Android השתלמות מתחילים

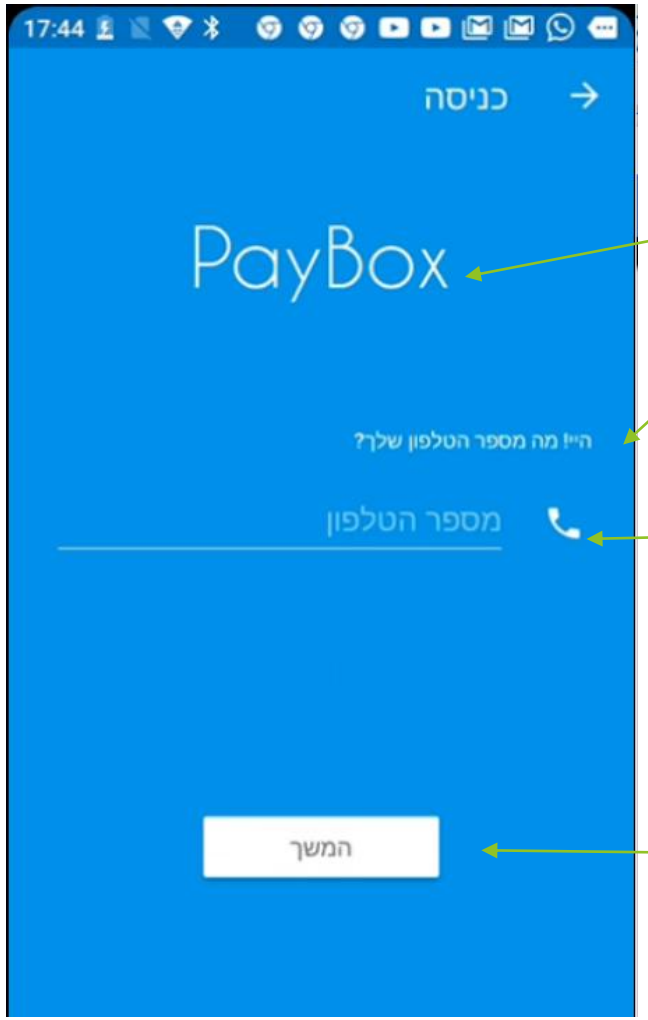
UI Layouts

אלון חימוביץ

תיכון אהל-שם/ עירוני ל"ד שמיר

מכללת הרמלין להנדסה

אילו רכיבים יוצרים כאן את ממשק המשתמש - UI?



תיבת טקסט

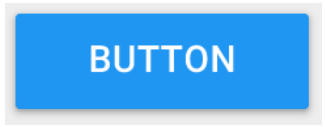
תיבת קלט

כפתור

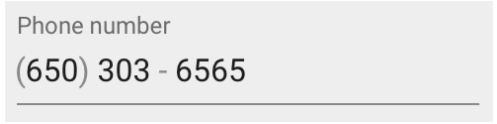
רכיב - View

- דוגמה לרכיבים על המסך ליצירת אינטראקציה עם המשתמש
- מהם המאפיינים שיכולים להיות לכל רכיב?
- הערה: יש מאות רכיבים שמוגדרים על ידי אנדרואיד

Button



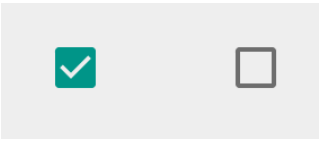
EditText



Slider

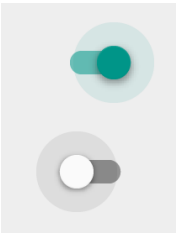
TextView

CheckBox



RadioButton

Switch

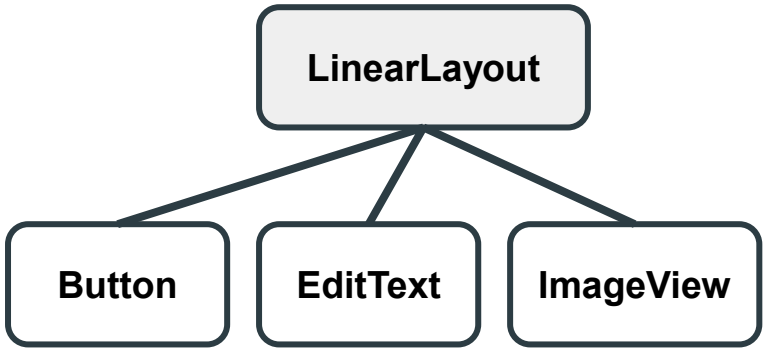


ImageView

Layout

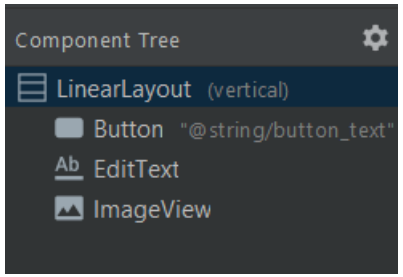
- Layout מיכל שעוטף Views בהתאם לקריטריונים ומגדיר את הקשרים ביניהם.
- נקרא גם ViewGroup - כלומר מכיל קבוצת רכיבים בתוכו
- ישנם מספר סוגים של Layout, אין חובה להכיר את כולם - לפי צורך.

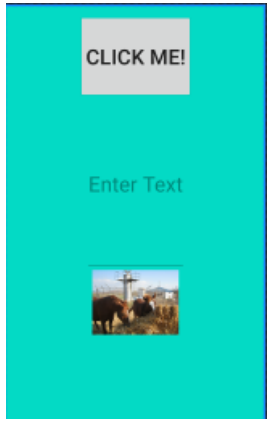
- Linear Layout
- Relative Layout
- Table Layout
- Constraint Layout

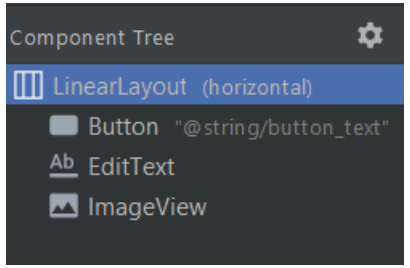


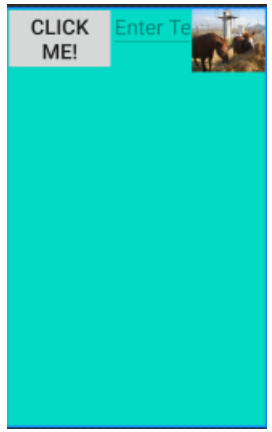
Linear Layout

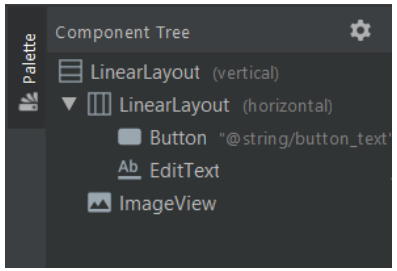
- הרכיבים מסודרים אחד אחרי השני
- אנכי או אופקי - בהתאם להגדרה
- יכולים להיות Nested

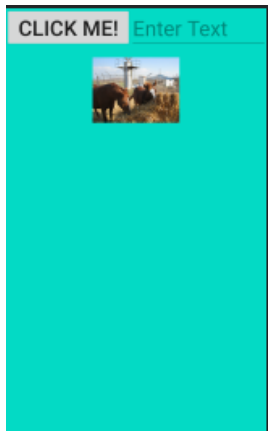






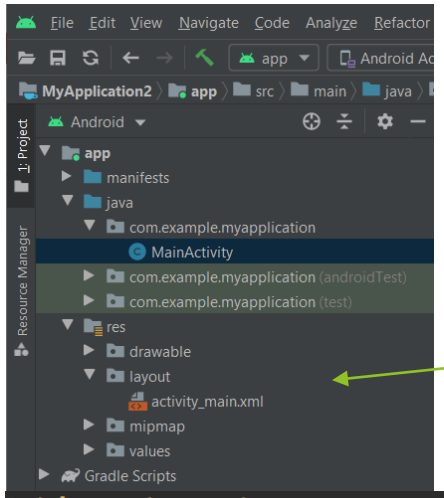






מתיאוריה לפרקטיקה

ה Layout- מוגדר בקובץ XML




```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
        <Button
            android:layout_height="match_parent"
            android:layout_width="wrap_content"
            android:text="@string/button_text"
            android:textSize="40sp"
            />
        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="40sp"
            android:hint="@string/enter_text"
            android:inputType="text"
            />
</LinearLayout>
```

1. רוחב/גובה נקבע בהתאם ל View שמכיל אותו
2. רוחב/גובה נקבע בהתאם לתוכן שנמצא
3. סידור הרכיבים - אופקי/אנכי
4. נמצא בקובץ מחרוזות קבועות - בהמשך...
5. dp- הגודל נקבע בהתאם לצפיפות הפיקסלים במסך המכשיר
   sp - זהה לdp תוך התייחסות להעדפת המשתמש מבחינת גודל הfont במכשיר
6. הגדרת סוג הקלט שמתקבל ב EditText
7. תגית פתיחה וסיום מחוייבת בכל רכיב שמכיל בתוכו View.

Always use sp for text.
regular paragraph: 14-16s

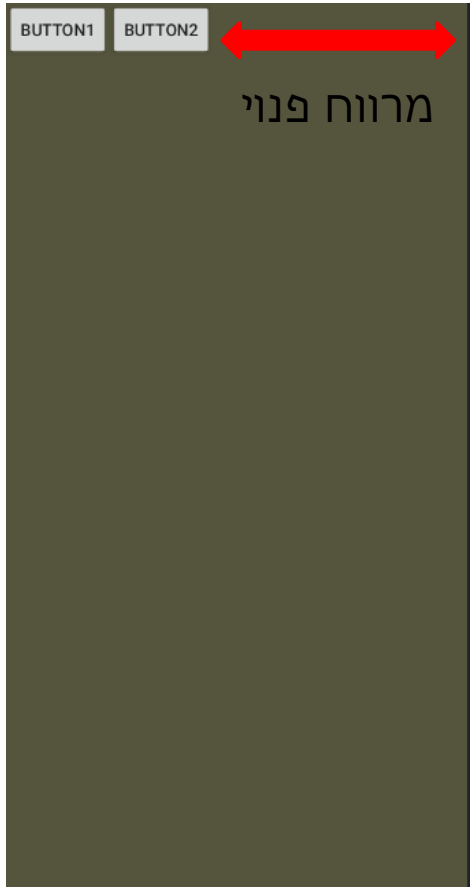
https://medium.com/@saranyaan2710/android-fill-parent-and-match-parent-with-example-71154425b5b3

http://www.differencebetween.info/difference-between-sp-and-dp-android

https://claude.ai/share/752dc405-bbd0-422d-bdff-f97da6a

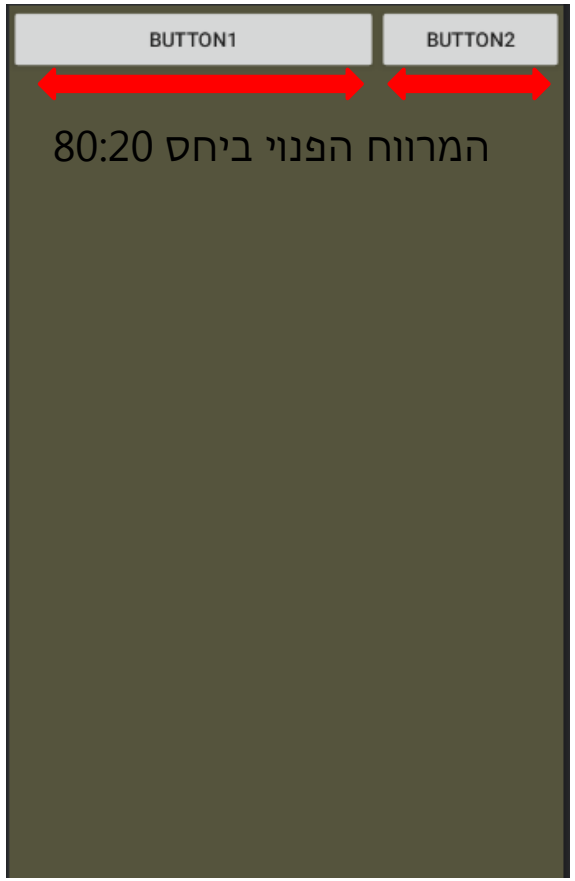
ברירת מחדל - layout_weight

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
<Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="button1"
    />

<Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="button2"
    />
</LinearLayout>
```



חלוקת מרווח פנוי - layout_weight

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="horizontal">
  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button1"
      android:layout_weight="80"
      />

  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button2"
      android:layout_weight="20"
      />
</LinearLayout>
```

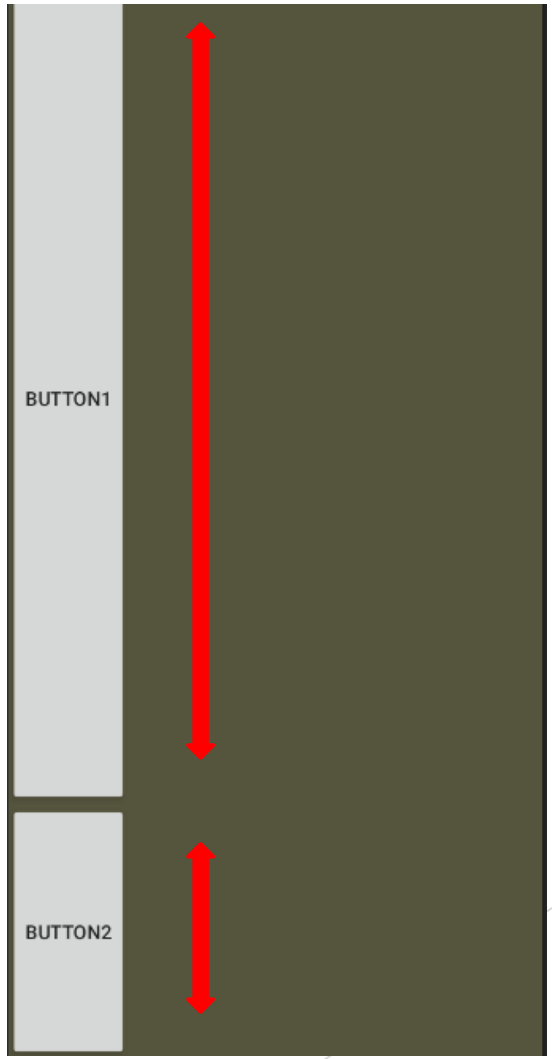


layout_weight - יחס חלוקה של ה layout

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    >
<Button
    android:layout_height="wrap_content"
    android:layout_width="0dp"
    android:text="button1"
    android:layout_weight="80"
    />

<Button
    android:layout_height="wrap_content"
    android:layout_width="0dp"
    android:text="button2"
    android:layout_weight="25"

    />
</LinearLayout>
```

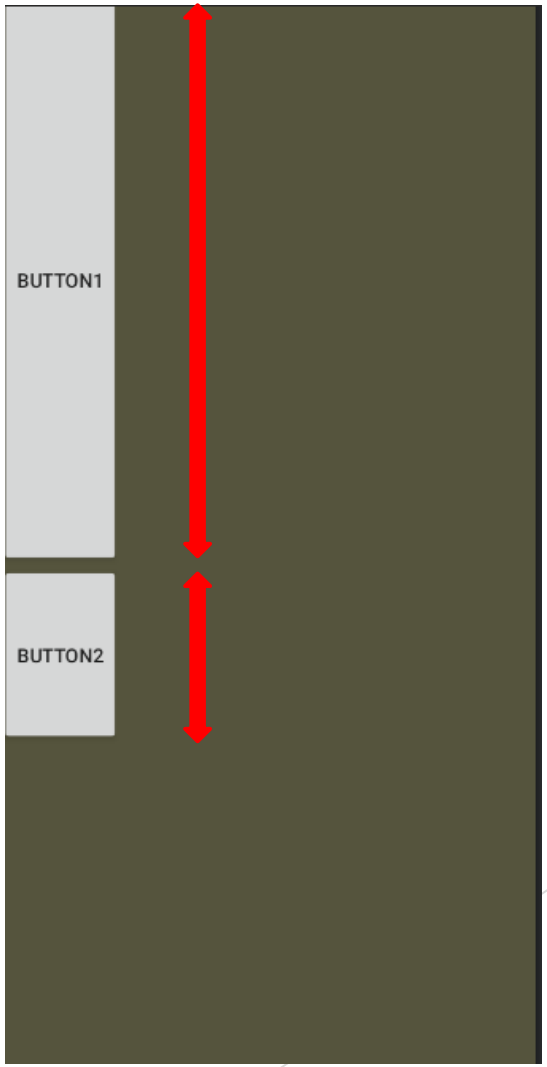
Vertical דוגמה ל - layout_weight

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="vertical"
      >
  <Button
      android:layout_height="0dp"
      android:layout_width="wrap_content"
      android:text="button1"
      android:layout_weight="80"
      />

  <Button
      android:layout_height="0dp"
      android:layout_width="wrap_content"
      android:text="button2"
      android:layout_weight="25"

      />
</LinearLayout>
```



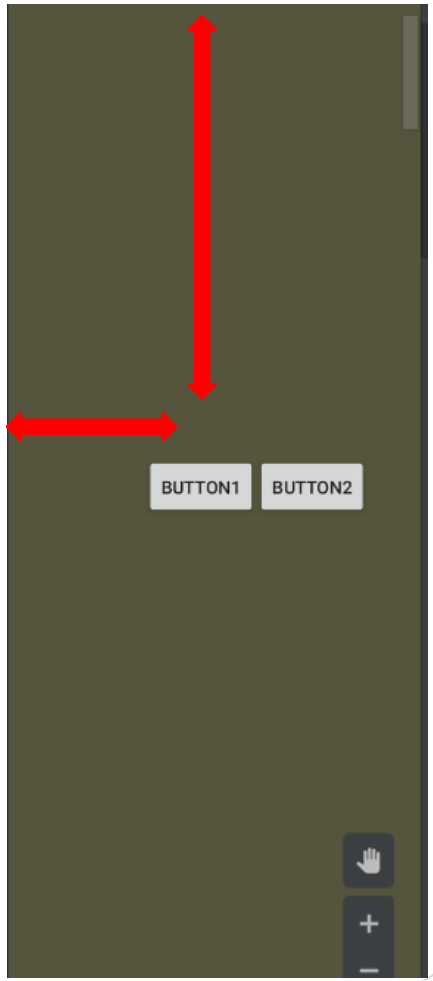
weight_sum שימוש ב - layout_weight

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="vertical“
      android:weightSum="150"
      >
  <Button
      android:layout_height="0dp"
      android:layout_width="wrap_content"
      android:text="button1"
      android:layout_weight="80"
      />

  <Button
      android:layout_height="0dp"
      android:layout_width="wrap_content"
      android:text="button2"
      android:layout_weight="25"

      />
```



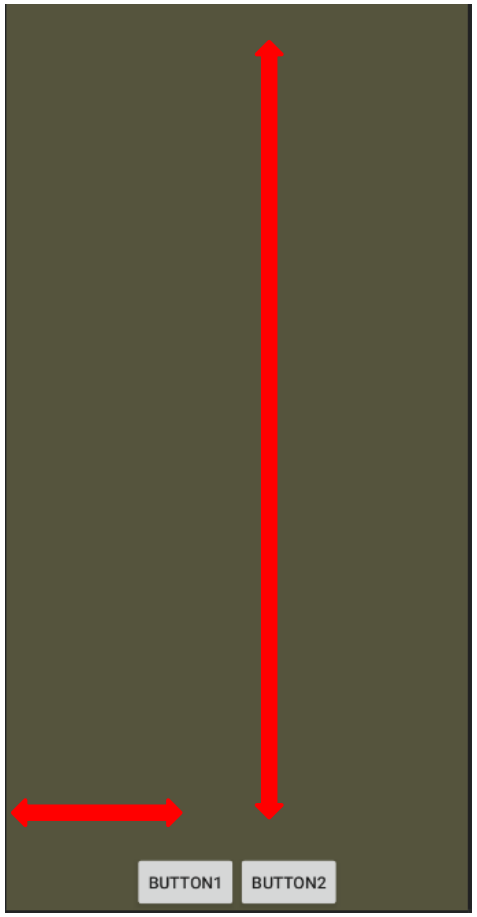
center -gravity

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="horizontal"
      android:gravity="center"              >
  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button1"
      />

  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button2"

      />
</LinearLayout>
```



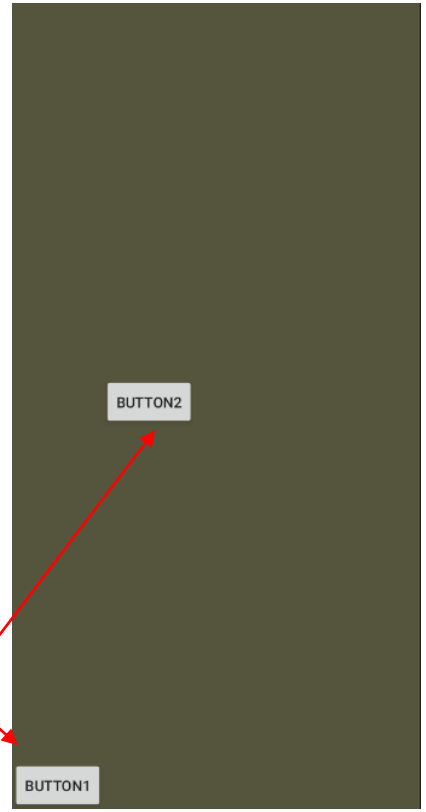
center|bottom-gravity

```
<LinearLayout
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:orientation="horizontal"
      android:gravity="center|bottom"
>
  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button1"
      />

  <Button
      android:layout_height="wrap_content"
      android:layout_width="wrap_content"
      android:text="button2"

      />
</LinearLayout>
```

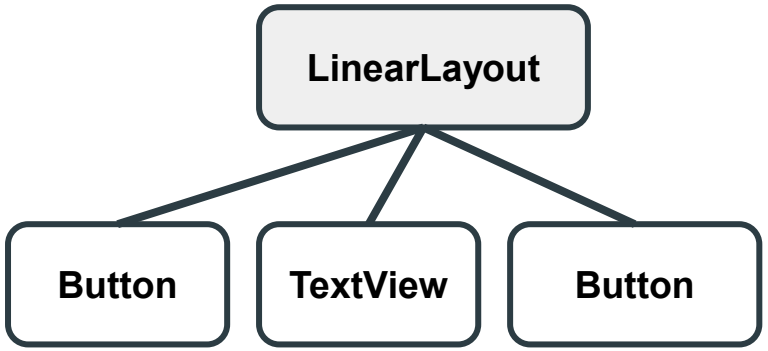


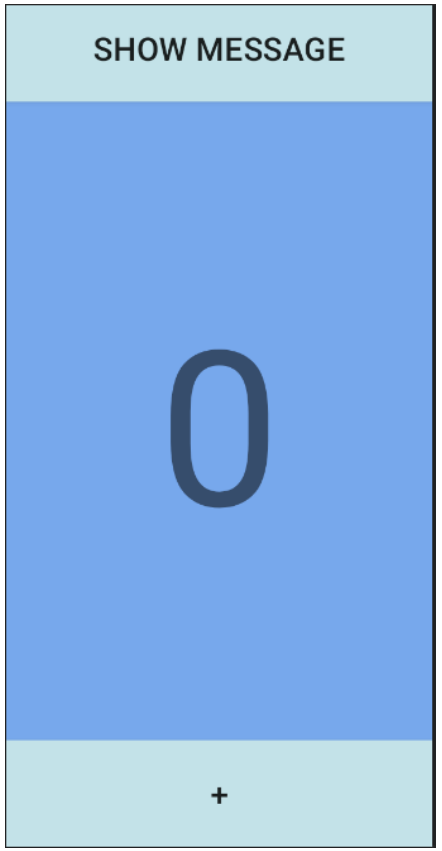
view layout_gravity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
 <Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="button1"
    android:layout_gravity="bottom"
    />

 <Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="button2"
    android:layout_gravity="center_vertical"
    />
</LinearLayout>
```



תרגיל Layout ראשון





- צרו LinearLayout שמכיל את ה Views הבאים:
- Button - כפתור עם טקסט "show"
- TextView - שמופיע בו המספר 0
- Button - כפתור עם טקסט "+"
- על המסך להראות באופן הבא:

מבוא - Constraint Layout

- Layout גמיש לארגון views
- הנפוץ ביותר בפיתוח לאנדרואיד
- מאפשר למקם רכיבים על ידי הגדרת יחסים (אילוצים - constraints) בין רכיבים
- יצירת ממשקי משתמש מורכבים במבנה שטוח

- https://developer.android.com/develop/ui/views/layout/constraint-layout

Basic Constraints

- אילוץ constraint - מגדיר כיצד רכיב ממוקם יחסית לרכיב אחר.

- כל view חייב להכיל לפחות 2 אילוצים - אופקי ואנכי
- הנפוץ ביותר בפיתוח לאנדרואיד
- מאפשר למקם רכיבים על ידי הגדרת יחסים (אילוצים - constraints) בין רכיבים
- יצירת ממשקים מורכבים במבנה שטוח

דוגמה כפתור במרכז המסך

```
<androidx.constraintlayout.widget.ConstraintLayout

   android:layout_width="match_parent"
   android:layout_height="match_parent"

      <Button
         android:id="@+id/centerButton"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="כפתור במרכז"
         app:layout_constraintTop_toTopOf="parent"
         app:layout_constraintBottom_toBottomOf="parent"
         app:layout_constraintStart_toStartOf="parent"
         app:layout_constraintEnd_toEndOf="parent" />


</androidx.constraintlayout.widget.ConstraintLayout>
```

הסבר:

app:layout_constraintTop_toTopOf="parent":
צד עליון של הכפתור מיושר לצד עליון של ההורה.

app:layout_constraintBottom_toBottomOf="parent":
צד תחתון של הכפתור מיושר לצד תחתון של ההורה.

app:layout_constraintStart_toStartOf="parent":
צד התחלה (שמאל) של הכפתור מיושר לצד התחלה של ההורה.

app:layout_constraintEnd_toEndOf="parent":
צד סיום (ימין) של הכפתור מיושר לצד סיום של ההורה.

כאשר רכיב מאולץ לשני כיוונים מנוגדים (לדוגמה, גם ל- top וגם ל- bottom), הוא מתמרכז ביניהם.

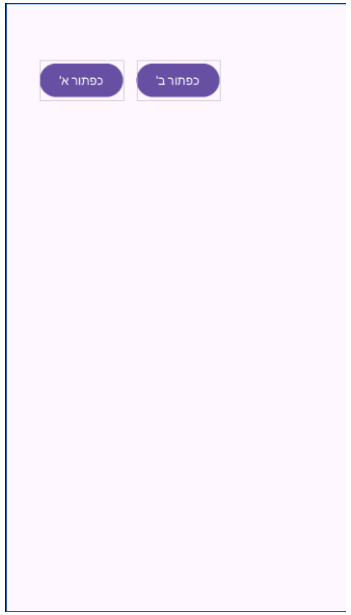
דוגמה 2 כפתורים מקושרים

```
<androidx.constraintlayout.widget.ConstraintLayout

   android:layout_width="match_parent"
   android:layout_height="match_parent"
<Button
   android:id="@+id/buttonA"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_marginStart="40dp"
   android:layout_marginTop="68dp"
   android:text="כפתור א'"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent" />

<Button
   android:id="@+id/buttonB"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="כפתור ב'"
   app:layout_constraintTop_toTopOf="@+id/buttonA"
   app:layout_constraintStart_toEndOf="@+id/buttonA"
   android:layout_marginStart="16dp"/>



</androidx.constraintlayout.widget.ConstraintLayout>
```

app:layout_constraintTop_toTopOf="@+id/buttonA":
החלק העליון של "כפתור ב'" מיושר לחלק העליון של "כפתור א'".

app:layout_constraintStart_toEndOf="@+id/buttonA":
הצד השמאלי של "כפתור ב'" מיושר לצד הימני של "כפתור א'"

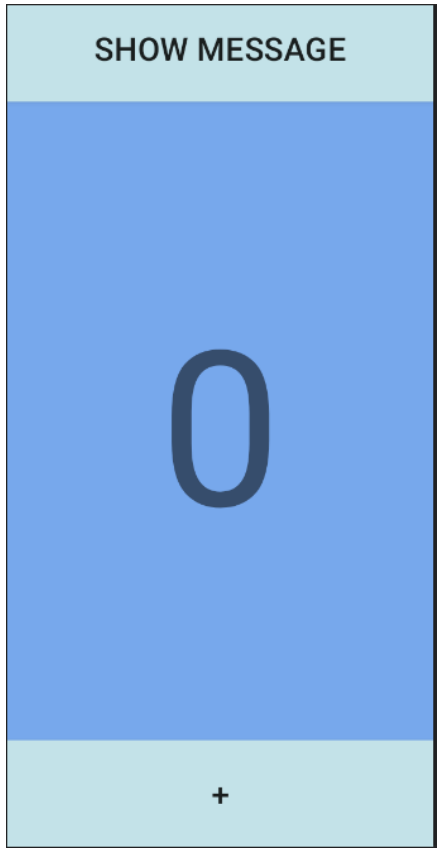
android:layout_marginStart="16dp":
מוסיף מרווח של 16dp בין הרכיבים.



Constraint Layout- מימדים

- Wrap Content: הרכיב מתאים עצמו לגודל התוכן שבו
- Match_parent: הרכיב תופס את כל הרוחב/גובה של ה parent layout
- 0dp (match_constraint) - כאשר יש אילוצים לשני כיוונים (למשל לימין ולשמאל) - ימתח עצמו על מנת למלא את המרווח שבין האילוצים.

תרגיל Constraint Layout



- צרו ConstraintLayout שמכיל את ה Views הבאים:
- Button - כפתור עם טקסט "show"
- TextView - שמופיע בו המספר 0
- Button - כפתור עם טקסט "+"
- על המסך להראות באופן הבא
- שימו לב להשתמש באילוצים מתאימים
- לא לשכוח כל רכיב חייב לפחות שני אילוצים (אופקי ואנכי)