

**שם הפרויקט:** סולמות ונחשים

**שם המבצע:** בר ויצמן

**שם המנחה:** ענת בן משה

**שם בית הספר:** מקיף ג אשדוד

**שנת הגשה:** 2016

תוכן עניינים

מבוא........................................................2

מדריך למשתמש....................................3

מדריך למפתח........................................6

סיכום אישי..........................................12

מבוא

בכניסה למשחק יצטרכו המשתמשים להזין את שמותיהם ולהתחבר למשחק. המשחק יהיה בנוי בשרת לקוח , כאשר כל אחד מהמשתמשים מתחבר ממחשב אחר. המשחק בנוי ל2 משתמשים בלבד.

כאשר המשתמשים יזינו את שמותיהם וייכנסו למשחק, מולם ייפתח לוח המשחק, המכיל 100 משבצות הממוספרות מ-1 עד 100, ומטרת המשחק היא להגיע למשבצת ה100 והאחרונה ובכך לנצח . על לוח המשחק משובצים סולמות ונחשים . כאשר שחקן הגיע למשבצת בה יש סולם, יטפס השחקן בסולם – אך אם יגיע לנחש, ירד השחקן למשבצת אליה מוביל הנחש.

בתורו של כל משתמש, יצטרך המשתמש "לגלגל" את הקובייה שתהיה בצד המסך, אשר תגריל מספר בין 1-6. השחקן על לוח המשחק יתקדם קדימה בהתאם למספר שהוגרל על ידי הקוביות, וכן יוצג בצד המסך על צג "תורו של:" את שמו של השחקן שהגיע תורו לשחק. כך ימשיכו לשחק המשתמשים במשך זמן מסוים עד שאחד מהם יגיע למשבצת ה-100 והאחרונה וינצח .

מדריך למשתמש

**הפעלת השרת :**

על מנת לשחק במשחק נצטרך תחילה להפעיל את השרת באחד המחשבים עך ששני השחקנים יוכלו להתחבר אליו על פי כתובת ה IP וה Port שלו ולשחק אחד נגד השני. כל מה שנצטרך לעשות יהיה ללחוץ על קיצור הדרך שנקרא Server .

לאחר הלחיצה יופיע חלון בו נצטרך ללחוץ על "אפשר גישה" או "בטל חסימה" על מנת להפעיל את הסרבר ומיידית יופיע מסך הסרבר:



בחלון זה כתובה כתובת הIp וה Port אותם יצטרכו השחקנים לרשום בטופס ההתחברות.

בחלון זה יופיעו פרטי ההתחברות וההתנתקות. כששחקן יתחבר הסרבר יפלוט על החלון "ברוך הבא למשחק" וכששחקן יתנתק הסרבר יודיע על כך ויכתוב את שעת ההתנתקות המדויקת .

**הפעלת טפסי השחקנים והרצת המשחק:**



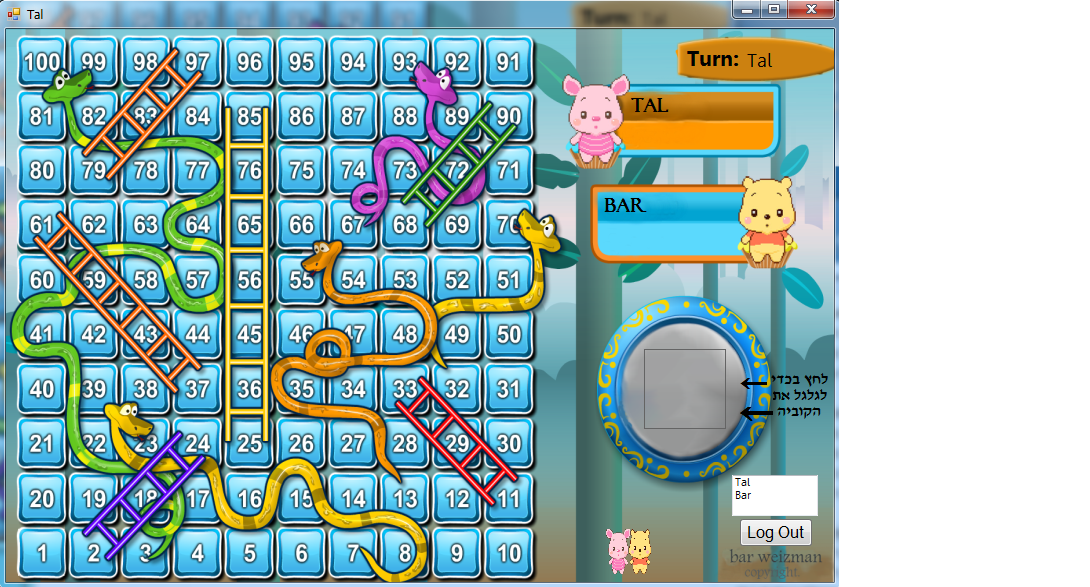
לאחר שווידאת כי השרת פועל, כל שחקן ילחץ על קיצור הדרך למשחק עצמו שנקרא SnakesAndLadders. מיד יופיע על גבי המסך טופס התפריט:



ברגע שנלחץ על start ייפתח טופס ההתחברות אליו נתבקש להזין את שם המשתמש שלנו, כתובת IP וה Port שהופיעו קודם לכן על טופס השרת ולאחר מכן ללחוץ על Connect .

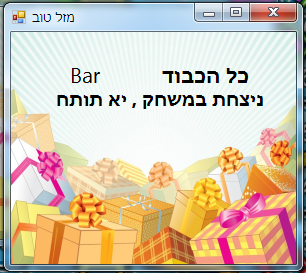
כעת ייפתח לנו טופס המשחק ונותר רק לחכות, אנו מחוברים לשרת וממתינים ששחקן נוסף יתחבר כדי שנוכל לשחק מולו.

טופס המשחק נראה כך לאחר ששני שחקנים התחברו , עכשיו נתחיל לשחק. בצד שמאל של המסך רשום תורו של איזה שחקן. כל שחקן בתורו יצטרך ללחוץ על הקוביה שתגריל מספר בין 1-6 ותקדם אותו על פני לוח המשחק:



המשחק מתחיל מכאן

נוכל להתנתק מהמשחק באמצעות לחיצה על כפתור ה Log Out שבתחתית המסך.



כאשר אחד מהשחקנים מגיע למשבצת ה100 , ייגמר המשחק ויופיע שם המנצח .

תהנו ובהצלחה!

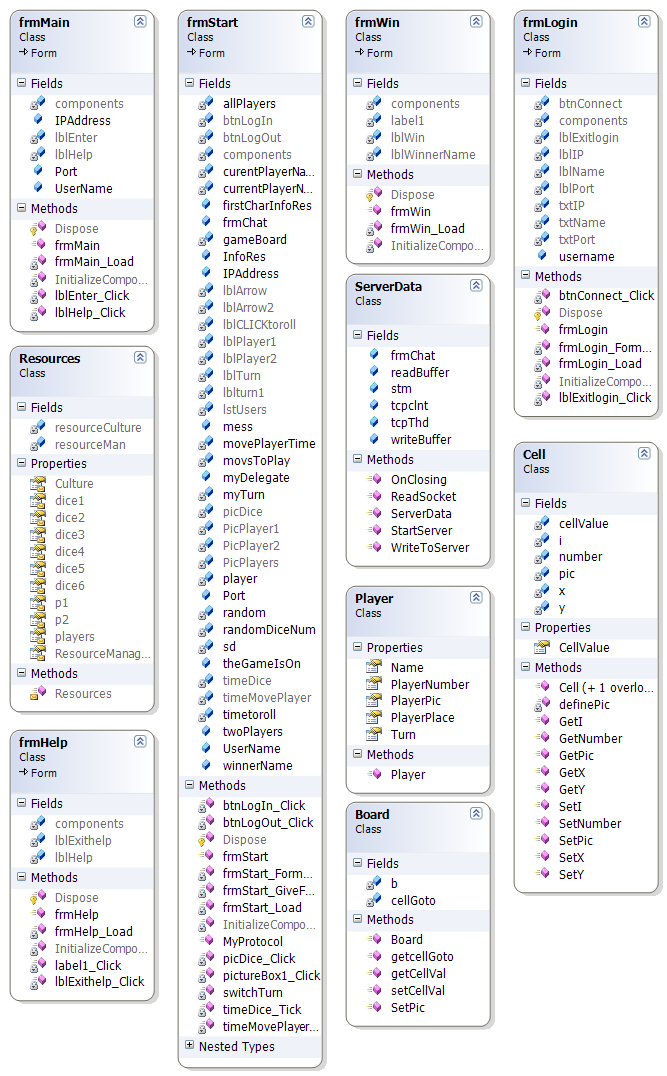
מדריך למפתח

**מבנה השרת-לקוח**

עיקר הפרויקט שלי מבוסס על המודל שרת-לקוח. מרבית ישומי האינטרנט המבוזרים בנויים על פי מודל זה ומורכבים משני חלקים: שרת ולקוח, המתקשרים זה עם זה באמצעות העברת הודעות כפי שנקבע על ידי הפרוטוקול(אוסף של חוקים המגדירים את אופן בקשת וקבלת הנתונים במערכת תקשורת מסוימת). תהליך לקוח הוא תהליך היוזם את הקשר. תהליך השרת מאזין באופן פסיבי ליצירת קשר. כאשר לקוח מעוניין לקבל שירות הוא שולח לשרת מתאים בקשה ליצירת קשר. השרת מקבל את הבקשה ומשיבה עליה וכך נוצר קשר בין השרת ללקוח. השרת מספק ללקוח מידע או שירות.

בדרך כלל, תוכנות השרת ולקוח רצות על גבי מחשבים שונים והתקשורת מתבצעת דרך רשת המחשבים. עם זאת, הן יכולות לפעול גם על גבי אותו מחשב.

**תרשימי UML**



**תיעוד הפונקציות ב- SERVER**

private void GetIpAdDress()

פונקציה ששומרת כתובת מחשב במשתנה IPADRESS

public void NewClientConnected()

הפונקציה מופעלת בתוך תהליך ומתבצעת באופו אוטומטי כאשר מתחבר לקוח חדש. הפונקציה מופעלת לכל לקוח פעם אחת בלבד. מטרתה היא לשמור בתוך טבלת DATAHOLDER את נתוני הלקוח.

public void ReadSocket()

התהליך של כל לקוח. הפונקציה מופעלת באופן אוטומטי כאשר לקוח מסויים שולח מידע .

private void CloseTheThread(long realId)

הפונקציה מקבלת את מספר הID של הלקוח שגרם לשגיאה במשחק וסוגרת את התהליך של הלקוח הספציפי הזה.

private void OnClosing()

הפונקציה מחסלת תהליכים קיימים וסוגרת את המאזין.

public void UpDateDataGrid(string displayString)

פונקציה שמעדכנת בטופס השרת את פעילות השחקנים לגבי התחברות והתנתקות משרת.

**תיעוד פונקציות ב-Snakes and Ladders**

**frmMain:**

private void lblEnter\_Click(object sender, EventArgs e)

פונקציה זו פותחת את טופס ההתחברות(Login) למשחק

private void lblHelp\_Click(object sender, EventArgs e)

פונקציה זו פותחת את טופס ההוראות של המשחק , שם מוסברים כללי וחוקי המשחק

**frmLogin:**

private void frmLogin\_Load(object sender, EventArgs e)

פונקציה זו מתבצעת כאשר נפתח הטופס. היא מציגה את נתוני ההתחברות הדרושים על גבי טופס ההתחברות למשחק ,מתרגמת ומעבירה את כתובת ה Ip אל הסרבר.

private void btnConnect\_Click(object sender, EventArgs e)

פונקציה זו יוצרת קשר בין השרת ללקוח, כלומר, שומרת את כל הנתונים שהוזנו ע"י המשתמש לטופס המשחק ומעבירה אותם אל הסרבר.

**frmStart:**

private void frmStart\_Load(object sender, EventArgs e)

פונקציה זו מתבצעת כאשר נפתח הטופס. תפקידה הוא בהגדרת המשתנים ומצביעים שונים שישמשו בהמשך למטרות שונות.

private void frmStart\_FormClosing(object sender, FormClosingEventArgs e)

פונקציה זו אחראית על סגירת הטופס

private void picDice\_Click(object sender, EventArgs e)

פונקציה זו אחראית על הפעלה ראשונית של קוביית המשחק (הגרלת מספר + הפעלת טיימר)

private void timeDice\_Tick(object sender, EventArgs e)

פונקציה זו מופעלת כאשר מופעל הטיימר של הקוביה. תפקידה הוא הפעלת קוביית המשחק על גבי הלוח (להחליף תמונות רנדומליות של הקוביה ולעצור כאשר הטיימר נגמר בתמונה שהוגרלה באמת).

private void timeMovePlayer\_Tick(object sender, EventArgs e)

פונקציה זו אחראית על תזוזה של השחקנים על גבי הלוח. היא מטפלת בתא שיוצאים ממנו ובתא שמגיעים אליו (עצירה , מעבר בין תא לתא, החלפת תמונה מתאימה).

private void switchTurn()

זוהי פונקציה שמחליפה תור כאשר השחקן הגיע אל התא המיועד. מופעלת כאשר קוראים לה לביצוע.

private void btnLogOut\_Click(object sender, EventArgs e)

פונקציה זו מנתקת את הלקוח מהשרת.

private void btnLogIn\_Click(object sender, EventArgs e)

פונקציה זו אחראית על התחברות השחקנים אל השרת אך אינה רלוונטית מאחר שהמשחק נפתח כבר לאחר שהם התחברו (אין אופציה להתחברות מחודשת).

public void MyProtocol()

פרוטוקול תקשורת בין הלקוח לבין השרת

**תיעוד מחלקות ב-Snakes and Ladders**

**Class Cell:**

public Cell(int i, int x, int y, PictureBox pic, int number)

מחלקה זו אחראית לבניית אובייקט מטיפוס תא שיכיל תכונות שונות כגון מיקום , תמונה וערך .

במחלקה זו פעולות Get וSet המשמשות אותנו במהלך כל הפרויקט.

**Class Board:**

public Board(frmStart frm)

מחלקה זו אחראית לבניית אובייקט מטיפוס לוח שיכיל מערך של תאים (הוא מכיל את המחלקה Cell) וכמו כן את מיקומי הסולמות והנחשים במשחק.

במחלקה זו פעולות Get וSet המשמשות אותנו במהלך כל הפרויקט.

**Class Player:**

public Player(string Name, int PlayerNum, bool Turn, Image PlayerPic, int PlayerPlace)

מחלקה זו אחראית לבניית אובייקט מטיפוס שחקן שיכיל תכונות שונות כגון שם , מספר סידורי של השחקן, תור, מיקום על הלוח ותמונה.

במחלקה זו פעולות Get וSet המשמשות אותנו במהלך כל הפרויקט.

**Class ServerData:**

מחלקה זו אחראית על יצירת השרת ויצירת ה Thread שמאזין דרכו ללקוחות. זהו קישור והעברת הנתונים מהלקוח אל השרת , כלומר אל הסרבר.

public void StartServer(string user, string ip, int port)

הפעולה מקבלת את פרטי ההתחברות ופותחת את תהליך ההתקשרות בין השרת למשתמש(לקוח)

public void WriteToServer(string strMessage)

הפעולה שולחת לשרת את ההודעה שמתקבלת

public void OnClosing()

הפעולה סוגרת את תהליך ההתקשרות ומאזינה כאשר לקוח מתנתק

public void ReadSocket()

הפעולה קוראת את המידע שנשלח אל הלקוח

סיכום אישי

הצדדים החלשים בפרויקט שלי הם אורך ויעילות הקוד ואילו החזקים יותר הם הגרפיקה והעיצוב של הפרויקט.

אם היה לי עוד זמן לעבוד על הפרויקט, הייתי מוסיפה אופציה של בחירת שחקן ע"י המשתמש וכן בחירת גודל לוח משחק. בחירת גודל לוח משחק יכולה להאריך את המשחק/לקצר אותו , ובחירת שחקן ע"י המשתמש יכולה להיות נחמדה מאוד.

למדתי מהפרויקט המון מבחינת ידע בגרפיקה : למדתי על עצמי שאני יכולה להתמודד עם גרפיקה לא פשוטה בכלל ואני מאוד גאה בעצמי על הגרפיקה במשחק. כמו כן מבחינת תכנות היה מעניין מאוד לעבוד עם מחלקות שונות מכיוון שלא התנסיתי בזה רבות בעבר מה שהיה קצת קשה אך מועיל בהחלט. ארגון העבודה בפרויקט שלי היה יעיל מכיוון שכל פעם עבדתי על חלק קטן ולאט לאט נבנה פרויקט שלם ונהניתי מאוד לאורך כל העבודה על הפרויקט .