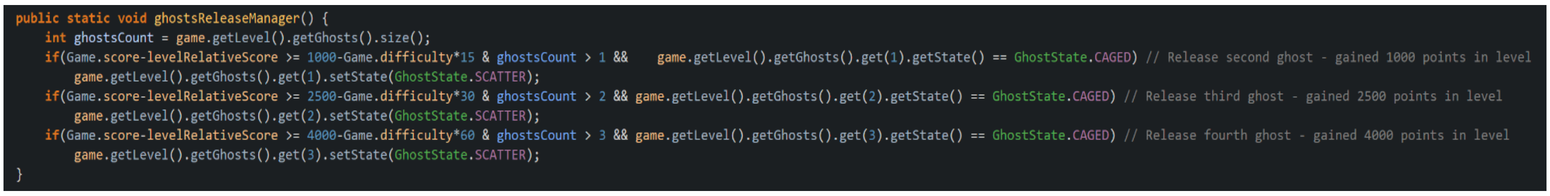
Principles of Object Oriented Programming – Assignment 4 Implementation notes and UML chart

מגישים:
איתי בוגנים, 305278384
סהר ויה, 205583453

Description:

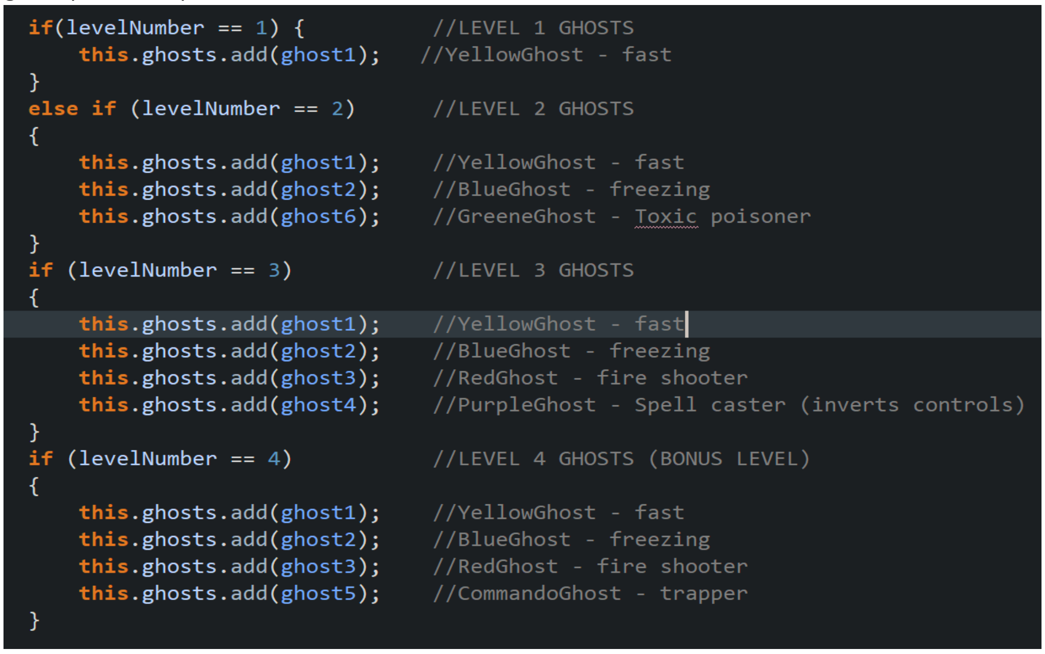
A few additions has been made over the demands of the current assignment in order to make the game more attractive and interesting for the end-user.
We tried giving the game A Retro Arcade machine feel by using some pixelated components and fonts and by using original Arcade Machines sounds that we gathered.
The game additions will be explained and details in the attached bonus.pdf document and the implementation of the different part will be explained here.
One of the additions is Game Difficulty which effects a few aspects of the game play and in involved in the ghosts release timings.
In order to make the game passable, in each level the ghost will release after a certain points step relative to the same level. (ex. In difficulty medium the second ghost (if exists) will release after the user managed to collect 880 points).

This is how we calculate when to release each ghost:



```java
public static void ghostsReleaseManager() {
    int ghostsCount = game.getLevel().getGhosts().size();
    if(Game.score-levelRelativeScore >= 1000-Game.difficulty*15 & ghostsCount > 1 &&    game.getLevel().getGhosts().get(1).getState() == GhostState.CAGED) // Release second ghost - gained 1000 points in level
        game.getLevel().getGhosts().get(1).setState(GhostState.SCATTER);
    if(Game.score-levelRelativeScore >= 2500-Game.difficulty*30 & ghostsCount > 2 && game.getLevel().getGhosts().get(2).getState() == GhostState.CAGED) // Release third ghost - gained 2500 points in level
        game.getLevel().getGhosts().get(2).setState(GhostState.SCATTER);
    if(Game.score-levelRelativeScore >= 4000-Game.difficulty*60 & ghostsCount > 3 && game.getLevel().getGhosts().get(3).getState() == GhostState.CAGED) // Release fourth ghost - gained 4000 points in level
        game.getLevel().getGhosts().get(3).setState(GhostState.SCATTER);
}
```

Each level has its own constant placed ghosts that are added to the board gradually by the level number, where in level 2, 3 and 4 (Bonus level) there is between 2-4 guest bonus implemented ghosts. This is the full ghost per level specification:

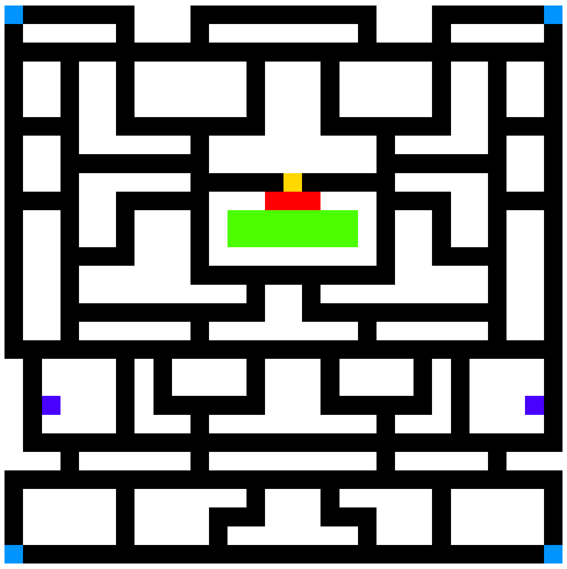


```java
if(levelNumber == 1) {          //LEVEL 1 GHOSTS
    this.ghosts.add(ghost1);   //YellowGhost - fast
}
else if (levelNumber == 2)      //LEVEL 2 GHOSTS
{
    this.ghosts.add(ghost1);    //YellowGhost - fast
    this.ghosts.add(ghost2);    //BlueGhost - freezing
    this.ghosts.add(ghost6);    //GreeneGhost - Toxic poisoner
}
if (levelNumber == 3)           //LEVEL 3 GHOSTS
{
    this.ghosts.add(ghost1);    //YellowGhost - fast
    this.ghosts.add(ghost2);    //BlueGhost - freezing
    this.ghosts.add(ghost3);    //RedGhost - fire shooter
    this.ghosts.add(ghost4);    //PurpleGhost - Spell caster (inverts controls)
}
if (levelNumber == 4)           //LEVEL 4 GHOSTS (BONUS LEVEL)
{
    this.ghosts.add(ghost1);    //YellowGhost - fast
    this.ghosts.add(ghost2);    //BlueGhost - freezing
    this.ghosts.add(ghost3);    //RedGhost - fire shooter
    this.ghosts.add(ghost5);    //CommandoGhost - trapper
}
```

IMPOTRANT NOTE: In order to ease the checking of the different levels we added a Developer Key Listener (Not for end-user use) – you can switch level programmatically **by pressing CTRL+P** while in level game-play.

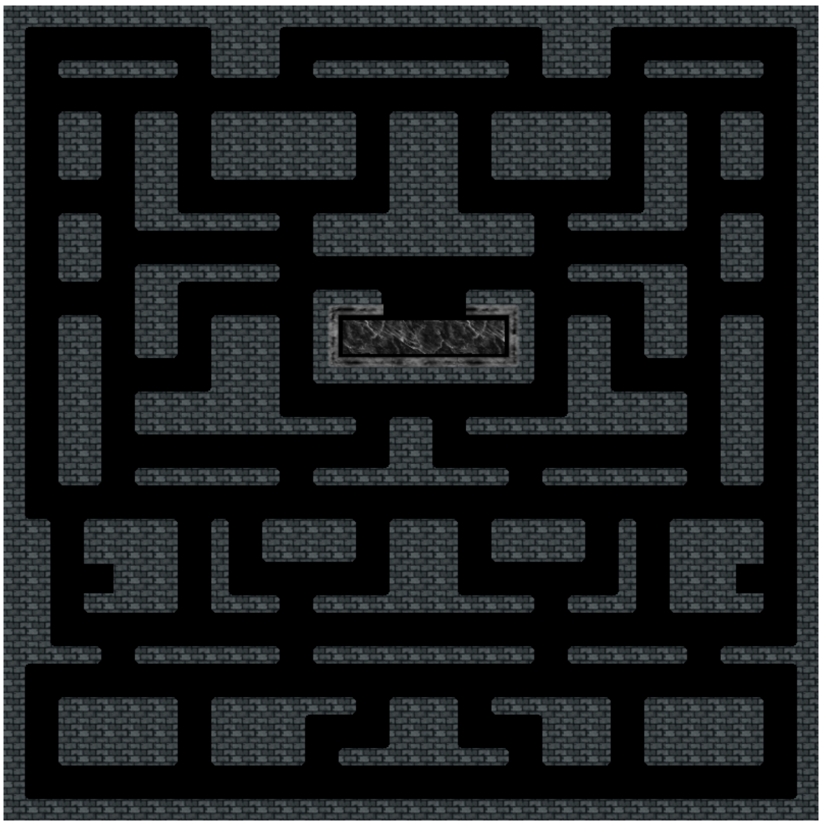
Board Implementation: The boards are read (Using the Maze Loader class) from a 32x32 pixels images in which every pixel is colored in a different color according to the current Tile property.
The board is represented by a 2D matrix of Tile types that has a different Tile Type property as read from the board logic image during game initialization.
Each Tile has an adjacency list of closest neighbors thus creating an adjacency matrix so that the board can be easily be represented as an Un-Directed Graph.

Maze behind logic image example: (Enlarged – original size is 32x32px)

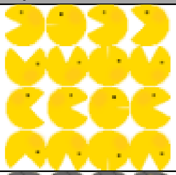
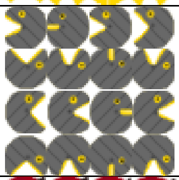


Board representation:
Black pixels – Paths
White Pixels – Walls
Green pixels – Ghosts Spawn
Red pixels – Door Tiles
Blue pixels – Board corners
Purple pixels – Portal positions (will be detailed in the bonus.pdf document).
Yellow pixel – Initial Pacman position.

The output for that logic image is: (Sized 800x800px)



Pacman Type sprites: The different Pacman types are as requested in the assignment demands, the sprites which represents each Pacman are changed and are as follows:

Pacman Type	Sprite
Normal Pacman	
Shielded Pacman	
Evil Pacman	

Interaction and encounters with all requested ghost are as in the assignment demands. When Pacman is in GOD-MODE (After eating an energy pill) a Sword sprite will appear on the Pacman character sprite:

Winning a level and gaining points

To win a level the Pacman character must collect all of the level COINS.
A LEVEL CLEARED notification will be presented in front of the player and he will have the option to continue to the next level by pressing SPACE or he can stop the game by pressing ESC (Those options and relevant keys to press are also notified on screen to the player). Each coin collected will gain the player 10 points.
Additional points can be achieved by collecting other additional items, detailed in the next sections.

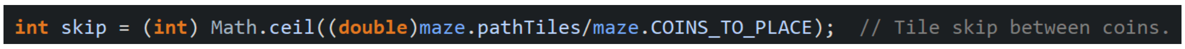
Collectable item Distribution and regeneration:

Collectable name	Amounts	Sprite
Coin	240 each level	
Energy Pill	4 each level – regenerates after all collected (will be detailed in bonus.pdf)	
Pineapple	**2**-level 1, **4**-level 2, **5**-level 3, **6**-level4	
Grapes	**2**-level 1, **4**-level 2, **5**-level 3, **6**-level4	
Strawberry	**1**-level 1, **2**-level 2, **3**-level 3, **4**-level4	
Bananas	**0**-level 1, **1**-level 2, **2**-level 3, **3**-level4	
Summoner (will be detailed in bonus.pdf)	One each time - regenerates	

*Other collectable amounts will be detailed in bonus.pdf

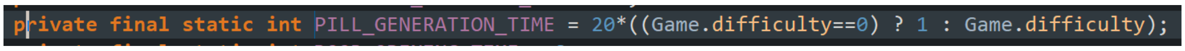
<u>Collectable items placements and details:</u> The game consists of different collectable items that are placed by the following logic:

Coins – Each board consists of 240 coins that are initially placed by skipping tiles according to the following formula: (COINS_TO_PLACE = 240)



```
int skip = (int) Math.ceil((double)maze.pathTiles/maze.COINS_TO_PLACE);  // Tile skip between coins.
```

Energy pills – Are placed in the board corners – once all energy pills are eaten by Pacman, they will regenerate according to:



```
private final static int PILL_GENERATION_TIME = 20*((Game.difficulty==0) ? 1 : Game.difficulty);
```

If the player ate an energy pill it will turn <u>GOD-MODE, in which the ghosts are vulnerable</u> and can be eaten by Pacman.

Fruits – Each level has a fixed amount of fruits for each kind that generates after 10 seconds from the moment the game started and start disposing gradually until completely vanished from board or eaten prior to vanishing.

The entire fruit collection that is relevant to the current level will then regenerate after 7 seconds from vanishing in order to give the player the opportunity to gain more score.

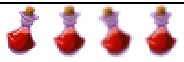
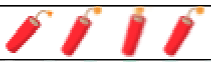
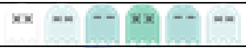
When a fruit collectable appears on board it will flicker with the following symbol: 🎁

Other collectables – will be details in the bonus.pdf document.

<u>Complete scoring table:</u>

As an item is collected its specific score will be added to the game score and a number of the score collected will pop for a short time in the place it was collected in.

The scoring is by the following table:

Collectable name	Score	Sprite
Coin	10 points	
Energy Pill	50 points	
Pineapple	100 points	
Grapes	200 points	
Strawberry	300 points	
Bananas	400 points	
Elixir (will be detailed in bonus.pdf)	10 points	
Summoner (will be detailed in bonus.pdf)	5000 points	
TNT (will be detailed in bonus.pdf)	3000 points	
Killing a ghost	1000 points	
Being frozen	-10 points	
Becoming dizzy (will be detailed in bonus.pdf)	-100 points	
Becoming poisoned (will be detailed in bonus.pdf)	-5 points per second	

<u>Ghost Movement and AI:</u> The ghosts move differently for each state that they are currently in, meaning the ghost behave differently according their current state:

```
public enum GhostState { CAGED, SCATTER, RANDOM, CHASE, FLEE, DIED, FROZEN, DISAPEAR; }
```

And also according to the current Pacman entity state:

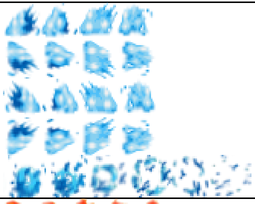
```
public enum PacmanState { NORMAL, FREEZE, DEAD, DIZZY, POISONED };
```

The ghosts can chase each certain Tile on the board representation (specifically the tile Pacman is currently holding) by using the game board Graph property and by using Breadth-First-Search Graph search to find the shortest path from every path defined tile to another.

<u>Ghosts FLEE mode -</u> If the Pacman entity ate an energy pill all ghost go into FLEE mode in which they are vulnerable and can be eaten.
If a ghost has been eaten it will increase speed, change animation and chase its initial spawn Tile to regenerate. After regeneration the ghost is “Angry” and will start chasing Pacman as soon as it is back to normal.

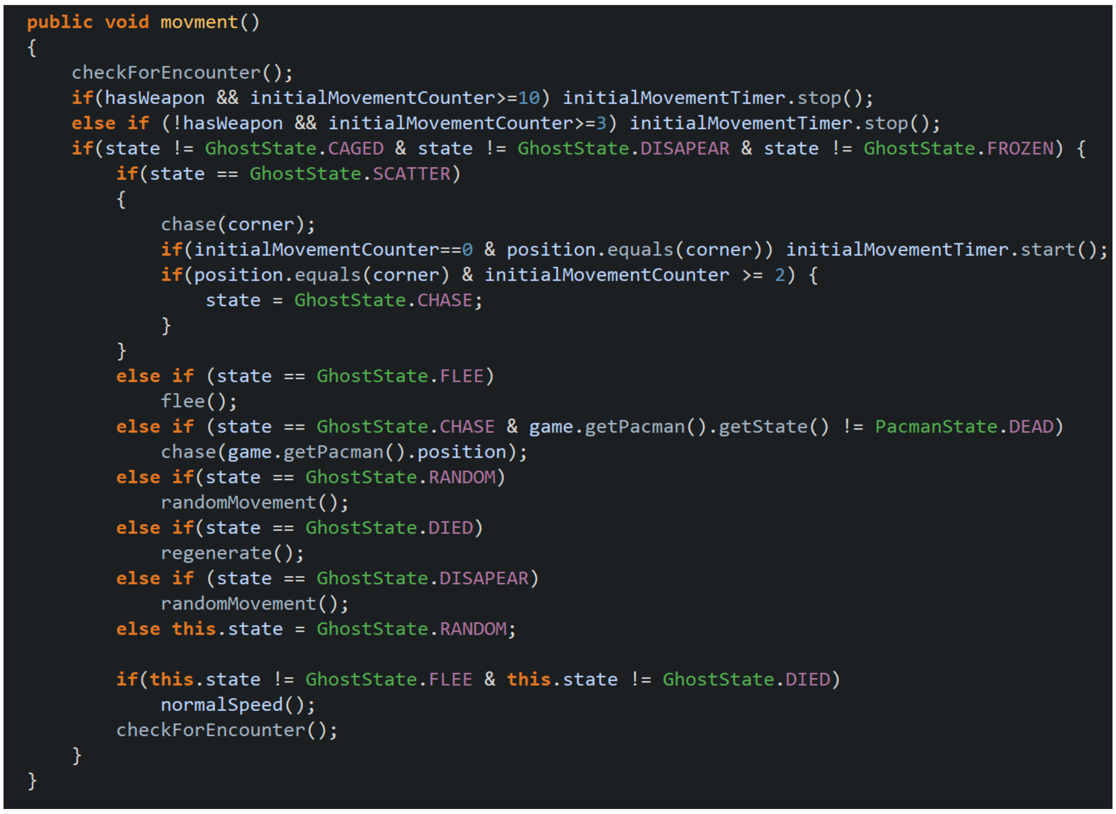
<u>Ghosts Arming –</u> If A ghost has a weapon it will arm itself after 2 second from the initial scatter to the board corners. As a ghost is armed it will flicker with the following symbol: ⊕
If a player has managed to kill the ghost before it was armed, after the ghost will regenerate it will try to reach board corner again to arm itself until successful.

<u>The ghost details as requested in the assignment:</u> (NOTE: ghost colors are changed from the original assignment colors but the properties are the same as the demands)

Ghost name	Ghost property	Sprite	Weapon	Weapon sprite	Weapon property
Yellow Ghost	Fast		NONE	NONE	NONE
Blue Ghost	Fast, Freezes		Watersplash		Freezes, medium speed
Red Ghost	Slow		Fireball		Kills all Pacman types, Fast

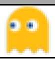
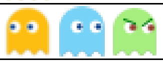
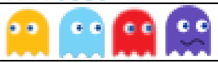
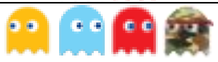
<u>More ghosts will be detailed in the bonus.pdf document.</u>

<u>Ghosts State switching -</u> In order to make the game more convenient, The ghosts switch movement states in a timed manner and according to the difficulty the player chose between Random state and Chasing state. (Hardest difficulty – the ghost are in a constant chasing behavior). If a ghost managed to kill the Pacman entity it will immediately go in Scatter state and chase one fixed board corner in order to allow the player to regenerate properly.
Also, If a ghost has managed to put an effect on Pacman, it will immediately change its state to random in order to not overload Pacman with the effect.
This is the ghosts main movement method which is used to govern the ghosts behavior in each state:



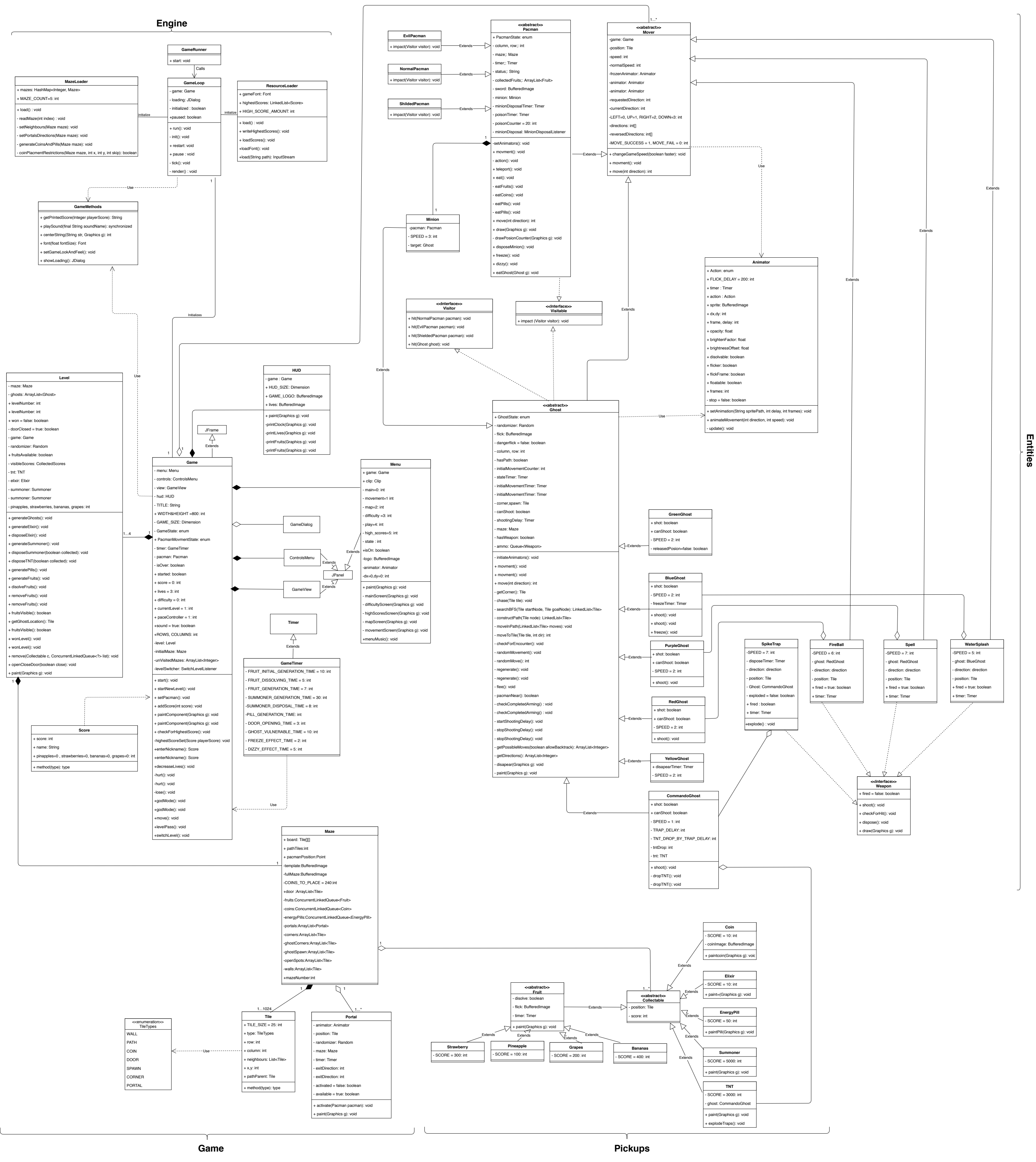
```
public void movment()
{
    checkForEncounter();
    if(hasWeapon && initialMovementCounter>=10) initialMovementTimer.stop();
    else if (!hasWeapon && initialMovementCounter>=3) initialMovementTimer.stop();
    if(state != GhostState.CAGED & state != GhostState.DISAPEAR & state != GhostState.FROZEN) {
        if(state == GhostState.SCATTER)
        {
            chase(corner);
            if(initialMovementCounter==0 & position.equals(corner)) initialMovementTimer.start();
            if(position.equals(corner) & initialMovementCounter >= 2) {
                state = GhostState.CHASE;
            }
        }
        else if (state == GhostState.FLEE)
            flee();
        else if (state == GhostState.CHASE & game.getPacman().getState() != PacmanState.DEAD)
            chase(game.getPacman().position);
        else if(state == GhostState.RANDOM)
            randomMovement();
        else if(state == GhostState.DIED)
            regenerate();
        else if (state == GhostState.DISAPEAR)
            randomMovement();
        else this.state = GhostState.RANDOM;

        if(this.state != GhostState.FLEE & this.state != GhostState.DIED)
            normalSpeed();
        checkForEncounter();
    }
}
```

<u>Ghosts Level Distribution, Releasing conditions And possible ways for a player to loose:</u>
Ghosts will be released from cage after a player reaches a certain amount of points each level(Relative to the level).
As mentioned the ghosts points amount to release from cage is game difficulty dependent (More will be detailed in bonus.pdf). This is the per level ghost distribution:

Level #	Ghosts	Sprites	Ways for a player to loose (Pacman dies)
1	Yellow		Interaction with ghost
2	Yellow, Blue, Green		Poisoning by green ghost
3	Yellow, Blue, Red, Purple		Red ghost interaction
4 (BONUS)	Yellow, Blue, Red, Commando		Yellow, Red Ghosts interaction or commando ghost traps

<u>More ghosts that are not detailed here will be detailed in the bonus.pdf document.</u>

PAC-MAN Adventures - OOP Assignment 4 UML Chart



Text