Pacman

Weisz Erik
Version v1.0
2019.12.01.

Bevezető

A program célja a pacman játékot utánozni. Azért így fogalmaztam, mivel több dologban is különbözik: a szellemeket nem lehet megenni, valamint a szörnyek máshogy mozognak. Ezeket leszámítva viszont egészen hasonlóra sikeredett.

A program elkészítéséhez egy raylib nevű grafikus, valamint játékkészítő könyvtárat használtam. Erről több információ található itt: https://www.raylib.com/
A játék fordításához természetesen szükséges a könyvtár, és a különböző programozó környezetekben való fordításához szükséges információ itt található: https://github.com/raysan5/raylib/wiki (a jobb oldalt található listában több környezethez is található segítség, valamint makefile-al való fordításról is szó van). Én személy szerint VSCode-al használtam.

A játékos, valamint a szellemek mozgását, illetve a pontok elhelyezését fekete-fehér pálya képek használatával készítettem el.

A hátteret, valamint az előbb említett pályák innen származnak: https://github.com/TechnoVisual/Pygame-Zero/tree/master/pacman1/images

Struktúra Index

Struktúra Lista

Struktúrák:

Fájl Index

Fájl Lista

Enemies Struktúra

`#include <enemies.h>`

Public Attributes

- **Enemy blue**
- **Enemy red**
- **Enemy orange**
- **Enemy pink**

Tagok leírása

Enemy Enemies::blue

kék szellem

Enemy Enemies::orange

narancs szellem

Enemy Enemies::pink

rózsaszín szellem

Enemy Enemies::red

vörös szellem

EnemiesPng Struktúra

```
#include <main.h>
```

Tagok

- Texture2D **redLeft**
- Texture2D **redRight**
- Texture2D **redUp**
- Texture2D **redDown**

- Texture2D **blueLeft**
- Texture2D **blueRight**
- Texture2D **blueUp**
- Texture2D **blueDown**

- Texture2D **orangeLeft**
- Texture2D **orangeRight**
- Texture2D **orangeUp**
- Texture2D **orangeDown**

- Texture2D **pinkLeft**
- Texture2D **pinkRight**
- Texture2D **pinkUp**
- Texture2D **pinkDown**

Enemy Struktúra

```
#include <enemies.h>
```

Tagok

- **Player plr**
- **PossibleDir pDir**

Tagok leírása

PossibleDir Enemy::pDir

- a szellem lehetséges irányai

Player Enemy::plr

- a szellem hitboxát, valamint pozícióját tároló struktúra

Eredmeny Struktúra

```
#include <main.h>
```

Tagok

- char * **nev**
- int **pont**

Tagok leírása

char* Eredmeny::nev

- az eredményt elért játékos neve

int Eredmeny::pont

- az eredményt elért játékos pontszáma

Player Struktúra

```
#include <main.h>
```

Tagok

- Rectangle **rec**
- **Dir dir**
- **Dir nextDir**
- **Dir lastDir**

Member Data Documentation

Dir Player::dir

- a játékos vagy szellem jelenlegi iránya

Dir Player::lastDir

- a játékos vagy szellem előző iránya – akkor használom, ha a játékosnak nincs jelenlegi iránya, ilyenkor ez alapján tudom, hogy merre kell néznie

Dir Player::nextDir

- a játékos következő iránya, ha még éppen nincs elágazásnál

Rectangle Player::rec

- a játékos hitboxa egy téglalap struktúraként – erről továábi információ megtalálható a raylib dokumentációjában

---

Pont Struktúra

```
#include <main.h>
```

Tagok

- Rectangle **rect**
- bool **van**

Tagok leírása

Rectangle Pont::rect

- egy pont hitboxa és pozíciója egy téglalap struktúraként

bool Pont::van

- a pont létezik-e még (van-e)

PossibleDir Struktúra

`#include <enemies.h>`

Tagok

- bool **l**
- bool **r**
- bool **u**
- bool **d**

Tagok leírása

bool PossibleDir::d

- a játékos vagy szellem mehet-e lefelé

bool PossibleDir::l

- a játékos vagy szellem mehet-e balra

bool PossibleDir::r

- a játékos vagy szellem mehet-e jobbra

bool PossibleDir::u

- a játékos vagy szellem mehet-e felfelé

Tavolsag Struktúra

```
#include <enemies.h>
```

Public Attributes

- int **up**
- int **down**
- int **left**
- int **right**

Member Data Documentation

int Tavolsag::down

- a szellem alatt lévő pozíció távolsága a játékostól

int Tavolsag::left

- a szellemtől balra lévő pozíció távolsága a játékostól

int Tavolsag::right

- a szellemtől jobbra levő pozíció távolsága a játékostól

int Tavolsag::up

- a szellem felett lévő pozíció távolsága a játékostól

Fájlok Dokumentációja

enemies.c Fájl

```
#include "raylib.h"
#include "main.h"
#include "player.h"
#include "enemies.h"
#include "pont.h"
#include <math.h>
#include <stdio.h>
#include "debugmalloc.h"
```

Függvények

- void **getPossibleDir** (**Enemy** *emy, Color *img)
- int **tavSzamit** (int kezdetX, int kezdetY, int celX, int celY)
- **Dir tavMin** (**PossibleDir** pDir, **Tavolsag** tav)
- void **utvalaszt** (**Enemy** *emy, int celX, int celY, Color *img)
- void **checkEnemyMove** (**Enemy** *emy, Color *img)
- void **moveEnemy** (**Enemy** *emy)
- void **handleRed** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- void **handlePink** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- void **handleBlue** (**Enemy** *emy, **Enemy** *red, **Player** *plr, Color *img, bool *init)
- void **handleOrange** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- bool **checkDeath** (**Enemies** *emys, **Player** *plr)

Függvények leírása

- A piros szellemet leszámítva minden szellem először manuálisan jut ki a pálya közepén találhatü szellemdobozból

bool checkDeath (Enemies * *emys*, Player * *plr*)

- a játékos halálát ellenőrzi azáltal, hogy megnézi, összeért-e a hitboxa a négy szellem valamelyikével

void checkEnemyMove (Enemy * *emy*, Color * *img*)

- ellenőrzi, hogy a szellemek tervezett mozgása lehetséges-e (vagyis, hogy van-e előttük fal)

void getPossibleDir (Enemy * *emy*, Color * *img*)

- a szellemk lehetséges irányait ellenőrzi, valamint beállítja a PossibleDir struktúrájukban

void handleBlue (Enemy * *emy*, Enemy * *red*, Player * *plr*, Color * *img*, bool * *init*)

- a kék szellem kezelése
- mozgása: a piros szellemtől húz egy vonalat a játékos előtt lévő ponthoz, ezután ezt a vektort megszorozza kettővel, és ehhez ponthoz tart, ezáltal ahogy a piros szellem egyre közelebb jut a játékoshoz, a kék is

void handleOrange (Enemy * *emy*, Player * *plr*, Color * *img*, bool * *init*)

- a narancssárga szellem kezelése
- mozgása: attól föggően, hogy közel, vagy messze van a játékostól
- ha közel: menekül vissza a pálya közepére
- ha messze: a játékos fele tart

void handlePink (Enemy * *emy*, Player * *plr*, Color * *img*, bool * *init*)

- a rózsaszín szellem kezelése
- mozgása: a játékos előtti ponthoz tart, ezáltal próbálja váratlanul letámadni

void handleRed (Enemy * *emy*, Player * *plr*, Color * *img*, bool * *init*)

- a piros szellem kezelése
- mozgása: egyszerűen csak a játékoshoz tart

void moveEnemy (Enemy * *emy*)

- a szellemek mozgatásához használt függvény

Dir tavMin (PossibleDir *pDir*, Tavolsag *tav*)

- a szellemek körül levő pontoktól a játékoshoz megnézi a távolságot, és a visszaadja a legrövidebbnek az irányát

int tavSzamit (int *kezdetX*, int *kezdetY*, int *celX*, int *celY*)

- két pont közötti távolságot számít ki

void utvalaszt (Enemy * *emy*, int *celX*, int *celY*, Color * *img*)

- egy szellemet próbál eljuttatni a megadott pozícióhoz

enemies.h Fájl

```
#include "main.h"
```

Struktúrák

- struct **PossibleDir**
- struct **Tavolsag**
- struct **Enemy**
- struct **Enemies**

Typedefek

- typedef struct **PossibleDir PossibleDir**
- typedef struct **Tavolsag Tavolsag**
- typedef struct **Enemy Enemy**
- typedef struct **Enemies Enemies**

Függvények

- void **handleRed** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- void **handlePink** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- void **handleBlue** (**Enemy** *emy, **Enemy** *red, **Player** *plr, Color *img, bool *init)
- void **handleOrange** (**Enemy** *emy, **Player** *plr, Color *img, bool *init)
- bool **checkDeath** (**Enemies** *emys, **Player** *plr)

Typedef Documentation

typedef struct Enemies Enemies

- a szellemek összességét tároló struktúra – elemei Enemy struktúrák

typedef struct Enemy Enemy

- egy szellem adatait tároló adatszerkezet – elemei egy PossibleDir és egy Player struktúra

typedef struct PossibleDir PossibleDir

- egy szellem lehetséges irányait tároló adatszerkezet

typedef struct Tavolsag Tavolsag

- egy szellem körüli pontok távolsága a játékoshoz

---

main.c Fájl

```
#include "raylib.h"
#include "pont.h"
#include "menu.h"
#include "render.h"
#include "main.h"
#include "player.h"
#include "enemies.h"
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "debugmalloc.h"
```

Függvények

- int **main** ()

---

main.h Fájl

Struktúrák

- struct **Pont**
- struct **Player**
- struct **EnemiesPng**
- struct **Eredmeny**

Typedefek

- typedef struct **Pont Pont**
- typedef struct **Player Player**
- typedef struct **EnemiesPng EnemiesPng**
- typedef struct **Eredmeny Eredmeny**

Enum-ok

- enum **GameState** { **MENU**, **GAME**, **SCOREBOARD** }
- enum **MenuState** { **IDLE**, **PLAY**, **EXIT** }
- enum **GameEnd** { **WIN**, **LOSE** }
- enum **Dir** { **UP**, **RIGHT**, **DOWN**, **LEFT**, **NONE** }

Változók

- enum **GameState gamestate**
- enum **MenuState menustate**

Typedef Documentation

typedef struct EnemiesPng EnemiesPng

- a szellemek textúráit tároló adatszerkezet

typedef struct Eredmeny Eredmeny

- az előző játékok eredményeit tároló adatszerkezet

typedef struct Player Player

- a játékos/szellem hitboxát, valamint pozícióját, irányát tároló adatszerkezet

typedef struct Pont Pont

- egy pont hitboxát, koordinátáit, valamint azt, hogy ott van-e még,s tároló adatszerkezet

---

Enum-ok dokumentációja

enum Dir

Lehetséges állapotok:

UP	
RIGHT	
DOWN	
LEFT	
NONE	

enum GameEnd

- a játék végének verzióját meghatározó változó

Lehetséges állapotok:

WIN	
LOSE	

enum GameState

- a játék állapotát kezelő változó

Lehetséges állapotok:

MENU	
GAME	
SCOREBOARD	

enum MenuState

- a kezdeti menü állapotát kezelő változó

Lehetséges állapotok:

IDLE	
PLAY	
EXIT	

---

menu.c File Reference

```
#include "raylib.h"
#include "render.h"
#include "main.h"
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "debugmalloc.h"
```

Függvények

- bool **HandleMenu** (int **menustate**, bool *pDebug)
- void **listabaBeilleszt** (**Eredmeny** eredmeny[], int score, char *nev)
- void **handleScoreboard** (enum **GameEnd** gameend, int score, Texture2D *bg_used, **Eredmeny** eredmeny[])

Function Documentation

bool HandleMenu (int *menustate*, bool * *pDebug*)

- A menüt kezelő függvény – kezeli a gombokat

void handleScoreboard (enum GameEnd *gameend*, int *score*, Texture2D * *bg_used*, Eredmeny *eredmeny*[])

- a játék végén található scoreboard kezelése – kezeli a kiírást, valamint a név megadását, és a fájlba írást is

void listabaBeilleszt (Eredmeny *eredmeny*[], int *score*, char * *nev*)

- az új adat az eredmenyek listába való beillesztéséhez van

menu.h Fájl

```
#include <stdio.h>
```

Függvények

- bool **HandleMenu** (int **menustate**, bool *debug)
- void **handleScoreboard** (enum **GameEnd** gameend, int score, Texture2D *bg_used, **Eredmeny** eredmeny[])

Függvények leírása

bool HandleMenu (int *menustate*, bool * *debug*)

- a menu kezelése

void handleScoreboard (enum GameEnd *gameend*, int *score*, Texture2D * *bg_used*, Eredmeny *eredmeny*[])

- a játék végén található eredmények panel kezelése, valamint a végső eredmény fájlba írása

player.c Fájl

```
#include "raylib.h"
#include "main.h"
#include "pont.h"
#include "render.h"
#include <stdlib.h>
#include "debugmalloc.h"
```

Függvények

- void **checkInput** (**Player** *plr, Color *img)
- void **updateDir** (**Player** *plr, Color *img)
- void **checkMove** (**Player** *plr, Color *img)
- void **move** (**Player** *plr)
- float **getRotation** (**Player** *plr)
- void **handlePlayer** (**Player** *plr, Color *img, Texture2D *pacman)

Függvények leírása

void checkInput (Player * *plr*, Color * *img*)

- a nyílbillentyűk megnyomásának ellenőrzése, és ezáltal a játékoz következő irányának megadása

void checkMove (Player * *plr*, Color * *img*)

- azt ellenőrzi, hogy a játékos előtt fal van-e

float getRotation (Player * *plr*)

- a pac-man figura forgatásához ad vissza egy float-ot a játékos irányából

void handlePlayer (Player * *plr*, Color * *img*, Texture2D * *pacman*)

- a játékos kezelése – a többi kezelő függvényt egyesíti

void move (Player * *plr*)

- a játékos mozgatása

void updateDir (Player * *plr*, Color * *img*)

- a játékos irányának frissítése a következő irány alapján

player.h Fájl

```
#include "main.h"
```

Függvények

- void **handlePlayer** (**Player** *plr, Color *img, Texture2D *pacman)
- float **getRotation** (**Player** *plr)
- void **move** (**Player** *plr)

---

pont.c Fájl

```
#include "raylib.h"
#include "main.h"
#include <stdlib.h>
#include "debugmalloc.h"
```

Függvények

- bool **colorCmp** (Color clr1, Color clr2)
- bool **checkPont** (int x, int y, Color *img)
- void **initPont** (**Pont** ***adatok)
- void **handlePoints** (**Player** *plr, **Pont** **pontok, int *score)

Függvények leírása

bool checkPont (int *x*, int *y*, Color * *img*)

- megnézi, hogy egy megadott pont fekete-e vagy fehér a Color* img fájlban (ami színek listája)

bool colorCmp (Color *clr1*, Color *clr2*)

- megnézi, hogy két szín ugyanaz-e

void handlePoints (Player * *plr*, Pont ** *pontok*, int * *score*)

- a pontokat kezeli, miután inicializálva lettek – nézi, hogy a játékos megette-e őket

void initPont (Pont *** *adatok*)

- a pontok inicializálása két-dimenziós listába

pont.h Fájl

```
#include "main.h"
```

Függvények

- bool **colorCmp** (Color clr1, Color clr2)
- bool **checkPont** (int x, int y, Color img[])
- void **initPont** (**Pont** ***pontok)
- void **handlePoints** (**Player** *plr, **Pont** **pontok, int *score)

---

render.c Fájl

```
#include "raylib.h"
#include "main.h"
#include "player.h"
#include "pont.h"
#include "enemies.h"
#include "debugmalloc.h"
```

Függvények

- void **renderBackground** (int **gamestate**, Texture2D *bg)
- void **drawMenu** (int **menustate**, bool debug)
- void **drawPont** (**Pont** **pontok)
- void **getPlusXY** (**Player** *plr, bool *plusX, bool *plusY)
- void **drawPacman** (**Player** *plr, Texture2D *pacman)
- void **drawEnemies** (**Enemies** *emys, **EnemiesPng** *enemyTextures)

Függvények leírása

void drawEnemies (Enemies * *emys*, EnemiesPng * *enemyTextures*)

- a szellemeket rajzolja ki az egyesített Enemies listából

void drawMenu (int *menustate*, bool *debug*)

- a menüt rajzolja ki a játék elején

void drawPacman (Player * *plr*, Texture2D * *pacman*)

- a játékost jeleníti meg

void drawPont (Pont ** *pontok*)

- a pontok sokaságát jeleníti meg

void getPlusXY (Player * *plr*, bool * *plusX*, bool * *plusY*)

- a játékos forgatásához szükséges függvény – azért kell, mert a beépített forgató függvény egy képet a bal felső sarka körül forgat, és ez a függvény adja meg az ahhoz szükséges eltolást, hogy a kép középen legyen

void renderBackground (int *gamestate*, Texture2D * *bg*)

- a háttér kirajzolása – attól függően, hogy menüben vagyunk-e, sötétítve is van

render.h Fájl

```
#include "main.h"
#include "enemies.h"
```

Függvények

- void **renderBackground** (int **gamestate**, Texture2D *bg)
- void **drawMenu** (int **menustate**, bool debug)
- void **drawPont** (**Pont** **pontok)
- void **drawPacman** (**Player** *plr, Texture2D *pacman)
- float **getRotation** (**Player** *plr)
- void **getPlusXY** (**Player** *plr, bool *plusX, bool *plusY)
- void **drawEnemies** (**Enemies** *emys, **EnemiesPng** *enemyTextures)
