Pathy Function Calls

Declaration Statements

The following must come before a "DecEnd" statement

@[ID]()

Node declaration

[ID] - An unused string identifier

%[ID]()

Junction declaration

[ID] - An unused string identifier

^[ID]([A], [B], {Weight}, {Direction})

Link Declaration

[ID] - An unused string identifier

[A], [B] - string identifiers of nodes or junctions on either end of the link

{Weight} - Optional, Default 0 - An Integer amount of energy required for an entity to use this link.

{Direction} - Optional, Default 0, Weight required if setting direction - Directional assignment. An integer from the following:

0 = Allows travel both ways.

1 = Allows travel from A to B only.

2 = Allows travel from B to A only.

3 = Blocked. No travel permitted.

$[ID]()

Action declaration

[ID] - An unused string identifier

£[ID]([Start])

[ID] - An unused string identifier

[Start] - The Node that the Entity starts on.

AssignAction([Action],[Node])

Assigns an action to a node to show that it is available at that node. If an action has not had this called on them, then it will simply exist, but no nodes will report they can provide it.

[Action] - A string identifier of an existing Action

[Node] - A string identifier of an existing Node.

Warning: shows a warning if [Node] already has that specific action assigned.

Throws: if [Action] and/or [Node] do not exist

DeleteItem([ID])

Removes [ID] from the model if possible

[ID] - A string identifier of an existing item

Warning: shows a warning if [ID] is a Node, and had one or more Actions associated with it.

Warning: shows a warning if [ID] is an Action, and was associated with one or more nodes.

Throws: if [ID] doesn't exist

Throws: if [ID] is a Node or Junction, and is attached to one or more Links

Throws: if [ID] is a Node, and an Entity is currently located at [ID]

Flexi Statements

The following can come before or after a "DecEnd" statement

SetWeight([ID], [Val])

Sets the weight of [ID] to be [Val]

[ID] - A string identifier of an existing Link

[Val] - An integer value

SetLinkOneWay([ID], [A], [B])

Restricts directional travel along [ID] so that it may only be passed going towards [Dest]

[ID] - A string identifier of an existing Link

[Dest] - A string identifiers of one the nodes/junctions on either end of [ID]

Throws: if [ID] or [Dest] doesn't exist

Throws: If [Dest] is not on one end of [ID]

SetLinkTwoWay([ID])

Sets directional travel to be passable both ways along [ID]

[ID] - A string identifier of an existing Link

Throws: if [ID] doesn't exist or isn't a Link

SetLinkBlocked([ID])

Blocks travel along [ID]

[ID] - A string identifier of an existing Link

Throws: if [ID] doesn't exist or isn't a Link

MoveEntity([Entity],[Node])

Changes the location of [Entity] to [Node] without respect to its energy or if it can logically get there

[Entity] - A string identifier of an existing Entity

[Node] - A string identifier of an existing Node

Throws: if [Entity] or [Node] doesn't exist

SetEnergy([ID], [Val])

Sets the energy of [ID] to be [Val]

[ID] - A string identifier of an existing Entity

[Val] - An integer value

Throws: if [ID] doesn't exist

SetJunctDirection([Junction], [A],[B], [Direction])

Sets the directional limitations within [Junction]. If this has not been called on a junction, that junction will default to two-way travel from and to all directions.

[Junction] - a string Identifier of an existing junction.

[A], [B] - string identifiers of links connected to [Junction].

[Direction] - Directional assignment. An string from the following:

TWOWAY = Allows travel both ways.

ATOB = Allows travel from A to B only.

BTOA = Allows travel from B to A only.

BLOCKED = Blocked. No travel permitted.

Throws: if [Jucntion], [A] or [B] does not exist, or [A] and/or [B] are not connected to [Junction]

Query Statements

The following must come after a "DecEnd" statement

PrintNodes()

Lists the identifiers of all the Nodes in the world.

PrintLinks()

Lists the identifiers of all the Links in the world.

PrintJunctions()

Lists the identifiers of all the Junctions in the world.

PrintActions()

Lists the identifiers of all the Actions in the world.

PrintEntities()

Lists the identifiers of all the Entities in the world.

LinkedTo([Place])

Lists the identifiers of the Junctions and Nodes that [Place] links directly to, regardless of travel restrictions on those links.

[Place] - A string identifier of an existing Node or Junction.

ConnectedBy([Place])

Lists the identifiers of the Links connected to [Place]

[Place] - A string identifier of an existing Node or Junction.

TypeOf([ID])

Returns the type of the ID supplied

ActionsThere([Node])

Lists the identifiers of the Actions associated with [Node]

[Node] - A string identifier of an existing Node

Links([Link])

Lists the identifiers of the items at either end of [Link]

[Link] - A string identifier of an existing Link

Direction([Link])

Returns the directional orientation of [Link]. One-way travel is returned in respect to the starting endpoint.

[Link] - A string identifier of an existing Link

Weight([Link])

Returns the weight of [Link]

[Link] - A string identifier of an existing Link

Connectivity([Junct])

Lists the internal direction allowances for [Junct].

[Junct] - A string identifier of an existing Junction

WhereIs([Entity])

Returns the Node that [Entity] is located at

[Entity] - A string identifier of an existing Entity

Availability([Action])

Returns a lits of Nodes where [Action] is available

[Action] - A string Identifier of an existing Action

SharedActions([A], [B])

Lists the actions that [A] and [B] have in common.

[A], [B] - string identifiers of two different existing nodes

Not Implemented yet

The following exist, but will not produce useful output in this version of Pathy.

PathTo([A], [B])

Returns true you can get from [A] to [B], false if you cannot. Ignores energy costs. Respects Link travel restrictions.

[A], [B] - string identifiers of two different existing nodes

CanMove([Entity], [Node])

Returns True if [Entity] can logically move to [Node], False if it cannot. Respects Energy requirements and Link directional allowance.

[Entity] - A string identifier of an existing Entity

[Node] - A string identifier of an existing Node

CanMoveWhy([Entity], [Node])

Returns True if [Entity] can logically move to [Node], returns a verbose reason if it cannot. Respects Energy requirements and Link directional allowance.

[Entity] - A string identifier of an existing Entity

[Node] - A string identifier of an existing Node

EnergyReq([A], [B])

Returns the lowest energy requirement of all routes from [A] to [B]. Returns -1 if no route. Respects Link directional allowance.

[A], [B] - string identifiers of two different existing nodes