Project Flip Jump

Group 17

Mikael Ahlberg
Daniel Ericsson
Axel Stenkula
Johannes Svensson
Fredrik Vretblad

5.5 Detailed Design

Class Audio

Fields

Field name: private OpenALSource soundSource
Comment: Handles the sound connection for the class.

Field name: private Settings settings;
Comment: A link/reference to the settings class for easier retrieval of game settings.

Methods

Method name: public Audio(Settings settings)
Parameter (Settings): The link/reference to the settings class.
Return value: None
Description: The constructor for the class. It initiates all OpenAL code so the game is ready to play sounds.
Pre-condition: The settings class has been initiated and settings loaded.
Post-condition: The OpenAL-engine has been initialized.
Called by: The Game class constructor.
Implements: -

Method name: public void play(OpenALWave soundClip)
Parameter (OpenALWave): The sound clip to be played.
Return value: None
Description: The method plays the specified sound clip that's sent as a parameter to the method.
Pre-condition: The OpenAL-enginge has been initialized.
Post-condition: The sound clip has been played.
Called by: Called by speedup-event, highscore-event and gameover-event.
Calls: Calls OpenALSource play()-method.
Implements: 6.1.3.5 High score sound

Method name: public void stopCurrent()
Parameter: None.
Return value: None.
Description: The method stops all current sound effects.
Pre-condition: The OpenAL-enginge has been initialized.
Post-condition: All sound clips are stopped.
Called by: The Game class menuLoop()-method.
Calls: Calls OpenALSource stop()-method.
Implements: 6.1.3.5 High score sound

Method name: public boolean toggleMute()
Parameter: None.
Return value (Boolean): Returns true if the sound is switched on and vice versa.

Description: The method changes the mute settings for the game.
Pre-condition: The OpenAL-enginge has been initialized.
Post-condition: The sound has been switched on or off by the method.
Called by: The Game class menuLoop()-method.
Calls: Calls the Settings class setMute()-method.
Implements: 6.1.1.4 Appearance of sound

Method name: public boolean isMuted()
Parameter: None.
Return value (Boolean): Returns true if the sound is muted and vice versa.
Description: The method returns the current state of the mute settings.
Pre-condition: The Settings for the mute function has been initiated.
Post-condition: The value has been returned.
Called by: Called by play-event and toggleMute-event.
Calls: Calls the Settings class isMute()-method.
Implements: 6.1.1.4 Apperance of sound

Class Block

Methods

Method name: public Block(int x, int y, int height, int width)
Parameter (int x): The blocks initial x value.
Parameter (int y): The blocks initial y value.
Parameter (int height): The blocks height value.
Parameter (int width): The blocks width value.
Return value: None
Description: The constructor for the class. It makes sure that the initial values for the block is set corrected, like dimension, texture and position in the world.
Pre-condition: The game world has been created.
Post-condition: A new block has been created in the world.
Called by: The World class constructor.
Calls: Calls the setTexture()-method in the superclass.
Implements: -

Abstract Class Entity

Fields

Field name: private int x;
Comment: The X-position of the entity in the world.

Field name: private int y;
Comment: The Y-position of the entity in the world.

Field name: private int velocityX;
Comment: The current speed in the X-direction.

Field name: private int velocityY;
Comment: The current speed in the Y-direction.

Field name: private int height;
Comment: The entitys height in the world.

Field name: private int width;
Comment: The entitys width in the world.

Field name: private String texture;
Comment: A path to the entitys texture.

Methods

Method name: public Entity(int x, int y, int height, int width)
Parameter (int x): The entity's initial x value.
Parameter (int y): The entity's initial y value.
Parameter (int height): The entity's height value.
Parameter (int width): The entity's width value.
Return value: None
Description: The constructor for the class. This class is an abstract class and is not instanced.
Pre-condition: The game world has been created.
Post-condition: A new entity of some kind has been created in the world.
Called by: The classes that inherits from this class.
Implements: -

Method name: public void setTexture(String texture)
Parameter (String): The path to the texture.
Return value: None
Description: The method sets the texture for the object to the specified path.
Pre-condition: A object that inherits from entity has been created.
Post-condition: The texture has been set.
Called by: The specified objects constructor.
Calls: None.
Implements: -

Method name: public String getTexture()
Parameter: None
Return value (String): Returns the path to the objects texture.
Description: The method returns the path to the objects texture.
Pre-condition: An object that inherits from entity has been created and texture has been set.
Post-condition: The texture path has been returned.
Called by: The Graphics class draw()-methods.
Calls: None.
Implements: -

Method name: public String getTexture()
Parameter: None
Return value (String): Returns the path to the objects texture.
Description: The method returns the path to the objects texture.
Pre-condition: An object that inherits from entity has been created and texture has been set.
Post-condition: The texture path has been returned.
Called by: The Graphics class draw()-methods.
Calls: None.
Implements:

Method name: public void setX(int x)
Parameter (int x): The entity's X-position.
Return value: None.
Description: The method sets the entity's X-position in the world.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The X-position has been set.
Called by: The World class collisionDetection()-method and the local update()-method.
Calls: None.
Implements: -

Method name: public int getX()
Parameter: None.
Return value (int): Returns the entity's X-position in the world.
Description: The method returns the entity's X-position.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The X-position has been returned.
Called by: The World class collisionDetection()-method.
Calls: None.
Implements: -

Method name: public void setY(int y)
Parameter (int y): The entity's Y-position.
Return value: None.
Description: The method sets the entity's Y-position in the world.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The Y-position has been set.
Called by: The World class collisionDetection()-method and the local update()-method.
Calls: None.
Implements: -

Method name: public int getY()
Parameter: None.
Return value (int): Returns the entity's Y-position in the world.
Description: The method returns the entity's Y-position.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The Y-position has been returned.

Called by: The World class collisionDetection()-method.
Calls: None.
Implements: -

Method name: public void setVelocityX(int velocityX)
Parameter (int velocityX): The entity's speed in X-direction.
Return value: None.
Description: The method sets the entity's speed in X-direction.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The X-velocity has been set.
Called by: The World class updateWorld()-method.
Calls: None.
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.5 Screen movement
6.1.2.8 Screen speed

Method name: public int getVelocityX()
Parameter: None.
Return value (int): Returns the entity's speed in X-direction.
Description: The method returns the entity's speed in X-direction.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The speed in X-direction has been returned.
Called by: The World class updateWorld()-method.
Calls: None.
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.5 Screen movement
6.1.2.8 Screen speed

Method name: public void setVelocityY(int velocityY)
Parameter (int velocityY): The entity's speed in Y-direction.
Return value: None.
Description: The method sets the entity's speed in Y-direction.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The Y-velocity has been set.
Called by: The World class updateWorld()-method.
Calls: None.
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.5 Screen movement
6.1.2.8 Screen speed

Method name: public int getVelocityY()
Parameter: None.

Return value (int): Returns the entity's speed in Y-direction.
Description: The method returns the entity's speed in Y-direction.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The speed in Y-direction has been returned.
Called by: The World class updateWorld()-method.
Calls: None.
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.5 Screen movement
6.1.2.8 Screen speed

Method name: public void setHeight(int height)
Parameter (int height): The entity's height.
Return value: None.
Description: The method sets the entity's height.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The height has been set.
Called by: The Entity's class constructor.
Calls: None.
Implements: -

Method name: public int getHeight()
Parameter: None.
Return value (int): Returns the entity's height.
Description: The method returns the height.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The height has been returned.
Called by: The World class collisionDetection()-method.
Calls: None.
Implements: -

Method name: public void setWidth(int width)
Parameter (int width): The entity's width.
Return value: None.
Description: The method sets the entity's width.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The width has been set.
Called by: The Entity's class constructor.
Calls: None.
Implements: -

Method name: public int getWidth()
Parameter: None.
Return value (int): Returns the entity's width.
Description: The method returns the width.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The width has been returned.

Called by: The World class collisionDetection()-method.
Calls: None.
Implements: -

Method name: public void update()
Parameter: None.
Return value: None.
Description: The method converts the velocity variables to actual coordinates for each frame in the game and thereby moves the object in the world.
Pre-condition: An object that inherits from entity has been created.
Post-condition: The coordinates has been updated with respect to the velocity vectors.
Called by: The World class updateWorld()-method and the collisionDetection()-method.
Calls: setX(), setY(), getX(), getY(), setVelocityX(), setVelocityY(), getVelocityX() and getVelocityY()-methods.
Implements: -

Class Game

Fields

Field name: private World world;
Comment: The world object.

Field name: private Graphics graphics;
Comment: The graphics object.

Field name: private Audio audio;
Comment: The audio object.

Field name: private Menu currentMenu;
Comment: Keeps track of the current menu that's displayed to the screen.

Field name: private Menu startMenu;
Comment: The startMenu object.

Field name: private Menu optionsMenu;
Comment: The optionsMenu object.

Field name: private Menu difficultyMenu;
Comment: The difficultyMenu object.

Field name: private Menu tutorialMenu;
Comment: The tutorialMenu object.

Field name: private HighScore highScore;
Comment: The highScore object.

Field name: private Settings settings;
Comment: The settings object.

Field name: private boolean enter;
Comment: Keeps track if the enter key has been pressed.

Field name: private boolean escape;
Comment: Keeps track if the escape key has been pressed.

Field name: private boolean up;
Comment: Keeps track if the up key has been pressed.

Field name: private boolean down;
Comment: Keeps track if the down key has been pressed.

Field name: private boolean right;
Comment: Keeps track if the right key has been pressed.

Field name: private boolean left;
Comment: Keeps track if the left key has been pressed.

Field name: private boolean space;
Comment: Keeps track if the space key has been pressed.

Methods

Method name: public Game()
Parameter: None
Return value: None
Description: The constructor for the class. It initializes all objects and contains the game loop. In other words, the core of the game.
Pre-condition: The program has been started.
Post-condition: All vital objects has been initialized.
Called by: The main()-method.
Calls: initiateMenus()-method and object constructors.
Implements: -

Method name: public void initiateMenus()
Parameter: None
Return value: None
Description: The method sets up all menu items..
Pre-condition: The Game class has been created.
Post-condition: All menus has been set-up.
Called by: The Game constructor.
Calls: Set-up functions in menu.
Implements: -

Method name: public void actOnInput()
Parameter: None
Return value: None
Description: The method checks for input and acts accordingly.
Pre-condition: The gameLoop()-method is running.
Post-condition: Input has been checked.
Called by: The gameLoop()-method.

Calls: The Player class setVelocityX() and setVelocityy()-methods
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.4 Use special item
6.1.3.2 Enter high score list

Method name: public void resetInput()
Parameter: None
Return value: None
Description: The method resets the input booleans.
Pre-condition: The gameLoop()-method is running.
Post-condition: Input has been reset.
Called by: The gameLoop()-method and the menuLoop()-method.
Calls: None
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.4 Use special item
6.1.3.2 Enter high score list

Method name: public void keyPressed(KeyEvent arg0)
Parameter (KeyEvent): A key code for the pressed key
Return value: None
Description: The method is a part of the KeyListener-events.
Pre-condition: The Game class has been created.
Post-condition: Input has been confirmed.
Called by: Java underlying system.
Calls: None
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.4 Use special item
6.1.3.2 Enter high score list

Method name: public void keyReleased(KeyEvent arg0)
Parameter (KeyEvent): A key code for the released key
Return value: None
Description: The method is a part of the KeyListener-events.
Pre-condition: The Game class has been created.
Post-condition: Input has been confirmed.
Called by: Java underlying system.
Calls: None
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.4 Use special item
6.1.3.2 Enter high score list

Method name: public void keyTyped(KeyEvent arg0)
Parameter (KeyEvent): A key code for the typed key
Return value: None
Description: The method is a part of the KeyListener-events.
Pre-condition: The Game class has been created.
Post-condition: Input has been confirmed.
Called by: Java underlying system.
Calls: None
Implements:
6.1.2.1 Sideways movement
6.1.2.2 Character jump
6.1.2.4 Use special item
6.1.3.2 Enter high score list

Method name: public boolean isGameOver()
Parameter: None
Return value (boolean): Returns true if the game is over and vice versa.
Description: The method checks if the game is over, typically the player is dead or the user has pressed the escape key.
Pre-condition: The gameLoop() is running.
Post-condition: The state of the game has been checked.
Called by: The gameLoop()-method.
Calls: The Players isAlive()-method.
Implements: 6.1.1.2 Durance of play

Method name: public void menuLoop()
Parameter: None
Return value: None
Description: The method handles the menu system and constantly checks for input and calls the appropriete methods. It also calls the graphics draw()-method to draw the menu to the screen.
Pre-condition: The Game class has been created.
Post-condition: The Game has exited.
Called by: The main()-method.
Calls: Methods concerning the game play, like the gameLoop(), and menu functions.
Implements: -

Method name: public void gameLoop()
Parameter: None
Return value: None
Description: The method handles the in game functionality and constantly checks for input through the actOnInput()-method. It also make sure that the world is updated and that the game is drawn to the screen.
Pre-condition: The game is running.
Post-condition: The menu is running.
Called by: The menuLoop()-method.
Calls: actOnInput(), isGameOver(), resetInput(), world.updateWorld() and the graphics.draw()-methods.
Implements: -

Method name: public void main(String[] args)
Parameter (String[] args): Unused input to the program
Return value: None
Description: The main()-method for the program. It creates an Game object.
Pre-condition: The program has been started.
Post-condition: The menuLoop() is running.
Called by: The Java subsystem.
Calls: menuLoop()-method
Implements: -

Class Graphics

Fields

Field name: private HashMap<String,Object> textures;
Comment: Keeps track of the texture used in game.

Field name: private Settings setting;
Comment: A reference back to the settings object.

Methods

Method name: public Graphics(Settings settings)
Parameter (Settings): None
Return value: None
Description: The constructor for the class. It initializes all OpenGL code.
Pre-condition: The Game class has been created.
Post-condition: All OpenGL systems has been initialized.
Called by: The Game class constructor.
Calls: None
Implements: -

Method name: public void initOpenGL()
Parameter: None
Return value: None
Description: Intitalizes OpenGL systems.
Pre-condition: The Graphics class has been created.
Post-condition: All OpenGL systems is ready for use.
Called by: The Graphics constructor.
Calls: None
Implements: -

Method name: public void loadTexture(String textureName)
Parameter (String): A path to the texture to be loaded.
Return value: None
Description: Loads the specified texture from disk into the HashMap.
Pre-condition: The Graphics class has been created.
Post-condition: The texture has been loaded into the HashMap.
Called by: The Graphics draw()-method.
Calls: None
Implements: -

Method name: public void drawMenu(Menu currentMenu)
Parameter (Menu): The menu to be drawn to the screen.
Return value: None
Description: The method draws the specified menu to the screen.
Pre-condition: The menuLoop() is running.

Post-condition: The menu has been drawn to the screen.
Called by: The menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public void fadeOut(int time)
Parameter (int): The time it takes before the screen becomes completely black.
Return value: None
Description: The method fades all graphics to black background.
Pre-condition: The Graphics class has been created and the screen is visible.
Post-condition: The screen is black.
Called by: The menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public void fadeIn(int time)
Parameter (int): The time it takes before the screen becomes completely visible.
Return value: None
Description: The method fades in all graphics from a black background.
Pre-condition: The Graphics class has been created and the screen is black.
Post-condition: The screen is visible.
Called by: The menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public void draw(World world)
Parameter (World): A reference back to the world object.
Return value: None
Description: The method draws a frame of the world to the screen (player, items, blocks, background etc).
Pre-condition: The gameLoop() is running.
Post-condition: The world has been drawn to the screen.
Called by: The gameLoop() in the Game class.
Calls: drawWorld() and drawObjects()
Implements: -

Method name: public void drawObjects(LinkedList<Entity> entities)
Parameter (LinkedList): A reference to the list of entities placed in the world.
Return value: None
Description: The method draws a frame of the world objects to the screen.
Pre-condition: The gameLoop() is running.
Post-condition: The world objects has been drawn to the screen.
Called by: The draw()-method.
Calls: None
Implements: -

Method name: public void drawWorld(World world)

Parameter (World): A reference back to the world object.
Return value: None
Description: The method draws a frame of the world to the screen.
Pre-condition: The gameLoop() is running.
Post-condition: The world has been drawn to the screen.
Called by: The draw()-method.
Calls: None
Implements: -

Method name: public void rotateCamera(boolean left)
Parameter (boolean): Direction of the flip, left = true, right = false
Return value: None
Description: The method rotates the camera/viewport in the given direction.
Pre-condition: The gameLoop() is running.
Post-condition: The camera has been rotated.
Called by: The updateWorld()-method in the World class.
Calls: None
Implements: 6.1.2.12 Flip function

Method name: public boolean toggleFullscreen()
Parameter: None
Return value (boolean): Returns true if fullscreen is activated.
Description: The method switches fullscreen on or off.
Pre-condition: The menuLoop() is running.
Post-condition: The fullscreen mode has been changed.
Called by: The menuLoop().
Calls: setFullscreen() in the Settings class.
Implements: 6.1.1.3 Screen size

Method name: public boolean isFullscreen()
Parameter: None
Return value (boolean): Returns true if fullscreen is activated.
Description: The method returns the current state of the fullscreen value.
Pre-condition: The menuLoop() is running.
Post-condition: The fullscreen value has been returned.
Called by: The menuLoop().
Calls: getFullscreen in the Settings class.
Implements: 6.1.1.3 Screen size

Class Highscore

Fields

Field name: private LinkedList<ScoreData> score
Comment: Keeps track of the different high scores.

Methods

Method name: public HighScore(String title)
Parameter (String): Title name for the graphical interface.
Return value: None
Description: The constructor for the class. It initializes all components used for the high score list.
Pre-condition: The high score option has been selected.
Post-condition: All components has been created.
Called by: The Game class constructor.
Calls: None
Implements: -

Method name: public void save()
Parameter: None
Return value: None
Description: Saves the current high score list down to a locally stored file.
Pre-condition: The high score list is loaded into memory.
Post-condition: The high score list is stored to file.
Called by: Entered new highscore-event.
Calls: None
Implements:
6.1.3.1 Storing of high score
6.1.3.2 Enter high score list
6.1.3.4 Reset high score

Method name: public void load()
Parameter: None
Return value: None
Description: Loads the locally stored list to memory.
Pre-condition: The high score list is in a readable file.
Post-condition: The high score list is loaded.
Called by: The Game constructor.
Calls: None
Implements: 6.1.3.3 Check high score

Method name: public boolean isHighscore()
Parameter: None
Return value (boolean): Always returns true
Description: An identifier method for the high score list.
Pre-condition: The high score object has been created.
Post-condition: The value true has been returned.

Called by: The Graphics drawMenu()-method.
Calls: None
Implements: 6.1.3.3 Check high score

Class Item

Methods

Method name: public Item(int x, int y, int height, int width)
Parameter (int x): The items initial x value.
Parameter (int y): The items initial y value.
Parameter (int height): The items height value.
Parameter (int width): The items width value.
Return value: None
Description: The constructor for the class. It makes sure that the initial values for the item is set corrected, like dimension, texture and position in the world.
Pre-condition: The game world has been created.
Post-condition: A new item has been created in the world.
Called by: The World class constructor.
Calls: Calls the setTexture()-method in the superclass.
Implements: -

Class Menu

Fields

Field name: private ArrayList<MenuData> optionList
Comment: Keeps track of the different selectable menu options.

Field name: private String title
Comment: The name of the menu screen.

Field name: private int currentSelection
Comment: Keeps track of the current selected menu alternative.

Methods

Method name: public Menu(String title)
Parameter (String): Title name for the graphical interface.
Return value: None
Description: The constructor for the class. It initializes all components used for the menu list.
Pre-condition: The Game class has been created.
Post-condition: All components for the menu has been created.
Called by: The Game class constructor.
Calls: None
Implements: -

Method name: public void addComponent(String name, Menu menuLink)
Parameter (String): The text to be displayed for the component in the GUI.
Parameter (Menu): Link to a new menu window.

Return value: None
Description: Adds a new selectable menu component that links to a new menu window.
Pre-condition: The menu object has been created.
Post-condition: A new component has been added to the menu window.
Called by: initiateMenus() in the Game class.
Calls: None
Implements: -

Method name: public void addComponent(String name, int funcIndex)
Parameter (String): The text to be displayed for the component in the GUI.
Parameter (int): A func index so the propper function gets called when selecting this alternative.
Return value: None
Description: Adds a new selectable menu component that doesn't link to a new menu window.
Pre-condition: The menu object has been created.
Post-condition: A new component has been added to the menu window.
Called by: initiateMenus() in the Game class.
Calls: None
Implements: -

Method name: public void changeComponent(String name, String newName)
Parameter (String): The name for the component to change (GUI name).
Parameter (String): The new name for the component (GUI name).
Return value: None
Description: Changes an already defined component in the menu.
Pre-condition: The menu object has been created.
Post-condition: The component has changed its name.
Called by: menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public void changeCurrentComponent(String newName)
Parameter (String): The new name for the component (GUI name).
Return value: None
Description: Changes the currently selected component in the menu.
Pre-condition: The menu object has been created and a component has been selected.
Post-condition: The component has changed its name.
Called by: menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public void removeComponent(String name)
Parameter (String): The name for the component to be removed (GUI name).
Return value: None
Description: Removes the specified component in the menu.
Pre-condition: The menu object has been created.
Post-condition: The component has been removed.
Called by: menuLoop() in the Game class.
Calls: None

Implements: -

Method name: public ArrayList<MenuData> getComponentList()
Parameter: None
Return value (ArrayList): The list of components in this menu object.
Description: Returns the component list containing all objects for this menu.
Pre-condition: The menu object has been created.
Post-condition: The component list has been returned.
Called by: drawMenu() in the Graphics class.
Calls: None
Implements: -

Method name: public int getAction(boolean enter, boolean space, boolean escape, boolean up, boolean down)
Parameter (all booleans): Is true if a key has been pressed.
Return value (int): Returns the funcIndex for the currently selected menu option.
Description: Returns an index so that the caller can call the right method when a menu option has been selected.
Pre-condition: The menu object has been created.
Post-condition: The funcIndex has been returned.
Called by: menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public Menu getNewMenu()
Parameter: None
Return value (Menu): Returns a reference to a new menu window.
Description: If the currently selected menu option links to a new menu window, this method will return the new menu window. The function should only be called when that's true.
Pre-condition: The menu object has been created and the currently selected menu option links to a new menu window.
Post-condition: The reference to the new menu window has been returned..
Called by: menuLoop() in the Game class.
Calls: None
Implements: -

Method name: public boolean isHighscore()
Parameter: None
Return value (boolean): Always returns false
Description: An identifier method for the high score list.
Pre-condition: The menu object has been created.
Post-condition: The value false has been returned.
Called by: The Graphics drawMenu()-method.
Calls: None
Implements: -

Class Player

Fields

Field name: private String t_facingLeft
Comments: The path to the texture for the character when facing left.

Field name: private String t_facingRight
Comments: The path to the texture for the character when facing right.

Field name: private String t_jumpLeft
Comments: The path to the texture for the character when jumping left.

Field name: private String t_jumpRight
Comments: The path to the texture for the character when jumping right.

Field name: private int collectedItems
Comments: The amount of items the user currently have.

Field name: private int score
Comments: The users current score.

Field name: private boolean isAlive
Comments: True if the user is still alive (in the game).

Methods

Method name: public Player(int x, int y, int height, int width)
Parameter (int x): The players initial x value.
Parameter (int y): The players initial y value.
Parameter (int height): The players height value.
Parameter (int width): The players width value.
Return value: None
Description: The constructor for the class. It makes sure that the initial values for the player is set corrected, like dimension, texture and position in the world.
Pre-condition: The game world has been created.
Post-condition: A new item has been created in the world.
Called by: The World class constructor.
Calls: Calls the setTexture()-method in the superclass.
Implements: -

Method name: public String getTexture()
Return value(String): Returns the path to the current picture.
Description: Overloads Entitys getTexture() to return the picture that represents the current state of the player.
Pre-condition: The game been initialized.
Post-condition: The texture path has been returned.
Called by: The Graphics class draw()-methods.
Calls: None.
Implements: -

Method name: public void addScore(int score)
Parameter (int score): The score to be added.
Return value: None.
Description: Adds the specified score to the users score.
Pre-condition: The game been initialized.
Post-condition: The score has been updated.
Called by: The World class collisionDetection() method.
Calls: None.
Implements: 6.1.2.11 Showing high score

Method name: public int getScore()
Parameter (int score): The score to be added.
Return value(int): Returns the current score.
Description: Returns the current score for the user.
Pre-condition: The game been initialized.
Post-condition: The score has been returned.
Called by: The Graphics class drawWorld() method.
Calls: None.
Implements: 6.1.2.11 Showing high score

Method name: public boolean hasMaxItems()
Return value(boolean): Is true if the user has filled up his/hers items stock.
Description: Checks if the user has filled up his/hers item stock.
Pre-condition: The game been initialized.
Post-condition: Has returned true or false.
Called by: The World class collisionDetection() method.
Calls: None.
Implements: 6.1.2.3 Collect special item

Method name: public void addItem()
Return value: None.
Description: Adds one more item to the users stock of items.
Pre-condition: The game been initialized.
Post-condition: The new item has been added.
Called by: The World class collisionDetection() method.
Calls: None.
Implements: 6.1.2.3 Collect special item

Method name: public void removeItem()
Return value: None.
Description: Removes one more item to the users stock of items.
Pre-condition: The game been initialized.
Post-condition: One item has been removed.
Called by: The World class collisionDetection() method.
Calls: None.
Implements: 6.1.2.4 Use special item

Method name: public int getCollectedItems()
Return value(int): Returns the amount of items the user has in stock.
Description: Returns the amount of items the has in stock.
Pre-condition: The game been initialized.
Post-condition: The amount of items has been returned.
Called by: The Graphics class drawWorld() method.
Calls: None.
Implements: -

Method name: public boolean isAlive()
Return value(boolean): Returns true if the character is alive.
Description: Returns true if the character is alive.
Pre-condition: The game been initialized.
Post-condition: The caller knows if the character is alive.
Called by: The Game class isGameOver() method.
Calls: None.
Implements: 6.1.1.2 Durance of play

Method name: public void setAlive(boolean isAlive)
Parameter (boolean isAlive): Is true if the player should be alive.
Return value: None.
Description: Can change the state of the character (if the character is dead or alive).
Pre-condition: The game been initialized.
Post-condition: The users alive state has been changed.
Called by: The World class collisionDetection() method.
Calls: None.
Implements: 6.1.1.2 Durance of play

Class Settings

Fields

Field name: private boolean fullscreen
Comments: Is true if the game is in fullscreen mode.

Field name: private boolean mute
Comments: Is true if the game is in mute mode.

Methods

Method name: public Settings()
Return value: None.
Description: The constructor of Settings class. Initiates the settings.
Pre-condition: The game been initialized.
Post-condition: The settings has been initialized.
Called by: The Game class constructor.
Calls: None.
Implements: -

Method name: public void load()
Return value: None.
Description: Loads settings from a file.
Pre-condition: The game been initialized.
Post-condition: The settings has been loaded from file.
Called by: The Game class constructor.
Calls: None.
Implements: -

Method name: public void save()
Return value: None.
Description: Saves settings to a file.
Pre-condition: The game been initialized.
Post-condition: The settings has been saved to a file.
Called by: The Settings class setFullscreen(boolean fullscreen) and setMute(boolean mute).
Calls: None.
Implements: -

Method name: public boolean getFullscreen()
Return value(boolean): Returns true if the game is in fullscreen mode.
Description: Checks if the game is in fullscreen mode.
Pre-condition: The game been initialized.
Post-condition: The calling method knows if the game is in fullscreen.
Called by: Graphic class draw methods.
Calls: None.
Implements: 6.1.1.3 Screen size

Method name: public void setFullscreen(boolean fullscreen)
Parameter (boolean fullscreen): The value to set the fullscreen to, false for window mode.
Return value: None.
Description: Sets the fullscreen variable to the specified value.
Pre-condition: The game been initialized.
Post-condition: The fullscreen value has been changed to the new value.
Called by: menuLoop() in the Game class.
Calls: None.
Implements: -

Method name: public boolean getMute()
Parameter: None
Return value (boolean): Returns the value of the mute settings..
Description: The method returns the value of the mute settings.
Pre-condition: The game been initialized.
Post-condition: The mute value has been returned.
Called by: menuLoop() in the Game class.
Calls: None.
Implements: 6.1.1.4 Appearance of sound

Method name: public void setMute(boolean mute)
Parameter (boolean mute): The value to set the mute to, true for mute.
Return value: None.
Description: Sets the mute variable to the specified value.
Pre-condition: The game been initialized.
Post-condition: The mute value has been changed to the new value.
Called by: menuLoop() in the Game class.
Calls: None.
Implements: 6.1.1.4 Appearance of sound

Class World

Fields

Field name: private Player player
Comment: The player object.

Field name: private LinkedList<Entity> entityList
Comment: Contains all entity's in the world, both blocks, items and the player.

Field name: private LinkedList<Block> blockList
Comment: Contains all blocks in the world.

Field name: private LinkedList<Item> itemList
Comment: Contains all items in the world.

Field name: private String t_collectedItem
Comment: Contains a path to the corresponding texture.

Field name: private String t_itemPlaceHolder
Comment: Contains a path to the corresponding texture.

Field name: private String t_rotateLeft
Comment: Contains a path to the corresponding texture.

Field name: private String t_rotateRight
Comment: Contains a path to the corresponding texture.

Field name: private String t_speedup
Comment: Contains a path to the corresponding texture.

Field name: private String t_gameover
Comment: Contains a path to the corresponding texture.

Field name: private OpenALWave s_speedup
Comment: Contains a path to the corresponding audio file.

Field name: private OpenALWave s_highScore;
Comment: Contains a path to the corresponding audio file.

Field name: private OpenALWave s_gameOver;
Comment: Contains a path to the corresponding audio file.

Field name: private int difficultylevel
Comment: Keeps track of the current difficulty level.

Field name: private int gravity
Comment: The constant gravity for the game.

Methods

Method name: public World(Graphics graphics)
Parameter (Graphics): A reference to the graphic object.
Return value: None
Description: The constructor for the class. It initializes all world objects and the player.
Pre-condition: The Game class has been created.
Post-condition: All world objects has been initialized and set-up.
Called by: The Game class constructor.
Calls: None
Implements: -

Method name: public LinkedList<Entity> getEntityList()
Parameter: None
Return value (LinkedList): Returns the entity list.
Description: Returns the list of entity's placed in the world.
Pre-condition: The world object has been initialized.
Post-condition: The entity list has been returned.
Called by: drawObjects() in the Graphics class.
Calls: None
Implements: -

Method name: public Player getPlayer()
Parameter: None

Return value (Player): Returns a reference to the player object.
Description: Returns the reference to the player.
Pre-condition: The player object has been created.
Post-condition: The player reference has been returned.
Called by: actOnInput() and isGameOver() in the Game class.
Calls: None
Implements: -

Method name: public void updateWorld()
Parameter: None
Return value: None
Description: Updates all the worlds objects, like positions, speed etc in the world. It also keeps track of when to increase the difficulty level.
Pre-condition: The gameLoop()-method is running,
Post-condition: The objects has been updated in the world.
Called by: gameLoop()-method in the Game class.
Calls: update()-method for each object in the world, collisionDetection()-method, rotateCamera(), speedUp and createNewItem()-methods.
Implements:
6.1.2.5 Screen movement
6.1.2.12 Flip function

Method name: public void collisionDetection()
Parameter: None
Return value: None
Description: Check each object against the player object to handle collisions in the game. It also make sure that items are collected if they collide with the player.
Pre-condition: The gameLoop()-method is running,
Post-condition: The objects has been updated in the world.
Called by: updateWorld()-method.
Calls: Each objects move methods (setX, setY, getX, getY).
Implements: 6.1.2.6 Ability to jump through blocks

Method name: public void blockGenerator(Block block)
Parameter (Block): The block to be moved on the screen to a new position.
Return value: None
Description: The method moves used blocks to a new position above the screen for reuse.
Pre-condition: The gameLoop()-method is running,
Post-condition: The objects has been updated in the world.
Called by: updateWorld()-method.
Calls: Each objects move methods (setX, setY, getX, getY).
Implements: -

Method name: public void removeItem()
Parameter: None
Return value: None
Description: The method removes an item from the world, or typically, the item and entity list.
Pre-condition: The gameLoop()-method is running,

Post-condition: The objects has been removed from the world.
Called by: collisionDetection()-method.
Calls: None.
Implements: 6.1.2.3 Collect special item

Method name: public void createNewItem()
Parameter: None
Return value: None
Description: The method creates a new item in the world at a random position.
Pre-condition: The gameLoop()-method is running and it's time for a new item to be created.
Post-condition: The objects has been created and positioned in the world.
Called by: updateWorld()-method.
Calls: None.
Implements: -

Method name: public void speedup()
Parameter: None
Return value: None
Description: The method increases the motion speed of the world screen.
Pre-condition: The gameLoop()-method is running and it's time for a speedup.
Post-condition: The screen has speeded up.
Called by: updateWorld()-method.
Calls: None.
Implements:
6.1.2.8 Screen speed
6.1.2.9 Screen speed increase indication

5.6 Package Diagram