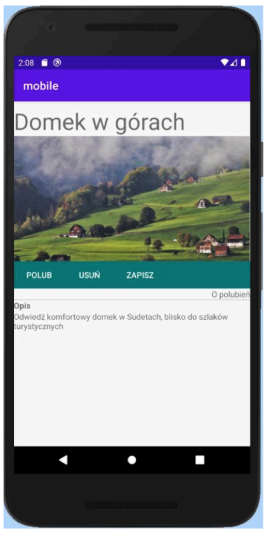
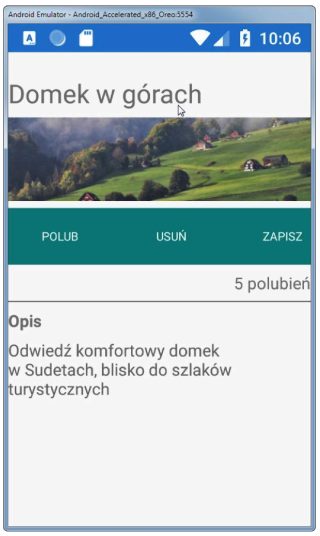
INF.04 czerwiec 2022



Obraz 1a.



Obraz 1b.

Elementy aplikacji:

- Tytuł o treści: „Domek w górach”.
- Obraz o nazwie *obraz.jpg* wypakowany z archiwum.
- Trzy przyciski o treści: „POLUB”, „USUŃ”, „ZAPISZ” umiejscowione obok siebie.
- Napis o treści „0 polubień”.
- Linia horyzontalna.
- Napis o treści „Opis”.
- Napis o treści „Odwiedź komfortowy domek w Sudetach, blisko do szlaków turystycznych”.

Założenia aplikacji:

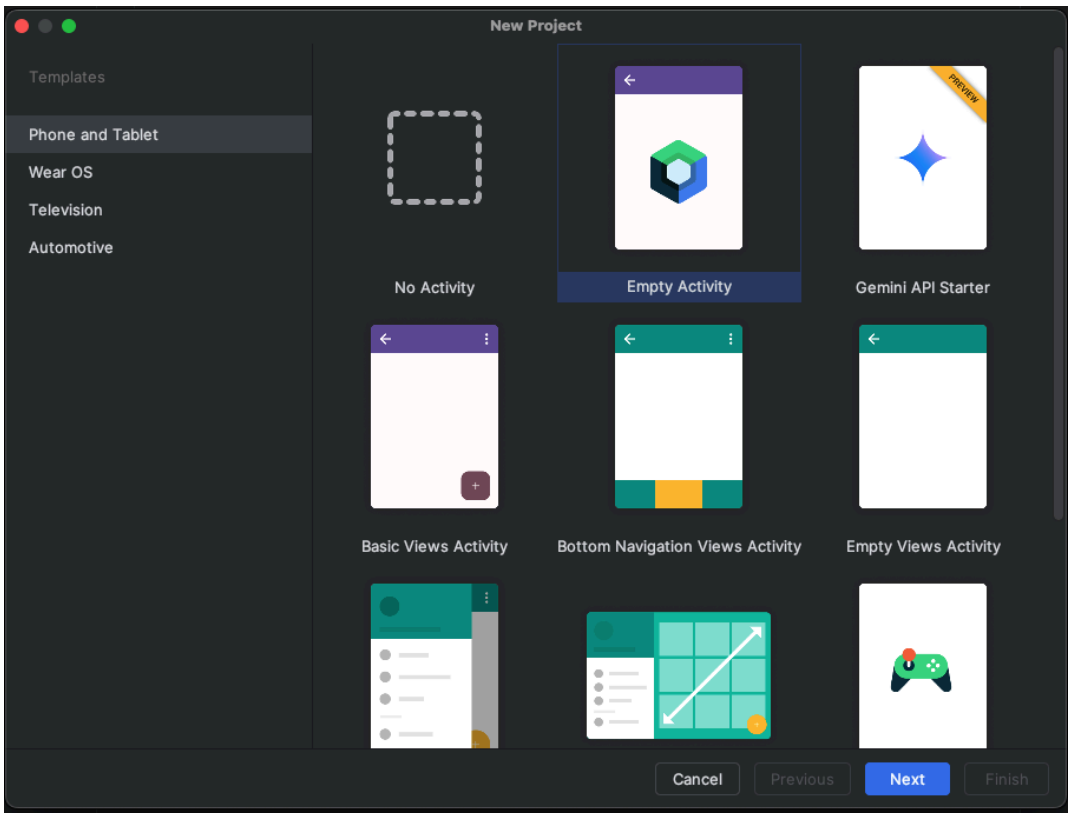
- Interfejs użytkownika zapisany za pomocą języka znaczników wspieranego w danym środowisku (np. XAML, XML).
- Zastosowany typ rozkładu liniowy wertykalny (Linear / Stack lub inny o tej idei) z zagłębionym rozkładem liniowym horyzontalnym dla przycisków.
- Margines wewnętrzny górny dla całej strony lub rozkładu wertykalnego: 20 px (lub dp)
- Kolor tła przycisków i rozkładu, w którym się znajdują: Teal (#008080), zgodnie z Obrazem 1a.

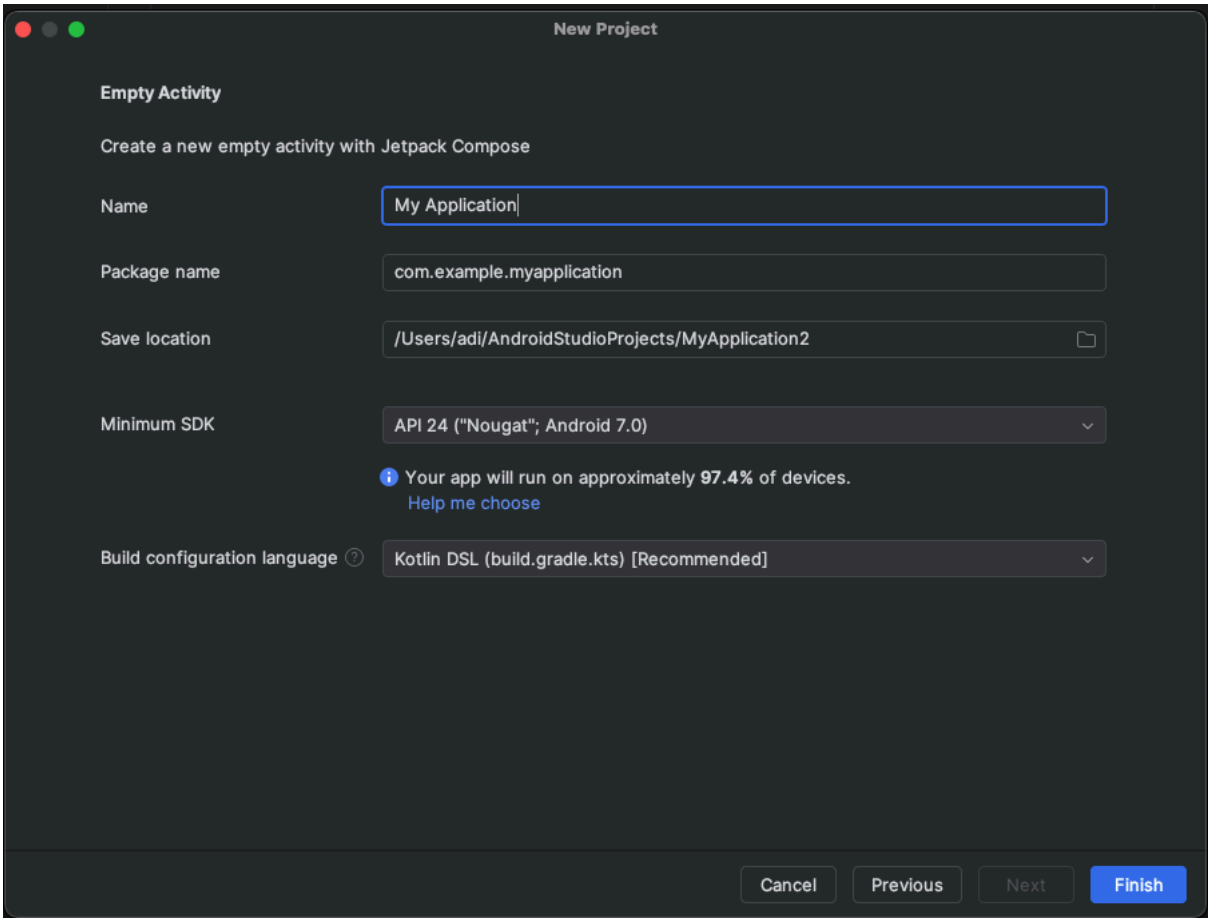
- Kolory czcionki: biały dla przycisków oraz Gray (#808080) dla napisu „Odwiedź...”, zgodnie z Obrazem 1a.
- Czcionka tytułu ma rozmiar największy spośród użytych w aplikacji.
- Czcionka napisu „Opis” jest pogrubiona.
- Napis o liczbie polubień jest wyrównany do prawej.
- Obraz wypełnia całą szerokość strony (zależnie od zastosowanego aspektu może być automatycznie obcięty przez emulator – zobacz obraz 1b).
- Linia horyzontalna jest koloru Gray (#808080), dopuszcza się również prostokąt o wysokości 1.
- Aplikacja powinna być zapisana czytelnie, z zachowaniem zasad czystego formatowania kodu, należy stosować znaczące nazwy zmiennych i funkcji.

Działanie aplikacji:

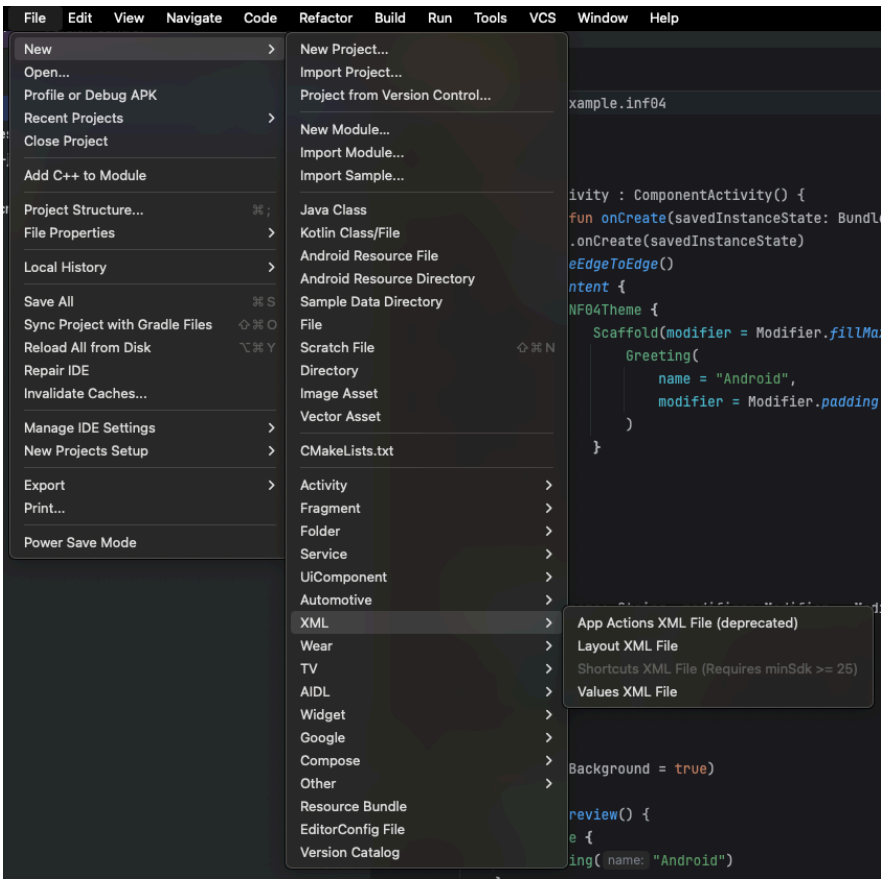
- Aplikacja implementuje licznik polubień, który w stanie początkowym aplikacji jest równy 0, następnie jego stan jest:
  - inkrementowany po wciśnięciu przycisku „POLUB”,
  - dekrementowany po wciśnięciu przycisku „USUŃ”. Licznik nie może być niższy niż 0.
- Stan licznika jest wyświetlany pod przyciskami, w formie napisu „<x> polubień”, gdzie <x> oznacza aktualną wartość licznika.

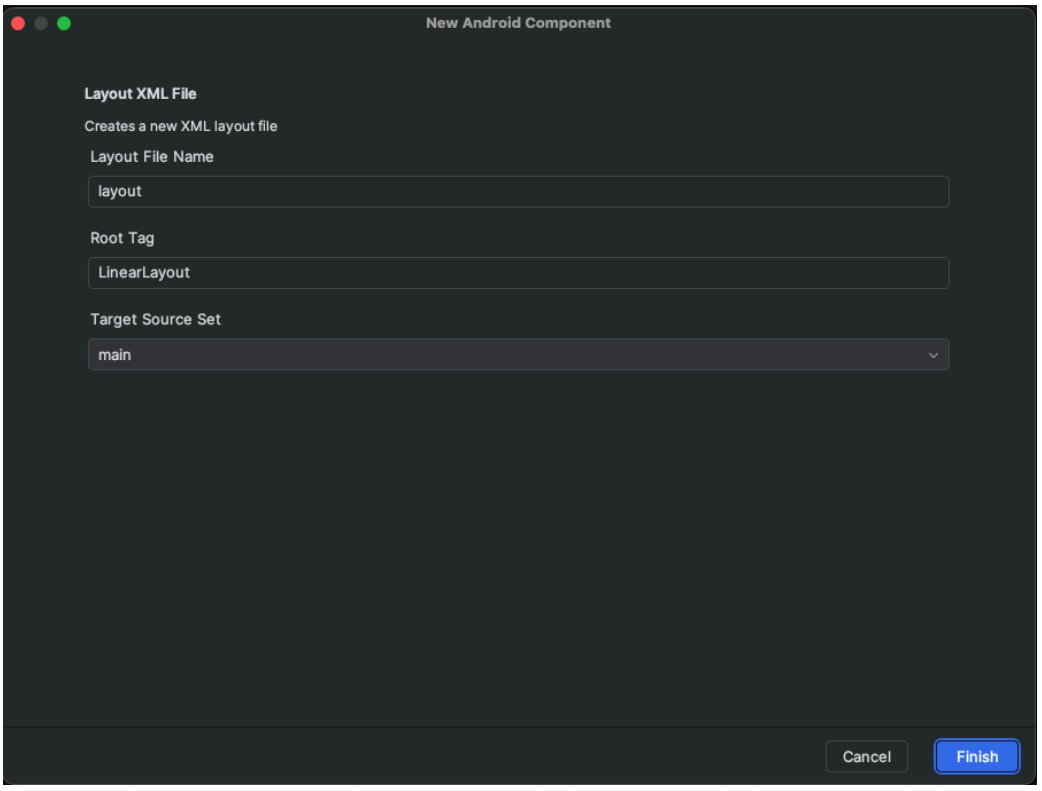
1. Zaczniemy od utworzenia nowego projektu



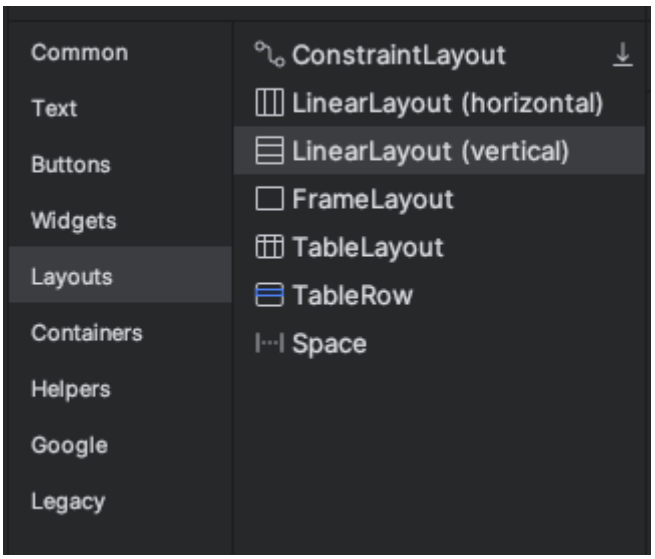


2. Następnie utworzymy plik .xml, w którym będziemy definiować wygląda naszej aplikacji

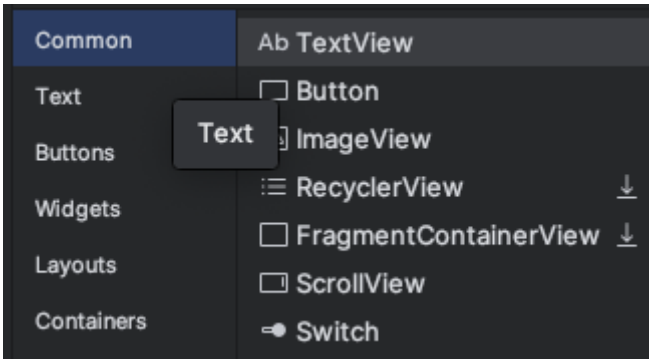


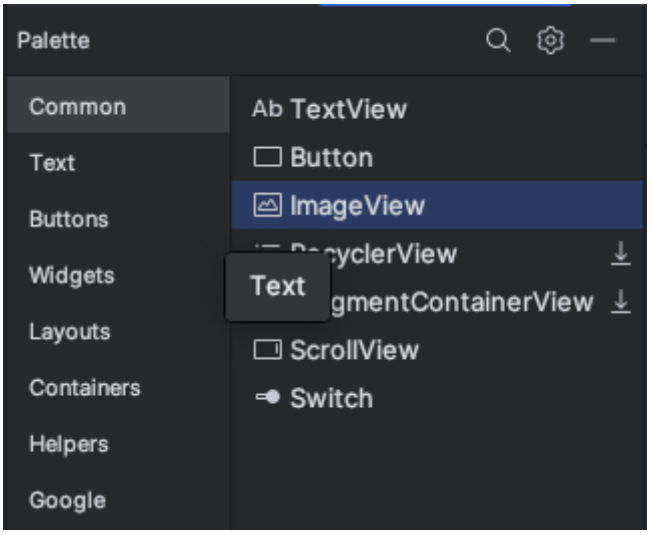


3. Do naszego layoutu dodamy element “LinearLayout (vertical)”, który pozwoli nam układać elementy w kolejnych wierszach

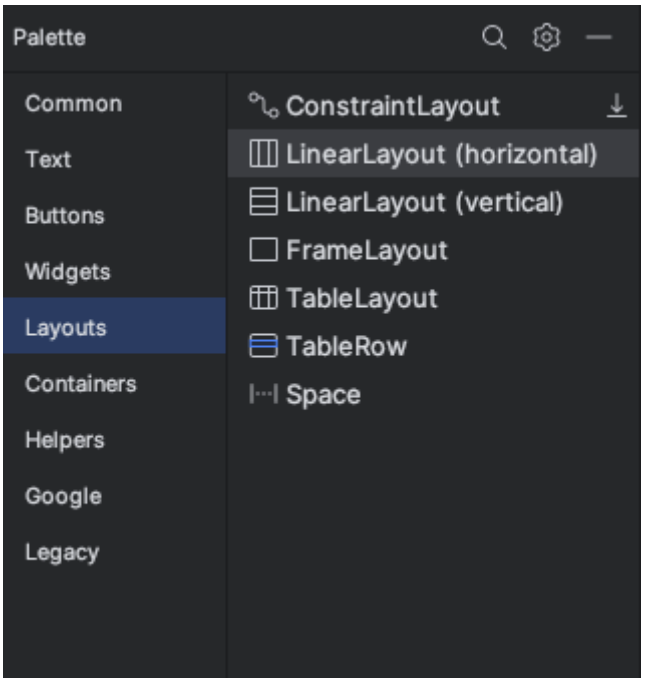


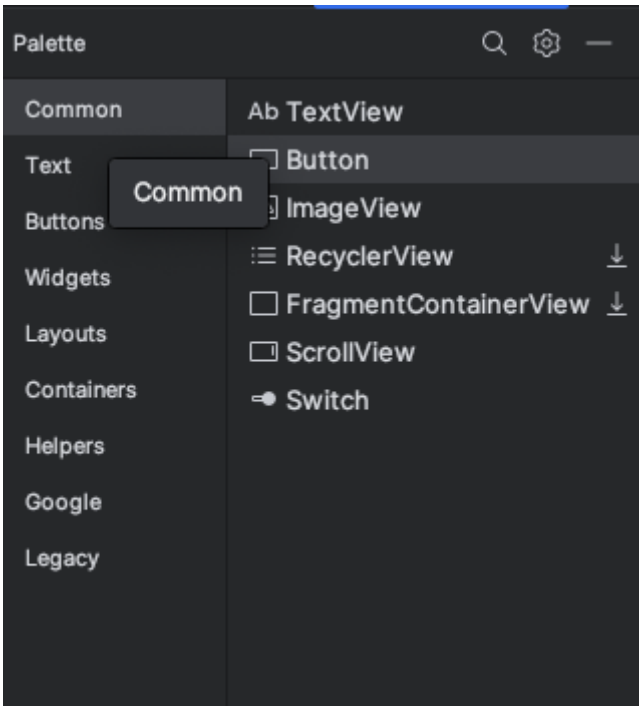
4. Następnie do naszego LinearLayout dodamy elementy wymagane w założeniach aplikacji - text i obrazek



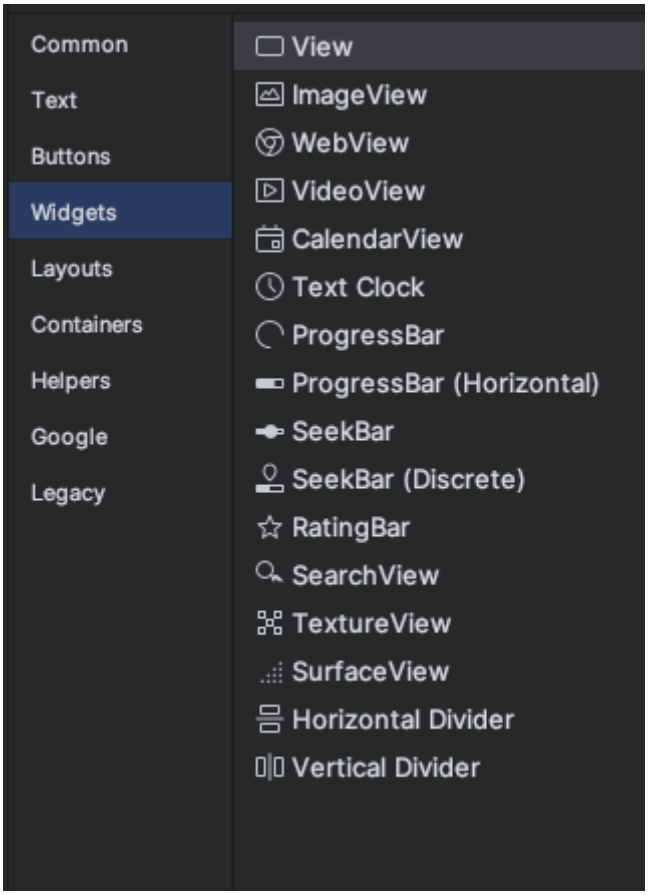


5. Pod obrazkiem dodamy LinearLayout (horizontal) i umieścimy w nim guziki - pozwoli to nam na równe ułożenie guzików

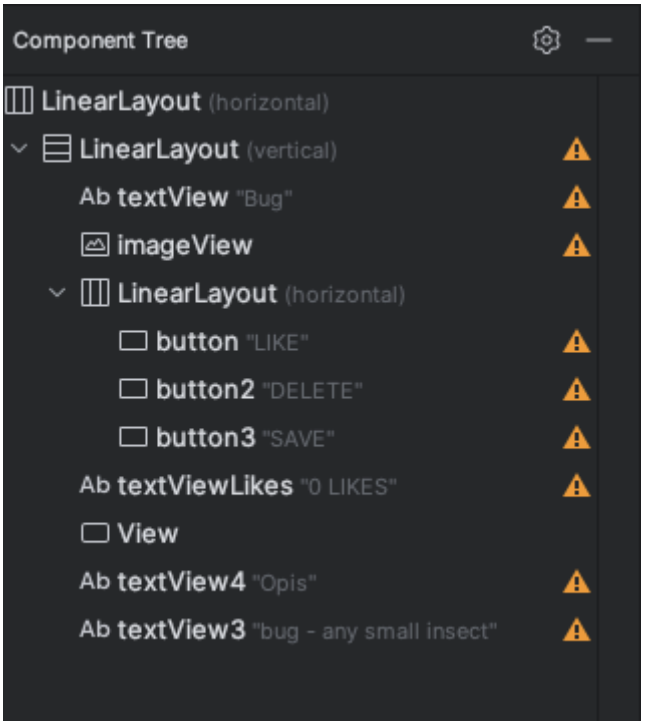




6. Następnie dodajemy nasz licznik polubień
7. Kolejnym krokiem będzie dodanie poziomej kreski, a pod nią dwóch pól tekstowych



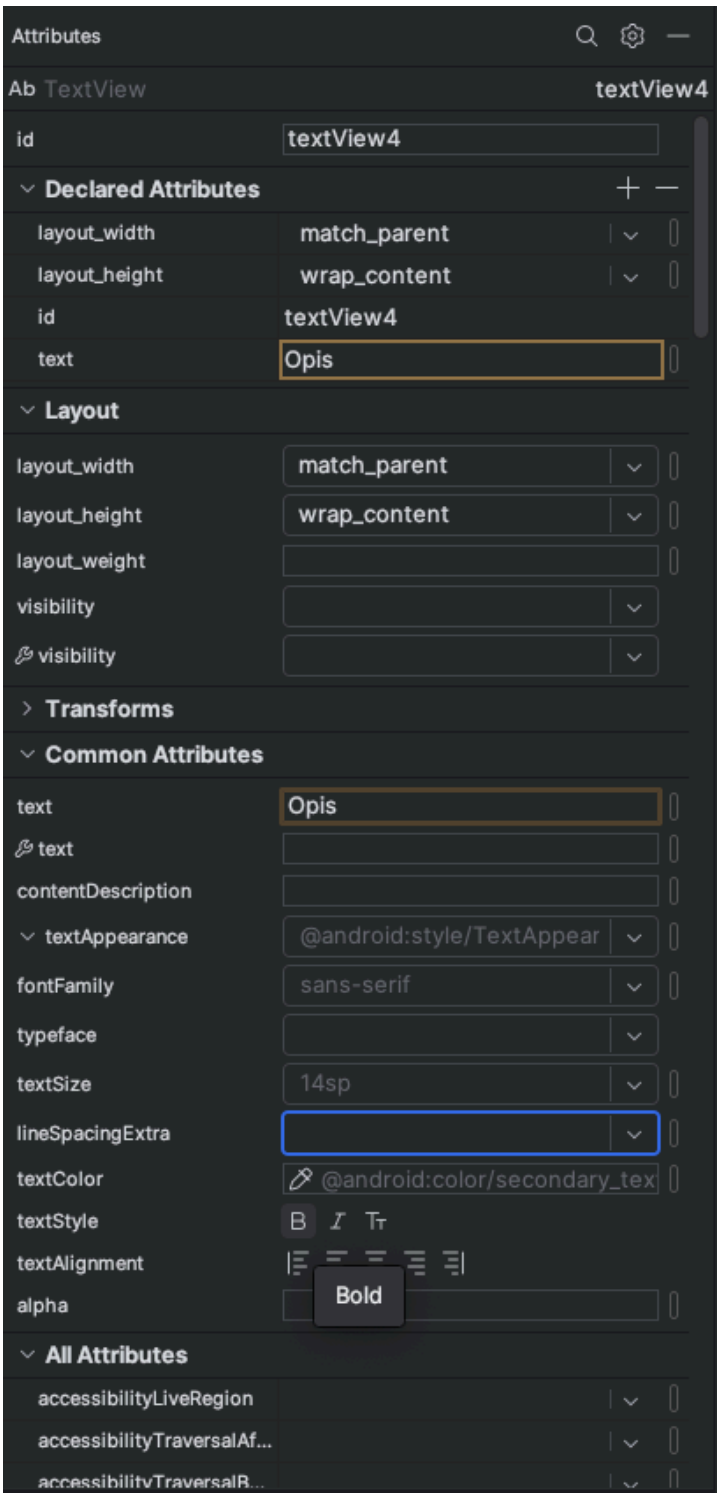
8. Oto struktura naszej aplikacji:



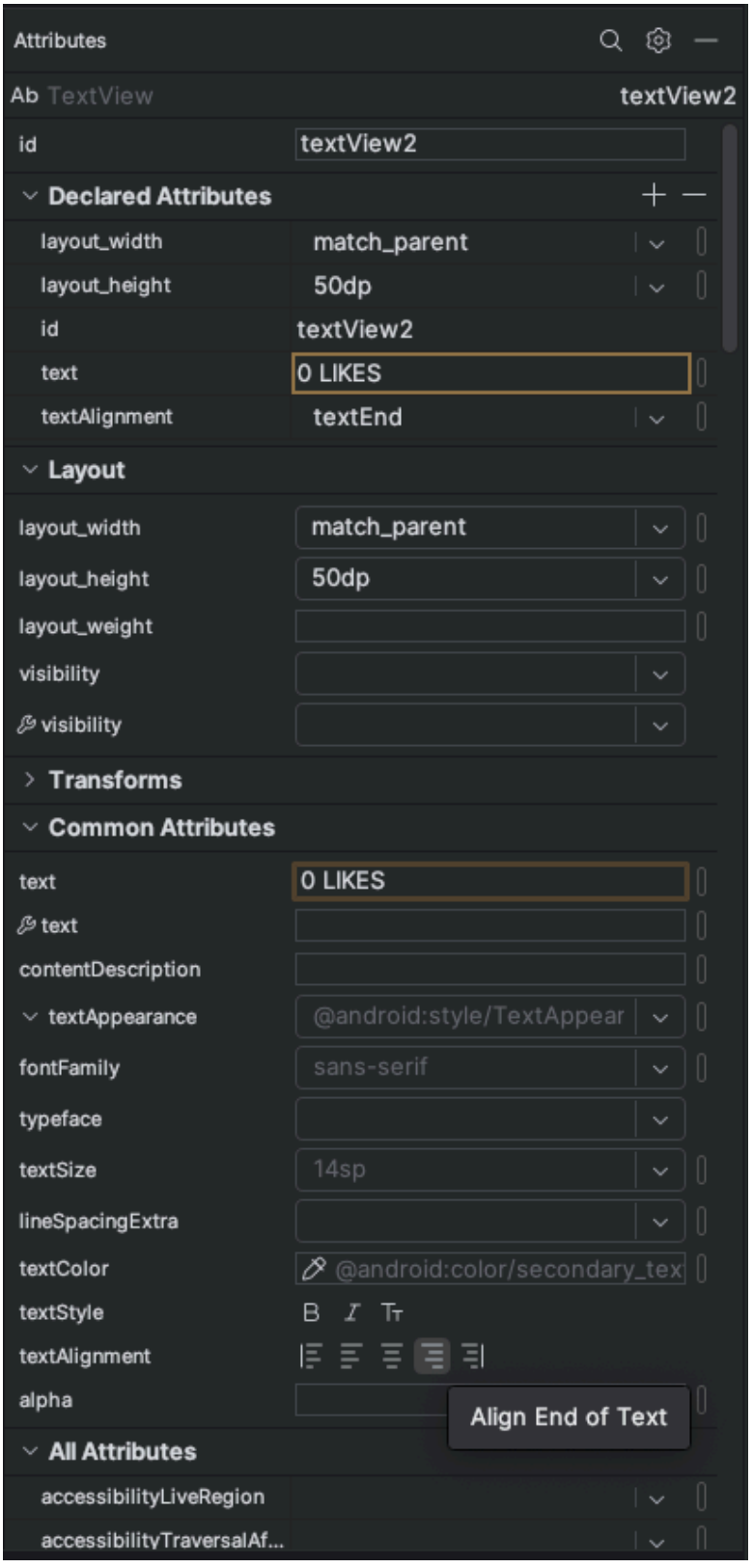
9. Przechodzimy do kwestii estetycznych - ustawienie górnego marginesu dla całej aplikacji

```
android:layout_marginTop="20dp">
```

10. Pogrubienie tekstu “Opis”

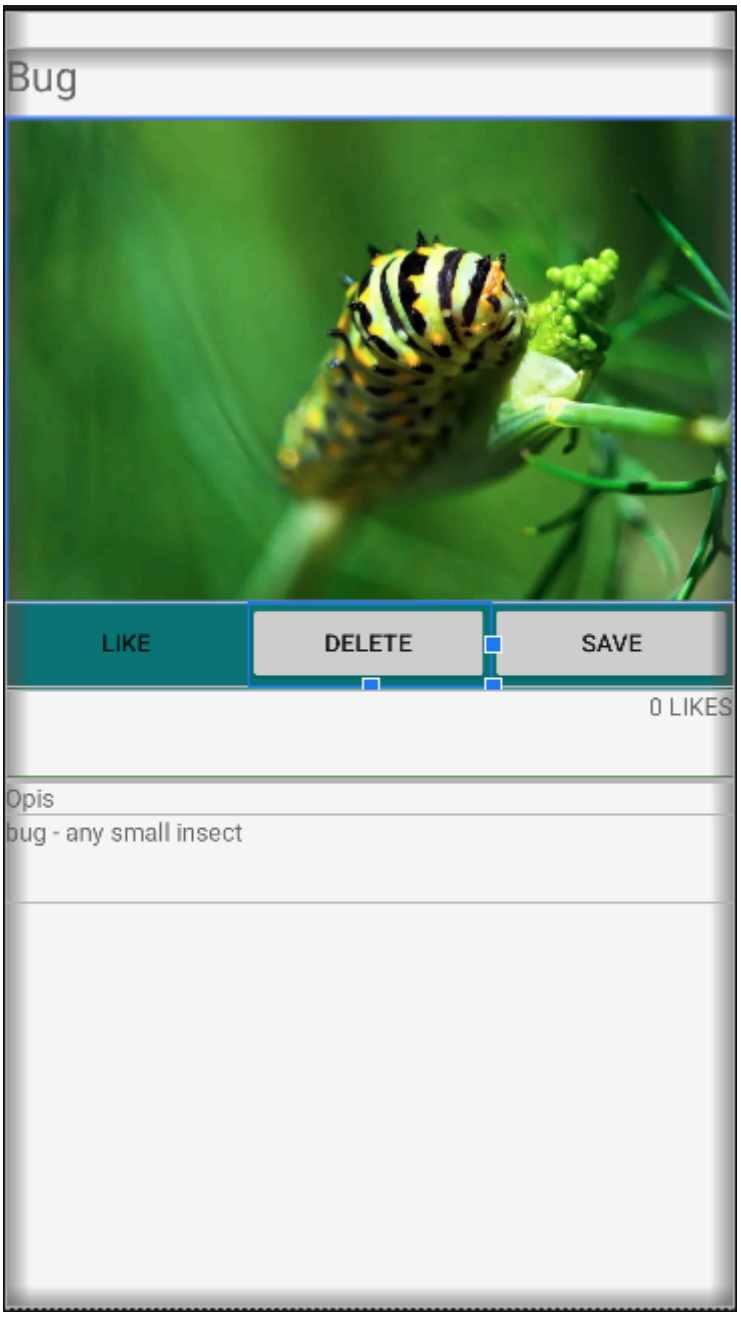

Attributes
Ab TextView
textView4
id
textView4
Declared Attributes
layout_width
match_parent
layout_height
wrap_content
id
textView4
text
Opis
Layout
layout_width
match_parent
layout_height
wrap_content
layout_weight
visibility
visibility
Transforms
Common Attributes
text
Opis
text
contentDescription
textAppearance
@android:style/TextAppear
fontFamily
sans-serif
typeface
textSize
14sp
lineSpacingExtra
textColor
@android:color/secondary_tex
textStyle
textAlignment
alpha
Bold
All Attributes
accessibilityLiveRegion
accessibilityTraversalAf...

11. Wyrównanie do lewej “licznika Like’ów”

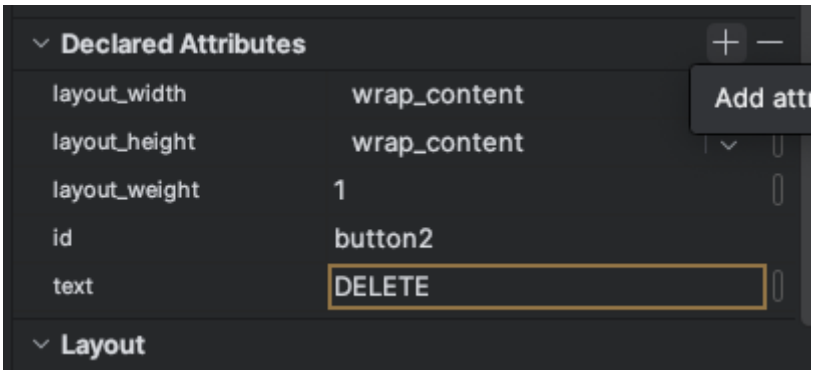


12. Ustawienie szerokości obrazka na cały ekran

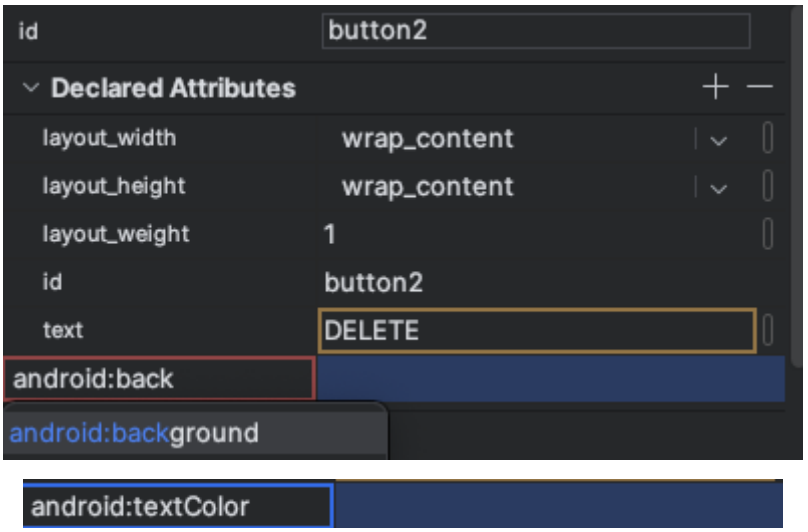
13. Teraz przejdziemy do zmiany koloru tła guzika (działa też do innych obiektów)



14. W sekcji “Attributes” rozwijamy listę “Declared Attributes” i klikamy “+”



15. Wyszukujemy właściwości “android:background” - dla tła i “android:textColor” - dla koloru czcionki



16. Wygląd naszej aplikacji jest gotowy - przejdźmy do funkcjonalności. Tworzymy zmienną pomocniczą - licznik polubień

```
private var likeCounter = 0
```

17. Funkcja zliczająca kliknięcia guziora “LIKE”

```
// Obsługa przycisku "LIKE"
fun onLikeClick(view: android.view.View) {
    likeCounter++
    updateLikesTextView()
}
```

18. Funkcja zliczająca kliknięcia guziora “DELETE”

```
// Obsługa przycisku "DELETE"
fun onDeleteClick(view: android.view.View) {
    if (likeCounter > 0) {
        likeCounter--
    }
    updateLikesTextView()
}
```

19. Funkcja odpowiadająca za aktualizację tekstu w zależności od wartości naszej zmiennej pomocniczej

```
// Aktualizacja tekstu z licznikiem polubień
private fun updateLikesTextView() {
    val likesTextView = findViewById<TextView>(R.id.textViewLikes)
    likesTextView.text = "$likeCounter LIKES"
}
```

20. Do działania naszej aplikacji brakuje tylko jednej rzeczy - wywołania odpowiednich funkcji na wciśnięcie poszczególnych guziorów. W pliku .xml dodajemy takie dwie linijki (w sekcjach odpowiadających konkretnym guziorom)

```
android:onClick="onLikeClick" />
```

```
android:onClick="onDeleteClick" />
```

Zadanie dla chętnych (dodatkowe na 6):

1. Na podstawie powyższej instrukcji zamień tekst na guziorze “SAVE” na “COMMENT”
2. Po kliknięciu guziora powinno wyświetlić się nowe okienko gdzie możemy wybrać ilość gwiazdek i wpisać komentarz. Ponadto w tym okienku powinny znajdować się dwa guziory: CANCEL - zamyka okienko, ADD - zapisuje ocenę i komentarz i zapisuje okienko. Po wciśnięciu guzika ADD pod opisem powinna wyświetlić się wybrana ilość gwiazdek i komentarz.