Documentatie Proiect PAOO

Generated by Doxygen 1.9.1

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 Package database

Classes

- class InsertGet

5.2 Package entities

Classes

- class Banana

  Reprezintă un obiect colectabil de tip "banană" în joc.
- class Coconut

  Reprezintă un obiect colectabil de tip "nucă de cocos" în joc.
- class Enemy

  Clasă abstractă de bază pentru toți inamicii din joc.
- class EnemyManager

  Gestionează toți inamicii din joc, inclusiv Nanites, Karagors, Goblins, GoblinBosses și GolemBosses.
- class Entity

  Clasă abstractă de bază pentru toate entitățile din joc (de ex., jucător, inamici, obiecte).
- class Gem

  Reprezintă un obiect de tip "Gem" (piatră prețioasă) în joc.
- class Goblin

  Reprezintă entitatea Goblin în joc.
- class GoblinBoss

  Reprezintă entitatea Goblin Boss în joc.
- class GolemBoss

  Reprezintă entitatea Golem Boss în joc.
- class Karagor

  Reprezintă entitatea Karagor în joc, care poate funcționa ca un boss.
- class Nanite

  Reprezintă entitatea Nanite în joc.
- class Player
- class Projectile

  Reprezintă un proiectil în joc.

5.3 Package gamestates

Classes

- class EnterNameOverlay

  Reprezintă un overlay pentru introducerea numelui jucătorului.
- class GameOverOverlay

  Reprezintă overlay-ul afișat la sfârșitul jocului (Game Over).
- enum Gamestate

  Enumerație ce definește diferitele stări posibile ale jocului.
- class Leaderboard

  Reprezintă starea de joc pentru afișarea clasamentului (Leaderboard).
- class LevelFinishedOverlay

  Reprezintă overlay-ul afișat la finalizarea cu succes a unui nivel.
- class Loadgame

  Reprezintă starea de joc pentru ecranul de încărcare a unui joc salvat.
- class Menu

  Reprezintă starea de joc pentru meniul principal.
- class Options

  Reprezintă starea de joc pentru meniul de opțiuni.
- class Playing

  Reprezintă starea principală de joc (gameplay-ul propriu-zis).
- class State

  Clasă de bază pentru diferitele stări ale jocului (de ex., Meniu, Joc propriu-zis).
- interface Statemethods

  Interfață ce definește metodele comune pe care toate stările de joc (gamestates) trebuie să le implementeze.

5.4 Package inputs

Classes

- class KeyboardInputs

  Gestionează input-ul de la tastatură pentru joc.
- class MouseInputs

  Gestionează input-ul de la mouse pentru joc.

5.5 Package levels

Classes

- class Level

  Reprezintă un nivel individual în joc.
- class LevelFactory

  Clasă de tip Factory responsabilă pentru crearea obiectelor de tip Level.
- class LevelManager

  Gestionează încărcarea, desenarea și tranziția între nivelurile jocului.

5.6 Package main

Classes

- class Game

  Clasa principală a jocului, responsabilă pentru gestionarea stărilor de joc, bucla principală a jocului (game loop) și inițializarea componentelor cheie.
- class GamePanel

  Panoul principal al jocului, extinzând JPanel.
- class GameWindow

  Reprezintă fereastra principală a jocului.
- class Main

  Clasa principală a aplicației, conținând punctul de intrare (metoda main).

5.7 Package ui

Classes

- class LevelButton

  Reprezintă un buton specific pentru selectarea nivelurilor în interfața utilizator.
- class MenuButton

  Reprezintă un buton generic pentru meniurile din interfața utilizator.
- class ProgressBar

  Reprezintă o bară de progres personalizată pentru interfața utilizator.

5.8 Package utilz

Classes

- class Constants

  Clasă ce conține constante globale utilizate în diverse părți ale jocului.
- enum Enemy_Animation_Rows

  Enumerație ce definește rândurile de animație și numărul de cadre pentru inamici.
- enum Gorilla_Animation_rows

  Enumerație ce definește rândurile de animație și numărul de cadre pentru personajul jucător (Gorila).
- class HelpMethods

  Clasă utilitară ce conține metode statice ajutătoare, în principal pentru detecția coliziunilor și poziționarea entităților în cadrul nivelului.
- class LevelProgress

  Clasă simplă de date (POJO - Plain Old Java Object) pentru a stoca informațiile despre progresul jucătorului într-un anumit nivel.
- class LoadSave

  Clasă utilitară responsabilă pentru încărcarea resurselor jocului, cum ar fi imaginile (sprite atlas-uri) și datele nivelurilor din fișiere.

Chapter 6

Class Documentation

6.1 entities.GoblinBoss.ActionState Enum Reference

Enumerare pentru stările de acțiune posibile ale Goblin Boss-ului.

Public Attributes

- IDLE
- DETECTED
- CHASING
- PREPARING_ATTACK
- ATTACKING_MELEE
- ATTACKING_RUN_SLASH
- REPOSITIONING_WALK
- REPOSITIONING_SLIDE
- HURT
- DYING

6.1.1 Detailed Description

Enumerare pentru stările de acțiune posibile ale Goblin Boss-ului.

6.1.2 Member Data Documentation

6.1.2.1 ATTACKING_MELEE

`entities.GoblinBoss.ActionState.ATTACKING_MELEE`

6.1.2.2 ATTACKING_RUN_SLASH

entities.GoblinBoss.ActionState.ATTACKING_RUN_SLASH

6.1.2.3 CHASING

entities.GoblinBoss.ActionState.CHASING

6.1.2.4 DETECTED

entities.GoblinBoss.ActionState.DETECTED

6.1.2.5 DYING

entities.GoblinBoss.ActionState.DYING

6.1.2.6 HURT

entities.GoblinBoss.ActionState.HURT

6.1.2.7 IDLE

entities.GoblinBoss.ActionState.IDLE

6.1.2.8 PREPARING_ATTACK

entities.GoblinBoss.ActionState.PREPARING_ATTACK

6.1.2.9 REPOSITIONING_SLIDE

entities.GoblinBoss.ActionState.REPOSITIONING_SLIDE

6.1.2.10 REPOSITIONING_WALK

`entities.GoblinBoss.ActionState.REPOSITIONING_WALK`

The documentation for this enum was generated from the following file:

- src/entities/GoblinBoss.java

6.2 entities.GolemBoss.ActionState Enum Reference

Enumerare pentru stările de acțiune specifice Golem Boss-ului.

Public Attributes

- IDLE
- DETECTED
- WALKING_TOWARDS_PLAYER
- CHASING
- PREPARING_ATTACK
- ATTACKING_MELEE
- HURT
- DYING

6.2.1 Detailed Description

Enumerare pentru stările de acțiune specifice Golem Boss-ului.

6.2.2 Member Data Documentation

6.2.2.1 ATTACKING_MELEE

`entities.GolemBoss.ActionState.ATTACKING_MELEE`

6.2.2.2 CHASING

`entities.GolemBoss.ActionState.CHASING`

6.2.2.3 DETECTED

entities.GolemBoss.ActionState.DETECTED

6.2.2.4 DYING

entities.GolemBoss.ActionState.DYING

6.2.2.5 HURT

entities.GolemBoss.ActionState.HURT

6.2.2.6 IDLE

entities.GolemBoss.ActionState.IDLE

6.2.2.7 PREPARING_ATTACK

entities.GolemBoss.ActionState.PREPARING_ATTACK

6.2.2.8 WALKING_TOWARDS_PLAYER

entities.GolemBoss.ActionState.WALKING_TOWARDS_PLAYER

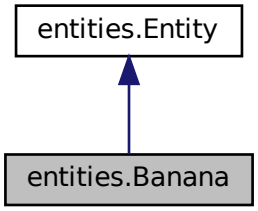
The documentation for this enum was generated from the following file:

- src/entities/GolemBoss.java

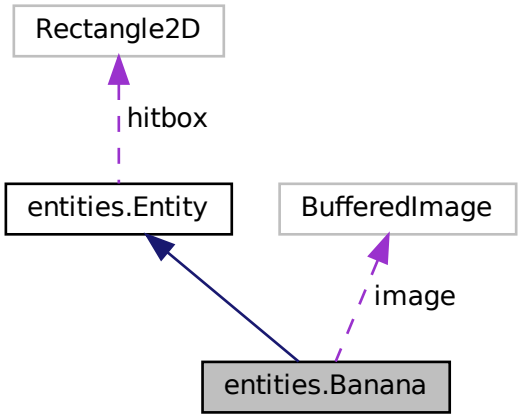
6.3 entities.Banana Class Reference

Reprezintă un obiect colectabil de tip "banană" în joc.

Inheritance diagram for entities.Banana:



Collaboration diagram for entities.Banana:



Public Member Functions

- Banana (float x, float y, int[ ][ ] levelData, BufferedImage image)
  Constructor pentru clasa Banana.
- void update ()
  Actualizează starea bananei.
- void draw (Graphics g, int xLvlOffset)
  Desenează banana pe ecran.
- boolean isActive ()
  Verifică dacă banana este activă.
- void setActive (boolean active)
  Setează starea de activare a bananei.

Protected Member Functions

- void drawHitbox (Graphics g, int xLvlOffset)

  Desenează hitbox-ul bananei (suprascris din Entity).

Private Attributes

- BufferedImage image
- boolean active = true

  Indică dacă banana este activă (poate fi colectată și desenată).
- int[ ][ ] levelData

  Datele nivelului, potențial pentru interacțiuni viitoare cu terenul (momentan neutilizat activ pentru banane).
- float originalY

  Coordonata Y inițială (de bază) pentru animația de plutire.
- float floatSpeed = 0.05f ∗ Game.SCALE

  Viteza animației de plutire.
- float floatAmplitude = 2.0f ∗ Game.SCALE

  Amplitudinea mișcării de plutire (cât de mult se mișcă sus/jos).
- float floatAngle = 0

  Unghiul curent în ciclul de plutire (pentru funcția sinus).
- float originalWidth
- float originalHeight
- float scaleFactor = 1.0f

  Factorul de scalare curent pentru animația de respirație.
- float scaleSpeed = 0.005f

  Viteza cu care se schimbă factorul de scalare.
- float minScale = 0.9f

  Factorul minim de scalare.
- float maxScale = 1.1f

  Factorul maxim de scalare.
- boolean scalingUp = true

  Indică dacă banana se mărește (true) sau se micșorează (false) în animația de respirație.

Additional Inherited Members

6.3.1 Detailed Description

Reprezintă un obiect colectabil de tip "banană" în joc.

Bananele pot oferi bonusuri jucătorului la colectare. Această clasă gestionează animația de plutire și scalare a bananei. Extinde clasa Entity.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 Banana()

```
entities.Banana.Banana (
            float x,
            float y,
            int levelData[][],
            BufferedImage image )
```

Constructor pentru clasa Banana.

Parameters

x	Poziția x inițială a colțului stânga-sus al bananei.
y	Poziția y inițială a colțului stânga-sus al bananei.
levelData	Datele nivelului (momentan neutilizate activ pentru logica bananei).
image	Imaginea (sprite-ul) pentru banană.

6.3.3 Member Function Documentation

6.3.3.1 draw()

```
void entities.Banana.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează banana pe ecran.

Aplică animația de scalare pentru efectul de "respirație".

Parameters

g	Contextul grafic Graphics pe care se va desena.
xLvlOffset	Offset-ul orizontal al nivelului pentru scrolling.

6.3.3.2 drawHitbox()

```
void entities.Banana.drawHitbox (
            Graphics g,
            int xLvlOffset )  [protected]
```

Desenează hitbox-ul bananei (suprascris din Entity).

Folosit pentru depanare.

Parameters

g	Contextul grafic.
xLvlOffset	Offset-ul orizontal al nivelului.

6.3.3.3 isActive()

```
boolean entities.Banana.isActive ( )
```

Verifică dacă banana este activă.

Returns

`true`
dacă banana este activă,
`false`
altfel.

6.3.3.4 setActive()

```
void entities.Banana.setActive (
            boolean active )
```

Setează starea de activare a bananei.

O banană inactivă nu este actualizată sau desenată și nu poate fi colectată.

Parameters

active	Noua stare de activare.

6.3.3.5 update()

```
void entities.Banana.update ( )
```

Actualizează starea bananei.

Gestionează animația de plutire și de "respirație" (scalare). Rulează doar dacă banana este activă și are o imagine validă.

6.3.4 Member Data Documentation

6.3.4.1 active

```
boolean entities.Banana.active = true  [private]
```

Indică dacă banana este activă (poate fi colectată și desenată).

6.3.4.2 floatAmplitude

```
float entities.Banana.floatAmplitude = 2.0f * Game.SCALE  [private]
```

Amplitudinea mișcării de plutire (cât de mult se mișcă sus/jos).

6.3.4.3 floatAngle

```
float entities.Banana.floatAngle = 0  [private]
```

Unghiul curent în ciclul de plutire (pentru funcția sinus).

6.3.4.4 floatSpeed

```
float entities.Banana.floatSpeed = 0.05f * Game.SCALE  [private]
```

Viteza animației de plutire.

6.3.4.5 image

```
BufferedImage entities.Banana.image  [private]
```

6.3.4.6 levelData

```
int [][] entities.Banana.levelData  [private]
```

Datele nivelului, potențial pentru interacțiuni viitoare cu terenul (momentan neutilizat activ pentru banane).

6.3.4.7 maxScale

```
float entities.Banana.maxScale = 1.1f  [private]
```

Factorul maxim de scalare.

6.3.4.8 minScale

```
float entities.Banana.minScale = 0.9f  [private]
```

Factorul minim de scalare.

6.3.4.9 originalHeight

```
float entities.Banana.originalHeight  [private]
```

6.3.4.10 originalWidth

```
float entities.Banana.originalWidth  [private]
```

6.3.4.11 originalY

```
float entities.Banana.originalY  [private]
```

Coordonata Y inițială (de bază) pentru animația de plutire.

6.3.4.12 scaleFactor

```
float entities.Banana.scaleFactor = 1.0f  [private]
```

Factorul de scalare curent pentru animația de respirație.

6.3.4.13 scaleSpeed

```
float entities.Banana.scaleSpeed = 0.005f  [private]
```

Viteza cu care se schimbă factorul de scalare.

6.3.4.14 scalingUp

```
boolean entities.Banana.scalingUp = true  [private]
```

Indică dacă banana se mărește (true) sau se micșorează (false) în animația de respirație.

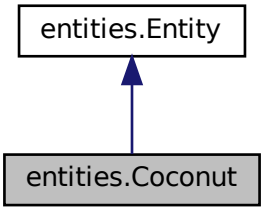
The documentation for this class was generated from the following file:

- src/entities/Banana.java

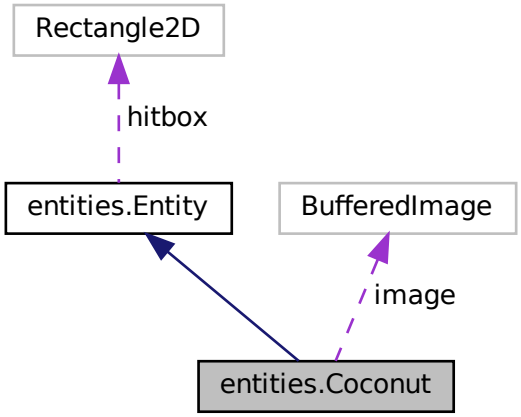
6.4 entities.Coconut Class Reference

Reprezintă un obiect colectabil de tip "nucă de cocos" în joc.

Inheritance diagram for entities.Coconut:



Collaboration diagram for entities.Coconut:



Public Member Functions

- Coconut (float x, float y, int[ ][ ] levelData, BufferedImage image)

  Constructor pentru clasa Coconut.
- void update ()

  Actualizează starea nucii de cocos.
- void draw (Graphics g, int xLvlOffset)

  Desenează nuca de cocos pe ecran.
- boolean isActive ()

  Verifică dacă nuca de cocos este activă.
- void setActive (boolean active)

  Setează starea de activare a nucii de cocos.

Private Attributes

- BufferedImage image
- boolean active = true

  Indică dacă nuca de cocos este activă (poate fi colectată și desenată).
- int[ ][ ] levelData

  Datele nivelului, potențial pentru interacțiuni viitoare cu terenul (momentan neutilizat activ pentru nuci de cocos).
- float originalY

  Coordonata Y inițială (de bază) pentru animația de plutire.
- float floatSpeed = 0.05f ∗ Game.SCALE

  Viteza animației de plutire.
- float floatAmplitude = 2.0f ∗ Game.SCALE

  Amplitudinea mișcării de plutire.
- float floatAngle = 0

  Unghiul curent în ciclul de plutire.
- float originalWidth
- float originalHeight
- float scaleFactor = 1.0f

  Factorul de scalare curent.
- float scaleSpeed = 0.005f

  Viteza de schimbare a factorului de scalare.
- float minScale = 0.9f

  Factorul minim de scalare.
- float maxScale = 1.1f

  Factorul maxim de scalare.
- boolean scalingUp = true

  Indică direcția scalării (mărire sau micșorare).

Additional Inherited Members

6.4.1 Detailed Description

Reprezintă un obiect colectabil de tip "nucă de cocos" în joc.

Nucile de cocos pot fi folosite de jucător, probabil ca proiectile. Această clasă gestionează animația de plutire și scalare a nucii de cocos. Extinde clasa Entity.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 Coconut()

```
entities.Coconut.Coconut (
            float x,
            float y,
            int levelData[][],
            BufferedImage image )
```

Constructor pentru clasa Coconut.

Parameters

x	Poziția x inițială a colțului stânga-sus al nucii de cocos.
y	Poziția y inițială a colțului stânga-sus al nucii de cocos.
levelData	Datele nivelului (momentan neutilizate activ).
image	Imaginea (sprite-ul) pentru nuca de cocos.

6.4.3 Member Function Documentation

6.4.3.1 draw()

```
void entities.Coconut.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează nuca de cocos pe ecran.

Aplică animația de scalare pentru efectul de "respirație".

Parameters

g	Contextul grafic Graphics pe care se va desena.
xLvlOffset	Offset-ul orizontal al nivelului pentru scrolling.

6.4.3.2 isActive()

```
boolean entities.Coconut.isActive ( )
```

Verifică dacă nuca de cocos este activă.

Returns

`true`

dacă este activă,
`false`

altfel.

6.4.3.3 setActive()

```
void entities.Coconut.setActive (
            boolean active )
```

Setează starea de activare a nucii de cocos.

O nucă de cocos inactivă nu este actualizată sau desenată.

Parameters

active	Noua stare de activare.

6.4.3.4 update()

```
void entities.Coconut.update ( )
```

Actualizează starea nucii de cocos.

Gestionează animația de plutire și de "respirație" (scalare). Rulează doar dacă nuca de cocos este activă și are o imagine validă.

6.4.4 Member Data Documentation

6.4.4.1 active

```
boolean entities.Coconut.active = true  [private]
```

Indică dacă nuca de cocos este activă (poate fi colectată și desenată).

6.4.4.2 floatAmplitude

```
float entities.Coconut.floatAmplitude = 2.0f * Game.SCALE  [private]
```

Amplitudinea mișcării de plutire.

6.4.4.3 floatAngle

```
float entities.Coconut.floatAngle = 0  [private]
```

Unghiul curent în ciclul de plutire.

6.4.4.4 floatSpeed

```
float entities.Coconut.floatSpeed = 0.05f * Game.SCALE  [private]
```

Viteza animației de plutire.

6.4.4.5 image

```
BufferedImage entities.Coconut.image  [private]
```

6.4.4.6 levelData

```
int [][] entities.Coconut.levelData  [private]
```

Datele nivelului, potențial pentru interacțiuni viitoare cu terenul (momentan neutilizat activ pentru nuci de cocos).

6.4.4.7 maxScale

```
float entities.Coconut.maxScale = 1.1f  [private]
```

Factorul maxim de scalare.

6.4.4.8 minScale

```
float entities.Coconut.minScale = 0.9f  [private]
```

Factorul minim de scalare.

6.4.4.9 originalHeight

```
float entities.Coconut.originalHeight  [private]
```

6.4.4.10 originalWidth

```
float entities.Coconut.originalWidth  [private]
```

6.4.4.11 originalY

```
float entities.Coconut.originalY  [private]
```

Coordonata Y inițială (de bază) pentru animația de plutire.

6.4.4.12 scaleFactor

```
float entities.Coconut.scaleFactor = 1.0f  [private]
```

Factorul de scalare curent.

6.4.4.13 scaleSpeed

```
float entities.Coconut.scaleSpeed = 0.005f  [private]
```

Viteza de schimbare a factorului de scalare.

6.4.4.14 scalingUp

```
boolean entities.Coconut.scalingUp = true  [private]
```

Indică direcția scalării (mărire sau micșorare).

The documentation for this class was generated from the following file:

- src/entities/Coconut.java

6.5 utilz.Constants Class Reference

Clasă ce conține constante globale utilizate în diverse părți ale jocului.

Classes

- class **Collectibles**

  Constante legate de obiectele colectabile.
- class **Directions**

  Constante pentru direcțiile de mișcare sau orientare.
- class **EnemyConstants**

  Constante legate de inamici.
- class **Tiles**

  Constante legate de tile-urile din nivel.
- class **UI**

  Constante legate de interfața utilizator (UI).

6.5.1 Detailed Description

Clasă ce conține constante globale utilizate în diverse părți ale jocului.

Constantele sunt grupate în clase interne statice pentru o mai bună organizare.

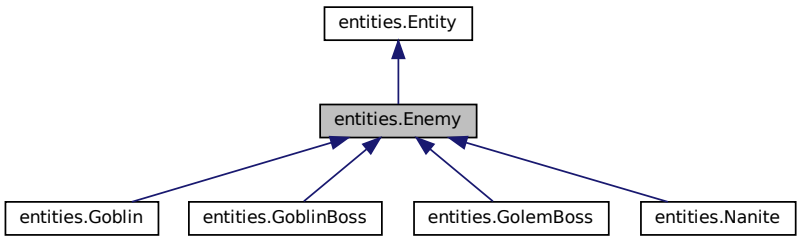
The documentation for this class was generated from the following file:

- src/utilz/Constants.java

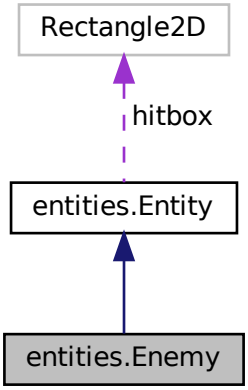
6.6 entities.Enemy Class Reference

Clasă abstractă de bază pentru toți inamicii din joc.

Inheritance diagram for entities.Enemy:



Collaboration diagram for entities.Enemy:



Public Member Functions

- Enemy (float x, float y, int width, int height, int enemyType)

  Constructor pentru clasa abstractă Enemy.
- void update (Rectangle2D.Float playerHitbox)

  Metodă de actualizare generală pentru inamic.
- void drawHitbox (Graphics g, int xLvlOffset)

  Desenează hitbox-ul inamicului dacă flag-ul.
- int getAniIndex ()

  Returnează indexul frame-ului curent al animației.
- int getEnemyState ()

Returnează starea curentă a animației inamicului.

- void setEnemyState (int state)

  Setează starea de animație a inamicului și resetează indexul și contorul animației.
- int getEnemyType ()

  Returnează tipul specific al inamicului.
- Rectangle2D.Float getHitbox ()

  Returnează hitbox-ul inamicului (moștenit din Entity).
- boolean isActive ()

  Verifică dacă inamicul este activ în joc.

Protected Member Functions

- void updateAnimationTick ()

  Actualizează contorul și indexul animației inamicului.

Protected Attributes

- int aniIndex

  Indexul frame-ului curent în animația inamicului.
- int enemyState

  Starea curentă a animației inamicului (de ex., IDLE, RUNNING, ATTACK).
- int enemyType

  Tipul specific al inamicului (de ex., GOBLIN, NANITE).
- int aniTick

  Contor pentru ciclul de animație.
- int aniSpeed = 5

  Viteza animației (numărul de tick-uri de joc per frame de animație).
- boolean drawHitbox = false

  Flag pentru a desena sau nu hitbox-ul inamicului (pentru depanare).
- boolean isActive = true

  Indică dacă inamicul este activ în joc (viu și participă la logică/desenare).

6.6.1 Detailed Description

Clasă abstractă de bază pentru toți inamicii din joc.

Definește proprietăți și comportamente comune pentru inamici, cum ar fi gestionarea animațiilor, starea (activ/inactiv) și tipul inamicului. Extinde clasa Entity.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Enemy()

```
entities.Enemy.Enemy (
            float x,
            float y,
            int width,
            int height,
            int enemyType )
```

Constructor pentru clasa abstractă Enemy.

Parameters

x	Poziția x inițială a inamicului.
y	Poziția y inițială a inamicului.
width	Lățimea inamicului.
height	Înălțimea inamicului.
enemyType	Tipul specific al inamicului.

6.6.3 Member Function Documentation

6.6.3.1 drawHitbox()

```
void entities.Enemy.drawHitbox (
            Graphics g,
            int xLvlOffset )
```

Desenează hitbox-ul inamicului dacă flag-ul.
drawHitbox

este activ. Util pentru depanare.

Parameters

g	Contextul grafic Graphics.
xLvlOffset	Offset-ul orizontal al nivelului pentru scrolling.

6.6.3.2 getAniIndex()

```
int entities.Enemy.getAniIndex ( )
```

Returnează indexul frame-ului curent al animației.

Returns

Indexul frame-ului.

6.6.3.3 getEnemyState()

```
int entities.Enemy.getEnemyState ( )
```

Returnează starea curentă a animației inamicului.

Returns

Starea animației (valoare din utilz.Enemy_Animation_Rows).

6.6.3.4 getEnemyType()

```
int entities.Enemy.getEnemyType ( )
```

Returnează tipul specific al inamicului.

Returns

Tipul inamicului.

6.6.3.5 getHitbox()

```
Rectangle2D.Float entities.Enemy.getHitbox ( )
```

Returnează hitbox-ul inamicului (moștenit din Entity).

Returns

Hitbox-ul.

Reimplemented from entities.Entity.

6.6.3.6 isActive()

```
boolean entities.Enemy.isActive ( )
```

Verifică dacă inamicul este activ în joc.

Returns

`true`
dacă este activ,
`false`
altfel.

Reimplemented in entities.Nanite, and entities.Goblin.

6.6.3.7 setEnemyState()

```
void entities.Enemy.setEnemyState (
            int state )
```

Setează starea de animație a inamicului și resetează indexul și contorul animației.

Parameters

state	Noua stare de animație.

6.6.3.8 update()

```
void entities.Enemy.update (
            Rectangle2D.Float playerHitbox )
```

Metodă de actualizare generală pentru inamic.

Momentan, actualizează doar animația. Subclasele ar trebui să suprascrie această metodă pentru a adăuga logica specifică de comportament (AI).

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru interacțiuni.

Reimplemented in entities.Nanite, and entities.Goblin.

6.6.3.9 updateAnimationTick()

```
void entities.Enemy.updateAnimationTick ( )  [protected]
```

Actualizează contorul și indexul animației inamicului.

Gestionează buclarea animațiilor și tranziția la starea inactivă după animația de moarte.

6.6.4 Member Data Documentation

6.6.4.1 aniIndex

```
int entities.Enemy.aniIndex  [protected]
```

Indexul frame-ului curent în animația inamicului.

6.6.4.2 aniSpeed

```
int entities.Enemy.aniSpeed = 5  [protected]
```

Viteza animației (numărul de tick-uri de joc per frame de animație).

6.6.4.3 aniTick

```
int entities.Enemy.aniTick  [protected]
```

Contor pentru ciclul de animație.

6.6.4.4 drawHitbox

```
boolean entities.Enemy.drawHitbox = false  [protected]
```

Flag pentru a desena sau nu hitbox-ul inamicului (pentru depanare).

6.6.4.5 enemyState

```
int entities.Enemy.enemyState  [protected]
```

Starea curentă a animației inamicului (de ex., IDLE, RUNNING, ATTACK).

Folosește valori din utilz.Enemy_Animation_Rows.

6.6.4.6 enemyType

```
int entities.Enemy.enemyType  [protected]
```

Tipul specific al inamicului (de ex., GOBLIN, NANITE).

Folosește constante definite în subclase sau utilz.Constants.EnemyConstants.

6.6.4.7 isActive

```
boolean entities.Enemy.isActive = true  [protected]
```

Indică dacă inamicul este activ în joc (viu și participă la logică/desenare).

Reimplemented in entities.Nanite, and entities.Goblin.

The documentation for this class was generated from the following file:

- src/entities/Enemy.java

6.7 utilz.Enemy_Animation_Rows Enum Reference

Enumerație ce definește rândurile de animație și numărul de cadre pentru inamici.

Classes

- class **Directions**

  Clasă internă statică ce definește constante pentru direcții.

Public Member Functions

- Enemy_Animation_Rows (int rowIndex)

  Constructor pentru constantele enum care nu au un număr specific de cadre definit (implicit 0).
- Enemy_Animation_Rows (int rowIndex, int frameCount)

  Constructor pentru constantele enum.
- int getRowIndex ()

  Returnează indexul rândului din spritesheet pentru această animație.
- int getFrameCount ()

  Returnează numărul de cadre (frame-uri) pentru această animație.

Public Attributes

- DYING =(0, 15)

  Animația de moarte.
- FALLING_DOWN =(1, 6)

  Animația de cădere.
- HURT =(2, 12)

  Animația de rănire.
- IDLE =(3, 18)

  Animația de repaus (idle), cu clipire.
- IDLE_NO_BLINK =(4, 18)

  Animația de repaus (idle), fără clipire.
- JUMP_LOOP =(5, 6)

  Animația de buclă a săriturii (în aer).
- JUMP_START =(6,6)

  Animația de început a săriturii.
- KICKING =(7, 12)

  Animația de lovitură cu piciorul.
- RUN_SLASING =(8,12)

  Animația de atac cu tăietură în timpul alergării.
- RUN_THROWING =(9,12)

  Animația de aruncare în timpul alergării.
- RUNNING =(10,12)

  Animația de alergare.
- SLASHING_IN_THE_AIR =(11,12)

  Animația de atac cu tăietură în aer.
- SLASHING =(12,12)

  Animația de atac cu tăietură (la sol).

- SLIDING =(13,6)

  Animația de alunecare.
- THROWING_IN_THE_AIR =(14,6)

  Animația de aruncare în aer.
- THROWING =(15,6)

  Animația de aruncare (la sol).
- WALKING =(16, 24)

  Animația de mers.

Private Attributes

- final int rowIndex

  Indexul rândului din spritesheet corespunzător acestei animații.
- final int frameCount

  Numărul de cadre (frame-uri) pentru această animație.

6.7.1 Detailed Description

Enumerație ce definește rândurile de animație și numărul de cadre pentru inamici.

Fiecare constantă corespunde unui rând specific dintr-un spritesheet de animații pentru inamici. Este utilizată pentru a gestiona și accesa secvențele de animație.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Enemy_Animation_Rows() [1/2]

```
utilz.Enemy_Animation_Rows.Enemy_Animation_Rows (
            int rowIndex )
```

Constructor pentru constantele enum care nu au un număr specific de cadre definit (implicit 0).

Parameters

rowIndex	Indexul rândului animației.

6.7.2.2 Enemy_Animation_Rows() [2/2]

```
utilz.Enemy_Animation_Rows.Enemy_Animation_Rows (
            int rowIndex,
            int frameCount )
```

Constructor pentru constantele enum.

Parameters

rowIndex	Indexul rândului animației în spritesheet.
frameCount	Numărul de cadre pentru animație.

6.7.3 Member Function Documentation

6.7.3.1 getFrameCount()

```
int utilz.Enemy_Animation_Rows.getFrameCount ( )
```

Returnează numărul de cadre (frame-uri) pentru această animație.

Returns

Numărul de cadre.

6.7.3.2 getRowIndex()

```
int utilz.Enemy_Animation_Rows.getRowIndex ( )
```

Returnează indexul rândului din spritesheet pentru această animație.

Returns

Indexul rândului.

6.7.4 Member Data Documentation

6.7.4.1 DYING

```
utilz.Enemy_Animation_Rows.DYING =(0, 15)
```

Animația de moarte.

6.7.4.2 FALLING_DOWN

```
utilz.Enemy_Animation_Rows.FALLING_DOWN =(1, 6)
```

Animația de cădere.

6.7.4.3 frameCount

```
final int utilz.Enemy_Animation_Rows.frameCount  [private]
```

Numărul de cadre (frame-uri) pentru această animație.

6.7.4.4 HURT

```
utilz.Enemy_Animation_Rows.HURT =(2, 12)
```

Animația de rănire.

6.7.4.5 IDLE

```
utilz.Enemy_Animation_Rows.IDLE =(3, 18)
```

Animația de repaus (idle), cu clipire.

6.7.4.6 IDLE_NO_BLINK

```
utilz.Enemy_Animation_Rows.IDLE_NO_BLINK =(4, 18)
```

Animația de repaus (idle), fără clipire.

6.7.4.7 JUMP_LOOP

```
utilz.Enemy_Animation_Rows.JUMP_LOOP =(5, 6)
```

Animația de buclă a săriturii (în aer).

6.7.4.8 JUMP_START

```
utilz.Enemy_Animation_Rows.JUMP_START =(6,6)
```

Animația de început a săriturii.

6.7.4.9 KICKING

```
utilz.Enemy_Animation_Rows.KICKING =(7, 12)
```

Animația de lovitură cu piciorul.

6.7.4.10 rowIndex

```
final int utilz.Enemy_Animation_Rows.rowIndex  [private]
```

Indexul rândului din spritesheet corespunzător acestei animații.

6.7.4.11 RUN_SLASING

```
utilz.Enemy_Animation_Rows.RUN_SLASING =(8,12)
```

Animația de atac cu tăietură în timpul alergării.

6.7.4.12 RUN_THROWING

```
utilz.Enemy_Animation_Rows.RUN_THROWING =(9,12)
```

Animația de aruncare în timpul alergării.

6.7.4.13 RUNNING

```
utilz.Enemy_Animation_Rows.RUNNING =(10,12)
```

Animația de alergare.

6.7.4.14 SLASHING

```
utilz.Enemy_Animation_Rows.SLASHING =(12,12)
```

Animația de atac cu tăietură (la sol).

6.7.4.15 SLASHING_IN_THE_AIR

```
utilz.Enemy_Animation_Rows.SLASHING_IN_THE_AIR =(11,12)
```

Animația de atac cu tăietură în aer.

6.7.4.16 SLIDING

```
utilz.Enemy_Animation_Rows.SLIDING =(13,6)
```

Animația de alunecare.

6.7.4.17 THROWING

```
utilz.Enemy_Animation_Rows.THROWING =(15,6)
```

Animația de aruncare (la sol).

6.7.4.18 THROWING_IN_THE_AIR

```
utilz.Enemy_Animation_Rows.THROWING_IN_THE_AIR =(14,6)
```

Animația de aruncare în aer.

6.7.4.19 WALKING

```
utilz.Enemy_Animation_Rows.WALKING =(16, 24)
```

Animația de mers.

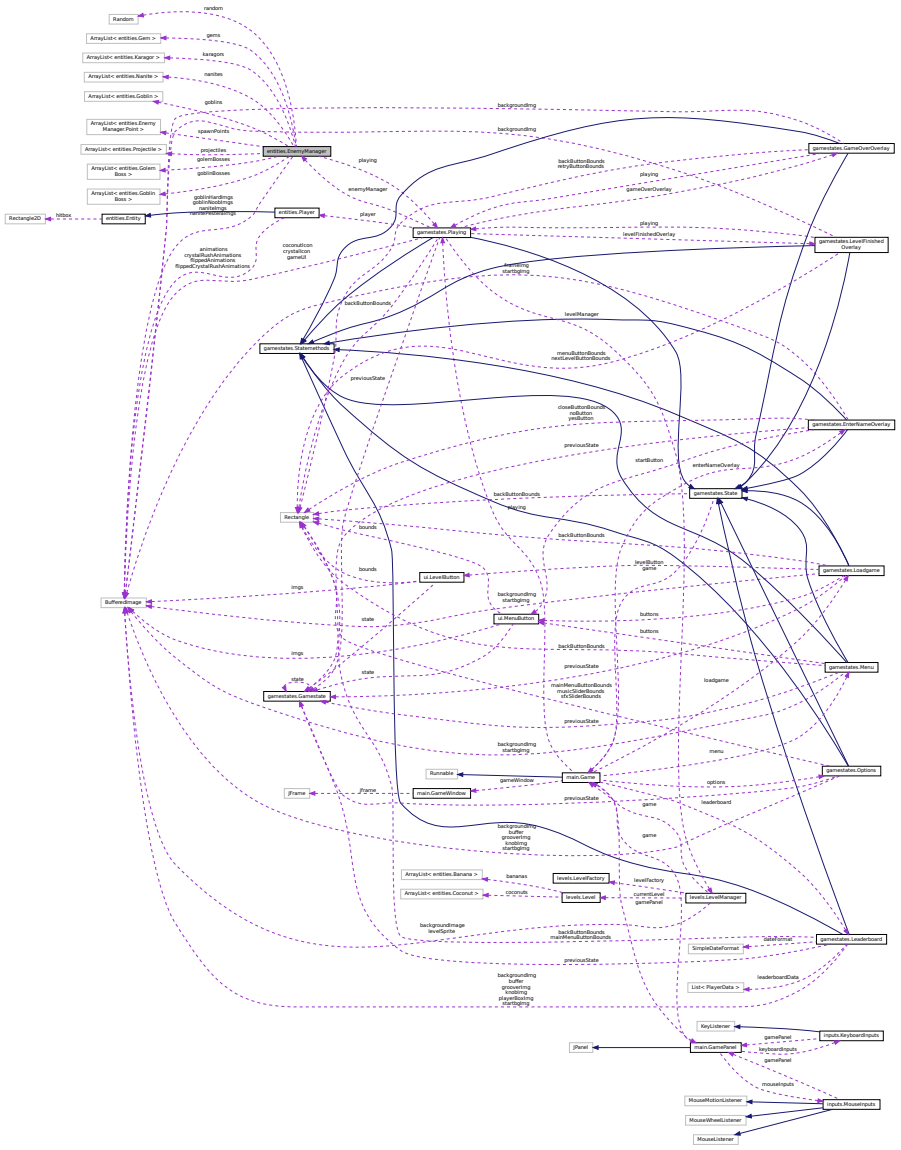
The documentation for this enum was generated from the following file:

- src/utilz/Enemy_Animation_Rows.java

6.8 entities.EnemyManager Class Reference

Gestionează toți inamicii din joc, inclusiv Nanites, Karagors, Goblins, GoblinBosses și GolemBosses.

Collaboration diagram for entities.EnemyManager:



Classes

- class Point

  Clasă internă simplă pentru a stoca informații despre un punct de spawn: coordonatele (x, y) și codul numeric al inamicului care trebuie spawnat.

Public Member Functions

- EnemyManager (Playing playing)

Constructor pentru EnemyManager.

- void update (Rectangle2D.Float playerHitbox)

  Actualizează starea tuturor inamicilor activi, proiectilelor și gem-urilor.
- void draw (Graphics g, int xLvlOffset)

  Desenează toți inamicii activi, proiectilele și gem-urile pe ecran.
- void loadEnemiesFromLevelData (int[ ][ ] levelData, Level level)

  Încarcă inamicii dintr-un nivel pe baza datelor acestuia.
- void spawnAllEnemies ()

  Spawnează toți inamicii identificați în punctele de spawn.
- void resetEnemies ()

  Resetează toți inamicii din nivel.
- int getCurrentLevel ()

  Returnează ID-ul nivelului curent.
- ArrayList< Nanite > getNanites ()

  Returnează lista de Nanites activi.
- ArrayList< Karagor > getKaragors ()

  Returnează lista de Karagors activi.
- ArrayList< Goblin > getGoblins ()

  Returnează lista de Goblini activi.
- ArrayList< GoblinBoss > getGoblinBosses ()

  Returnează lista de GoblinBosses activi.
- ArrayList< Gem > getGems ()

  Returnează lista de Gem-uri active.
- ArrayList< GolemBoss > getGolemBosses ()

  Returnează lista de GolemBosses activi.
- void addProjectile (Projectile p)

  Adaugă un proiectil la lista de proiectile active gestionate de EnemyManager.

Private Member Functions

- void loadEnemyImgs ()

  Încarcă imaginile pentru toți inamicii (Nanites, Goblins).
- BufferedImage[ ][ ] loadSpriteSheet (String path, int spriteWidth, int spriteHeight)

  Încarcă un sprite sheet dintr-o cale specificată și îl împarte în frame-uri individuale.
- void drawGolemBosses (Graphics g, int xLvlOffset)

  Desenează toți GolemBosses activi pe ecran, inclusiv bara lor de viață.
- void drawProjectiles (Graphics g, int xLvlOffset)

  Desenează toate proiectilele active pe ecran.
- void drawGoblinBosses (Graphics g, int xLvlOffset)

  Desenează toți GoblinBosses activi pe ecran, inclusiv bara lor de viață.
- void drawGems (Graphics g, int xLvlOffset)

  Desenează toate gem-urile active pe ecran.
- void drawGoblins (Graphics g, int xLvlOffset)

  Desenează toți Goblinii activi (Noob și Hard) pe ecran, inclusiv bara lor de viață și hitbox-ul (dacă este activat).
- void drawNanites (Graphics g, int xLvlOffset)

  Desenează toți Nanites activi (Jungla și Pestera) pe ecran, inclusiv bara lor de viață și hitbox-ul (dacă este activat).
- void drawKaragors (Graphics g, int xLvlOffset)

  Desenează toți Karagors activi pe ecran, inclusiv bara lor de viață.
- void drawHealthBar (Graphics g, Karagor karagor, int xLvlOffset)

  Desenează bara de viață pentru un Karagor.

- void drawHealthBar (Graphics g, Nanite nanite, int xLvlOffset)

  Desenează bara de viață pentru un Nanite.
- void drawHealthBar (Graphics g, Goblin goblin, int xLvlOffset)

  Desenează bara de viață pentru un Goblin.
- void drawHealthBar (Graphics g, GoblinBoss boss, int xLvlOffset)

  Desenează bara de viață pentru un GoblinBoss.
- void drawHealthBar (Graphics g, GolemBoss boss, int xLvlOffset)

  Desenează bara de viață pentru un GolemBoss.
- void scanLevelForSpawnPoints ()

  Scanează datele nivelului pentru a identifica punctele de spawn ale inamicilor.
- void spawnEnemy (float x, float y, int enemyCode)

  Spawnează un inamic la o poziție specificată, pe baza codului său.
- void checkForNewEnemySpawn (Rectangle2D.Float playerHitbox)

  Verifică dacă trebuie spawnați noi inamici pe baza proximității jucătorului față de punctele de spawn rămase.
- void applyKnockback (Player player, boolean enemyHitFromLeft)

  Aplică un efect de knockback jucătorului.
- void spawnGem (float x, float y)

  Spawnează un gem la o poziție specificată.
- void trySpawnCollectible (float x, float y)

  Încearcă să spawneze un obiect colectabil (banană sau nucă de cocos) la o poziție specificată, cu o șansă de 50% pentru fiecare.

Private Attributes

- Playing playing
- BufferedImage[ ][ ] naniteImgs
- BufferedImage[ ][ ] nanitePesteraImgs
- BufferedImage[ ][ ] goblinNoobImgs
- BufferedImage[ ][ ] goblinHardImgs
- ArrayList< Nanite > nanites = new ArrayList<>()
- ArrayList< Karagor > karagors = new ArrayList<>()
- ArrayList< Goblin > goblins = new ArrayList<>()
- ArrayList< GoblinBoss > goblinBosses = new ArrayList<>()
- ArrayList< GolemBoss > golemBosses = new ArrayList<>()
- ArrayList< Projectile > projectiles = new ArrayList<>()
- ArrayList< Gem > gems = new ArrayList<>()
- Random random = new Random()
- int[ ][ ] levelData
- int currentLevel
- ArrayList< Point > spawnPoints = new ArrayList<>()
- boolean allEnemiesSpawned = false
- float playerDetectionDistance = 400

6.8.1 Detailed Description

Gestionează toți inamicii din joc, inclusiv Nanites, Karagors, Goblins, GoblinBosses și GolemBosses.

Se ocupă de încărcarea imaginilor inamicilor, actualizarea stării lor, desenarea lor pe ecran, gestionarea coliziunilor cu jucătorul și proiectilele, și spawn-ul inamicilor și al obiectelor colectabile.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 EnemyManager()

```
entities.EnemyManager.EnemyManager (
            Playing playing )
```

Constructor pentru EnemyManager.

Inițializează managerul cu o referință la starea de joc "Playing", încarcă imaginile inamicilor și setează nivelul curent la 1.

Parameters

playing	Referință la starea de joc "Playing".

6.8.3 Member Function Documentation

6.8.3.1 addProjectile()

```
void entities.EnemyManager.addProjectile (
            Projectile p )
```

Adaugă un proiectil la lista de proiectile active gestionate de EnemyManager.

Parameters

p	Proiectilul care trebuie adăugat.

6.8.3.2 applyKnockback()

```
void entities.EnemyManager.applyKnockback (
            Player player,
            boolean enemyHitFromLeft )  [private]
```

Aplică un efect de knockback jucătorului.

Direcția knockback-ului depinde de partea din care a fost lovit jucătorul.

Parameters

player	Jucătorul care primește knockback.
enemyHitFromLeft	true dacă inamicul a lovit din stânga jucătorului, false altfel.

6.8.3.3 checkForNewEnemySpawn()

```
void entities.EnemyManager.checkForNewEnemySpawn (
            Rectangle2D.Float playerHitbox )  [private]
```

Verifică dacă trebuie spawnați noi inamici pe baza proximității jucătorului față de punctele de spawn rămase.

Dacă un punct de spawn este suficient de aproape de jucător, inamicul corespunzător este spawnat și punctul este eliminat din listă.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru a calcula distanța.

6.8.3.4 draw()

```
void entities.EnemyManager.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează toți inamicii activi, proiectilele și gem-urile pe ecran.

Parameters

g	Contextul grafic pentru desenare.
xLvlOffset	Decalajul pe axa X al nivelului, pentru scrolling.

6.8.3.5 drawGems()

```
void entities.EnemyManager.drawGems (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toate gem-urile active pe ecran.

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.6 drawGoblinBosses()

```
void entities.EnemyManager.drawGoblinBosses (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toți GoblinBosses activi pe ecran, inclusiv bara lor de viață.

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.7 drawGoblins()

```
void entities.EnemyManager.drawGoblins (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toți Goblinii activi (Noob și Hard) pe ecran, inclusiv bara lor de viață și hitbox-ul (dacă este activat).

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.8 drawGolemBosses()

```
void entities.EnemyManager.drawGolemBosses (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toți GolemBosses activi pe ecran, inclusiv bara lor de viață.

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.9 drawHealthBar() [1/5]

```
void entities.EnemyManager.drawHealthBar (
            Graphics g,
            Goblin goblin,
            int xLvlOffset )  [private]
```

Desenează bara de viață pentru un Goblin.

Parameters

g	Contextul grafic.
goblin	Goblin-ul pentru care se desenează bara de viață.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.10 drawHealthBar() [2/5]

```
void entities.EnemyManager.drawHealthBar (
            Graphics g,
            GoblinBoss boss,
            int xLvlOffset )  [private]
```

Desenează bara de viață pentru un GoblinBoss.

Parameters

g	Contextul grafic.
boss	GoblinBoss-ul pentru care se desenează bara de viață.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.11 drawHealthBar() [3/5]

```
void entities.EnemyManager.drawHealthBar (
            Graphics g,
            GolemBoss boss,
            int xLvlOffset )  [private]
```

Desenează bara de viață pentru un GolemBoss.

Parameters

g	Contextul grafic.
boss	GolemBoss-ul pentru care se desenează bara de viață.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.12 drawHealthBar() [4/5]

```
void entities.EnemyManager.drawHealthBar (
            Graphics g,
            Karagor karagor,
            int xLvlOffset )  [private]
```

Desenează bara de viață pentru un Karagor.

Parameters

g	Contextul grafic.
karagor	Karagor-ul pentru care se desenează bara de viață.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.13 drawHealthBar() [5/5]

```
void entities.EnemyManager.drawHealthBar (
            Graphics g,
            Nanite nanite,
            int xLvlOffset )  [private]
```

Desenează bara de viață pentru un Nanite.

Parameters

g	Contextul grafic.
nanite	Nanite-ul pentru care se desenează bara de viață.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.14 drawKaragors()

```
void entities.EnemyManager.drawKaragors (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toți Karagors activi pe ecran, inclusiv bara lor de viață.

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.15 drawNanites()

```
void entities.EnemyManager.drawNanites (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toți Nanites activi (Jungla și Pestera) pe ecran, inclusiv bara lor de viață și hitbox-ul (dacă este activat).

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.16 drawProjectiles()

```
void entities.EnemyManager.drawProjectiles (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează toate proiectilele active pe ecran.

Parameters

g	Contextul grafic.
xLvlOffset	Decalajul nivelului pe axa X.

6.8.3.17 getCurrentLevel()

```
int entities.EnemyManager.getCurrentLevel ( )
```

Returnează ID-ul nivelului curent.

Returns

ID-ul nivelului.

6.8.3.18 getGems()

```
ArrayList<Gem> entities.EnemyManager.getGems ( )
```

Returnează lista de Gem-uri active.

Returns

O ArrayList de obiecte Gem.

6.8.3.19 getGoblinBosses()

```
ArrayList<GoblinBoss> entities.EnemyManager.getGoblinBosses ( )
```

Returnează lista de GoblinBosses activi.

Returns

O ArrayList de obiecte GoblinBoss.

6.8.3.20 getGoblins()

```
ArrayList<Goblin> entities.EnemyManager.getGoblins ( )
```

Returnează lista de Goblini activi.

Returns

O ArrayList de obiecte Goblin.

6.8.3.21 getGolemBosses()

```
ArrayList<GolemBoss> entities.EnemyManager.getGolemBosses ( )
```

Returnează lista de GolemBosses activi.

Returns

O ArrayList de obiecte GolemBoss.

6.8.3.22 getKaragors()

```
ArrayList<Karagor> entities.EnemyManager.getKaragors ( )
```

Returnează lista de Karagors activi.

Returns

O ArrayList de obiecte Karagor.

6.8.3.23 getNanites()

```
ArrayList<Nanite> entities.EnemyManager.getNanites ( )
```

Returnează lista de Nanites activi.

Returns

O ArrayList de obiecte Nanite.

6.8.3.24 loadEnemiesFromLevelData()

```
void entities.EnemyManager.loadEnemiesFromLevelData (
            int levelData[][],
            Level level )
```

Încarcă inamicii dintr-un nivel pe baza datelor acestuia.

Golește listele de inamici existenți și scanează datele nivelului pentru puncte de spawn. Pentru anumite niveluri (1, 2, 3), toți inamicii sunt spawnați imediat.

Parameters

levelData	Datele nivelului (matrice de tile-uri).
level	Obiectul Level curent, pentru a obține ID-ul nivelului.

6.8.3.25 loadEnemyImgs()

```
void entities.EnemyManager.loadEnemyImgs ( )  [private]
```

Încarcă imaginile pentru toți inamicii (Nanites, Goblins).

Boss-ii (GoblinBoss, Karagor) își încarcă propriile sprite-uri în clasele lor.

6.8.3.26 loadSpriteSheet()

```
BufferedImage [][] entities.EnemyManager.loadSpriteSheet (
            String path,
            int spriteWidth,
            int spriteHeight )  [private]
```

Încarcă un sprite sheet dintr-o cale specificată și îl împarte în frame-uri individuale.

Determină numărul maxim de coloane necesar pe baza definițiilor din Enemy_Animation_Rows.

Parameters

path	Calea către fișierul sprite sheet.
spriteWidth	Lățimea unui singur sprite.
spriteHeight	Înălțimea unui singur sprite.

Returns

O matrice bidimensională de imagini (frame-uri de animație). Returnează o matrice goală dacă încărcarea eșuează pentru a evita NullPointerException.

6.8.3.27 resetEnemies()

```
void entities.EnemyManager.resetEnemies ( )
```

Resetează toți inamicii din nivel.

Golește listele de inamici și reîncarcă punctele de spawn din datele nivelului. Pentru anumite niveluri (1, 2, 3), toți inamicii sunt spawnați imediat după resetare.

6.8.3.28 scanLevelForSpawnPoints()

```
void entities.EnemyManager.scanLevelForSpawnPoints ( )  [private]
```

Scanează datele nivelului pentru a identifica punctele de spawn ale inamicilor.

Stochează aceste puncte (coordonate și codul inamicului) într-o listă pentru spawn ulterior. Afișează în consolă numărul de puncte de spawn găsite.

6.8.3.29 spawnAllEnemies()

```
void entities.EnemyManager.spawnAllEnemies ( )
```

Spawnează toți inamicii identificați în punctele de spawn.

Marchează `allEnemiesSpawned` ca true după ce toți inamicii au fost spawnați. Afișează în consolă numărul de inamici spawnați.

6.8.3.30 spawnEnemy()

```
void entities.EnemyManager.spawnEnemy (
            float x,
            float y,
            int enemyCode )  [private]
```

Spawnează un inamic la o poziție specificată, pe baza codului său.

Adaugă inamicul nou creat la lista corespunzătoare și îi setează datele nivelului.

Parameters

x	Coordonata X a punctului de spawn.
y	Coordonata Y a punctului de spawn.
enemyCode	Codul numeric care identifică tipul de inamic.

6.8.3.31 spawnGem()

```
void entities.EnemyManager.spawnGem (
            float x,
            float y )  [private]
```

Spawnează un gem la o poziție specificată.

Tipul gem-ului depinde de nivelul curent.

Parameters

x	Coordonata X a punctului de spawn pentru gem.
y	Coordonata Y a punctului de spawn pentru gem.

6.8.3.32 trySpawnCollectible()

```
void entities.EnemyManager.trySpawnCollectible (
            float x,
            float y )  [private]
```

Încearcă să spawneze un obiect colectabil (banană sau nucă de cocos) la o poziție specificată, cu o șansă de 50% pentru fiecare.

Verifică dacă imaginile pentru colectabile sunt încărcate.

Parameters

x	Coordonata X a punctului de spawn pentru obiectul colectabil.
y	Coordonata Y a punctului de spawn pentru obiectul colectabil.

6.8.3.33 update()

```
void entities.EnemyManager.update (
            Rectangle2D.Float playerHitbox )
```

Actualizează starea tuturor inamicilor activi, proiectilelor și gem-urilor.

Gestionează spawn-ul inamicilor pe baza proximității jucătorului (dacă este cazul pentru nivelul curent), coliziunile, aplicarea daunelor și efectele de knockback. De asemenea, gestionează drop-urile de la inamici.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru interacțiuni și detectare.

6.8.4 Member Data Documentation

6.8.4.1 allEnemiesSpawned

```
boolean entities.EnemyManager.allEnemiesSpawned = false  [private]
```

6.8.4.2 currentLevel

```
int entities.EnemyManager.currentLevel  [private]
```

6.8.4.3 gems

```
ArrayList<Gem> entities.EnemyManager.gems = new ArrayList<>()  [private]
```

6.8.4.4 goblinBosses

```
ArrayList<GoblinBoss> entities.EnemyManager.goblinBosses = new ArrayList<>()  [private]
```

6.8.4.5 goblinHardImgs

```
BufferedImage [][] entities.EnemyManager.goblinHardImgs  [private]
```

6.8.4.6 goblinNoobImgs

```
BufferedImage [][] entities.EnemyManager.goblinNoobImgs  [private]
```

6.8.4.7 goblins

```
ArrayList<Goblin> entities.EnemyManager.goblins = new ArrayList<>()  [private]
```

6.8.4.8 golemBosses

```
ArrayList<GolemBoss> entities.EnemyManager.golemBosses = new ArrayList<>()  [private]
```

6.8.4.9 karagors

```
ArrayList<Karagor> entities.EnemyManager.karagors = new ArrayList<>()  [private]
```

6.8.4.10 levelData

```
int [][] entities.EnemyManager.levelData  [private]
```

6.8.4.11 naniteImgs

```
BufferedImage [][] entities.EnemyManager.naniteImgs  [private]
```

6.8.4.12 nanitePesteraImgs

```
BufferedImage [][] entities.EnemyManager.nanitePesteraImgs  [private]
```

6.8.4.13 nanites

```
ArrayList<Nanite> entities.EnemyManager.nanites = new ArrayList<>()  [private]
```

6.8.4.14 playerDetectionDistance

```
float entities.EnemyManager.playerDetectionDistance = 400  [private]
```

6.8.4.15 playing

```
Playing entities.EnemyManager.playing  [private]
```

6.8.4.16 projectiles

```
ArrayList<Projectile> entities.EnemyManager.projectiles = new ArrayList<>()  [private]
```

6.8.4.17 random

```
Random entities.EnemyManager.random = new Random()  [private]
```

6.8.4.18 spawnPoints

```
ArrayList<Point> entities.EnemyManager.spawnPoints = new ArrayList<>()  [private]
```

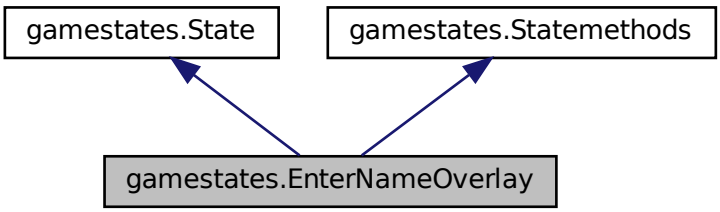
The documentation for this class was generated from the following file:

- src/entities/EnemyManager.java

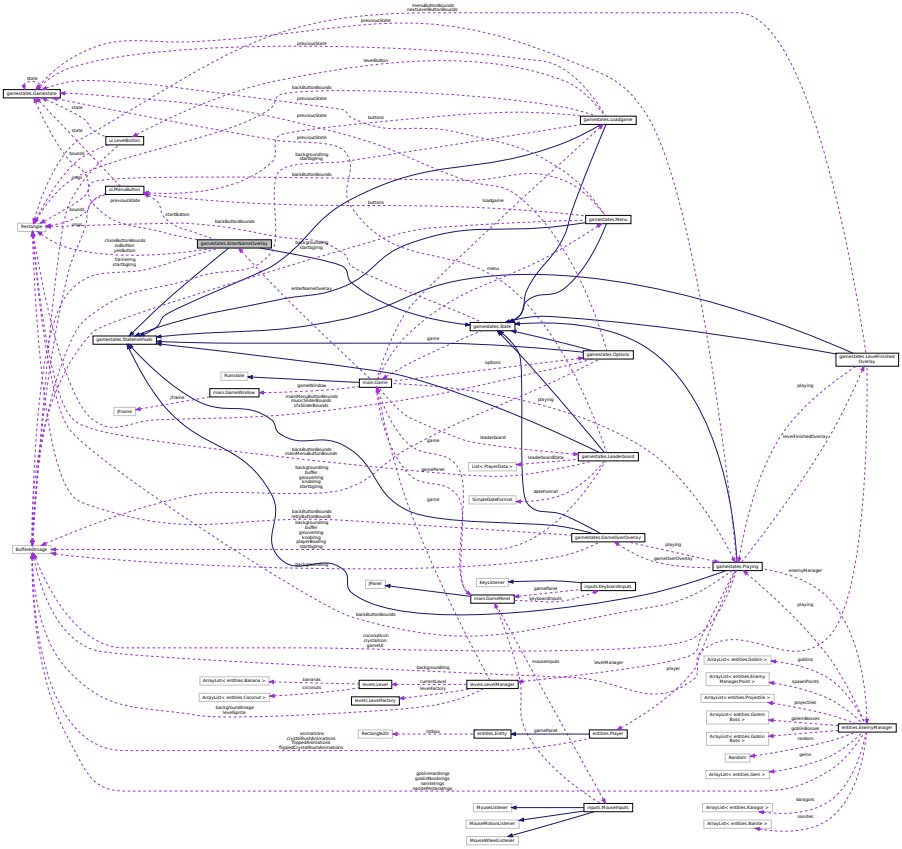
6.9 gamestates.EnterNameOverlay Class Reference

Reprezintă un overlay pentru introducerea numelui jucătorului.

Inheritance diagram for gamestates.EnterNameOverlay:



Collaboration diagram for gamestates.EnterNameOverlay:



Public Member Functions

- EnterNameOverlay (Game game)

Constructor pentru EnterNameOverlay.

- void update ()

  Actualizează starea overlay-ului.
- void draw (Graphics g)

  Desenează elementele overlay-ului pe ecran.
- void mouseClicked (MouseEvent e)

  Gestionează evenimentele de click al mouse-ului.
- void mousePressed (MouseEvent e)

  Gestionează evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionează evenimentele de eliberare a butonului mouse-ului.
- void mouseMoved (MouseEvent e)

  Gestionează evenimentele de mișcare a mouse-ului.
- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionează evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).

Private Member Functions

- void loadStartImg ()

  Încarcă și configurează imaginea de fundal.
- void loadFrame ()

  Încarcă și configurează imaginea cadrului pentru introducerea numelui.
- void loadButtons ()

  Inițializează butonul de start.
- void processUsername (String playerName)

  Procesează numele de utilizator introdus.
- void loadCustomFont ()

  Încarcă fontul personalizat folosit în acest overlay.

Private Attributes

- BufferedImage startbgImg
- BufferedImage frameImg
- int startbgX
- int frameX
- MenuButton startButton
- Rectangle closeButtonBounds
- StringBuilder username = new StringBuilder()
- final int MAX_NAME_LENGTH = 15

  Lungimea maximă permisă pentru numele de utilizator.
- boolean inputActive = true

  Indică dacă input-ul de la tastatură este activ pentru acest overlay.
- Gamestate previousState = null
- Font airstrikeFont
- String errorMessage = ""
- boolean showLoadPrompt = false

Indică dacă se afișează prompt-ul pentru încărcarea unui joc salvat existent.

- Rectangle yesButton = new Rectangle(775, 650, 100, 40)
- Rectangle noButton = new Rectangle(925, 650, 100, 40)
- int existingLevel = 0

  Numele tabelului principal pentru progresul jucătorului (neutilizat direct, se folosește username-ul).
- int existingScore = 0
- int existingHealth = 0
- int existingCoconuts = 0
- float existingPosX = 0
- float existingPosY = 0

Static Private Attributes

- static final String DB_FILE = "data/gamedatabase.db"

  Calea către fișierul bazei de date.

Additional Inherited Members

6.9.1 Detailed Description

Reprezintă un overlay pentru introducerea numelui jucătorului.

Această stare permite jucătorului să introducă un nume, care este apoi folosit pentru a salva sau încărca progresul jocului. Extinde State și implementează Statemethods.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 EnterNameOverlay()

```
gamestates.EnterNameOverlay.EnterNameOverlay (
            Game game )
```

Constructor pentru EnterNameOverlay.

Parameters

game	Referință la instanța principală a jocului Game.

6.9.3 Member Function Documentation

6.9.3.1 draw()

```
void gamestates.EnterNameOverlay.draw (
            Graphics g )
```

Desenează elementele overlay-ului pe ecran.

Include fundalul, cadrul, câmpul pentru nume, mesajele de eroare și prompt-ul de încărcare.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.9.3.2 keyPressed()

```
void gamestates.EnterNameOverlay.keyPressed (
            KeyEvent e )
```

Gestionează evenimentele de apăsare a tastelor.

Permite introducerea numelui de utilizator, ștergerea caracterelor (Backspace) și confirmarea numelui (Enter).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.9.3.3 keyReleased()

```
void gamestates.EnterNameOverlay.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.9.3.4 loadButtons()

```
void gamestates.EnterNameOverlay.loadButtons ( )  [private]
```

Inițializează butonul de start.

6.9.3.5 loadCustomFont()

```
void gamestates.EnterNameOverlay.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat folosit în acest overlay.

6.9.3.6 loadFrame()

```
void gamestates.EnterNameOverlay.loadFrame ( )  [private]
```

Încarcă și configurează imaginea cadrului pentru introducerea numelui.

6.9.3.7 loadStartImg()

```
void gamestates.EnterNameOverlay.loadStartImg ( )  [private]
```

Încarcă și configurează imaginea de fundal.

6.9.3.8 mouseClicked()

```
void gamestates.EnterNameOverlay.mouseClicked (
            MouseEvent e )
```

Gestionează evenimentele de click al mouse-ului.

Procesează click-urile pe butonul de start sau pe opțiunile din prompt-ul de încărcare.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.9.3.9 mouseDragged()

```
void gamestates.EnterNameOverlay.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.9.3.10 mouseMoved()

```
void gamestates.EnterNameOverlay.mouseMoved (
            MouseEvent e )
```

Gestionează evenimentele de mișcare a mouse-ului.

Setează starea "mouse over" pentru butonul de start.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.9.3.11 mousePressed()

```
void gamestates.EnterNameOverlay.mousePressed (
            MouseEvent e )
```

Gestionează evenimentele de apăsare a butonului mouse-ului.

Setează starea de apăsare pentru butonul de start și gestionează butonul de închidere.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.9.3.12 mouseReleased()

```
void gamestates.EnterNameOverlay.mouseReleased (
            MouseEvent e )
```

Gestionează evenimentele de eliberare a butonului mouse-ului.

Finalizează acțiunea butonului de start dacă a fost apăsat.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.9.3.13 processUsername()

```
void gamestates.EnterNameOverlay.processUsername (
            String playerName )  [private]
```

Procesează numele de utilizator introdus.

Verifică dacă există un joc salvat pentru acest nume. Dacă da, afișează un prompt pentru încărcare. Dacă nu, creează o nouă salvare și trece la meniul principal.

Parameters

playerName	Numele de utilizator de procesat.

6.9.3.14 update()

```
void gamestates.EnterNameOverlay.update ( )
```

Actualizează starea overlay-ului.

Momentan, actualizează doar butonul de start.

Implements gamestates.Statemethods.

6.9.4 Member Data Documentation

6.9.4.1 airstrikeFont

```
Font gamestates.EnterNameOverlay.airstrikeFont  [private]
```

6.9.4.2 closeButtonBounds

```
Rectangle gamestates.EnterNameOverlay.closeButtonBounds  [private]
```

6.9.4.3 DB_FILE

```
final String gamestates.EnterNameOverlay.DB_FILE = "data/gamedatabase.db"  [static], [private]
```

Calea către fișierul bazei de date.

6.9.4.4 errorMessage

```
String gamestates.EnterNameOverlay.errorMessage = ""  [private]
```

6.9.4.5 existingCoconuts

```
int gamestates.EnterNameOverlay.existingCoconuts = 0  [private]
```

6.9.4.6 existingHealth

```
int gamestates.EnterNameOverlay.existingHealth = 0  [private]
```

6.9.4.7 existingLevel

```
int gamestates.EnterNameOverlay.existingLevel = 0  [private]
```

Numele tabelului principal pentru progresul jucătorului (neutilizat direct, se folosește username-ul).

6.9.4.8 existingPosX

```
float gamestates.EnterNameOverlay.existingPosX = 0  [private]
```

6.9.4.9 existingPosY

```
float gamestates.EnterNameOverlay.existingPosY = 0  [private]
```

6.9.4.10 existingScore

```
int gamestates.EnterNameOverlay.existingScore = 0  [private]
```

6.9.4.11 frameImg

```
BufferedImage gamestates.EnterNameOverlay.frameImg  [private]
```

6.9.4.12 frameX

```
int gamestates.EnterNameOverlay.frameX  [private]
```

6.9.4.13 inputActive

```
boolean gamestates.EnterNameOverlay.inputActive = true  [private]
```

Indică dacă input-ul de la tastatură este activ pentru acest overlay.

6.9.4.14 MAX_NAME_LENGTH

```
final int gamestates.EnterNameOverlay.MAX_NAME_LENGTH = 15  [private]
```

Lungimea maximă permisă pentru numele de utilizator.

6.9.4.15 noButton

```
Rectangle gamestates.EnterNameOverlay.noButton = new Rectangle(925, 650, 100, 40)  [private]
```

6.9.4.16 previousState

```
Gamestate gamestates.EnterNameOverlay.previousState = null  [private]
```

6.9.4.17 showLoadPrompt

```
boolean gamestates.EnterNameOverlay.showLoadPrompt = false  [private]
```

Indică dacă se afișează prompt-ul pentru încărcarea unui joc salvat existent.

6.9.4.18 startbgImg

```
BufferedImage gamestates.EnterNameOverlay.startbgImg  [private]
```

6.9.4.19 startbgX

```
int gamestates.EnterNameOverlay.startbgX  [private]
```

6.9.4.20 startButton

```
MenuButton gamestates.EnterNameOverlay.startButton  [private]
```

6.9.4.21 username

```
StringBuilder gamestates.EnterNameOverlay.username = new StringBuilder()  [private]
```

6.9.4.22 yesButton

```
Rectangle gamestates.EnterNameOverlay.yesButton = new Rectangle(775, 650, 100, 40)  [private]
```

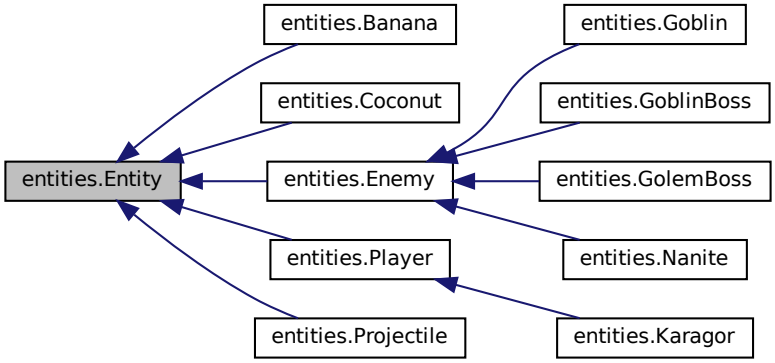
The documentation for this class was generated from the following file:

- src/gamestates/EnterNameOverlay.java

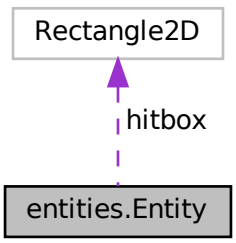
6.10 entities.Entity Class Reference

Clasă abstractă de bază pentru toate entitățile din joc (de ex., jucător, inamici, obiecte).

Inheritance diagram for entities.Entity:



Collaboration diagram for entities.Entity:



Public Member Functions

- Entity (float x, float y, int width, int height)

  Constructor pentru clasa Entity.
- Rectangle2D.Float getHitbox ()

  Returnează hitbox-ul entității.

Protected Member Functions

- void initHitbox (float x, float y, float width, float height)

  Inițializează hitbox-ul entității cu coordonatele și dimensiunile specificate.
- void drawHitbox (Graphics g)

  Desenează hitbox-ul entității pe ecran.

Protected Attributes

- float x

  Coordonata x a colțului stânga-sus al entității (poate fi pentru desenare).
- float y

  Coordonata y a colțului stânga-sus al entității (poate fi pentru desenare).
- int width

  Lățimea entității (poate fi pentru desenare).
- int height

  Înălțimea entității (poate fi pentru desenare).
- Rectangle2D.Float hitbox

  Dreptunghiul de coliziune (hitbox) al entității.

6.10.1 Detailed Description

Clasă abstractă de bază pentru toate entitățile din joc (de ex., jucător, inamici, obiecte).

Definește proprietăți comune precum poziția (x, y), dimensiunile (lățime, înălțime) și un hitbox pentru detecția coliziunilor.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 Entity()

```
entities.Entity.Entity (
            float x,
            float y,
            int width,
            int height )
```

Constructor pentru clasa Entity.

Parameters

x	Poziția x inițială a entității.
y	Poziția y inițială a entității.
width	Lățimea inițială a entității.
height	Înălțimea inițială a entității.

6.10.3 Member Function Documentation

6.10.3.1 drawHitbox()

```
void entities.Entity.drawHitbox (
            Graphics g )  [protected]
```

Desenează hitbox-ul entității pe ecran.

Utilizată în principal pentru depanare.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.10.3.2 getHitbox()

```
Rectangle2D.Float entities.Entity.getHitbox ( )
```

Returnează hitbox-ul entității.

Returns

Obiectul Rectangle2D.Float reprezentând hitbox-ul.

Reimplemented in entities.Projectile, and entities.Enemy.

6.10.3.3 initHitbox()

```
void entities.Entity.initHitbox (
            float x,
            float y,
            float width,
            float height )  [protected]
```

Inițializează hitbox-ul entității cu coordonatele și dimensiunile specificate.

Parameters

x	Poziția x a colțului stânga-sus al hitbox-ului.
y	Poziția y a colțului stânga-sus al hitbox-ului.
width	Lățimea hitbox-ului.
height	Înălțimea hitbox-ului.

6.10.4 Member Data Documentation

6.10.4.1 height

```
int entities.Entity.height  [protected]
```

Înălțimea entității (poate fi pentru desenare).

6.10.4.2 hitbox

```
Rectangle2D.Float entities.Entity.hitbox  [protected]
```

Dreptunghiul de coliziune (hitbox) al entității.

6.10.4.3 width

```
int entities.Entity.width  [protected]
```

Lățimea entității (poate fi pentru desenare).

6.10.4.4 x

```
float entities.Entity.x  [protected]
```

Coordonata x a colțului stânga-sus al entității (poate fi pentru desenare).

6.10.4.5 y

```
float entities.Entity.y  [protected]
```

Coordonata y a colțului stânga-sus al entității (poate fi pentru desenare).

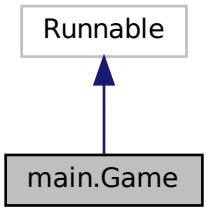
The documentation for this class was generated from the following file:

- src/entities/Entity.java

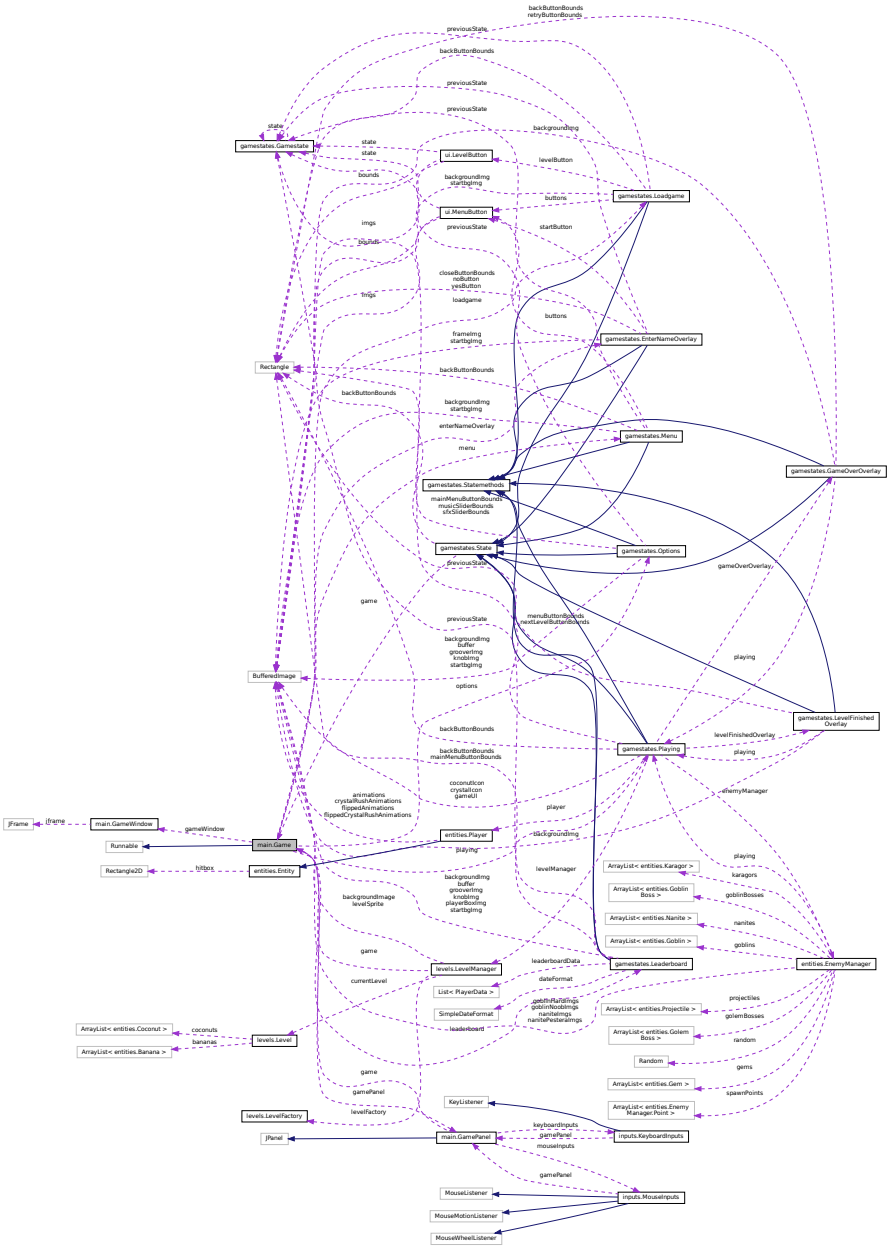
6.11 main.Game Class Reference

Clasa principală a jocului, responsabilă pentru gestionarea stărilor de joc, bucla principală a jocului (game loop) și inițializarea componentelor cheie.

Inheritance diagram for main.Game:



Collaboration diagram for main.Game:



Public Member Functions

- void setSessionUsername (String username)

  Setează numele de utilizator pentru sesiunea curentă.
- String getSessionUsername ()

  Returnează numele de utilizator al sesiunii curente.
- Game ()

  Constructor pentru clasa Game.
- void update ()

  Actualizează logica jocului în funcție de starea curentă (Gamestate).
- void render (Graphics g)

Desenează conținutul jocului în funcție de starea curentă (Gamestate).

- void run ()

  Metoda principală a buclei jocului, implementată din interfața Runnable.
- void windowFocusLost ()

  Metodă apelată când fereastra jocului pierde focusul.
- Menu getMenu ()
- Playing getPlaying ()
- Options getOptions ()
- Loadgame getLoadgame ()
- Leaderboard getLeaderboard ()
- EnterNameOverlay getEnterNameOverlay ()

Static Public Attributes

- static final int TILES_DEFAULT_SIZE = 60

  Dimensiunea implicită a unui tile (în pixeli) înainte de scalare.
- static final float SCALE = 1.0f

  Factorul de scalare aplicat elementelor jocului.
- static final int TILES_IN_WIDTH = 32

  Numărul de tile-uri vizibile pe lățimea ecranului.
- static final int TILES_IN_HEIGHT = 18

  Numărul de tile-uri vizibile pe înălțimea ecranului.
- static final int TILES_SIZE = (int) (TILES_DEFAULT_SIZE ∗ SCALE)

  Dimensiunea finală a unui tile (în pixeli) după aplicarea scalei.
- static final int GAME_WIDTH = TILES_SIZE ∗ TILES_IN_WIDTH

  Lățimea totală a ferestrei jocului în pixeli.
- static final int GAME_HEIGHT = TILES_SIZE ∗ TILES_IN_HEIGHT

  Înălțimea totală a ferestrei jocului în pixeli.

Private Member Functions

- void initClasses ()

  Inițializează toate instanțele stărilor de joc.
- void startGameLoop ()

  Pornește bucla principală a jocului într-un nou fir de execuție.

Private Attributes

- String sessionUsername

  Numele de utilizator pentru sesiunea curentă de joc.
- GameWindow gameWindow
- GamePanel gamePanel
- Thread gameThread
- final double FPS_SET = 60

  Numărul țintă de cadre pe secundă (Frames Per Second).
- final double UPS_SET = 120

  Numărul țintă de actualizări logice pe secundă (Updates Per Second).
- Playing playing
- Menu menu
- Options options
- Loadgame loadgame
- Leaderboard leaderboard
- EnterNameOverlay enterNameOverlay

6.11.1 Detailed Description

Clasa principală a jocului, responsabilă pentru gestionarea stărilor de joc, bucla principală a jocului (game loop) și inițializarea componentelor cheie.

Implementează interfața Runnable pentru a rula într-un fir de execuție separat.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Game()

```
main.Game.Game ( )
```

Constructor pentru clasa Game.

Inițializează toate clasele necesare, panoul de joc, fereastra și pornește bucla principală a jocului.

6.11.3 Member Function Documentation

6.11.3.1 getEnterNameOverlay()

```
EnterNameOverlay main.Game.getEnterNameOverlay ( )
```

Returns

Instanța overlay-ului de introducere a numelui.

6.11.3.2 getLeaderboard()

```
Leaderboard main.Game.getLeaderboard ( )
```

Returns

Instanța stării de clasament.

6.11.3.3 getLoadgame()

```
Loadgame main.Game.getLoadgame ( )
```

Returns

Instanța stării de încărcare a jocului.

6.11.3.4 getMenu()

```
Menu main.Game.getMenu ( )
```

Returns

Instanța stării de meniu.

6.11.3.5 getOptions()

```
Options main.Game.getOptions ( )
```

Returns

Instanța stării de opțiuni.

6.11.3.6 getPlaying()

```
Playing main.Game.getPlaying ( )
```

Returns

Instanța stării de joc propriu-zis.

6.11.3.7 getSessionUsername()

```
String main.Game.getSessionUsername ( )
```

Returnează numele de utilizator al sesiunii curente.

Returns

Numele de utilizator.

6.11.3.8 initClasses()

```
void main.Game.initClasses ( )  [private]
```

Inițializează toate instanțele stărilor de joc.

6.11.3.9 render()

```
void main.Game.render (
            Graphics g )
```

Desenează conținutul jocului în funcție de starea curentă (Gamestate).

Deleagă desenarea către metoda
`draw`

a stării active.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.11.3.10 run()

```
void main.Game.run ( )
```

Metoda principală a buclei jocului, implementată din interfața Runnable.

Gestionează sincronizarea actualizărilor logice (UPS) și a cadrelor desenate (FPS).

6.11.3.11 setSessionUsername()

```
void main.Game.setSessionUsername (
            String username )
```

Setează numele de utilizator pentru sesiunea curentă.

Parameters

username	Noul nume de utilizator.

6.11.3.12 startGameLoop()

```
void main.Game.startGameLoop ( )  [private]
```

Pornește bucla principală a jocului într-un nou fir de execuție.

6.11.3.13 update()

```
void main.Game.update ( )
```

Actualizează logica jocului în funcție de starea curentă (Gamestate).

Deleagă actualizarea către metoda
update

a stării active. În cazul stării QUIT, închide aplicația.

6.11.3.14 windowFocusLost()

```
void main.Game.windowFocusLost ( )
```

Metodă apelată când fereastra jocului pierde focusul.

Dacă starea curentă este PLAYING, resetează flag-urile de direcție ale jucătorului.

6.11.4 Member Data Documentation

6.11.4.1 enterNameOverlay

```
EnterNameOverlay main.Game.enterNameOverlay  [private]
```

6.11.4.2 FPS_SET

```
final double main.Game.FPS_SET = 60  [private]
```

Numărul țintă de cadre pe secundă (Frames Per Second).

6.11.4.3 GAME_HEIGHT

```
final int main.Game.GAME_HEIGHT = TILES_SIZE * TILES_IN_HEIGHT  [static]
```

Înălțimea totală a ferestrei jocului în pixeli.

6.11.4.4 GAME_WIDTH

```
final int main.Game.GAME_WIDTH = TILES_SIZE * TILES_IN_WIDTH  [static]
```

Lățimea totală a ferestrei jocului în pixeli.

6.11.4.5 gamePanel

```
GamePanel main.Game.gamePanel  [private]
```

6.11.4.6 gameThread

```
Thread main.Game.gameThread  [private]
```

6.11.4.7 gameWindow

```
GameWindow main.Game.gameWindow  [private]
```

6.11.4.8 leaderboard

```
Leaderboard main.Game.leaderboard  [private]
```

6.11.4.9 loadgame

```
Loadgame main.Game.loadgame  [private]
```

6.11.4.10 menu

```
Menu main.Game.menu  [private]
```

6.11.4.11 options

```
Options main.Game.options  [private]
```

6.11.4.12 playing

```
Playing main.Game.playing  [private]
```

6.11.4.13 SCALE

```
final float main.Game.SCALE = 1.0f  [static]
```

Factorul de scalare aplicat elementelor jocului.

6.11.4.14 sessionUsername

```
String main.Game.sessionUsername  [private]
```

Numele de utilizator pentru sesiunea curentă de joc.

6.11.4.15 TILES_DEFAULT_SIZE

```
final int main.Game.TILES_DEFAULT_SIZE = 60  [static]
```

Dimensiunea implicită a unui tile (în pixeli) înainte de scalare.

6.11.4.16 TILES_IN_HEIGHT

```
final int main.Game.TILES_IN_HEIGHT = 18  [static]
```

Numărul de tile-uri vizibile pe înălțimea ecranului.

6.11.4.17 TILES_IN_WIDTH

```
final int main.Game.TILES_IN_WIDTH = 32  [static]
```

Numărul de tile-uri vizibile pe lățimea ecranului.

6.11.4.18 TILES_SIZE

```
final int main.Game.TILES_SIZE = (int) (TILES_DEFAULT_SIZE * SCALE)  [static]
```

Dimensiunea finală a unui tile (în pixeli) după aplicarea scalei.

6.11.4.19 UPS_SET

```
final double main.Game.UPS_SET = 120  [private]
```

Numărul țintă de actualizări logice pe secundă (Updates Per Second).

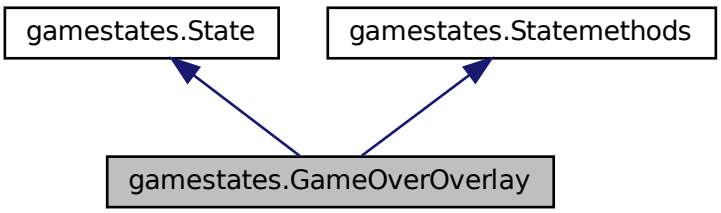
The documentation for this class was generated from the following file:

- src/main/Game.java

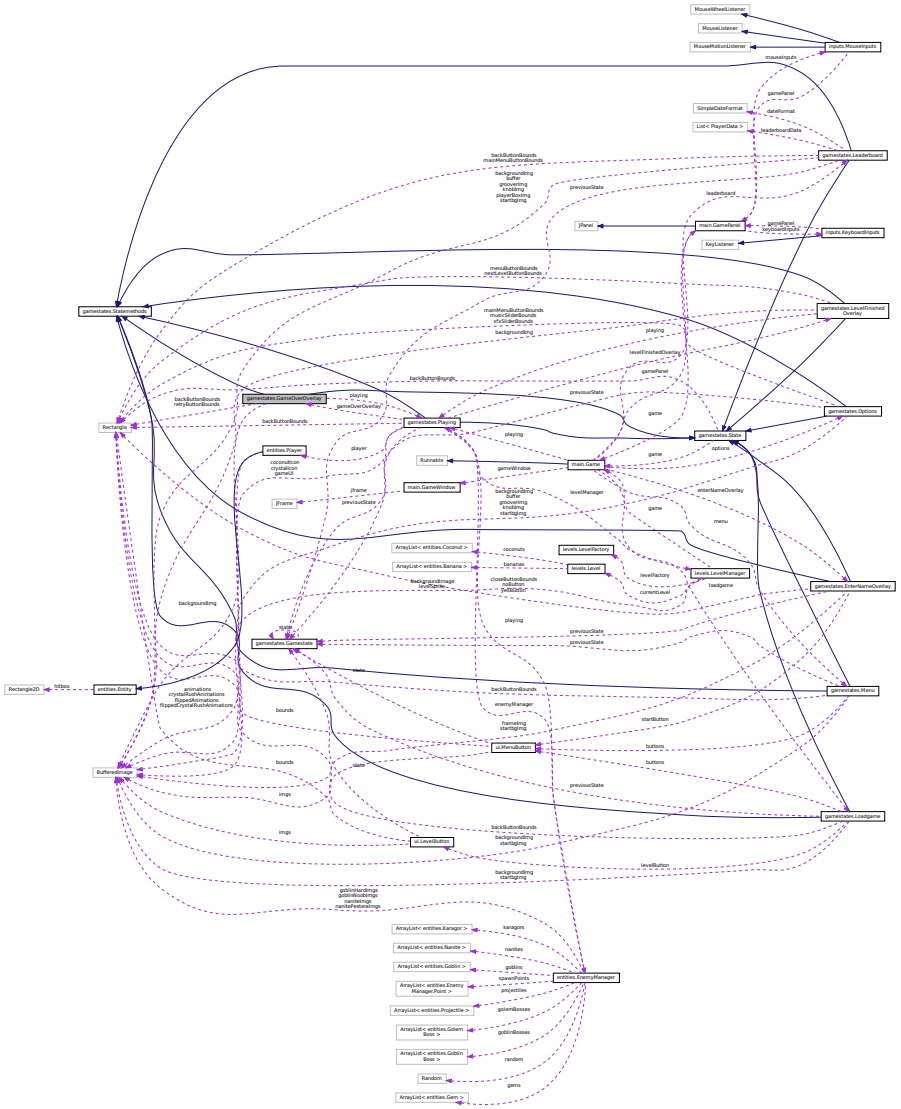
6.12 gamestates.GameOverOverlay Class Reference

Reprezintă overlay-ul afișat la sfârșitul jocului (Game Over).

Inheritance diagram for gamestates.GameOverOverlay:



Collaboration diagram for gamestates.GameOverOverlay:



Public Member Functions

- GameOverOverlay (Playing playing)

  Constructor pentru GameOverOverlay.
- void update ()

  Actualizează starea overlay-ului.
- void draw (Graphics g)

  Desenează elementele overlay-ului Game Over.
- void mouseClicked (MouseEvent e)

  Gestionarea evenimentului de click al mouse-ului (neutilizat).
- void mousePressed (MouseEvent e)

  Gestionează evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).
- void mouseMoved (MouseEvent e)

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).

- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionază evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" (suprascris din State pentru a folosi backButtonBounds local).
- void drawRetryButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Retry".

Private Member Functions

- void loadBackground ()

  Încarcă și configurează imaginea de fundal pentru ecranul Game Over.
- void loadCustomFont ()

  Încarcă fontul personalizat folosit pentru afișarea textului în overlay.

Private Attributes

- BufferedImage backgroundImg
- int bgX
- Rectangle retryButtonBounds
- Rectangle backButtonBounds
- Font overlayFont
- Playing playing

Static Private Attributes

- static final String DB_FILE = "data/gamedatabase.db"

  Calea către fișierul bazei de date pentru încărcarea scorului.

Additional Inherited Members

6.12.1 Detailed Description

Reprezintă overlay-ul afișat la sfârșitul jocului (Game Over).

Permite jucătorului să reîncerce nivelul sau să se întoarcă la meniul principal. Afișează scorul final. Extinde State și implementează Statemethods.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 GameOverOverlay()

```
gamestates.GameOverOverlay.GameOverOverlay (
            Playing playing )
```

Constructor pentru GameOverOverlay.

Parameters

playing	Referință la starea de joc Playing din care s-a ajuns aici.

6.12.3 Member Function Documentation

6.12.3.1 draw()

```
void gamestates.GameOverOverlay.draw (
            Graphics g )
```

Desenează elementele overlay-ului Game Over.

Include imaginea de fundal și scorul jucătorului. Opțional, desenează hitbox-urile butoanelor pentru depanare.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.12.3.2 drawBackButton()

```
void gamestates.GameOverOverlay.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" (suprascris din State pentru a folosi backButtonBounds local).

Folosit în scopuri de depanare.

Parameters

g2d	Contextul grafic 2D Graphics2D.

Reimplemented from gamestates.State.

6.12.3.3 drawRetryButton()

```
void gamestates.GameOverOverlay.drawRetryButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Retry".

Folosit în scopuri de depanare.

Parameters

g2d	Contextul grafic 2D Graphics2D.

6.12.3.4 keyPressed()

```
void gamestates.GameOverOverlay.keyPressed (
            KeyEvent e )
```

Gestionază evenimentele de apăsare a tastelor.

Permite reîncercarea nivelului (Enter) sau întoarcerea la meniu (Escape).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.12.3.5 keyReleased()

```
void gamestates.GameOverOverlay.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.12.3.6 loadBackground()

```
void gamestates.GameOverOverlay.loadBackground ( )  [private]
```

Încarcă și configurează imaginea de fundal pentru ecranul Game Over.

6.12.3.7 loadCustomFont()

```
void gamestates.GameOverOverlay.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat folosit pentru afișarea textului în overlay.

6.12.3.8 mouseClicked()

```
void gamestates.GameOverOverlay.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.12.3.9 mouseDragged()

```
void gamestates.GameOverOverlay.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.12.3.10 mouseMoved()

```
void gamestates.GameOverOverlay.mouseMoved (
            MouseEvent e )
```

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.12.3.11 mousePressed()

```
void gamestates.GameOverOverlay.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Verifică dacă s-a apăsat pe butonul de reîncercare sau pe cel de întoarcere la meniu.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.12.3.12 mouseReleased()

```
void gamestates.GameOverOverlay.mouseReleased (
            MouseEvent e )
```

Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.12.3.13 update()

```
void gamestates.GameOverOverlay.update ( )
```

Actualizează starea overlay-ului.

Momentan, nu face nimic deoarece overlay-ul este static.

Implements gamestates.Statemethods.

6.12.4 Member Data Documentation

6.12.4.1 backButtonBounds

```
Rectangle gamestates.GameOverOverlay.backButtonBounds  [private]
```

6.12.4.2 backgroundImg

```
BufferedImage gamestates.GameOverOverlay.backgroundImg  [private]
```

6.12.4.3 bgX

```
int gamestates.GameOverOverlay.bgX  [private]
```

6.12.4.4 DB_FILE

```
final String gamestates.GameOverOverlay.DB_FILE = "data/gamedatabase.db"  [static], [private]
```

Calea către fișierul bazei de date pentru încărcarea scorului.

6.12.4.5 overlayFont

```
Font gamestates.GameOverOverlay.overlayFont  [private]
```

6.12.4.6 playing

```
Playing gamestates.GameOverOverlay.playing  [private]
```

6.12.4.7 retryButtonBounds

```
Rectangle gamestates.GameOverOverlay.retryButtonBounds  [private]
```

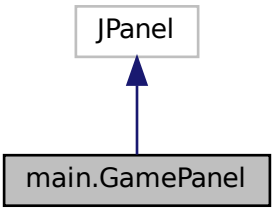
The documentation for this class was generated from the following file:

- src/gamestates/GameOverOverlay.java

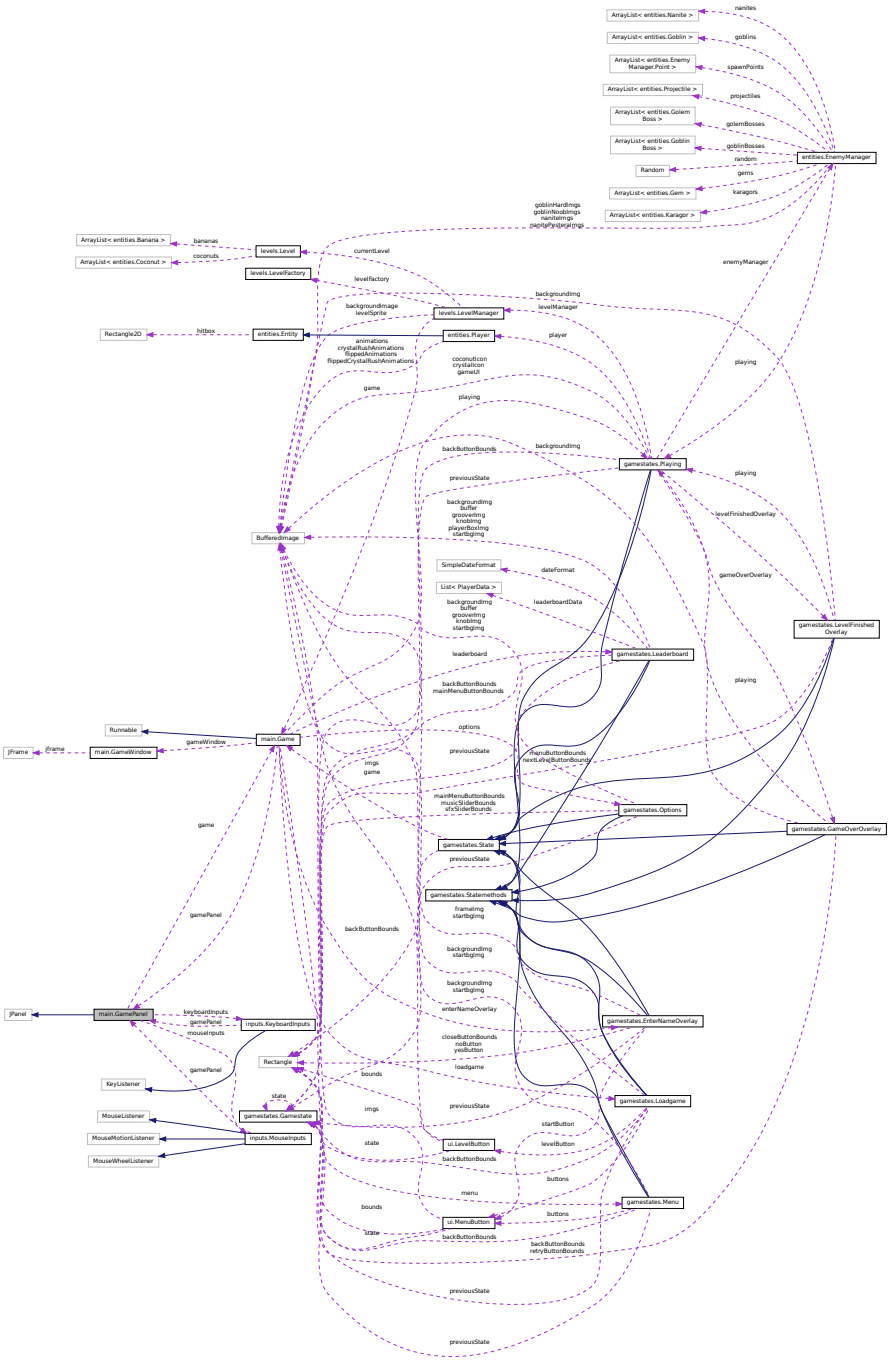
6.13 main.GamePanel Class Reference

Panoul principal al jocului, extinzând JPanel.

Inheritance diagram for main.GamePanel:



Collaboration diagram for main.GamePanel:



Public Member Functions

- GamePanel (Game game)

  Constructor pentru GamePanel.
- void updateGame ()

  Metodă destinată actualizării logicii jocului (momentan goală).
- void paintComponent (Graphics g)

  Suprascrie metoda JComponent#paintComponent(Graphics) pentru a desena conținutul jocului.
- Game getGame ()

  Returnează referința la instanța principală a jocului.

Private Member Functions

- void setPanelSize ()

  Setează dimensiunile preferate, maxime și minime ale panoului pe baza constantelor.

Private Attributes

- KeyboardInputs keyboardInputs

  Listener pentru input-ul de la tastatură.
- MouseInputs mouseInputs

  Listener pentru input-ul de la mouse.
- Game game

  Referință la instanța principală a jocului Game.

6.13.1 Detailed Description

Panoul principal al jocului, extinzând JPanel.

Acesta este componenta Swing pe care se desenează întregul joc. De asemenea, inițializează și atașează listener-ii pentru input-ul de la tastatură și mouse.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 GamePanel()

```
main.GamePanel.GamePanel (
            Game game )
```

Constructor pentru GamePanel.

Inițializează referința la joc, listener-ii de input, setează dimensiunea panoului și adaugă listener-ii la panou.

Parameters

game	Referință la instanța principală a jocului Game.

6.13.3 Member Function Documentation

6.13.3.1 getGame()

```
Game main.GamePanel.getGame ( )
```

Returnează referința la instanța principală a jocului.

Returns

Obiectul Game.

6.13.3.2 paintComponent()

```
void main.GamePanel.paintComponent (
            Graphics g )
```

Suprascrie metoda JComponent#paintComponent(Graphics) pentru a desena conținutul jocului.

Apelează metoda
`render`

a obiectului Game.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.13.3.3 setPanelSize()

```
void main.GamePanel.setPanelSize ( )  [private]
```

Setează dimensiunile preferate, maxime și minime ale panoului pe baza constantelor.
`GAME_WIDTH`

și
`GAME_HEIGHT`

din clasa Game.

6.13.3.4 updateGame()

```
void main.GamePanel.updateGame ( )
```

Metodă destinată actualizării logicii jocului (momentan goală).

Ar putea fi apelată din bucla principală a jocului dacă GamePanel ar gestiona direct actualizările.

6.13.4 Member Data Documentation

6.13.4.1 game

```
Game main.GamePanel.game  [private]
```

Referință la instanța principală a jocului Game.

6.13.4.2 keyboardInputs

```
KeyboardInputs main.GamePanel.keyboardInputs  [private]
```

Listener pentru input-ul de la tastatură.

6.13.4.3 mouseInputs

```
MouseInputs main.GamePanel.mouseInputs  [private]
```

Listener pentru input-ul de la mouse.

The documentation for this class was generated from the following file:

- src/main/GamePanel.java

6.14 gamestates.Gamestate Enum Reference

Enumerație ce definește diferitele stări posibile ale jocului.

Collaboration diagram for gamestates.Gamestate:

Public Attributes

- PLAYING

  Starea în care jucătorul controlează personajul și interacționează cu nivelul.
- MENU

  Starea meniului principal al jocului.
- OPTIONS

  Starea meniului de opțiuni/setări.
- QUIT

  Starea care indică intenția de a părăsi jocul.
- LOADGAME

  Starea pentru încărcarea unui joc salvat.
- LEADERBOARD

  Starea pentru afișarea clasamentului (leaderboard).
- ENTER_NAME

  Starea inițială sau pentru introducerea numelui jucătorului.

Static Public Attributes

- static Gamestate state = ENTER_NAME

  Starea curentă a jocului.

6.14.1 Detailed Description

Enumerație ce definește diferitele stări posibile ale jocului.

Fiecare constantă reprezintă o secțiune distinctă a jocului, cum ar fi meniul principal, jocul propriu-zis, etc.

6.14.2 Member Data Documentation

6.14.2.1 ENTER_NAME

`gamestates.Gamestate.ENTER_NAME`

Starea inițială sau pentru introducerea numelui jucătorului.

6.14.2.2 LEADERBOARD

`gamestates.Gamestate.LEADERBOARD`

Starea pentru afișarea clasamentului (leaderboard).

6.14.2.3 LOADGAME

`gamestates.Gamestate.LOADGAME`

Starea pentru încărcarea unui joc salvat.

6.14.2.4 MENU

`gamestates.Gamestate.MENU`

Starea meniului principal al jocului.

6.14.2.5 OPTIONS

`gamestates.Gamestate.OPTIONS`

Starea meniului de opțiuni/setări.

6.14.2.6 PLAYING

`gamestates.Gamestate.PLAYING`

Starea în care jucătorul controlează personajul și interacționează cu nivelul.

6.14.2.7 QUIT

`gamestates.Gamestate.QUIT`

Starea care indică intenția de a părăsi jocul.

6.14.2.8 state

`Gamestate gamestates.Gamestate.state = ENTER_NAME [static]`

Starea curentă a jocului.

Aceasta determină ce logică și randare sunt active.

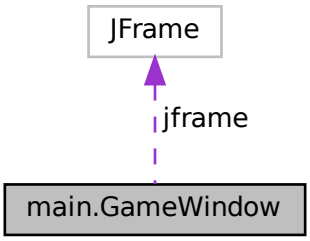
The documentation for this enum was generated from the following file:

- src/gamestates/Gamestate.java

6.15 main.GameWindow Class Reference

Reprezintă fereastra principală a jocului.

Collaboration diagram for main.GameWindow:



Public Member Functions

- GameWindow (GamePanel gamePanel)
  Constructor pentru GameWindow.

Private Attributes

- JFrame jframe
  Fereastra principală a jocului (obiectul JFrame).

6.15.1 Detailed Description

Reprezintă fereastra principală a jocului.

Această clasă creează și configurează obiectul JFrame care conține panoul jocului (GamePanel).

6.15.2 Constructor & Destructor Documentation

6.15.2.1 GameWindow()

```
main.GameWindow.GameWindow (
            GamePanel gamePanel )
```

Constructor pentru GameWindow.

Inițializează și configurează JFrame-ul, adaugă GamePanel-ul la acesta și setează un listener pentru focusul ferestrei.

Parameters

gamePanel	Panoul principal al jocului (GamePanel) care va fi afișat în fereastră.

Apelată când fereastra câștigă focusul.

Parameters

e	Evenimentul de focus al ferestrei.

Apelată când fereastra pierde focusul. Notifică jocul pentru a gestiona această situație (de ex., pauză, resetare input).

Parameters

e	Evenimentul de focus al ferestrei.

6.15.3 Member Data Documentation

6.15.3.1 jframe

```
JFrame main.GameWindow.jframe  [private]
```

Fereastra principală a jocului (obiectul JFrame).

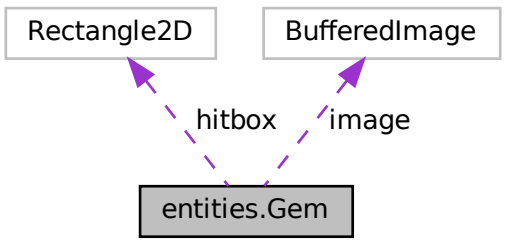
The documentation for this class was generated from the following file:

- src/main/GameWindow.java

6.16 entities.Gem Class Reference

Reprezintă un obiect de tip "Gem" (piatră prețioasă) în joc.

Collaboration diagram for entities.Gem:



Public Member Functions

- Gem (float x, float y, int levelId)

  Constructor pentru clasa Gem.
- void update ()

  Actualizează starea gem-ului.
- void draw (Graphics g, int xLvlOffset)

  Desenează gem-ul pe ecran.
- Rectangle2D.Float getHitbox ()

  Returnează hitbox-ul gem-ului.
- boolean isActive ()

  Verifică dacă gem-ul este activ.
- void setActive (boolean active)

  Setează starea de activare a gem-ului.

Private Attributes

- float x

  Coordonata x a colțului stânga-sus al gem-ului (pentru poziționarea inițială a hitbox-ului).
- float y

  Coordonata y a colțului stânga-sus al gem-ului (pentru poziționarea inițială a hitbox-ului).
- BufferedImage image

  Imaginea (sprite-ul) gem-ului.
- Rectangle2D.Float hitbox

  Dreptunghiul de coliziune (hitbox) al gem-ului.
- float originalY

  Coordonata Y inițială (de bază) pentru animația de plutire.
- float floatSpeed = 0.05f ∗ Game.SCALE

  Viteza animației de plutire.
- float floatAmplitude = 1.0f ∗ Game.SCALE

  Amplitudinea mișcării de plutire.
- float floatAngle = 0

  Unghiul curent în ciclul de plutire.
- float originalWidth

  Lățimea originală a imaginii gem-ului, scalată cu Game#SCALE.
- float originalHeight

  Înălțimea originală a imaginii gem-ului, scalată cu Game#SCALE.
- float scaleFactor = 1.0f

  Factorul de scalare curent pentru animația de respirație.
- float scaleSpeed = 0.005f

  Viteza cu care se schimbă factorul de scalare.
- float minScale = 0.9f

  Factorul minim de scalare.
- float maxScale = 1.1f

  Factorul maxim de scalare.
- boolean scalingUp = true

  Indică dacă gem-ul se mărește (true) sau se micșorează (false) în animația de respirație.
- boolean active = true

  Indică dacă gem-ul este activ (poate fi colectat și desenat).
- int levelId

  ID-ul nivelului în care a fost generat gem-ul, pentru a determina tipul de gem (culoarea).

6.16.1 Detailed Description

Reprezintă un obiect de tip "Gem" (piatră prețioasă) în joc.

Gem-urile sunt probabil obiecte speciale, posibil lăsate de boși la înfrângere. Această clasă gestionează animația de plutire și scalare a gem-ului. Nu extinde Entity direct, ci gestionează propriul hitbox.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 Gem()

```
entities.Gem.Gem (
            float x,
            float y,
            int levelId )
```

Constructor pentru clasa Gem.

Inițializează gem-ul la o poziție specificată, cu un sprite corespunzător ID-ului nivelului.

Parameters

x	Poziția x unde va fi centrat gem-ul.
y	Poziția y unde va fi centrat gem-ul.
levelId	ID-ul nivelului, folosit pentru a alege sprite-ul gem-ului.

6.16.3 Member Function Documentation

6.16.3.1 draw()

```
void entities.Gem.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează gem-ul pe ecran.

Imaginea este desenată la dimensiunile și poziția hitbox-ului curent (care include efectele de animație).

Parameters

g	Contextul grafic Graphics pe care se va desena.
xLvlOffset	Offset-ul orizontal al nivelului pentru scrolling.

6.16.3.2 getHitbox()

```
Rectangle2D.Float entities.Gem.getHitbox ( )
```

Returnează hitbox-ul gem-ului.

Returns

Obiectul Rectangle2D.Float reprezentând hitbox-ul.

6.16.3.3 isActive()

```
boolean entities.Gem.isActive ( )
```

Verifică dacă gem-ul este activ.

Returns

`true`
dacă gem-ul este activ,
`false`
altfel.

6.16.3.4 setActive()

```
void entities.Gem.setActive (
            boolean active )
```

Setează starea de activare a gem-ului.

Un gem inactiv nu este actualizat sau desenat și nu poate fi colectat.

Parameters

active	Noua stare de activare.

6.16.3.5 update()

```
void entities.Gem.update ( )
```

Actualizează starea gem-ului.

Gestionează animația de plutire și de "respirație" (scalare). Rulează doar dacă gem-ul este activ și are o imagine validă.

6.16.4 Member Data Documentation

6.16.4.1 active

```
boolean entities.Gem.active = true  [private]
```

Indică dacă gem-ul este activ (poate fi colectat și desenat).

6.16.4.2 floatAmplitude

```
float entities.Gem.floatAmplitude = 1.0f * Game.SCALE  [private]
```

Amplitudinea mișcării de plutire.

6.16.4.3 floatAngle

```
float entities.Gem.floatAngle = 0  [private]
```

Unghiul curent în ciclul de plutire.

6.16.4.4 floatSpeed

```
float entities.Gem.floatSpeed = 0.05f * Game.SCALE  [private]
```

Viteza animației de plutire.

6.16.4.5 hitbox

```
Rectangle2D.Float entities.Gem.hitbox  [private]
```

Dreptunghiul de coliziune (hitbox) al gem-ului.

6.16.4.6 image

```
BufferedImage entities.Gem.image  [private]
```

Imaginea (sprite-ul) gem-ului.

6.16.4.7 levelId

```
int entities.Gem.levelId  [private]
```

ID-ul nivelului în care a fost generat gem-ul, pentru a determina tipul de gem (culoarea).

6.16.4.8 maxScale

```
float entities.Gem.maxScale = 1.1f  [private]
```

Factorul maxim de scalare.

6.16.4.9 minScale

```
float entities.Gem.minScale = 0.9f  [private]
```

Factorul minim de scalare.

6.16.4.10 originalHeight

```
float entities.Gem.originalHeight  [private]
```

Înălțimea originală a imaginii gem-ului, scalată cu Game#SCALE.

6.16.4.11 originalWidth

```
float entities.Gem.originalWidth  [private]
```

Lățimea originală a imaginii gem-ului, scalată cu Game#SCALE.

6.16.4.12 originalY

```
float entities.Gem.originalY  [private]
```

Coordonata Y inițială (de bază) pentru animația de plutire.

6.16.4.13 scaleFactor

```
float entities.Gem.scaleFactor = 1.0f  [private]
```

Factorul de scalare curent pentru animația de respirație.

6.16.4.14 scaleSpeed

```
float entities.Gem.scaleSpeed = 0.005f  [private]
```

Viteza cu care se schimbă factorul de scalare.

6.16.4.15 scalingUp

```
boolean entities.Gem.scalingUp = true  [private]
```

Indică dacă gem-ul se mărește (true) sau se micșorează (false) în animația de respirație.

6.16.4.16 x

```
float entities.Gem.x  [private]
```

Coordonata x a colțului stânga-sus al gem-ului (pentru poziționarea inițială a hitbox-ului).

6.16.4.17 y

```
float entities.Gem.y  [private]
```

Coordonata y a colțului stânga-sus al gem-ului (pentru poziționarea inițială a hitbox-ului).

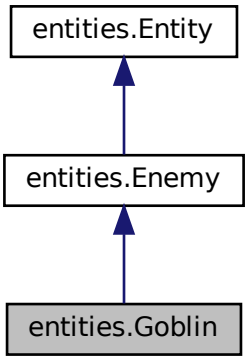
The documentation for this class was generated from the following file:

- src/entities/Gem.java

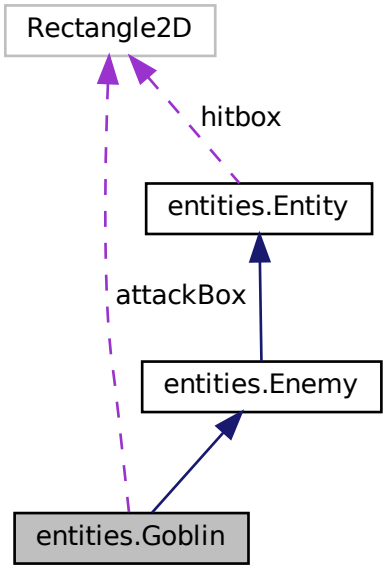
6.17 entities.Goblin Class Reference

Reprezintă entitatea Goblin în joc.

Inheritance diagram for entities.Goblin:



Collaboration diagram for entities.Goblin:



Public Member Functions

- Goblin (float x, float y, int width, int height, int enemyType)

  Constructor pentru clasa Goblin.
- void update (Rectangle2D.Float playerHitbox)

  Actualizează starea Goblinului.
- void takeDamage (int damage)

  Aplică daune Goblinului atunci când este atacat de jucător.
- boolean checkPlayerHit (Rectangle2D.Float playerHitbox)

  Verifică dacă jucătorul atinge Goblinul și aplică daune dacă este cazul.
- boolean canAttackPlayer (Rectangle2D.Float playerHitbox)

  Verifică dacă Goblinul poate ataca jucătorul (dacă jucătorul este în raza de atac).
- void playerDetected (float playerX, float playerY)

  Detectează jucătorul dacă se află în raza de vizualizare și ajustează direcția Goblinului pentru a se îndrepta către jucător.
- void setLevelData (int[ ][ ] levelData)

  Setează datele nivelului pentru Goblin.
- boolean isActive ()

  Verifică dacă Goblinul este activ în joc.
- int getHealth ()

  Returnează sănătatea curentă a Goblinului.
- int getMaxHealth ()

  Returnează sănătatea maximă a Goblinului.
- int getDamage ()

  Returnează daunele de atac ale Goblinului.
- int getDirection ()

  Returnează direcția curentă a Goblinului.
- void makeBoss ()

  Transformă acest Goblin într-un boss (Karagor).
- Rectangle2D.Float getAttackBox ()

  Returnează hitbox-ul de atac al Goblinului.
- int getGoblinType ()

  Returnează tipul Goblinului (de exemplu, NOOB, HARD).

Static Public Attributes

- static final int GOBLIN_NOOB = 0

  Tipul de Goblin "Noob".
- static final int GOBLIN_HARD = 1

  Tipul de Goblin "Hard".
- static final int IDLE = 3

  Starea IDLE a Goblinului.
- static final int RUNNING = 10

  Starea RUNNING (alergare) a Goblinului.
- static final int ATTACK = 12

  Starea ATTACK (atac) a Goblinului (corespunde animației SLASHING).
- static final int HURT = 2

  Starea HURT (lovit) a Goblinului.
- static final int DYING = 0

  Starea DYING (moarte) a Goblinului.

Private Member Functions

- void updateAttackBox ()

  Actualizează poziția și dimensiunea hitbox-ului de atac în funcție de direcția Goblinului.
- void updateCooldowns ()

  Actualizează cronometrele de cooldown pentru atac și atingerea jucătorului.
- void initPatrolBoundaries ()

  Inițializează limitele de patrulare pentru Goblin pe baza poziției sale curente.
- boolean canSeePlayer (Rectangle2D.Float playerHitbox)

  Verifică dacă Goblinul poate vedea jucătorul.
- void updateBehavior (Rectangle2D.Float playerHitbox)

  Actualizează comportamentul Goblinului pe baza stării curente și a interacțiunii cu jucătorul.
- void updatePosition ()

  Actualizează poziția Goblinului.
- boolean willLandOnGround (float x, float y)

  Verifică dacă Goblinul va ateriza pe o suprafață solidă la poziția specificată.
- void checkAttackHit ()

  Verifică dacă atacul Goblinului a lovit jucătorul.
- void setState (int state)

  Setează starea curentă a Goblinului și resetează cronometrul pentru starea respectivă.

Private Attributes

- int health

  Sănătatea curentă a Goblinului.
- int maxHealth

  Sănătatea maximă a Goblinului.
- int damage

  Daunele provocate de atacul Goblinului.
- boolean isActive = true

  Indicator dacă Goblinul este activ în joc.
- int attackRange = 30

  Distanța de atac a Goblinului.
- int ticksInState = 0

  Numărul de tick-uri petrecute în starea curentă.
- int attackCooldown = 0

  Cooldown-ul dintre atacuri.
- boolean attackChecked = false

  Indicator dacă lovitura de atac a fost verificată în animația curentă.
- boolean playerDetected = false

  Indicator dacă jucătorul a fost detectat de Goblin.
- float leftPatrolLimit

  Limita stângă a zonei de patrulare.
- float rightPatrolLimit

  Limita dreaptă a zonei de patrulare.
- boolean patrolBoundariesSet = false

  Indicator dacă limitele de patrulare au fost setate.
- boolean isMoving = false

  Indicator dacă Goblinul se mișcă.
- int direction = 1

Direcția de mișcare a Goblinului (1 = dreapta, -1 = stânga).

- float moveSpeed = 0.5f

  Viteza de mișcare curentă a Goblinului.
- float patrolMoveSpeed

  Viteza de mișcare în timpul patrulării.
- float chaseMoveSpeed

  Viteza de mișcare în timpul urmăririi jucătorului.
- int[ ][ ] levelData

  Datele nivelului curent, folosite pentru coliziuni și navigație.
- boolean inAir = false

  Indicator dacă Goblinul se află în aer.
- float airSpeed = 0f

  Viteza verticală a Goblinului în aer.
- float gravity = 0.04f

  Valoarea gravitației aplicate Goblinului.
- float jumpSpeed = -3.5f ∗ Game.SCALE

  Viteza de săritură a Goblinului.
- Rectangle2D.Float attackBox

  Hitbox-ul pentru atacul Goblinului.
- int playerTouchCooldown = 0

  Cooldown-ul pentru daunele provocate prin atingerea jucătorului.
- int touchDamageCooldown = 60

  Cooldown-ul dintre daunele provocate prin atingere (în frame-uri).
- float patrolDistance = 100.0f

  Distanța pe care Goblinul o patrulează în fiecare direcție de la punctul de start.
- int detectionRange = 300

  Distanța la care Goblinul poate detecta jucătorul.

Static Private Attributes

- static final int ATTACK_COOLDOWN_MAX = 120

  Cooldown-ul maxim dintre atacuri (aproximativ 2 secunde la 60 FPS).

Additional Inherited Members

6.17.1 Detailed Description

Reprezintă entitatea Goblin în joc.

Această clasă extinde clasa Enemy și definește comportamentul specific și atributele pentru Goblini, inclusiv tipurile (Noob, Hard), stările, logica de patrulare, atac și interacțiunea cu jucătorul.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 Goblin()

```
entities.Goblin.Goblin (
            float x,
            float y,
            int width,
            int height,
            int enemyType )
```

Constructor pentru clasa Goblin.

Inițializează Goblinul cu o poziție, dimensiuni și tip specific. Setează atributele în funcție de tip (Noob sau Hard), inițializează parametrii de patrulare, starea inițială și hitbox-ul.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
width	Lățimea entității.
height	Înălțimea entității.
enemyType	Tipul Goblinului (GOBLIN_NOOB sau GOBLIN_HARD).

6.17.3 Member Function Documentation

6.17.3.1 canAttackPlayer()

```
boolean entities.Goblin.canAttackPlayer (
            Rectangle2D.Float playerHitbox )
```

Verifică dacă Goblinul poate ataca jucătorul (dacă jucătorul este în raza de atac).

Dacă da, inițiază starea de atac.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă Goblinul poate ataca și a inițiat atacul, false altfel.

6.17.3.2 canSeePlayer()

```
boolean entities.Goblin.canSeePlayer (
            Rectangle2D.Float playerHitbox )  [private]
```

Verifică dacă Goblinul poate vedea jucătorul.

Condițiile includ distanța, alinierea verticală și direcția în care privește Goblinul.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă jucătorul este vizibil, false altfel.

6.17.3.3 checkAttackHit()

```
void entities.Goblin.checkAttackHit ( )  [private]
```

Verifică dacă atacul Goblinului a lovit jucătorul.

Într-un joc real, aici s-ar verifica coliziunea cu hitbox-ul jucătorului și s-ar aplica daune dacă există o coliziune.

6.17.3.4 checkPlayerHit()

```
boolean entities.Goblin.checkPlayerHit (
            Rectangle2D.Float playerHitbox )
```

Verifică dacă jucătorul atinge Goblinul și aplică daune dacă este cazul.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă jucătorul a lovit Goblinul și daunele au fost aplicate, false altfel.

6.17.3.5 getAttackBox()

```
Rectangle2D.Float entities.Goblin.getAttackBox ( )
```

Returnează hitbox-ul de atac al Goblinului.

Returns

Dreptunghiul reprezentând hitbox-ul de atac.

6.17.3.6 getDamage()

```
int entities.Goblin.getDamage ( )
```

Returnează daunele de atac ale Goblinului.

Returns

Daunele de atac.

6.17.3.7 getDirection()

```
int entities.Goblin.getDirection ( )
```

Returnează direcția curentă a Goblinului.

Returns

Direcția (1 pentru dreapta, -1 pentru stânga).

6.17.3.8 getGoblinType()

```
int entities.Goblin.getGoblinType ( )
```

Returnează tipul Goblinului (de exemplu, NOOB, HARD).

Acesta corespunde valorii `enemyType` transmise în constructor.

Returns

Tipul Goblinului.

6.17.3.9 getHealth()

```
int entities.Goblin.getHealth ( )
```

Returnează sănătatea curentă a Goblinului.

Returns

Sănătatea curentă.

6.17.3.10 getMaxHealth()

```
int entities.Goblin.getMaxHealth ( )
```

Returnează sănătatea maximă a Goblinului.

Returns

Sănătatea maximă.

6.17.3.11 initPatrolBoundaries()

```
void entities.Goblin.initPatrolBoundaries ( )  [private]
```

Inițializează limitele de patrulare pentru Goblin pe baza poziției sale curente.

Această metodă ar trebui apelată după ce Goblinul s-a așezat pe o platformă. Caută marginile platformei pentru a stabili limitele.

6.17.3.12 isActive()

```
boolean entities.Goblin.isActive ( )
```

Verifică dacă Goblinul este activ în joc.

Returns

true dacă Goblinul este activ, false altfel.

Reimplemented from entities.Enemy.

6.17.3.13 makeBoss()

```
void entities.Goblin.makeBoss ( )
```

Transformă acest Goblin într-un boss (Karagor).

Mărește dimensiunea, sănătatea, daunele și alte atribute. Această metodă modifică instanța curentă de Goblin.

6.17.3.14 playerDetected()

```
void entities.Goblin.playerDetected (
            float playerX,
            float playerY )
```

Detectează jucătorul dacă se află în raza de vizualizare și ajustează direcția Goblinului pentru a se îndrepta către jucător.

Trece în modul de urmărire (chase) dacă jucătorul este detectat.

Parameters

playerX	Coordonata X a jucătorului.
playerY	Coordonata Y a jucătorului.

6.17.3.15 setLevelData()

```
void entities.Goblin.setLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Goblin.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

6.17.3.16 setState()

```
void entities.Goblin.setState (
            int state )  [private]
```

Setează starea curentă a Goblinului și resetează cronometrul pentru starea respectivă.

Parameters

state	Noua stare a Goblinului.

6.17.3.17 takeDamage()

```
void entities.Goblin.takeDamage (
            int damage )
```

Aplică daune Goblinului atunci când este atacat de jucător.

Reduce sănătatea și gestionează tranziția la starea HURT sau DYING.

Parameters

damage	Cantitatea de daune primite.

6.17.3.18 update()

```
void entities.Goblin.update (
            Rectangle2D.Float playerHitbox )
```

Actualizează starea Goblinului.

Aceasta include actualizarea hitbox-ului de atac, cooldown-urilor, comportamentului și poziției. De asemenea, inițializează limitele de patrulare dacă este necesar.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru interacțiuni.

Reimplemented from entities.Enemy.

6.17.3.19 updateAttackBox()

```
void entities.Goblin.updateAttackBox ( )  [private]
```

Actualizează poziția și dimensiunea hitbox-ului de atac în funcție de direcția Goblinului.

6.17.3.20 updateBehavior()

```
void entities.Goblin.updateBehavior (
            Rectangle2D.Float playerHitbox )  [private]
```

Actualizează comportamentul Goblinului pe baza stării curente și a interacțiunii cu jucătorul.

Gestionează tranzițiile între stări (IDLE, RUNNING, ATTACK, HURT, DYING) și logica specifică fiecărei stări.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru a lua decizii.

6.17.3.21 updateCooldowns()

```
void entities.Goblin.updateCooldowns ( )  [private]
```

Actualizează cronometrele de cooldown pentru atac și atingerea jucătorului.

6.17.3.22 updatePosition()

```
void entities.Goblin.updatePosition ( )  [private]
```

Actualizează poziția Goblinului.

Aplică gravitația, gestionează mișcarea verticală și orizontală, verifică coliziunile cu pereții și podeaua, și previne căderea de pe platforme.

6.17.3.23 willLandOnGround()

```
boolean entities.Goblin.willLandOnGround (
            float x,
            float y )  [private]
```

Verifică dacă Goblinul va ateriza pe o suprafață solidă la poziția specificată.

Parameters

x	Coordonata X a poziției viitoare.
y	Coordonata Y a poziției viitoare.

Returns

true dacă Goblinul va ateriza pe sol, false altfel.

6.17.4 Member Data Documentation

6.17.4.1 airSpeed

```
float entities.Goblin.airSpeed = 0f  [private]
```

Viteza verticală a Goblinului în aer.

6.17.4.2 ATTACK

```
final int entities.Goblin.ATTACK = 12  [static]
```

Starea ATTACK (atac) a Goblinului (corespunde animației SLASHING).

6.17.4.3 ATTACK_COOLDOWN_MAX

```
final int entities.Goblin.ATTACK_COOLDOWN_MAX = 120  [static], [private]
```

Cooldown-ul maxim dintre atacuri (aproximativ 2 secunde la 60 FPS).

6.17.4.4 attackBox

```
Rectangle2D.Float entities.Goblin.attackBox  [private]
```

Hitbox-ul pentru atacul Goblinului.

6.17.4.5 attackChecked

```
boolean entities.Goblin.attackChecked = false  [private]
```

Indicator dacă lovitura de atac a fost verificată în animația curentă.

6.17.4.6 attackCooldown

```
int entities.Goblin.attackCooldown = 0  [private]
```

Cooldown-ul dintre atacuri.

6.17.4.7 attackRange

```
int entities.Goblin.attackRange = 30  [private]
```

Distanța de atac a Goblinului.

6.17.4.8 chaseMoveSpeed

```
float entities.Goblin.chaseMoveSpeed  [private]
```

Viteza de mișcare în timpul urmăririi jucătorului.

6.17.4.9 damage

```
int entities.Goblin.damage  [private]
```

Daunele provocate de atacul Goblinului.

6.17.4.10 detectionRange

```
int entities.Goblin.detectionRange = 300  [private]
```

Distanța la care Goblinul poate detecta jucătorul.

6.17.4.11 direction

```
int entities.Goblin.direction = 1  [private]
```

Direcția de mișcare a Goblinului (1 = dreapta, -1 = stânga).

6.17.4.12 DYING

```
final int entities.Goblin.DYING = 0  [static]
```

Starea DYING (moarte) a Goblinului.

6.17.4.13 GOBLIN_HARD

```
final int entities.Goblin.GOBLIN_HARD = 1  [static]
```

Tipul de Goblin "Hard".

6.17.4.14 GOBLIN_NOOB

```
final int entities.Goblin.GOBLIN_NOOB = 0  [static]
```

Tipul de Goblin "Noob".

6.17.4.15 gravity

```
float entities.Goblin.gravity = 0.04f  [private]
```

Valoarea gravitației aplicate Goblinului.

6.17.4.16 health

```
int entities.Goblin.health  [private]
```

Sănătatea curentă a Goblinului.

6.17.4.17 HURT

```
final int entities.Goblin.HURT = 2  [static]
```

Starea HURT (lovit) a Goblinului.

6.17.4.18 IDLE

```
final int entities.Goblin.IDLE = 3  [static]
```

Starea IDLE a Goblinului.

6.17.4.19 inAir

```
boolean entities.Goblin.inAir = false  [private]
```

Indicator dacă Goblinul se află în aer.

6.17.4.20 isActive

```
boolean entities.Goblin.isActive = true  [private]
```

Indicator dacă Goblinul este activ în joc.

6.17.4.21 isMoving

```
boolean entities.Goblin.isMoving = false  [private]
```

Indicator dacă Goblinul se mișcă.

6.17.4.22 jumpSpeed

```
float entities.Goblin.jumpSpeed = -3.5f * Game.SCALE  [private]
```

Viteza de săritură a Goblinului.

6.17.4.23 leftPatrolLimit

```
float entities.Goblin.leftPatrolLimit  [private]
```

Limita stângă a zonei de patrulare.

6.17.4.24 levelData

```
int [][] entities.Goblin.levelData  [private]
```

Datele nivelului curent, folosite pentru coliziuni și navigație.

6.17.4.25 maxHealth

```
int entities.Goblin.maxHealth  [private]
```

Sănătatea maximă a Goblinului.

6.17.4.26 moveSpeed

```
float entities.Goblin.moveSpeed = 0.5f  [private]
```

Viteza de mișcare curentă a Goblinului.

6.17.4.27 patrolBoundariesSet

```
boolean entities.Goblin.patrolBoundariesSet = false  [private]
```

Indicator dacă limitele de patrulare au fost setate.

6.17.4.28 patrolDistance

```
float entities.Goblin.patrolDistance = 100.0f  [private]
```

Distanța pe care Goblinul o patrulează în fiecare direcție de la punctul de start.

6.17.4.29 patrolMoveSpeed

```
float entities.Goblin.patrolMoveSpeed  [private]
```

Viteza de mișcare în timpul patrulării.

6.17.4.30 playerDetected

```
boolean entities.Goblin.playerDetected = false  [private]
```

Indicator dacă jucătorul a fost detectat de Goblin.

6.17.4.31 playerTouchCooldown

```
int entities.Goblin.playerTouchCooldown = 0  [private]
```

Cooldown-ul pentru daunele provocate prin atingerea jucătorului.

6.17.4.32 rightPatrolLimit

```
float entities.Goblin.rightPatrolLimit  [private]
```

Limita dreaptă a zonei de patrulare.

6.17.4.33 RUNNING

```
final int entities.Goblin.RUNNING = 10  [static]
```

Starea RUNNING (alergare) a Goblinului.

6.17.4.34 ticksInState

```
int entities.Goblin.ticksInState = 0  [private]
```

Numărul de tick-uri petrecute în starea curentă.

6.17.4.35 touchDamageCooldown

```
int entities.Goblin.touchDamageCooldown = 60  [private]
```

Cooldown-ul dintre daunele provocate prin atingere (în frame-uri).

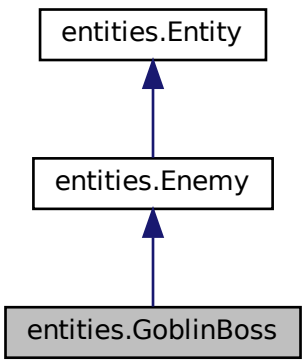
The documentation for this class was generated from the following file:

- src/entities/Goblin.java

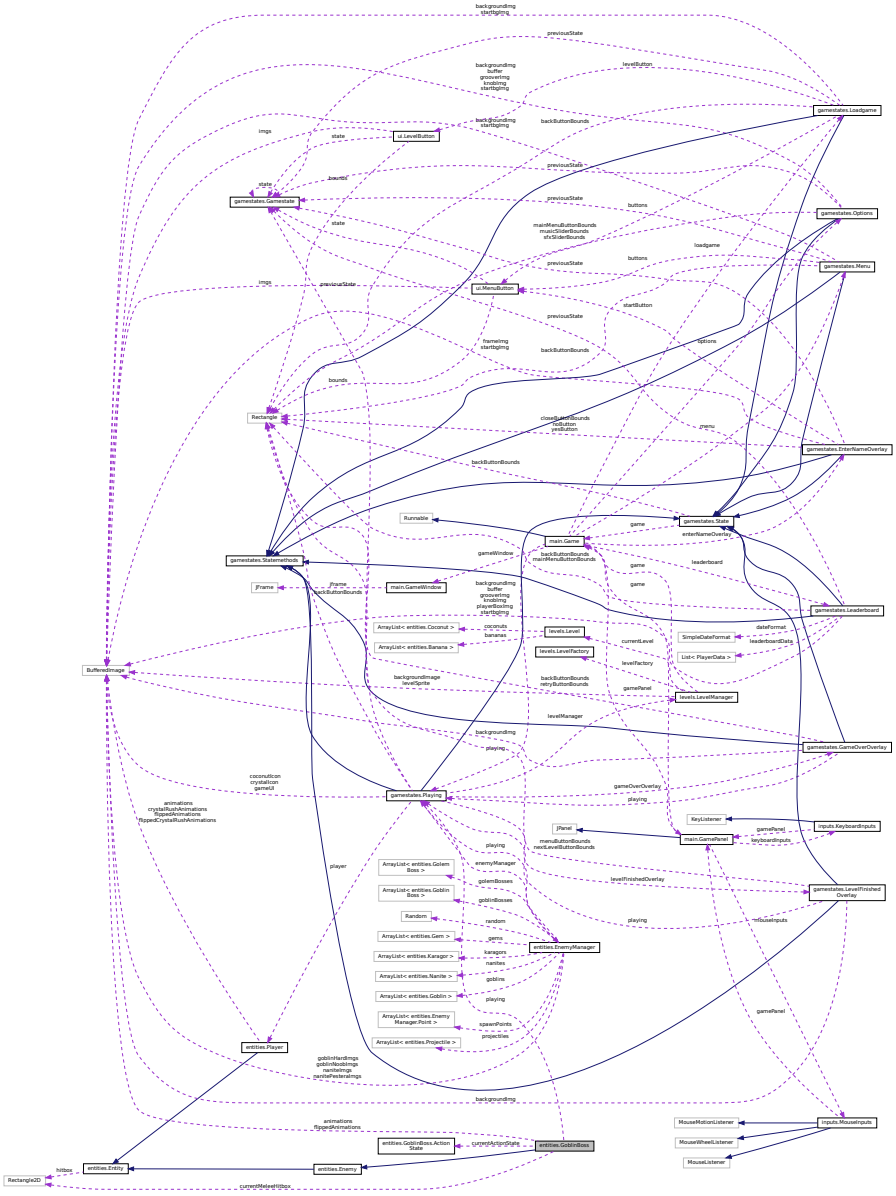
6.18 entities.GoblinBoss Class Reference

Reprezintă entitatea Goblin Boss în joc.

Inheritance diagram for entities.GoblinBoss:



Collaboration diagram for entities.GoblinBoss:



Classes

- enum ActionState

  Enumerare pentru stările de acțiune posibile ale Goblin Boss-ului.

Public Member Functions

- GoblinBoss (float x, float y, Playing playing)

  Constructor pentru clasa GoblinBoss.
- void update (Player player, int[ ][ ] lvlData)

  Actualizează starea Goblin Boss-ului.
- void setLevelData (int[ ][ ] levelData)

Setează datele nivelului pentru Goblin Boss.

- void render (Graphics g, int xLvlOffset)

  Randează Goblin Boss-ul pe ecran.
- boolean isAlive ()

  Verifică dacă Goblin Boss-ul este în viață.
- void takeDamage (int damage)

  Aplică daune Goblin Boss-ului.
- int getAttackDamage ()

  Returnează daunele de atac ale Goblin Boss-ului.
- int getCurrentHealth ()

  Returnează sănătatea curentă a Goblin Boss-ului.
- int getMaxHealth ()

  Returnează sănătatea maximă a Goblin Boss-ului.

Protected Attributes

- int direction = Enemy_Animation_Rows.Directions.LEFT

  Direcția curentă a Goblin Boss-ului (stânga sau dreapta).

Private Member Functions

- float getDistance (Rectangle2D.Float r1, Rectangle2D.Float r2)

  Calculează distanța euclidiană între centrele a două dreptunghiuri.
- void loadAnimations ()

  Încarcă animațiile pentru Goblin Boss din sprite sheet.
- void updatePlayerDetection (Player player)

  Actualizează starea de detectare a jucătorului.
- void handleStateMachine (Player player)

  Gestionează mașina de stări a Goblin Boss-ului.
- void decideNextAction (Player player)

  Decide următoarea acțiune a boss-ului când jucătorul este detectat.
- void initiateMeleeAttack ()

  Inițiază un atac melee standard (lovitură sau picior).
- void initiateRunSlashAttack ()

  Inițiază un atac de tip "run slash".
- void initiateReposition (boolean preferFastSlide)

  Inițiază o acțiune de repoziționare.
- void initiateSlideReposition ()

  Inițiază o repoziționare prin alunecare (slide).
- void initiateWalkReposition ()

  Inițiază o repoziționare prin mers.
- void handleRepositionMovementLogic (float speed, Enemy_Animation_Rows anim, float timeoutTicks)

  Gestionează logica de mișcare pentru repoziționare (mers sau alunecare).
- void decideIdleAction ()

  Decide o acțiune aleatorie pentru starea IDLE.
- void moveTowardsPlayer (Rectangle2D.Float playerHitbox, float speed)

  Mută Goblin Boss-ul către jucător.
- void updateCurrentMeleeHitbox ()

  Actualizează hitbox-ul pentru atacurile melee.

- void applyMeleeDamage (Player player)

  Aplică daune jucătorului dacă atacul melee îl lovește.
- void setBossAnimation (Enemy_Animation_Rows animType)

  Setează animația curentă a Goblin Boss-ului.
- void applyGravity ()

  Aplică gravitația Goblin Boss-ului dacă acesta nu se află pe o suprafață solidă.
- void updateHitbox ()

  Actualizează poziția hitbox-ului.

Static Private Member Functions

- static int calculateBossHitboxWidth ()

  Calculează lățimea hitbox-ului pentru Goblin Boss.
- static int calculateBossHitboxHeight ()

  Calculează înălțimea hitbox-ului pentru Goblin Boss.
- static BufferedImage flipImage (BufferedImage image)

  Inversează (flip) o imagine pe orizontală.

Private Attributes

- BufferedImage[ ][ ] animations

  Matrice bidimensională pentru stocarea animațiilor normale ale boss-ului.
- BufferedImage[ ][ ] flippedAnimations

  Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale boss-ului.
- int maxHealthBoss = 300

  Sănătatea maximă a Goblin Boss-ului.
- int currentHealthBoss

  Sănătatea curentă a Goblin Boss-ului.
- int attackDamageBoss = 10

  Daunele provocate de atacul Goblin Boss-ului.
- final int DRAW_WIDTH = (int) (Constants.EnemyConstants.GOBLIN_DRAW_WIDTH_DEFAULT ∗ BOSS_SCALE_FACTOR ∗ Game.SCALE)

  Lățimea de desenare a Goblin Boss-ului, ajustată cu factorul de scalare și scala jocului.
- final int DRAW_HEIGHT = (int) (Constants.EnemyConstants.GOBLIN_DRAW_HEIGHT_DEFAULT ∗ BOSS_SCALE_FACTOR ∗ Game.SCALE)

  Înălțimea de desenare a Goblin Boss-ului, ajustată cu factorul de scalare și scala jocului.
- float xDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_X ∗ BOSS_SCALE_FACTOR

  Decalajul pe axa X pentru desenarea sprite-ului Goblin Boss-ului.
- float yDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_Y ∗ BOSS_SCALE_FACTOR

  Decalajul pe axa Y pentru desenarea sprite-ului Goblin Boss-ului.
- int[ ][ ] levelData

  Datele nivelului curent, folosite pentru coliziuni și navigație.
- Playing playing

  Referință la starea de joc "Playing".
- ActionState currentActionState = ActionState.IDLE

  Starea de acțiune curentă a Goblin Boss-ului.
- boolean playerDetected = false

  Indicator dacă jucătorul a fost detectat de Goblin Boss.
- float detectionRange = 450f ∗ Game.SCALE

Distanța la care Goblin Boss-ul poate detecta jucătorul.

- float sightRange = 500f ∗ Game.SCALE

  Distanța vizuală a Goblin Boss-ului.
- float meleeAttackRange = hitbox.width ∗ 0.7f

  Distanța de la care Goblin Boss-ul poate efectua un atac melee.
- float walkSpeed = 0.6f ∗ Game.SCALE

  Viteza de mers a Goblin Boss-ului.
- float runSpeed = 1.8f ∗ Game.SCALE

  Viteza de alergare a Goblin Boss-ului.
- float slideSpeed = 3.0f ∗ Game.SCALE

  Viteza de alunecare (slide) a Goblin Boss-ului.
- int actionTimer = 0

  Cronometru pentru durata acțiunilor curente.
- int actionCooldown = 0

  Cronometru pentru cooldown-ul dintre acțiuni.
- final int IDLE_DURATION_MIN = 60

  Durata minimă a stării IDLE.
- final int IDLE_DURATION_MAX = 120

  Durata maximă a stării IDLE.
- final int PREPARE_ATTACK_DURATION = 45

  Durata pregătirii unui atac.
- final int ATTACK_COOLDOWN_MAX = 90

  Cooldown-ul maxim după un atac.
- final int REPOSITION_COOLDOWN_MAX = 75

  Cooldown-ul maxim după o repoziționare.
- boolean isPerformingAction = false

  Indicator dacă Goblin Boss-ul execută o acțiune în prezent.
- int attackCheckFrame

  Frame-ul specific din animația de atac la care se verifică aplicarea daunelor.
- boolean attackDamageAppliedThisAttack

  Indicator dacă daunele au fost aplicate în timpul atacului curent.
- Rectangle2D.Float currentMeleeHitbox

  Hitbox-ul curent pentru atacurile melee.
- float targetX

  Coordonata X țintă pentru mișcările de repoziționare.

Static Private Attributes

- static final float BOSS_SCALE_FACTOR = 2.0f

  Factorul de scalare specific pentru dimensiunile Goblin Boss-ului.

Additional Inherited Members

6.18.1 Detailed Description

Reprezintă entitatea Goblin Boss în joc.

Această clasă extinde clasa Enemy și definește comportamentul specific și atributele pentru Goblin Boss, inclusiv animațiile, stările de acțiune, logica de atac și interacțiunea cu jucătorul.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 GoblinBoss()

```
entities.GoblinBoss.GoblinBoss (
            float x,
            float y,
            Playing playing )
```

Constructor pentru clasa GoblinBoss.

Inițializează Goblin Boss-ul cu o poziție specifică și o referință la starea de joc "Playing". Setează sănătatea, încarcă animațiile și inițializează hitbox-ul.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
playing	Referință la obiectul Playing, pentru interacțiuni cu starea jocului.

6.18.3 Member Function Documentation

6.18.3.1 applyGravity()

```
void entities.GoblinBoss.applyGravity ( )  [private]
```

Aplică gravitația Goblin Boss-ului dacă acesta nu se află pe o suprafață solidă.

6.18.3.2 applyMeleeDamage()

```
void entities.GoblinBoss.applyMeleeDamage (
            Player player )  [private]
```

Aplică daune jucătorului dacă atacul melee îl lovește.

Verifică intersecția dintre hitbox-ul de atac și hitbox-ul jucătorului. Aplică și un efect de knockback jucătorului.

Parameters

player	Jucătorul care poate fi lovit.

6.18.3.3 calculateBossHitboxHeight()

```
static int entities.GoblinBoss.calculateBossHitboxHeight ( )  [static], [private]
```

Calculează înălțimea hitbox-ului pentru Goblin Boss.

Aceasta se bazează pe dimensiunile sprite-ului sursă și factorii de scalare.

Returns

Înălțimea calculată a hitbox-ului.

6.18.3.4 calculateBossHitboxWidth()

```
static int entities.GoblinBoss.calculateBossHitboxWidth ( )  [static], [private]
```

Calculează lățimea hitbox-ului pentru Goblin Boss.

Aceasta se bazează pe dimensiunile sprite-ului sursă și factorii de scalare.

Returns

Lățimea calculată a hitbox-ului.

6.18.3.5 decideIdleAction()

```
void entities.GoblinBoss.decideIdleAction ( )  [private]
```

Decide o acțiune aleatorie pentru starea IDLE.

Boss-ul poate să se întoarcă, să meargă puțin sau să rămână pe loc.

6.18.3.6 decideNextAction()

```
void entities.GoblinBoss.decideNextAction (
            Player player )  [private]
```

Decide următoarea acțiune a boss-ului când jucătorul este detectat.

Aceasta poate fi un atac, urmărirea jucătorului sau repoziționarea.

Parameters

player	Jucătorul, pentru a evalua distanța și a lua decizii.

6.18.3.7 flipImage()

```
static BufferedImage entities.GoblinBoss.flipImage (
            BufferedImage image )  [static], [private]
```

Inversează (flip) o imagine pe orizontală.

Parameters

image	Imaginea care trebuie inversată.

Returns

Imaginea inversată sau null dacă imaginea de intrare este null.

6.18.3.8 getAttackDamage()

```
int entities.GoblinBoss.getAttackDamage ( )
```

Returnează daunele de atac ale Goblin Boss-ului.

Returns

Daunele de atac.

6.18.3.9 getCurrentHealth()

```
int entities.GoblinBoss.getCurrentHealth ( )
```

Returnează sănătatea curentă a Goblin Boss-ului.

Returns

Sănătatea curentă.

6.18.3.10 getDistance()

```
float entities.GoblinBoss.getDistance (
            Rectangle2D.Float r1,
            Rectangle2D.Float r2 )  [private]
```

Calculează distanța euclidiană între centrele a două dreptunghiuri.

Parameters

r1	Primul dreptunghi.
r2	Al doilea dreptunghi.

Returns

Distanța dintre centrele celor două dreptunghiuri.

6.18.3.11 getMaxHealth()

```
int entities.GoblinBoss.getMaxHealth ( )
```

Returnează sănătatea maximă a Goblin Boss-ului.

Returns

Sănătatea maximă.

6.18.3.12 handleRepositionMovementLogic()

```
void entities.GoblinBoss.handleRepositionMovementLogic (
            float speed,
            Enemy_Animation_Rows anim,
            float timeoutTicks )  [private]
```

Gestionează logica de mișcare pentru repoziționare (mers sau alunecare).

Mută boss-ul către `targetX` cu viteza specificată și gestionează timeout-ul acțiunii.

Parameters

speed	Viteza de mișcare.
anim	Animația care trebuie redată în timpul mișcării.
timeoutTicks	Numărul de tick-uri după care acțiunea de repoziționare expiră.

6.18.3.13 handleStateMachine()

```
void entities.GoblinBoss.handleStateMachine (
            Player player )  [private]
```

Gestionează mașina de stări a Goblin Boss-ului.

Determină acțiunile curente și tranzițiile între stări pe baza detectării jucătorului, cooldown-urilor și altor condiții.

Parameters

player	Jucătorul, pentru a lua decizii bazate pe poziția și starea acestuia.

6.18.3.14 initiateMeleeAttack()

```
void entities.GoblinBoss.initiateMeleeAttack ( )  [private]
```

Inițiază un atac melee standard (lovitură sau picior).

Setează animația corespunzătoare și starea de atac.

6.18.3.15 initiateReposition()

```
void entities.GoblinBoss.initiateReposition (
            boolean preferFastSlide )  [private]
```

Inițiază o acțiune de repoziționare.

Decide dacă repoziționarea va fi o alunecare rapidă sau un mers.

Parameters

preferFastSlide	Indică dacă alunecarea rapidă este preferată.

6.18.3.16 initiateRunSlashAttack()

```
void entities.GoblinBoss.initiateRunSlashAttack ( )  [private]
```

Inițiază un atac de tip "run slash".

Setează animația și starea corespunzătoare.

6.18.3.17 initiateSlideReposition()

```
void entities.GoblinBoss.initiateSlideReposition ( )  [private]
```

Inițiază o repoziționare prin alunecare (slide).

Calculează o poziție țintă și setează starea și animația corespunzătoare.

6.18.3.18 initiateWalkReposition()

```
void entities.GoblinBoss.initiateWalkReposition ( )  [private]
```

Inițiază o repoziționare prin mers.

Calculează o poziție țintă și setează starea și animația corespunzătoare.

6.18.3.19 isAlive()

```
boolean entities.GoblinBoss.isAlive ( )
```

Verifică dacă Goblin Boss-ul este în viață.

Returns

true dacă sănătatea curentă este mai mare decât 0, false altfel.

6.18.3.20 loadAnimations()

```
void entities.GoblinBoss.loadAnimations ( )  [private]
```

Încarcă animațiile pentru Goblin Boss din sprite sheet.

Sprite sheet-ul este încărcat folosind clasa LoadSave. Animațiile sunt stocate în matricile `animations` și `flippedAnimations`.

6.18.3.21 moveTowardsPlayer()

```
void entities.GoblinBoss.moveTowardsPlayer (
            Rectangle2D.Float playerHitbox,
            float speed )  [private]
```

Mută Goblin Boss-ul către jucător.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru a determina direcția.
speed	Viteza cu care se mișcă boss-ul.

6.18.3.22 render()

```
void entities.GoblinBoss.render (
            Graphics g,
            int xLvlOffset )
```

Randează Goblin Boss-ul pe ecran.

Desenează animația curentă la poziția corectă, luând în considerare decalajul nivelului. Poate desena și hitbox-ul dacă `drawHitbox` este true.

Parameters

g	Contextul grafic pentru desenare.
xLvlOffset	Decalajul pe axa X al nivelului, pentru scrolling.

6.18.3.23 setBossAnimation()

```
void entities.GoblinBoss.setBossAnimation (
            Enemy_Animation_Rows animType )  [private]
```

Setează animația curentă a Goblin Boss-ului.

Resetează indexul și tick-ul animației dacă noua animație este diferită sau dacă boss-ul nu execută o acțiune (pentru a reseta animațiile IDLE).

Parameters

animType	Tipul de animație (rândul din sprite sheet) care trebuie setat.

6.18.3.24 setLevelData()

```
void entities.GoblinBoss.setLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Goblin Boss.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

6.18.3.25 takeDamage()

```
void entities.GoblinBoss.takeDamage (
            int damage )
```

Aplică daune Goblin Boss-ului.

Reduce sănătatea curentă și gestionează tranziția la starea HURT sau DYING.

Parameters

damage	Cantitatea de daune primite.

6.18.3.26 update()

```
void entities.GoblinBoss.update (
            Player player,
            int lvlData[][] )
```

Actualizează starea Goblin Boss-ului.

Aceasta include actualizarea animației, cronometrelor de acțiune, detectarea jucătorului, gestionarea mașinii de stări, aplicarea gravitației și actualizarea hitbox-ului.

Parameters

player	Jucătorul, pentru interacțiuni și detectare.
lvlData	Datele nivelului curent, pentru coliziuni.

6.18.3.27 updateCurrentMeleeHitbox()

```
void entities.GoblinBoss.updateCurrentMeleeHitbox ( )  [private]
```

Actualizează hitbox-ul pentru atacurile melee.

Poziția și dimensiunea hitbox-ului de atac depind de direcția boss-ului.

6.18.3.28 updateHitbox()

```
void entities.GoblinBoss.updateHitbox ( )  [private]
```

Actualizează poziția hitbox-ului.

În implementarea curentă, metodele de mișcare modifică direct `hitbox.x` și `hitbox.y`, deci această metodă ar putea fi goală sau folosită pentru ajustări fine dacă este necesar.

6.18.3.29 updatePlayerDetection()

```
void entities.GoblinBoss.updatePlayerDetection (
            Player player )  [private]
```

Actualizează starea de detectare a jucătorului.

Verifică dacă jucătorul se află în raza de vizualizare și setează direcția boss-ului către jucător.

Parameters

player	Jucătorul care trebuie detectat.

6.18.4 Member Data Documentation

6.18.4.1 actionCooldown

```
int entities.GoblinBoss.actionCooldown = 0  [private]
```

Cronometru pentru cooldown-ul dintre acțiuni.

6.18.4.2 actionTimer

```
int entities.GoblinBoss.actionTimer = 0  [private]
```

Cronometru pentru durata acțiunilor curente.

6.18.4.3 animations

```
BufferedImage [][] entities.GoblinBoss.animations  [private]
```

Matrice bidimensională pentru stocarea animațiilor normale ale boss-ului.

6.18.4.4 ATTACK_COOLDOWN_MAX

```
final int entities.GoblinBoss.ATTACK_COOLDOWN_MAX = 90  [private]
```

Cooldown-ul maxim după un atac.

6.18.4.5 attackCheckFrame

```
int entities.GoblinBoss.attackCheckFrame  [private]
```

Frame-ul specific din animația de atac la care se verifică aplicarea daunelor.

6.18.4.6 attackDamageAppliedThisAttack

```
boolean entities.GoblinBoss.attackDamageAppliedThisAttack  [private]
```

Indicator dacă daunele au fost aplicate în timpul atacului curent.

6.18.4.7 attackDamageBoss

```
int entities.GoblinBoss.attackDamageBoss = 10  [private]
```

Daunele provocate de atacul Goblin Boss-ului.

6.18.4.8 BOSS_SCALE_FACTOR

```
final float entities.GoblinBoss.BOSS_SCALE_FACTOR = 2.0f  [static], [private]
```

Factorul de scalare specific pentru dimensiunile Goblin Boss-ului.

6.18.4.9 currentActionState

```
ActionState entities.GoblinBoss.currentActionState = ActionState.IDLE  [private]
```

Starea de acțiune curentă a Goblin Boss-ului.

6.18.4.10 currentHealthBoss

```
int entities.GoblinBoss.currentHealthBoss  [private]
```

Sănătatea curentă a Goblin Boss-ului.

6.18.4.11 currentMeleeHitbox

```
Rectangle2D.Float entities.GoblinBoss.currentMeleeHitbox  [private]
```

Hitbox-ul curent pentru atacurile melee.

6.18.4.12 detectionRange

```
float entities.GoblinBoss.detectionRange = 450f * Game.SCALE  [private]
```

Distanța la care Goblin Boss-ul poate detecta jucătorul.

6.18.4.13 direction

```
int entities.GoblinBoss.direction = Enemy_Animation_Rows.Directions.LEFT  [protected]
```

Direcția curentă a Goblin Boss-ului (stânga sau dreapta).

6.18.4.14 DRAW_HEIGHT

```
final int entities.GoblinBoss.DRAW_HEIGHT = (int) (Constants.EnemyConstants.GOBLIN_DRAW_↩
HEIGHT_DEFAULT * BOSS_SCALE_FACTOR * Game.SCALE)  [private]
```

Înălțimea de desenare a Goblin Boss-ului, ajustată cu factorul de scalare și scala jocului.

6.18.4.15 DRAW_WIDTH

```
final int entities.GoblinBoss.DRAW_WIDTH = (int) (Constants.EnemyConstants.GOBLIN_DRAW_WIDTH↩
_DEFAULT * BOSS_SCALE_FACTOR * Game.SCALE)  [private]
```

Lățimea de desenare a Goblin Boss-ului, ajustată cu factorul de scalare și scala jocului.

6.18.4.16 flippedAnimations

```
BufferedImage [][] entities.GoblinBoss.flippedAnimations  [private]
```

Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale boss-ului.

6.18.4.17 IDLE_DURATION_MAX

```
final int entities.GoblinBoss.IDLE_DURATION_MAX = 120  [private]
```

Durata maximă a stării IDLE.

6.18.4.18 IDLE_DURATION_MIN

```
final int entities.GoblinBoss.IDLE_DURATION_MIN = 60  [private]
```

Durata minimă a stării IDLE.

6.18.4.19 isPerformingAction

```
boolean entities.GoblinBoss.isPerformingAction = false  [private]
```

Indicator dacă Goblin Boss-ul execută o acțiune în prezent.

6.18.4.20 levelData

```
int [][] entities.GoblinBoss.levelData  [private]
```

Datele nivelului curent, folosite pentru coliziuni și navigație.

6.18.4.21 maxHealthBoss

```
int entities.GoblinBoss.maxHealthBoss = 300  [private]
```

Sănătatea maximă a Goblin Boss-ului.

6.18.4.22 meleeAttackRange

```
float entities.GoblinBoss.meleeAttackRange = hitbox.width * 0.7f  [private]
```

Distanța de la care Goblin Boss-ul poate efectua un atac melee.

6.18.4.23 playerDetected

```
boolean entities.GoblinBoss.playerDetected = false  [private]
```

Indicator dacă jucătorul a fost detectat de Goblin Boss.

6.18.4.24 playing

```
Playing entities.GoblinBoss.playing  [private]
```

Referință la starea de joc "Playing".

6.18.4.25 PREPARE_ATTACK_DURATION

```
final int entities.GoblinBoss.PREPARE_ATTACK_DURATION = 45  [private]
```

Durata pregătirii unui atac.

6.18.4.26 REPOSITION_COOLDOWN_MAX

```
final int entities.GoblinBoss.REPOSITION_COOLDOWN_MAX = 75  [private]
```

Cooldown-ul maxim după o repoziționare.

6.18.4.27 runSpeed

```
float entities.GoblinBoss.runSpeed = 1.8f * Game.SCALE  [private]
```

Viteza de alergare a Goblin Boss-ului.

6.18.4.28 sightRange

```
float entities.GoblinBoss.sightRange = 500f * Game.SCALE  [private]
```

Distanța vizuală a Goblin Boss-ului.

6.18.4.29 slideSpeed

```
float entities.GoblinBoss.slideSpeed = 3.0f * Game.SCALE  [private]
```

Viteza de alunecare (slide) a Goblin Boss-ului.

6.18.4.30 targetX

```
float entities.GoblinBoss.targetX  [private]
```

Coordonata X țintă pentru mișcările de repoziționare.

6.18.4.31 walkSpeed

```
float entities.GoblinBoss.walkSpeed = 0.6f * Game.SCALE  [private]
```

Viteza de mers a Goblin Boss-ului.

6.18.4.32 xDrawOffset

```
float entities.GoblinBoss.xDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_X * BOSS_SCALE_FACTOR
[private]
```

Decalajul pe axa X pentru desenarea sprite-ului Goblin Boss-ului.

6.18.4.33 yDrawOffset

```
float entities.GoblinBoss.yDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_Y * BOSS_SCALE_FACTOR
[private]
```

Decalajul pe axa Y pentru desenarea sprite-ului Goblin Boss-ului.

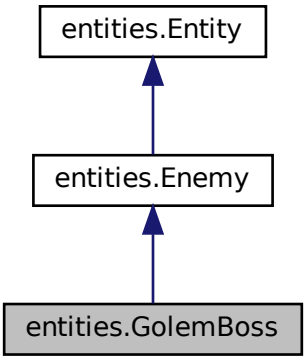
The documentation for this class was generated from the following file:

- src/entities/GoblinBoss.java

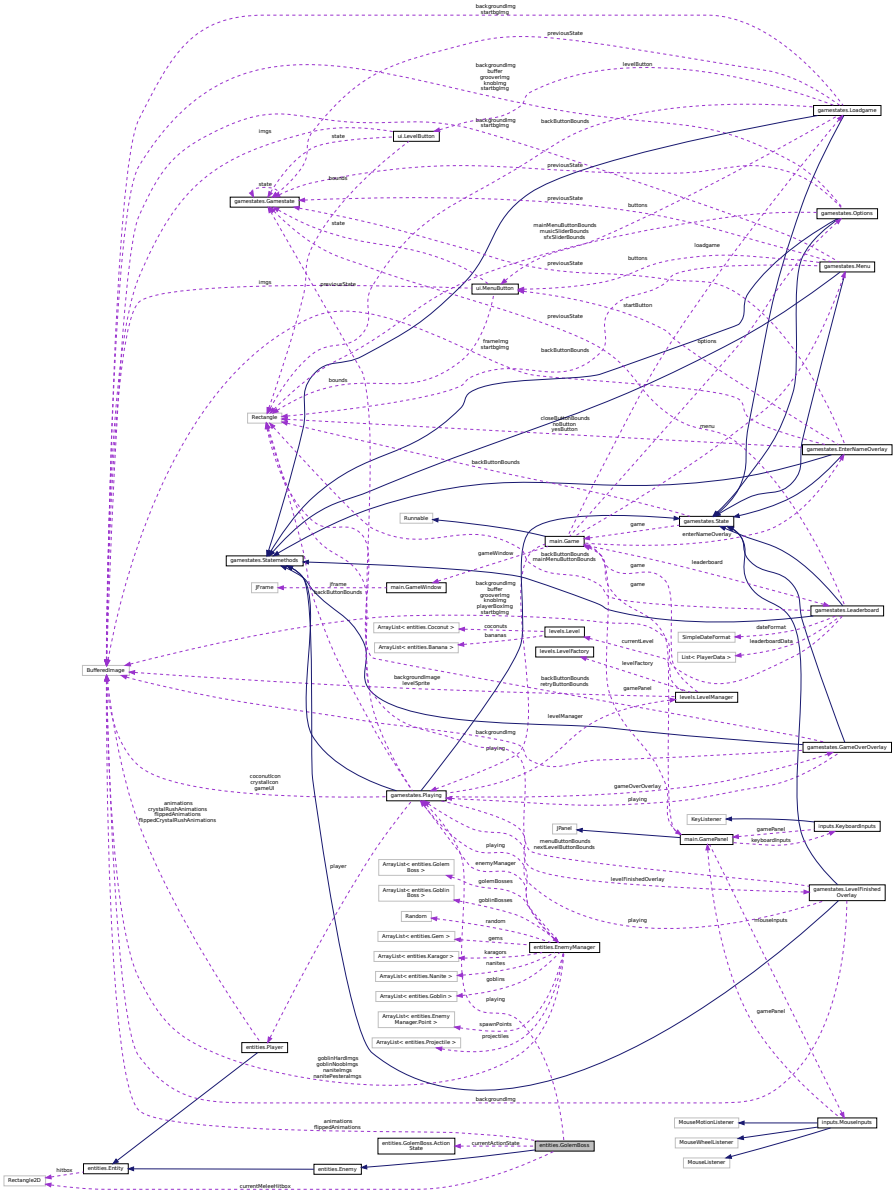
6.19 entities.GolemBoss Class Reference

Reprezintă entitatea Golem Boss în joc.

Inheritance diagram for entities.GolemBoss:



Collaboration diagram for entities.GolemBoss:



Classes

- enum ActionState

  Enumerare pentru stările de acțiune specifice Golem Boss-ului.

Public Member Functions

- GolemBoss (float x, float y, Playing playing)

  Constructor pentru clasa GolemBoss.
- void update (Player player, int[ ][ ] lvlData)

  Actualizează starea Golem Boss-ului.
- void setLevelData (int[ ][ ] levelData)

Setează datele nivelului pentru Golem Boss.

- void render (Graphics g, int xLvlOffset)

  Randează Golem Boss-ul pe ecran.
- boolean isAlive ()

  Verifică dacă Golem Boss-ul este în viață.
- void takeDamage (int damage)

  Aplică daune Golem Boss-ului.
- int getAttackDamage ()

  Returnează daunele de atac ale Golem Boss-ului.
- int getCurrentHealth ()

  Returnează sănătatea curentă a Golem Boss-ului.
- int getMaxHealth ()

  Returnează sănătatea maximă a Golem Boss-ului.

Protected Attributes

- int direction = Enemy_Animation_Rows.Directions.LEFT

  Direcția curentă a Golem Boss-ului (stânga sau dreapta).

Private Member Functions

- void loadAnimations ()

  Încarcă animațiile pentru Golem Boss din sprite sheet-ul specific.
- float getDistance (Rectangle2D.Float r1, Rectangle2D.Float r2)

  Calculează distanța euclidiană între centrele a două dreptunghiuri.
- void updatePlayerDetection (Player player)

  Actualizează starea de detectare a jucătorului.
- void handleStateMachine (Player player)

  Gestionează mașina de stări a Golem Boss-ului.
- void decideNextAction (Player player)

  Decide următoarea acțiune a Golem Boss-ului când jucătorul este detectat.
- void initiateMeleeAttack ()

  Inițiază un atac melee.
- void decideIdleAction ()

  Decide o acțiune aleatorie pentru starea IDLE a Golemului.
- void moveTowardsPlayer (Rectangle2D.Float playerHitbox, float speed)

  Mută Golem Boss-ul către jucător cu viteza specificată.
- void updateCurrentMeleeHitbox ()

  Actualizează hitbox-ul pentru atacurile melee ale Golemului.
- void applyMeleeDamage (Player player)

  Aplică daune jucătorului dacă atacul melee al Golemului îl lovește.
- void setBossAnimation (Enemy_Animation_Rows animType)

  Setează animația curentă a Golem Boss-ului.
- void applyGravity ()

  Aplică gravitația Golem Boss-ului dacă acesta nu se află pe o suprafață solidă.
- void updateHitbox ()

  Actualizează poziția hitbox-ului.

Static Private Member Functions

- static int calculateBossHitboxWidth ()

  Calculează lățimea hitbox-ului pentru Golem Boss.
- static int calculateBossHitboxHeight ()

  Calculează înălțimea hitbox-ului pentru Golem Boss.
- static BufferedImage flipImage (BufferedImage image)

  Inversează (flip) o imagine pe orizontală.

Private Attributes

- BufferedImage[ ][ ] animations

  Matrice bidimensională pentru stocarea animațiilor normale ale boss-ului.
- BufferedImage[ ][ ] flippedAnimations

  Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale boss-ului.
- int maxHealthBoss = 450

  Sănătatea maximă a Golem Boss-ului.
- int currentHealthBoss

  Sănătatea curentă a Golem Boss-ului.
- int attackDamageBoss = 20

  Daunele provocate de atacul Golem Boss-ului.
- final int DRAW_WIDTH = (int) (Constants.EnemyConstants.GOBLIN_BOSS_SPRITE_SOURCE_WIDTH_↩ DEFAULT ∗ BOSS_SCALE_FACTOR ∗ Game.SCALE)

  Lățimea de desenare a Golem Boss-ului, ajustată cu factorul de scalare și scala jocului.
- final int DRAW_HEIGHT = (int) (Constants.EnemyConstants.GOBLIN_BOSS_SPRITE_SOURCE_↩ HEIGHT_DEFAULT ∗ BOSS_SCALE_FACTOR ∗ Game.SCALE)

  Înălțimea de desenare a Golem Boss-ului, ajustată cu factorul de scalare și scala jocului.
- float xDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_X ∗ BOSS_SCALE_FACTOR

  Decalajul pe axa X pentru desenarea sprite-ului Golem Boss-ului.
- float yDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_Y ∗ BOSS_SCALE_FACTOR

  Decalajul pe axa Y pentru desenarea sprite-ului Golem Boss-ului.
- int[ ][ ] levelData

  Datele nivelului curent, folosite pentru coliziuni și navigație.
- Playing playing

  Referință la starea de joc "Playing".
- ActionState currentActionState = ActionState.IDLE

  Starea de acțiune curentă a Golem Boss-ului.
- boolean playerDetected = false

  Indicator dacă jucătorul a fost detectat de Golem Boss.
- float detectionRange = 500f ∗ Game.SCALE

  Distanța la care Golem Boss-ul poate detecta jucătorul.
- float sightRange = 600f ∗ Game.SCALE

  Distanța vizuală a Golem Boss-ului.
- float meleeAttackRange

  Distanța de la care Golem Boss-ul poate efectua un atac melee.
- float walkSpeed = 0.4f ∗ Game.SCALE

  Viteza de mers a Golem Boss-ului.
- float runSpeed = 1.2f ∗ Game.SCALE

  Viteza de alergare a Golem Boss-ului.
- int actionTimer = 0

Cronometru pentru durata acțiunilor curente.

- int actionCooldown = 0

  Cronometru pentru cooldown-ul dintre acțiuni.
- final int IDLE_DURATION_MIN = 80

  Durata minimă a stării IDLE.
- final int IDLE_DURATION_MAX = 160

  Durata maximă a stării IDLE.
- final int PREPARE_ATTACK_DURATION = 60

  Durata pregătirii unui atac (telegraph).
- final int ATTACK_COOLDOWN_MAX = 120

  Cooldown-ul maxim după un atac.
- boolean isPerformingAction = false

  Indicator dacă Golem Boss-ul execută o acțiune în prezent.
- int attackCheckFrame

  Frame-ul specific din animația de atac la care se verifică aplicarea daunelor.
- boolean attackDamageAppliedThisAttack

  Indicator dacă daunele au fost aplicate în timpul atacului curent.
- Rectangle2D.Float currentMeleeHitbox

  Hitbox-ul curent pentru atacurile melee.

Static Private Attributes

- static final float BOSS_SCALE_FACTOR = 3.0f

  Factorul de scalare specific pentru dimensiunile Golem Boss-ului.

Additional Inherited Members

6.19.1 Detailed Description

Reprezintă entitatea Golem Boss în joc.

Această clasă extinde clasa Enemy și definește comportamentul specific și atributele pentru Golem Boss, un inamic puternic cu atacuri devastatoare. Include gestionarea animațiilor, stărilor de acțiune, logicii de atac și interacțiunii cu jucătorul.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 GolemBoss()

```
entities.GolemBoss.GolemBoss (
            float x,
            float y,
            Playing playing )
```

Constructor pentru clasa GolemBoss.

Inițializează Golem Boss-ul cu o poziție specifică și o referință la starea de joc "Playing". Setează sănătatea, încarcă animațiile și inițializează hitbox-ul.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
playing	Referință la obiectul Playing, pentru interacțiuni cu starea jocului.

6.19.3 Member Function Documentation

6.19.3.1 applyGravity()

```
void entities.GolemBoss.applyGravity ( )  [private]
```

Aplică gravitația Golem Boss-ului dacă acesta nu se află pe o suprafață solidă.

Golemul este mai greu și cade mai repede.

6.19.3.2 applyMeleeDamage()

```
void entities.GolemBoss.applyMeleeDamage (
            Player player )  [private]
```

Aplică daune jucătorului dacă atacul melee al Golemului îl lovește.

Verifică intersecția dintre hitbox-ul de atac și hitbox-ul jucătorului. Aplică și un efect de knockback puternic jucătorului.

Parameters

player	Jucătorul care poate fi lovit.

6.19.3.3 calculateBossHitboxHeight()

```
static int entities.GolemBoss.calculateBossHitboxHeight ( )  [static], [private]
```

Calculează înălțimea hitbox-ului pentru Golem Boss.

Se bazează pe dimensiunile sprite-ului sursă și factorii de scalare.

Returns

Înălțimea calculată a hitbox-ului.

6.19.3.4 calculateBossHitboxWidth()

```
static int entities.GolemBoss.calculateBossHitboxWidth ( )  [static], [private]
```

Calculează lățimea hitbox-ului pentru Golem Boss.

Se bazează pe dimensiunile sprite-ului sursă și factorii de scalare.

Returns

Lățimea calculată a hitbox-ului.

6.19.3.5 decideIdleAction()

```
void entities.GolemBoss.decideIdleAction ( )  [private]
```

Decide o acțiune aleatorie pentru starea IDLE a Golemului.

Golemul este mai puțin agitat; poate să se întoarcă sau să rămână pe loc.

6.19.3.6 decideNextAction()

```
void entities.GolemBoss.decideNextAction (
            Player player )  [private]
```

Decide următoarea acțiune a Golem Boss-ului când jucătorul este detectat.

Aceasta poate fi un atac, urmărirea jucătorului (mers sau alergare).

Parameters

player	Jucătorul, pentru a evalua distanța și a lua decizii.

6.19.3.7 flipImage()

```
static BufferedImage entities.GolemBoss.flipImage (
            BufferedImage image )  [static], [private]
```

Inversează (flip) o imagine pe orizontală.

Parameters

image	Imaginea care trebuie inversată.

Returns

Imaginea inversată sau null dacă imaginea de intrare este null.

6.19.3.8 getAttackDamage()

```
int entities.GolemBoss.getAttackDamage ( )
```

Returnează daunele de atac ale Golem Boss-ului.

Returns

Daunele de atac.

6.19.3.9 getCurrentHealth()

```
int entities.GolemBoss.getCurrentHealth ( )
```

Returnează sănătatea curentă a Golem Boss-ului.

Returns

Sănătatea curentă.

6.19.3.10 getDistance()

```
float entities.GolemBoss.getDistance (
            Rectangle2D.Float r1,
            Rectangle2D.Float r2 )  [private]
```

Calculează distanța euclidiană între centrele a două dreptunghiuri.

Parameters

r1	Primul dreptunghi.
r2	Al doilea dreptunghi.

Returns

Distanța dintre centrele celor două dreptunghiuri.

6.19.3.11 getMaxHealth()

```
int entities.GolemBoss.getMaxHealth ( )
```

Returnează sănătatea maximă a Golem Boss-ului.

Returns

Sănătatea maximă.

6.19.3.12 handleStateMachine()

```
void entities.GolemBoss.handleStateMachine (
            Player player )  [private]
```

Gestionează mașina de stări a Golem Boss-ului.

Determină acțiunile curente și tranzițiile între stări pe baza detectării jucătorului, cooldown-urilor și altor condiții specifice Golemului.

Parameters

player	Jucătorul, pentru a lua decizii bazate pe poziția și starea acestuia.

6.19.3.13 initiateMeleeAttack()

```
void entities.GolemBoss.initiateMeleeAttack ( )  [private]
```

Inițiază un atac melee.

Golemul poate efectua un "Heavy Swing" (animația SLASHING) sau un "Ground Stomp" (animația KICKING). Setează animația corespunzătoare, starea de atac și actualizează hitbox-ul de atac.

6.19.3.14 isAlive()

```
boolean entities.GolemBoss.isAlive ( )
```

Verifică dacă Golem Boss-ul este în viață.

Returns

true dacă sănătatea curentă este mai mare decât 0, false altfel.

6.19.3.15 loadAnimations()

```
void entities.GolemBoss.loadAnimations ( )  [private]
```

Încarcă animațiile pentru Golem Boss din sprite sheet-ul specific.

Sprite sheet-ul este încărcat folosind clasa LoadSave. Animațiile sunt stocate în matricile `animations` și `flippedAnimations`.

6.19.3.16 moveTowardsPlayer()

```
void entities.GolemBoss.moveTowardsPlayer (
            Rectangle2D.Float playerHitbox,
            float speed )  [private]
```

Mută Golem Boss-ul către jucător cu viteza specificată.

Include verificarea limitelor nivelului.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru a determina direcția.
speed	Viteza cu care se mișcă boss-ul.

6.19.3.17 render()

```
void entities.GolemBoss.render (
            Graphics g,
            int xLvlOffset )
```

Randează Golem Boss-ul pe ecran.

Desenează animația curentă la poziția corectă, luând în considerare decalajul nivelului. Poate desena și hitbox-ul și hitbox-ul de atac (dacă `drawHitbox` este true).

Parameters

g	Contextul grafic pentru desenare.
xLvlOffset	Decalajul pe axa X al nivelului, pentru scrolling.

6.19.3.18 setBossAnimation()

```
void entities.GolemBoss.setBossAnimation (
            Enemy_Animation_Rows animType )  [private]
```

Setează animația curentă a Golem Boss-ului.

Resetează indexul și tick-ul animației dacă noua animație este diferită.

Parameters

animType	Tipul de animație (rândul din sprite sheet) care trebuie setat.

6.19.3.19 setLevelData()

```
void entities.GolemBoss.setLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Golem Boss.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

6.19.3.20 takeDamage()

```
void entities.GolemBoss.takeDamage (
            int damage )
```

Aplică daune Golem Boss-ului.

Reduce sănătatea curentă și gestionează tranziția la starea HURT sau DYING.

Parameters

damage	Cantitatea de daune primite.

6.19.3.21 update()

```
void entities.GolemBoss.update (
            Player player,
            int lvlData[][] )
```

Actualizează starea Golem Boss-ului.

Aceasta include actualizarea animației, cronometrelor de acțiune, detectarea jucătorului, gestionarea mașinii de stări, aplicarea gravitației și actualizarea hitbox-ului.

Parameters

player	Jucătorul, pentru interacțiuni și detectare.
lvlData	Datele nivelului curent, pentru coliziuni.

6.19.3.22 updateCurrentMeleeHitbox()

```
void entities.GolemBoss.updateCurrentMeleeHitbox ( )  [private]
```

Actualizează hitbox-ul pentru atacurile melee ale Golemului.

Poziția și dimensiunea hitbox-ului de atac depind de tipul de atac (stomp sau swing) și direcția boss-ului.

6.19.3.23 updateHitbox()

```
void entities.GolemBoss.updateHitbox ( )  [private]
```

Actualizează poziția hitbox-ului.

Momentan nu este necesară, deoarece metodele de mișcare actualizează direct `hitbox.x` și `hitbox.y`.

6.19.3.24 updatePlayerDetection()

```
void entities.GolemBoss.updatePlayerDetection (
            Player player )  [private]
```

Actualizează starea de detectare a jucătorului.

Verifică dacă jucătorul se află în raza de vizualizare și setează direcția boss-ului către jucător.

Parameters

player	Jucătorul care trebuie detectat.

6.19.4 Member Data Documentation

6.19.4.1 actionCooldown

```
int entities.GolemBoss.actionCooldown = 0  [private]
```

Cronometru pentru cooldown-ul dintre acțiuni.

6.19.4.2 actionTimer

```
int entities.GolemBoss.actionTimer = 0  [private]
```

Cronometru pentru durata acțiunilor curente.

6.19.4.3 animations

```
BufferedImage [][] entities.GolemBoss.animations  [private]
```

Matrice bidimensională pentru stocarea animațiilor normale ale boss-ului.

6.19.4.4 ATTACK_COOLDOWN_MAX

```
final int entities.GolemBoss.ATTACK_COOLDOWN_MAX = 120  [private]
```

Cooldown-ul maxim după un atac.

6.19.4.5 attackCheckFrame

```
int entities.GolemBoss.attackCheckFrame  [private]
```

Frame-ul specific din animația de atac la care se verifică aplicarea daunelor.

6.19.4.6 attackDamageAppliedThisAttack

```
boolean entities.GolemBoss.attackDamageAppliedThisAttack  [private]
```

Indicator dacă daunele au fost aplicate în timpul atacului curent.

6.19.4.7 attackDamageBoss

```
int entities.GolemBoss.attackDamageBoss = 20  [private]
```

Daunele provocate de atacul Golem Boss-ului.

6.19.4.8 BOSS_SCALE_FACTOR

```
final float entities.GolemBoss.BOSS_SCALE_FACTOR = 3.0f  [static], [private]
```

Factorul de scalare specific pentru dimensiunile Golem Boss-ului.

6.19.4.9 currentActionState

```
ActionState entities.GolemBoss.currentActionState = ActionState.IDLE  [private]
```

Starea de acțiune curentă a Golem Boss-ului.

6.19.4.10 currentHealthBoss

```
int entities.GolemBoss.currentHealthBoss  [private]
```

Sănătatea curentă a Golem Boss-ului.

6.19.4.11 currentMeleeHitbox

```
Rectangle2D.Float entities.GolemBoss.currentMeleeHitbox  [private]
```

Hitbox-ul curent pentru atacurile melee.

6.19.4.12 detectionRange

```
float entities.GolemBoss.detectionRange = 500f * Game.SCALE  [private]
```

Distanța la care Golem Boss-ul poate detecta jucătorul.

6.19.4.13 direction

```
int entities.GolemBoss.direction = Enemy_Animation_Rows.Directions.LEFT  [protected]
```

Direcția curentă a Golem Boss-ului (stânga sau dreapta).

6.19.4.14 DRAW_HEIGHT

```
final int entities.GolemBoss.DRAW_HEIGHT = (int) (Constants.EnemyConstants.GOBLIN_BOSS_↩
SPRITE_SOURCE_HEIGHT_DEFAULT * BOSS_SCALE_FACTOR * Game.SCALE)  [private]
```

Înălțimea de desenare a Golem Boss-ului, ajustată cu factorul de scalare și scala jocului.

6.19.4.15 DRAW_WIDTH

```
final int entities.GolemBoss.DRAW_WIDTH = (int) (Constants.EnemyConstants.GOBLIN_BOSS_SPRITE↩
_SOURCE_WIDTH_DEFAULT * BOSS_SCALE_FACTOR * Game.SCALE)  [private]
```

Lățimea de desenare a Golem Boss-ului, ajustată cu factorul de scalare și scala jocului.

6.19.4.16 flippedAnimations

```
BufferedImage [][] entities.GolemBoss.flippedAnimations  [private]
```

Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale boss-ului.

6.19.4.17 IDLE_DURATION_MAX

```
final int entities.GolemBoss.IDLE_DURATION_MAX = 160  [private]
```

Durata maximă a stării IDLE.

6.19.4.18 IDLE_DURATION_MIN

```
final int entities.GolemBoss.IDLE_DURATION_MIN = 80  [private]
```

Durata minimă a stării IDLE.

6.19.4.19 isPerformingAction

```
boolean entities.GolemBoss.isPerformingAction = false  [private]
```

Indicator dacă Golem Boss-ul execută o acțiune în prezent.

6.19.4.20 levelData

```
int [][] entities.GolemBoss.levelData  [private]
```

Datele nivelului curent, folosite pentru coliziuni și navigație.

6.19.4.21 maxHealthBoss

```
int entities.GolemBoss.maxHealthBoss = 450  [private]
```

Sănătatea maximă a Golem Boss-ului.

6.19.4.22 meleeAttackRange

```
float entities.GolemBoss.meleeAttackRange  [private]
```

Distanța de la care Golem Boss-ul poate efectua un atac melee.

6.19.4.23 playerDetected

```
boolean entities.GolemBoss.playerDetected = false  [private]
```

Indicator dacă jucătorul a fost detectat de Golem Boss.

6.19.4.24 playing

```
Playing entities.GolemBoss.playing  [private]
```

Referință la starea de joc "Playing".

6.19.4.25 PREPARE_ATTACK_DURATION

```
final int entities.GolemBoss.PREPARE_ATTACK_DURATION = 60  [private]
```

Durata pregătirii unui atac (telegraph).

6.19.4.26 runSpeed

```
float entities.GolemBoss.runSpeed = 1.2f * Game.SCALE  [private]
```

Viteza de alergare a Golem Boss-ului.

6.19.4.27 sightRange

```
float entities.GolemBoss.sightRange = 600f * Game.SCALE  [private]
```

Distanța vizuală a Golem Boss-ului.

6.19.4.28 walkSpeed

```
float entities.GolemBoss.walkSpeed = 0.4f * Game.SCALE  [private]
```

Viteza de mers a Golem Boss-ului.

6.19.4.29 xDrawOffset

```
float entities.GolemBoss.xDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_X * BOSS_SCALE_FACTOR
[private]
```

Decalajul pe axa X pentru desenarea sprite-ului Golem Boss-ului.

6.19.4.30 yDrawOffset

```
float entities.GolemBoss.yDrawOffset = Constants.EnemyConstants.GOBLIN_DRAW_OFFSET_Y * BOSS_SCALE_FACTOR
[private]
```

Decalajul pe axa Y pentru desenarea sprite-ului Golem Boss-ului.

The documentation for this class was generated from the following file:

- src/entities/GolemBoss.java

6.20 utilz.Gorilla_Animation_rows Enum Reference

Enumerație ce definește rândurile de animație și numărul de cadre pentru personajul jucător (Gorila).

Classes

- class **Directions**
  Clasă internă statică ce definește constante pentru direcții.

Public Member Functions

- Gorilla_Animation_rows (int rowIndex)

  Constructor pentru constantele enum care nu au un număr specific de cadre definit (de ex., animațiile de liană).
- Gorilla_Animation_rows (int rowIndex, int frameCount)

  Constructor pentru constantele enum.
- int getRowIndex ()

  Returnează indexul rândului din spritesheet pentru această animație.
- int getFrameCount ()

  Returnează numărul de cadre (frame-uri) pentru această animație.

Public Attributes

- PUNCH_CROUCHED =(0,10)

  Animația de pumn în poziție ghemuită.
- COMBO_STANDING =(1,12)

  Animația de combo în picioare.
- CROUCH_TO_STAND =(2,5)

  Animația de tranziție de la ghemuit la în picioare.
- STANDING_BLOCK =(3,5)

  Animația de blocare în picioare.
- DIE_CROUCHED =(4,5)

  Animația de moarte în poziție ghemuită.
- HURT_CROUCHED =(5,5)

  Animația de rănire în poziție ghemuită.
- HURT_STANDING =(6,5)

  Animația de rănire în picioare.
- IDLE_CROUCHED =(7,20)

  Animația de repaus (idle) în poziție ghemuită.
- STANDING_JUMP_SLAM =(8,12)

  Animația de atac "Jump Slam" în picioare.
- JUMP_STANDING =(9,12)

  Animația de săritură în picioare.
- PUNCH_STANDING =(10, 10)

  Animația de pumn în picioare.
- CROUCH_RUN =(11,12)

  Animația de alergare în poziție ghemuită.
- STANDING_RUN =(12, 12)

  Animația de alergare în picioare.
- DIE_STANDING =(13,5)

  Animația de moarte în picioare.
- IDLE_STANDING =(14, 20)

  Animația de repaus (idle) în picioare.
- STAND_TO_CROUCH =(15,5)

  Animația de tranziție de la în picioare la ghemuit.
- CROUCH_THROW =(16,10)

  Animația de aruncare în poziție ghemuită.
- STAND_THROW =(17,10)

  Animația de aruncare în picioare.
- VINE_18 =(18)

Rândul 18 pentru animația de liană.

- VINE_19 =(19)

  Rândul 19 pentru animația de liană.
- VINE_20 =(20)

  Rândul 20 pentru animația de liană.
- VINE_21 =(21)

  Rândul 21 pentru animația de liană.
- VINE_22 =(22)

  Rândul 22 pentru animația de liană.
- VINE_23 =(23)

  Rândul 23 pentru animația de liană.
- CROUCH_WALK =(24,12)

  Animația de mers în poziție ghemuită.
- STAND_WALK =(25, 16)

  Animația de mers în picioare.
- CROUCH_SLAM =(26,10)

  Animația de atac "Slam" în poziție ghemuită.
- STAND_SLAM =(27,10)

  Animația de atac "Slam" în picioare.

Private Attributes

- final int rowIndex

  Indexul rândului din spritesheet corespunzător acestei animații.
- final int frameCount

  Numărul de cadre (frame-uri) pentru această animație.

6.20.1 Detailed Description

Enumerație ce definește rândurile de animație și numărul de cadre pentru personajul jucător (Gorila).

Fiecare constantă corespunde unui rând specific dintr-un spritesheet de animații pentru jucător. Este utilizată pentru a gestiona și accesa secvențele de animație ale jucătorului.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 Gorilla_Animation_rows() [1/2]

```
utilz.Gorilla_Animation_rows.Gorilla_Animation_rows (
            int rowIndex )
```

Constructor pentru constantele enum care nu au un număr specific de cadre definit (de ex., animațiile de liană).

Implicit, numărul de cadre este 0.

Parameters

rowIndex	Indexul rândului animației.

6.20.2.2 Gorilla_Animation_rows() [2/2]

```
utilz.Gorilla_Animation_rows.Gorilla_Animation_rows (
            int rowIndex,
            int frameCount )
```

Constructor pentru constantele enum.

Parameters

rowIndex	Indexul rândului animației în spritesheet.
frameCount	Numărul de cadre pentru animație.

6.20.3 Member Function Documentation

6.20.3.1 getFrameCount()

```
int utilz.Gorilla_Animation_rows.getFrameCount ( )
```

Returnează numărul de cadre (frame-uri) pentru această animație.

Returns

Numărul de cadre.

6.20.3.2 getRowIndex()

```
int utilz.Gorilla_Animation_rows.getRowIndex ( )
```

Returnează indexul rândului din spritesheet pentru această animație.

Returns

Indexul rândului.

6.20.4 Member Data Documentation

6.20.4.1 COMBO_STANDING

```
utilz.Gorilla_Animation_rows.COMBO_STANDING =(1,12)
```

Animația de combo în picioare.

6.20.4.2 CROUCH_RUN

```
utilz.Gorilla_Animation_rows.CROUCH_RUN =(11,12)
```

Animația de alergare în poziție ghemuită.

6.20.4.3 CROUCH_SLAM

```
utilz.Gorilla_Animation_rows.CROUCH_SLAM =(26,10)
```

Animația de atac "Slam" în poziție ghemuită.

6.20.4.4 CROUCH_THROW

```
utilz.Gorilla_Animation_rows.CROUCH_THROW =(16,10)
```

Animația de aruncare în poziție ghemuită.

6.20.4.5 CROUCH_TO_STAND

```
utilz.Gorilla_Animation_rows.CROUCH_TO_STAND =(2,5)
```

Animația de tranziție de la ghemuit la în picioare.

6.20.4.6 CROUCH_WALK

```
utilz.Gorilla_Animation_rows.CROUCH_WALK =(24,12)
```

Animația de mers în poziție ghemuită.

6.20.4.7 DIE_CROUCHED

```
utilz.Gorilla_Animation_rows.DIE_CROUCHED =(4,5)
```

Animația de moarte în poziție ghemuită.

6.20.4.8 DIE_STANDING

```
utilz.Gorilla_Animation_rows.DIE_STANDING =(13,5)
```

Animația de moarte în picioare.

6.20.4.9 frameCount

```
final int utilz.Gorilla_Animation_rows.frameCount  [private]
```

Numărul de cadre (frame-uri) pentru această animație.

6.20.4.10 HURT_CROUCHED

```
utilz.Gorilla_Animation_rows.HURT_CROUCHED =(5,5)
```

Animația de rănire în poziție ghemuită.

6.20.4.11 HURT_STANDING

```
utilz.Gorilla_Animation_rows.HURT_STANDING =(6,5)
```

Animația de rănire în picioare.

6.20.4.12 IDLE_CROUCHED

```
utilz.Gorilla_Animation_rows.IDLE_CROUCHED =(7,20)
```

Animația de repaus (idle) în poziție ghemuită.

6.20.4.13 IDLE_STANDING

```
utilz.Gorilla_Animation_rows.IDLE_STANDING =(14, 20)
```

Animația de repaus (idle) în picioare.

6.20.4.14 JUMP_STANDING

```
utilz.Gorilla_Animation_rows.JUMP_STANDING =(9,12)
```

Animația de săritură în picioare.

6.20.4.15 PUNCH_CROUCHED

```
utilz.Gorilla_Animation_rows.PUNCH_CROUCHED =(0,10)
```

Animația de pumn în poziție ghemuită.

6.20.4.16 PUNCH_STANDING

```
utilz.Gorilla_Animation_rows.PUNCH_STANDING =(10, 10)
```

Animația de pumn în picioare.

6.20.4.17 rowIndex

```
final int utilz.Gorilla_Animation_rows.rowIndex  [private]
```

Indexul rândului din spritesheet corespunzător acestei animații.

6.20.4.18 STAND_SLAM

```
utilz.Gorilla_Animation_rows.STAND_SLAM =(27,10)
```

Animația de atac "Slam" în picioare.

6.20.4.19 STAND_THROW

```
utilz.Gorilla_Animation_rows.STAND_THROW =(17,10)
```

Animația de aruncare în picioare.

6.20.4.20 STAND_TO_CROUCH

```
utilz.Gorilla_Animation_rows.STAND_TO_CROUCH =(15,5)
```

Animația de tranziție de la în picioare la ghemuit.

6.20.4.21 STAND_WALK

```
utilz.Gorilla_Animation_rows.STAND_WALK =(25, 16)
```

Animația de mers în picioare.

6.20.4.22 STANDING_BLOCK

```
utilz.Gorilla_Animation_rows.STANDING_BLOCK =(3,5)
```

Animația de blocare în picioare.

6.20.4.23 STANDING_JUMP_SLAM

```
utilz.Gorilla_Animation_rows.STANDING_JUMP_SLAM =(8,12)
```

Animația de atac "Jump Slam" în picioare.

6.20.4.24 STANDING_RUN

```
utilz.Gorilla_Animation_rows.STANDING_RUN =(12, 12)
```

Animația de alergare în picioare.

6.20.4.25 VINE_18

```
utilz.Gorilla_Animation_rows.VINE_18 =(18)
```

Rândul 18 pentru animația de liană.

6.20.4.26 VINE_19

```
utilz.Gorilla_Animation_rows.VINE_19 =(19)
```

Rândul 19 pentru animația de liană.

6.20.4.27 VINE_20

```
utilz.Gorilla_Animation_rows.VINE_20 =(20)
```

Rândul 20 pentru animația de liană.

6.20.4.28 VINE_21

```
utilz.Gorilla_Animation_rows.VINE_21 =(21)
```

Rândul 21 pentru animația de liană.

6.20.4.29 VINE_22

```
utilz.Gorilla_Animation_rows.VINE_22 =(22)
```

Rândul 22 pentru animația de liană.

6.20.4.30 VINE_23

```
utilz.Gorilla_Animation_rows.VINE_23 =(23)
```

Rândul 23 pentru animația de liană.

The documentation for this enum was generated from the following file:

- src/utilz/Gorilla_Animation_rows.java

6.21 utilz.HelpMethods Class Reference

Clasă utilitară ce conține metode statice ajutătoare, în principal pentru detecția coliziunilor și poziționarea entităților în cadrul nivelului.

Static Public Member Functions

- static boolean canMoveHere (float x, float y, float width, float height, int[ ][ ] lvlData)

  Verifică dacă o entitate se poate deplasa la o anumită poziție (x, y) fără a intra în coliziune cu tile-uri solide.
- static boolean isSolid (int[ ][ ] lvlData, float x, float y)

  Verifică dacă un punct specific (x, y) din lume corespunde unui tile solid.
- static boolean isEntityOnFloor (Rectangle2D.Float hitbox, int[ ][ ] lvlData)

  Verifică dacă o entitate se află pe sol.
- static boolean isEntityOnWall (int[ ][ ] lvlData, int x, int y)

  Verifică dacă o entitate se află lângă un perete (la dreapta sa).
- static boolean isEntityOnCeiling (int[ ][ ] lvlData, int x, int y)

  Verifică dacă o entitate se află sub un tavan.
- static float getEntityXPosNextToWall (Rectangle2D.Float hitbox, float xSpeed)

  Calculează poziția X a unei entități astfel încât să fie exact lângă un perete, fără a-l penetra, în funcție de direcția de mișcare.
- static float getEntityYPosUnderRoofOrAboveFloor (Rectangle2D.Float hitbox, float airSpeed)

  Calculează poziția Y a unei entități astfel încât să fie exact sub un tavan sau deasupra unei podele, fără a le penetra, în funcție de direcția de mișcare verticală.
- static boolean canBossMoveHere (float x, float y, float width, float height, int[ ][ ] lvlData, int direction)

  O versiune mai puțin strictă a metodei canMoveHere(float, float, float, float, int[ ][ ]), potențial pentru entități mai mari precum boșii.

6.21.1 Detailed Description

Clasă utilitară ce conține metode statice ajutătoare, în principal pentru detecția coliziunilor și poziționarea entităților în cadrul nivelului.

6.21.2 Member Function Documentation

6.21.2.1 canBossMoveHere()

```
static boolean utilz.HelpMethods.canBossMoveHere (
            float x,
            float y,
            float width,
            float height,
            int lvlData[][],
            int direction )  [static]
```

O versiune mai puțin strictă a metodei canMoveHere(float, float, float, float, int[ ][ ]), potențial pentru entități mai mari precum boșii.

Verifică mai puține puncte: cele 4 colțuri și centrul marginii frontale (în direcția de mișcare).

Parameters

x	Poziția x dorită a colțului stânga-sus al hitbox-ului.
y	Poziția y dorită a colțului stânga-sus al hitbox-ului.
width	Lățimea hitbox-ului entității.
height	Înălțimea hitbox-ului entității.
lvlData	Matricea 2D cu datele tile-urilor nivelului.
direction	Direcția de mișcare a entității (de ex., 1 pentru dreapta, -1 sau 0 pentru stânga).

Returns

`true`

dacă entitatea se poate deplasa la noua poziție,
`false`

altfel.

6.21.2.2 canMoveHere()

```
static boolean utilz.HelpMethods.canMoveHere (
            float x,
            float y,
            float width,
            float height,
            int lvlData[][] )  [static]
```

Verifică dacă o entitate se poate deplasa la o anumită poziție (x, y) fără a intra în coliziune cu tile-uri solide.

Verifică colțurile și puncte intermediare pe marginile hitbox-ului entității.

Parameters

x	Poziția x dorită a colțului stânga-sus al hitbox-ului.
y	Poziția y dorită a colțului stânga-sus al hitbox-ului.
width	Lățimea hitbox-ului entității.
height	Înălțimea hitbox-ului entității.
lvlData	Matricea 2D cu datele tile-urilor nivelului.

Returns

`true`

dacă entitatea se poate deplasa la noua poziție,
`false`

altfel.

6.21.2.3 getEntityXPosNextToWall()

```
static float utilz.HelpMethods.getEntityXPosNextToWall (
            Rectangle2D.Float hitbox,
            float xSpeed )  [static]
```

Calculează poziția X a unei entități astfel încât să fie exact lângă un perete, fără a-l penetra, în funcție de direcția de mișcare.

Parameters

hitbox	Dreptunghiul de coliziune al entității.
xSpeed	Viteza orizontală a entității (pozitivă pentru dreapta, negativă pentru stânga).

Returns

Noua coordonată X a hitbox-ului.

6.21.2.4 getEntityYPosUnderRoofOrAboveFloor()

```
static float utilz.HelpMethods.getEntityYPosUnderRoofOrAboveFloor (
            Rectangle2D.Float hitbox,
            float airSpeed )  [static]
```

Calculează poziția Y a unei entități astfel încât să fie exact sub un tavan sau deasupra unei podele, fără a le penetra, în funcție de direcția de mișcare verticală.

Parameters

hitbox	Dreptunghiul de coliziune al entității.
airSpeed	Viteza verticală a entității (pozitivă pentru cădere, negativă pentru săritură).

Returns

Noua coordonată Y a hitbox-ului.

6.21.2.5 isEntityOnCeiling()

```
static boolean utilz.HelpMethods.isEntityOnCeiling (
            int lvlData[][],
            int x,
            int y )  [static]
```

Verifică dacă o entitate se află sub un tavan.

Această metodă pare incompletă sau specifică unui anumit context, deoarece verifică doar un singur tile deasupra coordonatelor date.

Parameters

lvlData	Matricea 2D cu datele tile-urilor nivelului.
x	Coordonata x a tile-ului curent al entității (în unități de tile-uri).
y	Coordonata y a tile-ului curent al entității (în unități de tile-uri).

Returns

`true`

dacă tile-ul deasupra nu este gol (ID != 0),
`false`

altfel.

6.21.2.6 isEntityOnFloor()

```
static boolean utilz.HelpMethods.isEntityOnFloor (
            Rectangle2D.Float hitbox,
            int lvlData[][] )  [static]
```

Verifică dacă o entitate se află pe sol.

Verifică punctele de sub colțurile stânga-jos și dreapta-jos ale hitbox-ului.

Parameters

hitbox	Dreptunghiul de coliziune al entității.
lvlData	Matricea 2D cu datele tile-urilor nivelului.

Returns

`true`

dacă entitatea este pe sol,
`false`

altfel.

6.21.2.7 isEntityOnWall()

```
static boolean utilz.HelpMethods.isEntityOnWall (
            int lvlData[][],
            int x,
            int y )  [static]
```

Verifică dacă o entitate se află lângă un perete (la dreapta sa).

Această metodă pare incompletă sau specifică unui anumit context, deoarece verifică doar un singur tile la dreapta coordonatelor date.

Parameters

lvlData	Matricea 2D cu datele tile-urilor nivelului.
x	Coordonata x a tile-ului curent al entității (în unități de tile-uri).
y	Coordonata y a tile-ului curent al entității (în unități de tile-uri).

Returns

`true`

dacă tile-ul din dreapta nu este gol (ID != 0),
`false`

altfel.

6.21.2.8 isSolid()

```
static boolean utilz.HelpMethods.isSolid (
            int lvlData[][],
            float x,
            float y )  [static]
```

Verifică dacă un punct specific (x, y) din lume corespunde unui tile solid.

Un tile este considerat solid dacă valoarea sa în
`lvlData`

este între 0 și 95 (inclusiv). De asemenea, verifică dacă punctul este în afara limitelor nivelului.

Parameters

lvlData	Matricea 2D cu datele tile-urilor nivelului.
x	Coordonata x a punctului de verificat (în pixeli).
y	Coordonata y a punctului de verificat (în pixeli).

Returns

`true`

dacă punctul este solid sau în afara limitelor,
`false`

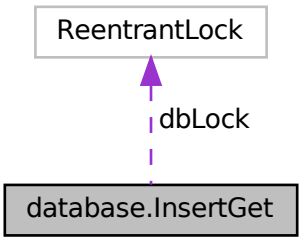
altfel.

The documentation for this class was generated from the following file:

- src/utilz/HelpMethods.java

6.22 database.InsertGet Class Reference

Collaboration diagram for database.InsertGet:



Static Public Member Functions

- static void ensurePlayerTableExists (String dbPath, String numeTabela)
- static void SaveIntoDatabase (String nume_fisier, String nume_tabela, int levelIndex, int Scor, int currentHealth, int coconutNumber, float posX, float posY, int timer)

  Functia se ocupa de salvarea datelor in baza de date.
- static int LoadLevelIndex (String nume_fisier, String nume_tabela)

  Functia se ocupa de incarcarea indexului nivelului din baza de date.
- static int LoadScore (String nume_fisier, String nume_tabela)
- static int LoadCurrentHealth (String nume_fisier, String nume_tabela)
- static int LoadCoconutNumber (String nume_fisier, String nume_tabela)
- static float LoadXPosition (String nume_fisier, String nume_tabela)
- static float LoadYPosition (String nume_fisier, String nume_tabela)
- static int LoadTimer (String nume_fisier, String nume_tabela)
- static List< String > getPlayerList ()
- static String LoadUsername (String nume_fisier)
- static void SaveUsername (String nume_fisier, String username)
- static void createLevelProgressTable (String tableName)
- static boolean checkIfTableExists (String dbPath, String tableName)

Static Private Member Functions

- static Connection getConnection () throws SQLException
- static int loadLevelData (String nume_fisier, String nume_tabela, String column)

Static Private Attributes

- static final ReentrantLock dbLock = new ReentrantLock()
- static final String DB_URL = "jdbc:sqlite:data/gamedatabase.db"

6.22.1 Member Function Documentation

6.22.1.1 checkIfTableExists()

```
static boolean database.InsertGet.checkIfTableExists (
            String dbPath,
            String tableName )  [static]
```

6.22.1.2 createLevelProgressTable()

```
static void database.InsertGet.createLevelProgressTable (
            String tableName )  [static]
```

6.22.1.3 ensurePlayerTableExists()

```
static void database.InsertGet.ensurePlayerTableExists (
            String dbPath,
            String numeTabela )  [static]
```

6.22.1.4 getConnection()

```
static Connection database.InsertGet.getConnection ( ) throws SQLException  [static], [private]
```

6.22.1.5 getPlayerList()

```
static List<String> database.InsertGet.getPlayerList ( )  [static]
```

6.22.1.6 LoadCoconutNumber()

```
static int database.InsertGet.LoadCoconutNumber (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.7 LoadCurrentHealth()

```
static int database.InsertGet.LoadCurrentHealth (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.8 loadLevelData()

```
static int database.InsertGet.loadLevelData (
            String nume_fisier,
            String nume_tabela,
            String column )  [static], [private]
```

6.22.1.9 LoadLevelIndex()

```
database.InsertGet.LoadLevelIndex (
            String nume_fisier,
            String nume_tabela )  [static]
```

Functia se ocupa de incarcarea indexului nivelului din baza de date.

6.22.1.10 LoadScore()

```
static int database.InsertGet.LoadScore (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.11 LoadTimer()

```
static int database.InsertGet.LoadTimer (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.12 LoadUsername()

```
static String database.InsertGet.LoadUsername (
            String nume_fisier )  [static]
```

6.22.1.13 LoadXPosition()

```
static float database.InsertGet.LoadXPosition (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.14 LoadYPosition()

```
static float database.InsertGet.LoadYPosition (
            String nume_fisier,
            String nume_tabela )  [static]
```

6.22.1.15 SaveIntoDatabase()

```
database.InsertGet.SaveIntoDatabase (
            String nume_fisier,
            String nume_tabela,
            int levelIndex,
            int Scor,
            int currentHealth,
            int coconutNumber,
            float posX,
            float posY,
            int timer )  [static]
```

Functia se ocupa de salvarea datelor in baza de date.

6.22.1.16 SaveUsername()

```
static void database.InsertGet.SaveUsername (
            String nume_fisier,
            String username )  [static]
```

6.22.2 Member Data Documentation

6.22.2.1 DB_URL

```
final String database.InsertGet.DB_URL = "jdbc:sqlite:data/gamedatabase.db"  [static], [private]
```

6.22.2.2 dbLock

```
final ReentrantLock database.InsertGet.dbLock = new ReentrantLock()  [static], [private]
```

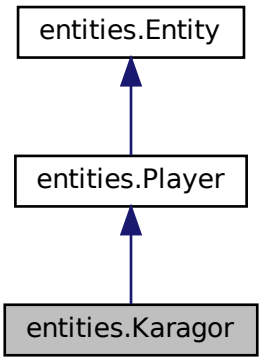
The documentation for this class was generated from the following file:

- src/database/InsertGet.java

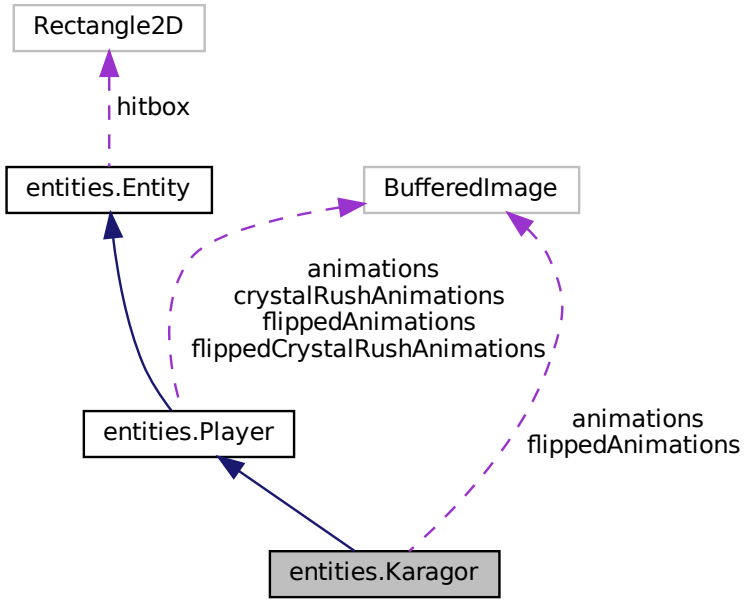
6.23 entities.Karagor Class Reference

Reprezintă entitatea Karagor în joc, care poate funcționa ca un boss.

Inheritance diagram for entities.Karagor:



Collaboration diagram for entities.Karagor:



Public Member Functions

- Karagor (float x, float y, int width, int height, boolean isBoss)

  Constructor pentru clasa Karagor.
- void setPlatformBounds (float left, float right)

  Setează limitele platformei pe care Karagor poate patrula.
- void updatePlayerPosition (Rectangle2D.Float playerHitbox)

  Actualizează poziția lui Karagor în funcție de poziția jucătorului (dacă este boss).
- void update (Rectangle2D.Float playerHitbox)

  Metoda principală de actualizare pentru Karagor.
- void render (Graphics g, int lvlOffsetX)

  Randează Karagor pe ecran.
- void loadAnimations ()

  Încarcă animațiile pentru Karagor din sprite sheet.
- void loadLevelData (int[ ][ ] levelData)

  Setează datele nivelului pentru Karagor.
- boolean isLeft ()
- void setLeft (boolean left)

  Setează starea de mișcare la stânga.
- boolean isRight ()
- void setRight (boolean right)

  Setează starea de mișcare la dreapta.
- void resetDirBooleans ()

  Resetează toate flag-urile de direcție și atac.
- void resetInAir ()

Resetează starea "inAir" și inițiază animația de aterizare.

- void setAttack (boolean attack)

  Setează starea de atac.
- java.awt.geom.Rectangle2D.Float getAttackHitbox ()

  Returnează hitbox-ul de atac al lui Karagor dacă acesta atacă și se află într-un frame valid al animației de atac.
- void setHasHit (boolean hasHit)

  Marchează atacul curent ca fiind efectuat (a lovit ceva), pentru a preveni lovituri multiple cu același atac.
- boolean hasHit ()

  Verifică dacă atacul curent a lovit deja ceva.
- int getAttackDamage ()

  Returnează daunele provocate de atacurile lui Karagor.
- boolean isAttack ()
- boolean isCrouch ()
- void setCrouch (boolean crouch)

  Setează starea de ghemuire a lui Karagor.
- boolean isUp ()
- void setUp (boolean up)

  Setează flag-ul "up".
- boolean isDown ()
- void setDown (boolean down)

  Setează flag-ul "down".
- boolean isJump ()
- void setJump (boolean jump)

  Setează starea de săritură.
- int getCurrentHealth ()

  Returnează sănătatea curentă a lui Karagor.
- int getMaxHealth ()

  Returnează sănătatea maximă a lui Karagor.
- void heal (int amount)

  Crește sănătatea lui Karagor cu o anumită valoare, fără a depăși sănătatea maximă.
- boolean takeDamage (int amount)

  Aplică daune lui Karagor și inițiază starea "hurt" și efectul de knockback.
- boolean isAlive ()

  Verifică dacă Karagor este în viață (are sănătate mai mare decât 0).
- void resetHealth ()

  Resetează sănătatea lui Karagor la valoarea maximă.
- void setLevelData (int[ ][ ] levelData)

  Setează datele nivelului pentru Karagor, folosite pentru coliziuni.
- boolean isAttacking ()

  Verifică dacă Karagor execută un atac.

Public Attributes

- boolean drawHitbox = true

  Indicator dacă hitbox-ul trebuie desenat (pentru debugging).

Protected Member Functions

- void updateGravity ()

  Actualizează gravitația și mișcarea verticală a lui Karagor.

Private Member Functions

- void performAttack ()

  Efectuează un atac aleatoriu din lista de animații de atac valide.
- void updateHurtState ()

  Actualizează starea "hurt" (lovit).
- void updateAnimationTick ()

  Actualizează tick-ul de animație și indexul frame-ului curent.
- void setAnimation ()

  Setează animația curentă a lui Karagor pe baza stării sale.
- void resetAnimationTick ()

  Resetează contorul de tick-uri și indexul frame-ului animației.
- void updatePos ()

  Actualizează poziția orizontală a lui Karagor pe baza input-ului (stânga/dreapta) și a săriturii.
- void updateXPos (float xSpeed)

  Actualizează poziția X a lui Karagor, verificând coliziunile cu pereții.
- void jump ()

  Inițiază o săritură dacă Karagor nu este deja în aer.

Private Attributes

- BufferedImage[ ][ ] animations

  Matrice bidimensională pentru stocarea animațiilor normale ale lui Karagor.
- int animationTick

  Contor pentru tick-urile de animație.
- int animationIndex

  Indexul frame-ului curent al animației.
- int animationSpeed = 3

  Viteza animației (numărul de tick-uri per frame).
- int karagorAction = Gorilla_Animation_rows.PUNCH_STANDING.getRowIndex()

  Acțiunea curentă a lui Karagor (indexul rândului din sprite sheet).
- int karagorDirection = -1

  Direcția curentă a lui Karagor (-1 pentru stânga, 1 pentru dreapta - deși neutilizat direct în logica de mișcare).
- boolean moving = false

  Indicator dacă Karagor se mișcă.
- boolean attack = false

  Indicator dacă Karagor atacă.
- boolean isPunching = false

  Indicator dacă animația de pumn este în desfășurare.
- boolean hasHit = false

  Indicator dacă atacul curent a lovit deja ceva.
- int attackDamage = 5

  Daunele provocate de atacurile lui Karagor.
- boolean left = false

  Indicator dacă Karagor se mișcă la stânga.
- boolean right = false

  Indicator dacă Karagor se mișcă la dreapta.
- boolean up = false

  Indicator dacă Karagor se mișcă în sus (neutilizat pentru mișcare, poate pentru săritură).
- boolean down = false

Indicator dacă Karagor se mișcă în jos (neutilizat pentru mișcare, poate pentru ghemuire).

- boolean facingRight = false

  Indicator dacă Karagor este orientat spre dreapta.
- float playerSpeed = 1.5f

  Viteza de mișcare a lui Karagor.
- boolean crouch = false

  Indicator dacă Karagor este ghemuit.
- boolean wasCrouchPressed = false

  Starea anterioară a butonului de ghemuire.
- boolean isTransitioning = false

  Indicator dacă Karagor este într-o animație de tranziție (ex: stand to crouch).
- int levelData [ ][ ]

  Datele nivelului curent, folosite pentru coliziuni și navigație.
- boolean jump = false

  Indicator dacă Karagor încearcă să sară.
- int maxHealth = 150

  Sănătatea maximă a lui Karagor.
- int currentHealth = 150

  Sănătatea curentă a lui Karagor.
- boolean isHurt = false

  Indicator dacă Karagor este în starea "hurt" (lovit).
- int hurtTimer = 0

  Cronometru pentru durata animației "hurt".
- final int HURT_ANIMATION_DURATION = 30

  Durata animației "hurt" în frame-uri.
- float xDrawOffset = 66 ∗ Game.SCALE

  Decalajul pe axa X pentru desenarea sprite-ului lui Karagor.
- float yDrawOffset = 39 ∗ Game.SCALE

  Decalajul pe axa Y pentru desenarea sprite-ului lui Karagor.
- float airSpeed = 0f

  Viteza verticală a lui Karagor în aer.
- float gravity = 0.12f ∗ Game.SCALE

  Valoarea gravitației aplicate lui Karagor.
- float jumpSpeed = -6.5f ∗ Game.SCALE

  Viteza inițială a săriturii lui Karagor.
- float fallSpeedAfterCollision = 2.4f ∗ Game.SCALE

  Viteza de cădere după o coliziune verticală (cu tavanul).
- boolean inAir = false

  Indicator dacă Karagor se află în aer.
- boolean isLanding = false

  Indicator dacă Karagor este în animația de aterizare.
- int landingFrame = 7

  Frame-ul curent al animației de aterizare.
- BufferedImage[ ][ ] flippedAnimations

  Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale lui Karagor.
- boolean isBoss = true

  Indicator dacă această instanță de Karagor este un boss.
- float platformRightBound

  Limita dreaptă a platformei pe care se află Karagor (pentru patrulare).
- float platformLeftBound

  Limita stângă a platformei pe care se află Karagor (pentru patrulare).

- int attackCooldown = 120

  Cooldown-ul dintre atacuri.
- float detectionRange = 500f

  Distanța la care Karagor poate detecta jucătorul.
- boolean isAttacking = false

  Indicator dacă Karagor execută un atac.
- final int ATTACK_COOLDOWN = 120

  Cooldown-ul maxim dintre atacuri (în tick-uri).

Additional Inherited Members

6.23.1 Detailed Description

Reprezintă entitatea Karagor în joc, care poate funcționa ca un boss.

Această clasă extinde clasa Player, sugerând că poate împărtăși unele mecanici cu jucătorul, dar are propriul set de animații, comportament de atac și atribute. Gestionează stările, mișcarea, atacurile și interacțiunea cu mediul și jucătorul.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Karagor()

```
entities.Karagor.Karagor (
            float x,
            float y,
            int width,
            int height,
            boolean isBoss )
```

Constructor pentru clasa Karagor.

Inițializează Karagor cu o poziție, dimensiuni și specifică dacă este un boss. Setează atributele (sănătate, daune) în funcție de rolul de boss. Încarcă animațiile și inițializează hitbox-ul.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
width	Lățimea entității.
height	Înălțimea entității.
isBoss	true dacă această instanță este un boss, false altfel.

6.23.3 Member Function Documentation

6.23.3.1 getAttackDamage()

```
int entities.Karagor.getAttackDamage ( )
```

Returnează daunele provocate de atacurile lui Karagor.

Returns

Cantitatea de daune.

Reimplemented from entities.Player.

6.23.3.2 getAttackHitbox()

```
java.awt.geom.Rectangle2D.Float entities.Karagor.getAttackHitbox ( )
```

Returnează hitbox-ul de atac al lui Karagor dacă acesta atacă și se află într-un frame valid al animației de atac.

Returns

Un obiect Rectangle2D.Float reprezentând zona de atac, sau null dacă nu atacă sau nu este un frame valid.

Reimplemented from entities.Player.

6.23.3.3 getCurrentHealth()

```
int entities.Karagor.getCurrentHealth ( )
```

Returnează sănătatea curentă a lui Karagor.

Returns

Valoarea sănătății curente.

Reimplemented from entities.Player.

6.23.3.4 getMaxHealth()

```
int entities.Karagor.getMaxHealth ( )
```

Returnează sănătatea maximă a lui Karagor.

Returns

Valoarea sănătății maxime.

Reimplemented from entities.Player.

6.23.3.5 hasHit()

```
boolean entities.Karagor.hasHit ( )
```

Verifică dacă atacul curent a lovit deja ceva.

Returns

true dacă atacul a lovit, false altfel.

Reimplemented from entities.Player.

6.23.3.6 heal()

```
void entities.Karagor.heal (
            int amount )
```

Crește sănătatea lui Karagor cu o anumită valoare, fără a depăși sănătatea maximă.

Parameters

amount	Cantitatea de sănătate adăugată.

Reimplemented from entities.Player.

6.23.3.7 isAlive()

```
boolean entities.Karagor.isAlive ( )
```

Verifică dacă Karagor este în viață (are sănătate mai mare decât 0).

Returns

true dacă Karagor este în viață, false altfel.

Reimplemented from entities.Player.

6.23.3.8 isAttack()

```
boolean entities.Karagor.isAttack ( )
```

Returns

true dacă Karagor este în starea de atac (flag general), false altfel.

Reimplemented from entities.Player.

6.23.3.9 isAttacking()

```
boolean entities.Karagor.isAttacking ( )
```

Verifică dacă Karagor execută un atac.

Returns

true dacă Karagor atacă, false altfel.

6.23.3.10 isCrouch()

```
boolean entities.Karagor.isCrouch ( )
```

Returns

true dacă Karagor este ghemuit, false altfel.

Reimplemented from entities.Player.

6.23.3.11 isDown()

```
boolean entities.Karagor.isDown ( )
```

Returns

true dacă flag-ul "down" este activat, false altfel (neutilizat pentru mișcare).

Reimplemented from entities.Player.

6.23.3.12 isJump()

```
boolean entities.Karagor.isJump ( )
```

Returns

true dacă Karagor încearcă să sară, false altfel.

Reimplemented from entities.Player.

6.23.3.13 isLeft()

```
boolean entities.Karagor.isLeft ( )
```

Returns

true dacă Karagor se mișcă la stânga, false altfel.

Reimplemented from entities.Player.

6.23.3.14 isRight()

```
boolean entities.Karagor.isRight ( )
```

Returns

true dacă Karagor se mișcă la dreapta, false altfel.

Reimplemented from entities.Player.

6.23.3.15 isUp()

```
boolean entities.Karagor.isUp ( )
```

Returns

true dacă flag-ul "up" este activat, false altfel (neutilizat pentru mișcare).

Reimplemented from entities.Player.

6.23.3.16 jump()

```
void entities.Karagor.jump ( )  [private]
```

Inițiază o săritură dacă Karagor nu este deja în aer.

Setează starea "inAir" și viteza verticală inițială.

Reimplemented from entities.Player.

6.23.3.17 loadAnimations()

```
void entities.Karagor.loadAnimations ( )
```

Încarcă animațiile pentru Karagor din sprite sheet.

Sprite sheet-ul este încărcat folosind clasa LoadSave. Animațiile sunt stocate în matricile `animations` și `flippedAnimations`.

Reimplemented from entities.Player.

6.23.3.18 loadLevelData()

```
void entities.Karagor.loadLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Karagor.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

Reimplemented from entities.Player.

6.23.3.19 performAttack()

```
void entities.Karagor.performAttack ( )  [private]
```

Efectuează un atac aleatoriu din lista de animații de atac valide.

Setează acțiunea Karagorului la animația de atac aleasă și resetează cooldown-ul.

6.23.3.20 render()

```
void entities.Karagor.render (
            Graphics g,
            int lvlOffsetX )
```

Randează Karagor pe ecran.

Desenează frame-ul curent al animației la poziția corectă, luând în considerare decalajul nivelului.

Parameters

g	Contextul grafic pentru desenare.
lvlOffsetX	Decalajul pe axa X al nivelului, pentru scrolling.

Reimplemented from entities.Player.

6.23.3.21 resetAnimationTick()

```
void entities.Karagor.resetAnimationTick ( )  [private]
```

Resetează contorul de tick-uri și indexul frame-ului animației.

Folosit la schimbarea tipului de animație.

Reimplemented from entities.Player.

6.23.3.22 resetDirBooleans()

```
void entities.Karagor.resetDirBooleans ( )
```

Resetează toate flag-urile de direcție și atac.

Reimplemented from entities.Player.

6.23.3.23 resetHealth()

```
void entities.Karagor.resetHealth ( )
```

Resetează sănătatea lui Karagor la valoarea maximă.

Reimplemented from entities.Player.

6.23.3.24 resetInAir()

```
void entities.Karagor.resetInAir ( )
```

Resetează starea "inAir" și inițiază animația de aterizare.

Reimplemented from entities.Player.

6.23.3.25 setAnimation()

```
void entities.Karagor.setAnimation ( )  [private]
```

Setează animația curentă a lui Karagor pe baza stării sale.

Gestionează prioritățile animațiilor (ex: "hurt" are prioritate maximă).

Reimplemented from entities.Player.

6.23.3.26 setAttack()

```
void entities.Karagor.setAttack (
            boolean attack )
```

Setează starea de atac.

Dacă se activează atacul și Karagor nu atacă deja, inițiază animația de pumn și resetează starea de lovitură.

Parameters

attack	true pentru a iniția un atac.

Reimplemented from entities.Player.

6.23.3.27 setCrouch()

```
void entities.Karagor.setCrouch (
            boolean crouch )
```

Setează starea de ghemuire a lui Karagor.

Dacă se activează ghemuirea și Karagor nu era ghemuit și nu este în aer, inițiază animația de tranziție corespunzătoare.

Parameters

crouch	true pentru a activa ghemuirea.

Reimplemented from entities.Player.

6.23.3.28 setDown()

```
void entities.Karagor.setDown (
            boolean down )
```

Setează flag-ul "down".

Parameters

down	true pentru a activa flag-ul.

Reimplemented from entities.Player.

6.23.3.29 setHasHit()

```
void entities.Karagor.setHasHit (
            boolean hasHit )
```

Marchează atacul curent ca fiind efectuat (a lovit ceva), pentru a preveni lovituri multiple cu același atac.

Parameters

hasHit	true dacă atacul a lovit, false altfel.

Reimplemented from entities.Player.

6.23.3.30 setJump()

```
void entities.Karagor.setJump (
            boolean jump )
```

Setează starea de săritură.

Parameters

jump	true pentru a iniția o săritură.

Reimplemented from entities.Player.

6.23.3.31 setLeft()

```
void entities.Karagor.setLeft (
            boolean left )
```

Setează starea de mișcare la stânga.

Parameters

left	true pentru a activa mișcarea la stânga.

Reimplemented from entities.Player.

6.23.3.32 setLevelData()

```
void entities.Karagor.setLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Karagor, folosite pentru coliziuni.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

6.23.3.33 setPlatformBounds()

```
void entities.Karagor.setPlatformBounds (
            float left,
            float right )
```

Setează limitele platformei pe care Karagor poate patrula.

Parameters

left	Limita stângă a platformei.
right	Limita dreaptă a platformei.

6.23.3.34 setRight()

```
void entities.Karagor.setRight (
            boolean right )
```

Setează starea de mișcare la dreapta.

Parameters

right	true pentru a activa mișcarea la dreapta.

Reimplemented from entities.Player.

6.23.3.35 setUp()

```
void entities.Karagor.setUp (
            boolean up )
```

Setează flag-ul "up".

Parameters

up	true pentru a activa flag-ul.

Reimplemented from entities.Player.

6.23.3.36 takeDamage()

```
boolean entities.Karagor.takeDamage (
            int amount )
```

Aplică daune lui Karagor și inițiază starea "hurt" și efectul de knockback.

Parameters

amount	Cantitatea de daune primite.

Returns

true dacă Karagor este încă în viață după primirea daunelor, false altfel.

Reimplemented from entities.Player.

6.23.3.37 update()

```
void entities.Karagor.update (
            Rectangle2D.Float playerHitbox )
```

Metoda principală de actualizare pentru Karagor.

Actualizează poziția în funcție de jucător, gravitația, knockback-ul, animațiile și starea de "hurt".

Parameters

playerHitbox	Hitbox-ul jucătorului.

6.23.3.38 updateAnimationTick()

```
void entities.Karagor.updateAnimationTick ( )  [private]
```

Actualizează tick-ul de animație și indexul frame-ului curent.

Gestionează diferite tipuri de animații (în aer, aterizare, tranziție, pumn, normală).

Reimplemented from entities.Player.

6.23.3.39 updateGravity()

```
void entities.Karagor.updateGravity ( )  [protected]
```

Actualizează gravitația și mișcarea verticală a lui Karagor.

Gestionează starea "inAir" și coliziunile cu podeaua sau tavanul.

Reimplemented from entities.Player.

6.23.3.40 updateHurtState()

```
void entities.Karagor.updateHurtState ( )  [private]
```

Actualizează starea "hurt" (lovit).

Dacă Karagor este lovit, incrementează un cronometru; după o durată specifică, iese din starea "hurt".

6.23.3.41 updatePlayerPosition()

```
void entities.Karagor.updatePlayerPosition (
            Rectangle2D.Float playerHitbox )
```

Actualizează poziția lui Karagor în funcție de poziția jucătorului (dacă este boss).

Karagor se va întoarce către jucător și se va deplasa spre el dacă nu este în raza de atac. Dacă este în raza de atac și cooldown-ul permite, va efectua un atac.

Parameters

playerHitbox	Hitbox-ul jucătorului.

6.23.3.42 updatePos()

```
void entities.Karagor.updatePos ( )  [private]
```

Actualizează poziția orizontală a lui Karagor pe baza input-ului (stânga/dreapta) și a săriturii.

Gestionează coliziunile cu pereții.

Reimplemented from entities.Player.

6.23.3.43 updateXPos()

```
void entities.Karagor.updateXPos (
            float xSpeed )  [private]
```

Actualizează poziția X a lui Karagor, verificând coliziunile cu pereții.

Parameters

xSpeed	Viteza orizontală care trebuie aplicată.

Reimplemented from entities.Player.

6.23.4 Member Data Documentation

6.23.4.1 airSpeed

```
float entities.Karagor.airSpeed = 0f  [private]
```

Viteza verticală a lui Karagor în aer.

6.23.4.2 animationIndex

```
int entities.Karagor.animationIndex  [private]
```

Indexul frame-ului curent al animației.

6.23.4.3 animations

```
BufferedImage [][] entities.Karagor.animations  [private]
```

Matrice bidimensională pentru stocarea animațiilor normale ale lui Karagor.

6.23.4.4 animationSpeed

```
int entities.Karagor.animationSpeed = 3  [private]
```

Viteza animației (numărul de tick-uri per frame).

6.23.4.5 animationTick

```
int entities.Karagor.animationTick  [private]
```

Contor pentru tick-urile de animație.

6.23.4.6 attack

```
boolean entities.Karagor.attack = false  [private]
```

Indicator dacă Karagor atacă.

6.23.4.7 ATTACK_COOLDOWN

```
final int entities.Karagor.ATTACK_COOLDOWN = 120  [private]
```

Cooldown-ul maxim dintre atacuri (în tick-uri).

6.23.4.8 attackCooldown

```
int entities.Karagor.attackCooldown = 120  [private]
```

Cooldown-ul dintre atacuri.

6.23.4.9 attackDamage

```
int entities.Karagor.attackDamage = 5  [private]
```

Daunele provocate de atacurile lui Karagor.

6.23.4.10 crouch

```
boolean entities.Karagor.crouch = false  [private]
```

Indicator dacă Karagor este ghemuit.

6.23.4.11 currentHealth

```
int entities.Karagor.currentHealth = 150  [private]
```

Sănătatea curentă a lui Karagor.

6.23.4.12 detectionRange

```
float entities.Karagor.detectionRange = 500f  [private]
```

Distanța la care Karagor poate detecta jucătorul.

6.23.4.13 down

```
boolean entities.Karagor.down = false  [private]
```

Indicator dacă Karagor se mișcă în jos (neutilizat pentru mișcare, poate pentru ghemuire).

6.23.4.14 drawHitbox

```
boolean entities.Karagor.drawHitbox = true
```

Indicator dacă hitbox-ul trebuie desenat (pentru debugging).

6.23.4.15 facingRight

```
boolean entities.Karagor.facingRight = false  [private]
```

Indicator dacă Karagor este orientat spre dreapta.

6.23.4.16 fallSpeedAfterCollision

```
float entities.Karagor.fallSpeedAfterCollision = 2.4f * Game.SCALE  [private]
```

Viteza de cădere după o coliziune verticală (cu tavanul).

6.23.4.17 flippedAnimations

```
BufferedImage [][] entities.Karagor.flippedAnimations  [private]
```

Matrice bidimensională pentru stocarea animațiilor inversate (flipped) ale lui Karagor.

6.23.4.18 gravity

```
float entities.Karagor.gravity = 0.12f * Game.SCALE  [private]
```

Valoarea gravitației aplicate lui Karagor.

6.23.4.19 hasHit

```
boolean entities.Karagor.hasHit = false  [private]
```

Indicator dacă atacul curent a lovit deja ceva.

6.23.4.20 HURT_ANIMATION_DURATION

```
final int entities.Karagor.HURT_ANIMATION_DURATION = 30  [private]
```

Durata animației "hurt" în frame-uri.

6.23.4.21 hurtTimer

```
int entities.Karagor.hurtTimer = 0  [private]
```

Cronometru pentru durata animației "hurt".

6.23.4.22 inAir

```
boolean entities.Karagor.inAir = false  [private]
```

Indicator dacă Karagor se află în aer.

6.23.4.23 isAttacking

```
boolean entities.Karagor.isAttacking = false  [private]
```

Indicator dacă Karagor execută un atac.

6.23.4.24 isBoss

```
boolean entities.Karagor.isBoss = true  [private]
```

Indicator dacă această instanță de Karagor este un boss.

6.23.4.25 isHurt

```
boolean entities.Karagor.isHurt = false  [private]
```

Indicator dacă Karagor este în starea "hurt" (lovit).

6.23.4.26 isLanding

```
boolean entities.Karagor.isLanding = false  [private]
```

Indicator dacă Karagor este în animația de aterizare.

6.23.4.27 isPunching

```
boolean entities.Karagor.isPunching = false  [private]
```

Indicator dacă animația de pumn este în desfășurare.

6.23.4.28 isTransitioning

```
boolean entities.Karagor.isTransitioning = false  [private]
```

Indicator dacă Karagor este într-o animație de tranziție (ex: stand to crouch).

6.23.4.29 jump

```
boolean entities.Karagor.jump = false  [private]
```

Indicator dacă Karagor încearcă să sară.

6.23.4.30 jumpSpeed

```
float entities.Karagor.jumpSpeed = -6.5f * Game.SCALE  [private]
```

Viteza inițială a săriturii lui Karagor.

6.23.4.31 karagorAction

```
int entities.Karagor.karagorAction = Gorilla_Animation_rows.PUNCH_STANDING.getRowIndex()  [private]
```

Acțiunea curentă a lui Karagor (indexul rândului din sprite sheet).

6.23.4.32 karagorDirection

```
int entities.Karagor.karagorDirection = -1  [private]
```

Direcția curentă a lui Karagor (-1 pentru stânga, 1 pentru dreapta - deși neutilizat direct în logica de mișcare).

6.23.4.33 landingFrame

```
int entities.Karagor.landingFrame = 7  [private]
```

Frame-ul curent al animației de aterizare.

6.23.4.34 left

```
boolean entities.Karagor.left = false  [private]
```

Indicator dacă Karagor se mișcă la stânga.

6.23.4.35 levelData

```
int entities.Karagor.levelData[][]  [private]
```

Datele nivelului curent, folosite pentru coliziuni și navigație.

6.23.4.36 maxHealth

```
int entities.Karagor.maxHealth = 150  [private]
```

Sănătatea maximă a lui Karagor.

6.23.4.37 moving

```
boolean entities.Karagor.moving = false  [private]
```

Indicator dacă Karagor se mișcă.

6.23.4.38 platformLeftBound

```
float entities.Karagor.platformLeftBound  [private]
```

Limita stângă a platformei pe care se află Karagor (pentru patrulare).

6.23.4.39 platformRightBound

```
float entities.Karagor.platformRightBound  [private]
```

Limita dreaptă a platformei pe care se află Karagor (pentru patrulare).

6.23.4.40 playerSpeed

```
float entities.Karagor.playerSpeed = 1.5f  [private]
```

Viteza de mișcare a lui Karagor.

6.23.4.41 right

```
boolean entities.Karagor.right = false  [private]
```

Indicator dacă Karagor se mișcă la dreapta.

6.23.4.42 up

```
boolean entities.Karagor.up = false  [private]
```

Indicator dacă Karagor se mișcă în sus (neutilizat pentru mișcare, poate pentru săritură).

6.23.4.43 wasCrouchPressed

```
boolean entities.Karagor.wasCrouchPressed = false  [private]
```

Starea anterioară a butonului de ghemuire.

6.23.4.44 xDrawOffset

```
float entities.Karagor.xDrawOffset = 66 * Game.SCALE  [private]
```

Decalajul pe axa X pentru desenarea sprite-ului lui Karagor.

6.23.4.45 yDrawOffset

```
float entities.Karagor.yDrawOffset = 39 * Game.SCALE  [private]
```

Decalajul pe axa Y pentru desenarea sprite-ului lui Karagor.

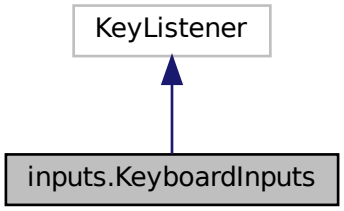
The documentation for this class was generated from the following file:

- src/entities/Karagor.java

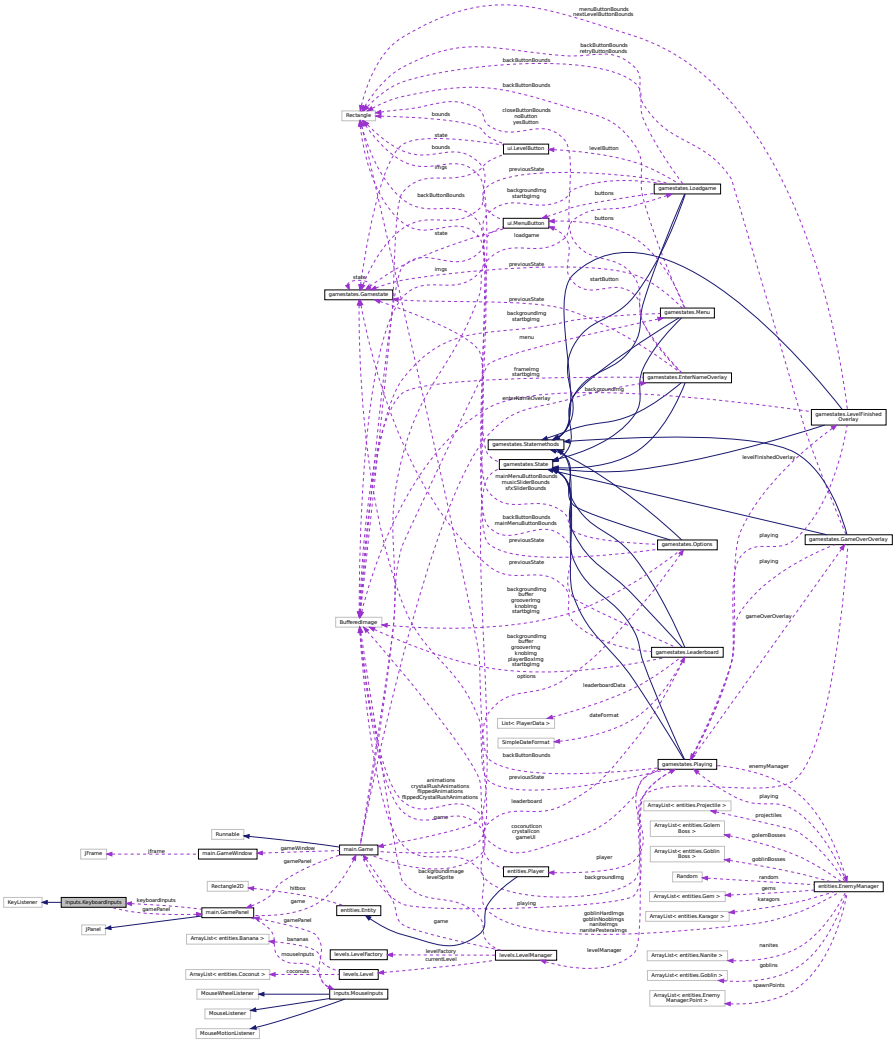
6.24 inputs.KeyboardInputs Class Reference

Gestionează input-ul de la tastatură pentru joc.

Inheritance diagram for inputs.KeyboardInputs:



Collaboration diagram for inputs.KeyboardInputs:



Public Member Functions

- KeyboardInputs (GamePanel gamePanel)

  Constructor pentru KeyboardInputs.
- void keyTyped (KeyEvent e)

  Metodă apelată când o tastă este "typed" (apăsată și eliberată, generând un caracter).
- void keyPressed (KeyEvent e)

  Metodă apelată la apăsarea unei taste.
- void keyReleased (KeyEvent e)

  Metodă apelată la eliberarea unei taste.

Private Attributes

- GamePanel gamePanel

6.24.1 Detailed Description

Gestionează input-ul de la tastatură pentru joc.

Implementează interfața KeyListener și redirecționează evenimentele de tastatură către starea de joc activă corespunzătoare.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 KeyboardInputs()

```
inputs.KeyboardInputs.KeyboardInputs (
            GamePanel gamePanel )
```

Constructor pentru KeyboardInputs.

Parameters

gamePanel	Panoul principal al jocului GamePanel căruia i se atașează acest listener.

6.24.3 Member Function Documentation

6.24.3.1 keyPressed()

```
void inputs.KeyboardInputs.keyPressed (
            KeyEvent e )
```

Metodă apelată la apăsarea unei taste.

Redirecționează evenimentul către metoda
keyPressed

a stării de joc active.

Parameters

e	Evenimentul KeyEvent.

6.24.3.2 keyReleased()

```
void inputs.KeyboardInputs.keyReleased (
            KeyEvent e )
```

Metodă apelată la eliberarea unei taste.

Redirecționează evenimentul către metoda
keyReleased

a stării de joc active.

Parameters

e	Evenimentul KeyEvent.

6.24.3.3 keyTyped()

```
void inputs.KeyboardInputs.keyTyped (
            KeyEvent e )
```

Metodă apelată când o tastă este "typed" (apăsată și eliberată, generând un caracter).

Momentan neutilizată.

Parameters

e	Evenimentul KeyEvent.

6.24.4 Member Data Documentation

6.24.4.1 gamePanel

```
GamePanel inputs.KeyboardInputs.gamePanel  [private]
```

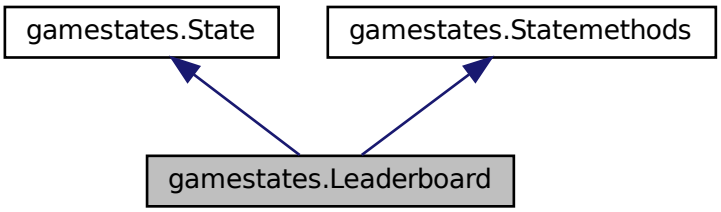
The documentation for this class was generated from the following file:

- src/inputs/KeyboardInputs.java

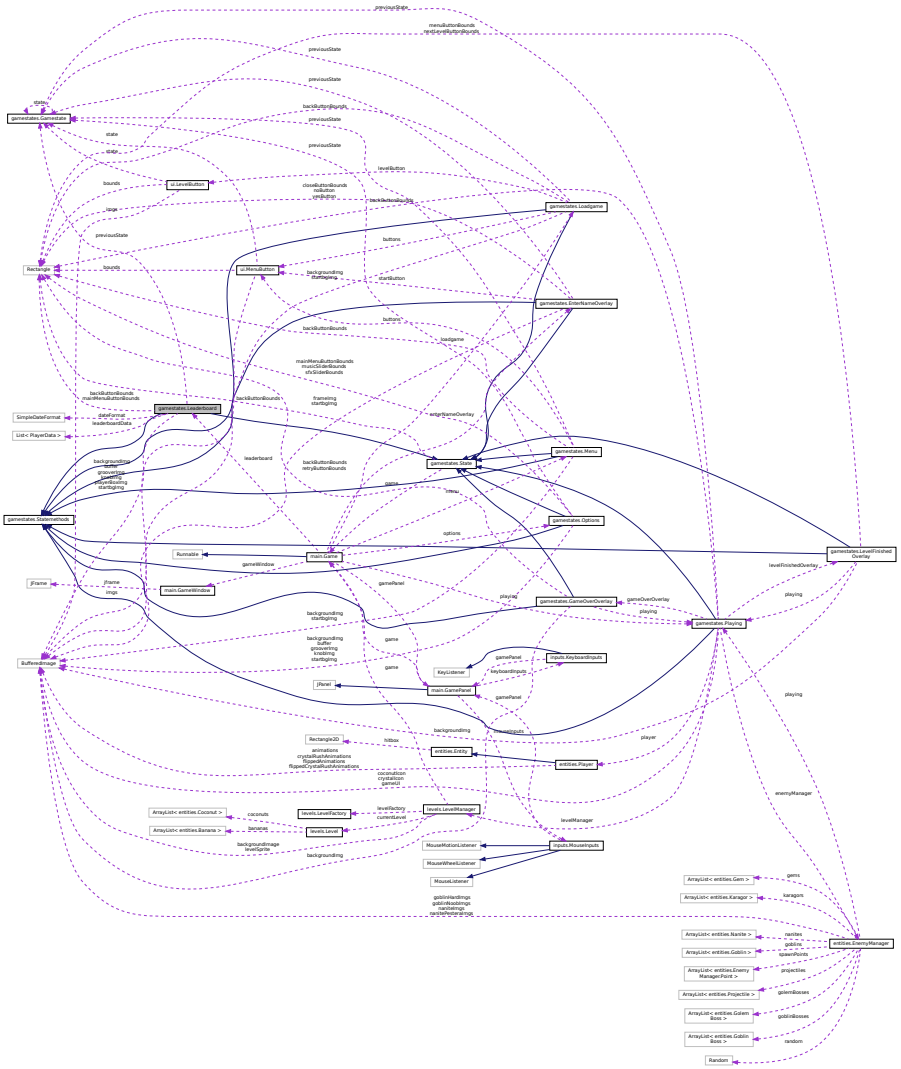
6.25 gamestates.Leaderboard Class Reference

Reprezintă starea de joc pentru afișarea clasamentului (Leaderboard).

Inheritance diagram for gamestates.Leaderboard:



Collaboration diagram for gamestates.Leaderboard:



Classes

- class **PlayerData**

  Clasă internă statică pentru a stoca datele unui jucător afișate în clasament.

Public Member Functions

- Leaderboard (Game game)

  Constructor pentru starea Leaderboard.
- void update ()

  Actualizează starea clasamentului.
- void draw (Graphics g)

  Desenează clasamentul pe ecran.
- void mouseWheelMoved (MouseWheelEvent e)

  Gestionează evenimentele de rotiță a mouse-ului pentru derularea listei clasamentului.
- void mouseClicked (MouseEvent e)

Gestionarea evenimentului de click al mouse-ului (neutilizat).

- void mousePressed (MouseEvent e)

  Gestionază evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).
- void mouseMoved (MouseEvent e)

  Gestionarea evenimentului de mișcare a mouse-ului (neutilizat, butoanele nu au efect de hover aici).
- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionarea evenimentului de apăsare a tastei (neutilizat).
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).
- void setPreviousState (Gamestate state)

  Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" (X).

Private Member Functions

- void refreshLeaderboardData ()

  Reîmprospătează datele clasamentului prin încărcarea informațiilor despre toți jucătorii din baza de date și sortarea lor după scor.
- void loadCustomFont ()

  Încarcă fontul personalizat folosit pentru textul din clasament.
- void loadStartImg ()

  Încarcă și configurează imaginea de fundal generală.
- void loadBackground ()

  Încarcă și configurează imaginea specifică a cadrului clasamentului.

Private Attributes

- BufferedImage backgroundImg
- int menuX
- BufferedImage startbgImg
- int startbgX
- Rectangle mainMenuButtonBounds
- BufferedImage playerBoxImg
- Font airstrikeFont
- BufferedImage knobImg
- BufferedImage grooverImg
- int scrollOffset = 0

  Offset-ul curent de derulare verticală a listei.
- int boxSpacing = 110

  Spațiul vertical (în pixeli) între fiecare intrare din clasament.
- int visibleAreaHeight = boxSpacing ∗ 5

Înălțimea zonei vizibile a listei derulabile.

- BufferedImage buffer

  Un buffer grafic pentru a desena conținutul înainte de a-l afișa pe ecran (pentru performanță/evitarea pâlpâirii).
- boolean needsRedraw = true

  Flag care indică dacă este necesară redesenarea conținutului buffer-ului.
- Rectangle backButtonBounds
- Gamestate previousState = null
- List< PlayerData > leaderboardData = new ArrayList<>()
- SimpleDateFormat dateFormat

Static Private Attributes

- static final String DB_FILE = "data/gamedatabase.db"

  Calea către fișierul bazei de date.

Additional Inherited Members

6.25.1 Detailed Description

Reprezintă starea de joc pentru afișarea clasamentului (Leaderboard).

Încarcă datele jucătorilor din baza de date, le sortează după scor și le afișează într-o listă derulabilă. Extinde State și implementează Statemethods.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 Leaderboard()

```
gamestates.Leaderboard.Leaderboard (
            Game game )
```

Constructor pentru starea Leaderboard.

Inițializează elementele grafice, încarcă fonturile și datele din clasament.

Parameters

game	Referință la instanța principală a jocului Game.

6.25.3 Member Function Documentation

6.25.3.1 draw()

```
void gamestates.Leaderboard.draw (
            Graphics g )
```

Desenează clasamentul pe ecran.

Utilizează un buffer pentru a desena elementele doar atunci când este necesar (needsRedraw). Afișează intrările jucătorilor, clasamentul, scorul și alte detalii, precum și o bară de derulare.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.25.3.2 drawBackButton()

```
void gamestates.Leaderboard.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" (X).

Suprascrie metoda din State pentru a folosi backButtonBounds specific acestei clase.

Parameters

g2d	Contextul grafic 2D.

Reimplemented from gamestates.State.

6.25.3.3 keyPressed()

```
void gamestates.Leaderboard.keyPressed (
            KeyEvent e )
```

Gestionarea evenimentului de apăsare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.25.3.4 keyReleased()

```
void gamestates.Leaderboard.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.25.3.5 loadBackground()

```
void gamestates.Leaderboard.loadBackground ( )  [private]
```

Încarcă și configurează imaginea specifică a cadrului clasamentului.

6.25.3.6 loadCustomFont()

```
void gamestates.Leaderboard.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat folosit pentru textul din clasament.

6.25.3.7 loadStartImg()

```
void gamestates.Leaderboard.loadStartImg ( )  [private]
```

Încarcă și configurează imaginea de fundal generală.

6.25.3.8 mouseClicked()

```
void gamestates.Leaderboard.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.25.3.9 mouseDragged()

```
void gamestates.Leaderboard.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.25.3.10 mouseMoved()

```
void gamestates.Leaderboard.mouseMoved (
            MouseEvent e )
```

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat, butoanele nu au efect de hover aici).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.25.3.11 mousePressed()

```
void gamestates.Leaderboard.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Verifică dacă s-a apăsat pe butonul de întoarcere la meniul principal sau pe butonul "X".

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.25.3.12 mouseReleased()

```
void gamestates.Leaderboard.mouseReleased (
            MouseEvent e )
```

Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.25.3.13 mouseWheelMoved()

```
void gamestates.Leaderboard.mouseWheelMoved (
            MouseWheelEvent e )
```

Gestionează evenimentele de rotiță a mouse-ului pentru derularea listei clasamentului.

Parameters

e	Evenimentul MouseWheelEvent.

6.25.3.14 refreshLeaderboardData()

```
void gamestates.Leaderboard.refreshLeaderboardData ( )  [private]
```

Reîmprospătează datele clasamentului prin încărcarea informațiilor despre toți jucătorii din baza de date și sortarea lor după scor.

6.25.3.15 setPreviousState()

```
void gamestates.Leaderboard.setPreviousState (
            Gamestate state )
```

Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Parameters

state	Starea anterioară Gamestate.

6.25.3.16 update()

```
void gamestates.Leaderboard.update ( )
```

Actualizează starea clasamentului.

Momentan, nu necesită actualizări logice per frame, deoarece redesenarea este gestionată de flag-ul needsRedraw

.

Implements gamestates.Statemethods.

6.25.4 Member Data Documentation

6.25.4.1 airstrikeFont

```
Font gamestates.Leaderboard.airstrikeFont  [private]
```

6.25.4.2 backButtonBounds

```
Rectangle gamestates.Leaderboard.backButtonBounds  [private]
```

6.25.4.3 backgroundImg

```
BufferedImage gamestates.Leaderboard.backgroundImg  [private]
```

6.25.4.4 boxSpacing

```
int gamestates.Leaderboard.boxSpacing = 110  [private]
```

Spațiul vertical (în pixeli) între fiecare intrare din clasament.

6.25.4.5 buffer

```
BufferedImage gamestates.Leaderboard.buffer  [private]
```

Un buffer grafic pentru a desena conținutul înainte de a-l afișa pe ecran (pentru performanță/evitarea pâlpâirii).

6.25.4.6 dateFormat

```
SimpleDateFormat gamestates.Leaderboard.dateFormat  [private]
```

6.25.4.7 DB_FILE

```
final String gamestates.Leaderboard.DB_FILE = "data/gamedatabase.db"  [static], [private]
```

Calea către fișierul bazei de date.

6.25.4.8 grooverImg

```
BufferedImage gamestates.Leaderboard.grooverImg  [private]
```

6.25.4.9 knobImg

```
BufferedImage gamestates.Leaderboard.knobImg  [private]
```

6.25.4.10 leaderboardData

```
List<PlayerData> gamestates.Leaderboard.leaderboardData = new ArrayList<>()  [private]
```

6.25.4.11 mainMenuButtonBounds

```
Rectangle gamestates.Leaderboard.mainMenuButtonBounds  [private]
```

6.25.4.12 menuX

```
int gamestates.Leaderboard.menuX  [private]
```

6.25.4.13 needsRedraw

```
boolean gamestates.Leaderboard.needsRedraw = true  [private]
```

Flag care indică dacă este necesară redesenarea conținutului buffer-ului.

6.25.4.14 playerBoxImg

```
BufferedImage gamestates.Leaderboard.playerBoxImg  [private]
```

6.25.4.15 previousState

```
Gamestate gamestates.Leaderboard.previousState = null  [private]
```

6.25.4.16 scrollOffset

```
int gamestates.Leaderboard.scrollOffset = 0  [private]
```

Offset-ul curent de derulare verticală a listei.

6.25.4.17 startbgImg

```
BufferedImage gamestates.Leaderboard.startbgImg  [private]
```

6.25.4.18 startbgX

```
int gamestates.Leaderboard.startbgX  [private]
```

6.25.4.19 visibleAreaHeight

```
int gamestates.Leaderboard.visibleAreaHeight = boxSpacing * 5  [private]
```

Înălțimea zonei vizibile a listei derulabile.

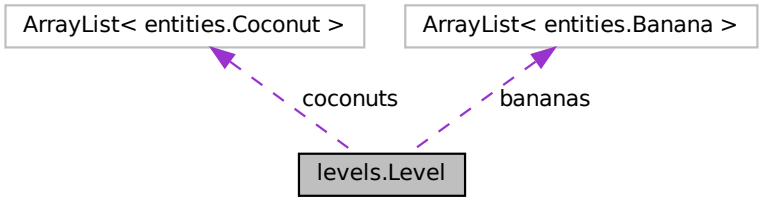
The documentation for this class was generated from the following file:

- src/gamestates/Leaderboard.java

6.26 levels.Level Class Reference

Reprezintă un nivel individual în joc.

Collaboration diagram for levels.Level:



Public Member Functions

- Level (int[ ][ ] lvlData, int levelId)
  Constructor principal pentru clasa Level.
- Level (int[ ][ ] lvlData)
  Constructor secundar pentru compatibilitate cu codul existent, care nu specifică un ID de nivel.
- int getSpriteIndex (int x, int y)
  Returnează indexul (ID-ul) sprite-ului pentru un tile la coordonatele specificate.
- void update (int playerX)
  Actualizează offset-ul de derulare al nivelului pe baza poziției jucătorului.
- int getLevelOffset ()
  Returnează offset-ul curent de derulare orizontală a nivelului, în tile-uri.
- int getMaxLevelOffsetX ()
  Returnează offset-ul maxim de derulare orizontală a nivelului, în pixeli.
- int[ ][ ] getLevelData ()
  Returnează matricea 2D cu datele tile-urilor nivelului.
- int getLevelId ()
  Returnează ID-ul nivelului curent.
- void setLevelId (int levelId)
  Setează ID-ul nivelului.
- ArrayList< Banana > getBananas ()

Returnează lista de banane din nivel.

- void setBananas (ArrayList< Banana > bananas)

  Setează lista de banane pentru nivel.
- void addBanana (Banana b)

  Adaugă o banană la lista de banane a nivelului.
- ArrayList< Coconut > getCoconuts ()

  Returnează lista de nuci de cocos din nivel.
- void setCoconuts (ArrayList< Coconut > coconuts)

  Setează lista de nuci de cocos pentru nivel.
- void addCoconut (Coconut c)

  Adaugă o nucă de cocos la lista de nuci de cocos a nivelului.

Private Attributes

- int[ ][ ] lvlData

  Matrice bidimensională ce conține datele despre tile-urile nivelului (ID-urile sprite-urilor).
- ArrayList< Banana > bananas

  Lista bananelor prezente în nivel.
- ArrayList< Coconut > coconuts

  Lista nucilor de cocos prezente în nivel.
- int levelOffset

  Offset-ul curent de derulare orizontală a nivelului, în tile-uri.
- int maxTilesOffset

  Offset-ul maxim de derulare orizontală a nivelului, în tile-uri.
- int maxLevelOffsetX

  Offset-ul maxim de derulare orizontală a nivelului, în pixeli.
- int levelId

  Identificatorul numeric al nivelului (de ex., 1, 2, 3).

6.26.1 Detailed Description

Reprezintă un nivel individual în joc.

Stochează datele despre tile-urile nivelului, obiectele colectabile (banane, nuci de cocos) și gestionează offset-ul de derulare (scrolling) al nivelului.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 Level() [1/2]

```
levels.Level.Level (
            int lvlData[][],
            int levelId )
```

Constructor principal pentru clasa Level.

Parameters

lvlData	Matricea 2D cu datele tile-urilor nivelului.
levelId	Identificatorul numeric al nivelului.

6.26.2.2 Level() [2/2]

```
levels.Level.Level (
            int lvlData[][] )
```

Constructor secundar pentru compatibilitate cu codul existent, care nu specifică un ID de nivel.

Implicit, se consideră nivelul 1.

Parameters

lvlData	Matricea 2D cu datele tile-urilor nivelului.

6.26.3 Member Function Documentation

6.26.3.1 addBanana()

```
void levels.Level.addBanana (
            Banana b )
```

Adaugă o banană la lista de banane a nivelului.

Inițializează lista dacă este null.

Parameters

b	Banana de adăugat.

6.26.3.2 addCoconut()

```
void levels.Level.addCoconut (
            Coconut c )
```

Adaugă o nucă de cocos la lista de nuci de cocos a nivelului.

Inițializează lista dacă este null.

Parameters

c	Nuca de cocos de adăugat.

6.26.3.3 getBananas()

```
ArrayList<Banana> levels.Level.getBananas ( )
```

Returnează lista de banane din nivel.

Returns

O listă de obiecte Banana. Poate fi null dacă nu au fost adăugate banane.

6.26.3.4 getCoconuts()

```
ArrayList<Coconut> levels.Level.getCoconuts ( )
```

Returnează lista de nuci de cocos din nivel.

Returns

O listă de obiecte Coconut. Poate fi null dacă nu au fost adăugate nuci de cocos.

6.26.3.5 getLevelData()

```
int [][] levels.Level.getLevelData ( )
```

Returnează matricea 2D cu datele tile-urilor nivelului.

Returns

Datele nivelului.

6.26.3.6 getLevelId()

```
int levels.Level.getLevelId ( )
```

Returnează ID-ul nivelului curent.

Returns

ID-ul nivelului (de ex., 1, 2, 3).

6.26.3.7 getLevelOffset()

```
int levels.Level.getLevelOffset ( )
```

Returnează offset-ul curent de derulare orizontală a nivelului, în tile-uri.

Returns

Offset-ul nivelului.

6.26.3.8 getMaxLevelOffsetX()

```
int levels.Level.getMaxLevelOffsetX ( )
```

Returnează offset-ul maxim de derulare orizontală a nivelului, în pixeli.

Returns

Offset-ul maxim al nivelului.

6.26.3.9 getSpriteIndex()

```
int levels.Level.getSpriteIndex (
            int x,
            int y )
```

Returnează indexul (ID-ul) sprite-ului pentru un tile la coordonatele specificate.

Parameters

x	Coordonata x a tile-ului (în unități de tile-uri).
y	Coordonata y a tile-ului (în unități de tile-uri).

Returns

ID-ul sprite-ului pentru tile-ul respectiv.

6.26.3.10 setBananas()

```
void levels.Level.setBananas (
            ArrayList< Banana > bananas )
```

Setează lista de banane pentru nivel.

Parameters

bananas	Noua listă de banane.

6.26.3.11 setCoconuts()

```
void levels.Level.setCoconuts (
            ArrayList< Coconut > coconuts )
```

Setează lista de nuci de cocos pentru nivel.

Parameters

coconuts	Noua listă de nuci de cocos.

6.26.3.12 setLevelId()

```
void levels.Level.setLevelId (
            int levelId )
```

Setează ID-ul nivelului.

Parameters

levelId	Noul ID pentru nivel.

6.26.3.13 update()

```
void levels.Level.update (
            int playerX )
```

Actualizează offset-ul de derulare al nivelului pe baza poziției jucătorului.

Această metodă nu este apelată direct în fluxul principal de actualizare al jocului, logica de scrolling fiind gestionată în clasa Playing.

Parameters

playerX	Poziția x a jucătorului (în pixeli).

6.26.4 Member Data Documentation

6.26.4.1 bananas

```
ArrayList<Banana> levels.Level.bananas  [private]
```

Lista bananelor prezente în nivel.

6.26.4.2 coconuts

```
ArrayList<Coconut> levels.Level.coconuts  [private]
```

Lista nucilor de cocos prezente în nivel.

6.26.4.3 levelId

```
int levels.Level.levelId  [private]
```

Identificatorul numeric al nivelului (de ex., 1, 2, 3).

6.26.4.4 levelOffset

```
int levels.Level.levelOffset  [private]
```

Offset-ul curent de derulare orizontală a nivelului, în tile-uri.

6.26.4.5 lvlData

```
int [][] levels.Level.lvlData  [private]
```

Matrice bidimensională ce conține datele despre tile-urile nivelului (ID-urile sprite-urilor).

6.26.4.6 maxLevelOffsetX

```
int levels.Level.maxLevelOffsetX  [private]
```

Offset-ul maxim de derulare orizontală a nivelului, în pixeli.

6.26.4.7 maxTilesOffset

```
int levels.Level.maxTilesOffset  [private]
```

Offset-ul maxim de derulare orizontală a nivelului, în tile-uri.

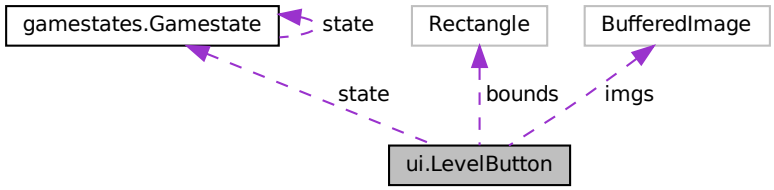
The documentation for this class was generated from the following file:

- src/levels/Level.java

6.27 ui.LevelButton Class Reference

Reprezintă un buton specific pentru selectarea nivelurilor în interfața utilizator.

Collaboration diagram for ui.LevelButton:



Public Member Functions

- LevelButton (int xPos, int yPos, int rowIndex, Gamestate state)

  Constructor pentru LevelButton.
- void draw (Graphics g)

  Desenează butonul pe ecran, folosind imaginea corespunzătoare stării curente.
- void update ()

  Actualizează starea vizuală a butonului (indexul imaginii) pe baza interacțiunii cu mouse-ul.
- boolean isMouseOver ()

  Verifică dacă mouse-ul este deasupra butonului.
- void setMouseOver (boolean mouseOver)

  Setează starea "mouse over" a butonului.
- boolean isMousePressed ()

Verifică dacă butonul este apăsat.

- void setMousePressed (boolean mousePressed)

  Setează starea de apăsare a butonului.
- Rectangle getBounds ()

  Returnează limitele dreptunghiulare (hitbox) ale butonului.
- void applyGamestate ()

  Aplică starea de joc asociată cu acest buton, schimbând starea globală a jocului.
- void resetBools ()

  Resetează flag-urile de interacțiune cu mouse-ul (mouseOver și mousePressed) la valorile implicite (false).
- int getRowIndex ()

  Returnează indexul rândului din spritesheet folosit pentru imaginile acestui buton.

Private Member Functions

- void initBounds ()

  Inițializează limitele dreptunghiulare (hitbox) ale butonului.
- void loadImgs ()

  Încarcă imaginile pentru buton din spritesheet-ul global de butoane.

Private Attributes

- int xPos
- int rowIndex

  Indexul rândului din spritesheet-ul de butoane de unde se încarcă imaginile pentru acest tip de buton.
- int index

  Indexul imaginii curente din array-ul.
- int xOffsetCenter = L_WIDTH / 2

  Offset orizontal pentru centrarea imaginii butonului față de.
- Gamestate state

  Starea de joc Gamestate care va fi activată la apăsarea butonului.
- BufferedImage[ ] imgs

  Array de imagini pentru diferitele stări ale butonului (normal, mouse over, apăsat).
- boolean mouseOver
- Rectangle bounds

  Dreptunghiul care definește limitele butonului pentru detecția coliziunilor cu mouse-ul.

6.27.1 Detailed Description

Reprezintă un buton specific pentru selectarea nivelurilor în interfața utilizator.

Gestionează stările vizuale ale butonului (normal, mouse over, apăsat) și acțiunea asociată.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 LevelButton()

```
ui.LevelButton.LevelButton (
            int xPos,
            int yPos,
            int rowIndex,
            Gamestate state )
```

Constructor pentru LevelButton.

Parameters

xPos	Poziția x a centrului butonului.
yPos	Poziția y a colțului de sus-stânga al butonului.
rowIndex	Indexul rândului din spritesheet-ul de butoane.
state	Starea de joc Gamestate asociată cu acest buton.

6.27.3 Member Function Documentation

6.27.3.1 applyGamestate()

```
void ui.LevelButton.applyGamestate ( )
```

Aplică starea de joc asociată cu acest buton, schimbând starea globală a jocului.

6.27.3.2 draw()

```
void ui.LevelButton.draw (
            Graphics g )
```

Desenează butonul pe ecran, folosind imaginea corespunzătoare stării curente.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.27.3.3 getBounds()

```
Rectangle ui.LevelButton.getBounds ( )
```

Returnează limitele dreptunghiulare (hitbox) ale butonului.

Returns

Obiectul Rectangle reprezentând limitele.

6.27.3.4 getRowIndex()

```
int ui.LevelButton.getRowIndex ( )
```

Returnează indexul rândului din spritesheet folosit pentru imaginile acestui buton.

Returns

Indexul rândului.

6.27.3.5 initBounds()

```
void ui.LevelButton.initBounds ( )  [private]
```

Inițializează limitele dreptunghiulare (hitbox) ale butonului.

6.27.3.6 isMouseOver()

```
boolean ui.LevelButton.isMouseOver ( )
```

Verifică dacă mouse-ul este deasupra butonului.

Returns

`true`
dacă mouse-ul este deasupra,
`false`
altfel.

6.27.3.7 isMousePressed()

```
boolean ui.LevelButton.isMousePressed ( )
```

Verifică dacă butonul este apăsat.

Returns

`true`
dacă butonul este apăsat,
`false`
altfel.

6.27.3.8 loadImgs()

```
void ui.LevelButton.loadImgs ( )  [private]
```

Încarcă imaginile pentru buton din spritesheet-ul global de butoane.

Folosește
`rowIndex`

pentru a selecta setul corect de imagini.

6.27.3.9 resetBools()

```
void ui.LevelButton.resetBools ( )
```

Resetează flag-urile de interacțiune cu mouse-ul (mouseOver și mousePressed) la valorile implicite (false).

6.27.3.10 setMouseOver()

```
void ui.LevelButton.setMouseOver (
            boolean mouseOver )
```

Setează starea "mouse over" a butonului.

Parameters

mouseOver	Noua stare "mouse over".

6.27.3.11 setMousePressed()

```
void ui.LevelButton.setMousePressed (
            boolean mousePressed )
```

Setează starea de apăsare a butonului.

Parameters

mousePressed	Noua stare de apăsare.

6.27.3.12 update()

```
void ui.LevelButton.update ( )
```

Actualizează starea vizuală a butonului (indexul imaginii) pe baza interacțiunii cu mouse-ul.

6.27.4 Member Data Documentation

6.27.4.1 bounds

```
Rectangle ui.LevelButton.bounds  [private]
```

Dreptunghiul care definește limitele butonului pentru detecția coliziunilor cu mouse-ul.

6.27.4.2 imgs

```
BufferedImage [] ui.LevelButton.imgs  [private]
```

Array de imagini pentru diferitele stări ale butonului (normal, mouse over, apăsat).

6.27.4.3 index

```
int ui.LevelButton.index  [private]
```

Indexul imaginii curente din array-ul.
imgs

, determinat de starea mouse-ului.

6.27.4.4 mouseOver

```
boolean ui.LevelButton.mouseOver  [private]
```

6.27.4.5 rowIndex

```
int ui.LevelButton.rowIndex  [private]
```

Indexul rândului din spritesheet-ul de butoane de unde se încarcă imaginile pentru acest tip de buton.

6.27.4.6 state

```
Gamestate ui.LevelButton.state  [private]
```

Starea de joc Gamestate care va fi activată la apăsarea butonului.

6.27.4.7 xOffsetCenter

```
int ui.LevelButton.xOffsetCenter = L_WIDTH / 2  [private]
```

Offset orizontal pentru centrarea imaginii butonului față de.
xPos

.

6.27.4.8 xPos

```
int ui.LevelButton.xPos  [private]
```

The documentation for this class was generated from the following file:

- src/ui/LevelButton.java

6.28 levels.LevelFactory Class Reference

Clasă de tip Factory responsabilă pentru crearea obiectelor de tip Level.

Public Member Functions

- int getTilesetRows (int levelNumber)
  Returnează numărul de rânduri din tileset pentru nivelul specificat.
- int getTilesetCols (int levelNumber)
  Returnează numărul de coloane din tileset pentru nivelul specificat.
- int getTileSize (int levelNumber)
  Returnează dimensiunea unui tile (în pixeli) pentru nivelul specificat.
- Level createLevel (int levelNumber)
  Creează un obiect Level pe baza numărului de nivel specificat.
- String getLevelAtlasPath (int levelNumber)
  Returnează calea către atlasul de tile-uri (spritesheet) corespunzător numărului de nivel.
- String getBackgroundPath (int levelNumber)
  Returnează calea către imaginea de fundal corespunzătoare numărului de nivel.

Static Public Attributes

- static final int LEVEL1_ROWS = 8
  Numărul de rânduri în tileset-ul pentru Nivelul 1.
- static final int LEVEL1_COLS = 12
  Numărul de coloane în tileset-ul pentru Nivelul 1.
- static final int LEVEL1_TILE_SIZE = 32
  Dimensiunea unui tile (în pixeli) pentru Nivelul 1.
- static final int LEVEL2_ROWS = 9
  Numărul de rânduri în tileset-ul pentru Nivelul 2.
- static final int LEVEL2_COLS = 9
  Numărul de coloane în tileset-ul pentru Nivelul 2.
- static final int LEVEL2_TILE_SIZE = 32
  Dimensiunea unui tile (în pixeli) pentru Nivelul 2.
- static final int LEVEL3_ROWS = 18
  Numărul de rânduri în tileset-ul pentru Nivelul 3.
- static final int LEVEL3_COLS = 15
  Numărul de coloane în tileset-ul pentru Nivelul 3.
- static final int LEVEL3_TILE_SIZE = 16
  Dimensiunea unui tile (în pixeli) pentru Nivelul 3.

Private Member Functions

- void preprocessEnemySpawnPoints (int[ ][ ] levelData)

  Pre-procesează datele nivelului pentru a gestiona punctele de spawn ale inamicilor.
- Level createLevel1 ()

  Creează și returnează Nivelul 1.
- Level createLevel2 ()

  Creează și returnează Nivelul 2.
- Level createLevel3 ()

  Creează și returnează Nivelul 3.

6.28.1 Detailed Description

Clasă de tip Factory responsabilă pentru crearea obiectelor de tip Level.

Implementează modelul de proiectare Factory pentru a abstractiza procesul de creare a nivelurilor.

6.28.2 Member Function Documentation

6.28.2.1 createLevel()

```
Level levels.LevelFactory.createLevel (
            int levelNumber )
```

Creează un obiect Level pe baza numărului de nivel specificat.

Parameters

levelNumber	Numărul nivelului de creat.

Returns

Obiectul Level creat.

6.28.2.2 createLevel1()

```
Level levels.LevelFactory.createLevel1 ( )  [private]
```

Creează și returnează Nivelul 1.

Încarcă datele specifice nivelului și le pre-procesează.

Returns

Obiectul Level pentru Nivelul 1.

6.28.2.3 createLevel2()

```
Level levels.LevelFactory.createLevel2 ( )  [private]
```

Creează și returnează Nivelul 2.

Încarcă datele specifice nivelului și le pre-procesează.

Returns

Obiectul Level pentru Nivelul 2.

6.28.2.4 createLevel3()

```
Level levels.LevelFactory.createLevel3 ( )  [private]
```

Creează și returnează Nivelul 3.

Încarcă datele specifice nivelului și le pre-procesează.

Returns

Obiectul Level pentru Nivelul 3.

6.28.2.5 getBackgroundPath()

```
String levels.LevelFactory.getBackgroundPath (
            int levelNumber )
```

Returnează calea către imaginea de fundal corespunzătoare numărului de nivel.

Parameters

levelNumber	Numărul nivelului.

Returns

Calea către fișierul imaginii de fundal.

6.28.2.6 getLevelAtlasPath()

```
String levels.LevelFactory.getLevelAtlasPath (
            int levelNumber )
```

Returnează calea către atlasul de tile-uri (spritesheet) corespunzător numărului de nivel.

Parameters

levelNumber	Numărul nivelului.

Returns

Calea către fișierul atlas.

6.28.2.7 getTilesetCols()

```
int levels.LevelFactory.getTilesetCols (
            int levelNumber )
```

Returnează numărul de coloane din tileset pentru nivelul specificat.

Parameters

levelNumber	Numărul nivelului.

Returns

Numărul de coloane din tileset.

6.28.2.8 getTilesetRows()

```
int levels.LevelFactory.getTilesetRows (
            int levelNumber )
```

Returnează numărul de rânduri din tileset pentru nivelul specificat.

Parameters

levelNumber	Numărul nivelului.

Returns

Numărul de rânduri din tileset.

6.28.2.9 getTileSize()

```
int levels.LevelFactory.getTileSize (
            int levelNumber )
```

Returnează dimensiunea unui tile (în pixeli) pentru nivelul specificat.

Parameters

levelNumber	Numărul nivelului.

Returns

Dimensiunea unui tile în pixeli.

6.28.2.10 preprocessEnemySpawnPoints()

```
void levels.LevelFactory.preprocessEnemySpawnPoints (
            int levelData[][] )  [private]
```

Pre-procesează datele nivelului pentru a gestiona punctele de spawn ale inamicilor.

Această metodă este apelată înainte de crearea obiectului Level pentru a se asigura că tile-urile corespunzătoare punctelor de spawn ale inamicilor nu sunt interpretate ca blocuri solide sau alte elemente de decor.

Parameters

levelData	Datele nivelului de procesat.

6.28.3 Member Data Documentation

6.28.3.1 LEVEL1_COLS

```
final int levels.LevelFactory.LEVEL1_COLS = 12  [static]
```

Numărul de coloane în tileset-ul pentru Nivelul 1.

6.28.3.2 LEVEL1_ROWS

```
final int levels.LevelFactory.LEVEL1_ROWS = 8  [static]
```

Numărul de rânduri în tileset-ul pentru Nivelul 1.

6.28.3.3 LEVEL1_TILE_SIZE

```
final int levels.LevelFactory.LEVEL1_TILE_SIZE = 32  [static]
```

Dimensiunea unui tile (în pixeli) pentru Nivelul 1.

6.28.3.4 LEVEL2_COLS

```
final int levels.LevelFactory.LEVEL2_COLS = 9  [static]
```

Numărul de coloane în tileset-ul pentru Nivelul 2.

6.28.3.5 LEVEL2_ROWS

```
final int levels.LevelFactory.LEVEL2_ROWS = 9  [static]
```

Numărul de rânduri în tileset-ul pentru Nivelul 2.

6.28.3.6 LEVEL2_TILE_SIZE

```
final int levels.LevelFactory.LEVEL2_TILE_SIZE = 32  [static]
```

Dimensiunea unui tile (în pixeli) pentru Nivelul 2.

6.28.3.7 LEVEL3_COLS

```
final int levels.LevelFactory.LEVEL3_COLS = 15  [static]
```

Numărul de coloane în tileset-ul pentru Nivelul 3.

6.28.3.8 LEVEL3_ROWS

```
final int levels.LevelFactory.LEVEL3_ROWS = 18  [static]
```

Numărul de rânduri în tileset-ul pentru Nivelul 3.

6.28.3.9 LEVEL3_TILE_SIZE

```
final int levels.LevelFactory.LEVEL3_TILE_SIZE = 16  [static]
```

Dimensiunea unui tile (în pixeli) pentru Nivelul 3.

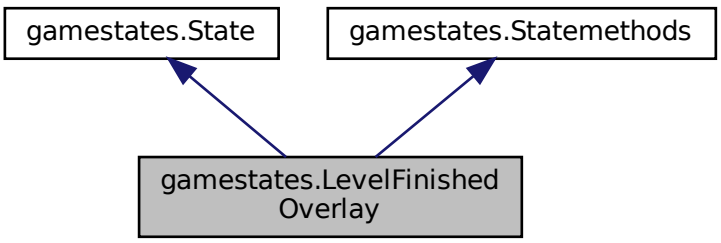
The documentation for this class was generated from the following file:

- src/levels/LevelFactory.java

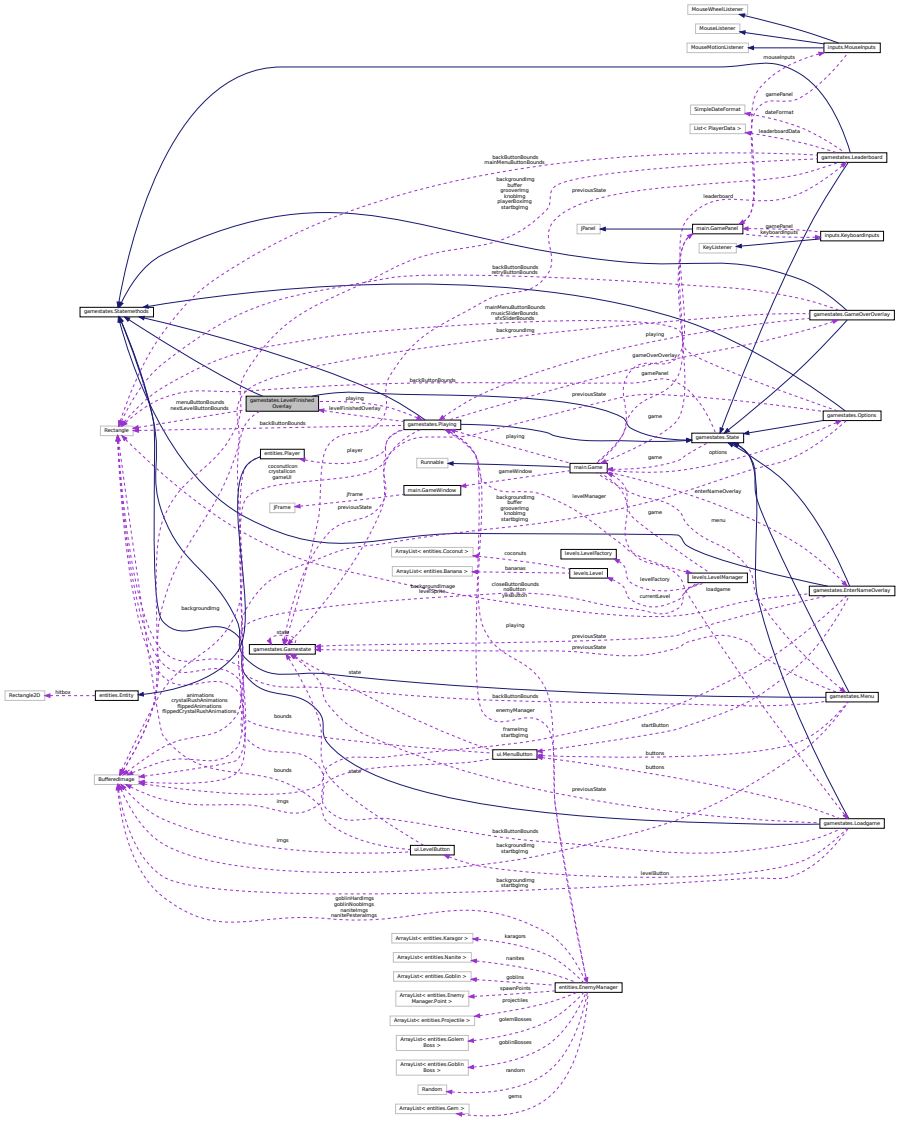
6.29 gamestates.LevelFinishedOverlay Class Reference

Reprezintă overlay-ul afișat la finalizarea cu succes a unui nivel.

Inheritance diagram for gamestates.LevelFinishedOverlay:



Collaboration diagram for gamestates.LevelFinishedOverlay:



Public Member Functions

- LevelFinishedOverlay (Playing playing)

  Constructor pentru LevelFinishedOverlay.
- void update ()

  Actualizează starea overlay-ului.
- void draw (Graphics g)

  Desenează elementele overlay-ului de finalizare a nivelului.
- void mouseClicked (MouseEvent e)

  Gestionarea evenimentului de click al mouse-ului (neutilizat).
- void mousePressed (MouseEvent e)

  Gestionează evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).
- void mouseMoved (MouseEvent e)

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).

- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionază evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).

Private Member Functions

- void loadBackground ()

  Încarcă și configurează imaginea de fundal pentru ecranul de finalizare a nivelului.
- void loadButtons ()

  Inițializează limitele butoanelor pentru "Nivelul Următor" și "Meniu" (butonul X).
- void loadCustomFont ()

  Încarcă fontul personalizat folosit pentru afișarea textului în overlay.

Private Attributes

- BufferedImage backgroundImg
- int bgX
- Rectangle nextLevelButtonBounds
- Rectangle menuButtonBounds
- Font overlayFont
- Playing playing

Static Private Attributes

- static final String DB_FILE = "data/gamedatabase.db"

  Calea către fișierul bazei de date pentru salvarea/încărcarea scorului.

Additional Inherited Members

6.29.1 Detailed Description

Reprezintă overlay-ul afișat la finalizarea cu succes a unui nivel.

Permite jucătorului să treacă la nivelul următor sau să se întoarcă la meniul principal. Afișează scorul obținut pentru nivelul finalizat. Extinde State și implementează Statemethods.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 LevelFinishedOverlay()

```
gamestates.LevelFinishedOverlay.LevelFinishedOverlay (
            Playing playing )
```

Constructor pentru LevelFinishedOverlay.

Parameters

playing	Referință la starea de joc Playing din care s-a ajuns aici.

6.29.3 Member Function Documentation

6.29.3.1 draw()

```
void gamestates.LevelFinishedOverlay.draw (
            Graphics g )
```

Desenează elementele overlay-ului de finalizare a nivelului.

Include imaginea de fundal și scorul jucătorului.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.29.3.2 keyPressed()

```
void gamestates.LevelFinishedOverlay.keyPressed (
            KeyEvent e )
```

Gestionază evenimentele de apăsare a tastelor.

Permite trecerea la starea de joc (Enter) sau la meniu (Escape). Notă: Trecerea directă la PLAYING cu Enter poate sări peste logica de avansare la nivelul următor.

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.29.3.3 keyReleased()

```
void gamestates.LevelFinishedOverlay.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.29.3.4 loadBackground()

```
void gamestates.LevelFinishedOverlay.loadBackground ( )  [private]
```

Încarcă și configurează imaginea de fundal pentru ecranul de finalizare a nivelului.

6.29.3.5 loadButtons()

```
void gamestates.LevelFinishedOverlay.loadButtons ( )  [private]
```

Inițializează limitele butoanelor pentru "Nivelul Următor" și "Meniu" (butonul X).

Coordonatele sunt specifice imaginii de fundal "level_finished.png".

6.29.3.6 loadCustomFont()

```
void gamestates.LevelFinishedOverlay.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat folosit pentru afișarea textului în overlay.

6.29.3.7 mouseClicked()

```
void gamestates.LevelFinishedOverlay.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Acțiunile sunt gestionate în mousePressed(MouseEvent).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.29.3.8 mouseDragged()

```
void gamestates.LevelFinishedOverlay.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.29.3.9 mouseMoved()

```
void gamestates.LevelFinishedOverlay.mouseMoved (
            MouseEvent e )
```

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.29.3.10 mousePressed()

```
void gamestates.LevelFinishedOverlay.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Verifică dacă s-a apăsat pe butonul pentru nivelul următor sau pe cel de întoarcere la meniu. Salvează progresul jucătorului înainte de a trece la nivelul următor.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.29.3.11 mouseReleased()

```
void gamestates.LevelFinishedOverlay.mouseReleased (
            MouseEvent e )
```

Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.29.3.12 update()

```
void gamestates.LevelFinishedOverlay.update ( )
```

Actualizează starea overlay-ului.

Momentan, nu face nimic deoarece overlay-ul este static.

Implements gamestates.Statemethods.

6.29.4 Member Data Documentation

6.29.4.1 backgroundImg

```
BufferedImage gamestates.LevelFinishedOverlay.backgroundImg  [private]
```

6.29.4.2 bgX

```
int gamestates.LevelFinishedOverlay.bgX  [private]
```

6.29.4.3 DB_FILE

```
final String gamestates.LevelFinishedOverlay.DB_FILE = "data/gamedatabase.db"  [static], [private]
```

Calea către fișierul bazei de date pentru salvarea/încărcarea scorului.

6.29.4.4 menuButtonBounds

```
Rectangle gamestates.LevelFinishedOverlay.menuButtonBounds  [private]
```

6.29.4.5 nextLevelButtonBounds

```
Rectangle gamestates.LevelFinishedOverlay.nextLevelButtonBounds  [private]
```

6.29.4.6 overlayFont

```
Font gamestates.LevelFinishedOverlay.overlayFont  [private]
```

6.29.4.7 playing

```
Playing gamestates.LevelFinishedOverlay.playing  [private]
```

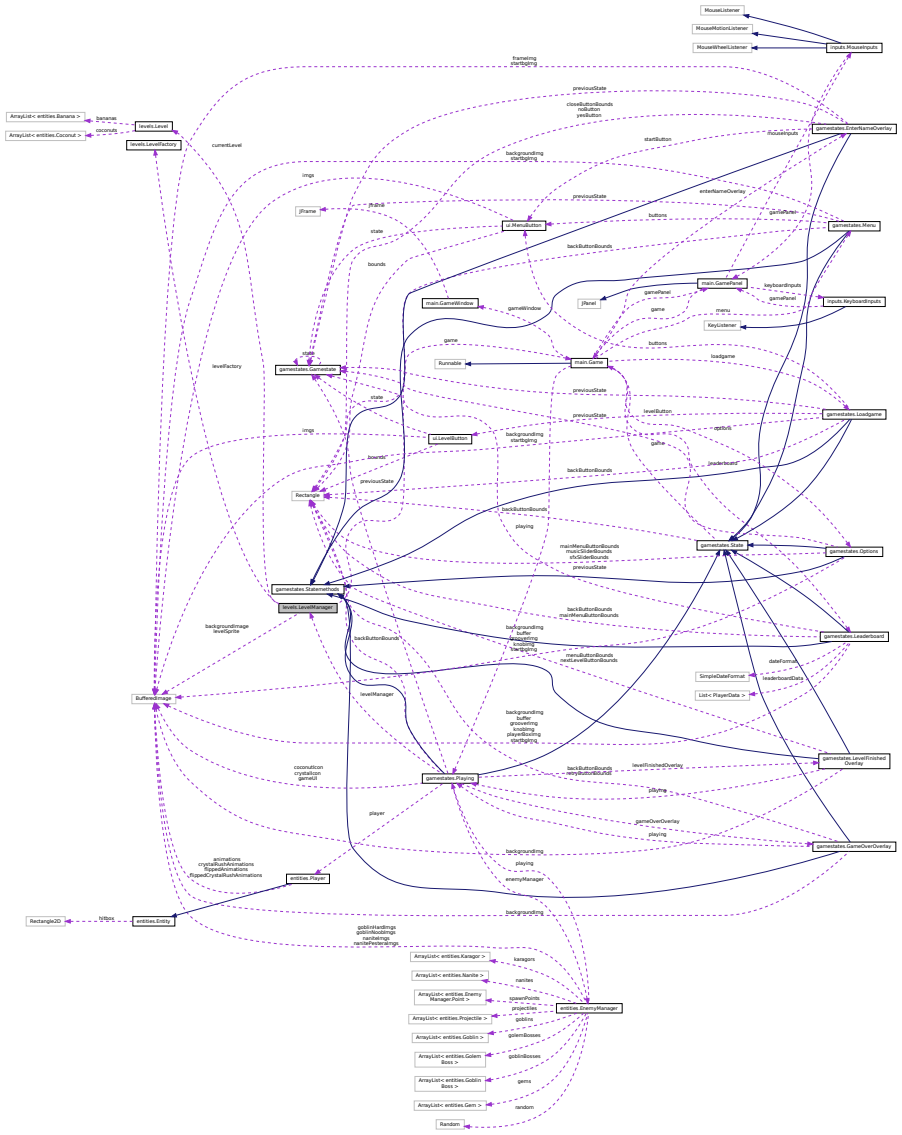
The documentation for this class was generated from the following file:

- src/gamestates/LevelFinishedOverlay.java

6.30 levels.LevelManager Class Reference

Gestionează încărcarea, desenarea și tranziția între nivelurile jocului.

Collaboration diagram for levels.LevelManager:



Public Member Functions

- LevelManager (Game game)

  Constructor pentru LevelManager.
- void loadLevel (int levelNumber)

  Încarcă un nivel specific pe baza numărului său.
- void draw (Graphics g, int xLvlOffset)

  Desenează nivelul curent pe ecran.
- void update ()

  Actualizează logica specifică managerului de niveluri.
- Level getCurrentLevel ()

  Returnează obiectul Level curent încărcat.
- int getCurrentLevelNumber ()

  Returnează numărul (indexul) nivelului curent.
- boolean nextLevel ()

Trece la următorul nivel, dacă există.

- boolean previousLevel ()

  Trece la nivelul anterior, dacă există.

Private Member Functions

- void importBackgroundForLevel (int levelNumber)

  Încarcă imaginea de fundal pentru nivelul specificat.
- void importSpritesForLevel (int levelNumber)

  Încarcă și prelucrează atlasul de sprite-uri (tileset) pentru nivelul specificat.

Private Attributes

- Game game
- BufferedImage[ ] levelSprite

  Array de imagini pentru tile-urile sprite-urilor nivelului curent.
- BufferedImage backgroundImage

  Imaginea de fundal pentru nivelul curent.
- Level currentLevel

  Obiectul Level curent încărcat.
- LevelFactory levelFactory

  Fabrica utilizată pentru a crea obiecte de tip Level.
- int currentLevelNumber

  Numărul (indexul) nivelului curent.

6.30.1 Detailed Description

Gestionează încărcarea, desenarea și tranziția între nivelurile jocului.

Utilizează LevelFactory pentru a crea instanțe de Level.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 LevelManager()

```
levels.LevelManager.LevelManager (
            Game game )
```

Constructor pentru LevelManager.

Inițializează fabrica de niveluri și încarcă primul nivel implicit.

Parameters

game	Referință la instanța principală a jocului Game.

6.30.3 Member Function Documentation

6.30.3.1 draw()

```
void levels.LevelManager.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează nivelul curent pe ecran.

Mai întâi desenează imaginea de fundal, apoi tile-urile nivelului.

Parameters

g	Contextul grafic Graphics pe care se va desena.
xLvlOffset	Offset-ul orizontal de derulare al nivelului.

6.30.3.2 getCurrentLevel()

```
Level levels.LevelManager.getCurrentLevel ( )
```

Returnează obiectul Level curent încărcat.

Returns

Nivelul curent.

6.30.3.3 getCurrentLevelNumber()

```
int levels.LevelManager.getCurrentLevelNumber ( )
```

Returnează numărul (indexul) nivelului curent.

Returns

Numărul nivelului curent.

6.30.3.4 importBackgroundForLevel()

```
void levels.LevelManager.importBackgroundForLevel (
            int levelNumber )  [private]
```

Încarcă imaginea de fundal pentru nivelul specificat.

Parameters

levelNumber	Numărul nivelului pentru care se încarcă fundalul.

6.30.3.5 importSpritesForLevel()

```
void levels.LevelManager.importSpritesForLevel (
            int levelNumber )  [private]
```

Încarcă și prelucrează atlasul de sprite-uri (tileset) pentru nivelul specificat.

Extrage fiecare tile individual și îl stochează în array-ul
`levelSprite`

.

Parameters

levelNumber	Numărul nivelului pentru care se încarcă sprite-urile.

6.30.3.6 loadLevel()

```
void levels.LevelManager.loadLevel (
            int levelNumber )
```

Încarcă un nivel specific pe baza numărului său.

Importă sprite-urile și imaginea de fundal corespunzătoare și creează obiectul Level.

Parameters

levelNumber	Numărul nivelului de încărcat.

6.30.3.7 nextLevel()

```
boolean levels.LevelManager.nextLevel ( )
```

Trece la următorul nivel, dacă există.

Returns

`true`

dacă s-a trecut la următorul nivel cu succes,
`false`

dacă nu mai sunt niveluri.

6.30.3.8 previousLevel()

```
boolean levels.LevelManager.previousLevel ( )
```

Trece la nivelul anterior, dacă există.

Returns

`true`
dacă s-a trecut la nivelul anterior cu succes,
`false`
dacă nu există un nivel anterior.

6.30.3.9 update()

```
void levels.LevelManager.update ( )
```

Actualizează logica specifică managerului de niveluri.

Momentan, nu conține logică de actualizare specifică.

6.30.4 Member Data Documentation

6.30.4.1 backgroundImage

```
BufferedImage levels.LevelManager.backgroundImage  [private]
```

Imaginea de fundal pentru nivelul curent.

6.30.4.2 currentLevel

```
Level levels.LevelManager.currentLevel  [private]
```

Obiectul Level curent încărcat.

6.30.4.3 currentLevelNumber

```
int levels.LevelManager.currentLevelNumber  [private]
```

Numărul (indexul) nivelului curent.

6.30.4.4 game

```
Game levels.LevelManager.game  [private]
```

6.30.4.5 levelFactory

```
LevelFactory levels.LevelManager.levelFactory  [private]
```

Fabrica utilizată pentru a crea obiecte de tip Level.

6.30.4.6 levelSprite

```
BufferedImage [] levels.LevelManager.levelSprite  [private]
```

Array de imagini pentru tile-urile sprite-urilor nivelului curent.

The documentation for this class was generated from the following file:

- src/levels/LevelManager.java

6.31 utilz.LevelProgress Class Reference

Clasă simplă de date (POJO - Plain Old Java Object) pentru a stoca informațiile despre progresul jucătorului într-un anumit nivel.

Public Member Functions

- LevelProgress ()
  Constructor implicit.
- LevelProgress (int score, int health, int coconuts, int posX, int posY)
  Constructor cu parametri pentru a inițializa progresul cu valori specifice.

Public Attributes

- int score
  Scorul obținut de jucător în nivel.
- int health
  Viața rămasă a jucătorului.
- int coconuts
  Numărul de nuci de cocos colectate de jucător.
- int posX
  Poziția X a jucătorului în nivel.
- int posY
  Poziția Y a jucătorului în nivel.

6.31.1 Detailed Description

Clasă simplă de date (POJO - Plain Old Java Object) pentru a stoca informațiile despre progresul jucătorului într-un anumit nivel.

Utilizată pentru salvarea și încărcarea stării jocului.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 LevelProgress() [1/2]

```
utilz.LevelProgress.LevelProgress ( )
```

Constructor implicit.

Inițializează progresul cu valori default (scor 0, viață 100, nuci de cocos 0, poziție 0,0).

6.31.2.2 LevelProgress() [2/2]

```
utilz.LevelProgress.LevelProgress (
            int score,
            int health,
            int coconuts,
            int posX,
            int posY )
```

Constructor cu parametri pentru a inițializa progresul cu valori specifice.

Parameters

score	Scorul jucătorului.
health	Viața jucătorului.
coconuts	Numărul de nuci de cocos.
posX	Poziția X a jucătorului.
posY	Poziția Y a jucătorului.

6.31.3 Member Data Documentation

6.31.3.1 coconuts

```
int utilz.LevelProgress.coconuts
```

Numărul de nuci de cocos colectate de jucător.

6.31.3.2 health

```
int utilz.LevelProgress.health
```

Viața rămasă a jucătorului.

6.31.3.3 posX

```
int utilz.LevelProgress.posX
```

Poziția X a jucătorului în nivel.

6.31.3.4 posY

```
int utilz.LevelProgress.posY
```

Poziția Y a jucătorului în nivel.

6.31.3.5 score

```
int utilz.LevelProgress.score
```

Scorul obținut de jucător în nivel.

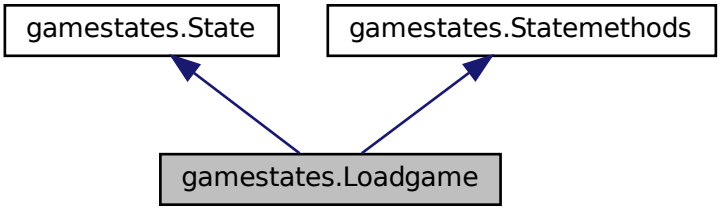
The documentation for this class was generated from the following file:

- src/utilz/LevelProgress.java

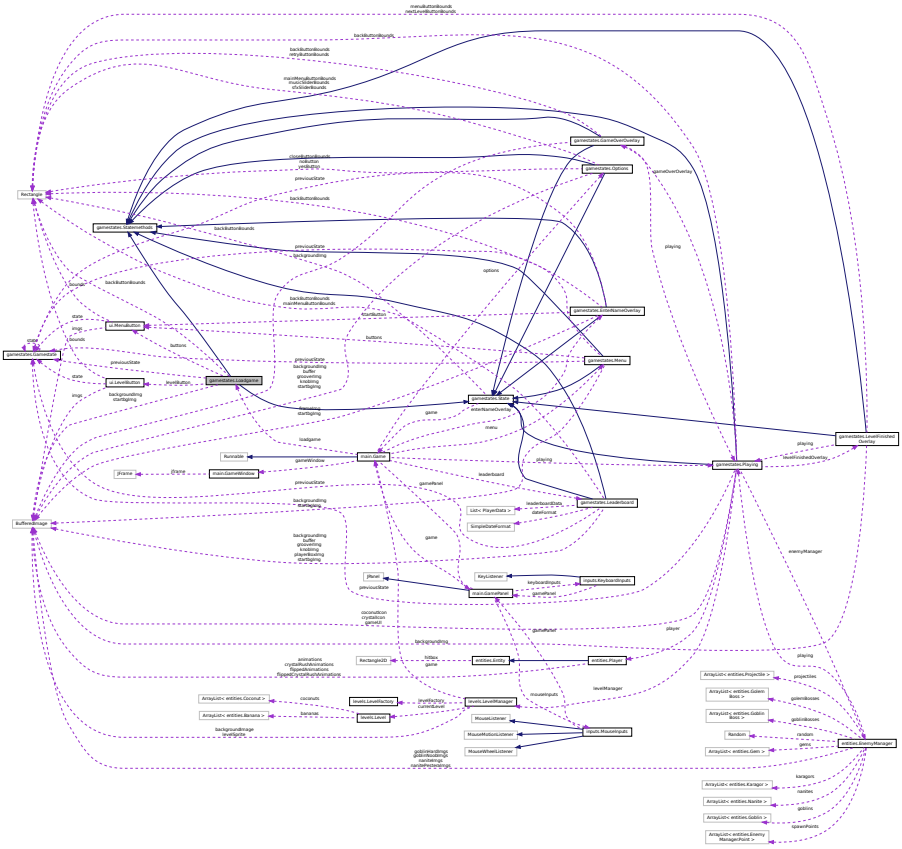
6.32 gamestates.Loadgame Class Reference

Reprezintă starea de joc pentru ecranul de încărcare a unui joc salvat.

Inheritance diagram for gamestates.Loadgame:



Collaboration diagram for gamestates.Loadgame:



Public Member Functions

- Loadgame (Game game)

Constructor pentru starea Loadgame.

- void update ()

  Actualizează starea butoanelor din ecranul de încărcare.
- void draw (Graphics g)

  Desenează elementele ecranului de încărcare a jocului.
- void mouseClicked (MouseEvent e)

  Gestionarea evenimentului de click al mouse-ului (neutilizat).
- void mousePressed (MouseEvent e)

  Gestionază evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionază evenimentele de eliberare a butonului mouse-ului.
- void mouseMoved (MouseEvent e)

  Gestionază evenimentele de mișcare a mouse-ului.
- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionază evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).
- void setPreviousState (Gamestate state)

  Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" (X).

Private Member Functions

- void loadStartImg ()

  Încarcă și configurează imaginea de fundal generală.
- void loadBackground ()

  Încarcă și configurează imaginea specifică a cadrului pentru ecranul de încărcare.
- void loadLevelButtons ()

  Inițializează și poziționează butoanele de selecție a nivelului.
- void loadButtons ()

  Inițializează butoanele de meniu (de ex., un buton "Load" general, dacă ar fi necesar).
- void resetButtons ()
- void loadLevel (int levelNumber, String username)

  Încarcă un nivel specific pentru un utilizator dat.

Private Attributes

- MenuButton[ ] buttons = new MenuButton[5]
- LevelButton[ ] levelButton = new LevelButton[3]
- BufferedImage backgroundImg
- int menuX
- BufferedImage startbgImg
- int startbgX
- Rectangle backButtonBounds
- Gamestate previousState = null

Additional Inherited Members

6.32.1 Detailed Description

Reprezintă starea de joc pentru ecranul de încărcare a unui joc salvat.

Permite jucătorului să selecteze un nivel salvat pentru a continua jocul. Extinde State și implementează Statemethods.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 Loadgame()

```
gamestates.Loadgame.Loadgame (
            Game game )
```

Constructor pentru starea Loadgame.

Inițializează butoanele, imaginile de fundal și alte elemente UI.

Parameters

game	Referință la instanța principală a jocului Game.

6.32.3 Member Function Documentation

6.32.3.1 draw()

```
void gamestates.Loadgame.draw (
            Graphics g )
```

Desenează elementele ecranului de încărcare a jocului.

Include fundalul, cadrul și butoanele.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.32.3.2 drawBackButton()

```
void gamestates.Loadgame.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" (X).

Suprascrie metoda din State.

Parameters

g2d	Contextul grafic 2D.

Reimplemented from gamestates.State.

6.32.3.3 keyPressed()

```
void gamestates.Loadgame.keyPressed (
            KeyEvent e )
```

Gestionază evenimentele de apăsare a tastelor.

Tasta Enter poate fi folosită pentru a confirma o acțiune (de ex., încărcarea unui slot implicit).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.32.3.4 keyReleased()

```
void gamestates.Loadgame.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.32.3.5 loadBackground()

```
void gamestates.Loadgame.loadBackground ( )  [private]
```

Încarcă și configurează imaginea specifică a cadrului pentru ecranul de încărcare.

6.32.3.6 loadButtons()

```
void gamestates.Loadgame.loadButtons ( )  [private]
```

Inițializează butoanele de meniu (de ex., un buton "Load" general, dacă ar fi necesar).

Momentan, pare să inițializeze un buton care duce la starea PLAYING, posibil pentru un slot de salvare general.

6.32.3.7 loadLevel()

```
void gamestates.Loadgame.loadLevel (
            int levelNumber,
            String username )  [private]
```

Încarcă un nivel specific pentru un utilizator dat.

Preia datele salvate din baza de date și configurează starea jocului (Playing).

Parameters

levelNumber	Numărul nivelului de încărcat.
username	Numele utilizatorului pentru care se încarcă nivelul.

6.32.3.8 loadLevelButtons()

```
void gamestates.Loadgame.loadLevelButtons ( )  [private]
```

Inițializează și poziționează butoanele de selecție a nivelului.

6.32.3.9 loadStartImg()

```
void gamestates.Loadgame.loadStartImg ( )  [private]
```

Încarcă și configurează imaginea de fundal generală.

6.32.3.10 mouseClicked()

```
void gamestates.Loadgame.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Acțiunile sunt gestionate în mousePressed(MouseEvent) și mouseReleased(MouseEvent).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.32.3.11 mouseDragged()

```
void gamestates.Loadgame.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.32.3.12 mouseMoved()

```
void gamestates.Loadgame.mouseMoved (
            MouseEvent e )
```

Gestionază evenimentele de mișcare a mouse-ului.

Setează starea "mouse over" pentru butoanele interactive.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.32.3.13 mousePressed()

```
void gamestates.Loadgame.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Setează starea de apăsare pentru butoanele interactive și gestionează butonul "X".

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.32.3.14 mouseReleased()

```
void gamestates.Loadgame.mouseReleased (
            MouseEvent e )
```

Gestionază evenimentele de eliberare a butonului mouse-ului.

Aplică acțiunea corespunzătoare butonului apăsat (încărcare nivel sau altă acțiune de meniu).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.32.3.15 resetButtons()

```
void gamestates.Loadgame.resetButtons ( )  [private]
```

6.32.3.16 setPreviousState()

```
void gamestates.Loadgame.setPreviousState (
            Gamestate state )
```

Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Parameters

state	Starea anterioară Gamestate.

6.32.3.17 update()

```
void gamestates.Loadgame.update ( )
```

Actualizează starea butoanelor din ecranul de încărcare.

Implements gamestates.Statemethods.

6.32.4 Member Data Documentation

6.32.4.1 backButtonBounds

```
Rectangle gamestates.Loadgame.backButtonBounds  [private]
```

6.32.4.2 backgroundImg

```
BufferedImage gamestates.Loadgame.backgroundImg  [private]
```

6.32.4.3 buttons

```
MenuButton [] gamestates.Loadgame.buttons = new MenuButton[5]  [private]
```

6.32.4.4 levelButton

```
LevelButton [] gamestates.Loadgame.levelButton = new LevelButton[3]  [private]
```

6.32.4.5 menuX

```
int gamestates.Loadgame.menuX  [private]
```

6.32.4.6 previousState

```
Gamestate gamestates.Loadgame.previousState = null  [private]
```

6.32.4.7 startbgImg

```
BufferedImage gamestates.Loadgame.startbgImg  [private]
```

6.32.4.8 startbgX

```
int gamestates.Loadgame.startbgX  [private]
```

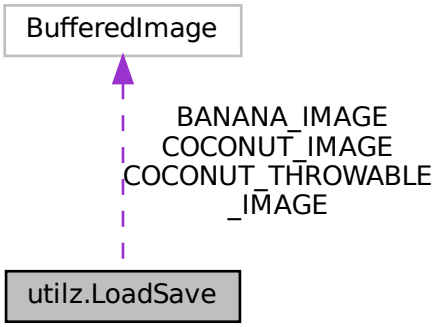
The documentation for this class was generated from the following file:

- src/gamestates/Loadgame.java

6.33 utilz.LoadSave Class Reference

Clasă utilitară responsabilă pentru încărcarea resurselor jocului, cum ar fi imaginile (sprite atlas-uri) și datele nivelurilor din fișiere.

Collaboration diagram for utilz.LoadSave:



Static Public Member Functions

- static BufferedImage getSpriteAtlas (String filename)

  Încarcă o imagine (sprite atlas) din directorul de resurse.
- static int[ ][ ] getLevelData (String filePath)

  Încarcă datele unui nivel dintr-un fișier CSV specificat.

Static Public Attributes

- static final String PLAYER_ATLAS = "koba_compressed_spritesheet.png"

  Calea către atlasul de sprite-uri pentru jucător (Koba).
- static final String KOBA_RUSH = "koba_compressed_spritesheet_cristal.png"

  Calea către atlasul de sprite-uri pentru Koba în starea "Crystal Rush".
- static final String LEVEL1_ATLAS = "Tileset_niv1.png"

  Calea către atlasul de tile-uri pentru Nivelul 1.
- static final String LEVEL1_BACKGROUND = "lvl1_bg.png"

  Calea către imaginea de fundal pentru Nivelul 1.
- static final String LEVEL1_DATA = "level1.csv"

  Calea către fișierul CSV cu datele pentru Nivelul 1.
- static final String LEVEL2_ATLAS = "Tileset_niv2.png"

  Calea către atlasul de tile-uri pentru Nivelul 2.
- static final String LEVEL2_BACKGROUND = "lvl2_bg.png"

  Calea către imaginea de fundal pentru Nivelul 2.
- static final String LEVEL2_DATA = "level2.csv"

  Calea către fișierul CSV cu datele pentru Nivelul 2.
- static final String LEVEL3_ATLAS = "Tileset_niv3.png"

  Calea către atlasul de tile-uri pentru Nivelul 3.
- static final String LEVEL3_BACKGROUND = "lvl3_bg.png"

  Calea către imaginea de fundal pentru Nivelul 3.
- static final String LEVEL3_DATA = "level3.csv"

  Calea către fișierul CSV cu datele pentru Nivelul 3.
- static final String MENU_BUTTONS = "sheet_but.png"

  Calea către spritesheet-ul pentru butoanele de meniu.
- static final String MENU_BACKGROUND = "menu _background.png"

  Calea către imaginea de fundal pentru meniul principal.
- static final String START_BACKGROUND = "start_bg.png"

  Calea către imaginea de fundal pentru ecranul de start/introducere nume.
- static final String FRAME_LOADGAME = "frame_load_game.png"

  Calea către imaginea cadrului pentru ecranul de încărcare a jocului.
- static final String LEVEL_BUTTONS = "level_but.png"

  Calea către spritesheet-ul pentru butoanele de selecție a nivelului.
- static final String ENTER_NAME_FRAME = "NAME.png"

  Calea către imaginea cadrului pentru ecranul de introducere a numelui.
- static final String NANITE_JUNGLA = "compressed_Nanite_Negre_Jungla.png"

  Calea către spritesheet-ul pentru Nanite-ii din junglă.
- static final String NANITE_PESTERA = "compressed_Nanite_Negre_Pestera.png"

  Calea către spritesheet-ul pentru Nanite-ii din peșteră.
- static final String GAME_UI = "game UI_clean.png"

  Calea către imaginea pentru interfața utilizator (HUD).
- static final String KARAGOR_SPRITESHEET = "karagor_compressed_spritesheet.png"

Calea către spritesheet-ul pentru inamicul Karagor.

- static final String GREEN_GEM = "green_gem.png"

  Calea către imaginea pentru piatra prețioasă verde.
- static final String ORANGE_GEM = "orange_gem.png"

  Calea către imaginea pentru piatra prețioasă portocalie.
- static final String PURPLE_GEM = "purple_gem.png"

  Calea către imaginea pentru piatra prețioasă mov.
- static final String BANANA_SPRITE = "banana.png"

  Calea către sprite-ul pentru banană (colectabil).
- static final String COCONUT_SPRITE = "coconut_static.png"

  Calea către sprite-ul pentru nuca de cocos statică (colectabil).
- static final String COCONUT_THROWABLE_SPRITE = "coconut.png"

  Calea către sprite-ul pentru nuca de cocos aruncabilă (proiectil).
- static final String GOBLIN_NOOB_SPRITESHEET = "compressed_goblin_mob_noob.png"

  Calea către spritesheet-ul pentru Goblinii de tip "noob".
- static final String GOBLIN_HARD_SPRITESHEET = "compressed_goblin_mob_hard.png"

  Calea către spritesheet-ul pentru Goblinii de tip "hard".
- static final String GOBLIN_BOSS_SPRITESHEET = "compressed_goblin_boss.png"

  Calea către spritesheet-ul pentru Goblin Boss.
- static final String GOLEM_BOSS_SPRITESHEET = "compressed_golem_boss_purple.png"

  Calea către spritesheet-ul pentru Golem Boss.
- static BufferedImage BANANA_IMAGE

  Imaginea preîncărcată pentru banană.
- static BufferedImage COCONUT_IMAGE

  Imaginea preîncărcată pentru nuca de cocos statică (colectabil).
- static BufferedImage COCONUT_THROWABLE_IMAGE

  Imaginea preîncărcată pentru nuca de cocos aruncabilă.

6.33.1 Detailed Description

Clasă utilitară responsabilă pentru încărcarea resurselor jocului, cum ar fi imaginile (sprite atlas-uri) și datele nivelurilor din fișiere.

Conține constante pentru căile către fișierele de resurse și metode statice pentru a accesa aceste resurse.

6.33.2 Member Function Documentation

6.33.2.1 getLevelData()

```
static int [][] utilz.LoadSave.getLevelData (
            String filePath )  [static]
```

Încarcă datele unui nivel dintr-un fișier CSV specificat.

Fișierul CSV trebuie să fie localizat în directorul de resurse ("/res/"). Fiecare linie din CSV reprezintă un rând de tile-uri, iar valorile sunt separate prin spații.

Parameters

filePath	Calea către fișierul CSV, relativă la directorul "/res/" (de ex., "level1.csv").

Returns

O matrice bidimensională de întregi (`int[][]`) reprezentând datele nivelului. Returnează un array gol (`new int[0][0]`) dacă fișierul nu este găsit sau apare o eroare.

6.33.2.2 getSpriteAtlas()

```
static BufferedImage utilz.LoadSave.getSpriteAtlas (
            String filename )  [static]
```

Încarcă o imagine (sprite atlas) din directorul de resurse.

Parameters

filename	Numele fișierului imagine (de ex., "player_atlas.png").

Returns

Un obiect BufferedImage reprezentând imaginea încărcată, sau `null` dacă încărcarea eșuează.

6.33.3 Member Data Documentation

6.33.3.1 BANANA_IMAGE

```
BufferedImage utilz.LoadSave.BANANA_IMAGE  [static]
```

Imaginea preîncărcată pentru banană.

6.33.3.2 BANANA_SPRITE

```
final String utilz.LoadSave.BANANA_SPRITE = "banana.png"  [static]
```

Calea către sprite-ul pentru banană (colectabil).

6.33.3.3 COCONUT_IMAGE

```
BufferedImage utilz.LoadSave.COCONUT_IMAGE  [static]
```

Imaginea preîncărcată pentru nuca de cocos statică (colectabil).

6.33.3.4 COCONUT_SPRITE

```
final String utilz.LoadSave.COCONUT_SPRITE = "coconut_static.png"  [static]
```

Calea către sprite-ul pentru nuca de cocos statică (colectabil).

6.33.3.5 COCONUT_THROWABLE_IMAGE

```
BufferedImage utilz.LoadSave.COCONUT_THROWABLE_IMAGE  [static]
```

Imaginea preîncărcată pentru nuca de cocos aruncabilă.

6.33.3.6 COCONUT_THROWABLE_SPRITE

```
final String utilz.LoadSave.COCONUT_THROWABLE_SPRITE = "coconut.png"  [static]
```

Calea către sprite-ul pentru nuca de cocos aruncabilă (proiectil).

6.33.3.7 ENTER_NAME_FRAME

```
final String utilz.LoadSave.ENTER_NAME_FRAME = "NAME.png"  [static]
```

Calea către imaginea cadrului pentru ecranul de introducere a numelui.

6.33.3.8 FRAME_LOADGAME

```
final String utilz.LoadSave.FRAME_LOADGAME = "frame_load_game.png"  [static]
```

Calea către imaginea cadrului pentru ecranul de încărcare a jocului.

6.33.3.9 GAME_UI

```
final String utilz.LoadSave.GAME_UI = "game UI_clean.png"  [static]
```

Calea către imaginea pentru interfața utilizator (HUD).

6.33.3.10 GOBLIN_BOSS_SPRITESHEET

```
final String utilz.LoadSave.GOBLIN_BOSS_SPRITESHEET = "compressed_goblin_boss.png"  [static]
```

Calea către spritesheet-ul pentru Goblin Boss.

6.33.3.11 GOBLIN_HARD_SPRITESHEET

```
final String utilz.LoadSave.GOBLIN_HARD_SPRITESHEET = "compressed_goblin_mob_hard.png"  [static]
```

Calea către spritesheet-ul pentru Goblinii de tip "hard".

6.33.3.12 GOBLIN_NOOB_SPRITESHEET

```
final String utilz.LoadSave.GOBLIN_NOOB_SPRITESHEET = "compressed_goblin_mob_noob.png"  [static]
```

Calea către spritesheet-ul pentru Goblinii de tip "noob".

6.33.3.13 GOLEM_BOSS_SPRITESHEET

```
final String utilz.LoadSave.GOLEM_BOSS_SPRITESHEET = "compressed_golem_boss_purple.png"  [static]
```

Calea către spritesheet-ul pentru Golem Boss.

6.33.3.14 GREEN_GEM

```
final String utilz.LoadSave.GREEN_GEM = "green_gem.png"  [static]
```

Calea către imaginea pentru piatra prețioasă verde.

6.33.3.15 KARAGOR_SPRITESHEET

```
final String utilz.LoadSave.KARAGOR_SPRITESHEET = "karagor_compressed_spritesheet.png"  [static]
```

Calea către spritesheet-ul pentru inamicul Karagor.

6.33.3.16 KOBA_RUSH

```
final String utilz.LoadSave.KOBA_RUSH = "koba_compressed_spritesheet_cristal.png"  [static]
```

Calea către atlasul de sprite-uri pentru Koba în starea "Crystal Rush".

6.33.3.17 LEVEL1_ATLAS

```
final String utilz.LoadSave.LEVEL1_ATLAS = "Tileset_niv1.png"  [static]
```

Calea către atlasul de tile-uri pentru Nivelul 1.

6.33.3.18 LEVEL1_BACKGROUND

```
final String utilz.LoadSave.LEVEL1_BACKGROUND = "lvl1_bg.png"  [static]
```

Calea către imaginea de fundal pentru Nivelul 1.

6.33.3.19 LEVEL1_DATA

```
final String utilz.LoadSave.LEVEL1_DATA = "level1.csv"  [static]
```

Calea către fișierul CSV cu datele pentru Nivelul 1.

6.33.3.20 LEVEL2_ATLAS

```
final String utilz.LoadSave.LEVEL2_ATLAS = "Tileset_niv2.png"  [static]
```

Calea către atlasul de tile-uri pentru Nivelul 2.

6.33.3.21 LEVEL2_BACKGROUND

```
final String utilz.LoadSave.LEVEL2_BACKGROUND = "lvl2_bg.png"  [static]
```

Calea către imaginea de fundal pentru Nivelul 2.

6.33.3.22 LEVEL2_DATA

```
final String utilz.LoadSave.LEVEL2_DATA = "level2.csv"  [static]
```

Calea către fișierul CSV cu datele pentru Nivelul 2.

6.33.3.23 LEVEL3_ATLAS

```
final String utilz.LoadSave.LEVEL3_ATLAS = "Tileset_niv3.png"  [static]
```

Calea către atlasul de tile-uri pentru Nivelul 3.

6.33.3.24 LEVEL3_BACKGROUND

```
final String utilz.LoadSave.LEVEL3_BACKGROUND = "lvl3_bg.png"  [static]
```

Calea către imaginea de fundal pentru Nivelul 3.

6.33.3.25 LEVEL3_DATA

```
final String utilz.LoadSave.LEVEL3_DATA = "level3.csv"  [static]
```

Calea către fișierul CSV cu datele pentru Nivelul 3.

6.33.3.26 LEVEL_BUTTONS

```
final String utilz.LoadSave.LEVEL_BUTTONS = "level_but.png"  [static]
```

Calea către spritesheet-ul pentru butoanele de selecție a nivelului.

6.33.3.27 MENU_BACKGROUND

```
final String utilz.LoadSave.MENU_BACKGROUND = "menu _background.png"  [static]
```

Calea către imaginea de fundal pentru meniul principal.

6.33.3.28 MENU_BUTTONS

```
final String utilz.LoadSave.MENU_BUTTONS = "sheet_but.png"  [static]
```

Calea către spritesheet-ul pentru butoanele de meniu.

6.33.3.29 NANITE_JUNGLA

```
final String utilz.LoadSave.NANITE_JUNGLA = "compressed_Nanite_Negre_Jungla.png"  [static]
```

Calea către spritesheet-ul pentru Nanite-ii din junglă.

6.33.3.30 NANITE_PESTERA

```
final String utilz.LoadSave.NANITE_PESTERA = "compressed_Nanite_Negre_Pestera.png"  [static]
```

Calea către spritesheet-ul pentru Nanite-ii din peșteră.

6.33.3.31 ORANGE_GEM

```
final String utilz.LoadSave.ORANGE_GEM = "orange_gem.png"  [static]
```

Calea către imaginea pentru piatra prețioasă portocalie.

6.33.3.32 PLAYER_ATLAS

```
final String utilz.LoadSave.PLAYER_ATLAS = "koba_compressed_spritesheet.png"  [static]
```

Calea către atlasul de sprite-uri pentru jucător (Koba).

6.33.3.33 PURPLE_GEM

```
final String utilz.LoadSave.PURPLE_GEM = "purple_gem.png"  [static]
```

Calea către imaginea pentru piatra prețioasă mov.

6.33.3.34 START_BACKGROUND

```
final String utilz.LoadSave.START_BACKGROUND = "start_bg.png"  [static]
```

Calea către imaginea de fundal pentru ecranul de start/introducere nume.

The documentation for this class was generated from the following file:

- src/utilz/LoadSave.java

6.34 main.Main Class Reference

Clasa principală a aplicației, conținând punctul de intrare (metoda main).

Static Public Member Functions

- static void main (String[ ] args)
  Punctul de intrare principal pentru aplicația jocului.

6.34.1 Detailed Description

Clasa principală a aplicației, conținând punctul de intrare (metoda main).

Responsabilitatea sa este de a inițializa și porni jocul.

6.34.2 Member Function Documentation

6.34.2.1 main()

```
static void main.Main.main (
            String[] args )  [static]
```

Punctul de intrare principal pentru aplicația jocului.

Creează o nouă instanță a clasei Game, care la rândul său inițializează fereastra jocului, panoul și începe bucla principală a jocului.

Parameters

args	Argumentele liniei de comandă (neutilizate în acest joc).

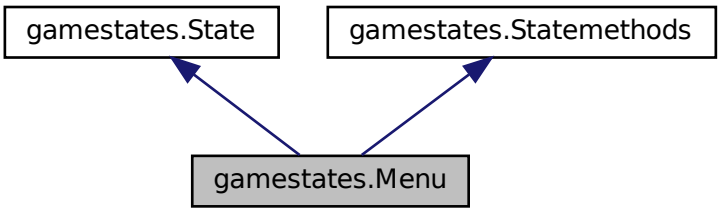
The documentation for this class was generated from the following file:

- src/main/Main.java

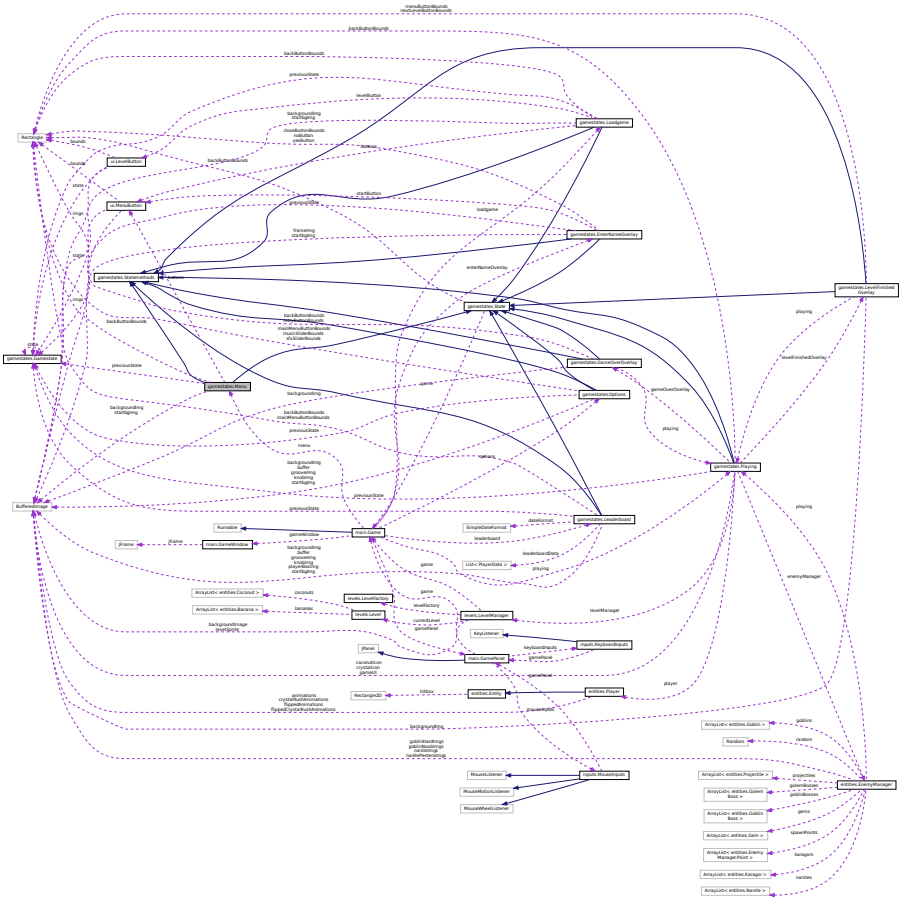
6.35 gamestates.Menu Class Reference

Reprezintă starea de joc pentru meniul principal.

Inheritance diagram for gamestates.Menu:



Collaboration diagram for gamestates.Menu:



Public Member Functions

- Menu (Game game)

  Constructor pentru starea Menu.
- void update ()

  Actualizează starea butoanelor din meniu.
- void draw (Graphics g)

  Desenează elementele meniului principal.
- void mouseClicked (MouseEvent e)

  Gestionarea evenimentului de click al mouse-ului (neutilizat).
- void mousePressed (MouseEvent e)

  Gestionază evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionază evenimentele de eliberare a butonului mouse-ului.
- void mouseMoved (MouseEvent e)

  Gestionază evenimentele de mișcare a mouse-ului.
- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat).
- void keyPressed (KeyEvent e)

  Gestionază evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

Gestionarea evenimentului de eliberare a tastei (neutilizat).

- void setPlayerName (String name)

  Setează numele jucătorului.
- String getPlayerName ()

  Returnează numele jucătorului.

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" (X).

Private Member Functions

- void loadStartImg ()

  Încarcă și configurează imaginea de fundal generală.
- void loadBackground ()

  Încarcă și configurează imaginea specifică a cadrului meniului.
- void loadButtons ()

  Inițializează și poziționează butoanele din meniul principal.
- void resetButtons ()

Private Attributes

- String playerName = ""

  Numele jucătorului, poate fi afișat în meniu.
- MenuButton[ ] buttons = new MenuButton[5]
- BufferedImage backgroundImg
- int menuX
- BufferedImage startbgImg
- int startbgX
- Rectangle backButtonBounds
- Gamestate previousState = null

Additional Inherited Members

6.35.1 Detailed Description

Reprezintă starea de joc pentru meniul principal.

Afișează opțiuni precum "Start Joc", "Încarcă Joc", "Clasament", "Opțiuni" și "Ieșire". Extinde State și implementează Statemethods.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 Menu()

```
gamestates.Menu.Menu (
            Game game )
```

Constructor pentru starea Menu.

Inițializează butoanele și imaginile de fundal.

Parameters

game	Referință la instanța principală a jocului Game.

6.35.3 Member Function Documentation

6.35.3.1 draw()

```
void gamestates.Menu.draw (
            Graphics g )
```

Desenează elementele meniului principal.

Include fundalul, cadrul și butoanele.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.35.3.2 drawBackButton()

```
void gamestates.Menu.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" (X).

Suprascrie metoda din State.

Parameters

g2d	Contextul grafic 2D.

Reimplemented from gamestates.State.

6.35.3.3 getPlayerName()

```
String gamestates.Menu.getPlayerName ( )
```

Returnează numele jucătorului.

Returns

Numele jucătorului.

6.35.3.4 keyPressed()

```
void gamestates.Menu.keyPressed (
            KeyEvent e )
```

Gestionază evenimentele de apăsare a tastelor.

Tasta Enter poate fi folosită pentru a începe un joc nou.

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.35.3.5 keyReleased()

```
void gamestates.Menu.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.35.3.6 loadBackground()

```
void gamestates.Menu.loadBackground ( )  [private]
```

Încarcă și configurează imaginea specifică a cadrului meniului.

6.35.3.7 loadButtons()

```
void gamestates.Menu.loadButtons ( )  [private]
```

Inițializează și poziționează butoanele din meniul principal.

6.35.3.8 loadStartImg()

```
void gamestates.Menu.loadStartImg ( )  [private]
```

Încarcă și configurează imaginea de fundal generală.

6.35.3.9 mouseClicked()

```
void gamestates.Menu.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Acțiunile sunt gestionate în mousePressed(MouseEvent) și mouseReleased(MouseEvent).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.35.3.10 mouseDragged()

```
void gamestates.Menu.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.35.3.11 mouseMoved()

```
void gamestates.Menu.mouseMoved (
            MouseEvent e )
```

Gestionază evenimentele de mișcare a mouse-ului.

Setează starea "mouse over" pentru butoanele interactive.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.35.3.12 mousePressed()

```
void gamestates.Menu.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Setează starea de apăsare pentru butoanele interactive și gestionează butonul "X".

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.35.3.13 mouseReleased()

```
void gamestates.Menu.mouseReleased (
            MouseEvent e )
```

Gestionază evenimentele de eliberare a butonului mouse-ului.

Aplică starea de joc corespunzătoare butonului apăsat. Setează starea anterioară pentru ecranele de Opțiuni și Încarcă Joc. Resetează jocul dacă se intră în starea PLAYING.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.35.3.14 resetButtons()

```
void gamestates.Menu.resetButtons ( )  [private]
```

6.35.3.15 setPlayerName()

```
void gamestates.Menu.setPlayerName (
            String name )
```

Setează numele jucătorului.

Parameters

name	Numele jucătorului.

6.35.3.16 update()

```
void gamestates.Menu.update ( )
```

Actualizează starea butoanelor din meniu.

Implements gamestates.Statemethods.

6.35.4 Member Data Documentation

6.35.4.1 backButtonBounds

```
Rectangle gamestates.Menu.backButtonBounds  [private]
```

6.35.4.2 backgroundImg

```
BufferedImage gamestates.Menu.backgroundImg  [private]
```

6.35.4.3 buttons

```
MenuButton [] gamestates.Menu.buttons = new MenuButton[5]  [private]
```

6.35.4.4 menuX

```
int gamestates.Menu.menuX  [private]
```

6.35.4.5 playerName

```
String gamestates.Menu.playerName = ""  [private]
```

Numele jucătorului, poate fi afișat în meniu.

6.35.4.6 previousState

```
Gamestate gamestates.Menu.previousState = null  [private]
```

6.35.4.7 startbgImg

```
BufferedImage gamestates.Menu.startbgImg  [private]
```

6.35.4.8 startbgX

```
int gamestates.Menu.startbgX  [private]
```

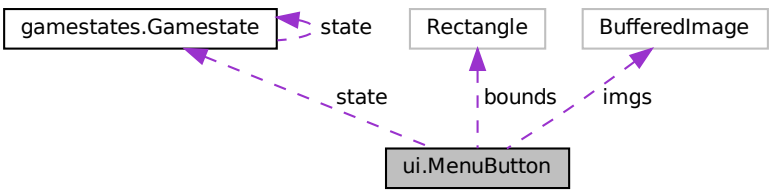
The documentation for this class was generated from the following file:

- src/gamestates/Menu.java

6.36 ui.MenuButton Class Reference

Reprezintă un buton generic pentru meniurile din interfața utilizator.

Collaboration diagram for ui.MenuButton:



Public Member Functions

- MenuButton (int xPos, int yPos, int rowIndex, Gamestate state)

  Constructor pentru MenuButton.
- void draw (Graphics g)

  Desenează butonul pe ecran, folosind imaginea corespunzătoare stării curente.
- void update ()

  Actualizează starea vizuală a butonului (indexul imaginii) pe baza interacțiunii cu mouse-ul.
- boolean isMouseOver ()

  Verifică dacă mouse-ul este deasupra butonului.
- void setMouseOver (boolean mouseOver)

  Setează starea "mouse over" a butonului.
- boolean isMousePressed ()

  Verifică dacă butonul este apăsat.
- void setMousePressed (boolean mousePressed)

  Setează starea de apăsare a butonului.
- Rectangle getBounds ()

  Returnează limitele dreptunghiulare (hitbox) ale butonului.
- void applyGamestate ()

  Aplică starea de joc asociată cu acest buton, schimbând starea globală a jocului.
- Gamestate getState ()

  Returnează starea de joc Gamestate asociată cu acest buton.
- void resetBools ()

  Resetează flag-urile de interacțiune cu mouse-ul (mouseOver și mousePressed) la valorile implicite (false).

Private Member Functions

- void initBounds ()

  Inițializează limitele dreptunghiulare (hitbox) ale butonului.
- void loadImgs ()

  Încarcă imaginile pentru buton din spritesheet-ul global de butoane de meniu.

Private Attributes

- int xPos
- int rowIndex

  Indexul rândului din spritesheet-ul de butoane de unde se încarcă imaginile pentru acest tip de buton.
- int index

  Indexul imaginii curente din array-ul.
- int xOffsetCenter = B_WIDTH / 2

  Offset orizontal pentru centrarea imaginii butonului față de.
- Gamestate state

  Starea de joc Gamestate care va fi activată la apăsarea butonului.
- BufferedImage[ ] imgs

  Array de imagini pentru diferitele stări ale butonului (normal, mouse over, apăsat).
- boolean mouseOver
- Rectangle bounds

  Dreptunghiul care definește limitele butonului pentru detecția coliziunilor cu mouse-ul.

6.36.1 Detailed Description

Reprezintă un buton generic pentru meniurile din interfața utilizator.

Gestionează stările vizuale ale butonului (normal, mouse over, apăsat) și acțiunea asociată (schimbarea stării de joc).

6.36.2 Constructor & Destructor Documentation

6.36.2.1 MenuButton()

```
ui.MenuButton.MenuButton (
            int xPos,
            int yPos,
            int rowIndex,
            Gamestate state )
```

Constructor pentru MenuButton.

Parameters

xPos	Poziția x a centrului butonului.
yPos	Poziția y a colțului de sus-stânga al butonului.
rowIndex	Indexul rândului din spritesheet-ul de butoane (specific pentru tipul de buton de meniu).
state	Starea de joc Gamestate asociată cu acest buton.

6.36.3 Member Function Documentation

6.36.3.1 applyGamestate()

```
void ui.MenuButton.applyGamestate ( )
```

Aplică starea de joc asociată cu acest buton, schimbând starea globală a jocului.

6.36.3.2 draw()

```
void ui.MenuButton.draw (
            Graphics g )
```

Desenează butonul pe ecran, folosind imaginea corespunzătoare stării curente.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.36.3.3 getBounds()

```
Rectangle ui.MenuButton.getBounds ( )
```

Returnează limitele dreptunghiulare (hitbox) ale butonului.

Returns

Obiectul Rectangle reprezentând limitele.

6.36.3.4 getState()

```
Gamestate ui.MenuButton.getState ( )
```

Returnează starea de joc Gamestate asociată cu acest buton.

Returns

Starea de joc.

6.36.3.5 initBounds()

```
void ui.MenuButton.initBounds ( )  [private]
```

Inițializează limitele dreptunghiulare (hitbox) ale butonului.

Folosește constantele
`B_WIDTH`

și
`B_HEIGHT`

pentru dimensiuni.

6.36.3.6 isMouseOver()

```
boolean ui.MenuButton.isMouseOver ( )
```

Verifică dacă mouse-ul este deasupra butonului.

Returns

`true`
dacă mouse-ul este deasupra,
`false`
altfel.

6.36.3.7 isMousePressed()

```
boolean ui.MenuButton.isMousePressed ( )
```

Verifică dacă butonul este apăsat.

Returns

`true`
dacă butonul este apăsat,
`false`
altfel.

6.36.3.8 loadImgs()

```
void ui.MenuButton.loadImgs ( )  [private]
```

Încarcă imaginile pentru buton din spritesheet-ul global de butoane de meniu.

Folosește
`rowIndex`
pentru a selecta setul corect de imagini.

6.36.3.9 resetBools()

```
void ui.MenuButton.resetBools ( )
```

Resetează flag-urile de interacțiune cu mouse-ul (mouseOver și mousePressed) la valorile implicite (false).

6.36.3.10 setMouseOver()

```
void ui.MenuButton.setMouseOver (
            boolean mouseOver )
```

Setează starea "mouse over" a butonului.

Parameters

mouseOver	Noua stare "mouse over".

6.36.3.11 setMousePressed()

```
void ui.MenuButton.setMousePressed (
            boolean mousePressed )
```

Setează starea de apăsare a butonului.

Parameters

mousePressed	Noua stare de apăsare.

6.36.3.12 update()

```
void ui.MenuButton.update ( )
```

Actualizează starea vizuală a butonului (indexul imaginii) pe baza interacțiunii cu mouse-ul.

6.36.4 Member Data Documentation

6.36.4.1 bounds

```
Rectangle ui.MenuButton.bounds  [private]
```

Dreptunghiul care definește limitele butonului pentru detecția coliziunilor cu mouse-ul.

6.36.4.2 imgs

```
BufferedImage [] ui.MenuButton.imgs  [private]
```

Array de imagini pentru diferitele stări ale butonului (normal, mouse over, apăsat).

6.36.4.3 index

```
int ui.MenuButton.index  [private]
```

Indexul imaginii curente din array-ul.
imgs

, determinat de starea mouse-ului.

6.36.4.4 mouseOver

```
boolean ui.MenuButton.mouseOver  [private]
```

6.36.4.5 rowIndex

```
int ui.MenuButton.rowIndex  [private]
```

Indexul rândului din spritesheet-ul de butoane de unde se încarcă imaginile pentru acest tip de buton.

6.36.4.6 state

```
Gamestate ui.MenuButton.state  [private]
```

Starea de joc Gamestate care va fi activată la apăsarea butonului.

6.36.4.7 xOffsetCenter

```
int ui.MenuButton.xOffsetCenter = B_WIDTH / 2  [private]
```

Offset orizontal pentru centrarea imaginii butonului față de.
xPos

.

6.36.4.8 xPos

```
int ui.MenuButton.xPos  [private]
```

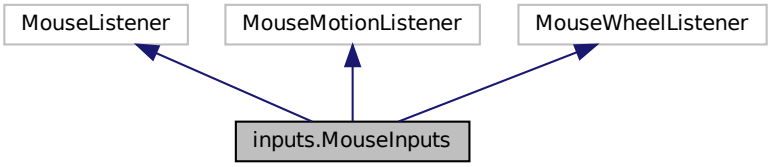
The documentation for this class was generated from the following file:

- src/ui/MenuButton.java

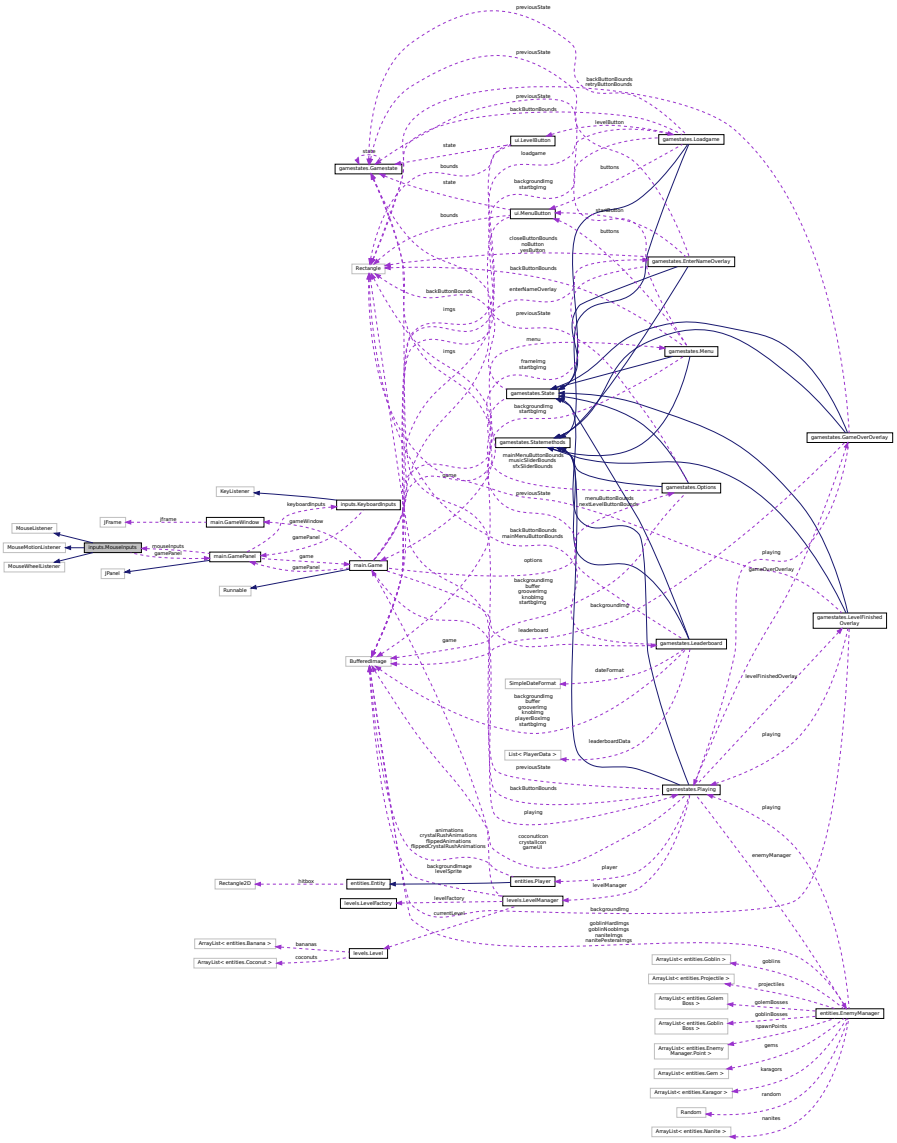
6.37 inputs.MouseInputs Class Reference

Gestionează input-ul de la mouse pentru joc.

Inheritance diagram for inputs.MouseInputs:



Collaboration diagram for inputs.MouseInputs:



Public Member Functions

- MouseInputs (GamePanel gamePanel)

  Constructor pentru MouseInputs.
- void mouseClicked (MouseEvent e)

  Metodă apelată la un click de mouse (apăsare și eliberare).
- void mousePressed (MouseEvent e)

  Metodă apelată la apăsarea unui buton al mouse-ului.
- void mouseReleased (MouseEvent e)

  Metodă apelată la eliberarea unui buton al mouse-ului.
- void mouseEntered (MouseEvent e)

  Metodă apelată când cursorul mouse-ului intră în componentă.
- void mouseExited (MouseEvent e)

  Metodă apelată când cursorul mouse-ului iese din componentă.
- void mouseDragged (MouseEvent e)

  Metodă apelată când mouse-ul este mișcat cu un buton apăsat (drag).
- void mouseMoved (MouseEvent e)

  Metodă apelată când mouse-ul este mișcat (fără butoane apăsate).
- void mouseWheelMoved (MouseWheelEvent e)

  Metodă apelată la mișcarea rotiței mouse-ului.

Private Attributes

- GamePanel gamePanel

6.37.1 Detailed Description

Gestionează input-ul de la mouse pentru joc.

Implementează interfețele MouseListener, MouseMotionListener și MouseWheelListener și redirecționează evenimentele de mouse către starea de joc activă corespunzătoare.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 MouseInputs()

```
inputs.MouseInputs.MouseInputs (
            GamePanel gamePanel )
```

Constructor pentru MouseInputs.

Parameters

gamePanel	Panoul principal al jocului GamePanel căruia i se atașează acest listener.

6.37.3 Member Function Documentation

6.37.3.1 mouseClicked()

```
void inputs.MouseInputs.mouseClicked (
            MouseEvent e )
```

Metodă apelată la un click de mouse (apăsare și eliberare).

Redirecționează evenimentul către metoda
mouseClicked

a stării de joc active.

Parameters

e	Evenimentul MouseEvent.

6.37.3.2 mouseDragged()

```
void inputs.MouseInputs.mouseDragged (
            MouseEvent e )
```

Metodă apelată când mouse-ul este mișcat cu un buton apăsat (drag).

Redirecționează evenimentul către metoda
mouseDragged

a stării de joc active.

Parameters

e	Evenimentul MouseEvent.

6.37.3.3 mouseEntered()

```
void inputs.MouseInputs.mouseEntered (
            MouseEvent e )
```

Metodă apelată când cursorul mouse-ului intră în componentă.

Momentan neutilizată.

Parameters

e	Evenimentul MouseEvent.

6.37.3.4 mouseExited()

```
void inputs.MouseInputs.mouseExited (
            MouseEvent e )
```

Metodă apelată când cursorul mouse-ului iese din componentă.

Momentan neutilizată.

Parameters

e	Evenimentul MouseEvent.

6.37.3.5 mouseMoved()

```
void inputs.MouseInputs.mouseMoved (
            MouseEvent e )
```

Metodă apelată când mouse-ul este mișcat (fără butoane apăsate).

Redirecționează evenimentul către metoda
mouseMoved

a stării de joc active.

Parameters

e	Evenimentul MouseEvent.

6.37.3.6 mousePressed()

```
void inputs.MouseInputs.mousePressed (
            MouseEvent e )
```

Metodă apelată la apăsarea unui buton al mouse-ului.

Redirecționează evenimentul către metoda
mousePressed

a stării de joc active.

Parameters

e	Evenimentul MouseEvent.

6.37.3.7 mouseReleased()

```
void inputs.MouseInputs.mouseReleased (
            MouseEvent e )
```

Metodă apelată la eliberarea unui buton al mouse-ului.

Redirecționează evenimentul către metoda
`mouseReleased`

a stării de joc active.

Parameters

e	Evenimentul MouseEvent.

6.37.3.8 mouseWheelMoved()

```
void inputs.MouseInputs.mouseWheelMoved (
            MouseWheelEvent e )
```

Metodă apelată la mișcarea rotiței mouse-ului.

Redirecționează evenimentul către metoda
`mouseWheelMoved`

a stării de joc active (de ex., pentru derularea clasamentului).

Parameters

e	Evenimentul MouseWheelEvent.

6.37.4 Member Data Documentation

6.37.4.1 gamePanel

```
GamePanel inputs.MouseInputs.gamePanel  [private]
```

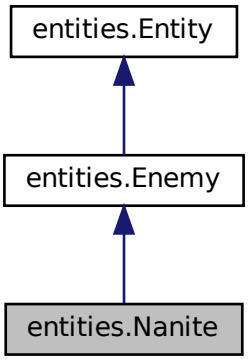
The documentation for this class was generated from the following file:

- src/inputs/MouseInputs.java

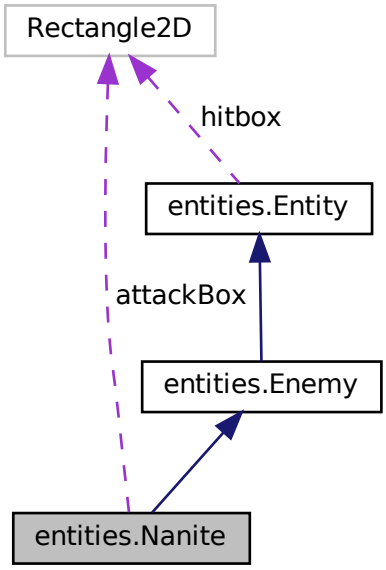
6.38 entities.Nanite Class Reference

Reprezintă entitatea Nanite în joc.

Inheritance diagram for entities.Nanite:



Collaboration diagram for entities.Nanite:



Public Member Functions

- Nanite (float x, float y, int width, int height, int enemyType)

  Constructor pentru clasa Nanite.
- void update (Rectangle2D.Float playerHitbox)

  Actualizează starea Nanite-ului.
- void takeDamage (int damage)

  Aplică daune Nanite-ului atunci când este atacat.
- boolean checkPlayerHit (Rectangle2D.Float playerHitbox)

  Verifică dacă jucătorul atinge Nanite-ul și aplică daune dacă este cazul și nu este în cooldown.
- boolean canAttackPlayer (Rectangle2D.Float playerHitbox)

  Verifică dacă Nanite-ul poate ataca jucătorul (dacă jucătorul este în raza de atac și Nanite-ul nu este în cooldown).
- void playerDetected (float playerX, float playerY)

  Detectează jucătorul dacă se află în raza de vizualizare și ajustează direcția Nanite-ului pentru a se îndrepta către jucător.
- void setLevelData (int[ ][ ] levelData)

  Setează datele nivelului pentru Nanite, folosite pentru coliziuni și navigație.
- boolean isActive ()

  Verifică dacă Nanite-ul este activ în joc.
- int getHealth ()

  Returnează sănătatea curentă a Nanite-ului.
- int getMaxHealth ()

  Returnează sănătatea maximă a Nanite-ului.
- int getDamage ()

  Returnează daunele de atac ale Nanite-ului.
- int getDirection ()

  Returnează direcția curentă a Nanite-ului.
- void makeBoss ()

  Transformă acest Nanite într-un boss (Karagor).
- Rectangle2D.Float getAttackBox ()

  Returnează hitbox-ul de atac al Nanite-ului.

Static Public Attributes

- static final int NANITE_JUNGLA = 0

  Tipul de Nanite "Jungla" (implicit).
- static final int NANITE_PESTERA = 1

  Tipul de Nanite "Pestera" (variantă de peșteră).
- static final int IDLE = 3

  Starea IDLE a Nanite-ului.
- static final int RUNNING = 10

  Starea RUNNING (alergare) a Nanite-ului.
- static final int ATTACK = 12

  Starea ATTACK (atac) a Nanite-ului (corespunde animației SLASHING).
- static final int HURT = 2

  Starea HURT (lovit) a Nanite-ului.
- static final int DYING = 0

  Starea DYING (moarte) a Nanite-ului.

Private Member Functions

- void updateAttackBox ()

  Actualizează poziția și dimensiunea hitbox-ului de atac în funcție de direcția Nanite-ului.
- void updateCooldowns ()

  Actualizează cronometrele de cooldown pentru atac și atingerea jucătorului.
- void initPatrolBoundaries ()

  Inițializează limitele de patrulare pentru Nanite pe baza poziției sale curente.
- boolean canSeePlayer (Rectangle2D.Float playerHitbox)

  Verifică dacă Nanite-ul poate vedea jucătorul.
- void updateBehavior (Rectangle2D.Float playerHitbox)

  Actualizează comportamentul Nanite-ului pe baza stării curente și a interacțiunii cu jucătorul.
- void updatePosition ()

  Actualizează poziția Nanite-ului.
- boolean willLandOnGround (float x, float y)

  Verifică dacă Nanite-ul va ateriza pe o suprafață solidă la poziția specificată.
- void checkAttackHit ()

  Verifică dacă atacul Nanite-ului a lovit jucătorul.
- void setState (int state)

  Setează starea curentă a Nanite-ului și resetează cronometrul pentru starea respectivă.

Private Attributes

- int health

  Sănătatea curentă a Nanite-ului.
- int maxHealth

  Sănătatea maximă a Nanite-ului.
- int damage

  Daunele provocate de atacul Nanite-ului.
- boolean isActive = true

  Indicator dacă Nanite-ul este activ în joc.
- int attackRange = 30

  Distanța de atac a Nanite-ului.
- int ticksInState = 0

  Numărul de tick-uri petrecute în starea curentă.
- int attackCooldown = 0

  Cooldown-ul dintre atacuri.
- boolean attackChecked = false

  Indicator dacă lovitura de atac a fost verificată în animația curentă.
- boolean playerDetected = false

  Indicator dacă jucătorul a fost detectat de Nanite.
- float leftPatrolLimit

  Limita stângă a zonei de patrulare.
- float rightPatrolLimit

  Limita dreaptă a zonei de patrulare.
- boolean patrolBoundariesSet = false

  Indicator dacă limitele de patrulare au fost setate.
- boolean isMoving = false

  Indicator dacă Nanite-ul se mișcă.
- int direction = 1

Direcția de mișcare a Nanite-ului (1 = dreapta, -1 = stânga).

- float moveSpeed = 0.5f

  Viteza de mișcare curentă a Nanite-ului.
- float patrolMoveSpeed

  Viteza de mișcare în timpul patrulării.
- float chaseMoveSpeed

  Viteza de mișcare în timpul urmăririi jucătorului.
- int[ ][ ] levelData

  Datele nivelului curent, folosite pentru coliziuni și navigație.
- boolean inAir = false

  Indicator dacă Nanite-ul se află în aer.
- float airSpeed = 0f

  Viteza verticală a Nanite-ului în aer.
- float gravity = 0.04f

  Valoarea gravitației aplicate Nanite-ului.
- float jumpSpeed = -3.5f ∗ Game.SCALE

  Viteza de săritură a Nanite-ului.
- Rectangle2D.Float attackBox

  Hitbox-ul pentru atacul Nanite-ului.
- int playerTouchCooldown = 0

  Cooldown-ul pentru daunele provocate prin atingerea jucătorului.
- int touchDamageCooldown = 60

  Cooldown-ul dintre daunele provocate prin atingere (în frame-uri).
- float patrolDistance = 100.0f

  Distanța pe care Nanite-ul o patrulează în fiecare direcție de la punctul de start.
- int detectionRange = 300

  Distanța la care Nanite-ul poate detecta jucătorul.

Static Private Attributes

- static final int ATTACK_COOLDOWN_MAX = 120

  Cooldown-ul maxim dintre atacuri (aproximativ 2 secunde la 60 FPS).

Additional Inherited Members

6.38.1 Detailed Description

Reprezintă entitatea Nanite în joc.

Această clasă extinde clasa Enemy și definește comportamentul specific și atributele pentru Nanites, inamici mici și agili. Include tipuri de Nanite (Jungla, Pestera), stări, logică de patrulare, atac și interacțiune cu jucătorul.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 Nanite()

```
entities.Nanite.Nanite (
            float x,
            float y,
            int width,
            int height,
            int enemyType )
```

Constructor pentru clasa Nanite.

Inițializează Nanite-ul cu o poziție, dimensiuni și tip specific (Jungla sau Pestera). Setează atributele în funcție de tip, inițializează parametrii de patrulare, starea inițială și hitbox-ul.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
width	Lățimea entității (de obicei dimensiunea sprite-ului).
height	Înălțimea entității (de obicei dimensiunea sprite-ului).
enemyType	Tipul Nanite-ului (NANITE_JUNGLA sau NANITE_PESTERA).

6.38.3 Member Function Documentation

6.38.3.1 canAttackPlayer()

```
boolean entities.Nanite.canAttackPlayer (
            Rectangle2D.Float playerHitbox )
```

Verifică dacă Nanite-ul poate ataca jucătorul (dacă jucătorul este în raza de atac și Nanite-ul nu este în cooldown).

Dacă da, inițiază starea de atac.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă Nanite-ul poate ataca și a inițiat atacul, false altfel.

6.38.3.2 canSeePlayer()

```
boolean entities.Nanite.canSeePlayer (
            Rectangle2D.Float playerHitbox )  [private]
```

Verifică dacă Nanite-ul poate vedea jucătorul.

Condițiile includ distanța orizontală, alinierea verticală (aproximativ același nivel) și direcția în care privește Nanite-ul.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă jucătorul este vizibil, false altfel.

6.38.3.3 checkAttackHit()

```
void entities.Nanite.checkAttackHit ( )  [private]
```

Verifică dacă atacul Nanite-ului a lovit jucătorul.

Într-un joc real, aici s-ar verifica coliziunea cu hitbox-ul jucătorului și s-ar aplica daune dacă există o coliziune. (Momentan, este un placeholder).

6.38.3.4 checkPlayerHit()

```
boolean entities.Nanite.checkPlayerHit (
            Rectangle2D.Float playerHitbox )
```

Verifică dacă jucătorul atinge Nanite-ul și aplică daune dacă este cazul și nu este în cooldown.

Parameters

playerHitbox	Hitbox-ul jucătorului.

Returns

true dacă jucătorul a lovit Nanite-ul și daunele au fost aplicate (sau ar trebui aplicate), false altfel.

6.38.3.5 getAttackBox()

```
Rectangle2D.Float entities.Nanite.getAttackBox ( )
```

Returnează hitbox-ul de atac al Nanite-ului.

Returns

Dreptunghiul reprezentând hitbox-ul de atac.

6.38.3.6 getDamage()

```
int entities.Nanite.getDamage ( )
```

Returnează daunele de atac ale Nanite-ului.

Returns

Daunele de atac.

6.38.3.7 getDirection()

```
int entities.Nanite.getDirection ( )
```

Returnează direcția curentă a Nanite-ului.

Returns

Direcția (1 pentru dreapta, -1 pentru stânga).

6.38.3.8 getHealth()

```
int entities.Nanite.getHealth ( )
```

Returnează sănătatea curentă a Nanite-ului.

Returns

Sănătatea curentă.

6.38.3.9 getMaxHealth()

```
int entities.Nanite.getMaxHealth ( )
```

Returnează sănătatea maximă a Nanite-ului.

Returns

Sănătatea maximă.

6.38.3.10 initPatrolBoundaries()

```
void entities.Nanite.initPatrolBoundaries ( )  [private]
```

Inițializează limitele de patrulare pentru Nanite pe baza poziției sale curente.

Această metodă ar trebui apelată după ce Nanite-ul s-a așezat pe o platformă. Caută marginile platformei pentru a stabili limitele, asigurând o zonă minimă de patrulare.

6.38.3.11 isActive()

```
boolean entities.Nanite.isActive ( )
```

Verifică dacă Nanite-ul este activ în joc.

Returns

true dacă Nanite-ul este activ, false altfel.

Reimplemented from entities.Enemy.

6.38.3.12 makeBoss()

```
void entities.Nanite.makeBoss ( )
```

Transformă acest Nanite într-un boss (Karagor).

Această metodă pare a fi un placeholder sau o funcționalitate specifică jocului unde un Nanite se poate transforma. Mărește atributele și dimensiunea.

6.38.3.13 playerDetected()

```
void entities.Nanite.playerDetected (
            float playerX,
            float playerY )
```

Detectează jucătorul dacă se află în raza de vizualizare și ajustează direcția Nanite-ului pentru a se îndrepta către jucător.

Trece în modul de urmărire (chase) dacă jucătorul este detectat și Nanite-ul era în IDLE. Revine la modul de patrulare dacă jucătorul iese din raza de detecție extinsă.

Parameters

playerX	Coordonata X a jucătorului.
playerY	Coordonata Y a jucătorului.

6.38.3.14 setLevelData()

```
void entities.Nanite.setLevelData (
            int levelData[][] )
```

Setează datele nivelului pentru Nanite, folosite pentru coliziuni și navigație.

Parameters

levelData	O matrice bidimensională reprezentând tile-urile nivelului.

6.38.3.15 setState()

```
void entities.Nanite.setState (
            int state )  [private]
```

Setează starea curentă a Nanite-ului și resetează cronometrul pentru starea respectivă.

Parameters

state	Noua stare a Nanite-ului (folosind constantele definite în clasă, ex: IDLE, RUNNING).

6.38.3.16 takeDamage()

```
void entities.Nanite.takeDamage (
            int damage )
```

Aplică daune Nanite-ului atunci când este atacat.

Reduce sănătatea și gestionează tranziția la starea HURT sau DYING.

Parameters

damage	Cantitatea de daune primite.

6.38.3.17 update()

```
void entities.Nanite.update (
            Rectangle2D.Float playerHitbox )
```

Actualizează starea Nanite-ului.

Aceasta include actualizarea hitbox-ului de atac, cooldown-urilor, comportamentului și poziției. De asemenea, inițializează limitele de patrulare dacă este necesar.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru interacțiuni.

Reimplemented from entities.Enemy.

6.38.3.18 updateAttackBox()

```
void entities.Nanite.updateAttackBox ( )  [private]
```

Actualizează poziția și dimensiunea hitbox-ului de atac în funcție de direcția Nanite-ului.

6.38.3.19 updateBehavior()

```
void entities.Nanite.updateBehavior (
            Rectangle2D.Float playerHitbox )  [private]
```

Actualizează comportamentul Nanite-ului pe baza stării curente și a interacțiunii cu jucătorul.

Gestionează tranzițiile între stări (IDLE, RUNNING, ATTACK, HURT, DYING) și logica specifică fiecărei stări, inclusiv patrularea și atacul.

Parameters

playerHitbox	Hitbox-ul jucătorului, pentru a lua decizii.

6.38.3.20 updateCooldowns()

```
void entities.Nanite.updateCooldowns ( )  [private]
```

Actualizează cronometrele de cooldown pentru atac și atingerea jucătorului.

Incrementează și contorul de tick-uri în starea curentă.

6.38.3.21 updatePosition()

```
void entities.Nanite.updatePosition ( )  [private]
```

Actualizează poziția Nanite-ului.

Aplică gravitația, gestionează mișcarea verticală și orizontală, verifică coliziunile cu pereții și podeaua, și previne căderea de pe platforme.

6.38.3.22 willLandOnGround()

```
boolean entities.Nanite.willLandOnGround (
            float x,
            float y )  [private]
```

Verifică dacă Nanite-ul va ateriza pe o suprafață solidă la poziția specificată.

Creează un hitbox de test la poziția viitoare și verifică dacă este pe podea.

Parameters

x	Coordonata X a poziției viitoare.
y	Coordonata Y a poziției viitoare.

Returns

true dacă Nanite-ul va ateriza pe sol, false altfel.

6.38.4 Member Data Documentation

6.38.4.1 airSpeed

```
float entities.Nanite.airSpeed = 0f  [private]
```

Viteza verticală a Nanite-ului în aer.

6.38.4.2 ATTACK

```
final int entities.Nanite.ATTACK = 12  [static]
```

Starea ATTACK (atac) a Nanite-ului (corespunde animației SLASHING).

6.38.4.3 ATTACK_COOLDOWN_MAX

```
final int entities.Nanite.ATTACK_COOLDOWN_MAX = 120  [static], [private]
```

Cooldown-ul maxim dintre atacuri (aproximativ 2 secunde la 60 FPS).

6.38.4.4 attackBox

```
Rectangle2D.Float entities.Nanite.attackBox  [private]
```

Hitbox-ul pentru atacul Nanite-ului.

6.38.4.5 attackChecked

```
boolean entities.Nanite.attackChecked = false  [private]
```

Indicator dacă lovitura de atac a fost verificată în animația curentă.

6.38.4.6 attackCooldown

```
int entities.Nanite.attackCooldown = 0  [private]
```

Cooldown-ul dintre atacuri.

6.38.4.7 attackRange

```
int entities.Nanite.attackRange = 30  [private]
```

Distanța de atac a Nanite-ului.

6.38.4.8 chaseMoveSpeed

```
float entities.Nanite.chaseMoveSpeed  [private]
```

Viteza de mișcare în timpul urmăririi jucătorului.

6.38.4.9 damage

```
int entities.Nanite.damage  [private]
```

Daunele provocate de atacul Nanite-ului.

6.38.4.10 detectionRange

```
int entities.Nanite.detectionRange = 300  [private]
```

Distanța la care Nanite-ul poate detecta jucătorul.

6.38.4.11 direction

```
int entities.Nanite.direction = 1  [private]
```

Direcția de mișcare a Nanite-ului (1 = dreapta, -1 = stânga).

6.38.4.12 DYING

```
final int entities.Nanite.DYING = 0  [static]
```

Starea DYING (moarte) a Nanite-ului.

6.38.4.13 gravity

```
float entities.Nanite.gravity = 0.04f  [private]
```

Valoarea gravitației aplicate Nanite-ului.

6.38.4.14 health

```
int entities.Nanite.health  [private]
```

Sănătatea curentă a Nanite-ului.

6.38.4.15 HURT

```
final int entities.Nanite.HURT = 2  [static]
```

Starea HURT (lovit) a Nanite-ului.

6.38.4.16 IDLE

```
final int entities.Nanite.IDLE = 3  [static]
```

Starea IDLE a Nanite-ului.

6.38.4.17 inAir

```
boolean entities.Nanite.inAir = false  [private]
```

Indicator dacă Nanite-ul se află în aer.

6.38.4.18 isActive

```
boolean entities.Nanite.isActive = true  [private]
```

Indicator dacă Nanite-ul este activ în joc.

6.38.4.19 isMoving

```
boolean entities.Nanite.isMoving = false  [private]
```

Indicator dacă Nanite-ul se mișcă.

6.38.4.20 jumpSpeed

```
float entities.Nanite.jumpSpeed = -3.5f * Game.SCALE  [private]
```

Viteza de săritură a Nanite-ului.

6.38.4.21 leftPatrolLimit

```
float entities.Nanite.leftPatrolLimit  [private]
```

Limita stângă a zonei de patrulare.

6.38.4.22 levelData

```
int [][] entities.Nanite.levelData  [private]
```

Datele nivelului curent, folosite pentru coliziuni și navigație.

6.38.4.23 maxHealth

```
int entities.Nanite.maxHealth  [private]
```

Sănătatea maximă a Nanite-ului.

6.38.4.24 moveSpeed

```
float entities.Nanite.moveSpeed = 0.5f  [private]
```

Viteza de mișcare curentă a Nanite-ului.

6.38.4.25 NANITE_JUNGLA

```
final int entities.Nanite.NANITE_JUNGLA = 0  [static]
```

Tipul de Nanite "Jungla" (implicit).

6.38.4.26 NANITE_PESTERA

```
final int entities.Nanite.NANITE_PESTERA = 1  [static]
```

Tipul de Nanite "Pestera" (variantă de peșteră).

6.38.4.27 patrolBoundariesSet

```
boolean entities.Nanite.patrolBoundariesSet = false  [private]
```

Indicator dacă limitele de patrulare au fost setate.

6.38.4.28 patrolDistance

```
float entities.Nanite.patrolDistance = 100.0f  [private]
```

Distanța pe care Nanite-ul o patrulează în fiecare direcție de la punctul de start.

6.38.4.29 patrolMoveSpeed

```
float entities.Nanite.patrolMoveSpeed  [private]
```

Viteza de mișcare în timpul patrulării.

6.38.4.30 playerDetected

```
boolean entities.Nanite.playerDetected = false  [private]
```

Indicator dacă jucătorul a fost detectat de Nanite.

6.38.4.31 playerTouchCooldown

```
int entities.Nanite.playerTouchCooldown = 0  [private]
```

Cooldown-ul pentru daunele provocate prin atingerea jucătorului.

6.38.4.32 rightPatrolLimit

```
float entities.Nanite.rightPatrolLimit  [private]
```

Limita dreaptă a zonei de patrulare.

6.38.4.33 RUNNING

```
final int entities.Nanite.RUNNING = 10  [static]
```

Starea RUNNING (alergare) a Nanite-ului.

6.38.4.34 ticksInState

```
int entities.Nanite.ticksInState = 0  [private]
```

Numărul de tick-uri petrecute în starea curentă.

6.38.4.35 touchDamageCooldown

```
int entities.Nanite.touchDamageCooldown = 60  [private]
```

Cooldown-ul dintre daunele provocate prin atingere (în frame-uri).

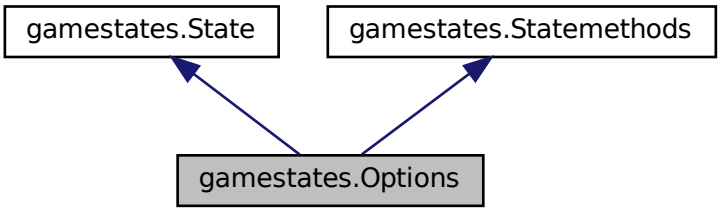
The documentation for this class was generated from the following file:

- src/entities/Nanite.java

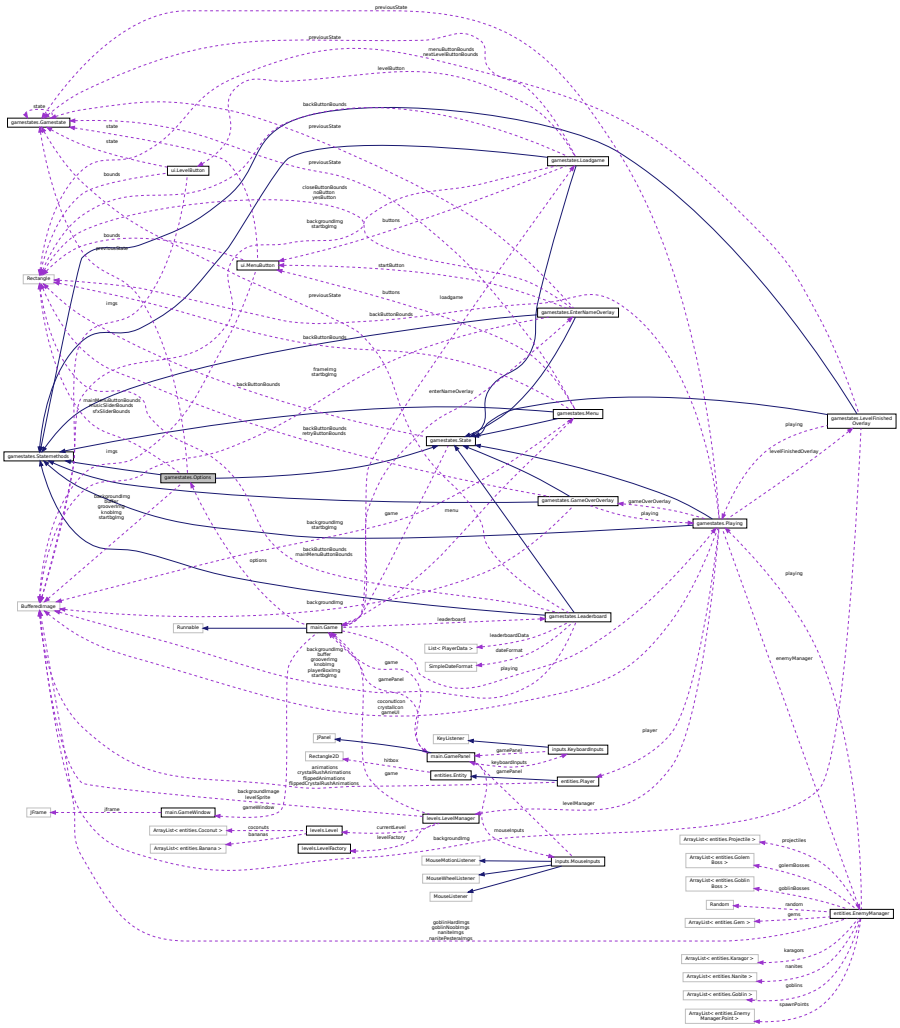
6.39 gamestates.Options Class Reference

Reprezintă starea de joc pentru meniul de opțiuni.

Inheritance diagram for gamestates.Options:



Collaboration diagram for gamestates.Options:



Public Member Functions

- Options (Game game)

  Constructor pentru starea Options.
- void update ()

  Actualizează starea meniului de opțiuni.
- void draw (Graphics g)

  Desenează elementele meniului de opțiuni.
- void mouseClicked (MouseEvent e)

  Gestionarea evenimentului de click al mouse-ului (neutilizat).
- void mousePressed (MouseEvent e)

  Gestionează evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionează evenimentele de eliberare a butonului mouse-ului.
- void mouseMoved (MouseEvent e)

  Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).
- void mouseDragged (MouseEvent e)

Gestionează evenimentele de tragere a mouse-ului (drag).

- int getMusicVolume ()

  Returnează valoarea curentă a volumului muzicii.
- int getSfxVolume ()

  Returnează valoarea curentă a volumului efectelor sonore.
- void keyPressed (KeyEvent e)

  Gestionarea evenimentului de apăsare a tastei (neutilizat).
- void keyReleased (KeyEvent e)

  Gestionarea evenimentului de eliberare a tastei (neutilizat).
- void setPreviousState (Gamestate state)

  Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Private Member Functions

- void loadStartImg ()

  Încarcă și configurează imaginea de fundal generală.
- void loadBackground ()

  Încarcă și configurează imaginea specifică a cadrului pentru meniul de opțiuni.
- void loadSliderImages ()

  Încarcă imaginile pentru componentele slider-elor (buton și șanț).
- void loadCustomFont ()

  Încarcă fontul personalizat folosit pentru textul din meniul de opțiuni.
- void updateMusicValue (int x)

  Actualizează valoarea volumului muzicii pe baza poziției x a mouse-ului.
- void updateSfxValue (int x)

  Actualizează valoarea volumului efectelor sonore pe baza poziției x a mouse-ului.

Private Attributes

- BufferedImage backgroundImg
- int menuX
- BufferedImage startbgImg
- int startbgX
- Rectangle mainMenuButtonBounds
- BufferedImage buffer

  Un buffer grafic pentru a desena conținutul înainte de a-l afișa pe ecran.
- boolean needsRedraw = true

  Flag care indică dacă este necesară redesenarea conținutului buffer-ului.
- Gamestate previousState
- BufferedImage knobImg
- BufferedImage grooverImg
- int musicValue = 50

  Valoarea curentă a volumului muzicii (0-100).
- int sfxValue = 50

  Valoarea curentă a volumului efectelor sonore (0-100).
- Rectangle musicSliderBounds

  Limitele dreptunghiulare pentru slider-ul de muzică.
- Rectangle sfxSliderBounds

  Limitele dreptunghiulare pentru slider-ul de efecte sonore.
- boolean musicSliderDragging = false

  Indică dacă utilizatorul trage de slider-ul de muzică.
- boolean sfxSliderDragging = false

  Indică dacă utilizatorul trage de slider-ul de efecte sonore.
- Font airstrikeFont

Additional Inherited Members

6.39.1 Detailed Description

Reprezintă starea de joc pentru meniul de opțiuni.

Permite jucătorului să ajusteze setări precum volumul muzicii și al efectelor sonore. Extinde State și implementează Statemethods.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 Options()

```
gamestates.Options.Options (
            Game game )
```

Constructor pentru starea Options.

Inițializează elementele UI, inclusiv sliderele de volum.

Parameters

game	Referință la instanța principală a jocului Game.

6.39.3 Member Function Documentation

6.39.3.1 draw()

```
void gamestates.Options.draw (
            Graphics g )
```

Desenează elementele meniului de opțiuni.

Utilizează un buffer pentru a desena elementele doar atunci când este necesar (needsRedraw

). Include fundalul, cadrul, sliderele de volum și valorile acestora.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.39.3.2 getMusicVolume()

```
int gamestates.Options.getMusicVolume ( )
```

Returnează valoarea curentă a volumului muzicii.

Returns

Volumul muzicii (0-100).

6.39.3.3 getSfxVolume()

```
int gamestates.Options.getSfxVolume ( )
```

Returnează valoarea curentă a volumului efectelor sonore.

Returns

Volumul SFX (0-100).

6.39.3.4 keyPressed()

```
void gamestates.Options.keyPressed (
            KeyEvent e )
```

Gestionarea evenimentului de apăsare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.39.3.5 keyReleased()

```
void gamestates.Options.keyReleased (
            KeyEvent e )
```

Gestionarea evenimentului de eliberare a tastei (neutilizat).

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.39.3.6 loadBackground()

```
void gamestates.Options.loadBackground ( )  [private]
```

Încarcă și configurează imaginea specifică a cadrului pentru meniul de opțiuni.

6.39.3.7 loadCustomFont()

```
void gamestates.Options.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat folosit pentru textul din meniul de opțiuni.

6.39.3.8 loadSliderImages()

```
void gamestates.Options.loadSliderImages ( )  [private]
```

Încarcă imaginile pentru componentele slider-elor (buton și șanț).

6.39.3.9 loadStartImg()

```
void gamestates.Options.loadStartImg ( )  [private]
```

Încarcă și configurează imaginea de fundal generală.

6.39.3.10 mouseClicked()

```
void gamestates.Options.mouseClicked (
            MouseEvent e )
```

Gestionarea evenimentului de click al mouse-ului (neutilizat).

Acțiunile sunt gestionate în mousePressed(MouseEvent) și mouseDragged(MouseEvent).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.39.3.11 mouseDragged()

```
void gamestates.Options.mouseDragged (
            MouseEvent e )
```

Gestionază evenimentele de tragere a mouse-ului (drag).

Actualizează valorile slider-elor dacă sunt trase.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.39.3.12 mouseMoved()

```
void gamestates.Options.mouseMoved (
            MouseEvent e )
```

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat).

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.39.3.13 mousePressed()

```
void gamestates.Options.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Verifică dacă s-a apăsat pe un slider de volum sau pe butonul de întoarcere.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.39.3.14 mouseReleased()

```
void gamestates.Options.mouseReleased (
            MouseEvent e )
```

Gestionează evenimentele de eliberare a butonului mouse-ului.

Oprește tragerea slider-elor.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.39.3.15 setPreviousState()

```
void gamestates.Options.setPreviousState (
            Gamestate state )
```

Setează starea anterioară a jocului, pentru a permite revenirea la aceasta.

Parameters

state	Starea anterioară Gamestate.

6.39.3.16 update()

```
void gamestates.Options.update ( )
```

Actualizează starea meniului de opțiuni.

Momentan, nu necesită actualizări logice per frame, deoarece redesenarea este gestionată de flag-ul `needsRedraw`

și interacțiunile cu mouse-ul.

Implements gamestates.Statemethods.

6.39.3.17 updateMusicValue()

```
void gamestates.Options.updateMusicValue (
            int x )  [private]
```

Actualizează valoarea volumului muzicii pe baza poziției x a mouse-ului.

Parameters

x	Poziția x a mouse-ului.

6.39.3.18 updateSfxValue()

```
void gamestates.Options.updateSfxValue (
            int x )  [private]
```

Actualizează valoarea volumului efectelor sonore pe baza poziției x a mouse-ului.

Parameters

x	Poziția x a mouse-ului.

6.39.4 Member Data Documentation

6.39.4.1 airstrikeFont

```
Font gamestates.Options.airstrikeFont  [private]
```

6.39.4.2 backgroundImg

```
BufferedImage gamestates.Options.backgroundImg  [private]
```

6.39.4.3 buffer

```
BufferedImage gamestates.Options.buffer  [private]
```

Un buffer grafic pentru a desena conținutul înainte de a-l afișa pe ecran.

6.39.4.4 grooverImg

```
BufferedImage gamestates.Options.grooverImg  [private]
```

6.39.4.5 knobImg

```
BufferedImage gamestates.Options.knobImg  [private]
```

6.39.4.6 mainMenuButtonBounds

```
Rectangle gamestates.Options.mainMenuButtonBounds  [private]
```

6.39.4.7 menuX

```
int gamestates.Options.menuX  [private]
```

6.39.4.8 musicSliderBounds

```
Rectangle gamestates.Options.musicSliderBounds  [private]
```

Limitele dreptunghiulare pentru slider-ul de muzică.

6.39.4.9 musicSliderDragging

```
boolean gamestates.Options.musicSliderDragging = false  [private]
```

Indică dacă utilizatorul trage de slider-ul de muzică.

6.39.4.10 musicValue

```
int gamestates.Options.musicValue = 50  [private]
```

Valoarea curentă a volumului muzicii (0-100).

6.39.4.11 needsRedraw

```
boolean gamestates.Options.needsRedraw = true  [private]
```

Flag care indică dacă este necesară redesenarea conținutului buffer-ului.

6.39.4.12 previousState

```
Gamestate gamestates.Options.previousState  [private]
```

6.39.4.13 sfxSliderBounds

```
Rectangle gamestates.Options.sfxSliderBounds  [private]
```

Limitele dreptunghiulare pentru slider-ul de efecte sonore.

6.39.4.14 sfxSliderDragging

```
boolean gamestates.Options.sfxSliderDragging = false  [private]
```

Indică dacă utilizatorul trage de slider-ul de efecte sonore.

6.39.4.15 sfxValue

```
int gamestates.Options.sfxValue = 50  [private]
```

Valoarea curentă a volumului efectelor sonore (0-100).

6.39.4.16 startbgImg

```
BufferedImage gamestates.Options.startbgImg  [private]
```

6.39.4.17 startbgX

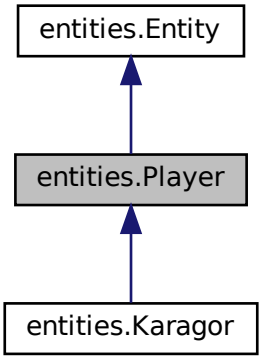
```
int gamestates.Options.startbgX  [private]
```

The documentation for this class was generated from the following file:

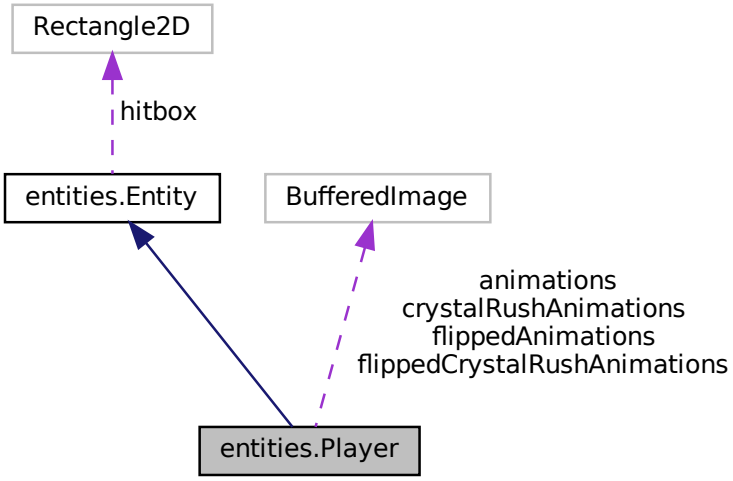
- src/gamestates/Options.java

6.40 entities.Player Class Reference

Inheritance diagram for entities.Player:



Collaboration diagram for entities.Player:



Public Member Functions

- Player (float x, float y, int width, int height)
- void update ()
- void render (Graphics g, int lvlOffsetX)
- void loadAnimations ()
- void loadLevelData (int[][] levelData)
- boolean isLeft ()
- void setLeft (boolean left)
- boolean isRight ()
- void setRight (boolean right)
- void resetDirBooleans ()
- void resetInAir ()
- void setAttack (boolean attack)
- void setThrowAttack (boolean throwAttack)
- void setWhackAttack (boolean whackAttack)
- void setJumpSlamAttack (boolean jumpSlamAttack)
- java.awt.geom.Rectangle2D.Float getAttackHitbox ()
- java.awt.geom.Rectangle2D.Float getWhackHitbox ()
- java.awt.geom.Rectangle2D.Float getJumpSlamHitbox ()
- void setHasHit (boolean hasHit)
- boolean hasHit ()
- int getAttackDamage ()
- void collectBananaEffect ()
- void collectCoconutEffect ()
- boolean isAttack ()
- boolean isCrouch ()
- void setCrouch (boolean crouch)
- boolean isUp ()
- void setUp (boolean up)
- boolean isDown ()
- void setDown (boolean down)
- boolean isJump ()
- void setJump (boolean jump)
- int getCurrentHealth ()
- int getMaxHealth ()
- void heal (int amount)
- boolean takeDamage (int amount)
- boolean isAlive ()
- boolean isDamaged ()
- void applyKnockback (float knockbackX, float knockbackY)
- void resetHealth ()
- void setCurrentHealth (int health)
- void setPosition (float x, float y)
- void resetToStartPosition ()
- boolean isThrowing ()
- boolean isPunching ()
- boolean isWhacking ()
- boolean isJumpSlamming ()
- void setJumpSlamUnlocked (boolean unlocked)
- boolean isCrystalRushUnlocked ()
- void unlockCrystalRush ()
- boolean isCrystalRushActive ()
- void activateCrystalRush ()
- boolean isFacingRight ()
- int getAnimationIndex ()
- int getWhackIndex ()
- boolean canSpawnProjectileAndConsume ()

Protected Member Functions

- void updateKnockback ()

Private Member Functions

- void updateDamageEffect ()
- void updateGravity ()
- void updateAnimationTick ()
- void setAnimation ()
- void resetAnimationTick ()
- void updatePos ()
- void updateXPos (float xSpeed)
- void jump ()

Private Attributes

- BufferedImage[ ][ ] animations
- BufferedImage[ ][ ] flippedAnimations
- BufferedImage[ ][ ] crystalRushAnimations
- BufferedImage[ ][ ] flippedCrystalRushAnimations
- int animationTick
- int playerAction = Gorilla_Animation_rows.PUNCH_STANDING.getRowIndex()
- int playerDirection = -1
- boolean moving = false
- boolean isPunching = false
- boolean isThrowing = false
- boolean isWhacking = false
- boolean isJumpSlamming = false
- boolean hasHit = false
- boolean hasThrown = false
- int attackDamage = 10
- boolean left = false
- boolean right = false
- boolean up = false
- boolean down = false
- boolean facingRight = false
- float playerSpeed = 2.4f
- boolean crouch = false
- boolean wasCrouchPressed = false
- boolean isTransitioning = false
- int levelData [ ][ ]
- boolean jump = false
- int maxHealth = 100
- int currentHealth = 100
- boolean isDamaged = false
- int damageFlashTimer = 0
- final int DAMAGE_FLASH_DURATION = 20
- int permanentMaxHpBonus = 0
- int permanentAttackDamageBonus = 0
- float knockbackX = 0
- float knockbackY = 0
- int knockbackDuration = 0

- final int KNOCKBACK_DURATION = 15
- float xDrawOffset = 66 ∗ Game.SCALE
- float yDrawOffset = 39 ∗ Game.SCALE
- float airSpeed = 0f
- float gravity = 0.12f ∗ Game.SCALE
- float jumpSpeed = -6.5f ∗ Game.SCALE
- float fallSpeedAfterCollision = 2.4f ∗ Game.SCALE
- boolean inAir = false
- boolean isLanding = false
- int landingFrame = 7
- int punchTick = 0
- int throwTick = 0
- int whackTick = 0
- int jumpSlamTick = 0
- boolean jumpSlamUnlocked = false
- final float JUMP_SLAM_DASH_SPEED = 8.0f ∗ Game.SCALE
- int jumpSlamCooldownTimer = 0
- final int JUMP_SLAM_COOLDOWN_DURATION = 1200
- int whackCooldownTimer = 0
- final int WHACK_COOLDOWN_DURATION = 45
- boolean isCrystalRushActive = false
- int crystalRushTimer = 0
- final int CRYSTAL_RUSH_DURATION = 1200
- int crystalRushCooldownTimer = 0
- final int CRYSTAL_RUSH_COOLDOWN_DURATION = 3600
- boolean crystalRushUnlocked = false
- float originalPlayerSpeed

Additional Inherited Members

6.40.1 Constructor & Destructor Documentation

6.40.1.1 Player()

```
entities.Player.Player (
            float x,
            float y,
            int width,
            int height )
```

6.40.2 Member Function Documentation

6.40.2.1 activateCrystalRush()

```
void entities.Player.activateCrystalRush ( )
```

6.40.2.2 applyKnockback()

```
void entities.Player.applyKnockback (
            float knockbackX,
            float knockbackY )
```

6.40.2.3 canSpawnProjectileAndConsume()

```
boolean entities.Player.canSpawnProjectileAndConsume ( )
```

6.40.2.4 collectBananaEffect()

```
void entities.Player.collectBananaEffect ( )
```

6.40.2.5 collectCoconutEffect()

```
void entities.Player.collectCoconutEffect ( )
```

6.40.2.6 getAnimationIndex()

```
int entities.Player.getAnimationIndex ( )
```

6.40.2.7 getAttackDamage()

```
int entities.Player.getAttackDamage ( )
```

Reimplemented in entities.Karagor.

6.40.2.8 getAttackHitbox()

```
java.awt.geom.Rectangle2D.Float entities.Player.getAttackHitbox ( )
```

Reimplemented in entities.Karagor.

6.40.2.9 getCurrentHealth()

```
int entities.Player.getCurrentHealth ( )
```

Reimplemented in entities.Karagor.

6.40.2.10 getJumpSlamHitbox()

```
java.awt.geom.Rectangle2D.Float entities.Player.getJumpSlamHitbox ( )
```

6.40.2.11 getMaxHealth()

```
int entities.Player.getMaxHealth ( )
```

Reimplemented in entities.Karagor.

6.40.2.12 getWhackHitbox()

```
java.awt.geom.Rectangle2D.Float entities.Player.getWhackHitbox ( )
```

6.40.2.13 getWhackIndex()

```
int entities.Player.getWhackIndex ( )
```

6.40.2.14 hasHit()

```
boolean entities.Player.hasHit ( )
```

Reimplemented in entities.Karagor.

6.40.2.15 heal()

```
void entities.Player.heal (
            int amount )
```

Reimplemented in entities.Karagor.

6.40.2.16 isAlive()

```
boolean entities.Player.isAlive ( )
```

Reimplemented in entities.Karagor.

6.40.2.17 isAttack()

```
boolean entities.Player.isAttack ( )
```

Reimplemented in entities.Karagor.

6.40.2.18 isCrouch()

```
boolean entities.Player.isCrouch ( )
```

Reimplemented in entities.Karagor.

6.40.2.19 isCrystalRushActive()

```
boolean entities.Player.isCrystalRushActive ( )
```

6.40.2.20 isCrystalRushUnlocked()

```
boolean entities.Player.isCrystalRushUnlocked ( )
```

6.40.2.21 isDamaged()

```
boolean entities.Player.isDamaged ( )
```

6.40.2.22 isDown()

```
boolean entities.Player.isDown ( )
```

Reimplemented in entities.Karagor.

6.40.2.23 isFacingRight()

```
boolean entities.Player.isFacingRight ( )
```

6.40.2.24 isJump()

```
boolean entities.Player.isJump ( )
```

Reimplemented in entities.Karagor.

6.40.2.25 isJumpSlamming()

```
boolean entities.Player.isJumpSlamming ( )
```

6.40.2.26 isLeft()

```
boolean entities.Player.isLeft ( )
```

Reimplemented in entities.Karagor.

6.40.2.27 isPunching()

```
boolean entities.Player.isPunching ( )
```

6.40.2.28 isRight()

```
boolean entities.Player.isRight ( )
```

Reimplemented in entities.Karagor.

6.40.2.29 isThrowing()

```
boolean entities.Player.isThrowing ( )
```

6.40.2.30 isUp()

```
boolean entities.Player.isUp ( )
```

Reimplemented in entities.Karagor.

6.40.2.31 isWhacking()

```
boolean entities.Player.isWhacking ( )
```

6.40.2.32 jump()

```
void entities.Player.jump ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.33 loadAnimations()

```
void entities.Player.loadAnimations ( )
```

Reimplemented in entities.Karagor.

6.40.2.34 loadLevelData()

```
void entities.Player.loadLevelData (
            int levelData[][] )
```

Reimplemented in entities.Karagor.

6.40.2.35 render()

```
void entities.Player.render (
            Graphics g,
            int lvlOffsetX )
```

Reimplemented in entities.Karagor.

6.40.2.36 resetAnimationTick()

```
void entities.Player.resetAnimationTick ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.37 resetDirBooleans()

```
void entities.Player.resetDirBooleans ( )
```

Reimplemented in entities.Karagor.

6.40.2.38 resetHealth()

```
void entities.Player.resetHealth ( )
```

Reimplemented in entities.Karagor.

6.40.2.39 resetInAir()

```
void entities.Player.resetInAir ( )
```

Reimplemented in entities.Karagor.

6.40.2.40 resetToStartPosition()

```
void entities.Player.resetToStartPosition ( )
```

6.40.2.41 setAnimation()

```
void entities.Player.setAnimation ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.42 setAttack()

```
void entities.Player.setAttack (
            boolean attack )
```

Reimplemented in entities.Karagor.

6.40.2.43 setCrouch()

```
void entities.Player.setCrouch (
            boolean crouch )
```

Reimplemented in entities.Karagor.

6.40.2.44 setCurrentHealth()

```
void entities.Player.setCurrentHealth (
            int health )
```

6.40.2.45 setDown()

```
void entities.Player.setDown (
            boolean down )
```

Reimplemented in entities.Karagor.

6.40.2.46 setHasHit()

```
void entities.Player.setHasHit (
            boolean hasHit )
```

Reimplemented in entities.Karagor.

6.40.2.47 setJump()

```
void entities.Player.setJump (
            boolean jump )
```

Reimplemented in entities.Karagor.

6.40.2.48 setJumpSlamAttack()

```
void entities.Player.setJumpSlamAttack (
            boolean jumpSlamAttack )
```

6.40.2.49 setJumpSlamUnlocked()

```
void entities.Player.setJumpSlamUnlocked (
            boolean unlocked )
```

6.40.2.50 setLeft()

```
void entities.Player.setLeft (
            boolean left )
```

Reimplemented in entities.Karagor.

6.40.2.51 setPosition()

```
void entities.Player.setPosition (
            float x,
            float y )
```

6.40.2.52 setRight()

```
void entities.Player.setRight (
            boolean right )
```

Reimplemented in entities.Karagor.

6.40.2.53 setThrowAttack()

```
void entities.Player.setThrowAttack (
            boolean throwAttack )
```

6.40.2.54 setUp()

```
void entities.Player.setUp (
            boolean up )
```

Reimplemented in entities.Karagor.

6.40.2.55 setWhackAttack()

```
void entities.Player.setWhackAttack (
            boolean whackAttack )
```

6.40.2.56 takeDamage()

```
boolean entities.Player.takeDamage (
            int amount )
```

Reimplemented in entities.Karagor.

6.40.2.57 unlockCrystalRush()

```
void entities.Player.unlockCrystalRush ( )
```

6.40.2.58 update()

```
void entities.Player.update ( )
```

6.40.2.59 updateAnimationTick()

```
void entities.Player.updateAnimationTick ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.60 updateDamageEffect()

```
void entities.Player.updateDamageEffect ( )  [private]
```

6.40.2.61 updateGravity()

```
void entities.Player.updateGravity ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.62 updateKnockback()

```
void entities.Player.updateKnockback ( )  [protected]
```

6.40.2.63 updatePos()

```
void entities.Player.updatePos ( )  [private]
```

Reimplemented in entities.Karagor.

6.40.2.64 updateXPos()

```
void entities.Player.updateXPos (
            float xSpeed )  [private]
```

Reimplemented in entities.Karagor.

6.40.3 Member Data Documentation

6.40.3.1 airSpeed

```
float entities.Player.airSpeed = 0f  [private]
```

6.40.3.2 animations

```
BufferedImage [][] entities.Player.animations  [private]
```

6.40.3.3 animationTick

```
int entities.Player.animationTick  [private]
```

6.40.3.4 attackDamage

```
int entities.Player.attackDamage = 10  [private]
```

6.40.3.5 crouch

```
boolean entities.Player.crouch = false  [private]
```

6.40.3.6 CRYSTAL_RUSH_COOLDOWN_DURATION

```
final int entities.Player.CRYSTAL_RUSH_COOLDOWN_DURATION = 3600  [private]
```

6.40.3.7 CRYSTAL_RUSH_DURATION

```
final int entities.Player.CRYSTAL_RUSH_DURATION = 1200  [private]
```

6.40.3.8 crystalRushAnimations

```
BufferedImage [][] entities.Player.crystalRushAnimations  [private]
```

6.40.3.9 crystalRushCooldownTimer

```
int entities.Player.crystalRushCooldownTimer = 0  [private]
```

6.40.3.10 crystalRushTimer

```
int entities.Player.crystalRushTimer = 0  [private]
```

6.40.3.11 crystalRushUnlocked

```
boolean entities.Player.crystalRushUnlocked = false  [private]
```

6.40.3.12 currentHealth

```
int entities.Player.currentHealth = 100  [private]
```

6.40.3.13 DAMAGE_FLASH_DURATION

```
final int entities.Player.DAMAGE_FLASH_DURATION = 20  [private]
```

6.40.3.14 damageFlashTimer

```
int entities.Player.damageFlashTimer = 0  [private]
```

6.40.3.15 down

```
boolean entities.Player.down = false  [private]
```

6.40.3.16 facingRight

```
boolean entities.Player.facingRight = false  [private]
```

6.40.3.17 fallSpeedAfterCollision

```
float entities.Player.fallSpeedAfterCollision = 2.4f * Game.SCALE  [private]
```

6.40.3.18 flippedAnimations

```
BufferedImage [][] entities.Player.flippedAnimations  [private]
```

6.40.3.19 flippedCrystalRushAnimations

```
BufferedImage [][] entities.Player.flippedCrystalRushAnimations  [private]
```

6.40.3.20 gravity

```
float entities.Player.gravity = 0.12f * Game.SCALE  [private]
```

6.40.3.21 hasHit

```
boolean entities.Player.hasHit = false  [private]
```

Reimplemented in entities.Karagor.

6.40.3.22 hasThrown

```
boolean entities.Player.hasThrown = false  [private]
```

6.40.3.23 inAir

```
boolean entities.Player.inAir = false  [private]
```

6.40.3.24 isCrystalRushActive

```
boolean entities.Player.isCrystalRushActive = false  [private]
```

6.40.3.25 isDamaged

```
boolean entities.Player.isDamaged = false  [private]
```

6.40.3.26 isJumpSlamming

```
boolean entities.Player.isJumpSlamming = false  [private]
```

6.40.3.27 isLanding

```
boolean entities.Player.isLanding = false  [private]
```

6.40.3.28 isPunching

```
boolean entities.Player.isPunching = false  [private]
```

6.40.3.29 isThrowing

```
boolean entities.Player.isThrowing = false  [private]
```

6.40.3.30 isTransitioning

```
boolean entities.Player.isTransitioning = false  [private]
```

6.40.3.31 isWhacking

```
boolean entities.Player.isWhacking = false  [private]
```

6.40.3.32 jump

```
boolean entities.Player.jump = false  [private]
```

Reimplemented in entities.Karagor.

6.40.3.33 JUMP_SLAM_COOLDOWN_DURATION

```
final int entities.Player.JUMP_SLAM_COOLDOWN_DURATION = 1200  [private]
```

6.40.3.34 JUMP_SLAM_DASH_SPEED

```
final float entities.Player.JUMP_SLAM_DASH_SPEED = 8.0f * Game.SCALE  [private]
```

6.40.3.35 jumpSlamCooldownTimer

```
int entities.Player.jumpSlamCooldownTimer = 0  [private]
```

6.40.3.36 jumpSlamTick

```
int entities.Player.jumpSlamTick = 0  [private]
```

6.40.3.37 jumpSlamUnlocked

```
boolean entities.Player.jumpSlamUnlocked = false  [private]
```

6.40.3.38 jumpSpeed

```
float entities.Player.jumpSpeed = -6.5f * Game.SCALE  [private]
```

6.40.3.39 KNOCKBACK_DURATION

```
final int entities.Player.KNOCKBACK_DURATION = 15  [private]
```

6.40.3.40 knockbackDuration

```
int entities.Player.knockbackDuration = 0  [private]
```

6.40.3.41 knockbackX

```
float entities.Player.knockbackX = 0  [private]
```

6.40.3.42 knockbackY

```
float entities.Player.knockbackY = 0  [private]
```

6.40.3.43 landingFrame

```
int entities.Player.landingFrame = 7  [private]
```

6.40.3.44 left

```
boolean entities.Player.left = false  [private]
```

6.40.3.45 levelData

```
int entities.Player.levelData[][]  [private]
```

6.40.3.46 maxHealth

```
int entities.Player.maxHealth = 100  [private]
```

6.40.3.47 moving

```
boolean entities.Player.moving = false  [private]
```

6.40.3.48 originalPlayerSpeed

```
float entities.Player.originalPlayerSpeed  [private]
```

6.40.3.49 permanentAttackDamageBonus

```
int entities.Player.permanentAttackDamageBonus = 0  [private]
```

6.40.3.50 permanentMaxHpBonus

```
int entities.Player.permanentMaxHpBonus = 0  [private]
```

6.40.3.51 playerAction

```
int entities.Player.playerAction = Gorilla_Animation_rows.PUNCH_STANDING.getRowIndex()  [private]
```

6.40.3.52 playerDirection

```
int entities.Player.playerDirection = -1  [private]
```

6.40.3.53 playerSpeed

```
float entities.Player.playerSpeed = 2.4f  [private]
```

6.40.3.54 punchTick

```
int entities.Player.punchTick = 0  [private]
```

6.40.3.55 right

```
boolean entities.Player.right = false  [private]
```

6.40.3.56 throwTick

```
int entities.Player.throwTick = 0  [private]
```

6.40.3.57 up

```
boolean entities.Player.up = false  [private]
```

6.40.3.58 wasCrouchPressed

```
boolean entities.Player.wasCrouchPressed = false  [private]
```

6.40.3.59 WHACK_COOLDOWN_DURATION

```
final int entities.Player.WHACK_COOLDOWN_DURATION = 45  [private]
```

6.40.3.60 whackCooldownTimer

```
int entities.Player.whackCooldownTimer = 0  [private]
```

6.40.3.61 whackTick

```
int entities.Player.whackTick = 0  [private]
```

6.40.3.62 xDrawOffset

```
float entities.Player.xDrawOffset = 66 * Game.SCALE  [private]
```

6.40.3.63 yDrawOffset

```
float entities.Player.yDrawOffset = 39 * Game.SCALE  [private]
```

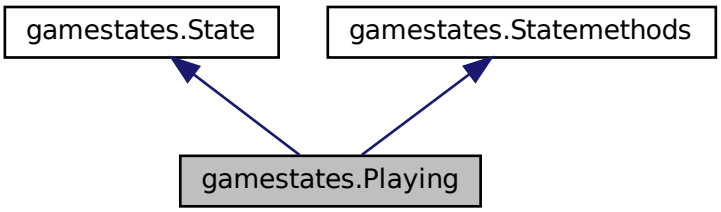
The documentation for this class was generated from the following file:

- src/entities/Player.java

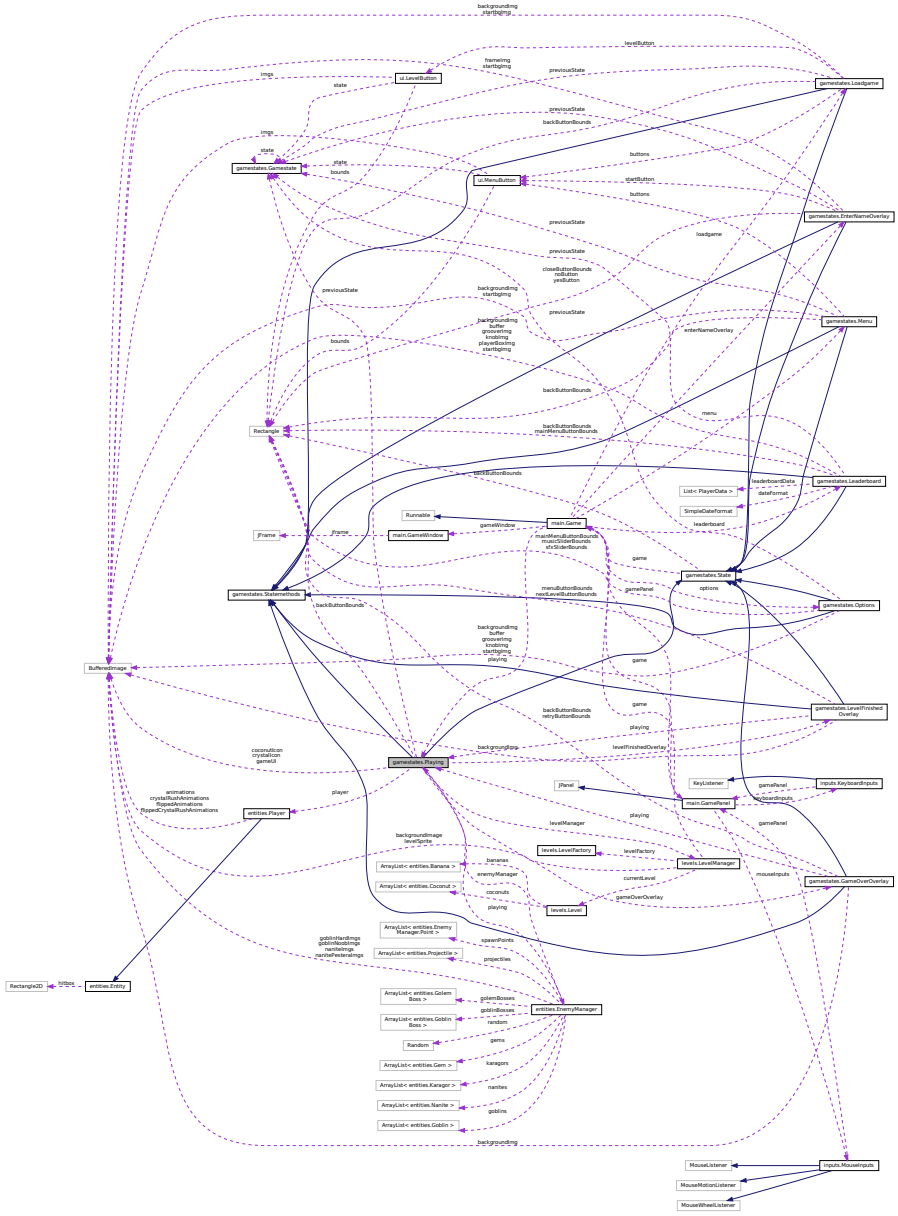
6.41 gamestates.Playing Class Reference

Reprezintă starea principală de joc (gameplay-ul propriu-zis).

Inheritance diagram for gamestates.Playing:



Collaboration diagram for gamestates.Playing:



Public Member Functions

- Playing (Game game)

  Constructor pentru starea Playing.
- void update ()

  Actualizează logica stării de joc.
- void advanceToNextLevel ()

  Avansează la următorul nivel.
- void draw (Graphics g)

  Desenează toate elementele stării de joc, inclusiv nivelul, inamicii, jucătorul și HUD-ul.
- void mouseClicked (MouseEvent e)

  Gestionează evenimentele de click al mouse-ului.

- void keyPressed (KeyEvent e)

  Gestionează evenimentele de apăsare a tastelor.
- void keyReleased (KeyEvent e)

  Gestionează evenimentele de eliberare a tastelor.
- void mousePressed (MouseEvent e)

  Gestionază evenimentele de apăsare a butonului mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat aici).
- void mouseMoved (MouseEvent e)

  Gestionarea evenimentului de mișcare a mouse-ului (neutilizat aici).
- void mouseDragged (MouseEvent e)

  Gestionarea evenimentului de tragere a mouse-ului (neutilizat aici).
- void windowFocusLost ()

  Apelată când fereastra jocului pierde focusul, pentru a reseta input-urile jucătorului.
- Player getPlayer ()
- EnemyManager getEnemyManager ()
- void resetAll ()

  Resetează starea jocului (parțial, păstrând progresul).
- void resetAll (boolean fullReset)

  Resetează starea jocului.
- void setPlayerName (String name)

  Setează numele jucătorului (informativ, username este folosit pentru salvare/încărcare).
- String getPlayerName ()
- void setPreviousState (Gamestate state)

  Setează starea anterioară a jocului.
- void setUsername (String username)

  Setează numele de utilizator curent al sesiunii.
- void showLevelFinishedOverlay ()

  Afișează overlay-ul de final de nivel.
- void setLevelFinished (boolean finished)

  Setează starea de finalizare a nivelului.
- String getUsername ()
- LevelManager getLevelManager ()
- void showGameOverOverlay ()

  Afișează overlay-ul de Game Over și salvează progresul.
- void setGameOver (boolean over)

  Setează starea de Game Over.
- void setCurrentScore (int score)

  Setează scorul curent.
- void setCurrentCoconuts (int coconuts)

  Setează numărul curent de nuci de cocos.
- void setTimer (int timer)

  Setează valoarea cronometrului (timpul scurs).
- int getElapsedSeconds ()
- int getCurrentScore ()
- int getCurrentCoconuts ()

Public Attributes

- int currentCoconuts = 0

  Numărul curent de nuci de cocos deținute de jucător.

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" din HUD (dacă showDebugHitbox este true).

Private Member Functions

- void initClasses ()

  Inițializează clasele principale necesare pentru starea de joc: LevelManager, Player, EnemyManager.
- void checkPlayerAttackHits ()

  Verifică dacă atacurile jucătorului lovesc inamici și aplică daune.
- void checkGemCollision ()

  Verifică coliziunea jucătorului cu gem-urile (pietre prețioase).
- void updateBananas ()

  Actualizează starea bananelor active din nivel.
- void checkBananaCollision ()

  Verifică coliziunea jucătorului cu bananele și aplică efectul.
- void updateCoconuts ()

  Actualizează starea nucilor de cocos active din nivel.
- void checkCoconutCollision ()

  Verifică coliziunea jucătorului cu nucile de cocos și actualizează contorul.
- void checkPlayerThrow ()

  Verifică dacă jucătorul a inițiat o aruncare și generează un proiectil (nucă de cocos).
- void checkCloseToBorder ()

  Verifică dacă jucătorul este aproape de marginile ecranului și ajustează offset-ul de scrolling al nivelului (.
- void drawBananas (Graphics g, int xLvlOffset)

  Desenează bananele active din nivel.
- void drawCoconuts (Graphics g, int xLvlOffset)

  Desenează nucile de cocos active din nivel.
- void drawHUD (Graphics g)

  Desenează elementele HUD (Head-Up Display).
- void loadCustomFont ()

  Încarcă fontul personalizat.

Private Attributes

- Player player
- LevelManager levelManager
- EnemyManager enemyManager
- boolean paused = false
- String playerName = ""
- String username = ""
- int xLvlOffset = 0

  Offset-ul orizontal al nivelului pentru scrolling.
- int leftBorder = (int) (0.2 ∗ Game.GAME_WIDTH)

  Limita stângă a ecranului pentru inițierea scrolling-ului.
- int rightBorder = (int) (0.8 ∗ Game.GAME_WIDTH)

  Limita dreaptă a ecranului pentru inițierea scrolling-ului.
- int lvlTilesWide

  Lățimea totală a nivelului curent, în tile-uri.

- int maxTilesOffset

  Offset-ul maxim posibil al tile-urilor pentru scrolling.
- int maxLvlOffsetX

  Offset-ul maxim posibil al nivelului în pixeli pentru scrolling.
- BufferedImage gameUI
- BufferedImage coconutIcon
- BufferedImage crystalIcon
- long levelStartTime

  Timpul de start al nivelului curent, în milisecunde.
- int elapsedSeconds

  Numărul de secunde scurs de la începerea nivelului.
- Font airstrikeFont
- boolean timerStarted = false

  Indică dacă cronometrul nivelului a pornit.
- Rectangle backButtonBounds
- Gamestate previousState = null
- LevelFinishedOverlay levelFinishedOverlay
- boolean levelFinished = false

  Indică dacă nivelul curent a fost finalizat.
- boolean gameOver = false

  Indică dacă jocul s-a terminat (Game Over).
- GameOverOverlay gameOverOverlay
- int currentLevel = 1

  Nivelul curent la care se joacă (index).
- int currentScore = 0

  Scorul curent al jucătorului în nivel.

Additional Inherited Members

6.41.1 Detailed Description

Reprezintă starea principală de joc (gameplay-ul propriu-zis).

Gestionează jucătorul, nivelurile, inamicii, interfața utilizator (HUD), coliziunile, colectabilele și logica generală a jocului în desfășurare. Extinde State și implementează Statemethods.

6.41.2 Constructor & Destructor Documentation

6.41.2.1 Playing()

```
gamestates.Playing.Playing (
            Game game )
```

Constructor pentru starea Playing.

Inițializează managerii, jucătorul și încarcă datele inițiale.

Parameters

game	Referință la instanța principală a jocului Game.

6.41.3 Member Function Documentation

6.41.3.1 advanceToNextLevel()

```
void gamestates.Playing.advanceToNextLevel ( )
```

Avansează la următorul nivel.

Salvează progresul nivelului curent, încarcă datele pentru noul nivel și resetează starea jucătorului și a jocului.

6.41.3.2 checkBananaCollision()

```
void gamestates.Playing.checkBananaCollision ( )  [private]
```

Verifică coliziunea jucătorului cu bananele și aplică efectul.

6.41.3.3 checkCloseToBorder()

```
void gamestates.Playing.checkCloseToBorder ( )  [private]
```

Verifică dacă jucătorul este aproape de marginile ecranului și ajustează offset-ul de scrolling al nivelului (. xLvlOffset

).

6.41.3.4 checkCoconutCollision()

```
void gamestates.Playing.checkCoconutCollision ( )  [private]
```

Verifică coliziunea jucătorului cu nucile de cocos și actualizează contorul.

6.41.3.5 checkGemCollision()

```
void gamestates.Playing.checkGemCollision ( )  [private]
```

Verifică coliziunea jucătorului cu gem-urile (pietre prețioase).

La coliziune, activează overlay-ul de final de nivel.

6.41.3.6 checkPlayerAttackHits()

```
void gamestates.Playing.checkPlayerAttackHits ( )  [private]
```

Verifică dacă atacurile jucătorului lovesc inamici și aplică daune.

Asigură că un atac lovește o singură dată.

6.41.3.7 checkPlayerThrow()

```
void gamestates.Playing.checkPlayerThrow ( )  [private]
```

Verifică dacă jucătorul a inițiat o aruncare și generează un proiectil (nucă de cocos).

6.41.3.8 draw()

```
void gamestates.Playing.draw (
            Graphics g )
```

Desenează toate elementele stării de joc, inclusiv nivelul, inamicii, jucătorul și HUD-ul.

De asemenea, desenează overlay-urile de final de nivel sau game over, dacă sunt active.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implements gamestates.Statemethods.

6.41.3.9 drawBackButton()

```
void gamestates.Playing.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" din HUD (dacă showDebugHitbox este true).

Reimplemented from gamestates.State.

6.41.3.10 drawBananas()

```
void gamestates.Playing.drawBananas (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează bananele active din nivel.

6.41.3.11 drawCoconuts()

```
void gamestates.Playing.drawCoconuts (
            Graphics g,
            int xLvlOffset )  [private]
```

Desenează nucile de cocos active din nivel.

6.41.3.12 drawHUD()

```
void gamestates.Playing.drawHUD (
            Graphics g )  [private]
```

Desenează elementele HUD (Head-Up Display).

6.41.3.13 getCurrentCoconuts()

```
int gamestates.Playing.getCurrentCoconuts ( )
```

Returns

Numărul curent de nuci de cocos.

6.41.3.14 getCurrentScore()

```
int gamestates.Playing.getCurrentScore ( )
```

Returns

Scorul curent.

6.41.3.15 getElapsedSeconds()

```
int gamestates.Playing.getElapsedSeconds ( )
```

Returns

Timpul scurs în secunde pentru nivelul curent.

6.41.3.16 getEnemyManager()

```
EnemyManager gamestates.Playing.getEnemyManager ( )
```

Returns

Managerul de inamici.

6.41.3.17 getLevelManager()

```
LevelManager gamestates.Playing.getLevelManager ( )
```

Returns

Managerul de niveluri.

6.41.3.18 getPlayer()

```
Player gamestates.Playing.getPlayer ( )
```

Returns

Instanța jucătorului.

6.41.3.19 getPlayerName()

```
String gamestates.Playing.getPlayerName ( )
```

Returns

Numele informativ al jucătorului.

6.41.3.20 getUsername()

```
String gamestates.Playing.getUsername ( )
```

Returns

Numele de utilizator curent.

6.41.3.21 initClasses()

```
void gamestates.Playing.initClasses ( )  [private]
```

Inițializează clasele principale necesare pentru starea de joc: LevelManager, Player, EnemyManager.

Încarcă datele nivelului și inamicii. Configurează variabilele legate de dimensiunea nivelului și scrolling. Încarcă elementele UI și fonturile.

6.41.3.22 keyPressed()

```
void gamestates.Playing.keyPressed (
            KeyEvent e )
```

Gestionează evenimentele de apăsare a tastelor.

Controlează mișcarea jucătorului, săritura, ghemuirea, atacurile speciale și pauza.

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.41.3.23 keyReleased()

```
void gamestates.Playing.keyReleased (
            KeyEvent e )
```

Gestionează evenimentele de eliberare a tastelor.

Oprește mișcarea jucătorului sau ghemuirea.

Parameters

e	Evenimentul KeyEvent.

Implements gamestates.Statemethods.

6.41.3.24 loadCustomFont()

```
void gamestates.Playing.loadCustomFont ( )  [private]
```

Încarcă fontul personalizat.

6.41.3.25 mouseClicked()

```
void gamestates.Playing.mouseClicked (
            MouseEvent e )
```

Gestionează evenimentele de click al mouse-ului.

Inițiază cronometrul nivelului la primul click. Activează atacurile jucătorului (pumn sau whack) la click stânga/dreapta.

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.41.3.26 mouseDragged()

```
void gamestates.Playing.mouseDragged (
            MouseEvent e )
```

Gestionarea evenimentului de tragere a mouse-ului (neutilizat aici).

Implements gamestates.Statemethods.

6.41.3.27 mouseMoved()

```
void gamestates.Playing.mouseMoved (
            MouseEvent e )
```

Gestionarea evenimentului de mișcare a mouse-ului (neutilizat aici).

Implements gamestates.Statemethods.

6.41.3.28 mousePressed()

```
void gamestates.Playing.mousePressed (
            MouseEvent e )
```

Gestionază evenimentele de apăsare a butonului mouse-ului.

Specific pentru butonul "Înapoi" din HUD, dacă este cazul. Salvează progresul la apăsarea butonului "Înapoi".

Parameters

e	Evenimentul MouseEvent.

Implements gamestates.Statemethods.

6.41.3.29 mouseReleased()

```
void gamestates.Playing.mouseReleased (
            MouseEvent e )
```

Gestionarea evenimentului de eliberare a butonului mouse-ului (neutilizat aici).

Implements gamestates.Statemethods.

6.41.3.30 resetAll() [1/2]

```
void gamestates.Playing.resetAll ( )
```

Resetează starea jocului (parțial, păstrând progresul).

6.41.3.31 resetAll() [2/2]

```
void gamestates.Playing.resetAll (
            boolean fullReset )
```

Resetează starea jocului.

Parameters

fullReset	True pentru o resetare completă (la reîncercare nivel), false pentru o resetare parțială (păstrând scorul, etc. la încărcare).

6.41.3.32 setCurrentCoconuts()

```
void gamestates.Playing.setCurrentCoconuts (
            int coconuts )
```

Setează numărul curent de nuci de cocos.

Parameters

coconuts	Noul număr de nuci.

6.41.3.33 setCurrentScore()

```
void gamestates.Playing.setCurrentScore (
            int score )
```

Setează scorul curent.

Parameters

score	Noul scor.

6.41.3.34 setGameOver()

```
void gamestates.Playing.setGameOver (
            boolean over )
```

Setează starea de Game Over.

6.41.3.35 setLevelFinished()

```
void gamestates.Playing.setLevelFinished (
            boolean finished )
```

Setează starea de finalizare a nivelului.

6.41.3.36 setPlayerName()

```
void gamestates.Playing.setPlayerName (
            String name )
```

Setează numele jucătorului (informativ, username este folosit pentru salvare/încărcare).

6.41.3.37 setPreviousState()

```
void gamestates.Playing.setPreviousState (
            Gamestate state )
```

Setează starea anterioară a jocului.

6.41.3.38 setTimer()

```
void gamestates.Playing.setTimer (
            int timer )
```

Setează valoarea cronometrului (timpul scurs).

Parameters

timer	Timpul în secunde.

6.41.3.39 setUsername()

```
void gamestates.Playing.setUsername (
            String username )
```

Setează numele de utilizator curent al sesiunii.

6.41.3.40 showGameOverOverlay()

```
void gamestates.Playing.showGameOverOverlay ( )
```

Afișează overlay-ul de Game Over și salvează progresul.

6.41.3.41 showLevelFinishedOverlay()

```
void gamestates.Playing.showLevelFinishedOverlay ( )
```

Afișează overlay-ul de final de nivel.

6.41.3.42 update()

```
void gamestates.Playing.update ( )
```

Actualizează logica stării de joc.

Include actualizarea nivelului, jucătorului, inamicilor, colectabilelor, verificarea coliziunilor și a condițiilor de final de joc. Rulează doar dacă jocul nu este în pauză.

Implements gamestates.Statemethods.

6.41.3.43 updateBananas()

```
void gamestates.Playing.updateBananas ( )  [private]
```

Actualizează starea bananelor active din nivel.

6.41.3.44 updateCoconuts()

```
void gamestates.Playing.updateCoconuts ( )  [private]
```

Actualizează starea nucilor de cocos active din nivel.

6.41.3.45 windowFocusLost()

```
void gamestates.Playing.windowFocusLost ( )
```

Apelată când fereastra jocului pierde focusul, pentru a reseta input-urile jucătorului.

6.41.4 Member Data Documentation

6.41.4.1 airstrikeFont

```
Font gamestates.Playing.airstrikeFont  [private]
```

6.41.4.2 backButtonBounds

```
Rectangle gamestates.Playing.backButtonBounds  [private]
```

6.41.4.3 coconutIcon

```
BufferedImage gamestates.Playing.coconutIcon  [private]
```

6.41.4.4 crystalIcon

```
BufferedImage gamestates.Playing.crystalIcon  [private]
```

6.41.4.5 currentCoconuts

```
int gamestates.Playing.currentCoconuts = 0
```

Numărul curent de nuci de cocos deținute de jucător.

6.41.4.6 currentLevel

```
int gamestates.Playing.currentLevel = 1  [private]
```

Nivelul curent la care se joacă (index).

6.41.4.7 currentScore

```
int gamestates.Playing.currentScore = 0  [private]
```

Scorul curent al jucătorului în nivel.

6.41.4.8 elapsedSeconds

```
int gamestates.Playing.elapsedSeconds  [private]
```

Numărul de secunde scurs de la începerea nivelului.

6.41.4.9 enemyManager

```
EnemyManager gamestates.Playing.enemyManager  [private]
```

6.41.4.10 gameOver

```
boolean gamestates.Playing.gameOver = false  [private]
```

Indică dacă jocul s-a terminat (Game Over).

6.41.4.11 gameOverOverlay

```
GameOverOverlay gamestates.Playing.gameOverOverlay  [private]
```

6.41.4.12 gameUI

```
BufferedImage gamestates.Playing.gameUI  [private]
```

6.41.4.13 leftBorder

```
int gamestates.Playing.leftBorder = (int) (0.2 * Game.GAME_WIDTH)  [private]
```

Limita stângă a ecranului pentru inițierea scrolling-ului.

6.41.4.14 levelFinished

```
boolean gamestates.Playing.levelFinished = false  [private]
```

Indică dacă nivelul curent a fost finalizat.

6.41.4.15 levelFinishedOverlay

```
LevelFinishedOverlay gamestates.Playing.levelFinishedOverlay  [private]
```

6.41.4.16 levelManager

```
LevelManager gamestates.Playing.levelManager  [private]
```

6.41.4.17 levelStartTime

```
long gamestates.Playing.levelStartTime  [private]
```

Timpul de start al nivelului curent, în milisecunde.

6.41.4.18 lvlTilesWide

```
int gamestates.Playing.lvlTilesWide  [private]
```

Lățimea totală a nivelului curent, în tile-uri.

6.41.4.19 maxLvlOffsetX

```
int gamestates.Playing.maxLvlOffsetX  [private]
```

Offset-ul maxim posibil al nivelului în pixeli pentru scrolling.

6.41.4.20 maxTilesOffset

```
int gamestates.Playing.maxTilesOffset  [private]
```

Offset-ul maxim posibil al tile-urilor pentru scrolling.

6.41.4.21 paused

```
boolean gamestates.Playing.paused = false  [private]
```

6.41.4.22 player

```
Player gamestates.Playing.player  [private]
```

6.41.4.23 playerName

```
String gamestates.Playing.playerName = ""  [private]
```

6.41.4.24 previousState

```
Gamestate gamestates.Playing.previousState = null  [private]
```

6.41.4.25 rightBorder

```
int gamestates.Playing.rightBorder = (int) (0.8 * Game.GAME_WIDTH)  [private]
```

Limita dreaptă a ecranului pentru inițierea scrolling-ului.

6.41.4.26 timerStarted

```
boolean gamestates.Playing.timerStarted = false  [private]
```

Indică dacă cronometrul nivelului a pornit.

6.41.4.27 username

```
String gamestates.Playing.username = ""  [private]
```

6.41.4.28 xLvlOffset

```
int gamestates.Playing.xLvlOffset = 0  [private]
```

Offset-ul orizontal al nivelului pentru scrolling.

The documentation for this class was generated from the following file:

- src/gamestates/Playing.java

6.42 entities.EnemyManager.Point Class Reference

Clasă internă simplă pentru a stoca informații despre un punct de spawn: coordonatele (x, y) și codul numeric al inamicului care trebuie spawnat.

Public Member Functions

- Point (float x, float y, int enemyCode)

  Constructor pentru clasa Point.

6.42.1 Detailed Description

Clasă internă simplă pentru a stoca informații despre un punct de spawn: coordonatele (x, y) și codul numeric al inamicului care trebuie spawnat.

6.42.2 Constructor & Destructor Documentation

6.42.2.1 Point()

```
entities.EnemyManager.Point.Point (
            float x,
            float y,
            int enemyCode )
```

Constructor pentru clasa Point.

Parameters

x	Coordonata X a punctului de spawn.
y	Coordonata Y a punctului de spawn.
enemyCode	Codul inamicului.

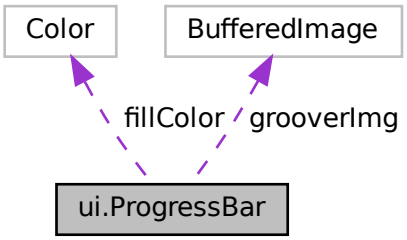
The documentation for this class was generated from the following file:

- src/entities/EnemyManager.java

6.43 ui.ProgressBar Class Reference

Reprezintă o bară de progres personalizată pentru interfața utilizator.

Collaboration diagram for ui.ProgressBar:



Public Member Functions

- ProgressBar (int x, int y, int width, int height)

  Constructor pentru ProgressBar.
- void draw (Graphics g)

  Desenează bara de progres pe ecran.
- void setProgress (float progress)

  Setează progresul curent al barei.
- float getProgress ()

  Returnează progresul curent al barei.
- void setFillColor (Color color)

  Setează culoarea de umplere pentru partea de progres a barei.
- void setPosition (int x, int y)

  Setează poziția colțului de sus-stânga al barei de progres.
- void setDimensions (int width, int height)

  Setează dimensiunile barei de progres.

Private Member Functions

- void loadImages ()

  Încarcă imaginea de fundal ("șanțul") pentru bara de progres.

Private Attributes

- int x
- float progress

  Progresul curent al barei, o valoare între 0.0f (0%) și 1.0f (100%).
- BufferedImage grooverImg

  Imaginea de fundal ("șanțul") pentru bara de progres.
- Color fillColor

  Culoarea de umplere pentru partea de progres a barei.

6.43.1 Detailed Description

Reprezintă o bară de progres personalizată pentru interfața utilizator.

Poate fi folosită pentru a afișa diverse tipuri de progres, cum ar fi încărcarea, viața, experiența etc.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 ProgressBar()

```
ui.ProgressBar.ProgressBar (
            int x,
            int y,
            int width,
            int height )
```

Constructor pentru ProgressBar.

Parameters

x	Poziția x a colțului de sus-stânga al barei de progres.
y	Poziția y a colțului de sus-stânga al barei de progres.
width	Lățimea barei de progres.
height	Înălțimea barei de progres.

6.43.3 Member Function Documentation

6.43.3.1 draw()

```
void ui.ProgressBar.draw (
            Graphics g )
```

Desenează bara de progres pe ecran.

Include fundalul și partea de umplere corespunzătoare progresului.

Parameters

g	Contextul grafic Graphics pe care se va desena.

6.43.3.2 getProgress()

```
float ui.ProgressBar.getProgress ( )
```

Returnează progresul curent al barei.

Returns

Progresul (între 0.0 și 1.0).

6.43.3.3 loadImages()

```
void ui.ProgressBar.loadImages ( )  [private]
```

Încarcă imaginea de fundal ("șanțul") pentru bara de progres.

6.43.3.4 setDimensions()

```
void ui.ProgressBar.setDimensions (
            int width,
            int height )
```

Setează dimensiunile barei de progres.

Parameters

width	Noua lățime.
height	Noua înălțime.

6.43.3.5 setFillColor()

```
void ui.ProgressBar.setFillColor (
            Color color )
```

Setează culoarea de umplere pentru partea de progres a barei.

Parameters

color	Noua culoare de umplere.

6.43.3.6 setPosition()

```
void ui.ProgressBar.setPosition (
            int x,
            int y )
```

Setează poziția colțului de sus-stânga al barei de progres.

Parameters

x	Noua coordonată x.
y	Noua coordonată y.

6.43.3.7 setProgress()

```
void ui.ProgressBar.setProgress (
            float progress )
```

Setează progresul curent al barei.

Valoarea este limitată între 0.0f și 1.0f.

Parameters

progress	Noua valoare a progresului (între 0.0 și 1.0).

6.43.4 Member Data Documentation

6.43.4.1 fillColor

```
Color ui.ProgressBar.fillColor  [private]
```

Culoarea de umplere pentru partea de progres a barei.

6.43.4.2 grooverImg

```
BufferedImage ui.ProgressBar.grooverImg  [private]
```

Imaginea de fundal ("șanțul") pentru bara de progres.

6.43.4.3 progress

```
float ui.ProgressBar.progress  [private]
```

Progresul curent al barei, o valoare între 0.0f (0%) și 1.0f (100%).

6.43.4.4 x

```
int ui.ProgressBar.x  [private]
```

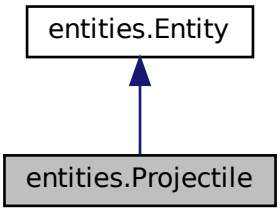
The documentation for this class was generated from the following file:

- src/ui/ProgressBar.java

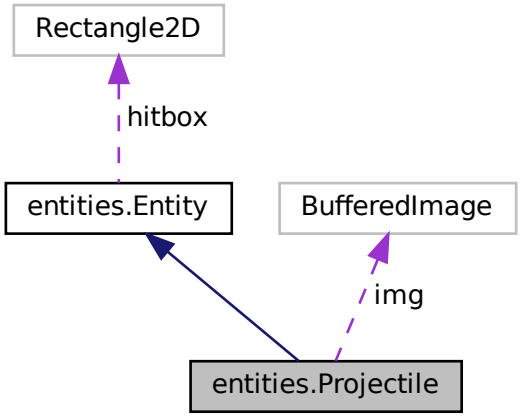
6.44 entities.Projectile Class Reference

Reprezintă un proiectil în joc.

Inheritance diagram for entities.Projectile:



Collaboration diagram for entities.Projectile:



Public Member Functions

- Projectile (float x, float y, int width, int height, int direction, int damage)

  Constructor pentru proiectile simple, desenate ca dreptunghiuri.
- Projectile (float x, float y, int direction, BufferedImage img, int damage)

  Constructor pentru proiectile bazate pe sprite-uri.
- void update ()

  Actualizează poziția proiectilului.
- void draw (Graphics g, int xLvlOffset)

  Desenează proiectilul pe ecran.
- boolean isActive ()

  Verifică dacă proiectilul este activ.
- void setActive (boolean active)

  Setează starea de activitate a proiectilului.
- Rectangle2D.Float getHitbox ()

  Returnează hitbox-ul proiectilului.
- int getDamage ()

  Returnează daunele pe care le provoacă proiectilul.

Private Attributes

- boolean active = true

  Indicator dacă proiectilul este activ și ar trebui actualizat/desenat.
- int direction

  Direcția proiectilului (0 pentru stânga, 1 pentru dreapta, folosind constantele din Enemy_Animation_Rows.Directions).
- float speed = 4.0f ∗ main.Game.SCALE

  Viteza de deplasare a proiectilului, scalată cu dimensiunea jocului.
- BufferedImage img

  Imaginea (sprite-ul) pentru proiectil.
- int damage

  Daunele pe care le provoacă proiectilul la impact.

Additional Inherited Members

6.44.1 Detailed Description

Reprezintă un proiectil în joc.

Această clasă extinde clasa Entity și definește comportamentul specific și atributele pentru proiectile, cum ar fi direcția, viteza, daunele și starea (activ/inactiv). Poate fi desenat ca un simplu dreptunghi sau folosind un sprite.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 Projectile() [1/2]

```
entities.Projectile.Projectile (
            float x,
            float y,
            int width,
            int height,
            int direction,
            int damage )
```

Constructor pentru proiectile simple, desenate ca dreptunghiuri.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
width	Lățimea proiectilului.
height	Înălțimea proiectilului.
direction	Direcția de mișcare (0 = stânga, 1 = dreapta).
damage	Daunele provocate de proiectil.

6.44.2.2 Projectile() [2/2]

```
entities.Projectile.Projectile (
            float x,
            float y,
            int direction,
            BufferedImage img,
            int damage )
```

Constructor pentru proiectile bazate pe sprite-uri.

Lățimea și înălțimea sunt derivate din imagine și scalate.

Parameters

x	Poziția inițială pe axa X.
y	Poziția inițială pe axa Y.
direction	Direcția de mișcare (0 = stânga, 1 = dreapta).
img	Imaginea (sprite-ul) pentru proiectil.
damage	Daunele provocate de proiectil.

6.44.3 Member Function Documentation

6.44.3.1 draw()

```
void entities.Projectile.draw (
            Graphics g,
            int xLvlOffset )
```

Desenează proiectilul pe ecran.

Dacă proiectilul nu este activ, nu desenează nimic. Desenează sprite-ul dacă este disponibil, altfel desenează un dreptunghi galben. Include logica pentru inversarea sprite-ului dacă se mișcă spre stânga.

Parameters

g	Contextul grafic pentru desenare.
xLvlOffset	Decalajul pe axa X al nivelului, pentru scrolling.

6.44.3.2 getDamage()

```
int entities.Projectile.getDamage ( )
```

Returnează daunele pe care le provoacă proiectilul.

Returns

Cantitatea de daune.

6.44.3.3 getHitbox()

```
Rectangle2D.Float entities.Projectile.getHitbox ( )
```

Returnează hitbox-ul proiectilului.

Returns

Un obiect Rectangle2D.Float reprezentând hitbox-ul.

Reimplemented from entities.Entity.

6.44.3.4 isActive()

```
boolean entities.Projectile.isActive ( )
```

Verifică dacă proiectilul este activ.

Returns

true dacă proiectilul este activ, false altfel.

6.44.3.5 setActive()

```
void entities.Projectile.setActive (
            boolean active )
```

Setează starea de activitate a proiectilului.

Parameters

active	Noua stare de activitate (true pentru activ, false pentru inactiv).

6.44.3.6 update()

```
void entities.Projectile.update ( )
```

Actualizează poziția proiectilului.

Dacă proiectilul nu este activ, nu face nimic. Mișcă proiectilul în direcția specificată cu viteza sa. Verificarea ieșirii din ecran este gestionată de EnemyManager.

6.44.4 Member Data Documentation

6.44.4.1 active

```
boolean entities.Projectile.active = true  [private]
```

Indicator dacă proiectilul este activ și ar trebui actualizat/desenat.

6.44.4.2 damage

```
int entities.Projectile.damage  [private]
```

Daunele pe care le provoacă proiectilul la impact.

6.44.4.3 direction

```
int entities.Projectile.direction  [private]
```

Direcția proiectilului (0 pentru stânga, 1 pentru dreapta, folosind constantele din Enemy_Animation_Rows.↩ Directions).

6.44.4.4 img

```
BufferedImage entities.Projectile.img  [private]
```

Imaginea (sprite-ul) pentru proiectil.

Poate fi null dacă se dorește un dreptunghi simplu.

6.44.4.5 speed

```
float entities.Projectile.speed = 4.0f * main.Game.SCALE  [private]
```

Viteza de deplasare a proiectilului, scalată cu dimensiunea jocului.

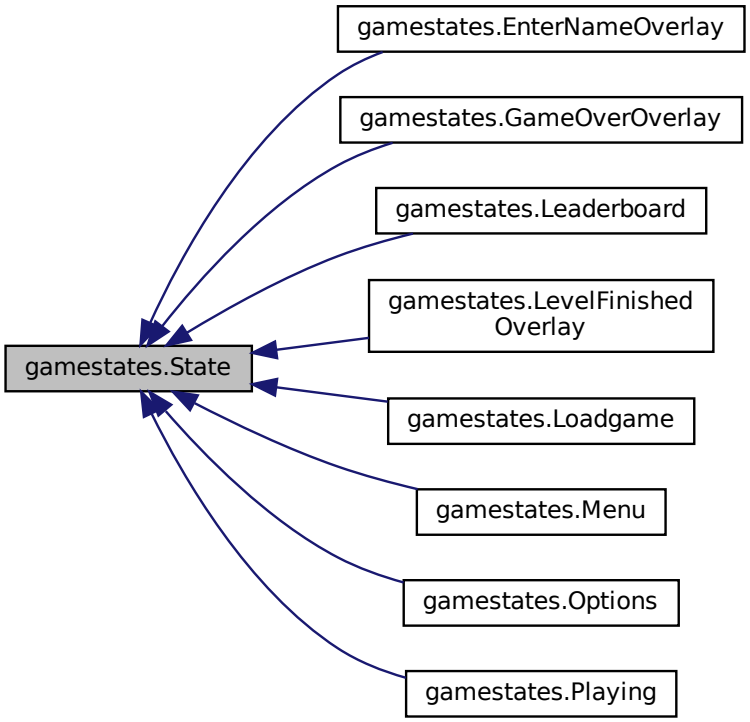
The documentation for this class was generated from the following file:

- src/entities/Projectile.java

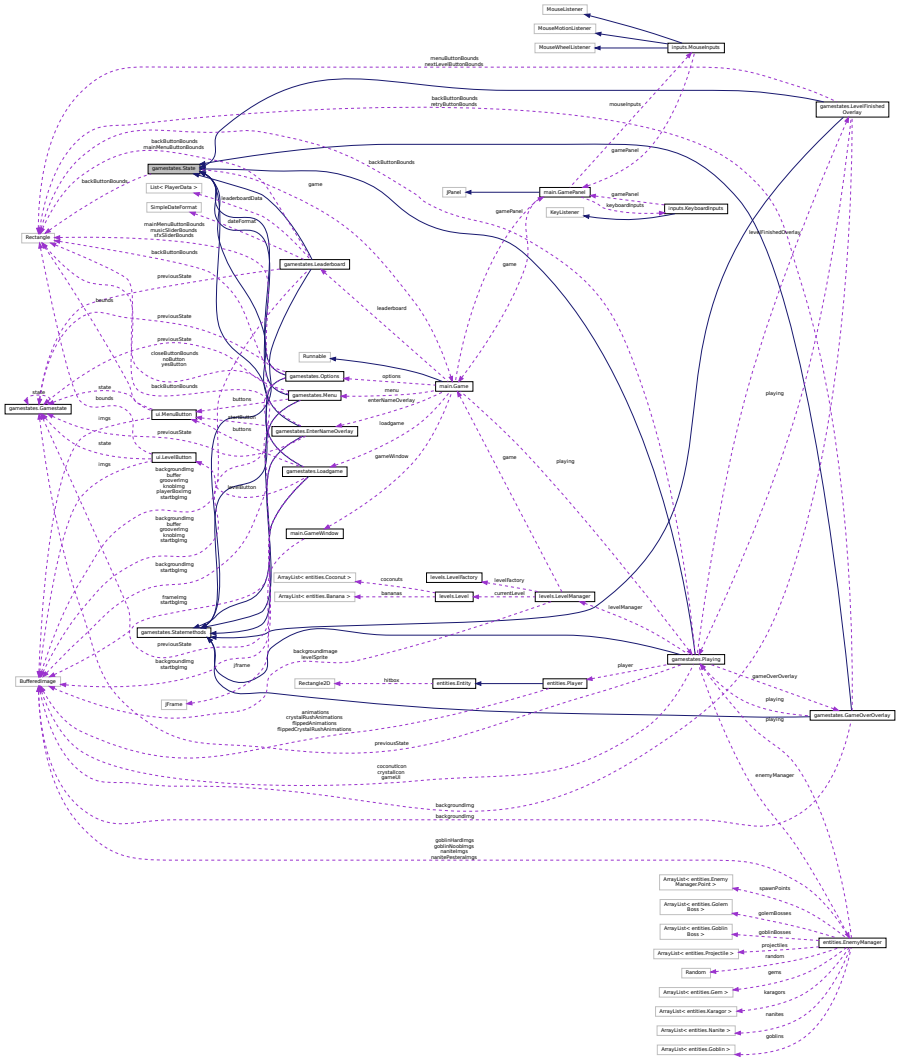
6.45 gamestates.State Class Reference

Clasă de bază pentru diferitele stări ale jocului (de ex., Meniu, Joc propriu-zis).

Inheritance diagram for gamestates.State:



Collaboration diagram for gamestates.State:



Public Member Functions

- State (Game game)

  Constructor pentru clasa State.
- boolean isIn (MouseEvent e, MenuButton mb)

  Verifică dacă evenimentul de mouse s-a produs în interiorul limitelor unui buton de meniu.
- boolean isIn (MouseEvent e, LevelButton lb)

  Verifică dacă evenimentul de mouse s-a produs în interiorul limitelor unui buton de nivel.
- Game getGame ()

  Returnează instanța principală a jocului.

Protected Member Functions

- void drawBackButton (Graphics2D g2d)

  Desenează hitbox-ul butonului "Înapoi" dacă
- boolean isBackButtonPressed (MouseEvent e)

  Verifică dacă butonul "Înapoi" a fost apăsat, pe baza coordonatelor evenimentului de mouse.

Protected Attributes

- Game game

  Referință la instanța principală a jocului Game.
- Rectangle backButtonBounds

  Dreptunghiul care definește limitele butonului "Înapoi", dacă există în această stare.
- boolean showDebugHitbox = true

  Flag pentru a afișa sau nu hitbox-urile în scop de depanare (implicit true).

6.45.1 Detailed Description

Clasă de bază pentru diferitele stări ale jocului (de ex., Meniu, Joc propriu-zis).

Oferă funcționalități comune și o referință la obiectul principal al jocului. Deși nu este declarată abstractă, este concepută pentru a fi extinsă de stări specifice.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 State()

```
gamestates.State.State (
            Game game )
```

Constructor pentru clasa State.

Parameters

game	Referință la instanța principală a jocului Game.

6.45.3 Member Function Documentation

6.45.3.1 drawBackButton()

```
void gamestates.State.drawBackButton (
            Graphics2D g2d )  [protected]
```

Desenează hitbox-ul butonului "Înapoi" dacă
showDebugHitbox

este true și
backButtonBounds

este definit. Folosit în scopuri de depanare.

Parameters

g2d	Contextul grafic 2D Graphics2D pe care se va desena.

Reimplemented in gamestates.Playing, gamestates.Menu, gamestates.Loadgame, gamestates.Leaderboard, and gamestates.GameOverOverlay.

6.45.3.2 getGame()

```
Game gamestates.State.getGame ( )
```

Returnează instanța principală a jocului.

Returns

Obiectul Game.

6.45.3.3 isBackButtonPressed()

```
boolean gamestates.State.isBackButtonPressed (
            MouseEvent e )  [protected]
```

Verifică dacă butonul "Înapoi" a fost apăsat, pe baza coordonatelor evenimentului de mouse.

Parameters

e	Evenimentul de mouse MouseEvent.

Returns

`true`
dacă butonul "Înapoi" a fost apăsat,
`false`
altfel sau dacă nu este definit.

6.45.3.4 isIn() [1/2]

```
boolean gamestates.State.isIn (
            MouseEvent e,
            LevelButton lb )
```

Verifică dacă evenimentul de mouse s-a produs în interiorul limitelor unui buton de nivel.

Parameters

e	Evenimentul de mouse MouseEvent.
lb	Butonul de nivel LevelButton de verificat.

Returns

`true`
dacă mouse-ul este în interiorul butonului,
`false`
altfel.

6.45.3.5 isIn() [2/2]

```
boolean gamestates.State.isIn (
            MouseEvent e,
            MenuButton mb )
```

Verifică dacă evenimentul de mouse s-a produs în interiorul limitelor unui buton de meniu.

Parameters

e	Evenimentul de mouse MouseEvent.
mb	Butonul de meniu MenuButton de verificat.

Returns

`true`
dacă mouse-ul este în interiorul butonului,
`false`
altfel.

6.45.4 Member Data Documentation

6.45.4.1 backButtonBounds

```
Rectangle gamestates.State.backButtonBounds  [protected]
```

Dreptunghiul care definește limitele butonului "Înapoi", dacă există în această stare.

6.45.4.2 game

```
Game gamestates.State.game  [protected]
```

Referință la instanța principală a jocului Game.

6.45.4.3 showDebugHitbox

```
boolean gamestates.State.showDebugHitbox = true  [protected]
```

Flag pentru a afișa sau nu hitbox-urile în scop de depanare (implicit true).

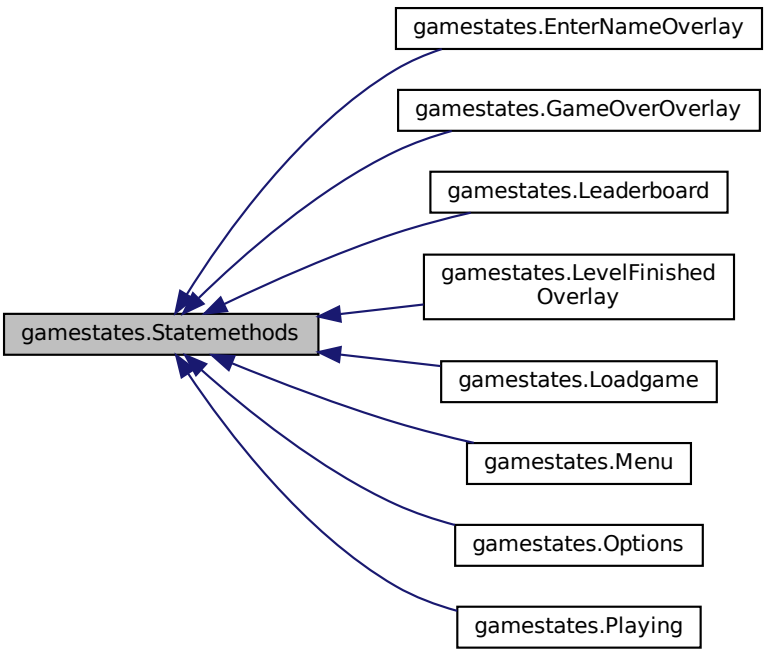
The documentation for this class was generated from the following file:

- src/gamestates/State.java

6.46 gamestates.Statemethods Interface Reference

Interfață ce definește metodele comune pe care toate stările de joc (gamestates) trebuie să le implementeze.

Inheritance diagram for gamestates.Statemethods:



Public Member Functions

- void update ()

  Actualizează logica specifică stării de joc.
- void draw (Graphics g)

  Desenează elementele grafice specifice stării de joc.
- void mouseClicked (MouseEvent e)

  Gestionază evenimentul de click al mouse-ului.
- void mousePressed (MouseEvent e)

  Gestionază evenimentul de apăsare a unui buton al mouse-ului.
- void mouseReleased (MouseEvent e)

  Gestionază evenimentul de eliberare a unui buton al mouse-ului.
- void mouseMoved (MouseEvent e)

  Gestionază evenimentul de mișcare a mouse-ului (fără butoane apăsate).
- void mouseDragged (MouseEvent e)

  Gestionază evenimentul de mișcare a mouse-ului cu un buton apăsat (drag).
- void keyPressed (KeyEvent e)

  Gestionază evenimentul de apăsare a unei taste.
- void keyReleased (KeyEvent e)

  Gestionază evenimentul de eliberare a unei taste.

6.46.1 Detailed Description

Interfață ce definește metodele comune pe care toate stările de joc (gamestates) trebuie să le implementeze.

Aceste metode gestionează actualizarea logicii, desenarea și procesarea input-ului (mouse și tastatură).

6.46.2 Member Function Documentation

6.46.2.1 draw()

```
void gamestates.Statemethods.draw (
            Graphics g )
```

Desenează elementele grafice specifice stării de joc.

Parameters

g	Contextul grafic Graphics pe care se va desena.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.2 keyPressed()

```
void gamestates.Statemethods.keyPressed (
            KeyEvent e )
```

Gestionază evenimentul de apăsare a unei taste.

Parameters

e	Evenimentul KeyEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.3 keyReleased()

```
void gamestates.Statemethods.keyReleased (
            KeyEvent e )
```

Gestionază evenimentul de eliberare a unei taste.

Parameters

e	Evenimentul KeyEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.4 mouseClicked()

```
void gamestates.Statemethods.mouseClicked (
            MouseEvent e )
```

Gestionază evenimentul de click al mouse-ului.

Parameters

e	Evenimentul MouseEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.5 mouseDragged()

```
void gamestates.Statemethods.mouseDragged (
            MouseEvent e )
```

Gestionază evenimentul de mișcare a mouse-ului cu un buton apăsat (drag).

Parameters

e	Evenimentul MouseEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.6 mouseMoved()

```
void gamestates.Statemethods.mouseMoved (
            MouseEvent e )
```

Gestionază evenimentul de mișcare a mouse-ului (fără butoane apăsate).

Parameters

e	Evenimentul MouseEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.7 mousePressed()

```
void gamestates.Statemethods.mousePressed (
            MouseEvent e )
```

Gestionază evenimentul de apăsare a unui buton al mouse-ului.

Parameters

e	Evenimentul MouseEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.8 mouseReleased()

```
void gamestates.Statemethods.mouseReleased (
            MouseEvent e )
```

Gestionază evenimentul de eliberare a unui buton al mouse-ului.

Parameters

e	Evenimentul MouseEvent generat.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

6.46.2.9 update()

```
void gamestates.Statemethods.update ( )
```

Actualizează logica specifică stării de joc.

Apelată în fiecare ciclu al buclei principale a jocului.

Implemented in gamestates.Playing, gamestates.Options, gamestates.Menu, gamestates.Loadgame, gamestates.LevelFinishedOverl
gamestates.Leaderboard, gamestates.GameOverOverlay, and gamestates.EnterNameOverlay.

The documentation for this interface was generated from the following file:

- src/gamestates/Statemethods.java

Chapter 7

File Documentation

7.1 src/database/InsertGet.java File Reference

Classes

- class database.InsertGet

Packages

- package database

7.2 src/entities/Banana.java File Reference

Classes

- class entities.Banana

  Reprezintă un obiect colectabil de tip "banană" în joc.

Packages

- package entities

7.3 src/entities/Coconut.java File Reference

Classes

- class entities.Coconut

  Reprezintă un obiect colectabil de tip "nucă de cocos" în joc.

Packages

- package entities

7.4 src/entities/Enemy.java File Reference

Classes

- class entities.Enemy

  Clasă abstractă de bază pentru toți inamicii din joc.

Packages

- package entities

7.5 src/entities/EnemyManager.java File Reference

Classes

- class entities.EnemyManager

  Gestionează toți inamicii din joc, inclusiv Nanites, Karagors, Goblins, GoblinBosses și GolemBosses.
- class entities.EnemyManager.Point

  Clasă internă simplă pentru a stoca informații despre un punct de spawn: coordonatele (x, y) și codul numeric al inamicului care trebuie spawnat.

Packages

- package entities

7.6 src/entities/Entity.java File Reference

Classes

- class entities.Entity

  Clasă abstractă de bază pentru toate entitățile din joc (de ex., jucător, inamici, obiecte).

Packages

- package entities

7.7 src/entities/Gem.java File Reference

Classes

- class entities.Gem

  Reprezintă un obiect de tip "Gem" (piatră prețioasă) în joc.

Packages

- package entities

7.8 src/entities/Goblin.java File Reference

Classes

- class entities.Goblin

  Reprezintă entitatea Goblin în joc.

Packages

- package entities

7.9 src/entities/GoblinBoss.java File Reference

Classes

- class entities.GoblinBoss

  Reprezintă entitatea Goblin Boss în joc.
- enum entities.GoblinBoss.ActionState

  Enumerare pentru stările de acțiune posibile ale Goblin Boss-ului.

Packages

- package entities

7.10 src/entities/GolemBoss.java File Reference

Classes

- class entities.GolemBoss

  Reprezintă entitatea Golem Boss în joc.
- enum entities.GolemBoss.ActionState

  Enumerare pentru stările de acțiune specifice Golem Boss-ului.

Packages

- package entities

7.11 src/entities/Karagor.java File Reference

Classes

- class entities.Karagor

  Reprezintă entitatea Karagor în joc, care poate funcționa ca un boss.

Packages

- package entities

7.12 src/entities/Nanite.java File Reference

Classes

- class entities.Nanite

  Reprezintă entitatea Nanite în joc.

Packages

- package entities

7.13 src/entities/Player.java File Reference

Classes

- class entities.Player

Packages

- package entities

7.14 src/entities/Projectile.java File Reference

Classes

- class entities.Projectile

  Reprezintă un proiectil în joc.

Packages

- package entities

7.15 src/gamestates/EnterNameOverlay.java File Reference

Classes

- class gamestates.EnterNameOverlay

  Reprezintă un overlay pentru introducerea numelui jucătorului.

Packages

- package gamestates

7.16 src/gamestates/GameOverOverlay.java File Reference

Classes

- class gamestates.GameOverOverlay

  Reprezintă overlay-ul afișat la sfârșitul jocului (Game Over).

Packages

- package gamestates

7.17 src/gamestates/Gamestate.java File Reference

Classes

- enum gamestates.Gamestate

  Enumerație ce definește diferitele stări posibile ale jocului.

Packages

- package gamestates

7.18 src/gamestates/Leaderboard.java File Reference

Classes

- class gamestates.Leaderboard

  Reprezintă starea de joc pentru afișarea clasamentului (Leaderboard).
- class **gamestates.Leaderboard.PlayerData**

  Clasă internă statică pentru a stoca datele unui jucător afișate în clasament.

Packages

- package gamestates

7.19 src/gamestates/LevelFinishedOverlay.java File Reference

Classes

- class gamestates.LevelFinishedOverlay

  Reprezintă overlay-ul afișat la finalizarea cu succes a unui nivel.

Packages

- package gamestates

7.20 src/gamestates/Loadgame.java File Reference

Classes

- class gamestates.Loadgame

  Reprezintă starea de joc pentru ecranul de încărcare a unui joc salvat.

Packages

- package gamestates

7.21 src/gamestates/Menu.java File Reference

Classes

- class gamestates.Menu

  Reprezintă starea de joc pentru meniul principal.

Packages

- package gamestates

7.22 src/gamestates/Options.java File Reference

Classes

- class gamestates.Options

  Reprezintă starea de joc pentru meniul de opțiuni.

Packages

- package gamestates

7.23 src/gamestates/Playing.java File Reference

Classes

- class gamestates.Playing

  Reprezintă starea principală de joc (gameplay-ul propriu-zis).

Packages

- package gamestates

7.24 src/gamestates/State.java File Reference

Classes

- class gamestates.State

  Clasă de bază pentru diferitele stări ale jocului (de ex., Meniu, Joc propriu-zis).

Packages

- package gamestates

7.25 src/gamestates/Statemethods.java File Reference

Classes

- interface gamestates.Statemethods

  Interfață ce definește metodele comune pe care toate stările de joc (gamestates) trebuie să le implementeze.

Packages

- package gamestates

7.26 src/inputs/KeyboardInputs.java File Reference

Classes

- class inputs.KeyboardInputs

  Gestionează input-ul de la tastatură pentru joc.

Packages

- package inputs

7.27 src/inputs/MouseInputs.java File Reference

Classes

- class inputs.MouseInputs

  Gestionează input-ul de la mouse pentru joc.

Packages

- package inputs

7.28 src/levels/Level.java File Reference

Classes

- class levels.Level

  Reprezintă un nivel individual în joc.

Packages

- package levels

7.29 src/levels/LevelFactory.java File Reference

Classes

- class levels.LevelFactory

  Clasă de tip Factory responsabilă pentru crearea obiectelor de tip Level.

Packages

- package levels

7.30 src/levels/LevelManager.java File Reference

Classes

- class levels.LevelManager

  Gestionează încărcarea, desenarea și tranziția între nivelurile jocului.

Packages

- package levels

7.31 src/main/Game.java File Reference

Classes

- class main.Game

  Clasa principală a jocului, responsabilă pentru gestionarea stărilor de joc, bucla principală a jocului (game loop) și inițializarea componentelor cheie.

Packages

- package main

7.32 src/main/GamePanel.java File Reference

Classes

- class main.GamePanel

  Panoul principal al jocului, extinzând JPanel.

Packages

- package main

7.33 src/main/GameWindow.java File Reference

Classes

- class main.GameWindow

  Reprezintă fereastra principală a jocului.

Packages

- package main

7.34 src/main/Main.java File Reference

Classes

- class main.Main

  Clasa principală a aplicației, conținând punctul de intrare (metoda main).

Packages

- package main

7.35 src/ui/LevelButton.java File Reference

Classes

- class ui.LevelButton

  Reprezintă un buton specific pentru selectarea nivelurilor în interfața utilizator.

Packages

- package ui

7.36 src/ui/MenuButton.java File Reference

Classes

- class ui.MenuButton

  Reprezintă un buton generic pentru meniurile din interfața utilizator.

Packages

- package ui

7.37 src/ui/ProgressBar.java File Reference

Classes

- class ui.ProgressBar

  Reprezintă o bară de progres personalizată pentru interfața utilizator.

Packages

- package ui

7.38 src/utilz/Constants.java File Reference

Classes

- class utilz.Constants

  Clasă ce conține constante globale utilizate în diverse părți ale jocului.
- class **utilz.Constants.UI**

  Constante legate de interfața utilizator (UI).
- class **utilz.Constants.UI.Buttons**

  Constante pentru dimensiunile butoanelor.
- class **utilz.Constants.Directions**

  Constante pentru direcțiile de mișcare sau orientare.
- class **utilz.Constants.EnemyConstants**

  Constante legate de inamici.
- class **utilz.Constants.Collectibles**

  Constante legate de obiectele colectabile.
- class **utilz.Constants.Tiles**

  Constante legate de tile-urile din nivel.

Packages

- package utilz

7.39 src/utilz/Enemy_Animation_Rows.java File Reference

Classes

- enum utilz.Enemy_Animation_Rows

  Enumerație ce definește rândurile de animație și numărul de cadre pentru inamici.
- class **utilz.Enemy_Animation_Rows.Directions**

  Clasă internă statică ce definește constante pentru direcții.

Packages

- package utilz

7.40 src/utilz/Gorilla_Animation_rows.java File Reference

Classes

- enum utilz.Gorilla_Animation_rows

  Enumerație ce definește rândurile de animație și numărul de cadre pentru personajul jucător (Gorila).
- class **utilz.Gorilla_Animation_rows.Directions**

  Clasă internă statică ce definește constante pentru direcții.

Packages

- package utilz

7.41 src/utilz/HelpMethods.java File Reference

Classes

- class utilz.HelpMethods

  Clasă utilitară ce conține metode statice ajutătoare, în principal pentru detecția coliziunilor și poziționarea entităților în cadrul nivelului.

Packages

- package utilz

7.42 src/utilz/LevelProgress.java File Reference

Classes

- class utilz.LevelProgress

  Clasă simplă de date (POJO - Plain Old Java Object) pentru a stoca informațiile despre progresul jucătorului într-un anumit nivel.

Packages

- package utilz

7.43 src/utilz/LoadSave.java File Reference

Classes

- class utilz.LoadSave

  Clasă utilitară responsabilă pentru încărcarea resurselor jocului, cum ar fi imaginile (sprite atlas-uri) și datele nivelurilor din fișiere.

Packages

- package utilz

Index