BlueJ / Java Cheatsheet

by Juraj Masar

Imports

```
import java.io.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.JLabel;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.Color;
import java.awt.Font;
import javax.swing.Timer;
```

Switch statement

Break must be used! Otherwise following code will be executed as well.

```
switch (dices[i].getValue())
{
   case 1: diods[i].setBackground (Color.RED); break;
   case 2: diods[i].setBackground (Color.BLUE); break;
   default: diods[i].setBackground (Color.WHITE); break;
}
```

Working with an array

```
int[] p;
p = new int[3];
for (int i=0;i<=p.length;i++) p[i]=1;
```

for objects:

```
JButton[] buttons;
buttons = new JButton[6]; //we want to allocate memory of 6 buttons
```

```
buttons[4] = new JButton(); //create particular instance for array element
```

Printing out strings (text output)

```
System.out.print(“string”);
System.out.println(“string”);
```

Ending program when JFrame closes

```
Frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
```

Or normally as a command `System.exit(0);`

Converting variable types

char to String

```
char c = 'A';
String s;
s = String.valueOf(c);
```

String to int

```
int i;
String s;
s = “123.12”;
i = Integer.parseInt(s);
```

String to double

```
double d;
String s;
s = “123”;
d = Double.parseDouble(s);
```

JFrame

```
frame = new JFrame("Mickey The Tester");
frame.setLayout(null);
frame.setSize(540,580);
frame.setVisible(true);
frame.setResizable(false);
```

JLabel

```
label = new JLabel();
label.setText("init text");
label.setHorizontalAlignment(JLabel.LEFT);
//label.setSize(40,570);
//label.setLocation(0,260);
label.setBounds (0,260,540,40);

//assign label to JFrame
frame.getContentPane().add(label);

label.setVisible (true);
label.setForeground(Color.blue);
label.setFont(new Font("Dialog", Font.PLAIN, 24));
label.setText ("ANSWER: "+input);
```

Using button + catching events

```
public class Application implements ActionListener
{
    private JButton btn;
    private JFrame f;
    public void actionPerformed (ActionEvent e)
    {
        if (e.getSource() == btn)
        {
           //btn was pressed, do some action
        }
     }
     public static void main ()
     {
           f = new Jframe();
           f.setSize(100,100);
```

```
        //creating instance of JButton object
         btn = new JButton("caption");
        btn.setSize(530,70);
        btn.setLocation(0,480);
        frame.getContentPane().add(btn);
        btn.setVisible(true);

        //important part: setting instance of ActionListener
        btn.addActionListener(this);
    }
```

Writing to text files

```
try
{
    pen = new PrintWriter(new FileOutputStream("filename.txt"));
    pen.print("some string");
    pen.println("another string");
    pen.flush(); //data are flushed to the file from the stream
    pen.close();
} catch (FileNotFoundException e)
{
    System.out.println(e.getMessage());
}
```

Reading from text file

```
try
{
    glasses = new BufferedReader (new FileReader ("filename.txt"));
    String line;
    while ((line = glasses.readLine()) != null)
    {
```

```
     }
} catch (FileNotFoundException e) //or IOExpection
{
     System.out.println(e.getMessage());
}
```

Executing modal error dialog

```
JOptionPane.showMessageDialog(this, “The content of the window”, "Title of the window", JOptionPane.ERROR_MESSAGE);
```

this – instance of Jframe

JoptionPane.ERROR_MESSAGE – type of dialog

Throwing an exception

```
public static boolean checkInterval(double left, double right, double check) throws NumberFormatException
    //if it is false, than it throws NFE
    {
        if ((check < left) || (check > right)) throw new NumberFormatException("<html>You entered either too small or too high number.");
        return (check >= left) && (check <= right);
    }
```

Generating random numbers

```
    public static int random (int min, int max)
    {
        if (min>max)
        {
          min = min + max;
          max = min - max;
          min = min - max;
        }
```

```
        return (int)(Math.random()*(max-min+1))+min;
    }
```

Math.random() - returns value in type of double from 0 to 1; 1 excluding

Getting list of files in a directory

Object variable:

```
private static final String soundsdir = System.getProperty("user.dir")
+"/words/";
```

final – constant which can't be changed in future inherited classes

```
File f = new File (soundsdir);
files = f.list();

if (files == null)
{
     System.out.println("Error - No files present");
     System.exit(0); //program ends
}
```

Using a Timer

```
Timer t; //preferably defined as object variable
t = new Timer (100,this);//interval in ms, action listener
t.start();
t.stop();
```

Calling parent's constructor

```
public class HahaDice extends Dice
{
    private int x;
    public HahaDice()
    {
```

```
        // initialize instance variables
        super();
    }
    public HahaDice (int size)
    {
        super(size);
    }
}
```

Random Access File

```
import java.io.*;
class DirectAccessFile
{
     RandomAccessFile f;
     long recordLength;
     public DirectAccessFile (String fileName, long recordLength)
     {
          try
          {
               f = new RandomAccessFile (fileName, “rw”);
               this.recordLength = recordLength;
          }
          catch (IOException e)
          {
               System.out.println (e.getMessage());
          }
     }
     public long lengthInBytes()
     {
          //use of try required
          return f.length();
     }
```

```
	public long numberOfRecords ()
	{
		//use of try required
		return f.length()/recordLength;
	}
	public long goToRecordNum (long num)
	{
		if (num >= 0 && num <= numOfRecords)
			f.seek (num*recordLength);
	}
	public void writeString (String s)
	{
		f.writeUTF(s);
	}
}
```

Bubble sort

```
procedure bubbleSort( A : list of sortable items ) defined as:
  do
    swapped := false
    for each i in 0 to length(A) - 2 inclusive do:
      if A[i] > A[i+1] then
        swap( A[i], A[i+1] )
        swapped := true
      end if
    end for
  while swapped
end procedure
```