Házi feladat dokumentáció

Felhasználói kézikönyv

A program elindítása után a felhasználó előtt megjelenik a játék menüje, melyben három menüpont közül lehet választani: Play, Leaderboard, Quit. A Play menüpontot választva a felhasználó elindítja a játékot, miután megadott egy felhasználónevet. A Leaderboard–ot választva a felhasználó megtekintheti azon játékosok eredményeinek listáját, akik már játszottak a játékkal. A Quit pontot választva a játék bezárásra kerül.

A játék során a cél, hogy a játékos a Sárga Pac-Man figurával felszedje az összes a pályán található fehér pontot. Ezt a pályán mászkáló 4 szellem nehezíti meg. Ha a játékost megérinti egy szellem akkor a játékos 3 életéből egy levonásra kerül. Amennyiben a játékos elveszíti mind a három életét, mielőtt sikerülne összegyűjteni a fehér pontokat, akkor a játékos vesztett, a próbálkozása nem kerül elmentésre a dicsőséglistába, valamint a menübe való visszalépés után újra próbálkozhat. Amennyiben a játékos összeszedi az összes pontot a pályán, akkor nyert. Ilyenkor a játék előtt megadott felhasználónévvel felkerül a dicsőséglistára, majd visszakerül a főmenübe.

A program osztálydiagrammja

mellékelve a .zip file-ban

A programban megvalósított metódusok leírásai

Container.java:

> public void tick(): Ha a gameState enumeráció Win vagy Dead értéket vesz fel akkor a felhasználó által megadott nevet és a játék alatt elért pontszámot belehelyezi az osztály TreeMap típusú változójába.
>
> public static Map<Integer, String> getMap(): Visszaadja az osztály treemap nevű változóját.

Game.java:

> public synchronized void start(): Ez a függvény elindítja a game threadet és beállítja a játék elindításához szükséges boolean változót.
>
> public synchronized void stop(): Megállítja a game thread-et.
>
> public void run(): Ebben a metódusban van a game loop melynek segítségével másodpercenként 60-szor frissül a játék logika és rajzolódik ki a megfelelő kép a ablakba a szintén ebben az osztályban lévő tick() és render() metódusok segítségével.
>
> private void tick(): Ez a metódus amelyik összefoglalja a játék működéséhez szükséges többi osztály tick() metódusát.

private void render(): Ez a metódus foglalja össze a többi osztály render() metódusait, melyek rajzolnak valamit a megjelenített ablakba.

public static void main(String[] args): Ez a main metódus, amely létrehoz egy példányt a Game osztályból.

GameObject.java:

public void setX(): Beállítja az objektum x koordinátáját.

public void setY(): Beállítja az objektum y koordinátáját.

public int getX(): Visszaadja az adott objektum x koordinátáját.

public int getY(): Visszaadja az adott objektum y koordinátáját.

public void setID(GameObjectID id): Beállítja az objektum GameObjectID típusú változóját.

public GameObjectID getID(): Visszaadja az objektum GameObjectID típusú változóját.

public void setSpeedX(int speedX): Beállítja az objektum x irányú sebességét.

public void setSpeedY(int speedY): Beállítja az objektum y irányú sebességét.

public int getSpeedX(): Visszaadja az objektum x irányú sebességét.

public int getSpeedY(): Visszaadja az objektum y irányú sebességét.

GameObjectID.java: Ez egy enumeráció mely az egyes előforduló objektumok típusát tartalmazza. Az értéke lehet PacMan vagy Ghost.

Ghost.java:

public void tick(): A szellem ezen metódus segítségével mozog a pályán.

public void render(Graphics g): Ez a metódus az szellemet megtestesítő gif-et tölti be és rajzolja ki a játéktérre.

public Rectangle getBounds(): Visszaad egy téglalapot, mely a szellem pillanatnyi helyében a hitbox-át reprezentálja.

Handler.java:

public void tick(): Ez a metódus hajtja végig minden a pályán lévő GameObject-nek a tick() metódusát.

public void render(Graphics g): Ez a metódus hajtja végig minden a pályán lévő GameObject-nek a render() metódusát.

public void clearHandler(): A handler láncolt listájában lévő objektumokat kitörli.

public void addGameObject(GameObject obj): Hozzáad egy GameObject-et a handler lácolt listájához.

public void removeGameObject(GameObject obj): Eltávolít egy GameObject-et a handler lácolt listájából.

KeyInput.java:

public static void setPlayerName(String s): Beállítja a playerName nevű változó értékét a megadott Stringre.

public void keyPressed(KeyEvent e): Ez a metódus az egyes gombok lenyomásának érzékelését kezeli. Az egyes gombok lenyomásait a játék gameState változójától függően kezeli.

public String getPlayerName(): Visszaadja a playerName változó értékét.

PacMan.java:

public Rectangle getBounds(): Visszaad egy téglalapot, mely a PacMan pillanatnyi helyében a hitbox-át reprezentálja.

private void collision(): Ez a metódus azt vizsgálja, hogy a PacMan egy adott időpillanatban ütközött-e valamilyen a Handler osztály láncolt listájában lévő objektummal, azaz ütközött-e szellemmel.

public void tick(): Ez a metódus vizsgálja, hogy az adott időpillanatban a PacMan ütköztt-e fallal, coinnal, vagy szellemmel.

public void render(Graphics g): Ez a metódus tölti be és rajzolja ki a PacMan egy es irányaihoz tartozó gif fiájlokat.

STATE.java: EZ egy enumeráció, melyben a játék különböző állapotai vannak eltárolva. A következtő étékeket veheti fel: Menu, GiveName, Game, Leaderboard, Dead, Win.

HUD.java:

public void tick(): Ez a metódus figyeli a már összegyűjtött coinokat, valamint, hogy a játékos felszedte-e már az összes coin-t.

public void render(Graphics g): Ez a metódus rajzolja ki az ablakba a nyert, vesztett felíratokat, valamint ez írja ki a képernyőre a játékos aktuális pontszámát és életét.

public static void setLives(int i): Beállítja az életek számát a megadott értékre.

public static void setScore(int i): Beállítja a megszerzett pontokat a megadott értékre.

public int getScore(): Visszaadja a megszerzett pontokat.

Maze.java:

public void render(Graphics g): Ez a metódus rajzolja ki a pályát és a rajta lévő pontokat az ablkaba.

public void setScreenMaze(): Visszaállítja alaphelyzetbe a pálya változóit.

public int getFieldSize(): Visszaadja a pályát felépítő mezők méretét.

public int getfieldNumberHor(): Visszaadja, hogy vízszintesen hány mezőből áll a pálya.

public int getfieldNumberVer(): Visszaadja, hogy vertikálisan hány mezőből áll a pálya.

public int[][] getScreenMaze(): Visszaadja a pályát leíró két dimenziós tömböt.

public int getScreenMazeElement(int x, int y): Visszaadja a pálya adott koordinátájához tartozó mező értékét.

public void setScreenMazeElement(int x, int y, int element): Beállítja a pálya adott koordinátájához tartozó mező értékét a megadott értékre.

public List<Rectangle> getRectWall(): Visszaadja a falakat alkotó téglalapok listáját.

public List<Rectangle> getRectCoin(): Visszaadja a coinokat tartalmazó téglalapok listáját.

Menu.java:

public void setPlayerName(String s): Beállítja az osztály playerName nevű változóját a megadott Stringre.

public String getPlayerName(): Visszaadja az osztály playerName nevű változóját.

public void tick(): Ha a játék gameState változoja Menu értéket vesz fel, akkor a Container nevű osztály TreeMap-jét egyenlővé tesz a saját TreeMap-jével.

public void render(Graphics g): Kirajzolja az ablakba a játék gameState változója alapján a megfelelő menüt vagy almenüt.

private boolean mouseOver(int mx, int my, int x, int y, int width, int height): Visszatér egy igaz vagy egy hamis értékkel annak függvényében, hogy az egér pozíciója a megadott koordinátáknak megfelelő téglalapon belül helyezkedik-e el.

public void mousePressed(MouseEvent e): Az egér kattintását kezeli a játék egy adott állapotában.

public void load(): A leaderboard.txt fájlból betölti a Container nevű osztály TreeMap-jába az adatokat.

public void save(): Elmenti a leaderboard.txt fájlba az aktuális leaderboard adatait.

PacManWindow.java: Ebben az osztályban hozom létre a JFrame-et, amely a program alapját képezi.