

Segundo Projeto: Castle Defense
MAC0346/6960 Programação para Jogos Digitais, 2019-2



Autores:

Alexandre L. Martins - 503780,
Arthur Pires da Fonseca - 10773096,
Édio Cerati Neto - 9762678,
Rubens Douglas Roccia - 9793590

Universidade de São Paulo - Dep. Ciência da Computação
Responsáveis: Prof. Dr. Fabio Kon e Prof. Ms. Wilson Kazuo Mizutani

São Paulo, 06 de Dezembro 2019

1. Apresentação (Ax)

- (A1) Atender o formato de entrega.*
- (A2) Executar sem erros.*
- (A3) Explicar organização do código no relatório.*
- (A4) Listar tarefas cumpridas no relatório.*

2. Qualidade do Código (Qx)

- (Q1) Passar no luacheck*
- (Q2) Separação clara entre Model, View e os estados*
- (Q5) Linhas com até 100 caracteres*

3. Juiciness (Jx)

- (J1) Efeitos de som ocorrem quando as teclas 'up', 'down' e 'enter' são ativadas no menu e um tema é executado durante as fases de combate.*
- (J2) Implementamos um algoritmo de pathfinder que guia os monstros.*
- (J3) Os obstáculos naturais surgem em posições aleatórias cada vez que uma fase é iniciada.*

4. Dano e derrota (Ex)

- (E1) Unidades inimigas perdem vida por tempo quando próximas.*
- (E2) Unidades sem vida morrem.*
- (E3) Invasores se movem em direção ao castelo dos jogadores.*
- (E4) Invasores não se amontoam em cima do castelo.*
 - As unidades vão em direção ao castelo não importa a posição do castelo, pois são orientadas por meio de um algoritmo de pathfinder.*
 - Unidades colidem com obstáculos e são retardadas devido a colisão.*
 - O castelo é destruído se for atingido por pelo menos 12 monstros. Os corações na lateral esquerda da tela fazem a contagem de vidas, a cada três monstros que chegam ao castelo um coração é apagado.*

5. Ondas e vitória (Vx)

- Todos os dados da unidade são armazenados e atualizados via banco de dados.*
- A sequência de ondas e unidades utilizadas bem como obstáculos são armazenadas no BD.*

6. Variação de unidades (Ux)

- (U1) Seleção de unidades durante a partida.*
- (U2) Poder.*
- (U3) Alcance.*
- (U5) Carregar estatísticas do banco de dados.*
 - As unidades são selecionadas em tempo de combate, para tanto o player deve clicar sobre a figura desejada no painel lateral, que esta será atualizada como unidade a ser inserida.*
 - Criamos quatro unidades heróis e três unidades monstros.*
 - Todas as unidades possuem poder, alcance e HP.*

7. Dinheiro e custo das unidades (Cx)

(C1) Cada fase especifica uma quantidade inicial de dinheiro.

(C2) Unidades custam dinheiro.

(C4) Carregar essas informações do banco de dados.

-- A compra de unidades implica em gasto de dinheiro.

-- Sem dinheiro novas unidades não podem ser inseridas em campo.

8. Pré-requisito de unidades (Rx)

Não implementamos.

9. Manobras táticas (Mx)

Não implementamos.

10. Invasores com prioridade (Ix)

-- Invasores são barrados por obstáculos no meio, retardando o avanço.

11. Unidades de Suporte (Sx)

(S1) Algumas unidades afetam outras unidades em seu alcance.

(S2) Unidades afetadas ficam com estatísticas alteradas.

(S3) Carregar efeitos do banco de dados.

-- O raio de alcance dos efeitos das unidades é exibido.

-- O herói Healer cura de tempos em tempos os heróis dentro de seu raio de alcance.

-- O herói Dengue diminui a cada delay o ataque dos inimigos dentro de seu raio de alcance. O efeito é permanente.

-- O herói Dobby muda todos os slimes azuis dentro de seu raio de alcance para slimes verdes, que são mais fracos. O efeito é permanente.

-- Os heróis suportes e seus efeitos são resgatados do banco de dados.

12. Evolução de Unidades (Nx)

(N1) Seleção de evoluções durante a partida.

(N2) Evoluções custam dinheiro.

(N3) Unidades do tipo evoluído ficam mais fortes.

(N4) Carregar evoluções do banco de dados.

-- Ao evoluir, o personagem aumenta seu HP, seu custo e seu dano / efeito.

-- Cada unidade herói possui 3 ou 2 níveis de evolução.

13. Elementos do Mapa (Tx)

(T1) Obstáculos naturais.

14. Invasores aprimorados (Px)

Não implementamos.

15. Conteúdo adicional (Dx)

(D1) Battle of Bastards

(D2) Clash Royale

16. Estrutura da implementação

Implementamos no arquivo `play_stage.lua` a função 'go' que realiza a execução do pathfinder e a função `_load_Landscape` que carrega os obstáculos de forma aleatória na tela, a partir das quantidades especificadas no banco de dados. Na pasta 'state', implementamos `chose_start_monster_coord.lua`, que determina a posição inicial dos monstros no campo, o arquivo `collision.lua` que identifica quando uma unidade colide com um obstáculo.

