
ESPECIFICAÇÃO DE REQUISITOS DE SOFTWARE

para

Stonk Miner

Req_PR_StonkMiner_2021nov09

Versão 1.0

Universidade Federal de Santa Catarina

09/11/2021

Versão	Autor(es)	Data	Ação
1.0	Antonio Luiz, Gabriel Jacinto, Murillo Guindani	09/11/2021	Estabelecimento dos requisitos

Tabela de Conteúdos

1. Introdução

- 1.1. Objetivo do Desenvolvimento
- 1.2. Abreviações
- 1.3. Regras do Jogo
- 1.4. Referências

2. Visão Geral

- 2.1. Arquitetura do Software
- 2.2. Premissas de Desenvolvimento

3. Requisitos da Aplicação

- 3.1. Regras de Negócio
- 3.2. Requisitos Funcionais
- 3.3. Requisitos Não Funcionais

1. Introdução

1.1. Objetivo do Desenvolvimento

O seguinte projeto tem como objetivo a produção de uma paródia multiplayer do jogo de ação “Gold Miner”. As partidas são uma competição entre dois jogadores, na qual ganha quem fizer mais pontos ao final da partida. A temática da paródia é inspirada no fórum r/WallStreetBets e o *short squeeze* nas ações da GameStop que foi feito pelos usuários desse fórum.

1.2. Definições, abreviações

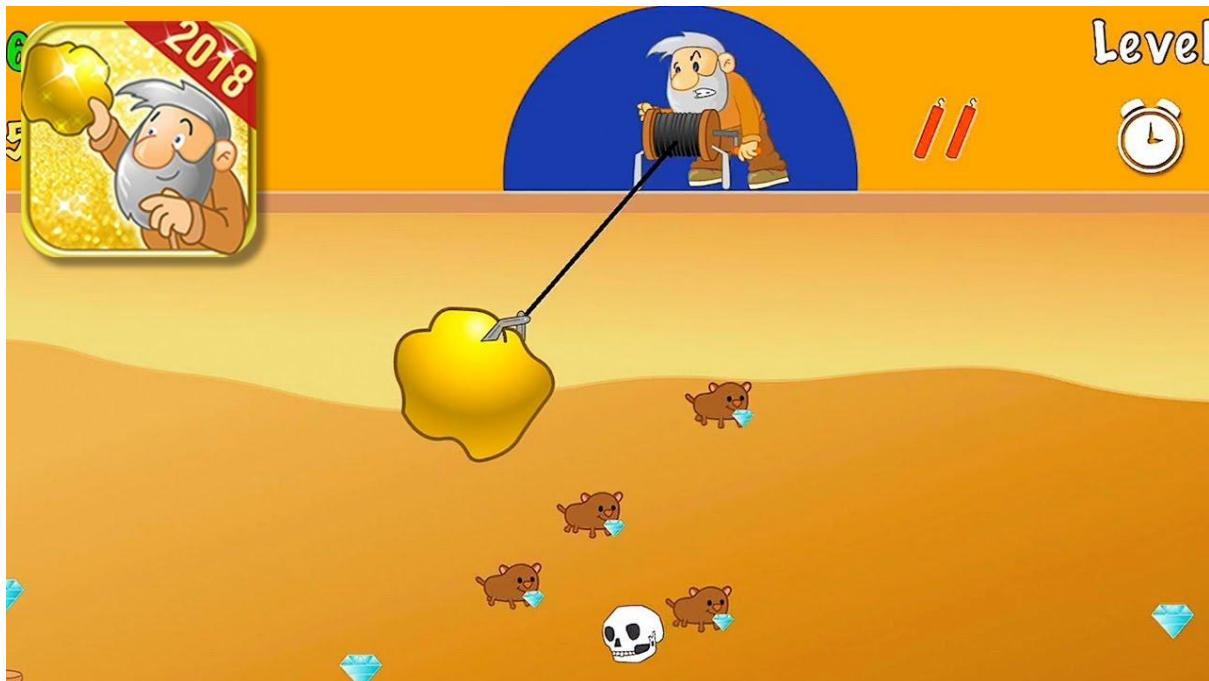
Nesse documento, as seguintes abreviações estarão sendo utilizadas:

- RF: Requisito Funcional
- RNF: Requisito Não Funcional

1.3. Regras do jogo

A partida inicia assim que o jogador aperta o botão de jogar. Uma vez gerado o mapa que será compartilhado pelos dois jogadores simultaneamente, o cronômetro inicia. Os jogadores devem lançar suas garras para capturar quaisquer objetos de valor gerados pelo mapa até o final do cronômetro. As garras movem automaticamente para a esquerda e direita, ao apertar um botão a garra é lançada para a direção que estiver apontando, podendo ou não capturar um objeto. É preciso esperar que a garra volte para sua posição inicial para lançá-la novamente.

Cada jogador ainda poderá receber *power-ups* (objetos especiais) aleatórios para auxiliar na captura dos objetos durante a partida.



1.4. Referências

- Gold miner (descrição) :
https://thatguywiththeglasses.fandom.com/wiki/Gold_Miner
- Gold miner (jogo):
<https://www.clickjogos.com.br/Jogos-online/Acao-e-Aventura/Gold-Miner>
- Pygame: <https://www.pygame.org/docs/>
- Template de documento de especificação de requisitos de software::
https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc
- Fórum Wall Street Bets (subreddit): <https://www.reddit.com/r/wallstreetbets/>
- Noticia sobre o caso *short squeeze* da GameStop:
<https://br.investing.com/analysis/o-que-e-short-squeeze-e-o-que-aconteceu-com-as-acoes-da-gamestop-200439821>

2. Visão Geral do Sistema

2.1. Arquitetura do software

Execução Centralizada.

2.2. Premissas de desenvolvimento

O software deverá ser escrito em python, utilizando o pygame.

O software deverá ter execução centralizada.

O software deverá ser capaz de executar em qualquer plataforma que possua um interpretador python e a biblioteca pygame instalada.

3. Requisitos da Aplicação

3.1. Regras de Negócio

Do funcionamento do jogo:

- O jogo é multiplayer
- As partidas são jogadas com um temporizador.
- Ao fim do tempo é determinado o vencedor da partida.
- Os pontos são calculados como valores monetários dos itens coletados.
- O jogador com mais pontos ao final do tempo é o vencedor.
- Cada nível é construído aleatoriamente antes da partida.
- O tempo é fixado em um minuto

Do funcionamento da garra:

- Quando está pronta, a garra oscila em movimento pendular uniforme a partir de sua posição fixa.
- Ao ser lançada, a garra para de oscilar e percorre o mapa até encontrar um objeto ou o fim do mapa.
- A garra captura itens quando os toca.
- A garra iça itens após os capturar.
- A velocidade de içamento depende do peso do item.
- Se não encontra item em seu percurso, a garra retorna vazia rapidamente.
- Se lançada uma dinamite, o item capturado é destruído e a garra retorna vazia rapidamente.

Da natureza dos itens valiosos:

- Podem ser explosivos ou não (como se fosse uma bolha especulativa)
- Pedra
 - \$50 por unidade de tamanho
- Dólar
 - \$100 por unidade de tamanho
- Ações TSLA
 - \$300 por unidade
- Caixas
 - Contêm 3 dinamites ou Bitcoin (\$500 por unidade)

3.2. Requisitos Funcionais

RF 01: O software deverá ser capaz de administrar as partidas

- Por que? Para que cada partida tenha começo, meio e fim.
- Quem? A aplicação do jogo
- Onde? Na interface do jogo
- Quando? Antes, durante e depois das partidas
- Como? Carregando o nível, iniciando o contador regressivo, terminando a partida ao final do tempo, comparando ganhos de cada jogador e determinando vencedor.

RF 02: O software deverá ser capaz de criar mapas com itens distribuídos de maneira ideal.

- Por que? Para aumentar o dinamismo do jogo, assim cada partida terá um mapa diferente eliminando a necessidade de confecção de múltiplos mapas.
- Quem? A instância de mapa
- Onde? Na interface do jogo
- Quando? Antes de iniciar a partida multiplayer
- Como? Inserindo itens preciosos e não preciosos no mapa de maneira pseudo-aleatória dentro de um intervalo e evitando que estes itens se sobreponham.

RF 03: O software deverá ser capaz de administrar a física do jogo.

- Por que? Para que ocorra impacto entre os objetos e movimento da garra.
- Quem? O motor de física do jogo
- Onde? Na interface do jogo
- Quando? Durante as partidas
- Como? Verificando continuamente por intersecção entre itens e mudando o estado destes itens dependendo do impacto.

RF 04: O software deverá ser capaz de processar o lançamento da garra.

- Por que? Para que o jogador consiga interagir com os itens do mapa.
- Quem? O Jogador
- Onde? No mapa
- Quando? Durante as partidas
- Como? Recebendo comandos do teclado e lançando a garra somente se a garra estiver pronta, isto é, não está retornando de um lançamento.

RF 05: O software deverá ser capaz de processar o retorno da garra.

- Por que? Para que o jogador consiga capturar itens
- Quem? O motor de física
- Onde? No mapa
- Quando? Quando a garra capturar um item ou chegar aos limites do mapa.
- Como? Se captura item, calculando a velocidade de içamento que depende do peso do item. Senão, retornando a garra na velocidade de lançamento.

RF 06: O software deverá ser capaz de processar o lançamento de dinamites.

- Por que? Para que o jogador seja capaz de destruir um item capturado
- Quem? O Jogador
- Onde? Na interface do jogo
- Quando? Somente quando o jogador estiver içando um objeto
- Como? Verificando a existência de uma dinamite no inventário do jogador. Se há dinamite, destrói o item sendo içado o que aumenta a velocidade de retorno da garra.

3.3. Requisitos Não Funcionais

RNF 01: O software deverá ser escrito em python

RNF 02: O software deverá ter execução centralizada

RNF 03: O software deverá ser capaz de executar em qualquer plataforma que possua um interpretador python e a biblioteca pygame instalada.

RNF 04: O Software deve exibir um menu inicial com explicação sobre como o jogo funciona e botão para iniciar o jogo.

RNF 05: A janela onde o jogo será exibido deverá ter resolução 1200x600 pixels