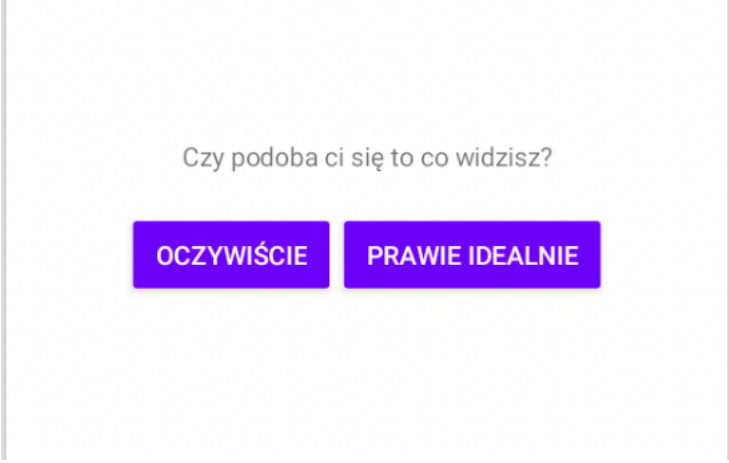
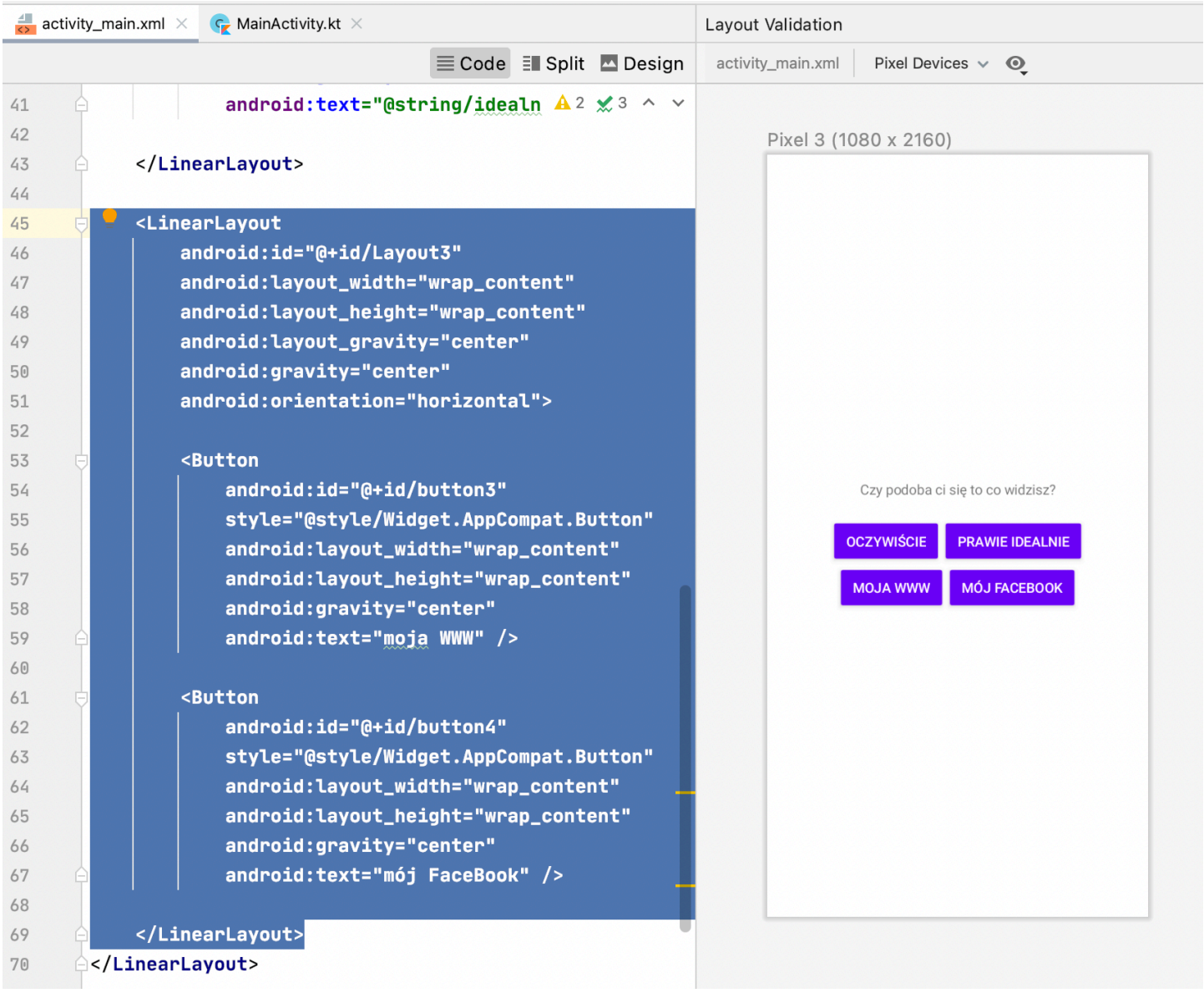
Intencje w Android

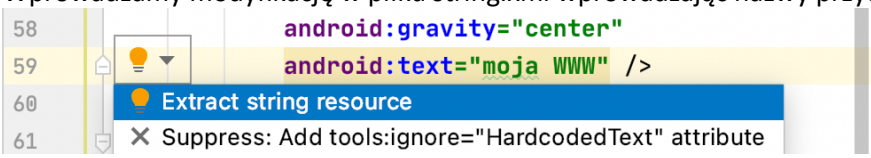
Podstawą tej instrukcji jest projekt „Hello World” wykonany poprzednio.



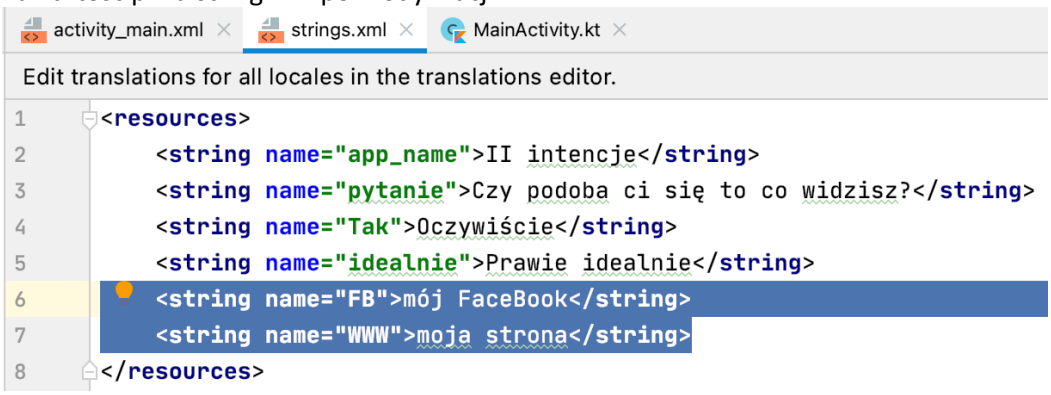
Na końcu pliku activity_main.xml dodamy kolejny Layout w którym umieścimy dodatkowe przyciski. Można to zrobić kopiując już istniejący wewnętrzny LinerarLayory zmieniając odpowiednio dane.



Wprowadzamy modyfikację w pliku string.xml wprowadzając nazwy przycisków



Zawartość pliku string.xml po modyfikacji:



Przechodzimy do pliku MainActivity.kt i tworzymy obsługę przycisków uruchamiające stronę WWW oraz FB:

```
        Toast.makeText( context: this,  text: "Przycisk 2 został kliknięty", Toast.LENGTH_SHORT).show()
    }
    val btn3_click = findViewById<Button>(R.id.button3)
    val btn4_click = findViewById<Button>(R.id.button4)
    btn3_click.setOnClickListener(){  it: View!
        var message3 = Toast.makeText( context: this, text: "Dziękuję za odwiedzenie mojej strony",Toast.LENGTH_SHORT)
        message3.show()
    }
    btn4_click.setOnClickListener(){  it: View!
        var message4 = Toast.makeText( context: this, text: "Dołącz do moich znajomych",Toast.LENGTH_SHORT)
        message4.show()
    }
```

Teraz dodajemy dla każdego przycisku linki do stron jako zmienne typu uri. Jednocześnie uruchamiamy obiekt *intence* jako intencję niejawną. (intencja niejawna jest odwołaniem do intencji funkcjonującej w systemie, czy my jej nie tworzymy).

```
    btn3_click.setOnClickListener(){  it: View!
        var message3 = Toast.makeText( context: this, text: "Dziękuję za odwiedzenie mojej strony",Toast.LENGTH_SHORT)
        message3.show()
        var adres = "https://bab.la/" //przykładowy słownik ang.
        var channelRevolShen =Intent(ACTION_VIEW, Uri.parse(adres))
    }
    btn4_click.setOnClickListener(){  it: View!
        var message4 = Toast.makeText( context: this, text: "Dołącz do moich znajomych",Toast.LENGTH_SHORT)
        message4.show()
        var adres = "https://pl-pl.facebook.com/" //strona główna FB
        var channelRevolShen =Intent(ACTION_VIEW, Uri.parse(adres))
    }
```

Dodatkowo nasze „import …” powinny zostać uzupełnione do postaci:

```
import android.annotation.SuppressLint
import android.content.Intent
import android.content.Intent.ACTION_VIEW
import android.net.Uri
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.Toast
import android.widget.Button
import android.widget.TextView
```

Teraz musimy uruchomić naszą aktywność. Dopisujemy przy **każdym** przycisku linię startową.

```
51        var channelRevolShen =Intent(ACTION_VIEW, Uri.parse(adres))
52        startActivity(channelRevolShen)
```

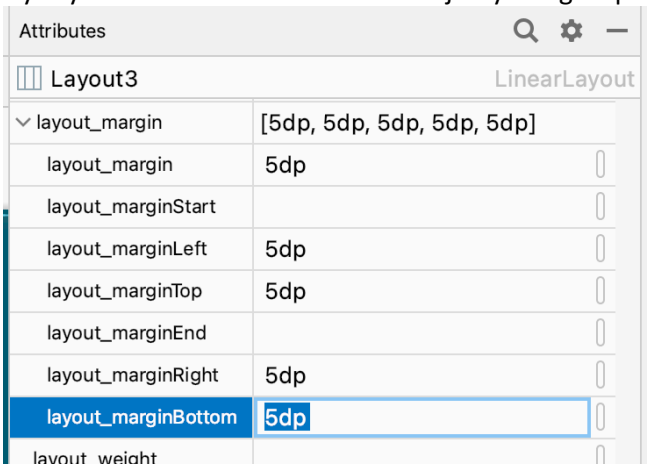
Przyciski powinny już działać

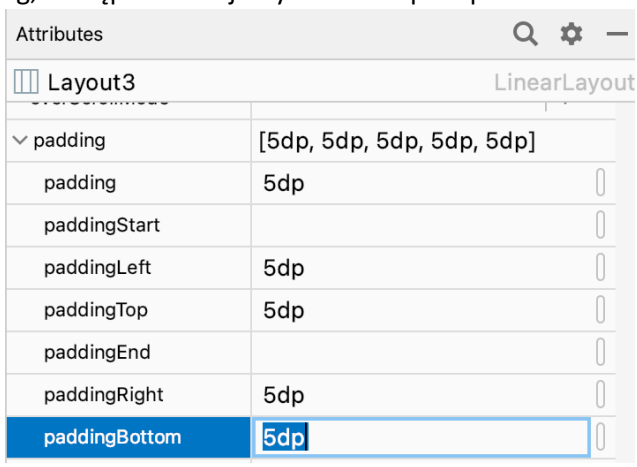
Zmieńmy wygląd aplikacji

1. Odsunięcie przycisków od siebie za pomocą paddingu i marginu
2. Zmiana koloru przycisków
3. Zmiana tła aplikacji
4. Powiększenie czcionki napisu nad przyciskami.

Ad.1

Klikamy Layout3 i w All Attributes odszukujemy margin I padding, następnie dodajemy wartość np. 5dp





W kodzie zostaną dodane atrybuty, które możemy jednocześnie skopiować np. do obu przycisków. Ja skopiuję tylko padding.

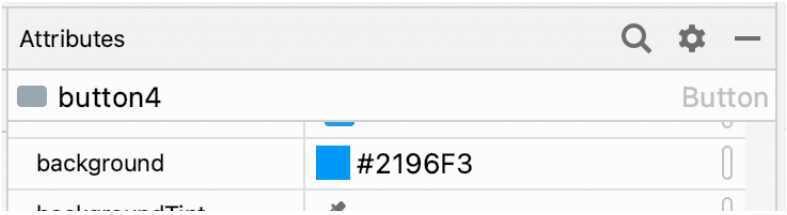
Możemy również wyedytować kod xml podając tylko margin i padding bez rozgraniczenia kierunku: top, botton, left lub right. Otrzymamy kod:

```
49      <LinearLayout
50          android:id="@+id/Layout3"
51          android:layout_width="wrap_content"
52          android:layout_height="wrap_content"
53          android:layout_gravity="center"
54          android:gravity="center"
55          android:orientation="horizontal"
56          android:layout_margin="5dp"
57          android:padding="5dp" >
```

To samo robimy dla przycisków stosując margin np. 5dp:

```
59      <Button
60          android:id="@+id/button3"
61          style="@style/Widget.AppCompat.Button"
62          android:layout_width="wrap_content"
63          android:layout_height="wrap_content"
64          android:gravity="center"
65          android:text="@string/WWW"
66          android:layout_margin="8dp" />
67
68      <Button
69          android:id="@+id/button4"
70          style="@style/Widget.AppCompat.Button"
71          android:layout_width="wrap_content"
72          android:layout_height="wrap_content"
73          android:gravity="center"
74          android:text="@string/FB"
75          android:layout_margin="8dp"/>
```

Ad.2 Kolory przycisków zmieniamy również w atrybutach podając backgroundcolor. Wybiorę przycisk WWW jako zielony, a przycisk FB niebieski (można wpisać wartości szesnastkowe:

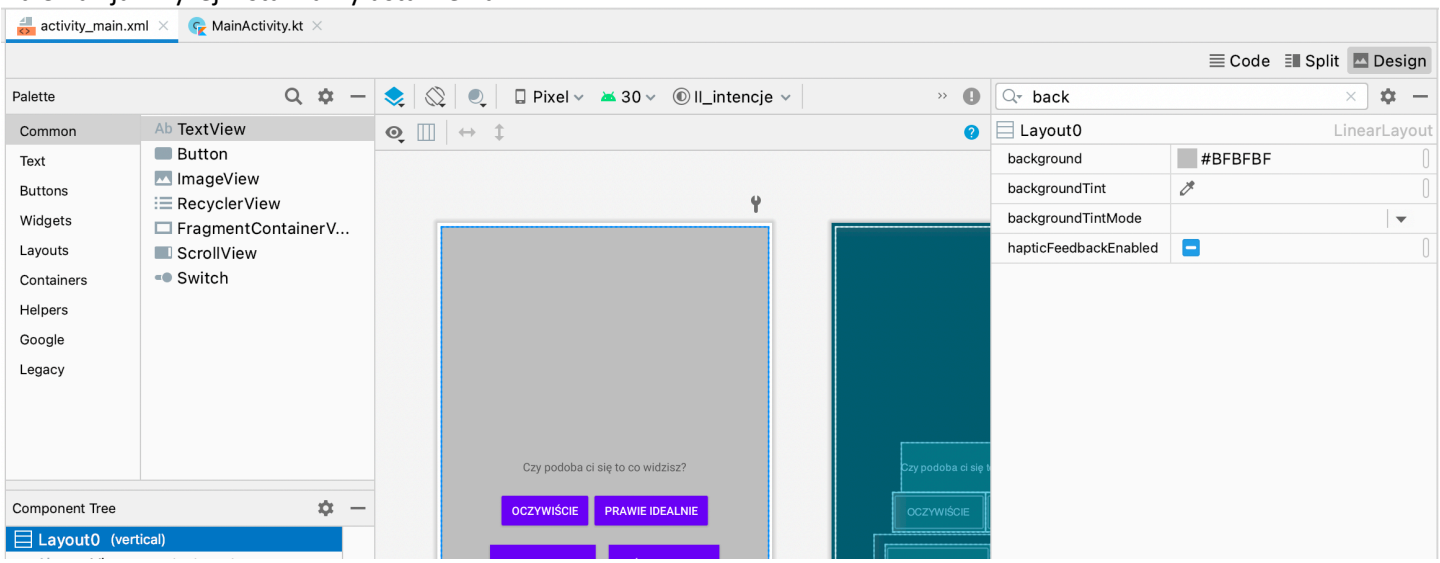


Oczywiście w kodzie zostanie dodana linia kodu definiująca kolor i pokazująca go na marginesie:

```
81      <Button
82          android:id="@+id/button4"
83          style="@style/Widget.AppCompat.Button"
84          android:layout_width="wrap_content"
85          android:layout_height="wrap_content"
86          android:background="#2196F3"
87          android:gravity="center"
```

(Uwaga: Tutaj można utworzyć własny styl. Jak utworzyć styl jest pokazane po pkt. 4.)

Ad.3 Tak jak wyżej wstawiamy ustawienia:



Można stworzyć również własny styl i w nim umieścić atrybuty. (własny styl zostanie utworzony po pkt. 4.)

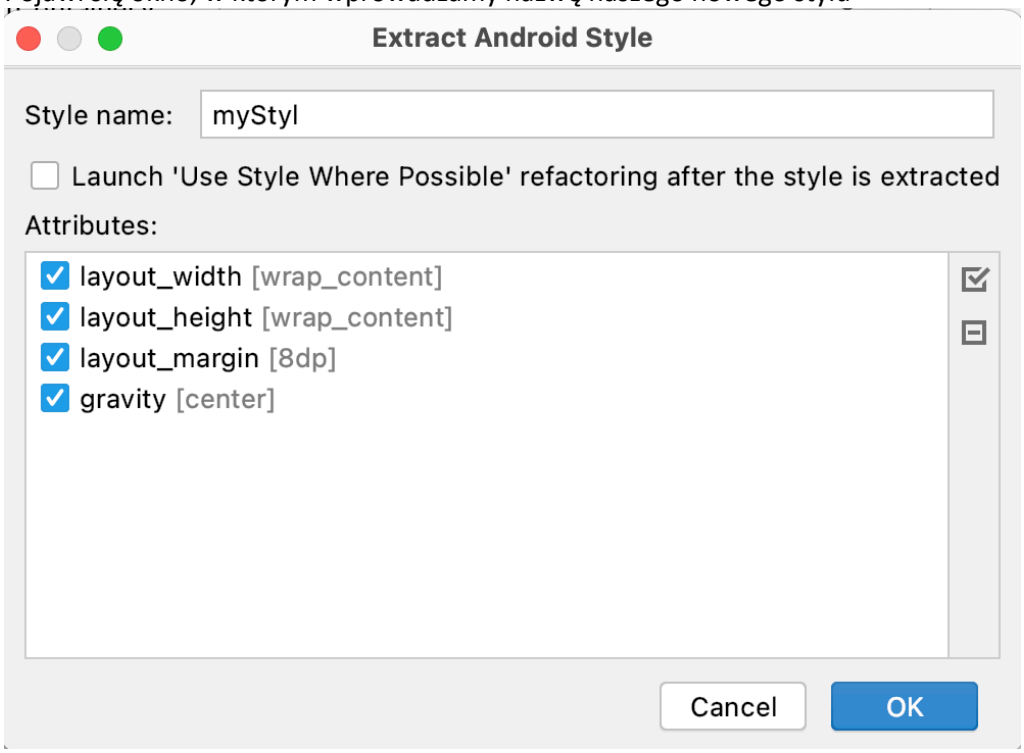
Ad.4

W standardowy sposób zmienimy wielkość i kolor czcionki napisu. W kodzie można dopisać dwie linie:

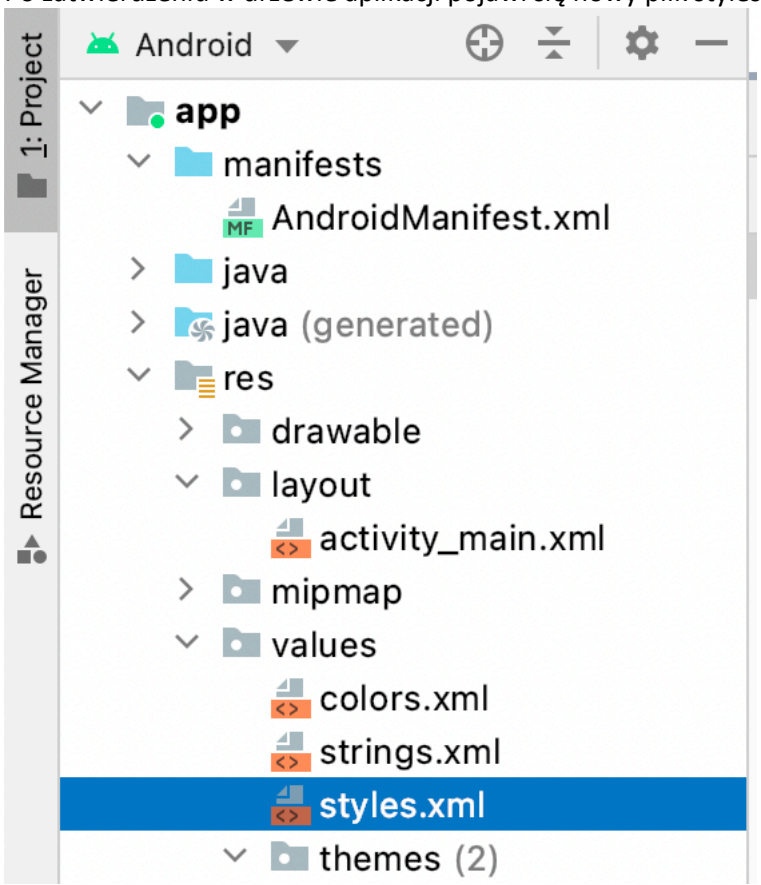
```
15      <TextView
16          android:id="@+id/textView"
17          android:layout_width="wrap_content"
18          android:layout_height="wrap_content"
19          android:layout_gravity="center"
20          android:gravity="center"
21          android:padding="20dp"
22          android:text="Czy podoba ci się to co widzisz?"
23          android:textColor="#FFFF00"
24          android:textSize="20sp" />
```

Utworzenie własnego stylu

W celu utworzenia własnego motywu/stylu należy prawym przyciskiem myszy kliknąć w przycisk i wybrać:
Refactor > Extract Style
Pojawi się okno, w którym wprowadzamy nazwę naszego nowego stylu



Po zatwierdzeniu w drzewie aplikacji pojawi się nowy plik styles.xml



Plik **styles.xml** ma zawartość:

activity_main.xml | styles.xml | MainActivity.kt

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="myStyl" parent="Widget.AppCompat.Button">
        <item name="android:layout_width">wrap_content</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:layout_margin">8dp</item>
        <item name="android:gravity">center</item>
    </style>
</resources>
```

Można dopisać linii kolejne linie kodu stylu. Np.:

activity_main.xml | styles.xml | MainActivity.kt

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="myStyl" parent="Widget.AppCompat.Button">
        <item name="android:layout_width">wrap_content</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:layout_margin">8dp</item>
        <item name="android:gravity">center</item>
        <item name="android:background">@color/naWWW</item>
    </style>
</resources>
```

Postać naszego kody przycisków możemy zmienić na:

```xml
        <Button
            android:id="@+id/button3"
            style="@style/myStyl"
            android:text="@string/WWW" />

        <Button
            android:id="@+id/button4"
            style="@style/myStyl"
            android:text="@string/FB" />
```