פיתוח אפליקציות לאנדרואיד בסביבת Android Studio

למה פיתחתי ספר זה

החזון הוא לאפשר לכל אדם ואדם לשנות את חייו באמצעות השכלה פרקטית גם בחינם ללא עלות.

איך מוציאים את החזון לפועל:
באמצעות פלטפורמה אינטרנטית המחברת בין מורים לתלמידים. הטכנולוגיה מנגישה את הידע והחינוך לכל אדם, בכל זמן ובכל מקום.

הקדשה ספר זה מוקדש לנעמה שרת עמיר, אור עמיר, יהל עמיר.3 בנות שסובלות אותי במהלך חיי.תודה על התמיכה!

בנוסף אני רוצה להודות ולהקדיש ספר זה, גם לכל המורים והמדריכים לפיתוח אפליקציות במדינת ישראל, שמלווים את ילדינו בחטיבות, בתיכונים ובאוניברסיטאות.יצא לי לעבוד עם מדריכים רבים, מדובר באנשים שעושים עבודת קודש, תורמים הרבה מעבר למצופה מהם ומשקיעים את זמנם בפיתוח היכולות של הדור הבא!
תודה רבה לכל המדריכים והמורים במדינת ישראל, שמשקיעים מזמנם ולא ישנים לילות ועסוקים במציאת באגים של סטודנטים להנדסת תוכנה.

תודה לארד פאר, על עריכת והגהת הסרטונים.ארד בקרוב תתגייס ל 8200 ואני בטוח שתתרום רבות למדינת ישראל בכל תחום שתעסוק בו.

תודה לריזורט אנגל על עריכת כריכה לספר

מה אפשר לעשות עם ידע במערכת הפעלה אנדרואיד?

לפני שנים רבות פגשתי מורה מעולה למתמטיקה.היא סיפרה לי שלפני שהיא מלמדת נושא מסוים היא קודם כל מספרת שיעור שלם לתלמידים מה ניתן לעשות עם אותו נושא שהיא מלמדת.
היו לה דפים שלמים שהיא מסבירה מתה ניתן לעשות עם סדרה חשבונית, טריגונומטריה וכו.היא תמיד התרכזה בלמה זה חשוב ואחרי זה באיך עושים את זה.
אז אני אתחיל במה אפשר לעשות עם ידע במערכת הפעלה אנדרואיד.

- בניית אפליקציות
- בניית רובוטים
- בניית מוצרים לבישים
- תקשורת עם רכיבי i.o.t
- סדר במידע
- משחקים
- בניית חדרי בריחה
- פתיחת sockets
- פתיחה תקשורת bluetooth
- פתיחת תקשורת wifi
- תקשורת עם שפות שונות כגון פייתון, nodejs ,#c
- ועוד .....

לסיכום : אפשר לעשות המון עם מערכת הפעלה אנדרואיד

אני מאחל לכם

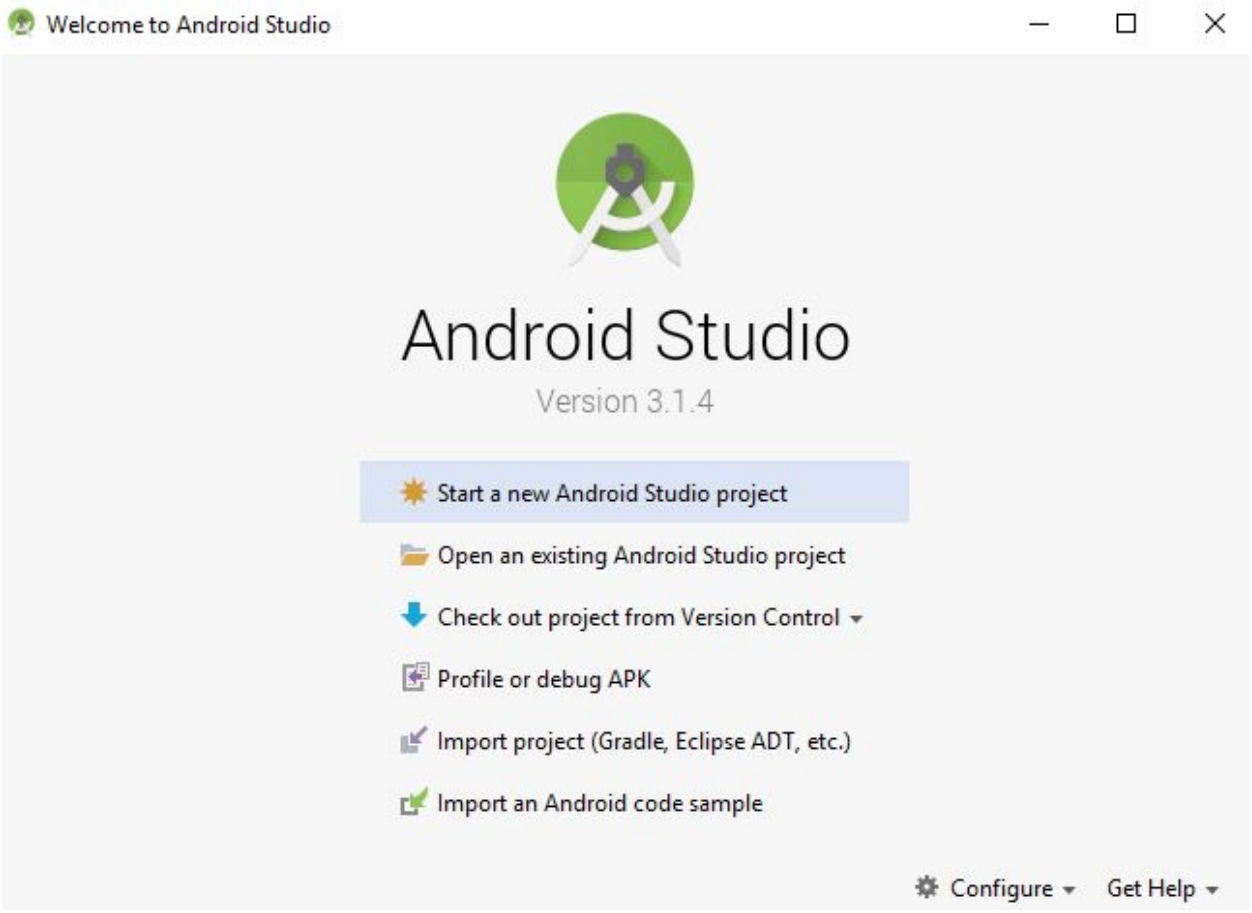
אני מאחל לכם המון הצלחה בלימודים בספר זה.לאחר שתעברו על כל הפרקים אני ממליץ לכם לבנות מספר אפליקציות, לעלות אותם לחנות האפליקציות ולרכוש ניסיון ראשוני בעולם פיתוח האפליקציות.לאחר ספר זה יהיו ספרים נוספים רבים שתוכלו לקרוא וללמוד .....

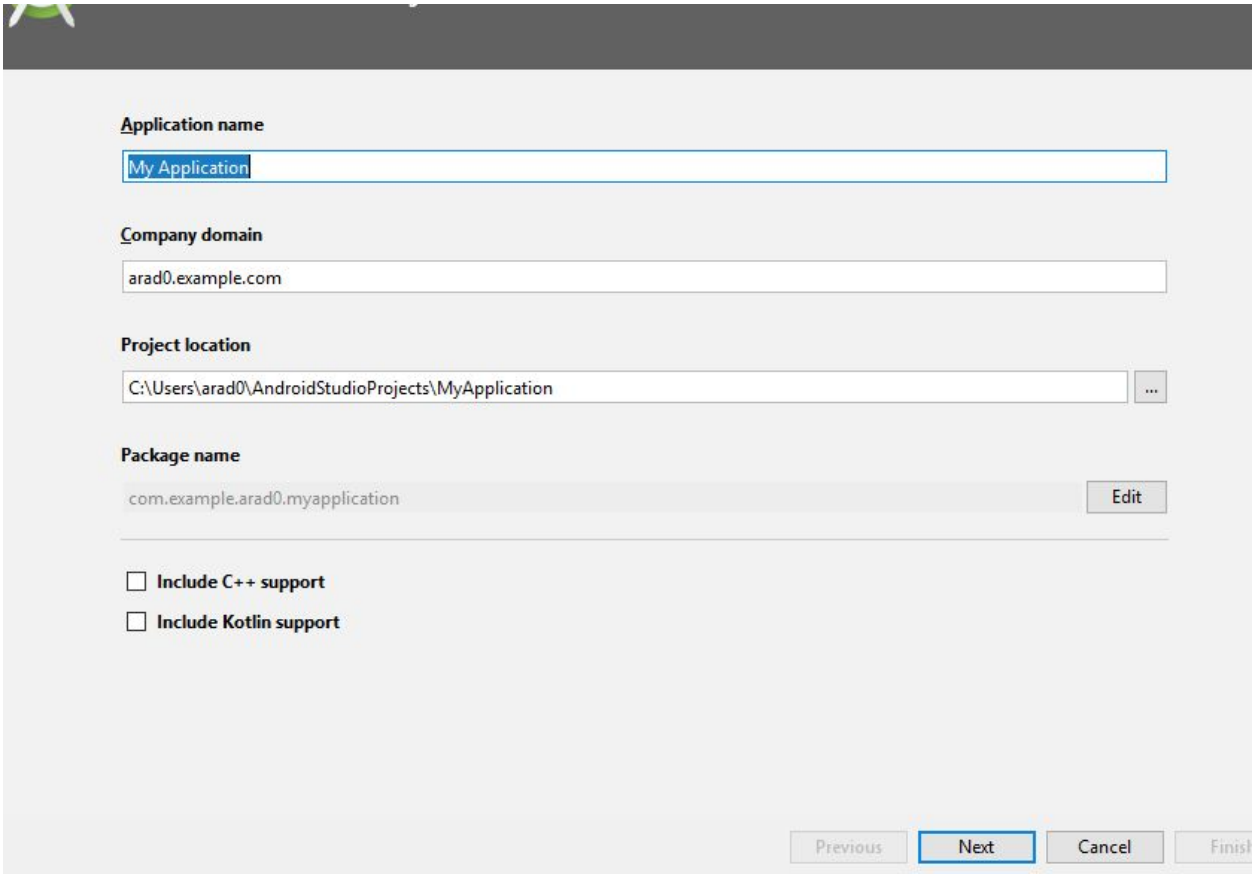
בהצלחה אסף!

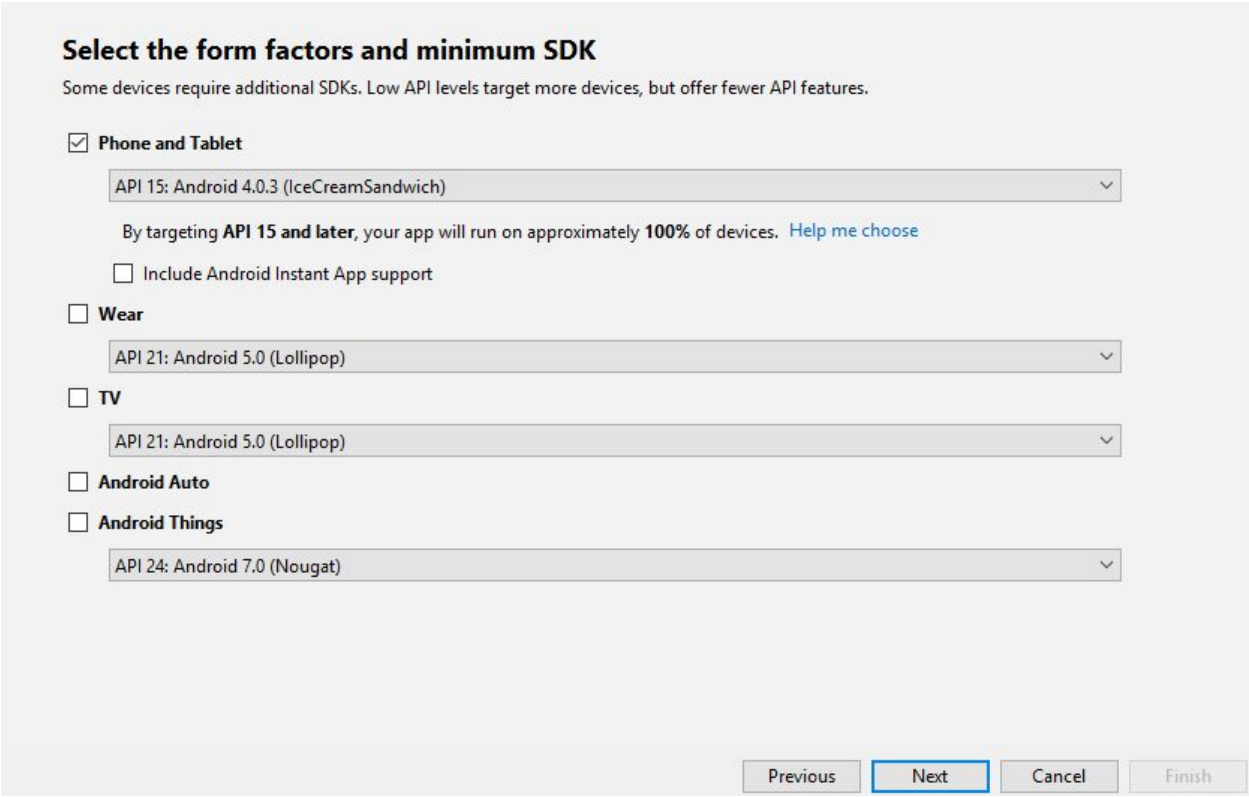
תוכן עניינים

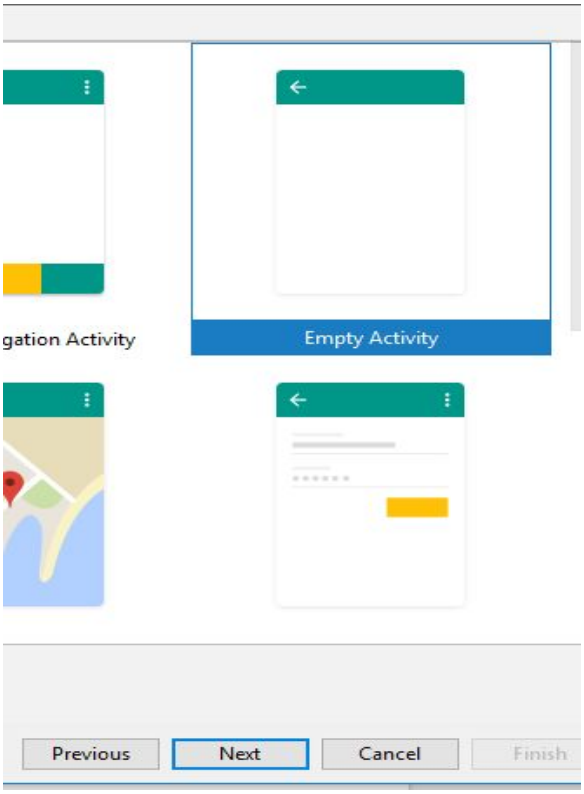
פתיחת תכנית ראשונה ב Android Studio :

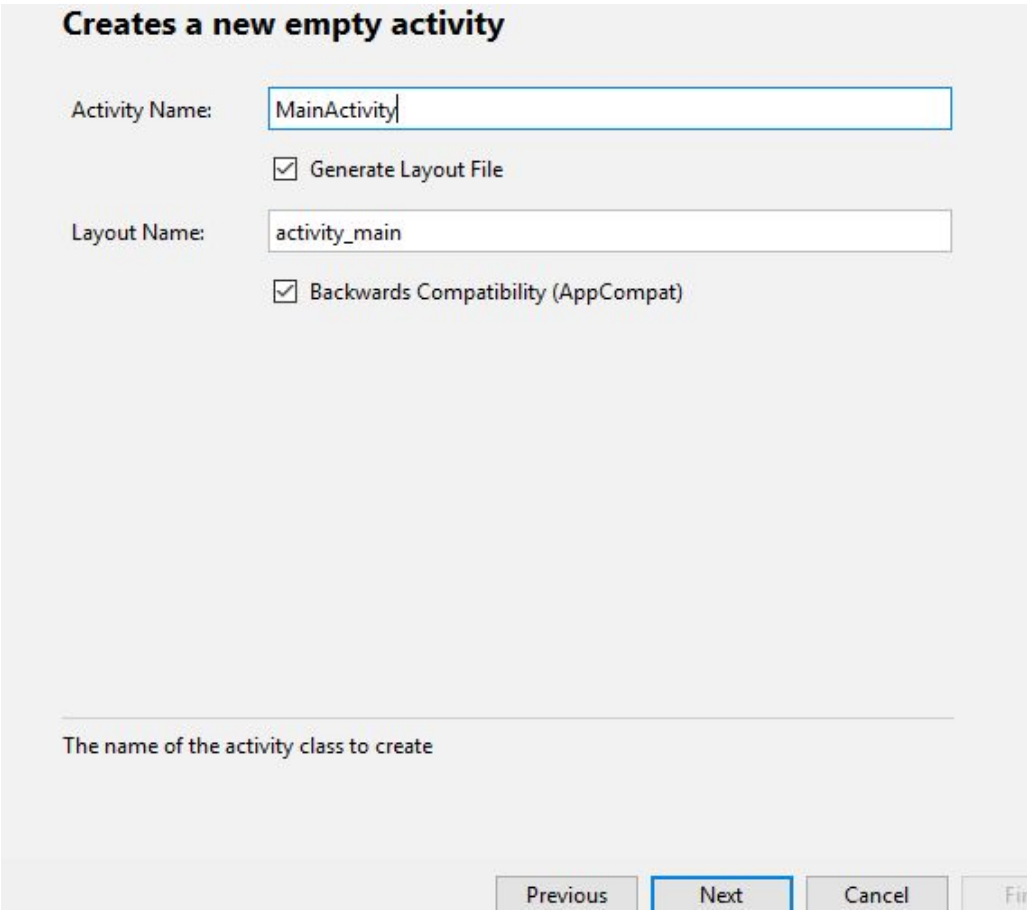
ראשית, נלמד איך לפתוח תוכנית חדשה לאחר ההתקנה הראשונית של Android Studio:





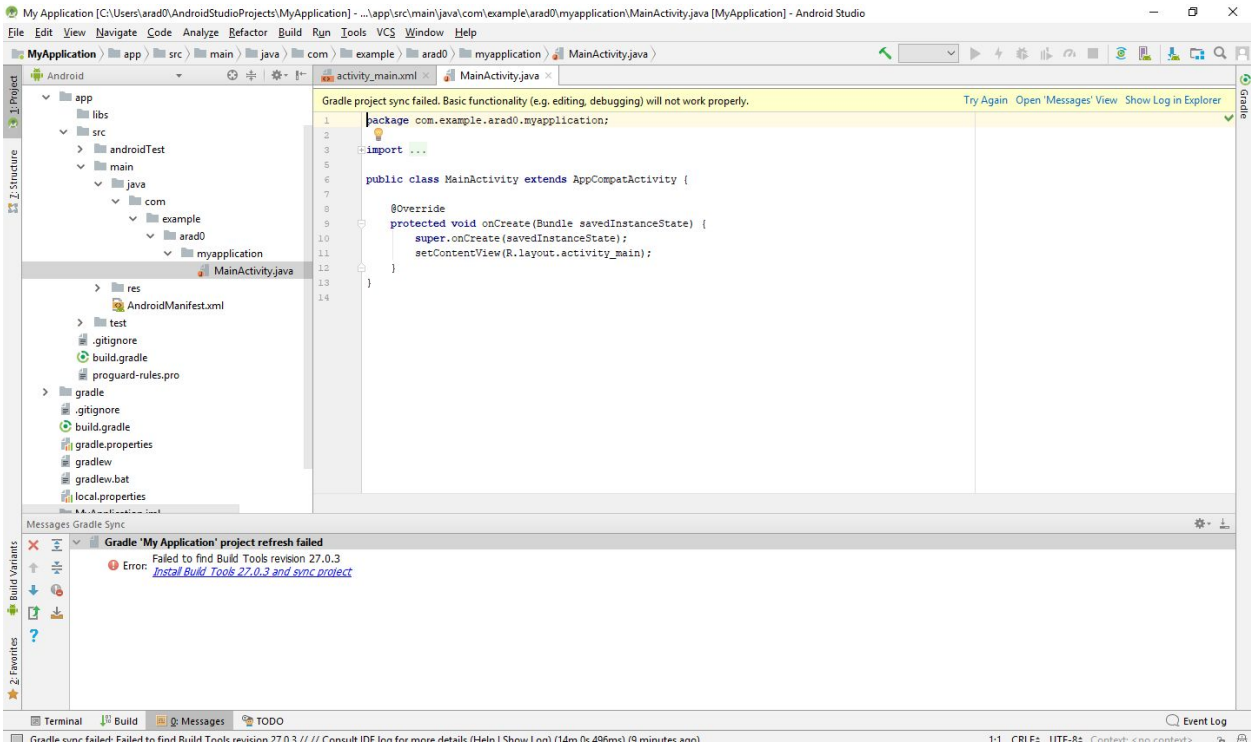






מבנה תכנית באנדרואיד סטודיו:

כל תכנית שנבנה באנדרואיד סטודיו תיפתח בצורה כזו:



כך יראה המסך כאשר נפתח תוכנית חדשה.

למטה במסך ניתן לראות את הoutput של התוכנית. במקרה של באג בקוד ניתן לבדוק בעזרת תיבה זו היכן כשל הקוד וקרסה התוכנית.

במידה ומופיע לכם קוד אדום עם הודעה על ה SDK מיד כשתפתחו את התוכנית הראשונה, ,תלחצו על ההודעה ותורידו את ה SDK.

מצד שמאל ניתן לראות את הקובץ הפתוח.

מצד ימין ניתן לראות את הספריה של התוכנית שלנו.

בכל תיקיה נמצאים קבצים שונים מקטגוריה מסוימת.

לדוגמה, תיקיית ה values מכילה קבועים שבונים את האפליקציה. כמו צבע האפליקציה, אייקון, string'ים ושם האפליקציה.

ניתן לראות שבתחילת הקוד ממומשים הקבועים שבתיקיית values, כגון שם האפליקציה ו style.

נוכל לראות שימוש בתיקיה זו בתחילת כל activity שנפתח.

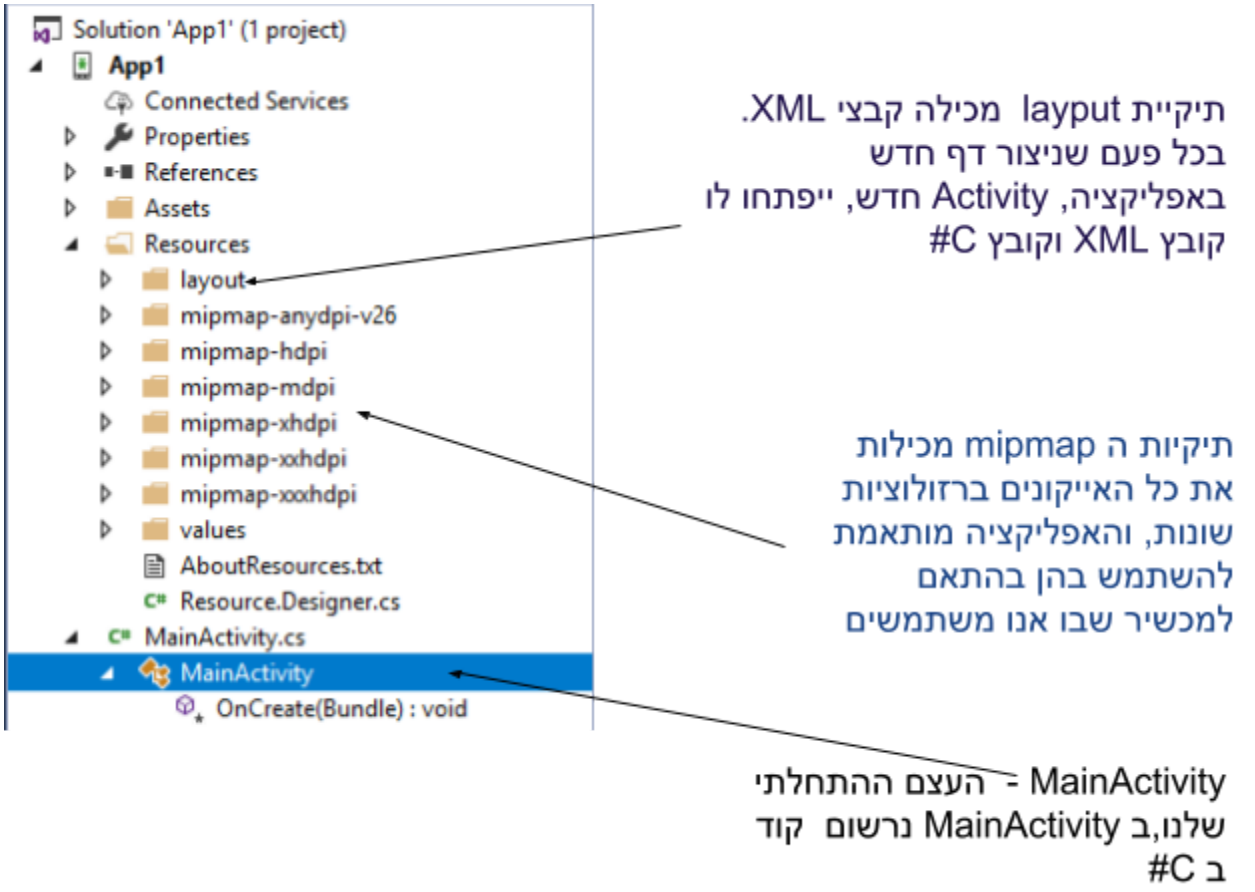
Activity הוא דף לאפליקציה שלנו. בכל פעם שאנו מגיעים למסך חדש באפליקציה, אנו מגיעים בעצם ל Activity אחר. כל Activity בנוי מקובץ JAVA וקובץ XML שקשורים אחד לשני.

קבצי ה XML מופיעים תחת תיקיית layout שבתיקיית res ובהם נערוך את הגרפיקה של Activity.

קבצי ה JAVA נקראים גם Class ובהם נצהיר על משתנים וקבועים ובהם נערוך את הקוד שמאחורי התוכנית.

כל class של Activity יורש מ class שנקרא AppCompatActivity.

הפונקציה הראשונה שתיקרא בפתיחת כל Activity היא onCreate, שם מוגדר בעזרת פונקציה שנקראת setContentView קובץ ה XML שיהיה ה layout של ה Activity.

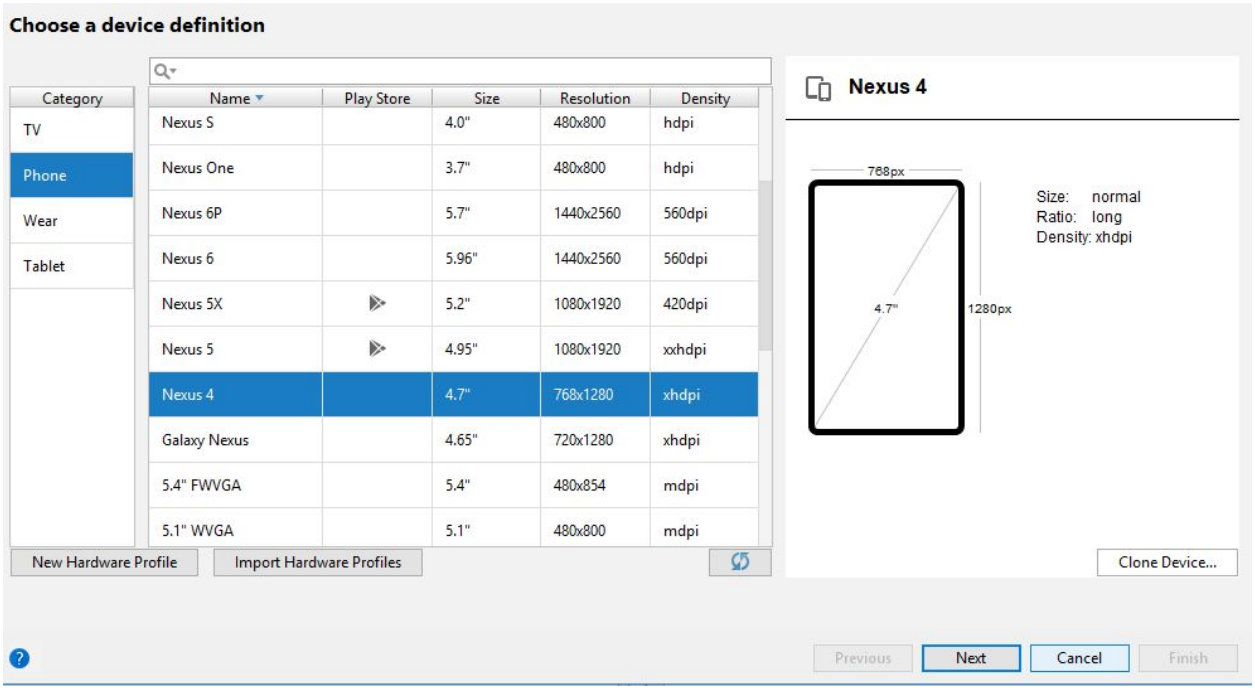


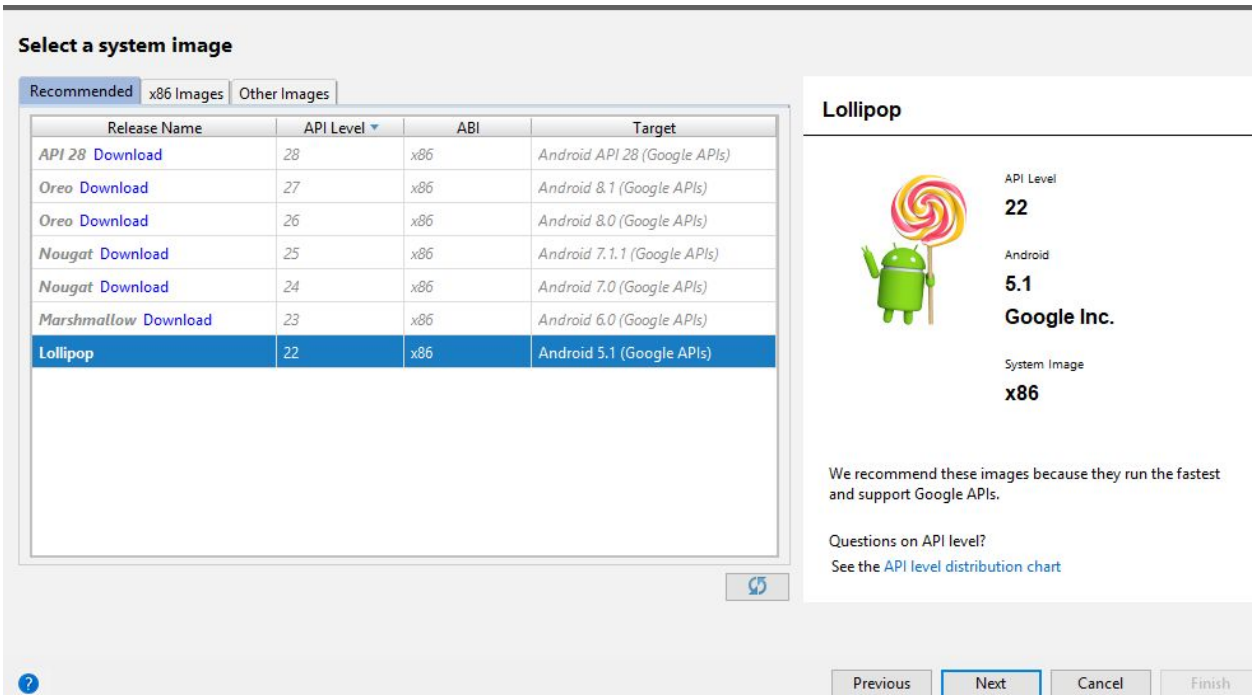
איך מריצים את האפליקציה?

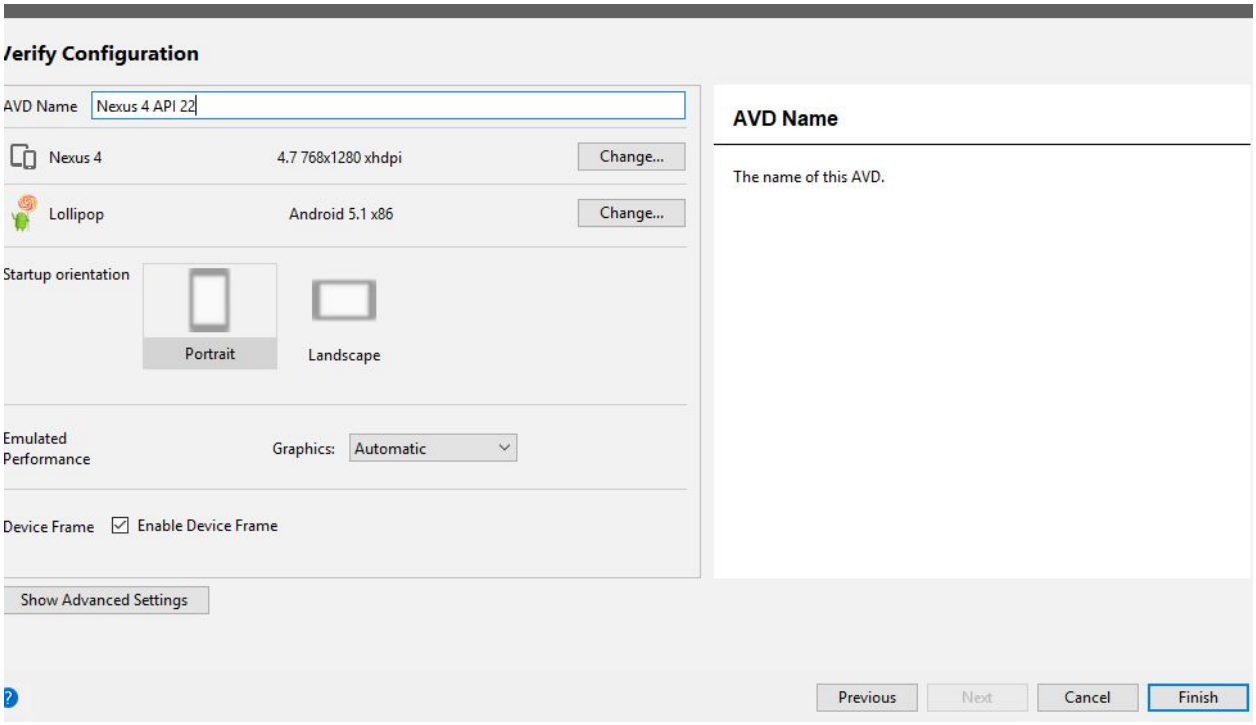
בשביל להריץ את האפליקציה נשתמש באחת משתי הדרכים הבאות - בעזרת אימולטור או בעזרת הטלפון שלנו.

נוכל לחבר את הטלפון למחשב ולהריץ את האפליקציה עליו (יותר מהיר, עדיף למחשבים איטיים), או שנוכל להריץ על אימולטור במחשב שלנו.
אימולטור הוא בעצם טלפון בעל מערכת הפעלה של אנדרואיד שישמש כטלפון הניסוי שלנו בו נוכל להריץ את האפליקציה שבנינו.

בשביל להריץ את האפליקציה על האימולטור נוריד קודם את כל המשאבים שידרשו לנו: (קיים אימולטור שמגיע עם ההתקנה של AndroidStudio, אך הוא חזק ולא ירוץ על הרבה מחשבים)

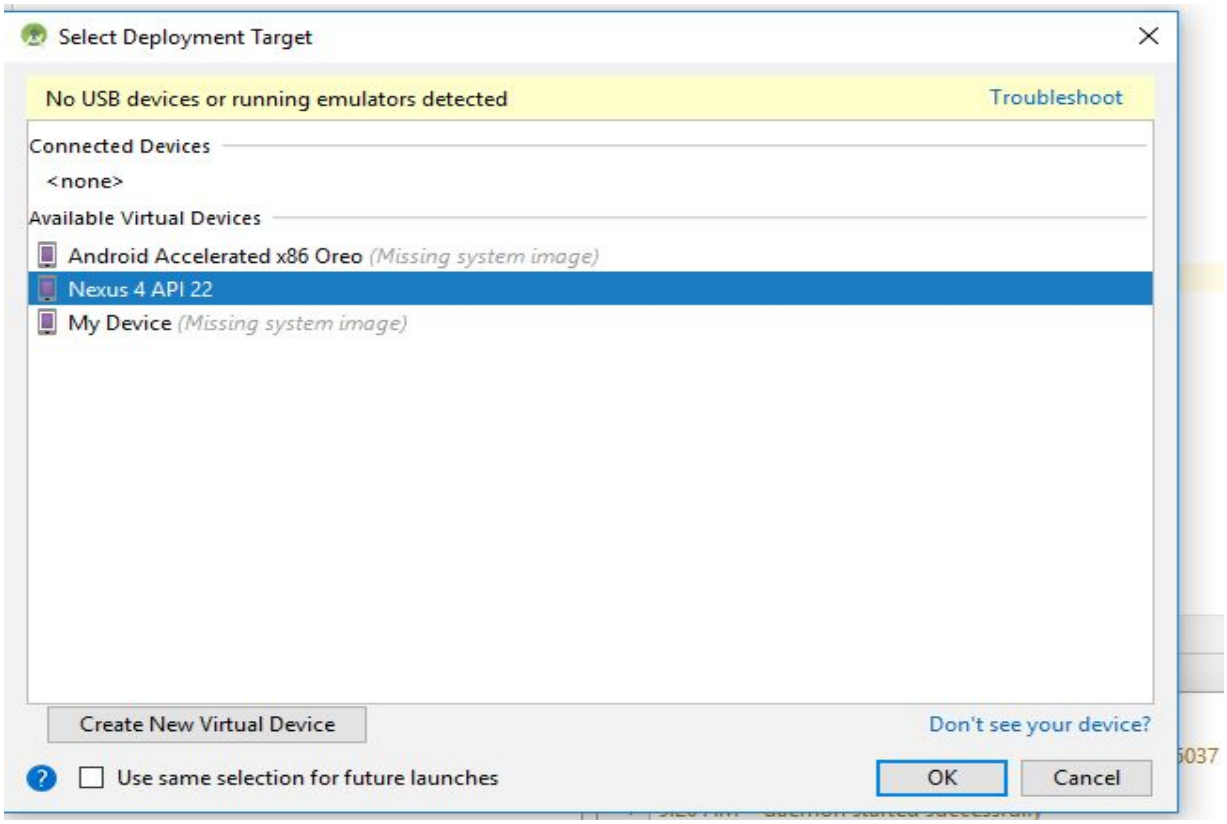






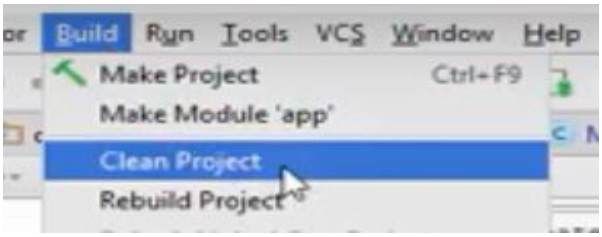
כעת נראה שהאימולטור שלנו נשמר ונטען (יכול לקחת כמה דקות, תלוי בחוזק מחשב), לאחר מכן נלחץ על start.

בפעם הראשונה שנריץ את האימולטור התהליך יקח כמה דקות מרגע שנלחץ start.



כשאנחנו רואים ב Output (החלון מצד שמאל למטה במסך) שיש לנו שגיאה בתוכנית, לעיתים זה קורה בגלל ששינינו שמות ו id (נלמד בהמשך) בכל מיני מקומות באפליקציה והתוכנית לא הספיקה לעדכן זאת, או שאין סנכרון בין קובץ ה XML לקובץ ה JAVA בגלל שינוי שמות, וכו'.

בשביל לטפל בזה כדאי לשמור את קובץ ה XML לאחר שאנו מסיימים לעדכן אותו, ואם בכל מקרה יש שגיאה, נוכל לנקות את האפליקציה בדרך הבאה: (ניתן ללחוץ על clean ואם עדיין לא עובד - גם rebuild, או להפך)



בדרך זו אנחנו מנקים את הקוד שבנינו לפני כן וכבר לא משומש.

הרצת האפליקציה על הסמארטפון שלכם:

תוכלו להריץ את האפליקציה שתבנו גם על הטלפון שלכם. אפשרות זו עדיפה מאשר הרצה על האימולטור כי היא יותר מהירה ודורשת פחות כוח מהמחשב (מומלץ בחום להריץ על הטלפון למי שיש מחשב לא כל כך חזק).

בשביל להריץ על הטלפון:

1. ודאו שהטלפון מחובר למחשב עם כבל USB (שימו לב שיש כבלי USB שמאפשרים רק טעינה)
2. פתחו את ההגדרות בטלפון -> מידע על הטלפון
3. תלחצו על build number 7 פעמים

4. תחזרו למסך הקודם
5. תוכלו לראות שנוספה לכם עכשיו אפשרות ללחוץ על 'אפשרויות מפתח'
6. כנסו ל 'אפשרויות מפתח'
7. הפעילו את usb debbuging
8. הפעם, כשתריצו את האפליקציה, תבחרו במכשיר שלכם במקום באימולטור שיצרנו לפני כן.

פרק 1 - תכנות בסיסי באנדרואיד

בפרק זה נבנה את האפליקציה הראשונה שלנו. נבנה אפליקציה שמעלה ומורידה ניקוד באמצעות לחיצה על כפתורים והצגת הניקוד על TextView

בניית אפליקציה באנדרואיד סטודיו תתחלק ל 4 שלבים :

שלב 1 : עיצוב layout

שלב 2 : הצהרה על תכונות ב JAVA

שלב 3 : קבלת הפניה לתכונות שב XML

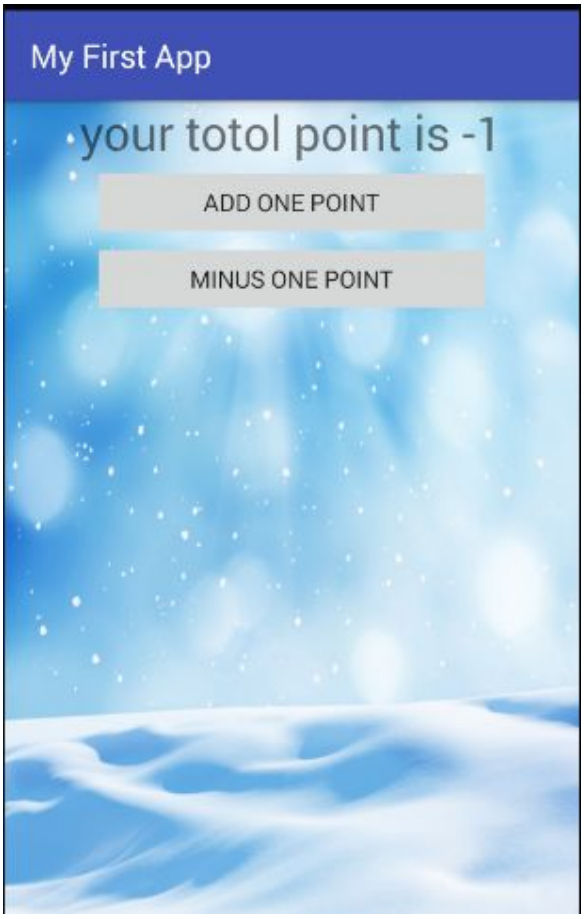
שלב 4 : מימוש events (אובייקטים כגון כפתורים וטקסטים)

שלב 1 - עיצוב layout

השלב הראשון של האפליקציה שלנו הוא החלק הגרפי.

בשלב זה נלמד איך לעצב את האפליקציה שלנו בדרך שנרצה.

נרצה שהאפליקציה שלנו בפרק זה תראה כך:



נעשה זאת בעזרת לחיצה בספריית התיקיות על
res -> layout -> activity_main.axml.
אנו נלמד לעבוד בעזרת סידור מסך שנקרא LinearLayout, לכן שנו את המילה
RelativeLayout ל LinearLayout. מה שמייחד את צורת סידור LinearLayout הוא
שכאשר נשים בה אובייקטים, הם יופיעו על המסך אחד אחרי השני, מלמעלה למטה או
משמאל לימין - בצורה שאנו נקבע.
סידור משמאל לימין נקרא horizontal.
אנו נשתמש לבניית האפליקציה שלנו בסידור מלמעלה למטה שנקרא vertical.
כך יראה הקוד שלנו לאחר החלפת המילה RelativeLayout במילה LinearLayout:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="fill_parent"
```

```
android:layout_height="fill_parent"
android:orientation="vertical"
```

בהמשך נראה איך להוסיף תמונה מהמחשב בתור הרקע, היא תופיע בשורה הבאה:

```
android:background="@drawable/back1"
```

בכל פעם שנוסיף אובייקט למסך נוסיף אותו **לפני** השורה </LinearLayout> .

לדוגמה, במידה ונרצה להוסיף טקסט למסך נרשום <TextView ונעצב אותו.

נוכל להוסיף לאובייקט שלנו תכונות בעזרת כתיבת המילה android ולחיצה על CTRL + SPACE. אז יופיעו לנו רשימת התכונות של האובייקט.

כשמוסיפים תכונות לאובייקט חשוב לשים לב שאנו כותבים זאת **לפני** ה /> שסוגר את האובייקט.

```
<TextView
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="you have 0 points"
   android:textSize="30dp"
   android:layout_gravity="center"
   android:gravity="center"
   android:id="@+id/txtdisplay"
   />
```

ניצור גם שני כפתורים:

```
<Button
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:text="add one point"
    android:textSize="15dp"
    android:layout_gravity="center"
    android:id="@+id/btnplus"
    />
<Button
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:text="minus one point"
    android:textSize="15dp"
    android:layout_gravity="center"
    android:id="@+id/btnminus"
    />
</LinearLayout>
```

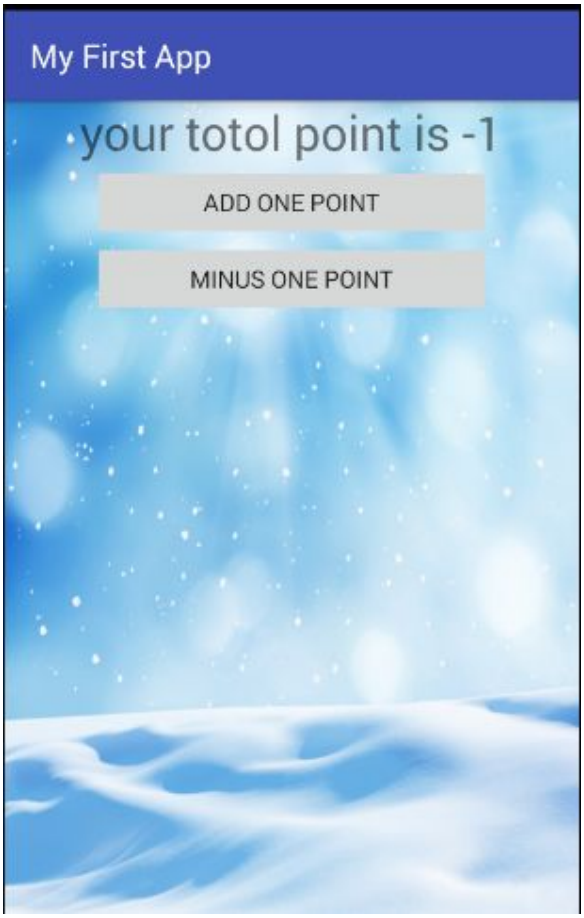
הוספת תמונה לאפליקציה

בשביל להעלות תמונה לאפליקציה נעשה העתק לתמונה מהמחשב. לאחר מכן נלחץ קליק ימני על התיקייה drawable שתחת res והדבק.

בשביל להשתמש בתמונה נרשום "android:background = "@drawable/picName בתוך האובייקט שאת הרקע שלנו נרצה לשנות. (בXML)

סיימנו את שלב מספר 1.

כך נראית האפליקציה שלנו בינתיים, ללא קוד שעובד מאחורי הקלעים.



שלב 2 - הצהרה על תכונות בקובץ ה(class) JAVA:

ניצור אובייקטים ב JAVA בהתאם לאובייקטים שיצרנו ב layout, ולאחר מכן ניתן להם הפניה לאובייקטים שיצרנו ב XML (רפרנס) בשביל שנוכל להשתלט עליהם.

הכרזה על אובייקטים בקובץ MainActivity, על מנת לקבל הפניה לאובייקטים בlayout

```
public class MainActivity extends AppCompatActivity {
 Button add;
 Button minus;
 TextView display;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

 }
```

הצהרה על המשתנים שבהם נשתמש בהמשך. (במקרה שלנו האובייקטים)

אם אחד האובייקטים מופיע בצבע אדום, תעמדו עליו עם סמן העכבר, וליחצו alt + space ואז תבחרו את האפשרות הראשונה (enter)

שלב 3 - קבלת הפניה לתכונות שב XML

בשלב זה נקשר את האובייקטים שיצרנו בשלב 2 בקובץ ה JAVA עם האובייקטים שיצרנו בשלב 1 ב XML

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
 Button add;
 Button minus;
 TextView display;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

        add=(Button)findViewByld(R.id.btnplus);
        minus=(Button)findViewByld(R.id.btnminus);
        display=(TextView)findViewByld(R.id.txtdisplay);
 }
```

קבלת הפניה מההצהרות ב JAVA לאובייקטים שיצרנו ב XML

שלב 4 - האזנה ללחיצה על כפתור:

בשלב זה נראה איך לממש את הפעולות שנרצה לבצע על הכפתורים בעזרת האזנה.
האזנה לכפתור מאפשרת לנו לשלוט מה יקרה לאחר שנלחץ על הכפתור..
ראשית, מתחת לשורות שהבאנו לכפתורים בOnCreate נרשום לכפתורים אפשרות האזנה:

נרשום את השורות:

```
add=(Button)findViewById(R.id.btnplus);
minus=(Button)findViewById(R.id.btnminus);
display=(TextView)findViewById(R.id.txtdisplay);
```

```
add.setOnClickListener(this);
minus.setOnClickListener(this);
}
```

נותנים לכפתורים אפשרות האזנה

This אומר שהפונקציה שמאזינה ללחיצה על כפתור תהיה ב class שאנחנו נמצאים בו כרגע.
לכן נצטרך לממש את הממשק View.OnClickListener
ואז את הפונקציה onClick:

תעמדו עם הסמן של העכבר על this ואז תלחצו על הנורה האדומה בצד שמאל בשביל לממש את הממשק:

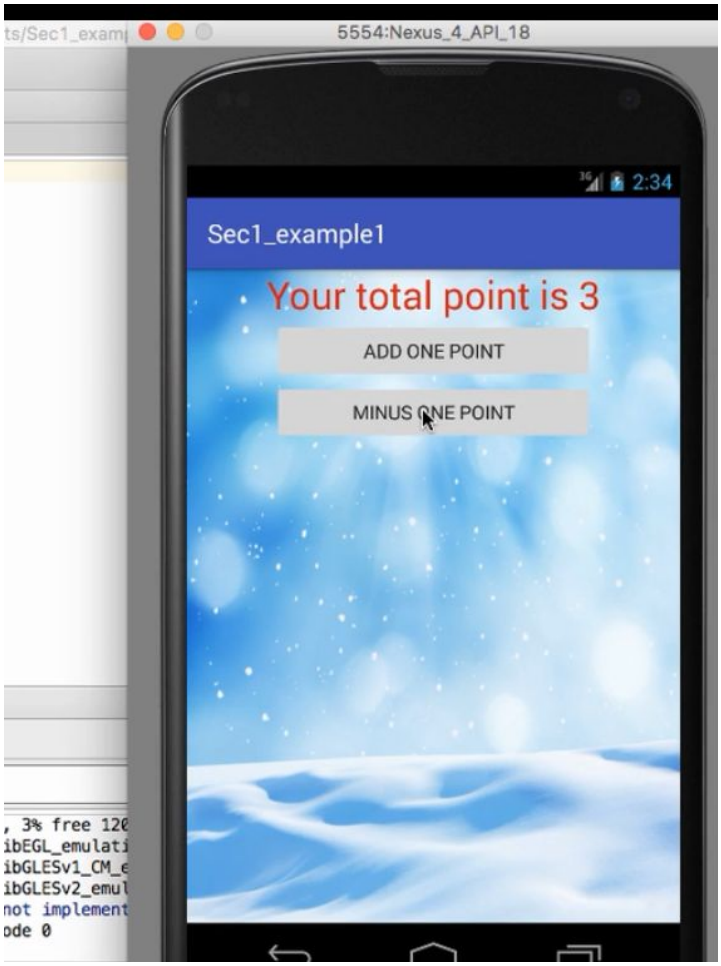
```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    @Override
    public void onClick(View v) {


        if(v==minus)
        {
            point--;
            display.setText("your totol point is " + point);
        }
        if(v==add)
        {
            point++;
            display.setText("your totol point is " + point);
        }
    }
```

הפונקציה onClick תמומש מתחת ל onCreate

במקרה שלנו נגדיר שבכל פעם שילחצו על הכפתורים יגדל/ יקטן ה counter בהתאם, ויעודכן הTextView בהתאם.

הריצו את האפליקציה :)

זהו זה, נריץ עכשיו את האפליקציה.



מצורפות כל שורות הקוד:

JAVA:

```
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
  Button add;
  Button minus;
  TextView display;
```

```
  @Override
  protected void onCreate(Bundle savedInstanceState)  implements
View.OnClickListener
{
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);

add=(Button)findViewById(R.id.btnplus);
minus=(Button)findViewById(R.id.btnminus);
display=(TextView)findViewById(R.id.txtdisplay);
 add.setOnClickListener(this);
 minus.setOnClickListener(this);
 }


  @Override
  public void onClick(View v) {
     if(v==minus)
     {
        point--;
        display.setText("your totol point is " + point);
     }
     if(v==add)
     {
        point++;
        display.setText("your totol point is " + point);

     }
```

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:orientation="vertical"
  android:background="@drawable/back1"
  >
  <TextView
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:text="you have 0 points"
     android:textSize="30dp"
     android:layout_gravity="center"
     android:gravity="center"
     android:id="@+id/txtdisplay"
     />
  <Button
     android:layout_width="250dp"
     android:layout_height="wrap_content"
     android:text="add one point"
     android:textSize="15dp"
     android:layout_gravity="center"
     android:id="@+id/btnplus"
     />
```

```
 <Button
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:text="minus one point"
    android:textSize="15dp"
    android:layout_gravity="center"
    android:id="@+id/btnminus"
    />
</LinearLayout>
```

דוגמה 2:

האפליקציה השנייה שנבנה בפרק זה תקלוט את השם שנרשום לה, ותציג אותו על המסך עם טקסט.

נעבוד לפי השלבים שלמדנו בדוגמה הראשונה:

שלב 1 : עיצוב layout

שלב 2 : הצהרה על תכונות ב JAVA

שלב 3 : קבלת הפניה לתכונות שב XML

שלב 4 : מימוש events (אובייקטים כגון כפתורים וטקסטים)

שלב 1 - עיצוב layout

האפליקציה שנבנה בדוגמה זו תראה כך:

ראשית נפתח את קובץ ה XML ונחליף את הRelativeLayout ל LinearLayout.

נוסיף גם vertical orientation.

לאחר מכן נוסיף אובייקט בשם EditText למסך.

ה EditText הוא אובייקט שהמשתמש יכול לרשום בו פרטים.

אנו נבנה את ה EditText שלנו כך:

```
<EditText
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:id="@+id/etFname"
    android:hint="First Name"
    android:layout_gravity="center"
    android:textSize="30sp"
    android:background="@android:drawable/editbox_background_normal"
```

כעת ניצור עוד EditText שבו המשתמש ירשום את השם משפחה.
ניצור גם button שעליו ילחץ המשתמש כדי שהשם שרשם יופיע על המסך.
לבסוף ניצור TextView שיציג על המסך את השם שיכתוב המשתמש.

שימו לב! סדר כתיבת האובייקטים במסך הXML משפיע על סדר הופעתם על המסך

הקוד הסופי הסופי שלנו בקובץ הXML ייראה כך:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:background="@drawable/back1"          <-  הוספנו כאן תמונה לרקע
  >
  <EditText
     android:layout_width="200dp"
     android:layout_height="wrap_content"
     android:id="@+id/etFname"
     android:hint="First Name"
     android:layout_gravity="center"
     android:textSize="30sp"
     />
  <EditText
     android:layout_width="200dp"
     android:layout_height="wrap_content"
     android:id="@+id/etLname"
     android:layout_gravity="center"
     android:hint="Last Name"
     android:textSize="30sp"
     android:layout_marginTop="10dp" קביעת מרחק האובייקט מהאובייקט שמעל
```

```
    />
  <Button
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:text="Save"
    android:layout_gravity="center"
    android:id="@+id/btnSave"
    android:textSize="30sp"
    android:layout_marginTop="10dp"  קביעת מרחק האובייקט מהאובייקט שמעל
    />
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/tvDisplay"
      android:text=""
      android:textSize="30sp"
      ></TextView>
</LinearLayout>
```

שלב 2 - הצהרה על תכונות בקובץ ה(class) JAVA:

כעת נצהיר בקובץ הJAVA על המשתנים שנרצה שישלטו לנו על האובייקטים מאחורי הקלעים.

נצטרך בשביל זה להגדיר משתנה לכל אחד מהאובייקטים שנרצה שיפעלו במהלך השימוש באפליקציה:

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
  EditText etFirstName,etLastName;
  TextView tvDisplay;
  Button btnSave;
```

שלב 3 - קבלת הפניה לתכונות שב XML

נבצע הפניה לאובייקטים שיצרנו בשלב 1 בXML לאובייקטים שיצרנו בשלב 2 בקובץ JAVA:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    etFirstName = (EditText)findViewById(R.id.etFname);
    etLastName = (EditText)findViewById(R.id.etLname);
    btnSave = (Button)findViewById(R.id.btnSave);
    tvDisplay=(TextView)findViewById(R.id.tvDisplay);
}
```

שלב 4 - האזנה באמצעות setOnClickListener:

כעת הגיע החלק שבו אנו כותבים את הקוד שיפעל בזמן שנשתמש באפליקציה, הקוד שיחיה את הlayout שעיצבנו.

תחילה נביא לbutton אפשרות האזנה לפונקציית לחיצה על כפתור.

חשוב מאוד - ללא אפשרות האזנה הכפתור לא יעבוד ויהיה אובייקט חסר שימוש על המסך!

נכתוב את שורה זו בתוך פונקציית הonCreate אחרי שהפנינו את הbutton לאחד שיצרנו בקובץ הXML

```
btnSave.setOnClickListener(this);
```

וכעת נרשום בתוך פונקציית ההאזנה פקודה לפיה כאשר המשתמש ילחץ על כפתור save והוא מלא את התיבות הטקסט של שם פרטי ושם משפחה, יופיע שמו המלא על המסך בתוך תיבת הטקסט.

נעשה זאת כך:

```
@Override
 public void onClick(View v) {

    if(v==btnSave)
    {
       if(etLastName.getText().toString().length()>0 && etFirstName.getText().toString().length()>0)
       {
          tvDisplay.setText(etFirstName.getText().toString() + "  " + etLastName.getText().toString());
          etFirstName.setText("");
          etLastName.setText("");
       }
    }
 }
```

הקוד הסופי שלנו לדוגמה 2 ייראה כך:

XML:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:orientation="vertical"
 android:background="@drawable/back1"          <-  הוספנו כאן תמונה לרקע
 >
 <EditText
    android:layout_width="200dp"
```

```
    android:layout_height="wrap_content"
    android:id="@+id/etFname"
    android:hint="First Name"
    android:layout_gravity="center"
    android:textSize="30sp"
    />
<EditText
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:id="@+id/etLname"
    android:layout_gravity="center"
    android:hint="Last Name"
    android:textSize="30sp"
    android:layout_marginTop="10dp" קביעת מרחק האובייקט מהאובייקט שמעל
    />
<Button
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:text="Save"
    android:layout_gravity="center"
    android:id="@+id/btnSave"
    android:textSize="30sp"
    android:layout_marginTop="10dp"  קביעת מרחק האובייקט מהאובייקט שמעל
    />
   <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
```

```
        android:id="@+id/tvDisplay"
        android:text=""
        android:textSize="30sp"
        ></TextView>
</LinearLayout>
```

JAVA:

```
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
  EditText etFirstName,etLastName;
  TextView tvDisplay;
  Button btnSave;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     etFirstName = (EditText)findViewById(R.id.etFname);
     etLastName = (EditText)findViewById(R.id.etLname);
     btnSave = (Button)findViewById(R.id.btnSave);
     tvDisplay=(TextView)findViewById(R.id.tvDisplay);
     btnSave.setOnClickListener(this);
  }
  @Override
  public void onClick(View v) {


     if(v==btnSave){
        if(etLastName.getText().toString().length()>0 &&
                        etFirstName.getText().toString().length()>0){
           tvDisplay.setText(etFirstName.getText().toString() + "  " +
                        etLastName.getText().toString());
           etFirstName.setText("");
           etLastName.setText("");
        }
```

```
    }
  }
}
```

כעת, לאחר שסיימנו את שתי הדוגמאות הראשונות, נלמד מושגים חשובים.

Context:

אז מהו Context בעצם?

Context הוא בעצם הפניה למי שיצר את המשתנה. הוא יכול להיות הפניה לממשק על או מחלקת על שיצרה את המשתנה.

אנו יודעים שכל אובייקט יורש מObject, וב visual studio כל אובייקט יורש גם ממחלקת העל Context, שיורשת מ Object.

אז למה בעצם לא להשתמש פשוט בObject?

כיוון שלמחלקת Context יש יותר פונקציות ותכונות שימושיות, וכמובן, לפי חוקי הירושה שאנו מכירים, מופע של מחלקת הבת יכול גם להשתמש בפונקציות ובתכונות של מחלקת האב.

מתי נשתמש בContext?

כאשר ניצור מחלקה, (כלומר קובץ JAVA שלא קשור לActivity) ונרצה להשתמש באובייקט כללי.

לדוגמה:

נרצה לשנות בכמה מסכים את הטקסט שמופיע ב TextView שבמסך לטקסט מסוים, ולא נרצה בכל פעם לרשום את אותו הקוד מחדש.

בשביל זה ניצור מחלקה שבה נרשום את הקוד, וניגש ל מסך שבו נמצא ה TextView בעזרת context.

איך נעשה זאת?

ניעזר בדוגמה הבא:

דוגמה 3:

בדוגמה זו נלמד איך להאזין ללחיצה על הכפתור שבנינו בדוגמה 1 בעזרת מחלקה חיצונית (מחוץ לActivity).

נעשה זאת לפי השלבים שלמדנו:

- שלב 1

נשאיר את הlayout זהה לזה שבדוגמה הראשונה (אפשר גם לבנות באותו פרויקט)

- שלב 2

נצהיר על אותם המשתנים כבדוגמה הראשונה.

- שלב 3

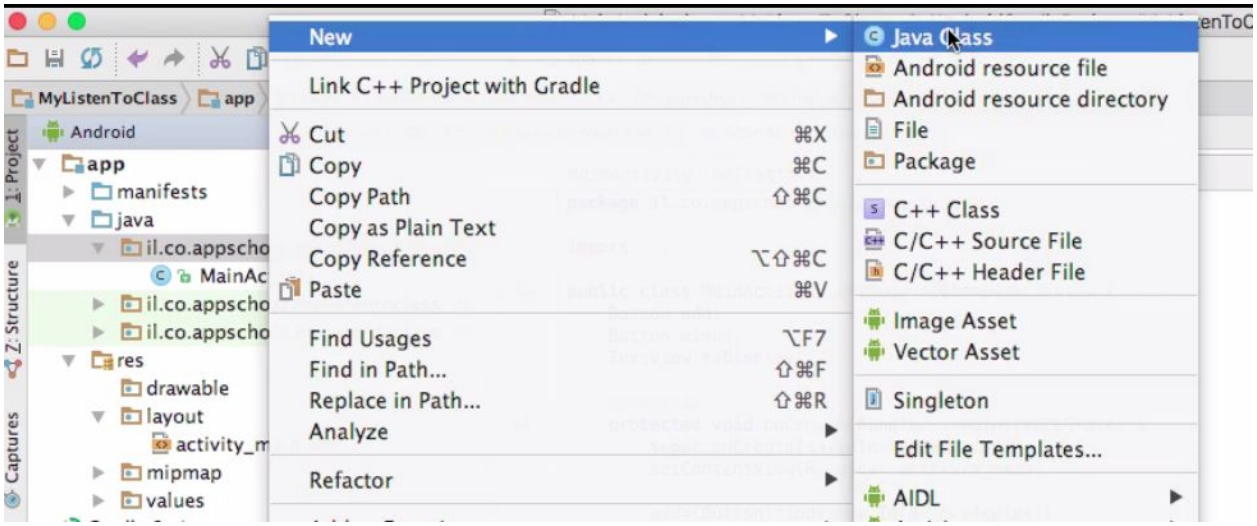
נתן הפניה באותה הצורה כבדוגמה 1.

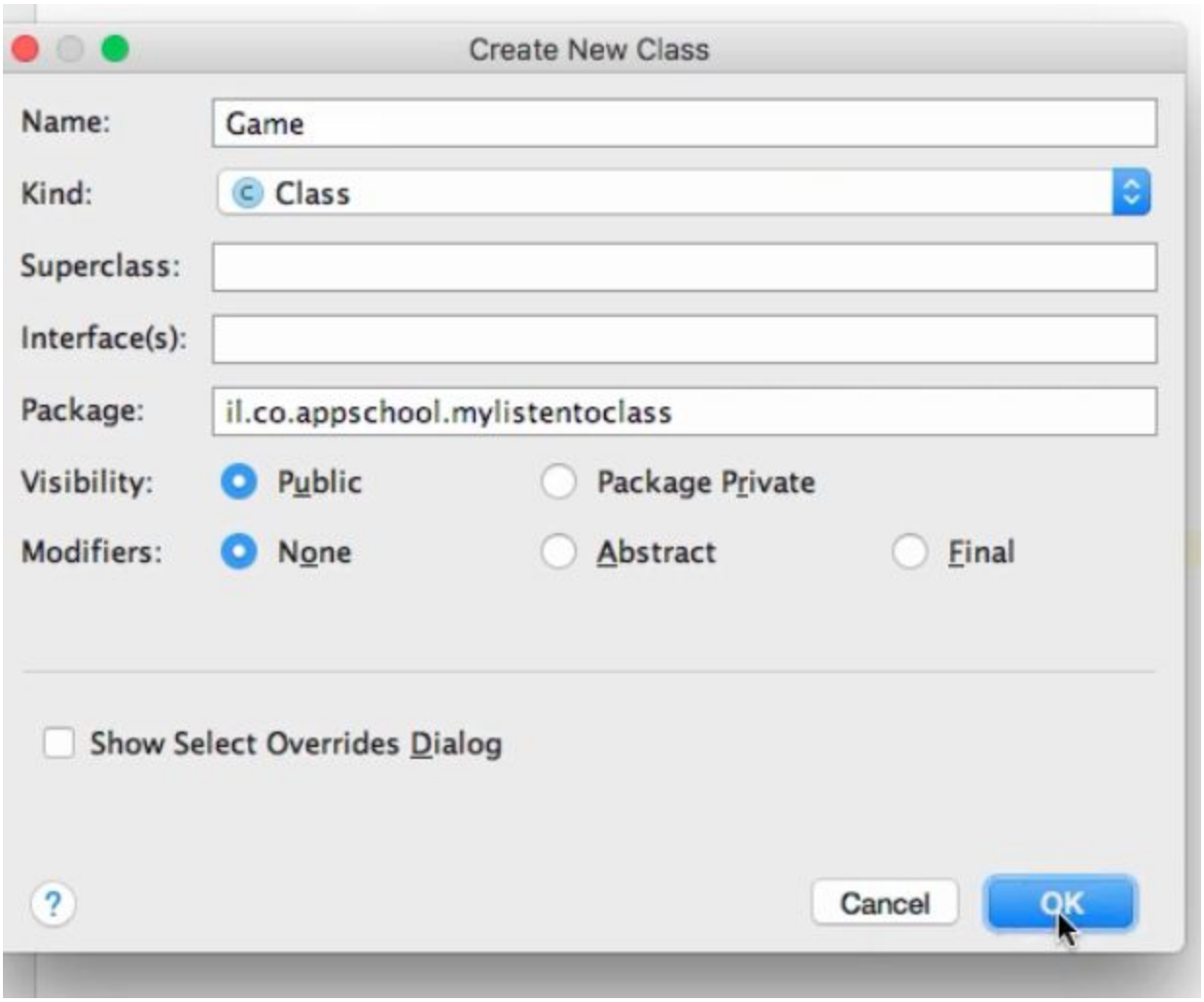
עד כה האפליקציה שלנו זהה לאפליקציה הראשונה שבנינו.

- שלב 4

בשלב זה ניתן לתוכנית להאזין ללחיצה על כפתור במחלקה אחרת.

ראשית כדי ליצור מחלקה נוספת נעקוב אחר השלבים הבאים:





ניצור מחלקה בשם Game שיורשת מ Button וניתן לה מספר תכונות:

```
public class Game implements View.OnClickListener{

   TextView tvDisplay;

   int point = 0;

   String message;
```

ו Constructor:

```
public Game(int point, String message, TextView tvDisplay) {

      this.point = point;

      this.message = message;

      this.tvDisplay = tvDisplay;

   }
```

ניתן גם getters וsetters:

```
      public int getPoint()
      {
          return point;
      }
      public void setPoint(int point)
      {
          this.point = point;
      }
      public String getMessage()
      {
```

```
        return message;
    }
    public void setMessage(String message)
    {
        this.message = message;
    }
```

ולבסוף אפשר להוסיף גם את פונקציית הtoString:

```
@Override
public String toString() {
    return "Game{" +
            "point=" + point +
            ", message='" + message + '\'' +
            '}';
}
```

עכשיו לאחר שיצרנו את הבסיס למחלקה, נבנה פונקציית OnClick שתאזין לכפתור:

נגדיר שהמחלקה שלנו תממש את הממשק View.OnClickListener

שבו נמצאת פונקציית ההאזנה OnClick:

public class Game **implements** View.OnClickListener{

עימדו עם סמן העכבר על השורה שבה מימשנו את הממשק, וליחצו על הנורה האדומה בצד שמאל בשביל לממש את הפונקציה onClick:

```
public void OnClick(object sender, System.EventArgs e){
        Button btn = (Button)sender;
        if (btn.Id == R.Id.btnplus)
            point++;
        else if (btn.Id == R.Id.btnminus){
            point--;
        }
        this.message = "You Have " + this.point + " points";
        tvDisplay.setText(this.message);
    }
  }
}
```

עכשיו נראה איך יראה קובץ הMainActivity:

```
public class MainActivity extends AppCompatActivity {
  Button add;
  Button minus;
  TextView tvDisplay;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    add=(Button)findViewById(R.id.btnplus);
    minus=(Button)findViewById(R.id.btnminus);
    tvDisplay=(TextView)findViewById(R.id.txtdisplay);
    Game game = new Game(0, "" , tvDisplay);
    add.setOnClickListener(game);
    minus.setOnClickListener(game);
  }
}
```

בשיעור זה נלמד מהו Toast.

Toast הוא בעצם הודעה שמופיעה על המסך למספר שניות ולאחר מכן נעלמת.
כאשר נרצה להשתמש בToast נכתוב אותו בקובץ הJAVA במקום שבו נרצה שהוא יפעל.

לדוגמה, במידה ונרצה שבכל פעם שהמשתמש ילחץ על כפתור תופיע הודעה קצרה, נכתוב קוד כזה:

```
Toast.makeText(this,"Please fill all fields",Toast.LENGTH_LONG).show();
```

סיכום המושגים:

לסיום הפרק הראשון נעבור על כל המושגים שלמדנו עד כה:

layout - קבצי UI שמכילים את כל הגרפיקה של האפליקציה

Activity-מסך באפליקציה. בנוי מקובץ XML (החלק הגרפי) וקובץ JAVA (מאחורי הקלעים)

onCreate - הפונקציה הראשונה שנקראת כשמגיעים למסך חדש.

Button ,TextView,ImageView,EditText - אובייקטים שלמדנו איך להשתמש בהם

wrap_content - יחידת מידה לפיה האובייקט תופס את השטח המינימלי שדרוש לתוכן

match_parent -יחידת מידה לפיה האובייקט מתפרש על פני כל השטח הפנוי (לפי רזולוציית המסך)

dp - יחידת מידה של אורך

sp - יחידת מידה של גודל הטקסט

hint - רמז על תיבת טקסט

layout_width - רוחב האובייקט (נקבע לפי match_parent/ wrap_content /dp)

layout_height - אורך האובייקט (נקבע לפי match_parent/ wrap_content /dp)

gravity_layout - סדר האובייקט על המסך (שמאל/ מרכז/ ימין)

gravity - סדר התוכן בתוך האובייקט (שמאל/ מרכז/ ימין…)

margin - מרחק האובייקט מאובייקט אחר/ דופן המסך (לפי השוליים)

padding - מרחק התוכן מהשוליים של האובייקט

onClick - פונקציה שמאזינה ללחיצה על כפתור. בשביל לממש אותה צריך לממש את הממשק בו היא נמצאת. (View.OnClickListener)

res - תיקיה בה מופיעים כל קבצי הXML (הקבצים שקשורים לגרפיקה של האפליקציה)

drawable- תיקיה המכילה את התמונות

values - תיקיה המכילה הגדרות מרכזיות של האפליקציה כגון צבע, שם האפליקציה וכו'

manifest - קובץ בהגדרות של האפליקציה. בו נרשמים המחלקות והקבצים שנוצרים