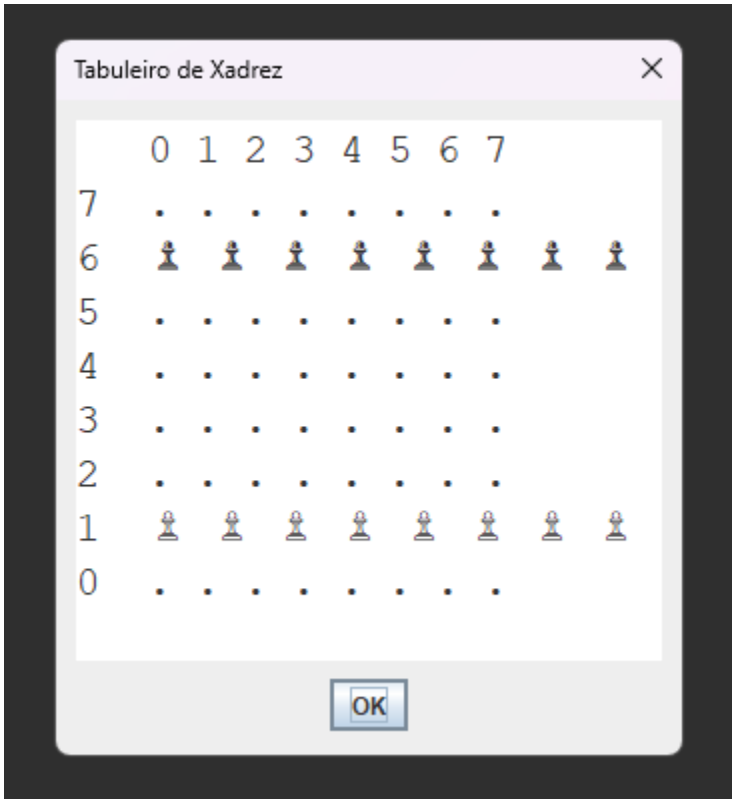
Documentação do Sistema de Xadrez.

Aluna: Bruna Xavier Alves de Mello.

Tabuleiro.



Sugestões de Entradas:
Peças Brancas

- **(1,0) - Linha/Coluna de Origem**
- **(3,0) - Linha/Coluna Destino**

Peças Pretas: :

- **(6,0) - Linha/Coluna de Origem**
- **(4,0) - Linha/Coluna Destino**

Explicações Técnicas Para Consulta.

Pacote: `xadrez.peca`

Classe: `Peao`

Responsabilidade: Representa o peão no tabuleiro, encapsulando sua posição, cor e regras de movimento/captura.

Atributos:

- `linha` e `coluna` – posição atual do peão.
- `cor` – cor da peça (`BRANCO` ou `PRETO`).
- `primeiroMovimento` – indica se o peão ainda não se moveu (permite avançar duas casas).

Métodos principais:

- `validarMovimento(int linhaDestino, int colunaDestino, Tabuleiro tabuleiro)`
  Valida se o movimento é permitido:
  - Avanço reto (1 ou 2 casas)
  - Captura diagonal (1 casa) - Não permitida
  - Não permite avançar para trás nem capturar aliado
- `validarMovimentoFrente(...)`
  Verifica avanço reto, bloqueio de peças e primeira jogada.
- `validarCaptura(...)`
  Verifica se há peça adversária na diagonal para capturar.
- `mover(int linhaDestino, int colunaDestino)`
  Atualiza posição e marca que o peão já se moveu.

- `getRepresentacao()`
  Retorna representação Unicode: ♙ (branco), ♟ (preto).

Enum interno: `Cor`

- `BRANCO`, `PRETO` – define cores das peças e permite comparação de turnos.

Pacote: `xadrez.tabuleiro`

Classe: `Tabuleiro`

Responsabilidade: Representa o tabuleiro de xadrez e gerencia as posições das peças.

Atributos:

- `matriz` – matriz 8x8 de `Peao` que representa o tabuleiro.

Métodos principais:

- `obterPeca(int linha, int coluna)` – retorna a peça na posição.
- `possuiPeca(int linha, int coluna)` – verifica se há peça na posição.
- `colocarPeca(Peao peca, int linha, int coluna)` – adiciona uma peça ao tabuleiro.
- `removerPeca(int linha, int coluna)` – remove peça da posição.
- `moverPeca(int linhaOrigem, int colunaOrigem, int linhaDestino, int colunaDestino)` – realiza movimento de peça, chamando `validarMovimento` do peão.
- `validarPosicao(int linha, int coluna)` – garante que a posição esteja dentro do tabuleiro.
- `getMatriz()` – retorna a matriz completa do tabuleiro.

Regras implementadas:

- Movimento válido dentro do tabuleiro (0–7).
- Não permite sobrescrever peça aliada.

Pacote: `xadrez.controle`

Classe: `Jogo`

Responsabilidade: Controla o fluxo do jogo, entrada do usuário e alternância de turnos.

Atributos:

- `tabuleiro` – instância do tabuleiro.
- `turnoBranco` – indica se é turno das brancas ou pretas.
- `tentativasMovimento` – conta tentativas de movimento no turno atual.

Métodos principais:

- `inicializarPecas()` – posiciona os peões no tabuleiro (linha 1 = brancas, linha 6 = pretas).
- `iniciar()` – loop principal do jogo:
    - Solicita origem e destino do movimento
    - Valida peça do turno
    - Executa movimento ou exibe erro
    - Alterna turno
- `solicitarPosicao(String mensagem)` – caixa de diálogo para entrada de posição.
- `parsearCoordenadas(String input)` – converte string em coordenadas do tabuleiro.

- `validarPecaDoTurno(int linha, int coluna)` – valida se peça pertence ao jogador.
- `executarMovimento(...)` – chama `tabuleiro.moverPeca` e exibe sucesso.
- `encerrarJogo()` e `cancelarMovimento()` – mensagens de controle.
- `exibirErroRegra(XadrezException e)` e `exibirErroInput()` – feedback de erro ao jogador.
- `exibirTabuleiro()` – mostra o tabuleiro em `JTextArea` com representação Unicode dos peões.

Fluxo do jogo:

1. Exibe tabuleiro.
2. Solicita entrada do jogador.
3. Valida movimento do peão.
4. Move peça se válido ou exibe erro.
5. Alterna turno entre brancas e pretas.
6. Repete até encerramento.

Regras implementadas no sistema

- Peões podem avançar 1 casa ou 2 casas no primeiro movimento.
- Não é permitido capturar peça aliada.
- Movimentos fora do tabuleiro são inválidos.
- Entrada de coordenadas inválida gera exceção.