Universidade do Porto

Faculdade de Engenharia

FEUP

Bomberman

Relatório Final

MIEIC-LPOO

Turma 3, Grupo 5

Rui Figueira – ei11021
Pedro Faria – ei11167

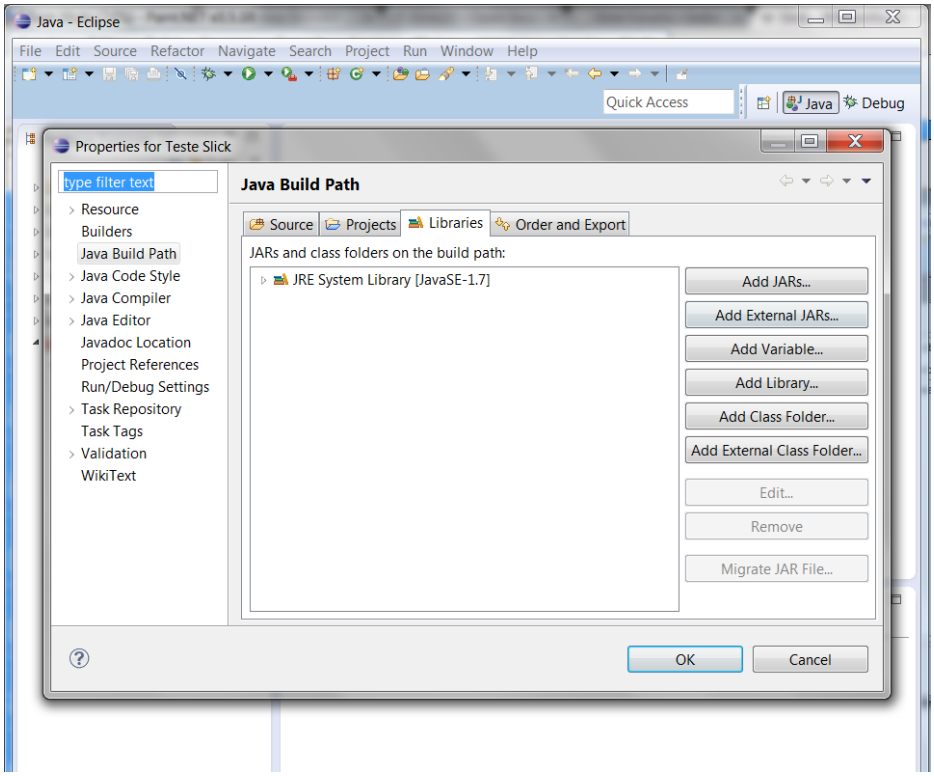
Introdução

Para o projecto final da disciplina, decidimos fazer uma remake do clássico jogo Bomberman. O jogador controla esta personagem que tem como objectivo destruir blocos até encontrar a saída do mapa, destruindo pelo seu caminho todos os monstros que tentem impedir o seu caminho. A saída estará escondida debaixo de um dos vários blocos destrutíveis do nível e ficará visualmente presente quando o bloco que a oculta for destruído.

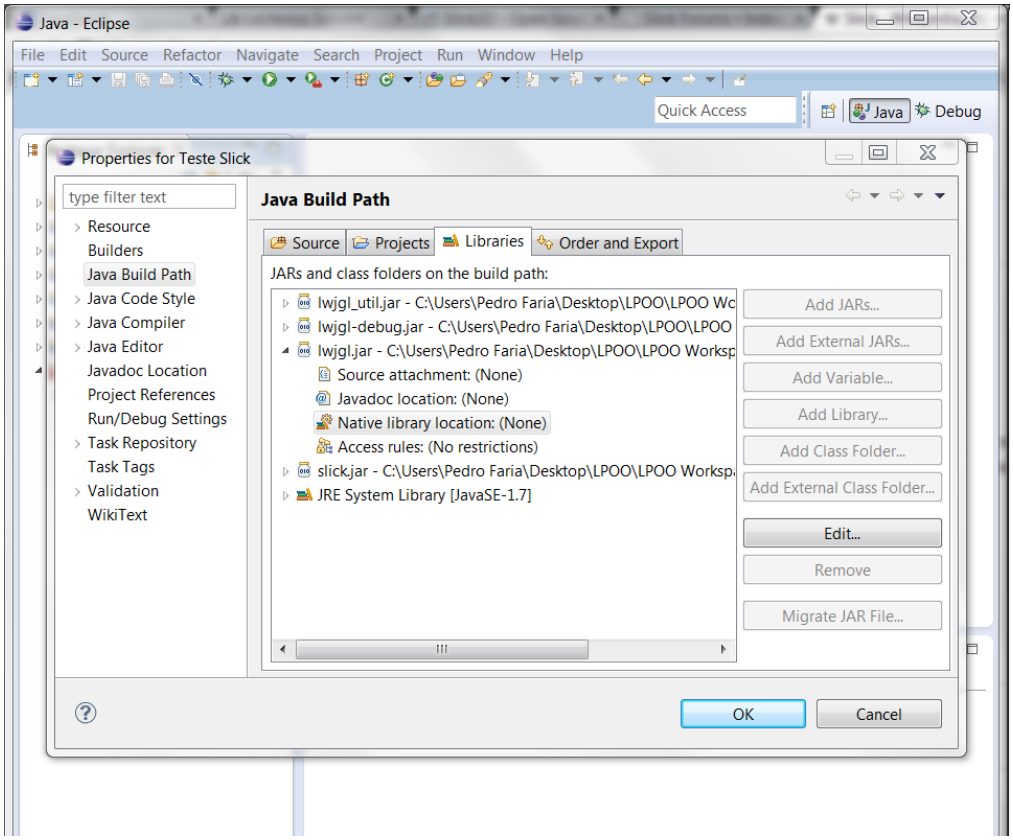
Manual de configuração

Neste projecto decidimos criar o jogo com a ajuda do Slick2D, uma biblioteca de “game development” de java.

Em anexo ao projecto estará a pasta com os .jar necessários para correr a aplicação. Para configurar basta ir à pasta do projecto no eclipse, escolher Properties>Java Build Path>Libraies>Add External JARs:



Escolher os seguintes .jar: lwjgl_util.jar, lwjgl-debug.jar, lwjgl.jar , jogg-0.0.7.jar, jorbis-0.0.15.jar, e slick.jar. Na lwjgl.jar é necessário escolher a Native library (escolher o SO respectivo, na pasta native) como demostra a figura:



Após configuração, basta executar no eclipse: Run As>Java Application>Initiate Game-bomberman.gui.

Descrição

Start Menu

O jogo inicia com um menu que dá a oportunidade ao jogador de escolher entre diferentes tipos de jogabilidade (”Campaign”, “Duel”). No Campaign, aparece a opção de começar um novo jogo com vários stages em que o jogador tem que descobrir a saída. No 2 players, o jogo é feito entre dois jogadores (localmente), num match entre os dois jogadores, até um dos dois ganhar.

Gameplay

O jogo é constituido por um player (Bomberman) que começa numa posição pré-definida do mapa, com uma bomba e 2 vidas. Os monstros são inicializados em posições aleatórias, o mesmo se aplica aos blocos do jogo. A saída é escondida aleatoriamente por baixo de um dos blocos do jogo. O Bomberman pode colocar bombas de acordo com os seus power-ups de bombas em posições livres do mapa. Estas bombas explodem passado um tempo pré-definido e tem um range associado, destruindo apenas a primeira interseção com objetos que encontrem no mapa ou então o range máximo no caso de não haver objetos. Os monstros movem-se aleatoriamente pelo mapa de acordo com os caminhos livres do mesmo. O jogo termina quando o player tiver descoberto a saída e todos os monstros tiverem sido mortos.

No modo 2 players, cada jogador começa na sua parte do ecrã, inicializados com 2 vidas. O objectivo é conseguir reduzir as vidas do outro jogador a 0 para ganhar o duelo.

Teclas Pré-defenidas

Player 1:

W, A, S, D / Up, Left, Down, Right arrows: Movimentação

Barra de espaços: Colocar bomba

Player 2:

I, J, K, L: Movimentação

O: Colocar bomba

GUI e Descrição dos Componentes

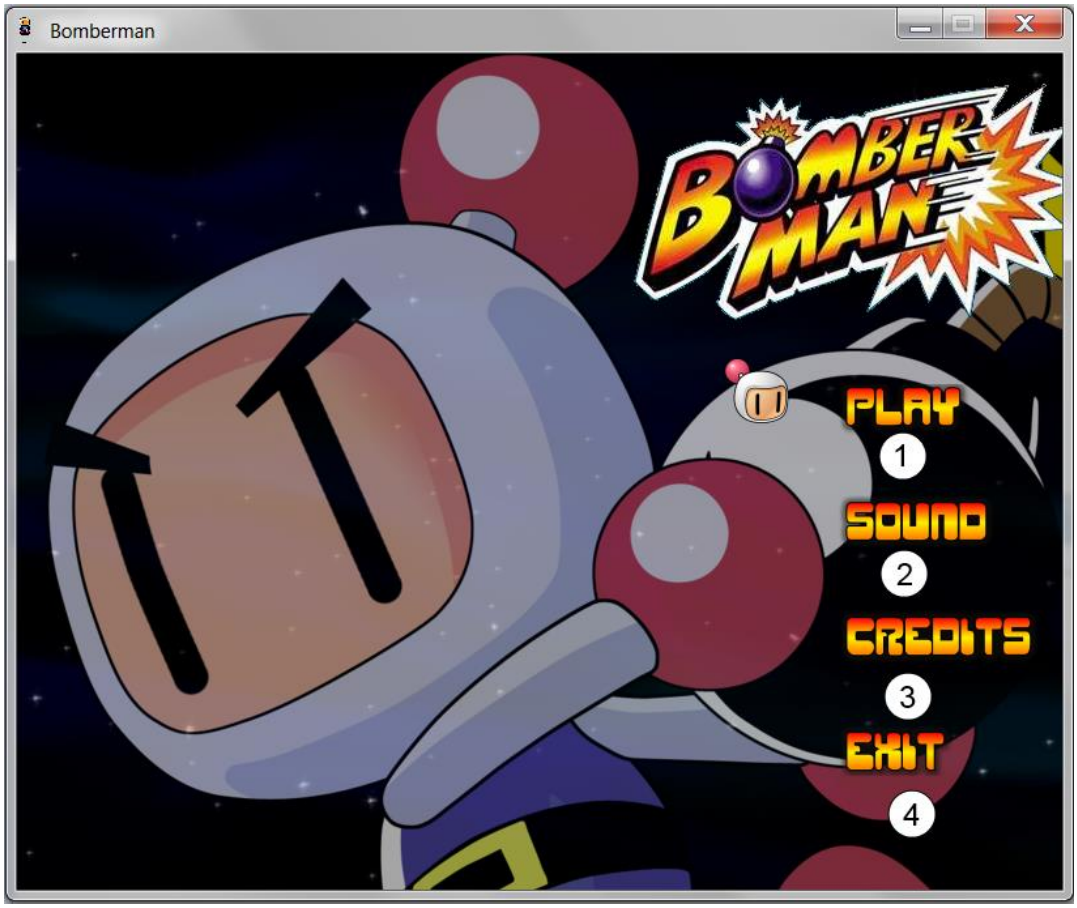


Figura 1- Imagem do menu principal do jogo

Menu principal

1-Play: Permite escolher o modo de jogo

- "Campaign" (1 player)
- "Duel" (2 players)

2-Sound: Permite ligar/desligar o som no jogo

3-Credits: Aparecem os créditos do jogo;

4-Exit: Fecha a aplicação.

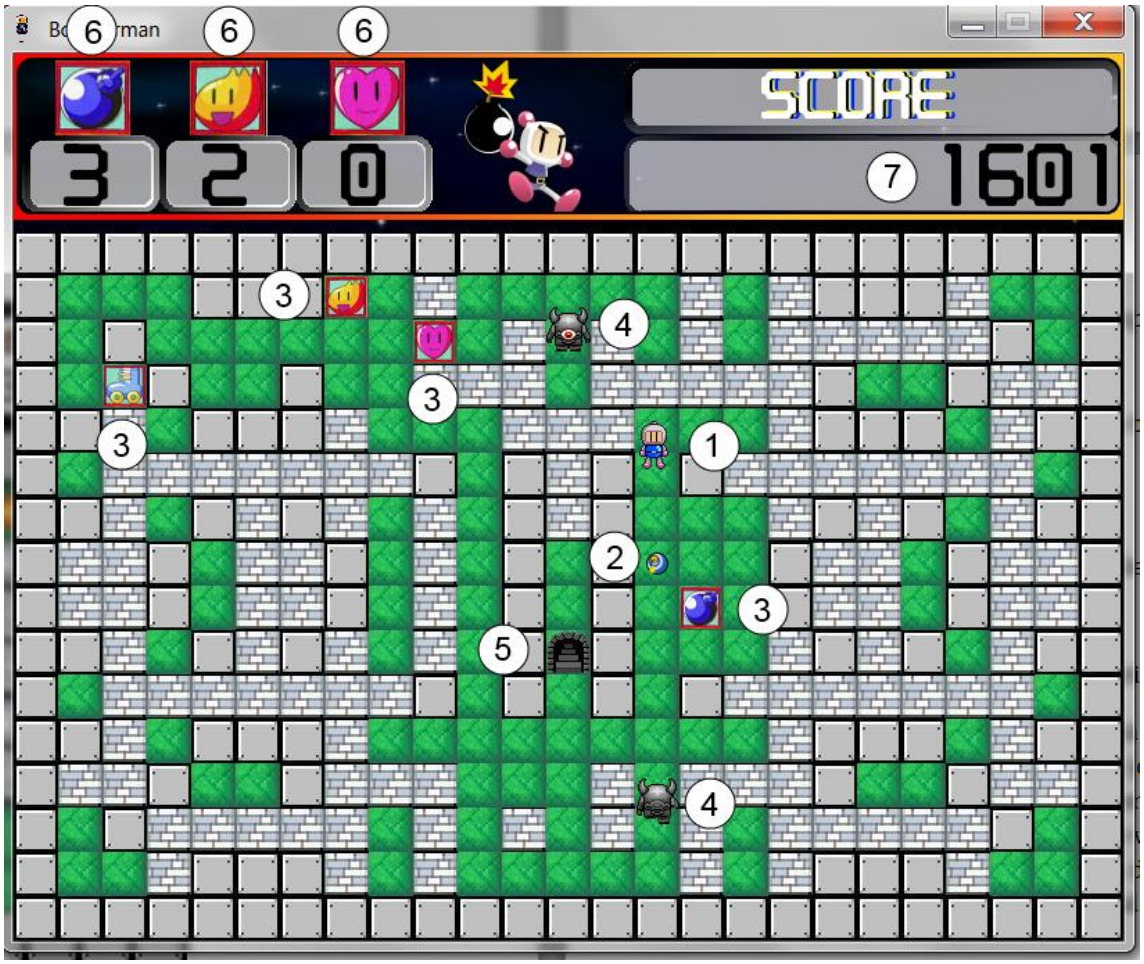


Figura 2 - Imagem do “Gameplay” de um nível

Painel de jogo

1-Bombarman (o jogador)

2-Bomba colocada

3-Power-ups

4-Inimigos

5-Saída

6-Quantidade de power-ups no jogador

7-Score

GamePlay

Power-ups

 **Fire Up:** Aumenta o raio de explosão das bombas em 1.

 **Bomb:** Aumenta o número de bombas que o jogador pode colocar simultaneamente em 1.

 **Skate:** Aumenta a velocidade de movimento do jogador.

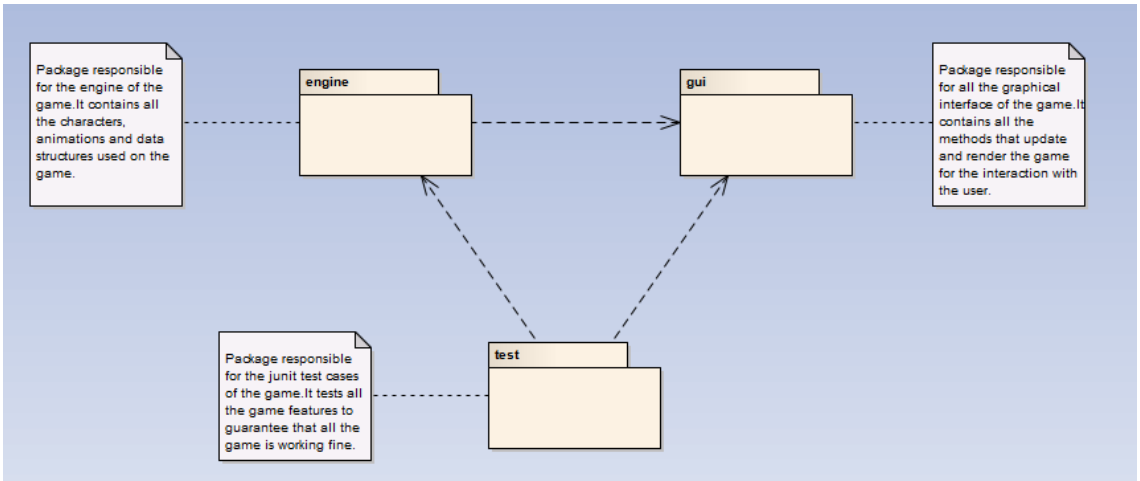
 **Skull:** Provoca um efeito negativo no bomberman. Pode fazer o seguinte:

- Diminuir o raio de explosão em 1;
- Diminuir a velocidade do jogador;
- Diminuir o número de bombas disponíveis em 1;
- Diminuir a quantidade de vidas em 1.
  (Nota: Em caso de o raio de explosão ou o número de bombas diminuir para 0, serão postos os valores em 1).

 **Saída:** Completa o Stage actual.

Concepção

Estrutura de Packages



Package	Descrição
Engine	Motor do jogo,com todas as estruturas de dados e funções para o respectivo funcionamento.
Gui	Interface gráfica do jogo.
Test	Package de testes unitários.

Bibliotecas Utilizadas	Finalidade
`org.newdawn.slick.AppGameContainer`	Usado para criar a aplicação.
`org.newdawn.slick.GameContainer`	Usado para criar um container para a aplicação.
`org.newdawn.slick.SlickException`	Usado para excpeções de slick 2D.
`org.newdawn.slick.state.StateBasedGame`	Implementa um State Based Game.Este possui vários states com as suas próprias propriedades.Os estados podem mudar entre si.
`org.newdawn.slick.Graphics`	Usado para a parte gráfica do jogo.
`org.newdawn.slick.Input`	Usado para receber input s do jogador.
`org.newdawn.slick.state.BasicGameState`	Estado basico de um State Based Game.
`org.newdawn.slick.Image`	Usado para criar imagens estáticas no jogo.
`org.newdawn.slick.Animation`	Usado para criar imagens dinâmicas no jogo.
`org.newdawn.slick.SpriteSheet`	Usado para criar spritesheets a serem usadas para a criação de animações.
`org.newdawn.slick.tiled.TiledMap`	Usado para ler e manipular tiled maps.

org.newdawn.slick.openal.Audio	Usado para criar o audio do jogo.
org.newdawn.slick.openal.AudioLoader	Usado para carregar os ficheiros audio.
org.newdawn.slick.util.ResourceLoader	Usado para carregar recuros diversos entre os quais,a música do jogo.
java.util.ArrayList	Usado para criar um array list.
java.util.List	Usado para criar uma list.
java.util.Random	Usado para geração aleatória de números.

Estrutura de Classes

Engine

Classe	Descrição
Bomb	Classe reponsável pela criação e manipulação de bombas usadas no jogo pelo bomberman.Possui as funçoes de manipulaçao e alteração das bombas.
Bomberman	Classe responsável pela criaçao e manipulação de todo o funcionamento do bomberman do jogo.
Level	Classe responsável pelo funcionamento dos níveis usadas ao longo do jogo.
Monster	Classe reponsável pela criação e manipulação de monstros usadas no level.
MusicEngine	Classe responsável pela criação e manipulação da música e efeitos sonoros do jogo.
PinkBomberman	Classe reponsável pela criação e manipulação de um segundo bomberman para o multiplayer.Possui um metodo de movimento próprio e as suas próprias animações.Extends Bomberman.
PowerUp	Classe responsável pelos powerUps.Possui as caracteristicas a serem guardadas por cada powerUp no level.

Gui

Classe	Descrição
Campaign	Classe responsável pelo modo campaign do jogo.
InitialMenu	Classe responsável pelo menu inicial do jogo.
InitiateGame	Classe responsável por iniciar o jogo e os seus estados.
PlayMenu	Classe responsável pelo menu play do jogo.
SoundMenu	Classe responsável pelo menu sound do jogo.
TwoPlayersMode	Classe responsável pelo modo multiplayer do jogo.

Test

Classe	Descrição
BombermanTest	Testes unitários ao funcionamento do bomberman.
BombTest	Testes unitários ao funcionamento da bomba.
LevelTest	Testes unitários ao funcionamento do nível.
MonsterTest	Testes unitários ao funcionamento do monstro.

Padrões de Desenho

Implementamos o padrão de desenho Singleton para a manipulação do audio do jogo, usando uma classe que apenas pode ter uma instância e que é usada transversalmente por várias classes para a manipulação da música do jogo. Apesar de só termos implementado este padrão de desenho, devido a não termos sentido uma necessidade de implementar outros na planificação do nosso projecto, achamos que se tivessemos abordado inicialmente os problemas de outra forma poderiamos ter tido mais facilidade no desenvolvimento do projecto ao verificar que podiamos utilizar outros padrões de desenho para a resoluçao desses problemas.

Testes

BombermanTest

Nome	Descrição
testFreeMovement	Testa o movimento do bomberman para um espaço vazio no nivel.
testMovementAgainstWall	Testa o movimento do bomberman contra uma parede.
testPuttingBomb	Testa se o bomberman poe uma bomba quando a space bar é carregada.
testPuttingMultipleBombs	Testa se o bomberman coloca varias bombas ao mesmo tempo.
testBombermanLosesALifeToAMonster	Testa se o número de vidas do bomberman diminui em 1 quando encontra um monstro.
testBombermanLosesALifeToABomb	Testa se o número de vidas do bomberman diminui em 1 quando encontra no raio de uma explosão de uma bomba.
testBombermanDies	Testa se o bomberman morre quando o seu número de vidas chega a 0;
testBombermanExitsWithMonstersAlive	Testa se o bomberman consegue passar de nível com monstros ainda vivos.
testBombermanExitsWithMonstersDead	Testa se o bomberman consegue passar de nível com os monstros todos mortos.
testBombermanCatchesSpeedPowerUp	Testa se a velocidade do bomberman aumenta quando apanha o speed powerUp.
testBombermanCatchesFirePowerUp	Testa se a o range das bombas do bomberman aumentam em 1 quando apanha o speed firePowerUp.
testBombermanCatchesBombPowerUp	Testa se o número de bombas do bomberman aumenta em 1 quando apanha o bomb powerUp.
testBombermanCatchesSkullPowerUp	Testa se a velocidade/número de bombas/velocidade/vidas do bomberman diminui quando apanha o skull powerUp.

BombTest

Nome	Descrição
testBombExplode	Testa se a bomba explode após o bomberman a por.
testBombRange	Testa se a explosao da bomba está de acordo com o seu range.
testBombDestroysDestructibleWall	Testa se a bomba destrói uma parede destrutivel.
testBombDoesntDestroyIndestructibleWall	Testa se a bomba não destrói uma parede indestrutivel.
testBombActivatesBomb	Testa se uma bomba detona outra bomba.

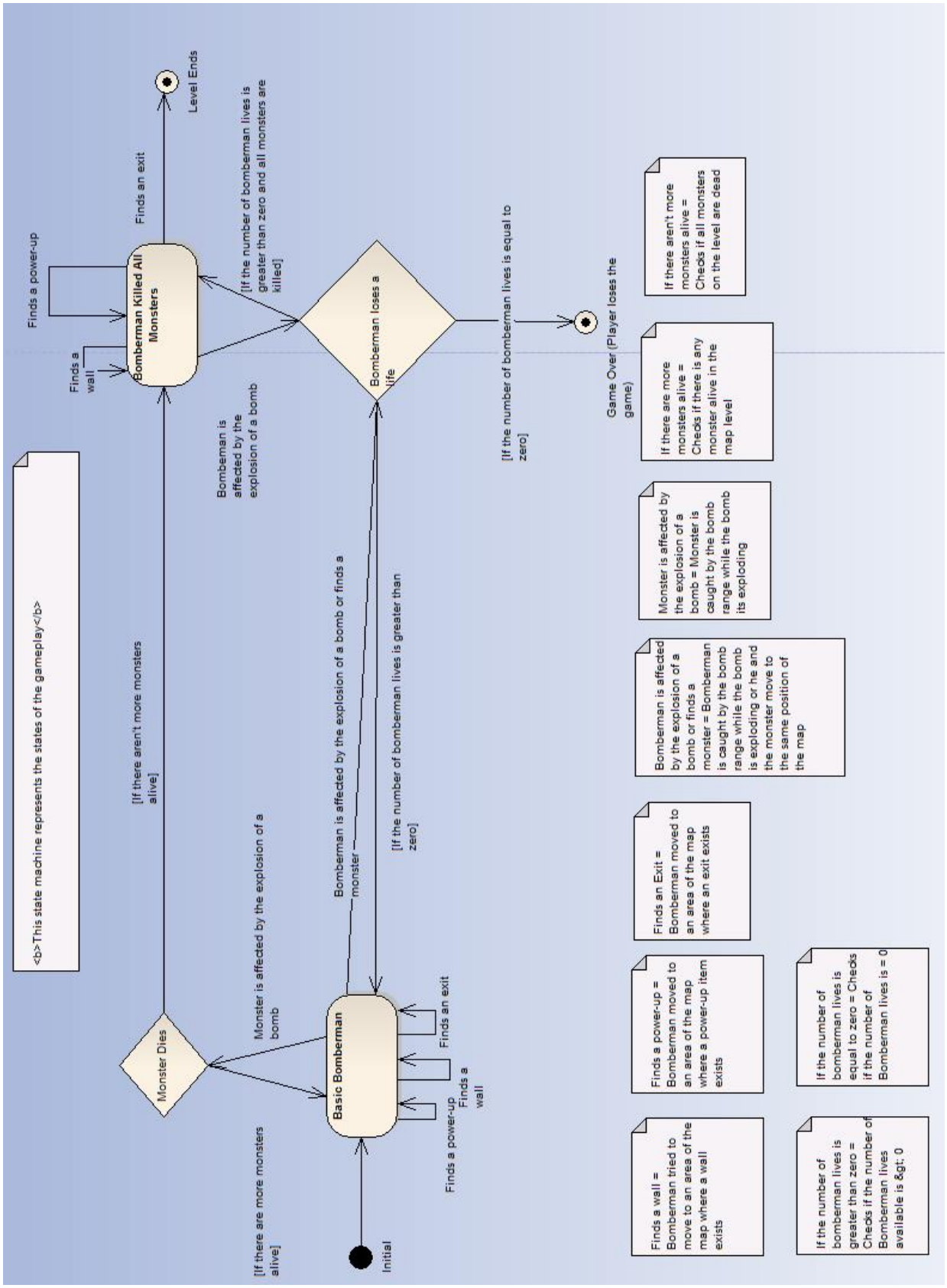
LevelTest

Nome	Descrição
testLevelEnd	Testa a condição de final do nível.
testChangeLevel	Testa a mudança de nível quando um nível acaba.

MonsterTest

Nome	Desrição
testFreeMonsterMovement	Testa o movimento de um monstro para um espaço vazio no nível.
testMovementAgainstWall	Testa o movimento de um monstro contra uma parede no nível.
testMonsterKillsBomberman	Testa de o monstro mata o bomberman.
testMonsterDies	Testa se o monstro morre para a explosão de uma bomba.
testMultipleMonsters	Testa a existência de múltiplos monstros no mesmo nível.

Comportamento do Bomberman



Conclusões

Comparando o nosso plano inicial com o que efectivamente implementamos neste jogo, achamos que fizemos um bom trabalho, conseguindo implementar todas as funcionalidades pretendidas em termos de engine do jogo e conseguindo trabalhar numa interface gráfica acima do esperado. Apesar de tudo, achamos que o jogo ainda apresenta várias possibilidades de melhoria, nomeadamente:

- A implementação de um sistema de high scores;
- O aperfeiçoamento do comportamento das bombas aquando da sua explosão;
- O aperfeiçoamento do comportamento sonoro no jogo;
- A implementaçao de um modo muliplayer online.

Ainda assim, achamos que conseguimos tornar o jogo suficientemente versátil com o apoio do Tiled para qualquer programador facilmente inserir novas funcionalidades e niveis no jogo, o que para nós foi uma boa prática de aplicação de refactoring aquando da avaliação intermédia do código. Em geral, achamos que desenvolvemos um bom projeto, mas como que em tudo, poderia ser melhorado com mais algum tempo de trabalho.