Design Document

# Main Approach

I use OOP to finish this project.

## Save

I use pickle module to solve the save problem. It seems very easy, but it’s really not. When I used pygame to develop this project, the pygame.surface cannot be pickled. Therefore, I have to do it manually and it took me a lot of time to figure it out because when I was saving it, it’s a surface, but when I loaded it back, it’s no longer a surface.

## Map

To generate the map, I first generate a 2d list that contains integer. This integer indicates what kind of the tile is. Then, I generate a real map that contains Tiles

## Configure

This is a file to store constants.

## Character, Player and Enemies

Player and Enemies are inherited from Characters.

Character solves the problem of the image figure, movements, what direction to face

Players can deploy bombs and have lives

Enemies can randomly move

# User Interface

## Button Design

Initially, I tried to use real button but I found it so exhausting. And I changed it to this way:

1. Use PS to draw back
2. Set an area as effective click area to trigger events

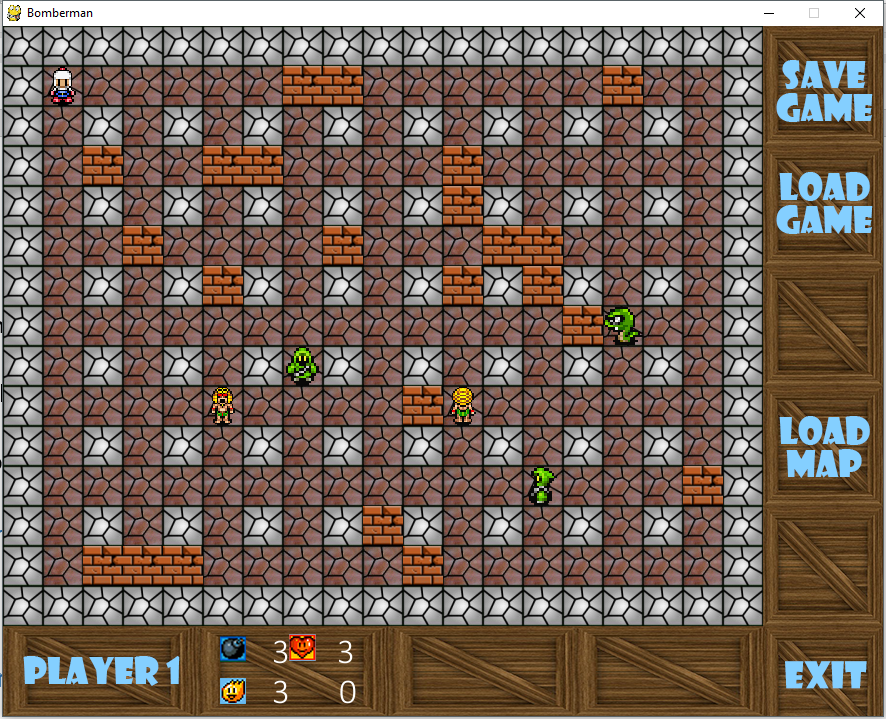


Actually, I found this way looks more elegant and easier to implement so I design in this way.

If you notice, there are a lot original graphic work in “/resources/ images/ Originals”

I really put a lot of effort to work on this.

## Menu Design



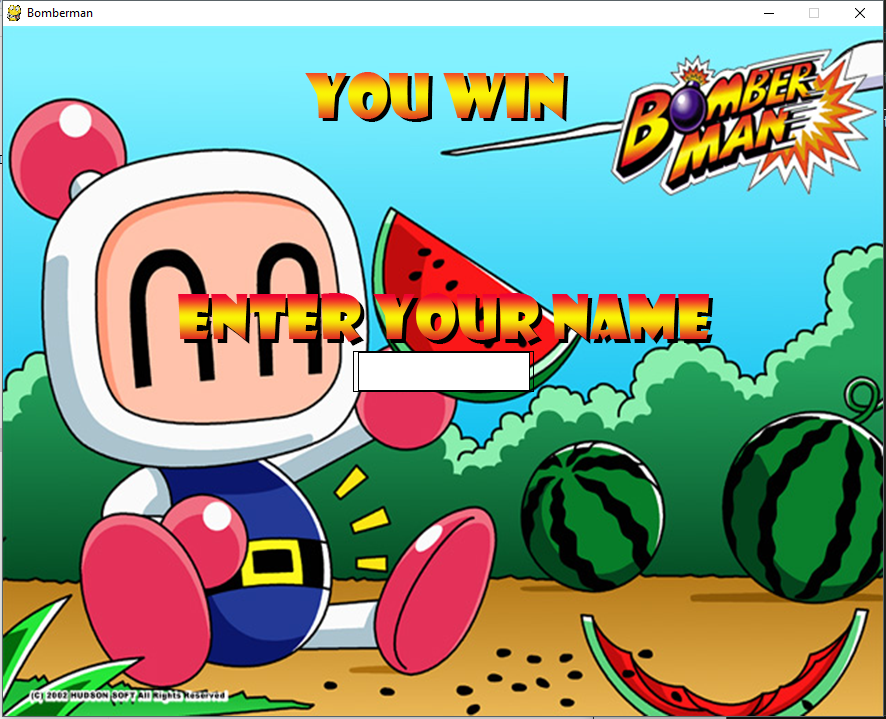
As shown, I used some box to indicate the boundary these button. Users are allowed to load map, saving or loading the game and go back

The bottom shows the players status, the max bombs, power range, and lives.

## Win / Lost

Basically, every page can go back to its parent page. What is worth to mention in Win Page, I designed a really sophisticated dialog box.

As shown, when a user wins the game in single mode, he can input his name on high scores. In the dialog box, he can input his name and it can in-time display and won’t go out of boundary. And it won’t crush and exit when it’s empty.



## Jump Logic

I put some efforts to predict users’ behavior so that I can make the game page jump more logically. For example, when a user wins and input his name, he probably wants to view score board. So when the user hits enter, it shows the score board; when a user tries to click the close button of the window, he probably wants to leave the game ASAP, so it’s unnecessary to go back to the title screen. It exits; however, when the user click the “exit” or “back” button, it will go back rather than exit.

## Controls

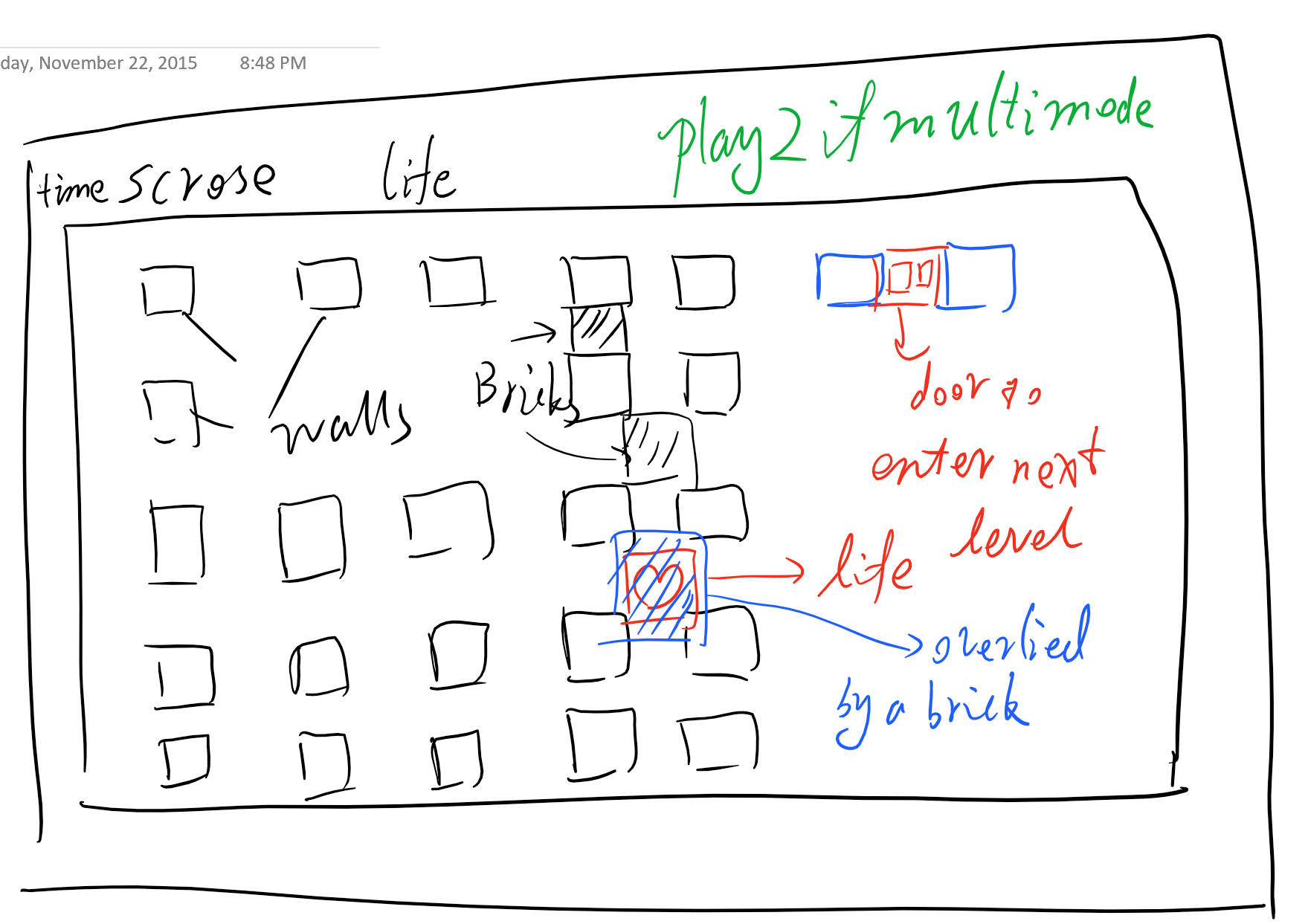
|  |  |
| --- | --- |
|  |  |

I used two sets of keyboard instruction to help user feel comfortable. It’s kind of weird if single mode uses the key instruction of player 1’s in multiplayer version. Keys are too close so I change it.

## Color Choice

I like Material Design, so I choose a lot of colors that contains a bit white, rather than so called hard color.

## Original Board Template



# Pities

There are some pities in this project. I have to demonstrate them. I wish I can make up some day.

# AI

Initially, I try to calculate all possible ways so that it’s possible to calculate the most efficient way (due to bricks, the shortest is not necessarily the most efficient way) to get the user in single player. However, it’s an exponential way and sometimes, it runs into “MemoError”. It took me too long to realize it’s my method problem.

Then, I try to implement a naïve AI, but running out of time… which is include in the zip file. You can click “init AI” to run it. It’s not very smart.

# Animations

Due to pygame limitation, I cannot play animation in the game… Manually implement it drives me crazy.

I probably next time choose another module rather than pygame.

## Sockets

I did put some effort to implement sockets. Due to the complexity of this game, it becomes rather annoying when I implemented sockets to transmit the object. It often goes wrong. My work not socket friendly to code…