Android – style i odstępy

Anna Gogolińska

Wymiary i Marginesy

- Wszystkie elementy na ekranie (Widżety i Rozkłady) muszą mieć zdefiniowaną szerokość (**layout_width**) i wysokość (**layout_height**).
- Do zewnętrznych odstępów używa się marginesów (**margin**).
- Jednostka**:** Używamy **dp.**

Cecha	Atrybut XML	Opis i Wartości
Szerokość elementu	**android:layout_width**	**match_parent** (wypełnij szerokość rodzica), **wrap_content** (dopasuj się do zawartości) lub konkretna wartość (np. "150dp").
Wysokość elementu	**android:layout_height**	**match_parent, wrap_content** lub konkretna wartość (np. "50dp").
Margines (wszędzie)	**android:layout_margin**	Zewnętrzny odstęp ze wszystkich 4 stron (np. "16dp").
Margines (wybrana strona)	**android:layout_marginBottom**	Margines tylko od dołu (np. "8dp"). Dostępne są też **layout_marginLeft, layout_marginRight, layout_marginTop.**

Przykład

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">

   <Button
      android:layout_width="200dp"
      android:layout_height="wrap_content"
      android:text="Przycisk"
      android:layout_margin="12dp" />
    <TextView
      android:layout_width="match_parent
      android:layout_height="wrap_content"
      android:text="Tekst z marginesem u góry"
      android:layout_marginTop="20dp" />

</LinearLayout>
```

Padding – Margines wewnętrzny

- Atrybut **padding** definiuje przestrzeń między **zawartością** elementu (np. tekstem w TextView) a jego **własną krawędzią/obramowaniem**. Działa to jak wewnętrzny margines.
- **Jednostka:** Używamy **dp.**

Cecha	Atrybut XML	Opis i Wartości
Wypełnienie (wszędzie)	android:padding	Wewnętrzny odstęp ze wszystkich 4 stron (np. "16dp").
Wypełnienie (wybrana strona)	android:paddingStart	Wypełnienie tylko z lewej strony (lub początku, np. "10dp"). Dostępne są też paddingTop, paddingBottom, paddingEnd (prawa strona).

Przykład

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Tekst z paddingiem"
    android:padding="20dp"
    android:layout_margin="10dp" />
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Przycisk"
    android:paddingTop="15dp"
    android:paddingBottom="15dp" />
```

Kolor tła

- Atrybut **android:background** służy do nadawania koloru tła zarówno rozkładom (grupom widoków), jak i pojedynczym widżetom (np. napisom).
- **Kolory:** Używamy kodów szesnastkowych (np. #RRGGBB lub #AARRGGBB).

Przykład

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical"
   android:background="#EAECEE">
  <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Napis z czerwonym tłem"
      android:padding="8dp"
      android:background="#E74C3C" />
</LinearLayout>
```

Czcionka – Kolor, Rozmiar i Styl

- **Jednostka:** Rozmiar tekstu określamy w **sp.**
- Styl textStyle pozwala na łączenie wartości znakiem pionowej kreski, np. "bold|italic".

Cecha	Atrybut XML	Opis i Wartości
Kolor Czcionki	**android:textColor**	Kolor w formacie szesnastkowym (np. "#000000").
Wielkość Czcionki	**android:textSize**	Rozmiar w jednostce sp (np. "24sp").
Pogrubienie/Italik	**android:textStyle**	Wartości: bold, italic.

Przykład

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Duży, niebieski, pogrubiony tekst."
      android:textSize="26sp"
      android:textColor="#3498DB"
      android:textStyle="bold" />
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Tekst pochylony i pogrubiony."
      android:textSize="18sp"
      android:textStyle="bold|italic" />
</LinearLayout>
```

Jednostki sp vs dp

Cecha	dp (Density-independent Pixels)	sp (Scale-independent Pixels)
Przeznaczenie	Używana do wszystkich wymiarów geometrycznych.	Używana wyłącznie do rozmiaru tekstu (android:textSize).
Co Oznacza	Piksel Niezależny od Gęstości (skaluje się z rozdzielczością ekranu).	Piksel Niezależny od Skalowania (skaluje się z rozdzielczością i ustawieniami systemowymi).
Dostępność (Accessibility)	Nie skaluje się, gdy użytkownik zmienia systemową wielkość czcionki.	Skaluje się automatycznie, aby dostosować się do preferencji użytkownika.
Przykłady Użycia	android:layout_width="100dp" android:padding="8dp" android:layout_margin="16dp"	android:textSize="20sp"

Wyrównywanie Elementów

- Atrybuty wyrównania kontrolują pozycję elementu względem jego rodzica (**layout_gravity**) lub pozycję zawartości elementu względem jego własnych granic (**gravity**).
  - Rodzic (LinearLayout) musi mieć ustawioną orientację (android:orientation="vertical" lub "horizontal"). Wyrównywanie działa tylko w **kierunku przeciwnym** do orientacji.
    - orientation="vertical" – można wyrównywać element **horyzontalnie** (lewo, prawo, środek), orientation="horizontal" – można wyrównywać element **wertykalnie** (góra, dół, środek).

Atrybut	Gdzie Ustawiany	Co Wyrównuje	Przykładowe Wartości
android: layout_gravity	Na DZIECKU (TextView, Button, itp.)	Wyrównuje ten element względem jego Rodzica (LinearLayout).	center_horizontal, end (prawo), start (lewo)
android:gravity	Na RODZICU (LinearLayout) lub DZIECKU (np. TextView)	Wyrównuje zawartość (TextView – tekst, LinearLayout – dzieci) wewnątrz własnych granic.	center, right, left, bottom

Przykład

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:background="#ECF0F1"
   android:padding="10dp">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Wyrównanie do lewej (Start)"
      android:layout_gravity="start"
      android:padding="5dp" />
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Wyrównanie do Środka"
      android:layout_gravity="center_horizontal"
      android:padding="5dp"/>
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Wyrównanie do prawej (End)"
      android:layout_gravity="end"
      android:padding="5dp"/>
</LinearLayout>
```

Horyzontalna Linia Oddzielająca

- W Androidzie nie ma dedykowanego znacznika <hr>. Taki efekt uzyskujemy, używając podstawowego elementu **View**, któremu nadajemy:
  - Dużą szerokość (match_parent).
  - Bardzo małą wysokość (np. "1dp" lub "2dp").
  - Kolor za pomocą atrybutu tła (android:background).

Przykład

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical">
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Sekcja Pierwsza"
      android:textSize="20sp"
      android:layout_marginBottom="5dp" />
   <View
      android:layout_width="match_parent"
      android:layout_height="2dp"
      android:background="#95A5A6"
      android:layout_marginTop="5dp"
      android:layout_marginBottom="10dp" />
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Sekcja Druga"
      android:textSize="20sp" />
</LinearLayout>
```

Separator listy

- Jeśli używa się ListView to można kontrolować przestrzeń (wysokość) i kolor linii między elementami.

Cecha	Atrybut XML	Element
Wysokość Separatora	android:dividerHeight	ListView
Kolor Separatora	android:divider	ListView

- <ListView
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:divider="#F1C40F"
  android:dividerHeight="4dp" />

Definicja styli

- Style pozwalają na wielokrotne wykorzystywanie tych samych ustawień wizualnych (kolor, rozmiar czcionki, tło, itp.) dla wielu widżetów. Definiuje się je w pliku **res/values/styles.xml**.
  1. Definicja stylu: Tworzysz nowy znacznik <style> z zestawem atrybutów (w res/values/styles.xml).
  2. Dodajesz atrybut **style** do widżetu (w pliku activity_main.xml).

Przykład

```
<resources>
   <style name="MojeTekstowePrzyciski">
   <item name="android:background">#3498DB</item>
   <item name="android:textSize">22sp</item>
   <item name="android:textColor">#FFFFFF</item>
   <item name="android:textStyle">bold</item>
   <item name="android:padding">10dp</item>
</style>
</resources>
```

res/values/styles.xml

```
<LinearLayout
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical">
<Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Inny element z tym samym stylem"
      android:layout_margin="10dp"
      style="@style/MojeTekstowePrzyciski" />
</LinearLayout>
```

activity_main.xml