

Unblock Me Solver

API Documentation

July 20, 2017

Contents

Contents	1
1 Package UnblockMeSolver	2
1.1 Modules	2
1.2 Variables	2
2 Package UnblockMeSolver.Map	3
2.1 Modules	3
2.2 Variables	3
3 Module UnblockMeSolver.Map.Map	4
3.1 Variables	4
3.2 Class Map	4
3.2.1 Methods	4
3.2.2 Properties	8
3.2.3 Class Variables	8
4 Module UnblockMeSolver.Map.MapReader	9
4.1 Variables	9
4.2 Class MapReader	9
4.2.1 Methods	9
4.2.2 Properties	9
4.2.3 Class Variables	10
5 Module UnblockMeSolver.Map.Move	11
5.1 Variables	11
5.2 Class Move	11
5.2.1 Methods	11
5.2.2 Properties	13
6 Module UnblockMeSolver.Map.Piece	14
6.1 Variables	14
6.2 Class Piece	14
6.2.1 Methods	14
6.2.2 Properties	15
7 Package UnblockMeSolver.PathFinder	16
7.1 Modules	16

7.2	Variables	16
8	Module UnblockMeSolver.PathFinder.AStar	17
8.1	Variables	17
8.2	Class AStar	17
8.2.1	Methods	17
8.2.2	Properties	18
9	Module UnblockMeSolver.PathFinder.AStarNode	19
9.1	Variables	19
9.2	Class AStarNode	19
9.2.1	Methods	19
9.2.2	Properties	20
9.2.3	Class Variables	20
10	Module UnblockMeSolver.PathFinder.Heuristics	21
10.1	Functions	21
10.2	Variables	21
11	Module UnblockMeSolver.PathFinder.Node	22
11.1	Variables	22
11.2	Class Node	22
11.2.1	Methods	22
11.2.2	Properties	23
12	Module UnblockMeSolver.PathFinder.PathFinder	24
12.1	Variables	24
12.2	Class PathFinder	24
12.2.1	Methods	24
12.2.2	Properties	24
13	Module UnblockMeSolver.PathFinder.Solver	26
13.1	Variables	26
14	Module UnblockMeSolver.PathFinder.bfs	27
14.1	Variables	27
14.2	Class BFS	27
14.2.1	Methods	27
14.2.2	Properties	28
15	Module UnblockMeSolver.PathFinder.dfs	29
15.1	Variables	29
15.2	Class DFS	29
15.2.1	Methods	29
15.2.2	Properties	30
16	Package UnblockMeSolver.test	31
16.1	Modules	31
16.2	Variables	31
17	Package UnblockMeSolver.test.MapTests	32
17.1	Modules	32
17.2	Variables	32

18 Module UnblockMeSolver.test.MapTests.mapTests	33
18.1 Variables	33
18.2 Class FakeMapReader	33
18.2.1 Methods	33
18.2.2 Properties	33
18.3 Class TestMap	34
18.3.1 Methods	34
18.3.2 Properties	35
18.3.3 Class Variables	36
19 Module UnblockMeSolver.test.MapTests.moveTests	37
19.1 Variables	37
19.2 Class TestMove	37
19.2.1 Methods	37
19.2.2 Properties	38
19.2.3 Class Variables	38
20 Module UnblockMeSolver.test.MapTests.pieceTests	39
20.1 Variables	39
20.2 Class TestPiece	39
20.2.1 Methods	39
20.2.2 Properties	40
20.2.3 Class Variables	40
21 Module UnblockMeSolver.test.MapTests.readerTests	41
21.1 Variables	41
21.2 Class TestReader	41
21.2.1 Methods	41
21.2.2 Properties	42
21.2.3 Class Variables	42
22 Package UnblockMeSolver.test.PathFinding	43
22.1 Modules	43
22.2 Variables	43
23 Module UnblockMeSolver.test.PathFinding.AStarNodeTests	44
23.1 Variables	44
23.2 Class NodeTest	44
23.2.1 Methods	44
23.2.2 Properties	44
23.2.3 Class Variables	45
24 Module UnblockMeSolver.test.PathFinding.heuristicTests	46
24.1 Variables	46
24.2 Class TestMap	46
24.2.1 Methods	46
24.2.2 Properties	47
24.2.3 Class Variables	47
25 Module UnblockMeSolver.test.PathFinding.nodeTests	48
25.1 Variables	48
25.2 Class NodeTest	48
25.2.1 Methods	48

25.2.2	Properties	48
25.2.3	Class Variables	49
26	Module UnblockMeSolver.test.PathFinding.pathFindingTests	50
26.1	Functions	50
26.2	Variables	50
26.3	Class PathFindingTest	50
26.3.1	Methods	50
26.3.2	Properties	51
26.3.3	Class Variables	51
27	Module UnblockMeSolver.test.files	52
27.1	Variables	52

1 Package UnblockMeSolver

1.1 Modules

- **Map** (Section 2, p. 3)
 - **Map** (Section 3, p. 4)
 - **MapReader** (Section 4, p. 9)
 - **Move** (Section 5, p. 11)
 - **Piece** (Section 6, p. 14)
- **PathFinder** (Section 7, p. 16)
 - **AStar** (Section 8, p. 17)
 - **AStarNode** (Section 9, p. 19)
 - **Heuristics** (Section 10, p. 21)
 - **Node** (Section 11, p. 22)
 - **PathFinder** (Section 12, p. 24)
 - **Solver** (Section 13, p. 26)
 - **bfs** (Section 14, p. 27)
 - **dfs** (Section 15, p. 29)
- **test** (Section 16, p. 31)
 - **MapTests** (Section 17, p. 32)
 - * **mapTests** (Section 18, p. 33)
 - * **moveTests** (Section 19, p. 37)
 - * **pieceTests** (Section 20, p. 39)
 - * **readerTests** (Section 21, p. 41)
 - **PathFinding** (Section 22, p. 43)
 - * **AStarNodeTests** (Section 23, p. 44)
 - * **heuristicTests** (Section 24, p. 46)
 - * **nodeTests** (Section 25, p. 48)
 - * **pathFindingTests** (Section 26, p. 50)
 - **files** (Section 27, p. 52)

1.2 Variables

Name	Description
<code>--package--</code>	Value: None

2 Package UnBlockMeSolver.Map

2.1 Modules

- **Map** (*Section 3, p. 4*)
- **MapReader** (*Section 4, p. 9*)
- **Move** (*Section 5, p. 11*)
- **Piece** (*Section 6, p. 14*)

2.2 Variables

Name	Description
<code>--package--</code>	Value: None

3 Module UnblockMeSolver.Map.Map

3.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.Map'</code>

3.2 Class Map



3.2.1 Methods

`__init__(self, graph, delimiter='\n')`

Initialize the class by setting the graph and delimiters variables.

Parameters

graph: Represents the board that the game will be played on
(*type=string | MapReader*)

delimiter: how to split the string
(*type=string*)

Overrides: `object.__init__`

`convertToMap(self)`

Convert the graph into a matrix that can be viewed and modified.

`setUpPieces(self)`

Find every piece available in the board and add it to an array to keep track of them for future use.

`setUp(self)`

Set up the Map class and set the graph by either getting a a string from either a map reader or string as the graph and breaking if not valid.

isWallOrGoal(*self, point, num_goals_found*)

This will check if the point is a wall or goal and if not return False. Additionally, it will update the number of goals that have been found.

Parameters

point: the point being checked in the graph.
(type=character)

num_goals_found: number of goals in the graph currently found
(type=integer)

Return Value

if the point is a wall or a goal and number of goals currently found
(type=boolean, integer)

largeRowValid(*self, row_index, num_goals_found*)

This will test on the larger rows to see if it is valid. A larger row indicates that it is either the top or the bottom of the graph.

Parameters

row_index: index for the row to be checked
(type=integer)

num_goals_found: the number of goals that have been found
(type=integer)

Return Value

If a large row provided is valid and number of goals currently found
(type=boolean, integer)

numColumnsMatch(*self*)

Ensure that the number of columns matches for each row.

Return Value

if the number of columns matches for each row
(type=boolean)

topBottomRowsValid(*self, num_goals_found*)

Make sure that both the top and bottom rows are valid according to the topBottomRowsValid function.

Parameters

num_goals_found: number of goals presently found
(type=integer)

Return Value

If the top and bottom rows are valid and number of goals currently found
(type=boolean, integer)

midRowsValid(*self*, *num_goals_found*)

Ensure that all the middle rows are valid entries with walls or a goal on both sides.

Parameters

num_goals_found: number of goals presently found
(*type=integer*)

Return Value

if the middle rows are valid and the number of goals currently found
(*type=boolean, integer*)

playerFound(*self*)

Check to see if the player, defined in Map.py, is in the map.

Return Value

player found in map
(*type=boolean*)

isValid(*self*)

Test if the graph in the map class is valid or not.

Return Value

graph is valid or not
(*type=boolean*)

validAdditionMoves(*self*, *move*)

This will handle move verificatin for moving to the right or down in the board. It will return an array of the the moves that can be made in the direction given.

Parameters

move: Array of moves in the direction of the move given
(*type=[Move]*)

validSubtractionMoves(*self*, *move*)

This will handle move verificatin for moving to the left or up in the board. It will return an array of the the moves that can be made in the direction given.

Parameters

move: Array of moves in the direction of the move given
(*type=[Move]*)

isValidMove(*self*, *move*)

Check if the move given is valid or not to make the move.

Return Value

if the move is valid or not
(*type=boolean*)

makeConfidentMove(*self*, *move*)

Make a move on the board without checking for its validity.

Parameters

move: move to make on the board
(*type=Move*)

makeMove(*self*, *move*)

Make a move on the board. Raise syntax error on bad move given

Parameters

move: move to make on the board
(*type=Move*)

getMoves(*self*)

Get the moves available on the board.

Return Value

Array of moves available
(*type=[Move]*)

isSolved(*self*)

If there is a goal found then the board is not solved. Complete checking is done as there may be a goal in any location.

Return Value

if the game has been solved or not
(*type=boolean*)

copy(*self*)

Create a copy of this map and return it

Return Value

Copy of this map
(*type=Map*)

copyMove(*self*, *move*)

Create a copy of the map and make a move on it; return the result.

Parameters

move: move to be made on copied board
(*type=Move*)

Return Value

The new map with the move made on it
(*type=Map*)

copyConfidentMove(*self*, *move*)

Create a copy of the map and make a move on it without checking for its validity. Return the result.

Parameters

move: move to be made on copied board
(*type=Move*)

Return Value

The new map with the move made on it
(*type=Map*)

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

3.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

3.2.3 Class Variables

Name	Description
wall	Value: ' '
goal	Value: '\$'
empty	Value: '0'
player	Value: '**'
playerPiece	Value: '*'

4 Module UnblockMeSolver.Map.MapReader

4.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.Map'</code>

4.2 Class MapReader



4.2.1 Methods

<code>--init--(self)</code>
Initialize MapReader class
Overrides: <code>object.--init--</code>

<code>load(self, file_name)</code>
Set flags and test if the file given exists. Throws IOError on file not existing.
Parameters
<code>file_name</code>: Name of the file that contains definition of the map.
<i>(type=string)</i>

<code>get(self)</code>
Read the file and return the string inside
Return Value
string inside of the specified file
<i>(type=string)</i>

Inherited from object

`--delattr--()`, `--format--()`, `--getattr--()`, `--hash--()`, `--new--()`, `--reduce--()`, `--reduce_ex--()`,
`--repr--()`, `--setattr--()`, `--sizeof--()`, `--str--()`, `--subclasshook--()`

4.2.2 Properties

Name	Description
<i>Inherited from object</i> __class__	

4.2.3 Class Variables

Name	Description
loaded	Value: False
file_found	Value: False

5 Module UnblockMeSolver.Map.Move

5.1 Variables

Name	Description
<code>--package--</code>	Value: None

5.2 Class Move



5.2.1 Methods

<code>--init--</code> (<i>self</i> , <i>piece</i> , <i>right</i> , <i>up</i>)
Initialize data structure with required variables. Throws an error on incorrect value ranges and types given
Parameters
<i>piece</i> : The piece to be moved (<i>type=character</i>)
<i>right</i> : If the piece should be to the right by a space (<i>type=integer</i>)
<i>up</i> : If the piece should be moved up by a space (<i>type=integer</i>)
Overrides: <code>object.--init--</code>

<code>isValid</code> (<i>self</i>)
Ensure that the move is valid for the board and the piece is a character.

equals(*self*, *move*)

Test whether the two moves are equal.

Parameters

move: Move being tested for equality
(*type=Move*)

Return Value

Whether the two moves are equal
(*type=boolean*)

toStr(*self*)

Convert self to string.

Return Value

String representation of the move
(*type=str*)

left(*piece*, *size=1*)

Instantiate a move that will go left

Parameters

piece: Character that represents the piece to be moved
(*type=character*)

right(*piece*, *size=1*)

Instantiate a move that will go right

Parameters

piece: Character that represents the piece to be moved
(*type=character*)

up(*piece*, *size=1*)

Instantiate a move that will go up

Parameters

piece: Character that represents the piece to be moved
(*type=character*)

down (<i>piece</i> , <i>size=1</i>)
--

Instantiate a move that will go down

Parameters

<p>piece: Character that represents the piece to be moved <i>(type=character)</i></p>

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`,
`__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

5.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

6 Module UnblockMeSolver.Map.Piece

6.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.Map'</code>

6.2 Class Piece



6.2.1 Methods

<code>--init--</code> <i>(self, x, y, vertical)</i>
Initialize piece class
Parameters
x: x coordinate <i>(type=integer)</i>
y: y coordinate <i>(type=integer)</i>
vertical: marks whether this piece can move vertically or not <i>(type=boolean)</i>
Overrides: <code>object.--init--</code>

<code>move</code> <i>(self, move)</i>
Move the piece if the given move is valid
Parameters
move: Move that will change the location of the piece <i>(type=Move)</i>

copy (<i>self</i>)

Create a copy of self.

Return Value

Copy of this piece

<i>(type=Piece)</i>

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`,
`__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

6.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

7 Package *UnBlockMeSolver.PathFinder*

7.1 Modules

- **AStar** (*Section 8, p. 17*)
- **AStarNode** (*Section 9, p. 19*)
- **Heuristics** (*Section 10, p. 21*)
- **Node** (*Section 11, p. 22*)
- **PathFinder** (*Section 12, p. 24*)
- **Solver** (*Section 13, p. 26*)
- **bfs** (*Section 14, p. 27*)
- **dfs** (*Section 15, p. 29*)

7.2 Variables

Name	Description
--package--	Value: None

8 Module UnblockMeSolver.PathFinder.AStar

8.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.PathFinder'</code>

8.2 Class AStar

object

UnblockMeSolver.PathFinder.PathFinder.PathFinder

UnblockMeSolver.PathFinder.AStar.AStar

8.2.1 Methods

heuristic(*self*, *board*)

Calculate heuristic cost for the given board.

Parameters

board: Board being analyzed
(*type=Map*)

Return Value

Heuristic cost
(*type=number*)

populateQueue(*self*, *graph*, *queue*, *parent*)

Populate the priority queue with nodes and their cost.

Parameters

graph: Graph to get moves from to populate queue
(*type=Map*)

queue: Queue with nodes being added to it with the cost
(*type=Queue*)

parent: Parent for the nodes that will be created
(*type=AStarNode*)

getPath(self)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle

(type=[Move])

Overrides: UnblockMeSolver.PathFinder.PathFinder.getPath

Inherited from UnblockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

__init__()

Inherited from object

__delattr__(), __format__(), __getattr__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(),
__repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

8.2.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

9 Module UnblockMeSolver.PathFinder.AStarNode

9.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.PathFinder'</code>

9.2 Class AStarNode

object └─

UnblockMeSolver.PathFinder.Node.Node └─

UnblockMeSolver.PathFinder.AStarNode.AStarNode

9.2.1 Methods

<code>--init--</code> (<i>self</i> , <i>graph</i> , <i>move</i> , <i>parent</i> , <i>g</i> , <i>h</i>)
Initialize AStarNode class
Parameters
graph: Map that moves can be made on (<i>type</i> = <i>Map</i>)
move: Move that resulted in the above graph (<i>type</i> = <i>Move</i>)
parent: Parent node of this. Contains previous moves to get the current board. (<i>type</i> = <i>AStarNode</i>)
g: Step cost for this node (<i>type</i> = <i>number</i>)
h: Heuristic cost for this node (<i>type</i> = <i>number</i>)
Overrides: <code>object.__init__</code>

cost (<i>self</i>)
Cost for this node.
Return Value
The cost to get to this state
(<i>type=number</i>)

Inherited from UnblockMeSolver.PathFinder.Node.Node(Section 11.2)

reconstructPath()

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

9.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

9.2.3 Class Variables

Name	Description
<code>g</code>	Value: 0
<code>h</code>	Value: 0

10 Module `UnBlockMeSolver.PathFinder.Heuristics`

10.1 Functions

<code>manhattan($x1$, $y1$, $x2$, $y2$)</code>
--

<code>euclidian($x1$, $y1$, $x2$, $y2$)</code>
--

10.2 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnBlockMeSolver.PathFinder'</code>

11 Module UnblockMeSolver.PathFinder.Node

11.1 Variables

Name	Description
<code>--package--</code>	Value: None

11.2 Class Node



Known Subclasses: UnblockMeSolver.PathFinder.AStarNode.AStarNode

11.2.1 Methods

<code>--init--(self, graph, move, parent)</code> Initialize AStarNode class Parameters graph: Map that moves can be made on (<i>type=Map</i>) move: Move that resulted in the above graph (<i>type=Move</i>) parent: Parent node of this. Contains previous moves to get the current board. (<i>type=Node</i>) Overrides: object. <code>--init--</code>
<code>reconstructPath(self)</code> Construct the path from here that it took to make it here. Return Value array of moves to get to this point from the root node (<i>type=[Move]</i>)

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`,
`__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

11.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

12 Module *UnBlockMeSolver.PathFinder.PathFinder*

12.1 Variables

Name	Description
<code>--package--</code>	Value: None

12.2 Class *PathFinder*



Known Subclasses: *UnBlockMeSolver.PathFinder.bfs.BFS*, *UnBlockMeSolver.PathFinder.dfs.DFS*, *UnBlockMeSolver.PathFinder.AStar.AStar*

12.2.1 Methods

<code>--init--</code> (<i>self</i> , <i>board</i>)
Initialize Pathfinder class instance.
Parameters
<i>board</i> : board to be solved
(<i>type=Map</i>)
Overrides: <i>object.--init--</i>

<code>getPath</code> (<i>self</i>)
Get the best path to solve the given graph.
Return Value
Array of moves which represent the path found to solve the puzzle
(<i>type=[Move]</i>)

Inherited from object

`--delattr--()`, `--format--()`, `--getattribute--()`, `--hash--()`, `--new--()`, `--reduce--()`, `--reduce_ex--()`,
`--repr--()`, `--setattr--()`, `--sizeof--()`, `--str--()`, `--subclasshook--()`

12.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>--class--</code>	

13 Module UnBlockMeSolver.PathFinder.Solver

13.1 Variables

Name	Description
<code>--package--</code>	Value: None

14 Module UnblockMeSolver.PathFinder.bfs

14.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.PathFinder'</code>

14.2 Class BFS

object

UnblockMeSolver.PathFinder.PathFinder.PathFinder

UnblockMeSolver.PathFinder.bfs.BFS

14.2.1 Methods

populateQueue(*self*, *graph*, *move*, *queue*, *parent*)

Populate the queue with nodes and their cost.

Parameters

graph: Graph to get moves from to populate queue
(type=Map)

move: Move to get to the
(type=Move)

queue: Stack with nodes being added to it with the cost
(type=Queue)

parent: Parent for the nodes that will be created
(type=Node)

getPath(*self*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle
(type=[Move])

Overrides: UnblockMeSolver.PathFinder.PathFinder.PathFinder.getPath

Inherited from UnBlockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

`__init__()`

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

14.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

15 Module UnblockMeSolver.PathFinder.dfs

15.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.PathFinder'</code>

15.2 Class DFS



15.2.1 Methods

populateQueue (<i>self</i> , <i>graph</i> , <i>move</i> , <i>stack</i> , <i>parent</i>)
Populate the stack with nodes and their cost.
Parameters
graph: Graph to get moves from to populate queue (<i>type</i> = <i>Map</i>)
move: Move to get to the (<i>type</i> = <i>Move</i>)
stack: Stack with nodes being added to it with the cost (<i>type</i> = <i>[Node]</i>)
parent: Parent for the nodes that will be created (<i>type</i> = <i>Node</i>)

getPath (<i>self</i>)
Get the best path to solve the given graph.
Return Value
Array of moves which represent the path found to solve the puzzle (<i>type</i> = <i>[Move]</i>)
Overrides: <code>UnblockMeSolver.PathFinder.PathFinder.PathFinder.getPath</code>

Inherited from UnblockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

`__init__()`

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`, `__str__()`, `__subclasshook__()`

15.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

16 Package UnblockMeSolver.test

16.1 Modules

- **MapTests** (*Section 17, p. 32*)
 - **mapTests** (*Section 18, p. 33*)
 - **moveTests** (*Section 19, p. 37*)
 - **pieceTests** (*Section 20, p. 39*)
 - **readerTests** (*Section 21, p. 41*)
- **PathFinding** (*Section 22, p. 43*)
 - **AStarNodeTests** (*Section 23, p. 44*)
 - **heuristicTests** (*Section 24, p. 46*)
 - **nodeTests** (*Section 25, p. 48*)
 - **pathFindingTests** (*Section 26, p. 50*)
- **files** (*Section 27, p. 52*)

16.2 Variables

Name	Description
--package--	Value: None

17 Package `UnBlockMeSolver.test.MapTests`

17.1 Modules

- `mapTests` (*Section 18, p. 33*)
- `moveTests` (*Section 19, p. 37*)
- `pieceTests` (*Section 20, p. 39*)
- `readerTests` (*Section 21, p. 41*)

17.2 Variables

Name	Description
<code>--package--</code>	Value: <code>None</code>

18 Module UnblockMeSolver.test.MapTests.mapTests

18.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnblockMeSolver.test.MapTests'</code>

18.2 Class FakeMapReader

object └─ **UnblockMeSolver.test.MapTests.mapTests.FakeMapReader**

Fake object with method and member similar to mapreader class that the map class relies on

18.2.1 Methods

```
--init--(self)

x.--init--(...) initializes x; see help(type(x)) for signature
Overrides: object.--init-- extit(inherited documentation)
```

```
get(self)
```

Inherited from object

```
--delattr--(), --format--(), --getattribute--(), --hash--(), --new--(), --reduce--(), --reduce_ex--(),
--repr--(), --setattr--(), --sizeof--(), --str--(), --subclasshook--()
```

18.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>--class--</code>	

18.3 Class *TestMap*



18.3.1 Methods

<code>test_setUpPieces(<i>self</i>)</code>
--

<code>test_init(<i>self</i>)</code>

<code>test_convertToMap(<i>self</i>)</code>

<code>test_setUp(<i>self</i>)</code>

<code>test_isWallOrGoal(<i>self</i>)</code>

<code>test_largeRowValid(<i>self</i>)</code>
--

<code>test_numColumnsMatch(<i>self</i>)</code>
--

<code>test_topBottomRowsValid(<i>self</i>)</code>

<code>test_midRowsValid(<i>self</i>)</code>

<code>test_playerFound(<i>self</i>)</code>
--

<code>test_isValid(<i>self</i>)</code>
--

<code>validate_move(<i>self</i>, <i>move1</i>, <i>move2</i>)</code>

<code>validate_moves(<i>self</i>, <i>moves1</i>, <i>moves2</i>)</code>
--

<code>test_isValidMove(<i>self</i>)</code>
--

<code>test_makeConfidentMove(<i>self</i>)</code>
--

<code>test_makeMove(self)</code>

<code>test_getMoves(self)</code>

<code>test_isSolved(self)</code>

<code>test_copy(self)</code>

<code>test_copyMove(self)</code>

<code>test_copyConfidentMove(self)</code>

<code>test_validSubractionMoves(self)</code>
--

<code>test_validAdditionMoves(self)</code>
--

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEquals(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattr__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

18.3.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

18.3.3 Class Variables

Name	Description
sample_string	Value: ' \n 000 \n **0\$\n ',
default_delimeter	Value: '\n'
new_delimeter	Value: ', '
<i>Inherited from unittest.case.TestCase</i> longMessage, maxDiff	

19 Module `UnBlockMeSolver.test.MapTests.moveTests`

19.1 Variables

Name	Description
<code>__package__</code>	Value: <code>'UnBlockMeSolver.test.MapTests'</code>

19.2 Class `TestMove`



19.2.1 Methods

<code>test_init(self)</code>

<code>test_isValid(self)</code>

<code>test_equals(self)</code>

<code>test_toStr(self)</code>

<code>test_left(self)</code>

<code>test_right(self)</code>

<code>test_up(self)</code>

<code>test_down(self)</code>

Inherited from `unittest.case.TestCase`

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`,

assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattr__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

19.2.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

19.2.3 Class Variables

Name	Description
<i>Inherited from unittest.case.TestCase</i>	
longMessage, maxDiff	

20 Module `UnBlockMeSolver.test.MapTests.pieceTests`

20.1 Variables

Name	Description
<code>__package__</code>	Value: <code>'UnBlockMeSolver.test.MapTests'</code>

20.2 Class `TestPiece`



20.2.1 Methods

<code>test_init(self)</code>

<code>test_move(self)</code>

Inherited from `unittest.case.TestCase`

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`, `assertRaises()`, `assertRaisesRegexp()`, `assertRegexpMatches()`, `assertSequenceEqual()`, `assertSetEqual()`, `assertTrue()`, `assertTupleEqual()`, `assert_()`, `countTestCases()`, `debug()`, `defaultTestResult()`, `doCleanups()`, `fail()`, `failIf()`, `failIfAlmostEqual()`, `failIfEqual()`, `failUnless()`, `failUnlessAlmostEqual()`, `failUnlessEqual()`, `failUnlessRaises()`, `id()`, `run()`, `setUp()`, `setUpClass()`, `shortDescription()`, `skipTest()`, `tearDown()`, `tearDownClass()`

Inherited from `object`

`__delattr__()`, `__format__()`, `__getattr__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__setattr__()`, `__sizeof__()`, `__subclasshook__()`

20.2.2 Properties

Name	Description
<i>Inherited from object</i> __class__	

20.2.3 Class Variables

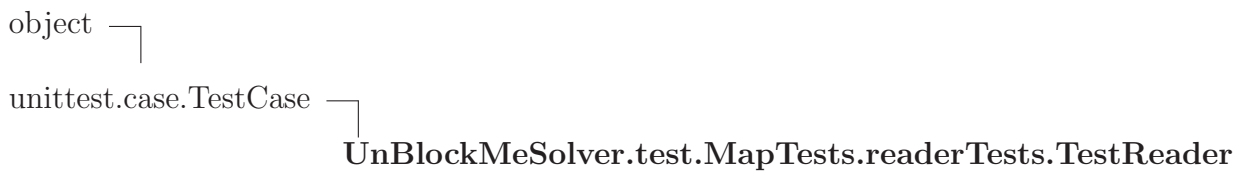
Name	Description
<i>Inherited from unittest.case.TestCase</i> longMessage, maxDiff	

21 Module `UnBlockMeSolver.test.MapTests.readerTests`

21.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnBlockMeSolver.test.MapTests'</code>

21.2 Class `TestReader`



21.2.1 Methods

<code>test_is_loaded(self)</code>

<code>test_is_found(self)</code>

<code>test_get(self)</code>

Inherited from `unittest.case.TestCase`

`--call--()`, `--eq--()`, `--hash--()`, `--init--()`, `--ne--()`, `--repr--()`, `--str--()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`, `assertRaises()`, `assertRaisesRegexp()`, `assertRegexpMatches()`, `assertSequenceEqual()`, `assertSetEqual()`, `assertTrue()`, `assertTupleEqual()`, `assert_()`, `countTestCases()`, `debug()`, `defaultTestResult()`, `doCleanups()`, `fail()`, `failIf()`, `failIfAlmostEqual()`, `failIfEqual()`, `failUnless()`, `failUnlessAlmostEqual()`, `failUnlessEqual()`, `failUnlessRaises()`, `id()`, `run()`, `setUp()`, `setUpClass()`, `shortDescription()`, `skipTest()`, `tearDown()`, `tearDownClass()`

Inherited from `object`

`--delattr--()`, `--format--()`, `--getattrattribute--()`, `--new--()`, `--reduce--()`, `--reduce_ex--()`,

`__setattr__()`, `__sizeof__()`, `__subclasshook__()`

21.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

21.2.3 Class Variables

Name	Description
<i>Inherited from unittest.case.TestCase</i>	
<code>longMessage</code> , <code>maxDiff</code>	

22 Package UnblockMeSolver.test.PathFinding

22.1 Modules

- **AStarNodeTests** (*Section 23, p. 44*)
- **heuristicTests** (*Section 24, p. 46*)
- **nodeTests** (*Section 25, p. 48*)
- **pathFindingTests** (*Section 26, p. 50*)

22.2 Variables

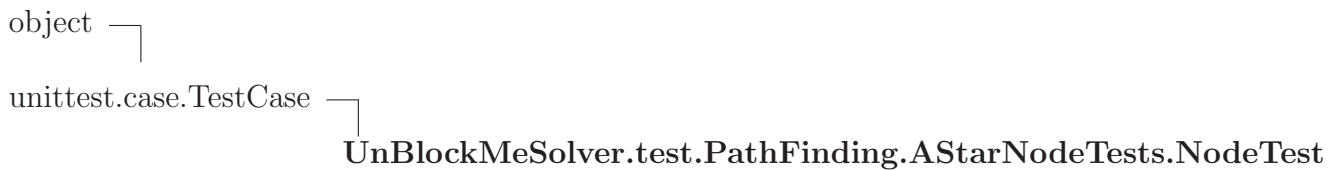
Name	Description
<code>--package--</code>	Value: None

23 Module `UnBlockMeSolver.test.PathFinding.AStarNodeTests`

23.1 Variables

Name	Description
<code>__package__</code>	Value: <code>'UnBlockMeSolver.test.PathFinding'</code>

23.2 Class `NodeTest`



23.2.1 Methods

<code>test_cost(self)</code>

Inherited from `unittest.case.TestCase`

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`, `assertRaises()`, `assertRaisesRegexp()`, `assertRegexpMatches()`, `assertSequenceEqual()`, `assertSetEqual()`, `assertTrue()`, `assertTupleEqual()`, `assert_()`, `countTestCases()`, `debug()`, `defaultTestResult()`, `doCleanups()`, `fail()`, `failIf()`, `failIfAlmostEqual()`, `failIfEqual()`, `failUnless()`, `failUnlessAlmostEqual()`, `failUnlessEqual()`, `failUnlessRaises()`, `id()`, `run()`, `setUp()`, `setUpClass()`, `shortDescription()`, `skipTest()`, `tearDown()`, `tearDownClass()`

Inherited from `object`

`__delattr__()`, `__format__()`, `__getattr__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__setattr__()`, `__sizeof__()`, `__subclasshook__()`

23.2.2 Properties

Name	Description
<i>Inherited from object</i> __class__	

23.2.3 Class Variables

Name	Description
<i>Inherited from unittest.case.TestCase</i> longMessage, maxDiff	

24 Module `UnBlockMeSolver.test.PathFinding.heuristicTests`

24.1 Variables

Name	Description
<code>--package--</code>	Value: <code>'UnBlockMeSolver.test.PathFinding'</code>

24.2 Class `TestMap`



24.2.1 Methods

`test_manhattan(self)`

`test_euclidian(self)`

Inherited from `unittest.case.TestCase`

`--call--()`, `--eq--()`, `--hash--()`, `--init--()`, `--ne--()`, `--repr--()`, `--str--()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`, `assertRaises()`, `assertRaisesRegexp()`, `assertRegexpMatches()`, `assertSequenceEqual()`, `assertSetEqual()`, `assertTrue()`, `assertTupleEqual()`, `assert_()`, `countTestCases()`, `debug()`, `defaultTestResult()`, `doCleanups()`, `fail()`, `failIf()`, `failIfAlmostEqual()`, `failIfEqual()`, `failUnless()`, `failUnlessAlmostEqual()`, `failUnlessEqual()`, `failUnlessRaises()`, `id()`, `run()`, `setUp()`, `setUpClass()`, `shortDescription()`, `skipTest()`, `tearDown()`, `tearDownClass()`

Inherited from `object`

`--delattr--()`, `--format--()`, `--getattribute--()`, `--new--()`, `--reduce--()`, `--reduce_ex--()`, `--setattr--()`, `--sizeof--()`, `--subclasshook--()`

24.2.2 Properties

Name	Description
<i>Inherited from object</i> __class__	

24.2.3 Class Variables

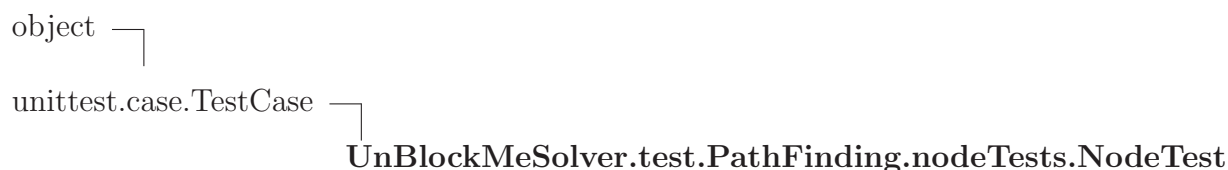
Name	Description
<i>Inherited from unittest.case.TestCase</i> longMessage, maxDiff	

25 Module `UnBlockMeSolver.test.PathFinding.nodeTests`

25.1 Variables

Name	Description
<code>__package__</code>	Value: <code>'UnBlockMeSolver.test.PathFinding'</code>

25.2 Class `NodeTest`



25.2.1 Methods

<code>test_reconstructPath(<i>self</i>)</code>
--

Inherited from `unittest.case.TestCase`

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`, `assertRaises()`, `assertRaisesRegexp()`, `assertRegexpMatches()`, `assertSequenceEqual()`, `assertSetEqual()`, `assertTrue()`, `assertTupleEqual()`, `assert_()`, `countTestCases()`, `debug()`, `defaultTestResult()`, `doCleanups()`, `fail()`, `failIf()`, `failIfAlmostEqual()`, `failIfEqual()`, `failUnless()`, `failUnlessAlmostEqual()`, `failUnlessEqual()`, `failUnlessRaises()`, `id()`, `run()`, `setUp()`, `setUpClass()`, `shortDescription()`, `skipTest()`, `tearDown()`, `tearDownClass()`

Inherited from `object`

`__delattr__()`, `__format__()`, `__getattr__()`, `__new__()`, `__reduce__()`, `__reduce_ex__()`, `__setattr__()`, `__sizeof__()`, `__subclasshook__()`

25.2.2 Properties

Name	Description
<i>Inherited from object</i> __class__	

25.2.3 Class Variables

Name	Description
<i>Inherited from unittest.case.TestCase</i> longMessage, maxDiff	

26 Module *UnBlockMeSolver.test.PathFinding.pathFindingTests*

26.1 Functions

<code>testPaths(<i>self</i>, <i>solver</i>, <i>checkLength</i>=True)</code>

26.2 Variables

Name	Description
<code>--package--</code>	Value: 'UnBlockMeSolver.test.PathFinding'

26.3 Class *PathFindingTest*

```

object └─
unittest.case.TestCase └─
                                UnBlockMeSolver.test.PathFinding.pathFindingTests.PathFindingTest

```

26.3.1 Methods

<code>solve(<i>self</i>, <i>board</i>, <i>moves</i>)</code>

<code>test_pathFinder(<i>self</i>)</code>

<code>test_dfs(<i>self</i>)</code>

<code>test_bfs(<i>self</i>)</code>

<code>test_AStar(<i>self</i>)</code>

Inherited from unittest.case.TestCase

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`,

assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattr__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

26.3.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

26.3.3 Class Variables

Name	Description
<i>Inherited from unittest.case.TestCase</i>	
longMessage, maxDiff	

27 Module `UnBlockMeSolver.test.files`

27.1 Variables

Name	Description
<code>good</code>	Value: <code>'test/Maps/good_map.map'</code>
<code>not_real</code>	Value: <code>'test/Maps/doesnt_exist.map'</code>
<code>sample</code>	Value: <code>'test/Maps/sample.string.map'</code>
<code>no_player</code>	Value: <code>'test/Maps/no_player.map'</code>
<code>empty</code>	Value: <code>'test/Maps/empty_map.map'</code>
<code>bad_col</code>	Value: <code>'test/Maps/bad_map.map'</code>
<code>impossible</code>	Value: <code>'test/Maps/impossible.map'</code>
<code>right</code>	Value: <code>'test/Maps/right.map'</code>
<code>left</code>	Value: <code>'test/Maps/left.map'</code>
<code>up</code>	Value: <code>'test/Maps/up.map'</code>
<code>down</code>	Value: <code>'test/Maps/down.map'</code>
<code>no_move</code>	Value: <code>'test/Maps/no_move.map'</code>
<code>bad_mid_row</code>	Value: <code>'test/Maps/bad_mid_row.map'</code>
<code>multiple_goals</code>	Value: <code>'test/Maps/multiple_goals.map'</code>
<code>no_goals</code>	Value: <code>'test/Maps/no_goals.map'</code>
<code>top_row_bad</code>	Value: <code>'test/Maps/top_row_bad.map'</code>
<code>bottom_row_bad</code>	Value: <code>'test/Maps/bottom_row_bad.map'</code>
<code>quick_solve</code>	Value: <code>'test/Maps/solve.map'</code>
<code>simple_solve</code>	Value: <code>'test/Maps/simple.map'</code>
<code>easy_solve</code>	Value: <code>'test/Maps/easy.map'</code>
<code>left_goal</code>	Value: <code>'test/Maps/left_goal.map'</code>
<code>__package__</code>	Value: <code>None</code>

Index

- UnBlockMeSolver (*package*), 2
 - UnBlockMeSolver.Map (*package*), 3
 - UnBlockMeSolver.Map.Map (*module*), 4–8
 - UnBlockMeSolver.Map.MapReader (*module*), 9–10
 - UnBlockMeSolver.Map.Move (*module*), 11–13
 - UnBlockMeSolver.Map.Piece (*module*), 14–15
 - UnBlockMeSolver.PathFinder (*package*), 16
 - UnBlockMeSolver.PathFinder.AStar (*module*), 17–18
 - UnBlockMeSolver.PathFinder.AStarNode (*module*), 19–20
 - UnBlockMeSolver.PathFinder.bfs (*module*), 27–28
 - UnBlockMeSolver.PathFinder.dfs (*module*), 29–30
 - UnBlockMeSolver.PathFinder.Heuristics (*module*), 21
 - UnBlockMeSolver.PathFinder.Node (*module*), 22–23
 - UnBlockMeSolver.PathFinder.PathFinder (*module*), 24–25
 - UnBlockMeSolver.PathFinder.Solver (*module*), 26
 - UnBlockMeSolver.test (*package*), 31
 - UnBlockMeSolver.test.files (*module*), 52
 - UnBlockMeSolver.test.MapTests (*package*), 32
 - UnBlockMeSolver.test.PathFinding (*package*), 43