Game API Specification Document

Use Cases:

1. User can choose their lives from 1 to 5
2. By default, user will have 3 lives
3. User can choose spawning position for pacman
4. User can choose the speed for ghost from slow, medium, and fast
5. User can press the arrow key on keyboard to control the movement of pacman
6. All passageway should be the width of the pacman
7. Pacman can eat small dots, which is 10 points each
8. There are approx. 240 small dots in total
9. Pacman can eat large dots, which is 50 points each
10. There are 4 larger blinking dots near the 4 corners
11. The box where ghosts start should be in the middle
12. Large dots can increase value if they collide with dark blue or flashing ghosts
13. Pacman will lose one live if they collide with normal ghost
14. Pacman can eat fruit, which is 100 points each
15. Pacman will win and go to the next level if he eats all the dots before losing all lives
16. Game will get over once pac man loses all lives
17. There needs to be one exit from one side of the game to the opposite side of the game
18. Pacman stops moving when hitting a wall if no direction input

Design Decision:

1. Different ghosts have different behaviors: (random, chase pacman)

a. Ghost → 3 directions to choose → Takes the one which takes to PacMan (based on PacMan's position)

b. Ghost → 2 directions to choose → Take the one from where it didn't come.

c. Ghost → 1 direction to choose → No Choice. Revert on the same path back.

2. Every 12 Seconds a fruit appear at a random location, the fruit disappears when pacman eats it
3. Each Non moving Object takes one block (1 unit space in canvas)
4. Walls are created from blocks
5. Difficulty → Increase Speed with each level. Reduce the time of Big Dot with each level.

6. User Customization/Extensibility → User can select pacman's spawning position. Select the lives at start and the speed of the ghosts.
7. Each level → Backend to store the state of the game and return the required state for the next level . For example increase the speed of ghosts,reduce invincible mode time.
8. Backend needs to send the updated list of Objects to the frontend.

Possible API Endpoints:

1. /setGame // For customization
2. /updateGameStore // Call every 0.1 second
3. /init // Restart
4. /changeDirection // Call whenever the user presses a direction key

API Documentation:

endpoint	method	payload	response	description

/setGame	POST	{
		X,
		y,
		lives,
		speed // 'slow','med','fast'
		}
			OK	Set customizable items (Spawning pt, speed, lives)

/updateGameStore	POST	none	{
			Score,
			Lives,
			Difficulty,
			isGameOver,
			PacMan,
			Fruit,
			NormalDot[],
			BigDot[],
			Ghost[]
			collectedFruits[](boolean)
			}
				Update the Game object every 0.1 second

/init	POST	none	gameStoreObject	Restart the game after game over
/changeDirection	POST	direction	OK	Get the arrow key that user pressed

---

1. POST /setGame

DESC: Get the user pac-man location for starting the game.
BACKEND :set the location of the pac-man in GameStore and update Matrix.

REQUEST
{
	“pos”: <Point>,
	“lives”: integer,
	“speed”: 1 - Slow, 2 - Medium, 3 - Fast
}

RESPONSE
{
	"errCode" : 0,
	"data" : <string>
},
{
	"errCode" : 1,
	"data" : <string>
}

2. POST /updateGameStore

DESC: Update the Game Objects.
BACKEND: return the updated game objects..

RESPONSE
{
	"errCode" : 0,

"data" : <GameStoreObj> // score,lives,isGameOver, pacMan,Ghosts[]
normalDots[];BigDot[] bigDots,fruits,
collectedFruits[](bool);
},
{
"errCode" : 1,
"data" : <string> [a string of fail reason]
}

3. POST /init

DESC: Restart game after game over.
BACKEND: return the initial game objects..

RESPONSE
{
"errCode" : 0,
"data" : <GameStoreObj> // score,lives,isGameOver, pacMan,Ghosts[]
normalDots[];BigDot[] bigDots,fruits;
},
{
"errCode" : 1,
"data" : <string> [a string of fail reason]
}

4. POST /changeDirection

DESC:get the arrow key which the user pressed.
BACKEND: return the initial game objects..

REQUEST
{
”direction”: <integer>
}
RESPONSE
{
"errCode" : 0,
"data" : <string>
},
{
"errCode" : 1,

"data" : <string> [a string of fail reason]
}

Models and Classes:

Objects

AMovingObjects:
{
String type;
Point position;
Point speed; // Will only be in (x, 0) or (0, y)
MovementStrategy movementStrategy;
CollisionStrategy collisionStrategy;
}

PacMan extends AMovingObjects
Ghost extends AMovingObjects
TwoEyes extends AMoving Objects

AInteractableObjects:
{
Point position;
String type;
CollisionStrategy collisionStrategy;
Int score;

}

NormalDot extends AInteractableObjects // 10 score each
BigDot extends AInteractableObjects // 50 score
Fruit extends AInteractableObjects //100 score
WallObj extends AInteractableObjects // 0 score

Game Store

Final AObjects[] … // A backup of everything in the game for initialization

```
Class GameStore {
        Int score;
        Int lives;
        Int difficulty;
        Boolean[] collectedFruits; // cherry, orange, banana, apple, avocado
        Boolean isGameOver; // if lives == 0 return true.
        Int fruitTimeInterval; // Every 12 seconds fruit appears. 120
        Int invincibleTimeout; // Activated when pac man eats a big dot 7 seconds
        Boolean isInvincible; // Pac man can collide with ghosts
        Int combo; // How many ghosts collided in one big dot period

        // Every object in the game <Should be stored as PropertyListeners?>
        PacMan pacMan;
        Fruit fruit;
        NormalDot[] normalDots;
        BigDot[] bigDots;
        Ghost[] ghosts;
        WallObject[] wall; // Static positions
        Point pacManStart;
        Point ghostStart;
        Int nextDirection //1 -- R , 2 -- L , 3 -- U & 4 -- D
}
```

Movement Strategy

```
MovementStrategy {

}

        PacManNormal {


                updatePosition(Point position){
```

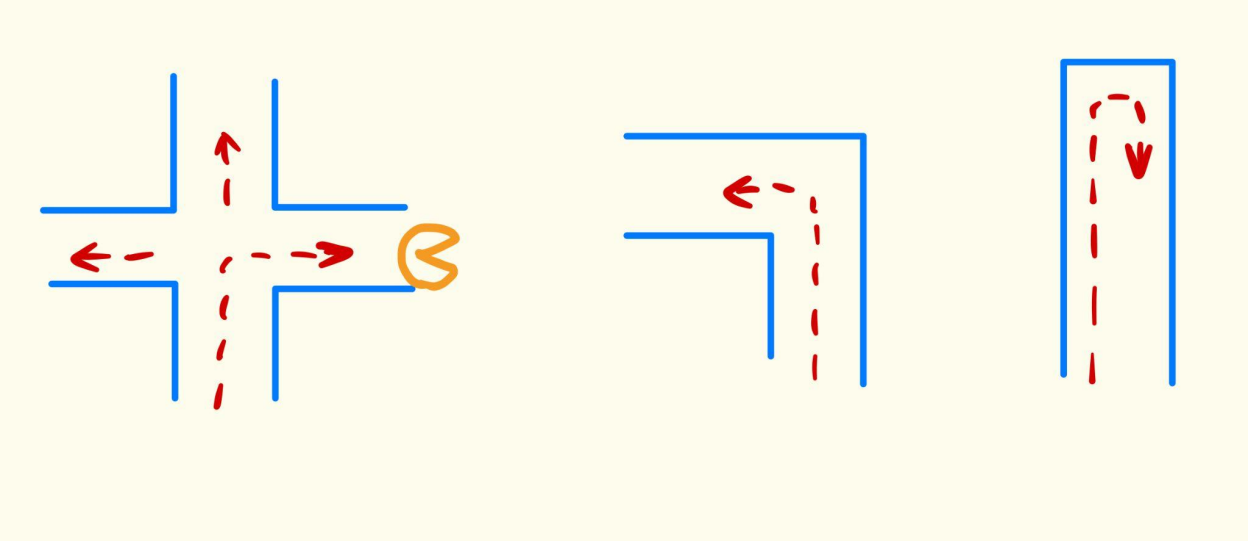
```
            //according to the keypress change the speed and add to current
            position
        }

}
```

GhostNormal {

Ghost ---> 3 directions to choose → Takes the one which **takes to** PacMan (based on PacMan's position)

Ghost ---> 2 directions to choose → Take the one from where it didn't come.

Ghost ---> 1 direction to choose → No Choice.



```
        updatePosition(Point position){

            //Update according to the above.
        }

}
```

GhostRandom {

```
        //Ghost moves in a random direction.

}

GhostRandomNormalMix {

        //Ghost moves in a random direction,
        //or chase Pacman based on a random variable

}

GhostPacManEatsBigDot {

    updateColour{

        colour:Blue
    }

    Ghost  ---> 3 directions to choose → Takes the one which takes away fromPacMan (based on PacMan's position)
    Ghost  ---> 2 directions to choose → Take the one from where it didn't come. (Maybe it does have to turn around?)
    Ghost ---> 1 direction to choose → No Choice.

    updatePosition(Point position){

        //Update according to the above.
    }


}

GhostPacManEatsBigDotBlinking {
    //Initiate when timeout < 2 seconds
    updateColour{

        colour:Blue/White every update // Can FE do this?
    }
```

Ghost ---> 3 directions to choose → Takes the one which **takes away** fromPacMan (based on PacMan's position)

Ghost ---> 2 directions to choose → Take the one from where it didn't come. (Maybe it does have to turn around?)

Ghost ---> 1 direction to choose → No Choice.

```
updatePosition(Point position){

        //Update according to the above.
}
}
```

```
TwoEyeStrategy {


//Go to ghostStart Position
updatePosition(Point position){

        //Update according to the above.
}
}
```

Collision Strategy

```
CollisionStrategy{
}
```

```
PacManWallCollision{

        //No Update to position
}
GhostWallCollision{

        //Change direction
}
GhostPacManCollision{

        //Eats PacMan, Restart Game
```

```
        //Lives should decrease.
        //Ghosts go to starting position
        //Pac-Man goes to starting position

}

GhostInvinciblePacManCollision{

        //PacMan eats ghosts,
        //Call TwoEyeStrategy for Ghosts.
        //Ghosts fly straight back to starting position
        // Increase points by combo*200

}

InteractableObjectsCollision {
                // PacMan gets reward
                // object removed
                // Call corresponding function(
                Normal dots: none // 10 points

                Big dots: PacMan enter invincible mode, start
                invincibleTimeout, set isInvincible to true
                Change Ghosts Movement Strategy to
                GhostPacManEatsBigDot
                Fruit: add fruit to game store // 50 points
                )

}
```