The Game Of Life

Generated by Doxygen 1.12.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

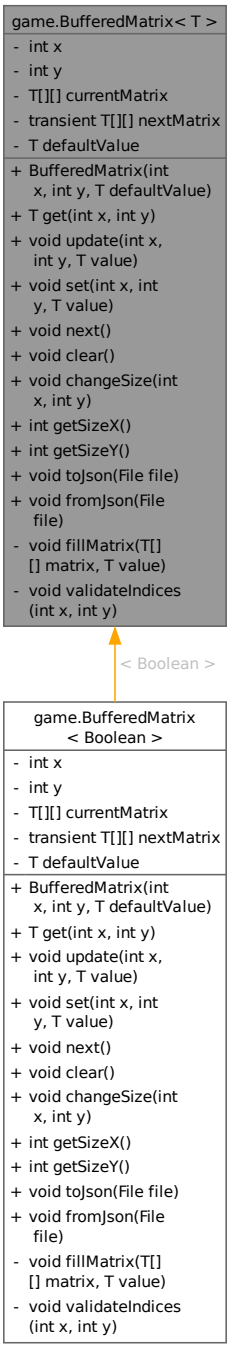
Chapter 3

Data Structure Documentation

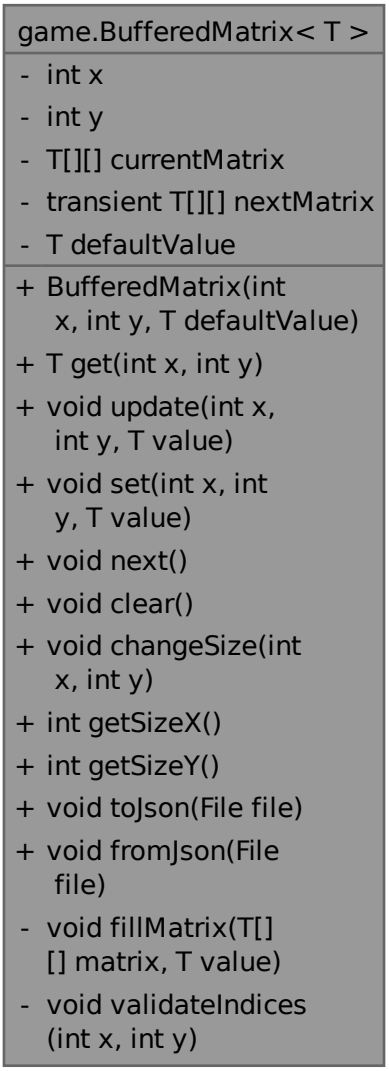
3.1 game.BufferedMatrix< T > Class Template Reference

A generic class representing a two-dimensional buffered matrix that supports operations such as getting and setting values, updating the current matrix, and saving/loading the matrix to/from JSON files.

Inheritance diagram for game.BufferedMatrix< T >:



Collaboration diagram for game.BufferedMatrix< T >:



Public Member Functions

- BufferedMatrix (int x, int y, T defaultValue)

  Constructs a new BufferedMatrix with the specified dimensions and default value.
- T get (int x, int y)

  Retrieves the value at the specified position in the current matrix.
- void update (int x, int y, T value)

  Updates the value at the specified position in the current matrix.
- void set (int x, int y, T value)

  Sets a value at the specified position in the next matrix.
- void **next** ()

  Advances the current matrix to the next state and clears the next matrix.

- void **clear** ()

  Clears both the current and next matrices, resetting them to the default value.
- void changeSize (int x, int y)

  Changes the size of the matrix to the specified dimensions, clearing the current and next matrices.
- int getSizeX ()

  Returns the current number of rows in the matrix.
- int getSizeY ()

  Returns the current number of columns in the matrix.
- void toJson (File file)

  Serializes the current state of the matrix to a JSON file.
- void fromJson (File file)

  Deserializes the matrix state from a JSON file, updating the current instance.

Private Member Functions

- void fillMatrix (T[ ][ ] matrix, T value)

  Fills the specified matrix with the given value.
- void validateIndices (int x, int y)

  Validates the specified indices to ensure they are within the matrix bounds.

Private Attributes

- int **x**

  The number of rows in the matrix.
- int **y**

  The number of columns in the matrix.
- T[ ][ ] **currentMatrix**

  The current state of the matrix.
- transient T[ ][ ] **nextMatrix**

  The next state of the matrix, used for double buffering.
- T **defaultValue**

  The default value used to fill the matrix.

3.1.1 Detailed Description

A generic class representing a two-dimensional buffered matrix that supports operations such as getting and setting values, updating the current matrix, and saving/loading the matrix to/from JSON files.

Parameters

<T>	the type of elements in the matrix

3.1.2 Constructor & Destructor Documentation

3.1.2.1 BufferedMatrix()

```
game.BufferedMatrix< T >.BufferedMatrix (
            int x,
            int y,
            T defaultValue)
```

Constructs a new BufferedMatrix with the specified dimensions and default value.

Parameters

x	the number of rows in the matrix
y	the number of columns in the matrix
defaultValue	the default value to fill the matrix

3.1.3 Member Function Documentation

3.1.3.1 changeSize()

```
void game.BufferedMatrix< T >.changeSize (
            int x,
            int y)
```

Changes the size of the matrix to the specified dimensions, clearing the current and next matrices.

Parameters

x	the new number of rows
y	the new number of columns

Exceptions

IndexOutOfBoundsException	if the new size is less than 1

3.1.3.2 fillMatrix()

```
void game.BufferedMatrix< T >.fillMatrix (
            T matrix[][],
            T value)  [private]
```

Fills the specified matrix with the given value.

Parameters

matrix	the matrix to fill
value	the value to fill the matrix with

3.1.3.3 fromJson()

```
void game.BufferedMatrix< T >.fromJson (
            File file)
```

Deserializes the matrix state from a JSON file, updating the current instance.

Parameters

file	the file from which to load the matrix

3.1.3.4 get()

```
T game.BufferedMatrix< T >.get (
            int x,
            int y)
```

Retrieves the value at the specified position in the current matrix.

Parameters

x	the row index
y	the column index

Returns

the value at the specified position

Exceptions

IndexOutOfBoundsException	if the indices are out of bounds

3.1.3.5 getSizeX()

```
int game.BufferedMatrix< T >.getSizeX ()
```

Returns the current number of rows in the matrix.

Returns

the number of rows

3.1.3.6 getSizeY()

```
int game.BufferedMatrix< T >.getSizeY ()
```

Returns the current number of columns in the matrix.

Returns

the number of columns

3.1.3.7 set()

```
void game.BufferedMatrix< T >.set (
            int x,
            int y,
            T value)
```

Sets a value at the specified position in the next matrix.

Parameters

x	the row index
y	the column index
value	the value to set

Exceptions

IndexOutOfBoundsException	if the indices are out of bounds

3.1.3.8 toJson()

```
void game.BufferedMatrix< T >.toJson (
            File file)
```

Serializes the current state of the matrix to a JSON file.

Parameters

file	the file to which the matrix should be saved

3.1.3.9 update()

```
void game.BufferedMatrix< T >.update (
            int x,
            int y,
            T value)
```

Updates the value at the specified position in the current matrix.

Parameters

x	the row index
y	the column index
value	the new value to set

Exceptions

IndexOutOfBoundsException	if the indices are out of bounds

3.1.3.10 validateIndices()

```
void game.BufferedMatrix< T >.validateIndices (
            int x,
            int y)  [private]
```

Validates the specified indices to ensure they are within the matrix bounds.

Parameters

x	the row index
y	the column index

Exceptions

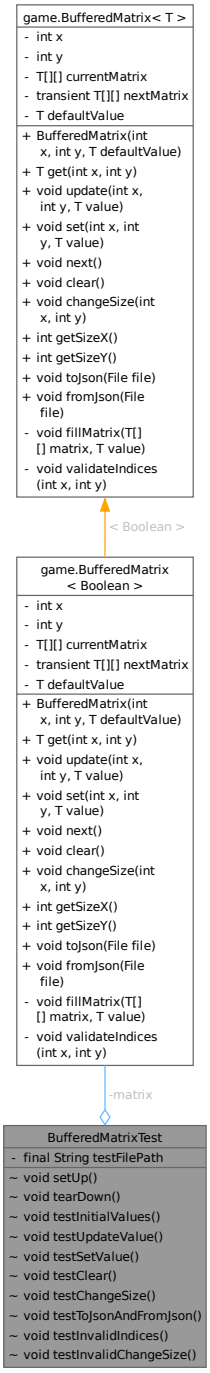
IndexOutOfBoundsException	if the indices are out of bounds

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

- BufferedMatrix.java

3.2 BufferedMatrixTest Class Reference

Collaboration diagram for BufferedMatrixTest:



Package Functions

- void **setUp** ()
- void **tearDown** ()

- void **testInitialValues** ()
- void **testUpdateValue** ()
- void **testSetValue** ()
- void **testClear** ()
- void **testChangeSize** ()
- void **testToJsonAndFromJson** ()
- void **testInvalidIndices** ()
- void **testInvalidChangeSize** ()

Private Attributes

- BufferedMatrix< Boolean > **matrix**
- final String **testFilePath** = "testMatrix.json"

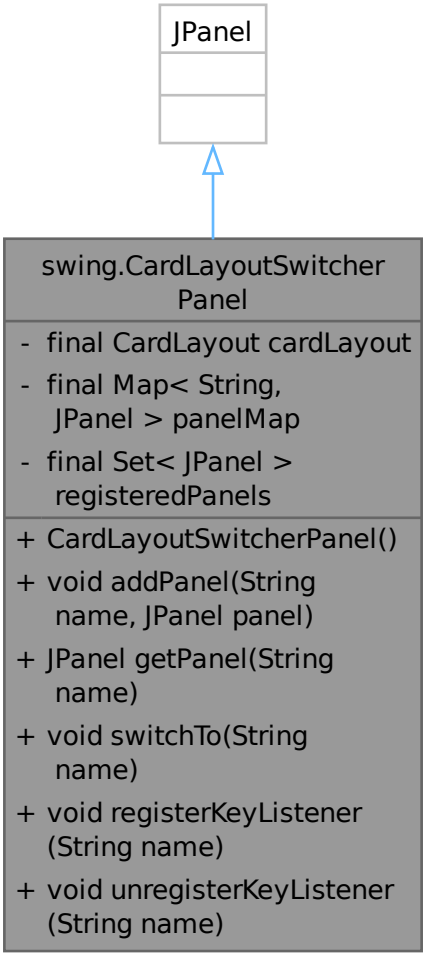
The documentation for this class was generated from the following file:

- BufferedMatrixTest.java

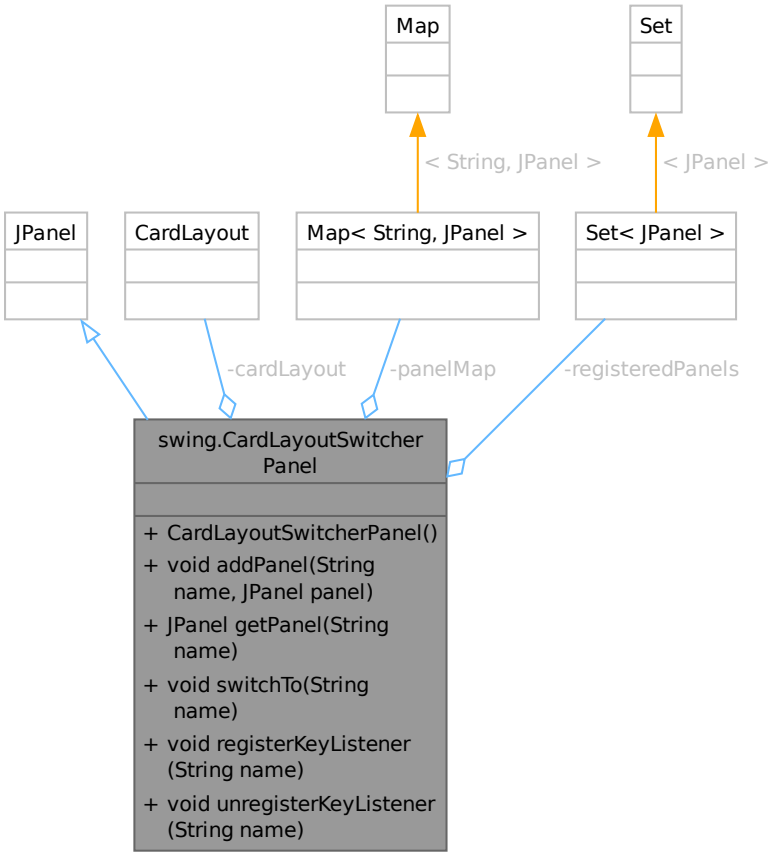
3.3 swing.CardLayoutSwitcherPanel Class Reference

A JPanel that utilizes a CardLayout to switch between multiple panels.

Inheritance diagram for swing.CardLayoutSwitcherPanel:



Collaboration diagram for swing.CardLayoutSwitcherPanel:



Public Member Functions

- CardLayoutSwitcherPanel ()

  Constructs a new CardLayoutSwitcherPanel with an initialized CardLayout.
- void addPanel (String name, JPanel panel)

  Adds a panel to the CardLayoutSwitcherPanel.
- JPanel getPanel (String name)

  Retrieves a panel by its name.
- void switchTo (String name)

  Switches the visible panel to the one associated with the given name.
- void registerKeyListener (String name)

  Registers a JPanel to receive key events when it is not focused.
- void unregisterKeyListener (String name)

  Unregisters a JPanel from receiving key events.

Private Attributes

- final CardLayout **cardLayout**

The CardLayout used to manage the layout of this panel.

- final Map< String, JPanel > **panelMap**

  A map that holds the association between panel names and their corresponding JPanel objects.

- final Set< JPanel > **registeredPanels**

  A set that keeps track of registered panels for key listener management.

3.3.1 Detailed Description

A JPanel that utilizes a CardLayout to switch between multiple panels.

This class allows adding, switching, and managing key listeners for different panels. It captures key events and dispatches them to registered panels that are not currently focused.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 CardLayoutSwitcherPanel()

```
swing.CardLayoutSwitcherPanel.CardLayoutSwitcherPanel ()
```

Constructs a new CardLayoutSwitcherPanel with an initialized CardLayout.

Sets up a KeyEventDispatcher to handle key events for registered panels.

3.3.3 Member Function Documentation

3.3.3.1 addPanel()

```
void swing.CardLayoutSwitcherPanel.addPanel (
            String name,
            JPanel panel)
```

Adds a panel to the CardLayoutSwitcherPanel.

Parameters

name	The name associated with the panel.
panel	The JPanel to be added.

3.3.3.2 getPanel()

```
JPanel swing.CardLayoutSwitcherPanel.getPanel (
            String name)
```

Retrieves a panel by its name.

Parameters

name	The name of the panel to retrieve.

Returns

The JPanel associated with the given name, or null if no such panel exists.

3.3.3.3 registerKeyListener()

```
void swing.CardLayoutSwitcherPanel.registerKeyListener (
            String name)
```

Registers a JPanel to receive key events when it is not focused.

Parameters

name	The name of the panel to register.

3.3.3.4 switchTo()

```
void swing.CardLayoutSwitcherPanel.switchTo (
            String name)
```

Switches the visible panel to the one associated with the given name.

Parameters

name	The name of the panel to switch to.

3.3.3.5 unregisterKeyListener()

```
void swing.CardLayoutSwitcherPanel.unregisterKeyListener (
            String name)
```

Unregisters a JPanel from receiving key events.

Parameters

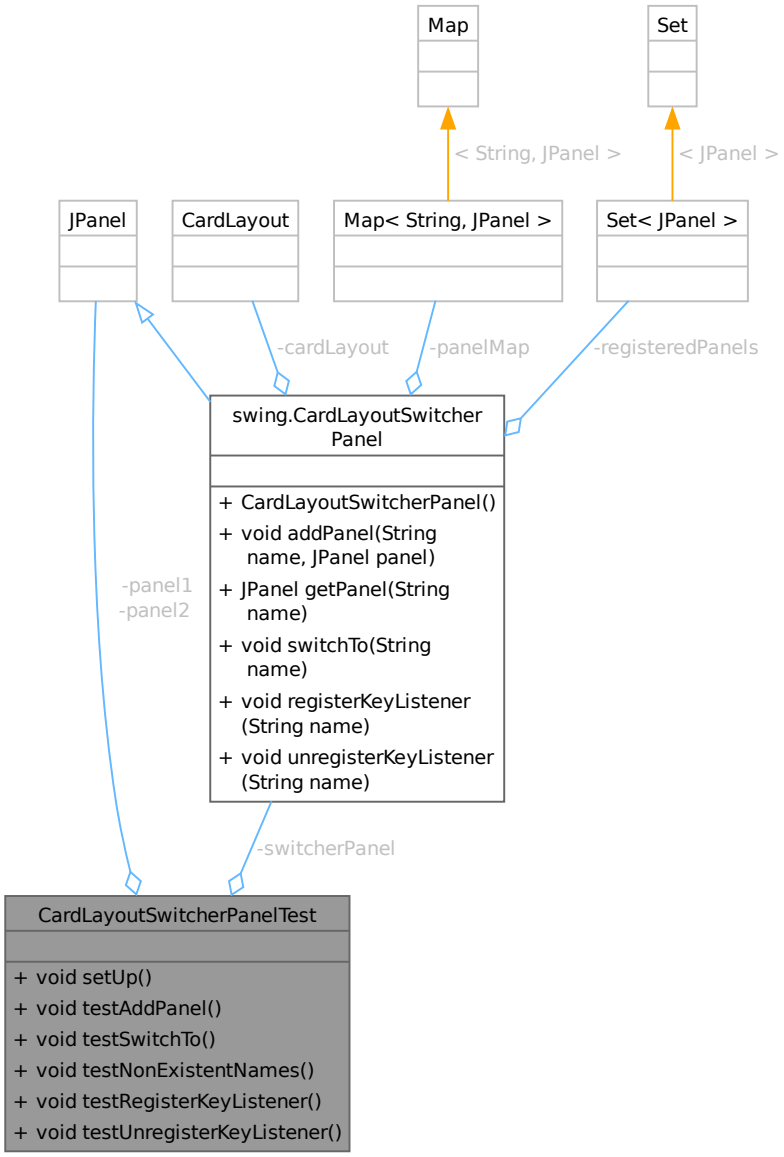
name	The name of the panel to unregister.

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

- CardLayoutSwitcherPanel.java

3.4 CardLayoutSwitcherPanelTest Class Reference

Collaboration diagram for CardLayoutSwitcherPanelTest:



Public Member Functions

- void **setUp** ()
- void **testAddPanel** ()
- void **testSwitchTo** ()
- void **testNonExistentNames** ()
- void **testRegisterKeyListener** ()
- void **testUnregisterKeyListener** ()

Private Attributes

- CardLayoutSwitcherPanel **switcherPanel**
- JPanel **panel1**
- JPanel **panel2**

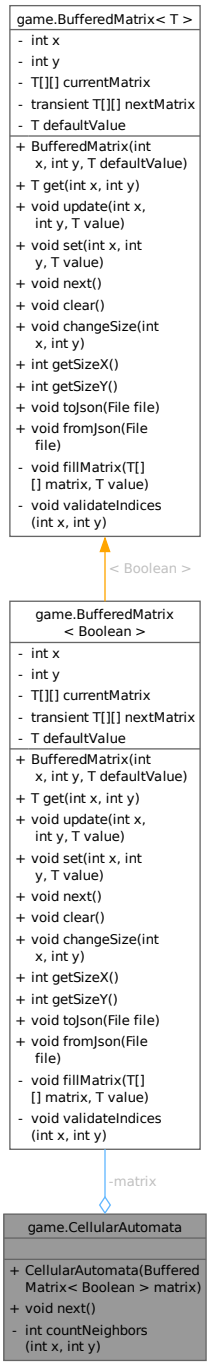
The documentation for this class was generated from the following file:

- CardLayoutSwitcherPanelTest.java

3.5 game.CellularAutomata Class Reference

The `CellularAutomata` class represents a cellular automaton system, where each cell can be either alive or dead.

Collaboration diagram for game.CellularAutomata:



Public Member Functions

- CellularAutomata (BufferedMatrix< Boolean > matrix)

  Constructs a `CellularAutomata` instance with the specified matrix.
- void next ()

  Updates the state of the cellular automaton to the next generation.

Private Member Functions

- int countNeighbors (int x, int y)

  Counts the number of alive neighbors around a specified cell in the matrix.

Private Attributes

- final BufferedMatrix< Boolean > **matrix**

  The matrix representing the current state of the cellular automaton, where each cell is either alive (true) or dead (false).

3.5.1 Detailed Description

The `CellularAutomata` class represents a cellular automaton system, where each cell can be either alive or dead.

This class implements the rules of Conway's Game of Life on a 2D matrix of boolean values.

The state of the cells is updated in discrete time steps, where the next state is determined by the current state and the number of alive neighbors.

The rules for updating the state of each cell are as follows:

- Any live cell with two or three live neighbors survives.
- Any dead cell with exactly three live neighbors becomes a live cell.
- All other live cells die in the next generation. Similarly, all other dead cells remain dead.

See also

BufferedMatrix

3.5.2 Constructor & Destructor Documentation

3.5.2.1 CellularAutomata()

```
game.CellularAutomata.CellularAutomata (
            BufferedMatrix< Boolean > matrix)
```

Constructs a `CellularAutomata` instance with the specified matrix.

Parameters

matrix	the initial state of the cellular automaton, represented as a `BufferedMatrix` of Boolean values.

3.5.3 Member Function Documentation

3.5.3.1 countNeighbors()

```
int game.CellularAutomata.countNeighbors (
            int x,
            int y)  [private]
```

Counts the number of alive neighbors around a specified cell in the matrix.

Parameters

x	the x-coordinate of the cell whose neighbors are to be counted.
y	the y-coordinate of the cell whose neighbors are to be counted.

Returns

the number of alive neighbors surrounding the cell at (x, y). The count includes wrapping around the edges of the matrix.

3.5.3.2 next()

```
void game.CellularAutomata.next ()
```

Updates the state of the cellular automaton to the next generation.

This method applies the rules of the cellular automaton to all cells in the matrix and updates their states accordingly. After updating the state, it invokes the `next` method of the matrix to prepare for the next generation.

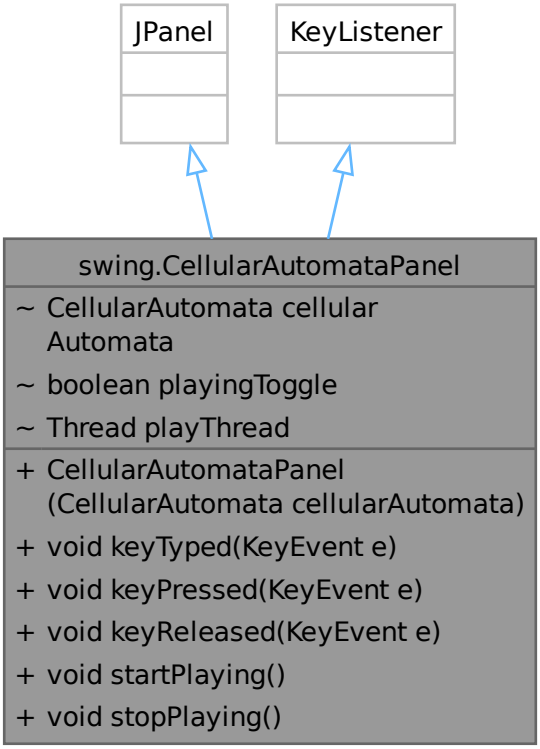
The documentation for this class was generated from the following file:

- CellularAutomata.java

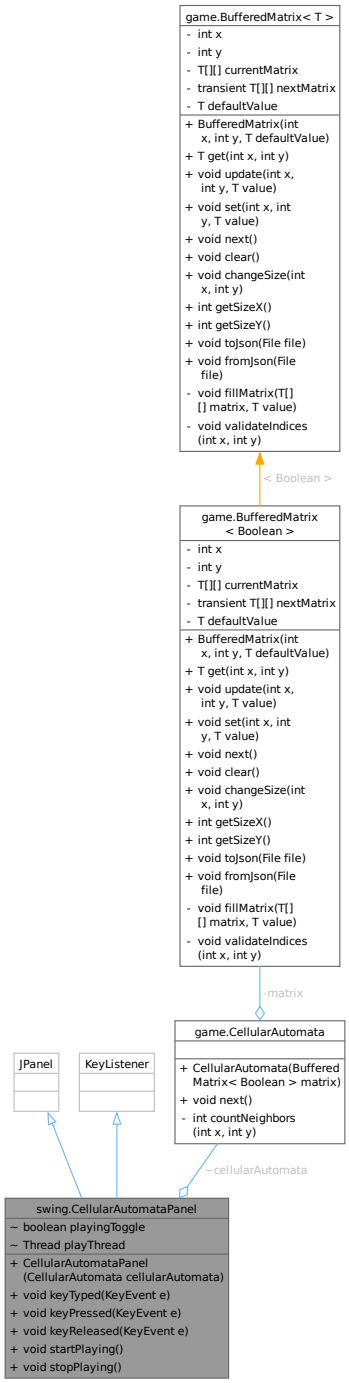
3.6 swing.CellularAutomataPanel Class Reference

A JPanel that visualizes and controls a Cellular Automaton.

Inheritance diagram for swing.CellularAutomataPanel:



Collaboration diagram for swing.CellularAutomataPanel:



Public Member Functions

- CellularAutomataPanel (CellularAutomata cellularAutomata)

  Constructs a CellularAutomataPanel with the specified CellularAutomata.
- void keyTyped (KeyEvent e)

  Invoked when a key has been typed.
- void keyPressed (KeyEvent e)

Invoked when a key has been pressed.

- void keyReleased (KeyEvent e)

  Invoked when a key has been released.

- void **startPlaying** ()

  Starts the simulation in a separate thread, updating the CellularAutomata instance at regular intervals.

- void **stopPlaying** ()

  Stops the simulation and waits for the playing thread to terminate.

Package Attributes

- CellularAutomata **cellularAutomata**

  The instance of the CellularAutomata that this panel visualizes and controls.

- boolean **playingToggle** = false

  A flag indicating whether the simulation is currently playing (true) or paused (false).

- Thread **playThread** = null

  The thread that runs the simulation when playing.

3.6.1 Detailed Description

A JPanel that visualizes and controls a Cellular Automaton.

It allows users to start and stop the simulation, as well as step through the simulation one generation at a time using keyboard input.

This panel listens for key events and responds to specific key presses:

- P: Toggles the play/pause state of the simulation.
- N: Advances the simulation to the next generation.

See also

CellularAutomata

3.6.2 Constructor & Destructor Documentation

3.6.2.1 CellularAutomataPanel()

```
swing.CellularAutomataPanel.CellularAutomataPanel (
            CellularAutomata cellularAutomata)
```

Constructs a CellularAutomataPanel with the specified CellularAutomata.

Parameters

cellularAutomata	the CellularAutomata instance to be visualized and controlled by this panel.

3.6.3 Member Function Documentation

3.6.3.1 keyPressed()

```
void swing.CellularAutomataPanel.keyPressed (
            KeyEvent e)
```

Invoked when a key has been pressed.

Responds to specific key events:

- P: Toggles the play/pause state of the simulation.
- N: Advances the simulation to the next generation.

Parameters

e	the key event to be processed.

3.6.3.2 keyReleased()

```
void swing.CellularAutomataPanel.keyReleased (
            KeyEvent e)
```

Invoked when a key has been released.

This implementation does not perform any action.

Parameters

e	the key event to be processed.

3.6.3.3 keyTyped()

```
void swing.CellularAutomataPanel.keyTyped (
            KeyEvent e)
```

Invoked when a key has been typed.

This implementation does not perform any action.

Parameters

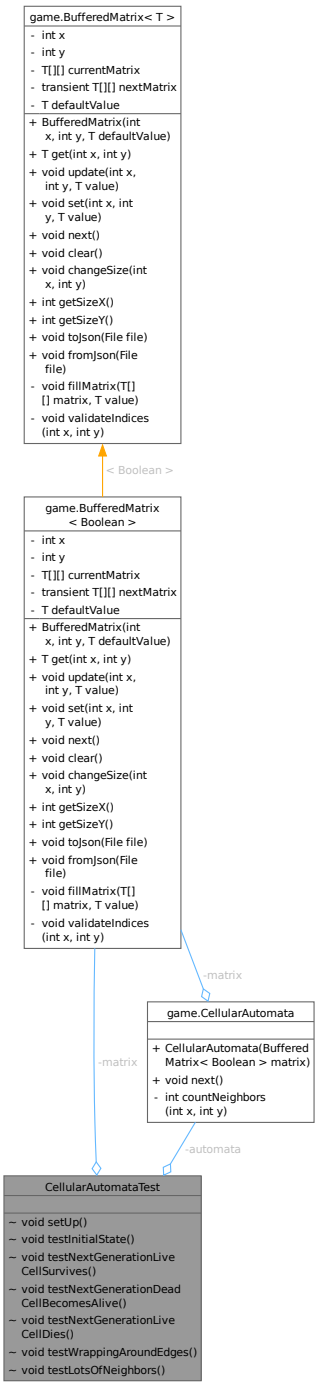
e	the key event to be processed.

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

- CellularAutomataPanel.java

3.7 CellularAutomataTest Class Reference

Collaboration diagram for CellularAutomataTest:



Package Functions

- void **setUp** ()
- void **testInitialState** ()

- void **testNextGenerationLiveCellSurvives** ()
- void **testNextGenerationDeadCellBecomesAlive** ()
- void **testNextGenerationLiveCellDies** ()
- void **testWrappingAroundEdges** ()
- void **testLotsOfNeighbors** ()

Private Attributes

- BufferedMatrix< Boolean > **matrix**
- CellularAutomata **automata**

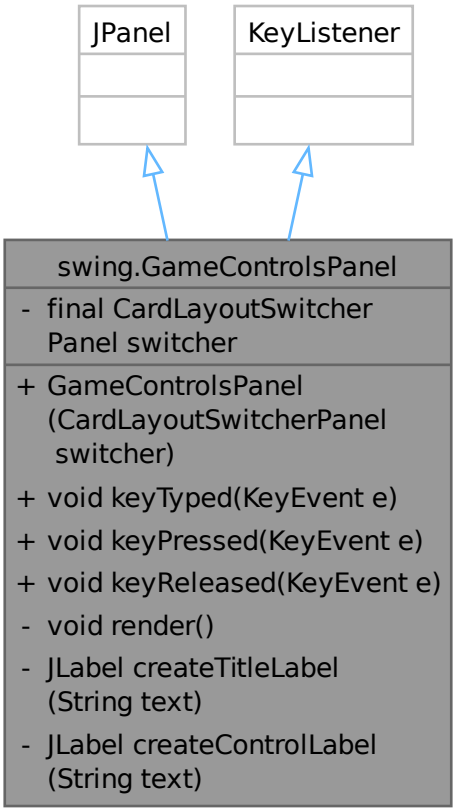
The documentation for this class was generated from the following file:

- CellularAutomataTest.java

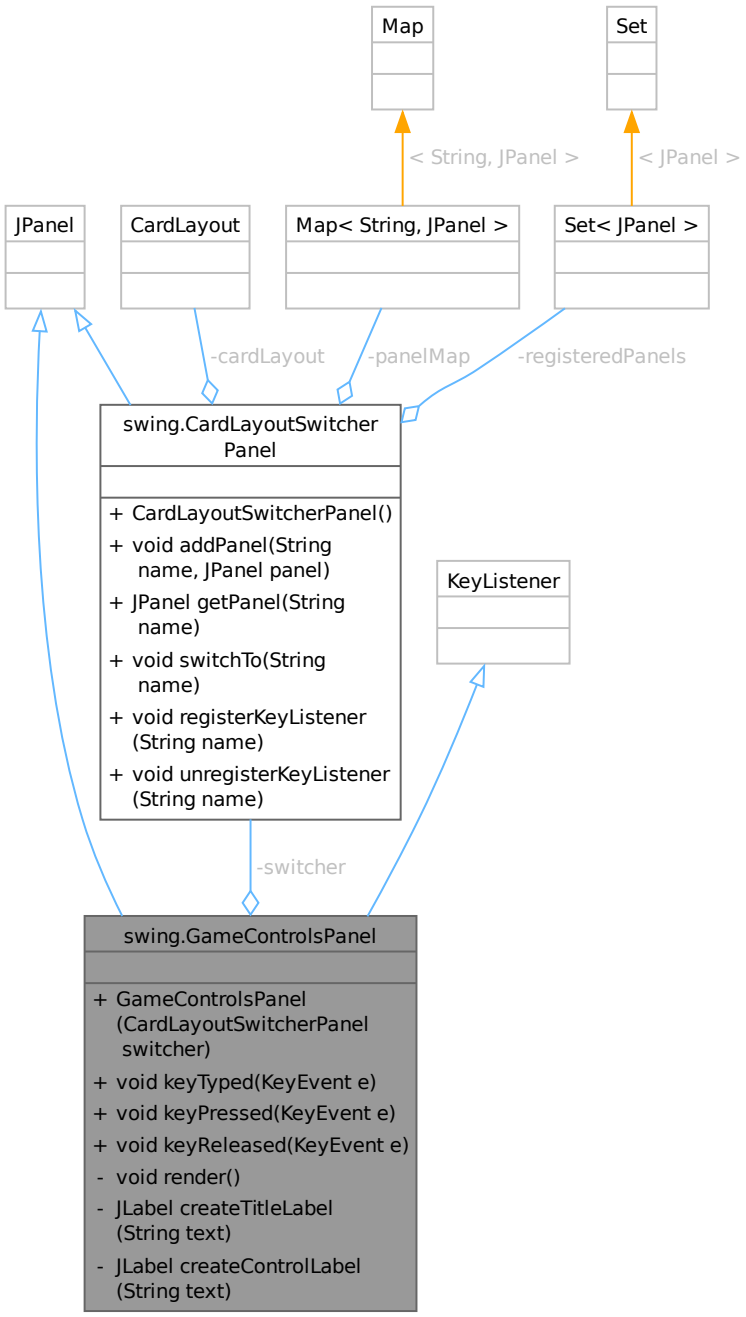
3.8 swing.GameControlsPanel Class Reference

The GameControlsPanel class provides a user interface for displaying the controls of a game.

Inheritance diagram for swing.GameControlsPanel:



Collaboration diagram for swing.GameControlsPanel:



Public Member Functions

- GameControlsPanel (CardLayoutSwitcherPanel switcher)

  Constructs a GameControlsPanel with the specified CardLayoutSwitcherPanel.
- void keyTyped (KeyEvent e)

  Processes key typed events.
- void keyPressed (KeyEvent e)

Processes key pressed events.

- void keyReleased (KeyEvent e)

  Processes key released events.

Private Member Functions

- void render ()

  Renders the user interface components of the panel, including the back button and control labels.
- JLabel createTitleLabel (String text)

  Creates a JLabel for section titles within the controls panel.
- JLabel createControlLabel (String text)

  Creates a JLabel for individual control instructions within the controls panel.

Private Attributes

- final CardLayoutSwitcherPanel **switcher**

3.8.1 Detailed Description

The GameControlsPanel class provides a user interface for displaying the controls of a game.

It extends JPanel and implements KeyListener to handle keyboard events.

This panel includes a back button that allows the user to return to the pause menu and displays labels for keyboard and mouse controls, including movement commands, game controls, and mouse interaction commands.

The panel listens for key events and responds to the ESC key by switching to the pause screen.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 GameControlsPanel()

```
swing.GameControlsPanel.GameControlsPanel (
            CardLayoutSwitcherPanel switcher)
```

Constructs a GameControlsPanel with the specified CardLayoutSwitcherPanel.

Parameters

switcher	The CardLayoutSwitcherPanel used to switch between different views of the application.

3.8.3 Member Function Documentation

3.8.3.1 createControlLabel()

```
JLabel swing.GameControlsPanel.createControlLabel (
            String text)  [private]
```

Creates a JLabel for individual control instructions within the controls panel.

Parameters

text	The text to display in the label.

Returns

A JLabel with the specified text and a plain font style.

3.8.3.2 createTitleLabel()

```
JLabel swing.GameControlsPanel.createTitleLabel (
            String text)  [private]
```

Creates a JLabel for section titles within the controls panel.

Parameters

text	The text to display in the label.

Returns

A JLabel with the specified text and a bold font style.

3.8.3.3 keyPressed()

```
void swing.GameControlsPanel.keyPressed (
            KeyEvent e)
```

Processes key pressed events.

If the ESC key is pressed, this method switches the view to the pause screen.

Parameters

e	the event to be processed

3.8.3.4 keyReleased()

```
void swing.GameControlsPanel.keyReleased (
            KeyEvent e)
```

Processes key released events.

This method is empty and can be overridden in subclasses to provide specific functionality.

Parameters

e	the event to be processed

3.8.3.5 keyTyped()

```
void swing.GameControlsPanel.keyTyped (
            KeyEvent e)
```

Processes key typed events.

This method is empty and can be overridden in subclasses to provide specific functionality.

Parameters

e	the event to be processed

3.8.3.6 render()

```
void swing.GameControlsPanel.render ()  [private]
```

Renders the user interface components of the panel, including the back button and control labels.

This method sets the layout and adds all necessary components to the panel.

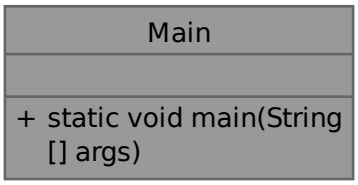
The documentation for this class was generated from the following file:

- GameControlsPanel.java

3.9 Main Class Reference

The Main class serves as the entry point for the Game of Life application.

Collaboration diagram for Main:



Static Public Member Functions

- static void main (String[ ] args)

  The main method is the entry point for the Game of Life application.

3.9.1 Detailed Description

The Main class serves as the entry point for the Game of Life application.

It sets up the main frame and the panels for different game states, such as the main menu, pause menu, matrix size selection, the game grid, and game controls. The game uses a card layout system to switch between these different panels.

The application is based on John Conway's Game of Life, a cellular automaton that simulates the evolution of a grid of cells according to specific rules. This class initializes the main game grid and control panels, handling the flow between different game states.

3.9.2 Member Function Documentation

3.9.2.1 main()

```
static void Main.main (
            String[] args)  [static]
```

The main method is the entry point for the Game of Life application.

It creates the main application window, sets its size, and initializes the various panels that allow users to control the game.

This includes:

- A main menu for starting or loading a game
- A pause menu for saving and controlling game behavior
- A matrix size selection panel to configure the game grid dimensions
- The main game grid where the simulation takes place
- Controls for running or pausing the simulation

The game uses a `CardLayoutSwitcherPanel` to switch between different screens in the game, such as the main menu, the game grid, and other control panels.

Parameters

args	command-line arguments (not used)

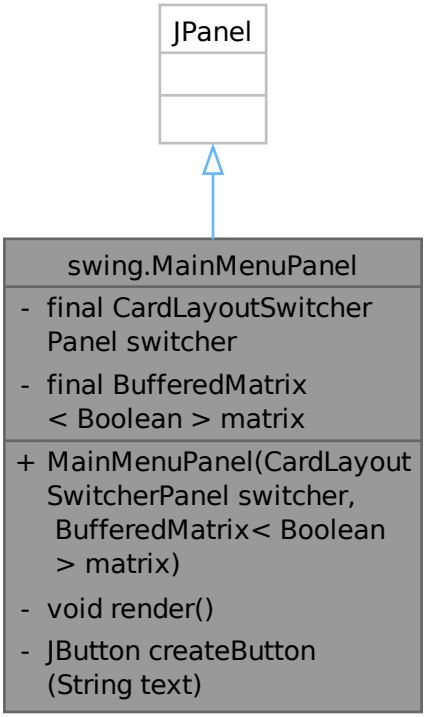
The documentation for this class was generated from the following file:

- Main.java

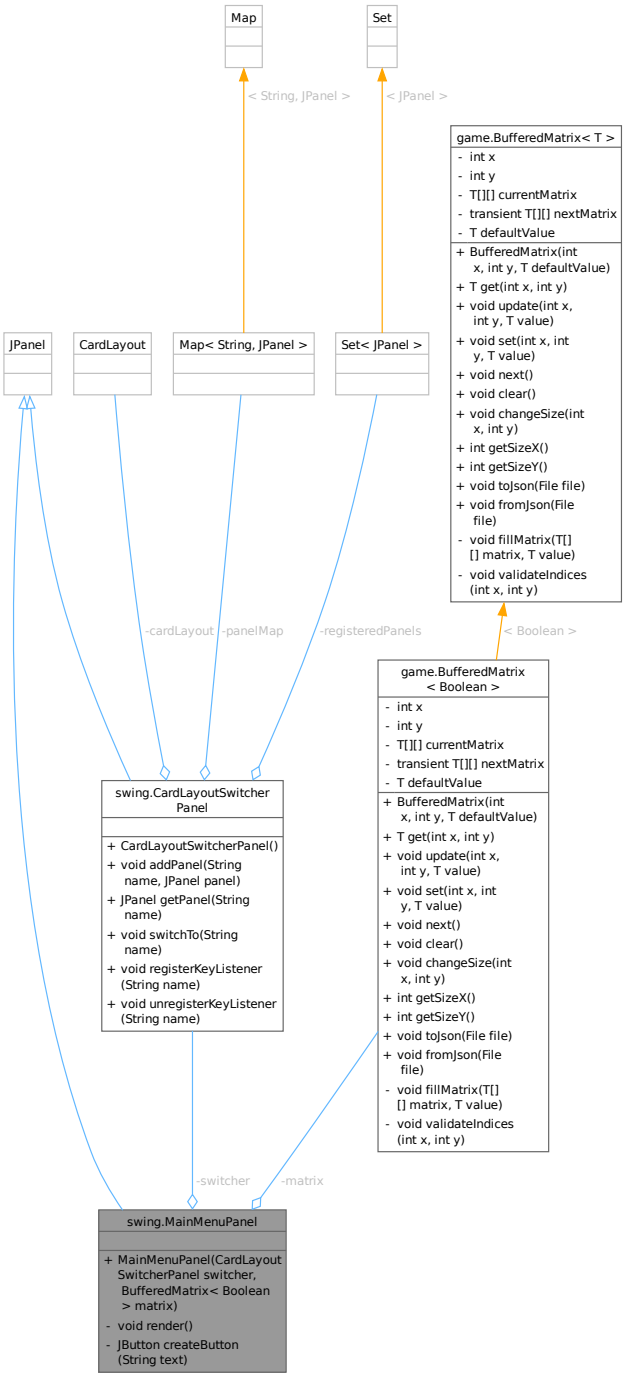
3.10 swing.MainMenuPanel Class Reference

MainMenuPanel represents the main menu of the game, allowing users to start a new game, load a saved game, or exit the application.

Inheritance diagram for swing.MainMenuPanel:



Collaboration diagram for swing.MainMenuPanel:



Public Member Functions

- MainMenuPanel (CardLayoutSwitcherPanel switcher, BufferedMatrix< Boolean > matrix)

  Constructs a MainMenuPanel with the specified CardLayoutSwitcherPanel and BufferedMatrix.

Private Member Functions

- void render ()

Sets up the layout and components of the main menu.

- JButton createButton (String text)

  Creates a JButton with the specified text.

Private Attributes

- final CardLayoutSwitcherPanel **switcher**

  The panel that handles switching between different game panels.
- final BufferedMatrix< Boolean > **matrix**

  The matrix representing the game state.

3.10.1 Detailed Description

MainMenuPanel represents the main menu of the game, allowing users to start a new game, load a saved game, or exit the application.

This panel contains buttons for navigating to different functionalities of the game. It utilizes a CardLayoutSwitcherPanel to switch between different game panels.

3.10.2 Constructor & Destructor Documentation

3.10.2.1 MainMenuPanel()

```
swing.MainMenuPanel.MainMenuPanel (
            CardLayoutSwitcherPanel switcher,
            BufferedMatrix< Boolean > matrix)
```

Constructs a MainMenuPanel with the specified CardLayoutSwitcherPanel and BufferedMatrix.

Parameters

switcher	The panel used for switching between different game views.
matrix	The matrix that holds the game's state data.

3.10.3 Member Function Documentation

3.10.3.1 createButton()

```
JButton swing.MainMenuPanel.createButton (
            String text)  [private]
```

Creates a JButton with the specified text.

Parameters

text	The text to be displayed on the button.

Returns

A JButton instance with the specified text.

3.10.3.2 render()

```
void swing.MainMenuPanel.render ()  [private]
```

Sets up the layout and components of the main menu.

This includes adding buttons for playing, loading a game, and exiting.

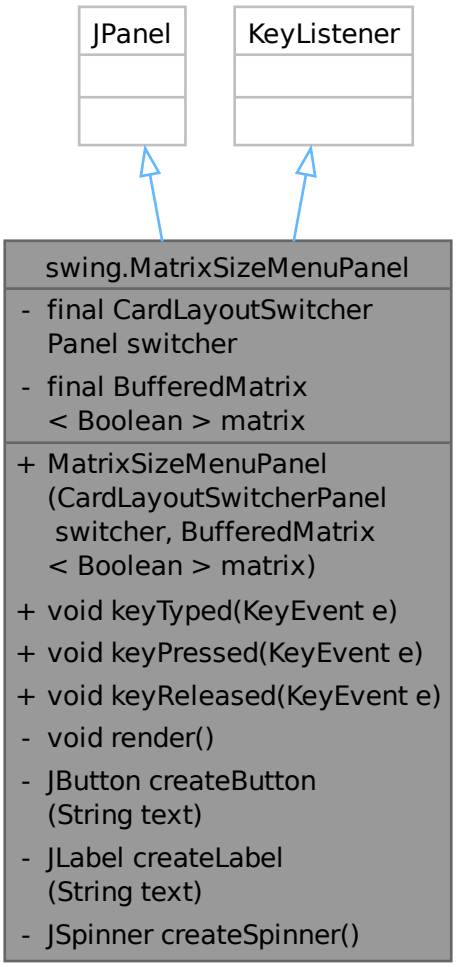
The documentation for this class was generated from the following file:

- MainMenuPanel.java

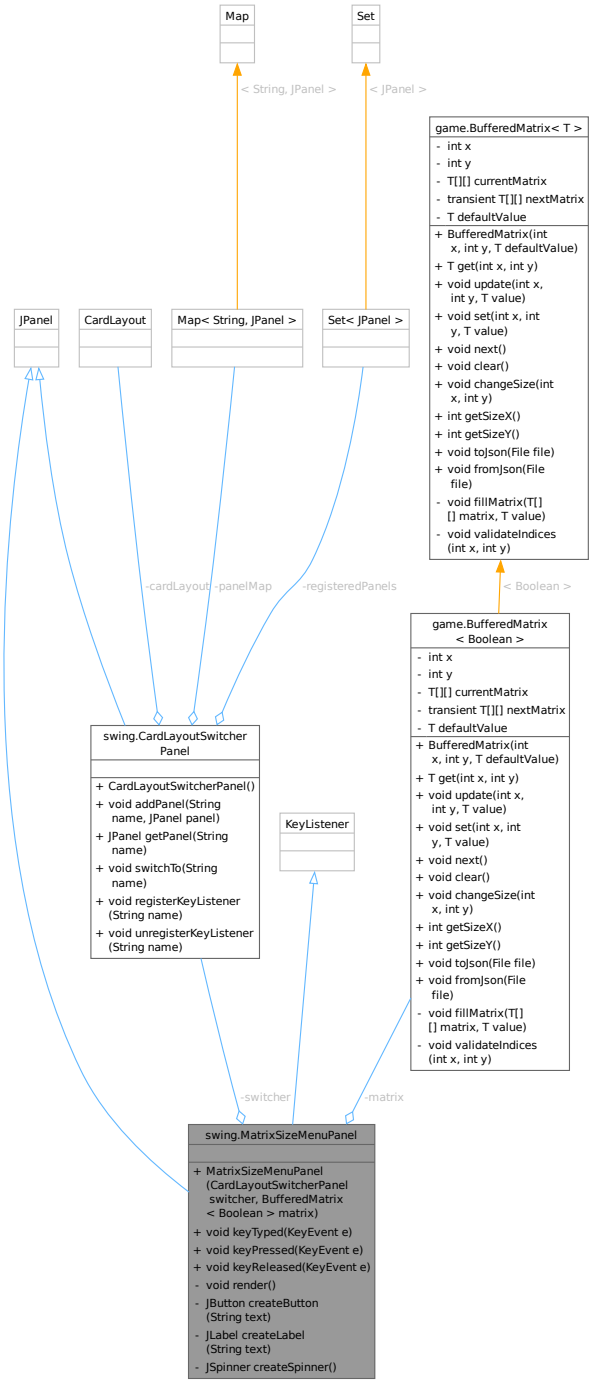
3.11 swing.MatrixSizeMenuPanel Class Reference

This class represents the menu panel for setting the size of the matrix in the Game of Life application.

Inheritance diagram for swing.MatrixSizeMenuPanel:



Collaboration diagram for swing.MatrixSizeMenuPanel:



Public Member Functions

- MatrixSizeMenuPanel (CardLayoutSwitcherPanel switcher, BufferedMatrix< Boolean > matrix)

  Constructs a new MatrixSizeMenuPanel with the specified switcher and matrix.
- void keyTyped (KeyEvent e)

  Processes a key typed event.
- void keyPressed (KeyEvent e)

Processes a key pressed event.

- void keyReleased (KeyEvent e)

  Processes a key released event.

Private Member Functions

- void **render** ()

  Initializes and arranges the components of the panel, including labels, spinners, and buttons for user interaction.
- JButton createButton (String text)

  Creates a JButton with the specified text and default font settings.
- JLabel createLabel (String text)

  Creates a JLabel with the specified text and default font settings.
- JSpinner createSpinner ()

  Creates a JSpinner configured to allow numeric input for specifying matrix dimensions.

Private Attributes

- final CardLayoutSwitcherPanel **switcher**

  The CardLayoutSwitcherPanel that manages the switching between different panels.
- final BufferedMatrix< Boolean > **matrix**

  The BufferedMatrix that holds the state of the grid.

3.11.1 Detailed Description

This class represents the menu panel for setting the size of the matrix in the Game of Life application.

It allows the user to specify the number of rows and columns for the grid. The panel includes buttons for navigation and input fields for user interaction.

3.11.2 Constructor & Destructor Documentation

3.11.2.1 MatrixSizeMenuPanel()

```
swing.MatrixSizeMenuPanel.MatrixSizeMenuPanel (
            CardLayoutSwitcherPanel switcher,
            BufferedMatrix< Boolean > matrix)
```

Constructs a new MatrixSizeMenuPanel with the specified switcher and matrix.

Parameters

switcher	The CardLayoutSwitcherPanel used for navigating between panels.
matrix	The BufferedMatrix to be configured with user-defined dimensions.

3.11.3 Member Function Documentation

3.11.3.1 createButton()

```
JButton swing.MatrixSizeMenuPanel.createButton (
            String text)  [private]
```

Creates a JButton with the specified text and default font settings.

Parameters

text	The text to be displayed on the button.

Returns

A JButton configured with the specified text.

3.11.3.2 createLabel()

```
JLabel swing.MatrixSizeMenuPanel.createLabel (
            String text)  [private]
```

Creates a JLabel with the specified text and default font settings.

Parameters

text	The text to be displayed on the label.

Returns

A JLabel configured with the specified text.

3.11.3.3 createSpinner()

```
JSpinner swing.MatrixSizeMenuPanel.createSpinner ()  [private]
```

Creates a JSpinner configured to allow numeric input for specifying matrix dimensions.

Returns

A JSpinner configured with a default value and range.

3.11.3.4 keyPressed()

```
void swing.MatrixSizeMenuPanel.keyPressed (
            KeyEvent e)
```

Processes a key pressed event.

Allows the user to navigate back to the home panel using the ESC key.

Parameters

e	The key event to be processed.

3.11.3.5 keyReleased()

```
void swing.MatrixSizeMenuPanel.keyReleased (
            KeyEvent e)
```

Processes a key released event.

Parameters

e	The key event to be processed.

3.11.3.6 keyTyped()

```
void swing.MatrixSizeMenuPanel.keyTyped (
            KeyEvent e)
```

Processes a key typed event.

Parameters

e	The key event to be processed.

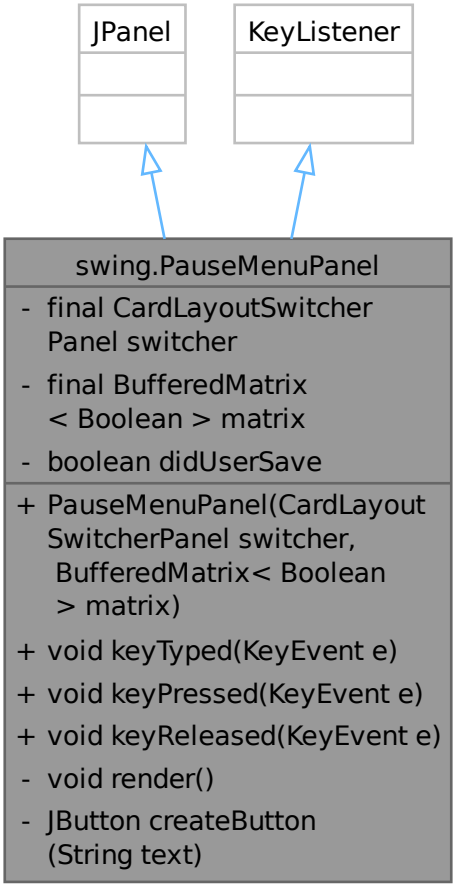
The documentation for this class was generated from the following file:

- MatrixSizeMenuPanel.java

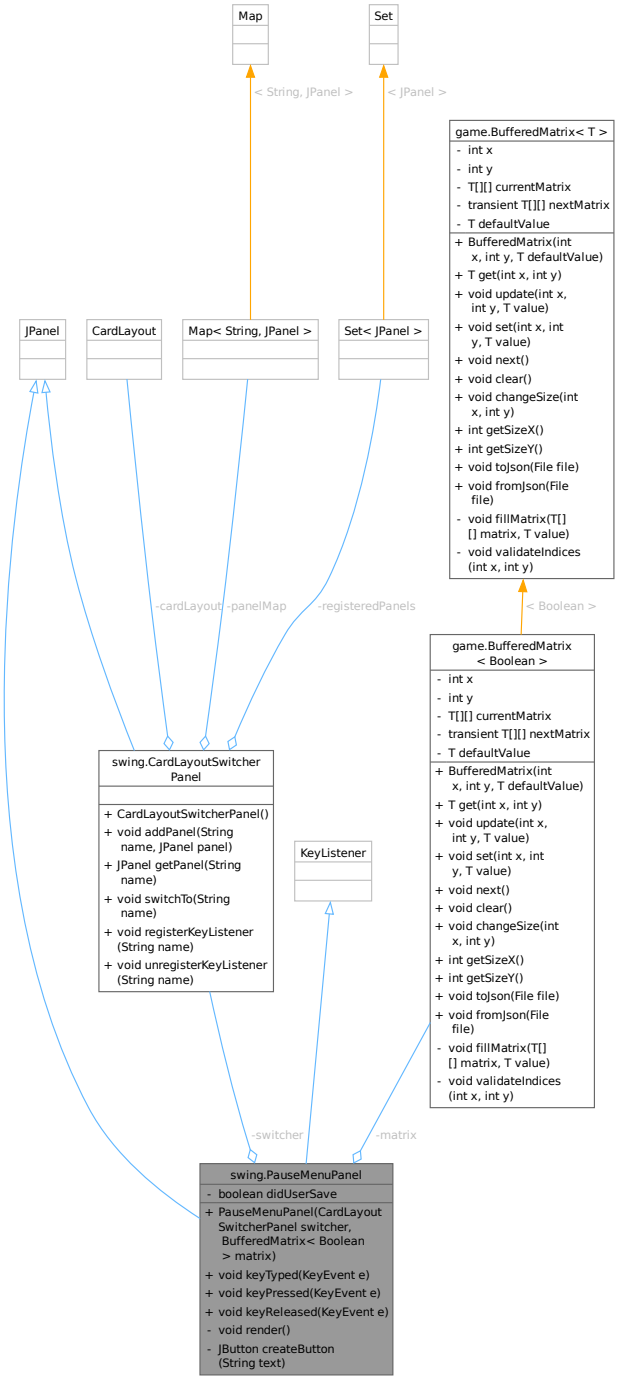
3.12 swing.PauseMenuPanel Class Reference

The `PauseMenuPanel` class represents a pause menu in the game.

Inheritance diagram for swing.PauseMenuPanel:



Collaboration diagram for swing.PauseMenuPanel:



Public Member Functions

- PauseMenuPanel (CardLayoutSwitcherPanel switcher, BufferedMatrix< Boolean > matrix)

  Constructs a new `PauseMenuPanel` with the specified switcher and matrix.
- void keyTyped (KeyEvent e)

  Handles the key typed event.
- void keyPressed (KeyEvent e)

Handles the key pressed event.

- void keyReleased (KeyEvent e)

  Handles the key released event.

Private Member Functions

- void render ()

  Initializes and lays out the components in the pause menu.
- JButton createButton (String text)

  Creates a JButton with the specified text.

Private Attributes

- final CardLayoutSwitcherPanel **switcher**

  The `CardLayoutSwitcherPanel` used to switch between different panels.
- final BufferedMatrix< Boolean > **matrix**

  The `BufferedMatrix` that represents the game state.
- boolean **didUserSave** = false

  A flag to indicate whether the user has saved their progress.

3.12.1 Detailed Description

The `PauseMenuPanel` class represents a pause menu in the game.

It provides options for the player to navigate back to the game, access game controls, save the current game state, or return to the main menu. This panel also listens for keyboard events to allow for quick navigation using keyboard keys.

3.12.2 Constructor & Destructor Documentation

3.12.2.1 PauseMenuPanel()

```
swing.PauseMenuPanel.PauseMenuPanel (
            CardLayoutSwitcherPanel switcher,
            BufferedMatrix< Boolean > matrix)
```

Constructs a new `PauseMenuPanel` with the specified switcher and matrix.

Parameters

switcher	the `CardLayoutSwitcherPanel` used for panel switching
matrix	the `BufferedMatrix` representing the current game state

3.12.3 Member Function Documentation

3.12.3.1 createButton()

```
JButton swing.PauseMenuPanel.createButton (
            String text)  [private]
```

Creates a JButton with the specified text.

Parameters

text	the text to be displayed on the button

Returns

a JButton configured with the specified text

3.12.3.2 keyPressed()

```
void swing.PauseMenuPanel.keyPressed (
            KeyEvent e)
```

Handles the key pressed event.

If the Escape key is pressed, the user is returned to the game grid.

Parameters

e	the event to be processed

3.12.3.3 keyReleased()

```
void swing.PauseMenuPanel.keyReleased (
            KeyEvent e)
```

Handles the key released event.

This method is not implemented.

Parameters

e	the event to be processed

3.12.3.4 keyTyped()

```
void swing.PauseMenuPanel.keyTyped (
            KeyEvent e)
```

Handles the key typed event.

This method is not implemented.

Parameters

e	the event to be processed

3.12.3.5 render()

```
void swing.PauseMenuPanel.render ()  [private]
```

Initializes and lays out the components in the pause menu.

This method sets up the buttons for navigating the pause menu, and it defines their corresponding actions when clicked.

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

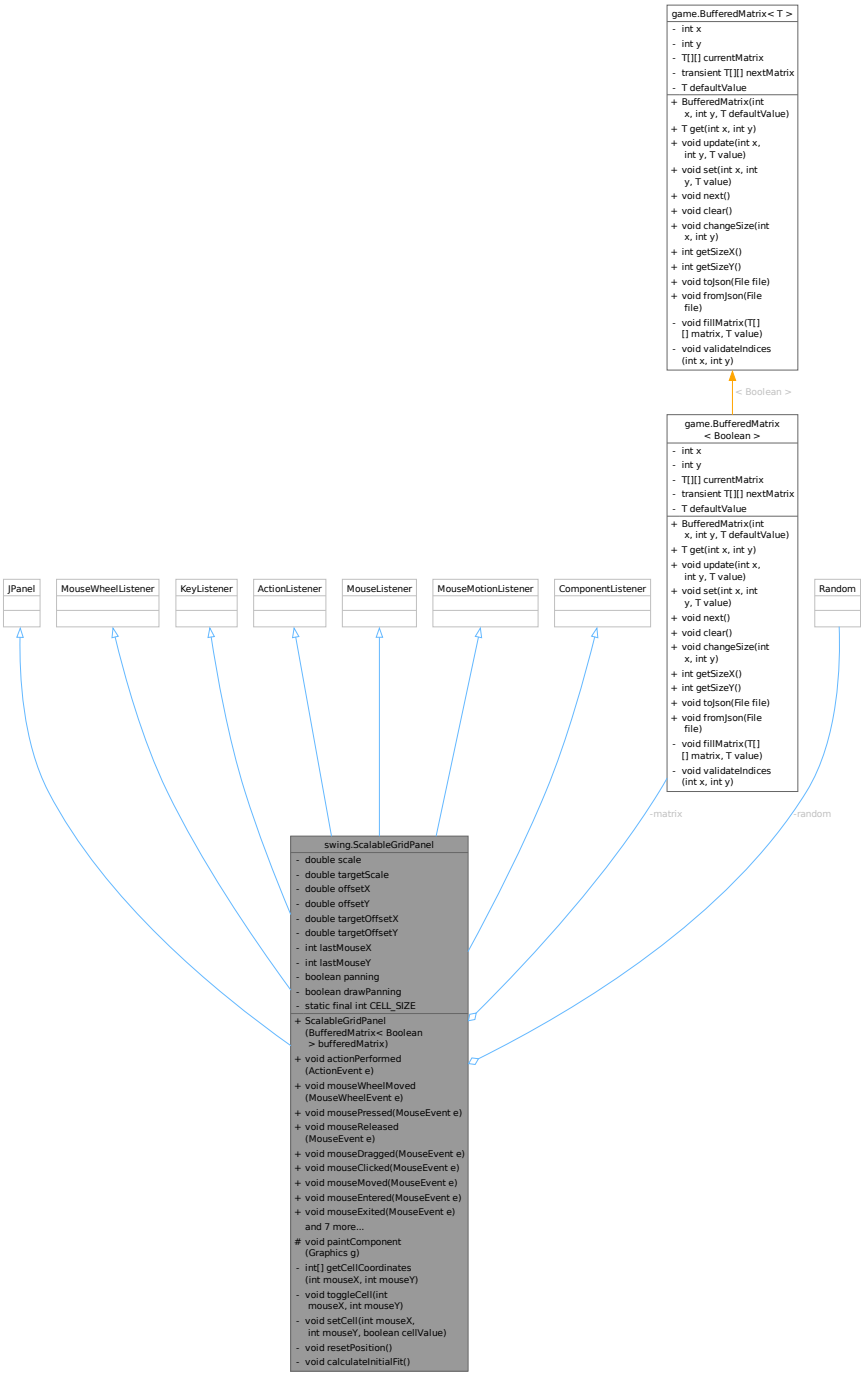
- PauseMenuPanel.java

3.13 swing.ScalableGridPanel Class Reference

ScalableGridPanel is a custom JPanel designed to display a grid based on a BufferedMatrix of Boolean values.

Inheritance diagram for swing.ScalableGridPanel:

Collaboration diagram for swing.ScalableGridPanel:



Public Member Functions

- ScalableGridPanel (BufferedMatrix< Boolean > bufferedMatrix)

  Constructs a new ScalableGridPanel with the given BufferedMatrix.
- void actionPerformed (ActionEvent e)

  Smoothly updates the zoom and pan offsets in response to user interactions.
- void mouseWheelMoved (MouseWheelEvent e)

Responds to mouse wheel scrolling to zoom in or out of the grid.

- void mousePressed (MouseEvent e)

  Handles mouse press events.
- void mouseReleased (MouseEvent e)

  Handles mouse release events.
- void mouseDragged (MouseEvent e)

  Handles mouse drag events.
- void mouseClicked (MouseEvent e)

  Responds to mouse click events.
- void mouseMoved (MouseEvent e)

  Responds to mouse move events.
- void mouseEntered (MouseEvent e)

  Responds to mouse enter events.
- void mouseExited (MouseEvent e)

  Responds to mouse exit events.
- void keyPressed (KeyEvent e)

  Handles key press events.
- void keyReleased (KeyEvent e)

  Responds to key release events.
- void keyTyped (KeyEvent e)

  Responds to key typed events.
- void componentResized (ComponentEvent e)

  Handles component resize events.
- void componentMoved (ComponentEvent e)

  Responds to component moved events.
- void componentShown (ComponentEvent e)

  Responds to component shown events.
- void componentHidden (ComponentEvent e)

  Responds to component hidden events.

Protected Member Functions

- void paintComponent (Graphics g)

  Paints the grid of cells onto the panel.

Private Member Functions

- int[ ] getCellCoordinates (int mouseX, int mouseY)

  Converts screen coordinates (mouse X, Y) to grid coordinates (row, column).
- void toggleCell (int mouseX, int mouseY)

  Toggles the state of the cell (alive or dead) at the given mouse position.
- void setCell (int mouseX, int mouseY, boolean cellValue)

  Sets the value of the cell at the given mouse position to a specified value.
- void **resetPosition** ()

  Resets the zoom and pan position to the default view.
- void **calculateInitialFit** ()

  Automatically calculates the best initial zoom to fit the entire grid within the panel.

Private Attributes

- final BufferedMatrix< Boolean > **matrix**

  The matrix representing the grid where each cell holds a Boolean value indicating whether the cell is alive (true) or dead (false).
- final Random **random** = new Random()

  Random instance used for randomizing the grid.
- double **scale** = 1.0

  The current zoom scale factor applied to the grid.
- double **targetScale** = 1.0

  The target zoom scale factor for smooth zooming transitions.
- double **offsetX** = 0

  The current horizontal and vertical offsets for panning.
- double **offsetY** = 0
- double **targetOffsetX** = 0

  The target horizontal and vertical offsets for smooth panning transitions.
- double **targetOffsetY** = 0
- int **lastMouseX**

  The last recorded mouse coordinates used for panning the grid.
- int **lastMouseY**
- boolean **panning** = false

  A flag indicating whether panning mode is currently active.
- boolean **drawPanning** = false

  A flag indicating whether draw-panning mode is active for drawing cells by dragging.

Static Private Attributes

- static final int **CELL_SIZE** = 50

  The size of each grid cell in pixels.

3.13.1 Detailed Description

ScalableGridPanel is a custom JPanel designed to display a grid based on a BufferedMatrix of Boolean values.

This panel supports zooming, panning, and modifying the grid cells via mouse interactions and key presses. It is mainly used for visualizing and interacting with cellular automata in the context of Conway's Game of Life or similar grid-based games.

The class allows users to zoom in and out using the mouse wheel, pan around the grid by dragging with the right mouse button, and toggle cells on or off by clicking them. Additional functionality includes resetting the grid, randomizing cells, and moving the viewport using keyboard inputs.

3.13.2 Constructor & Destructor Documentation

3.13.2.1 ScalableGridPanel()

```
swing.ScalableGridPanel.ScalableGridPanel (
            BufferedMatrix< Boolean > bufferedMatrix)
```

Constructs a new ScalableGridPanel with the given BufferedMatrix.

Parameters

bufferedMatrix	The matrix representing the state of the grid.

3.13.3 Member Function Documentation

3.13.3.1 actionPerformed()

```
void swing.ScalableGridPanel.actionPerformed (
            ActionEvent e)
```

Smoothly updates the zoom and pan offsets in response to user interactions.

Parameters

e	The ActionEvent triggered by the movement timer.

3.13.3.2 componentHidden()

```
void swing.ScalableGridPanel.componentHidden (
            ComponentEvent e)
```

Responds to component hidden events.

Currently does nothing.

Parameters

e	The ComponentEvent triggered when the component is hidden.

3.13.3.3 componentMoved()

```
void swing.ScalableGridPanel.componentMoved (
            ComponentEvent e)
```

Responds to component moved events.

Currently does nothing.

Parameters

e	The ComponentEvent triggered when the component is moved.

3.13.3.4 componentResized()

```
void swing.ScalableGridPanel.componentResized (
            ComponentEvent e)
```

Handles component resize events.

Resets the position of the view to fit the resized component.

Parameters

e	The ComponentEvent triggered by resizing the component.

3.13.3.5 componentShown()

```
void swing.ScalableGridPanel.componentShown (
            ComponentEvent e)
```

Responds to component shown events.

Resets the position of the view when the component becomes visible.

Parameters

e	The ComponentEvent triggered when the component is shown.

3.13.3.6 getCellCoordinates()

```
int[] swing.ScalableGridPanel.getCellCoordinates (
            int mouseX,
            int mouseY)  [private]
```

Converts screen coordinates (mouse X, Y) to grid coordinates (row, column).

Parameters

mouseX	The X-coordinate of the mouse.
mouseY	The Y-coordinate of the mouse.

Returns

An array with row and column coordinates, or null if outside the grid.

3.13.3.7 keyPressed()

```
void swing.ScalableGridPanel.keyPressed (
            KeyEvent e)
```

Handles key press events.

It allows the user to move the view using arrow keys or WASD, reset the view with the Home key, clear the grid with the 'R' key, or randomize the grid with the 'F' key.

Parameters

e	The KeyEvent triggered by a key press.

3.13.3.8 keyReleased()

```
void swing.ScalableGridPanel.keyReleased (
            KeyEvent e)
```

Responds to key release events.

Currently does nothing.

Parameters

e	The KeyEvent triggered by a key release.

3.13.3.9 keyTyped()

```
void swing.ScalableGridPanel.keyTyped (
            KeyEvent e)
```

Responds to key typed events.

Currently does nothing.

Parameters

e	The KeyEvent triggered by a key typed action.

3.13.3.10 mouseClicked()

```
void swing.ScalableGridPanel.mouseClicked (
            MouseEvent e)
```

Responds to mouse click events.

Currently does nothing.

Parameters

e	The MouseEvent triggered by a mouse click.

3.13.3.11 mouseDragged()

```
void swing.ScalableGridPanel.mouseDragged (
            MouseEvent e)
```

Handles mouse drag events.

If panning is active, it updates the target offsets based on the mouse movement. If draw-panning is active, it sets the cell state at the current mouse position to alive.

Parameters

e	The MouseEvent triggered by dragging the mouse.

3.13.3.12 mouseEntered()

```
void swing.ScalableGridPanel.mouseEntered (
            MouseEvent e)
```

Responds to mouse enter events.

Currently does nothing.

Parameters

e	The MouseEvent triggered when the mouse enters the component.

3.13.3.13 mouseExited()

```
void swing.ScalableGridPanel.mouseExited (
            MouseEvent e)
```

Responds to mouse exit events.

Currently does nothing.

Parameters

e	The MouseEvent triggered when the mouse exits the component.

3.13.3.14 mouseMoved()

```
void swing.ScalableGridPanel.mouseMoved (
            MouseEvent e)
```

Responds to mouse move events.

Currently does nothing.

Parameters

e	The MouseEvent triggered by moving the mouse.

3.13.3.15 mousePressed()

```
void swing.ScalableGridPanel.mousePressed (
            MouseEvent e)
```

Handles mouse press events.

Depending on the button pressed, it either toggles the cell at the mouse location, enables draw-panning, or activates panning mode.

Parameters

e	The MouseEvent triggered by the mouse press.

3.13.3.16 mouseReleased()

```
void swing.ScalableGridPanel.mouseReleased (
            MouseEvent e)
```

Handles mouse release events.

It deactivates panning or draw-panning mode when the corresponding mouse buttons are released.

Parameters

e	The MouseEvent triggered by the mouse release.

3.13.3.17 mouseWheelMoved()

```
void swing.ScalableGridPanel.mouseWheelMoved (
            MouseWheelEvent e)
```

Responds to mouse wheel scrolling to zoom in or out of the grid.

Parameters

e	The MouseWheelEvent triggered by scrolling.

3.13.3.18 paintComponent()

```
void swing.ScalableGridPanel.paintComponent (
            Graphics g)  [protected]
```

Paints the grid of cells onto the panel.

Cells are drawn as black (alive) or white (dead).

Parameters

g	The Graphics object used for drawing.

3.13.3.19 setCell()

```
void swing.ScalableGridPanel.setCell (
            int mouseX,
            int mouseY,
            boolean cellValue)  [private]
```

Sets the value of the cell at the given mouse position to a specified value.

Parameters

mouseX	The X-coordinate of the mouse.
mouseY	The Y-coordinate of the mouse.
cellValue	The value to set the cell to (true for alive, false for dead).

3.13.3.20 toggleCell()

```
void swing.ScalableGridPanel.toggleCell (
            int mouseX,
            int mouseY)  [private]
```

Toggles the state of the cell (alive or dead) at the given mouse position.

Parameters

mouseX	The X-coordinate of the mouse.
mouseY	The Y-coordinate of the mouse.

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

- ScalableGridPanel.java

Index