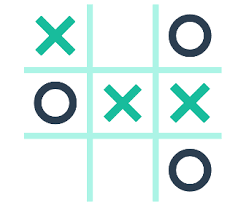
****ספר פרויקט גמר י"ב

שם האפליקציה:Tik Tak Tok



שם המגיש/ה: שירה לוינסון

תעודת זהות: 326173309

שנת הלימודים: 2022

בית ספר: קציר – על יסודי משגב (640565)

שם המנחה: הדס יגאל BSc הנדסת תוכנה

תוכן עניינים

# 

# מבוא

האפליקציה שהכנתי היא משחק איקס עיגול. בחרתי לממש את המשחק הזה מכיוון שתמיד אהבתי לשחק אותו עם חברים, וחשבתי שיהיה מהנה מאוד לממש את האלגוריתם שעומד מאחוריו. בנוסף המשחק שלי מאפשר לחברים לשחק אותו אחד עם שני מרחוק.  
איקס עיגול (או בשמותיו באנגלית: noughts and crosses = בריטית, או tic-tac-toe = אמריקאית), הנקרא גם איקס מיקס דריקס הוא משחק לשני שחקנים המשוחק על לוח של 3X3 משבצות. כל שחקן בתורו מסמן סימן באחת המשבצות (לרוב איקס או עיגול, ומכאן שם המשחק) השחקן המנצח הוא זה שהצליח למלא שורה, טור או אלכסון בסימנים שלו. אם אף שחקן לא הצליח לעשות זאת עד שכל המשבצות מלאות, המשחק נגמר בתיקו.

בדרך כלל נהוג להתחיל במשבצת קיצונית וזאת כדי ליצור בהמשך שורה וטור (או שורה ואלכסון, וכדומה) שבכל אחד מהם שני סימונים, כאשר אם יבחר השחקן השני למנוע אחד מהם, ינצח השחקן המתחיל בעזרת שני הסימונים האחרים (לדוגמה: השחקן ראשון שם ב-A1, A3, ו-C3). בשל טכניקות אלו בדרך כלל מגיע השחקן שהתחיל שני למצב של הפסד או תיקו.

במונחי תורת המשחקים, המשחק הוא משחק פתור: לשני השחקנים במשחק יש תכסיס תיקו. כלומר, כל שחקן יכול במקרה הגרוע להבטיח לעצמו תיקו אם ישחק משחק מושלם. לפיכך, אם שני השחקנים משחקים בצורה הטובה ביותר, הם יגיעו לתיקו.

באפליקציה שלי כל שחקן יכול לשחק עם חבר, בשני טלפונים שונים, והאפליקציה תשמור את תוצאותיהן של השחקנים.

מאז שחשבתי סתם על הרעיון לעשות את המשחק הזה באפליקציה עברו שנתיים ואני בכיתה יב' ונודע לנו שצריך לעשות פרויקט שהוא לבנות אפליקציה משלך ואז ישר הכל התחבר לי והתחלתי לעבוד. באפליקציה שלי ממש אפשר לראות את המשחק מבוסס אבל עם שינויים משלי, כל משתמש שנכנס לאפליקציה מתחיל במסך הראשי שבו הוא יכול כל פעם מחדש לשים לעצמו איזה שם שבא לו.

ואז יש לו שתי אופציות ליצור חדר משחק או להצטרף לאחד קיים דרך קוד. אם ובחר ליצור חדר משחק אז הוא יוכל לשחק עם ההגדרות של אותו משחק, לבחור עם איזה שחקן לשחק ומי לאתחל את הלוח. ואם בחר להצטרף לחדר משחק קיים דרך קוד הוא פשוט יחכה שמי שיצר את אותו החדר יתחיל את המשחק.

בשביל להתחיל משחק חייב 2 משתתפים בלבד. משתמש גם לא יוכל להכנס לחדר משחק אם השם שבחר כבר קיים או אם החדר מלא.

**פעולות נוספות של האפליקציה:**

* מוזיקת רקע
* התראה על חוסר חיבור לאינטרנט

בספר זה אציג את המסמכים, המחלקות, מבנה נתונים של האפליקציה וגם אסביר כיצד להשתמש בה.

**סביבת עבודה:** android studio

**גרסת האנדרואיד סטודיו:** 4.1

**שפת תכנות:** java

קישור לרשימות תוכנה מלאות באתר גיטהב: <https://github.com/ShiraLev100/Yud-Bet-Project.git>

# מדריך למשתמש

# ברקוד להורדת האפליקציה:



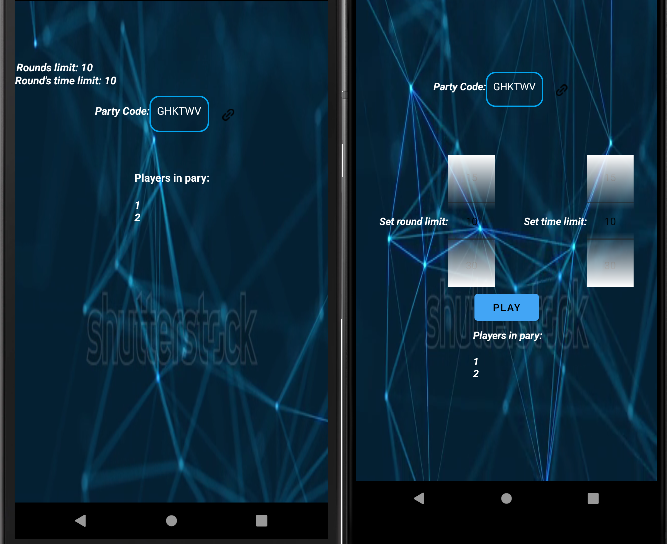
# תיאור מסכי האפליקציה

**מסך הבית-** בו אפשר להכניס איזה שם כרצון המשתמש, ליצור חדר משחק או להצטרף לאחד קיים דרך קוד ולבחור בתפריט האם להפעיל מוזיקת רקע שתפעל בזמן המשחק או להכנס למדריך שמסביר למשתתף איך עובד המשחק.



**(מרגע זה ועד שיסתיים המשחק המשתתפים רואים מסכים שונים)**

**מסך הצטרפות לחדר משחק-** בו אפשר לראות את הגדרות המשחק, את המשתתפים שהצטרפו ולהעתיק את קוד החדר.



**מסך יצירת חדר משחק-** בו אפשר לשנות את ההגדרות של המשחק, לראות את המשתתפים שהצטרפו להעתיק את קוד החדר וללחוץ על הכפתור שמתחיל את המשחק.

**מסך הבוחר-** בו אפשר לשלוח את הנושא לכל המשתתפים ולראות את הציורים שלהם ברגע שתם הזמן (הוא גם רואה את הזמן רץ) ולבסוף לדרג אותם בלחיצה על הציורים על פי הסדר מהמקום הראשון ועד האחרון.

**מסך המציירים-** בו אפשר לצייר את הנושא כאשר נשלח עד לרגע תום הזמן המוקצב. אפשר לבחור צבע, לבחור גודל מברשת צבע ולחזור אחורה בציור.

תמונה שמכילה טקסט, צג, צילום מסך

התיאור נוצר באופן אוטומטי

**מסך הצופה-** בו אפשר לראות את הבוחר מדרג את המשתתפים ואת הציורים שלהם.

תמונה שמכילה טקסט, חוץ

התיאור נוצר באופן אוטומטי

**מסך הבוחר-** דוגמה לאיך זה נראה כשהוא מדרג.

**מסך טבלת הדירוג לאחר כל סבב-** בו אפשר לראות את הניקוד הכללי של כל אחד מהמשתתפים ולאחר ספירה לאחור של 10 שניות עוברים לסבב הבא.

**מסך טבלת הדירוג הסופי-** בו אפשר לראות את הניקוד הכללי והסופי של כל אחד מהמשתתפים ולאחר ספירה של 20 שניות לאחור חוזרים למסך הבית מה שמסיים את המשחק.

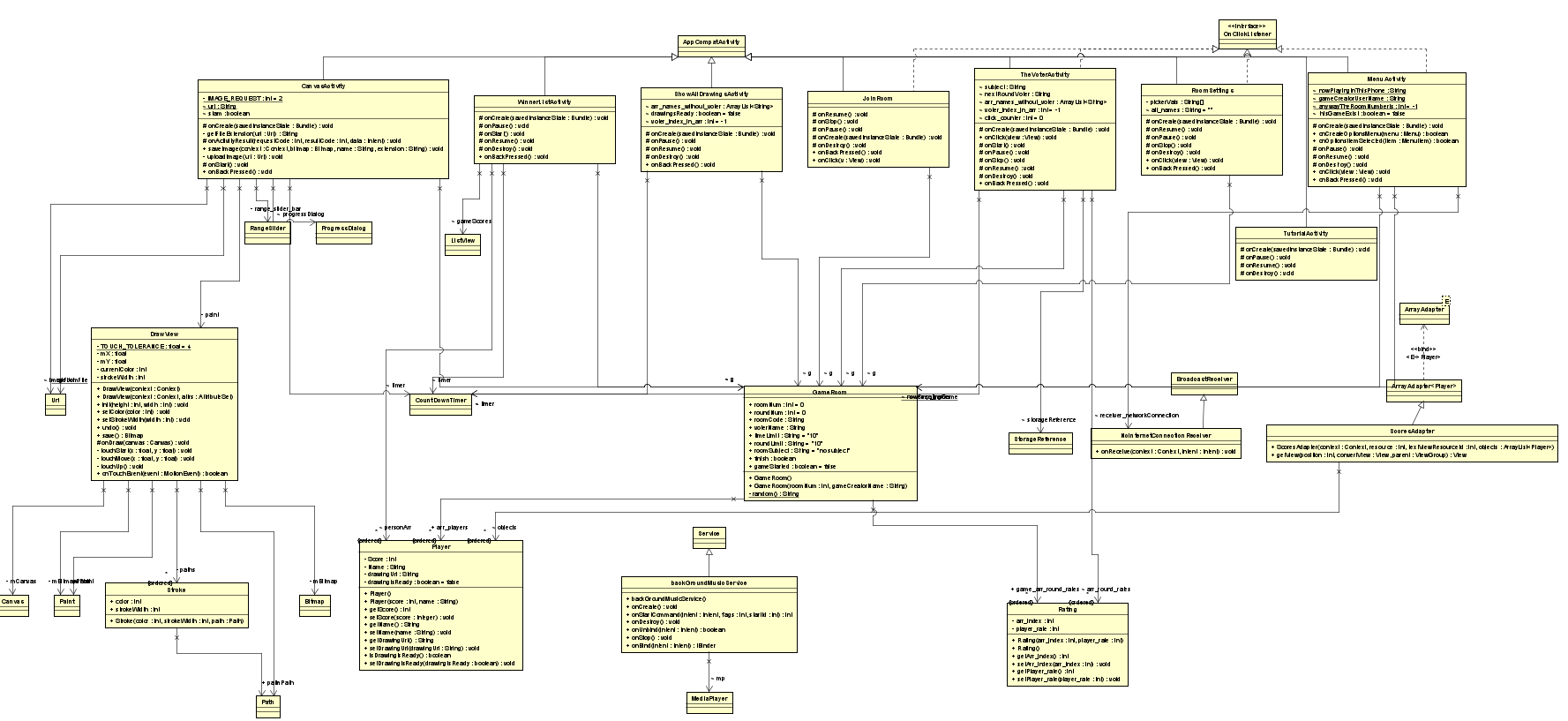
תמונה שמכילה טקסט, סצינה, במה, לייזר

התיאור נוצר באופן אוטומטי

תמונה שמכילה סצינה, לייזר

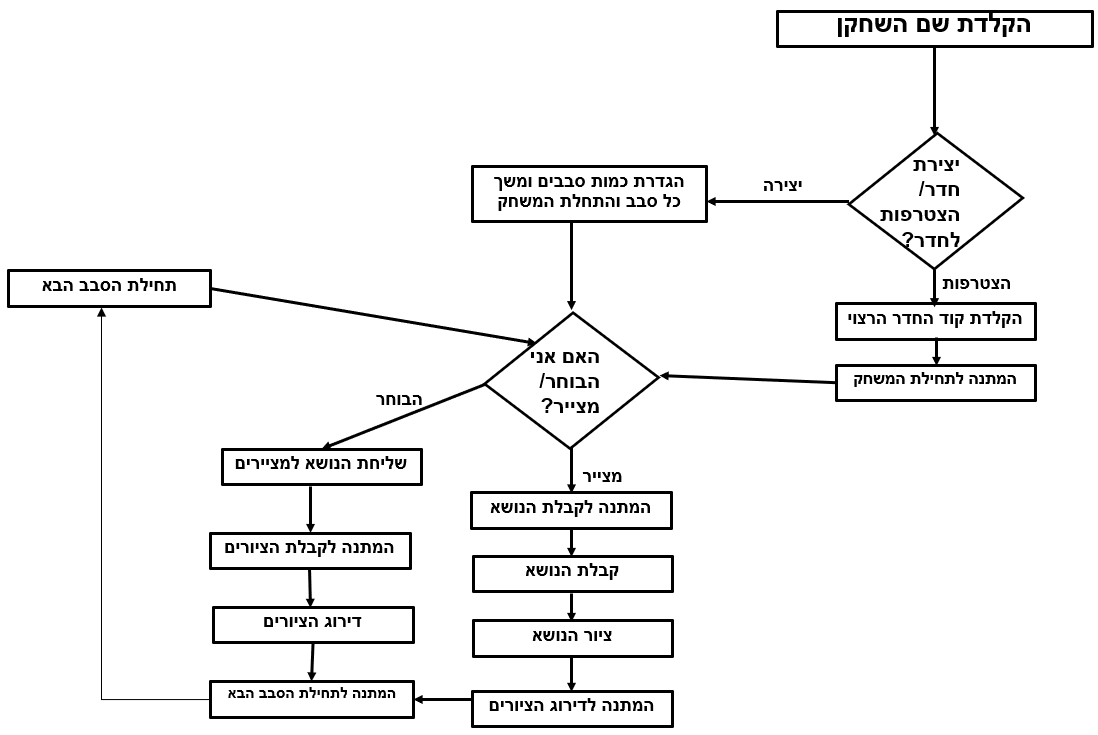
התיאור נוצר באופן אוטומטי

# תרשים מחלקות UML



# 

# תרשים זרימה



כן

האם נגמרו הסבבים?

לא

**והצגת טבלת הניקוד**

# 

# עמידה בדרישות הפרויקט check list

|  |  |
| --- | --- |
| **הדרישה** | **פירוט מענה על הדרישה בפרויקט** |
| לפחות 5 מסכים שונים | **המסכים שלי הם:** |
| לפחות 5 מחלקות | **המחלקות שלי הן:** |
| שימוש בתפריטים | **תפריטים אצלי בפרויקט:**  יש לי תפריט ב action bar, הפריטים שלו הם מוזיקת רקע ומדריך איך לשחק. |
| שימוש בתיבות דו שיח | **תיבות דו שיח אצלי בפרויקט:**  הדיאלוג משמש להזנת קוד החדר אליו מעוניינים להצטרף. |
| service | **אצלי בפרויקט מימשתי את הservice הבא:**  backgroundMusicService |
| intent | **אצלי בפרויקט הintent הבאים:**  בשימוש למעבר בין חלונות והפעלת הסרביס. |
| רכיבים מתקדמים חובה לפחות אחד מהרשימה הבאה:  · גרפיקה  · אנימציה  · שימוש במצלמה של הטלפון  · טלפוניה: חיוג, קבלת שיחה נכנסת, קבלת / שליחת SMS  · חיישני תנועה  · חיישני מיקום ומפות  · מיקרופון | **אצלי בפרויקט:**  **-גרפיקה**  יש לי מסך צייר כולל בחירת צבע ועובי קו ובנוסף אני מציג את תמונות של כל הציורים של כל המשתמשים. |
| לפחות אחד משני הנושאים הבאים:  Broadcast Receiver (= מחלקה אשר יכולה להאזין למסרים) לדוגמא:  · האזנה למצב הבטריה  · האזנה לשיחה נכנסת או יוצאת  · האזנה לשעון  · האזנה לכניסה של sms  Content provider (= ספקי תוכן שמאפשרים שיתוף נתונים בין אפליקציות. במידה ואתם מעוניינים שהאפליקציה שלכם תשתף נתונים עם אפליקציות אחרות אזי נפתח ספק תוכן)  לדוגמא :  · אנשי קשר  · יומן שיחות  · הגדרות התקן  · יומן לוח שנה  · ועוד | **דוגמא לBroadcast Receiver בפרויקט שלי:**  האזנה למצב האינטרנט - NoInterntConnectiontReceiver |

# הסבר מפורט על קטעי קוד

1. **Service**

בפרויקט השתמשתי בservice בשביל מוזיקת רקע לאפליקציה גם כאשר המסכים מתחלפים או משתנים המוזיקה ממשיכה להתנגן.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

1. **Broadcast Receiver**

זו יכולת של האנדרואיד להאזין כל הזמן ובכל המסכים של האפליקציה לאירועי מערכת שמעניינים אותנו. בפרויקט שלי ה Broadcast Receiver משמש לבדיקת מצב האינטרנט של הטלפון, במידה ואין חיבור לאינטרנט ובמידה ויש חיבור תופיעה הודעה למשתמש.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

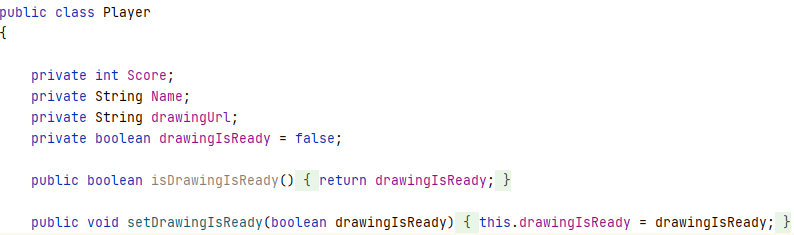
.

1. **מבנה נתונים עיקריים**

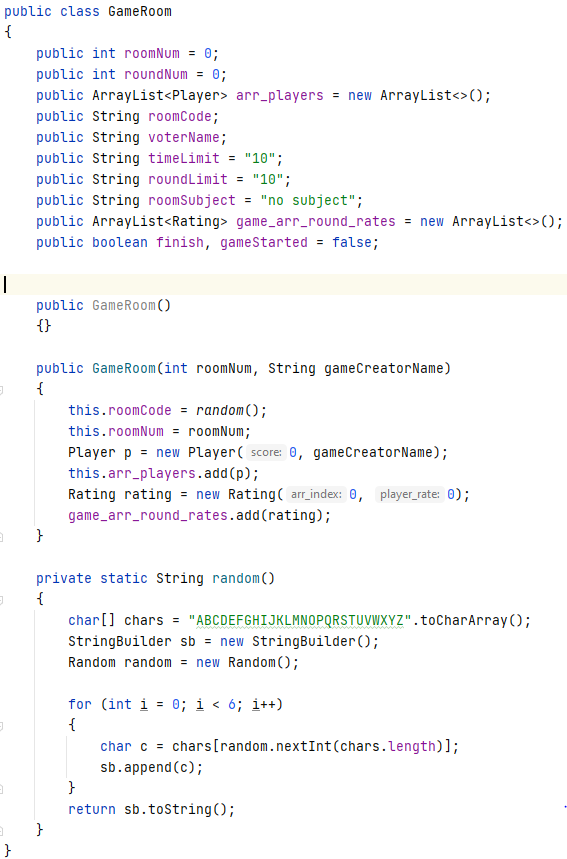
המחלקות המהותיות ביותר הן**Player** וה-**GameRoom**

**Player:**

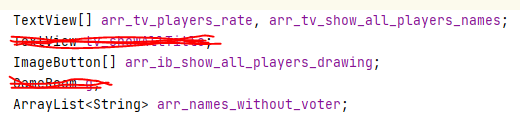
וכמובן יש למחלקה פונקציה בונה setter ו- getter.

ניתן לראות בקוד שכל שחקן יודע מה הניקוד שלו, מה השם שלו ומה כתובת האינטרנט בה שמור הציור האחרון שלו(הכתובת של הstorage-).

**: GameRoom**

ניתן לראות בקוד את התכונות והפעולות של חדר המשחק.

בנוסף יש לי מערכים שבהם אני שומר את השמות של השחקנים, הציורים שלהם והדירוגים שלהם.

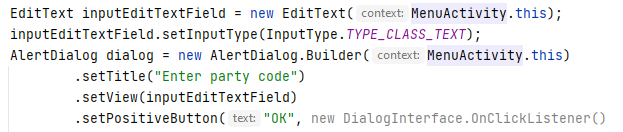


**4)** **הסרת חדר משחק שלם מהפיירבייס**

ניתן לראות בקוד שכאשר ונגמרים הסבבים פועלת הפעולה להסרת חדר המשחק מהפיירבייס.

מה שהפעולה עושה זה בעצם הולכת לכל המשחקים הקיימים בפיירבייס מחפשת את מספר החדר הנוכחי ומסירה את הערך שלו מהפיירבייס.

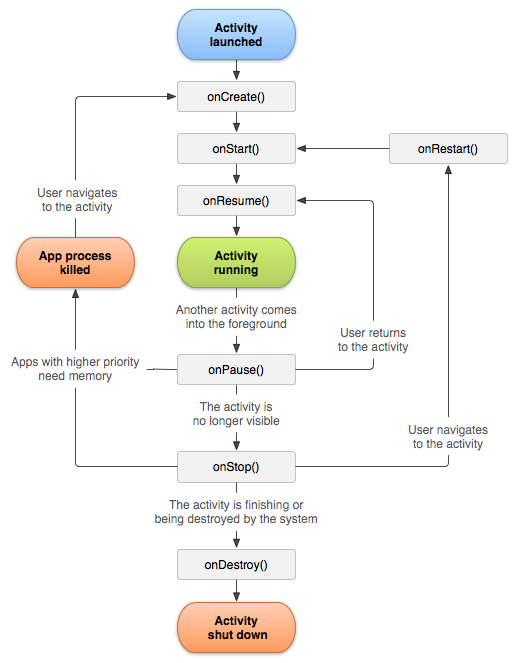
**5) כתיבת טקסט בתוך דיאלוג**

ניתן לראות בקוד איך נוצר תיבת טקסט בתוך הדיאלוג.

**6) CountDownTimer**

ה- CountDownTimerהיא פעולה שרצה במקביל לכל מה שקורה ולא תוקעת או מפריעה למסביב.

בקוד ניתן לראות שהפעולה רצה על פי הזמן שהוקצב בתחילת המשחק ויחידות מידה של מילי השנייה כל "טיק" שזה כל שנייה אחת שעוברת או כל 1000 מילי השנייה רואים כמה זמן נשאר ובסיום הפעולה הבוחר יכול לדרג את הציורים.

**7) activity life cycle**

האיור בתמונה מציג את מחזור החיים של ה activity-, בפרויקט שלי התעוררה בעיה של ריענון תצוגה לא רצוי משהו שגרם להבהובים. כשדיבגתי את הבעיה הבנתי שהבעיה נוצרת משום שאני ממשיך להאזין לחדר אחרי שכבר עברתי חלון וזה גורם לאירוע ולתגובה של הבהוב. הפתרון שלי היה להגדיר את האזנה בפעולת onResume ולהסיר את האזנה בפעולת onStop וכך ברגע שאני עובר חלון אני כבר לא מאזין.

**8) Custom array adapter**

משמעות המילה adapter היא מתאם. מטרת המתאם היא להציג בlistView- את המידע שנמצא במערך, כלומר לתאם בין מערך שהוא מבנה נתונים רעיוניים לבין ה-listView שנמצא על המסך שהמשתמש יכול לצפות בו.

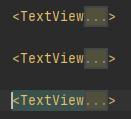
בפרויקט שלי אני מציג את השמות, הנקודות והמקום (מקום כלומר מקום ראשון, מקום שני וכו') של השחקנים.

תמונה שמכילה טקסט

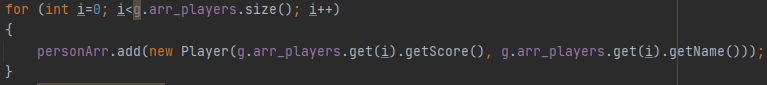
התיאור נוצר באופן אוטומטי

**שלבי העבודה שלי היו:**

**שלב 1:**

לבנות קובץ xml שמתאר מה המבנה של כל שורה באותה טבלה.

**שלב 2:**

הכנתי מערך שבו אני רוצה להציג את הנתונים בטבלה.

**שלב 3:**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**כתבתי קובץ java שמסביר איך הadapter ממלא את השורות של הטבלה.

**שלב 4:**

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיאני ממיין את השורות שיש בטבלה לפי הניקוד שיש לשחקנים.

**שלב 5:**

אני מבקש מהמתאם לקחת את המערך ולהציגו ב-listView.

**#הערה:**

יכולתי גם להשתמש בadapter בסיסי שהשפה נותנת לי אבל העדפתי לעצב בעצמי את הטבלה ולכן החלטתי לכתוב את ה adapter בעצמי.

**9) להסביר היטב איך עובד כל עניין הציור**

**10) קובץ המניפסט**

תמונה שמכילה טקסט

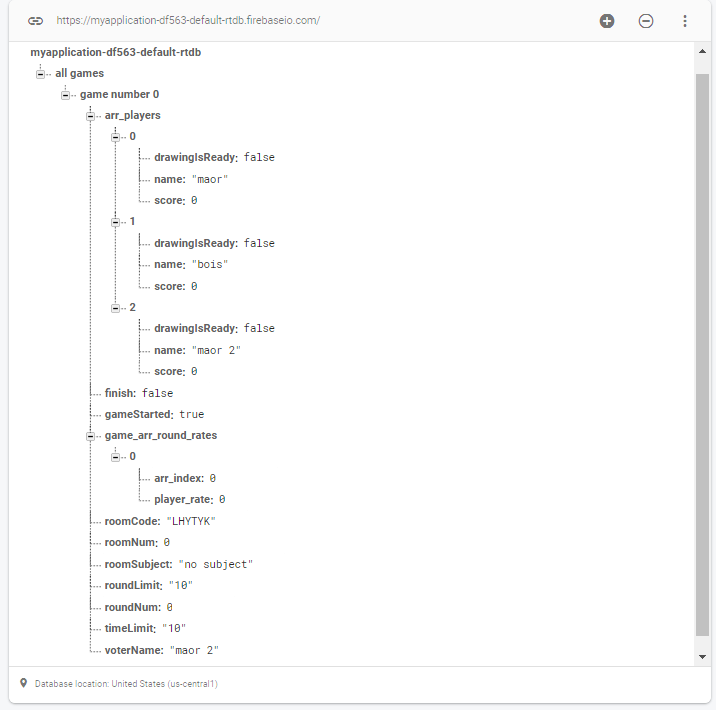
התיאור נוצר באופן אוטומטימטרתו של קובץ זה לתאר את כל ה activities, Broadcast Receivers, services וההרשאות שקיימות אצלי בפרויקט.

**11) הפיירבייס**

הפיירבייס הוא שרת חינמי של גוגל שמאפשר לנו שמירת נתונים, תמונות וכו' בלי התעסקות **בכתיבת קוד צד השרת. אני השתמשתי בפיירבייס כדי לאחסן את כל חדרי המשחק שלי וכדי שהשחקנים יוכלו לשחק זה עם זה ממכשירים שונים.**

# מבנה הנתונים בפיירבייס

בפרויקט השתמשתי בfirebase כמבנה נתונים שיכול לקשר בין אנשים ומרכיבים באפליקציה.

השתמשתי גם בstorage של firebase בשביל לשמור את התמונות של הציורים.

# 

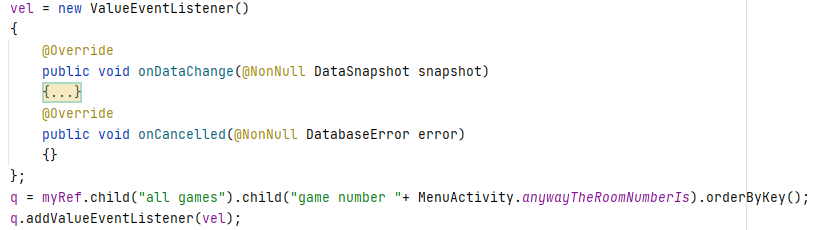
תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

**דוגמה לקוד:**

****

**בקוד זה אני ניגש לפיצול ספציפי בפיירבייס שמתייחס לחדר משחק מסוים ואני מעדכן אותו בעזרת פקודת setValue.**



**בקוד זה אני מאזין לאירועים שמתרחשים בחדר מסוים.**

**דוגמה לאירועים בקוד שלי:**

* **הצטרף שחקן חדש למשחק**
* **הועלה ציור לשרת**
* **הבוחר חילק ניקוד**

לקוד האזנה לאירועים יש שני סוגים:

**הסוג הראשון והפשוט יותר מבין השניים נקרא SingleEventListener ומה שהוא עושה זה מאזין לאירוע פעם אחת וזהו לא יותר. והסוג השני נקרא valueEventListener שלא כמו הסוג הראשון מאזין כל פעם שמתחרש אירוע, ניתן לעצור את ההאזנה עזרת הקוד-**

 קוד לאחסון בפיירבייס:





**בקוד הנ"ל ניתן לראות כי אני ניגש לאחסון בפיירבייס ובתוכו אני ניגש לפיצול ספציפי שמתייחס לחדר משחק מסוים ומעלה את השם של השחקן ביחד עם הקובץ URI של ציורו.**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיקישור בין הציורים שב-*storage* של פיירבייס לבין ה-*real-time-database* של פיירבייס:**

**בקוד הנ"ל ניתן לראות כיצד אני מקשר את הציור** שב-**storage** **של פיירבייס ל-real-time-database** של פיירבייס **בכך שאני ניגש לקישור URL של הקובץ URI** שב-**storage** ואז מחפש את השחקן המתאים לאותו קובץ וניגש לפיצול שלו ב-  **real-time-database**של הפיירבייס ומעדכן בעזרת פקודת **setValue**.

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיהנה דוגמה לאיך זה נראה ב- **real-time-database**

# תמונה שמכילה טקסט התיאור נוצר באופן אוטומטיקבצים נלווים-תמונות, צבעים ומחרוזות

**תמונות של האייקונים באפליקציה**

**קובץ ווידאו ומוזיקה באפליקציה**

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

**צבעים באפליקציה**



**מחרוזות באפליקציה**

# סיכום

ממש נהניתי לבנות את האפליקציה שלי, לפתח מוצר מתחילת הרעיון הראשוני ועד הביצוע והתוצר הסופי. הפרויקט הזה היה מאוד משמעותי עבורי משום שזאת פעם ראשונה שפיתחתי אשכרה אפליקציה משלי שעובדת ממש והכל אם זה לפתח רעיון ולתכנון, לעצב את המבנה הכללי של האפליקציה, לעצב את כל החלונות, ליצור מבנה נתונים מתאים ולכתוב את הקוד של כל החלקים.

אבל לא הכל ורוד בעולם התכנות ובפרויקט שלי התעוררה בעיה של ריענון תצוגה לא רצוי משהו שגרם ל"הבהובים" במסך המשתמש (הבעיה תקעה אותי כמעט חודש שלם עד שמצאתי את הפתרון). לבסוף מצאתי את הבעיה והבנתי שהיא נוצרת משום שאני ממשיך להאזין לחדר אחרי שכבר עברתי חלון מה שגרם לאירוע ולתגובה של ה"הבהוב". הפתרון שלי היה להגדיר את ההאזנה בפעולת onResume ולהסיר את אותה בפעולת onStop וכך ברגע שאני עובר חלון אני כבר לא מאזין יותר.

מהפרויקט הזה למדתי איך לתכנן אפליקציה ושיטות שונות ודרכים שונות לעצב חלונות. למדתי איך לתכנן מבנה נתונים עם כמה טבלאות והרבה משתנים שונים ואיך לעבוד איתם. ובעיקר קיבלתי מהפרויקט הזה הרבה תחושת מסוגלות וסיפוק מההצלחה והסיום של האתגר.

# נספחים- רשימות תוכנה מלאות