🎮 Memory Game - משחק זיכרון רב-משתתפים

משחק זיכרון תחרותי לשני שחקנים עם ארכיטקטורת Client-Server, פותח ב-Java עם JavaFX.

Show Image Show Image Show Image

📋 תוכן עניינים

🎯 אודות הפרויקט

משחק זיכרון קלאסי בגרסה מתקדמת שמאפשרת לשני שחקנים להתחרות בזמן אמת דרך רשת.

השחקנים מתחברים לשרת מרכזי ומשחקים בתורות - כל שחקן מנסה למצוא זוגות של קלפים תואמים. השחקן שמוצא את מספר הזוגות הגדול ביותר - מנצח!

🎲 חוקי המשחק

1. שני שחקנים מתחברים לשרת
2. הלוח מכיל קלפים הפוכים עם תמונות זהות בזוגות
3. בכל תור, שחקן בוחר שני קלפים
4. אם הקלפים תואמים - השחקן זוכה בנקודה וממשיך לשחק
5. אם הקלפים לא תואמים - התור עובר לשחקן השני
6. המשחק מסתיים כשכל הזוגות נמצאו
7. השחקן עם הניקוד הגבוה ביותר מנצח!

✨ תכונות עיקריות

🌐 רשת מתקדמת

- **ארכיטקטורת Client-Server** - שרת מנהל מספר משחקים במקביל
- **תקשורת בזמן אמת** - עדכונים מיידיים בין השחקנים
- **ניהול תורות אוטומטי** - השרת מנהל את התורות והניקוד
- **מתקדם**-ExecutorService לניהול חיבורים מרובים - שימוש ב **Threading**

🎨 ממשק משתמש

- **JavaFX GUI** - ממשק משתמש גרפי מושקע

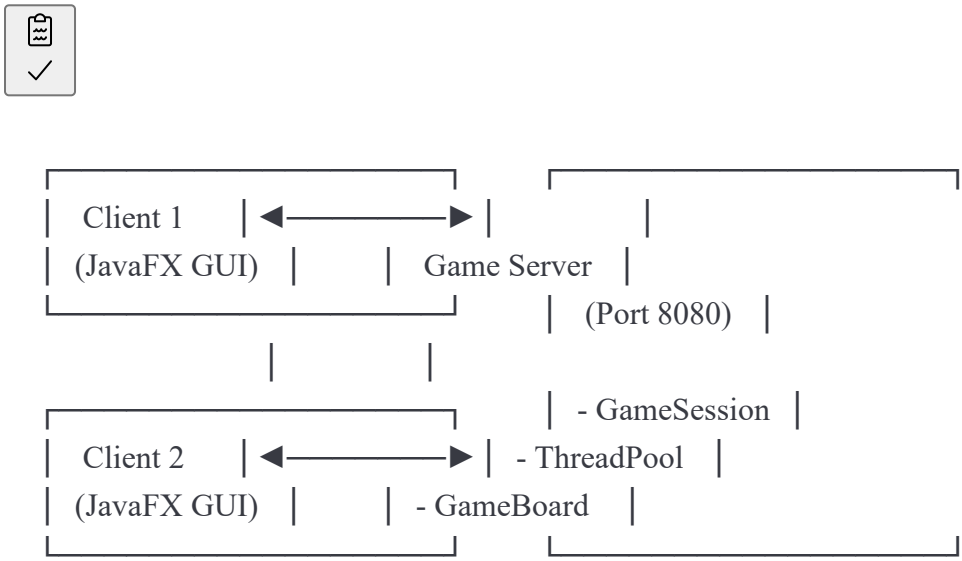
- **עיצוב רספונסיבי** - הלוח מתאים את עצמו לגודל (4x4, 5x5, 6x6...)
- **תמונות דינמיות** - תמיכה ב-40 תמונות שונות
- **אנימציות ויזואליות** - צבעים משתנים (ירוק=זוג נכון, אדום=שגוי)
- **עדכוני סטטוס בזמן אמת** - הודעות על מצב המשחק והתור הנוכחי

📐 קוד איכותי

- **אובייקטים סריאליזביליים** - העברת נתונים מובנית בין שרת ללקוח
- **הפרדת אחריות (Separation of Concerns)** - קוד מאורגן ומודולרי
- **טיפול בשגיאות** - ניהול חיבורים ושגיאות רשת
- **תיעוד מקיף** - JavaDoc לכל המחלקות והפונקציות

📐 ארכיטקטורה

מבנה המערכת



רכיבי המערכת

צד שרת

- **MemoryGameServer** - שרת ראשי שמאזין לחיבורים
- **GameSession** - מנהל משחק בין שני שחקנים
- **GameBoard** - לוגיקת הלוח והקלפים

צד לקוח

- **MemoryGameClient** - אפליקציית JavaFX
- **GameController** - מנהל את הממשק והלוגיקה
- **MemoryGameView.fxml** - עיצוב הממשק

מודלים משותפים

- **GameMessage** - הודעות בין שרת ללקוח
- **CardSelection** - בחירת קלפים של שחקן
- **TurnResult** - תוצאות תור
- **GameBoard** - מצב הלוח

💻 דרישות מערכת

- **Java JDK 8** ומעלה
- **JavaFX SDK** (כלול ב-JDK 8-10, או להורדה נפרדת ב-JDK 11+)
- תיקיית **תמונות** - `images/` עם קבצי תמונות (img1.jpg ... img40.jpg)
- **מערכת הפעלה**: Windows / macOS / Linux

🚀 התקנה והרצה

1️⃣ הכנת תיקיית תמונות

בתיקיית הפרויקט והכנס 40 תמונות `images` צור תיקייה בשם:

```
project/
├── images/
│   ├── img1.jpg
│   ├── img2.jpg
│   ├── img3.jpg
│   └── ... (עד img40.jpg)
└── src/
```

חשוב: התמונות צריכות להיות בפורמט JPG או PNG.

2️⃣ קומפילציה

```bash
# קומפילציה של כל הקבצים
javac *.java
```

3️⃣ הרצת השרת

```bash
```

```
# הרצה עם ברירת מחדל (4 לוח, 8080 פורט x4)
java MemoryGameServer

# הרצה עם הגדרות מותאמות אישית
java MemoryGameServer 8080 5

# פורמט: java MemoryGameServer <port> <board_size>
```

פרמטרים:

- `port` - מספר הפורט (ברירת מחדל: 8080)
- `board_size` - גודל הלוח NxN (ברירת מחדל: 4, מקסימום: 6)

4️⃣ הרצת הלקוחות

פתח שני טרמינלים (עבור שני שחקנים):

bash

```
# טרמינל 1 - שחקן 1
java MemoryGameClient

# טרמינל 2 - שחקן 2
java MemoryGameClient

# התחברות לשרת מרוחק
java MemoryGameClient 192.168.1.100 8080
```

פרמטרים:

- `host` - כתובת השרת (ברירת מחדל: localhost)
- `port` - מספר הפורט (ברירת מחדל: 8080)

📁 מבנה הקבצים

```
memory-game/
│
├── 📄 README.md                # התיעוד הזה
│
├── 🖥️ Server Side
│   ├── MemoryGameServer.java     # שרת ראשי
│   ├── GameSession.java          # ניהול משחק בין 2 שחקנים
│   └── GameBoard.java            # לוגיקת לוח המשחק
│
├── 💻 Client Side
│   ├── MemoryGameClient.java     # אפליקציית JavaFX
│   ├── GameController.java       # בקר ממשק המשתמש
│   └── MemoryGameView.fxml       # עיצוב UI
│
├── 📦 Shared Models
│   ├── GameMessage.java          # הודעות רשת
│   ├── CardSelection.java        # בחירת קלף
│   └── TurnResult.java           # תוצאות תור
│
└── 🖼️ images/
    ├── img1.jpg
    ├── img2.jpg
    └── ... (40 תמונות)
```

🛠️ טכנולוגיות

<table> <tr> <td>שפת תכנות</td> <td>Java 8+</td> </tr> <tr> <td>ממשק משתמש</td> <td>JavaFX + FXML</td> </tr> <tr> <td>תקשורת רשת</td> <td>Java Sockets + ObjectStreams</td> </tr> <tr> <td>Concurrency</td> <td>ExecutorService, Threads, BlockingQueue</td> </tr> <tr> <td>Serialization</td> <td>Java Serializable</td> </tr> <tr> <td>Design Patterns</td> <td>MVC, Client-Server, Observer</td> </tr> </table>

📸 צילומי מסך

כאן תוכל/י להוסיף תמונות או GIF בפעולה של המשחק

📋 markdown

```
![Game Screenshot](screenshots/game.png)
![Winner Screen](screenshots/winner.png)
```

🎯 תכונות טכניות מתקדמות

ניהול חיבורים

- לניהול תור שחקנים ממתינים **BlockingQueue**-שימוש ב
- לניהול דינמי של חיבורים **CachedThreadPool**
- טיפול בניתוקים פתאומיים

סנכרון

- **synchronized methods** למניעת race conditions
- Platform.runLater() לעדכוני UI thread-safe ב-JavaFX

Responsive Design

- התאמה אוטומטית של גודל הקלפים לפי מספר השורות והעמודות
- התאמת גודל החלון לגודל הלוח
- לתמיכה בלוחות גדולים ScrollPane

🐛 פתרון בעיות נפוצות

"Connection refused"

- ודא שהשרת רץ לפני שהלקוחות מתחברים
- בדוק שהפורט נכון (8080 כברירת מחדל)

"Images not loading"

- קיימת `images/` ודא שתיקיית
- בדוק ששמות הקבצים תואמים: img1.jpg ... img40.jpg
- תמיכה ב-JPG ו-PNG

"Board size error"

- גודל הלוח חייב להיות זוגי (4x4, 6x6 וכו')
- 6 מקסימוmx6 (36 זוגות = 18 קלפים)
- 2 מינימוmx2

🎓 מטרת הפרויקט

פרויקט זה פותח במסגרת לימודי תכנות מתקדם והדגמת:

- ✅ תכנות רשת ב-Java (Sockets, Streams)
- ✅ תכנות מקבילי (Threading, Concurrency)
- ✅ בניית GUI עם JavaFX
- ✅ ארכיטקטורת Client-Server
- ✅ Object-Oriented Programming
- ✅ עיצוב ממשקי משתמש רספונסיביים

🤝 תרומה

רוצה לתרום לפרויקט? מוזמנ/ת!

1. Fork את הפרויקט
2. צור branch חדש (`git checkout -b feature/AmazingFeature`)
3. Commit את השינויים (`git commit -m 'Add some AmazingFeature'`)
4. Push ל-branch (`git push origin feature/AmazingFeature`)
5. פתח Pull Request

📝 רישיון

פרויקט זה הוא בקוד פתוח ללא רישיון ספציפי. ניתן להשתמש, לשנות ולהפיץ לצרכים לימודיים.

👨‍💻 יוצר/ת הפרויקט

[שמך/שמך]

- 📧 Email: your.email@example.com
- 🔗 GitHub: @yourUsername
- 💼 LinkedIn: linkedin.com/in/yourprofile

⭐ אם אהבת את הפרויקט, תן/י לו כוכב! ⭐

Made with ❤️ and ☕ by [שמך]