

Projeto Cucco Race

Especificação de Requisitos de Software

Versão 1.1

26/04/2016

Versão	Autores	Data	Ação
1.0	Diego Feijó	07/04/2016	Estabelecimento dos requisitos.
1.1	Diego Feijó	26/04/2016	Revisão sobre requisitos conforme avaliação da primeira entrega.

Conteúdo:

1. Introdução;
2. Visão geral;
3. Requisitos de software;
4. Esboço Interface Gráfica;
5. Referências.

1 Introdução:

Objetivo: desenvolvimento de um software distribuído que emule uma disputa de dois a quatro jogadores em um tabuleiro que propõe uma corrida por turnos, permitindo que jogadores se enfrentem em confrontos eliminatórios.

Regras do jogo:

Nomenclatura utilizada neste documento:

- **Jogador:** o personagem que responde a comandos.
- **Casas:** parte jogável do mapa, onde os jogadores poderão interagir com outros personagens e itens de cenário. Cada casa é indicada como uma fração do mapa, chamada de *tile*,

nomenclatura padrão de jogos baseados em tiles. Algumas casas podem disparar eventos que decidirão a sorte dos jogadores. Estas casas serão chamadas de **casas de evento**. Uma casa pode ser ocupada por apenas um personagem.

- **NPC:** do inglês *Non Playable Character* ou personagem não jogável, são personagens não controlados pelos jogadores. Em Cucco Race, esses personagens podem disparar eventos que decidirão a sorte dos jogadores.
- **Inimigos:** personagens que disparam eventos de confronto.
- **Personagem:** jogadores, NPCs ou inimigos.
- **PdV:** pontos de vida do personagem. Determinam a saúde e podem ser alterados durante o jogo. Um personagem sem PdV é eliminado da partida.
- **Encontro:** quando um jogador finaliza seu movimento em uma casa ocupada por outro personagem acontece um encontro. Um evento de confronto é disparado, no caso de jogador ou inimigo. Para NPCs um evento de sorte é disparado.
- **Confronto:** eventos de confronto decidem quem ocupará a casa onde o evento foi disparado. O jogador atual pode decidir entre enfrentar ou conceder a vitória. No primeiro caso, cada personagem rolará um dado de seis faces para decidir o ataque e a defesa. O personagem que conseguir o maior número vence e permanece na casa, enquanto o que rolou o menor número perde uma quantidade de seus PdV e volta uma casa.
- **Sorte:** eventos de sorte podem auxiliar ou atrapalhar a caminhada dos jogadores. Quando disparados por um encontro com NPCs, finalizam com o NPC concedendo a casa ao jogador ou permitindo que o jogador avance uma casa, evitando que dois personagens ocupem a mesma casa.

Os jogadores iniciam na casa inicial do mapa e tiram a sorte para decidir a ordem dos turnos. A cada turno os jogadores poderão executar as seguintes ações:

- Rolar dados para determinar o número de casas que pode andar no turno;
- Andar exatamente o número de casas sorteado, sendo que se o movimento finalizar em uma casa ocupada por outro personagem, um evento é disparado para decidir quem ocupará a casa.
- Decidir sobre eventos disparáveis por casas de evento, NPCs ou outros jogadores, entre eles:
 - Escolher enfrentar outro jogador, se estiver na mesma casa;
 - Escolher enfrentar inimigos, se estiver na mesma casa;
 - Interagir com NPCs;

Cada jogador inicia com uma quantidade de PdV que pode ser alterada por eventos em casas especiais do mapa ou em encontros com outros jogadores e/ou inimigos não jogáveis. Os jogadores sem pontos de vida são eliminados da partida.

Vence o primeiro jogador que chegar ao fim do mapa ou o jogador remanescente.

2 Visão Geral

Arquitetura do programa: programa orientado a objetos, distribuído.

Premissas de desenvolvimento:

- O programa deve apresentar uma interface gráfica bidimensional, dividida em uma matriz onde cada posição será referenciada como um *tile*, com uma paleta de 256 cores, similar aos jogos 16 bits.
- O programa fará uso da linguagem de programação Java em sua versão SE8, com o paradigma de Orientação a Objetos, do software para cliente-servidor NetGames.
- A estrutura básica do jogo será gerida pelo software TiledGame2D, um *framework* que utiliza as bibliotecas do Java para controlar as regras básicas, comuns a jogos bidimensionais baseados em *tiles*.

3 Requisitos de Software

3.1 Requisitos Funcionais:

1 – Conectar: como será um software distribuído, deve apresentar uma interface gráfica que permita ao usuário conectar-se ao servidor NetGames que fará a gestão dos usuários conectados.

2 – Desconectar: deve permitir ao usuário que possa se desconectar de um servidor via interface gráfica.

3 – Iniciar partida: uma vez conectado, o programa deve possibilitar a inicialização de uma nova partida, onde será indicado o número de jogadores desta partida. O programa deve permitir partidas contendo de dois a quatro jogadores.

4 – Estabelecer ordem dos turnos: ao definir os jogadores, o programa deve decidir, de forma aleatória, a ordem dos seus turnos e apresentar aos jogadores via interface gráfica.

5 – Composição do mapa: o mapa deve ser composto de tiles jogáveis (casas) e tiles de cenários. O conjunto de casas servirá como o caminho entre o ponto de partida até o fim do mapa, podendo haver mais de um caminho possível. Os tiles de cenário servem apenas como imagem de fundo, não permitindo que os jogadores caminhem por eles.

6 – Rolar Dados: para decidir a quantidade de casas que será possível andar no turno, o programa deve simular ao usuário uma rolagem de dados, mostrando na interface gráfica o resultado desta

rolagem e guardar este valor como movimentação restante do personagem.

7 – Movimento do jogador: o jogador deve ter a possibilidade de se movimentar no tabuleiro utilizando as teclas direcionais do teclado, sendo que cada movimentação corresponde a um tile e deve respeitar a quantidade restante de movimentações possíveis. Um movimento bem-sucedido segue as seguintes condições:

- O jogador deve ter movimentação disponível;
- O tile ao qual deseja mover-se deve ser uma casa do tabuleiro.
- Ao fim do movimento as seguintes ações devem ser executadas:
 - Decrescer a movimentação disponível do personagem;
 - Conferir se existem movimentos a serem utilizados por esse jogador;
 - Se não houverem movimentos restantes, conferir se existem eventos a serem disparados por personagens ou casas de evento;
 - Se um evento foi disparado, decidir se executará ou não o evento. A opção pela não execução deve fazer com que o jogador regresse uma casa;
 - Conferir se a condição de vitória foi satisfeita e atribuir vitória ao jogador que executou o movimento em caso positivo;
 - Se não houverem movimentos restantes e a condição de vitória não foi satisfeita, finalizar o turno do jogador;

3.2 Requisitos não Funcionais

1 – Especificação de projeto: além do código Java, deve ser produzida especificação de projeto baseada em UML segunda versão.

2 – Identificação dos jogadores: os jogadores devem ser identificados por um nome inserido ao iniciar uma nova partida.

3 – Tecnologia da GUI: *graphic user interface* baseada na biblioteca nativa *Graphics2D*, auxiliado pelo software TiledGame2D.

4 Esboço da Interface Gráfica

A interface gráfica do jogo (*GUI*), será composta de recursos disponíveis nas comunidades de desenvolvedores de jogos na web, sendo suas respectivas fontes referenciadas ao fim deste

documento. A ferramenta Tiled será utilizada para criação dos mapas.



Figura 1: Esboço da GUI

5 Referências

MORGAN, Leonardo. **NetGames**. Disponível em: <<http://www.inf.ufsc.br/~netgames/>>. Acesso em: 07 abr. 2016.

LINDEIJER, Thorbjørn. **Tiled Map Editor**. Disponível em: <<http://www.mapeditor.org/>>. Acesso em: 07 abr. 2016.

THEVGRESOURCE. **The VG Resource**. Disponível em: <<http://www.sprisers-resource.com/>>. Acesso em: 07 abr. 2016.