תיק פרויקט

15 puzzle

מגיש: ישראל שראל

ת.ז: 214762494

בית הספר: ישיבת בני עקיבא השומרון

שם המנחה: דבורה יעיש

שם החלופה: טלפונים ניידים

**תוכן העניינים**

תוכן

[**תוכן העניינים** 2](#_Toc106282459)

[**מבוא:** 3](#_Toc106282460)

[יכולות המערכת: 3](#_Toc106282461)

[תיאור תחום הידע – פרק מילולי 4](#_Toc106282462)

[**מבנה / ארכיטקטורה של הפרויקט** 6](#_Toc106282463)

[Splash Activity – מסך פתיחה 6](#_Toc106282464)

[Main Activity – המסך הראשי 7](#_Toc106282466)

[Solved Dialog - דיאלוג לוח פתור 8](#_Toc106282467)

[Menu main – תפריט ראשי 9](#_Toc106282468)

[Setting Activity – מסך הגדרות 10](#_Toc106282469)

[Record Dialog - דיאלוג שיאים 11](#_Toc106282470)

[Sms Activity – מסך שליחת הודעת sms למפתח 12](#_Toc106282471)

[**Screen flow diagram** 13](#_Toc106282472)

[**תיאור מחלקות הפרויקט - UML** 14](#_Toc106282473)

[**מימוש הפרויקט** 15](#_Toc106282474)

[חלק גרפי (קבצי תצורה/קונפיגורציה) 15](#_Toc106282475)

[**חלק לוגי (קבצי קוד Java)** 25](#_Toc106282476)

[קבצי קוד - Activities 25](#_Toc106282477)

[קבצי קוד – מחלקות העצמים 31](#_Toc106282478)

[קבצי קוד – מחלקות עזר 42](#_Toc106282479)

[**מדריך למשתמש** 43](#_Toc106282480)

[**סיכום אישי/רפלקציה** 45](#_Toc106282481)

[**רשימת מקורות/ביבליוגרפיה** 46](#_Toc106282482)

[**נספחים** 47](#_Toc106282483)

**מבוא:**

הרקע לפרויקט:

שם הפרויקט הוא ,15 puzzle זהו משחק ישן שבנוי על לוח עם אריחים ומטרתו היא לסדר את המספרים לפי הסדר מ1 עד המספר הכי גדול.

**כללי המשחק:** המשחק מתחיל כשהלוח מבולגן והמספרים לא במקום, בנוסף יש מקום אחד ריק וניתן להזיז רק אליו אריחים, המטרה היא לסדר את המספרים מ1 ועד המספר הכי גדול לפי הגודל שנקבע(הdefault הוא 4x4).

קהל היעד של המשחק הוא כל אחד שרוצה לשחק במשחק בזמנו החופשי ולהפעיל את המוח.

הסיבה שבחרתי את הנושא הנ"ל זה שעניין אותי לבנות משהו שלא רק מצריך ידע במושגים בתכנות ובכל מיני דברים טכניים אלא גם חשיבה מעמיקה ואלגוריתמיקה.

במשחקים כמו זה יש הרבה דברים שצריך לחשוב עליהם ולהפעיל את המוח.

יכולות המערכת:

* לשחק במשחק
* לערבב את המשחק ולהתחיל מהתחלה בכל שלב במשחק
* להשהות את המשחק ולהמשיך
* לשנות את הגודל של הלוח וכך גם את הצבע של האריחים
* לראות את השיאים מסודרים לפי כמות המהלכים ולפי הזמן שלקח לפתור עבור כל גודל של לוח בנפרד
* לשמור את השיאים מתועדים עם התאריך בו הם בוצעו.
* לשלוח הודעת sms למפתח

תהליך המחקר:

בהתחלה לקחתי את המשחק ושיחקתי איתו קצת. ניסיתי לחשוב אילו קשיים יכולים להיות בבניית המשחק, לאחר מכן הסתכלתי על הערך שלו בוויקיפדיה ולמדתי עוד קצת על אופי המשחק.

לאחר מכן ניסיתי לחשוב באיזו צורה של תכנות אני אצטרך להשתמש בשביל לבנות את המשחק בצורה הכי אידיאלית ואלגנטית. הסתכלתי קצת באתר github על פרויקטים דומים בשביל לקבל איזו נקודת מבט על איך אני רוצה שהפרויקט יהיה בנוי. בgithub ראיתי שדרך מאוד טובה לעבוד עם משחקים כמו המשחק הזה היא בניית class שיורש מView . התחלתי קצת ללמוד על זה ולהתנסות בזה.

כמובן שהסתכלתי גם על אפליקציות דומות וראיתי מה המודל הכי טוב בשביל הפרויקט הזה מבחינת כל המובנים.

אתגרים מרכזיים:

בעיה רצינית שהייתה לי היא איך לבנות את הלוח של המשחק בצורה הכי אלגנטית וקלה לשימוש.

בעיה נוספת שאיתה התמודדתי היא שיכול לצאת מצב שבו סידור המספרים בלוח יהיה כך שהלוח יהיה לא פתיר. הייתי צריך לבנות אלגוריתם שפותר את הבעיה וגם התמודדתי עם המורכבות של לשלב בסיס נתונים וsql באפליקציה.

עוד אתגר מאוד רציני שנתקלתי בו הוא המעבר בין מצב שהמשחק עוצר למצב שהוא רץ או מצב שהוא מתחיל מחדש. מכיוון שיש הרבה ואריאציות של מעברים היה לי קשה להצליח לתפעל את כולם בצורה הנכונה. בשביל להתמודד עם בעיה זו ממש הייתי צריך לתכנן ולבנות סוג של מודל חישובי שיקל עלי בהתייחסות לכל המצבים. (מצורף בנספחים דף שאיתו תכננתי את המעברים בכתב יד)

אלמנט טכנולוגי חדשני שהשתמשתי בו הוא recyclerView במקום בlistView ובשביל זה הייתי צריך לייבא ספרייה בgradle.

הרחבה נוספת שהכנסתי היא אנימציה במסך פתיחה בשביל שהאפליקציה תיפתח בצורה יפה שתעניין את המשתמש.

תיאור תחום הידע – פרק מילולי

אובייקטים:

בשביל משחק זה בניתי מספר אובייקטים:

Tile: אובייקט שמייצג אריח אחד עם מספר על הלוח.

Square: אובייקט שמייצג מקום בלוח שבו יכול להיכנס אריח (Tile).

BoardGame: אובייקט שיורש מView ומכיל מערך דו ממדי של Tile ומערך דו ממדי של Square.

Game: אובייקט ששולט בכל המשחק מבחינת המצב ריצה/יצירת BoardGame ועוד.

Time: אובייקט שיורש מThread ואחראי על הטיימר שסופר את הזמן.

Record: אובייקט שמייצג יחידה אחת של שיא.

RecordAdapter: אובייקט שיורש מ RecyclerViewודואג לסדר את הרקורדים כך שנוכל להציג אותם ברשימה

RecordHelper: אובייקט שיורש מ SQLiteOpenHelperודואג לשמור את הנתונים בdataBase פנימי של האפליקציה.

כאמור השתמשתי בייצוג מידע של מערך דו ממדי בעיקר בשביל לייצג נתונים כמו האריחים (Tile) / המקומות על הלוח (Square) וגם במערך חד ממדי של מספרים בשביל הסדר שלהם על הלוח. בנוסף השתמשתי בטבלה(database) עבור השיאים עם פעולה של הוספה של שיא:createRecord(Record r) ועוד פעולה של קבלת כל השיאים: getAllRecord().

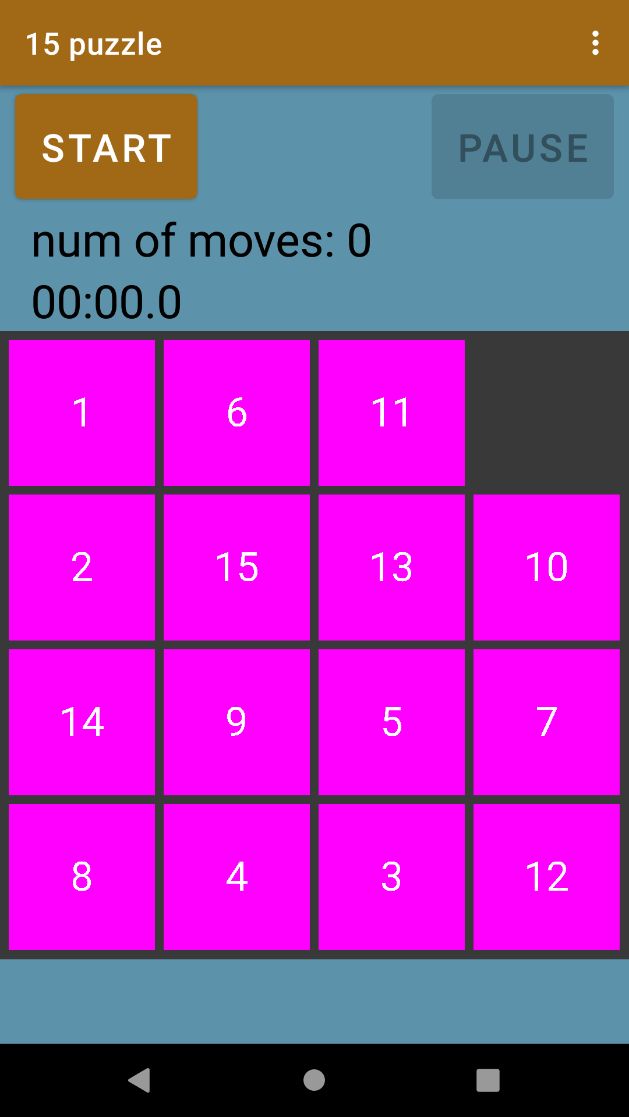
**מבנה / ארכיטקטורה של הפרויקט**

Splash Activity – מסך פתיחה

## 

המסך הראשון שמופיע כשמופעל האפליקציה, הוא מראה את המספר 15 ואת המילה "puzzle" באנימציה אחד אחרי השני. לאחר מספר שניות הוא מעביר אותנו למסך הראשי(MainActivity).

Main Activity – המסך הראשי



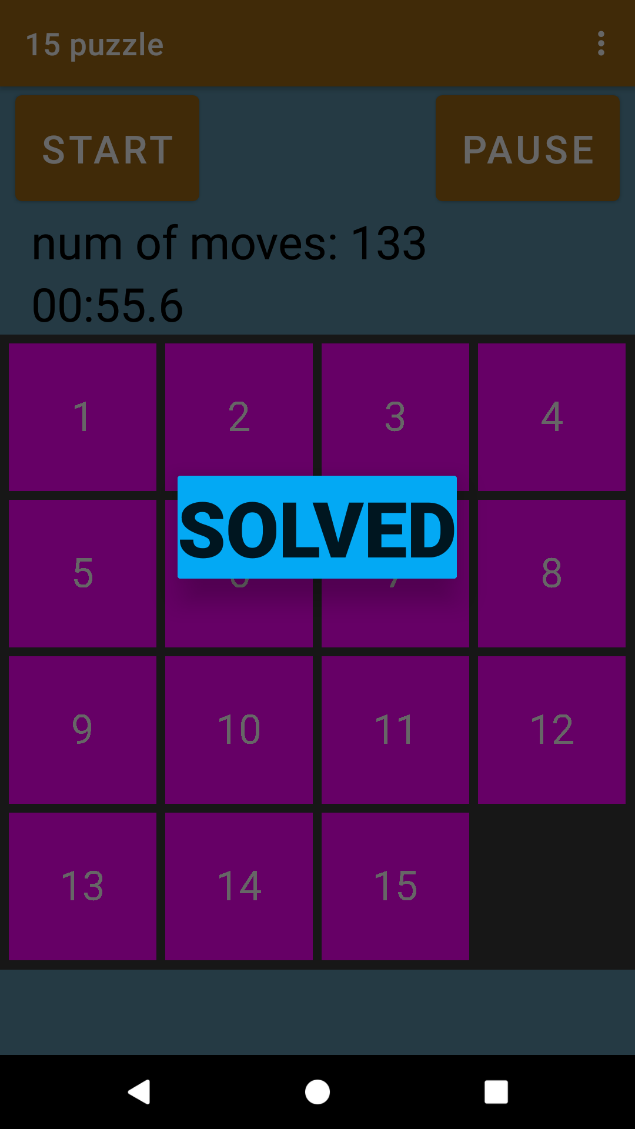
המסך הראשי של המשחק ובו מוטמע האובייקט של הלוח BoardGame בLinearLayout. כשלוחצים על הלוח הטיימר מתחיל לרוץ ואז נהיה אפשרי ללחוץ על הpause.

אם לוחצים על אריח ליד המקום הריק, האריח זז לשם וה"number of moves" עולה ב1.

אם לוחצים על הstart הטיימר והמהלכים מתאפסים והלוח מתערבב.

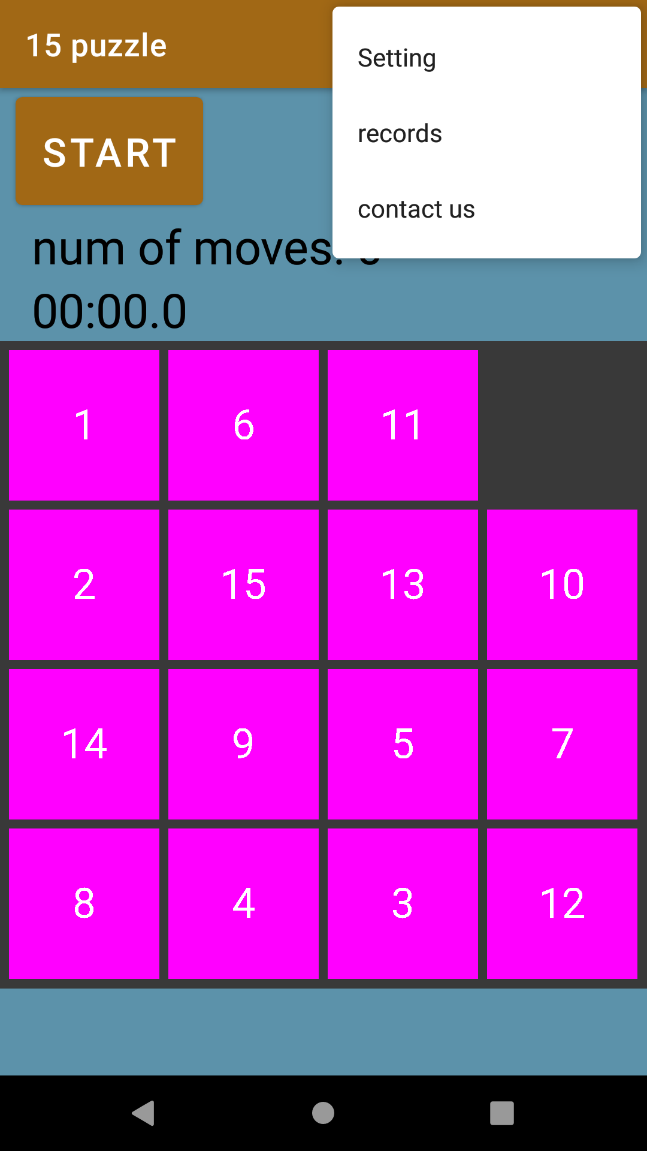
אם לוחצים על הpause הכיתוב משתנה לcontinue, הטיימר נעצר ולא ניתן להמשיך את המשחק עד שלוחצים על הcontinue. אז המשחק ממשיך והכיתוב משתנה חזרה לpause. בonPause המשחק נעצר ובonCreate המשחק נוצר. לחיצה על ה3 נקודות תפתח תפריט.

Solved Dialog - דיאלוג לוח פתור



הדיאלוג הזה נפתח מתי שהשחקן פותר את הלוח וגומר את המשחק. המשחק נעצר והתוצאה שלו נשמרת בבסיס נתונים, בשביל לשחק שוב צריך ללחוץ על solved ואז זה מאפס את הטיימר והמהלכים ומערבב את הלוח.

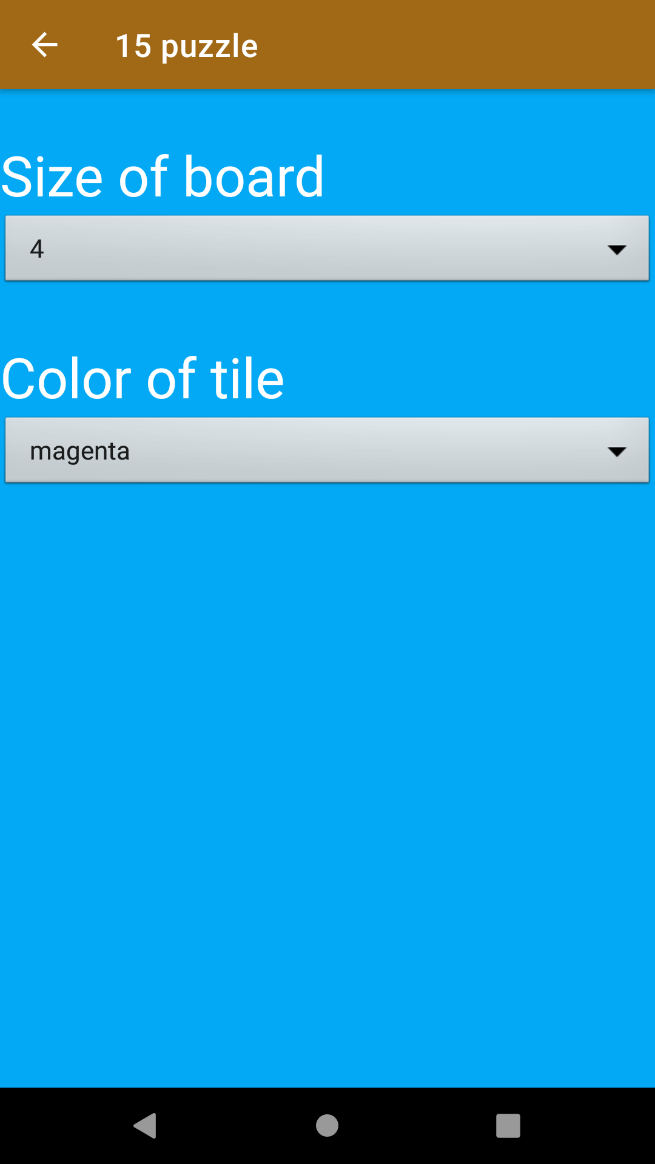
Menu main – תפריט ראשי



בתפריט הראשי יש שלושה אפשרויות

1. מעבר למסך הגדרות על ידי Intent(SettingActivity) ואיפוס המשחק.
2. פתיחה של dialog של השיאים שמתאים לגודל הלוח הספיציפי שמוגדר ועצירת המשחק.
3. מעבר למסך של שליחת sms למפתח(SmsActivity) ואיפוס המשחק.

Setting Activity – מסך הגדרות

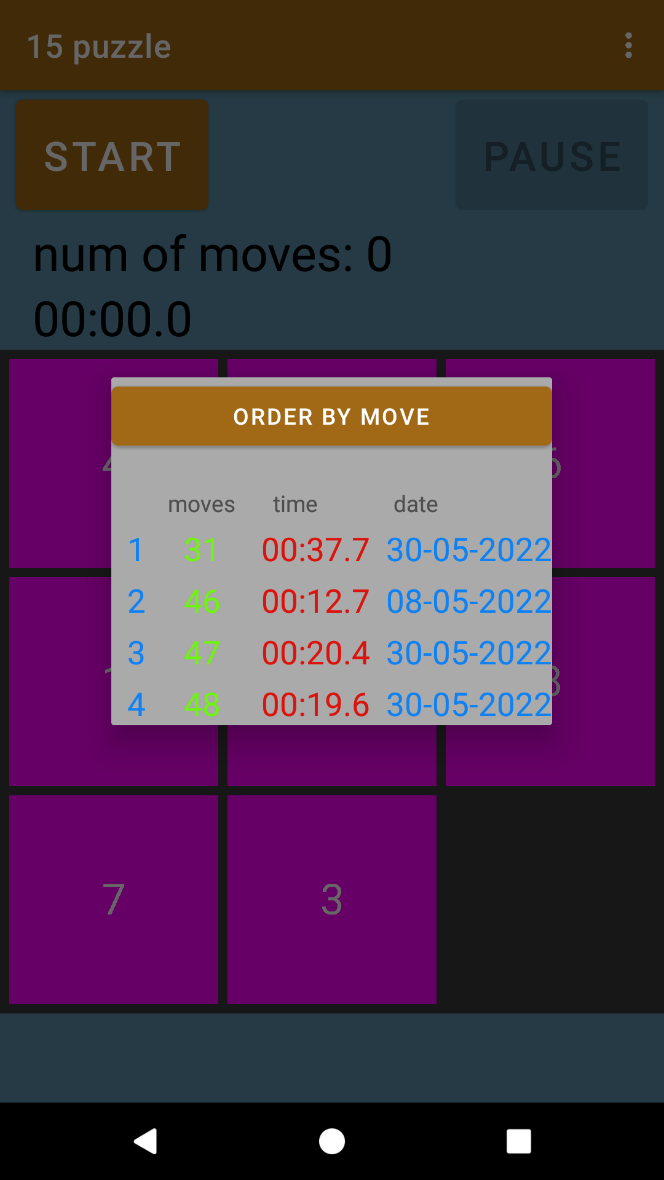


במסך זה ניתן לשנות את גודל הלוח של המשחק (Size of board) מגודל של 3x3 עד גודל של 10x10.

בנוסף ניתן לשנות את צבע האריחים לשלל צבעים כגון ירוק, אדום, סגול ועוד...

בלחיצה על החץ למעלה ניתן לחזור למסך הראשי (MainActivity) עם ההגדרות שנשמרו.

Record Dialog - דיאלוג שיאים



לאחר שלוחצים על הכפתור record בתפריט נפתח דיאלוג שמראה את כל השיאים שנשמרו בבסיס נתונים מסודרים בהתחלה לפי מספר מהלכים. אפשר לשנות כך שהשיאים יהיו מסודרים לפי הזמן ולאחר מכן לשנות חזרה. כשהדיאלוג נפתח המשחק נעצר, בשביל לצאת מהדיאלוג יש ללחוץ על המסך סתם לא איפה שהדיאלוג נמצא.

Sms Activity – מסך שליחת הודעת sms למפתח

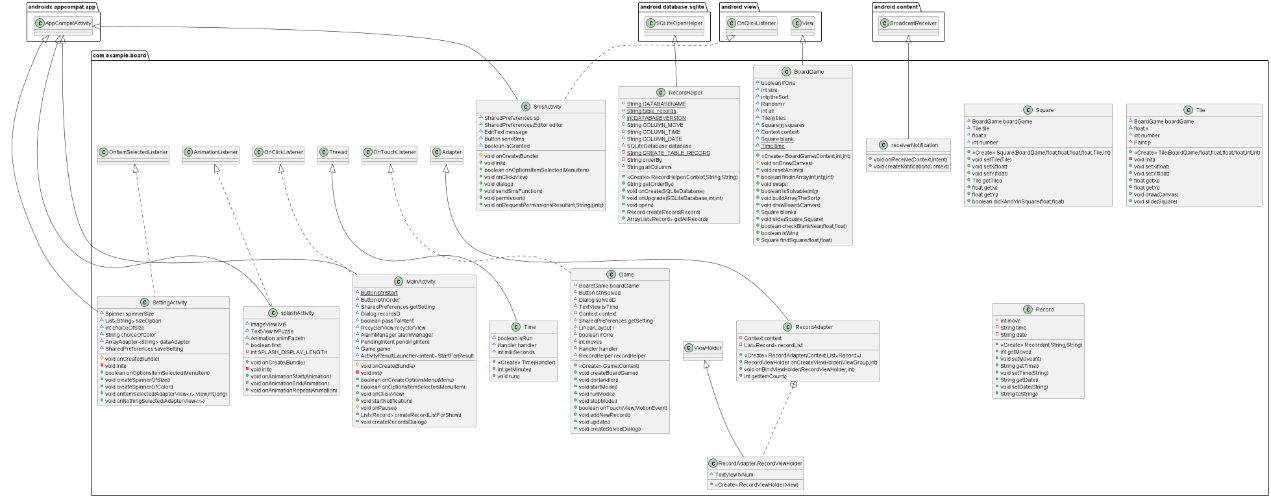


במסך זה ניתן לכתוב הודעה למפתח ולשלוח לו sms. אם מנסים לשלוח הודעה ריקה מוקפץ Toast שמסביר שלא ניתן לשלוח הודעה ריקה. בפעם הראשונה שמנסים לשלוח הודעת זה מפעיל בקשת permission לשליחת sms, בלחיצה על החץ למעלה ניתן לחזור למסך הראשי(MainActivity).

**Screen flow diagram**

**תיאור מחלקות הפרויקט - UML**

לתמונה באיכות יותר גבוהה:

<https://drive.google.com/file/d/1Oz3Ue1M0WyHpmiIpVNFb02DP9eQ2xY6W/view?usp=sharing>

**מימוש הפרויקט**

חלק גרפי (קבצי תצורה/קונפיגורציה)

**activity\_main.xml:**

קובץ התצוגה הגרפית במסך הפתיחה (main activity), בו מוצג הלוח של המשחק.

יש בו כפתור start שאחראי לערבב את הלוח ולאפס את הטיימר והמהלכים, כפתור stop שעוצר את המשחק אם הוא רץ וממשיך אותו אם הוא בעצירה, text view שמראה את מספר המהלכים ועוד text view שמראה את הטיימר.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#5C92AA"  
 tools:context=".MainActivity">  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <Button  
 android:id="@+id/btnStart"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:text="start"  
 android:textSize="25sp"/>  
  
 <Button  
 android:id="@+id/btnPause"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="80dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:layout\_alignParentRight="true"  
 android:text="pause"  
 android:enabled="false"  
 android:textSize="25sp" />  
  
 </RelativeLayout>  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/tvMoves"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="right"  
 android:layout\_marginLeft="20dp"  
 android:text="number of moves: 0"  
 android:textColor="@color/black"  
 android:textSize="30sp" />  
  
 <TextView  
 android:id="@+id/tvTime"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="20dp"  
 android:layout\_gravity="left"  
 android:text="00:00.0"  
 android:textColor="@color/black"  
 android:textSize="30sp" />  
 </LinearLayout>  
   
 </LinearLayout>  
 <!--בתוך הLinearLayout מכניסים את הלוח של המשחק-->  
 <LinearLayout  
 android:layout\_gravity="center"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/lGame"  
 android:orientation="vertical" >  
 </LinearLayout>  
</LinearLayout>

**activity\_setting.xml:**

קובץ התצוגה הגרפית של מסך ההגדרות (setting activity) בו מוצגות ההגדרות של המשחק: text view שכתוב עליו "size of board" ומתחתיו spinner בו ניתן לבחור את גודל הלוח, text view שכתוב עליו "color of tile" ומתחתיו spinner בו ניתן לבחור את צבע האריחים.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#03A9F4"  
 tools:context=".SettingActivity">  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="30dp"  
 android:text="Size of board"  
 android:textSize="35sp"  
 android:textColor="#fff"/>  
  
 <Spinner  
 android:id="@+id/spinnerSize"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:background="@android:drawable/btn\_dropdown"  
 android:spinnerMode="dropdown"  
 android:prompt="@string/spinner\_title"  
 />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="30dp"  
 android:text="Color of tile"  
 android:textSize="35sp"  
 android:textColor="#fff"/>  
  
 <Spinner  
 android:id="@+id/spinnerColor"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:background="@android:drawable/btn\_dropdown"  
 android:spinnerMode="dropdown"  
 android:prompt="@string/spinner\_title"  
 />  
</LinearLayout>

**activity\_sms.xml:**

קובץ התצוגה הגרפית במסך השליחת sms (sms activity), בו מוצג edit text עבור כתיבת ההודעה של המשתמש וbutton בשביל לשלוח את ההודעת sms למפתח.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".SmsActivity">  
   
 <EditText  
 android:id="@+id/etSms"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:hint="your message here"  
 android:ems="10"  
 android:inputType="textMultiLine" />  
   
 <Button  
 android:id="@+id/btnSms"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:text="Send SMS" />  
  
</LinearLayout>

**custom\_dialog\_records.xml:**

קובץ התצוגה הגרפית שמיועד להצגת דיאלוג השיאים, הוא מכיל כפתור שאחראי לשנות את סידור השיאים, table row עם כותרות העמודות move, time, date. מתחתן מוצגים הדברים שהם מתארים בתוך Recycler View, הם השיאים שנשמרו בבסיס נתונים.

<?xml version="1.0" encoding="utf-8"?>  
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:background="#aaa"  
 android:layout\_height="match\_parent">  
  
 <Button  
 android:id="@+id/btnOrder"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="order by move"  
 android:layout\_marginBottom="20sp"/>  
 <TableRow  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="35dp"  
 >  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="moves"/>  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="2"  
 android:text="time"/>  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
  
 android:layout\_weight="3"  
 android:text="date"/>  
   
 </TableRow>  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recyclerView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>  
  
</TableLayout>

**custom\_solved.xml:**

קובץ התצוגה הגרפית שמיועד להציג דיאלוג עם הכיתוב solved שקופץ מתי שהשחקן פותר את המשחק.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <android.widget.Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#03A9F4"  
 android:text="solved"  
 android:textSize="50sp"  
 android:textStyle="bold"  
 android:id="@+id/btnSolved"  
 />  
  
  
</LinearLayout>

**records\_layout.xml:**

קובץ התצוגה הגרפית שמיועד להציג פריסה של Record אחד, יש בו ארבע text view המייצגים תאריך, זמן, מהלכים וid, מממשים אותו בRecycler View לכל שיא ושיא בנפרד.

<?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="horizontal">  
  
  
 <TextView  
 android:id="@+id/tvId"  
 android:layout\_width="25dp"  
 android:layout\_height="wrap\_content"  
  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="5dp"  
 android:text="00"  
 android:textColor="#0e82f5"  
 android:textSize="20dp" />  
  
 <TextView  
 android:id="@+id/tvMoves"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
  
 android:layout\_marginTop="5dp"  
 android:text="000"  
 android:textColor="#6ef50e"  
 android:textSize="20dp" />  
  
 <TextView  
  
 android:id="@+id/tvTime"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="25dp"  
 android:layout\_marginTop="5dp"  
 android:text="00:00.0"  
 android:textColor="#dc130c"  
 android:textSize="20sp" />  
  
 <TextView  
 android:id="@+id/tvDate"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
  
 android:layout\_marginTop="5dp"  
 android:text="10.02.2022"  
 android:textColor="#0e82f5"  
 android:textSize="20dp" />  
  
</LinearLayout>

**menu\_main.xml:**

קובץ התצוגה הגרפית שמציג את התפריט הראשי המוצג במסך הפתיחה (main activity), יש בו item שמוביל למסך הגדרות (setting activity),item שמקפיץ את הדיאלוג של השיאים וitem שמוביל למסך שליחת sms activity) sms).

<?xml version="1.0" encoding="utf-8"?>  
<menu xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto">  
  
 <item android:id="@+id/setting"  
 android:icon="@android:drawable/ic\_menu\_manage"  
 android:title="Setting"  
 app:showAsAction="never"/>  
  
 <item android:id="@+id/records"  
 android:icon="@android:drawable/ic\_menu\_sort\_by\_size"  
 android:title="records"  
 app:showAsAction="never"/>  
  
 <item android:id="@+id/contact"  
 android:icon="@android:drawable/ic\_menu\_sort\_by\_size"  
 android:title="contact us"  
 app:showAsAction="never"/>  
  
</menu>

**Fade\_in.xml:**

קובץ xml שעושה אנימציה של פייד אין כלומר שהרכיב שעליו עושים את האנימציה מופיע בהדרגה.

<?xml version="1.0" encoding="utf-8"?>  
<set xmlns:android="http://schemas.android.com/apk/res/android">  
  
 <alpha  
 android:duration="1000"  
 android:fromAlpha="0.0"  
 android:toAlpha="1.0"  
 android:fillBefore="false"  
  
 >  
 </alpha>  
  
</set>

**strings.xml:**

קובץ xml שאחראי לשמור מחרוזות האפליקציה.

<resources>  
 <string name="app\_name">15 puzzle</string>  
 <string name="spinner\_title">spinner title</string>  
 <string name="message1">Don\'t forget to play today !!</string>  
 <string name="message2">Playing the game will improve your mind !</string>  
 <string name="message3">You still did not play today !? Go ahead !!</string>  
 <string name="message4">sad?? Bored?? This game will improve your mood instantly !!!</string>  
 <string name="message5">Come on! Take a minute and enjoy</string>  
</resources>

**themes.xml:**

קובץ xml שאחראי על כל העיצובים והצבעים של האפליקציה

<resources xmlns:tools="http://schemas.android.com/tools" xmlns:android="http://schemas.android.com/apk/res/android">  
 <!-- Base application theme. -->  
 <style name="Theme.Board" parent="Theme.MaterialComponents.Light.DarkActionBar">  
 <!-- Primary brand color. -->  
  
  
 <item name="colorPrimary">#A16815</item>  
 <item name="colorPrimaryVariant">#976113</item>  
 <item name="colorOnPrimary">@color/white</item>  
 <!-- Secondary brand color. -->  
 <item name="colorSecondary">@color/teal\_200</item>  
 <item name="colorSecondaryVariant">@color/teal\_700</item>  
 <item name="colorOnSecondary">#976113</item>  
 <!-- Status bar color. -->  
 <item name="android:statusBarColor" tools:targetApi="l">?attr/colorPrimaryVariant</item>  
 <!-- Customize your theme here. -->  
 </style>  
   
 <style name="splashScreenTheme" parent="Theme.MaterialComponents.Light.NoActionBar">  
 <item name="android:windowBackground">#1FCEE4</item>  
 <item name="android:statusBarColor">#1FCEE4</item>  
 </style>  
</resources>

**androidMaifest.xml:**

הקובץ של ההגדרות של המערכת שבו מגדירים את האקטיביטים, ברודכסטים וגם כותבים אילו הרשאות האפליקציה צריכה.

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 package="com.example.board">  
  
 <uses-permission android:name="android.permission.SEND\_SMS" />  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.Board">  
 <activity  
 android:name=".splashActivity"  
 android:screenOrientation="portrait"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 <activity  
 android:name=".SmsActivity"  
 android:screenOrientation="portrait"  
 tools:ignore="LockedOrientationActivity" />  
 <activity  
 android:name=".SettingActivity"  
 android:screenOrientation="portrait"  
 tools:ignore="LockedOrientationActivity" />  
 <activity  
 android:name=".MainActivity"  
 android:screenOrientation="portrait"  
 tools:ignore="IntentFilterExportedReceiver,LockedOrientationActivity">  
  
 </activity>  
  
 <receiver  
 android:name=".receiverNotification"  
 android:exported="false">  
 <intent-filter>  
 <action android:name="android.intent.action.BOOT\_COMPLETED" />  
 </intent-filter>  
 </receiver>  
 </application>  
  
</manifest>

**חלק לוגי (קבצי קוד Java)**

קבצי קוד – Activities

**SplashActivity:**

מחלקה של Activity שמתחברת לקובץ xml – activity\_splash.xml. זה המסך שעולה בהתחלה לכמה שניות כשפותחים את האפליקציה ומוצג בו השם של האפליקציה באנימציה.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| ImageView iv15 | תמונה של הלוגו של האפליקציה. |
| TextView tvPuzzle | רכיב טקסט שכתוב בתוכו את המילה "puzzle". |
| Animation animFadeIn; | אובייקט Animation שאחראי על האנימציות |
| boolean first | משתנה בוליאני המייצג אם כבר הוצגה האנימציה הראשונה |
| private final int SPLASH\_DISPLAY\_LENGTH = 2500 | הזמן שבו מופיעה המסך |
| public void onCreate(Bundle savedInstanceState) | פעולה מובנת שמופעלת מתי שהאקטיביטי מופעל ובה קוראים לפעולה init(), מפעילים את האנימציה הראשונה ומטפלים בהעברה לאקטיביטי הראשי עם handler. |
| private void init() | פעולה שמאתחלת את האובייקטים ומגדירה כמה הגדרות של המסך |
| public void onAnimationStart(Animation animation) | פעולה מובנת שנקראת מתי שהאנימציה מתחילה. |
| public void onAnimationEnd(Animation animation) | פעולה מובנת שנקראת מתי שהאנימציה נגמרת ובה אנחנו מפעילים את האנימציה השנייה. |
| public void onAnimationRepeat(Animation animation) | פעולה מובנת שמופעלת מתי שהאנימציה חוזרת על עצמה. |

**MainActivity:**

מחלקה ראשית שמתחברת לקובץ xml – activity\_main.xml. היא מייצגת את המסך הראשי ובו נמצא המשחק. ממנה מנווטים לשאר האקטיביטים.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| static Button btnStart | כפתור שמאפס את המשחק ומערבב אותו. |
| static Button btnPause | כפתור שעוצר את המשחק אם הוא רץ וממשיך אותו אם הוא בעצירה. |
| Button btnOrder | כפתור בדיאלוג של השיאים שהוא מסדר את השיאים לפי זמן או מהלכים. |
| Dialog recordsD | הדיאלוג של השיאים. |
| boolean passToIntent | משתנה שבודק אם המשתמש עבר לאקטיביטי אחר עם intent. |
| RecyclerView recyclerView | RecyclerView המציג את השיאים בדיאלוג השיאים ברשימה. |
| AlarmManager alarmManager | תכונה שמתחברת לService המערכת של תזמון ההתראות. |
| PendingIntent pendingIntent | תכונה שאחראית לשלוח את המשתמש לMain activity כשהוא לוחץ על ההתראה. |
| Game game | מופע של הclass שאחראי לתפעל את המשחק. |
| protected void onCreate(Bundle savedInstanceState) | פעולה מובנת של המערכת שמופעלת כשהאפליקציה נפתחת ובה קוראים לפעולה init(). |
| private void init() | פעולה שמאתחלת את האובייקטים, מעדכנת את הSharedPreferences, וקוראת ל startNotificationבפעם הראשונה בשביל להגדיר את ההתראה. |
| public boolean onCreateOptionsMenu(Menu menu) | פעולה שיוצרת את התפריט. |
| public boolean onOptionsItemSelected(@NonNull MenuItem item) | פעולה שנקראת כשלוחצים על אפשרות בתפריט. |
| public void onClick(View v) | פעולה שמופעלת כשלוחצים על כפתור. פה היא מטפלת בכמה כפתורים: כשולחצים על כפתור הstart אז מאתחלים את המשחק ומערבבים את הלוח.  כשלוחצים על כפתור הstop אז אם המשחק רץ מעבירים אותו למצב עצירה (stop mode) ואם המשחק עוצר אז מעברים אותו למצב ריצה (run mode). כשלוחצים על כפתור הsovled (שנמצא בדיאלוג שקופץ כשפותרים את הלוח) אז מאתחלים את המשחק ומערבבים את הלוח. |
| ActivityResultLauncher<Intent> StartForResult | אובייקט שאחראי לעשות דברים כשחוזרים מIntent. |
| public void onActivityResult(ActivityResult result) | פעולה בתוך class אנונימי שמופעל מתי שחוזרים מIntent ופה מאתחלים את המשחק. |
| public void startNotification() | פעולה שמגדירה חזרה של התראה שמזכירה לשחק כל יום ב20:30. הפעולה נקראת רק פעם אחת ויותר לא קוראים לה. |
| protected void onPause() | פעולה מובנת שמופעלת כשיוצאים מה.Activity כשהיא נקראת עוצרים את המשחק(stop mode) |
| public List<Record> createRecordListForShow() | פעולה שיוצרת רשימה של השיאים. אם יש יותר מ10 שיאים היא מצמצמת את הרשימה ל10 שיאים הראשונים. |
| public void createRecordsDialog() | פעולה שיוצרת את הדיאלוג של השיאים, ובתוכה יש פעולת onClick() שמטפלת במקרה בו לוחצים על הכפתור btnOrder. |

**SettingActivity:**

מחלקה של activity שמתחברת לקובץ xml – activity\_setting.xml. היא מייצגת מסך הגדרות לאפליקציה שבו יש אפשרויות של שינוי גודל הלוח ושינוי הצבע של האריחים בלוח.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| Spinner spinnerSize | ספינר לבחירת גודל הלוח |
| Spinner spinnerColor | ספינר צבע האריחים |
| List<String> sizeOption | רשימה שבתוכה יש את כל הגדלים ואותה מכניסים לתוך spinnerSize. |
| List<String> colorOption | רשימה שבתוכה יש את כל הצבעים ואותה מכניסים לתוך .spinnerColor |
| int choiceOfSize | משתנה שבו שמים את הגודל הנוכחי של הלוח ומכניסים לו את הגודל החדש אם המשתמש שינה את הגודל. |
| String choiceOfColor | משתנה שבו שמים את הצבע הנוכחי של הלוח ומכניסים לו את הצבע החדש אם המשתמש שינה את הצבע. |
| protected void onCreate(Bundle savedInstanceState) | פעולה מובנת של המערכת שמופעלת כשהאקטיביטי נפתח, קוראת לinit(). |
| private void init() | פעולה שמאתחלת את האובייקטים, מקבלת מהאחסון הפנימי את ההגדרות הנוכחיות של הצבע והגודל וקוראת ל createSpinnerOfSize() ול createSpinnerOfColor() |
| public boolean onOptionsItemSelected(@NonNull MenuItem item) | פעולה שמופעלת כשלוחצים על הכפתור חזרה שנמצא למעלה בAction bar, מחזיר אותנו למסך הראשי. |
| public void createSpinnerOfSize() | פעולה שיוצרת את הספינר של הגדרת הגודל של הלוח. |
| public void createSpinnerOfColor() | פעולה שיוצרת את הספינר של הגדרת הצבע של האריחים. |
| public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) | פעולה שמופעלת כשהמשתמש בוחר גודל או צבע והיא משנה באחסון הפנימי את ההגדרות לפי מה שהמשתמש בחר. |
| public void onNothingSelected(AdapterView<?> adapterView) | פעולה שנקראת מתי שלוחצים על אחד הספינרים ולא בוחרים כלום, (לא כתוב בה כלום אבל חייבים לממש אותה..) |

**SmsActivity:**

מחלקה של activity של שליחת sms למפתח שמתחברת לקובץ xml – activity\_sms.xml ובה המשתמש שולח הודעת sms למפתח.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| SharedPreferences sp | תכונה שאחראית להתחבר לאחסון הפנימי ולכתוב שם נתונים |
| SharedPreferences.Editor editor | תכונה שדרכה עורכים בקובץ באחסון הפנימי |
| EditText message | אובייקט שבו כותבים את ההודעה למפתח |
| Button sendSms | כפתור לשליחת sms. אם אין הרשאה לשליחת sms, האפליקציה מבקשת הרשאה ואם משתמש אישר הsms נשלח |
| boolean isGranted | משתנה שאומר אם ניתנה הרשאה או לא |
| protected void onCreate(Bundle savedInstanceState) | פעולה מובנת של המערכת שמופעלת כשהאקטיביטי נפתח. |
| public void init() | פעולה שמאתחלת את האובייקטים |
| public boolean onOptionsItemSelected(@NonNull MenuItem item) | פעולה שמופעלת כשלוחצים על הכפתור חזרה שנמצא למעלה בAction bar. |
| public void onClick(View v) | פעולה שמופעלת כשלוחצים על הכפתור sendSms והיא קוראת לפעולה permission() ואם יש הרשאה הפעולה קוראת ל sendSmsFunction(). |
| public void dialog() | פעולה שבונה דיאלוג הסבר אם המשתמש לא נתן הרשאה ומנסה עוד פעם לשלוח הודעה. |
| public void sendSmsFunction() | פעולה ששולחת sms למפתח. |
| public void permission() | פעולה שמטפלת בבקשת הרשאה, בפעם הראשונה היא מפעילה את הפעולה המובנת requestPermissions().  אם המשתמש דוחה את הבקשה ומנסה לשלוח שוב קוראים לפעולה dialog() להסביר למשתמש למה הוא לא יכול לשלוח את ההודעה.  אם הוא דוחה עוד פעם מפנים אותו להגדרות המערכת של האפליקציה  (תרשים של run time permission מצורף בנספחים). |
| public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) | פעולה שמופעלת אחרי בקשת הרשאה משנה את המשתנה isGranted. |

קבצי קוד – מחלקות העצמים

**Record:**

מחלקה המייצגת שיא.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| private int move | תכונה שמייצגת את מספר המהלכים שלקח למשתמש לפתור. |
| private String time | תכונה שמייצגת את הזמן שלקח לפתור. |
| private String date | תכונה שמייצגת את התאריך של אותו יום שבו המשתמש פתר. |
| public Record(int move, String time, String date) | פעולה בונה. |

**RecordHelper:**

מחלקה שיורשת מ SQLiteOpenHelper ומיועדת להתחברות לבסיס נתונים מקומי: .

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| public static final String DATABASENAME = "records.db" | תכונה שמייצגת את השם של הבסיס נתונים. |
| public static String table\_records | תכונה שמייצגת את שם הטבלה שאליה אנחנו רוצים לגשת. |
| public static final int DATABASEVERSION = 1 | תכונה שמייצגת את הגרסה הנוכחית של הבסיס נתונים. |
| public final String COLUMN\_MOVE = "move" | תכונה שמייצגת את העמודה של המהלכים בטבלה. |
| public final String COLUMN\_TIME = "time" | תכונה שמייצגת את העמודה של הזמן שלקח לפתור בטבלה. |
| public final String COLUMN\_DATE = "date" | תכונה שמייצגת את העמודה של התאריך שבו היה השיא בטבלה. |
| SQLiteDatabase database | מופע של המחלקה SQLiteDatabase ואיתו ניגשים לבסיס נתונים. |
| private static String CREATE\_TABLE\_RECORD | תכונה מסוג מחרוזת שאליה נשרשר את הפקודה של sql ליצירת טבלה. |
| private String orderBy | תכונה שמייצגת את הסדר שבו נרצה לסדר את השיאים, לפי הזמן או לפי המהלכים. |
| String[] allColumns = { COLUMN\_MOVE, COLUMN\_TIME, COLUMN\_DATE}; | מערך של string שבו יש את כל שמות העמודות בטבלה. |
| public RecordHelper(Context context, final String table\_records,String orderBy) | פעולה בונה שבתוכה יוצרים את הפקודה בsql ליצירת טבלה. |
| public void onCreate(SQLiteDatabase db) | פעולה מובנת שנקראת כשרוצים ליצור טבלה חדשה או להתחבר לטבלה קיימת בבסיס נתונים. |
| public void open() | פעולה שנותנת הרשאת כתיבה בבסיס נתונים. |
| public Record createRecord(Record r) | פעולה שמקבלת שיא ומוסיפה אותו לבסיס נתונים. |
| public ArrayList<Record> getAllRecord() | פעולה שמחזירה את כל השיאים. |

**RecordAdapter:**

מחלקה שיורשת מ RecyclerView.Adapter<RecordAdapter.RecordViewHolder>

ואחראית לסדר את המערך שיאים בתוך RecyclerView שיציג אותו בצורה גרפית יפה.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| private final Context context | תכונה שמייצגת את המקום שבו צריך לארגן את הRecyclerView. |
| private final List<Record> recordList | תכונה שמייצגת את הרשימה של השיאים. |
| public RecordAdapter(Context context, List<Record> recordList) | פעולה בונה. |
| public RecordViewHolder onCreateViewHolder(ViewGroup parent, int viewType) | פעולה שיוצרת אובייקט שיסדר בתוך הRecyclerView את השיאים לפי איך שמוגדר בקובץ records\_layout.xml |
| public void onBindViewHolder(RecordViewHolder holder, int position) | פעולה שנקראת כמספר השיאים שנמצאים ברשימה ומכניסה אותם אחד אחרי השני בRecyclerView. |
| public int getItemCount() | פעולה שמחזירה את הגודל של הרשימת שיאים. |
| public RecordViewHolder(View itemView) | פעולה שנמצאת בתוך הclass RecordViewHolder ומאתחלת עבור כל View של שיא את הרכיבים שנמצאים בתוכו. |

**Square:**

מחלקה המייצגת ריבוע בלוח שעליו יכול להיות אריח, בלוח של 4 על 4 לדוגמא יש 16 Squares.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| BoardGame boardGame | תכונה שמייצגת את הלוח שבו נמצא הריבוע. |
| Tile tile | תכונה שמייצגת את האריח שנמצא על הריבוע או null אם על הריבוע נמצא החור במשחק. |
| float x,y,w,h | תכונות שמייצגות את המיקום של הריבוע והאורך/רוחב שלו: מיקום בx,y ואורך/רוחב בw,h. |
| int number | תכונה שמייצגת את המספר של הריבוע, הריבועים ממוספרים מ1 עד גודל הלוח לפי סידורם בלוח. |
| public Square(BoardGame boardGame,float x,float y,float w,float h,Tile tile,int number) | פעולה בונה. |
| public boolean didXAndYInSquare(float xo, float yo) | פעולה שמקבלת x וy ובודקת אם הם נמצאים על הריבוע. |

**Tile:**

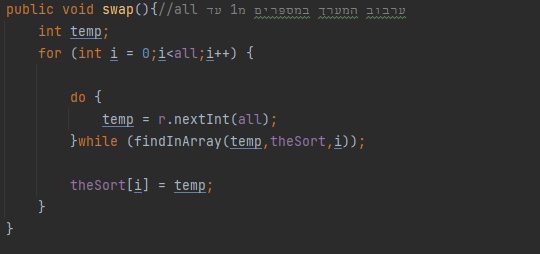
מחלקה המייצגת אריח בלוח. כל אריח נמצא על ריבוע(Square) אחד ויש מקום אחד שחסר אריח בשביל החור, לדוגמא אם הלוח בגודל 4 על 4 אז יש 15 Tiles.

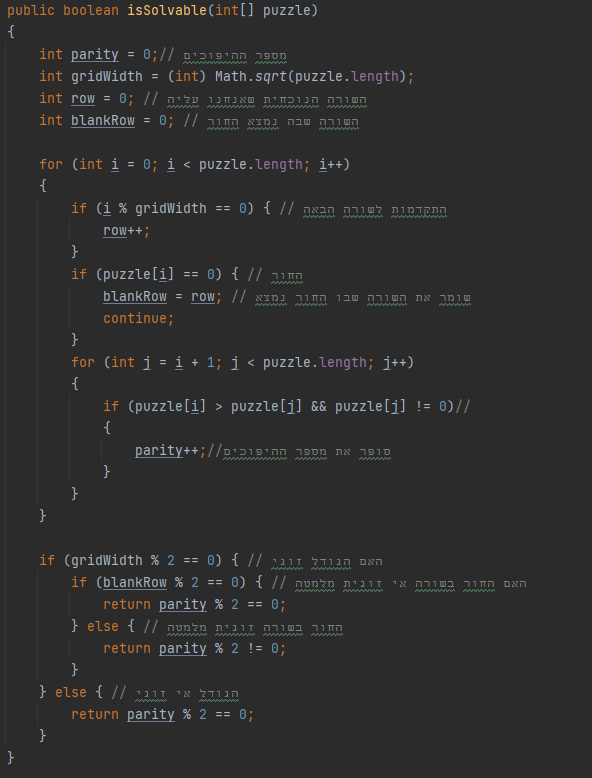
|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| BoardGame boardGame | תכונה שמייצגת את הלוח שבו נמצא האריח. |
| float x,y,w,h | תכונות שמייצגות את המיקום של האריח והאורך/רוחב שלו: מיקום בx,y ואורך/רוחב בw,h. |
| int color | תכונה שמייצגת את הצבע של האריח. |
| int number | תכונה שמייצגת את המספר שעל האריח, בסידור הסופי כל אריח יהיה על ריבוע שמספרו כמספר האריח. |
| Paint p, paintTitle | תכונות שמייצגות את הכלים שאיתם נצייר את האריחים ואת המספרים שעליהם. |
| public Tile(BoardGame boardGame, float x, float y, float w, float h, int num, int color) | פעולה בונה, בתוכה יש קריאה בפעולה init(). |
| init() | פעולה שמאתחלת ומגדירה את התצורה של הp ושל ה paintTitle. |
| public void draw(Canvas canvas) | פעולה שמציירת את האריח עם המספר שלו עליו. |
| public void slide(Square to) | פעולה שמקבלת ריבוע ומעבירה את האריח הנוכחי לריבוע הזה כלומר משנה את הx והy שלו לx והy של האריח. |

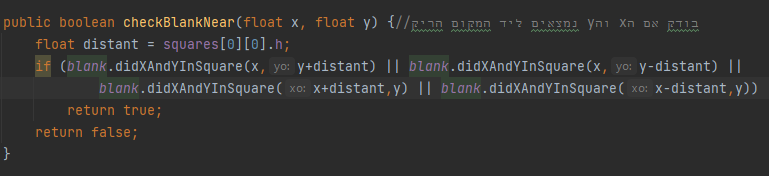
**BoardGame:**

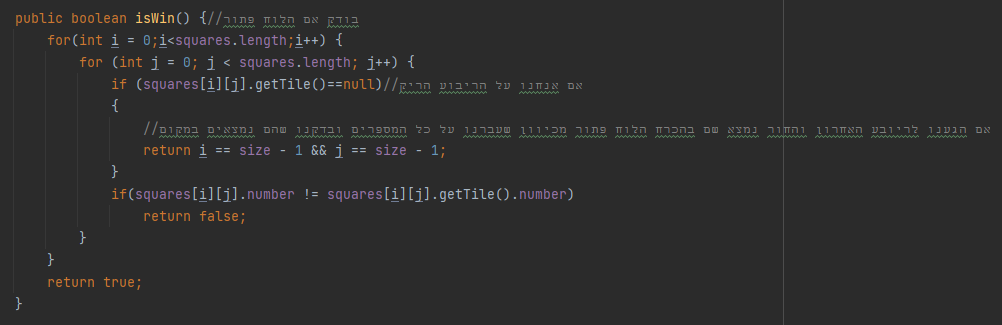
מחלקה שיורשת מView ומשמשת בתור הלוח של המשחק, בה נמצא כל הלוגיקה של הלוח והשינויים בו.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| boolean ifOne = true | תכונה שאומרת אם זה פעם ראשונה שמציירים את הלוח. |
| int size,color | תכונות שמייצגות את הגודל של הלוח והצבע של האריחים שמקבלים בפעולה הבונה ולפיהם בונים את הלוח. |
| int[] theSort | תכונה מסוג מערך שבתוכה נשמר הסדר של הבלגון של הלוח. |
| Random r | תכונה שאחראית להגריל את המספרים של הבלגון. |
| int all | תכונה שמייצגת את הגודל המלא של הלוח לדוגמא אם הלוח הוא 4 על 4 אז all = 16. |
| Tile[][]tiles | תכונה מסוג מערך דו ממדי של אריחים שמייצגת את כל האריחים שעל הלוח, אם הלוח בגודל 4 על 4 אז יש במערך 16 אריחים ובמקום אחד יש Null. |
| Square[][]squares | תכונה מסוג מערך דו ממדי של ריבועים שמייצגת את כל הריבועים שעל הלוח, אם הלוח בגודל 4 על 4 אז יש במערך 16 ריבועים. |
| Context context | הקישור למקום שבו צריך לבנות את הלוח, במקרה הזה בmain activity. |
| static Square blank | תכונה מסוג Square שמייצגת את הריבוע הריק, היא סטטית מכיוון שרוצים להשתמש בה גם במחלקה Game -. |
| static Time time | מופע של המחלקה Time שיורשת מThread ואחראי על הטיימר של המשחק, הוא סטטי מכיוון שרוצים להשתמש בו גם במחלקה Game -. |
| public BoardGame(Context context,int size,int color) | פעולה בונה. |
| protected void onDraw(Canvas canvas) | פעולה מובנת של View ובה מציירים על הלוח עם ,Canvasקוראת לdrawBoard(). |
| public void resetArr(int[] arr) | פעולה שמאתחלת את המערך שכל אבריו יהיו 1-. |
| findInArray(int n,int[] arr,int temp) | פעולה שמקבלת מספר, מערך ומיקום עכשווי במערך ובודקת אם המספר לא נמצא במערך עד המקום העכשווי. |
| public void swap() | פעולה שאחראית לערבב את המערך על ידי זה שהיא מגרילה כל פעם מספר, בודקת שהוא לא נמצא כבר במערך בעזרת הפעולה findInArray(). אם הוא עוד לא נמצא היא מכניסה אותו למערך.(**מצורף קוד מתחת לטבלה**) |
| public boolean isSolvable(int[] puzzle) | פעולה עם אלגוריתם מתקדם שמקבלת מערך חד ממדי עם הסדר של הבלגון ובודקת אם הבלגון פתיר.  הסבר בקצרה: הפעולה קודם כל מוצאת את מספר ההיפוכים במערך (היפוך אחד הוא מספר אחד שיותר קטן ממספר אחר שנמצא לפניו במערך.  לדוגמא במערך :  [2 1 3 4 5 6 7 8 9 10 11 12 13 14 15 X] שמסודר מימין לשמאל המספר 1 נצמא אחרי המספר 2 ולכן זה נחשב היפוך אחד).  בנוסף הפעולה מוצאת באיזה שורה נמצא המקום הריק.לפי שתי נתונים אלו היא בודקת בעזרת מספר כללים אם המערך פתיר או לא.  הסבר מורחב לאלגוריתם באתר - <https://www.geeksforgeeks.org/check-instance-15-puzzle-solvable>  (**מצורף קוד מתחת לטבלה**) |
| public void buildArrayTheSort() | פעולה שמשתמשת בפעולות,resetArr() swap() וisSolveable() בשביל ליצור את המערך הסופי של הבלגון. בסדר הזה: היא נכנסת ללולאת do while ומאפסת את המערך עם resetArr(), לאחר מכן מערבבת אותו עם swap() ולאחר מכן בודקת אם הוא פתיר עם isSolvable() וכל עוד הוא לא פתיר היא חוזרת על התהליך.  (**מצורף קוד מתחת לטבלה**) |
| public void drawBoard(Canvas canvas) | הפעולה שאחראית לצייר את הלוח, בפעם הראשונה היא קוראת ל buildArrayTheSort() בשביל ליצור את הבלגון של הלוח, לאחר מכן היא מאתחלת את המערכים tiles ו squaresומציירת את כל האריחים על הלוח אחד אחרי השני.  בנוסף קוראת לפעולה blank() בשביל למצוא את החור. בפעמים האחרות שהפעולה נקראת היא רק מציירת את כל האריחים לפי המיקומים החדשים שלהם במידה והלוח השתנה. |
| public Square blank() | פעולה שעוברת על כל הריבועים  ( (squaresומוצאת את המקום הריק כלומר המקום שבו אין אריח(tile). |
| public void slide(Square from,Square to) | פעולה שמקבלת שני ריבועים, אחד של האריח שלחצו עליו והשני של המקום הריק. הפעולה מזיזה את האריח לריבוע של המקום הריק בשתי שלבים:  1. היא מגדירה שהתכונה tile באובייקט to יהיה הtile שנמצא באובייקט from  2. היא לקרוא לפעולה slide() של Tile בשביל לשנות את הx והy של האריח למיקום של הריבוע החדש(to). |
| public boolean checkBlankNear(float x, float y) | פעולה שמקבלת x וy ובודקת אם הם נמצאים ליד הריבוע הריק עם הפעולה didXAndYInSquare() של Square.  (**מצורף קוד מתחת לטבלה**) |
| public boolean isWin() | פעולה שבודקת אם הלוח פתור בכך שהיא עוברת על כל הריבועים ובודקת אם המספר שלהם תואם למספר של האריחים שעליהם.  (**מצורף קוד מתחת לטבלה**) |
| public Square findSquare(float x, float y) | פעולה שמקבלת x וy ומחזירה את הsquare שנמצא שם. |





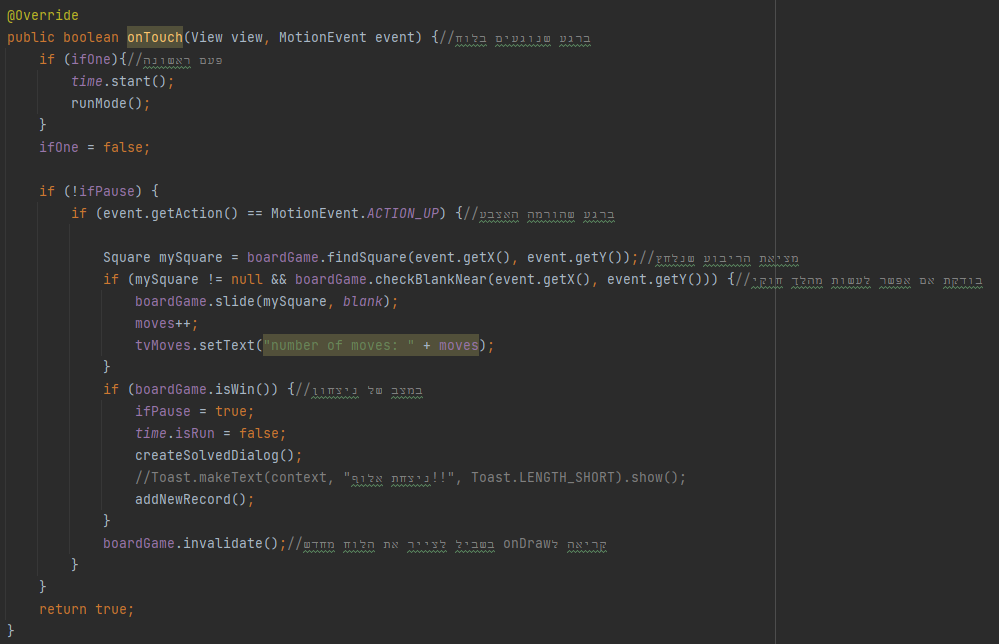




**Game:**

המחלקה המרכזית של הפרויקט, היא מייצגת את המשחק. היא אחראית לכל הלוגיקה שלו והשינויים במשחק.

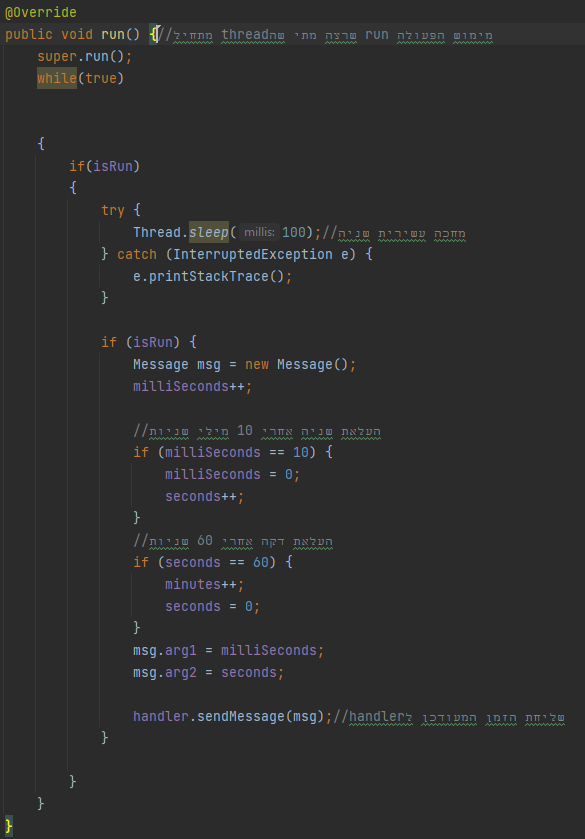
|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| private BoardGame boardGame | תכונה שמייצגת את לוח משחק. |
| Dialog solvedD | הדיאלוג שקופץ כשפותרים את הלוח. |
| Button btnSolved | הכפתור שנמצא בדיאלוג solved,  כשלוחצים עליו הוא מאתחל את המשחק. |
| TextView tvTime, tvMoves | הרכיבים שעליהם כותבים את הטיימר של המשחק ואת מספר המהלכים במשחק. |
| private final Context context | תכונה שמייצגת את המקום שבו נרצה לממש את המשחק (במקרה הזה בmainActivity). |
| SharedPreferences getSetting | תכונה שאיתה ניקח את ההגדרות שנשמרו באחסון הפנימי. |
| LinearLayout l | תכונה שמתחברת לlinearLayout שהגדרנו בactivity\_main.xml ובו נשים את האובייקט של הלוח(boardGame). |
| boolean ifOne = true | תכונה שבודקת אם זו פעם ראשונה שנוגעים בלוח, אם כן יופעל הthread Time ונעבור למצב ריצה(run mode). |
| Boolean ifStart=false, ifPause = true | תכונות שמעדכנות אותנו באיזה מצב אנחנו במשחק ומשתנות לפי המצב הנוכחי. ifStart מייצג אם המשחק כבר התחיל, ifPause מייצג אם המשחק נמצא בעצירה או לא. |
| int moves | תכונה שמייצגת את מספר המהלכים במשחק. |
| int sizeOfBoard | תכונה שמייצגת את גודל הלוח שהוגדר. |
| int colorOfTile | תכונה שמייצגת את צבע האריחים שהוגדר. |
| Handler handler | תכונה שאחראית לקבל את ההתקדמות של הטיימר מהThread Time. |
| RecordHelper recordHelper | מופע של המחלקה RecordHelper שאחראי להתחבר לבסיס נתונים ולהכניס אליו שיאים. |
| public Game(Context context) | פעולה בונה שמאתחלת את כל האובייקטים. היא קוראת לstartMode() ולupdate() בשביל לעדכן את ההגדרות ול createBoardGame() בשביל ליצור את הלוח של המשחק. |
| public void createBoardGame() | פעולה שאחראית ליצור את לוח המשחק.  ראשית היא מוציאה את ההגדרות של הצבע והגודל מהאחסון הפנימי, אם זו פעם ראשונה שהמשחק מופעל היא מגדירה כברירת מחדל את הגודל ל4 על 4 ואת הצבע לורוד.  לאחר מכן היא בונה את הלוח בהתאם לרוחב המסך של המכשיר עליו מופעלת האפליקציה. |
| public void doHandler() | פעולה שאחראית לקבל messages מהThread Time שבתוכם יש את ההתקדמות של הטיימר ומתי שה ifStart שווה לfalse היא מאתחלת את הטיימר  ל 00:00.0. |
| public void startMode() | פעולה שמכניסה את המשחק לstart mode כלומר למצב התחלתי לפני שהמשתמש מתחיל לשחק. במצב זה מאפסים את כל הרכיבים של המשחק כמו הטיימר או המהלכים ומאתחלים את האובייקט time. |
| public void runMode() | פעולה שמכניסה את המשחק למצב ריצה כלומר המצב בו המשחק רץ והמשתמש פותר את הלוח. במצב הזה ifStart = true ו  ifPause = false. |
| public void stopMode() | פעולה שמכניסה את המשחק למצב עצירה כלומר מצב שבו השחקן לא משחק ואם הוא רוצה להמשיך לשחק הוא צריך ללחוץ על הכפתור continue. במצב הזה  ifStart = true ו ifPause = true. |
| public boolean onTouch(View view, MotionEvent event) | פעולה מובנת של View ומופעלת על האובייקט .boardGame  היא אחראית לטפל במצב בו המשתמש נוגע בלוח.  בפעם הראשונה היא מריצה את הטיימר וקורא לפעולה runMode().  בפעמים הבאות היא רק בודקת אם המשתמש לחץ על ריבוע שלידו נמצא החור, אם כן היא מזיזה את האריח שעל הריבוע הזה לחור עם הפעולה slide()של boardGame. לאחר מכן היא בודקת אם הלוח פתור עם הפעולה isWin()של boardGame, אם הלוח פתור היא מקפיצה את הדיאלוג solved עם הפעולה createSolvedDialog() ומכניסה את השיא החדש לבסיס הנתונים בעזרת הפעולה addNewRecord().(**מצורף קוד מתחת לטבלה**) |
| public void addNewRecord() | פעולה שמתחברת לבסיס נתונים ומכניסה לו שיא חדש. |
| public void update() | פעולה שמעדכנת את ההגדרות צבע וגודל שנמצאות באחסון הפנימי. בנוסף מעדכנת את הסידור של השיאים בבסיס נתונים. |
| public void createSolvedDialog() | פעולה שיוצרת דיאלוג, מאפיינת אותו ומציגה אותו על המסך כשהלוח פתור. |



**Time:**

מחלקה שיורשת מThread ואחראית לקדם את הטיימר של הזמן.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| boolean isRun | תכונה שאומרת אם המשחק רץ. |
| Handler handler | תכונה שמקבלת את ההתקדמות של הטיימר ושלוחת אותו לHandler במחלקה game בשביל שיקדם את הטיימר. |
| int milliSeconds, seconds, minutes | תכונות שמייצגות דקות, שניות ועשירית שניה. |
| public Time() | פעולה בונה שמאתחלת את המשתנים milliSeconds, seconds, minutes ל0 |
| public void run() | פעולה מובנת שחייבים לממש כשיורשים מThread והיא מופעלת מתי שקוראים לפעולה start() שלThread.  בתוכה בודקים אם המשחק רץ ואם כן מחכים 0.1 שניות ואז מקדמים את המשתנים ושולחים את ההתקדמות בתוך message לhandler. (**מצורף קוד מתחת לטבלה**) |

  
}

קבצי קוד – מחלקות עזר

**ReceiverNotification:**

מחלקה שיורשת מ BroadcastReceiverוהיא מופעלת מתי שAlarmManager קורא לBroadcast, כל יום בשעה 20:30.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| public void onReceive(Context context, Intent intent) | פעולה שמופעלת כשהBroadcast נקרא, ובתוכה יש קריאה לפעולה createNotification(). |
| public void createNotification(Context context) | הפעולה שאחראית ליצור את ההתראה. היא יוצרת את כל הרכיבים ובונה את ההתראה. לאחר מכן היא שולחת אותה לService של ההתראות של המערכת. |

**מדריך למשתמש**

**התקנה**- כדי להתקין את האפליקציה על טלפון אנדרואיד נדרש לחבר את הטלפון לאנדרואיד סטודיו ולהריץ את האפליקציה עליו או לייצא קובץ apk מאנדרואיד סטודיו ולהתקין אותו על הטלפון.

**גרסאות עליהם נבדק הפרויקט:**

סוגי מכשירים – Xiaomi redmi note 9, Sumsung Galaxy A52

סוגי אמולטורים - Pixel\_2\_API\_28

**תיאור אופן השימוש בפרוייקט:**

לוחצים על האייקון של האפליקציה והמשחק פותח מסך שבו מוצג השם של האפליקציה עם אנימציה נחמדה ולאחר מספר שניות המשחק נפתח כשהלוח בגודל 4 על 4 והאריחים בצבע ורוד, עכשיו יש כמה דברים שאפשר לעשות:

1. אפשר פשוט להתחיל לשחק במשחק.

**חוקי המשחק**:

יש לוח בגודל מסוים והאריחים שעליו מבולגנים.

המטרה היא לסדר אותם לפי המספרים שעליהם מ1 עד המספר הכי גדול.

עכשיו איך בעצם תזיז את החלקים? אז ככה: בלוח יש מקום אחד שאין עליו אריח וזה ה"מקום הריק", רק אליו ניתן להזיז אריח. אם אתה רוצה להזיז אריח אל ה"מקום הריק", לחץ עליו. כל הזזה של אריח מעלה את מספר המהלכים. יש גם טיימר שסופר כמה זמן לקח לך לפתור את הלוח, הוא פועל ממתי שהתחלת לפתור. ברגע שהצלחת לפתור קופצת הודעה שמודיע לך שפתרת את המשחק בהצלחה, בשביל לשחר שוב לחץ על ההודעה. אחרי שפתרת את הלוח, הזמן וכמות המהלכים שהצטרכת לפתרון הלוח נשמרים במערכת ותוכל לראות את השיאים לאחר מכן.

1. אחרי שהתחלת לשחק, תוכל תמיד לעצור את הזמן על ידי לחיצה על הכפתור pause שיהפוך לcontinue ואם תרצה להמשיך במשחק תצטרך ללחוץ על continue והמשחק ימשיך.

בנוסף אם תצא בטעות מהאפליקציה בלי לסגור אותה, אל דאגה! המשחק ייעצר ותוכל להמשיך אותו בדיוק מאותה נקודה שהפסקת.

1. אם הבלגון לא מוצא חן בעיניך או שתרצה להתחיל משחק חדש בלי לגמור את המשחק הקודם, תוכל גם לבלגן את הלוח שוב על ידי לחיצה על הכפתור start.
2. תוכל גם לגשת למסך ההגדרות על ידי לחיצה על שלוש הנקודות בפינת המסך ולחיצה על setting. שם תוכל לשנות את הגודל של הלוח לשלל גדלים מ3 על 3 עד 10 על 10 (אזהרה, פתירת לוח של 10 על 10 זה דבר מאוד מתיש ומייאש 😉).

תוכל גם לשנות את הצבע של האריחים לשלל צבעים שונים.

לאחר מכן בעזרת לחיצה על החץ למעלה או על כפתור החזרה המובנה, תוכל לחזור למסך הראשי עם ההגדרות החדשות. אם תצא מהאפליקציה, כשתחזור הלוח יהיה לפי ההגדרות האחרונות שהגדרת.

1. אפשר גם לראות את השיאים שלך על ידי לחיצה על השלוש נקודות בפינת המסך ואז לחיצה על record. ייפתח דיאלוג שמציג את 10 השיאים ההכי טובים מסודרים לפי כמות מהלכים או לפי הזמן שלקח לפתור אותם. ניתן לבחור את צורת הסידור על ידי לחיצה על הכפתור הנמצא בחלק העליון של הדיאלוג.
2. בנוסף תוכל גם לשלוח הודעה למפתח על ידי לחיצה על השלוש נקודות בפינת המסך ולחיצה על contact us. ייפתח מסך בו תוכל לכתוב הודעה ולשלוח אותה למפתח. לאחר שתשלח את ההודעה האפליקציה אוטומטית תחזיר אותך למסך הראשי של האפליקציה.

לא ניתן לסובב את המסך מתי שהאפליקציה פתוחה מכיוון שזה יהרוס את העיצוב של המשחק.

בנוסף לכל זה מוגדר לאפליקציה לשלוח אליך כל יום ב20:30 תזכורת לשחק במשחק עם הודעה שתעודד אותך לשחק כל יום ולפתח את המוח שלך!

**סיכום אישי/רפלקציה**

בהתחלה כשניגשתי להתחיל את הפרוייקט זה נראה לי כמשהו שגדול ממני ולא ראיתי את עצמי יושב ומתכנת ממש משחק אמיתי, גם ידעתי שבסופו של דבר זה יחשב עוד 5 יחידות שלמות לבגרות ותיארתי לעצמי שאם זה שווה 5 יחידות זה בטח המון עבודה.

בסוף החלטתי שאני לוקח את זה על עצמי והתחלתי לעבוד.

אחרי שסיימנו ללמוד את החומר בכיתה התחלתי לחשוב על רעיון שיעניין אותי, בסוף הגעתי למסכנה שלבנות משחק שאולי אין בו הרבה חידושים טכנולוגיים אבל יש בו הרבה אלגוריתמיקה וחשיבה זה יכול להיות משהו שאני יאהב לעשות ולכן הלכתי על זה. בהתחלה ניסיתי להבין במה אני אצטרך להשתמש והגעתי למסכנה אחרי קריאה באינטרנט והתייעצויות שבבניית משחק כדאי ליצור מחלקה שיורשת מview ולעבוד עם זה. לכן הלכתי ללמוד את הנושא ולהתנסות בו. בנוסף קראתי בכל מיני מקומות על המשחק הספציפי הזה בשביל להבין איך אני מיישם אותו מבחינת האלגוריתמיקה, חקרתי את הנושא לעומק.

בזכות הפרוייקט למדתי הרבה דברים חדשים בתכנות ובandroid studio ואני אוכל ליישם אותם בפרוייקטים אחרים שארצה להכין, בנוסף הפרוייקט לימד אותי על עמידה ביעדים ולא להתייאש גם כשיש הרבה קשיים ואני מקווה שזה יעזור לי גם בדברים אחרים בחיים שלי.

הבנתי שבלי עזרה מהמורה ומחברים ובעיקר מהאינטרנט היה יכול להיות הרבה יותר קשה ואני מודה למורה שלי שהנחת אותי בכל מהלך הפרוייקט.

במבט לאחור אני חושב שאם הייתי מתחיל יותר מוקדם להשקיע בפרוייקט היה יכול להיות הרבה פיצ'רים שעכשיו כבר אין לי זמן להכניס וחבל.

אני חושב שאם היה לי ידע יותר מורחב בשפה האנגלית הייתי מתקדם יותר מהר בפרוייקט ואולי לא נתקע על דברים סתם ולכן אני מייעץ לכל מי שרוצה לעשות הנדסת תוכנה שיבנה לו בסיס טוב באנגלית וכך יהיה לו יותר קל בהרבה מובנים.

**רשימת מקורות/ביבליוגרפיה**

1. ויקיפדיה - חידת ה-15

<https://he.wikipedia.org/wiki/%D7%97%D7%99%D7%93%D7%AA_%D7%94-15>

1. אלעד שניר (2011), "חידת ה - 15" מתוך: "מאמרים"

<https://www.articles.co.il/article/97033/%D7%97%D7%99%D7%93%D7%AA+%D7%94-+15>

1. מאמר מערכת (2021), How to check if an instance of 15 puzzle is solvable? מתוך: "GeeksForGeeks"

<https://www.geeksforgeeks.org/check-instance-15-puzzle-solvable/>

1. Caio Lopes (לא ידוע שנה), Sliding-Tile-Puzzle פרוייקט מתוך: "Github"

<https://github.com/caiolopes/Sliding-Tile-Puzzle/tree/74b263c4e10fd12726723fc308388b89b7759c00>

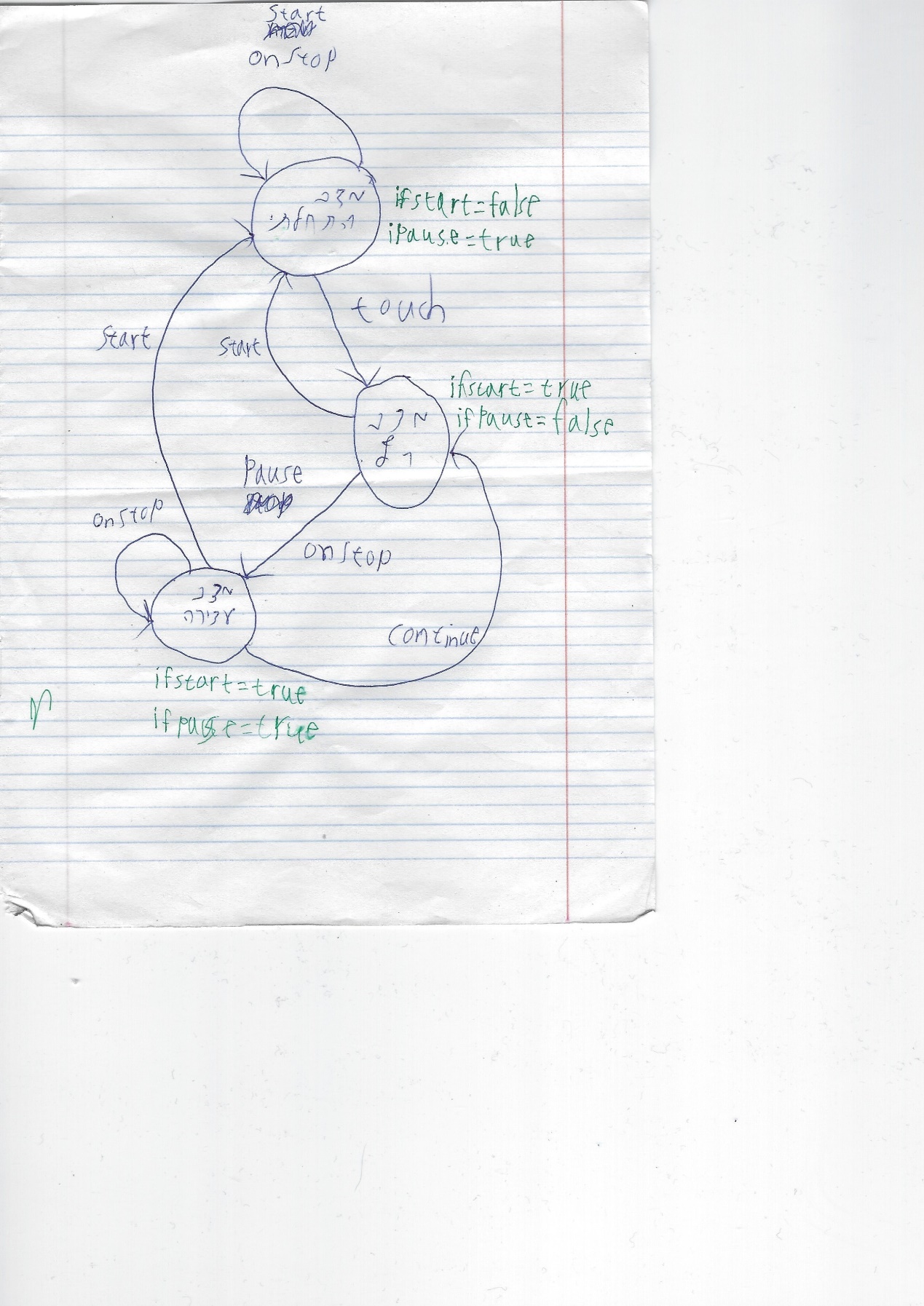
**נספחים**

קישור לתיקייה עם כל מחלקות הקוד:

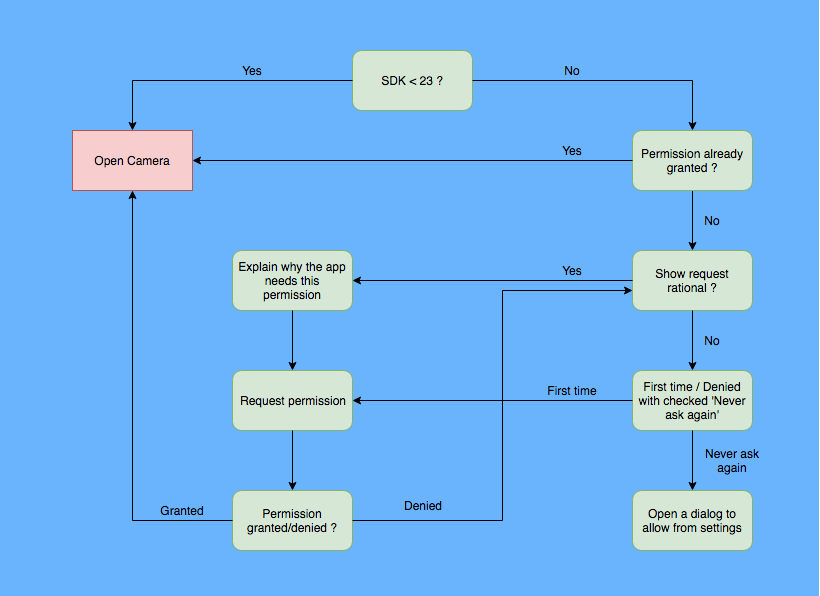
<https://drive.google.com/drive/folders/1-cQm8ACZdajVH_H6zLWH6Sn8eFBL1jvV?usp=sharing>

דף שכתבתי בכתב יד שמתאר את המעברים בין המצבים של המשחק

(start mode, stop mode…) בצורה של מודל חישובי.



תרשים של run time permission (תהליך בקשת הרשאה)



Send sms

**MainActivity.java**

package com.example.board;  
  
import androidx.activity.result.ActivityResult;  
import androidx.activity.result.ActivityResultCallback;  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
  
import android.annotation.SuppressLint;  
import android.app.Activity;  
import android.app.AlarmManager;  
import android.app.Dialog;  
import android.app.PendingIntent;  
import android.content.Intent;  
import android.content.SharedPreferences;  
  
  
import android.net.MacAddress;  
import android.os.Bundle;  
  
import android.view.Menu;  
import android.view.MenuInflater;  
import android.view.MenuItem;  
  
import android.view.View;  
  
import android.view.WindowManager;  
import android.widget.Button;  
import android.widget.ProgressBar;  
  
  
import java.util.ArrayList;  
import java.util.Calendar;  
  
import java.util.List;  
  
  
public class MainActivity extends AppCompatActivity implements View.OnClickListener{  
  
 static Button *btnStart*, *btnPause*;  
 Button btnOrder;  
  
 SharedPreferences getSetting;  
 Dialog recordsD;  
  
 boolean passToIntent = false;//בודק אם עשינו onStop דרך intent  
  
  
 RecyclerView recyclerView;  
  
 AlarmManager alarmManager;  
 PendingIntent pendingIntent;  
  
 Game game;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setTheme(R.style.*Theme\_Board*);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);//הופף את הactivity למסך מלא  
  
 setContentView(R.layout.*activity\_main*);  
  
 init();  
  
 }  
 //פעולה שמאתחלת את האובייקטים ומעדכנת את הSharedPreferences  
 private void init(){  
  
 getSetting = getSharedPreferences("data",0);  
 SharedPreferences.Editor editor = getSetting.edit();  
  
 if (getSetting.getString("orderBy",null) == null) {  
 editor.putString("orderBy", "move");  
 editor.commit();  
 }  
  
 if(getSetting.getBoolean("firstLaunch",true)) {  
 startNotification();  
 editor.putBoolean("firstLaunch", false);  
 editor.commit();  
 }  
  
 *btnStart* = findViewById(R.id.*btnStart*);  
 *btnStart*.setOnClickListener(this);  
 *btnPause* = findViewById(R.id.*btnPause*);  
 *btnPause*.setOnClickListener(this);  
  
 game = new Game(this);  
 game.update();  
  
 }  
  
 //פעולה שיוצרת את התפריט  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
  
 MenuInflater inflater = getMenuInflater();  
 inflater.inflate(R.menu.*menu\_main*,menu);  
 return true;  
 }  
  
  
 //כשלוחצים על אפשרות בתפריט  
 @Override  
 public boolean onOptionsItemSelected(@NonNull MenuItem item) {  
 if (item.getItemId() == R.id.*setting*) {//כניסה להגדרות  
 Intent intent = new Intent(MainActivity.this,SettingActivity.class);  
 passToIntent = true;  
 StartForResult.launch(intent);  
 }  
  
 else if (item.getItemId() == R.id.*records*) {//פתיחת הטבלת שיאים  
 if(game.ifStart)  
 game.stopMode();  
 createRecordsDialog();  
 }  
  
 else if (item.getItemId() == R.id.*contact*) {//פתיחת המסך של ה"contact us"  
 Intent intent = new Intent(MainActivity.this,SmsActivity.class);  
 passToIntent = true;  
 StartForResult.launch(intent);  
 }  
 return true;  
  
 }  
  
 //פעולה שמופעלת כשלוחצים על כפתור  
 @SuppressLint({"SetTextI18n", "Range"})  
 @Override  
 public void onClick(View v) {  
  
 if (v == *btnStart*) {//אם לחצו על start  
 game = new Game(this);  
 game.tvTime.setText("00:00.0");  
  
 }  
 else if (v == *btnPause*)//אם לחצו על pause  
 {  
 if (!game.ifPause)  
 game.stopMode();  
 else  
 game.runMode();  
  
 }  
 else {//טיפול בכפתור הsolved  
 game.solvedD.dismiss();  
 game = new Game(this);  
  
 }  
  
 }  
 ActivityResultLauncher<Intent> StartForResult = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(),  
 new ActivityResultCallback<ActivityResult>() {  
 @Override  
 public void onActivityResult(ActivityResult result) {//חזרה מintent  
 if (result.getResultCode() == Activity.*RESULT\_OK*) {  
  
  
 passToIntent = false;  
  
 game = new Game(MainActivity.this);//אתחול המשחק  
  
 }  
 }  
 });  
  
  
 //הגדרת ההתראות  
 public void startNotification(){  
  
 Intent intent = new Intent(this, receiverNotification.class);  
  
 //שליחה לברודקסט receiverNotification.class  
 pendingIntent = PendingIntent.*getBroadcast*(  
 this.getApplicationContext(), 234324243, intent, PendingIntent.*FLAG\_IMMUTABLE*);  
 alarmManager = (AlarmManager) getSystemService(*ALARM\_SERVICE*);//התחברות לservice של המערכת  
  
 Calendar calendar = Calendar.*getInstance*();  
  
 calendar.set(Calendar.*HOUR\_OF\_DAY*, 8);//הגדרת השעה של ההתראה ל20:30  
 calendar.set(Calendar.*MINUTE*, 30);  
 calendar.set(Calendar.*SECOND*, 0);  
  
  
 alarmManager.setRepeating(AlarmManager.*RTC\_WAKEUP*, calendar.getTimeInMillis(),  
 alarmManager.*INTERVAL\_DAY*, pendingIntent);//הפעלת החזרה היומית  
 }  
  
  
 @Override  
 protected void onPause() {  
 super.onPause();  
  
 if (!game.ifPause) {  
 game.stopMode();  
  
 if (passToIntent) *btnPause*.setText("pause");  
 }  
 }  
  
 //צמצום המערך שיאים ל10 שיאים הראשונים  
 public List<Record> createRecordListForShow()  
 {  
 game.recordHelper.open();  
 ArrayList<Record> list = game.recordHelper.getAllRecord();  
  
  
 //אם המערך של השיאים גדול מ10  
 if (list.size()>10) {  
 List<Record> newList = list.subList(0, 10);//לקיחת ה10 שיאים הראשונים  
 return newList;  
 }  
  
 game.recordHelper.close();  
 return list;  
  
 }  
  
 //יצירת הדיאלוג של השיאים  
 public void createRecordsDialog()  
 {  
  
 //אתחול והגדרת הדיאלוג  
 recordsD=new Dialog(this);  
 recordsD.setContentView(R.layout.*custom\_dialog\_records*);  
 recordsD.setCancelable(true);  
  
 //אתחול הrecyclerView  
 recyclerView = (RecyclerView) recordsD.findViewById(R.id.*recyclerView*);  
 recyclerView.setHasFixedSize(true);  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
  
 //יצירת adapter עם list של השיאים  
 RecordAdapter recordAdapter = new RecordAdapter(this, createRecordListForShow());  
  
 recyclerView.setAdapter(recordAdapter);  
 recordsD.show();  
  
  
 btnOrder = recordsD.findViewById(R.id.*btnOrder*);  
 if(game.recordHelper.getOrderBy().equals("time")) btnOrder.setText("order by time");  
 btnOrder.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 SharedPreferences.Editor editor = getSetting.edit();  
  
 //שינוי מסידור לפי מהלכים לסידור לפי זמן  
 if (getSetting.getString("orderBy",null).equals("move")) {  
 editor.putString("orderBy", "time");  
 btnOrder.setText("order by time");  
 }  
 else {//שינוי מסידור לפי זמן לסידור לפי מהלכים  
 editor.putString("orderBy", "move");  
 btnOrder.setText("order by move");  
 }  
 //אתחול הטבלה לפי ההגדרה החדשה  
 editor.commit();  
 //עדכון הסידור החדש והצגתו בדיאלוג  
 game.update();  
 RecordAdapter recordAdapter = new RecordAdapter(MainActivity.this,  
 createRecordListForShow());  
 recyclerView.setAdapter(recordAdapter);  
 recordsD.show();  
  
 }  
 });  
  
 }  
  
}

**SettingActivity.java**

package com.example.board;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.ActionBar;  
import androidx.appcompat.app.AppCompatActivity;  
  
  
import android.content.SharedPreferences;  
import android.graphics.Color;  
import android.os.Bundle;  
import android.view.MenuItem;  
import android.view.View;  
import android.view.WindowManager;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Spinner;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class SettingActivity extends AppCompatActivity implements AdapterView.OnItemSelectedListener {  
  
 Spinner spinnerSize, spinnerColor;  
 List<String> sizeOption, colorOption;  
  
 int choiceOfSize;  
 String choiceOfColor;  
  
  
 ArrayAdapter<String> dataAdapter;//אובייקט שמכין את הarrayList לspinner  
 SharedPreferences saveSetting;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setTheme(R.style.*Theme\_Board*);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);//הופף את הactivity למסך מלא  
  
 setContentView(R.layout.*activity\_setting*);  
  
 init();  
  
 }  
  
 private void init() {//פעולה שמאתחלת את האובייקטים ומקבלת מהSharedPreferences את ההגדרות הנוכחיות  
  
 ActionBar actionBar = getSupportActionBar();  
  
 actionBar.setDisplayHomeAsUpEnabled(true);//מראה כפתור חזרה בלמעלה של המסך  
  
  
 saveSetting = getSharedPreferences("data",0);  
  
 choiceOfSize = saveSetting.getInt("size",4);  
 choiceOfColor = saveSetting.getString("colorOfTile name",null);  
 spinnerSize = findViewById(R.id.*spinnerSize*);  
 spinnerColor = findViewById(R.id.*spinnerColor*);  
 createSpinnerOfSize();  
 createSpinnerOfColor();  
  
 }  
 @Override  
 public boolean onOptionsItemSelected(@NonNull MenuItem item) {//מה קורה כשלוחצים על הכפתור חזרה  
 if (item.getItemId() == android.R.id.*home*) {  
 this.finish();//סוגר את הactivity  
 return true;  
 }  
 return super.onOptionsItemSelected(item);  
 }  
  
  
 public void createSpinnerOfSize() {//יצירת הספינר של הגדרת הגודל של הלוח  
 sizeOption = new ArrayList<String>();  
 for (int i = 3; i <= 10; i++)  
 sizeOption.add(String.*valueOf*(i));//הכנסת הגדלים לArrayList  
  
 dataAdapter = new ArrayAdapter<String>(this, android.R.layout.*simple\_spinner\_item*, sizeOption);//יצירת המתאם עם הרשימה של הגדלים  
 dataAdapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);//שינוי העיצוב של התצוגה כשהספינר נפתח  
 spinnerSize.setAdapter(dataAdapter);  
 spinnerSize.setSelection(choiceOfSize - 3);//מכיוון שהגודל הכי קטן הוא 3 אז הבחירה צריכה להיות פחות 3  
  
  
 spinnerSize.setOnItemSelectedListener(this);  
 }  
  
 public void createSpinnerOfColor() {//יצירת הספינר של הגדרת הצבע של האריחים  
  
  
 colorOption = new ArrayList<String>();  
 colorOption.add("magenta");  
 colorOption.add("green");colorOption.add("yellow");  
 colorOption.add("black");colorOption.add("gray");  
 colorOption.add("purple");colorOption.add("red");//הכנסת סוגי הצבעים לרשימה  
  
  
 dataAdapter = new ArrayAdapter<String>(this, android.R.layout.*simple\_spinner\_item*, colorOption);  
 dataAdapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 spinnerColor.setAdapter(dataAdapter);  
  
  
  
 for (int i = 0;i<7;i++)  
 {  
 if (spinnerColor.getAdapter().getItem(i).equals(choiceOfColor)){  
 spinnerColor.setSelection(i);//מציאת הצבע שמוגדר עכשיו  
 break;  
 }  
 }  
 spinnerColor.setOnItemSelectedListener(this);  
 }  
  
 @Override  
 public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {//מתי שנבחר משהו  
 SharedPreferences.Editor editor = saveSetting.edit();  
  
 if (view.getParent() == spinnerSize) {//בודק אם נלחץ בספינר של הגדלים  
  
 String temp = adapterView.getItemAtPosition(i).toString();//מוצא את הבחירה  
 editor.putInt("size", Integer.*parseInt*(temp));//מעדכן את ההגדרה  
 editor.commit();  
  
 setResult(*RESULT\_OK*);  
 }  
 else//נלחץ בספינר של הצבעים  
 {  
 String temp = adapterView.getItemAtPosition(i).toString();  
  
 //מעדכן איזה צבע להגדיר לפי הבחירה  
 switch (temp) {  
 case "magenta":  
 editor.putInt("color",Color.*MAGENTA*);  
 editor.putString("colorOfTile name","magenta");  
 break;  
 case "green":  
 editor.putInt("color",Color.*parseColor*("#4CAF50"));  
 editor.putString("colorOfTile name","green");  
 break;  
 case "yellow":  
 editor.putInt("color",Color.*parseColor*("#FFC107"));  
 editor.putString("colorOfTile name","yellow");  
 break;  
 case "black":  
 editor.putInt("color",Color.*BLACK*);  
 editor.putString("colorOfTile name","black");  
 break;  
 case "gray":  
 editor.putInt("color",Color.*parseColor*("#8E8E8E"));  
 editor.putString("colorOfTile name","gray");  
 break;  
 case "purple":  
 editor.putInt("color",Color.*parseColor*("#5F4195"));  
 editor.putString("colorOfTile name","purple");  
 break;  
 case "red":  
 editor.putInt("color",Color.*parseColor*("#C61E1E"));  
 editor.putString("colorOfTile name","red");  
 break;  
 }  
 editor.commit();  
 setResult(*RESULT\_OK*);  
 }  
 }  
  
 @Override  
 public void onNothingSelected(AdapterView<?> adapterView) {  
  
 }  
}

**SmsActivity.java**

package com.example.board;  
  
import android.Manifest;  
  
import android.app.AlertDialog;  
import android.content.DialogInterface;  
import android.content.SharedPreferences;  
import android.content.pm.PackageManager;  
import android.net.Uri;  
import android.os.Build;  
import android.os.Bundle;  
  
import android.app.PendingIntent;  
import android.content.Intent;  
import android.provider.Settings;  
import android.telephony.SmsManager;  
  
import android.view.MenuItem;  
import android.view.View;  
import android.view.View.OnClickListener;  
import android.view.WindowManager;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.ActionBar;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
  
public class SmsActivity extends AppCompatActivity implements View.OnClickListener{  
  
 SharedPreferences sp;  
 SharedPreferences.Editor editor;  
  
 EditText message;  
 Button sendSms;  
 boolean isGranted = false;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setTheme(R.style.*Theme\_Board*);  
 setContentView(R.layout.*activity\_sms*);  
  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*,  
 WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);//הופף את הactivity למסך מלא  
 init();  
  
 }  
  
 //פעולה שמאתחלת את האובייקטים  
 public void init(){  
 ActionBar actionBar = getSupportActionBar();  
  
 // showing the back button in action bar  
 actionBar.setDisplayHomeAsUpEnabled(true);//מראה כפתור חזרה בלמעלה של המסך  
  
  
 sp = getSharedPreferences("data",0);  
 editor = sp.edit();  
  
 message=(EditText)findViewById(R.id.*etSms*);  
 sendSms =(Button)findViewById(R.id.*btnSms*);  
  
 sendSms.setOnClickListener(this);  
  
 }  
  
 //מה קורה כשלוחצים על הכפתור חזרה  
 @Override  
 public boolean onOptionsItemSelected(@NonNull MenuItem item) {  
 if (item.getItemId() == android.R.id.*home*) {  
 this.finish();  
 return true;  
 }  
 return super.onOptionsItemSelected(item);  
 }  
  
 @Override  
 public void onClick(View v) {  
 if (!message.getText().toString().equals("")) {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*) {  
 permission();//בקשת הרשאה  
 if (isGranted) {  
 sendSmsFunction();  
 }  
 }  
 else  
 sendSmsFunction();  
 }  
 else  
 Toast.*makeText*(getApplicationContext(), "message cannot br null", Toast.*LENGTH\_LONG*).show();  
  
 }  
 public void dialog()//בונה את הדיאלוג אם המשתמש לא נתן הרשאה ומנסה עוד פעם לשלוח הודעה  
 {  
 AlertDialog.Builder builder = new AlertDialog  
 .Builder(this);  
 builder.setMessage("to contact with developer, the app need to access sms");//הסבר למשתמש  
 builder.setTitle("Sms permission needed");  
 builder.setCancelable(true);  
  
  
 builder.setPositiveButton(  
 "Allow", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which)//אם המשתמש לוחץ allow  
 {  
 ActivityCompat.*requestPermissions*(SmsActivity.this, new String[]{Manifest.permission.*SEND\_SMS*}, 200);//מבקש הרשאה  
 }  
 });  
 builder.setNegativeButton(  
 "No", new DialogInterface.OnClickListener() {  
  
 @Override  
 public void onClick(DialogInterface dialog, int which)//אם המשתמש לוחץ no  
 {  
 dialog.cancel();//סוגר את הדיאלוג  
 }  
 });  
  
 AlertDialog alertDialog = builder.create();  
 alertDialog.show();//מציד את הדיאלוג  
 }  
  
  
 public void sendSmsFunction(){//פעולה ששולחת sms למפתח  
 String msg=message.getText().toString();//מוציא את ההודעה מהeditText  
  
  
 Intent intent=new Intent(getApplicationContext(),MainActivity.class);  
 PendingIntent pi=PendingIntent.*getActivity*(getApplicationContext(), 0, intent,0);  
  
  
 SmsManager sms=SmsManager.*getDefault*();//מתחבר לsms של המערכת  
 sms.sendTextMessage("0585115606", null, msg, pi,null);//שולח את ההודעה וחוזר למסך הראשי  
  
 Toast.*makeText*(getApplicationContext(), "Message Sent successfully!", Toast.*LENGTH\_LONG*).show();  
 }  
  
 public void permission() {//פעולה שמטפלת בהרשאה  
 if (ActivityCompat.*checkSelfPermission*(this, Manifest.permission.*SEND\_SMS*) != PackageManager.*PERMISSION\_GRANTED*) {//אם אין הרשאה  
 if (ActivityCompat.*shouldShowRequestPermissionRationale*(this, Manifest.permission.*SEND\_SMS*)) {//אם זה פעם שניה שמבקשים הרשאה  
 dialog();  
  
  
 }  
 else if (!sp.getBoolean("firstCheckPermission",false)){//אחרת אם זה פעם ראשונה שמבקשים הרשאה  
 ActivityCompat.*requestPermissions*(this, new String[]{Manifest.permission.*SEND\_SMS*}, 100);//מבקרש הרשאה  
 editor.putBoolean("firstCheckPermission",true);  
 editor.commit();  
  
 }  
 else{//אם זה כבר פעם שלישית מפנים את המשתמש להגדרות של האפליקציה  
 Toast.*makeText*(this, "Please allow sms permission setting", Toast.*LENGTH\_LONG*);  
 Intent intent = new Intent();  
 intent.setAction(Settings.*ACTION\_APPLICATION\_DETAILS\_SETTINGS*);//הגדרת הintent להגדרות של האפליקציה  
 Uri uri = Uri.*fromParts*("package", this.getPackageName(), null);  
 intent.setData(uri);//הפניה להגדרות של האפליקציה הנוכחית  
 this.startActivity(intent);  
 }  
  
 }  
 else//יש הרשאה  
 isGranted = true;  
 }  
  
 @Override  
 public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {//פעולה שמופעלת אחרי בקשת הרשאה  
 super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
 if (requestCode == 100) {  
 if (grantResults.length > 0 && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {//אם המשתמש אישר  
 isGranted = true;  
 System.*out*.println("PLAYGROUND Permission has been granted");  
  
 } else {//המשתמש לא אישר  
 System.*out*.println("PLAYGROUND Permission has been denied or request cancelled");  
 isGranted = false;  
 }  
 }  
 if (requestCode == 200) {  
 if (grantResults.length > 0 && grantResults[0] == PackageManager.*PERMISSION\_GRANTED*) {  
 isGranted = true;  
 System.*out*.println("PLAYGROUND Permission has been granted");  
  
 } else  
 isGranted = false;  
 }  
 }  
  
}

**Record.java**

package com.example.board;  
  
public class Record {  
 private int move;//מהלכים  
 private String time;//זמן  
 private String date;//תאריך  
  
  
  
 public Record(int move, String time, String date) {  
 this.move = move;  
 this.time = time;  
 this.date = date;  
  
  
 }  
 public int getMove() {  
 return move;  
 }  
 public void setMove(int move) {  
 this.move = move;  
 }  
  
 public String getTime() {  
 return time;  
 }  
 public void setTime(String time) {  
 this.time = time;  
 }  
  
 public String getDate() {  
 return date;  
 }  
 public void setDate(String date) {  
 this.date = date;  
 }  
  
 @Override  
 public String toString() {  
 return "Record{" +  
 "move='" + move + '\'' +  
 ", time='" + time + '\'' +  
 ", date=" + date +  
 '}';  
 }  
}

**RecordHelper.java**

package com.example.board;  
  
  
import android.annotation.SuppressLint;  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.util.Log;  
  
import java.util.ArrayList;  
  
public class RecordHelper extends SQLiteOpenHelper {  
  
 public static final String *DATABASENAME* = "records.db";  
 public static String *table\_records*;  
 public static final int *DATABASEVERSION* = 1;  
  
  
 public final String COLUMN\_MOVE = "move";  
 public final String COLUMN\_TIME = "time";  
 public final String COLUMN\_DATE = "date";  
  
 SQLiteDatabase database;  
  
 private static String *CREATE\_TABLE\_RECORD*;  
  
 private String orderBy;  
  
 String[] allColumns = { COLUMN\_MOVE, COLUMN\_TIME, COLUMN\_DATE};//כל העמודות  
  
  
 public RecordHelper(Context context, final String table\_records,String orderBy) {  
 super(context, *DATABASENAME*, null, *DATABASEVERSION*);  
 this.*table\_records* = table\_records;//הגדרת איזה טבלה  
 this.orderBy = orderBy;//הגדרת איזה סידור  
 *CREATE\_TABLE\_RECORD* = "CREATE TABLE IF NOT EXISTS " +  
 this.*table\_records* + "(" + COLUMN\_MOVE + " INTEGER," + COLUMN\_TIME + " VARCHAR,"  
 + COLUMN\_DATE + " INTEGER " + ");";//פקודה בsql ליצירת טבלה  
 }  
  
 public String getOrderBy() {  
 return orderBy;  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 db.execSQL(*CREATE\_TABLE\_RECORD*);//מריץ את הפקודה ליצירת טבלה  
 Log.*d*("data1", "Table customer created");  
 }  
  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {//עדכון הבסיס נתונים  
 db.execSQL("DROP TABLE IF EXISTS " + *table\_records*);  
 onCreate(db);  
 }  
  
 public void open() {//נותן הרשאת כתיבה בבסיס נתונים  
 database = this.getWritableDatabase();  
 Log.*d*("data1", "Database connection open");  
 }  
  
 public Record createRecord(Record r) {//פעולה שמקבלת שיא ומוסיפה אותו לבסיס נתונים  
 ContentValues values = new ContentValues();//אובייקט שמיועד להכניס נתונים לבסיס נתונים  
 values.put(this.COLUMN\_MOVE, r.getMove());  
 values.put(this.COLUMN\_TIME, r.getTime());  
 values.put(this.COLUMN\_DATE, r.getDate());  
  
 long insertId = database.insert(RecordHelper.*table\_records*, null, values);//הפקודה שמוסיפה את השיא החדש  
 Log.*d*("data1", "Record " + insertId + " insert to database");  
  
 return r;  
 }  
 @SuppressLint("Range")  
 public ArrayList<Record> getAllRecord() {//פעולה שמחזירה את כל השיאים  
  
 onCreate(database);  
 ArrayList<Record> l = new ArrayList<Record>();  
 Cursor cursor=database.query(this.*table\_records*, allColumns, null, null, null, null, orderBy+ " ASC");//פקודה שמוציאה את כל הנתונים מהטבלה  
  
 if(cursor.getCount()>0)  
 {  
 while(cursor.moveToNext())//כל עוד יש עוד שיא  
 {  
  
 int move=cursor.getInt(cursor.getColumnIndex(this.COLUMN\_MOVE));  
 String time=cursor.getString(cursor.getColumnIndex(this.COLUMN\_TIME));  
 String date=cursor.getString(cursor.getColumnIndex(this.COLUMN\_DATE));  
 Record r=new Record(move,time,date);  
 l.add(r);  
 }  
  
 }  
 return l;  
 }  
}

**RecordAdapter.java**

package com.example.board;  
  
import android.content.Context;  
  
  
import android.view.LayoutInflater;  
  
import android.view.View;  
  
import android.view.ViewGroup;  
  
import android.widget.TextView;  
  
  
import androidx.recyclerview.widget.RecyclerView;  
  
import java.util.List;  
  
  
  
public class RecordAdapter extends RecyclerView.Adapter<RecordAdapter.RecordViewHolder> {  
  
 private final Context context;  
 private final List<Record> recordList;//הרשימה של השיאים  
  
  
 public RecordAdapter(Context context, List<Record> recordList)//פעולה בונה  
  
 {  
 this.context = context;  
 this.recordList = recordList;  
 }  
  
  
 @Override  
 public RecordViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {//יצירת holder בשביל הview הכללי של השיאים  
  
  
 LayoutInflater inflater = LayoutInflater.*from*(context);  
 View view = inflater.inflate(R.layout.*records\_layuot*, null);  
 return new RecordViewHolder(view);  
  
 }  
  
 @Override  
 public void onBindViewHolder(RecordViewHolder holder, int position) {//הכנסת כל שיא בRecyclerView  
  
  
 Record record = recordList.get(position);  
  
 holder.tvNum.setText(String.*valueOf*(position+1));  
 holder.tvMove.setText(String.*valueOf*(record.getMove()));  
 holder.tvTime.setText(record.getTime());  
 holder.tvDate.setText(record.getDate());  
 }  
  
 @Override  
 public int getItemCount() {  
 return recordList.size();  
 }//פעולה שמחזירה את הגודל של הרשימה  
  
 class RecordViewHolder extends RecyclerView.ViewHolder {//הclass שיורש מViewHolder  
  
 TextView tvNum, tvMove, tvTime,tvDate;  
  
 public RecordViewHolder(View itemView) {//מאתחל את הview הנוכחי  
 super(itemView);  
  
 tvNum = itemView.findViewById(R.id.*tvId*);  
 tvMove = itemView.findViewById(R.id.*tvMoves*);  
 tvTime = itemView.findViewById(R.id.*tvTime*);  
 tvDate = itemView.findViewById(R.id.*tvDate*);  
 }  
 }  
  
}

**Square.java**

package com.example.board;  
  
import android.content.Context;  
import android.graphics.Canvas;  
import android.graphics.Color;  
import android.graphics.Paint;  
import android.view.View;  
  
public class Square {  
  
 BoardGame boardGame;  
 Tile tile;  
 float x,y,w,h;  
  
  
 int number;  
  
  
 public Square(BoardGame boardGame,float x,float y,float w,float h,Tile tile,int number)  
 {  
  
 this.number = number;//המספר של הריבוע  
 this.x = x;//מיקום לפי x  
 this.y = y;//מיקום לפי y  
  
 this.boardGame = boardGame;  
  
  
 this.w = w;//גודל של הריבוע  
 this.h = h;  
 this.tile = tile;  
 }  
 public void setTile(Tile tile) {  
 this.tile = tile;  
 }  
  
 public void setX(float x) {  
 this.x = x;  
 }  
  
 public void setY(float y) {  
 this.y = y;  
 }  
 public Tile getTile() {  
 return tile;  
 }  
  
 public float getX() {  
 return x;  
 }  
  
 public float getY() {  
 return y;  
 }  
  
 public boolean didXAndYInSquare(float xo, float yo)//פעולה שמקבלת x וy ובודקת אם הם נמצאים על הריבוע  
 {  
 if(xo > x && xo < x + w && yo > y && yo < y + h )  
 return true;  
 return false;  
 }  
  
  
}

**Tile.java**

package com.example.board;  
  
  
import android.graphics.Canvas;  
import android.graphics.Paint;  
  
  
public class Tile {  
  
  
 BoardGame boardGame;  
 float x, y, w, h;  
 int number, color;  
 Paint p, paintTitle;  
  
 public Tile(BoardGame boardGame, float x, float y, float w, float h, int num, int color) {  
 this.boardGame = boardGame;  
  
 this.x = x;  
 this.y = y;  
 this.w = w;  
 this.h = h;  
  
 this.number = num;  
 this.color = color;  
  
 init();  
  
 }  
  
 private void init() {  
  
 //אתחול הpaint של הרקע של האריח  
 p = new Paint();  
 p.setColor(color);  
  
 //אתחול הpaint של המספרים  
 paintTitle = new Paint();  
 paintTitle.setARGB(255, 255, 255, 255);  
 paintTitle.setTextAlign(Paint.Align.*CENTER*);  
 paintTitle.setTextSize(70);  
  
  
 }  
  
  
 public void setX(float x) {  
 this.x = x;  
 }  
  
 public void setY(float y) {  
 this.y = y;  
 }  
  
 public float getX() {  
 return x;  
 }  
  
 public float getY() {  
 return y;  
 }  
  
 public void draw(Canvas canvas) {//פעולה שמציירת את האריח  
 canvas.drawRect(x,y,x+w,y+h,p);//מצייר את הרקע של האריח  
  
 //drawRoundRect(x, y, x + w, y + h, p2, canvas);  
  
 //מצייר את המספר שעל האריח  
 canvas.drawText(String.*valueOf*(number), x + (w / 2), (y + (h / 2))-(paintTitle.descent()+paintTitle.ascent())/2, paintTitle);  
 }  
  
  
  
 public void slide(Square to) {//פעולה שמקבלת ריבוע ומעבירה את האריח הנוכחי לריבוע הזה  
  
 this.setX(to.getX());  
 this.setY(to.getY());  
 }  
  
}

**BoardGame.java**

package com.example.board;  
  
  
import android.content.Context;  
import android.graphics.Canvas;  
import android.graphics.Color;  
  
import android.view.View;  
import java.util.Arrays;  
import java.util.Random;  
  
  
  
  
  
public class BoardGame extends View {  
  
 boolean ifOne = true;//בודק אם זה פעם ראשונה לצייר את הלוח  
 int size,color;  
  
 int[] theSort;//מערך עם הסדר של הבלגון של המספרים  
 Random r;  
 int all;  
  
 Tile[][]tiles;  
 Square[][]squares;  
  
 Context context;  
  
 static Square *blank*;//המקום הריק  
  
 static Time *time*;  
  
 public BoardGame(Context context,int size,int color) {  
 super(context);  
 this.context = context;  
 this.size = size;  
 this.color = color;  
 squares = new Square[size][size];  
 tiles = new Tile[size][size];  
 r = new Random();  
 all = size\*size;//לדוגמא 4x4 = 16  
 theSort = new int[all];  
  
  
  
 this.setBackgroundColor(Color.*parseColor*("#393939"));//שינוי הצבע של הרקע לאפור  
  
 }  
  
  
 @Override  
 protected void onDraw(Canvas canvas) {//פעולה של view ובה מציירים על הלוח עם canvas  
 super.onDraw(canvas);  
 drawBoard(canvas);  
  
  
 }  
  
 public void resetArr(int[] arr){//ריסוט המערך שכל אבריו יהיו 1-  
 Arrays.*fill*(arr, -1);  
  
  
 }  
  
 public boolean findInArray(int n,int[] arr,int temp)//בדיקה אם מספר כבר נמצא במערך  
 {  
 for (int i = 0; i < temp; i++) {  
 if (arr[i] == n)  
 return true;  
 }  
 return false;  
 }  
  
 public void swap(){//ערבוב המערך במספרים מ1 עד all  
 int temp;  
 for (int i = 0;i<all;i++) {  
  
 do {  
 temp = r.nextInt(all);  
 }while (findInArray(temp,theSort,i));  
  
 theSort[i] = temp;  
 }  
 }  
  
  
  
 public boolean isSolvable(int[] puzzle)// פעולה עם אלגוריתמיקה מתקדמת שבודקת אם המערך פתיר במשחק הזה, קישור עם הסבר מפורט על האלגוריתם - https://www.geeksforgeeks.org/check-instance-15-puzzle-solvable/  
 {  
 int parity = 0;// מספר ההיפוכים  
 int gridWidth = (int) Math.*sqrt*(puzzle.length);  
 int row = 0; // השורה הנוכחית שאנחנו עליה  
 int blankRow = 0; // השורה שבה נמצא החור  
  
 for (int i = 0; i < puzzle.length; i++)  
 {  
 if (i % gridWidth == 0) { // התקדמות לשורה הבאה  
 row++;  
 }  
 if (puzzle[i] == 0) { // החור  
 blankRow = row; // שומר את השורה שבו החור נמצא  
 continue;  
 }  
 for (int j = i + 1; j < puzzle.length; j++)  
 {  
 if (puzzle[i] > puzzle[j] && puzzle[j] != 0)//  
 {  
 parity++;//סופר את מספר ההיפוכים  
 }  
 }  
 }  
  
 if (gridWidth % 2 == 0) { // האם הגודל זוגי  
 if (blankRow % 2 == 0) { // האם החור בשורה אי זוגית מלמטה  
 return parity % 2 == 0;  
 } else { // החור בשורה זוגית מלמטה  
 return parity % 2 != 0;  
 }  
 } else { // הגודל אי זוגי  
 return parity % 2 == 0;  
 }  
 }  
  
 public void buildArrayTheSort()//פעולה שמשתמשת בפעולות resetArr, swap וisSolveable בשביל ליצור את המערך הסופי של הבלגון  
 {  
 do {  
 resetArr(theSort);  
 swap();  
 }while (!isSolvable(theSort));  
 }  
  
  
 public void drawBoard(Canvas canvas){//הפעולה שמציירת את הלוח  
 if(ifOne) {  
  
  
 buildArrayTheSort();  
  
 int count = 1;  
 int x = 15;  
 int y = 15;  
  
 int h = ((canvas.getWidth()-15)/size)-15;//חישוב של הגודל של האריחים בצורה שיישאר רווחים שווים בהתחלה, באמצע ובסוף  
 int w = ((canvas.getWidth()-15)/size)-15;  
  
  
  
  
  
  
 for (int i = 0; i < squares.length; i++) {  
 for (int j = 0; j < squares.length; j++) {  
  
 if (theSort[count - 1] != 0) {//אם זה לא החור  
  
 tiles[i][j] = new Tile(this, x, y, w, h, theSort[count - 1], color);  
 tiles[i][j].draw(canvas);  
  
 }  
 squares[i][j] = new Square(this, x, y, w, h,tiles[i][j],count);  
 x = x + w + 15;//התקדמות אופקית  
 count++;  
  
 }  
 y = y + h + 15;//התקדמות אנכית  
 x = 15;  
 }  
 }  
  
 *blank* = blank();  
 if (!ifOne) {  
 for (int i = 0; i < squares.length; i++) {  
 for (int j = 0; j < squares.length; j++) {  
 if (squares[i][j].number != *blank*.number)  
 squares[i][j].getTile().draw(canvas);//ציור מחדש של כל האריחים  
  
 }  
 }  
 }  
 ifOne = false;  
 }  
  
 public Square blank(){//פעולה שמחזירה את המקום הריק  
 for (int i = 0; i < squares.length; i++) {  
 for (int j = 0; j < squares.length; j++) {  
 if (squares[i][j].getTile() == null)  
 return squares[i][j];  
  
 }  
 }  
 return null;  
  
 }  
 public void slide(Square from,Square to) {//פעולה שמזיזה אריח מקום למקום  
 if (from.getTile()!=null)  
 {  
 to.setTile(from.getTile());  
 from.setTile(null);  
 to.getTile().slide(to);  
  
 }  
 }  
  
  
  
 public boolean checkBlankNear(float x, float y) {//בודק אם הx והy נמצאים ליד המקום הריק  
 float distant = squares[0][0].h;  
 if (*blank*.didXAndYInSquare(x,y+distant) || *blank*.didXAndYInSquare(x,y-distant) || *blank*.didXAndYInSquare(x+distant,y) || *blank*.didXAndYInSquare(x-distant,y))  
 return true;  
 return false;  
 }  
  
 public boolean isWin() {//בודק אם הלוח פתור  
 for(int i = 0;i<squares.length;i++) {  
 for (int j = 0; j < squares.length; j++) {  
 if (squares[i][j].getTile()==null)//אם אנחנו על הריבוע הריק  
 {  
 //אם הגענו לריובע האחרון והחור נמצא שם בהכרח הלוח פתור מכיוון שעברנו על כל המספרים ובדקנו שהם נמצאים במקום  
 return i == size - 1 && j == size - 1;  
 }  
 if(squares[i][j].number != squares[i][j].getTile().number)  
 return false;  
 }  
 }  
 return true;  
 }  
  
 public Square findSquare(float x, float y) {//מקבלת x וy ומחזירה את הsquare שנמצא שם  
 for(int i = 0;i<squares.length;i++) {  
 for (int j = 0; j < squares.length; j++) {  
 if(squares[i][j].didXAndYInSquare(x,y))  
  
 return squares[i][j];  
 }  
 }  
 return null;  
 }  
  
  
  
  
}

**Game.java**

package com.example.board;  
  
import android.annotation.SuppressLint;  
import android.app.Activity;  
import android.app.Dialog;  
import android.content.Context;  
import android.content.SharedPreferences;  
import android.graphics.Color;  
import android.os.Handler;  
import android.os.Message;  
import android.util.DisplayMetrics;  
import android.view.MotionEvent;  
import android.view.View;  
import android.widget.Button;  
import android.widget.LinearLayout;  
import android.widget.TextView;  
  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.Locale;  
  
import static com.example.board.BoardGame.*blank*;  
import static com.example.board.BoardGame.*time*;  
import static com.example.board.MainActivity.*btnPause*;  
  
public class Game implements View.OnTouchListener {  
 private BoardGame boardGame;//הלוח  
 Button btnSolved;  
 Dialog solvedD;  
 TextView tvTime, tvMoves;  
 private final Context context;  
 SharedPreferences getSetting;  
 LinearLayout l;  
  
 boolean ifOne = true, ifStart=false, ifPause = true;  
 int moves, sizeOfBoard=4, colorOfTile;  
 Handler handler;  
 RecordHelper recordHelper;  
  
  
 public Game(Context context) {//הפעולה הבונה  
 this.context = context;  
  
 tvTime = ((Activity)context).findViewById(R.id.*tvTime*);  
 tvMoves = ((Activity)context).findViewById(R.id.*tvMoves*);  
 startMode();  
  
 l = ((Activity)context).findViewById(R.id.*lGame*);  
 l.removeView(l.getChildAt(0));  
  
 getSetting = context.getSharedPreferences("data",0);  
 update();  
 moves = 0;  
 createBoardGame();  
  
  
 }  
  
 @SuppressLint("ClickableViewAccessibility")  
 public void createBoardGame(){//יצירת הלוח  
  
 if (sizeOfBoard != 0 && colorOfTile != 0)  
 boardGame = new BoardGame(context,sizeOfBoard, colorOfTile);  
 else//במקרה שזה פעם ראשונה שמופעלת האפליקצה  
 boardGame = new BoardGame(context,4, Color.*MAGENTA*);  
  
  
 DisplayMetrics displayMetrics = new DisplayMetrics();  
 ((Activity)context).getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);//מציאת הרוחב של המסך  
 int width = displayMetrics.widthPixels;  
  
 LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(width,width);  
 boardGame.setLayoutParams(params);//הגדרת הגודל של הלוח  
 boardGame.setOnTouchListener(this);  
 l.addView(boardGame);  
 }  
  
 public void doHandler(){//פעולה שמקושרת לhandler ומקבלת ממנו messages  
 handler=new Handler(new Handler.Callback() {  
 @SuppressLint({"DefaultLocale", "SetTextI18n"})  
 @Override  
 public boolean handleMessage(Message msg)  
 {  
 if (*time*.isRun)  
 tvTime.setText(String.*format*("%02d",*time*.getMinute())+":"+String.*format*("%02d",msg.arg2) +"."+ msg.arg1);//שינוי הזמן בכל עשירית שניה  
 if (!ifStart)  
 tvTime.setText("00:00.0");  
 return true;  
 }  
  
 });  
 }  
 public void startMode(){//מצב התחלתי  
 if (*time* != null) {  
  
 *time*.isRun = false;  
 *time*.handler = null;  
 }  
  
 doHandler();  
 *time* = new Time(handler);  
 tvTime.setText("00:00.0");  
 tvMoves.setText("number of moves: 0");  
 *btnPause*.setText("pause");  
 *btnPause*.setEnabled(false);  
  
 }  
 public void runMode(){//מעבר למצב ריצה  
 if (*time*!=null) *time*.isRun = true;  
 *btnPause*.setEnabled(true);  
 *btnPause*.setText("pause");  
 ifStart = true;  
 ifPause = false;  
 }  
  
 public void stopMode(){//מעבר למצב עצירה  
 if (*time*!=null) *time*.isRun = false;  
 *btnPause*.setText("continue");  
 ifStart = true;  
 ifPause = true;  
  
 }  
  
  
 @Override  
 public boolean onTouch(View view, MotionEvent event) {//ברגע שנוגעים בלוח  
 if (ifOne){//פעם ראשונה  
 *time*.start();  
 runMode();  
 }  
 ifOne = false;  
  
 if (!ifPause) {  
 if (event.getAction() == MotionEvent.*ACTION\_UP*) {//ברגע שהורמה האצבע  
  
 Square mySquare = boardGame.findSquare(event.getX(), event.getY());//מציאת הריבוע שנלחץ  
 if (mySquare != null && boardGame.checkBlankNear(event.getX(), event.getY())) {//בודקת אם אפשר לעשות מהלך חוקי  
 boardGame.slide(mySquare, *blank*);  
 moves++;  
 tvMoves.setText("number of moves: " + moves);  
 }  
 if (boardGame.isWin()) {//במצב של ניצחון  
 ifPause = true;  
 *time*.isRun = false;  
 createSolvedDialog();  
 //Toast.makeText(context, "ניצחת אלוף!!", Toast.LENGTH\_SHORT).show();  
 addNewRecord();  
 }  
 boardGame.invalidate();//קריאה לonDraw בשביל לצייר את הלוח מחדש  
 }  
 }  
 return true;  
 }  
 public void addNewRecord(){//פעולה שמוסיפה שיא חדש  
 recordHelper.open();  
 String currentDate = new SimpleDateFormat("dd-MM-yyyy", Locale.*getDefault*()).format(new Date());//מציאת התאריך הנוכחי  
 Record r = new Record(moves,tvTime.getText().toString(),currentDate);//יצירת שיר חדש  
 recordHelper.createRecord(r);//הוספת השיא החדש  
 recordHelper.close();  
 }  
  
 public void update()//SharedPreferencesפעולה שמעדכנת את הנתונים מה  
 {  
 int size = getSetting.getInt("size", 0);  
 if (size != 0) sizeOfBoard =size;  
 int color = getSetting.getInt("color", 0);  
 if (color != 0) colorOfTile = color;  
 recordHelper = new RecordHelper(context,"tblrecords"+sizeOfBoard, getSetting.getString("orderBy",null));  
  
 }  
 public void createSolvedDialog()//פתיחת דיאלוג כשמנצחים  
 {  
 solvedD=new Dialog(context);  
 solvedD.setContentView(R.layout.*custom\_solved*);  
 solvedD.setCancelable(false);  
 btnSolved = solvedD.findViewById(R.id.*btnSolved*);  
 btnSolved.setOnClickListener((View.OnClickListener) context);  
 solvedD.show();  
 }  
}

**Time.java**

package com.example.board;  
  
import android.net.MacAddress;  
import android.os.Handler;  
import android.os.Message;  
  
public class Time extends Thread {  
  
  
  
 boolean isRun=true;  
  
 Handler handler;  
  
 int milliSeconds, seconds, minute;//מילי שניות,שניות,דקות  
 public Time(Handler handler)  
 {  
 this.handler=handler;  
 milliSeconds = 0;  
 seconds = 0;  
 minute = 0;  
 }  
  
 public int getMinute() {  
 return minute;  
 }  
  
 @Override  
 public void run() {//מימוש הפעולה run שרצה מתי שהthread מתחיל  
 super.run();  
 while(true)  
  
  
 {  
 if(isRun)  
 {  
 try {  
 Thread.*sleep*(100);//מחכה עשירית שניה  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 if (isRun) {  
 Message msg = new Message();  
 milliSeconds++;  
  
 //העלאת שניה אחרי 10 מילי שניות  
 if (milliSeconds == 10) {  
 milliSeconds = 0;  
 seconds++;  
 }  
 //העלאת דקה אחרי 60 שניות  
 if (seconds == 60) {  
 minute++;  
 seconds = 0;  
 }  
 msg.arg1 = milliSeconds;  
 msg.arg2 = seconds;  
  
 handler.sendMessage(msg);//שליחת הזמן המעודכן לhandler  
 }  
  
 }  
 }  
 }  
}

**ReciverNotification.java**

package com.example.board;  
  
import android.app.Notification;  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
  
import android.os.Build;  
import android.widget.Toast;  
  
import androidx.core.app.NotificationCompat;  
  
import java.util.Random;  
  
public class ReceiverNotification extends BroadcastReceiver {  
  
 @Override  
 public void onReceive(Context context, Intent intent) {//כשהBroadcast מופעל  
  
 createNotification(context);  
 }  
 public void createNotification(Context context){  
 int icon = android.R.drawable.*star\_on*;  
  
 long when = System.*currentTimeMillis*();//הזמן הנוכחי  
 String title = "important message";  
 String ticker = "ticker";  
 int random = new Random().nextInt(5)+1;//הגרלת מספר הודעה  
 String message = "";  
 switch (random) {//השמת הודעה רנדומלית  
 case 1:  
 message = context.getResources().getString(R.string.*message1*);  
 break;  
 case 2:  
 message = context.getResources().getString(R.string.*message2*);  
 break;  
 case 3:  
 message = context.getResources().getString(R.string.*message3*);  
 break;  
 case 4:  
 message = context.getResources().getString(R.string.*message4*);  
 break;  
 case 5:  
 message = context.getResources().getString(R.string.*message5*);  
 break;  
 }  
  
 //phase 2  
 Intent intent1 = new Intent(context, MainActivity.class);//זה intent ששולח מפה לmainActivity  
  
 PendingIntent pendingIntent = PendingIntent.*getActivity*(context, 0, intent1, 0);  
 NotificationManager notificationManager = (NotificationManager) context.getSystemService(Context.*NOTIFICATION\_SERVICE*);//חיבור לservice של המערכת  
  
 //הכנת הבנאי של ההתראה  
 NotificationCompat.Builder builder = new NotificationCompat.Builder(context, "M\_CH\_ID");  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {//אם הגרסה מעל אוראו חובה להוסיף channel  
 String channelId = "YOUR\_CHANNEL\_ID";  
 NotificationChannel channel = new NotificationChannel(channelId,  
 "Channel human readable title",  
 NotificationManager.*IMPORTANCE\_DEFAULT*);  
 notificationManager.createNotificationChannel(channel);  
 builder.setChannelId(channelId);  
 }  
 //יצירת אובייקט של התראה והכנסת אליו כל הרכיבים שהכנו קודם  
 Notification notification = builder.setContentIntent(pendingIntent)  
 .setSmallIcon(icon).setTicker(ticker).setWhen(when)  
 .setAutoCancel(true).setContentTitle(title).setStyle(new NotificationCompat.BigTextStyle()  
 .bigText(message)).build();  
  
 notificationManager.notify(3, notification);//קריאה להתראה  
  
 //Toast.makeText(context, "Alarm....", Toast.LENGTH\_LONG).show();  
 }  
}