פרויקט גמר במדעי המחשב – תכנון ותכנות מערכות מונחות עצמים ואפליקציות אסינכרוניות – 5 יחידות לימוד

שם העבודה: Risk Online

שם המגיש: איתי אלפסי

מספר זיהוי: 315188896

בית ספר: מקיף ו אשדוד

שם המורה: אולג כצנלסון

שנה: תשע"ח, 2018

מבוא

בפרויקט שלי בחרתי לבנות משחק מחשב המבוסס על משחק הלוח האייקוני "סיכון".

את התוכנה הצלחתי לבנות בעזרת הכלים שרכשתי בשיעורי מדעי במחשב במהלך שנות לימודי בתיכון ובאמצעות ידע שרכשתי באופן עצמאי. הפרויקט מורכב משני מודולים שונים – השרת והלקוח. השרת נבנה בשפת C# בארכיטקטורת .NET והלקוח נבנה בשפת C# בארכיטקטורת UWP.

הפרויקט דרש ממני לממש את כל הידע שלי בתכנות מונחה עצמים ובכלל: אלגוריתמיקה, תקשורת, מחלקות, עצמים, פעולות, הורשה, תהליכונים וכד'.

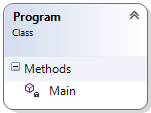
המשחק הוא משחק אסטרטגיה, בו מטרתו העיקרית של השחקן היא – להשתלט על העולם!

**מדריך למתכנת – שרת**

**דיאגרמת מבנה השרת:**



**מחלקות הפרויקט:**



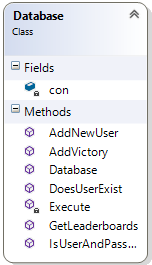
**Program.cs**

המחלקה הזו היא המחלקה הראשית. בה נוצא אובייקט הServer ובה הוא מופעל.

**פעולות המחלקה:**

static void Main(string[] args)

הפעולה הראשית של השרת. בפעולה זו יש שתי פקודות בלבד: בניית השרת והפעלתו.



**Database.cs**

מחלקה זו אחראית על ניהול ושימוש במסד הנתונים של השרת.

להלן פירוט של מבנה מסד הנתונים (המסד מכיל טבלה אחת בשם User):

|  |  |  |  |
| --- | --- | --- | --- |
| שם השדה | טיפוס | הסבר | מפתח ראשי? |
| username | TEXT | שם המשתמש | כן |
| password | TEXT | הסיסמה של המשתמש | לא |
| victories | INTEGER | מס' הניצחונות של המשתמש | לא |

**תכונות המחלקה:**

con – עצם מטיפוס SqliteConnection המייצג את החיבור של התוכנה למסד הנתונים.

**פעולות המחלקה:**

private SQLiteDataReader Execute(string query)

טענת כניסה: הפעולה מקבלת שאילתת SQL שאותה רוצה השרת להפעיל

טענת יציאה: הפעולה מריצה את השאילתה ומחזירה עצם לקריאת נתונים (הם תוצאת השאילתה).

public bool IsUserAndPassMatch(string username, string password)

טענת כניסה: הפעולה מקבלת שם משתמש וסיסמה שאותם רוצה השרת לבדוק

טענת יציאה: הפעולה בודקת האם ישנה רשומה בטבלה עם שם המשתמש והסיסמה הנתונים. הפעולה מחזירה true אם כן וfalse אחרת.

public bool DoesUserExist(string username)

טענת כניסה: הפעולה מקבלת שם משתמש.

טענת יציאה: הפעולה בודקת האם ישנה רשומה בטבלה עם שם המשתמש הנתון. הפעולה מחזירה true אם כן וfalse אחרת.

public void AddNewUser(string username, string password)

טענת כניסה: הפעולה מקבלת שם משתמש וסיסמה של משתמש.

טענת יציאה: הפעולה מוסיפה רשומה חדשה למסד של משתמש חדש (עם שם משתמש וסיסמה שהוא קבע).

public string[,] GetLeaderboards()

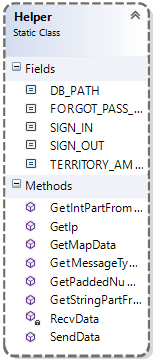
טענת כניסה: אין

טענת יציאה: הפעולה שולפת ממסד הנתונים את הנתונים את שמונת המשתמשים בעלי הניקוד הגבוה ביותר (בסדר ניקוד יורד), מכניסה אותם לתוך מערך דו-ממדי ומחירה את המערך.

public void AddVictory(string username)

טענת כניסה: הפעולה מקבלת שם משתמש.

טענת יציאה: הפעולה מעדכנת במסד הנתונים כי המשתמש ניצח במשחק ומעלה את מספר הניצחונות שלו.

**Helper.cs**

מחלקה סטטית המכילה קבועים ופעולות שונות העוזרים להתנהלות השרת.

**שדות המחלקה:**

SIGN\_IN … FORGOT\_PASS\_OTHER – מחרוזות קבועות המייצגות את ערכי פרוטוקול התקשורת לסוגי הודעות שונים (הפרוטוקול המלא מפורט בנספחים).

TERRITORY\_COUNT – מספר קבוע המתאר את מס' הטריטוריות במשחק.

DB\_PATH – כתובת הזיכרון היחסית של מסד הנתונים במחשב.

**פעולות המחלקה:**

public static void SendData(string message, TcpClient client)

טענת כניסה: הפעולה מקבלת הודעה וחיבור TCP (סוקט).

טענת יציאה: הפעולה שולחת את ההודעה דרך הסוקט כדי שתגיע ללקוח אליו שייך החיבור.

private static string RecvData(int size, TcpClient client)

טענת כניסה: הפעולה מקבלת גודל קלט וסוקט.

טענת יציאה: הפעולה קוראת מתוך הסוקט מידע בגודל size בתים ומחזירה אותו לקורא הפעולה כמחרוזת.

public static string GetPaddedNumber(int num, int size)

טענת כניסה: הפעולה מקבלת מספר וגודל מחרוזת.

טענת יציאה: הפעולה הופכת את המספר למחרוזת בגדל המצוין. אם כמות הספרות קטנה מהגודל הרצוי, המספר יוקדם באפסים (zero-padding).

public static string GetMessageTypeCode(TcpClient client)

טענת כניסה: הפעולה מקבלת סוקט של לקוח.

טענת יציאה: הפעולה משיגה מתוך הסוקט את קוד ההודעה הבאה (קוד הודעות נמצא בגודל קבוע של 3 בתים).

public static int GetIntPartFromSocket(TcpClient client, int size)

טענת כניסה: הפעולה מקבלת גודל קלט וסוקט.

טענת יציאה: הפעולה קוראת מתוך הסוקט מידע בגודל size בתים ומחזירה אותו לקורא הפעולה כמספר שלם.

public static string GetIp(TcpClient client)

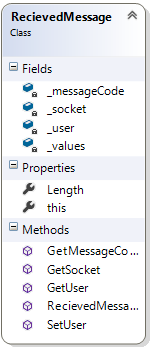
טענת כניסה: סוקט של לקוח.

טענת יציאה: הפעולה מחזירה את כתובת הIP של הלקוח שאליו מחובר הסוקט.

public static List<string>[] GetMapData()

טענת כניסה: אין

טענת יציאה: הפעולה מנתחת את הקובץ mapdata.csv המכיל את רשימת כל הטריטוריות, ועבור כל טריטוריה את אלו המחוברות אליה, ומחזירה את המידע כמערך של רשימות מחרוזות. (כל List מייצגת שורה בקובץ הCSV.)

**ReceivedMessage.cs**

מחלקה זו מייצגת הודעה שהשרת קיבל.

**תכונות המחלקה:**

\_messageCode – קוד (סוג) ההודעה.

\_socket – חיבור התקשורת (הסוקט) שממנו נשלחה ההודעה.

\_user – המשתמש ששלח את ההודעה. אם המשתמש עדיין לא נרשם (במקרה של הודעת הרבמה), ערך השדה יהיה null.

\_values – עצם מטיפוס List<string> המכיל את רשימת ערכי ההודעה.

Length – כמות ערכי ההודעה.

**פעולות המחלקה:**

public TcpClient GetSocket()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את הסוקט שממנו נשלחה ההודעה.

public User GetUser()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את המשתמש ששלח את ההודעה (null אם לא ידוע.)

public void SetUser(User user)

טענת כניסה: עצם מסוג משתמש השלח את ההודעה.

טענת יציאה: הפעולה קובעת את הערך \_user לuser שהועבר.

public string GetMessageCode()

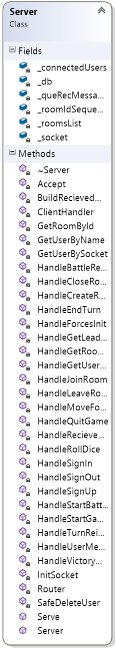
טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את קוד ההודעה.

public string this[int index]

טענת כניסה: אינדקס הערך ברשימת הערכים \_values.

טענת יציאה: פעולה זו היא העמסת האופרוטר []. כאשר מפעילים את האופרטור [] על עצם מסוג ReceivedMessage, יוחזר הערך ברשימה \_values במקום שצוין.

**Server.cs**

מחלקה זו היא המחלקה הראשית של הפרויקט. היא מיצגת את כל השרת ואת פעולותיו.

תכונות המחלקה:

\_socket – חיבור התקשורת (הסוקט) של השרת.

\_connectedUsers – מילון המכיל את הסוקט של כל משתמש כמפתח, ועצם המשתמש עצמו כערך.

\_roomsList – מילון המכיל את המספר המזהה של כל חדר כמפתח, ואת החדר עצמו כערך.

\_db – עצם הDatabase של ניהול הנתונים של השרת.

\_qurRecMessages – תור המכיל את כל ההודעות של הלקוחות המתקבלות לשרת (לפני שטיפלו בהן).

\_roomIdSequence – תכונה המכילה את המספר המזהה לחדר הבא בתור.

**פעולות המחלקה:**

public Server()

טענת כניסה: אין

טענת יציאה: הפעולה הבונה. בונה את כל מבני הנתונים של הפרויקט.

public void Serve()

טענת כניסה: אין

טענת יציאה: הפעולה פותחת תהליכון (Thread) להאזנה להודעות מהלקוחות ונכנס ללולאה אינסופית של קבלת לקוחות חדשים.

private void InitSocket()

טענת כיסה: אין

טענת יציאה: הפעולה בונה את חיבור התקשורת (הסוקט) של השרת.

private void Accept()

טענת כניסה: אין

טענת יציאה: הפעולה מקבלת לקוח שיצר קשר עם השרת ובונה תהליכון (Thread) שיטפל בהודעות המתקבלות ממנו.

private RecievedMessage BuildRecievedMessage(TcpClient client, string msgCode)

טענת כניסה: הסוקט שבו נמצאת ההודעה וקוד ההודעה המתקבלת.

טענת יציאה: הפעולה קוראת מתוך הסוקט את תוכן ההודעה לפי המבנה הספציפי של ההודעה שהתקבלה, ומחזירה את מבנה ההודעה שהתקבל.

private void HandleSignIn(RecievedMessage msg)

טענת כניסה: הודעת מלקוח.

טענת יציאה: הפעולה מנסה לחבר את הלקוח לשרת ושולחת לו הודעה אם הצליח (או לא הצליח) להתחבר. (לא מצליח להתחבר אם אין משתמש כזה, הסיסמה לא נכונה, או שהמשתמש כבר מחובר.)

private bool HandleSignUp(RecievedMessage msg)

טענת כניסה: הודעת מלקוח.

טענת יציאה: הפעולה מנסה לרשום את המשתמש למשחק ושולחת לו הודעה אם הצליח (או לא הצליח) להתחבר. (לא מצליח להתחבר אם המשתמש כבר קיים.)

private void HandleSignOut(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מוחקת את הלקוח מתוך המשחק / החדר שהוא נמצא בהם ומסירה אותו מרשימת המשתמשים המחוברים.

private bool HandleCreateRoom(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מנסה ליצור חדר חדש לפי דרישות המשתמש ושולחת אליו הודעה האם יצירת החדר הצליחה, ומחזירה את ההצלחה כbool. (היצירה נכשלת אם יש כבר חדר בשם הזה.)

private void HandleCloseRoom(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מנסה להפעיל את פעולת סגירת החדר, ואם הצליחה, מסירה את בחדר מרשימת החדרים. הפעולה מחזירה את הצלחתה כbool. (יכולה להיכשל אם הלקוח אינו מנהל החדר.)

private bool HandleLeaveRoom(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת ההוצאה מהחדר של החדר של הלקוח. (מחזירה false רק במקרה שהמשתמש או החדר שלו לא קיימים.)

private void HandleGetUsersInRoom(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת שליחת רשימת המשתמשים של החדר המבוקש.

private void HandleGetRooms(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה שולחת ללקוח רשימה של כל החדרים הזמינים בשרת.

private void HandleStartGame(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה יוצאת משחק חדש עם משתמשי החדר, שולחת הודעה לשחקנים על פתיחת המשחק, ומוחקת את החדר.

private void HandleQuitGame(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף העוזב.

private void HandleForcesInit(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף השולח את פיזור הכוחות הראשוני שלו.

private void HandleTurnReinforcements(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף שמבצע תגבור כוחות.

private void HandleMoveForces(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף שמעביר כוחות בין טריטוריות.

private void HandleEndTurn(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף שסיים את תורו.

private void HandleStartBattle(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול בפתיחת קרב בין משתתפים.

private void HandleRollDice(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתתף בקרב שהטיל את הקוביות שלו.

private void HandleBattleRetreat(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתמש שנסוג מקרב.

private void HandleVictoryMoveForces(RecievedMessage msg)

טענת כניסה: הודעה מלקוח.

טענת יציאה: הפעולה מפעילה את פעולת המשחק לטיפול במשתמש שמעביר כוחות למדינה שכבש.

private void HandleUserMessage(RecievedMessage msg)

טענת כניסה: הודעת צ'אט מלקוח.

טענת יציאה: הפעולה מעבירה את הודעת הצ'אט לשחר המשתתפים במשחק שלו.

private void HandleGetLeaderboards(RecievedMessage msg)

טענת כניסה: הודעה מלקוח המבקש את טבלת האלופים.

טענת יציאה: הפעולה משיגה את טבלת האלופים ממסד הנתונים ושולחת אותה למשתמש.

private void ClientHandler(object data)

טענת כניסה: הפעולה מקבלת את חיבור התקשורת (הסוקט) של משתמש (הטיפוס הוא object ולא TcpClient מאחר וביצירת תהליכון (Thread) ניתן להעביר אך רוק object.)

טענת יציאה: הפעולה רצה כל עוד המשתמש מתקשר עם השרת, קולטת את ההודעות שהמשתמש שולח, ומכניסה אותה לתור ההודעות. אם התקשורת עם המשתמש קורסת, הפעולה מכניסה הודעת EXIT\_APP המדמה יציאה רגילה של לקוח מהאפליקציה.

private void HandleRecievedMessages()

טענת כניסה: אין

טענת יציאה: הפעולה פועלת באופן אינסופי על תהליכון (Thread) נפרד, אשר מטרתו לזהות הודעות הנכנסות לתור ההודעות המשותף לכל המשתמשים, ולהעביר את ההודעה בראשית התור לפונקצית המיון (Router).

private void Router(RecievedMessage msg)

טענת כניסה: הודעה מלקוח כלשהו.

טענת יציאה: הפעולה קוראת את קוד ההודעה שהתקבלה ומעבירה אותה לפעולת הטיפול (Handler) המתאימה.

private User GetUserBySocket(TcpClient client)

טענת כניסה: סוקט של לקוח.

טענת יציאה: הפעולה מחזירה את המשתמש שלו שייך הסוקט.

private User GetUserByName(string username)

טענת כניסה: שם משתמש השייך ללקוח.

טענת יציאה: הפעולה מוצאת ומחזירה את המשתמש שלו שייך השם הנתון.

private Room GetRoomById(int id)

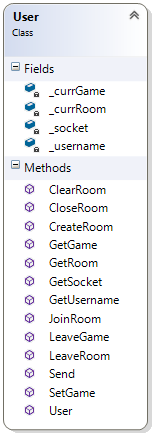
טענת כניסה: מספר מזהה של חדר.

טענת יציאה: הפעולה מחפשת ומחזירה את החדר בעל המספר המזהה הנתון.

private void SafeDeleteUser(RecievedMessage msg)

טענת כניסה: הודעה מלקוח אשר יצא מהאפליקציה.

טענת יציאה: הפעולה מפעילה את פעולת הSignOut כדי לדאוג להתנתקות של המשתמש מכל החדרים והמשחקים אליהם הוא מחובר, וסוגרת את ערוץ התקשורת (הסוקט).

**User.cs**

מחלקה זו מייצגת לקוח/משתמש/שחקן המחובר לאפליקציה ולשרת.

הפעולות שבה אחראיות על תקשורת עם השרת וחיבור של המשתמש לחדרים/משחקים בשרת.

**שדות המחלקה:**

\_currGame – הפניה לעצם מטיפוס Game המייצג את המשחק שבו כרגע משתתף המשתמש (null אם המשתמש לא משחק כרגע).

\_currRoom – הפניה לעצם מטיפוס Room המייצג את המשחק שבו כרגע נמצא המשתמש (null אם המשתמש לא נמצא בחדר כרגע).

\_socket – עצם מטיפוס TcpClient המייצג את ערוץ התקשורת (הסוקט) של אפליקציית הלקוח עם השרת.

\_username – משתה מטיפוס string המכיל את שם המשתמש של הלקוח.

**פעולות המחלקה:**

public void Send(string message)

**טענת כניסה: הודעה לשליחה.**

**טענת יציאה: הפעולה שולחת למשתמש את תוכן ההודעה שקיבלה.**

public void SetGame(Game game)

**טענת כניסה: משחק אליו המשתמש התחבר.**

**טענת יציאה: הפעולה קובעת את שדה** \_currGame **של המשתמש למשחק שהתקבל.**

public string GetUsername()

**טענת כניסה: אין**

**טענת יציאה: הפעולה מחזירה את שם המשתמש.**

public TcpClient GetSocket()

**טענת כניסה: אין**

**טענת יציאה: הפעולה מחזירה את הסוקט של המשתמש.**

public Game GetGame()

**טענת כניסה: אין**

**טענת יציאה: הפעולה מחזירה את ערך השדה** \_currGame**.**

public Room GetRoom()

**טענת כניסה: אין**

**טענת יציאה: הפעולה מחזירה את ערך המשתנה** \_currRoom**.**

public bool CreateRoom(int roomId, string name, int maxUsers)

**טענת כניסה: מספר מזהה, שם לחדר, ומס' שחקנים מקסימלי לחדר.**

**טענת יציאה: הפעולה בונה חדר לפי הפרמטרים שהועברו לפעולה, כאשר מנסה החדר הוא המשתמש הנוכחי (**this**).**

public bool JoinRoom(Room newRoom)

**טענת כניסה: חדר אליו המשתמש מנסה להיכנס.**

**טענת יציאה: הפעולה בודקת את המשתמש לא נמצא כבר בחדר אחר, ומנסה להפעיל את פעולת ההצטרפות לחדר של החדר עצמו. הפעולה מחזירה האם ההצטרפות הצליחה.**

public void LeaveRoom()

**טענת כניסה: אין**

**טענת יציאה: הפעולה מפעילה את פעולה עזיבת החדר של החדר על המשתמש הנוכחי (**this**) ומשנה את ערך השדה המתאים ל**null**.**

public int CloseRoom()

טענת כניסה: אין

טענת יציאה: הפעולה מנסה לסגור את החדר אליו מחובר המשתמש. הפעולה מחזירה את קוד החדר שנסדר אם הפעולה הצליחה, ו-1 אם הפעולה נכשלה.

public bool LeaveGame()

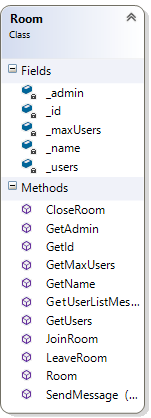
טענת כניסה: אין

טענת יציאה: הפעולה מוחקת את הערך של \_currGame, ומחזירה האם המשתמש היה מחובר למשחק מלכתחילה.

public void ClearRoom()

טענת כניסה: אין

טענת יציאה: הפעולה משנה את הערך של \_currRoom לnull.

**Room.cs**

מחלקה זו מייצגת חדר משחק, שבו מתאספים משתתפים לפני תחילתו של משחק.

**תכונות המחלקה:**

\_admin – עצם מטיפוס User המפנה למנהל החדר.

\_id – המספר המזהה של החדר.

\_maxUsers – מס' המשתמשים המקסימלי לחדר.

\_name – שם החדר.

\_users – עצם מטיפוס List<User> המכיל את רשימת כל המשתמשים בחדר.

**פעולות המחלקה:**

public void SendMessage(string message)

טענת כניסה: הפעולה מקבלת הודעה לשליחה.

טענת יציאה: הפעולה שולחת את ההודעה לכל המשתמשים בחדר.

public void SendMessage(User excludeUser, string message)

טענת כניסה: הפעולה מקבלת הודעה לשליחה ומשתמש מסוים.

טענת יציאה: הפעולה שולחת את ההודעה לכל משתמשי החדר חוץ מהמשתמש שצוין.

public List<User> GetUsers()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את רשימת המשתמשים של החדר.

public int GetId()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את המספר המזהה של החדר.

public string GetName()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את שם החדר.

public int GetMaxUsers()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את כמות המשתמשים המקסימלית שניתן לשים בחדר.

public string GetAdmin()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את מנהל החדר.

public string GetUserListMessage()

טענת כניסה: אין

טענת יציאה: הפעולה מייצאת ושולחת הודעה המתארת את רשימת המשתמשים המחוברים לחדר.

public bool JoinRoom(User user)

טענת כניסה: משתמש המנסה להתחבר לחדר.

טענת יציאה: הפעולה בודקת האם היא יכולה להוסיף את המשתמש לחדר. אם כן, מוסיפה אותו לרשימת המשתמשים ומעדכנת את שאר המשתמשים, ואם לא שולחת לו הודעת כישלון בהתאם.

public void LeaveRoom(User user)

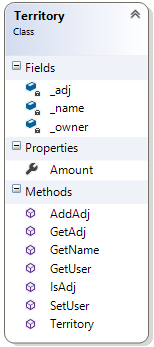
טענת כניסה: משתמש המנסה להתנתק מהחדר.

טענת יציאה: הפעולה מסירה את המשתמש מרשימת המשתמשים, שולחת הודעה שאומרת כי ההתנתקות הצליחה, ומעדכנת את שאר המשתמשים בחדר בנוגע לרשימת המשתמשים החדשה.

public int CloseRoom(User user)

טענת כניסה: משתמש המנסה לסגור את החדר.

טענת יציאה: הפעולה בודרת האם המשתמש הוא מנהל החדר. אם כן, היא סוגרת את החדר, שולחת הודעה מתאימה לכל המשתמשים ומוחקת את ערך הcurrRoom שלהם, ומחזירה את המספר המזהה של החדר. אם לא, הפעולה מחזירה -1.

**Territory.cs**

מחלקה זו מייצגת שטח (טריטוריה) במפת משחק.

**שדות המחלקה:**

\_name – שם הטריטוריה.

\_owner – עצם מסוג User המפנה להעלים של הטריטוריה כרגע במשחק.

\_adj – עצם מסוג Dictionary<string, Territory> המכיל את רשימת כל הטריטוריות הצמודות לטריטוריה הנוכחית (נתון קריטי במהלך המשחק.)

Amount – כמות יחידות הלוחמים בטריטוריה הנוכחית.

**פעולות המחלקה:**

public void AddAdj(Territory newTerritory)

טענת כניסה: טריטוריה נוספת הצמודה לטריטוריה הנוכחית.

טענת יציאה: הפעולה מוסיפה את הטריטוריה החדשה לרשימת \_adj.

public User GetUser()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את המשתמש שולט בטריטוריה.

public void SetUser(User owner)

טענת כניסה: המשתמש החדש ששולט בטריטוריה.

טענת יציאה: הפעולה מעדכנת את הערך \_owner להעלים החדשים של הטריטוריה.

public IEnumerable<Territory> GetAdj()

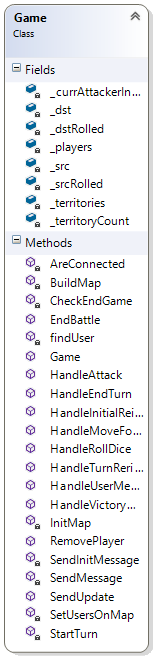
טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את רשימת הטריטוריות הצמודות לטריטוריה הנוכחית.

public bool IsAdj(Territory other)

טענת כניסה: טריטוריה שונה מהטריטוריה הנוכחית

טענת יציאה: הפעולה בודקת האם other צמודה לה, ומחזירה את התשובה כערך בוליאני.



**Game.cs**

מחלקה זו מייצגת משחק "סיכון" אקטיבי.

**שדות המחלקה:**

\_players – עצם מטיפוס List<User> המציג את רשימת השחקנים (המשתמשים) במשחק.

\_territories – עצם מטיפוס Dictionary<string, Territory> המכיל את שמות כל הטריטוריות כמפתחות ואת עצם הטריטוריות עצמן כערכים.

\_territoryCount – עצם מטיםוס Dictionary<User, int> המכיל את שחקני המשחק כמפתחות ואת כמות הטריטוריות שיש לכל אחד מהם כערך.

\_currAttackerIndex – האינדקס ברשימה \_players המפנה לשחקן התוקף כרגע.

\_src – כאשר ישנה אינטראקציה בין שתי טריטוריות, שדה זה מכיל את אחת מהן – לרוב את טריטורית המקור/התוקף.

\_dst - כאשר ישנה אינטראקציה בין שתי טריטוריות, שדה זה מכיל את אחת מהן – לרוב את טריטורית היעד/המגן.

\_srcRolled – שדה בוליאני הרלוונטי רק בשעת קרב. שדה זה מציין האם התוקף בקרב הטיל את קוביות הקרב.

\_dstRolled – שדה בוליאני הרלוונטי רק בשעת קרב. שדה זה מציין האם המגן בקרב הטיל את רוביות הקרב.

**פעולות המחלקה:**

private void InitMap()

טענת כניסה: אין

טענת יציאה: פעולה זו מפעילה מס' פעולות לאתחול המשחק ומפת המשחק.

private void BuildMap()

טענת כניסה: אין

טענת יציאה: הפעולה בונה את מבנה הנתונים של הטריטוריות (כך שכל טריטוריה תפנה לכל הטריטוריות הצמודות לה) לפי הנתונים הנמצאים בקובץ העזר mapdata.csv.

private void SetUsersOnMap()

טענת אניסה: אין

טענת יציאה: פעולה זו מחלקת את הטריטוריות השונות בין השחקנים וקובעת את הowner של כל טריטוריה בהתאם.

private void SendInitMessage()

טענת כניסה: אין

טענת יציאה: פעולה זו יוצאת ושולחת את הודעת אתחול המפה המעדכנת את כל המשתמשים במבנה המפה ההתחלתי.

public User RemovePlayer(User u)

טענת כניסה: משתמש שיש להסיר מהמשחק.

טוענת יציאה: הפעולה מוציאה את השחקן שהתקבל מרשימת השחקנים, בונה את המפה מחדש, ומאתחלת את המשחק מחדש.

private bool findUser(Territory t, string name)

טענת כניסה: טריטוריה ושם של אחד השחקנים.

טענת יציאה: הפעולה בודקת האם שם הבעלים של הטריטוריה הנתונה הוא הפרמטר name, ומחזירה את התשובה כערך בוליאני.

private void SendMessage(string message)

טענת כניסה: הודעה לשליחה.

טענת יציאה: הפעולה שולחת את ההודעה הנתונה לכל השחקנים.

public void HandleInitialReinforcements(RecievedMessage msg)

טענת כניסה: הודעה שנשלחה מלקוח.

טענת יציאה: הפעולה רושמת בתוך מבנה הטריטוריות את חלוקת הכוחות הראשונית של השחקן ששלח את ההודעה. כאשר כל השחקנים שלחו את חלוק הוקוחות שלהם, הפעולה תפעיל את פעולת "תחילת התור".

public void HandleUserMessage(string message)

טענת כניסה: הודעת צ'אט של שחקן.

טענת יציאה: הפעולה מפעילה את הפעולה האחראית להפצת הודעות לשאר השחקנים.

private void StartTurn()

טענת כניסה: אין

טענת יציאה: הפעולה אחראית על תחילתו של תור. היא שולחת עדכון כללי של המפה והודעת תחלת תור המציינת את השחקן התוקף הנוכחי.

public void SendUpdate()

טענת כניסה: אין

טענת יציאה: פעולה זו שולחת עדכון בנוגע למצב המפה לכל השחקנים.

public void HandleTurnRerinforcements(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: פעולה זו מקבלת הודעה על תגבור כוחות בתחילת תור, מכניסה את הערכים המעודכנים לטריטוריות המתאימות, ומעדכנת את המפה של כל השחקנים.

public void HandleMoveForces(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: פעולה זו מקבלת הודעה על העברת כוחות בסוף תור. הפעולה בודקת האם ההעברה חוקית, ושולחת הודעה בהתאם לשחקן שביצע את ההעברה.

public void HandleAttack(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: פעולה זו מקבלת הודעה על ניסיון תקיפה של שחקן אחד נגד שחקן אחר. הפעולה בודקת האם התקיפה חוקית (יש מספיק חיילים לתקיפה, הטריטוריות שייכות לשחקנים מתאימים, הטריטוריות צמודות אחת לשנייה). אם התקיפה חוקית, הודעה על תחילת קרב תישלח לכל החשקנים. אם לא, הודעת שגיאה מתאימה תישלח לתוקף בלבד.

public User HandleRollDice(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: פעולה זו מקבלת הודעה על תוקף/מגן שהטיל את הקוביות שלו. אם השני עוד לא הטיל, הפעולה "תמתין" לשחקן השני. אם שניהם הטילו, הפעולה תגריל את תוצאות הקוביות, תחשב את ההשלכות מבחינת חיילים, ותשלח את הקוביות והתוצאות לתוקף ולמגן.

public void HandleVictoryMoveForces(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: פעולה זו מקבלת הודעה על משתמש שמעביר כוחות לטריטוריה כבושה. הפעולה מעירה את הכוחות שצוינו לטריטוריה הכבושה ומעדכנת את השחקנים.

public User EndBattle(bool success)

טענת כניסה: ערך שמציין האם קרב הצליח (עבור התוקף).

טענת יציאה: פעולה זו מופעלת כאשר מסתיים קרב. הפעולה מעדכנת את מפת המשחק, שולחת את הגרסה המעודכנת לשחקנים, ואם התקיפה הצליחה, משנה את הבעלות של הטריטוריה שנכבשה.

private User CheckEndGame(User user)

טענת כניסה: אחד השחקנים.

טענת יציאה: הפעולה בודקת האם השחקן שהתקבל ניצח במשחק. אם כן, היא שולחת הודעה מתאימה לשחקנים, מוחקת את כולם מתוך המשחק, ומחזירה את השחקן המנצח. אם לא, הפעולה מחזירה null.

public void HandleEndTurn(RecievedMessage msg)

טענת כניסה: הודעה משחקן.

טענת יציאה: הפעולה מקבלת הודעה על משתמש שסיים את תורו. הפעולה מחפשת את המשתמש הבא בתור (שעדיין בעל לפחות טריטוריה אחת) ומתחילה תור חדש כאשר הוא התוקף.

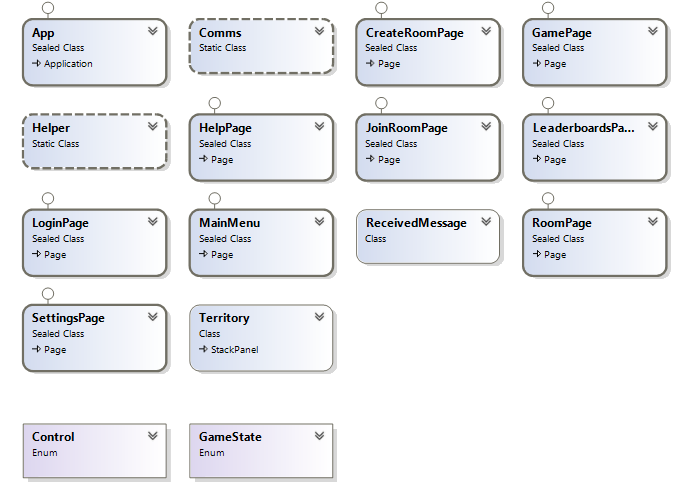
private bool AreConnected(Territory t1, Territory t2, int count = 10)

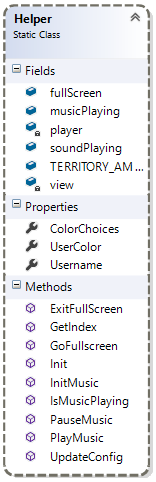
טענת כניסה: שתי טריטוריות, וספירת איטרציות (למטרת תנאי עצירכה רקורסיבי)

טענת יציאה: זוהי פעולה רקורסיבית הבודקת האם יש קשר רציף של טריטוריות השייכות למשתמש מסוים בין שתי הטריטוריות שצוינו. הפעולה תחזיר את התשובה כערך בוליאני.

**מדריך למתכנת – לקוח**

**דיאגרמת המחלקות:**



**מחלקות הלקוח:**

**Helper.cs**

מחלקה זו היא מחלקה סטטית המכילה נתונים, שדות ופעולות חשובות לפעולתה השוטפת של האפליקציה.

**שדות המחלקה:**

Player – עצם מטיפוס MediaPlayer האחראי על ניגון המוזיקה בפרויקט.

musicPlaying – ערך בוליאני המציין האם המוזיקה בפרויקט פועלת.

soundPlaying - ערך בוליאני המציין האם הצלילים בפרויקט פועלים.

fullScreen – ערך בוליאני המציין אם המשחק נמצא במצב של מסך מלא.

UserColor – הצבע הנוכחי של השחקן.

Username – שם המשתמש של המשתמש הנוכחי.

view – משתנה סטטי שמטרתו כניסה ויציאה ממצב full-screen.

TERRITORY\_AMOUNT – מספר הטריטוריות במשחק (קבוע כ42).

ColorChoices – מערך הצבעים השונים שמהם יכול לבחור המשתמש את צבע השחקן שלו.

**פעולות המחלקה:**

public static void PlayMusic()

טענת כניסה: אין

טענת יציאה: הפעולה מפעילה את המוזיקה בפרויקט.

public static void PauseMusic()

טענת כניסה: אין

טענת יציאה: הפעולה מפסיקה את המוזיקה בפרויקט.

public static void GoFullscreen()

טענת כניסה: אין

טענת יציאה: הפעולה מעבירה את האפליקציה למצב של מסך מלא.

public static void ExitFullScreen()

טענת כניסה: אין

טענת יציאה: הפעולה מוציאה את האפליקציה ממצב מסך מלא.

public static bool IsMusicPlaying()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה כערך בוליאני האם המוזיקה באפליקציה פועלת.

public static async void InitMusic()

טענת כניסה: אין

טענת יציאה: פעולה זו מאתחלת את עצם המוזיקה ומפעילה/משהה אותו (לפי קביעת ההגדרות המוקדמת).

public static void Init()

טענת כניסה: אין

טענת יציאה: פעולה זו מאתחלת את המחלקה ואת שדותיה לפי נתוני הLocalSettings של האפליקציה שנשמרים על הדיסק הקשיח, כך שהם ישמרו גם בין הפעלה להפעלה. הפעולה גם מפעילה שדות שצוינו לפי ההגדרות הקודמות.

public static void UpdateConfig()

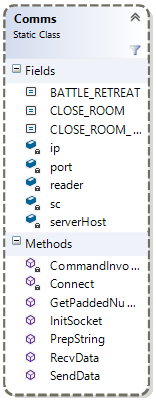
טענת כניסה: אין

טענת יציאה: הפעולה מעדכנת את הערכים של LocalSettings בהתאם לשינויים שהתבצעו בדף ההגדרות.

public static int GetIndex(Dictionary<string, Territory> dict, Territory val)

טענת כניסה: מילון הטריטוריות ושמותיהן, וטריטוריה ספציפית.

טענת יציאה: פעולה זו עוברת על תוכן המילון, ומחזירה את האינדקס במילון בו נמצאת הטריטוריה val.

**Comms.cs**

מחלקה זו היא מחלקה סטטית המכילה שדות ופעולות הקשורות לתקשורת של הלקוח עם השרת.

**שדות המחלקה:**

reader – עצם מטיפוס StreamReader המאפשר קריאה מחיבור התקשורת (הסוקט) של הלקוח עם השרת.

sc – עצם מטיפוס StreamSocket המייצג את חיבור התקשורת (הסוקט) של הלקוח עם השרת.

ip – משתנה המכיל את כתובת הIP של השרת כstring.

port – פורט שכבת התעבורה שדרכו תתבצע התקשורת עם השרת (על בסיס פרוטוקול TCP).

SIGN\_IN…. – אוסף הקבועים המתארים את כל סוגי ההודעות בפרוטוקול ואת ערכי הקודים התואמים (כולל ערכי פירוט ספציפיים).

serverHost – עצם מסוג HostName המייצג את כתובת הIP של השרת בצורה המאפשרת התחברות.

**פעולות המחלקה:**

public static void InitSocket()

טענת כניסה: אין

טענת יציאה: הפעולה בונה את עצם כתובת הIP ואת עצם הStreamSocket ומפעילה את פעולת ההתחברות.

private async static void Connect()

טענת כניסה: אין

טענת יציאה: פעולה זו מנסה לבצע חיבור לשרת. אם החיבור נכשל, הפעולה מציגה אפשרות להתחברות מחדש / יציאה מהאפליקציה.

private static void CommandInvokedHandler(IUICommand command)

טענת כניסה: האופציה שבה בחר המשתמש כאשר החיבור לשרת נכשל.

טענת יציאה: הפעולה יוצאת מהאפליקציה או מפעילה מחדש את פעולת Connect, לפי החלטתו של המשתמש.

public static void SendData(string message)

טענת כניסה: הודעה לשליחה לשרת.

טענת יציאה: הפעולה שולחת את ההודעה לשרת דרך הסוקט.

public static string RecvData(int size, int flags)

טענת כניסה: גודל ההודעה המצופה, "דגלים" המציינים את סוג קליטת ההודעה.

טענת יציאה: הפעולה קוראת מתוך הסוקט נתונים באורך שצוין. אם flags שווה ל0, תתבצע "נעילה" של הסוקט הקריאה כדי להבטיח שלא יהיו שיבושים בקריאה/שליחה של הודעות. אם לא, הקריאה תתבצע ללא נעילה, ומשתמשים במקרה שכזה בתקשורת בה לא ידוע מתי תתקבל ההודעה.

public static string GetPaddedNumber(int num, int size)

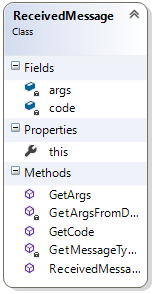
טענת כניסה: הפעולה מקבלת מספר וגודל מחרוזת.

טענת יציאה: הפעולה הופכת את המספר למחרוזת בגדל המצוין. אם כמות הספרות קטנה מהגודל הרצוי, המספר יוקדם באפסים (zero-padding).

public static string PrepString(string str)

טענת כניסה: מחרוזת לשליחה לשרת.

טענת יציאה: הפעולה מוסיפה לתחילתה של המחרוזת את גודלה בפורמט הפרוטוקול.

**ReceivedMessage.cs**

מחלקה זו מתארת הודעה אשר התקבלה מהשרת.

**שדות המחלקה:**

code – קוד/סוג ההודעה.

args – עצם מטיפוס List<string> המכיל את רשימת ערכי ההודעה.

**פעולות המחלקה:**

public string GetCode()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את קוד ההודעה.

public List<string> GetArgs()

טענת כניסה: אין

טענת יציאה: הפעולה מחזירה את רשימת ערכי ההודעה.

public string this[int index]

טענת כניסה: אינדקס של ערך מסוים מההודעה.

טענת יציאה: הפעולה מחזירה את הערל באינדקס שהתקבל. פעולה זו היא העמסה על האופרטור [].

private string GetMessageTypeCode(int flags)

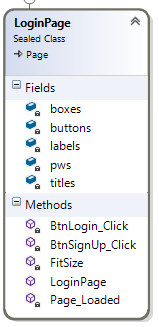
טענת כניסה: דגל סוג תקשורת שמועבר לפעולת Comms.

טענת יציאה: פעולה זו קולטת את קוד ההודעה הבאה שעומדת להתקבל.

private List<string> GetArgsFromData(int flags)

טענת כניסה: דגל סוג תקשורת שמועבר לפעולת Comms.

טענת יציאה: פעולה זו קולטת את ערכי ההודעה הנוכחית לפי קוד ההודעה.

**LoginPage.xaml.cs**

דף זה הוא הדף הראשון בפתיחת האפליקציה, שבו המשתמש נרשם למשחק.

**שדות הדף:**

titles – עצם מטיפוס List<TextBlock> המכיל את כל הכותרות בעמוד.

labels – עצם מטיפוס List<TextBlock> המכיל את כל קטעי הטקסט הרכילים בעמוד.

boxes – עצם מטיפוס List<TextBox> המכיל את כל קופסאות הזנת הנתונים בדף.

pws – עצם מטיפוס List<PasswordBox> המכיל את על קופסאות הזנת הסיסמאות בדף.

buttons – עצם מטיפוס List<Button> המכיל את כל הכפתורים בדף.

**פעולות הדף:**

private void Page\_Loaded(object sender, RoutedEventArgs e)

טענת כניסה: שולח הפעולה ומשתני האירוע.

טענת יציאה: הפעולה מפעילה את אתחול המחלקות Helper וComms ומפעילה את פעולת ההתאמה האדפטיבית.

private void FitSize(object sender, RoutedEventArgs e)

טענת כניסה: שולח הפעולה ומשתני האירוע.

טענת יציאה: הפעולה משנה את המיקום והגודל של כל הפקדים בדף כדי שיתאימו לגודל המסך המשתנה. הפעולה מופעלת כל פעם שמשתנה גודל המסך.

private void BtnLogin\_Click(object sender, RoutedEventArgs e)

טענת כניסה: שולח הפעולה ומשתני האירוע.

טענת יציאה: הפעולה מרכיבה את הודעת בקשה ההתחברות של המשתמש לפי הנתונים שהזין המשתמש, ומקבלת את תגובת השרת. אם ההתחברות הצליחה, המשתמש יועבר למסך הראשי. אם לא, תוצא הודעת שגיאה מתאימה.

private async void BtnSignUp\_Click(object sender, RoutedEventArgs e)

טענת כניסה: שולח הפעולה ומשתני האירוע.

טענת יציאה: הפעולה מרכיבה את הודעת בקשה ההרשמה של המשתמש לפי הנתונים שהזין המשתמש, ומקבלת את תגובת השרת. אם ההרשמה הצליחה, המשתמש יועבר למסך הראשי. אם לא, תוצא הודעת שגיאה מתאימה.