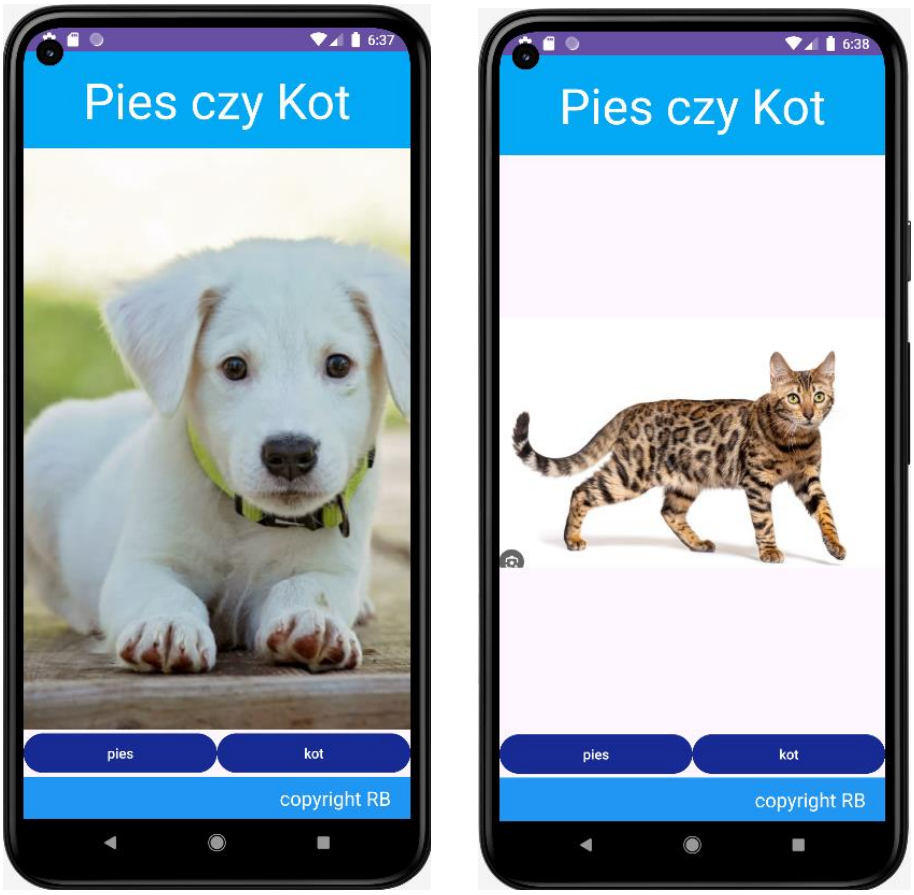
1. **Rozbudowa tekstu powitalnego:**
    - Zmodyfikuj funkcję **sayHello(View view)** tak, aby oprócz tekstu "Witam Cię w pierwszej aplikacji !!!" dodawała dodatkowy tekst informacyjny.
2. **Zmiana tła aplikacji:**
    - Zmień tło aplikacji na inny kolor.
3. **Dodanie obrazka powitalnego:**
    - Dodaj obrazek do tekstu powitalnego lub innego elementu.

```
mtxt.setText("Witam Cię w pierwszej aplikacji !!!");
obraz.setVisibility(View.VISIBLE);
tlo.setBackgroundColor(Color.YELLOW);

lub

tlo.setBackgroundColor(getResources().getColor(R.color.yellow1));
(jeśli kolor jest dodany do zasobów aplikacji i chcemy z niego
skorzystać)
```

Zadanie 3.

Stworzyć interaktywną aplikację mobilną w języku Java, która :

1. wyświetla pytanie "Pies czy Kot" w dużym napisie na górze ekranu.
2. Poniżej znajdują się dwa obrazy - psa (**pies**) i kota (**kot**).
3. Na początku widoczny jest obraz kota, ale po naciśnięciu przycisku "pies" ma się pojawić obraz psa, a obraz kota ma zniknąć.
4. Dodatkowo, na dole ekranu znajdują się dwa przyciski "pies" i "kot", które umożliwiają użytkownikowi przełączanie się między obrazami.
5. Na samym dole ekranu wyświetlony jest napis "copyright RB" w kolorze niebieskim z wyrównaniem do prawej strony.
6. Zadanie ma być zrealizowane w pliku XML opisującym layout ekranu.



ćwiczenie :

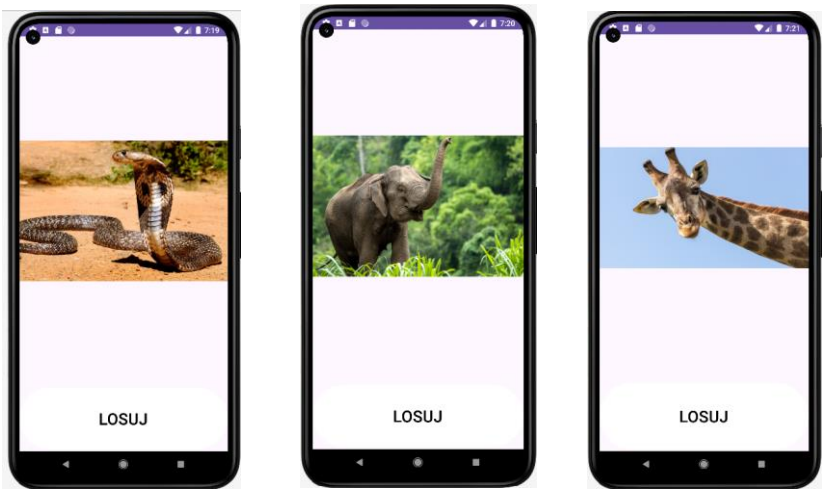
1. Uzupełnij projekt o element wyświetlania „kot” w nagłówku zamiast „pies czy kot” po kliknięciu buton „kot”
2. Uzupełnij projekt o element wyświetlania „pies” w nagłówku zamiast „pies czy kot” po kliknięciu buton „pies”

<u>Zadanie 4.</u> Losowy obraz.

Napisz aplikacje która na kliknieciе przycisku losuje obraz z 4 obrazów z zasobów aplikacji.

Wykorzystaj tablice obrazów i metodę Random.

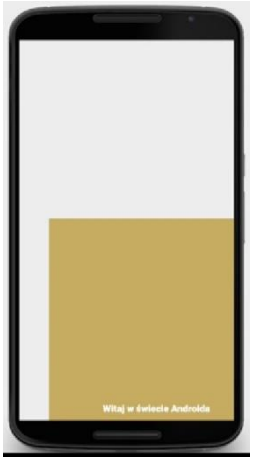
```
int[] images={R.drawable.kobra,R.drawable.slon,R.drawable.zyrafa,R.drawable.surokatka};
Random random = new Random();
int ind = random.nextInt(4);
imageView.setImageResource(images[ind]);
```



Zadanie dodatkowe: Dodatkowo przy kliknięciu buttona losuje się kolor tła tego komponentu.

```
private int randomButton()
{
    Random random = new Random();
    int r=random.nextInt(256);
    int g=random.nextInt(256);
    int b=random.nextInt(256);
    return Color.rgb(r,g,b);
}
```

<u>Zadanie 5.</u> Na podstawie widoku zaprojektuj Interfejs aplikacji Android.



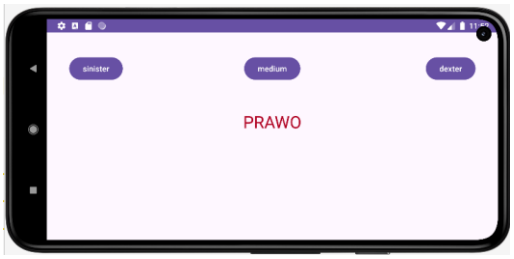
<u>Zadanie 6</u>. Tłumacz łaciny:

Stwórz aplikację w Android Studio, która będzie tłumaczyć słowa łacińskie na język polski. Poniżej znajduje się lista słów łacińskich oraz ich odpowiedników w języku polskim:

- sinister: lewo
- dexter: prawo
- medium: środek

Aplikacja powinna korzystać z interfejsu graficznego (GUI) i obejmować trzy przyciski, z których każdy będzie przypisany do jednego słowa łacińskiego. Po kliknięciu na dowolny przycisk, aplikacja powinna wyświetlić tłumaczenie odpowiedniego słowa w kontrolce typu TextView.

Możesz dostosować GIU zgodnie z własnymi preferencjami dotyczącymi wyglądu i funkcjonalności interfejsu użytkownika.



<u>Zadanie 7</u>. Przycisk Zmieniający Kolor:

Stwórz aplikację w Android Studio, która zawiera jeden przycisk. Po każdym kliknięciu na przycisk, jego kolor tła oraz kolor tekstu powinny się odwracać - jeśli przycisk miał białe tło z czarnym tekstem, to po kliknięciu powinien mieć czarne tło z białym tekstem.

Oto opis poszczególnych elementów:

RelativeLayout - Definiuje kontener dla elementów interfejsu użytkownika, w którym relacje między nimi są opisane poprzez ich położenie względem siebie.

Button (Przycisk):

- Nadaj przyciskowi unikalny identyfikator, który może być używany do odwoływania się do niego w kodzie Java.
- Określ szerokość i wysokość przycisku, aby dopasować się do zawartości.
- Ustal, że przycisk powinien być wyśrodkowany w obrębie rodzica (RelativeLayout).
- Ustaw kolor tła przycisku na czarny.
- Dodaj marginesy wew. o szerokości 60 dp do przycisku.
- Ustaw tekst wyświetlany na przycisku.
- Ustaw kolor tekstu na biały.
- Ustaw rozmiar tekstu na 40 sp
- Jeśli używasz tej struktury w projekcie, upewnij się, że odpowiednie kolory są zdefiniowane w pliku res/values/colors.xml.