Class BackgroundGame

java.lang.Object
  java.awt.Component
    java.awt.Container
      javax.swing.JComponent
        javax.swing.JPanel
          BackgroundGame

All Implemented Interfaces:

java.awt.event.KeyListener, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, java.util.EventListener, javax.accessibility.Accessible

---

```
public class BackgroundGame
extends javax.swing.JPanel
implements java.awt.event.KeyListener
```

The clean-up game in the background whilst the popups appear in the foreground

See Also:

Serialized Form

Nested Class Summary

Nested classes/interfaces inherited from class javax.swing.JPanel

`javax.swing.JPanel.AccessibleJPanel`

Nested classes/interfaces inherited from class javax.swing.JComponent

`javax.swing.JComponent.AccessibleJComponent`

Nested classes/interfaces inherited from class java.awt.Container

`java.awt.Container.AccessibleAWTContainer`

Nested classes/interfaces inherited from class java.awt.Component

`java.awt.Component.AccessibleAWTComponent, java.awt.Component.BaselineResizeBehavior, java.awt.Component.BltBufferStrategy, java.awt.Component.FlipBufferStrategy`

Field Summary

Fields

Modifier and Type	Field and Description
private double	**cpuUsage** A number which if exceeding 100 will cause the loss of the game
private boolean	**isOver** Whether the game has been lost;
private boolean	**isPaused** Whether the game is paused.
private boolean	**isStarted** Whether the game has been started.
private long	**lastLogicCycleTime** Some number of nanoseconds representing a moment in the past when the logic loop was run.
private java.lang.Object	**lock** A dummy object used for synchronization.
private int	**logicFps** How many cycles of game logic to execute per second
private java.util.ArrayList<**GameObject**>	**objects** A list of all of the GameObjects.
private java.util.ArrayList<**Question**>	**questions** List of questions that may appear in the pop-ups.
private **RecycleBin**	**rb** The sole RecyclingBin object in the game
private java.util.HashMap<java.lang.String,java.awt.image.BufferedImage>	**sprites** A map of string identifiers to BufferedImages
private long	**timeFirstPaused** A moment to be used later in offseting time paused from the time elapsed in-game.
private long	**timeGameEnded** A nanosecond moment representing when the game was lost.
private long	**timeGameStarted** Some number of nanoseconds representing a moment in the past when the game was started.

Fields inherited from class javax.swing.JComponent

accessibleContext, listenerList, TOOL_TIP_TEXT_KEY, ui, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Constructor Summary

Constructors

Constructor and Description
BackgroundGame(java.awt.Dimension d) The constructor.

Method Summary

Methods

Modifier and Type	Method and Description
void	**decreaseCpuUsage**(double val) Decreases the CPU usage by some amount.
private void	**drawGameOverScreen**(java.awt.Graphics g) Draws a BSOD with game information, signifying a game over.
private void	**drawTitleScreen**(java.awt.Graphics g) Draws the instructive title screen.
void	**endGame**() Routine for ending the game (showing the blue-screen).
private void	**gameCycle**() What to do whilst the game is running.
double	**getCpuUsage**() Accesses the cpuUsage variable.
java.util.HashMap<java.lang.String,java.awt.image.BufferedImage>	**getSprites**() Accesses the sprites member.
long	**getTimeGameStarted**() Accesses the timeGameStarted member.
void	**increaseCpuUsage**(double val) Increases the CPU usage by some amount.
boolean	**isOver**() Whether the game is over
boolean	**isPaused**() Whether the game is paused.
boolean	**isStarted**() whether the game has started

void	**keyPressed**(java.awt.event.KeyEvent e) Gives the recycle bin acceleration on depression of the left or right arrow keys.
void	**keyReleased**(java.awt.event.KeyEvent e) Remove the acceleration from the RecycleBin when the arrow keys are released.
void	**keyTyped**(java.awt.event.KeyEvent e)
private void	**loadQuestions**() Loads the questions from QuestionBank.txt.
private void	**loadSprites**() Loads all of the requisite images from the working directory, 7 in all.
private void	**makeDialog**() Create a pop-up question.
void	**paintComponent**(java.awt.Graphics g) Draws the sprites of all of the GameObjects
void	**startGame**() Begins the game proper!
private void	**togglePaused**() Toggles the paused state of the game.

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUI, getUIClassID, paramString, setUI, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, fireVetoableChange, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentGraphics, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingOrigin, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintBorder, paintChildren, paintImmediately, paintImmediately, print, printAll, printBorder, printChildren, printComponent, processComponentKeyEvent, processKeyBinding, processKeyEvent, processMouseEvent, processMouseMotionEvent, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setUI, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addImpl, addPropertyChangeListener,

addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getLayout, getMousePosition, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paintComponents, preferredSize, printComponents, processContainerEvent, processEvent, remove, remove, removeAll, removeContainerListener, setComponentZOrder, setFocusCycleRoot, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setLayout, transferFocusDownCycle, validate, validateTree

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, bounds, checkImage, checkImage, coalesceEvents, contains, createImage, createImage, createVolatileImage, createVolatileImage, disableEvents, dispatchEvent, enable, enableEvents, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getBackground, getBounds, getColorModel, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeysEnabled, getFont, getForeground, getGraphicsConfiguration, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getLocale, getLocation, getLocationOnScreen, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getToolkit, getTreeLock, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, isBackgroundSet, isCursorSet, isDisplayable, isEnabled, isFocusable, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, list, list, list, location, lostFocus, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paintAll, postEvent, prepareImage, prepareImage, processComponentEvent, processFocusEvent, processHierarchyBoundsEvent, processHierarchyEvent, processInputMethodEvent, processMouseWheelEvent, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, resize, resize, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setFocusable, setFocusTraversalKeysEnabled, setIgnoreRepaint, setLocale, setLocation, setLocation, setName, setSize, setSize, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

rb

```
private RecycleBin rb
```

The sole RecyclingBin object in the game

sprites

```
private java.util.HashMap<java.lang.String,java.awt.image.BufferedImage> sprites
```

A map of string identifiers to BufferedImages

objects

```
private java.util.ArrayList<GameObject> objects
```

A list of all of the GameObjects. Iterated through in the game loop.

cpuUsage

```
private double cpuUsage
```

A number which if exceeding 100 will cause the loss of the game

logicFps

```
private final int logicFps
```

How many cycles of game logic to execute per second

See Also:

Constant Field Values

lastLogicCycleTime

```
private long lastLogicCycleTime
```

Some number of nanoseconds representing a moment in the past when the logic loop was run. Used for pacing with logicFps.

isStarted

```
private boolean isStarted
```

Whether the game has been started. Note that the game goes to the bitter end (the user's loss).

isPaused

```
private boolean isPaused
```

Whether the game is paused.

isOver

```
private boolean isOver
```

Whether the game has been lost;

timeGameStarted

```
private long timeGameStarted
```

Some number of nanoseconds representing a moment in the past when the game was started.

timeFirstPaused

```
private long timeFirstPaused
```

A moment to be used later in offseting time paused from the time elapsed in-game.

lock

```
private final java.lang.Object lock
```

A dummy object used for synchronization. Used primarily to isolate adding GameObjects to objects and iterating through objects.

timeGameEnded

```
private long timeGameEnded
```

A nanosecond moment representing when the game was lost.

questions

```
private java.util.ArrayList<Question> questions
```

List of questions that may appear in the pop-ups.

Constructor Detail

BackgroundGame

```
public BackgroundGame(java.awt.Dimension d)
```

The constructor. Loads all of the sprites. Creates the recycle bin for the user to play with even before starting the game. Initializes the list of GameObjects. Begins a game loop in a separate thread. This loop processes: - Running time - Removal of GameObjects marked for removal - Calling handlers for when an object escapes its set boundaries - Handles collisions (in separate threads) - Calls the cycle() function of each GameObject - Calls { gameCycle} The bulk of the loop is in a synchronized block to prevent concurrent modification and access of the list of GameObjects. Also begins a paint thread for continuous redrawing.

Parameters:

`d` - The size of the game.

Method Detail

getSprites

```
public java.util.HashMap<java.lang.String,java.awt.image.BufferedImage> getSprites()
```

Accesses the sprites member.

Returns:

A map of string identifiers to BufferedImages

getTimeGameStarted

```
public long getTimeGameStarted()
```

Accesses the timeGameStarted member.

Returns:

A nano-second moment representing when the game started

keyTyped

```
public void keyTyped(java.awt.event.KeyEvent e)
```

Specified by:

`keyTyped` in interface `java.awt.event.KeyListener`

keyReleased

```
public void keyReleased(java.awt.event.KeyEvent e)
```

Remove the acceleration from the RecycleBin when the arrow keys are released.

Specified by:

`keyReleased` in interface `java.awt.event.KeyListener`

Parameters:

`e` - The KeyEvent object

keyPressed

```
public void keyPressed(java.awt.event.KeyEvent e)
```

Gives the recycle bin acceleration on depression of the left or right arrow keys. Space pauses, and escape closes. The Windows key will start the game too, fitting in with the Windows XP look-and-feel.

Specified by:

`keyPressed` in interface `java.awt.event.KeyListener`

Parameters:

`e` - The KeyEvent object

loadSprites

```
private void loadSprites()
                  throws java.io.IOException
```

Loads all of the requisite images from the working directory, 7 in all.

Throws:

`java.io.IOException`

paintComponent

```
public void paintComponent(java.awt.Graphics g)
```

Draws the sprites of all of the GameObjects

Overrides:

`paintComponent` in class `javax.swing.JComponent`

Parameters:

`g` - Graphics context

getCpuUsage

```
public double getCpuUsage()
```

Accesses the cpuUsage variable. Used to update the metre in the HUD.

Returns:

The current CPU usage or 100, whichever is least

makeDialog

```
private void makeDialog()
```

Create a pop-up question. Called repeatedly.

drawTitleScreen

```
private void drawTitleScreen(java.awt.Graphics g)
```

Draws the instructive title screen.

Parameters:

`g` - The graphics context

increaseCpuUsage

```
public void increaseCpuUsage(double val)
```

Increases the CPU usage by some amount.

Parameters:

`val` - The amount by which to increase CPU usage.

endGame

```
public void endGame()
```

Routine for ending the game (showing the blue-screen).

decreaseCpuUsage

```
public void decreaseCpuUsage(double val)
```

Decreases the CPU usage by some amount.

Parameters:

`val` - The amount by which to increase CPU usage.

togglePaused

```
private void togglePaused()
```

Toggles the paused state of the game.

startGame

```
public void startGame()
```

Begins the game proper!

gameCycle

```
private void gameCycle()
```

What to do whilst the game is running. Called in the background loop thread. This method is strictly for things specific to each game. e.g. Collision detection which is universal does not go here. Creation of the junk items and popups does go here. The difficulty increases exponentially as the recycling bin collects more objects. Let n be the number of objects collected. Then the chance of a popup being created during a call of gameCycle is $(1 - 1.1^{-0.002n})$ in 1. That of a large sysfile being created is $(0.1 + (2)3^{-0.2(n+20)})$ in 1. For a medium sysfile, it's $(1 - 1.2^{-0.002n})$ in 1. For the smallest one, it's $(1 - 2^{-0.002n})$ in 1. Basically, smaller items are created more frequently later in the game, whilst the large item is created less frequently and eventually vanishes. Finally, junk items have a set frequency of 0.005 in 1, or about 1 in 200 iterations. All of these functions were chosen by experimentation.

isPaused

```
public boolean isPaused()
```

Whether the game is paused.

Returns:

Whether the game is paused.

isStarted

```
public boolean isStarted()
```

whether the game has started

Returns:

Whether the game has started.

isOver

```
public boolean isOver()
```

Whether the game is over

Returns:

Whether the game is over

drawGameOverScreen

```
private void drawGameOverScreen(java.awt.Graphics g)
```

Draws a BSOD with game information, signifying a game over.

Parameters:

`g` - Grahpics context

loadQuestions

```
private void loadQuestions()
                     throws java.io.IOException,
                            java.io.FileNotFoundException
```

Loads the questions from QuestionBank.txt. The format is Question Choice Choice Choice Choice And the answer will be marked with two hyphens ("--"). Deviation will cause an exception to be raised.

Throws:

`java.io.IOException`

`java.io.FileNotFoundException`