Maze Solver

1.0

Generated by Doxygen 1.15.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

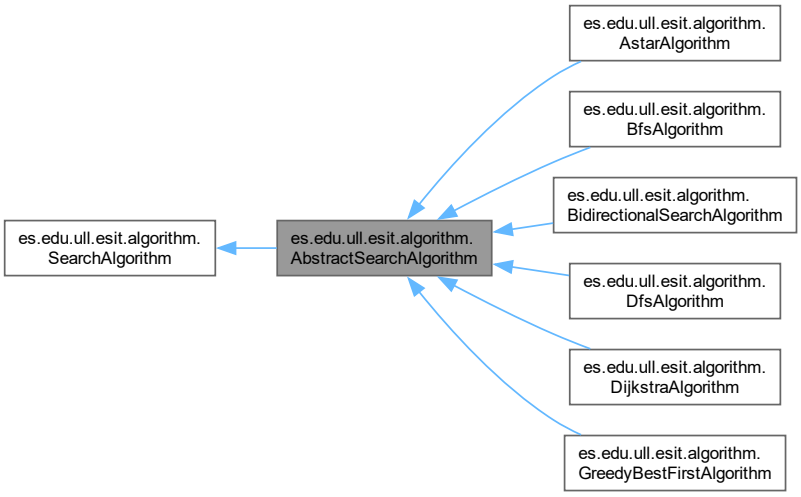
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

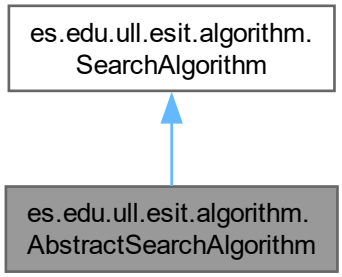
Class Documentation

3.1 es.edu.ull.esit.algorithm.AbstractSearchAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.AbstractSearchAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.AbstractSearchAlgorithm:



Protected Member Functions

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

Additional Inherited Members

Public Member Functions inherited from es.edu.ull.esit.algorithm.SearchAlgorithm

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

3.1.1 Detailed Description

Abstract base class for pathfinding algorithms. Provides common helper methods used by various search algorithms.

3.1.2 Member Function Documentation

3.1.2.1 getLeastHeuristic()

```
Node es.edu.ull.esit.algorithm.AbstractSearchAlgorithm.getLeastHeuristic (
            List< Node > nodes,
            Node end,
            Node start)  [protected]
```

Selects the node with the lowest heuristic cost from a list. Calculates both the heuristic distance to the end and distance from the start.

Parameters

nodes	The list of nodes to evaluate
end	The target/end node
start	The starting node

Returns

The node with the lowest total heuristic cost

3.1.2.2 shortpath()

```
void es.edu.ull.esit.algorithm.AbstractSearchAlgorithm.shortpath (
            Node prev[][],
            Node end,
            int searchTime)  [protected]
```

Reconstructs and displays the shortest path from start to end. Backtracks from the end node using the previous node array.

Parameters

prev	2D array storing the previous node for each position
end	The target/end node
searchTime	The delay time in milliseconds for visualization

The documentation for this class was generated from the following file:

- src/main/java/es/edu/ull/esit/algorithm/AbstractSearchAlgorithm.java

3.2 es.edu.ull.esit.Algorithm Class Reference

Public Member Functions

- void setStrategy (SearchAlgorithm strategy)
- void performSearch (Node start, Node end, int graphWidth, int graphHeight)
- int getSearchTime ()
- void setSearchTime (int searchtime)

3.2.1 Detailed Description

Context class for pathfinding algorithms using the Strategy pattern. Supports DFS, BFS, A∗, Dijkstra, Greedy Best-First Search, and Bidirectional Search. Each algorithm is implemented as a separate strategy class.

3.2.2 Member Function Documentation

3.2.2.1 getSearchTime()

```
int es.edu.ull.esit.Algorithm.getSearchTime ()
```

Gets the current search time delay in milliseconds.

Returns

The search time in milliseconds

3.2.2.2 performSearch()

```
void es.edu.ull.esit.Algorithm.performSearch (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight)
```

Performs the search using the currently set strategy.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid

3.2.2.3 setSearchTime()

```
void es.edu.ull.esit.Algorithm.setSearchTime (
            int searchtime)
```

Sets the search time delay for visualization. Controls the speed of the search animation.

Parameters

searchtime	The search time in milliseconds

3.2.2.4 setStrategy()

```
void es.edu.ull.esit.Algorithm.setStrategy (
            SearchAlgorithm strategy)
```

Sets the search algorithm strategy.

Parameters

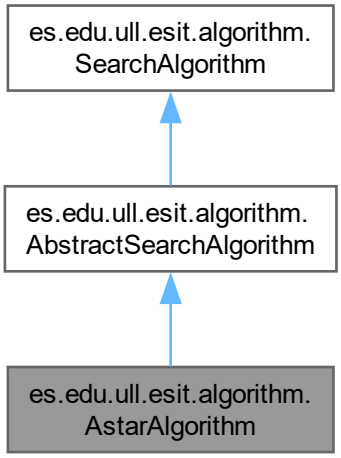
strategy	The search algorithm to use

The documentation for this class was generated from the following file:

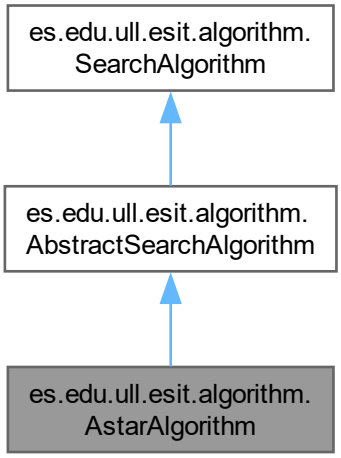
- src/main/java/es/edu/ull/esit/Algorithm.java

3.3 es.edu.ull.esit.algorithm.AstarAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.AstarAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.AstarAlgorithm:



Public Member Functions

- void search (Node start, Node targetNode, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.3.1 Detailed Description

A∗ pathfinding algorithm. Combines actual distance from start (g-cost) with estimated distance to end (h-cost).

3.3.2 Member Function Documentation

3.3.2.1 search()

```
void es.edu.ull.esit.algorithm.AstarAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

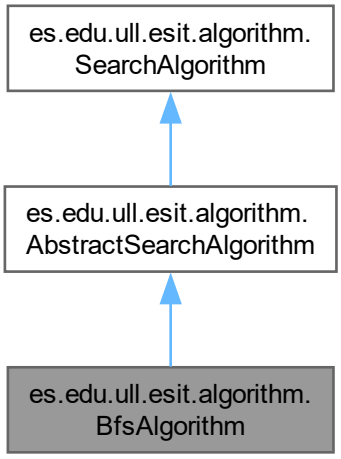
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

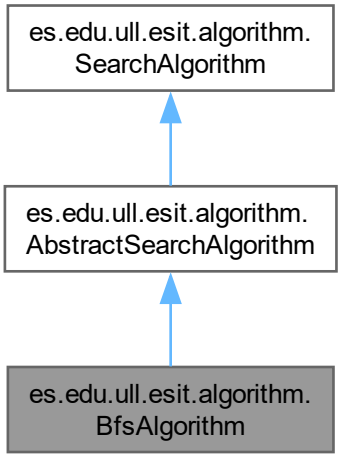
- src/main/java/es/edu/ull/esit/algorithm/AstarAlgorithm.java

3.4 es.edu.ull.esit.algorithm.BfsAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.BfsAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.BfsAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.4.1 Detailed Description

Breadth-First Search (BFS) pathfinding algorithm. Uses a queue to explore all neighbors at the current depth before moving to the next level.

3.4.2 Member Function Documentation

3.4.2.1 search()

```
void es.edu.ull.esit.algorithm.BfsAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

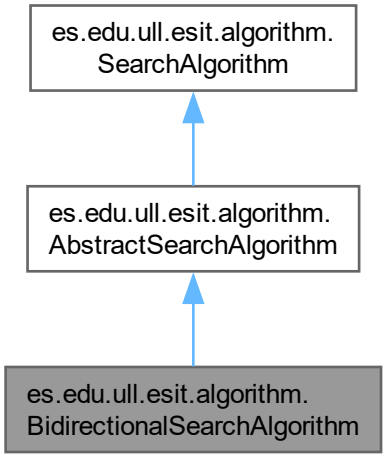
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

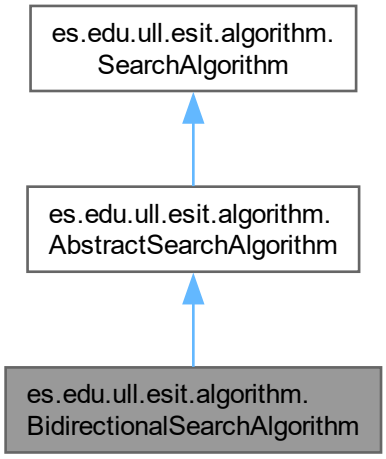
- src/main/java/es/edu/ull/esit/algorithm/BfsAlgorithm.java

3.5 es.edu.ull.esit.algorithm.BidirectionalSearchAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.BidirectionalSearchAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.BidirectionalSearchAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.5.1 Detailed Description

Bidirectional Search pathfinding algorithm. Searches from both start and end simultaneously until the searches meet. More efficient than unidirectional search for finding paths.

3.5.2 Member Function Documentation

3.5.2.1 search()

```
void es.edu.ull.esit.algorithm.BidirectionalSearchAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

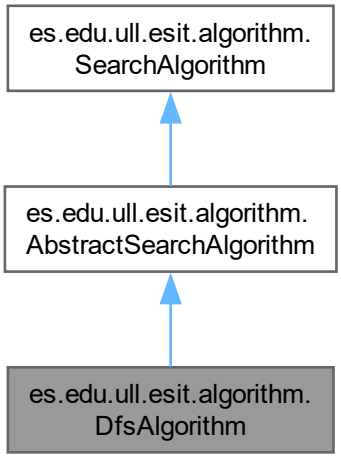
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

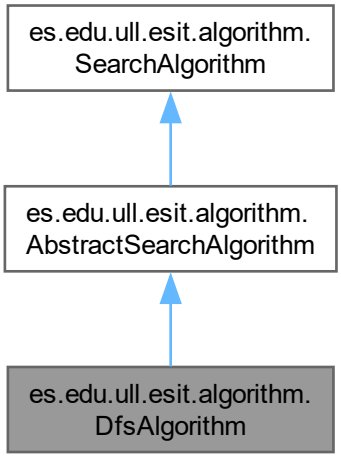
- src/main/java/es/edu/ull/esit/algorithm/BidirectionalSearchAlgorithm.java

3.6 es.edu.ull.esit.algorithm.DfsAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.DfsAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.DfsAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.6.1 Detailed Description

Depth-First Search (DFS) pathfinding algorithm. Uses a stack to explore as far as possible along each branch before backtracking.

3.6.2 Member Function Documentation

3.6.2.1 search()

```
void es.edu.ull.esit.algorithm.DfsAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

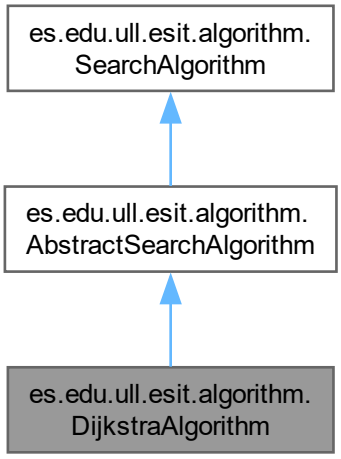
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

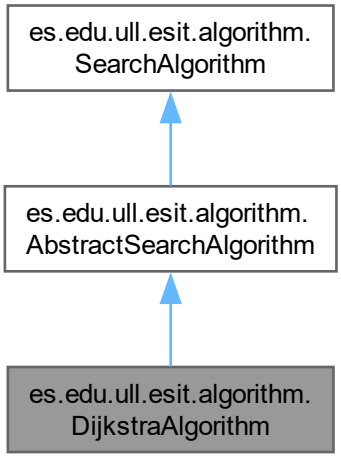
- src/main/java/es/edu/ull/esit/algorithm/DfsAlgorithm.java

3.7 es.edu.ull.esit.algorithm.DijkstraAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.DijkstraAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.DijkstraAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.7.1 Detailed Description

Dijkstra's pathfinding algorithm. Guarantees the shortest path by exploring nodes in order of their distance from start.

3.7.2 Member Function Documentation

3.7.2.1 search()

```
void es.edu.ull.esit.algorithm.DijkstraAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

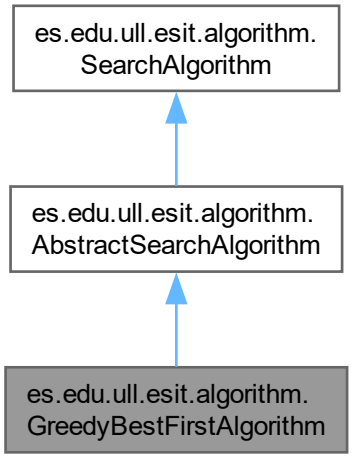
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

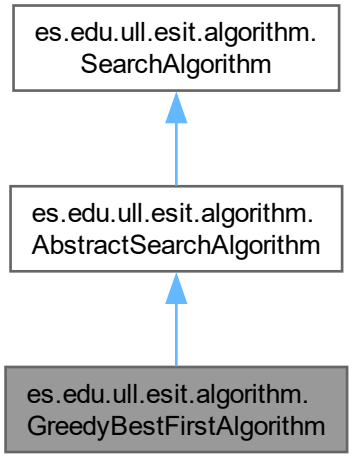
- src/main/java/es/edu/ull/esit/algorithm/DijkstraAlgorithm.java

3.8 es.edu.ull.esit.algorithm.GreedyBestFirstAlgorithm Class Reference

Inheritance diagram for es.edu.ull.esit.algorithm.GreedyBestFirstAlgorithm:



Collaboration diagram for es.edu.ull.esit.algorithm.GreedyBestFirstAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

Additional Inherited Members

Protected Member Functions inherited from es.edu.ull.esit.algorithm.AbstractSearchAlgorithm

- void shortpath (Node[ ][ ] prev, Node end, int searchTime)
- Node getLeastHeuristic (List< Node > nodes, Node end, Node start)

3.8.1 Detailed Description

Greedy Best-First Search pathfinding algorithm. Prioritizes nodes that appear to be closer to the goal based on heuristic. May not find the optimal path but is fast.

3.8.2 Member Function Documentation

3.8.2.1 search()

```
void es.edu.ull.esit.algorithm.GreedyBestFirstAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

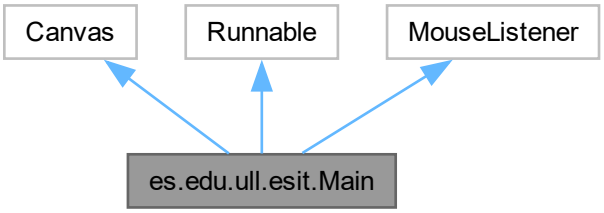
Implements es.edu.ull.esit.algorithm.SearchAlgorithm.

The documentation for this class was generated from the following file:

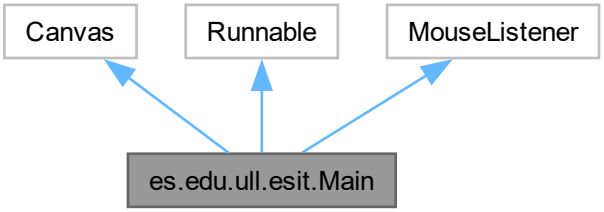
- src/main/java/es/edu/ull/esit/algorithm/GreedyBestFirstAlgorithm.java

3.9 es.edu.ull.esit.Main Class Reference

Inheritance diagram for es.edu.ull.esit.Main:



Collaboration diagram for es.edu.ull.esit.Main:



Public Member Functions

- void run ()
- void init ()
- void setMazeDirections ()
- void createNodes (boolean ref)
- void saveMaze () throws IOException
- void openMaze () throws IOException
- void clearSearchResults ()
- void resetCosts ()
- void render (Graphics2D g)
- void start ()
- void mousePressed (MouseEvent e)
- boolean isMazeValid ()
- Node getNodeAt (int x, int y)
- void mouseClicked (MouseEvent arg0)
- void mouseEntered (MouseEvent arg0)
- void mouseExited (MouseEvent arg0)
- void mouseReleased (MouseEvent arg0)

Static Public Member Functions

- static void main (String[ ] args)
- static void SetupMenu (JFrame frame)

3.9.1 Detailed Description

Main application class for the Maze Solver. Provides a graphical user interface for creating mazes, loading/saving mazes, and visualizing various pathfinding algorithms.

3.9.2 Member Function Documentation

3.9.2.1 clearSearchResults()

```
void es.edu.ull.esit.Main.clearSearchResults ()
```

Clears all search results from the maze. Removes search visualization while preserving walls, start, and end points.

3.9.2.2 createNodes()

```
void es.edu.ull.esit.Main.createNodes (
            boolean ref)
```

Creates or resets the node grid.

Parameters

ref	If true, only clears existing nodes; if false, creates new nodes

3.9.2.3 getNodeAt()

```
Node es.edu.ull.esit.Main.getNodeAt (
            int x,
            int y)
```

Gets the node at the specified pixel coordinates.

Parameters

x	The x-coordinate in pixels
y	The y-coordinate in pixels

Returns

The node at the coordinates, or null if out of bounds

3.9.2.4 init()

```
void es.edu.ull.esit.Main.init ()
```

Initializes the application components. Creates the node grid, sets up mouse listeners, and configures maze directions.

3.9.2.5 isMazeValid()

```
boolean es.edu.ull.esit.Main.isMazeValid ()
```

Validates that the maze has both a start and end point.

Returns

true if the maze has start and target nodes, false otherwise

3.9.2.6 main()

```
void es.edu.ull.esit.Main.main (
            String[] args)  [static]
```

Main entry point for the application. Initializes the GUI window, menu, and maze grid.

Parameters

args	Command line arguments (not used)

3.9.2.7 mouseClicked()

```
void es.edu.ull.esit.Main.mouseClicked (
            MouseEvent arg0)
```

Mouse clicked event handler (not used).

Parameters

arg0	The mouse event

3.9.2.8 mouseEntered()

```
void es.edu.ull.esit.Main.mouseEntered (
            MouseEvent arg0)
```

Mouse entered event handler (not used).

Parameters

arg0	The mouse event

3.9.2.9 mouseExited()

```
void es.edu.ull.esit.Main.mouseExited (
            MouseEvent arg0)
```

Mouse exited event handler (not used).

Parameters

arg0	The mouse event

3.9.2.10 mousePressed()

```
void es.edu.ull.esit.Main.mousePressed (
            MouseEvent e)
```

Handles mouse press events on the canvas. Allows users to create walls, set start/end points by clicking nodes.

Parameters

e	The mouse event

3.9.2.11 mouseReleased()

```
void es.edu.ull.esit.Main.mouseReleased (
            MouseEvent arg0)
```

Mouse released event handler (not used).

Parameters

arg0	The mouse event

3.9.2.12 openMaze()

```
void es.edu.ull.esit.Main.openMaze () throws IOException
```

Loads a maze configuration from a file. Reads the custom maze format and reconstructs the node grid.

Exceptions

IOException	If an I/O error occurs during file reading

3.9.2.13 render()

```
void es.edu.ull.esit.Main.render (
            Graphics2D g)
```

Renders all nodes on the graphics context.

Parameters

g	The Graphics2D context to render on

3.9.2.14 resetCosts()

```
void es.edu.ull.esit.Main.resetCosts ()
```

Resets the g-cost values for all nodes to maximum. Prepares the grid for a new pathfinding algorithm run.

3.9.2.15 run()

```
void es.edu.ull.esit.Main.run ()
```

Main render loop for the application. Continuously updates the display using double buffering.

3.9.2.16 saveMaze()

```
void es.edu.ull.esit.Main.saveMaze () throws IOException
```

Saves the current maze configuration to a file. The maze is saved in a custom format with:

- 0: Empty path
- 1: Wall
- 2: Start point
- 3: End point

Exceptions

IOException	If an I/O error occurs during file writing

3.9.2.17 setMazeDirections()

```
void es.edu.ull.esit.Main.setMazeDirections ()
```

Sets up the directional connections between adjacent nodes. Establishes left, right, up, and down neighbors for each node in the grid.

3.9.2.18 SetupMenu()

```
void es.edu.ull.esit.Main.SetupMenu (
            JFrame frame)  [static]
```

Sets up the menu bar with File, Board, and Algorithms menus. Configures all menu items and their action listeners.

Parameters

frame	The main application frame

3.9.2.19 start()

```
void es.edu.ull.esit.Main.start ()
```

Starts the rendering thread.

The documentation for this class was generated from the following file:

- src/main/java/es/edu/ull/esit/Main.java

3.10 es.edu.ull.esit.MazeGenerator Class Reference

Public Member Functions

- MazeGenerator (int width, int height, Node[ ][ ] grid)
- void generate ()

3.10.1 Detailed Description

Generates random mazes using a depth-first search algorithm. The maze generator creates complex paths by carving through a grid of walls.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 MazeGenerator()

```
es.edu.ull.esit.MazeGenerator.MazeGenerator (
            int width,
            int height,
            Node grid[][])
```

Constructs a new MazeGenerator with the specified dimensions.

Parameters

width	The width of the maze grid
height	The height of the maze grid
grid	The 2D array of nodes representing the grid

3.10.3 Member Function Documentation

3.10.3.1 generate()

```
void es.edu.ull.esit.MazeGenerator.generate ()
```

Generates a random maze using a depth-first search algorithm. The algorithm starts from a random cell and carves paths by removing walls between cells, creating a perfect maze with exactly one path between any two points.

The documentation for this class was generated from the following file:

- src/main/java/es/edu/ull/esit/MazeGenerator.java

3.11 es.edu.ull.esit.Node Class Reference

Public Member Functions

- Node (int x, int y)
- Node ()
- double getgCost ()
- void setgCost (double g)
- void render (Graphics2D g)
- void Clicked (int buttonCode)
- void setAsWall ()
- double getFCost ()
- void setFCost (double fcost)
- void setColor (Color c)
- Color getColor ()
- List< Node > getNeighbours ()
- void setDirections (Node l, Node r, Node u, Node d)
- void clearNode ()
- int getX ()
- int getY ()
- Node setX (int x)
- Node setY (int y)
- boolean isWall ()
- boolean isStart ()
- boolean isEnd ()
- boolean isPath ()
- boolean isSearched ()

Static Public Member Functions

- static double distance (Node a, Node b)

3.11.1 Detailed Description

Represents a node in the maze grid. Each node can be a wall, path, start point, end point, or searched node. Nodes maintain references to their neighbors and cost values for pathfinding algorithms.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 Node() [1/2]

```
es.edu.ull.esit.Node.Node (
            int x,
            int y)
```

Constructs a new Node with specified grid coordinates.

Parameters

x	The x-coordinate in the grid
y	The y-coordinate in the grid

3.11.2.2 Node() [2/2]

```
es.edu.ull.esit.Node.Node ()
```

Default constructor for Node.

3.11.3 Member Function Documentation

3.11.3.1 clearNode()

```
void es.edu.ull.esit.Node.clearNode ()
```

Clears the node to its default state (light gray path).

3.11.3.2 Clicked()

```
void es.edu.ull.esit.Node.Clicked (
            int buttonCode)
```

Handles mouse click events on the node. Different mouse buttons set different node types:

- Button 1: Wall (black)
- Button 2: Start point (green)
- Button 3: End point (red)
- Button 4: Clear node

Parameters

buttonCode	The mouse button code (1-4)

3.11.3.3 distance()

```
double es.edu.ull.esit.Node.distance (
            Node a,
            Node b)  [static]
```

Calculates the Euclidean distance between two nodes.

Parameters

a	The first node
b	The second node

Returns

The Euclidean distance between the nodes

3.11.3.4 getColor()

```
Color es.edu.ull.esit.Node.getColor ()
```

Gets the current color of this node.

Returns

The current node color

3.11.3.5 getFCost()

```
double es.edu.ull.esit.Node.getFCost ()
```

Gets the f-cost (total estimated cost) for pathfinding algorithms.

Returns

The f-cost value

3.11.3.6 getgCost()

```
double es.edu.ull.esit.Node.getgCost ()
```

Gets the g-cost (distance from start node) for pathfinding algorithms.

Returns

The g-cost value

3.11.3.7 getNeighbours()

```
List< Node > es.edu.ull.esit.Node.getNeighbours ()
```

Gets the list of neighboring nodes that are accessible paths. Only includes neighbors that are not walls.

Returns

List of accessible neighboring nodes

3.11.3.8 getX()

```
int es.edu.ull.esit.Node.getX ()
```

Gets the grid x-coordinate of this node.

Returns

The x-coordinate in the grid

3.11.3.9 getY()

```
int es.edu.ull.esit.Node.getY ()
```

Gets the grid y-coordinate of this node.

Returns

The y-coordinate in the grid

3.11.3.10 isEnd()

```
boolean es.edu.ull.esit.Node.isEnd ()
```

Checks if this node is the end/target point.

Returns

true if the node is the end point (red), false otherwise

3.11.3.11 isPath()

```
boolean es.edu.ull.esit.Node.isPath ()
```

Checks if this node is a traversable path.

Returns

true if the node is a path or the end point, false otherwise

3.11.3.12 isSearched()

```
boolean es.edu.ull.esit.Node.isSearched ()
```

Checks if this node has been searched by an algorithm.

Returns

true if the node has been visited during search, false otherwise

3.11.3.13 isStart()

```
boolean es.edu.ull.esit.Node.isStart ()
```

Checks if this node is the start point.

Returns

true if the node is the start point (green), false otherwise

3.11.3.14 isWall()

```
boolean es.edu.ull.esit.Node.isWall ()
```

Checks if this node is a wall.

Returns

true if the node is a wall (black), false otherwise

3.11.3.15 render()

```
void es.edu.ull.esit.Node.render (
            Graphics2D g)
```

Renders the node on the graphics context.

Parameters

g	The Graphics2D context to render on

3.11.3.16 setAsWall()

```
void es.edu.ull.esit.Node.setAsWall ()
```

Sets this node as a wall (black color).

3.11.3.17 setColor()

```
void es.edu.ull.esit.Node.setColor (
            Color c)
```

Sets the color of this node.

Parameters

c	The color to set

3.11.3.18 setDirections()

```
void es.edu.ull.esit.Node.setDirections (
            Node l,
            Node r,
            Node u,
            Node d)
```

Sets the directional neighbors for this node.

Parameters

l	The left neighbor node
r	The right neighbor node
u	The up neighbor node
d	The down neighbor node

3.11.3.19 setFCost()

```
void es.edu.ull.esit.Node.setFCost (
            double fcost)
```

Sets the f-cost (total estimated cost) for pathfinding algorithms.

Parameters

fcost	The f-cost value to set

3.11.3.20 setgCost()

```
void es.edu.ull.esit.Node.setgCost (
            double g)
```

Sets the g-cost (distance from start node) for pathfinding algorithms.

Parameters

g	The g-cost value to set

3.11.3.21 setX()

```
Node es.edu.ull.esit.Node.setX (
            int x)
```

Sets the pixel x-position of this node.

Parameters

x	The pixel x-position

Returns

This node for method chaining

3.11.3.22 setY()

```
Node es.edu.ull.esit.Node.setY (
            int y)
```

Sets the pixel y-position of this node.

Parameters

y	The pixel y-position

Returns

This node for method chaining

The documentation for this class was generated from the following file:

- src/main/java/es/edu/ull/esit/Node.java

3.12 es.edu.ull.esit.NodeTest Class Reference

Public Member Functions

- void testConstructorAndPositions ()
- void testSetXandY ()
- void testgCost ()
- void testFCost ()
- void testDistance ()
- void testClickedChangesColor ()
- void testSetAsWall ()
- void testClearNode ()
- void testColorFlags ()
- void testGetNeighbours ()

3.12.1 Detailed Description

Test unitarios para la clase Node.

Valida el funcionamiento de:

- Constructores y manejo de posiciones
- Costos (gCost y fCost)
- Cambio de color mediante interacciones
- Métodos booleanos de estado
- Obtención de nodos vecinos válidos
- Operaciones básicas como setX, setY, clearNode, etc.

3.12.2 Member Function Documentation

3.12.2.1 testClearNode()

```
void es.edu.ull.esit.NodeTest.testClearNode ()
```

Verifica que clearNode() restablezca el color LIGHT_GRAY.

3.12.2.2 testClickedChangesColor()

```
void es.edu.ull.esit.NodeTest.testClickedChangesColor ()
```

Prueba que el método Clicked() cambie el color del nodo dependiendo del código de botón recibido.

3.12.2.3 testColorFlags()

```
void es.edu.ull.esit.NodeTest.testColorFlags ()
```

Valida el funcionamiento de los métodos booleanos que identifican el estado del nodo según su color.

3.12.2.4 testConstructorAndPositions()

```
void es.edu.ull.esit.NodeTest.testConstructorAndPositions ()
```

Prueba que el constructor con parámetros asigna correctamente las posiciones X e Y y que los métodos getX() y getY() realizan la conversión adecuada.

3.12.2.5 testDistance()

```
void es.edu.ull.esit.NodeTest.testDistance ()
```

Verifica que el método estático distance() calcule correctamente la distancia euclidiana entre dos nodos.

3.12.2.6 testFCost()

```
void es.edu.ull.esit.NodeTest.testFCost ()
```

Comprueba que getFCost() y setFCost() funcionen correctamente.

3.12.2.7 testgCost()

```
void es.edu.ull.esit.NodeTest.testgCost ()
```

Comprueba que getgCost() y setgCost() funcionen correctamente.

3.12.2.8 testGetNeighbours()

```
void es.edu.ull.esit.NodeTest.testGetNeighbours ()
```

Prueba que getNeighbours() solo devuelva los nodos configurados como caminos válidos (isPath() == true).

3.12.2.9 testSetAsWall()

```
void es.edu.ull.esit.NodeTest.testSetAsWall ()
```

Verifica que setAsWall() establezca correctamente el color BLACK y que isWall() lo detecte.

3.12.2.10 testSetXandY()

```
void es.edu.ull.esit.NodeTest.testSetXandY ()
```

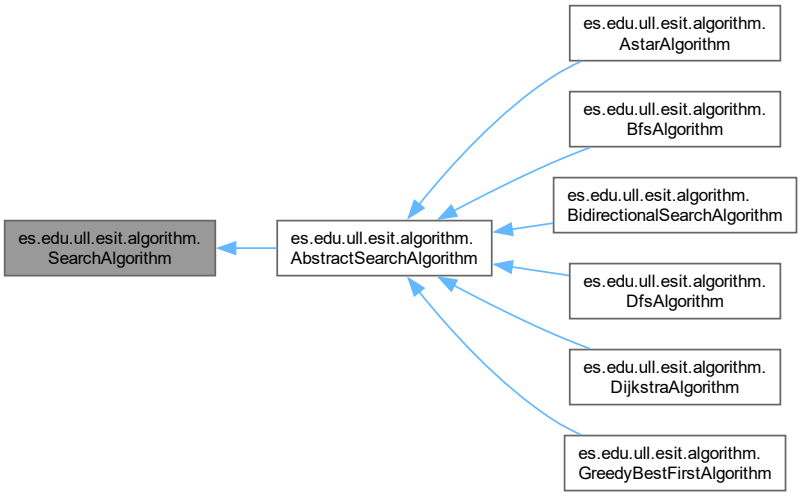
Verifica que los métodos setX() y setY() actualicen correctamente las coordenadas internas del nodo.

The documentation for this class was generated from the following file:

- src/test/java/es/edu/ull/esit/NodeTest.java

3.13 es.edu.ull.esit.algorithm.SearchAlgorithm Interface Reference

Inheritance diagram for es.edu.ull.esit.algorithm.SearchAlgorithm:



Public Member Functions

- void search (Node start, Node end, int graphWidth, int graphHeight, int searchTime)

3.13.1 Detailed Description

Strategy interface for pathfinding algorithms. Implementations of this interface provide different search strategies for finding paths in a maze grid.

3.13.2 Member Function Documentation

3.13.2.1 search()

```
void es.edu.ull.esit.algorithm.SearchAlgorithm.search (
            Node start,
            Node end,
            int graphWidth,
            int graphHeight,
            int searchTime)
```

Performs the search algorithm to find a path from start to end.

Parameters

start	The starting node
end	The target/end node
graphWidth	The width of the grid
graphHeight	The height of the grid
searchTime	The delay time in milliseconds for visualization

Implemented in es.edu.ull.esit.algorithm.AstarAlgorithm, es.edu.ull.esit.algorithm.BfsAlgorithm, es.edu.ull.esit.algorithm.BidirectionalS
es.edu.ull.esit.algorithm.DfsAlgorithm, es.edu.ull.esit.algorithm.DijkstraAlgorithm, and es.edu.ull.esit.algorithm.GreedyBestFirstAlgorit

The documentation for this interface was generated from the following file:

- src/main/java/es/edu/ull/esit/algorithm/SearchAlgorithm.java

Index