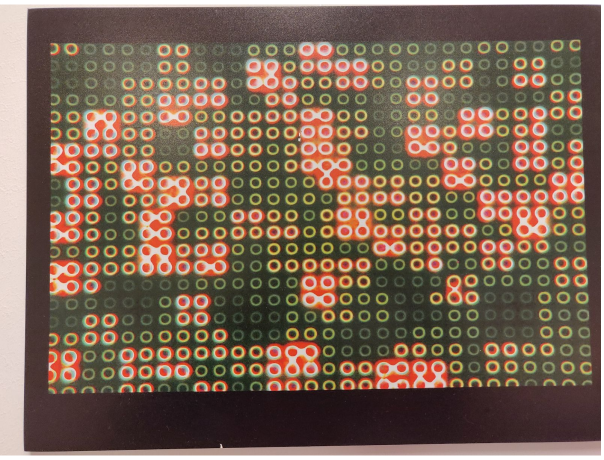
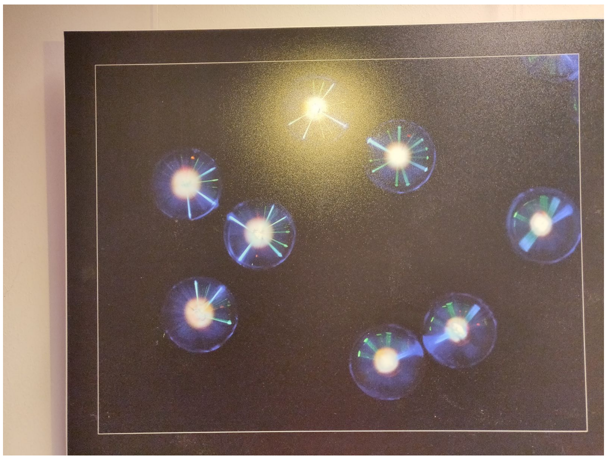
A-Maze-ing

By Jonathan Matarazzi

Synopsis

For this programming project I chose to use two artworks that I found in the NanoLab.





My program creates a maze made out of the circles in the left artwork and adds a character made out of one of the circles from the right artwork. The user can control where the character should go. The user can also destroy walls. The maze is generated using the Depth First Search algorithm and the character finds its path to its destination by using the A* algorithm.

Usage and interactions

There are two ways of interaction with the program:

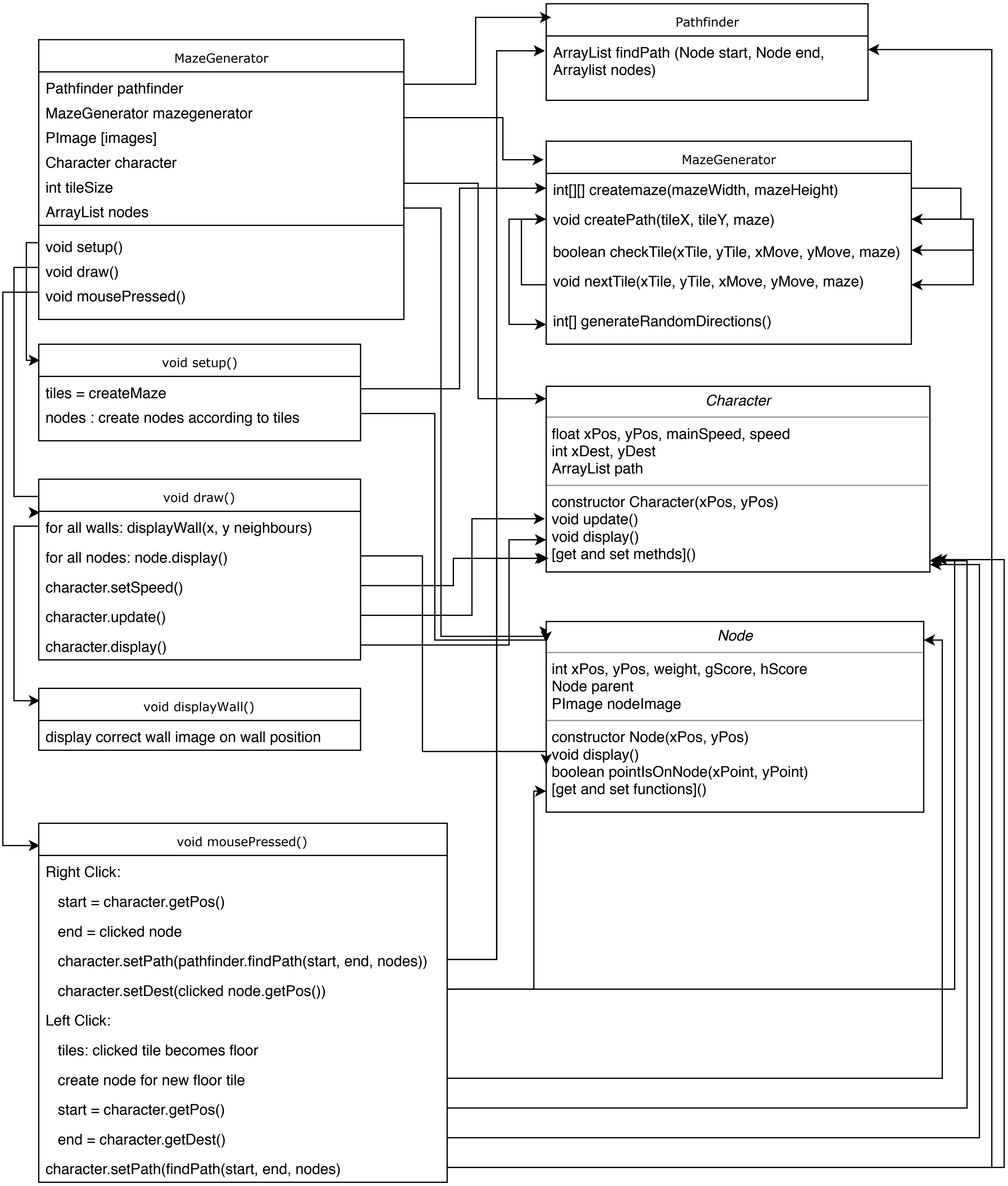
- Right clicking on a wall tile will destroy the clicked wall tile and replace it with a floor tile.
- Left clicking on a floor tile will set the clicked floor tile as the destination of the character.

Architecture

The program contains 4 classes:

- Pathfinder: A class that contains the pathfinding algorithm
    - This class contains one method called findPath that finds a path from a given start position to a given end position.
- MazeGenerator: A class that contains the maze generating algorithm
    - This class contains one public class called createMaze that creates a maze using the depth first search algorithm divided into separate private methods.
- Node: A class for all the nodes (floor tiles), it stores information about the node and contains methods for displaying the node and for setting en getting variables.
- Character: A class for the character, it stores information like the position, destination and speed. It contains methods for updating the position to move along the path, displaying the character and setting and getting variables.

The program itself also contains an important function called “displayWall” that handles the drawing of the walls to get nice connected walls using the correct images.



MazeGenerator

Pathfinder pathfinder
MazeGenerator mazegenerator
PImage [images]
Character character
int tileSize
ArrayList nodes

void setup()
void draw()
void mousePressed()

void setup()

tiles = createMaze
nodes : create nodes according to tiles

void draw()

for all walls: displayWall(x, y neighbours)
for all nodes: node.display()
character.setSpeed()
character.update()
character.display()

void displayWall()

display correct wall image on wall position

void mousePressed()

Right Click:
start = character.getPos()
end = clicked node
character.setPath(pathfinder.findPath(start, end, nodes))
character.setDest(clicked node.getPos())
Left Click:
tiles: clicked tile becomes floor
create node for new floor tile
start = character.getPos()
end = character.getDest()
character.setPath(findPath(start, end, nodes))

Pathfinder

ArrayList findPath (Node start, Node end, Arraylist nodes)

MazeGenerator

int[][] createmaze(mazeWidth, mazeHeight)
void createPath(tileX, tileY, maze)
boolean checkTile(xTile, yTile, xMove, yMove, maze)
void nextTile(xTile, yTile, xMove, yMove, maze)
int[] generateRandomDirections()

Character

float xPos, yPos, mainSpeed, speed
int xDest, yDest
ArrayList path

constructor Character(xPos, yPos)
void update()
void display()
[get and set methds]()

Node

int xPos, yPos, weight, gScore, hScore
Node parent
PImage nodeImage

constructor Node(xPos, yPos)
void display()
boolean pointIsOnNode(xPoint, yPoint)
[get and set functions]()