# **Build** (*Worker* worker, *Vector2* position, *string* buildingName)

Descrição: Este método é utilizado para mandar um trabalhado construir uma construção em um determinado local.

Parâmetros:

* *Worker*: Trabalhador que executará a ação.
* *Vector2*: Posição(x, y) no plano cartesiano para realizar a construção.
* *string*: Nome da construção que se deseja construir.

Retorno: Este método retorna **verdadeiro** se foi possível contruir. Caso contrário, retorna **falso**.

# AddUnit (*Building* building, *string* unitName)

Descrição: Este método é utilizado para adicionar uma unidade de determinada construção ao jogo.

Parâmetros:

* *Building*: A construção onde a unidade será treinada.
* *String:* O nome da unidade que se deseja criar. (A unidade deve estar na lista de unidades da contrução referida).

Retorno: Este método retorna **verdadeiro** se foi possível adicionar a unidade. Caso contrário, retorna **falso**.

# **Walk** (*Unit* unit, *Vector2* position)

Descrição: Este método é utilizado para fazer com que uma unidade ande para um determinado local.

Parâmetros:

* *Unit*: Unidade que deseja movimentar.
* *Vector2*: Posição(x, y) no plano cartesiano para onde deseja movimentar a unidade.

Retorno: Método sem retorno.

# GetNearTile (*Building* building)

Descrição: Este método é utilizado para fazer pegar a posição disponível(possível de construir) mais próxima de determinada construção.

Parâmetros:

* *Building*: Construção de referência para checar as Tiles em volta.

Retorno: Retorna um **Vector2** da posição da Tile disponível mais próxima para realizar uma construção.

# GetNearSource (*Worker* worker*, string* materialName)

Descrição: Este método é utilizado para fazer pegar a posição da fonte de recurso mais próxima de determinado trabalhador.

Parâmetros:

* *Worker*: Trabalhador de referência para obter a fonte mais próxima.
* *String*: Nome do material base que deseja obter.

Retorno: Retorna um **Vector2** com a posição da fonte mais próxima que produz o material desejado.

# Destroy (*Property* property)

Descrição: Este método é utilizado para destruir completamente uma propriedade...

Parâmetros:

* *Property*: Propriedade que deseja destruir.

Retorno: Este método retorna **verdadeiro** se foi possível destruir a propriedade. Caso contrário, retorna **falso**.

# GetUnitAvailable (*string* unitName)

Descrição: Este método é utilizado para pegar uma unidade disponível da lista de unidades do jogador a partir do nome. (Entende-se por unidade disponível aquela que não está andando, atacando, construindo e/ou coletando).

Parâmetros:

* *String:* Nome da unidade que o jogador deseja pegar da lista.

Retorno: Retorna a **lista com as unidades** com o nome indicado no método.

# GetBuilding (*string* buildingName)

Descrição: Este método é utilizado para pegar uma construção da lista de construções do jogador a partir do nome.

Parâmetros:

* *String*: Nome da construção que o jogador deseja pegar da lista.

Retorno: Retorna a **lista de construções** com o nome indicado no método.

# GetMaterialSourceAvailable (*string* sourceName)

Descrição: Este método é utilizado para pegar todas as fontes disponível de determinado recurso.

Parâmetros:

* *String*: Nome da fonte a ser buscada.

Retorno: Retorna uma **lista** com todas as fontes disponíveis com o nome indicado no método.

# GetAmountUnit (*string* unitName)

Descrição: Este método é utilizado para descobrir quantas unidades com determinado nome o jogador possui.

Parâmetros:

* *String*: Nome da unidade a ser contabilizada.

Retorno: Retorna um **inteiro** (a quantidade de determinada unidade).

# GetAmountTroop ()

Descrição: Este método é utilizado para descobrir a quantidade de tropa que o jogador possui.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna um **interio** (a quantidade de tropa).

# GetProductionRate (*string* materialName)

Descrição: Este método é utilizado para descobrir a taxa de produção de determinado recurso.

Parâmetros:

* *String*: Nome do material que se deseja obter a taxa de produção.

Retorno: Retorna um **float** (taxa de produção por unidade de tempo).

# VerifyPropertiesIsAttacked ()

Descrição: Este método é utilizado para descobrir se alguma propriedade está sendo atacada.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna **verdadeiro** se alguma propriedade estiver sendo atacada. Caso contrário, retorna **falso**.

# VerifyBuildingsIsAttacked ()

Descrição: Este método é utilizado para descobrir se alguma contrução está sendo atacada.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna **verdadeiro** se alguma construção estiver sendo atacada. Caso contrário, retorna **falso**.

# VerifyUnitsIsAttacked ()

Descrição: Este método é utilizado para descobrir se alguma unidade está sendo atacada.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna **verdadeiro** se alguma unidade estiver sendo atacada. Caso contrário, retorna **falso**.

# GetUnitsAttacked ()

Descrição: Este método é utilizado para pegar todas as unidade que estão sendo atacadas.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna uma **lista** com todas as unidades que estão sendo atacadas.

# GetBuildingsAttacked ()

Descrição: Este método é utilizado para pegar todas as construções que estão sendo atacadas.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna uma **lista** com todas as construções que estão sendo atacadas.

# VerifyBuildingExists (*string* buildingName)

Descrição: Este método é utilizado para verificar se o jogador já possui determinada construção.

Parâmetros:

* *String:* Nome da construção que se deseja verificar a existência.

Retorno: Este método retorna **verdadeiro** se foi possível encontrar a construção solicitada. Caso contrário, retorna **falso**.

# VerifyBuildingsLife ()

Descrição: Este método é utilizado para verificar se há construção com menos vida que o total da mesma.

Parâmetros:

* *Sem parâmetros*.

Retorno: Este método retorna **verdadeiro** se há alguma construção com menos vida que o total. Caso contrário, retorna **falso**.

# GetBuildingsLostLife ()

Descrição: Este método é utilizado para pegar a lista de todas as construções que possuem vida menor que o seu total.

Parâmetros:

* *Sem parâmetros*.

Retorno: Este método retorna uma **lista de construções** com vida inferior ao seu total.

# ExploreMap (*Unit* unit, *Vector2* point)

Descrição: Este método é utilizado para explorar o mapa a partir de um determinado ponto.

Parâmetros:

* *Unit:* Unidade que será enviada para explorar o mapa.
* *Vector2:* Ponto de referência para a exploração.

Retorno: Método sem retorno.

# GetVisibleEnemyBuildings ()

Descrição: Este método é utilizado para pegar a lista de construções inimigas visíveis no mapa.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna uma **lista de construções** com todas as construções inimigas visíveis no mapa.

# GetVisibleEnemyUnits ()

Descrição: Este método é utilizado para pegar a lista de unidades inimigas visíveis no mapa.

Parâmetros:

* *Sem parâmetros*.

Retorno: Retorna uma **lista de unidades** com todas as unidades inimigas visíveis no mapa.

# VerifyVisibleEnemyBuilding (*string* buildingName)

Descrição: Este método é utilizado para verificar a existência de uma construção inimiga.

Parâmetros:

* *String:* Nome da construção que se deseja verificar a existência.

Retorno: Este método retorna **verdadeiro** se há alguma construção com o nome indicado e se a mesma estiver visível. Caso contrário, retorna **falso**.