

SendHelp Snake 2.0

Generated by Doxygen 1.8.13

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	Board Class Reference	5
3.1.1	Constructor & Destructor Documentation	6
3.1.1.1	Board()	6
3.1.2	Member Function Documentation	6
3.1.2.1	actionPerformed()	6
3.1.2.2	checkCollision()	7
3.1.2.3	checkPellet()	7
3.1.2.4	loadImages()	7
3.1.2.5	locatePellet()	7
3.1.2.6	move()	8
3.1.2.7	paintComponent()	8
3.1.2.8	reset()	8
3.1.2.9	setDelay()	8
3.1.3	Member Data Documentation	9
3.1.3.1	ALL_DOTS	9
3.1.3.2	ball	9
3.1.3.3	BOARD_HEIGHT	9

3.1.3.4	BOARD_WIDTH	9
3.1.3.5	collision	9
3.1.3.6	collisionActive	9
3.1.3.7	collisionPellet	9
3.1.3.8	DELAY	10
3.1.3.9	DOT_SIZE	10
3.1.3.10	dots	10
3.1.3.11	downDirection	10
3.1.3.12	head	10
3.1.3.13	immunityCount	10
3.1.3.14	inGame	10
3.1.3.15	leftDirection	10
3.1.3.16	pellet	11
3.1.3.17	pellet_x	11
3.1.3.18	pellet_y	11
3.1.3.19	rand	11
3.1.3.20	RAND_POS	11
3.1.3.21	rightDirection	11
3.1.3.22	slowdown	11
3.1.3.23	slowdownPellet	11
3.1.3.24	speedup	12
3.1.3.25	speedupPellet	12
3.1.3.26	stepCount	12
3.1.3.27	timer	12
3.1.3.28	upDirection	12
3.1.3.29	x	12
3.1.3.30	y	12
3.2	Snake Class Reference	13
3.2.1	Constructor & Destructor Documentation	13
3.2.1.1	Snake()	13
3.2.2	Member Function Documentation	13
3.2.2.1	main()	13

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

ActionListener	
Board	5
JFrame	
Snake	13
JPanel	
Board	5

Chapter 2

Class Index

2.1 Class List

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

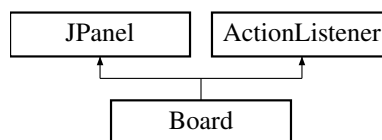
Board	5
Snake	13

Chapter 3

Class Documentation

3.1 Board Class Reference

Inheritance diagram for Board:



Public Member Functions

- `Board ()`
- `void setDelay (int delay)`
- `void loadImages ()`
- `void reset ()`
- `void paintComponent (Graphics g)`
- `void checkPellet ()`
- `void move ()`
- `void checkCollision ()` throws `InterruptedException`
- `void locatePellet ()`
- `void actionPerformed (ActionEvent e)`

Public Attributes

- `final int BOARD_WIDTH = 300`
- `final int BOARD_HEIGHT = 300`
- `final int DOT_SIZE = 10`
- `final int ALL_DOTS = 900`
- `final int RAND_POS = 29`
- `final int DELAY = 140`
- `final int x [] = new int[ALL_DOTS]`
- `final int y [] = new int[ALL_DOTS]`
- `int dots`
- `int pellet_x`

- int [pellet_y](#)
- Random [rand](#) = new Random()
- boolean [speedupPellet](#) = false
- boolean [slowdownPellet](#) = false
- boolean [collisionPellet](#) = false
- boolean [collisionActive](#) = false
- long [stepCount](#) = 0
- long [immunityCount](#) = 0
- boolean [leftDirection](#) = false
- boolean [rightDirection](#) = true
- boolean [upDirection](#) = false
- boolean [downDirection](#) = false
- boolean [inGame](#) = true
- Timer [timer](#)
- Image [ball](#)
- Image [pellet](#)
- Image [head](#)
- Image [speedup](#)
- Image [slowdown](#)
- Image [collision](#)

3.1.1 Constructor & Destructor Documentation

3.1.1.1 Board()

```
Board.Board ( )
```

Constructor for [Board](#).

```
out := self
```

3.1.2 Member Function Documentation

3.1.2.1 actionPerformed()

```
void Board.actionPerformed (
    ActionEvent e )
```

Called whenever an action occurs. If the game is currently running, checks if pellet was consumed by the snake, checks if collision has occurred and finally moves the snake. Repaints the scene afterwards.

Parameters

<i>e</i>	ActionEvent object
----------	--------------------

3.1.2.2 checkCollision()

```
void Board.checkCollision ( ) throws InterruptedException
```

Checks if snake collides with itself or the board's boundary. Typically ends game unless under collision immunity power-up.

```
transition := forall (z : int | z > 0 : z -= 1 and z > 4 and x[0] == x[z] and y[0] == y[z] and !this.collisionActive
and this.inGame = false); (this.y[0] >= this.BOARD_HEIGHT and this.inGame = false); (this.y[0] < 0 and this.inGame = false); (this.x[0] >= this.BOARD_WIDTH and this.inGame = false); (this.x[0] < 0 and this.inGame = false);
(!this.inGame and this.timer.stop());
```

Exceptions

<i>InterruptedException</i>	If thread's sleep is interrupted.
-----------------------------	-----------------------------------

3.1.2.3 checkPellet()

```
void Board.checkPellet ( )
```

If the snake's head touches a pellet, add to snake body count and applies power-up if applicable. Afterwards, locate the next pellet.

```
transition := (this.x[0] == this.pellet_x and this.y[0] == this.pellet_y) and (this.dots += 1 and (this.speedupPellet and this.setDelay(110) and speedupPellet = false) and (this.slowdownPellet and this.setDelay(170) and this.slowdownPellet = false) and (this.collisionPellet and this.collisionActive = true and this.collisionPellet = false))
```

3.1.2.4 loadImages()

```
void Board.loadImages ( )
```

Loads up the images for each significant component (body, head and pellets (power-up or otherwise))

```
transition := forall t : Image : t = t's corresponding image
```

```
exception := FileNotFoundException
```

3.1.2.5 locatePellet()

```
void Board.locatePellet ( )
```

Places next pellet randomly on the screen. (Either power-up or otherwise).

```
transition := (this.rand.nextInt(10) + 1 == 3 and this.speedupPellet = true) or (this.rand.nextInt(10) + 1 == 7 and this.slowdownPellet = true) or (this.rand.nextInt(10) + 1 == 4 and this.collisionPellet = true); this.pellet_x = Math.random() * this.RAND_POS * this.DOT_SIZE; this.pellet_y = Math.random() * this.RAND_POS * this.DOT_SIZE;
```

3.1.2.6 move()

```
void Board.move ( )
```

Moves the snake across the board. Also times out power-ups after an elapsed time.

```
transition := (this.timer.getDelay() != 140 and this.stepCount += 1); (this.stepCount > 35 and this.setDelay(140) and
this.stepCount = 0); (this.collisionActive and this.immunityCount += 1); (this.immunityCount > 35 and this.collisionActive = false and this.immunityCount = 0); forall z : int | z > 0 : z -= 1 and this.x[z] = x[(z-1)] and this.y[z] = y[(z-1)];
(this.leftDirection and x[0] -= this.DOT_SIZE); (this.rightDirection and x[0] += this.DOT_SIZE); (this.upDirection and y[0] -= DOT_SIZE); (this.downDirection and y[0] += this.DOT_SIZE)
```

3.1.2.7 paintComponent()

```
void Board.paintComponent (
    Graphics g )
```

Public access to drawing to the window the specified Graphics object.

```
out := forall t : Image : draw t
```

Parameters

<i>g</i>	Graphics object
----------	-----------------

3.1.2.8 reset()

```
void Board.reset ( )
```

Resets the game.

```
transition := this.dots = initialDotSize; forall z : int | z < this.dots : this.x[z] = initPosition - z * 10 and this.y[z] =
initPosition; this.rightDirection = true; this.leftDirection = false; this.upDirection = false; this.downDirection = false;
```

```
out := self
```

3.1.2.9 setDelay()

```
void Board.setDelay (
    int delay )
```

Sets the delay of the game, effectively controlling how quickly the game progresses.

```
transition := timer.setDelay(delay);
```

Parameters

<i>delay</i>	Delay of the game. The higher the number, the slower the game.
--------------	--

3.1.3 Member Data Documentation

3.1.3.1 ALL_DOTS

```
final int Board.ALL_DOTS = 900
```

However many dots on the board

3.1.3.2 ball

```
Image Board.ball
```

Image icon for snake body

3.1.3.3 BOARD_HEIGHT

```
final int Board.BOARD_HEIGHT = 300
```

Board height

3.1.3.4 BOARD_WIDTH

```
final int Board.BOARD_WIDTH = 300
```

Board width

3.1.3.5 collision

```
Image Board.collision
```

Image icon for collision immunity power-up pellet

3.1.3.6 collisionActive

```
boolean Board.collisionActive = false
```

If collision immunity power-up is in effect

3.1.3.7 collisionPellet

```
boolean Board.collisionPellet = false
```

Determines if current pellet is collision immunity power-up

3.1.3.8 DELAY

```
final int Board.DELAY = 140
```

Determines speed of game

3.1.3.9 DOT_SIZE

```
final int Board.DOT_SIZE = 10
```

Size of each dot or 'pixel'

3.1.3.10 dots

```
int Board.dots
```

Number of dots occupied by snake

3.1.3.11 downDirection

```
boolean Board.downDirection = false
```

Moves snake downwards

3.1.3.12 head

```
Image Board.head
```

Image icon for snake head

3.1.3.13 immunityCount

```
long Board.immunityCount = 0
```

Used to count duration of collision immunity power-up

3.1.3.14 inGame

```
boolean Board.inGame = true
```

If game is in process

3.1.3.15 leftDirection

```
boolean Board.leftDirection = false
```

Moves snake to the left

3.1.3.16 pellet

```
Image Board.pellet
```

Image icon for normal pellet

3.1.3.17 pellet_x

```
int Board.pellet_x
```

x-coordinate of pellet

3.1.3.18 pellet_y

```
int Board.pellet_y
```

y-coordinate of pellet

3.1.3.19 rand

```
Random Board.rand = new Random()
```

Used for calculating random power-up

3.1.3.20 RAND_POS

```
final int Board.RAND_POS = 29
```

Used to calculate next pellet position

3.1.3.21 rightDirection

```
boolean Board.rightDirection = true
```

Moves snake to the right

3.1.3.22 slowdown

```
Image Board.slowdown
```

Image icon for slow down power-up pellet

3.1.3.23 slowdownPellet

```
boolean Board.slowdownPellet = false
```

Determines if current pellet is slow down power-up

3.1.3.24 speedup

`Image Board.speedup`

Image icon for speed up power-up pellet

3.1.3.25 speedupPellet

`boolean Board.speedupPellet = false`

Determines if current pellet is speed up power-up

3.1.3.26 stepCount

`long Board.stepCount = 0`

Used to determine duration of speed up and slow down power-up

3.1.3.27 timer

`Timer Board.timer`

Timer (javax swing) object

3.1.3.28 upDirection

`boolean Board.upDirection = false`

Moves snake upwards

3.1.3.29 x

`final int Board.x[] = new int[ALL_DOTS]`

x-coordinates of board

3.1.3.30 y

`final int Board.y[] = new int[ALL_DOTS]`

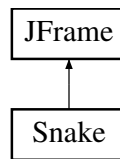
y-coordinates of board

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

- Board.java

3.2 Snake Class Reference

Inheritance diagram for Snake:



Public Member Functions

- [Snake](#) ()

Static Public Member Functions

- static void [main](#) (String[] args)

3.2.1 Constructor & Destructor Documentation

3.2.1.1 Snake()

```
Snake.Snake ( )
```

Constructor for [Snake](#). Sets up board, title of window, etc.

3.2.2 Member Function Documentation

3.2.2.1 main()

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

Main method to run the game.

Parameters

<i>args</i>	
-------------	--

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

- Snake.java

Index

ALL_DOTS
 Board, 9
actionPerformed
 Board, 6

BOARD_HEIGHT
 Board, 9
BOARD_WIDTH
 Board, 9
ball
 Board, 9
Board, 5
 ALL_DOTS, 9
 actionPerformed, 6
 BOARD_HEIGHT, 9
 BOARD_WIDTH, 9
 ball, 9
 Board, 6
 checkCollision, 7
 checkPellet, 7
 collision, 9
 collisionActive, 9
 collisionPellet, 9
 DELAY, 9
 DOT_SIZE, 10
 dots, 10
 downDirection, 10
 head, 10
 immunityCount, 10
 inGame, 10
 leftDirection, 10
 loadImages, 7
 locatePellet, 7
 move, 7
 paintComponent, 8
 pellet, 10
 pellet_x, 11
 pellet_y, 11
 RAND_POS, 11
 rand, 11
 reset, 8
 rightDirection, 11
 setDelay, 8
 slowdown, 11
 slowdownPellet, 11
 speedup, 11
 speedupPellet, 12
 stepCount, 12
 timer, 12
 upDirection, 12
 x, 12
 y, 12

checkCollision
 Board, 7
checkPellet
 Board, 7
collision
 Board, 9
collisionActive
 Board, 9
collisionPellet
 Board, 9

DELAY
 Board, 9
DOT_SIZE
 Board, 10
dots
 Board, 10
downDirection
 Board, 10

head
 Board, 10

immunityCount
 Board, 10
inGame
 Board, 10

leftDirection
 Board, 10
loadImages
 Board, 7
locatePellet
 Board, 7

main
 Snake, 13
move
 Board, 7

paintComponent
 Board, 8
pellet
 Board, 10
pellet_x
 Board, 11
pellet_y
 Board, 11

RAND_POS
 Board, [11](#)
rand
 Board, [11](#)
reset
 Board, [8](#)
rightDirection
 Board, [11](#)

setDelay
 Board, [8](#)
slowdown
 Board, [11](#)
slowdownPellet
 Board, [11](#)
Snake, [13](#)
 main, [13](#)
 Snake, [13](#)
speedup
 Board, [11](#)
speedupPellet
 Board, [12](#)
stepCount
 Board, [12](#)

timer
 Board, [12](#)

upDirection
 Board, [12](#)

x
 Board, [12](#)

y
 Board, [12](#)