שם: דניאל קגנוביץ
ת.ז. 211939293

Space invaders

Interfaces

Animation: ממשק זה מטרתו לייצג כיצד אנימציה צריכה להיות. עבור כל אנימציה יש את הפעולה של מה היא אמורה לעשות בכל רגע, ומתי היא נעצרת. האנימציה זה מה שהמשתמש רואה.

Collidable: הממשק זה מטרתו לייצג אובייקטים שניתן לפגוע בהם, ולכן לכל אובייקט כזה יהיה צריך להיות אובייקט שמתאר מה נפגע (המקרה שלנו זה מלבן) וגם מה קורה במקרה והאובייקט נפגע (hit)

FillableForBlock: הממשק זה מטרתו לייצג את האובייקטים שיכולים לצבוע את הblock ולכן יש להם פעולה של מילוי הblock.

HitListener: ממשק זה מטרתו לתאר מה קורה בעת פגיעה באובייקט ולכן יש לו פעולה של מה קורה כאשר נפגע שמקבל את האובייקט שנפגע (במקרה שלנו זה block) ומי שפגע בו (במקרה שלנו זה ball).

HitNotifier: ממשק זה מטרתו לתאר את האובייקטים שמכילים בהם רשימה של HitListener ולכן יש להם פעולה של הוספה של HitListener ומחיקה של HitListener.

LevelInformation: ממשק זה מטרתו לתאר את מה שאמור להיות בתוך שלב מסויים, ולכן יש לו את שם השלב, גודל הפאדל בו השחקן משתמש, מהירות הפאדל, הרקע של המשחק, ומספר החייזרים שצריך ופגוע בשביל לעבור את השלב.

Menu<T>: ממשק זה מטרתו לתאר כיצד תפריט משתמש אמור להיראות (במקרה שלנו במסך הפתיחה) נשים לב שממשק זה יורש מאנימציה וזאת כיוון שהמשתמש צריך לראות את התפריט. ניתן לראות שלתפריט יש אלמנטים של בחירה, קבלת האובייקט מהבחירה (T), והוספת תפריט משנה.

Shotable: ממשק זה מטרתו לתאר אובייקטים שיכולים לירות ולכן יש להם את הפקודה של לירות (shoot), במקרה שלנו הפאדל והחייזרים יכולים לירות ולכן יממשו ממשק זה.

Sprite: ממשק זה מטרתו לתאר אובייקטים שהמשתמש יכול ליראות על המסך, ולכן יש להן פעולה של ציור על המסך. כמו כן יש להם פעולה של מה הם יעשו אחרי dt שניות בהינן המשחק בו הם נמצאים.

Task<T>: ממשק זה מטרתו לתאר פעולה מסוג T והרצת פעולה זו, במקרה שלנו השתמשנו בTask כאשר רצינו להריץ את הAnimation החדש שאמור להתבצע כאשר לחצנו על אפשרות בתפריט הראשי או כאשר רצינו לצאת מהמשחק.

BlockCreator: ממשק זה מטרתו לתאר כיצד צריך לייצר Block (Block יפורט בהמשך).

Classes

Ass7Game: אחראי על יצירת המשחק, יצירת התפריט ובו יש את פעולת הmain. ומטרתה להפעיל את המשחק.

game:

GameLevel: מחלקה זו מייצגת ואחראית על הפעלת המשחק שאת התיאור שלה היא מקבלת בעת היצירה של המחלקה, כמו כן היא מממשת את Animation כיוון שהמשתמש צריך לראות כיצד השלב מתרחש. בנוסף היא מקבלת את מונה החיים של השחקן (Counter live) וגם את התוצאה של המשתמש (Counter scoreCounter). כמו כן היא מקבלת אובייקט מסוג AnimationRunner שיריץ את האנימציות השונות שיש בשלב (על האובייקט הזה יפורט בהמשך).

GameFlow: המחלקה הזאת מייצג ואחראית על מהלך המשחק והרצת האנימציות השונות מסביב, כמו התפריט, מסך התוצאות לאחר פסילה וקבלת השם במקרה והמשתמש השיג תוצאה טובה. כמו כן, היא גם אחראית על קובץ התוצאות של המשתמש במשחק. בנוסף יש לה פעולה של הרצת השלבים, במקר שלנו ירוצו אינסוף שלבים עד שהמשתמש יפסל במשחק ואז יבוצעו כל המעברים בין האנימציות השונות.

Indicators:

LevelIndicator: אחראי על הצגת השם של השלב במהלך המשחק.

LivesIndicator: אחראי על הצגת מספר החיים שנותרו למשתמש המשחק.

ScoreIndicator: אחראי על הצגת הניקוד שהמשתמש צבר עד כה.

Listeners:

המחלקות הבאות כולם מממשות את HitListener.

ScoreTrackingListener: אחראי עם הוספת/הורדת נקודות (במקרה שלנו רק להעלות נקודות) למשתמש במהלך המשחק כאשר הוא פוגע בחייזרים. ומטרתו היא להיות observer על הניקוד של השחקן.

PaddleHit: אחראי על מעקב ולהגיד האם הייתה פגיעה בפאדל. ומטרתה היא להיות observer על פגיעת החייזרים בפאדל.

InvaderRemover: אחראי על מעקב אחרי הפגיעות בחייזרים ומחיקת החייזר המתאים כאשר הוא נפגע. ומטרתו היא להיות observer על פגיעת המשתמש בחייזרים.

BallRemover: אחראי על מחיקת הכדור כאשר הכדור פוגע האובייקט מסויים. ומטרתו היא להיות observer על פגיעת הכדורים (היריות) בחפצים במשחק.

BlockRemover: אחראי על ספירת הבלוקים במשחק ומחיקתם כאשר מספר הפגיעות בהם הגיע ל-0. ומטרתו היא להיות observer על מחיקת שכבת ההגנה של המשתמש ומעקב אחרי מספר הblocks בשכבת ההגנה.

Objects:

AnimationRunner: אחראי על הרצת אנימציות במשחק. ומטרתו היא להריץ את האנימציות השונות שקיימיות במשחק (מתרחש בפעולה run).

Point: מייצג ואחראי על ייצוג נקודה במישור על ידי 2 ערכים (x,y).

Line: מייצג ואחראי על ייצוג ישר במישור על ידי 2 Point (p1,p2).

Rectangle: מייצג ואחראי על ייצוג מלבן במישור על ידי 4 Line או נקודת ימנית עליונה, אורך ורוחב.

Velocity: מייצג ואחראי על ייצוג מהירות בצורה של ווקטור (dx,dy).

Ball: מייצג ואחראי על ייצוג של כדור במישור על ידי נקודת מרכז, רדיוס וצבע. כמו כן ניתן להוסיף לו מהירות כך שינוע במרחב וניתן גם להוסיף תחום בו הכדור יזוז. ומטרתו לשמש הכדורים אותם יורים החייזרים והמשתמש.

Block: אחראי על ייצוג של בלוק במשחק, הוא מיוצג על ידי מלבן, רצף צבעים ונקודות פגיעה. מטרתו היא ברגע שיש פגיעה הוא מפעיל את הHitListeners שיש לו ובכך מוחק את שכבת ההגנה של המשתמש, או מודיע על פגיעה בחייזר. נשים לב שהוא יורש מ Collidable, Sprite, HitNotifier, כיוון שהוא מודיע על פגיעה, ניתן לראות אותו, וניתן לפגוע בו.

CollisionInfo: אחראי על ייצג מידע של פגיעה באובייקט. מטרתו היא לתת מידע על הפגיעה של הכדור באובייקט מסויים (אובייקט הפגיעה, ונקודת הפגיעה) וכך יוכל הכדור לשנות את מצבו בהתאם לפגיעה שהייתה.

Counter: אחראי על מניה של דברים. מטרתו היא למנות משתנים למיניהם כגון, חיים, ניקוד וכדומה.

FillBlockWithColor: מממש את FillableForBlock. אחראי על מילוי הבלוק בצבע. מטרתו למלא את הבלוק בצבע שהוגדר.

FillBlockWithImg: מממש את FillableForBlock. אחראי על מילוי הבלוק בתמונה. מטרתו למלא את הבלוק בתמונה שהוגדר.

GameEnvironment: אחראי על החזקת כל האובייקטים הניתנים לפגיעה (Collidable). מטרתו להוסיף, למחוק ולשמור אובייקטים שנמצאים במשחק שבהם הכדור יכול לפגוע.

SpriteCollection: אחראי על החזקת כל האובייקטים הניתנים לראות (Sprite). מטרתו להוסיף, למחוק ולשמור אובייקטים שנמצאים במשחק שאותם המשתמש יכול לראות, ולהודיע לאותם אובייקטים על כך שעבר dt שניות.

Invader: יורש מBlock ומממש את Shotable. אחראי על ייצוג של הפולש במשחק, על החזרת הפולש למקומו ההתחלתי בעת הצורך, ועל יריה של אותו הפולש. מטרתו להיות החייזר שתוקף את המשתמש במשחק, ונע בתנועה אופקית ואנכית לאורך המשחק.

InvaderColection: ממש את Sprite. אחראי על ייצוג של כל הפולשים שנמצאים במשחק. מטרתו להזיז את החייזרים ביחד במשחק (ניתן להגדיר מהירות), לירות יריות לכיוון השחקן ולהודיע לשחקן שנפסל כאשר הפולשים הגיעו לשכבת ההגנה.

Screen: אחראי על ייצוג המסך ועל הגבלת הגבולות שלו ברקע. מטרתו להגביל את גודל המסך כך שאובייקטים לא יחרגו ממנו ובנוסף גם לצייר רקע.

Paddle: ממש את Sprite, Collidable, HitNotifier, Shotable. אחראי על ייצוג המשתמש במשחק. מטרתו להיות המשתמש שמשחק, לירות כדורים על החייזרים, לנוע ביחס למקשי המקלדת של המשתמש ולהגיד מה לעשות ברגע שכדור פוגע בו.

ScoreInfo: ממש Serializable. אחראי על ייצוג של נתוני משתמש במשחק (שם וניקוד שצבר). מטרתו לשמור בזיכרון את נתוני אותו השחקן, ולייצג את הנתונים של אותו שחקן בטבלת הנקודות.

HighScoresTable: ממש Serializable. אחראי על ייצוג של השחקנים הטובים ביותר ששיחקו ולהציג את הנתונים שלהם, אחראי גם על הוספת משתמשים חדשים, בדיקה האם יש שחקן עם תוצאה טובה יותר משאר השחקנים בטבלה והחזרת הנתונים של המשתמשים (ScoreInfo). מטרתו לשמור ולטעון מזיכרון טבלה של השחקנים הטובים ביותר ששיחקו עד כה במשחק ובנוסף להוסיף שחקנים חדשים עם יש להם תוצאה טובה יותר משל אחרים בטבלה.

Operation<T>: אחראי על שמירת פעולה כלשהי שמיוצגת על ידי מפתח, שם הפעולה וערך הפעולה (T). מטרתו לשמור את נתוני הבחירה בתפריט בצורה נוחה ומסודרת.

ShowHiScoresTask: ממש את Task<Void>. אחראי על הרצה של אנימציית HighScoresAnimation (שיפורט בהמשך). מטרתו היא להריץ את האנימציה הזאת מתוך התפריט שבמקרה שלנו הוא מסוג Task<Void>.

BlocksFromSymbolsFactory: אחראי על ייצור בלוקים (וקבלת רווחים) בהינתן הסימן שלהם (זו גם המטרה שלו) זה הוא factory של בלוקים.

BlockTemplateCreator: מממש את BlockCreator. בהינתן הנתונים של הבלוקים המחלקה אחראית לייצר את אותו הבלוק (זה גם המטרה שלה).

Screens:

כל המחלקות הבאות ממשות את Animation.

KeyPressStoppableAnimation: אחראי על ייצוג אנימציה שאותה מקבל תוך כדי שברגע שנלחץ כפתור מסוים אם זה הכפתור שאותו קיבל כמפתח אז האנימציה תסתיים. מטרתו לגרום לכך שאנימציות יפסקו ברגע שיילחץ על כפתור.

GameOver: אחראי על הצגת סוף המשחק שם המשתמש נפסל. מטרתו להגיד למשתמש כי הוא נפסל ולהראות את התוצאה שלו.

PauseScreen: אחראי על תצוגה של מסך עצירה במהלך המשחק. מטרתו לעצור את המשחק עד שהמשתמש ימשיך את המשחק.

CountdownAnimation: אחראי על ספירה לאחור תוך כדאי כך שרואים את המסך של המשחק מאחוריו. מטרתו בתחילת כל משחק או לאחר פסילה, לספור לאחור ואז להמשיך את המשחק.

HighScoresAnimation: אחראי על ייצוג טבלת התוצאות (HighScoresTable) של המשחק. מטרתו להציג למשתמש את הטבלה של התוצאות הטובות ביותר ששחקנים עשו.

MenuAnimation<T>: ממש את Menu<T>. אחראי על ייצוג אנימציית התפריט עבור המשתמש. מטרתו להציג למשתמש את האפשרויות השונות שבתפריט.

EndScreen: אחראי על ייצוג תוצאת המשתמש (ניצח או הפסיד) וזו היא גם מטרתו.

readers:

LevelSpecificationReader: אחראי על קריאת השלב מהקובץ ולהחזיר את אותו השלב (זו היא גם מטרתו).

BlocksDefinitionReader: אחראי על קריאת בלוקים מהקובץ ולהחזיר את BlocksFromSymbolsFactory שאותו מייצג (זו היא גם מטרתו).

ColorsParser: אחראי על המרת מחרוזת (מהפורמט שהוגדר בקובץ) אל צבע (זו היא גם מטרתו).

ImageParser: אחראי על המרת מחרוזת (מהפורמט שהוגדר בקובץ) אל תמונה (זו היא גם מטרתו).

מימושים מיוחדים: (התשובות לשאלות):

The Aliens Formation:

יצרתי מחלקה בשם Invader שמייצג את הפולש במשחק שבתוכו יש Block (משתנה) שמייצג את הבלוק של אותו פולש על המסך. המחלקה יורשת מBlock כיוון שנוח לייצג אותו בצורה שכזאת.

לאחר מכן יצרתי את המחלקה של InvaderColection, שמחזיקה את כל הפולשים במבנה נתונים מסוג רשימה של מחסניות של פולשים (בשביל הירייה של הפולשים שתוסבר בהמשך). את הפולשים ואת הInvaderColection יצרתי בGameLevel על ידי לולאה מקוננות. את התנועה של כולם יחד עשיתי כאשר הגדרתי כי לInvaderColection ניתן שיהיה מהירות ובכך הוזזתי את כולם יחד, וכאשר הימיני ביותר או השמאלי ביותר יצאו מהמסך, שיניתי את המהירות לכיוון השני והגדלתי אותה ב10 אחוז, והורדתי את כל הפולשים למטה במסך במספר פיקסלים שהוגדר להם.

The Shields:

על מנת ליצור את החומות עשיתי לולאות מקוננות שייצרו לי את הבלוקים של ההגנה במקום המתאים, כאשר שבכל יצירת בלוק הוספתי לו listener של listener BallRemover של BlockRemover והוספתי לCounter של הבלוקים 1 שמונה את מספר הבלוקים שקיימים בshield וכאשר הורסים בלוק כזה אז הCounter קטן ב-1 ולכן ניתן לעקוב אחרי מצב הshield.

Shots by Aliens:

לInvaderColection הוספתי משתנה של רגע הירייה האחרונה ופעולה שבודקת האם הוא יכול לירות. באותה פעולה אם הזמן בין הירייה הקודמת לעכשיו גדול או שווה מהזמן שנאמר בתרגיל יוחזר אמת אחרת יוחזר שקר. ולכן אם יש אישור לירייה אז בניתי

פונקצייה שתחזיר לי רשימה של כל הראשונים שנמצאים עכשיו (כיוון שאלה מחסניות אז יוחזר הפולשים שבראש המחסניות) ואז הפעלתי Random שיקח מספר בין 0 לגודל אותה רשימה שקיבלתי פחות 1. ואז אותו הפולש שהוגרל יפעיל את פונקציית הירייה שלו שיוצרת כדור עם מהירות כלפי מטה, ואז עדכנתי את זמן הירייה האחרון.

Shots by Player:

הוספתי לפאדל משתנה של רגע הירייה האחרונה ופעולה שבודקת האם הוא יכול לירות. באותה פעולה אם הזמן בין הירייה הקודמת לעכשיו גדול או שווה מהזמן שנאמר בתרגיל יוחזר אמת אחרת יוחזר שקר. ולכן אם המשתמש לחץ על מקש רווח וגם יש אישור לירייה אז הפאדל יירה כדור בצורה אנכית כלפי מעלה (הפונקצייה יוצרת כדור), ואז עדכנתי את זמן הירייה האחרון.