***Arcade Game Maker*:**

Doc.

5

**Descrição Geral da Linha de Produto**

**I. Identificação**

A linha de produto de software (LPS) *Arcade Game Maker* (AGM) produz uma série de jogos arcade – ou seja, produtos com um ou mais jogos. Cada jogo é jogado por um único jogador que controla, parcialmente, os objetos que se movem. O objetivo é marcar pontos acertando obstáculos estáticos. Os jogos vão desde aqueles com obstáculos baixos até obstáculos altos e estão disponíveis para uma variedade de diferentes plataformas.

**II. Similaridades e Variabilidades**

Nesta seção são apresentadas as principais similaridades da LPS, ou seja, os aspectos comuns a todos os produtos desta LPS, bem como as variabilidades, que representam os aspectos que diferem de um produto em relação ao outro.

**II.1 Similaridades**

* Todos os jogos possuem elementos que recebem a interação do usuário (*Vector*);
* Todos os jogos possuem mesa (*Board*); e
* Os elementos não mencionados nas similaridades ou variabilidades da LPS são considerados obrigatórios para o correto funcionamento dos produtos.

**II.2 Variabilidades**

* Cada jogo, ao ser selecionado deverá ter sua mesa construída (*buildGameBoard*), logo a seleção de um ou mais jogos (mínimo 1 e máximo 3) necessita da construção de sua respectiva mesa. Assim, tal construção ocorrerá de acordo com o(s) jogo(s) que o produto suporta, devendo possuir no mínimo 1 mesa e máximo 3.
* A construção das mesas do jogo *Brickles* (*buildGameBoard()\_Brickles*) e *Pong* (*buildGameBoard()\_Pong*) poderá suportar ou não a existência de *pucks* de provisão (*PuckSupply)*, que correspondente ao número de bolas no jogos.
* Todo o jogo deverá ser salvo (*saveGame*), para isso é necessário que a funcionalidade dos jogos presentes no produto estejam presentes, por exemplo, se o jogo *Brickles* estiver presente no produto, a opção para salvar este jogo e seu respectivo menu deverá estar acessível, e assim por diante. Desta forma, esta funcionalidade estará presente no produto, de acordo com o jogo que este possui, sendo no mínimo 1 e máximo 3; e
* Todo o jogo deverá ser carregado, para isso, cada método correspondente ao carregamento do jogo (*loadGame*) deverá estar presente nos produtos, de acordo com a seleção de seus respectivos jogos (mínimo 1 e máximo 3). A seleção desta funcionalidade é análoga a opção de salvar o jogo, a qual estará presente, de acordo com o jogo que compor o produto.

**III. Descrição dos Elementos**

|  |  |
| --- | --- |
| **Elemento** | **Descrição** |
| *addMovablePiece()* | Adicionar peça móvel |
| *addStationaryPiece()* | Adicionar peça fixa |
| *Board* | Mesa/tabuleiro de jogo |
| *BottomPaddle()* | Elemento que movimenta a Puck do jogo, localizado na parte inferior da PongBoard. |
| *BowlingBoard* | Mesa/tabuleiro do jogo Bowling |
| *BowlingGameMenu* | Menu do Jogo de Bowling (Boliche) |
| *BricklesGameMenu* | Menu do Jogo Brickles |
| *buildGameBoard()* | Construir mesa/tabuleiro do jogo |
| *EndOfAlley()* | Fim da pista de boliche |
| *GamePlayer* | Jogador |
| *getGameOver()* | Obter fim de jogo |
| *getSaveData()* | Obter dados salvos |
| *Lane()* | Lona |
| *loadGame()* | Carregar jogo |
| *movableComponent* | Componente móvel |
| *New* | Novo |
| *playGame()* | Jogar jogo |
| *PongGameMenu* | Menu do Jogo Pong |
| *PuckSupply* | Puck Representa o principal elemento de um jogo como, por exemplo, a bola que derruba os BowlingPins no jogo Bowling, a bolinha que destrói os BrickPile no jogo Brickles, etc. PuckSupply é a quantidade de Pucks que o jogador tem direito em um jogo. |
| *rackPins()* | Empilhar pinos |
| *RecordStore* | Dados armazenados |
| *removeAllElements()* | Remover todos os elementos |
| *saveGame()* | Salvar jogo |
| *setMessage()* | Definir mensagem |
| *setSaveData()* | Definir dados salvos |
| *startMoving()* | Iniciar movimento |
| *stationaryComponents* | Componentes fixos |
| *TopPaddle()* | Elemento que movimenta a Puck do jogo, localizado na parte superior da PongBoard. |
| *Vector* | Vetor |