**Department of Information Technology**

**National Institute of Technology Karnataka, Surathkal**

****

Paradigms of Programming II (IT253)

Mini Project

**Title**- Picture Puzzle Game in Java

**Submitted by**

Supriya Sahoo 15IT145

Akriti Kumari 15IT107

Ankita Bhalavi 15IT108

Vandana Baidya 15IT250

**Submitted to**

Ms. Tulasi Gayatri Devi

(Course Instructor)

April 2017

**CERTIFICATE**

This is to certify that the project entitled **“**Picture Puzzle Game in Java**”** has been presented by Supriya Sahoo, Akriti Kumari, Ankita Bhalavi, Vandana Baidya, students of second year, B.Tech (IT), Department of Information Technology, National Institute of Technology Karnataka, Surathkal, on April \_2017, during the even semester of the academic year 2016- 2017, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology at NITK, Surathkal.

Place: NITK, Surathkal

Date: (Signature of the Examiner)

# INDEX

**Topic Page No.**

1. Introduction……………………………………………….5

2. Objective………………………………………………….5

3. Description………………………………………………..5

4. Software and Hardware requirement specification……….6

4.1. Software specification……………………………….6

4.2. Hardware specification………………………………6

5. System Code………………………………………………7

6. Code Explanation…………………………………………15

7. Results…………………………………………………......16

8. Discussion………………………………………………....19

9. Reference…………………………………………………..19

**ACKNOWLEDGEMENT**

This project would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them. We are highly indebted to the Department of Information Technology, NITK for their guidance and constant supervision as well as for providing necessary information and resources for the project and also for their support in completing the project. We would like to express our gratitude towards our teacher Ms. Tulasi Gayatri Devi for her kind co-operation and encouragement which helped us in completion of this project. Our thanks and appreciations also go to our group members in developing the project and people who have willingly helped us out with their abilities.

# 1. Introduction

Picture puzzle project is a game application which is implemented on java platform. In this project we are trying to create a single player puzzle game using java applets, GUI and swing packages. Pictures here are arranged in a haphazard manner throughout a grid and by tapping upon the pictures there positions can be swapped with the position of the adjacent pictures .With this tapping and swapping of the picture icons finally the required picture is obtained.

# 2. Objective

The main theme behind developing puzzle game using java is to provide a creative and competitive environment for the players who will use this system. It’s a game where two modules has been provided over a single platform. Here the users have the choice to select the game as per their choice and play at free of cost with full features. Some extensive graphical interface has been used to give excellent picture quality and its optimized code will help users to get processing work faster without any delay.

# 3. Description

The goal of this game is to arrange a given set of pictures in a grid into a final picture. Buttons containing images are moved by clicking on them. Only buttons adjacent to the empty button can be moved. First the image is scaled and cut into nine pieces. These pieces are used by *JButton* components. The last piece is not used; we have empty button instead. There is one button that we call the last button, which does not have an image where other buttons swap space with the help of this.

The image that we use to form is scaled to have desired width. Then we calculate the new height keeping the image-ratio. The solution array specialist stores the correct order of buttons which forms the image. Each button is identified by one point. The layout consists of 3 rows and 3 columns. The image is cut into a square shapes from the already resized image source. There is a point containing the button's correct row and column position in the picture.

These properties are used to find out if we have the correct order of buttons in the window. The button with no image is called the last button; it is placed at the end of the grid in the bottom-right corner. It is the button that swaps its position with the adjacent button that is being clicked. We randomly reorder the elements of the buttons list. The last button, i.e. the button with no image, is inserted at the end of the list. It is not supposed to be shuffled; it always goes at the end when we start the Puzzle game. Buttons are stored in an array list. This list is then mapped to the grid of the panel. We get the indexes of the last button and the clicked button and they are swapped.

Solution checking is done by comparing the list of points of the correctly ordered buttons with the current list containing the order of buttons from the window.

# 4. Software & Hardware Requirement specification

## 4.1. Hardware Specification

Hardware used during the development process. The configuration of the system, which was, used for the development process such coding is given as below:

|  |  |  |
| --- | --- | --- |
| S.No | Component | Description |
| 1. | Processor | Intel Dual-Core, 2.40 GHz |
| 2. | Cache | 128 KB |
| 3. | RAM | 4.00 GB |
| 4. | Hard Disk | 80 GB |

## 4.2. Software Specification

Operating System : Any Operating System

Language used : JAVA

IDE Tools : Eclipse, Netbeans

# 5. System Code

**import** java.awt.event.\*;

**import** java.awt.\*;

**import** javax.swing.\*;

**class** picpuzzle2 **extends** JFrame **implements** ActionListener{

JButton b1,b2,b3,b4,b5,b6,b7,b8,b9,sample,starB;

Icon star;

Icon ic0=**new** ImageIcon("pic/starB0.jpg");

Icon ic10=**new** ImageIcon("pic/starB10.jpg");

Icon ic20=**new** ImageIcon("pic/starB20.jpg");

Icon samicon1=**new** ImageIcon("pic/main.jpg");

Icon samicon2=**new** ImageIcon("pic/main2.jpg");

Icon samicon3=**new** ImageIcon("pic/main3.jpg");

Icon ic1=**new** ImageIcon("pic/1.jpg");

Icon ic2=**new** ImageIcon("pic/5.jpg");

Icon ic3=**new** ImageIcon("pic/2.jpg");

Icon ic4=**new** ImageIcon("pic/7.jpg");

Icon ic5=**new** ImageIcon("pic/4.jpg");

Icon ic6=**new** ImageIcon("pic/6.jpg");

Icon ic7=**new** ImageIcon("pic/8.jpg");

Icon ic8=**new** ImageIcon("pic/9.jpg");

Icon ic9=**new** ImageIcon("pic/3.jpg");

Icon ic11=**new** ImageIcon("pic/12.jpg");

Icon ic12=**new** ImageIcon("pic/13.jpg");

Icon ic13=**new** ImageIcon("pic/16.jpg");

Icon ic14=**new** ImageIcon("pic/11.jpg");

Icon ic15=**new** ImageIcon("pic/14.jpg");

Icon ic16=**new** ImageIcon("pic/19.jpg");

Icon ic17=**new** ImageIcon("pic/17.jpg");

Icon ic18=**new** ImageIcon("pic/15.jpg");

Icon ic19=**new** ImageIcon("pic/18.jpg");

Icon ic21=**new** ImageIcon("pic/24.jpg");

Icon ic22=**new** ImageIcon("pic/25.jpg");

Icon ic23=**new** ImageIcon("pic/21.jpg");

Icon ic24=**new** ImageIcon("pic/27.jpg");

Icon ic25=**new** ImageIcon("pic/23.jpg");

Icon ic26=**new** ImageIcon("pic/29.jpg");

Icon ic27=**new** ImageIcon("pic/28.jpg");

Icon ic28=**new** ImageIcon("pic/22.jpg");

Icon ic29=**new** ImageIcon("pic/26.jpg");

picpuzzle2(){

**super**("pic puzzle");

b1=**new** JButton(ic1);

b1.setBounds(10,80,100,100);   ***// setting button position***

b2=**new** JButton(ic2);

b2.setBounds(110,80,100,100);

b3=**new** JButton(ic3);

b3.setBounds(210,80,100,100);

b4=**new** JButton(ic4);

b4.setBounds(10,180,100,100);

b5=**new** JButton(ic5);

b5.setBounds(110,180,100,100);

b6=**new** JButton(ic6);

b6.setBounds(210,180,100,100);

b7=**new** JButton(ic7);

b7.setBounds(10,280,100,100);

b8=**new** JButton(ic8);

b8.setBounds(110,280,100,100);

b9=**new** JButton(ic9);

b9.setBounds(210,280,100,100);

sample=**new** JButton(samicon1);

sample.setBounds(380,100,200,200);

JLabel l1=**new** JLabel("Sample:");

l1.setBounds(330,200,70,20);

JLabel l2=**new** JLabel("NOTE:

icon has power to swap with neighbour icon=");

l2.setBounds(5,15,500,20);

JLabel l3=**new** JLabel("click sample picture to next puzzle");

l3.setBounds(380,320,200,20);

l3.setForeground(Color.red);

starB=**new** JButton(ic0);

starB.setBounds(330,5,50,50);

star=b9.getIcon();

add(b1);add(b2);add(b3);add(b4);add(b5);add(b6);add(b7);add(b8);

add(b9);add(sample);add(l1);add(l2);add(starB);add(l3);   ***// adding button into frame***

b1.addActionListener(**this**); b2.addActionListener(**this**);

 b3.addActionListener(**this**); b4.addActionListener(**this**);

b5.addActionListener(**this**); b6.addActionListener(**this**);

 b7.addActionListener(**this**); b8.addActionListener(**this**);

 b9.addActionListener(**this**);

sample.addActionListener(**this**);

setLayout(**null**);   ***// no layout manager***

setSize(600,500);   ***// frame size 600 width and 500 height***

setVisible(**true**);   ***// now frame will be visible, by default not visible***

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

**public** **void** actionPerformed(ActionEvent e){

**if**(e.getSource()==b1){

    Icon s1=b1.getIcon();

**if**(b2.getIcon()==star){

        b2.setIcon(s1);

        b1.setIcon(star);

      } **else** **if**(b4.getIcon()==star){

                    b4.setIcon(s1);

                    b1.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b2){

    Icon s1=b2.getIcon();

**if**(b1.getIcon()==star){

        b1.setIcon(s1);

        b2.setIcon(star);

      } **else** **if**(b5.getIcon()==star){

                    b5.setIcon(s1);

                    b2.setIcon(star);

                   }

**else** **if**(b3.getIcon()==star){

                    b3.setIcon(s1);

                    b2.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b3){

    Icon s1=b3.getIcon();

**if**(b2.getIcon()==star){

        b2.setIcon(s1);

        b3.setIcon(star);

      } **else** **if**(b6.getIcon()==star){

                    b6.setIcon(s1);

                    b3.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b4){

    Icon s1=b4.getIcon();

**if**(b1.getIcon()==star){

        b1.setIcon(s1);

        b4.setIcon(star);

      } **else** **if**(b5.getIcon()==star){

                    b5.setIcon(s1);

                    b4.setIcon(star);

                   }

**else** **if**(b7.getIcon()==star){

                    b7.setIcon(s1);

                    b4.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b5){

    Icon s1=b5.getIcon();

**if**(b2.getIcon()==star){

        b2.setIcon(s1);

        b5.setIcon(star);

      } **else** **if**(b4.getIcon()==star){

                    b4.setIcon(s1);

                    b5.setIcon(star);

                   }

**else** **if**(b6.getIcon()==star){

                    b6.setIcon(s1);

                    b5.setIcon(star);

                   }

**else** **if**(b8.getIcon()==star){

                    b8.setIcon(s1);

                    b5.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b6){

    Icon s1=b6.getIcon();

**if**(b3.getIcon()==star){

        b3.setIcon(s1);

        b6.setIcon(star);

      } **else** **if**(b5.getIcon()==star){

                    b5.setIcon(s1);

                    b6.setIcon(star);

                   }

**else** **if**(b9.getIcon()==star){

                    b9.setIcon(s1);

                    b6.setIcon(star);

                   }

}//end of if

**if**(e.getSource()==b7){

    Icon s1=b7.getIcon();

**if**(b4.getIcon()==star){

        b4.setIcon(s1);

        b7.setIcon(star);

      } **else** **if**(b8.getIcon()==star){

                    b8.setIcon(s1);

                    b7.setIcon(star);

                   }

 }//end of if

**if**(e.getSource()==b8){

    Icon s1=b8.getIcon();

**if**(b7.getIcon()==star){

        b7.setIcon(s1);

        b8.setIcon(star);

      } **else** **if**(b5.getIcon()==star){

                    b5.setIcon(s1);

                    b8.setIcon(star);

                   }

**else** **if**(b9.getIcon()==star){

                    b9.setIcon(s1);

                    b8.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==b9){

    Icon s1=b9.getIcon();

**if**(b8.getIcon()==star){

        b8.setIcon(s1);

        b9.setIcon(star);

      } **else** **if**(b6.getIcon()==star){

                    b6.setIcon(s1);

                    b9.setIcon(star);

                   }

  }//end of if

**if**(e.getSource()==sample){

Icon s1=sample.getIcon();

**if**(s1==samicon3){

sample.setIcon(samicon1);

b1.setIcon(ic1);

b2.setIcon(ic2);

b3.setIcon(ic3);

b4.setIcon(ic4);

b5.setIcon(ic5);

b6.setIcon(ic6);

b7.setIcon(ic7);

b8.setIcon(ic8);

b9.setIcon(ic9);

star=b9.getIcon();

starB.setIcon(ic0);

}//end of if

**else** **if**(s1==samicon1){

sample.setIcon(samicon2);

b1.setIcon(ic11);

b2.setIcon(ic12);

b3.setIcon(ic13);

b4.setIcon(ic14);

b5.setIcon(ic15);

b6.setIcon(ic16);

b7.setIcon(ic17);

b8.setIcon(ic18);

b9.setIcon(ic19);

star=b6.getIcon();

starB.setIcon(ic10);

}//end of else

**else**{

sample.setIcon(samicon3);

b1.setIcon(ic21);

b2.setIcon(ic22);

b3.setIcon(ic23);

b4.setIcon(ic24);

b5.setIcon(ic25);

b6.setIcon(ic26);

b7.setIcon(ic27);

b8.setIcon(ic28);

b9.setIcon(ic29);

star=b6.getIcon();

starB.setIcon(ic20);

}//end of else

}

}//end of actionPerformed

**public** **static** **void** main(String args[]){

**new** picpuzzle2();

}//end of main

}//end of class

# 6. Code Explanation:

This project focuses on how to make use of some basic Java Development Kit (JDK) classes and image-manipulation techniques to create a simple picture puzzle game. It is intended to give us a taste of how fun programming can be.

The game needs to be able to take an image and properly resize the game panel. The next step will be slicing the image into puzzle pieces. Making customizable the number of rows and columns for the puzzle will be useful. Now these pieces need to be randomly scrambled for the game. Here, we used a simple mouse click in this demonstration.

In the above code, the event handler class is JFrame which implements ActionListener. As we would like to handle the button-click event, so we added an action listener to the buttons with the help of **JButton()** method which creates a JButton instance, initializing it to have the specified icon/text/image/action, here we considered icon. The button is positioned by using **setbounds()** method.

**setbounds( int x, int y, int width, int height)** – sets the buttons size and location.

**setForeground()** – used to set foreground color of an applet window.

**void add([AbstractButton](https://docs.oracle.com/javase/8/docs/api/javax/swing/AbstractButton.html" \o "class in javax.swing) b)** – adds the button, where b is the button to be added

**void setIcon(Icon b)** – used to set the specified icon on the button.

**setLayout(null)** - To remove a layout manager, set the layout manager to null. "null layout manager" is also known as absolute positioning.

The size of a frame is defined by its width and height in pixels and we can set them using **setSize(int width, int height)** method.

**setVisible( boolean )** controls whether a Component and its children are displayed on the screen. setVisible( false ) hides a Component by marking it as invisible. setVisible( true ) reveals a Component by marking it as visible.

**setDefaultCloseOperation()** method is used to specify one of several options for the close button.

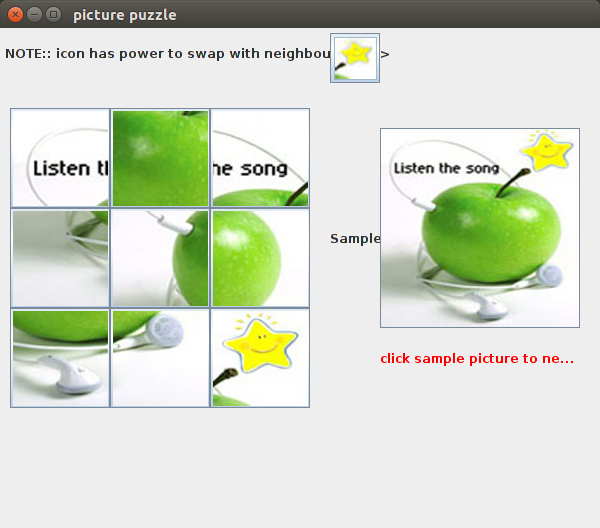
**void actionPerformed([ActionEvent](https://docs.oracle.com/javase/8/docs/api/java/awt/event/ActionEvent.html" \o "class in java.awt.event) e)** - Invoked when an action occurs.

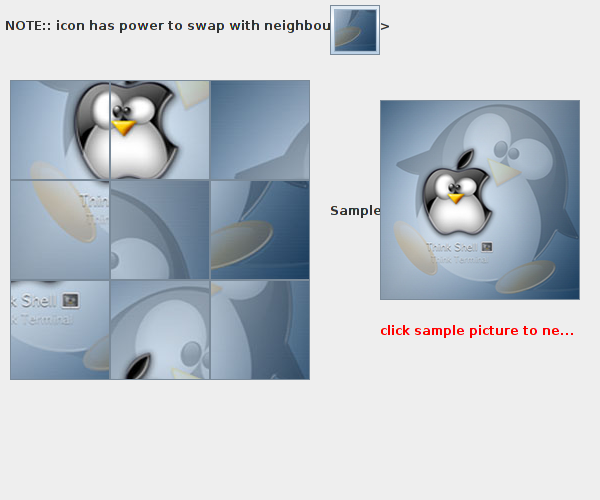
**ActionEvent.getSource()** - The EventObject contains the getSource( ) method. Suppose you have many buttons in your application. So, you can find which button is clicked by, using the getSource() method. The getSource( ) method returns the source of the event.

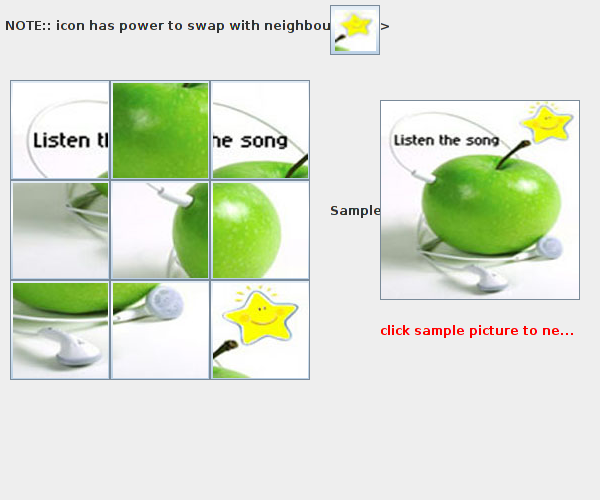
**void getIcon(Icon b)** – used to get the specified icon on the button.

**void setIcon(Icon b)** - used to set the specified Icon on the button.

# 7. Results







# 8. Discussion

* The main goal of the project was to provide a creative and competitive environment to the players.
* The system can also be upgraded into more complex system.
* The interface of the application can be improved to make it become friendlier.
* More algorithms can be implemented in puzzle game to make it more interesting.

# 9. References

1. <https://en.wikipedia.org/wiki/Swing_(Java)>

2. <https://en.wikipedia.org/wiki/Abstract_Window_Toolkit>

3. <https://docs.oracle.com/javase/tutorial/uiswing/components/frame.html>

4. <https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html>

5. <http://www.wideskills.com/java-tutorial/java-frame-class-example>