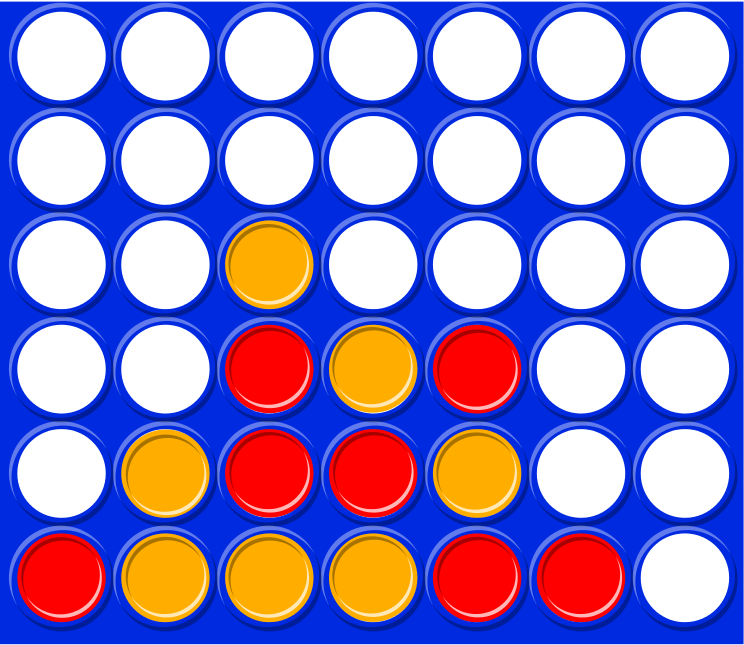
בס"ד



**ספר פרויקט המשחק**

**"ארבע בשורה"**

מגיש: דניאל עבאי

ת.ז: 207381724

מורה: אילן פרץ

מכללת קרית נוער ירושלים

תוכן עניינים

[הצעת פרויקט להנדסאים במגמת הנדסת תוכנה 2](#_Toc511229962)

[מדריך למשתמש 5](#_Toc511229963)

[מבוא 5](#_Toc511229964)

[דרישות למחשב והסביבה 5](#_Toc511229965)

[הוראות הרצת הפרויקט 5](#_Toc511229966)

[שימוש בתוכנה לקוח: 6](#_Toc511229967)

[13](#_Toc511229968)

[מדריך למתכנת 13](#_Toc511229969)

[כלי הפיתוח 13](#_Toc511229970)

[מודל שרת לקוח 13](#_Toc511229971)

[פרוטוקול תקשרות IP/TCP: 13](#_Toc511229972)

[שקע (תקשורת מחשבים): 14](#_Toc511229973)

[התחברות למוסד נתונים: 14](#_Toc511229974)

[פרוטוקול השיח: 15](#_Toc511229975)

[שימוש בתוכנת השרת: 16](#_Toc511229976)

[תרשים UML: 17](#_Toc511229977)

[רשימת המחלקות 18](#_Toc511229978)

[המחלקה Client 18](#_Toc511229979)

[המחלקה ClientGui 18](#_Toc511229980)

[המחלקה Utils 18](#_Toc511229981)

[המחלקה Sound 18](#_Toc511229982)

[המחלקה Server 18](#_Toc511229983)

[המחלקה GameLogic 18](#_Toc511229984)

[המחלקה UtilsDB 18](#_Toc511229985)

[המחלקה Place 18](#_Toc511229986)

[בעיות שהתמודדתי איתן: 19](#_Toc511229987)

[שיפורים עתידיים: 19](#_Toc511229988)

[סיכום 19](#_Toc511229989)

[תודות 19](#_Toc511229990)

[ביבליוגרפיה 19](#_Toc511229991)

[נספחים 20](#_Toc511229992)

סמל ביה"ס: 140129 שם ביה"ס: קרית נוער

# הצעת פרויקט להנדסאים במגמת הנדסת תוכנה

א. נושא: תכנון ופיתוח תוכנה למשחק “ארבע בשורה (Connect4) “עבור שני שחקנים.

ב. שם המנחה: אילן פרץ

ג. שם התלמיד: דניאל עבאי ת.ז: 207381724

ד. אופן ביצוע העבודה: עבודה ביחיד.

ה. אופי עבודת הפרויקט: תכנון אב טיפוס ובנייתו.

ו. מקום ביצוע פרויקט: קרית נוער.

ז. תיאור הנושא:

**מטרת הפרויקט:**

בניית משחק רשת לשני שחקנים, בשפת Java ובתכנות מונחה עצמים, תוך שימוש במודל שרת-לקוח (Client-Server) ופרוטוקול תקשורת TCP/IP (חבילת פרוטוקולי תקשורת שעליה מושתתת רשת האינטרנט).

הלקוחות הם השחקנים והשרת מנהל את המשחק והתקשורת בין השחקנים. לקוח ראשון מתחבר מחכה ללקוח השני שיתחבר למשחק. כל שחקן ישחק בתורו - בזמן שהשחקן הראשון משחק, השחקן השני מחכה עד שהשחקן הראשון יסיים בשביל להתחיל את תורו במשחק.

**תיאור המשחק:**

**ארבע בשורה** הוא משחק חשיבה לשני שחקנים. לכל שחקן יש אסימונים בצבע אחד, וכל שחקן בתורו מפיל אסימון

לתוך הלוח המורכב מ7 עמודות ו6 שורות.

בקישור הבא מופיעים כללי המשחק בשפה האנגלית: www.boardgamecapital.com/connect-four-rules.htm

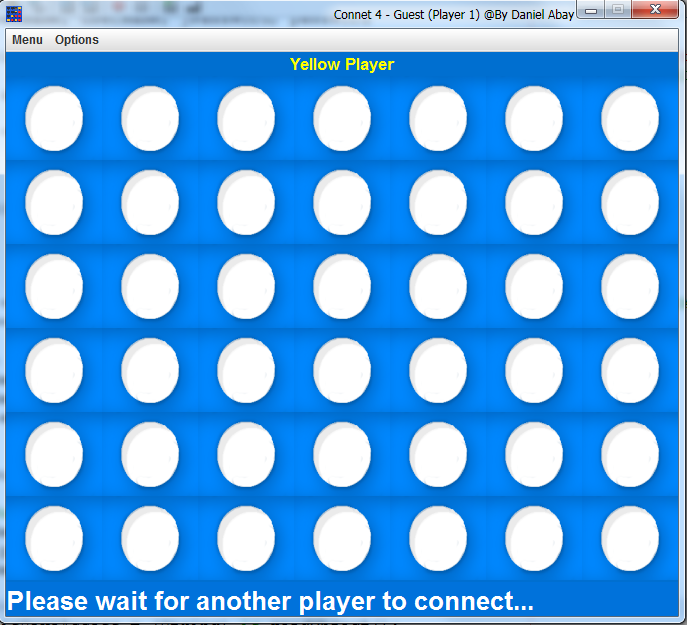
**הלוח והכלים:**

לוח המשחק הוא בגודל 7X6. במשחק יש 2 אסימונים, הם מחולקים לשני צבעים שונים(צהוב/אדום).

**המצב ההתחלתי:**

בחילת המשחק הלוח ריק.

בתמונה מופיעה המחשה של המצב התחלתי של הלוח אחרי ששחקן אחד התחבר.



**חוקי המשחק והמהלכים:**

חוקי המשחק, לכל שחקן יש תור לשים אסימון בצבע שלו על הלוח,

במטרה ליצור ארבע אסימונים זהים באותה שורה או אלכסון, והמטרה של השחקן הנגדי הוא למנוע מהשחקן השני ליצור את הרצף הזה ככה שהוא חוסם אותו עם אסימון בצבע שלו.

**מטרת המשחק וקביעת המנצח:**

המטרה היא למקם 4 חיילים מאותו צבע של השחקן בשורה, בעמודה או באלכסון. המנצח במשחק הוא השחקן אשר יצר רצף של ארבעה חיילים מאותו הצבע שייך לשחקן באותו קו.

ח. פירוט דרישות מהמערכת:

* תצוגת הלוח וקליטת המהלכים של המשתמש/ים.
* בדיקת חוקיות מהלכים- בדיקת חוקיות הנחת החלקים (עפ"י חוקי המשחק).
* ביצוע מהלך מתוכם על ידי המחשב באמצעות אלגוריתם של בינה מלאכותית.
* בדיקת סיום משחק.
* חישוב הניקוד של השחקנים (אנושי וממוחשב).
* הצגת תוצאת המשחק.

ט. מפרט טכני.

* מחשב המריץ את מערכת ההפעלה Windows XP ומעלה, זיכרון ראשי לפחות .1 GB
* מסך וכרטיס גרפי SVGA.
* דיסק קשיח
* עכבר + מקלדת
* התקנת Environment  Java Runtimeבגרסה 6 ומעלה.

י. לוח זמנים.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | אוקטובר | נובמבר | דצמבר | ינואר | פברואר | מרץ |
| ניתוח המערכת |  |  |  |  |  |  |
| תכנון מבני הנתונים ואלגוריתמים של האסטרטגיה |  |  |  |  |  |  |
| כתיבת הקוד |  |  |  |  |  |  |
| עיצוב ממשק המשתמש |  |  |  |  |  |  |
| ניפוי שגיאות ובדיקות איכות |  |  |  |  |  |  |
| הרצת התוכנית |  |  |  |  |  |  |

אישור המנחה\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ תאריך\_\_\_\_\_\_\_\_\_\_\_

אישור מרכז מגמה\_\_\_\_\_\_\_\_\_\_\_\_\_\_ תאריך\_\_\_\_\_\_\_\_\_\_\_

אישור הועדה\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ תאריך\_\_\_\_\_\_\_\_\_\_\_

# מדריך למשתמש

## מבוא

מטרת הפרויקט היא לממש את משחק הלוח "ארבע בשורה" בתוכנת מחשב שבה יהיה ניתן לשחק שחקן מול שחקן ברשת. בפרויקט מימשתי את כל הכללים של המשחק בהתאם לחוקים, המשתמש יכול ללמוד על חוקי המשחק בתוך המשחק ולהיעזר בהם.

## דרישות למחשב והסביבה

בכדי להריץ את התוכנה, כל מה שנדרש זה ש- java תהיה מותקנת על המחשב. התוכנה יכולה לרוץ על גביי מגוון מערכות הפעלה (Windows, Mac, Linux) מכיוון ששפת java פותחה בצורה כזאת שתעבוד במגוון פלטפורמות. התוכנה פותחה בסביבת העבודה NetBeans 8.1 ועל מנת לפתוח את הפרויקט ולערוך את הקוד צריכים שתוכנה זו תהיה מותקנת על המחשב, בנוסף דרוש שללקוח ולשרת יהיה חיבור תקין לרשת, קח שיהיה להם אייפי ופורט תקינים.

## הוראות הרצת הפרויקט

בכדי להריץ את התוכנה כל מה שנדרש זה לפתוח את קובץ ה - jar של התוכנה,

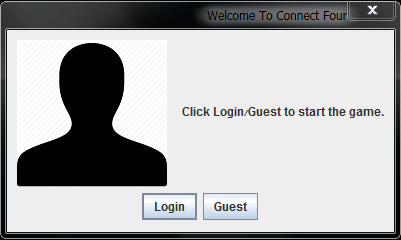
אין צורך בהתקנה בכדי להריץ את התוכנה.

בשביל להריץ את המשחק יש להיכנס לתיקהJARs , לאחר מכן יש להיכנס לתיקה Client,

ושם יש להריץ את הקובץ ClientConnect4 בשביל להריץ את המשחק.

## שימוש בתוכנה לקוח:

מיד בעת פתיחת קובץ התוכנית, מוצג אפשרות ההתחברות למשחק:



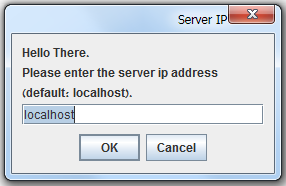
לשחקן יש **שתי** אפשרויות:

השחקן יכול להיכנס בתור אורח למשחק.

או שהוא יכול להיכנס למשחק בתור משתמש שרשום במערכת המשתמשים של המשחק.

***שחקן רשום:***

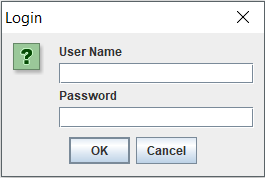
אם בחר בשחקן **רשום**, יופיע לו המסך הבא:



במסך הנ"ל, השחקן יצטרך להכניס את הכתובת של השרת, כולל פורט חוקי של השרת.

לדוגמא: 127.0.0.1:1234 או 1234:localhost.

אחרי שהשחקן יכניס כתובת חוקית שעובדת, זה החלון שיפתח לו:



אחרי שהשחקן יכניס שם משתמש וסיסמה חוקיים,

השחקן יתחבר למשחק.

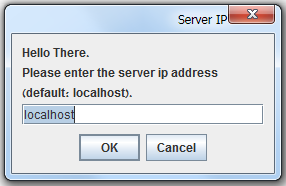
שמות משתמשים שרשומים במשחק הן:

שם: Daniel סיסמה: 1111

שם: Shirel סיסמה: 3333

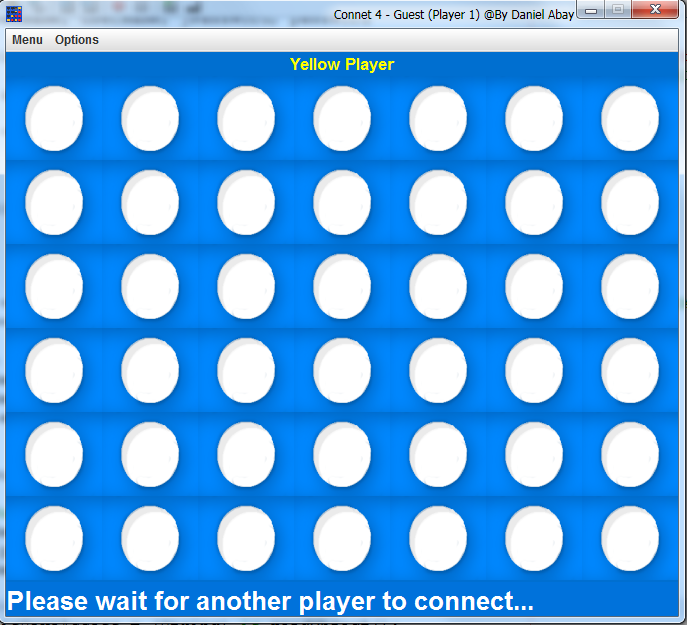
***אורח:***

ואם יבחר **באורח** יופיע לו אותו מסך כמו של הלקוח הרשום:



אחרי שהשחקן יכניס כתובת חוקית שעובדת, השחקן יתחבר למשחק.

וזה החלון שיפתח לו:

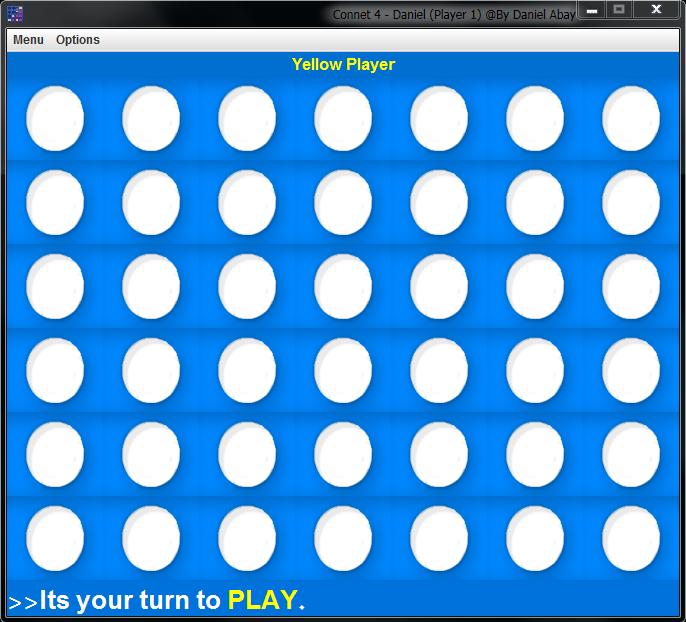


במקרה שהשחקן התחבר ראשון למשחק (Player 1), יצטרך להמתין לשחקן השני להתחבר (Player 2),

והמשחק יתחיל.

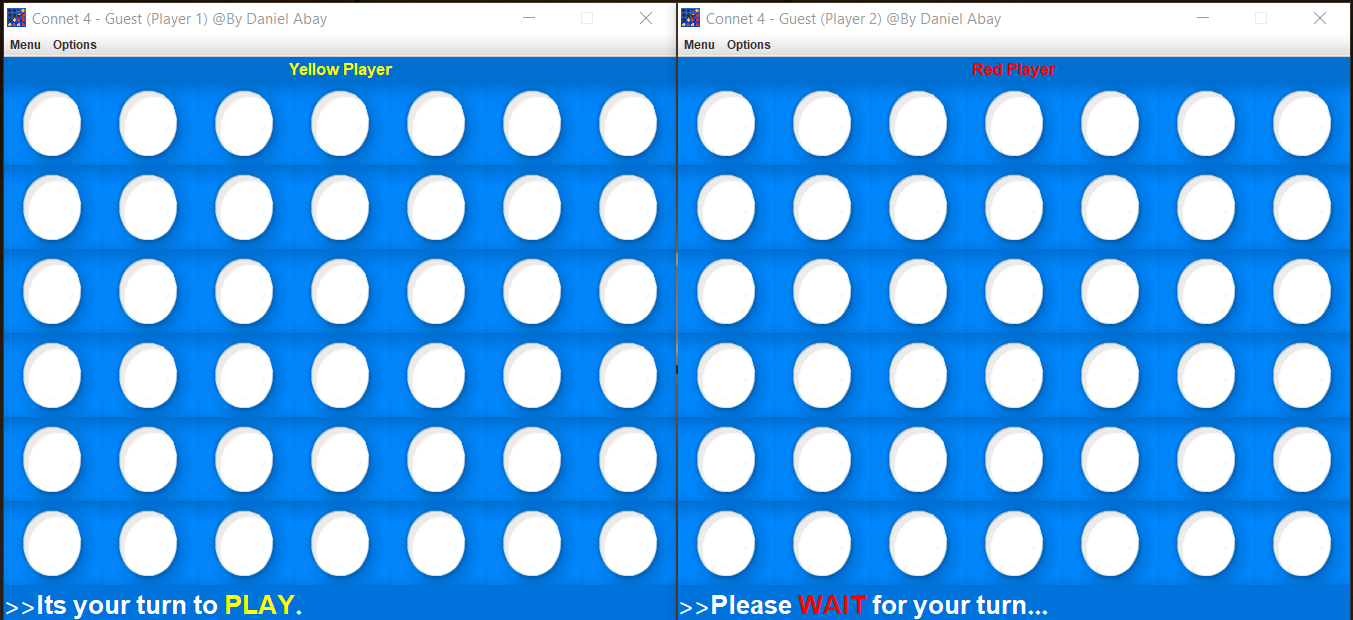
לאחר שהשחקן השני התחבר למשחק באותה דרך כפי שהסברנו מקודם,

המשחק יתחיל והמסך יראה ככה:

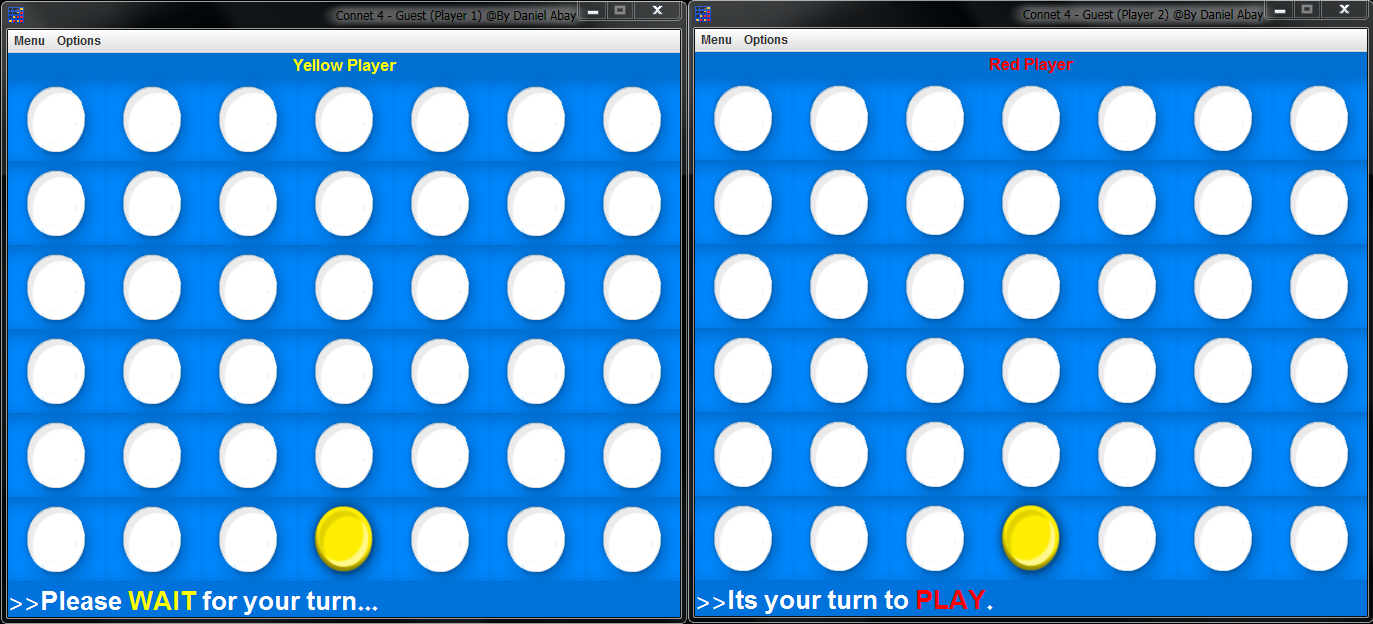


אנו יכולים לראות כי עכשיו זה תורו של השחקן הראשון, ניתן לראות ששורת המצב מודיעה על כך.

ולכן השחקן הראשון ישחק את תורו, והשחקן השני ימתין.

G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

((Player 2 (Player 1)

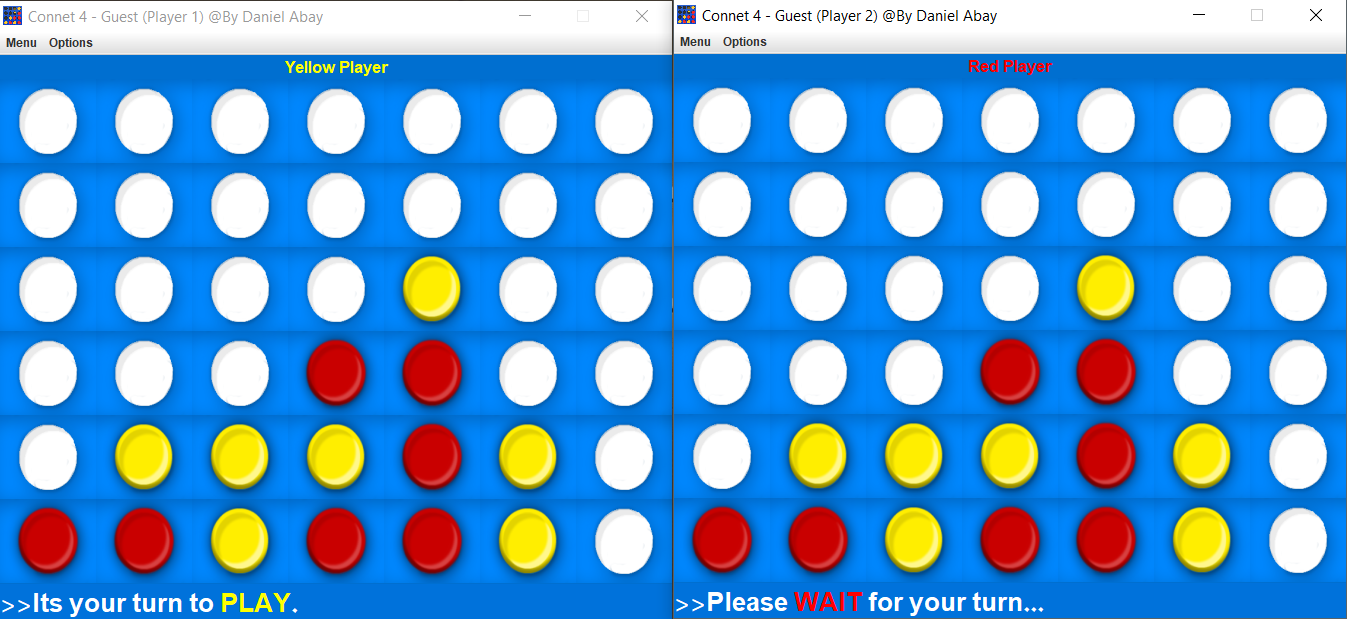
לאחר לחיצה על אחד הכפתורים שבלוח, נוכל לראות על הלוח את המיקום שהשחקן לחץ עליו, כמו באיור הבא:

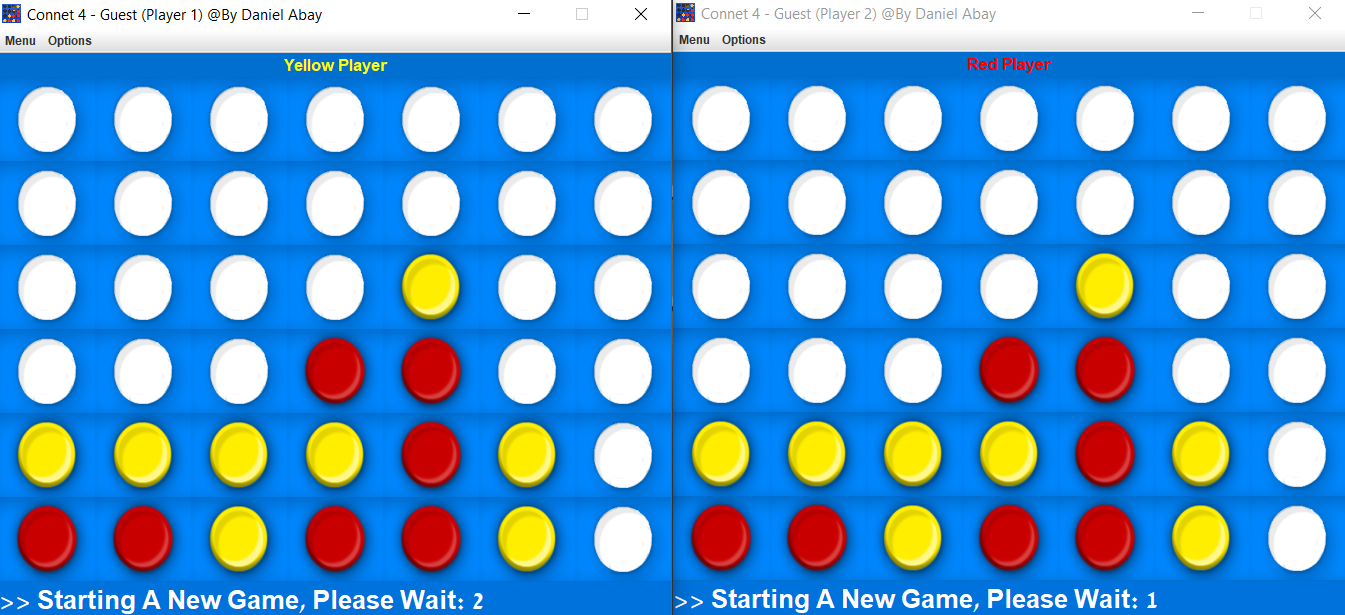
((Player 2 (Player 1)

כעט אנו רואים שזה תורו של השחקן השני ((Player 2, והשחקן הראשון (Player 1) ימתין לתורו.



אפשר לראות באיור מספר 1 כמה מהלכים קדימה של המשחק, ובאיור מספר 2 אפשר לראות שיש לנו ניצחון:





Player 1

Player 2

באיור הבא אנו רואים שמשחק נגמר, ואיזה שחקן ניצח:



ומשחק חדש יתחיל לאחר 3 שניות.G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

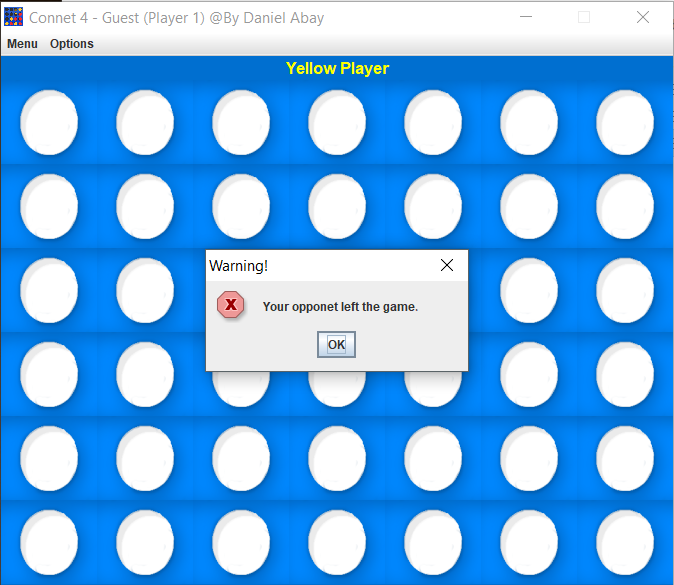
במקרה שאחד השחקנים רוצה לצאת מהמשחק, יוכל ללחוץ על כפתור ה"איקס" בחלון המשחק.

כמו באיור הבא:

G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

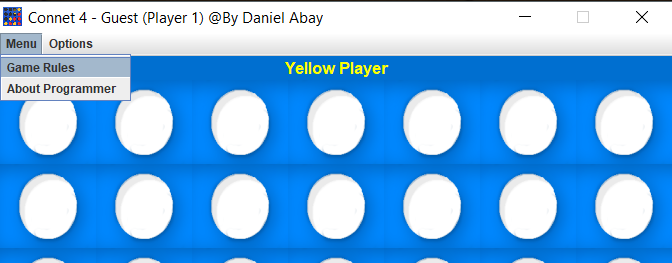
לאחר שאחד השחקנים יוצא מהמשחק אז ללקוח השני יוקפץ הודעה שהשחקן הראשון יצא, ולכן נגמר המשחק.

כמו באיור הבא:

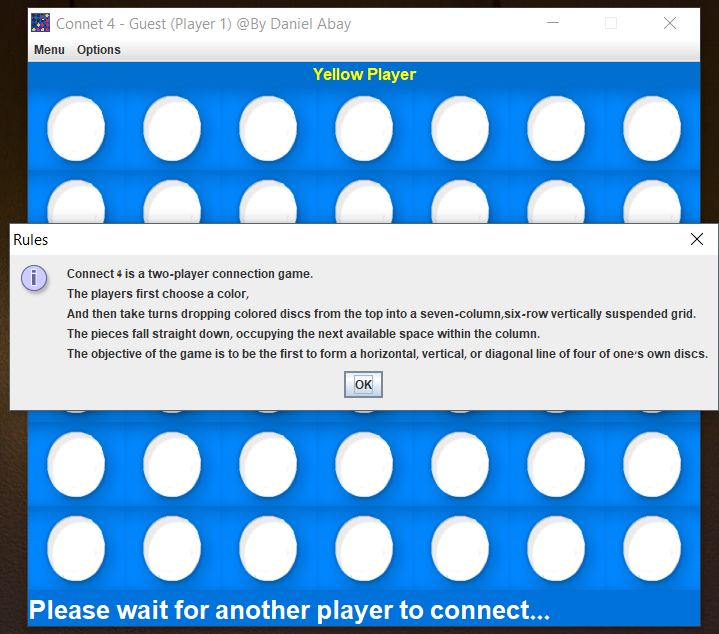
G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

המשחק יצא לאחר כמה שניות.

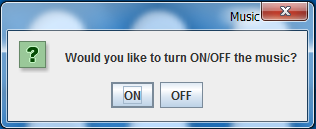
בכדי להציג את הוראות המשחק נלחץ על "Menu">"Game Rules":

G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

בכדי לצאת מהוראות המשחק צריך ללחוץ "Ok".

G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

ניתן להדליק ולכבות את המוזיקה של המשחק על ידי בחירת - " Options ">"Play Music":

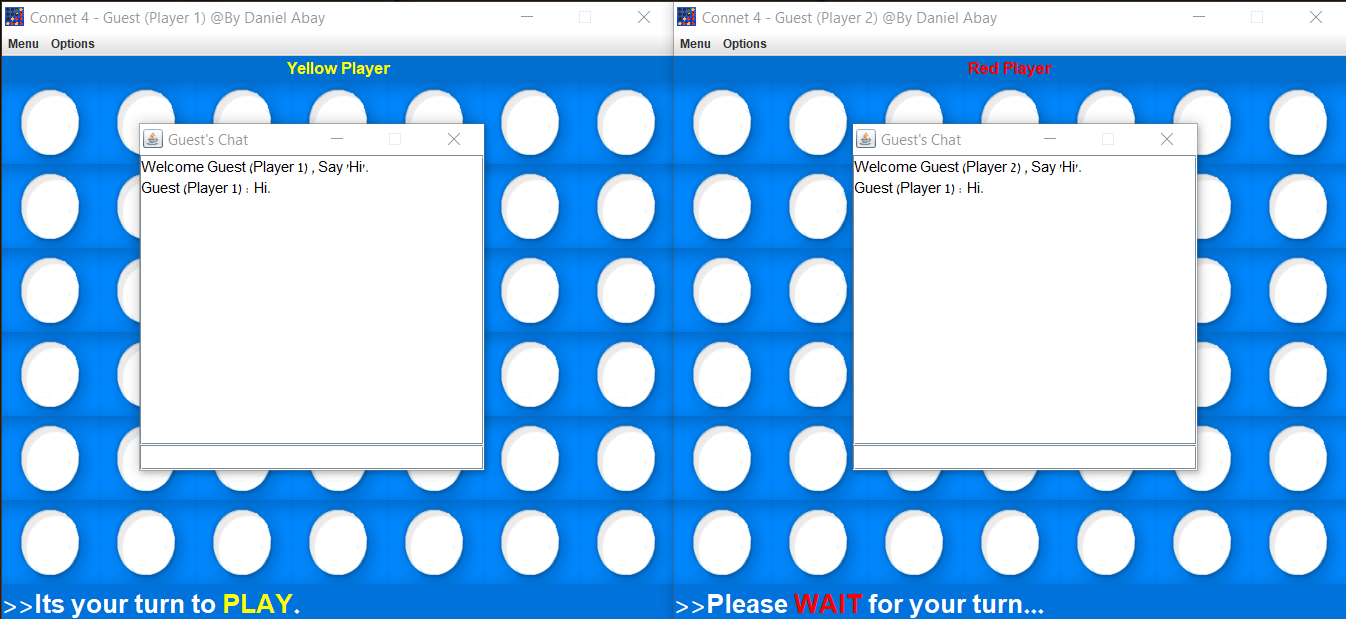
G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

ניתן גם לדבר במהלך המשחק עם השחקן השני על ידי בחירת" Options">"Chat" - :

(אי אפשר לפתוח את הצ'אט אם השחקן השני אינו מחובר).

G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png

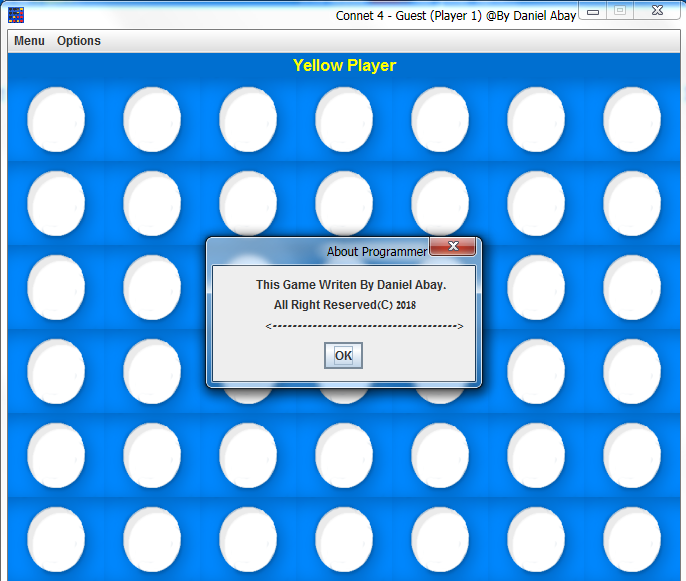
G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png



ניתן גם לראות את אודות מפתח המשחק וזכויות יוצרים על ידי בחירת" Menu">"About Programmer" - :



G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.png



# 

# G:\13th Grade\Java - Ilan Perets\תמונות לתיק פרויקט\mouse pointer.pngמדריך למתכנת

## כלי הפיתוח

המשחק נכתב כאמור בשפת Java התומכת במגוון פלטפורמות. הפרויקט פותחת בסביבת הפיתוח NetBeans 8.1. ניתן להוריד תוכנה זו באינטרנט.

מעבר לכתיבת הקוד בסביבת פיתוח זו, נדרשתי להשתמש במספר תוכנות להכנת המדיה של המשחק. השתמשתי בתוכנת PowerPoint על מנת ליצור את האייקונים של החיילים, את האייקון של החלון והאייקון של קיצור הדרך למשחק.

## מודל שרת לקוח

מודל שרת-לקוח (באנגלית: client-server model) היא ארכיטקטורת תוכנה לחישוב מבוזר, אשר מגדירה את היחס בין תוכנות משתפות פעולה. המודל מחלק את המשימות או עומס העבודה בין ספק השירות או המשאבים - השרת, לבין מבקש השירות - הלקוח. שרת-לקוח היא אחת מתצורות ההתקשרות הנפוצות ברשתות מחשבים. השרת הוא תוכנה פסיבית, המאזינה לרשת ומחכה לקבל בקשות. הלקוח לעומתו בדרך כלל מהווה את ממשק המשתמש - הוא מופעל על ידי המשתמש ופונה לשרת כאשר הוא זקוק למידע או שירותים ממנו.

בדרך כלל, תוכנות השרת והלקוח רצות על גבי מחשבים שונים והתקשורת ביניהן מתבצעת על גבי רשת מחשבים. עם זאת, תוכנות השרת והלקוח יכולות לפעול גם על גבי אותו מחשב. מכונת שרת הוא מחשב המריץ תוכנת שרת אחת או יותר, אשר חולקת את המשאבים שלה עם הלקוחות. הלקוח אינו חולק את המשאבים שלו, אלא מבקש תכנים ושירותים (ביצוע פונקציות) מהשרת. לפיכך, הלקוח הוא זה שיוזם את ההתקשרות עם השרת, אשר ממתין לבקשות נכנסות.

## פרוטוקול תקשורת IP/TCP:

פרוטוקול TCP הינו אחת מאבני הבניין החשובות והבסיסיות ביותר של רשת האינטרנט. מטרתו היא לאפשר העברת מידע אמינה ורציפה בין שתי ישויות רשת, מעל תשתית שאינה אמינה. למרות שמדובר בפרוטוקול מורכב, הרעיונות העומדים בבסיסו דווקא פשוטים יחסית, ולדעתי חשוב שאנחנו המפתחים, ובעצם כל מי שמתעסק עם רשתות מידע, נכיר אותם. בפוסט זה אציג מספר בעיות איתן מנסה להתמודד פרוטוקול TCP ואת הדרכים בהן הוא עושה זאת. אשתדל לעשות זאת בצורה פשוטה, בלי להיכנס לפרטים טכניים מדי.

## 

## שקע (תקשורת מחשבים):

בתקשורת נתונים, שקע (socket) הוא נקודת קצה עבור זרם נתונים בתקשורת בין תהליכים על גבי רשת מחשבים. כתובת שקע (socket address) היא שילוב של כתובת IP ומספר פורט, ניתן להקביל זאת לשיחת טלפון שבה כל קצה מזוהה על ידי מספר טלפון וקידומת מסוימת, זהו הServer Socket שנמצא בשרת.

ללקוח יש גם Socket שדרכו הוא מתחבר אל ה Server Socket, כך שנוצר שני חיבורים, יש את החיבור של הלקוח אל השרת, וגם הם מאזינים אחד לשני.

## התחברות למוסד נתונים:

תהליך ההתחברות למוסד הנתונים דורש כמה שלבים שהן:

* לאחר שלקוח מתחבר למשחק ובחור בתור משתמש רשום, המשחק יציג לו שאילתה שצריך למלות בה את השם משתמש שלו והסיסמה.
* לאחר מכן, אחרי שהלוקח רשם את שם המשתמש והסיסמה שלו, תוכנת הלקוח תשלח לתוכנת השרת את שם המשתמש וסיסמה, והיא בודקת במוסד הנתונים אם הנתונים שהוא רשם נכונים דרך טבלה של משתמשים רשומים.
* תהליך הבדיקה הוא כזה, השרת משתמש בשפת SQL בשביל לגשת לתוך הטבלה של מוסד הנתונים כדי לבדוק אם השם משתמש וסיסמה שהלקוח הקיש הם נכונים, הפעולה בודקת את הנתונים ואם הפרטים תואמים היא מחזירה אישור, ואם הם לא אז היא מחזירה שהנתונים שגויים.
* במקרה שהפרטים שהלקוח רשם הם נכונים, אז הפעולה תחזיר אישור השרת יוסיף את הלקוח לרשימת הלקוחות שמחוברים למשחק,

במקרה שהפרטים שהלקוח רשם הם אינם נכונים, הפעולה תחזיר לנו הודעה שהנתונים לא תואמים, והלקוח יצטרך לרשום את הפרטים שלו שוב פעם. כל עוד הפרטים של הלקוח הם לא תאומים לאחד המשתמשים שרשומים בטבלה, הוא לא יוכל להתחבר למשחק ולהתחיל לשחק.

## 

## הרשאות גישה סיסמה שם משתמש מספר זהות

## פרוטוקול השיח:

צד השרת וצד הלקוח מתקשרים באופן שבו השרת מחכה לבקשות מהלקוחות ולא יוזם אותן בעצמו, ועל צד הלקוח לשלוח בקשות לשרת, והשרת יבצע אותן לאחר קבלת הבקשות. השרת מוכן לקבל את כל סוגי הבקשות בכל זמן

(ביצוע מהלך/שליחת הודעת צא"ט ועוד).

* צד השרת אמור לרוץ כל הזמן (בניגוד לצד הלקוח שלא חייב לרוץ כך), ומחכה ללקוחות שיתחברו. במידה והשרת הגיע למספר המקסימלי של לקוחות שהוא יכול לשרת בזמן נתון (שהוא שתי שחקנים), השרת יסרב לחיבור של לקוחות נוספים ורק לאחר שאחד מהם יתנתק הוא יתפנה לקבלת לקוחות חדשים.
* לאחר התחברות של לקוח מסוים לשרת, הוא יוסף לרשימת הלקוחות המחוברים, לאחר שלקוח התחבר על הלקוח לבחור אם הוא רוצה להתחבר בתור משתמש רשום או בתור אורח, לאורח יש לרשום את כתובת השרת (הIP), ואז הוא יתחבר למשחק, במשתמש רשום הוא יהיה חייב לרשום גם את כתובת השרת (IP), והוא יהיה חייב לשלוח את פרטי המשתמש שלו לשרת (שם משתמש וסיסמא), כדי שהוא יוכל להתחיל לשחק. במידה ופרטי המשתמש נכונים, השחקן יכול להתחיל לשחק (במקרה שיש עוד לקוח שמחובר). אחרת, תישלח הודעה ללקוח על כך שהוזנו פרטי משתמש שגויים, ויצטרך להקליד אותם שוב.
* לאחר שההזדהות של המשתמש הצליחה, הלקוח יוכנס למשחק חדש אוטומטית.

ושרת יטפל במשחק באופן הבא:

המשתמש משחק נגד שחקן אנושי אקראי, אם קיים שחקן פנוי המחכה ליריב, השרת יצמיד את שניהם

אחד לשני, יאתחל משחק חדש ביניהם, ישלח להם את המצב ההתחלתי של המשחק ולכל שחקן ישלח הודעה לגבי

איזה שחקן צריך לשחק עכשיו. הכפתורים בלוח המשחק אצל שחקן זה יהפכו להיות פעילים, והוא יוכל לבחור את

המהלך שלו, בזמן הכפתורים אצל השחקן השני ימשיכו להיות בלתי פעילים.

מידה ולא קיים שחקן פנוי למשחק, השחקן יצטרך להמתין עד שעוד לקוח יתחבר.

* לאחר ששחקן מסוים בחר מהלך שהוא רוצה לבצע, המהלך יישלח לשרת. השרת יבדוק האם המהלך חוקי. במידה המהלך איננו חוקי, תישלח על כך הודעה ללקוח, והוא יתבקש לבחור מהלך אחר.

אם המהלך הינו חוקי, הכפתורים בלוח המשחק של אותו שחקן יהפכו לבלתי פעילים.

לאחר מכן, יישלח לשניהם המצב החדש של המשחק לאחר ביצוע המהלך, ואם המשחק לא הסתיים בעקבות מהלך זה, תישלח הודעה ליריב שזה תורו לשחק כעת, הכפתורים בלוח המשחק שלו יהפכו להיות פעילים והוא יידרש לבצע את המהלך שלו.

* במידה ולאחר ביצוע מהלך מסוים המשחק הגיע לסיום, במקום לשלוח הודעה לשחקן השני שזה תורו עכשיו, תישלח הודעה לגבי סיום המשחק – מי המנצח, או האם המשחק נגמר בשוויון.

בשלב זה ייגמר המשחק בין זוג שחקנים, והשרת יתחיל משחק חדש לאחר 3 שניות.

* בנוסף לכך, לאחר התחלת משחק חדש בין שני שחקנים יתחיל בניהם צא'ט. לאחר שליחת הודעת צא'ט לשרת, היא תישלח בחזרה לשחקן ששלח אותה ולשחקן היריב, והיא תוצג להם על גבי המסך.
* אם לקוח מסוים רוצה לצאת מן התוכנית, תישלח בקשת exit לשרת. בעקבות כך, אם הלקוח היה באמצע משחק, הוא יוצא מן המשחק ותישלח הודעה ליריב האנושי לגבי כך שהיריב שלו התנתק. לאחר מכן הלקוח יימחק מרשימת הלקוחות המחוברים, תישלח הודעת exit בחזרה ללקוח, החיבור בניהם ייסגר סופית ותוכנת צד הלקוח תחדל מלרוץ.

שלבים אלו יבוצעו גם כן במקרה שלקוח התנתק מהשרת בפתאומיות, נגיד במקרה המחשב שלו פתאום קרס.

**שלבי התחברות:**

-השרת ממתין ללקוחות..........

............לקוח ראשון מתחבר לשרת-

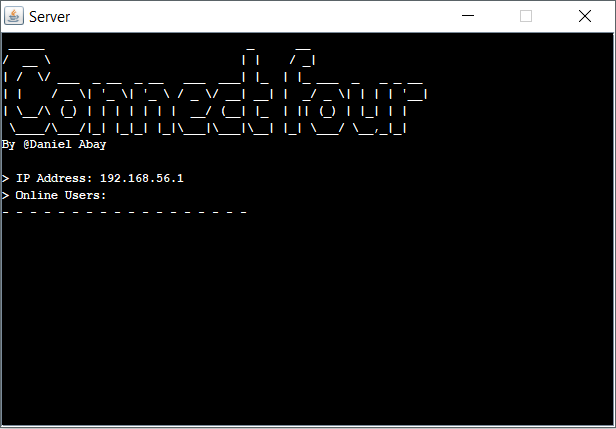
-השרת מראה שהתחבר לקוח אחד וממתין ללקוח השני..........

............לקוח שני מתחבר לשרת-

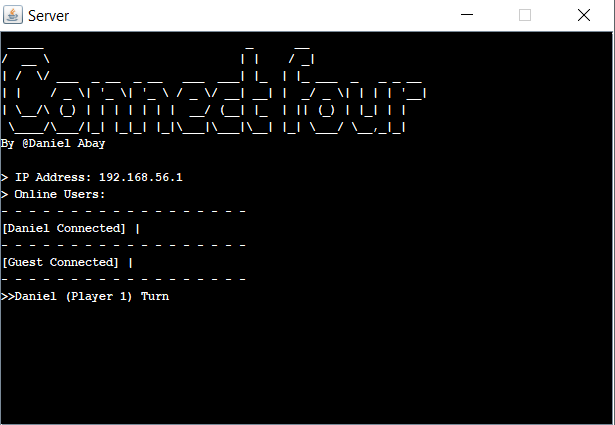
-השרת מראה ששני לקוחות מחוברים והמשחק התחיל..........

## שימוש בתוכנת השרת:

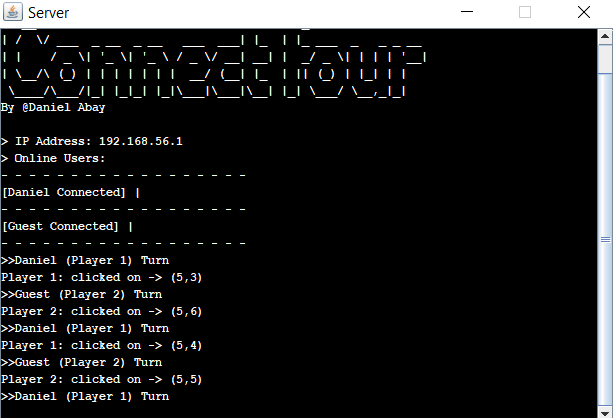
אחרי שהפעלנו את תוכנת השרת,הוא יבקש להכניס פורט, אפשר לראות את כתובת האייפי של השרת, ואת השחקנים המחוברים, באיור הבא:



כרגע אף שחקן לא מחובר, אבל ששני שחקנים יתחברו אל השרת, המשחק יתחיל, כמו באיור הבא:

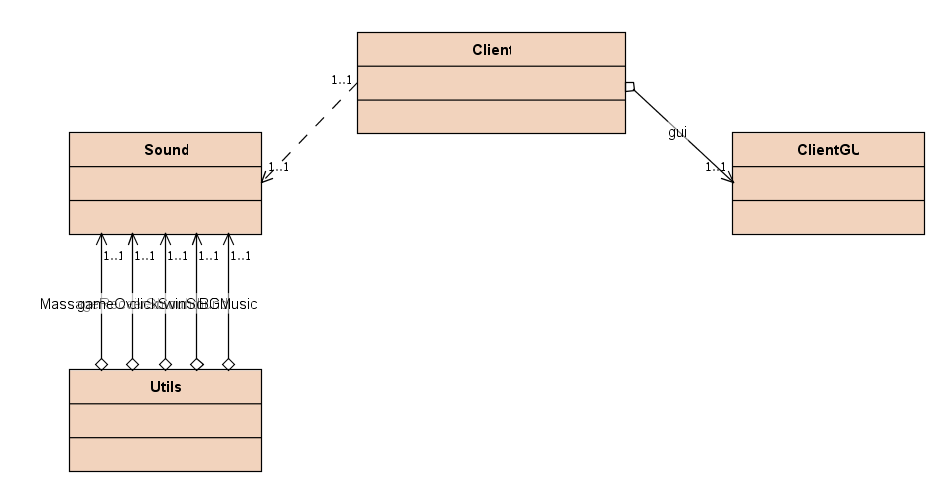


נוכל לראות את כל התנועות של שני השחקנים, ואת תור השחקן הנוכחי באיור הבא:

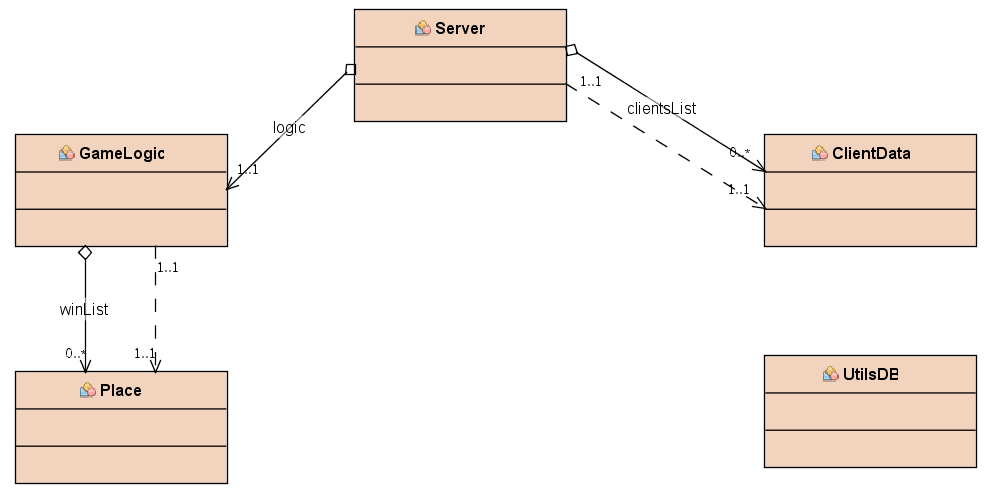


## תרשים UML:

תרשים של **לקוח**:



תרשים של **שרת**:



## רשימת המחלקות

רשימת מחלקות :**Client**

### המחלקה Client

זו המחלקה שאחראית להרצת התוכנית של הלוקח, במחלקה זו אנו שולחים כל מיני פקודות לשרת, והשרת מחזיר לנו את מה שרצינו בשביל לשחק במשחק או כל דבר אחר שקשור לתוכנה (למשל צ'אט), כל המשחק מתנהל אצל הלקוחות ונגמר אצלהם.

### המחלקה ClientGui

זוהי המחלקה של הממשק הגרפי של המשחק. היא יוצרת את חלון המשחק, את הפאנלים השונים בחלון, והיא מבצעת פקודות שונות המבצעות שינויים בממשק.

### המחלקה Utils

זוהי מחלקת שירות הטוענת את קבצי המדיה השונים של הפרויקט וכוללת משתנים וקבועים שונים הנדרשים להתנהלות המשחק, כדוגמת גודל הלוח, הייצוג המחרוזתי של השחקנים השונים במשחק, מהו גודל כפתור ועוד.

### המחלקה Sound

הפעולה טוענת קובץ סאונד מכתובת שמתקבלת כפרמטר. היא אחראית על הפעלת המוזיקה, הפסקתה, ועל ניגונה בלופ.

רשימת מחלקות :**Server**

### המחלקה Server

מחלקה שמנהלת את כל המשחק, כל לקוח שמבקש בקשה מהשרת, השרת בודק את הבקשה לו ומחליט איזה תשובה להחזיר ללקוח, המחלקה אחראית על כל ההתחברות של שחקנים ותחילת המשחק, לכל לקוח יש יחס אישי וכל תנועה שקוראת במשחק עוברת דרך השרת והוא מעדכן את שני הלקוחות בתנועה שקרה.

### המחלקה GameLogic

זוהי מחלקת לוגיקת המשחק, היא בודקת את המקום הנוכחי של השחקן, אחראית על ביצוע מהלכים על לוח המשחק, יש מערכים של ארבע מיקומים בשביל לבדוק אם יש ניצחון.

### המחלקה UtilsDB

המחלקה אחראית על מוסד הנתונים, כל הגישה למשתמשים רשומים או לאורחים בשביל להתחבר למשחק מתבצעת במחלקה זו.

המחלקה **ClientData**

מחלקה זו שומרת את כל הנתונים של לקוח שהתחבר למשחק, למשל כמו השם שלו, מספר השחקן שמחובר, והשקע שמנהל את הקלט פלט עם השרת.

### המחלקה Place

המחלקה מייצגת מקום מסוים במרחב המשחק. היא כוללת באיזה פאנל החייל נמצא – בלוח המשחק/אצל המחשב/אצל השחקן האנושי/ אצל השחקן האנושי השני, והיא כוללת את השורה והעמודה שבה הוא נמצא.

## 

## **בעיות שהתמודדתי איתן:**

הבעיות שאני התמודדתי איתן במהלך הכנת משחק הן:

* אחת הבעיות העיקריות שלי הייתה ליצור חיבור בין שני השחקנים, ולעביר בניהם מידע מסונכרן, למשל התחלתי עם לחיצת כפתור פשוטה של שחקן אחד, ולאחר שהוא לחץ אז השחקן השני יוכל גם ללחוץ. אולי זה נשמע פשוט אך בהתחלה לקח לי הרבה זמן עד שהצלחתי להשלים את המשימה.
* בעיה נוספת שנתקלתי בה, היא שבהתחלה ניסיתי לכתוב קוד אחד שכל הלקוחות ירוצו עליו, אבל היו לי בעיות במהלך הפיתוח ולא הצלחתי לעקוב אחרי מה שכל לקוח עושה, לכן הכנתי לכל לקוח קוד משלו. ככה יכלתי לעקוב יותר בקלות על מה שקורה לכל לקוח.

## **שיפורים עתידיים:**

שיפורים עתידיים שאפשר להוסיף למשחק הן:

* אפשרות לזיהוי כל השרתים שרצים במשחק, לדוגמה אם מריצים כמה שרתים מכמה מחשבים שונים, שהלקוח יוכל לבחור לאיזה שרת הוא רוצה להתחבר, וגם שיכול לראות כמה שחקנים מחוברים לכל שרת.
* אפשרות להריץ שרת אין סופי, שגם אחרי ששני לקוחות מתחברים הוא מתפנה לקבל עוד שני לקחות חדשים, וכך האלה. ככה שלא יהיה מצב שהוא יוכל לקבל רק שני לקוחות, ושאר הלקוחות האחרים יצטרכו לחכות עד שהם יתנתקו.

# סיכום

העבודה על המשחק ארבע בשורה הייתה מהנה ומעניינת, זו הרגשה אחרי שיצרתי תוכנה מתפקדת שהתחלתי מאפס. למדתי כיצד להשתמש בספריות הגרפיקה של Java בבקיאות, ומאוד נהניתי מהכרת המשחק ארבע בשורה. התגברתי על בעיות שצצו בעת כתיבת הפרויקט, ויצרתי תוכנה שאפשר ליהנות מהשימוש בה.

# תודות

רוב תודות למורה אילן פרץ שתמך בי ועזר לי לאחר כל בקשה במהלך העבודה על הפרויקט.

תודה רבה ליוסי לביא מנהל מכללת ההנדסאים בקרית נוער שנתן לנו את האפשרות ללמוד הנדסת תוכנה בכיתה י"ד.

תודה רבה לטכנאי מרג'י שנתן לנו להשתמש במעבדת המחשבים לאחר שעות הלימודים.

# ביבליוגרפיה

אתר סביבת הפיתוח NetBeans: https://netbeans.org

פורום קהילת התכנות StackOverflow: [http://stackoverflow.com](%20http://stackoverflow.com/)

# נספחים

**אלגוריתם ללקוח הראשון בשרת:**

הפעולה מטפלת בלקוח הראשון שהתחבר לשרת בפעולה מתהל כל הקלט והפלט בין השחקן

לבין השרת השחקן שולח מיקום שהוא לחץ בלוח והפעולה הנל בודקת אם המיקום הזה

פנוי או חוקי ומעדכן את הלוח ובודקת אם יש ניצחון לשחק הנוכחי אם לא אז

מחליפים לתור של השחקן השני.

private void handleClient1(final ClientData client) throws Exception

{

Thread t;

t = new Thread(new Runnable()

{

@Override

public void run()

{

try

{

client.getOutputStream().writeObject(client.getName());

client.getOutputStream().writeObject("Welcome, Please wait for another player to connect...");

updateServerGUI();

sem.acquire(); //Blocking HandleClient1 to countine. (Watinig for Player 2 to connect).

broadcastMsg("Two Connected"); // מודיע ששני שחקנים מחוברים ללקוחות

areaUsers.append(">>" + client.getUserRealName() + " (" + client.getName() + ")" + " Turn\n"); while (gameStatus != GAME\_OVER)

{

System.out.println("");

if (playerTurn == 1)

{

client.getOutputStream().writeObject("Play");

client.getOutputStream().flush();

Thread.sleep(200);

}

else

{

client.getOutputStream().writeObject("Please wait for your turn.");

}

//=====================<<<>>>========================//

Object msg = client.getInputStream().readObject(); // ממתין לקלט מהלקוח

if (msg instanceof String) // אם זה טקסט אז מתייחס לצאט

{

broadcastMsg(client.getUserRealName() + " (" + client.getUserName() + ") " + ": " + msg);

}

else if (msg instanceof Integer) // אם זה מספר אז הוא מתייחס ללוח המשחק

{

int row = (Integer)msg;

int col = (Integer)client.getInputStream().readObject();

if (logic.isColNotFull(col, board) && logic.isLegalPlace(row, col))

{

row = logic.getEmptyRow(col, board);

updateBoard(row, col, client.getName());

for (int i = 0; i < clientsList.size(); i++) // שולח את ההדעות לשני השחקנים(שידור)

{

clientsList.get(i).getOutputStream().writeObject("Place is Ok");

clientsList.get(i).getOutputStream().writeObject(row);

Thread.sleep(10);

clientsList.get(i).getOutputStream().writeObject(col);

clientsList.get(i).getOutputStream().writeObject(client.getName());

clientsList.get(i).getOutputStream().flush();

}

areaUsers.append(client.getName() + ": clicked on -> ("+row+","+col+ ")\n");

if (isPlayerWin(1) == true)

{

broadcastMsg(">> Player Yellow Won! <<");

updateServerGUI();

areaUsers.append("\n>> [GAME OVER] <<\n");

areaUsers.append(">> Player Yellow Won! <<\n");

clearBoard();

clientsList.get(1).getOutputStream().writeObject("Please wait for your turn.");

}

else if (logic.isTie(board))

{

broadcastMsg(">> It's a Tie! <<");

updateServerGUI();

areaUsers.append(">> It's a Tie! <<\n");

clearBoard();

}

else

{

clientsList.get(1).getOutputStream().writeObject("Play");

playerTurn = 2;

areaUsers.append(">>" + clientsList.get(1).getUserRealName()

+ "(" + clientsList.get(1).getName() + ")" + " Turn\n");

areaUsers.setCaretPosition(areaUsers.getDocument().getLength());

}

}

else

{

client.getOutputStream().writeObject("Ilegal Place, Try Again!");

}

Thread.sleep(200);

}

}

client.close(); // מנתק את החיבור עם אותו לקוח

}

catch (Exception ex)

{

try

{

if (clientsList.size() > 1)

{

clientsList.get(1).getOutputStream().writeObject("Disconnected");

}

client.close(); // מנתק את החיבור עם אותו לקוח

clientsList.removeAll(clientsList); // מסיר את כל השחקנים שברישמת שחקנים

updateServerGUI();

areaUsers.append("Wating for players..");

serverSocket = new ServerSocket(2222); // יצירת השקע של השרת

}

catch (Exception ex1)

{

ex1.printStackTrace();

}

}

}

}

);

t.start();

}

**אלגוריתם ללקוח השני בשרת:**

\* הפעולה מטפלת בלקוח שני שהתחבר לשרת בפעולה מתהל כל הקלט והפלט בין השחקן

\* לבין השרת השחקן שולח מיקום שהוא לחץ בלוח והפעולה הנל בודקת אם המיקום הזה

\* פנוי או חוקי ומעדכן את הלוח ובודקת אם יש ניצחון לשחק הנוכחי אם לא אז

\* מחליפים לתור של השחקן השני.

private void handleClient2(final ClientData client) throws Exception

{

Thread t = new Thread(new Runnable()

{

@Override

public void run()

{

try

{

client.getOutputStream().writeObject(client.getName()); // Sending the client user name.

client.getOutputStream().writeObject("Welcome, Please wait for another player to connect...");

updateServerGUI();

sem.release(); //relesing the block form HandleClient1.

Thread.sleep(500); //Wating for other brodcasts to finsh (don't want to interrupt them).

while (gameStatus != GAME\_OVER)

{

if (playerTurn == 2)

{

client.getOutputStream().writeObject("Play");

client.getOutputStream().flush();

Thread.sleep(200);

}

else

{

client.getOutputStream().writeObject("Please wait for your turn.");

}

//=====================<<<>>>========================//

Object msg = client.getInputStream().readObject(); // ממתין לקלט מהלקוח

if (msg instanceof String) // אם זה טקסט אז מתייחס לצאט

{

broadcastMsg(client.getUserRealName() + "("+client.getUserName()+")"+":"+ msg);

}

else if (msg instanceof Integer) // אם זה מספר אז הוא מתייחס ללוח המשחק

{

int row = (Integer)msg;

int col = (Integer)client.getInputStream().readObject();

if (logic.isColNotFull(col, board) && logic.isLegalPlace(row, col))

{

row = logic.getEmptyRow(col, board);

updateBoard(row, col, client.getName());

for (int i = 0; i < clientsList.size(); i++) // שולח את ההדעות לשני השחקנים(שידור)

{

clientsList.get(i).getOutputStream().writeObject("Place is Ok");

clientsList.get(i).getOutputStream().writeObject(row);

Thread.sleep(10);

clientsList.get(i).getOutputStream().writeObject(col);

clientsList.get(i).getOutputStream().writeObject(client.getName());

clientsList.get(i).getOutputStream().flush();

}

areaUsers.append(client.getName() + ": clicked on -> ("+row+","+col+ ")\n");

if (isPlayerWin(2) == true)

{

broadcastMsg(">> Player Red Won! <<");

updateServerGUI();

areaUsers.append("\>> [GAME OVER] <<\n");

areaUsers.append(">> Player Red Won! <<\n");

clearBoard();

clientsList.get(0).getOutputStream().writeObject("Please wait for your turn.");

}

else if (logic.isTie(board))

{

broadcastMsg(">> It's a Tie! <<");

updateServerGUI();

areaUsers.append(">> It's a Tie! <<\n");

clearBoard();

}

else

{

clientsList.get(0).getOutputStream().writeObject("Play");

playerTurn = 1;

areaUsers.append(">>"+clientsList.get(0).getUserRealName()+"("+clientsList.get(0).getName()+")"+"Turn\n");

areaUsers.setCaretPosition(areaUsers.getDocument().getLength());

}

}

else

{

client.getOutputStream().writeObject("Ilegal Place, Try Again!");

}

Thread.sleep(200);

}

}

client.close(); // מנתק את החיבור עם אותו לקוח

}

catch (Exception ex)

{

try

{

if (clientsList.size() > 1)

{

clientsList.get(0).getOutputStream().writeObject("Disconnected");

}

client.close(); // מנתק את החיבור עם אותו לקוח

clientsList.removeAll(clientsList); // מסיר את כל השחקנים שברישמת שחקנים

updateServerGUI();

areaUsers.append("Wating for players..");

serverSocket = new ServerSocket(2222); // יצירת השקע של השרת

}

catch (Exception ex1)

{

ex1.printStackTrace();

}

}

}

}

);

t.start();

}

**אלגוריתם לקוח שמאזין לשרת:**

פעולת האזנה הסרבר שולח הודעה ללקוח והלקוח בודק מה הסרבר רצה ממנו על פי

ההודעה שהוא שלח לו.

private void listenerFromServer() throws Exception

{

Object msgFromServer;

try

{

while (true)

{

try

{

msgFromServer = is.readObject(); // מחכה לסרבר שישלח לו הודעה מה לעשות

if (msgFromServer instanceof String)

{

if (((String)msgFromServer).contains("Please wait"))

{

if (clientID.equals("Player 1") && msgFromServer.equals("Please wait for your turn."))

{

gui.setLabelMsgText("<html>>>Please <font color='yellow'>WAIT</font> for your turn...</html>");

}

else if (clientID.equals("Player 2") && msgFromServer.equals("Please wait for your turn."))

{

gui.setLabelMsgText("<html>>>Please <font color='red'>WAIT</font> for your turn...</html>");

}

if (msgFromServer.equals("Welcome, Please wait for another player to connect..."))

{

gui.setLabelMsgText("<html>Please wait for another player to connect...</html>");

}

gui.enableAllButtons(false);

}

else if (((String)msgFromServer).equals("Play")) // במקרה שהסרבר שולח הודעה של שחק

{

if (clientID.equals("Player 1")) // אם השחקן הוא שחקן אחד אז תרשום בטקסט שתורו של שחקן צהוב

{

gui.setLabelMsgText("<html>>>Its your turn to <font color='yellow'>PLAY</font>.</html>");

}

else if (clientID.equals("Player 2"))

{

gui.setLabelMsgText("<html>>>Its your turn to <font color='red'>PLAY</font>.</html>");

}

gui.enableAllButtons(true);

}

else if (((String)msgFromServer).equals("Place is Ok"))

{

int row = (Integer)is.readObject(); // מקבל את המיקום בשורה בלוח

int col = (Integer)is.readObject(); // מקבל את המיקום בעמודה בלוח

String player = (String)is.readObject(); // שולח את השחקן שעשה את המהלך

if (player.equals("Player 1")) // אם השחקן שעשה את המהלך הוא השחקן אחד אז שים את הצבע צהוב

{

gui.setButtonOn(row, col, Utils.yellowIcon);

}

gui.enableAllButtons(false);

if (player.equals("Player 2"))

{

gui.setButtonOn(row, col, Utils.redIcon);

}

gui.enableAllButtons(false);

}

else if (((String)msgFromServer).equals(">> Player Yellow Won! <<"))

{

gui.setLabelMsgText("<html>>> Player <font color='yellow'>Yellow</font> Won! <<</html>");

playSound(Utils.winSound);

newGame();

}

else if (((String)msgFromServer).equals(">> Player Red Won! <<"))

{

// ששחקן האדום ניצח ומתכונן לרצה של משחק חדש

gui.setLabelMsgText("<html>>> Player <font color='red'>Red</font> Won! <<</html>");

playSound(Utils.winSound);

newGame();

}

else if (((String)msgFromServer).equals(">> It's a Tie! <<")) // במקרה שיש תיקו במשחק

{

gui.setLabelMsgText(msgFromServer.toString());

newGame(); // מתחיל משחק חדש

}

else if (((String)msgFromServer).equals("Disconnected")) // במקרה שאחד השחקנים מתנתק באמצע המשחק

{

gui.enableAllButtons(false); // מבטלת את כל הכפתורים

JOptionPane.showMessageDialog(gui.getWin(), "Your opponet left the game.", "Warning!", JOptionPane.ERROR\_MESSAGE); // מודיע שהשחקן עזב את המשחק

for (int i = 5; i > -1; i--)

{

// מריץ לולאה של חמש שניות עד שהמשחק יסתיים

// ומודיע לשחקן שהמשחק יוצא בעוד חמש שניות

gui.setLabelMsgText(">>Leaving the game in " + i + "...");

Thread.sleep(1000);

}

clientSocket.close(); // סוגר את החיבור לשרת

System.exit(0); //יוצא מהמשחק

}

else if (((String)msgFromServer).equals("Two Connected")) // מודיע ללוקח ששני השחקנים מחוברים

{

gui.getWin().setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

canOpenChat = true; // פותח אפשרות לפתוח את הצאט

}

else

if (msgFromServer.equals("Ilegal Place, Try Again!")) // מודיע שמיקום שהלוקח שם לא טוב

{

gui.setLabelMsgText("<html> <font color='red'>Ilegal Place, Try Again!</font> </html>");

Thread.sleep(1000);

}

else

{

area.append(msgFromServer + "\n");

area.setCaretPosition(area.getDocument().getLength());

chatWin.setVisible(true);

playSound(Utils.MassageReceived);

}

}

}

catch (Exception e)

{

JOptionPane.showMessageDialog(gui.getWin(), "Warning!\nThe Server is temporary down.", "Warning!", JOptionPane.WARNING\_MESSAGE);

System.exit(0);

}

}

}

catch (Exception e)

{

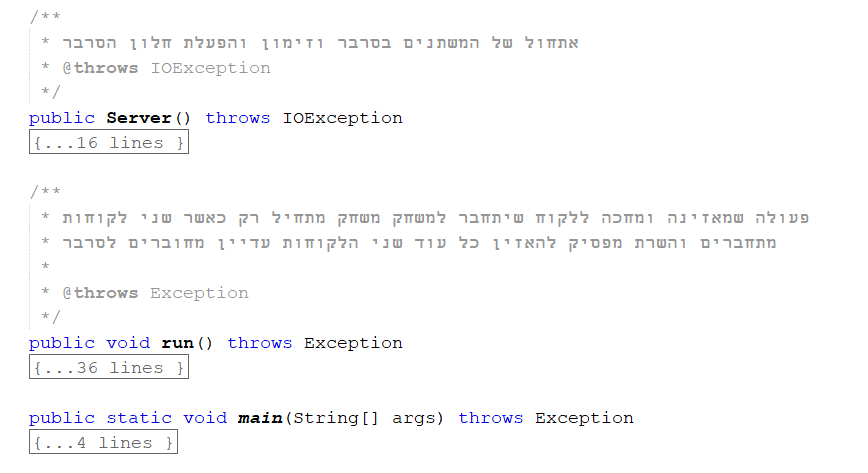
e.printStackTrace();

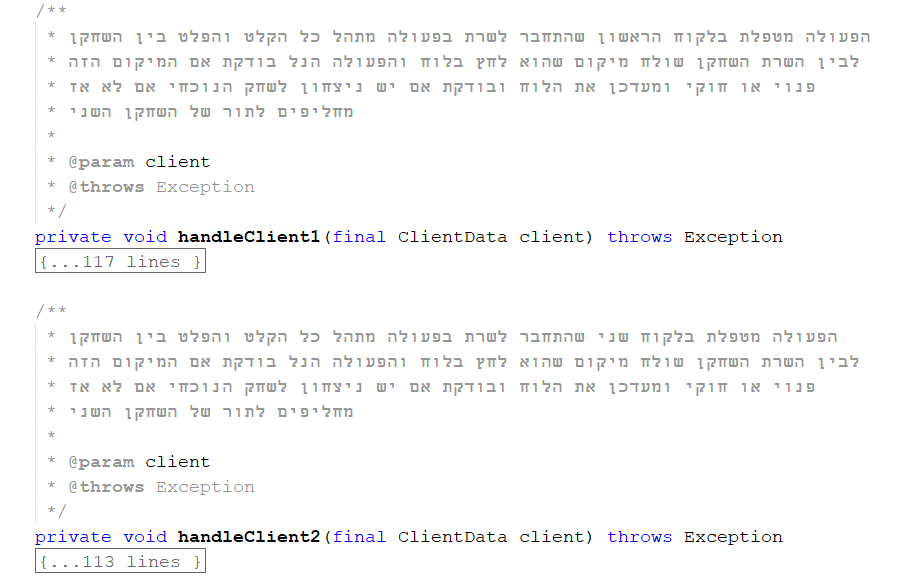
}

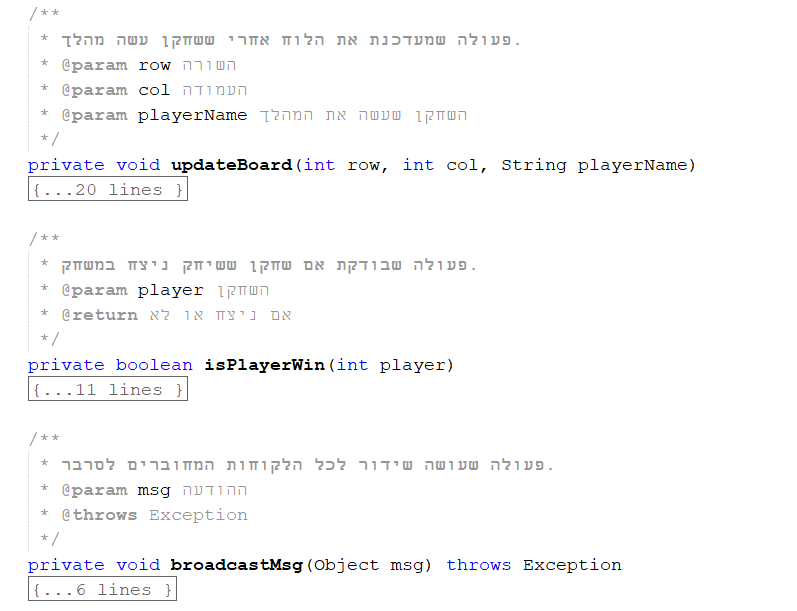
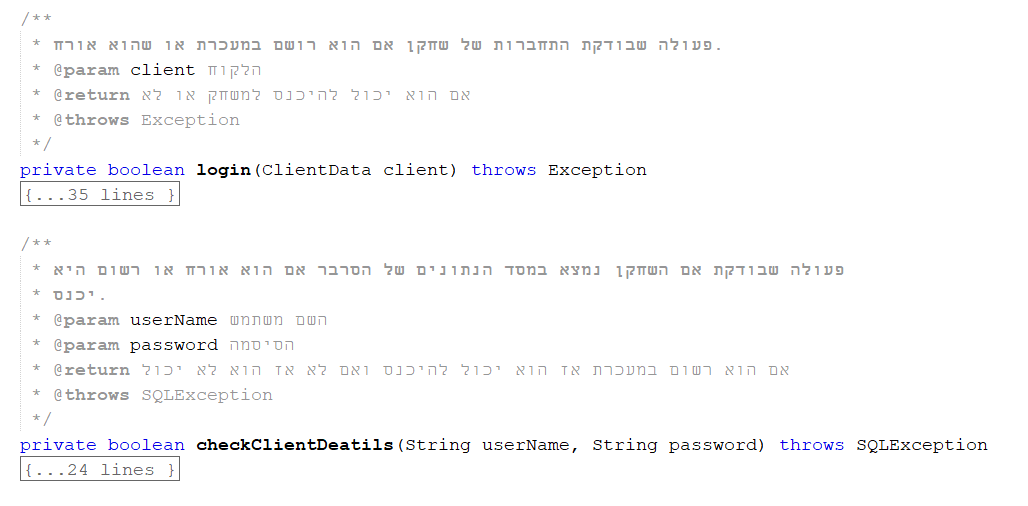
}

**תוכנית שרת:**

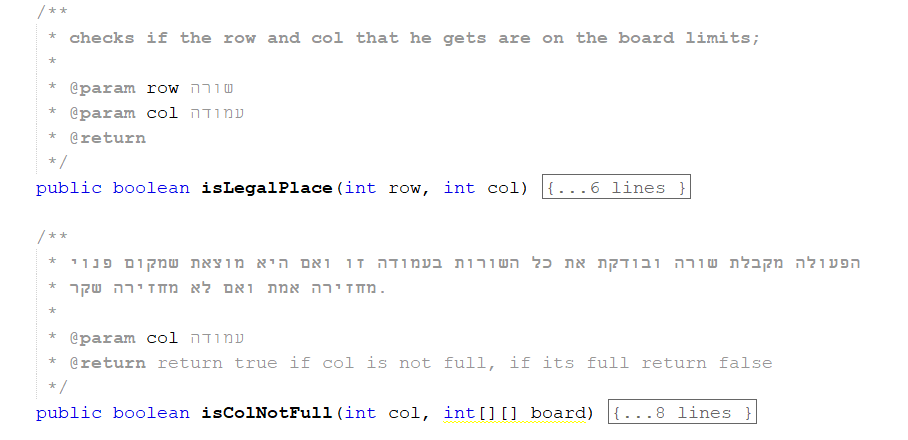
המחלקה :Server







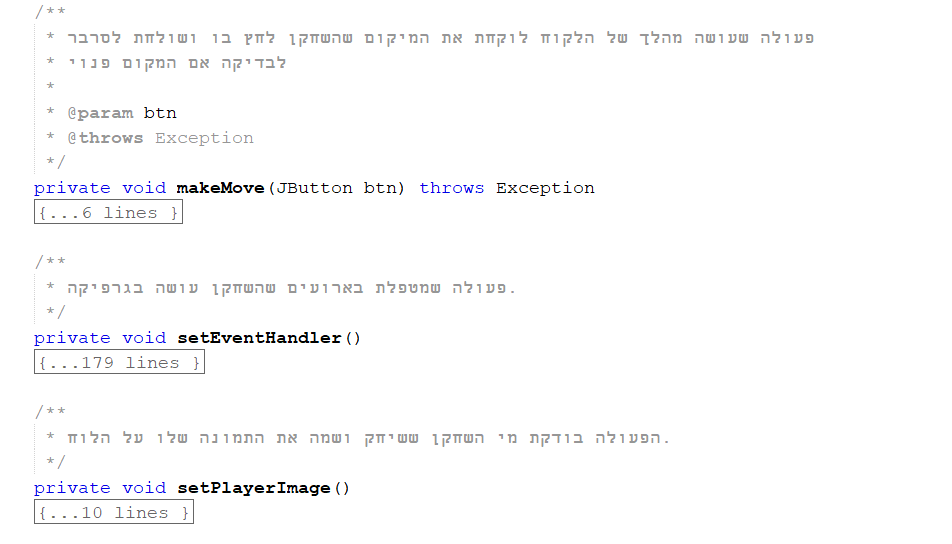
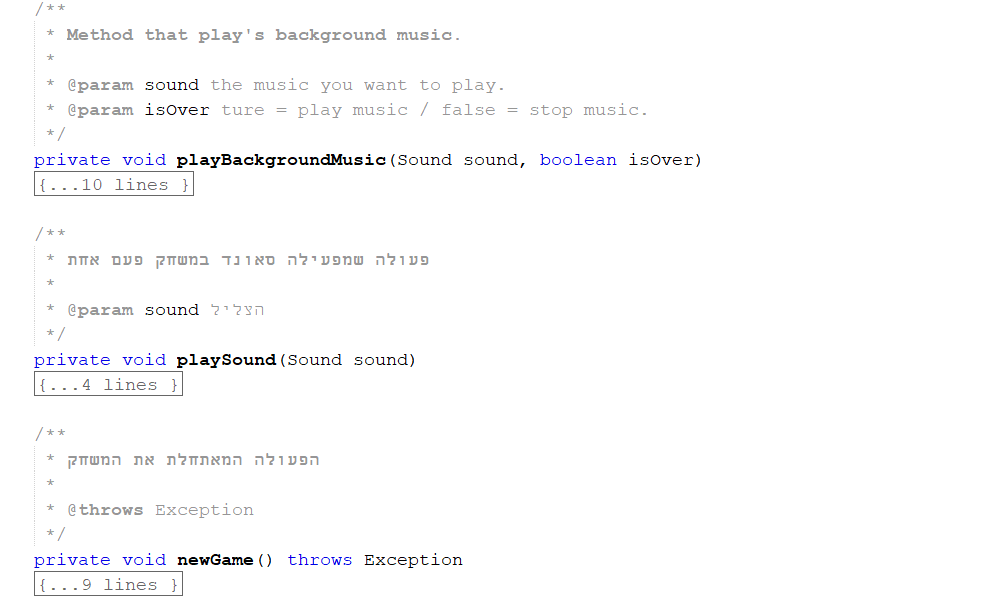
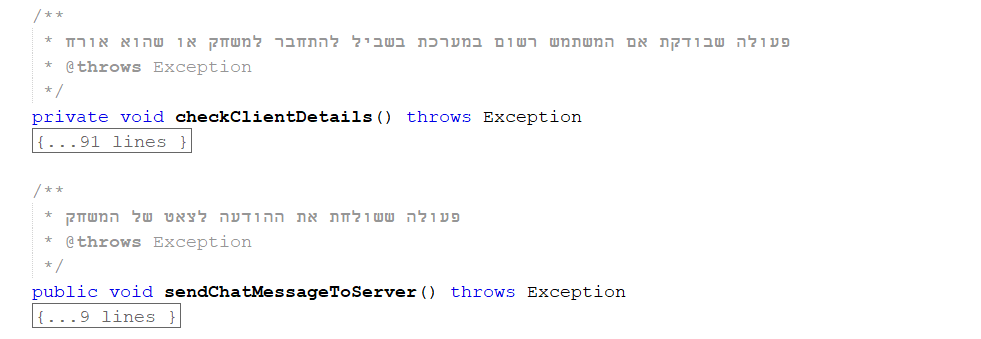


המחלקה GameLogic



**תוכנית לקוח:**

המחלקה Client





המחלקה ClientGUI

