

Notes from Well House Consultants

These notes are written by Well House Consultants and distributed under their Open Training Notes License. If a copy of this license is not supplied at the end of these notes, please visit

*<http://www.wellho.net/net/whcotnl.html>
for details.*

1.1 Well House Consultants

Well House Consultants provides niche training, primarily but not exclusively in Open Source programming languages. We offer public courses at our training centre and private courses at your offices. We also make some of our training notes available under our "Open Training Notes" license, such as we're doing in this document here.

1.2 Open Training Notes License

With an "Open Training Notes License", for which we make no charge, you're allowed to print, use and distribute these notes provided that you retain the complete and unaltered license agreement with them, including our copyright statement. This means that you can learn from the notes, and have others learn from them too.

You are NOT allowed to charge (directly or indirectly) for the copying or distribution of these notes, nor are you allowed to charge for presentations making any use of them.

1.3 Courses presented by the author

If you would like us to attend a course (Java, Perl, Python, PHP, Tcl/Tk, MySQL or Linux) presented by the author of these notes, please see our public course schedule at

<http://www.wellho.net/course/index.html>

If you have a group of 4 or more trainees who require the same course at the same time, it will cost you less to have us run a private course for you. Please visit our onsite training page at

<http://www.wellho.net/course/otc.html>

which will give you details and costing information

1.4 Contact Details

Well House Consultants may be found online at

<http://www.wellho.net>

graham@wellho.net

technical contact

lisa@wellho.net

administration contact

Our full postal address is

404 The Spa
Melksham
Wiltshire
UK SN12 6QL

Phone +44 (0) 1225 708225

Fax +44 (0) 1225 707126

Serialization

The information held in an object can be spread all around the data area of your Java program. If you want to save an object to a file, or transmit it over a connection to another computer, you'll need to be able to pick out the salient information and assemble it into a series of bytes, a process known as "serialization".

Saving objects to a file. 4

2.1 Saving objects to a file

An object needs to be written out as a series of bytes – a sequence or a serial stream. We achieve this by declaring the object class **Serializable**.

We then use an **ObjectOutputStream** to write the objects out, and an **ObjectInputStream** to read the objects back in.

writeObject and **readObject** methods are used for the actual writes and reads.

Let's put together a lot of the topics we have already studied on this course.

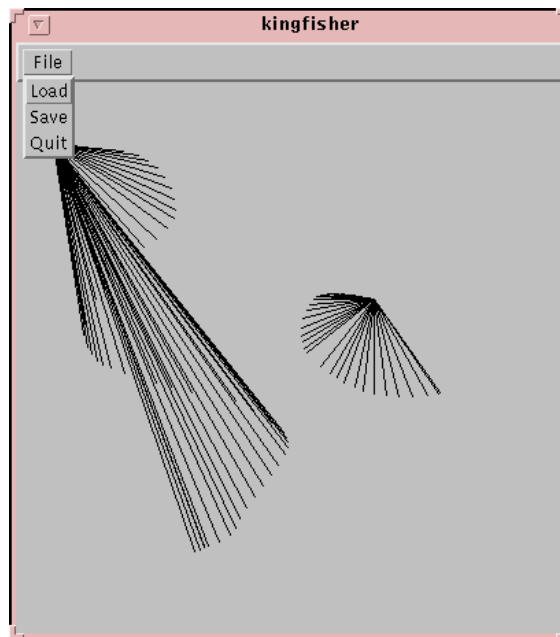


Figure 1 Running public course kingfisher on the AppletViewer

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.io.*;

public class kingfisher extends Frame {
    protected Drawing sketch;
    Point corner;

    public static void main(String [] args)
    {
        new kingfisher();
    }
    public kingfisher()
    {
        super("kingfisher");
        // final Drawing sketch;
        sketch = new Drawing(this, 400, 400);
        this.add(sketch, "Center");

        MenuBar menu = new MenuBar();
        this.setMenuBar(menu);
        Menu disc_actions = new Menu("File");
        menu.add(disc_actions);

        MenuItem load, save, finito;
        disc_actions.add(load = new MenuItem("Load"));
```

```

disc_actions.add(save = new MenuItem("Save"));
disc_actions.add(finito = new MenuItem("Quit"));

load.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        sketch.load();
    }
});

save.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        sketch.save();
    }
});

finito.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
});

MouseMotionListener happening =
    new mouse_handler(this);
sketch.addMouseMotionListener(happening);

this.pack();
this.show();
corner = sketch.getOffset();
}

////////////////////////////////////

class mouse_handler extends MouseMotionAdapter{

private kingfisher cyan_flash;
protected short x,y,xfrom,yfrom;

public mouse_handler(kingfisher george)
    { cyan_flash = george;}

public void mouseDragged(MouseEvent e) {
    // Graphics g = cyan_flash.getGraphics();
    Graphics g = getGraphics();
    x = (short) (corner.x + e.getX());
    y = (short) (corner.y + e.getY());
    g.drawLine(x,y,xfrom,yfrom);
    sketch.extend(x,y,xfrom,yfrom);
}

public void mouseMoved(MouseEvent e) {
    xfrom = (short) (corner.x + e.getX());
    yfrom = (short) (corner.y + e.getY());
}

}

////////////////////////////////////

```

```

static class Drawing extends Component {
    protected int wide,high;
    protected Frame box;
    protected Vector beads = new Vector();

    public Drawing(Frame box, int wide, int high) {
        this.box = box;
        this.wide = wide;
        this.high = high;
    }

    public Dimension getPreferredSize()
    {
        return new Dimension(wide,high);
    }

    public Point getOffset()
    {
        return box.getLocation();
    }

    public void extend(short x1, short y1,
                       short x2, short y2)
    {
        beads.addElement(new Bead(x1,y1,x2,y2));
    }

    public void paint(Graphics g)
    {
        for (int jp=0; jp<beads.size(); jp++)
        {
            Bead current = (Bead)beads.elementAt(jp);
            g.drawLine(current.x1, current.y1,
                       current.x2, current.y2);
        }
    }

    public void save(){
        String filename = "demo.data";
        try {
            FileOutputStream f_o_s =
                new FileOutputStream(filename);
            ObjectOutputStream o_o_s =
                new ObjectOutputStream(f_o_s);
            o_o_s.writeObject(beads);
            o_o_s.flush();
            o_o_s.close();
        } catch (Exception e){
            System.out.println(e);
        }
    }

    public void load(){
        String filename = "demo.data";
        try {
            FileInputStream f_i_s =
                new FileInputStream(filename);
            ObjectInputStream o_i_s =
                new ObjectInputStream(f_i_s);

```

```

        Vector incoming = (Vector)o_i_s.readObject();
        o_i_s.close();
        beads = incoming;
        repaint();
    } catch (Exception e){
        System.out.println(e);
    }
}
}

////////////////////////////////////

static class Bead implements Serializable {
    public short x1,x2,y1,y2;

    public Bead(short x1, short y1, short x2, short y2)
    {
        this.x1=x1; this.y1=y1;
        this.x2=x2; this.y2=y2;
    }
}
}

```

Quite a bit of work is done internally in serialisation. How can we be sure that a file saved in one program or version is read back with a compatible program?

When an object is serialized, some information about its class is stored with it, together with a version number ID.

You can declare this version number if you wish, within the class. Something like:

```
static final long serialVersionUID = 34877176437199L;
```

If you choose not to declare this constant, then the **ObjectOutputStream** will compute one for you based on the name, class, interfaces, fields and methods (except private methods) of the class.

Thus any change to the class will result in a new **serialVersionUID** and old data will throw an exception if you try to read it.

There may be variables within a class which you do not want serialized. For example, temporary variables that you would rather compute each time the data is reloaded, and platform-specific items such as file descriptors.

If you do not wish a variable or object to be serialized, you should declare it as **transient**.

```
protected transient int one, x, c;
```

 **Exercise**

License

*These notes are distributed under the **Well House Consultants Open Training Notes License**. Basically, if you distribute it and use it for free, we'll let you have it for free. If you charge for its distribution of use, we'll charge.*

3.1 Open Training Notes License

Training notes distributed under the **Well House Consultants Open Training Notes License** (WHCOTNL) may be reproduced for any purpose PROVIDE THAT:

- This License statement is retained, unaltered (save for additions to the change log) and complete.
- No charge is made for the distribution, nor for the use or application thereof. This means that you can use them to run training sessions or as support material for those sessions, but you cannot then make a charge for those training sessions.
- Alterations to the content of the document are clearly marked as being such, and a log of amendments is added below this notice.
- These notes are provided "as is" with no warranty of fitness for purpose. Whilst every attempt has been made to ensure their accuracy, no liability can be accepted for any errors of the consequences thereof.

Copyright is retained by Well House Consultants Ltd, of 404, The Spa, Melksham, Wiltshire, UK, SN12 6QL - phone number +44 (1) 1225 708225. Email contact - Graham Ellis (graham@wellho.net).

Please send any amendments and corrections to these notes to the Copyright holder - under the spirit of the Open Distribution license, we will incorporate suitable changes into future releases for the use of the community.

If you are charged for this material, or for presentation of a course (Other than by Well House Consultants) using this material, please let us know. It is a violation of the license under which this notes are distributed for such a charge to be made, except by the Copyright Holder.

If you would like Well House Consultants to use this material to present a training course for your organisation, or if you wish to attend a public course is one is available, please contact us or see our web site - <http://www.wellho.net> - for further details.

Change log
Original Version, Well House Consultants, 2004

Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____

License Ends.