Pacman

Készítette Doxygen 1.8.20

1. fejezet

Névtérmutató

1.1. Csomagok

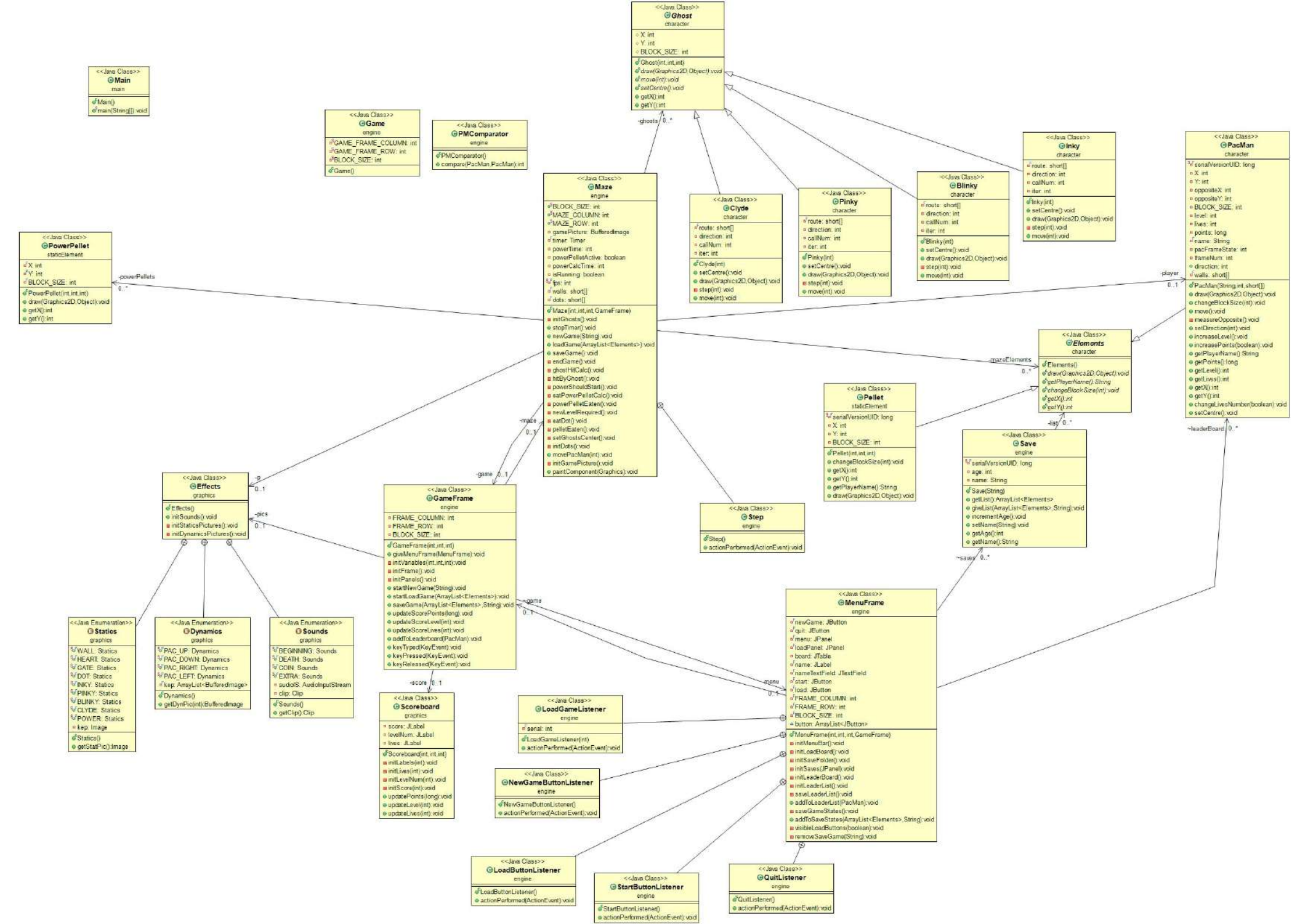
A csomagok rövid leírásai (ha léteznek):

2. fejezet

Hierarchikus mutató

2.1. Osztályhierarchia

Majdnem (de nem teljesen) betűrendbe szedett leszármazási lista:


<<Java Class>>
Main
main
Main()
main(String[]):void
<<Java Class>>
Game
engine
GAME_FRAME_COLUMN: int
GAME_FRAME_ROW: int
BLOCK_SIZE: int
Game()
<<Java Class>>
PMComparator
engine
PMComparator()
compare(PacMan,PacMan):int
<<Java Class>>
Ghost
character
X: int
Y: int
BLOCK_SIZE: int
Ghost(int,int,int)
draw(Graphics2D,Object):void
move(int):void
setCentre():void
getX():int
getY():int
-ghosts 0..*
<<Java Class>>
Maze
engine
BLOCK_SIZE: int
MAZE_COLUMN: int
MAZE_ROW: int
gamePicture: BufferedImage
timer: Timer
powerTime: int
powerPelletActive: boolean
powerCalcTime: int
isRunning: boolean
fps: int
walls: short[]
dots: short[]
Maze(int,int,int,GameFrame)
initGhosts():void
stopTimer():void
newGame(String):void
loadGame(ArrayList<Elements>):void
saveGame():void
endGame():void
ghostHitCalc():void
hitByGhost():void
powerShouldStart():void
eatPowerPelletCalc():void
powerPelletEaten():void
newLevelRequired():void
eatDot():void
pelletEaten():void
setGhostsCenter():void
initDots():void
movePacMan(int):void
initGamePicture():void
paintComponent(Graphics):void
<<Java Class>>
Clyde
character
route: short[]
direction: int
callNum: int
iter: int
Clyde(int)
setCentre():void
draw(Graphics2D,Object):void
step(int):void
move(int):void
<<Java Class>>
Pinky
character
route: short[]
direction: int
callNum: int
iter: int
Pinky(int)
setCentre():void
draw(Graphics2D,Object):void
step(int):void
move(int):void
<<Java Class>>
Blinky
character
route: short[]
direction: int
callNum: int
iter: int
Blinky(int)
setCentre():void
draw(Graphics2D,Object):void
step(int):void
move(int):void
<<Java Class>>
Inky
character
route: short[]
direction: int
callNum: int
iter: int
Inky(int)
setCentre():void
draw(Graphics2D,Object):void
step(int):void
move(int):void
<<Java Class>>
PacMan
character
serialVersionUID: long
X: int
Y: int
oppositeX: int
oppositeY: int
BLOCK_SIZE: int
level: int
lives: int
points: long
name: String
pacFrameState: int
frameNum: int
direction: int
walls: short[]
PacMan(String,int,short[])
draw(Graphics2D,Object):void
changeBlockSize(int):void
move():void
measureOpposite():void
setDirection(int):void
increaseLevel():void
increasePoints(boolean):void
getPlayerName():String
getPoints():long
getLevel():int
getLives():int
getX():int
getY():int
changeLivesNumber(boolean):void
setCentre():void
-player 0..1
~leaderBoard 0..*
<<Java Class>>
PowerPellet
staticElement
X: int
Y: int
BLOCK_SIZE: int
PowerPellet(int,int,int)
draw(Graphics2D,Object):void
getX():int
getY():int
-powerPellets 0..*
<<Java Class>>
Elements
character
Elements()
draw(Graphics2D,Object):void
getPlayerName():String
changeBlockSize(int):void
getX():int
getY():int
-mazeElements 0..*
-list 0..*
<<Java Class>>
Pellet
staticElement
serialVersionUID: long
X: int
Y: int
BLOCK_SIZE: int
Pellet(int,int,int)
changeBlockSize(int):void
getX():int
getY():int
getPlayerName():String
draw(Graphics2D,Object):void
<<Java Class>>
Save
engine
serialVersionUID: long
age: int
name: String
Save(String)
getList():ArrayList<Elements>
giveList(ArrayList<Elements>,String):void
incrementAge():void
setName(String):void
getAge():int
getName():String
~saves 0..*
-maze 0..1
-game 0..1
<<Java Class>>
Effects
graphics
Effects()
initSounds():void
initStaticsPictures():void
initDynamicsPictures():void
-p 0..1
-pics 0..1
<<Java Class>>
GameFrame
engine
FRAME_COLUMN: int
FRAME_ROW: int
BLOCK_SIZE: int
GameFrame(int,int,int)
giveMenuFrame(MenuFrame):void
initVariables(int,int,int):void
initFrame():void
initPanels():void
startNewGame(String):void
startLoadGame(ArrayList<Elements>):void
saveGame(ArrayList<Elements>,String):void
updateScorePoints(long):void
updateScoreLevel(int):void
updateScoreLives(int):void
addToLeaderboard(PacMan):void
keyTyped(KeyEvent):void
keyPressed(KeyEvent):void
keyReleased(KeyEvent):void
-game 0..1
-score 0..1
<<Java Class>>
Step
engine
Step()
actionPerformed(ActionEvent):void
<<Java Enumeration>>
Statics
graphics
WALL: Statics
HEART: Statics
GATE: Statics
DOT: Statics
INKY: Statics
PINKY: Statics
BLINKY: Statics
CLYDE: Statics
POWER: Statics
kep: Image
Statics()
getStatPic():Image
<<Java Enumeration>>
Dynamics
graphics
PAC_UP: Dynamics
PAC_DOWN: Dynamics
PAC_RIGHT: Dynamics
PAC_LEFT: Dynamics
kep: ArrayList<BufferedImage>
Dynamics()
getDynPic(int):BufferedImage
<<Java Enumeration>>
Sounds
graphics
BEGINNING: Sounds
DEATH: Sounds
COIN: Sounds
EXTRA: Sounds
audioIS: AudioInput(Stream
clip: Clip
Sounds()
getClip():Clip
<<Java Class>>
Scoreboard
graphics
score: JLabel
levelNum: JLabel
lives: JLabel
Scoreboard(int,int,int)
initLabels(int):void
initLives(int):void
initLevelNum(int):void
initScore(int):void
updatePoints(long):void
updateLevel(int):void
updateLives(int):void
<<Java Class>>
MenuFrame
engine
newGame: JButton
quit: JButton
menu: JPanel
loadPanel: JPanel
board: JTable
name: JLabel
nameTextField: JTextField
start: JButton
load: JButton
FRAME_COLUMN: int
FRAME_ROW: int
BLOCK_SIZE: int
button: ArrayList<JButton>
MenuFrame(int,int,int,GameFrame)
initMenuBar():void
initLoadBoard():void
initSaveFolder():void
initSaves(JPanel):void
initLeaderBoard():void
initLeadList():void
saveLeaderList():void
addToLeaderList(PacMan):void
saveGameStates():void
addToSaveStates(ArrayList<Elements>,String):void
usibleLoadButtons(boolean):void
removeSaveGame(String):void
-game 0..1
menu 0..1
<<Java Class>>
LoadGameListener
engine
serial: int
LoadGameListener(int)
actionPerformed(ActionEvent):void
<<Java Class>>
NewGameButtonListener
engine
NewGameButtonListener()
actionPerformed(ActionEvent):void
<<Java Class>>
LoadButtonListener
engine
LoadButtonListener()
actionPerformed(ActionEvent):void
<<Java Class>>
StartButtonListener
engine
StartButtonListener()
actionPerformed(ActionEvent):void
<<Java Class>>
QuitListener
engine
QuitListener()
actionPerformed(ActionEvent):void

3. fejezet

Osztálymutató

3.1. Osztálylista

Az összes osztály, struktúra, unió és interfész listája rövid leírásokkal:

4. fejezet

Fájlmutató

4.1. Fájllista

Az összes fájl listája rövid leírásokkal:

5. fejezet

Névterek dokumentációja

5.1. character csomag

Osztályok

- class Blinky

  Blinky szellemet megvalósító osztály.
- class Clyde

  Clyde szellemet megvalósító osztály.
- class Elements

  Ez az osztály a közös őse a PacMan-nek, és a Pellet-nek.
- class Ghost

  A szellemek közös absztrakt őse.
- class Inky

  Inky szellemet megvalósító osztály.
- class PacMan

  Ez az osztály valósítja meg a PacMan karaktert, és annak működését.
- class Pinky

  Pinky szellemet megvalósító osztály.

5.2. engine csomag

Osztályok

- class Game

  A játéknak összefoglaló keretet adó osztály.
- class GameFrame

  A játék keretét megvalósító osztály.
- class Maze

  Ez az osztály a játék lelke.
- class MenuFrame

  Ez az osztály felel a menü megvalósításáért.
- class PMComparator

  PacMan kompatárot a ranglistához.
- class Save

  Ez az osztály felel a mentés megvalósításáért.

5.3. graphics csomag

Osztályok

- class Effects
- class Scoreboard

  Ez az osztály felel az eredménytábláért.

5.4. staticElement csomag

Osztályok

- class Pellet

  Ez az osztály valósítja meg a játékban a felvehető pontokat.
- class PowerPellet

  Ez az osztály valósítja meg a játékban felvehető extra pontokat.

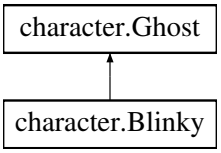
6. fejezet

Osztályok dokumentációja

6.1. character.Blinky osztályreferencia

Blinky szellemet megvalósító osztály.

A character.Blinky osztály származási diagramja:



Publikus tagfüggvények

- Blinky (int BLOCK_SIZE)

  Az osztály konstruktora.
- void setCentre ()

  Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.
- void draw (Graphics2D frame_g, Object c)

  Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.
- void move (int level)

  Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Privát tagfüggvények

- void step (int level)

  Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Privát attribútumok

- final short[ ] route

  A szellem útvonalát tároló objektum.
- int direction

  Az irány, amerre a szellem éppen halad.
- int callNum
- int iter

  Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

További örökölt tagok

6.1.1. Részletes leírás

Blinky szellemet megvalósító osztály.

6.1.2. Konstruktorok és destruktorok dokumentációja

6.1.2.1. Blinky()

```
character.Blinky.Blinky (
            int BLOCK_SIZE )
```

Az osztály konstruktora.

Paraméterek

BLOCK_SIZE	az objektum létrehozásakor érvényben lévő pályablokk méret

6.1.3. Tagfüggvények dokumentációja

6.1.3.1. draw()

```
void character.Blinky.draw (
            Graphics2D frame_g,
            Object c )
```

Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementált ősök: character.Ghost.

6.1.3.2. move()

```
void character.Blinky.move (
            int level )
```

Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Itt hívódik a step metódus is. Ezt a heterogén kollekciónak köszönhetően egy for ciklussal a Maze objektum el tudja végezni. A 26 azért kell, mert a központi helyről 13 blokkot megy a körkörös mozgás kezdőpontjáig, viszont nem akarjuk, hogy visszamenjen a középső "zárkába".

Paraméterek

level	hányas szinten tart éppen a pacman

Újraimplementált ősök: character.Ghost.

6.1.3.3. setCentre()

```
void character.Blinky.setCentre ( )
```

Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.

Újraimplementált ősök: character.Ghost.

6.1.3.4. step()

```
void character.Blinky.step (
            int level )  [private]
```

Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Paraméterek

level	a PacMan szintje

6.1.4. Adattagok dokumentációja

6.1.4.1. callNum

```
int character.Blinky.callNum  [private]
```

6.1.4.2. direction

```
int character.Blinky.direction  [private]
```

Az irány, amerre a szellem éppen halad.

6.1.4.3. iter

```
int character.Blinky.iter  [private]
```

Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

Ezen változó segítségével iterál végig az útvonalon a szellem.

6.1.4.4. route

```
final short [] character.Blinky.route  [private]
```

Kezdő érték:

```
= {
        8,9,8,8,7,8,7,7,7,6,6,6,6,5,6,4,5,4,4,4,3,4,2,4,1,4,
            1,3,1,2,1,1,2,1,3,1,4,1,4,2,4,3,4,4,5,4,6,4,7,4,
            7,3,7,2,7,1,6,1,5,1,4,1,4,2,4,3,4,4,5,4,6,4,
            6,5,6,6,7,6,7,7,7,8,6,8,6,9,6,10,5,10,4,10,
            4,9,4,8,4,7,4,6,3,6,2,6,1,6,1,5,1,4
    }
```

A szellem útvonalát tároló objektum.

X,Y koordináta párokat tárol, amik az oszlopszám, sorszám mintát követik. Fontos: ezek nem pixel koordináták, hanem blokk koordináták.

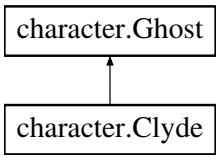
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Blinky.java

6.2. character.Clyde osztályreferencia

Clyde szellemet megvalósító osztály.

A character.Clyde osztály származási diagramja:



Publikus tagfüggvények

- Clyde (int BLOCK_SIZE)

  Az osztály konstruktora.
- void setCentre ()

  Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.
- void draw (Graphics2D frame_g, Object c)

  Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.
- void move (int level)

  Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Privát tagfüggvények

- void step (int level)

  Ez a metódus minden hívsára annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Privát attribútumok

- final short[ ] route

  A szellem útvonalát tároló objektum.
- int direction

  Az irány, amerre a szellem éppen halad.
- int callNum
- int iter

  Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

További örökölt tagok

6.2.1. Részletes leírás

Clyde szellemet megvalósító osztály.

6.2.2. Konstruktorok és destruktorok dokumentációja

6.2.2.1. Clyde()

```
character.Clyde.Clyde (
            int BLOCK_SIZE )
```

Az osztály konstruktora.

Paraméterek

BLOCK_SIZE	az objektum létrehozásakor érvényben lévő pályablokk méret

6.2.3. Tagfüggvények dokumentációja

6.2.3.1. draw()

```
void character.Clyde.draw (
            Graphics2D frame_g,
            Object c )
```

Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementált ősök: character.Ghost.

6.2.3.2. move()

```
void character.Clyde.move (
            int level )
```

Ez a metúdus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Itt hívódik a step metódus is. Ezt a heterogén kollekciónak köszönhetően egy for ciklussal a Maze objektum el tudja végezni. A 34 azért kell, mert a központi helyről 17 blokkot megy a körkörös mozgás kezdőpontjáig, viszont nem akarjuk, hogy visszamenjen a középső "zárkába".

Paraméterek

level	hányas szinten tart éppen a pacman

Újraimplementált ősök: character.Ghost.

6.2.3.3. setCentre()

```
void character.Clyde.setCentre ( )
```

Ez a metúdus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.

Újraimplementált ősök: character.Ghost.

6.2.3.4. step()

```
void character.Clyde.step (
            int level )  [private]
```

Ez a metódus minden hívsára annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Paraméterek

level	a PacMan szintje

6.2.4. Adattagok dokumentációja

6.2.4.1. callNum

```
int character.Clyde.callNum  [private]
```

6.2.4.2. direction

```
int character.Clyde.direction  [private]
```

Az irány, amerre a szellem éppen halad.

6.2.4.3. iter

```
int character.Clyde.iter  [private]
```

Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

Ezen változó segítségével iterál végig az útvonalon a szellem.

6.2.4.4. route

```
final short [] character.Clyde.route  [private]
```

Kezdő érték:

```
= {
        8,9,8,8,9,8,10,8,10,9,10,10,11,10,12,10,12,9,12,8,12,7,12,6,13,6,14,6,15,6,15,5,15,4,
            14,4,13,4,12,4,11,4,10,4,9,4,9,3,9,2,9,1,
            10,1,11,1,12,1,13,1,14,1,15,1,15,2,15,3,
            15,4,15,5,15,6,14,6,13,6,12,6,12,5,12,4,
            11,4,10,4,10,5,10,6,9,6,9,7,9,8,10,8,10,9,
            10,10,11,10,12,10,12,9,12,8,12,7,12,6,13,16,
            14,6,15,6,15,5,15,4
    }
```

A szellem útvonalát tároló objektum.

X,Y koordináta párokat tárol, amik az oszlopszám, sorszám mintát követik. Fontos: ezek nem pixel koordináták, hanem blokk koordináták.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Clyde.java

6.3. graphics.Effects.Dynamics felsoroló referencia

A dinamikusabb, a PacMan-hez tartalmazó képek enumerációja.

Publikus tagfüggvények

- BufferedImage getDynPic (int i)

  Getter a tároló típusonként tárolt elemeire, indexelhetően.

Publikus attribútumok

- PAC_UP

  A PacMan felfele irányának leképezése.
- PAC_DOWN

  A PacMan lefele irányának leképezése.
- PAC_RIGHT

  A PacMan jobbra irányának leképezése.
- PAC_LEFT

Privát attribútumok

- final ArrayList< BufferedImage > kep = new ArrayList<>()

  < *A PacMan balra irányának leképezése*

6.3.1. Részletes leírás

A dinamikusabb, a PacMan-hez tartalmazó képek enumerációja.

Irányonként tartozik hozzá egy tároló, aminek a segítségével a Timer minden tick-jére lépteti a tároló elemeit, így meganimálva a PacMan szájmozgását.

6.3.2. Tagfüggvények dokumentációja

6.3.2.1. getDynPic()

```
BufferedImage graphics.Effects.Dynamics.getDynPic (
            int i )
```

Getter a tároló típusonként tárolt elemeire, indexelhetően.

Elérés pl.: Pictures.Dynamics.PAC_UP.getDynPic(0);

Paraméterek

i	az indexe az irány szerint elérni kívánt képnek

Visszatérési érték

az adott indexhez, és irányhoz tartozó kép

6.3.3. Adattagok dokumentációja

6.3.3.1. kep

```
final ArrayList<BufferedImage> graphics.Effects.Dynamics.kep = new ArrayList<>()  [private]
```

< A PacMan balra irányának leképezése

Az iránytípusonkétni tároló

6.3.3.2. PAC_DOWN

`graphics.Effects.Dynamics.PAC_DOWN`

A PacMan lefele irányának leképezése.

6.3.3.3. PAC_LEFT

`graphics.Effects.Dynamics.PAC_LEFT`

6.3.3.4. PAC_RIGHT

`graphics.Effects.Dynamics.PAC_RIGHT`

A PacMan jobbra irányának leképezése.

6.3.3.5. PAC_UP

`graphics.Effects.Dynamics.PAC_UP`

A PacMan felfele irányának leképezése.

A dokumentáció ehhez az enum-hoz a következő fájl alapján készült:

- Effects.java

6.4. graphics.Effects osztályreferencia

Osztályok

- enum Dynamics

  A dinamikusabb, a PacMan-hez tartalmazó képek enumerációja.
- enum Sounds

  A PacMan hangjait tartalmazó enumeráció.
- enum Statics

  A statikus elemeket tartalmazó enumeráció.

Publikus tagfüggvények

- Effects () throws IOException, UnsupportedAudioFileException, LineUnavailableException

  Az osztály konstruktora.
- void initSounds () throws IOException, UnsupportedAudioFileException, LineUnavailableException

  Inicializáló függvény a PacMan hangjaira.

Privát tagfüggvények

- void initStaticsPictures ()

  Ez a metódus felel a statikus képek inicializálásért.
- void initDynamicsPictures () throws IOException

  Ez a metódus felel a dinamikus, PacMan-hez tartozó képek inicializálásáért.

6.4.1. Konstruktorok és destruktorok dokumentációja

6.4.1.1. Effects()

```
graphics.Effects.Effects ( ) throws IOException, UnsupportedAudioFileException, LineUnavailable↩
Exception
```

Az osztály konstruktora.

Inicializálja a Statikus, illetve Dinamikus (felhasználás szerint) képeket.

Kivételek

IOException	
UnsupportedAudioFileException	
LineUnavailableException	

6.4.2. Tagfüggvények dokumentációja

6.4.2.1. initDynamicsPictures()

```
void graphics.Effects.initDynamicsPictures ( ) throws IOException  [private]
```

Ez a metódus felel a dinamikus, PacMan-hez tartozó képek inicializálásáért.

Itt már sokkal jobban tudtam ciklusba szervezni a képek beolvasását.

Kivételek

IOException	

6.4.2.2. initSounds()

```
void graphics.Effects.initSounds ( ) throws IOException, UnsupportedAudioFileException, Line↩
UnavailableException
```

Inicializáló függvény a PacMan hangjaira.

Minden Timer tick-re lefut, hogy mindig le lehessen játszani a hangfájlokat.

Kivételek

IOException	
UnsupportedAudioFileException	
LineUnavailableException	

Beginning

DEATH

EXTRA

COIN

6.4.2.3. initStaticsPictures()

```
void graphics.Effects.initStaticsPictures ( )  [private]
```

Ez a metódus felel a statikus képek inicializálásért.

Sajnos, tekintve, hogy minden kép egyedi, és egy van belőlük, nem igazán tudtam kevésbé "sormintává" tenni.

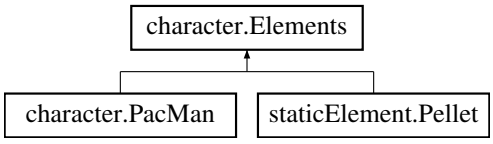
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Effects.java

6.5. character.Elements osztályreferencia

Ez az osztály a közös őse a PacMan-nek, és a Pellet-nek.

A character.Elements osztály származási diagramja:



Publikus tagfüggvények

- abstract void draw (Graphics2D frame_g, Object c)

  Absztrakt metódus az osztály leszármazottjainak kirajzolásához.
- abstract String getPlayerName ()

  Getter metódus, ami Pellet esetén null, PacMan esetén pedig a játékos által begépelt nevet adja vissza.
- abstract void changeBlockSize (int b)

  Ez a függvény szolgál arra, hogyha netán két programindítás között megváltozna a pályablokk mérete, akkor minden leszármazott megkapja az új méretet, és azzal tudjon tovább dolgozni.
- abstract int getX ()

  Getter az adott objektum X koordinátájára.
- abstract int getY ()

  Getter az adott objektum Y koordinátájára.

6.5.1. Részletes leírás

Ez az osztály a közös őse a PacMan-nek, és a Pellet-nek.

Azért van szükség erre a közös absztrakt ős osztályra, mert így a grafika megvalósítása sokkal könnyebb a heterogén kollekció révén.

6.5.2. Tagfüggvények dokumentációja

6.5.2.1. changeBlockSize()

```
abstract void character.Elements.changeBlockSize (
            int b )  [abstract]
```

Ez a függvény szolgál arra, hogyha netán két programindítás között megváltozna a pályablokk mérete, akkor minden leszármazott megkapja az új méretet, és azzal tudjon tovább dolgozni.

Paraméterek

b	az új pályablokk méret

Újraimplementáló leszármazottak: staticElement.Pellet és character.PacMan.

6.5.2.2. draw()

```
abstract void character.Elements.draw (
            Graphics2D frame_g,
            Object c )  [abstract]
```

Absztrakt metódus az osztály leszármazottjainak kirajzolásához.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementáló leszármazottak: staticElement.Pellet és character.PacMan.

6.5.2.3. getPlayerName()

```
abstract String character.Elements.getPlayerName ( )  [abstract]
```

Getter metódus, ami Pellet esetén null, PacMan esetén pedig a játékos által begépelt nevet adja vissza.

Visszatérési érték

a játékos által begépelt név

Újraimplementáló leszármazottak: staticElement.Pellet és character.PacMan.

6.5.2.4. getX()

```
abstract int character.Elements.getX ( )  [abstract]
```

Getter az adott objektum X koordinátájára.

Visszatérési érték

az X koordináta

Újraimplementáló leszármazottak: staticElement.Pellet és character.PacMan.

6.5.2.5. getY()

```
abstract int character.Elements.getY ( )  [abstract]
```

Getter az adott objektum Y koordinátájára.

Visszatérési érték

az Y koordináta

Újraimplementáló leszármazottak: staticElement.Pellet és character.PacMan.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Elements.java

6.6. engine.Game osztályreferencia

A játéknak összefglaló keretet adó osztály.

Publikus tagfüggvények

- Game () throws IOException, ClassNotFoundException, LineUnavailableException, UnsupportedAudioFile↩ Exception

  Az osztály konstruktora.

Statikus privát attribútumok

- static int GAME_FRAME_COLUMN = 17

  NE SZERKESZD A panelek szélessége blokkszámban.
- static int GAME_FRAME_ROW = 24

  NE SZERKESZD A panelek magassága blokkszámban.
- static int BLOCK_SIZE = 30

  Ez szerkeszthető, de legyen 3-mal osztható, és páros: a pályablokk mérete, pixelben.

6.6.1. Részletes leírás

A játéknak összefglaló keretet adó osztály.

Ez az osztály alkot keretet, a menü, és a játékpanel körül. Nincs sok funkciója, de itt állítható be a pályablokk méret is. A játékban az X koordináta tengely balról jobbra, az Y fentről lefele növekszik.

6.6.2. Konstruktorok és destruktorok dokumentációja

6.6.2.1. Game()

```
engine.Game.Game ( ) throws IOException, ClassNotFoundException, LineUnavailableException,
UnsupportedAudioFileException
```

Az osztály konstruktora.

Itt inicializálódik a játékkeret, és a menükeret is.

Kivételek

IOException	
ClassNotFoundException	
LineUnavailableException	
UnsupportedAudioFileException	

6.6.3. Adattagok dokumentációja

6.6.3.1. BLOCK_SIZE

```
int engine.Game.BLOCK_SIZE = 30  [static], [private]
```

Ez szerkeszthető, de legyen 3-mal osztható, és páros: a pályablokk mérete, pixelben.

6.6.3.2. GAME_FRAME_COLUMN

```
int engine.Game.GAME_FRAME_COLUMN = 17  [static], [private]
```

NE SZERKESZD A panelek szélessége blokkszámban.

6.6.3.3. GAME_FRAME_ROW

```
int engine.Game.GAME_FRAME_ROW = 24  [static], [private]
```

NE SZERKESZD A panelek magassága blokkszámban.

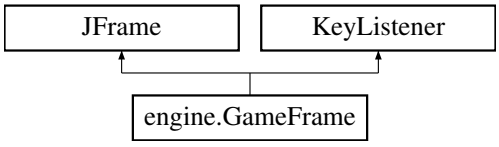
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Game.java

6.7. engine.GameFrame osztályreferencia

A játék keretét megvalósító osztály.

Az engine.GameFrame osztály származási diagramja:



Publikus tagfüggvények

- GameFrame (int column, int row, int block) throws IOException, LineUnavailableException, Unsupported↩ AudioFileException

  Az osztály konstruktora.

- void giveMenuFrame (MenuFrame f)

  Mivel a mentés/betöltésért a MenuFrame osztály felel, így a GameFrame-nek tudnia kell annak példányáról.

- void startNewGame (String name)

  Ha a menüből új játékot indít valaki, akkor ez a metódus hívódik meg, ami meghívja a pálya newGame metódusát.

- void startLoadGame (ArrayList< Elements > list)

  Ha a menüből valaki betölt egy játékot, akkor ez a metódus hívódik meg, ami meghívja a pálya loadGame metódusát.

- void saveGame (ArrayList< Elements > list, String name) throws IOException

  Minden egyes Escape billentyű leütésre a pálya elmetni a játékállást, amivel meghívja ezt a metódust, ez pedig meghívja a menü addToSaveStates metódusát.

- void updateScorePoints (long points)

  Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn a pontok állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updatePoints metódusát.

- void updateScoreLevel (int level)

  Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn a szintek állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updateLevel metódusát.

- void updateScoreLives (int num)

  Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn az életek számának állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updateLives metódusát.

- void addToLeaderboard (PacMan p)

  Ha egy játékos meghal, vagy a harmadik szinten is összeszedte az összes kicsi pontot, akkor vége a játéknak, és felkerül az eredményjelzőre.

- void keyTyped (KeyEvent e)

  Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus, de ezt nem használom semmire.

- void keyPressed (KeyEvent e)

  Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus.

- void keyReleased (KeyEvent e)

  Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus, de ezt nem használom semmire.

Privát tagfüggvények

- void initVariables (int c, int r, int b)

  Ez a metódus felel a változók inicializásáért.

- void initFrame ()

  A JFrame őshöz kapcsolódó inicializálások.

- void initPanels () throws UnsupportedAudioFileException, IOException, LineUnavailableException

  Itt jön létre a pályát magába foglaló Maze osztály egy példánya, illetve az eredményjelzésért felelős Scoreboard osztály egy példánya.

Privát attribútumok

- Scoreboard score

  Az eredményjelző Scoreboard osztály egy példánya.
- Maze maze

  A pálya (Maze) osztály egy példánya.
- MenuFrame menu

  A menüt megalkotó MenuFrame osztály.
- final Effects pics = new Effects()

  A képekért felelős, enumerációkat tartalmazó Pictures osztály egy példánya.
- int FRAME_COLUMN

  A játékpanel szélessége, blokkokban.
- int FRAME_ROW

  A játékpanel magassága, az eredményjelzővel együtt, blokkokban.
- int BLOCK_SIZE

  Az indításkor érvényben lévő blokkméret a Game osztálytól, pixelben.

6.7.1. Részletes leírás

A játék keretét megvalósító osztály.

Ez az osztály felel azért, hogy a pálya, és az ereményjelző szekció össze legyen fogva, és egyszerre legyenek kezelhetőek. Ha valaki megváltoztatja a páyablokk méretet a Game osztályban, akkor ez az ablak, benne a játékkal együtt átméreteződik. A JFrame ősből származik le, és a KeyListener interfészt valósítja meg. Ez az osztály érzékeli a játékos által lenyomott billentyűket.

6.7.2. Konstruktorok és destruktorok dokumentációja

6.7.2.1. GameFrame()

```
engine.GameFrame.GameFrame (
            int column,
            int row,
            int block ) throws IOException, LineUnavailableException, UnsupportedAudioFile↩
Exception
```

Az osztály konstruktora.

Itt kikapcsoljuk a dekorálást, azaz az operációs rendszer ablakkeretét, illetve az átméretezhetőséget is, majd középre igazítjuk a keretet. A jobb átláthatóság kedvéért az inicializálást három függvénybe szerveztem ki.

Paraméterek

column	a Game osztály példányától kapott szélesség, blokkokban
row	a Game osztály példányától kapott magasság, blokkokban
block	a Game osztály példányától kapott blokkméret, pixelben

Kivételek

IOException	
LineUnavailableException	
UnsupportedAudioFileException	

6.7.3. Tagfüggvények dokumentációja

6.7.3.1. addToLeaderboard()

```
void engine.GameFrame.addToLeaderboard (
            PacMan p )
```

Ha egy játékos meghal, vagy a harmadik szinten is összeszedte az összes kicsi pontot, akkor vége a játéknak, és felkerül az eredményjelzőre.

Ilyenkor ez a metódus hívódik meg, ami továbbítja a játékban résztvevő PacMan példányt a menünek.

Paraméterek

p	a PacMan példány

6.7.3.2. giveMenuFrame()

```
void engine.GameFrame.giveMenuFrame (
            MenuFrame f )
```

Mivel a mentés/betöltésért a MenuFrame osztály felel, így a GameFrame-nek tudnia kell annak példányáról.

Erre szolgál ez a metódus.

Paraméterek

f	a használandó MenuFrame objektum

6.7.3.3. initFrame()

```
void engine.GameFrame.initFrame ( )  [private]
```

A JFrame őshöz kapcsolódó inicializálások.

6.7.3.4. initPanels()

```
void engine.GameFrame.initPanels ( ) throws UnsupportedAudioFileException, IOException, Line↩
UnavailableException  [private]
```

Itt jön létre a pályát magába foglaló Maze osztály egy példánya, illetve az eredményjelzésért felelős Scoreboard osztály egy példánya.

Kivételek

UnsupportedAudioFileException	
IOException	
LineUnavailableException	

6.7.3.5. initVariables()

```
void engine.GameFrame.initVariables (
            int c,
            int r,
            int b )  [private]
```

Ez a metódus felel a változók inicializásáért.

Paraméterek

c	a játékkeret szélessége, blokkokban
r	a játékkeret magassága az eredményjelzővel együtt, blokkokban
b	az indításkor érvényben lévő blokkméret, pixelben

6.7.3.6. keyPressed()

```
void engine.GameFrame.keyPressed (
            KeyEvent e )
```

Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus.

Itt kezelem le, hogy játék közben milyen billentyűleütések történnek. Ez lehet az Escape, illetve a négy nyíl.

Paraméterek

e	billentyűleütés történt

6.7.3.7. keyReleased()

```
void engine.GameFrame.keyReleased (
            KeyEvent e )
```

Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus, de ezt nem használom semmire.

Paraméterek

e	billentyűfelengedés történt

6.7.3.8. keyTyped()

```
void engine.GameFrame.keyTyped (
            KeyEvent e )
```

Mivel az osztály megvalósítja a KeyListener intrefészt, ezért ez egy kötelezően implementálandó metódus, de ezt nem használom semmire.

Paraméterek

e	billentyűleütés történt

6.7.3.9. saveGame()

```
void engine.GameFrame.saveGame (
            ArrayList< Elements > list,
            String name ) throws IOException
```

Minden egyes Escape billentyű leütésre a pálya elmetni a játékállást, amivel meghívja ezt a metódust, ez pedig meghívja a menü addToSaveStates metódusát.

Paraméterek

list	a játékelemeket tartalmazó lista
name	a PacMan-ben eltárolt név

Kivételek

IOException	

6.7.3.10. startLoadGame()

```
void engine.GameFrame.startLoadGame (
            ArrayList< Elements > list )
```

Ha a menüből valaki betölt egy játékot, akkor ez a metódus hívódik meg, ami meghívja a pálya loadGame metódusát.

Paraméterek

list	a Save osztály megfelelő példányából kinyert játékelemeket tároló lista

6.7.3.11. startNewGame()

```
void engine.GameFrame.startNewGame (
            String name )
```

Ha a menüből új játékot indít valaki, akkor ez a metódus hívódik meg, ami meghívja a pálya newGame metódusát.

Paraméterek

name	a játékos által új játék kezdésekor begépelt név

6.7.3.12. updateScoreLevel()

```
void engine.GameFrame.updateScoreLevel (
            int level )
```

Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn a szintek állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updateLevel metódusát.

Paraméterek

level	int típusú szám, amire kell frissíteni

6.7.3.13. updateScoreLives()

```
void engine.GameFrame.updateScoreLives (
            int num )
```

Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn az életek számának állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updateLives metódusát.

Paraméterek

num	int típusú szám, amire kell frissíteni

6.7.3.14. updateScorePoints()

```
void engine.GameFrame.updateScorePoints (
            long points )
```

Ha a pálya azt mondja, hogy frissíteni kell az eredményjelzőn a pontok állását, akkor ezt a metódust hívja meg, ami meghívja a Scoreboard osztály példányának updatePoints metódusát.

Paraméterek

points	long típusú szám, amire kell frissíteni

6.7.4. Adattagok dokumentációja

6.7.4.1. BLOCK_SIZE

```
int engine.GameFrame.BLOCK_SIZE  [private]
```

Az indításkor érvényben lévő blokkméret a Game osztálytól, pixelben.

6.7.4.2. FRAME_COLUMN

```
int engine.GameFrame.FRAME_COLUMN  [private]
```

A játékpanel szélessége, blokkokban.

6.7.4.3. FRAME_ROW

```
int engine.GameFrame.FRAME_ROW  [private]
```

A játékpanel magassága, az eredményjelzővel együtt, blokkokban.

6.7.4.4. maze

```
Maze engine.GameFrame.maze  [private]
```

A pálya (Maze) osztály egy példánya.

6.7.4.5. menu

```
MenuFrame engine.GameFrame.menu  [private]
```

A menüt megalkotó MenuFrame osztály.

A MenuFrame objektum inicializálása után a GameFrame megkapja azt

6.7.4.6. pics

```
final Effects engine.GameFrame.pics = new Effects()  [private]
```

A képekért felelős, enumerációkat tartalmazó Pictures osztály egy példánya.

6.7.4.7. score

```
Scoreboard engine.GameFrame.score  [private]
```

Az eredményjelző Scoreboard osztály egy példánya.

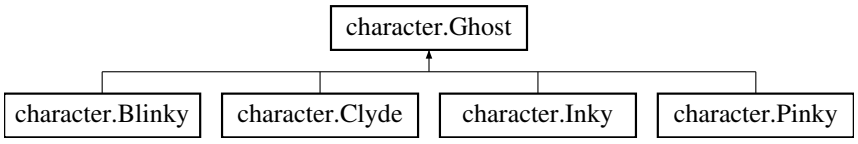
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- GameFrame.java

6.8. character.Ghost osztályreferencia

A szellemek közös absztrakt őse.

A character.Ghost osztály származási diagramja:



Publikus tagfüggvények

- Ghost (int x, int y, int block)

  Az osztály konstruktora.
- abstract void draw (Graphics2D frame_g, Object c)

  Absztrakt metódus az osztály leszármazottjainak kirajzolásához.
- abstract void move (int level)

  Absztrakt metódus az osztály leszármazottjainak mozgatásához.
- abstract void setCentre ()

  Ez a metódus a kiindulópontjába helyez minden egyes leszármazottat.
- int getX ()

  Getter az X koordinátához.
- int getY ()

  Getter az Y koordinátához.

Védett attribútumok

- int X

  Adott szellem objektum X koordinátája pixelben.
- int Y

  Adott szellem objektum Y koordinátája pixelben
- int BLOCK_SIZE

  A szellemek létrehozásakor érvényben lévő pályablokk méret.

6.8.1. Részletes leírás

A szellemek közös absztrakt őse.

Fontos, mivl ennek köszönhetően lehet heterogén kollekciót használni a szellemekhez. Minden szellem a program indítása után, a játék elkezdésekor inicializálódik. Sajnos egyik szellemet sem vezérli AI, előre meghatározott útvonalon közlekednek, de remélem, hogy ettől még kellően nehéz lesz vele játszani!:) Mindegyik szellem ütközéskor egyel csökenti a PacMan életeinek számát.

6.8.2. Konstruktorok és destruktorok dokumentációja

6.8.2.1. Ghost()

```
character.Ghost.Ghost (
            int x,
            int y,
            int block )
```

Az osztály konstruktora.

Paraméterek

x	az adott objektum X koordinátája
y	az adott objektum Y koordinátája
block	az adott objektum létrehozásakor érvényben lévő pályablokk méret

6.8.3. Tagfüggvények dokumentációja

6.8.3.1. draw()

```
abstract void character.Ghost.draw (
            Graphics2D frame_g,
            Object c )  [abstract]
```

Absztrakt metódus az osztály leszármazottjainak kirajzolásához.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementáló leszármazottak: character.Pinky, character.Inky, character.Clyde és character.Blinky.

6.8.3.2. getX()

```
int character.Ghost.getX ( )
```

Getter az X koordinátához.

Visszatérési érték

az X koordináta

6.8.3.3. getY()

```
int character.Ghost.getY ( )
```

Getter az Y koordinátához.

Visszatérési érték

az Y koordináta

6.8.3.4. move()

```
abstract void character.Ghost.move (
            int level )  [abstract]
```

Absztrakt metódus az osztály leszármazottjainak mozgatásához.

Paraméterek

level	hányas szinten tart éppen a pacman

Újraimplementáló leszármazottak: character.Pinky, character.Inky, character.Clyde és character.Blinky.

6.8.3.5. setCentre()

```
abstract void character.Ghost.setCentre ( )  [abstract]
```

Ez a metódus a kiindulópontjába helyez minden egyes leszármazottat.

Újraimplementáló leszármazottak: character.Pinky, character.Inky, character.Clyde és character.Blinky.

6.8.4. Adattagok dokumentációja

6.8.4.1. BLOCK_SIZE

```
int character.Ghost.BLOCK_SIZE  [protected]
```

A szellemek létrehozásakor érvényben lévő pályablokk méret.

6.8.4.2. X

```
int character.Ghost.X  [protected]
```

Adott szellem objektum X koordinátája pixelben.

6.8.4.3. Y

```
int character.Ghost.Y  [protected]
```

Adott szellem objektum Y koordinátája pixelben

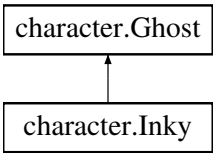
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Ghost.java

6.9. character.Inky osztályreferencia

Inky szellemet megvalósító osztály.

A character.Inky osztály származási diagramja:



Publikus tagfüggvények

- Inky (int BLOCK_SIZE)

  Az osztály konstruktora.
- void setCentre ()

  Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.
- void draw (Graphics2D frame_g, Object c)

  Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.
- void move (int level)

  Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Privát tagfüggvények

- void step (int level)

  Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Privát attribútumok

- final short[ ] route

  A szellem útvonalát tároló objektum.
- int direction

  Az irány, amerre a szellem éppen halad.
- int callNum
- int iter

  Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

További örökölt tagok

6.9.1. Részletes leírás

Inky szellemet megvalósító osztály.

6.9.2. Konstruktorok és destruktorok dokumentációja

6.9.2.1. Inky()

```
character.Inky.Inky (
            int BLOCK_SIZE )
```

Az osztály konstruktora.

Paraméterek

BLOCK_SIZE	az objektum létrehozásakor érvényben lévő pályablokk méret

6.9.3. Tagfüggvények dokumentációja

6.9.3.1. draw()

```
void character.Inky.draw (
            Graphics2D frame_g,
            Object c )
```

Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementált ősök: character.Ghost.

6.9.3.2. move()

```
void character.Inky.move (
            int level )
```

Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Itt hívódik a step metódus is. Ezt a heterogén kollekciónak köszönhetően egy for ciklussal a Maze objektum el tudja végezni. A 36 azért kell, mert a központi helyről 18 blokkot megy a körkörös mozgás kezdőpontjáig, viszont nem akarjuk, hogy visszamenjen a középső "zárkába".

Paraméterek

level	hányas szinten tart éppen a pacman

Újraimplementált ősök: character.Ghost.

6.9.3.3. setCentre()

```
void character.Inky.setCentre ( )
```

Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.

Újraimplementált ősök: character.Ghost.

6.9.3.4. step()

```
void character.Inky.step (
            int level )  [private]
```

Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Paraméterek

level	a PacMan szintje

6.9.4. Adattagok dokumentációja

6.9.4.1. callNum

```
int character.Inky.callNum  [private]
```

6.9.4.2. direction

```
int character.Inky.direction  [private]
```

Az irány, amerre a szellem éppen halad.

6.9.4.3. iter

```
int character.Inky.iter  [private]
```

Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

Ezen változó segítségével iterál végig az útvonalon a szellem.

6.9.4.4. route

```
final short [] character.Inky.route  [private]
```

Kezdő érték:

```
= {
        8,9,8,8,7,8,6,8,6,9,6,10,6,11,6,12,6,12,6,14,5,14,4,14,3,14,2,14,2,15,2,16,2,17,1,17,
            2,17,3,17,4,17,4,16,5,16,6,16,6,15,6,14,5,14,4,14,3,14,2,14,
            1,14,2,14,2,15,2,16,2,17,3,17,4,17,4,16,5,16,6,16,6,15,
            6,14,6,13,6,12,6,11,6,10,5,10,4,10,4,11,4,12,4,13,4,14,
            5,14,6,14,6,15,6,16,6,17,7,17,7,18,7,19,6,19,5,19,4,19,
            3,19,2,19,1,19,1,18,1,17
    }
```

A szellem útvonalát tároló objektum.

X,Y koordináta párokat tárol, amik az oszlopszám, sorszám mintát követik. Fontos: ezek nem pixel koordináták, hanem blokk koordináták.

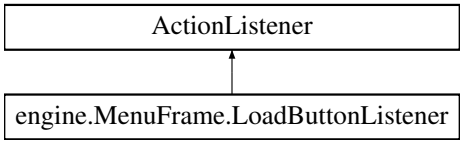
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Inky.java

6.10. engine.MenuFrame.LoadButtonListener osztályreferencia

Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a LOAD GAME Jbuttont, akkor megjelenjenek a mentéseket vezérlő gombok.

Az engine.MenuFrame.LoadButtonListener osztály származási diagramja:



Publikus tagfüggvények

- void actionPerformed (ActionEvent e)

  Megjeleníti a betöltési gombokat, és eltűnteti az új játékhoz szükséges elemeket.

6.10.1. Részletes leírás

Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a LOAD GAME Jbuttont, akkor megjelenjenek a mentéseket vezérlő gombok.

6.10.2. Tagfüggvények dokumentációja

6.10.2.1. actionPerformed()

```
void engine.MenuFrame.LoadButtonListener.actionPerformed (
            ActionEvent e )
```

Megjeleníti a betöltési gombokat, és eltűnteti az új játékhoz szükséges elemeket.

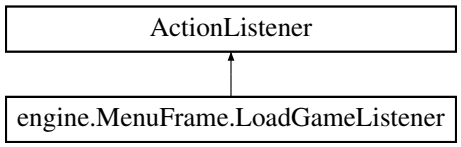
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.11. engine.MenuFrame.LoadGameListener osztályreferencia

Ez az ActionListener subclass felel azért, hogy a kiválasztott mentett állás betöltődjön, és induljon a játék.

Az engine.MenuFrame.LoadGameListener osztály származási diagramja:



Publikus tagfüggvények

- LoadGameListener (int s)

  Az osztály konstruktora.
- void actionPerformed (ActionEvent e)

  Ha megnyomjuk a gombot, elindítja a játékot a megfelelő Save objektumtól kapott játékelemeket tartalmazó listával, meghívja a game objektum startLoadGame metódusát.

Privát attribútumok

- final int serial

  Annak a mentésnek az index eleme, amiért ez a gomb felel.

6.11.1. Részletes leírás

Ez az ActionListener subclass felel azért, hogy a kiválasztott mentett állás betöltődjön, és induljon a játék.

6.11.2. Konstruktorok és destruktorok dokumentációja

6.11.2.1. LoadGameListener()

```
engine.MenuFrame.LoadGameListener.LoadGameListener (
            int s )
```

Az osztály konstruktora.

Itt kapja meg, hogy melyik indexű Save objektumért felel

Paraméterek

s	a Save objektum indexe

6.11.3. Tagfüggvények dokumentációja

6.11.3.1. actionPerformed()

```
void engine.MenuFrame.LoadGameListener.actionPerformed (
            ActionEvent e )
```

Ha megnyomjuk a gombot, elindítja a játékot a megfelelő Save objektumtól kapott játékelemeket tartalmazó listával, meghívja a game objektum startLoadGame metódusát.

6.11.4. Adattagok dokumentációja

6.11.4.1. serial

```
final int engine.MenuFrame.LoadGameListener.serial  [private]
```

Annak a mentésnek az index eleme, amiért ez a gomb felel.

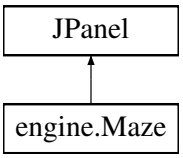
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.12. engine.Maze osztályreferencia

Ez az osztály a játék lelke.

Az engine.Maze osztály származási diagramja:



Osztályok

- class Step

  Ez a subclass felel azért, hogy a Timer minden tick-jére meghívódjanak a megfelelő metódusok.

Publikus tagfüggvények

- Maze (int FRAME_COLUMN, int FRAME_ROW, int BLOCK, GameFrame gameFrame) throws Unsupported↩ AudioFileException, IOException, LineUnavailableException

  Az osztály konstruktora.
- void stopTimer ()

  A függvény megállítjaa Timer-t.
- void newGame (String name)

  Ez a metódus hívódik meg, ha új játékot indítunk.
- void loadGame (ArrayList< Elements > list)

  Ez a metódus tölti be az elmentett játékot.
- void saveGame () throws IOException

  Ez a függvény hívódik meg a game játékkeretből, ha valaki megnyomja az Escape-et, majd meghívja a keret save↩ Game metódusát, és átadja neki a pályaelemeket tartalmazó tárolót, és a PacMan name attribútumát.
- void movePacMan (int direction)

  Ez a metódus hívódik meg, amikor a játékos megnyom egy nyilat, és átállítja a PacMan irányát.
- void paintComponent (Graphics g)

  Ez a függvény a JPanel egyik metódusa, amit felüldefiniálok itt.

Statikus publikus attribútumok

- static int BLOCK_SIZE

  A mindenkori blokkméret, pixelben.
- static int MAZE_COLUMN

  A pálya szélessége, blokkokban.
- static int MAZE_ROW

  A pálya magassága, blokkokban.

Privát tagfüggvények

- void initGhosts ()

  Ez a függvény minden játékinduláskor inicializálja a szellemeket, és hozzáadja pket a heterogén kolleckiójukhoz.
- void endGame ()

  Ez a függvény hívódik meg, ha véget ért a játék.
- void ghostHitCalc ()

  Ez a föggvény megnézi, hogy ütközik-e a PacMan és valamelyik szellem.
- void hitByGhost ()

  Ez a metódus hívódik meg akkor, ha a koordináták alapján a PacMan ütközik egy szellemmel.
- void powerShouldStart ()

  Ez a metódus dönti el, hogy elindítsa-e az extra pontok funkciót.
- void eatPowerPelletCalc ()

  Ha aktív az extra pont funkció, akkor ez a függvény hívódik meg minden tick-re, hogy ellenőrizze, megevett-e a PacMan egy PowerPellet-et.
- void powerPelletEaten ()

  Ha a PacMan megevett egy PowerPellet-et, akkor meghívódik ez a metódus, és növeli a PacMan pontszámát, életét, illetve ezek értékét frissíti az eredményjelzőn.
- void newLevelRequired ()

  A függvény megvizsgálja minden tick-re, hogy kell-e szintet növelni.
- void eatDot ()

  Ez a metódus hívódik meg minden tick-re, és ellenőrzi, hogy a PacMan-megevett-e egy pontot.
- void pelletEaten ()

  Ha a Pacman megevett egy pontot, meghívja ezt a függvényt, ami növeli a pontszámát, és frissíti az eredményjelzőt.
- void setGhostsCenter ()

  Ez a metódus az összes szellemet a kiinduló helyzetbe helyezi.
- void initDots ()

  Minden új játéknál, illetve szintlépésnél meghívódik ez a függvény, és létrehozza az új pontokat a játékmezőn, hozáadva őket a pályaelemek heterogén kollekciójához.
- void initGamePicture ()

  Amennyiben a gamePictre attribútum null lenne, ez a metódus inicializálja azt.

Privát attribútumok

- BufferedImage gamePicture = null

  A pálya megjeleníéséhez használt Bufferedimage attribútum, hogy ne szaggasson a kirajzolás.
- PacMan player

  A játékos által irányított PacMan példány.
- final GameFrame game

  A játék kerete, amiben fut.
- final Timer timer

Egy Swing Timer, aminek köszönhetően szisztematikusan időről időre meghívódhatnak a megfelelő metódusok.

- int powerTime

  Egy int változó, hogy meddig legyenek aktívak az extra pontok.
- boolean powerPelletActive = false

  Egy boolean változó, hogy éppen aktívak-e az extra pontok.
- int powerCalcTime

  Egy int változó, hogy milyen időközönként nézze meg a játék, hogy aktiválja-e az extra pontokat.
- final short[ ] walls

  oszlop,sor = x,y
- final short[ ] dots

  oszlop,sor = x,y
- ArrayList< Elements > mazeElements = new ArrayList<>()

  A pályaelemeket (Elements) tartalmazó heterogén kollekció
- ArrayList< PowerPellet > powerPellets = new ArrayList<>()

  Az extra pontokat tartalmazó tároló
- final ArrayList< Ghost > ghosts = new ArrayList<>()

  A szellemek heterogén kollekciója.
- final Effects p = new Effects()

  A játék effektjeit tartalmazó objektum.

Statikus privát attribútumok

- static final int fps = 120

  Egy "fps" változó, ami nagyjából annak felel meg, hogy milyen képfrissítéssel fut a játék.

6.12.1. Részletes leírás

Ez az osztály a játék lelke.

Gyakorlatilag ez az osztály tekinthető a játék engine-jének. Ez kommunikál a különböző JFrame elemekkel (játék, menü), itt történik a kirajzolás, itt történik a szellemekkel való ütközés, a pontok, vagy extra pontok megevése, az élet, a pontszám, a szint változtatása. Minden, ami a játék logikájához kapcsolódik. Leszármazik a JPanel ősből.

6.12.2. Konstruktorok és destruktorok dokumentációja

6.12.2.1. Maze()

```
engine.Maze.Maze (
            int FRAME_COLUMN,
            int FRAME_ROW,
            int BLOCK,
            GameFrame gameFrame ) throws UnsupportedAudioFileException, IOException, Line↩
UnavailableException
```

Az osztály konstruktora.

Átveszi a pálya méreteit, inicializálja a szellemeket, beállítja a JPanel típusához tartozó paramétereket, majd a powerCalcTime attribútumot beállítja az fps alapján, és a Timer-t is paraméterezi késleltetés, és frekvencia szempontjából.

Paraméterek

FRAME_COLUMN	a játékkeret szélessége, blokkokban
FRAME_ROW	a játékkeret magassága, blokkokban
BLOCK	a játékkeret blokkmérete, pixelben
gameFrame	a játék GameFrame kerete

Kivételek

UnsupportedAudioFileException	
IOException	
LineUnavailableException	

6.12.3. Tagfüggvények dokumentációja

6.12.3.1. eatDot()

```
void engine.Maze.eatDot ( )  [private]
```

Ez a metódus hívódik meg minden tick-re, és ellenőrzi, hogy a PacMan-megevett-e egy pontot.

Teszi mindezt a PacMan és a pontok koordinátáinak összehasonlításával.

6.12.3.2. eatPowerPelletCalc()

```
void engine.Maze.eatPowerPelletCalc ( )  [private]
```

Ha aktív az extra pont funkció, akkor ez a függvény hívódik meg minden tick-re, hogy ellenőrizze, megevett-e a PacMan egy PowerPellet-et.

Ehhez a két objektum koordinátáit használja egy bizonyos tűréshatárral.

6.12.3.3. endGame()

```
void engine.Maze.endGame ( )  [private]
```

Ez a függvény hívódik meg, ha véget ért a játék.

Ez két okból történhet, vagyeljutott a PacMan a harmadik szintre, és minden pontot megevett, vagy elfogyott minden élete. Ezután ez meghívja a game keret addToLeaderboard metódusát átadva a PacMan-t, és leállítja a Timer-t.

6.12.3.4. ghostHitCalc()

```
void engine.Maze.ghostHitCalc ( )  [private]
```

Ez a föggvény megnézi, hogy ütközik-e a PacMan és valamelyik szellem.

Ezt úgy teszi, hogy a PacMan és a szellem koordinátáit összeveti, és ha elegendően kicsi a két objektum közti távolság, meghívja a hitByGhost metódust.

6.12.3.5. hitByGhost()

```
void engine.Maze.hitByGhost ( )  [private]
```

Ez a metódus hívódik meg akkor, ha a koordináták alapján a PacMan ütközik egy szellemmel.

Csökkenti egyel a PacMan életét, majd frissíti ezt az értéket az eredményjelzől, megállítja a Timer-t, ellenőrzi, hogy maradt-e még élete a PacMan-nek, és ha nem, akkor hívja az endGame metódust, ha igen, akkor a kiindulópontjára helyez minden szellemet, és a PacMan-t, majd újraindítja a Timer-t.

6.12.3.6. initDots()

```
void engine.Maze.initDots ( )  [private]
```

Minden új játéknál, illetve szintlépésnél meghívódik ez a függvény, és létrehozza az új pontokat a játékmezőn, hozáadva őket a pályaelemek heterogén kollekciójához.

6.12.3.7. initGamePicture()

```
void engine.Maze.initGamePicture ( )  [private]
```

Amennyiben a gamePictre attribútum null lenne, ez a metódus inicializálja azt.

6.12.3.8. initGhosts()

```
void engine.Maze.initGhosts ( )  [private]
```

Ez a függvény minden játékinduláskor inicializálja a szellemeket, és hozzáadja pket a heterogén kolleckiójukhoz.

6.12.3.9. loadGame()

```
void engine.Maze.loadGame (
            ArrayList< Elements > list )
```

Ez a metódus tölti be az elmentett játékot.

Ehhez az átvett pályaelemeket tartalmazó listát beállítja az új heterogén kollekciónak, majd minden pályaelemnek átállítja a blokkméretét, ha netán az előző mentésnél ez más érték lett volna, középre helyezi a PacMan-t, majd frissíti az eredményjelző megfelelő részeit, a kiindulópontjaikba helyezi a szellemeket, újrarajzolja a pályát, és elindítja a Timer-t.

Paraméterek

list	a mentett pályaelemeket tartalmazó lista

6.12.3.10. movePacMan()

```
void engine.Maze.movePacMan (
            int direction )
```

Ez a metódus hívódik meg, amikor a játékos megnyom egy nyilat, és átállítja a PacMan irányát.

Paraméterek

direction	az új irány

6.12.3.11. newGame()

```
void engine.Maze.newGame (
            String name )
```

Ez a metódus hívódik meg, ha új játékot indítunk.

Új PacMan objektumot hoz létre, és új heterogén kollekciót a pályaelemeknek. Ezután reseteli az eredményjelzőn a pontszámot, életszámot, és szintszámot, létrehozza az új pontokat, beállítja a PacMan kezdőirányát, a szellemeket középre helyezi, újrarajzolja a pályát, és elindítja a Timer-t.

Paraméterek

name	a játékos által begépelt név

6.12.3.12. newLevelRequired()

```
void engine.Maze.newLevelRequired ( )  [private]
```

A függvény megvizsgálja minden tick-re, hogy kell-e szintet növelni.

Ezt úgy teszi, hogy a mazeElements változó darabszámát viozsgálja. Ha ez 1, azaz csak a PacMan maradt benne, akkor minden pont elfogyott, tehát vagy megnyerte a PacMan a játékot, ha már a harmadik szinten van, vagy pedig szintet lép. Ilyenkor újrainicializálja a pontokat, és frissíti az eredményjelzőt.

6.12.3.13. paintComponent()

```
void engine.Maze.paintComponent (
            Graphics g )
```

Ez a függvény a JPanel egyik metódusa, amit felüldefiniálok itt.

Ez inicializálja a gamePicture attribútumot, ha erre szükség van, majd elkéri tőle a grafikai sémáját, beállítja rajta az Anialiasing paramétereket, és rárajzolja a megfelelő elemeket. Ilyen például a falak, a kapu, a még meglévő pontok, a PacMan, és a szellemek. Ezután a gamePicture attribútumot megjeleníti a saját JPanelen. A pálya dinamikus részeinek kirajzolása Extra pontok

Pályaelemek (pontok, PacMan)

Szellemek

6.12.3.14. pelletEaten()

```
void engine.Maze.pelletEaten ( )  [private]
```

Ha a Pacman megevett egy pontot, meghívja ezt a függvényt, ami növeli a pontszámát, és frissíti az eredményjelzőt.

6.12.3.15. powerPelletEaten()

```
void engine.Maze.powerPelletEaten ( )  [private]
```

Ha a PacMan megevett egy PowerPellet-et, akkor meghívódik ez a metódus, és növeli a PacMan pontszámát, életét, illetve ezek értékét frissíti az eredményjelzőn.

6.12.3.16. powerShouldStart()

```
void engine.Maze.powerShouldStart ( )  [private]
```

Ez a metódus dönti el, hogy elindítsa-e az extra pontok funkciót.

Ehhez emgvizsgálja, hogy éppen érvényben van-e ez a funkció, és ha igen, akkor csökkenti a powerTime változó értékét, és ha ez eléri a nullát, akkor tudja, hogy lejárt az aktivitásra szánt idő, ezért törli az extra pontokat a pályáról. Ha nem volt éppen aktív ez a funkció, akkor csökkenti a powerCalcTime változó értékét, majd ha ez eléri a nullát, akkor generál egy random számot 0-1 között, és ha ez kisebb, mint 0.5, akkor elindítja a funkciót. A funkció 10 másodpercig aktív, és 5 másodpercenként nézi meg a pálya, hogy elindítsa-e.

6.12.3.17. saveGame()

```
void engine.Maze.saveGame ( ) throws IOException
```

Ez a függvény hívódik meg a game játékkeretből, ha valaki megnyomja az Escape-et, majd meghívja a keret save↩ Game metódusát, és átadja neki a pályaelemeket tartalmazó tárolót, és a PacMan name attribútumát.

Kivételek

IOException	

6.12.3.18. setGhostsCenter()

```
void engine.Maze.setGhostsCenter ( )  [private]
```

Ez a metódus az összes szellemet a kiinduló helyzetbe helyezi.

6.12.3.19. stopTimer()

```
void engine.Maze.stopTimer ( )
```

A függvény megállítjaa Timer-t.

ERre azért van szükség, mert néha a játékkeretnek kell ezt megtennie.

6.12.4. Adattagok dokumentációja

6.12.4.1. BLOCK_SIZE

```
int engine.Maze.BLOCK_SIZE  [static]
```

A mindenkori blokkméret, pixelben.

6.12.4.2. dots

```
final short [] engine.Maze.dots  [private]
```

Kezdő érték:

```
= {
            2,1,3,1,4,1,5,1,6,1,7,1,9,1,10,1,11,1,12,1,13,1,14,1,
            1,2,4,2,7,2,9,2,12,2,15,2,
            1,3,4,3,7,3,9,3,12,3,15,3,
            1,4,2,4,3,4,4,4,5,4,6,4,7,4,8,4,9,4,10,4,11,4,12,4,13,4,14,4,15,4,
            1,5,4,5,6,5,10,5,12,5,15,5,
            1,6,2,6,3,6,4,6,6,6,7,6,9,6,10,6,12,6,13,6,14,6,15,6,
            4,7,7,7,9,7,12,7,
            4,8,6,8,7,8,8,8,9,8,10,8,12,8,
            4,9,6,9,10,9,12,9,
            4,10,5,10,6,10,10,10,11,10,12,10,
            4,11,6,11,10,11,12,11,
            4,12,6,12,7,12,8,12,9,12,10,12,12,12,
            4,13,6,13,10,13,12,13,
            1,14,2,14,3,14,4,14,5,14,6,14,8,14,10,14,11,14,12,14,13,14,14,14,15,14,
            2,15,6,15,7,15,9,15,10,15,14,15,
            2,16,4,16,5,16,6,16,10,16,11,16,12,16,14,16,
            1,17,2,17,3,17,4,17,6,17,7,17,9,17,10,17,12,17,13,17,14,17,15,17,
            1,18,7,18,9,18,15,18,
            2,19,3,19,4,19,5,19,6,19,7,19,8,19,9,19,10,19,11,19,12,19,13,19,14,19
    }
```

oszlop,sor = x,y

A pálya pontjainak kezdőhelyeinek koordinátáit (oszlop;sor) tartalmazó short tömb

6.12.4.3. fps

```
final int engine.Maze.fps = 120  [static], [private]
```

Egy "fps" változó, ami nagyjából annak felel meg, hogy milyen képfrissítéssel fut a játék.

6.12.4.4. game

```
final GameFrame engine.Maze.game  [private]
```

A játék kerete, amiben fut.

6.12.4.5. gamePicture

```
BufferedImage engine.Maze.gamePicture = null  [private]
```

A pálya megjeleníéséhez használt Bufferedimage attribútum, hogy ne szaggasson a kirajzolás.

6.12.4.6. ghosts

```
final ArrayList<Ghost> engine.Maze.ghosts = new ArrayList<>()  [private]
```

A szellemek heterogén kollekciója.

6.12.4.7. MAZE_COLUMN

```
int engine.Maze.MAZE_COLUMN  [static]
```

A pálya szélessége, blokkokban.

6.12.4.8. MAZE_ROW

```
int engine.Maze.MAZE_ROW  [static]
```

A pálya magassága, blokkokban.

6.12.4.9. mazeElements

```
ArrayList<Elements> engine.Maze.mazeElements = new ArrayList<>()  [private]
```

A pályaelemeket (Elements) tartalmazó heterogén kollekció

6.12.4.10. p

```
final Effects engine.Maze.p = new Effects()  [private]
```

A játék effektjeit tartalmazó objektum.

6.12.4.11. player

```
PacMan engine.Maze.player  [private]
```

A játékos által irányított PacMan példány.

6.12.4.12. powerCalcTime

```
int engine.Maze.powerCalcTime  [private]
```

Egy int változó, hogy milyen időközönként nézze meg a játék, hogy aktiválja-e az extra pontokat.

6.12.4.13. powerPelletActive

```
boolean engine.Maze.powerPelletActive = false  [private]
```

Egy boolean változó, hogy éppen aktívak-e az extra pontok.

6.12.4.14. powerPellets

```
ArrayList<PowerPellet> engine.Maze.powerPellets = new ArrayList<>()  [private]
```

Az extra pontokat tartalmazó tároló

6.12.4.15. powerTime

```
int engine.Maze.powerTime  [private]
```

Egy int változó, hogy meddig legyenek aktívak az extra pontok.

6.12.4.16. timer

```
final Timer engine.Maze.timer  [private]
```

Egy Swing Timer, aminek köszönhetően szisztematikusan időről időre meghívódhatnak a megfelelő metódusok.

6.12.4.17. walls

```
final short [] engine.Maze.walls  [private]
```

Kezdő érték:

```
= {
            0,0,1,0,2,0,3,0,4,0,5,0,6,0,7,0,8,0,9,0,10,0,11,0,12,0,13,0,14,0,15,0,16,0,
            0,1,8,1,16,1,
            0,2,2,2,3,2,5,2,6,2,8,2,10,2,11,2,13,2,14,2,16,2,
            0,3,2,3,3,3,5,3,6,3,8,3,10,3,11,3,13,3,14,3,16,3,
            0,4,16,4,
            0,5,2,5,3,5,5,5,7,5,8,5,9,5,11,5,13,5,14,5,16,5,
            0,6,5,6,8,6,11,6,16,6,
            0,7,1,7,2,7,3,7,5,7,6,7,8,7,10,7,11,7,13,7,14,7,15,7,16,7,
            3,8,5,8,11,8,13,8,
            0,9,1,9,2,9,3,9,5,9,7,9,8,9,9,9,11,9,13,9,14,9,15,9,16,9,
            0,10,7,10,9,10,16,10,
            0,11,1,11,2,11,3,11,5,11,7,11,8,11,9,11,11,11,13,11,14,11,15,11,16,11,
            3,12,5,12,11,12,13,12,
            0,13,1,13,2,13,3,13,5,13,7,13,8,13,9,13,11,13,13,13,14,13,15,13,16,13,
            0,14,7,14,9,14,16,14,
            0,15,1,15,3,15,4,15,5,15,11,15,12,15,13,15,15,15,16,15,
            0,16,1,16,3,16,7,16,8,16,9,16,13,16,15,16,16,16,
            0,17,5,17,8,17,11,17,16,17,
            0,18,2,18,3,18,4,18,5,18,6,18,8,18,10,18,11,18,12,18,13,18,14,18,16,18,
            0,19,16,19,
            0,20,1,20,2,20,3,20,4,20,5,20,6,20,7,20,8,20,9,20,10,20,11,20,12,20,13,20,14,20,15,20,16,20
    }
```

oszlop,sor = x,y

A pálya falainak koordinátáit (oszlop;sor) tartalmazó short tömb

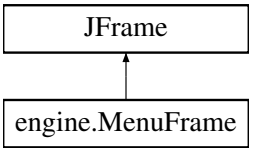
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Maze.java

6.13. engine.MenuFrame osztályreferencia

Ez az osztály felel a menü megvalósításáért.

Az engine.MenuFrame osztály származási diagramja:



Osztályok

- class LoadButtonListener

  Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a LOAD GAME Jbuttont, akkor megjelenjenek a mentéseket vezérlő gombok.
- class LoadGameListener

  Ez az ActionListener subclass felel azért, hogy a kiválasztott mentett állás betöltődjön, és induljon a játék.
- class NewGameButtonListener

  Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a NEW GAME Jbuttont, akkor megjelenjenek az új játék kezdését vezérlő elemet.
- class QuitListener

  Ez az ActionListener subclass felel azért, hogy a játékból ki lehessen lépni.
- class StartButtonListener

  Ez az ActionListener subclass felel azért, hogy az új játék elinduléjon, a játékos által megadott névvel.

Publikus tagfüggvények

- MenuFrame (int column, int row, int block, GameFrame g) throws IOException, ClassNotFoundException

  Az osztály konstruktora.
- void addToLeaderList (PacMan p)

  Ez a metódus felel azért, hogy a játék végeztével a ranglista frissüljön az új delikvenssel.
- void addToSaveStates (ArrayList< Elements > list, String n) throws IOException

  Ez egy viszonylag bonyolult metódus arra, hogy az adott játékállást elmentsük.

Csomag attribútumok

- GameFrame game

  Az a GameFrame objektum, ahol a játék fut.
- ArrayList< Save > saves

  A Save osztály mentéseket tartalmazó tároló
- ArrayList< PacMan > leaderBoard

  A ranglistához PacMan-eket tartalmazó tároló
- ArrayList< JButton > button = new ArrayList<>()

  A könnyebb kezelhetőség érdekében a betöltési gombokat így tárolom.

Privát tagfüggvények

- void initMenuBar ()

  Ebben a metódusban inicializálódik a mmenüsor, és a benne lévő gombokba is itt kerülnek bele az ActionListener implementációk.
- void initLoadBoard () throws IOException, ClassNotFoundException

  Ebben a metódusban inicializálódik a mentéseket, és az új játk indításához szükséges Swing elemeket tartalmazó panel, itt adódik a Start gombhoz a megfelelő ActionListener.
- void initSaveFolder ()

  Ez a metódus ellenőrzi, hogy van-e már a user.dir könyvtárban savegame mappa, és ha nincs, létrehozza.
- void initSaves (JPanel panel) throws IOException, ClassNotFoundException

  Ebben a metódusban inicializálódnak szerializálással a mentések.
- void initLeaderBoard () throws IOException, ClassNotFoundException

  Ebben a metódusban inicializálódik a ranglistát tároló JTable, majd meghívja az initLeadrList metódust.
- void initLeaderList () throws IOException, ClassNotFoundException

Ez a metódus inicializálja a ranglista tárolóját szerializálással.

- void saveLeaderList () throws IOException

  Ez a metódus felel azért, hogy a ranglista szerializálással egy fájlba kerüljön.
- void saveGameStates () throws IOException

  Ebben a metódusban zajlik a mentések szerializálása fájlba.
- void visibleLoadButtons (boolean visible)

  Ez a metódus mind a hat betöltési gombnak állítja a láthatóságát.
- void removeSaveGame (String s)

  Ez a metódus a játékmenet végével "eltűnteti" a játékot befejező PacMan-hez tartozó eddigi mentést, ha az létezik.

Privát attribútumok

- final JButton newGame = new JButton("NEW GAME")

  Az új játék felület indításáért felelős gomb.
- final JButton quit = new JButton("QUIT")

  A kilépésért felelős gomb.
- final JPanel menu = new JPanel()

  A menüsor JPanel-je.
- final JPanel loadPanel = new JPanel()

  A betöltés rész JPanel-je.
- JTable board

  A ranglista JTable táblázata.
- final JLabel name = new JLabel("PLEASE ENTER YOUR NAME")

  Új játékhot a név beírására felszólító felirat.
- final JTextField nameTextField = new JTextField(20)

  Az a JTextField, ahova beírhatja a nevét a játékos.
- final JButton start = new JButton("START")

  A játék kezdéséhez használható gomb.
- final JButton load = new JButton("LOAD GAME")

  A betöltés felület megjelenítéséhez használt JPanel.
- final int FRAME_COLUMN

  A menü szélessége, blokkokban.
- final int FRAME_ROW

  A menü magassága, blokkokban.
- final int BLOCK_SIZE

  A menü kirajzolásához tartozó blokkméret, pixelben.

6.13.1. Részletes leírás

Ez az osztály felel a menü megvalósításáért.

Itt jön létre minden a menüben látható elem, itt töltődnek be a mentett állások, és itt mentődnek el, a ranglistával együtt. A JFrame ősből származik le, és ActionListener subclass-okat használ a gombok érzékeléséhez.

6.13.2. Konstruktorok és destruktorok dokumentációja

6.13.2.1. MenuFrame()

```
engine.MenuFrame.MenuFrame (
            int column,
            int row,
            int block,
            GameFrame g ) throws IOException, ClassNotFoundException
```

Az osztály konstruktora.

Lényegében a JFrame őstől kapott elemeket állítja be, illetve inicializálja a menu, load, és leaderboard paneleket.

Paraméterek

column	a menü szélessége, blokkokban
row	a menü magassába, blokkokban
block	a menü blokkmérete, pixelben
g	az a GameFrame objektum, amit használ a játék

Kivételek

IOException	
ClassNotFoundException	

6.13.3. Tagfüggvények dokumentációja

6.13.3.1. addToLeaderList()

```
void engine.MenuFrame.addToLeaderList (
            PacMan p )
```

Ez a metódus felel azért, hogy a játék végeztével a ranglista frissüljön az új delikvenssel.

Egyszerre csak tizen lehetnek a ranglistán. A metódus rendezi is a listát a PMComparator osztály segítségével.

Paraméterek

p	az új PacMan objektum

6.13.3.2. addToSaveStates()

```
void engine.MenuFrame.addToSaveStates (
            ArrayList< Elements > list,
            String n ) throws IOException
```

Ez egy viszonylag bonyolult metódus arra, hogy az adott játékállást elmentsük.

Megnézi, hogy van-e ugyanolyan name attribútumot tartalmazó Save obektum a saves tárolóban, ha van, akkor azt felülírja, ha nincs, akkor megnézi, hogy van-e szabad ("empty") Save objektum benne, és oda ment, ha pedig nincs se üres, se ugyanolyan name attribútmú, akkor a Save objektumok kora alapján a legöregebbet felülírja. Ehhhez természetesen megfelelő pillanatokban növeli a saves-ben található objektumok korát.

Paraméterek

list	az elmentendő játékelemeket tartalmazó lista
n	az elmentetndő játékos neve

Kivételek

IOException	

6.13.3.3. initLeaderBoard()

```
void engine.MenuFrame.initLeaderBoard ( ) throws IOException, ClassNotFoundException  [private]
```

Ebben a metódusban inicializálódik a ranglistát tároló JTable, majd meghívja az initLeadrList metódust.

Kivételek

IOException	
ClassNotFoundException	

6.13.3.4. initLeaderList()

```
void engine.MenuFrame.initLeaderList ( ) throws IOException, ClassNotFoundException  [private]
```

Ez a metódus inicializálja a ranglista tárolóját szerializálással.

Kivételek

IOException	
ClassNotFoundException	

6.13.3.5. initLoadBoard()

```
void engine.MenuFrame.initLoadBoard ( ) throws IOException, ClassNotFoundException  [private]
```

Ebben a metódusban inicializálódik a mentéseket, és az új játk indításához szükséges Swing elemeket tartalmazó panel, itt adódik a Start gombhoz a megfelelő ActionListener.

Kivételek

IOException	
ClassNotFoundException	

6.13.3.6. initMenuBar()

```
void engine.MenuFrame.initMenuBar ( )  [private]
```

Ebben a metódusban inicializálódik a mmenüsor, és a benne lévő gombokba is itt kerülnek bele az ActionListener implementációk.

6.13.3.7. initSaveFolder()

```
void engine.MenuFrame.initSaveFolder ( )  [private]
```

Ez a metódus ellenőrzi, hogy van-e már a user.dir könyvtárban savegame mappa, és ha nincs, létrehozza.

6.13.3.8. initSaves()

```
void engine.MenuFrame.initSaves (
            JPanel panel ) throws IOException, ClassNotFoundException  [private]
```

Ebben a metódusban inicializálódnak szerializálással a mentések.

Itt töltödik fel a mentéseket tartalmazó tároló, és itt alakulnak meg az ezek eléréséhez szükséges JButton-ok is.

Paraméterek

panel	a mentések megjelenítéséért felelős panel

Kivételek

IOException	
ClassNotFoundException	

6.13.3.9. removeSaveGame()

```
void engine.MenuFrame.removeSaveGame (
            String s )  [private]
```

Ez a metódus a játékmenet végével "eltűnteti" a játékot befejező PacMan-hez tartozó eddigi mentést, ha az létezik.

Paraméterek

s	a PacMan neve

6.13.3.10. saveGameStates()

```
void engine.MenuFrame.saveGameStates ( ) throws IOException  [private]
```

Ebben a metódusban zajlik a mentések szerializálása fájlba.

Kivételek

IOException	

6.13.3.11. saveLeaderList()

```
void engine.MenuFrame.saveLeaderList ( ) throws IOException  [private]
```

Ez a metódus felel azért, hogy a ranglista szerializálással egy fájlba kerüljön.

Kivételek

IOException	

6.13.3.12. visibleLoadButtons()

```
void engine.MenuFrame.visibleLoadButtons (
            boolean visible )  [private]
```

Ez a metódus mind a hat betöltési gombnak álltíja a láthatóságát.

Paraméterek

visible	a láthatóság, true - látható, false - nem látható

6.13.4. Adattagok dokumentációja

6.13.4.1. BLOCK_SIZE

```
final int engine.MenuFrame.BLOCK_SIZE  [private]
```

A menü kirajzolásához tartozó blokkméret, pixelben.

6.13.4.2. board

```
JTable engine.MenuFrame.board  [private]
```

A ranglista JTable táblázata.

6.13.4.3. button

```
ArrayList<JButton> engine.MenuFrame.button = new ArrayList<>()  [package]
```

A könnyebb kezelhetőség érdekében a betöltési gombokat így tárolom.

6.13.4.4. FRAME_COLUMN

```
final int engine.MenuFrame.FRAME_COLUMN  [private]
```

A menü szélessége, blokkokban.

6.13.4.5. FRAME_ROW

```
final int engine.MenuFrame.FRAME_ROW  [private]
```

A menü magassága, blokkokban.

6.13.4.6. game

```
GameFrame engine.MenuFrame.game  [package]
```

Az a GameFrame objektum, ahol a játék fut.

6.13.4.7. leaderBoard

```
ArrayList<PacMan> engine.MenuFrame.leaderBoard  [package]
```

A ranglistához PacMan-eket tartalmazó tároló

6.13.4.8. load

```
final JButton engine.MenuFrame.load = new JButton("LOAD GAME")  [private]
```

A betöltés felület megjelenítéséhez használt JPanel.

6.13.4.9. loadPanel

```
final JPanel engine.MenuFrame.loadPanel = new JPanel()  [private]
```

A betöltés rész JPanel-je.

6.13.4.10. menu

```
final JPanel engine.MenuFrame.menu = new JPanel()  [private]
```

A menüsor JPanel-je.

6.13.4.11. name

```
final JLabel engine.MenuFrame.name = new JLabel("PLEASE ENTER YOUR NAME")  [private]
```

Új játékhot a név beírására felszólító felirat.

6.13.4.12. nameTextField

```
final JTextField engine.MenuFrame.nameTextField = new JTextField(20)  [private]
```

Az a JTextField, ahova beírhatja a nevét a játékos.

6.13.4.13. newGame

```
final JButton engine.MenuFrame.newGame = new JButton("NEW GAME")  [private]
```

Az új játék felület indításáért felelős gomb.

6.13.4.14. quit

```
final JButton engine.MenuFrame.quit = new JButton("QUIT")  [private]
```

A kilépésért felelős gomb.

6.13.4.15. saves

```
ArrayList<Save> engine.MenuFrame.saves  [package]
```

A Save osztály mentéseket tartalmazó tároló

6.13.4.16. start

```
final JButton engine.MenuFrame.start = new JButton("START")  [private]
```

A játék kezdéséhez használható gomb.

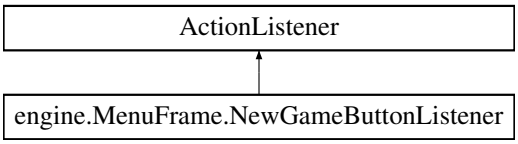
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.14. engine.MenuFrame.NewGameButtonListener osztályreferencia

Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a NEW GAME Jbuttont, akkor megjelenjenek az új játék kezdését vezérlő elemet.

Az engine.MenuFrame.NewGameButtonListener osztály származási diagramja:



Publikus tagfüggvények

- void actionPerformed (ActionEvent e)

  Eltűntei a betöltési gombokat, és megjelníti az új játékhoz szükséges elemeket.

6.14.1. Részletes leírás

Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a NEW GAME Jbuttont, akkor megjelenjenek az új játék kezdését vezérlő elemet.

6.14.2. Tagfüggvények dokumentációja

6.14.2.1. actionPerformed()

```
void engine.MenuFrame.NewGameButtonListener.actionPerformed (
            ActionEvent e )
```

Eltűntei a betöltési gombokat, és megjelníti az új játékhoz szükséges elemeket.

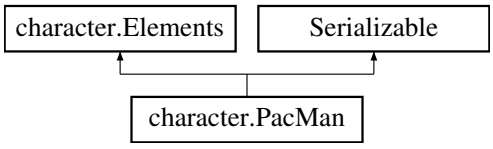
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.15. character.PacMan osztályreferencia

Ez az osztály valósítja meg a PacMan karaktert, és annak működését.

A character.PacMan osztály származási diagramja:



Publikus tagfüggvények

- PacMan (String n, int block, short[ ] w)

  Az osztály konstruktora.
- void draw (Graphics2D frame_g, Object c)

  Minden egyes Timer tick-re ez a metódus hívódik meg, ami kirajzolja a pályára a PacMan-t.
- void changeBlockSize (int d)

  A függvény megváltoztatja az objektumban eltárolt blokkméretet, és ezzel együtt középre helyezi a PacMan-t, így az a következő kirajzoláskor már alkalmazkodik a megváltozott pályaméretekhez.
- void move ()

  Ez a metódus felel a PacMan mozgatásáért.
- void setDirection (int d)

  Ez a metódus beállítja a PacMan irányát.
- void increaseLevel ()

  Ez a metódus egyel növeli a PacMan szintjét, majd középre pozícionálja a PacMan-t.
- void increasePoints (boolean powerTime)

  Ez a metódus annak megfelelően növeli a PacMan pontszámát, hogy éppen aktívak-e az extra PowerPelletek, illetve, hogy hányas szinten van a PacMan.
- String getPlayerName ()

  Getter a PacMan name attribútumára.
- long getPoints ()

  Getter a PacMan pontszámára.
- int getLevel ()

  Getter a PacMan szintjére.
- int getLives ()

  Getter a PacMan élteinek számára.
- int getX ()

  Getter az X koordinátára.
- int getY ()

  Getter az Y koordinátára.
- void changeLivesNumber (boolean increase)

  Ez a metódus a PacMan életeinek számát változtatja.
- void setCentre ()

  Ez a metódus a kezdőpozícióba helyezi a PacMan-t, ami a 8;15 (X;Y) koordinátájú blokk, illetve a kezdőirányt állítja be, és kiszámítja a szemközti fal koordinátáit.

Publikus attribútumok

- int direction = 1

  A karakter iránya, kezdetben 1, azaz balra.

Privát tagfüggvények

- void measureOpposite ()

  Ez talán az osztály legkomplexebb metódusa.

Privát attribútumok

- int X

  A karakter X koordinátája, pixelben.
- int Y

  A karakter Y koordinátája, pixelben.
- int oppositeX

  A karakterrel szemközti fal X koordinátája, pixelben.
- int oppositeY

  A karakterrel szemközti fal Y koordinátája, pixelben.
- int BLOCK_SIZE

  A mindenkori pályablokk méret.
- int level

  A karakter szintjének száma.
- int lives

  A karakter életeinek száma.
- long points

  A karakter pontjainak száma.
- final String name

  A játékos által kezdéskor begépelt név.
- int pacFrameState
- int frameNum = 0
- final short[ ] walls

  A játékban fixen elhelyezkedő falak koordinátáit tároló tömb.

Statikus privát attribútumok

- static final long serialVersionUID = 30L

  Szerializáláshoz nem kötelezően szükséges egyéni azonosító.

6.15.1. Részletes leírás

Ez az osztály valósítja meg a PacMan karaktert, és annak működését.

Itt történik annak az ellenőrzése, hogy a PacMan ne menjen át a falon, ahogy az is itt valósul meg, hogy mozogjon a karakter. Ez az osztály szerializálódik a mentésekben. A Pellet osztállyal közös őse az Elements osztály.

6.15.2. Konstruktorok és destruktorok dokumentációja

6.15.2.1. PacMan()

```
character.PacMan.PacMan (
            String n,
            int block,
            short[ ] w )
```

Az osztály konstruktora.

Paraméterek

n	a játékos által begépelt név
block	az objektum inicializálásakor érvényben lévő blokkméret
w	a fix falak koordinátáit tároló tömb a hívótól

6.15.3. Tagfüggvények dokumentációja

6.15.3.1. changeBlockSize()

```
void character.PacMan.changeBlockSize (
            int d )
```

A függvény megváltoztatja az objektumban eltárolt blokkméretet, és ezzel együtt középre helyezi a PacMan-t, így az a következő kirajzoláskor már alkalmazkodik a megváltozott pályaméretekhez.

Paraméterek

d	az új blokkméret

Újraimplementált ősök: character.Elements.

6.15.3.2. changeLivesNumber()

```
void character.PacMan.changeLivesNumber (
            boolean increase )
```

Ez a metódus a PacMan életeinek számát változtatja.

Ha az increase paraméter értéke true, akkor növeli az életek számát 1-gyel, de csak amennyiben az kevesebb volt, mint 3. Ha a paraméter értéke false, akkor csökkenti egyel az életek számát.

Paraméterek

increase	növeljen, vagy csökkentsen a fügvény, true - növeljen, false - ccsökkentsen

6.15.3.3. draw()

```
void character.PacMan.draw (
            Graphics2D frame_g,
            Object c )
```

Minden egyes Timer tick-re ez a metódus hívódik meg, ami kirajzolja a pályára a PacMan-t.

A PacMannek minden irányához tartozik egy képeket tároló tömb. A pacFrameState attribútum mondja meg, hogy egymás után ennek a tömbnek melyik elemét akarjuk megjeleníteni, a frameNum attribútum pedig azt, hogy ez a megjelenítés hány tick-enként változzon!

Paraméterek

frame_g	- az a Graphics2D elem, amire rajzolunk.
c	- ImageObserver a drawImage metódushoz, az az objektum, ahonnan hívták a draw-t

BALRA

FEL

LE

JOBRRA

Újraimplementált ősök: character.Elements.

6.15.3.4. getLevel()

```
int character.PacMan.getLevel ( )
```

Getter a PacMan szintjére.

Visszatérési érték

a szint

6.15.3.5. getLives()

```
int character.PacMan.getLives ( )
```

Getter a PacMan élteinek számára.

Visszatérési érték

az életek száma

6.15.3.6. getPlayerName()

```
String character.PacMan.getPlayerName ( )
```

Getter a PacMan name attribútumára.

Visszatérési érték

az attribútum értéke

Újraimplementált ősök: character.Elements.

6.15.3.7. getPoints()

```
long character.PacMan.getPoints ( )
```

Getter a PacMan pontszámára.

Visszatérési érték

a pontszám

6.15.3.8. getX()

```
int character.PacMan.getX ( )
```

Getter az X koordinátára.

Visszatérési érték

az X koordináta

Újraimplementált ősök: character.Elements.

6.15.3.9. getY()

```
int character.PacMan.getY ( )
```

Getter az Y koordinátára.

Visszatérési érték

az Y koordináta

Újraimplementált ősök: character.Elements.

6.15.3.10. increaseLevel()

```
void character.PacMan.increaseLevel ( )
```

Ez a metódus egyel növeli a PacMan szintjét, majd középre pozícionálja a PacMan-t.

6.15.3.11. increasePoints()

```
void character.PacMan.increasePoints (
            boolean powerTime )
```

Ez a metódus annak megfelelően növeli a PacMan pontszámát, hogy éppen aktívak-e az extra PowerPelletek, illetve, hogy hányas szinten van a PacMan.

Ha a powerTime paraméter értéke true, akkor 500-as növekszik a pontszám, ha ez false, akkor 1-es szint esetén 20, 2-es szint esetén 40, 3-as szint esetén 80-al nő a pontszám.

Paraméterek

powerTime	éppen aktívak-e a PowerPellet-ek, true - igen, false - nem

6.15.3.12. measureOpposite()

```
void character.PacMan.measureOpposite ( )  [private]
```

Ez talán az osztály legkomplexebb metódusa.

Az osztály konstruktorbáan megadott tömb alapján megkeresi a PacMan mindig adott irány szerinti szemközti falának koordinátáit, majd az oppositeX és oppositeY változókat ezekre a koordinátákra (pixelben) állítja be. A move metódus eszek ellenőrzése alapján mozgatja a PacMan-t. Ha nem talál szemközti falat, akkor -1;-1 koordinátákra állítja az előbbi két változót. Ennél jobban sajnos nem tudtam rövidíteni rajta, mert pl. más probléma megkeresni a balra legközelebb lévő falat, mint a jobbra lévőt. Szemközti fal balra

Szemközti fal felfele

Szemközti fal lefele

Szemközti fal jobbra

6.15.3.13. move()

```
void character.PacMan.move ( )
```

Ez a metódus felel a PacMan mozgatásáért.

Minden Timer tick-re meghívódik, és az adott szint értékével mozgatja a PacMan-t attól függően, hogy milyen irányba (direction: 1 - balra, 3 - le, 2 - fel, 4 - jobbra) áll. Fontos, hogyha a measureOpposite metódus -1;-1 koordinátákat adott vissza, akkor nem mozgatja a PacMant. Illetve akkor sem, hogyha az adott iránynak megfelelő koordináták különbségének abszolót értéke kisebb, mint a pályablokk mérete. Ez biztosítja, hogy a PacMan ne menjen bele a falba. Mozgás balra

Mozgás felfele

Mozgás lefele

Mozgás jobbra

6.15.3.14. setCentre()

```
void character.PacMan.setCentre ( )
```

Ez a metódus a kezdőpozícióba helyezi a PacMan-t, ami a 8;15 (X;Y) koordinátájú blokk, illetve a kezdőirányt állítja be, és kiszámítja a szemközti fal koordinátáit.

6.15.3.15. setDirection()

```
void character.PacMan.setDirection (
            int d )
```

Ez a metódus beállítja a PacMan irányát.

Amikor a játékos megnyom egy nyilat, akkor csak ezt állítja. Az átállítás után újraszámítja a szemközti fal koordinátáit.

Paraméterek

d	az új irány

6.15.4. Adattagok dokumentációja

6.15.4.1. BLOCK_SIZE

```
int character.PacMan.BLOCK_SIZE  [private]
```

A mindenkori pályablokk méret.

Elengedhetetlen a logikák működéséhez

6.15.4.2. direction

```
int character.PacMan.direction = 1
```

A karakter iránya, kezdetben 1, azaz balra.

6.15.4.3. frameNum

```
int character.PacMan.frameNum = 0  [private]
```

6.15.4.4. level

```
int character.PacMan.level  [private]
```

A karakter szintjének száma.

6.15.4.5. lives

```
int character.PacMan.lives  [private]
```

A karakter életeinek száma.

6.15.4.6. name

```
final String character.PacMan.name  [private]
```

A játékos által kezdéskor begépelt név.

6.15.4.7. oppositeX

```
int character.PacMan.oppositeX  [private]
```

A karakterrel szemközti fal X koordinátája, pixelben.

6.15.4.8. oppositeY

```
int character.PacMan.oppositeY  [private]
```

A karakterrel szemközti fal Y koordinátája, pixelben.

6.15.4.9. pacFrameState

```
int character.PacMan.pacFrameState  [private]
```

6.15.4.10. points

```
long character.PacMan.points  [private]
```

A karakter pontjainak száma.

6.15.4.11. serialVersionUID

```
final long character.PacMan.serialVersionUID = 30L  [static], [private]
```

Szerializáláshoz nem kötelezően szükséges egyéni azonosító.

6.15.4.12. walls

```
final short [] character.PacMan.walls  [private]
```

A játékban fixen elhelyezkedő falak koordinátáit tároló tömb.

6.15.4.13. X

```
int character.PacMan.X  [private]
```

A karakter X koordinátája, pixelben.

6.15.4.14. Y

```
int character.PacMan.Y  [private]
```

A karakter Y koordinátája, pixelben.

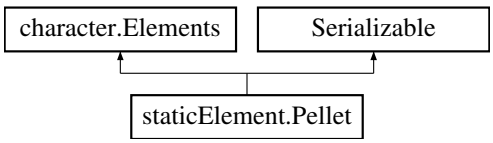
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- PacMan.java

6.16. staticElement.Pellet osztályreferencia

Ez az osztály valósítja meg a játékban a felvehető pontokat.

A staticElement.Pellet osztály származási diagramja:



Publikus tagfüggvények

- Pellet (int x, int y, int b)

  Az osztály konstruktora.
- void changeBlockSize (int d)

  A függvény megváltoztatja az objektumban eltárolt blokkméretet, és ezzel együtt az X,Y koordinátákat, így az következő kirajzoláskor már alkalmazkodik a megváltozott pályaméretekhez.
- int getX ()

  Getter az X koordinátára.
- int getY ()

  Getter az Y koordinátára.
- String getPlayerName ()

  Getter a pellet "nevére".
- void draw (Graphics2D frame_g, Object c)

  Minden egyes Timer tick-re ez a metódus íhvódik meg, ami kirajzolja a pályára a pelletet.

Privát attribútumok

- int X

  Az adot pellet X koordinátája, pixelben.
- int Y

  Az adot pellet Y koordinátája, pixelben.
- int BLOCK_SIZE

  Konstruktorban paraméterként átvett pályablokk méret.

Statikus privát attribútumok

- static final long serialVersionUID = 50L

  Szerializáláshoz nem kötelezően szükséges egyéni azonosító.

6.16.1. Részletes leírás

Ez az osztály valósítja meg a játékban a felvehető pontokat.

Az Elements ősből származik le. Ez az osztály szerializálódik a mentésekben.

6.16.2. Konstruktorok és destruktorok dokumentációja

6.16.2.1. Pellet()

```
staticElement.Pellet.Pellet (
            int x,
            int y,
            int b )
```

Az osztály konstruktora.

Paraméterek

x	az adott objektum X koordinátája
y	az adott objektum Y koordinátája
b	a az objektum létrejöttekor érvényben lévő pályablokk méret

6.16.3. Tagfüggvények dokumentációja

6.16.3.1. changeBlockSize()

```
void staticElement.Pellet.changeBlockSize (
            int d )
```

A függvény megváltoztatja az objektumban eltárolt blokkméretet, és ezzel együtt az X,Y koordinátákat, így az következő kirajzoláskor már alkalmazkodik a megváltozott pályaméretekhez.

Paraméterek

d	az új blokkméret

Újraimplementált ősök: character.Elements.

6.16.3.2. draw()

```
void staticElement.Pellet.draw (
            Graphics2D frame_g,
            Object c )
```

Minden egyes Timer tick-re ez a metódus íhvódik meg, ami kirajzolja a pályára a pelletet.

Paraméterek

frame_g	- az a Graphics2D elem, amire rajzolunk.
c	- ImageObserver a drawImage metódushoz, az az objektum, ahonnan hívták a draw-t

Újraimplementált ősök: character.Elements.

6.16.3.3. getPlayerName()

```
String staticElement.Pellet.getPlayerName ( )
```

Getter a pellet "nevére".

Erre a heterogén kollekció miatt van szükség.

Visszatérési érték

minden esetben 'null'

Újraimplementált ősök: character.Elements.

6.16.3.4. getX()

```
int staticElement.Pellet.getX ( )
```

Getter az X koordinátára.

Visszatérési érték

X értéke

Újraimplementált ősök: character.Elements.

6.16.3.5. getY()

```
int staticElement.Pellet.getY ( )
```

Getter az Y koordinátára.

Visszatérési érték

Y értéke

Újraimplementált ősök: character.Elements.

6.16.4. Adattagok dokumentációja

6.16.4.1. BLOCK_SIZE

```
int staticElement.Pellet.BLOCK_SIZE  [private]
```

Konstruktorban paraméterként átvett pályablokk méret.

6.16.4.2. serialVersionUID

```
final long staticElement.Pellet.serialVersionUID = 50L  [static], [private]
```

Szerializáláshoz nem kötelezően szükséges egyéni azonosító.

6.16.4.3. X

```
int staticElement.Pellet.X  [private]
```

Az adot pellet X koordinátája, pixelben.

6.16.4.4. Y

```
int staticElement.Pellet.Y  [private]
```

Az adot pellet Y koordinátája, pixelben.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Pellet.java

6.17. Pictures osztályreferencia

A játékban felhasznált képeket tároló osztály.

6.17.1. Részletes leírás

A játékban felhasznált képeket tároló osztály.

Ez az osztály felel a játékban használt képek és hangok tárolásáért. Három enumerációt használ, melyek segítségével könnyen lehet a PacMan, illetve az egyéb pályaelemek, szellemek, falak képeit, illetve hangjait tárolni, majd elérni. Minden kép a picture_context nevű mappában foglal helyet, minden hang, pedig az audio_context mappában. A képek 90%-át magamnak készítettem, a maradékot (WALL, GATE, HEART) pedig a Google-el találtam.

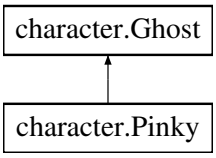
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Effects.java

6.18. character.Pinky osztályreferencia

Pinky szellemet megvalósító osztály.

A character.Pinky osztály származási diagramja:



Publikus tagfüggvények

- Pinky (int BLOCK_SIZE)

  Az osztály konstruktora.
- void setCentre ()

  Ez a metúdus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.
- void draw (Graphics2D frame_g, Object c)

  Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.
- void move (int level)

  Ez a metúdus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Privát tagfüggvények

- void step (int level)

  Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Privát attribútumok

- final short[ ] route

  A szellem útvonalát tároló objektum.
- int direction

  Az irány, amerre a szellem éppen halad.
- int callNum
- int iter

  Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

További örökölt tagok

6.18.1. Részletes leírás

Pinky szellemet megvalósító osztály.

6.18.2. Konstruktorok és destruktorok dokumentációja

6.18.2.1. Pinky()

```
character.Pinky.Pinky (
            int BLOCK_SIZE )
```

Az osztály konstruktora.

Paraméterek

BLOCK_SIZE	az objektum létrehozásakor érvényben lévő pályablokk méret

6.18.3. Tagfüggvények dokumentációja

6.18.3.1. draw()

```
void character.Pinky.draw (
            Graphics2D frame_g,
            Object c )
```

Ez a metódus rajzolja ki Pinky szellem képét a megfelelő koordinátákra, felüldefiniálva az ős ugyanilyen absztrakt metódusát.

Paraméterek

frame_g	Graphics2D objektum, amire rajzolunk
c	ImageObserver objektum, rendszerint a hívó objektum

Újraimplementált ősök: character.Ghost.

6.18.3.2. move()

```
void character.Pinky.move (
            int level )
```

Ez a metódus felel annak eldöntéséért, hogy a szellem elért-e már egy adott pályablokk végére, és ha igen, akkor merre folytassa útját, a route attribútum szerint.

Itt hívódik a step metódus is. Ezt a heterogén kollekciónak köszönhetően egy for ciklussal a Maze objektum el tudja végezni. A 36 azért kell, mert a központi helyről 18 blokkot megy a körkörös mozgás kezdőpontjáig, viszont nem akarjuk, hogy visszamenjen a középső "zárkába".

Paraméterek

level	hányas szinten tart éppen a pacman

Újraimplementált ősök: character.Ghost.

6.18.3.3. setCentre()

```
void character.Pinky.setCentre ( )
```

Ez a metódus a kiindulási pontjába helyezi a szellemet, visszaállítja a kezdőirányt, és nullázza a hívásszámot, és az iterátor értékét.

Újraimplementált ősök: character.Ghost.

6.18.3.4. step()

```
void character.Pinky.step (
            int level )  [private]
```

Ez a metódus minden hívására annyival növeli a direction attribútum szerinti irányba a szellem sebességét, amekkora szinten tart a PacMan.

Paraméterek

level	a PacMan szintje

6.18.4. Adattagok dokumentációja

6.18.4.1. callNum

```
int character.Pinky.callNum  [private]
```

6.18.4.2. direction

```
int character.Pinky.direction  [private]
```

Az irány, amerre a szellem éppen halad.

6.18.4.3. iter

```
int character.Pinky.iter  [private]
```

Ennek a változónak a feladata, hogy a szellem tudja, hogy hányadik lépését hajtja végre egy adott blokkon belül.

Ezen változó segítségével iterál végig az útvonalon a szellem.

6.18.4.4. route

```
final short [] character.Pinky.route  [private]
```

Kezdő érték:

```
= {

       8,9,8,8,9,8,10,8,10,9,10,10,10,11,10,12,10,13,10,14,10,15,10,16,11,16,12,16,12,17,13,17,14,17,15,17,
            15,18,15,19,14,19,13,19,12,19,11,19,10,19,9,19,
            9,18,9,17,10,17,10,16,11,16,12,16,12,17,13,17,14,17,
            14,16,14,15,14,14,13,14,12,14,12,13,12,12,12,11,12,10,11,10,10,10,
            10,11,10,12,10,13,10,14,10,15,10,16,11,16,12,16,
            12,17,13,17,14,17,15,17
    }
```

A szellem útvonalát tároló objektum.

X,Y koordináta párokat tárol, amik az oszlopszám, sorszám mintát követik. Fontos: ezek nem pixel koordináták, hanem blokk koordináták.

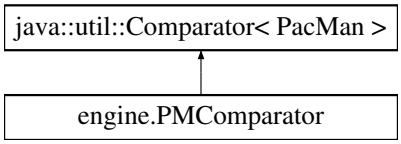
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Pinky.java

6.19. engine.PMComparator osztályreferencia

PacMan kompatárot a ranglistához.

Az engine.PMComparator osztály származási diagramja:



Publikus tagfüggvények

- int compare (PacMan o1, PacMan o2)

  A metódus összehasonlít két PacMan-t azok pontszáma alapján.

6.19.1. Részletes leírás

PacMan kompatárot a ranglistához.

Ezen osztályon keresztül tudunk összehasonlítani két PacMan-t, ehhez a Comparator interfészt valósítja meg.

6.19.2. Tagfüggvények dokumentációja

6.19.2.1. compare()

```
int engine.PMComparator.compare (
            PacMan o1,
            PacMan o2 )
```

A metódus összehasonlít két PacMan-t azok pontszáma alapján.

A csökkenő sorrendnek megfelelően hasonlít össze.

Paraméterek

o1	a bal oldali PacMan
o2	a jobb oldali PacMan

Visszatérési érték

negatív, ha o2 kisebb, mint o1; 0, ha egyenlőek; pozitív, ha 02 nagyobb, mint o1

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- PMComparator.java

6.20. staticElement.PowerPellet osztályreferencia

Ez az osztály valósítja meg a játékban felvehető extra pontokat.

Publikus tagfüggvények

- PowerPellet (int x, int y, int b)

  Az osztály konstruktora.
- void draw (Graphics2D frame_g, Object c)

  A metódus segítségével jeleníti meg az objektum a képét a paraméterként átvett bufferelt képre utaló Graphics2D objektummal.
- int getX ()

  Getter az X koordinátára.
- int getY ()

  Getter az Y koordinátára.

Privát attribútumok

- final int X

  Az objektum képének X koordinátája pixelben.
- final int Y

  Az objektum képének Y koordinátája pixelben.
- final int BLOCK_SIZE

  A játékban a létrehozáskor használatos pályablokk méret.

6.20.1. Részletes leírás

Ez az osztály valósítja meg a játékban felvehető extra pontokat.

Noha funkcionalitásban szinte ugyanazt tudja mint a Pellet osztály, a tárolás eltérősége miatt nem lenne hasznosabb, ha abból származna le. A Pellet osztállyal ellentétben, itt nem szükséges a changeBlockSize metódus, ugyanis ezek az objektumok mindig a játék futása során jönnek létre, tehát mindig az aktuális pályablokkméretet fogják használni.

6.20.2. Konstruktorok és destruktorok dokumentációja

6.20.2.1. PowerPellet()

```
staticElement.PowerPellet.PowerPellet (
            int x,
            int y,
            int b )
```

Az osztály konstruktora.

Paraméterek

x	az X koordináta értéke
y	az Y koordináta értéke
b	a játékban a létrehozáskor használatos pályablokk méret

6.20.3. Tagfüggvények dokumentációja

6.20.3.1. draw()

```
void staticElement.PowerPellet.draw (
            Graphics2D frame_g,
            Object c )
```

A metódus segítségével jeleníti meg az objektum a képét a paraméterként átvett bufferelt képre utaló Graphics2D objektummal.

Paraméterek

frame_g	Graphics2D objektum, aminek a segítségével kirajzolható a kép
c	ImageObserver objektum. Megfelel a hívó objektumnak.

6.20.3.2. getX()

```
int staticElement.PowerPellet.getX ( )
```

Getter az X koordinátára.

Visszatérési érték

az X koordináta

6.20.3.3. getY()

```
int staticElement.PowerPellet.getY ( )
```

Getter az Y koordinátára.

Visszatérési érték

az Y koordináta

6.20.4. Adattagok dokumentációja

6.20.4.1. BLOCK_SIZE

```
final int staticElement.PowerPellet.BLOCK_SIZE  [private]
```

A játékban a létrehozáskor használatos pályablokk méret.

6.20.4.2. X

```
final int staticElement.PowerPellet.X  [private]
```

Az objektum képének X koordinátája pixelben.

6.20.4.3. Y

```
final int staticElement.PowerPellet.Y  [private]
```

Az objektum képének Y koordinátája pixelben.

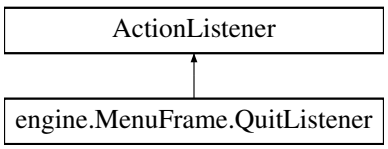
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- PowerPellet.java

6.21. engine.MenuFrame.QuitListener osztályreferencia

Ez az ActionListener subclass felel azért, hogy a játékból ki lehessen lépni.

Az engine.MenuFrame.QuitListener osztály származási diagramja:



Publikus tagfüggvények

- void actionPerformed (ActionEvent e)

  Ez a függvény menti el kilépéskor a mentéseket, illetve a ranglistát tartalmazó tárolókat, majd bezárja a programot.

6.21.1. Részletes leírás

Ez az ActionListener subclass felel azért, hogy a játékból ki lehessen lépni.

6.21.2. Tagfüggvények dokumentációja

6.21.2.1. actionPerformed()

```
void engine.MenuFrame.QuitListener.actionPerformed (
            ActionEvent e )
```

Ez a függvény menti el kilépéskor a mentéseket, illetve a ranglistát tartalmazó tárolókat, majd bezárja a programot.

Játékállások, azaz a Save ojbketumok mentése

A ranglista mentése

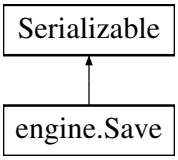
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.22. engine.Save osztályreferencia

Ez az osztály felel a mentés megvalósításáért.

Az engine.Save osztály származási diagramja:



Publikus tagfüggvények

- Save (String n)
  Az osztály konstrukora.
- ArrayList< Elements > getList ()
  Getter az objektumban eltárolt pályaelemeket (Elements) tartalmazó listára.
- void giveList (ArrayList< Elements > l, String n)
  Amikor mentésre kerül a sor, egy pályaelemeket tartalmazó listát adunk át az adott Save objektumnak.
- void incrementAge ()
  A metódus növeli az objektum age változójának értékét.
- void setName (String s)
  Setter az osztály name attribútumára.
- int getAge ()
  Getter a korra.
- String getName ()
  Getter a name attribútumra.

Privát attribútumok

- ArrayList< Elements > list

  A játékelemeket tartalmazó lista.
- int age

  A Save objektum "kora".
- String name

  A játékos neve, akinél elmentették az állást.

Statikus privát attribútumok

- static final long serialVersionUID = 40L

  Szerializáláshoz nem kötelezően szükséges egyéni azonosító

6.22.1. Részletes leírás

Ez az osztály felel a mentés megvalósításáért.

Ez egy szerializálható osztály, ami eltárol minden oplyan Elemnts típusú pályaelemet, ami a játék folytatása szempontjából releváns, plusz, a játékos nevét is, hogy visszatöltéskor könnyebben tudja azonosítani a játékmenetet. Az osztály két fájlt használ, ezek a gamestate és leaderboard nevű fájlok, a savegame mappában. Ha a mappa nem található indításkor, a program létrehozza azt. A gamestate fájlba szerializálódnak a mentések, a leaderboard fájlba pedig a ranglista.

6.22.2. Konstruktorok és destruktorok dokumentációja

6.22.2.1. Save()

```
engine.Save.Save (
            String n )
```

Az osztály konstrukora.

Paraméterek

n	Az adott játékos neve

6.22.3. Tagfüggvények dokumentációja

6.22.3.1. getAge()

```
int engine.Save.getAge ( )
```

Getter a korra.

Visszatérési érték

az adott Save objektum kora

6.22.3.2. getList()

```
ArrayList<Elements> engine.Save.getList ( )
```

Getter az objektumban eltárolt pályaelemeket (Elements) tartalmazó listára.

Visszatérési érték

a pályaelemeket tartalmazó lista

6.22.3.3. getName()

```
String engine.Save.getName ( )
```

Getter a name attribútumra.

Visszatérési érték

egy String, ami a játékos által begépelt név volt, amikor a mentés készült

6.22.3.4. giveList()

```
void engine.Save.giveList (
            ArrayList< Elements > l,
            String n )
```

Amikor mentésre kerül a sor, egy pályaelemeket tartalmazó listát adunk át az adott Save objektumnak.

Paraméterek

l	az átadandó lista
n	a játékos neve, aknél a mentés történik

6.22.3.5. incrementAge()

```
void engine.Save.incrementAge ( )
```

A metódus növeli az objektum age változójának értékét.

Ha ez nagyobb a növelés után, mint 6, akkor 1-re állítja.

6.22.3.6. setName()

```
void engine.Save.setName (
            String s )
```

Setter az osztály name attribútumára.

Paraméterek

s	beállítandó String, mint a játékos által begépelt név

6.22.4. Adattagok dokumentációja

6.22.4.1. age

```
int engine.Save.age  [private]
```

A Save objektum "kora".

Ennek értékére akkor van szükség, ha már mind a 6 mentés slot betelt. Ilyenkor ez alapján választ a játék, hogy melyiket írja felül

6.22.4.2. list

```
ArrayList<Elements> engine.Save.list  [private]
```

A játékelemeket tartalmazó lista.

6.22.4.3. name

```
String engine.Save.name  [private]
```

A játékos neve, akinél elmentették az állást.

6.22.4.4. serialVersionUID

```
final long engine.Save.serialVersionUID = 40L  [static], [private]
```

Szerializáláshoz nem kötelezően szükséges egyéni azonosító

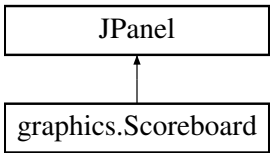
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Save.java

6.23. graphics.Scoreboard osztályreferencia

Ez az osztály felel az eredménytábláért.

A graphics.Scoreboard osztály származási diagramja:



Publikus tagfüggvények

- Scoreboard (int FRAME_COLUMN, int FRAME_ROW, int BLOCK_SIZE)

  Az osztály konstruktora.
- void updatePoints (long newScore)

  Ez a metódus frissíti a paraméterben étvett értékkel a score "kijelzőt".
- void updateLevel (int level)

  Ez a metódus frissíti a paraméterben átvett értékkel a levelNum "kijelzőt".
- void updateLives (int num)

  Ez a metódus frissíti a paraméterben átvett értékkel a lives "kijelzőt".

Privát tagfüggvények

- void initLabels (int BLOCK_SIZE)

  Ez a metódus hívja meg az erdménytábla részeinek inicializálásához szükséges metódusokat.
- void initLives (int BLOCK_SIZE)

  Ez a metódus inicializálja az életek kijelzéséhez szükséges JPanelt.
- void initLevelNum (int BLOCK_SIZE)

  Ez a metódus inicializálja a szint kijelzéséhez szükséges JPanelt.
- void initScore (int BLOCK_SIZE)

  Ez a metódus inicializálja a pontszámw kijelzéséhez szükséges JPanelt.

Privát attribútumok

- JLabel score

  Erre a JLabelre kerülnek a pontok.
- JLabel levelNum

  Erre a JLabelre kerül a szint száma.
- JLabel lives

  Erre a JLabelre kerülnek az életek.

6.23.1. Részletes leírás

Ez az osztály felel az eredménytábláért.

Ebben az osztályban zajlik a pontszámok, életek, szintek kiíratása, és frissítése, ehhez leszármazik a JPanel ősből.

6.23.2. Konstruktorok és destruktorok dokumentációja

6.23.2.1. Scoreboard()

```
graphics.Scoreboard.Scoreboard (
            int FRAME_COLUMN,
            int FRAME_ROW,
            int BLOCK_SIZE )
```

Az osztály konstruktora.

Itt állítja be a paramétereit.

Paraméterek

FRAME_COLUMN	a játékpanel szélessége, blokkokban
FRAME_ROW	a játékpanel magassága, blokkokban
BLOCK_SIZE	a pályablokk mérete

6.23.3. Tagfüggvények dokumentációja

6.23.3.1. initLabels()

```
void graphics.Scoreboard.initLabels (
            int BLOCK_SIZE )  [private]
```

Ez a metódus hívja meg az erdménytábla részeinek inicializálásához szükséges metódusokat.

Paraméterek

BLOCK_SIZE	a pályablokk mérete

6.23.3.2. initLevelNum()

```
void graphics.Scoreboard.initLevelNum (
            int BLOCK_SIZE )  [private]
```

Ez a metódus inicializálja a szint kijelzéséhez szükséges JPanelt.

Paraméterek

BLOCK_SIZE	a pályablokk mérete

6.23.3.3. initLives()

```
void graphics.Scoreboard.initLives (
            int BLOCK_SIZE )  [private]
```

Ez a metódus inicializálja az életek kijelzéséhez szükséges JPanelt.

Paraméterek

BLOCK_SIZE	a pályablokk mérete

6.23.3.4. initScore()

```
void graphics.Scoreboard.initScore (
            int BLOCK_SIZE )  [private]
```

Ez a metódus inicializálja a pontszámw kijelzéséhez szükséges JPanelt.

Paraméterek

BLOCK_SIZE	a pályablokk mérete

6.23.3.5. updateLevel()

```
void graphics.Scoreboard.updateLevel (
            int level )
```

Ez a metódus frissíti a paraméterben átvett értékkel a levelNum "kijelzőt".

Paraméterek

level	hányadik szinten jár a PacMan

6.23.3.6. updateLives()

```
void graphics.Scoreboard.updateLives (
            int num )
```

Ez a metódus frissíti a paraméterben átvett értékkel a lives "kijelzőt".

Paraméterek

num	hány életre frissítsen

6.23.3.7. updatePoints()

```
void graphics.Scoreboard.updatePoints (
            long newScore )
```

Ez a metódus frissíti a paraméterben étvett értékkel a score "kijelzőt".

Paraméterek

newScore	hány pontja van a játékosnak

6.23.4. Adattagok dokumentációja

6.23.4.1. levelNum

```
JLabel graphics.Scoreboard.levelNum  [private]
```

Erre a JLabelre kerül a szint száma.

6.23.4.2. lives

```
JLabel graphics.Scoreboard.lives  [private]
```

Erre a JLabelre kerülnek az életek.

6.23.4.3. score

```
JLabel graphics.Scoreboard.score  [private]
```

Erre a JLabelre kerülnek a pontok.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Scoreboard.java

6.24. graphics.Effects.Sounds felsoroló referencia

A PacMan hangjait tartalmazó enumeráció.

Publikus tagfüggvények

- Clip getClip ()

Publikus attribútumok

- BEGINNING
- DEATH
- COIN
- EXTRA

Privát attribútumok

- AudioInputStream audioIS
- Clip clip

6.24.1. Részletes leírás

A PacMan hangjait tartalmazó enumeráció.

6.24.2. Tagfüggvények dokumentációja

6.24.2.1. getClip()

```
Clip graphics.Effects.Sounds.getClip ( )
```

6.24.3. Adattagok dokumentációja

6.24.3.1. audioIS

```
AudioInputStream graphics.Effects.Sounds.audioIS  [private]
```

6.24.3.2. BEGINNING

```
graphics.Effects.Sounds.BEGINNING
```

6.24.3.3. clip

```
Clip graphics.Effects.Sounds.clip  [private]
```

6.24.3.4. COIN

```
graphics.Effects.Sounds.COIN
```

6.24.3.5. DEATH

```
graphics.Effects.Sounds.DEATH
```

6.24.3.6. EXTRA

```
graphics.Effects.Sounds.EXTRA
```

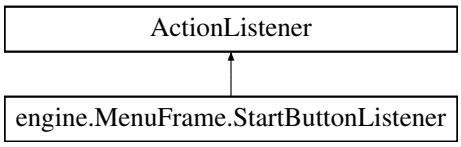
A dokumentáció ehhez az enum-hoz a következő fájl alapján készült:

- Effects.java

6.25. engine.MenuFrame.StartButtonListener osztályreferencia

Ez az ActionListener subclass felel azért, hogy az új játék elinduljon, a játékos által megadott névvel.

Az engine.MenuFrame.StartButtonListener osztály származási diagramja:



Publikus tagfüggvények

- void actionPerformed (ActionEvent e)

  Elindítja az őj játékot, a nameOfPlayer JTextFieldből kiolvasott névvel, meghívja a game objektum startNewGame metódusát.

6.25.1. Részletes leírás

Ez az ActionListener subclass felel azért, hogy az új játék elinduljon, a játékos által megadott névvel.

6.25.2. Tagfüggvények dokumentációja

6.25.2.1. actionPerformed()

```
void engine.MenuFrame.StartButtonListener.actionPerformed (
            ActionEvent e )
```

Elindítja az őj játékot, a nameOfPlayer JTextFieldből kiolvasott névvel, meghívja a game objektum startNewGame metódusát.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- MenuFrame.java

6.26. graphics.Effects.Statics felsoroló referencia

A statikus elemeket tartalmazó enumeráció.

Publikus tagfüggvények

- Image getStatPic ()

  Getter a típusonkétni képre - elérés pl.

Publikus attribútumok

- WALL

  A pályán megjelenő falak képe.
- HEART

  Az eredményjelző tálbán megjelenő szívek képe.
- GATE

  A pályán megjelenő kapu képe.
- DOT

  A pályán megjelenő pontok képe.
- INKY

  A pályán megjelenő Inky szellem képe.
- PINKY

  A pályán megjelenő Pinky szellem képe.
- BLINKY

  A pályán megjelenő Blinky szellem képe.
- CLYDE

  A pályán megjelenő Clyde szellem képe.
- POWER

Privát attribútumok

- Image kep = null

  < *A pályán megjelenő extra pontok képe*

6.26.1. Részletes leírás

A statikus elemeket tartalmazó enumeráció.

Minden elem statikus, ami nem a PacMan megjelenítéséhez tartozó kép. Ezek képtípus szerint érhetőek el az enumerációban.

6.26.2. Tagfüggvények dokumentációja

6.26.2.1. getStatPic()

```
Image graphics.Effects.Statics.getStatPic ( )
```

Getter a típusonkétni képre - elérés pl.

: Picture.Statics.INKY.getStatPic();

Visszatérési érték

a típusnak megfelelő kép

6.26.3. Adattagok dokumentációja

6.26.3.1. BLINKY

`graphics.Effects.Statics.BLINKY`

A pályán megjelenő Blinky szellem képe.

6.26.3.2. CLYDE

`graphics.Effects.Statics.CLYDE`

A pályán megjelenő Clyde szellem képe.

6.26.3.3. DOT

`graphics.Effects.Statics.DOT`

A pályán megjelenő pontok képe.

6.26.3.4. GATE

`graphics.Effects.Statics.GATE`

A pályán megjelenő kapu képe.

6.26.3.5. HEART

`graphics.Effects.Statics.HEART`

Az eredményjelző tálbán megjelenő szívek képe.

6.26.3.6. INKY

`graphics.Effects.Statics.INKY`

A pályán megjelenő Inky szellem képe.

6.26.3.7. kep

`Image graphics.Effects.Statics.kep = null [private]`

< A pályán megjelenő extra pontok képe

A típusonkénti kép attribútum

6.26.3.8. PINKY

`graphics.Effects.Statics.PINKY`

A pályán megjelenő Pinky szellem képe.

6.26.3.9. POWER

`graphics.Effects.Statics.POWER`

6.26.3.10. WALL

`graphics.Effects.Statics.WALL`

A pályán megjelenő falak képe.

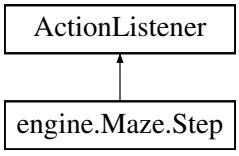
A dokumentáció ehhez az enum-hoz a következő fájl alapján készült:

- Effects.java

6.27. engine.Maze.Step osztályreferencia

Ez a subclass felel azért, hogy a Timer minden tick-jére meghívódjanak a megfelelő metódusok.

Az engine.Maze.Step osztály származási diagramja:



Publikus tagfüggvények

- void actionPerformed (ActionEvent e)

  Ez a függvény hívódik meg minden egyes tick-re.

6.27.1. Részletes leírás

Ez a subclass felel azért, hogy a Timer minden tick-jére meghívódjanak a megfelelő metódusok.

6.27.2. Tagfüggvények dokumentációja

6.27.2.1. actionPerformed()

```
void engine.Maze.Step.actionPerformed (
            ActionEvent e )
```

Ez a függvény hívódik meg minden egyes tick-re.

Mozgatja a PacMan-t a benne beállított irányba, majd mozgat minden szellemet a PacMan szintjének megfelelő sebességgel, ezután ellenőrzi, hogy megevett-e a PacMan egy pontot, majd ellenőrzi, hogy a PacMan ütközött-e szellemel. Ezután eldönti, hogy elindítsa-e az extra pontokat, és ha ezeket elindította, akkor utána mindaddig, amíg abba nem marad az extra pontok megjelenítése, ellenőrzi, hogy a PacMan megevett-e egyet. Ezek után megnézi, hogy kell-e növelni a PacMan szintjén, végezetül meghívja a repaint() metódust, amivel újrarajzolja a megfelelő helyváltoztatásokkal a pályát.

Paraméterek

e	meghívták adott tick-re

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- Maze.java

7. fejezet

Fájlok dokumentációja

7.1. Blinky.java fájlreferencia

Osztályok

- class character.Blinky

  Blinky szellemet megvalósító osztály.

Csomagok

- package character

7.2. Clyde.java fájlreferencia

Osztályok

- class character.Clyde

  Clyde szellemet megvalósító osztály.

Csomagok

- package character

7.3. Effects.java fájlreferencia

Osztályok

- class graphics.Effects
- enum graphics.Effects.Statics

  A statikus elemeket tartalmazó enumeráció.
- enum graphics.Effects.Dynamics

  A dinamikusabb, a PacMan-hez tartalmazó képek enumerációja.
- enum graphics.Effects.Sounds

  A PacMan hangjait tartalmazó enumeráció.

Csomagok

- package graphics

7.4. Elements.java fájlreferencia

Osztályok

- class character.Elements

  Ez az osztály a közös őse a PacMan-nek, és a Pellet-nek.

Csomagok

- package character

7.5. Game.java fájlreferencia

Osztályok

- class engine.Game

  A játéknak összefoglaló keretet adó osztály.

Csomagok

- package engine

7.6. GameFrame.java fájlreferencia

Osztályok

- class engine.GameFrame

  A játék keretét megvalósító osztály.

Csomagok

- package engine

7.7. Ghost.java fájlreferencia

Osztályok

- class character.Ghost

  A szellemek közös absztrakt őse.

Csomagok

- package character

7.8. Inky.java fájlreferencia

Osztályok

- class character.Inky

  Inky szellemet megvalósító osztály.

Csomagok

- package character

7.9. Maze.java fájlreferencia

Osztályok

- class engine.Maze

  Ez az osztály a játék lelke.
- class engine.Maze.Step

  Ez a subclass felel azért, hogy a Timer minden tick-jére meghívódjanak a megfelelő metódusok.

Csomagok

- package engine

7.10. MenuFrame.java fájlreferencia

Osztályok

- class engine.MenuFrame

  Ez az osztály felel a menü megvalósításáért.
- class engine.MenuFrame.LoadButtonListener

  Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a LOAD GAME Jbuttont, akkor megjelenjenek a mentéseket vezérlő gombok.
- class engine.MenuFrame.NewGameButtonListener

  Ez az ActionListener subclass felel azért, hogy amikor a menüsoron megnyomjuk a NEW GAME Jbuttont, akkor megjelenjenek az új játék kezdését vezérlő elemet.
- class engine.MenuFrame.StartButtonListener

  Ez az ActionListener subclass felel azért, hogy az új játék elinduljon, a játékos által megadott névvel.
- class engine.MenuFrame.LoadGameListener

  Ez az ActionListener subclass felel azért, hogy a kiválasztott mentett állás betöltődjön, és induljon a játék.
- class engine.MenuFrame.QuitListener

  Ez az ActionListener subclass felel azért, hogy a játékból ki lehessen lépni.

Csomagok

- package engine

7.11. PacMan.java fájlreferencia

Osztályok

- class character.PacMan

  Ez az osztály valósítja meg a PacMan karaktert, és annak működését.

Csomagok

- package character

7.12. Pellet.java fájlreferencia

Osztályok

- class staticElement.Pellet

  Ez az osztály valósítja meg a játékban a felvehető pontokat.

Csomagok

- package staticElement

7.13. Pinky.java fájlreferencia

Osztályok

- class character.Pinky

  Pinky szellemet megvalósító osztály.

Csomagok

- package character

7.14. PMComparator.java fájlreferencia

Osztályok

- class engine.PMComparator

  PacMan kompatárot a ranglistához.

Csomagok

- package engine

7.15. PowerPellet.java fájlreferencia

Osztályok

- class staticElement.PowerPellet

  Ez az osztály valósítja meg a játékban felvehető extra pontokat.

Csomagok

- package staticElement

7.16. Save.java fájlreferencia

Osztályok

- class engine.Save

  Ez az osztály felel a mentés megvalósításáért.

Csomagok

- package engine

7.17. Scoreboard.java fájlreferencia

Osztályok

- class graphics.Scoreboard

  Ez az osztály felel az eredménytábláért.

Csomagok

- package graphics

Tárgymutató

Cím	**Move PacMan**
Leírás	A játékos a PacMan-t irányítja egy labirintuson keresztül
Aktorok	Player
Forgatókönyv	**1.** A játékos a PacMan-t jobbra, balra, le, vagy felfele irányítja
Alternatív forgatókönyv	**1.A.1**. Ha a PacMan megeszik gy golyócskát pontot kap.
Alternatív forgatókönyv	**1.A.1.A.1.** Ha minden golyócska elfogyott, a PacMan a következő szintre jut.
Alternatív forgatókönyv	**1.B.1.** Ha a PacMan megeszik egy nagyobb golyócskát, extra életet, és plusz pontot kap.
Alternatív forgatókönyv	**1.C.1.** A PacMan meghal, ha szörnnyel ütközik.
Alternatív forgatókönyv	**1.C.1.A.1.**Ha a PacMan minden életét elveszíti, a játéknak vége.

Cím	**View Maze**
Leírás	A játékosmegtekinti a labirintust.
Aktorok	Player
Forgatókönyv	**1.** A rendszer kirajzolja a labirintus aktuális állapotát. **2.** A játékos megtekinti a rendszer aktuális állapotát.

Cím	**Control Monsters**
Leírás	A szörnyek mozognak a labirintusban
Aktorok	Contorller
Forgatókönyv	**1.** A szörnyek balra, le, jobbra, vagy felfele mozognak, a játéklogikának megfelelően.
Alternatív forgatókönyv	**1.A.1.** A PacMan meghal, ha szörnnyel ütközik.

Cím	**Control Power Pellets**
Leírás	Nagyobb golyócskák jelenhetnek meg, vagy tűnhetnek el a labirintusból
Aktorok	Controller
Forgatókönyv	**1.** A Controller új nagy golyócskát helyezhet el a labirintusban **2.** A Controller csökkenti a nagy golyócskák élettartamát
Alternatív forgatókönyv	**2.A.1.** Ha egy nagy golyócska élettartama lejár, eltűnik a labirintusból.