

Section 12: Applications

1) Applications

- a) free standing programs
- b) run compiled code with "java" interpreter

2) applications that use text io:

- a) have a main (like c++), which is done first, program over when main ends
- b) generally uses class System to invoke stuff

example:

```
public class chp18 {  
  
    public static void main(String [] args)  
    {  
        System.out.println("Blah!");  
    }  
  
}
```

Compilation and run:

```
$ javac chp18.java  
$ java chp18  
Blah!  
$
```

3) applications

- a) init method becomes constructor and remove void
- b) extend Frame to use window
- c) need public static void main(String[] args) and in main:
 - declare an object of the class
 - object.setSize(x,y);
 - object.setVisible(true);
- d) add methods windowClosing (must implements WindowListener)
- e) add this.addWindowListener(this); to register event handler
- f) setLayout(new FlowLayout());

4) applications that use AWT:

- a) extends class Frame
- b) main() acts like init()
- c) an object of the class is declared and the function(s) necessary to run the AWT application are called.

example:

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

public class chp18a extends Frame implements ActionListener,
WindowListener {

    private Button left, right;
    private Boxie box;

    public static void main(String [] args)
    {
        chp18a example = new chp18a();

        example.setSize(300,300);
        example.setVisible(true);

    }

    public chp18a() {
        box = new Boxie(100,100,10,10,Color.red);

        setTitle("Java Standalone with AWT");
        setLayout(new FlowLayout());

        left = new Button("Left");
        add(left);
        left.addActionListener(this);

        right = new Button("Right");
        add(right);
        right.addActionListener(this);

        this.addWindowListener(this);
    }

    public void actionPerformed(ActionEvent event) {
        if (event.getSource() == left)
            box.Left(5);
        if (event.getSource() == right)
            box.Right(5);
        repaint();
    }
}
```

```
    }

    public void windowClosing(WindowEvent event) {
        System.exit(0);
    }

    public void windowOpened(WindowEvent event) {
    }

    public void windowClosed(WindowEvent event) {
    }

    public void windowIconified(WindowEvent event) {
    }

    public void windowDeiconified(WindowEvent event) {
    }

    public void windowActivated(WindowEvent event) {
    }

    public void windowDeactivated(WindowEvent event) {
    }

    public void paint(Graphics g) {
        setBackground(Color.white);
        box.paint(g);
    }

    class Boxie {

        private int x, y, width, length;
        private Color c;

        public Boxie(int initX, int initY, int initW, int initL, Color initC)
    {
        x      = initX;
        y      = initY;
        width  = initW;
        length = initL;
        c      = initC;
    }

    public void Left(int n) {
        x -= n;
        if (x < 0) x = 0;
    }
}
```

```
public void Right(int n) {
    x += n;
}

    public void paint(Graphics g) {
        g.setColor(c);
        g.fillRect(x,y,width,length);
    }
}
}
```



5) menus in applications

- a) declare menu items: `private: MenuItem i;`
- b) create a menu bar: `MenuBar mb = new MenuBar();`
- c) create a menu on the menu bar: `Menu m = new Menu();`
- d) instantiate menu items: `i = new MenuItem("label");`
- e) add menu item to menu: `m.add(i);`
- f) register menu item: `i.addActionListener(this);`
- g) add menu to menu bar: `mb.add(m);`
- h) `setMenuBar(mb);`
- i) create event handler: `actionPerformed`

example:

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

public class chp18b extends Frame implements ActionListener,
WindowListener {

    private MenuItem exitProg;
    private MenuItem left, right;
    private Boxie box;

    public static void main(String [] args)
```

```
{
    chp18b example = new chp18b();

    example.setSize(300,300);
    example.setVisible(true);
}

public chp18b() {
    box = new Boxie(100,100,10,10,Color.red);

    setTitle("Java Standalone with AWT");
    setLayout(new FlowLayout());

    MenuBar mb = new MenuBar();

    Menu file = new Menu("File");

    exitProg = new MenuItem("Exit");
    file.add(exitProg);
    exitProg.addActionListener(this);

    mb.add(file);

    Menu move = new Menu("Move");

    left = new MenuItem("Left");
    move.add(left);
    left.addActionListener(this);

    right = new MenuItem("Right");
    move.add(right);
    right.addActionListener(this);

    mb.add(move);

    setMenuBar(mb);

    this.addWindowListener(this);
}

public void paint(Graphics g) {
    setBackground(Color.white);
    box.paint(g);
}
```

```
public void actionPerformed(ActionEvent event) {
    if (event.getSource() == exitProg)
        System.exit(0);
    if (event.getSource() == left)
        box.Left(5);
    if (event.getSource() == right)
        box.Right(5);
    repaint();
}

public void windowClosing(WindowEvent event) {
    System.exit(0);
}

public void windowOpened(WindowEvent event) {
}

public void windowClosed(WindowEvent event) {
}

public void windowIconified(WindowEvent event) {
}

public void windowDeiconified(WindowEvent event) {
}

public void windowActivated(WindowEvent event) {
}

public void windowDeactivated(WindowEvent event) {
}

class Boxie {

    private int x, y, width, length;
    private Color c;

    public Boxie(int initX, int initY, int initW, int initL,
                Color initC) {
        x      = initX;
        y      = initY;
        width  = initW;
        length = initL;
        c      = initC;
    }
}
```

```
public void Left(int n) {  
    x -= n;  
    if (x < 0) x = 0;  
}  
  
public void Right(int n) {  
    x += n;  
}  
  
    public void paint(Graphics g) {  
        g.setColor(c);  
        g.fillRect(x,y,width,length);  
    }  
}  
}
```

