

Especificação de requisitos

Quixo

v1.0

Versão	Autores	Data	Ação
1.0	Amanda Oliveira, Marco Cechinel e Willian Kruscinski	28/03/2023	Estabelecimento dos requisitos

Conteúdo

1. Introdução
 - 1.1 Objetivo
 - 1.2 Definições e abreviações
 - 1.3 Referência
2. Regras do jogo
3. Visão geral
4. Requisitos de software
 - 4.1 Requisitos funcionais
 - 4.2 Requisitos Não-Funcionais

Introdução

Objetivo

Desenvolvimento de um programa que suporte a disputa de partidas de Quixo na modalidade usuário contra usuário, estando ambos em diferentes máquinas.

Definições e abreviações

Neste documento, as seguintes abreviações serão utilizadas:

- RF: Requisito Funcional
- RNF: Requisito Não Funcional

Referência

Canal VemKa Jogar <https://youtu.be/oC_KUMylomc>.

Regras do jogo

1. O tabuleiro é composto por 25 peças e cada uma possui 3 possíveis faces (neutro, “X” e “O”). As 25 peças são organizadas formando um tabuleiro 5X5.
2. Quando o jogo se inicia todas as faces estão em posição neutra.
3. Um jogador será designado para ser o “X” e o outro o “O”.
4. As jogadas são alternadas.
5. Na vez do jogador:
 - a. O jogador deve retirar um cubo da borda do tabuleiro que possua a face neutra ou possua a sua designação (o “X” ou o “O”).
 - b. Caso o cubo seja de face neutra o cubo passa a ter a face do jogador.

- c. Agora é preciso reposicionar este cubo dentro do tabuleiro “empurrando” uma linha ou coluna (em qualquer direção na horizontal ou vertical) que tenha uma posição dada pela remoção da peça selecionada.
 - i. É expressamente proibido manter a peça na posição em que ela já estava.
6. O primeiro jogador que obter uma linha na horizontal, vertical ou ortogonal contendo peças das quais ele foi designado ganha o jogo.

Visão geral

Arquitetura do programa: programa orientado a objetos em arquitetura cliente-servidor distribuído.

Premissas de desenvolvimento:

- O programa deve apresentar uma interface gráfica bidimensional.
- O programa deve ser implementado em Python.
- O programa deve usar DOG como suporte para execução distribuída.
- Além do código, deve ser produzida especificação de projeto baseada em UML.

Requisitos de software

Requisitos Funcionais

RF 1. Iniciar programa: ao ser executado, o programa deve apresentar na interface o tabuleiro do jogo em seu estado inicial (todas as peças do tabuleiro em face neutra e a mensagem ‘Quixo’) e solicitar o nome do jogador. Após isso, deve solicitar conexão com DOG Server (utilizando os recursos de DOG). O resultado da tentativa de conexão deve ser

informado ao usuário. Apenas em caso de conexão bem sucedida as demais funcionalidades estarão habilitadas. No caso de conexão mal sucedida, a única alternativa deve ser encerrar o programa;

- RF 2. Iniciar jogo:** o programa deve apresentar a opção de menu “iniciar jogo” para o início de uma nova partida. O procedimento de início de partida consiste em enviar uma solicitação de início a Dog Server, que retornará o resultado, que será a identificação e a ordem dos jogadores, em caso de êxito, ou a razão da impossibilidade de início de partida, caso contrário. A interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance. Esta funcionalidade só deve estar habilitada se o programa estiver em seu estado inicial, isto é, sem partida em andamento e com o tabuleiro em seu estado inicial.
- RF 3. Restaurar estado inicial:** o programa deve apresentar a opção de menu “restaurar estado inicial” para levar o programa ao seu seu estado inicial, isto é, sem partida em andamento e com o tabuleiro em seu estado inicial. Esta funcionalidade só deve estar habilitada se o programa estiver com uma partida finalizada.
- RF 4. Selecionar peça:** o programa deve permitir a um jogador habilitado selecionar uma peça presente no tabuleiro. A peça selecionada deve ser visualmente destacada da interface do programa. Se a ação for executada após o jogador já ter uma peça selecionada, deve ser notificado lance irregular e o programa deve novamente aguardar a primeira ação do jogador habilitado (selecionar peça).
- RF 5. Selecionar destino:** o programa deve permitir a um jogador habilitado selecionar uma posição do tabuleiro, onde será colocada a peça previamente selecionada. Esta funcionalidade só deve estar habilitada se o programa estiver com peça selecionada (ver Requisito funcional 4). O

programa deve avaliar se a posição destino pode receber a peça selecionada, de acordo com as regras do jogo. Em caso positivo, deve efetuar a jogada - isto é, remover a peça de seu posicionamento inicial e colocá-la na posição destino - e, em caso negativo, notificar o lance irregular e o programa deve novamente aguardar a primeira ação do jogador habilitado (selecionar peça). No caso de êxito na colocação de peça em seu destino, o programa deve enviar a jogada ao adversário (utilizando os recursos de DOG) e avaliar o encerramento de partida. A jogada enviada deve conter a posição de seleção da peça do tabuleiro e a posição destino. No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor; no caso de não encerramento, deve ser desabilitado o jogador local e o programa fica no aguardo de jogada do adversário (RF 7) ou de notificação de abandono (RF 8);

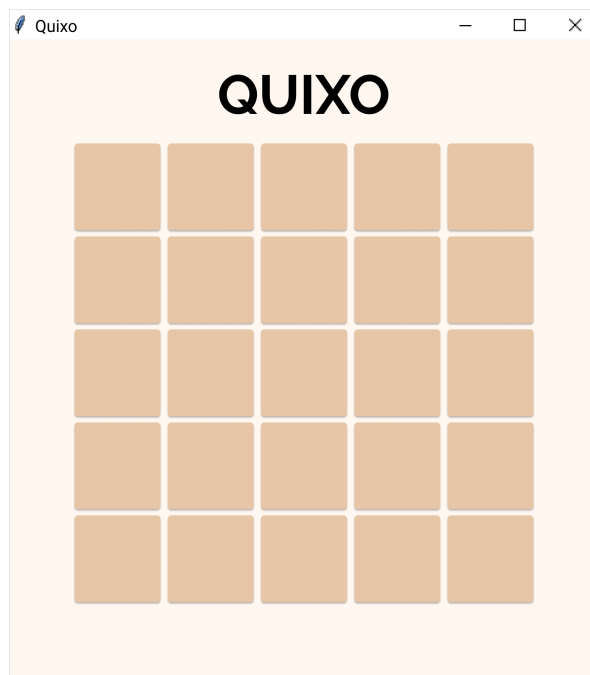
RF 6. Receber determinação de início: o programa deve poder receber uma notificação de início de partida, originada em Dog Server, em função de solicitação de início de partida por parte de outro jogador conectado ao servidor. O procedimento a partir do recebimento da notificação de início é o mesmo descrito no 'RF2 – Iniciar jogo', isto é, a interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance.

RF 7. Receber jogada: o programa deve poder receber uma jogada do adversário, enviada por Dog Server, quando for a vez do adversário do jogador local. A jogada recebida deve ser um lance regular e conter as informações especificadas para o envio de jogada no 'RF5 – Selecionar destino'. O programa deve remover a peça da origem definida e colocá-la no destino. Após isso, deve avaliar o encerramento de partida. No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor; no caso de não encerramento, deve ser habilitado o jogador local, para que possa proceder a seu lance.

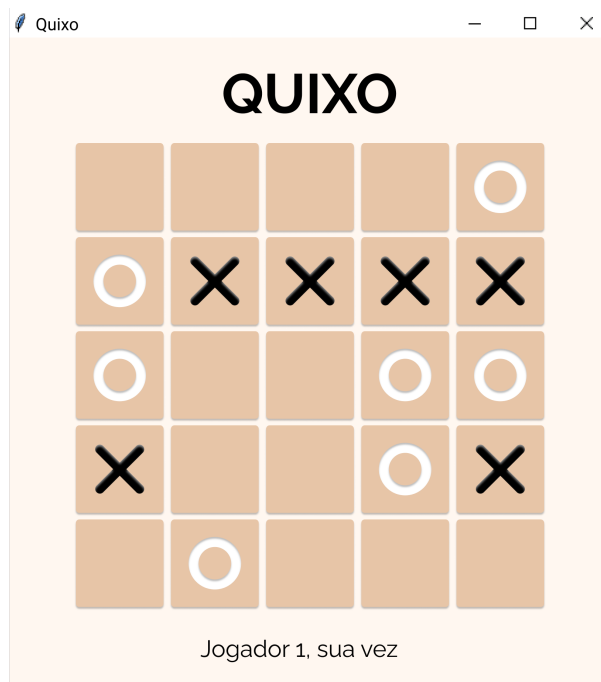
- RF 8. Receber notificação de abandono:** o programa deve poder receber uma notificação de abandono de partida por parte do adversário remoto, enviada por Dog Server. Neste caso, a partida deve ser considerada encerrada e o abandono notificado na interface.
- RF 9. Visualizar regras do jogo:** o programa deve possuir um botão de acesso que mostre todas as regras do jogo quando selecionado.

Requisitos Não-Funcionais

- RNF 1. Especificação de projeto:** além do código Python, deve ser produzida especificação de projeto baseada em UML com a ferramenta Visual Paradigm.
- RNF 2. Especificação de projeto:** o jogo deve ter funcionamento distribuído sob uso do DOG.
- RNF 3. Interface gráfica para usuário:** o programa deverá ter interface gráfica similar a do esboço abaixo, disponível e atualizada a cada nova jogada recebida a partir e pelo DOG;



Tabuleiro neutro (inicial)



Tabuleiro de jogo em progresso

RNF 4. Tecnologia de interface gráfica para usuário: A interface gráfica deve ser baseada em TKinter.