目次

グローバル配置のプリセット変数

his - 行動履歴オブジェクト

- RoundSetup フェーズに更新されます
- クラスインスタンスです(関数呼び出し時は「:(コロン)」を使う)

プロパティ

- his.xxx の形式で参照します

	型	説明
prev_action	string	直前のaction 例) "TurnLeft"
prev_object	string	直前のobject 例) "Key"
last_move_action	string	最後の移動action 例) "TurnLeft"

```
-- 直前の行動は「左を向く」か？
his.prev_action == "TurnLeft"

-- 直前に使ったのは「鍵」か？
his.prev_object == "Key"
```

- 直前の行動で object を使わなかった場合、his.prev_object の値は nil になります
- 直前の行動が "TurnLeft", "Key" だった場合、次の行動時の his.prev_object の値は "Key" になります

inv - 持ち物オブジェクト

- RoundSetup フェーズに更新されます
- クラスインスタンスです(関数呼び出し時は「:(コロン)」を使う)

プロパティ

- inv.xxx の形式で参照します

	型	説明
count	int	所持しているアイテムの種類数

```
-- アイテムを1つも持っていない？
inv.count < 1
```

関数

- inv:xxx() の形式で呼び出します

IsExist(string name)

- 指定のアイテムを所持しているかどうかを返します

戻り値	bool	所持しているなら true, 所持していないなら false
name	string	所持しているかを確認したいアイテム名

```
-- 鍵を持っているか？
inv:IsExist("Key")
```

GetItemCount(string name)

- 指定のアイテムを何個持っているかを返します

戻り値	int	所持数, そもそも持っていないならゼロを返す
name	string	確認したいアイテム名

```
-- 鍵を何個持っているか？
inv:GetItemCount("Key")
```

GetNameListTable()

- 所持しているアイテムのリストを返します
- アイテム名のリストなので、同じ名前のアイテムは1つにまとめられます

戻り値	LuaTable(配列)	アイテム名のリスト
引数なし		

```
-- 取得例
local ret = inv:GetNameListTable()

-- 戻り値と同等の構造を宣言した場合
ret = {"Key", "Orb"}

-- 戻り値の参照例 ------------------------------

-- "Key" 配列の1つめを指している
ret[1]

-- 格納数なので「2」
#ret
```

map - ウォーカーマップオブジェクト

- RoundSetup フェーズに更新されます
- クラスインスタンスです(関数呼び出し時は「:(コロン)」を使う)

関数

- map:xxx() の形式で呼び出します

Get(int x, int y)

- 指定座標のマップキャラクタ(文字)返します
- 視覚範囲外を指定した場合は "?" を返します

戻り値	string	1文字のマップキャラクタ
x, y	int	ウォーカー座標

```
-- 正面は壁か？
map:Get(0, 1) == "#"
```

CanEnter(int x, int y)

- 指定座標にウォーカーが進入(通過)出来るかを返します
- 視覚範囲外を指定した場合は false を返します

戻り値	bool	進入可能なら true
x, y	int	ウォーカー座標

```
-- 正面に進入できる(壁は無い)か？
map:CanEnter(0, 1)
```

CanPickup(int x, int y)

- 指定座標に拾えるもの(=アイテム)があるかどうかを返します
- 視覚範囲外を指定した場合は false を返します

戻り値	bool	取得可能なら true
x, y	int	ウォーカー座標

```
-- 足元にアイテムが落ちているか？
map:CanPickup(0, 0)
```

IsProwler(int x, int y)

- 指定座標にプロウラーがいるかどうかを返します
- 視覚範囲外を指定した場合は false を返します

戻り値	bool	いる場合は true
x, y	int	ウォーカー座標

```
-- 1歩先に敵がいる？
map:IsProwler(0, 2)
```

グローバル変数

walker - ウォーカー情報テーブル

※ his, inv, map を使っている限り、まず利用することはありません。

- RoundSetup フェーズに更新されます
- プログラムからは walker.~ か _G.walker.~ の形式で参照します
- WorldVM にある walker からコピーされたものですので、LocalVM 上のこれを変更してもゲームに影響しません

構造			型	説明	依存
walker					
-	actions		table	行動選択肢	Host
-	current_mapchip		string	現在地のマップチップ	Host
-	enterable_mapchip		string	進入可能なマップチップ	Host
-	inventory		table	所持アイテム	Host
-	map		table	walker視点でのマップ	Host
-	view		table	walkerの視覚	Host
-	state		table	walkerの状態	Host
-	-	life	int	行動可能回数の残り ゼロになったらクエスト失敗 -1:行動階数無限	Rule(LayerX)

- Host 依存の要素は削除できません
- Rule 依存の要素は Rule で追加/管理しているものです
  - state テーブルそのもの(中身はどうでもいい)は Host 依存です

ppMazeWalkerで使用している用語

RoundSetupフェーズ

ウォーカーの行動入力前に現在の情報を更新する処理です。
クエスト開始から終了まで、ウォーカー行動入力前に毎回実行されます。

ウォーカー座標

ウォーカーの現在地を(0, 0)とし、マップの上下左右に関係なくウォーカーの前後方向をY軸、左右方向をX軸とする座標系です。
前方方向にY増加、右方向にX増加となっています。

(-2, 2)	(-1, 2)	(0, 2) 一歩先	(1, 2)	(2, 2)
(-2, 1)	(-1, 1)	(0, 1) 正面	(1, 1)	(2, 1)
(-2, 0)	(-1, 0) 左	(0, 0) Walker	(1, 0) 右	(2, 0)
(-2, -1)	(-1, -1)	(0, -1) 後ろ	(1, -1)	(2, -1)
(-2, -2)	(-1, -2)	(0, -2)	(1, -2)	(2, -2)

- 2歩先は (0, 4)
- 1歩分左は (-2, 0)

※ ウォーカーは1歩で2マス分移動します、間のマスは壁配置用です