


Л2. View групи. Android Basics

У попередній лекції розглядався тільки один TextView або один ImageView на екрані. Якщо необхідно додати ще один View елемент нижче, то можна його скопіювати та вставити нижче. Однак виникає помилка, яка пов'язана з тим, що у xml-документу більше однієї кореневої View.

<https://labs.udacity.com/android-visualizer/>



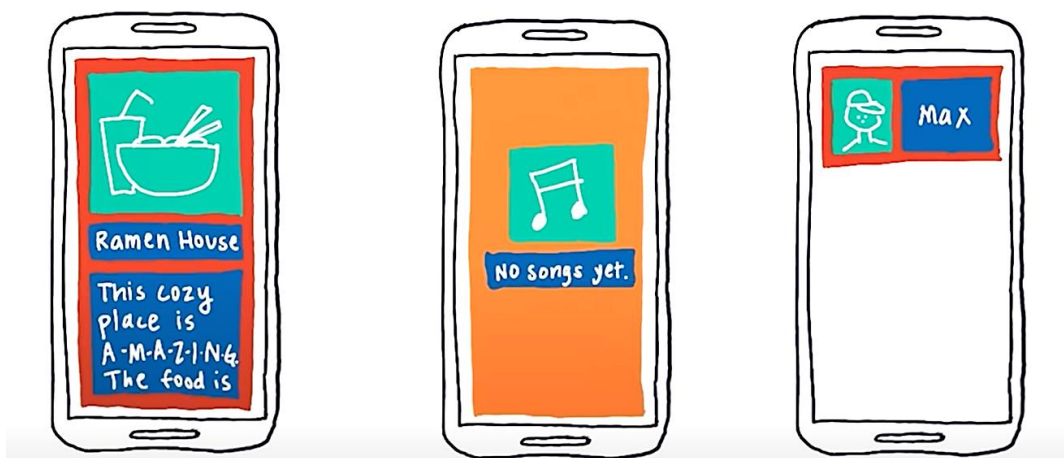
```
1 <TextView
2   android:text="I'm a lonely TextView!"
3   android:layout_width="wrap_content"
4   android:layout_height="wrap_content"
5   android:textSize="36sp" />
6
7 <TextView
8   android:text="I'm a lonely TextView!"
9   android:layout_width="wrap_content"
10  android:layout_height="wrap_content"
11  android:textSize="36sp" />
```

Your XML document has more than one root view. There can only be one root view, and it should enclose all of your other views.

The computer wasn't able to understand your XML. (If any of your code is colored red, the error might be just before it.)

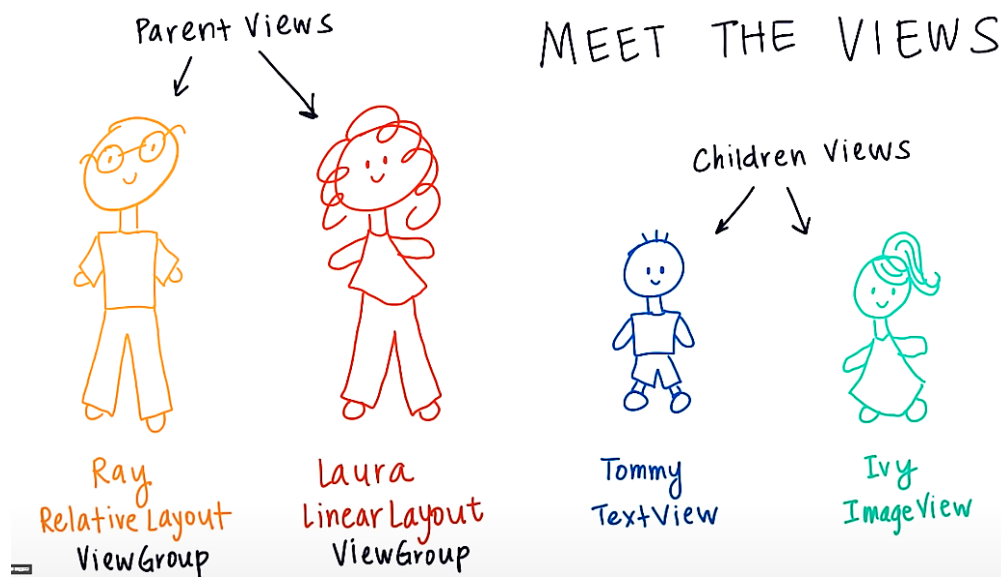
Коренева View може бути тільки одна, а всі інші повинні входити в неї. Якщо хочемо щоб обидві TextView залишилися на екрані, тоді потрібно помістити їх в групу View. В цьому випадку група View буде тією єдиною кореневою View для макета.

VIEW GROUPS



View Groups самі є View. На екрані вони мають прямокутну форму. Наприклад червоні та помаранчеві прямокутники. Це все View Groups. У них є ширина висота і колір фону, а також інші атрибути. View Groups можна

представити як контейнер для View. Якщо View містить інші View, вона називається батьком цих View.



Дитячі View розміщуються відносно їх батьків.



Батьки контролюють позицію своєї дитини, наприклад, одна дитяча TextView може бути у верхній частині екрану, а інша дитяча ImageView під нею або якщо батьки захотять, то може розташувати дітей по-іншому. Дитина ImageView може бути зліва, а тоді дитина TextView праворуч.



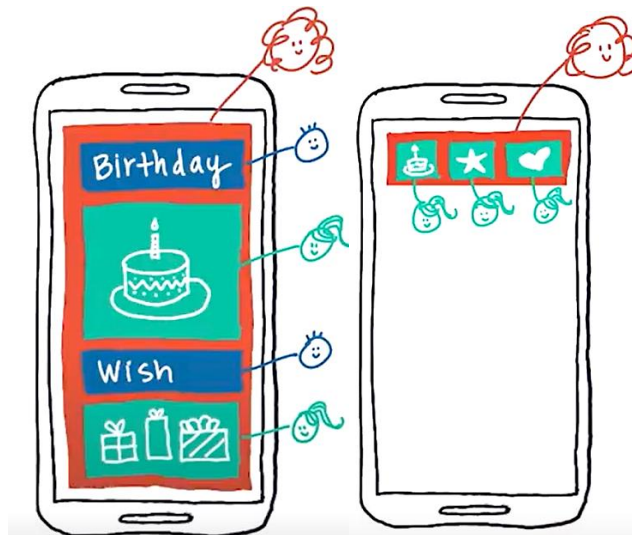
Даємо відповіді на запитання

1. How many views are there?
2. The mountain image is a _____ of the red ViewGroup.
 child parent
3. The red ViewGroup is the _____ of the "Hiking" TextView.
 child parent
4. Which views are siblings of each other?



Типи View-груп

У кожній ViewGroup є певні правила за якими дочірні елементи розташовуються усередині групи. У **LinearLayout** дочірні елементи можуть розташовуватися у вертикальній колонці. Немає обмежень по кількості TextView та ImageView. Їх можемо додавати у будьякій кількості як і інші типи View.



У **LinearLayout** дочірні елементи також можуть розташовуватися в горизонтальному ряду.

При **RelativeLayout** застосовуються свої правила розташування дочірніх елементів на екрані, наприклад, можна розташувати дочірній ImageView зверху над батьківським, TextView внизу щодо батьківського. Ще одна корисна функція **Relative Layout** - можливість розташувати одні дочірні

елементи щодо інших. `ImageView` можна помістити зліва від батька, а потім розташувати текст праворуч і ще текст нижче.

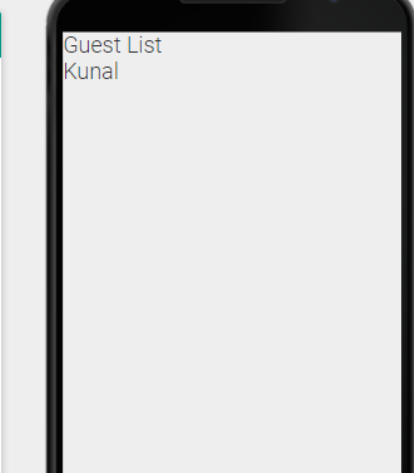


LinearLayout

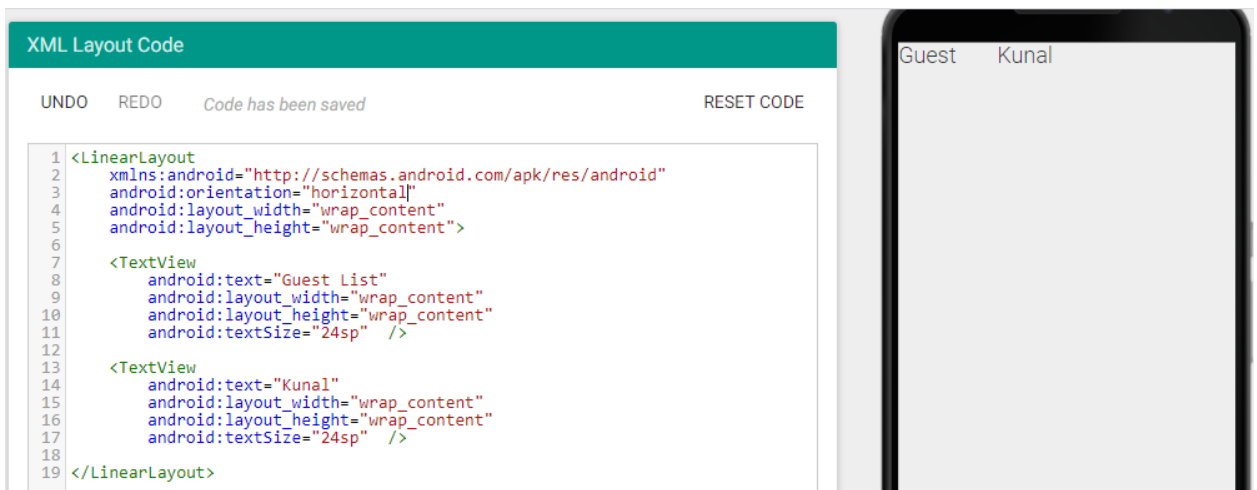
XML Layout Code

UNDO REDO Code has been saved RESET CODE

```
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="wrap_content"
5   android:layout_height="wrap_content">
6
7   <TextView
8     android:text="Guest List"
9     android:layout_width="wrap_content"
10    android:layout_height="wrap_content"
11    android:textSize="24sp" />
12
13   <TextView
14     android:text="Kunal"
15     android:layout_width="wrap_content"
16     android:layout_height="wrap_content"
17     android:textSize="24sp" />
18
19 </LinearLayout>
```



Давайте розглянемо код уважніше. Він починається з кутової дужки, що відкриває, потім ім'я View групи **LinearLayout**, потім перераховуються атрибути та в кінці кутова дужка, що закриває. Дочірні елементи вставляються між тегами, що відкривають та закривають. Атрибут **android:orientation** визначає чи є розмітка стовпцем або рядком. Значення цього атрибута **horizontal** для рядка та **vertical** для стовпця.



Рядок `xmlns:android=http://schemas.android.com/apk/res/android` це оголошення простору імен XML, щоб вказати що всі атрибути відносяться до Андроїд. Тому всі атрибути починаються з `android`.

match_Parent

Параметр атрибуту `match_Parent` вказує на відповідність батьківському View. Розмір по ширині та висоті дочірніх елементів буде відповідати батьківському.



Кілька прикладів, як різні настройки ширини дочірніх елементів в лінійній розмітці можуть вплинути на кінцевий інтерфейс користувача. Задаємо для кожного дочірнього елементу фіксовану ширину 200 dp. Якщо контент більше 200 dp, то обрізається його частина. Якщо задаємо для кожного дочірнього елементу ширину `wrap_content`, то ширина контенту кожного елементу різна. Якщо задати ширину `math_parent`, то ширина усіх дочірніх елементі буде однаковою та не залежить від їх контенту.

200 dp



wrap-content



match-parent



Аналогічно змінюється і висота дочірніх елементів, якщо встановити значення 200dp, wrap_content, match_parent.

Розглянемо XML код з атрибутом layout_widht, який має значення wrap_content та match_parent (1_match_Parent).

```
XML Layout Code
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:background="@android:color/darker_gray">
7
8   <TextView
9     android:text="VIP List"
10    android:layout_width="wrap_content"
11    android:layout_height="wrap_content"
12    android:background="#4CAF50"
13    android:textSize="24sp" />
14
15   <TextView
16     android:text="Kunal"
17     android:layout_width="wrap_content"
18     android:layout_height="wrap_content"
19     android:background="#4CAF50"
20     android:textSize="24sp" />
21
22   <TextView
23     android:text="Kagure"
24     android:layout_width="wrap_content"
25     android:layout_height="wrap_content"
26     android:background="#4CAF50"
27     android:textSize="24sp" />
```



```
XML Layout Code
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:background="@android:color/darker_gray">
7
8   <TextView
9     android:text="VIP List"
10    android:layout_width="match_parent"
11    android:layout_height="wrap_content"
12    android:background="#4CAF50"
13    android:textSize="24sp" />
14
15   <TextView
16     android:text="Kunal"
17     android:layout_width="match_parent"
18     android:layout_height="wrap_content"
19     android:background="#4CAF50"
20     android:textSize="24sp" />
21
22   <TextView
23     android:text="Kagure"
24     android:layout_width="match_parent"
25     android:layout_height="wrap_content"
26     android:background="#4CAF50"
27     android:textSize="24sp" />
28
29   <TextView
```




Рівномірний розподіл дочірніх View-елементів (android:layout_weight)

XML Layout Code

UNDO REDO Code has been saved RESET CODE

```
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="wrap_content"
5   android:layout_height="wrap_content">
6
7   <TextView
8     android:text="Tom"
9     android:layout_width="wrap_content"
10    android:layout_height="wrap_content"
11    android:textSize="24sp" />
12
13  <TextView
14    android:text="Tim"
15    android:layout_width="wrap_content"
16    android:layout_height="wrap_content"
17    android:textSize="24sp" />
18
19  <TextView
20    android:text="Todd"
21    android:layout_width="wrap_content"
22    android:layout_height="wrap_content"
23    android:textSize="24sp" />
24
25 </LinearLayout>
```

AVAILABLE IMAGES REPORT ISSUE CHEAT SHEET



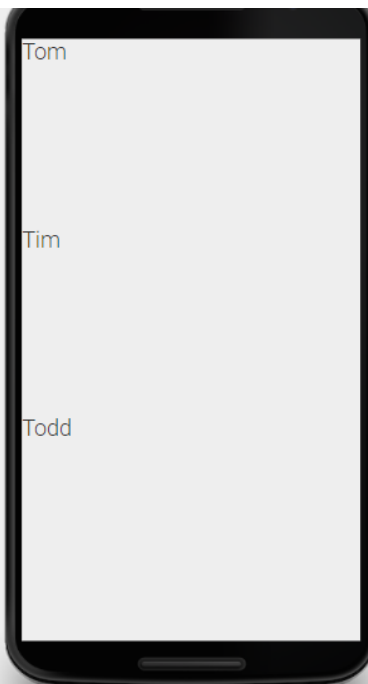
При такому рішенні TextView розподіляться більш рівномірно по екрану, однак таке рішення буде працювати, тільки для певного розміру екрану.

XML Layout Code

UNDO REDO Code has been saved RESET CODE

```
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="wrap_content"
5   android:layout_height="wrap_content">
6
7   <TextView
8     android:text="Tom"
9     android:layout_width="wrap_content"
10    android:layout_height="200dp"
11    android:textSize="24sp" />
12
13  <TextView
14    android:text="Tim"
15    android:layout_width="wrap_content"
16    android:layout_height="200dp"
17    android:textSize="24sp" />
18
19  <TextView
20    android:text="Todd"
21    android:layout_width="wrap_content"
22    android:layout_height="200dp"
23    android:textSize="24sp" />
24
25 </LinearLayout>
```

AVAILABLE IMAGES REPORT ISSUE CHEAT SHEET



Якщо взяти значення атрибуту **match_parent**, то перший елемент буде мати висоту батьківського елемента та виштовхує усі інші елементи за межі екрану (всі інші елементи не будуть видні). Досить часто застосовують атрибут **android:layout_weight**. У документації зазначається, для того щоб рівномірно розподілити усі елементи по вертикалі, необхідно задати для всіх

елементів `android:layout_height="0dp"` та `android:layout_weight="1"` (вагу елементів)

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@android:color/darker_gray"
    android:paddingLeft="16dp"
    android:paddingRight="16dp">

    <TextView
        android:text="VIP List"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="#000080"
        android:layout_weight="1"
        android:textSize="24sp" />

    <TextView
        android:text="Kunal"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="#008080"
        android:layout_weight="1"
        android:textSize="24sp" />

    <TextView
        android:text="Kagure"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:gravity="top"
        android:background="#00FF7F"
        android:textSize="24sp" />

    <TextView
        android:text="Lyla"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="#9ACD32"
        android:layout_weight="1"
        android:textSize="24sp" />
</LinearLayout>
```


XML Layout Code

UNDO

REDO

Code has been saved

RESET CODE

```
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:background="@android:color/darker_gray"
7   android:paddingLeft="16dp"
8   android:paddingRight="16dp">
9
10  <TextView
11     android:text="VIP List"
12     android:layout_width="0dp"
13     android:layout_height="0dp"
14     android:background="#000080"
15     android:layout_weight="1"
16     android:textSize="24sp" />
17
18  <TextView
19     android:text="Kunal"
20     android:layout_width="0dp"
21     android:layout_height="0dp"
22     android:background="#008080"
23     android:layout_weight="1"
24     android:textSize="24sp" />
25
26  <TextView
27     android:text="Kagure"
```

AVAILABLE IMAGES

REPORT ISSUE

CHEAT SHEET



Чим більше вага елемента, тим більше простір батьковського елемента буде зайнятий. Для Kunal, Kagure, Lyla висота не задана, тому між ними буде виставлена висота, а решта висоти буде відведено елементу з вагою 1.

XML Layout Code

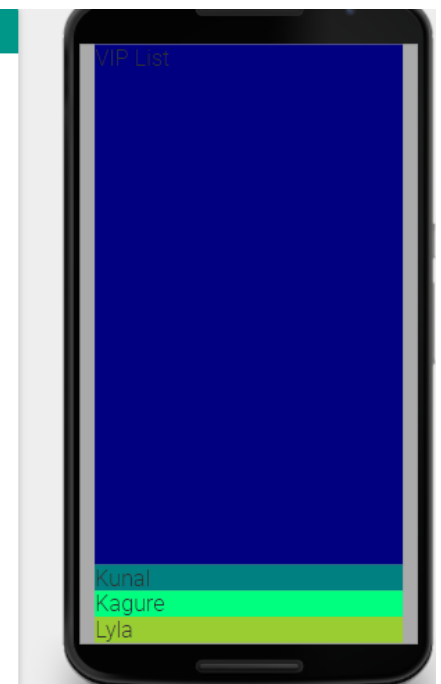
UNDO

REDO

Code has been saved

RESET CODE

```
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:background="@android:color/darker_gray"
7   android:paddingLeft="16dp"
8   android:paddingRight="16dp">
9
10  <TextView
11     android:text="VIP List"
12     android:layout_width="0dp"
13     android:layout_height="0dp"
14     android:background="#000080"
15     android:layout_weight="1"
16     android:textSize="24sp" />
17
18  <TextView
19     android:text="Kunal"
20     android:layout_width="match_parent"
21     android:layout_height="wrap_content"
22     android:background="#008080"
23     android:textSize="24sp" />
24
25  <TextView
26     android:text="Kagure"
27     android:layout_width="match_parent"
28     android:layout_height="wrap_content"
29     android:gravity="top"
30     android:background="#00FF7F"
31     android:textSize="24sp" />
32
33  <TextView
```

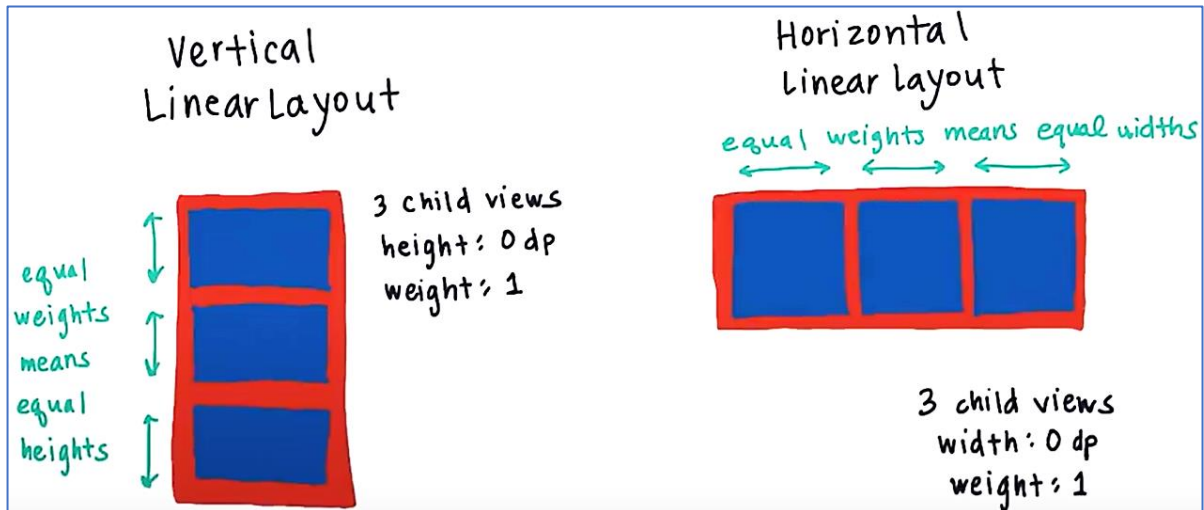


```

XML Layout Code
UNDO  REDO  Code has been saved  RESET CODE

1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:orientation="vertical"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent"
6   android:background="@android:color/darker_gray"
7   android:paddingLeft="16dp"
8   android:paddingRight="16dp">
9
10  <TextView
11   android:text="VIP List"
12   android:layout_width="0dp"
13   android:layout_height="0dp"
14   android:background="#000080"
15   android:layout_weight="1"
16   android:textSize="24sp" />
17
18  <TextView
19   android:text="Kunal"
20   android:layout_width="0dp"
21   android:layout_height="0dp"
22   android:background="#008080"
23   android:layout_weight="7"
24   android:textSize="24sp" />
25
26  <TextView
27   android:text="Kagure"
28   android:layout_width="0dp"
29   android:layout_height="0dp"
30   android:layout_weight="2"
31   android:gravity="top"
32   android:background="#00FF7F"
33   android:textSize="24sp" />

```




Розглянемо реальний додаток. Розглянемо нижній рядок, який має Horizontal Linear layout. Елемент по центру має вагу 1, тому ширина бокових елементів встановлюється по змісту, а він займає усю ширину.

Horizontal Linear layout

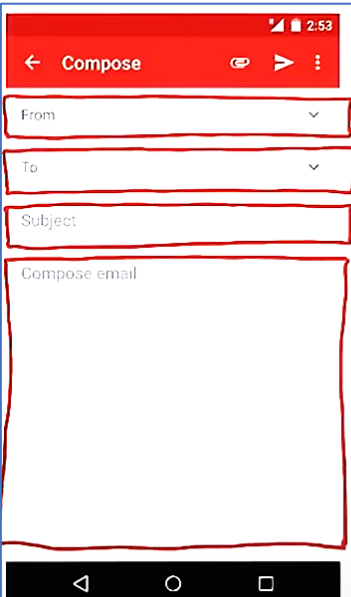
	ImageView	EditText	ImageView
Height	wrap_content	wrap_content	wrap_content
Width	wrap_content	0dp	wrap_content
weight	0	1	0

Ще один додаток. Розглянемо групу центральних елементів з Horizontal Linear layout. Важливо щоб усі три елементи рівномірно займали ширину, тому у них усіх вага 1.



Horizontal Linear layout

	CALL	SAVE	WEBSITE
Height	wrap_content	wrap_content	wrap_content
Width	Odp	Odp	Odp
weight	1	1	1



Vertical Linear layout

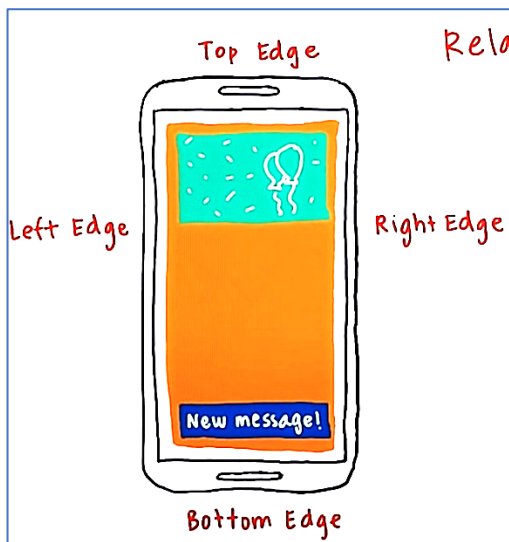
	Width	Height	Weight
Edit Text	match_parent	wrap_content	0
Edit Text	match_parent	wrap_content	0
Edit Text	match_parent	wrap_content	0
Edit Text	match_parent	Odp	1

Завдання. Змінити код, щоб отримати такий варіант розмітки (використовуємо Vizualizer). <https://labs.udacity.com/android-visualizer/#/android/linear-layout-weight>

Relative Layout

Дочірні елементи відносно батьківського можна розташувати по правому, лівому, верхньому або нижньому краю.

Relative to Parent



ImageView attributes:

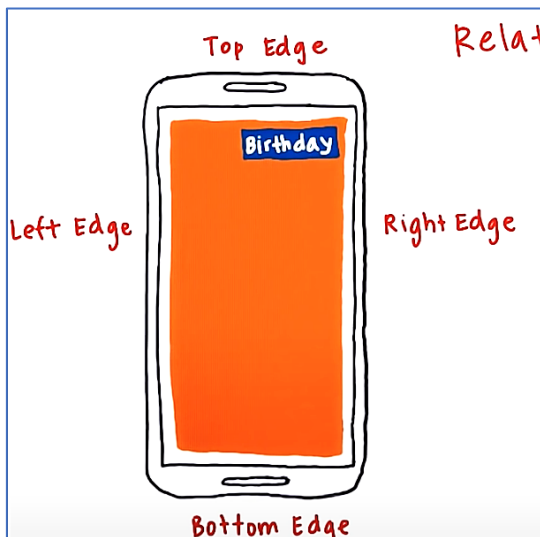
```

android:layout_alignParentTop = "true"
android:layout_alignParentBottom = "false"
android:layout_alignParentLeft = "true"
android:layout_alignParentRight = "true"

```

Для вирівнювання елемента по правому верхньому куту необхідно встановити такі атрибути

Relative to Parent



Child view attributes:

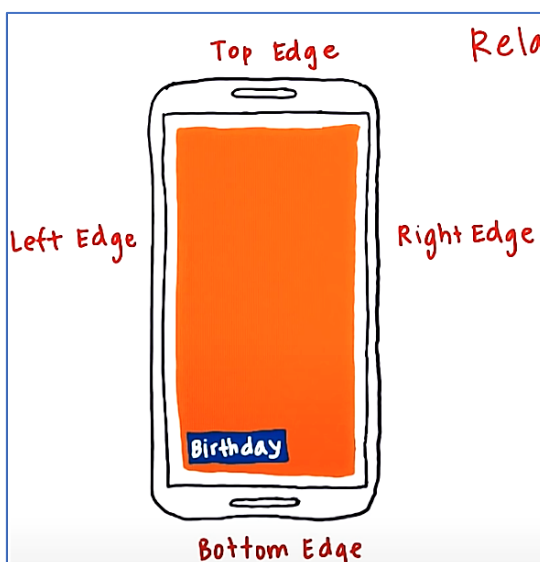
```

android:layout_alignParentTop = "true"
android:layout_alignParentBottom
android:layout_alignParentLeft
android:layout_alignParentRight = "true"

```

Для встановлення тексту в лівому нижньому куту використовуємо такі атрибути. По замовчуванню атрибути встановлені у значенні false.

Relative to Parent



Child view attributes:

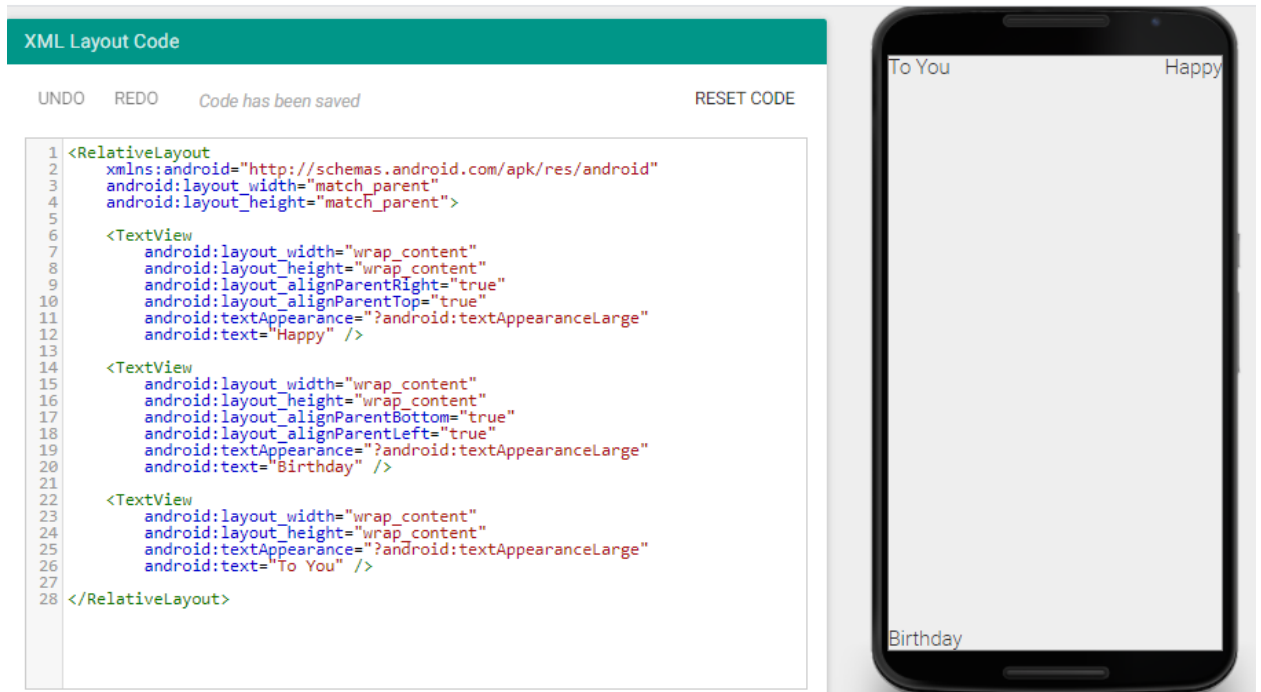
```

android:layout_alignParentTop
android:layout_alignParentBottom = "true"
android:layout_alignParentLeft = "true"
android:layout_alignParentRight

```

Є ще корисні атрибути, які центрують зображення: `layout_centerVertical="true"` та `layout_centerHorizontal="true"`. Усі атрибути називаються параметрами групи розмітки.

Розглянемо код з `RelativeLayout` (`4_relativeLayout`)



Є тег, що відкриває, **RelativeLayout** та тег, що закриває в кінці розмітки. Значення ширини та висоти для батьківського елемента задано **match parent**. Значення ширини та висоти кожного дочірнього елемента – **wrap_content** (по змісту). Для всіх дочірніх елементів є атрибути, які вказують відносно розташування (верхній правий кут, нижній лівий, верхній лівий).

Якщо не вказано розміщення дочірнього елемента, то за замовчуванням він додається в лівий верхній кут екрану.

Ставлення до інших View-елементів. Ідентифікатор

Припустимо, що один `TextView` має бути вище іншого або одне зображення має бути ліворуч від іншого. Припустимо, що надписи потрібно розташувати в такому порядку.



Робимо через `RelativeLayout`. `Ben` вирівнюється по верхньому краю по центру, `Lyla` - по нижньому лівому краю, `Jennie` - по нижньому правому краю. Вони називається представленням прив'язки.

Як розташувати `Kunal` ліворуч від `Ben`. Існує атрибут **`android:layout_toLeftOf`**, **`android:layout_toRightOf`**. Як задати відносно якого об'єкта? Для цього `View` елементам задають ID ім'я (ідентифікатор). Наприклад:

```
android:id="@+id/ben_text_view"
```

Символ `@` (at) відноситься до вказання на ресурс `android`. Далі вказуємо тип ресурсу. В даному випадку це **`id`**. Знак `+` вказуємо, що цей `id` оголошуємо в перше. Далі йде слеш та ім'я елемента, яке вибираємо для `ben`. Ім'я не повинно містити пробіли, повинно починатись з букви та не містити незрозумілі символи пунктуації. Можна цифри. Тоді, щоб розташувати `Kunal` ліворуч від `Ben` необхідно встановити такий атрибут

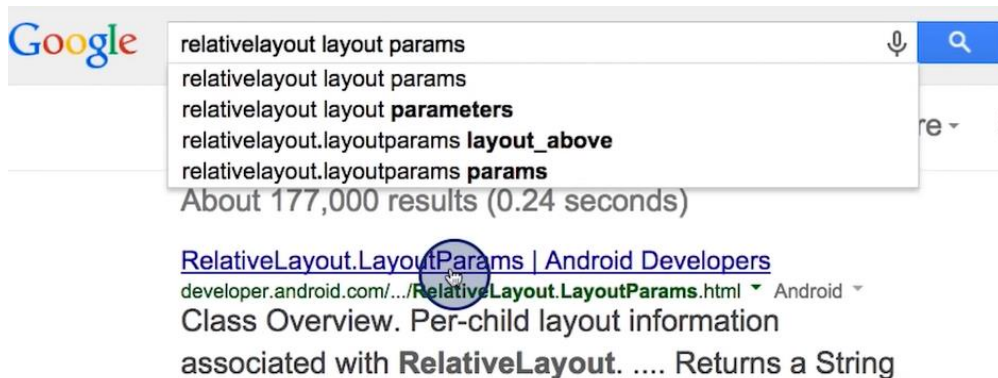
```
android:layout_toLeftOf="@id/ben_text_view"
```

Знак `+` у виразі **`@id/ben_text_view`** не вказуємо, тому `id` вже визначений.

Розглянемо як `Omoju` розташувати вище `Jennie`. Для цього `Jennie` потрібно задати ідентифікатор **`android:id="@+id/jennie_text_view"`**. Для `Omoju` вказуємо атрибут

```
android:layout_above="@id/jennie_text_view"
```

Атрибути розташування не потрібно запам'ятовувати. Їх завжди можна знайти в документації [RelativeLayout.LayoutParams](#)



Наприклад, можна знайти атрибут розташувати нижче

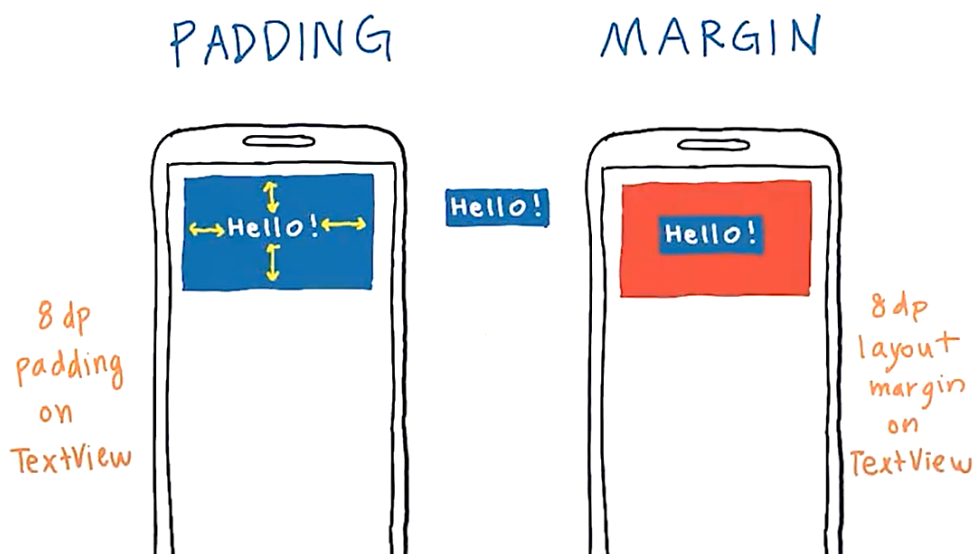
```
android:layout_below
```

Внутрішній і зовнішній відступ

Для того щоб зображення виглядало краще, необхідно додати відступи



Розглянемо, що буде якщо додати відступ до елемента TextView

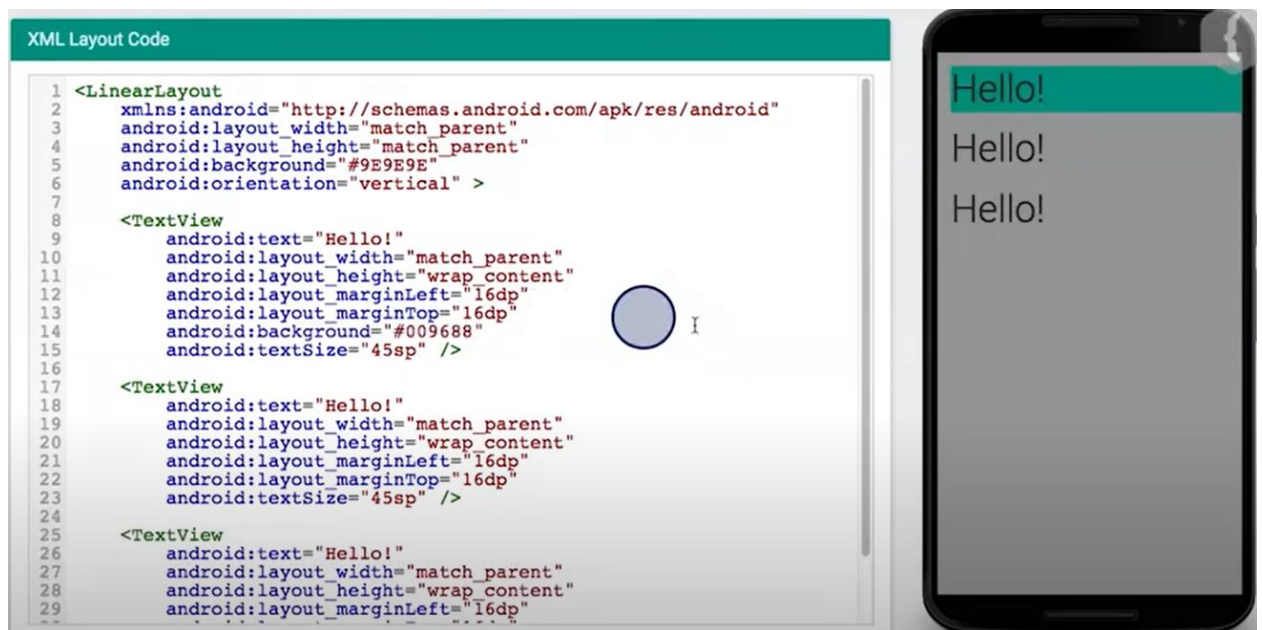


Якщо додати **padding** 8dp до елемента TextView, то він збільшиться з усіх сторін на 8dp; якщо додати **margin**, то результат буде як на картинці. Розмір тексту скорочується на 8dp. **Відступ створюється дочірнім елементом, а поля батьківським елементом.** Атрибути, що задають відступи

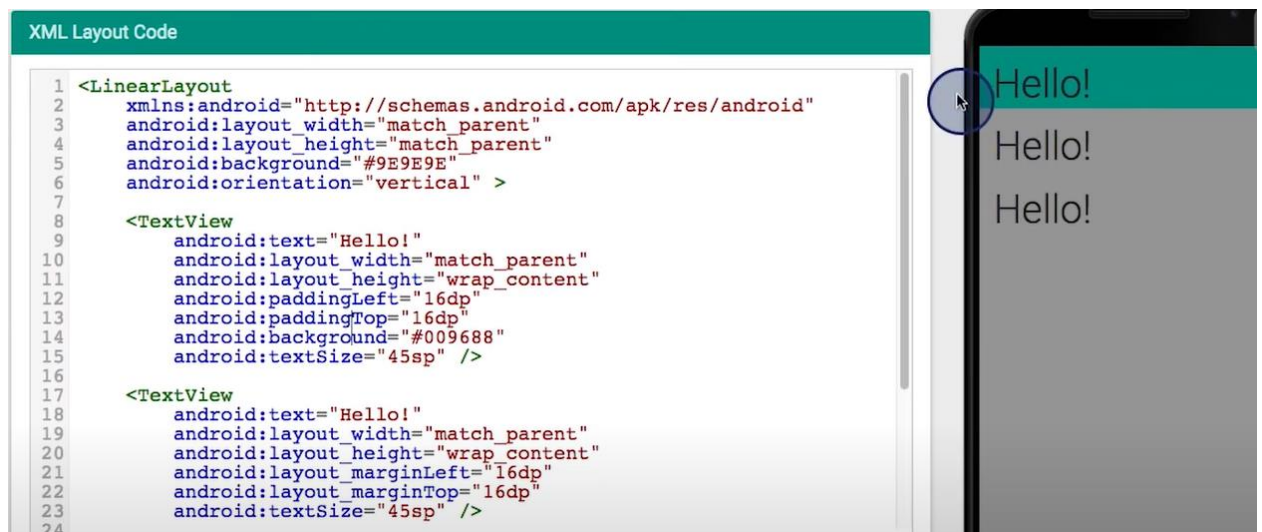
```
android:padding="8dp" або
    android:paddingLeft="8dp"
    android:paddingRight="8dp"
    android:paddingTop="8dp"
    android:paddingBottom="8dp"
```

Атрибути, що задають поля

```
android:layout_margin="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
```



Margin поміняли на padding (8_Layout_pading)



Заготовка коду для демонстрації
<LinearLayout


```
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="#9E9E9E"
android:orientation="vertical" >
```

```
<TextView
    android:text="Hello!"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingLeft="16dp"
    android:paddingTop="16dp"
    android:background="#009688"
    android:textSize="45sp" />
```

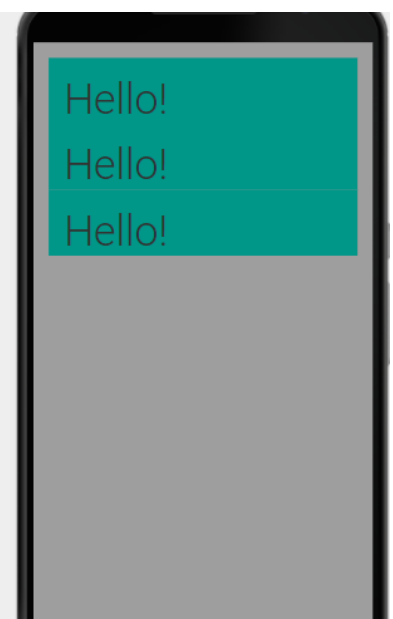
```
<TextView
    android:text="Hello!"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="16dp"
    android:layout_marginTop="16dp"
    android:textSize="45sp" />
```

```
<TextView
    android:text="Hello!"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="16dp"
    android:layout_marginTop="16dp"
    android:textSize="45sp" />
```

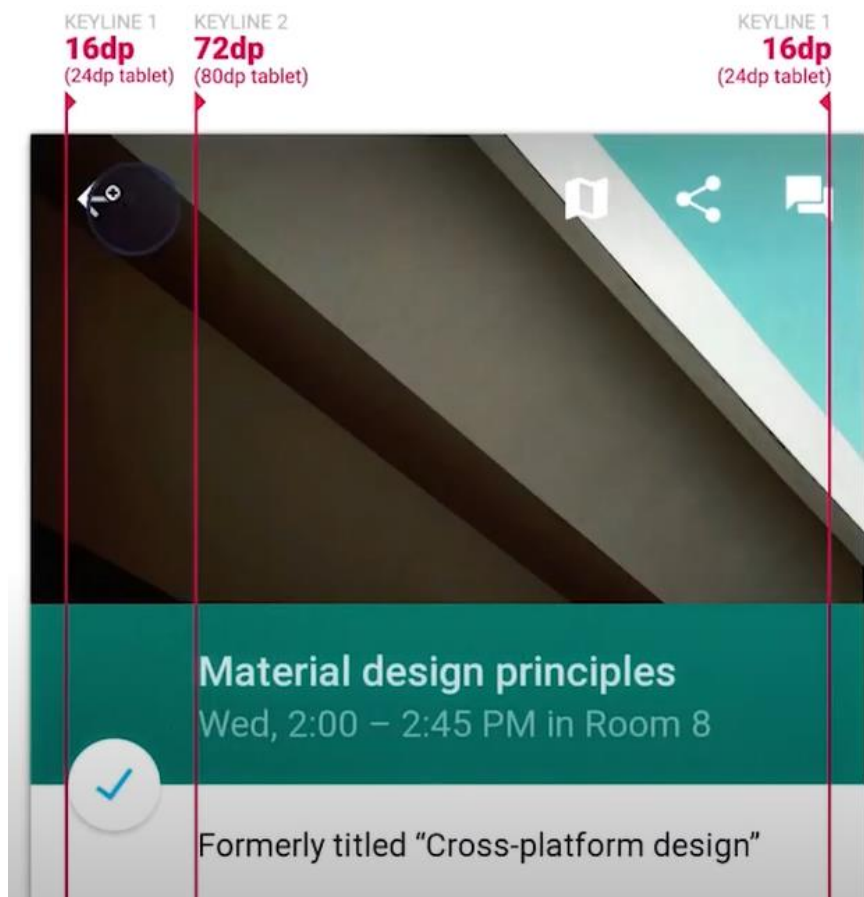
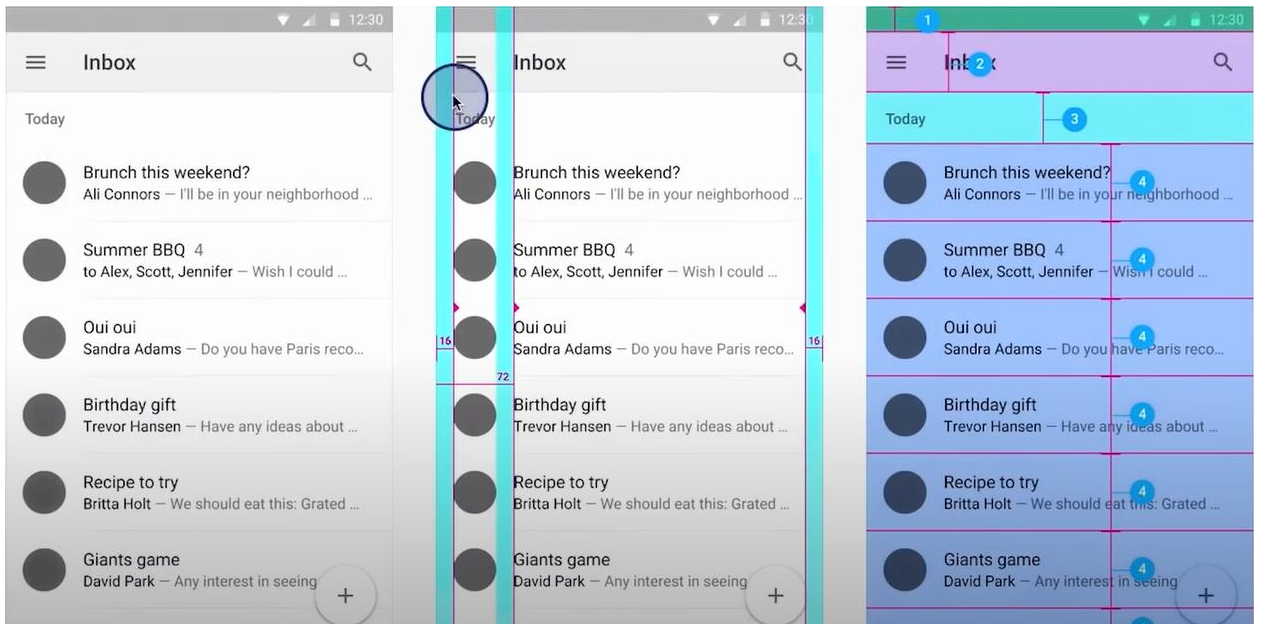
```
</LinearLayout>
```

Якщо додати padding в батьківському елементі, то появляться виступи з усіх сторін на 16dp.

```
XML Layout Code
UNDO  REDO  Code has been saved  RESET CODE
1 <LinearLayout
2   xmlns:android="http://schemas.android.com/apk/res/android"
3   android:layout_width="match_parent"
4   android:layout_height="match_parent"
5   android:background="#9E9E9E"
6   android:orientation="vertical"
7   android:padding="16dp">
8
9   <TextView
10    android:text="Hello!"
11    android:layout_width="match_parent"
12    android:layout_height="wrap_content"
13    android:paddingLeft="16dp"
14    android:paddingTop="16dp"
15    android:background="#009688"
16    android:textSize="45sp" />
17
18   <TextView
19    android:text="Hello!"
20    android:layout_width="match_parent"
21    android:layout_height="wrap_content"
22    android:paddingLeft="16dp"
23    android:paddingTop="16dp"
24    android:background="#009688"
25    android:textSize="45sp" />
26
27   <TextView
28    android:text="Hello!"
29    android:layout_width="match_parent"
30    android:paddingLeft="16dp"
31    android:paddingTop="16dp"
32    android:background="#009688"
33    android:textSize="45sp" />
34
35 </LinearLayout>
```



Рекомендується встановлювати відступи та поля кратні 8dp (material disaine)



В дужках для планшетів

5:00

July 16

15 Tue

Trip home

Flight to New York
7:15 AM - 3:35 PM
Los Angeles LAX

Drinks with Katie
5:45 - 7:30 PM
Balthazar

16 Wed

Sam Blitzstein
Birthday

John and Mark's wedding celebration
6:00 - 10:30 PM

18 Fri

Vintage clothes market
4 PM

5:00

Flight to New York

Virgin America
Flight 404
Tuesday, July 15
Delayed: departing 7:25 AM

Departs Los Angeles (LAX)
7:15 AM
Terminal 3
Gate E2

Arrives New York (JFK)
3:35 PM
Terminal 5
Gate 11