Emily Zhang

Design Proposal

The name of my term project is “Boomer”. It is a 2D strategic tile-based game. Player(s) can win by strategically dropping bombs to wrap the opponent between blocks. Player(s) can increase the chance of winning by collecting power-ups.

Competitive Analysis

I have seen similar games like Balloono and Bomberman and my game is a combination of both. Player(s) will be moving on a tile-based map, containing both solid blocks (non-destroyable) and explodable blocks (can be destroyed with bombs). The solid blocks will have fixed positions while the explodable blocks will be randomly generated each time the game starts. All players will also be spawned at a random corner each time. After the players place down a bomb, the bomb will explode with up to a certain range. All players will start off with an explosion range of 1 block and a limit of 1 bomb at a time. As the game progresses, players can collect power-ups underneath the explodable blocks to increase range of explosion, the number of bombs they have, and speed.

Different from Bomberman, where the map consists of multiple levels and mobs that move randomly within the map, my version of the game consists of only one level, but with an AI that tracks the player and places bombs strategically based on the player’s position. Different from Balloono, where the players are playing against each other online, my game consists of a single player mode, where the opponent will be a game AI, and a local multiplayer mode, with a maximum of four players. While a lot of tile-based games online only allows tile-to-tile movement, my game will allow players to move freely within the map constraints. I am also considering adding an extra feature that allows players to create and save their own maps to make the game more engaging and fun to play.

Structural Plan

The finalized project will consist of multiple files with multiple classes. There will be a main file containing the main game loop, a settings file initiating all the game dimensions and constants, a sprites file containing all sprite classes, and an AI file consisting of the search algorithms. The sprites file contains all the sprite classes, including the player, the AI, tiles, bombs, explosions, and etc.

Algorithmic Plan

The trickiest part of the project is the random map generation and game AI. The random map generation consists of a 2D list of letters and symbols representing different objects on the tile map. For example, letter ‘S’ represents solid blocks and letter “E” represents explodable blocks. The initial map will consist of all empty spaces, represented by “-”. Then, by using random.randrange, explodable blocks will be generated with an 80% chance. Then, the explodable blocks are set in place, the map will be destructively modified to include the solid blocks at certain locations. After that, the four corners will be cleared out to have enough space for the player to move when spawned. After constructing the 2D list, another function will translate the map and actually spawn the objects accordingly. The items map follows the same idea and logic, there is a 30% chance for an item to be placed under an explodable block.

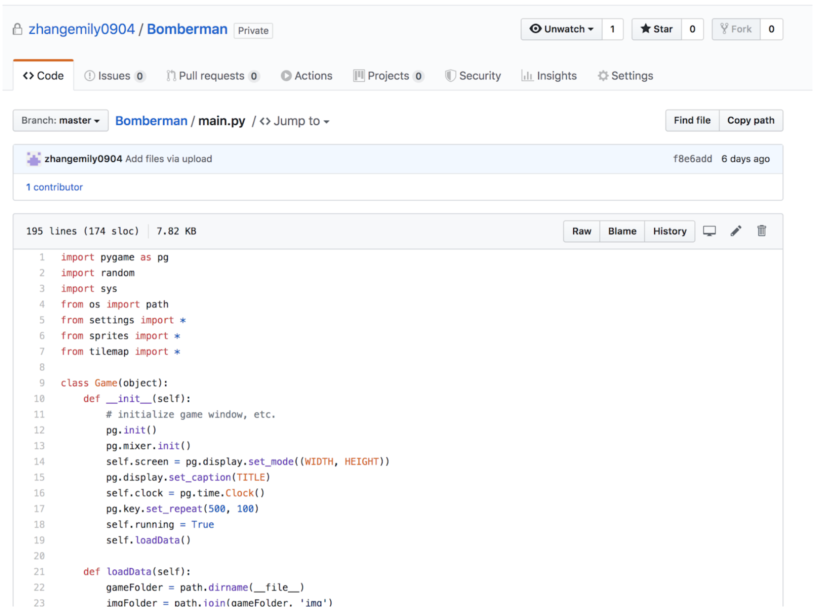
The game AI utilizes the A* search function to find the closest route to approach the player. Everytime the AI encounters an explodable block on the way toward the player, it will place a bomb and then find the closest safe place to hide from the explosion. When it is 2 blocks away from the player, it will place a bomb. When it is too close to the player, it will run away.

Timeline Plan

- **Tp0**: A working random generated map and basic game plays. The player will be allowed to move freely, place bombs, and collect power-ups.
- **Tp1**: A working game AI and a finished game play that allows the player to win/lose the game.
- **Tp2**: Reach MVP. Improve AI movement. Implement local multiplayer mode and allow players to create unique maps.
- **Tp3**: Improve player movement. Complete multiple screens (start screen, help screen, game screen, end screen).

Version Control Plan

I am backing up my code on github.



zhangemily0904 Add files via upload f8e6add 6 days ago

1 contributor

195 lines (174 sloc) 7.82 KB

```
import pygame as pg
import random
import sys
from os import path
from settings import *
from sprites import *
from tilemap import *

class Game(object):
    def __init__(self):
        # initialize game window, etc.
        pg.init()
        pg.mixer.init()
        self.screen = pg.display.set_mode((WIDTH, HEIGHT))
        pg.display.set_caption(TITLE)
        self.clock = pg.time.Clock()
        pg.key.set_repeat(500, 100)
        self.running = True
        self.loadData()

    def loadData(self):
        gameFolder = path.dirname(__file__)
        imgFolder = path.join(gameFolder, 'img')
```

Module List

- Pygame

Emily Zhang

TP2 Update

I implemented a start screen, game over screen, help screen, and a game control screen (player can hit tab during the game to see what keys they've set for movements). For local multiplayer mode, players can set their own keys for movements and bomb placements. At the end of the game, players can see their rank.

After implementing single player mode, where the player plays against an AI, and local multiplayer mode, where players can play against each other, I decide to implement sockets in my game to allow people to play the game online, which will make my game similar to Balloono. I also added background music to my game to make it more engaging and interesting.

I changed how I structured my code by making a separate file for each sprite instead of putting them all into one file. For example, I now have a players file, an AI file, a bombs file, etc. I believe this helps me organize the code in a better way and makes navigation and debugging easier.