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 עניינים

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(הdefult הוא 4x4).

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

הסיבה שבחרתי את הנושא הנל זה שעניין אותי לבנות משהו שלא רק מצריך ידע במושגים בתכנות ובכל מיני דברים טכניים אלא גם חשיבה מעמיקה ואלגוריתמיקה ובמשחקים כמו זה יש הרבה דברים שצריך לחשוב לגבם ולהפעיל את המוח.

יכולות המערכת:

* לשחק במשחק
* לערבב את המשחק ולהתחיל מהתחלה בכל שלב
* לעצור את המשחק ולהמשיך
* לשנות את הגודל של הלוח וכך גם את הצבע של האריחים
* לראות את השיאים מסודרים לפי מהלכים ולפי הזמן
* לשלוח הודעת sms למפתח.

תהליך המחקר:

בהתחלה לקחתי את המשחק ושיחקתי איתו קצת וניסיתי לחשוב איזה קשיים יכולים להיות בבניית המשחק, לאחר מכן הסתכלתי על הערך שלו בוויקיפדיה ולמדתי עוד קצת על האופי של המשחק.

לאחר מכן ניסיתי לחשוב באיזה צורה של תכנות אני אצטרך להשתמש בשביל לבנות את המשחק בצורה הכי אידיאלית ואלגנטית ולכן הסתכלתי קצת באתר github על פרויקטים דומים בשביל לקבל איזה נקודת מבט על איך אני רוצה שהפרויקט יהיה בנוי, שמה ראיתי שדרך מאוד טובה לעבוד עם משחקים כמו זה זה בניית class שיורש מview ועל זה לעבוד אז התחלתי קצת ללמוד על זה והתחלתי להתנסות עם זה.

כמובן שהסתכלתי גם על אפליקציות דומות וראיתי מה המודל הכי טוב בשביל הפרויקט הזה מכל מיני מובנים.

אתגרים מרכזיים:

בעיה רצינית שהיה לי זה איך לבנות את הלוח של המשחק בצורה הכי אלגנטית וקלה לשימוש ועוד בעיה שאיתה התמודדתי זה שיכול לצאת מצב שבו הלוח יהיה לא פתיר והייתי צריך למצוא אלגוריתם שפותר את הבעיה וגם התמודדתי עם המורכבות של לשלב בסיס נתונים וsql באפליקציה.

עוד אתגר מאוד רציני שנתקלתי בו זה המעבר בין מצב שהמשחק עוצר למצב שהוא רץ או מצב שהוא מתחיל מחדשת 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.

אלמנט טכנולוגי חדשני זה שהשתמשתי בrecycleView במקום בlistView ובשביל זה הייתי צריך ליבא ספרייה בgradle.

**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 שלו או היצירה בכלל של הBoardGame ועוד.

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

Record: אובייקט שמייצג יחידה אחת של שיא.

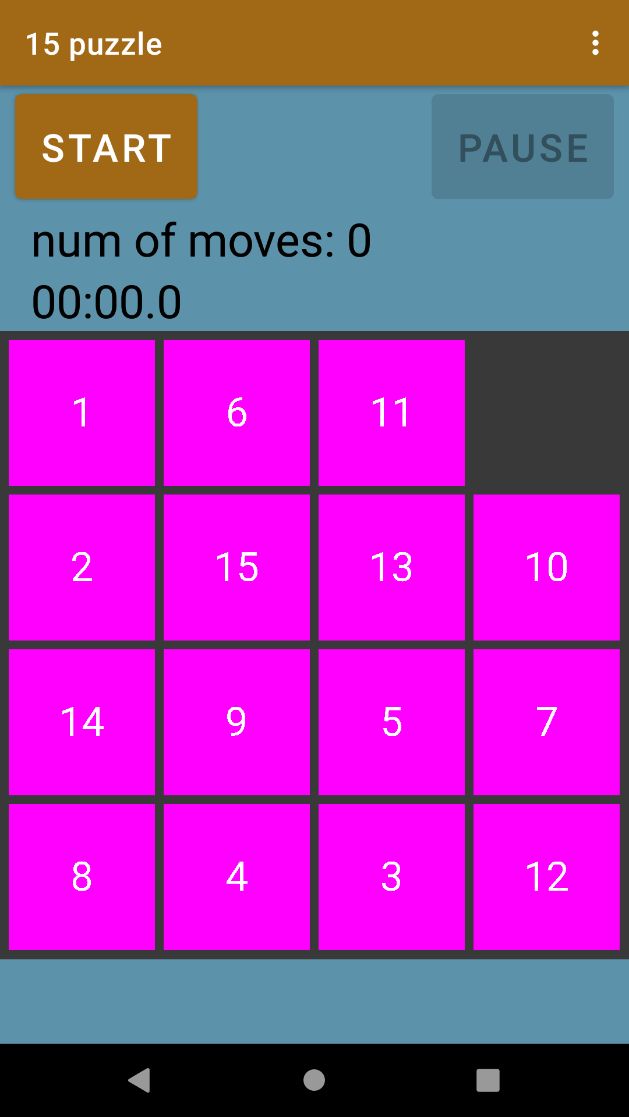
RecordAdapter: אובייקט שיורש מ RecyclerViewודואג לסדר את הרקורדים בסוג של רשימה.

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 או המקומות על הלוח(Square) או מערך של מספרים בשביל הסדר שלהם על הלוח וגם שימוש בטבלה(database) בשביל השיאים ויש לי פעולה של הוספה של שיא:createRecord(Record r) ושל קבלה של כל השיאים: getAllRecord()

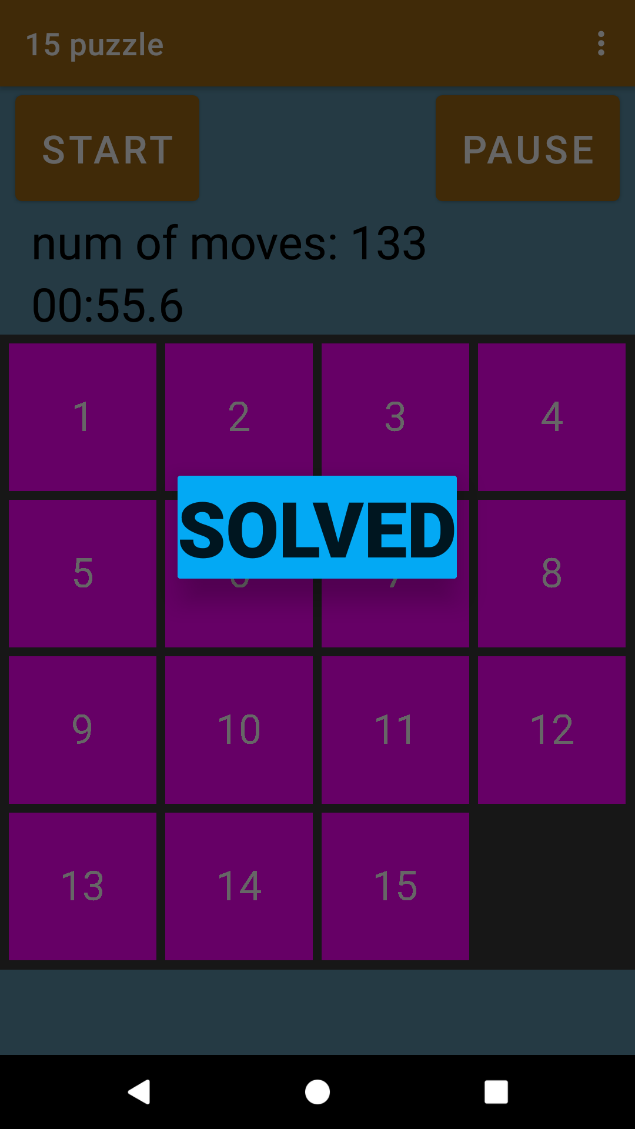
**מבנה / ארכיטקטורה של הפרויקט**

Main Activity – המסך הראשי



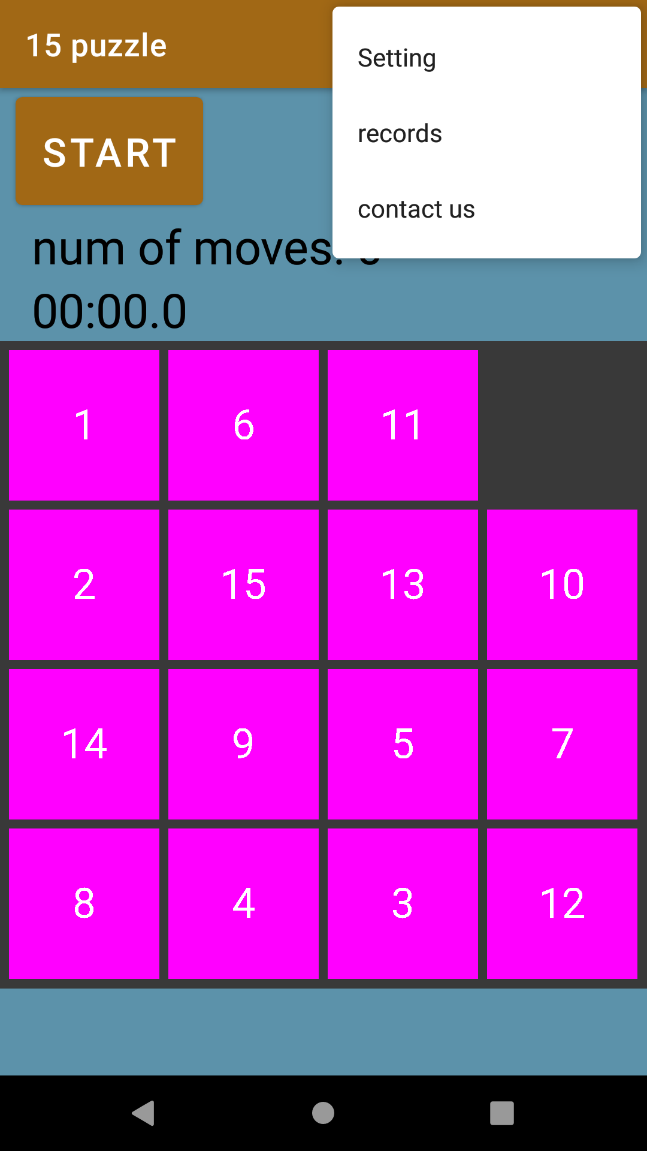
המסך הזה הוא המסך הראשי של המשחק ובו מוטמע האובייקט של הלוח BoardGame בLinearLayout. כשלוחצים על הלוח אז הזמן מתחיל לרוץ ונהיה אפשרי ללחוץ על הpause ואם לוחצים על אריח ליד המקום הריק האריח זז לשם והnum of mobes עולה ב1. אם לוחצים על הstart הזמן והמהלכים מתאפסים והלוח מתערבב. אם לוחצים על הpause הכיתוב משתנה לcontinue והזמן נעצר ולא אפשרי להמשיך את המשחק עד שלוחצים על הcontinue ואז המשחק ממשיך והכתובית משתנה חזרה לpause. בonPause המשחק נעצר ובonCreate המשחק נוצר. לחיצה על ה3 נקודות תפתח תפריט.

Record Dialog - דיאלוג שיאים



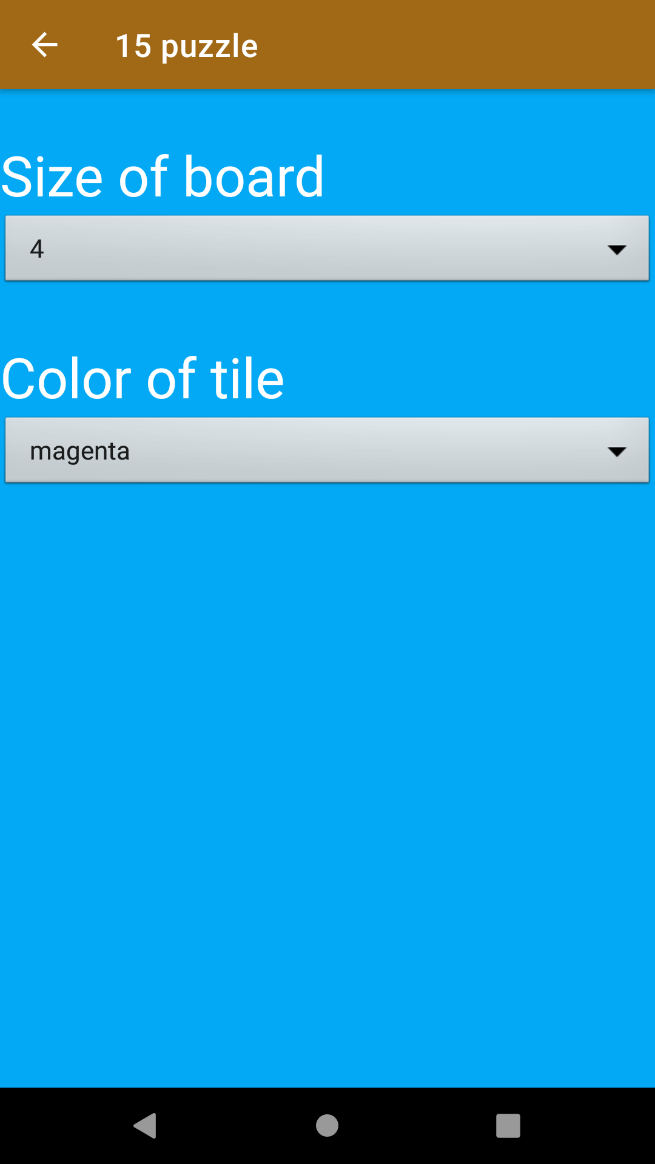
הדיאלוג הזה נפתח מתי שהשחקן פותר את הלוח וגומר את המשחק, המשחק נעצר והתוצאה שלו נשמרת בבסיס נתונים, בשביל לשחק שוב צריך ללחוץ על solved ואז זה מאפס את הזמן והמהלכים ומערבב את הלוח.

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 לגודל הלוח הספיציפי שמוגדר ועצירת המשחק.
3. מעבר למסך של שליחת sms למפתח(SmsActivity) ואיפוס המשחק.

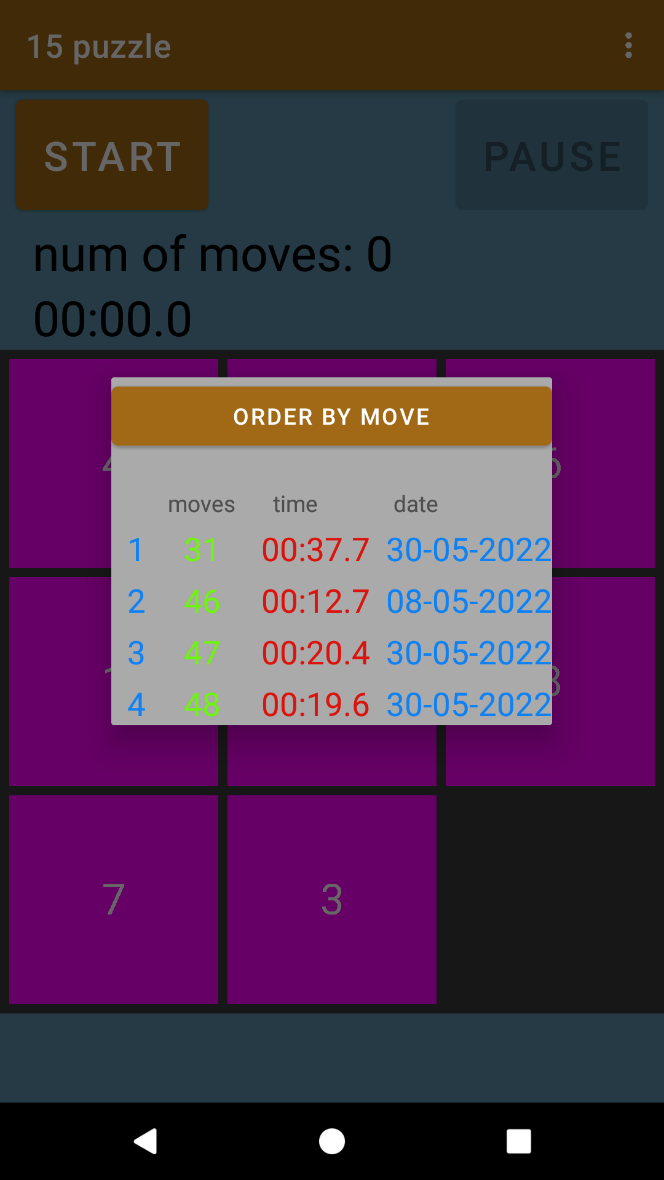
Setting Activity – מסך הגדרות

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

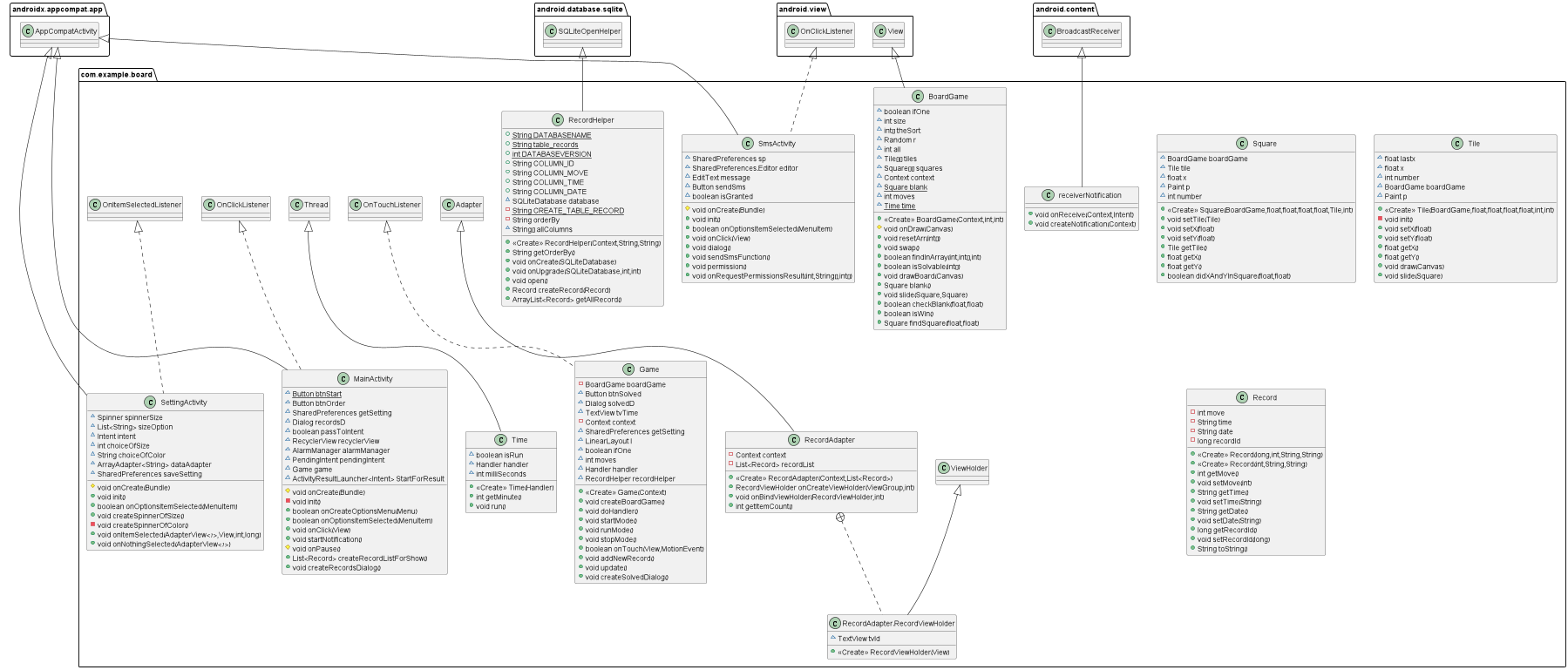
Sms Activity – מסך שליחת הודעת sms למפתח



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

**Screen flow diagram**

**uml**



לתמונה באיכות יותר גבוהה:

<https://drive.google.com/file/d/10WRi1fPlqbmrfSLoh5dAOD8gRf2oHcT2/view?usp=sharing>

**מימוש הפרויקט**

חלק גרפי (קבצי תצורה/קונפיגורציה)

**activity\_main.xml:**

קובץ התצוגה הגרפית במסך הפתיחה(main activity) שבו מוצג הלוח של המשחק, בנוסף יש בו כפתור start שאחראי לערבב את הלוח ולאפס את הזמן והמהלכים וכפתור stop שעוצר את המשחק אם הוא רץ וממשיך אותו אם הוא בעצירה, בנוסף יש text view שמראה את מספר המהלכים ועוד text view שמראה את הזמן שרץ מאז שהתחלת לפתור.

<?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 במסך ההגדרות(setting activity) שבו מוצג הגדרות של המשחק: text view שכתוב עליו size of board ומתחתיו spinner שאתה יכול לבחור בו איזה גודל של הלוח אתה רוצה ואותו דבר עם הגדרה של הצבע של האריחים, text view וspinner.

<?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) שבו מוצג edit text שבו המשתמש כותב את ההודעה 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 להציג את הדיאלוג שמציג את השיאים, הוא מכיל כפתור שאחראי לשנות את הסידור של השיאים, table row שבתוכו כתוב move, time, date שמתחתם מוצג הדברים שהם מתארים והrecycle view שבתוכו 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 שכתוב עליו 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 שבו יש ארבע text view שמייצגים תאריך, זמן, מהלכים וid, אחר כך מממשים אותו בrecycle view לכל שיא ושיא.

<?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 שמוצג במסך הפתיחה(main activiry), יש בו item שמוביל למסך הגדרות(setting activity),item שמקפיץ את הדיאלוג של השיאים וitem שמוביל למסך לשליחת 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>

**strings.xml:**

קובץ xml שאחראי לשמור מחרוזות במקום לכתוב אותם בקוד.

<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>

חלק לוג (קבצי קוד Java)

קבצי קוד - Activities

**MainActiviry:**

מחלקה ראשית שמתחברת לקובץ xml – activity\_main.xml והיא המסך הראשון שנכנסים אליו באפליקציה והוא המסך הראשי ובו נמצא המשחק ומשמה 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 המשתנה עבר לאקטיביטי אחר עם intent. |
| RecyclerView recyclerView | סוג של view שמציג את השיאים בדיאלוג ברשימה. |
| AlarmManager alarmManager | תכונה שמתחברת לservice של המערכת של התזמון של ההתראות. |
| PendingIntent pendingIntent | תכונה שאחראית לשלוח את המשתמש כשהוא לוחץ על ההתראה לMain activity. |
| Game game | מופע של הclass שאחראי לתפעל את המשחק. |
| protected void onCreate(Bundle savedInstanceState) | פעולה מובנת של המערכת שמופעלת כשהאפליקציה נפתחת. |
| private void init() | פעולה שמאתחלת את האובייקטים ומעדכנת את הSharedPreferences, בנוסף היא קוראת ל startNotificationבפעם הראשונה בשביל להגדיר את ההתראה. |
| public boolean onCreateOptionsMenu(Menu menu) | פעולה שיוצרת את התפריט. |
| public boolean onOptionsItemSelected(@NonNull MenuItem item) | פעולה כשלוחצים על אפשרות בתפריט. |
| public void onClick(View v) | פעולה שמופעלת כשלוחצים על כפתור, פה היא מטפלת במקרה כשולחצים על כפתור הstart, כפתור הstop וכתור הsovled שנמצא בדיאלוג שקופץ כשפותרים 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 כשחוזרים מהintent. |
| public void startNotification() | פעולה שמפעילה חזרה של התראה לשחק כל יום ב20:20, הפעולה נקראת רק פעם אחת ויותר לא קוראים לה. |
| protected void onPause() | פעולה מובנת שמופעלת כשיוצאים מהactivity וכשהיא נקראת עוצרים את המשחק(stop mode) |
| public List<Record> createRecordListForShow() | פעולה שיוצרת רשימה של השיאים ואם 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 שלוחצים על הכפתור btnOrder. |

**SettingActiviry:**

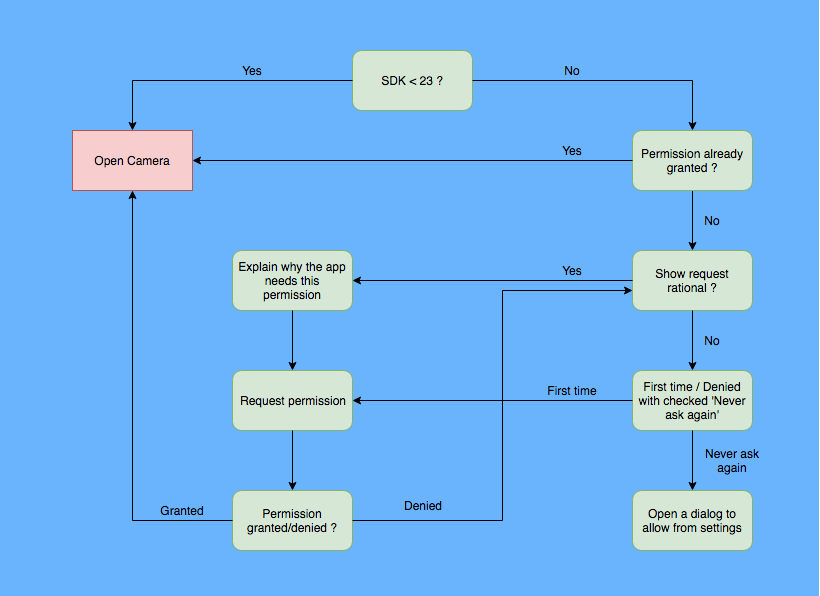
מחלקה של activity של הגדרות שמתחברת לקובץ xml – activity\_setting.xml ובה יש הגדרות לאפליקציה של שינוי גודל הלוח ושינוי הצבע של האריחים בלוח.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| Spinner spinnerSize | ספינר שאחראי להציג את כל הגדלים שאפשר לשנות אליהם. |
| Spinner spinnerColor | ספינר שאחראי להציג את כל הצבעים שאפשר לשנות אליהם. |
| 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 ל init() |
| private void init() | פעולה שמאתחלת את האובייקטים ומקבלת מהSharedPreferences את ההגדרות הנוכחיות וקוראת ל createSpinnerOfSize() ול createSpinnerOfColor() |
| public boolean onOptionsItemSelected(@NonNull MenuItem item) | מה קורה כשלוחצים על הכפתור חזרה שנמצא למעלה בAction bar. |
| 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 בספינר של הגדרת הגדלים או צבע בספינר של הגדרת הצבעים. |
| public void onNothingSelected(AdapterView<?> adapterView) | פעולה שנקראת מתי שלחצת על אחד הספינרים ולא בחרת כלום, לא הכנסתי בה כלום אבל חייבים לממש אותה אז היא כתובה |

**SmsActiviry:**

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 שיכולה לערוך בקובץ באחסון הפנימי |
| EditText message | אובייקט שבו כותבים את ההודעה למפתח |
| Button sendSms | כפתור שכשלוחצים עליו האפליקציה מבקשת הרשאה ואז אחרי שנותנים הרשאה 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) | מה קורה כשלוחצים על הכפתור חזרה שנמצא למעלה בAction bar. |
| public void onClick(View v) | פעולה שמופעלת כשלוחצים על הכפתור sendSms. |
| 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() ואם המשתמש דוחה את הבקשה ומנסה לשלוח שוב קוראים לפעולה dialog() להסביר למשתמש למה הוא לא יכול לשלוח את ההודעה ואם הוא דוחה עוד פעם מפנית אותו להגדרות המערכת של האפליקציה( תרשים של run time permission מצורף פה בעמוד הבא) |
| public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) | פעולה שמופעלת אחרי בקשת הרשאה והמשתמש אישר או שהוא לא אישר |



Send sms

קבצי קוד – מחלקות העצמים

**Record:**

מחלקה המייצגת שיא והתכונות שלו הם: הזמן שלקח לפתור, כמות המהלכים שהוא עשה והתאריך של אותו יום.

|  |  |
| --- | --- |
| 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 בהתחברות לבסיס נתונים מקומי בשביל להכניס אליו את השיאים של המשחקים ובשביל לקחת אותם מהבסיס נתונים בתור רשימה.

|  |  |
| --- | --- |
| 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 בsql 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 גם קוראים בsuper() לפעולה הבונה של SQLiteOpenHelper וגם 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 recycle view 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 הrecycle view. |
| 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 את שיציג את השיאים בצורה שכתובה בxml records\_layout בתוך הrecycle view. |
| public void onBindViewHolder(RecordViewHolder holder, int position) | פעולה שנקראת כמספר השיאים שנמצאים ברשימה ומכניסה אותם אחד אחרי השני בrecycle view. |
| public int getItemCount() | פעולה שמחזירה את הגודל של הרשימת שיאים. |
| public RecordViewHolder(View itemView) | פעולה שנמצאת בתוך הclass RecordViewHolder ומאתחל כל בשביל כל view את הרכיבים שנמצאים בתוכו. |

**Square:**

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

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| BoardGame boardGame | תכונה שמייצגת את הלוח שבו נמצא הריבוע. |
| Tile tile | תוכנה 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 והגודל שלו: מיקום בx,y וגודל 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 אריחים.

|  |  |
| --- | --- |
| 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 והגודל שלו: מיקום בx,y וגודל בw,h. |
| int color | תכונה שמייצגת את הצבע של האריח. |
| int number | תכונה שמייצגת את המספר שעל האריח, בסידור הסופי צריך שכל אריח יהיה על ריבוע עם אותו מספר. |
| 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 הpaint 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. |

**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 15 אריחים ובמקום של הריק 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 בclass game. |
| static Time time | מופע של המחלקה Time שיורשת מThread ואחראית על הטיימר של המשחק, היא סטטית מכיוון שרוצים להשתמש בה גם בclass 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 canvas, פה קוראים 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 temp. |
| public void swap() | פעולה שאחראית לערבב את המערך על ידי זה שהיא מגרילה כל פעם מספר, בודקת שהוא לא נמצא כבר במערך עם הפעולה findInArray ואם הוא עוד לא נמצא היא מכניסה אותו למערך.(**מצורף קוד מתחת לטבלה**) |
| public boolean isSolvable(int[] puzzle) | פעולה עם אלגוריתם מסובך שמקבלת מערך חד ממדי עם הסדר של הבלגון ובודקת אם הבלגון פתיר, בקצרה היא קודם כל מוצאת את מספר ההיפוכים במערך(היפוך אחד זה מספר אחד שיותר קטן ממספר אחר שנמצא מלפניו 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) ובאיזה שורה נמצא הריק ואז היא בודקת לפי 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 ובנוסף קוראת לפעולה blank() בשביל למצוא את החור. בפעים 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 שעל אחד יש אריח שלחצו עליו והשני זה הריבוע של המקום הריק ומזיזה את האריח לריבוע של המקום הריק בשתי שלבים: אחד היא מגדירה שהתכונה tile באובייקט to יהיה הtile שנמצא באובייקט from והשני זה 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.  (**מצורף קוד מתחת לטבלה**) |
| public boolean isWin() | פעולה שבודקת אם הלוח פתור.  (**מצורף קוד מתחת לטבלה**) |
| public Square findSquare(float x, float y) | פעולה שמקבלת x וy ומחזירה את הsquare שנמצא שם. |

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 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;  
}

**Game:**

המחלקה המרכזית של הפרויקט והיא מייצגת את המשחק ואחראית לכל הלוגיקה שלו והשינויים בו.

|  |  |
| --- | --- |
| כותרת הפעולה/תכונה מתוך java | תיאור הפעולה/תכונה |
| private BoardGame boardGame | תכונה שמייצגת את הלוח של המשחק. |
| Dialog solvedD | הדיאלוג שקופץ כשפותרים את הלוח. |
| Button btnSolved | הכפתור שנמצא בדיאלוג solved שכשלוחצים עליו הוא מאתחל את המשחק. |
| TextView tvTime, tvMoves | הרכיבים שעליהם כותבים את הזמן במשחק 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 ובה נשים את האובייקט של הלוח(boardGame). |
| boolean ifOne = true | תכונה שבודקת אם זה פעם ראשונה שנוגעים בלוח שאז צריך להפעיל את הthread Time ולהיכנס לrun mode |
| Boolean ifStart=false, ifPause = true | תכונות שמעדכנות אותנו באיזה מצב אנחנו במשחק ומשתנות לפי המצב הנוכחי במשחק, ifStart מייצג אם המשחק כבר התחיל וifPause מייצג אם המשחק נמצא בעצירה או לא. |
| int moves | תכונה שמייצגת את מספר המהלכים במשחק. |
| int sizeOfBoard | תכונה שמייצגת את הגודל של הלוח שהוגדר. |
| int colorOfTile | תכונה שמייצגת את הצבע של האריחים שהוגדר. |
| Handler handler | תכונה שאחראית לקבל את ההתקדמות של הזמן 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, קוראת לstartMode() לupdate() בשביל לעדכן את ההגדרות ול createBoardGame() בשביל ליצור את הלוח של המשחק. |
| public void createBoardGame() | פעולה שאחראית ליצור את המשחק, היא קודם כל מוציאה או את ההגדרות של הצבע והגודל מהאחסון הפנימי ואם זה פעם ראשונה שהמשחק מופעל היא מגדירה את הגודל ל4 על 4 ואת הצבע לורוד ולאחר מכן היא בונה את הלוח לפי הגודל של הרוחב של המסך. |
| public void doHandler() | פעולה שאחראית לקבל messages מהThread Time שבתוכם יש את ההתקדמות של הזמן ומתי שifStart = false היא מאתחלת את הזמן ל 00:00.0. |
| public void startMode() | פעולה שמכניסה את המשחק לstart mode כלומר למצב התחלתי לפני שהמשתמש התחיל לפתור ובו מאפסים את כל הדברים של המשחק כמו ההתקדמות בזמן או המהלכים ומאתחלים את האובייקט time. |
| public void runMode() | פעולה שמכניסה את המשחק למצב ריצה כלומר מתי שהמשחק 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 זה ifStart = true ו ifPause = true |
| public boolean onTouch(View view, MotionEvent event) | פעולה מובנת של view ומופעלת על האובייקט boardGame ואחראית לטפל במצב שהמשתמש נוגע בלוח, בפעם הראשונה הוא מריץ את הטיימר וקורא לפעולה runMode() ובפעמים הבאות הוא רק בודק אם המשתמש לחץ על ריבוע שלידו נמצא החור ואם כן אז הוא מזיז את האריח שעל הריבוע הזה לחור עם הפעולה של boardGame slide() ולאחר מכן בודק אם הלוח פתור עם הפעולה של boardGame isWin() ואם הלוח באמת פתור הוא מקפיץ את הדיאלוג solved עם הפעולה createSolvedDialog() ומכניס את השיא החדש לבסיס נתונים עם הפעולה addNewRecord().(**מצורף קוד מתחת לטבלה**) |
| public void addNewRecord() | פעולה שמחברת לבסיס נתונים ומכניסה לו שיא חדש. |
| public void update() | פעולה שמעדכנת את ההגדרות מהאחסון הפנימי של הגודל של הלוח, הצבע של האריחים והסידור של השיאים בבסיס נתונים. |
| public void createSolvedDialog() | פעולה שפותחת דיאלוג ומאתחלת את הרכיבים וההגדרות שלו כשהלוח פתור. |

@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;  
}

**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 וצריך לקדם את הזמן. |
| Handler handler | תכונה שמקבלת את ההתקדמות של הטיימר ושלוחת אותו לhandler במחלקה game בשביל שיקדם את הטיימר. |
| int milliSeconds, seconds, minute | תכונות שמייצגות דקות, שניות ומילי שניות |
| public Time() | פעולה בונה שמאתחלת את המשתנים milliSeconds, seconds, minute ל0 |
| public void run() | פעולה מובנת שחייבים לממש כשיורשים מthread והיא מופעלת מתי שקוראים לפעולה start() שלthread ובתוכה בודקים אם המשחק רץ ואם כן כל פעם מחכים 0.1 שניות ואז מקדמים את המשתנים ושולחים את ההתקדמות בתוך massage לhandler. (**מצורף קוד מתחת לטבלה**) |

@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  
 }  
  
 }  
 }  
}

**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 והיא יוצרת את כל הרכיבים ובונה את ההתראה ולאחר מכן 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,

סוגי אמולטורים - Pixel\_2\_API\_28,

**תיאור אופן השימוש בפרוייקט:**

לוחצים על האייקון של האפליקציה והמשחק 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 מזיזים את החלקים? אז ככה: בלוח יש מקום אחד שאין עליו אריח וזה המקום הריק ואם רוצים להזיז אריח צריך להזיז אליו על ידי לחיצה על האריח שלידו, אי אפשר להזיז אריחים שלא נמצאים ליד המקום הריק, כל הזזה של אריח מספר המהלכים עולה ובנוסף יש טיימר שסופר כמה זמן לקח לך לפתור את הלוח. ברגע שהצלחת לפתור קופץ דיאלוג שמודיע לך שפתרת את המשחק ובשביל לשחק שוב צריך ללחוץ על הדיאלוג. בנוסף גם אחרי שפתרת את הלוח הזמן שלקח לך והכמות מהלכים נשמרת במערכת ותוכל לראות את כל התוצאות לאחר מכן.

1. אחרי שהתחלת לשחק אתה תוכל תמיד לעצור את הזמן על ידי לחיצה על הכפתור pause שיהפוך לcontinue ואם תרצה להמשיך תצטרך ללחוץ על continue והמשחק ימשיך בנוסף אם תצא מהאפליקציה בלי לסגור אותה המשחק ייעצר ותוכל להמשיך אותו בדיוק מאותה נקודה שהפסקת.
2. תוכל גם לבלגן את הלוח שוב על ידי לחיצה על הכפתור start אם הבלגון לא מוצא חן בעיניך או שתרצה להתחיל משחק חדש בלי לגמור את המשחק הקודם.
3. תוכל גם לגשת למסך ההגדרות על ידי לחיצה על השלוש נקודות בפינת המסך ואז לחיצה על setting ושם 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 😉), בנוסף תוכל לשנות את הצבע של האריחים לשלל צבעים שונים ולאחר מכן תוכל לחזור למסך הראשי עם ההגדרות החדשות על ידי לחיצה על החץ למעלה או על ידי לחיצה על כפתור החזרה המובנה
4. אפשר גם לראות את כל השיאים שלך על ידי לחיצה על השלוש נקודות בפינת המסך ואז לחיצה על record ואז ייפתח דיאלוג שמציג את 10 השיאים הכי טובים מסודר לפי מהלכים ותוכל גם על ידי לחיצה על הכפתור שיופיע בלמעלה של הדיאלוג לשנות שהשיאים יהיו מסודרים לפי הזמן שלקח לפתור אותם.
5. בנוסף תוכל גם לשלוח הודעה למפתח על ידי לחיצה על השלוש נקודות בפינת המסך ואז לחיצה על contact us ואז ייפתח מסך שבו תוכל לכתוב הודעה ולשלוח אותה למפתח, לאחר שתשלח את ההודעה האפליקציה אוטומטית תחזיר אותך למסך הראשי של האפליקציה.

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!