

Guilherme Henrique Galdini Tosi - 11781587

Os UML em .png foram feitos usando a ferramenta StarUML

O jogo salva a fase atual em um arquivo chamado "save", para retornar a primeira fase é necessário excluir esse arquivo.

```

classDiagram
    class Main {
        +mainArea: String[]
    }
    class Tela {
        +hSkooter: Skooter
        +eElementos: Elemento[]
        +g2: Graphics
        +background: String
        +w: int = 5
        +h: int = 5
        +constructor: +Tela()
        +Tela faseAtual: int
        +addElemento(lim:Elemento)
        +removeElemento(cid:Elemento)
        +getGraphicsBuffer(): Graphics
        +paint(Graphics)
        +keyPressed: eKeyEvent
        +mousePressed: eMouseEvent
        +lerSave(): int
        +receberSave(valor: int)
        +moComponente()
        +mouseMove(e: MouseEvent)
        +mouseClicked: MouseEvent
        +mouseReleased: MouseEvent
        +mouseEntered: MouseEvent
        +mouseExited: MouseEvent
        +mouseDragged: eMouseEvent
        +keyTyped: eKeyEvent
        +keyReleased: eKeyEvent
    }
    class ControlDeJogo {
        +desenhaTudo: Elemento[]
        +processaTudo: Elemento[]
        +posicaoValida: Elemento[]
        +validaRobote: Elemento[]
        +posicaoMortal: Elemento[]
        +concordaValida: Elemento[]
        +posicaoValidaArrows: Elemento[]
    }
    class Posicao {
        +linha: int
        +coluna: int
        +linhaAnterior: int
        +colunaAnterior: int
        +constructor: +Posicao(linha: int, coluna: int)
        +setPosicao(linha: int, coluna: int)
        +getLinha(): int
        +getColuna(): int
        +valida(): boolean
        +setLinhaAnterior(): int
        +setColunaAnterior(): int
        +estaNaMesmaPosicao(posicao: Posicao)
        +copiar(posicao: Posicao)
        +moveUp(): boolean
        +moveDown(): boolean
        +moveLeft(): boolean
        +moveRight(): boolean
    }
    class Elemento {
        +rImage: ImageIcon
        +pPosicao: Posicao
        +bTranspavel: boolean
        +bMortal: boolean
        +bColatevel: boolean
        +bMovvel: boolean
        +bQuebravel: boolean
        +bata: boolean
        +constructor: +Elemento(nomeImagem: PNG String)
        +getPosicao(): Posicao
        +abTranspavel(): boolean
        +abMortal(): boolean
        +abColatevel(): boolean
        +abMovvel(): boolean
        +abQuebravel(): boolean
        +setBata: boolean
        +setTranspavel(bTranspavel: boolean)
        +setPosicao(linha: int, coluna: int)
        +moveUp(): boolean
        +moveDown(): boolean
        +moveLeft(): boolean
        +moveRight(): boolean
        +voltaUltimaPosicao()
        +autoDesenho()
    }
    class Arrow {
        +dir: int
        +Control: ControlDeJogo
        +constructor: +Arrow(nomeImagem: PNG String, dire: int, elementos: Elemento[])
        +empurar(): boolean
        +autoDesenho()
    }
    class Parede {
        +constructor: +Parede(nomeImagem: PNG String, breakIt: boolean)
    }
    class ParedeAtravessavel {
        +constructor: +ParedeAtravessavel(nomeImagem: PNG String, vemAtrai: boolean)
    }
    class Prizes {
        +constructor: +Prizes(nomeImagem: PNG String)
    }
    class Robo {
        +bRight: boolean
        +joPaCima: boolean
        +joPaBaixo: boolean
        +joPaEsquerda: boolean
        +joPaDireita: boolean
        +Control: ControlDeJogo
        +constructor: +Robo(nomeImagem: PNG String, eElementos: Elemento[])
        +autoDesenho()
        +voltaUltimaPosicao()
    }
    class Skooter {
        +constructor: +Skooter(nomeImagem: PNG String)
        +voltaUltimaPosicao()
    }
    Main --> Tela
    Tela --> ControlDeJogo
    ControlDeJogo --> Posicao
    ControlDeJogo --> Elemento
    ControlDeJogo --> Arrow
    ControlDeJogo --> Parede
    ControlDeJogo --> ParedeAtravessavel
    ControlDeJogo --> Prizes
    ControlDeJogo --> Robo
    ControlDeJogo --> Skooter
    Posicao --> Elemento
    Elemento --> Arrow
    Elemento --> Parede
    Elemento --> ParedeAtravessavel
    Elemento --> Prizes
    Elemento --> Robo
    Elemento --> Skooter
  
```

The diagram illustrates the architecture of the 'Corridos' game. It features several classes: **Main** (from Modelo), **Tela** (from Controlador), **ControlDeJogo** (from Controlador), **Posicao** (from Auxilia), **Elemento** (from Modelo), **Arrow** (from Modelo), **Parede** (from Modelo), **ParedeAtravessavel** (from Modelo), **Prizes** (from Modelo), **Robo** (from Modelo), and **Skooter** (from Modelo). The **ControlDeJogo** class acts as a central controller, managing the game state and interacting with the **Tela** and **Elemento** classes. The **Elemento** class is a base class for various game objects like **Arrow**, **Parede**, **ParedeAtravessavel**, **Prizes**, **Robo**, and **Skooter**. The **Posicao** class is used to track the position of the **Elemento** objects. The **Tela** class is responsible for rendering the game state and handling user input. The **Main** class is the entry point of the application. The diagram shows the relationships between these classes, including inheritance and associations.

Consts

