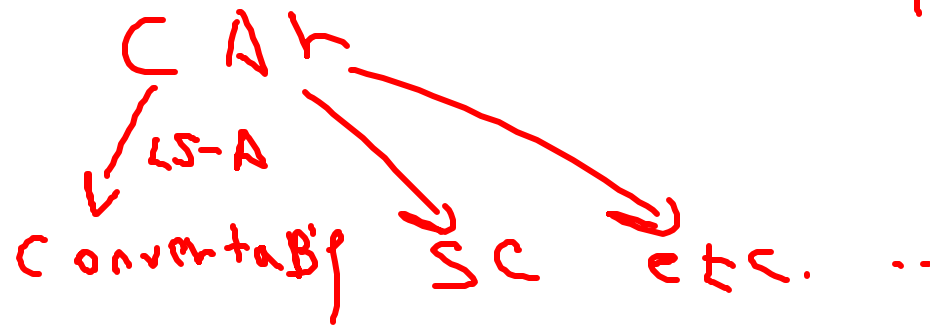


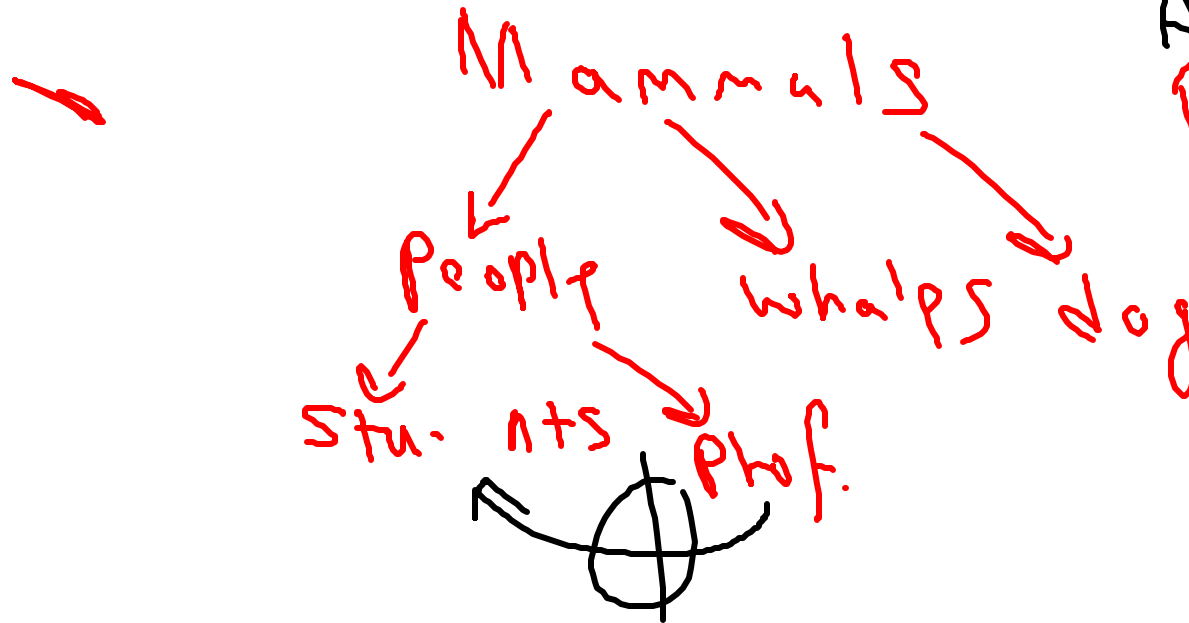
Ako

A-kind-of

Data Struct
+ Methods



Mammals
Have Hair



Acyclic
Digraph
Taxonomy

```

class Customer {
    private static int noc = 0;
    Customer () {
        noc++;
    }
    public int getNumberofCustomers () {
        return noc;
    }
}

```

Lect 4Static

- Methods
- Variables

Abstract

- Classes
- Methods

Final

- Classes
- Methods
- Constants

Wrappers

- Boolean
- Character
- Numeric

Strings

```

Public class Lamp {
    2. Static int MP = e * L; Power();
    Members
    are in
    Shared
    Memory
    Static int e = 110;
    Static int l = 1;
    // Does not compile.

```

1. First independent
 Second dependent

```

Static int power() {
    return e * l;
}

```

Interface

Interface Drawable {

public void draw(),



Abstract Classes

Provide for implementation

Reuse

```

Interface movable {
    public void
        move(int x, int y),
}
Public interface erasable {
    public void erase(),
}

```

A bitwired class GO
 Implements Movable, erasable ...

```

int x, y;
Public void move(int x, int y) {
    erase(); x = x, y = y;
}
draw();

```

1. Abstract Methods
Have no impl.

```
abstract class Foo {
```

```
    abstract void  
        print();
```

```
}
```

```
class Wow extends Foo {
```

```
    void print() { }  
}
```

2. classes containing
abstract methods
must be abstract



3. abstract classes
cannot be instantiated

Final
Method

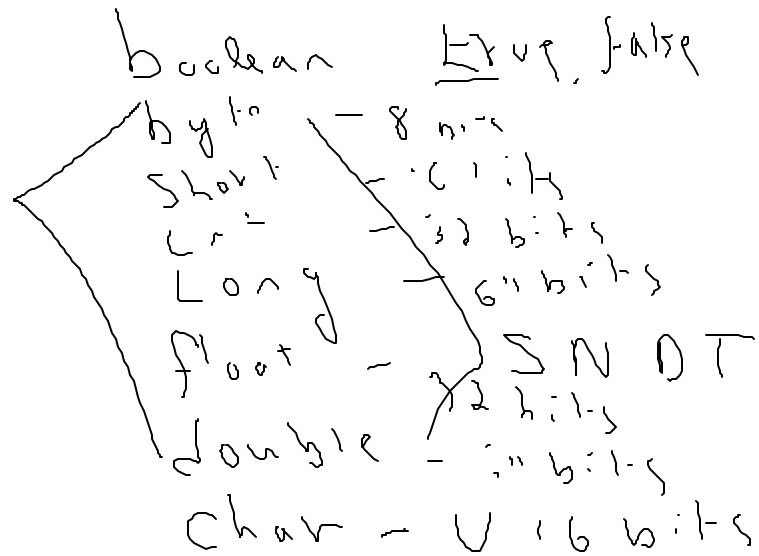
— in line EXPANSION

Final void print() { ... }

Can
Be
Final

Final method cannot be overridden →
Variables

Final classes cannot be subclassed



```
Boolean b = new Boolean("True") →  
boolean b = →  
isBoolean getBoolean("yes"); →
```

```
Character  
a = new Character('a');  
char aChar = a.charValue();  
aChar = Character.toUpperCase('A');  
Integer I = new Integer(2);  
OR X goto 100, 200, 300
```

String s = Integer.toString(i, 2);

SL = Long.toString(L, 16);

I = Integer.valueOf(S, 2);

L = Long.valueOf(SL, 16);

L = Integer.parseInt(SI);

Strings

a String =

String.valueOf(anObject);

Object.toString();

String.valueOf(charArray);

String s = "10 + ""

s = L + " " + j;
L + j + " = " + q;

String s1 = "Hi"

String s2 = "hi";

~~if (s1 == s2) {~~ ...

if (s1.equals(s2))
letop 1

s1.startsWith("foo");
s1.endsWith(".ink");
(s1.toLowerCase()).endsWith(".ink");

Container
Classes

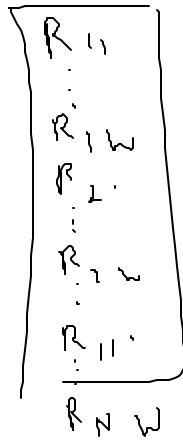
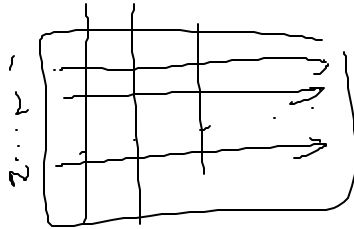
Array $O(n \cdot \text{mult}(h))$

int $l = 2$: int[][] mult(
 int deepArray int b[][])

[] [] [] ...
 = new int [l] [0] ... ;

deep Array . length
 deep Array [0] . length
 deep Array [0] [0] [0] length





```
class Array {
```

```
    int i(5);
```

```
    new int[10]{1,0};
```

```
    void print() {
```

```
        for (int x = 0; x < i.length; x++) {
```

```
            for (int y = 0; y < i[x].length; y++)
```

```
                System.out.println(i[x][y] + " ");
```

```
            System.out.println();
```

```
        }
```

1. arrays are immutable
2. length is read only
3. can't get # of Dim!
4. Monogeneous
for primitive data types
5. index starts at 0!
6. defaults Ref null
Number 0
boolean false

Vectors

java.util

```
Vector v = new Vector();
```

```
v.addElement(new Integer(10));
```

```
int i = v.size();
```

```
v.addElement("hello");
```

Containers
↳ Arrays
Vectors

```

* int getSize () {
    } Return v.size()

```

```

Class Shapes {
    Private Vector v = new Vector ();
    Public void add (Shape s) { if (s
    } V.addElement(s), ← Instance of
    Shape getAt (int i) { Shape)
    (Shape) v.elementAt(i),
}
*

```

```
void draw() {  
    for(int i=0; i < v.size(); i++)  
        ((Shape)v.elementAt(i)).draw();  
}
```

```
class No {  
    String name;  
  
    void print() {  
        .  
        .  
    }  
}
```

Exceptions

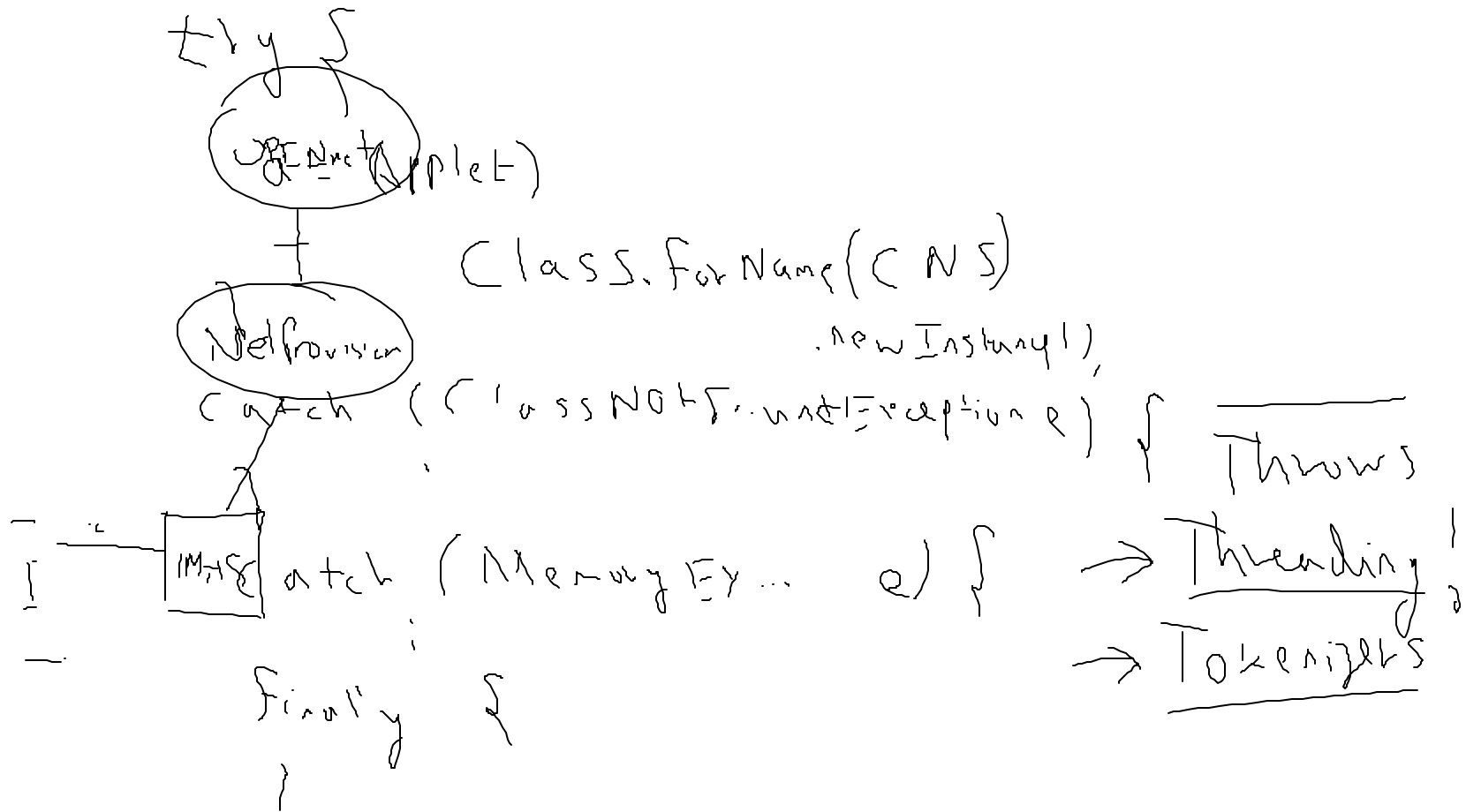
File f = null;
Exceptions

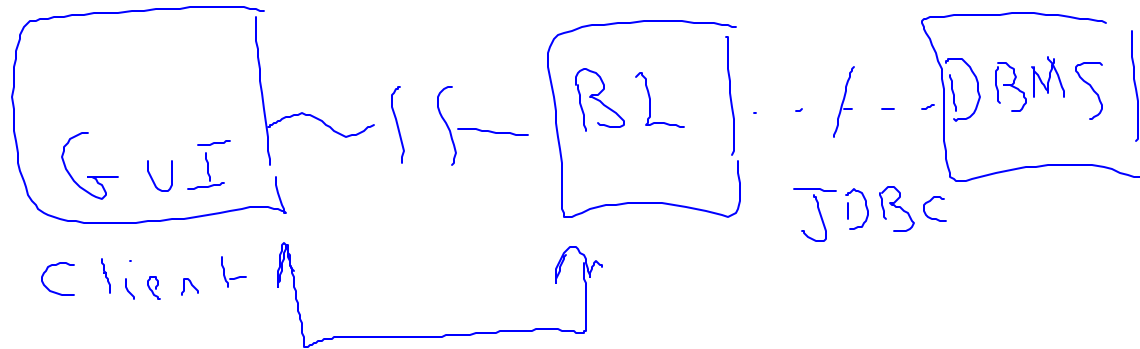
```
try {  
    f f = ...  
} catch (Exception e) {  
    System.out.println(e);  
}
```

f is undefined

```
String getFileName()  
    throws  
    IOException
```

```
try {  
    String fn = getFileName();  
} catch (IOException e) {
```

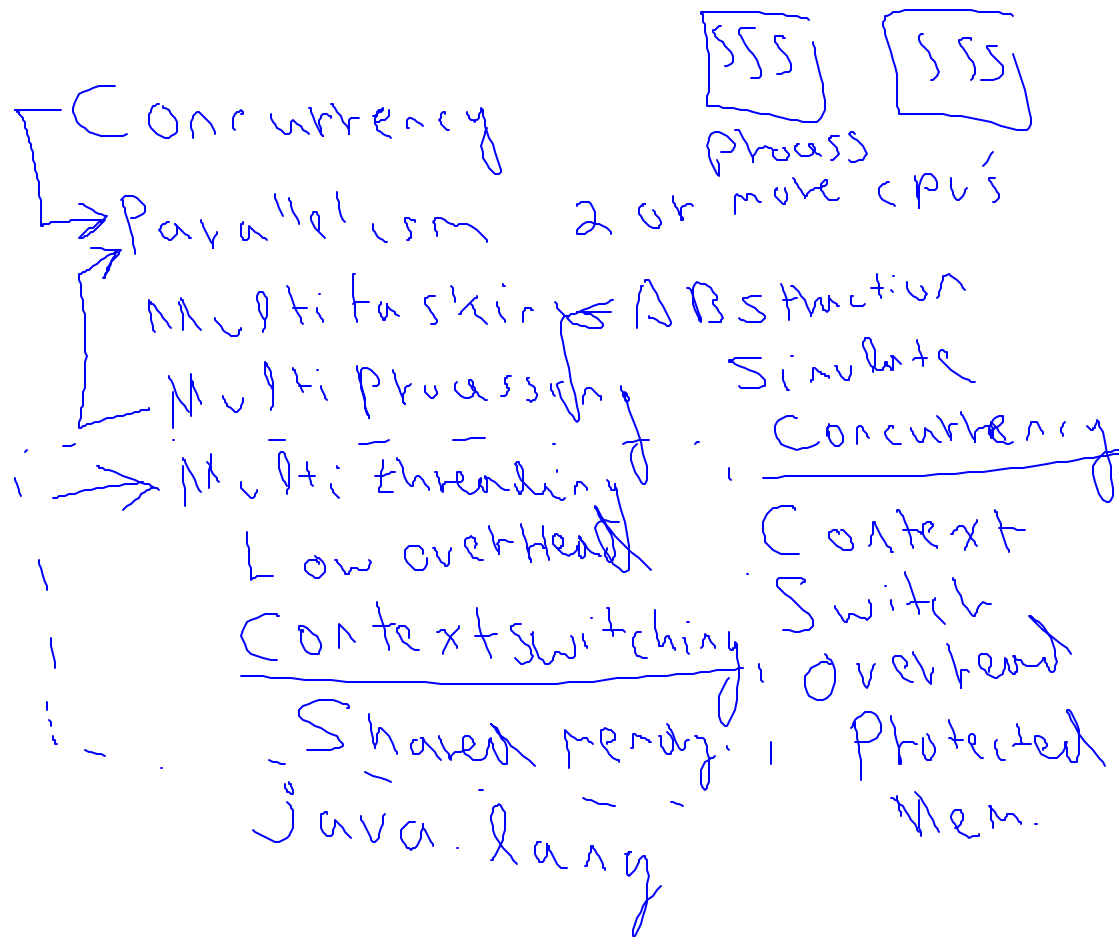




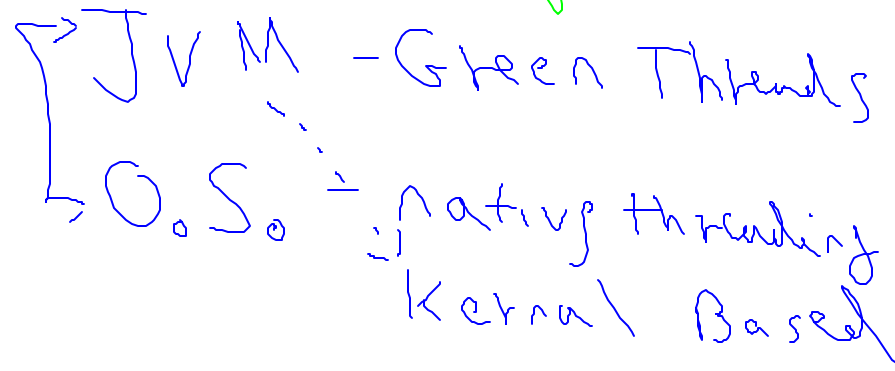
ASP
Servlets
JSP
EJB
CGI

The
Last
Time:
Strings
CONT.
Exception

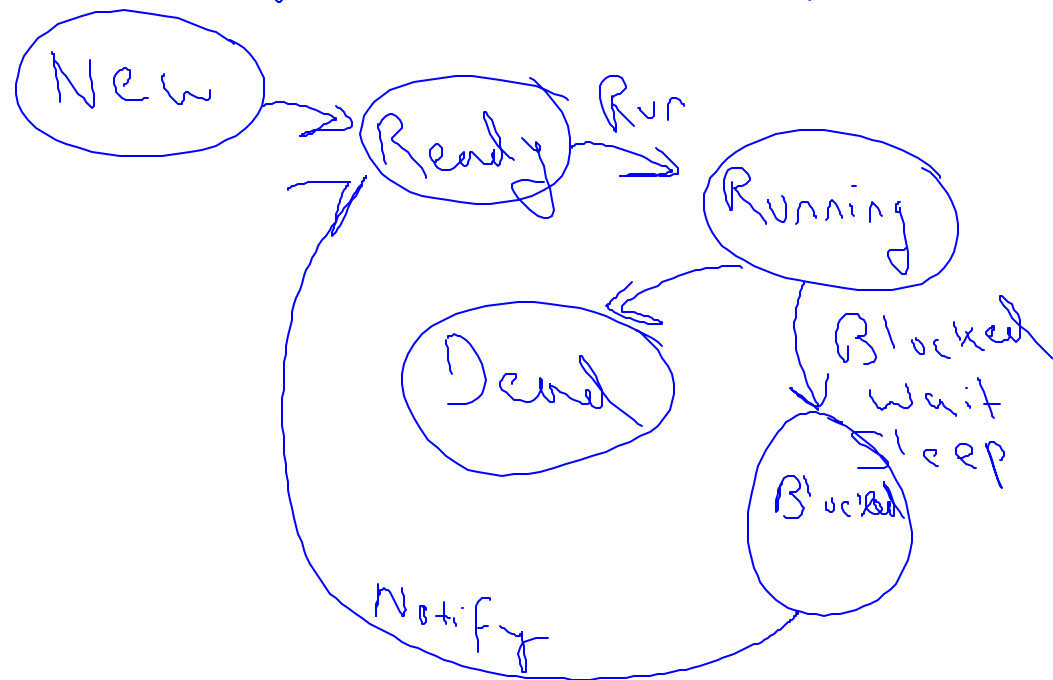
Threading
OS concepts



Mult. - Threading



Life cycle of a Thread



Runnable interface
- Run method

public void run();

Runnable t = new RC();

Class RC implements Runnable
Thread t = new Thread(this);

RC() {
t.start();
}

Thread.Sleep(long ms)
Thread.yield() // allows
others to Run

Synchronizing Threads

- | | |
|----------------------------|------------------|
| 1. Math major | |
| 2. No Beer 😞 | 1. RM comes home |
| 3. Go Buy Beer | 2. 😞 No Beer |
| 4. Comes Home | 3. Go Buy Beer |
| <u>Need Strictly Notes</u> | 4. Comes Home |

Synchronized

void BuyBeer() {
 i
}

Atomic - Runs all at once

IMV

Wait

```
Class Anim implements Runnable {
```

```
    public void
```

```
        synchronized run() {
```

```
            while (moreTimes()) {
```

```
                repaint();  
                try {
```

```
                    wait();  
                } catch (InterruptedException e) { ... }
```

```
public void paint(Graphics g) {  
    synchronized (this) {  
        g.drawImage(img, 0, 0, null);  
        Thread notify();  
    }  
}
```

Dead locks

Occur when 2 or

more threads

want for each other

↓ No progress is made

Synchronized methods

Have Cost

Synchronized refs
Synchronized (adt) of
for (each cust) }
Synchronized(cust) }
// locks cust
// unlocked cust

One Thread

→ many Locks

Suspend + Resume

Can Pause + Restart
a thread ... But Locks
are Not Released

⇒ Deadlocks possible!

Resume
Stop is now
Deprecated!

Use a Boolean

Flag in Run method.

yield, sleep(long n),
start(), run(), interrupt(),
work(), notify(), notifyAll()

```
Thread t = new Thread();  
t.start(),  
gets caller's priority
```

```
public void run() {  
    runFlag = true;  
    while (runFlag) {  
        runFlag = doStuff();  
        yield();  
    }  
}
```

The System
class

System.arraycopy(src, 0, dest, 0, src.length);

long = System.currentTimeMillis()

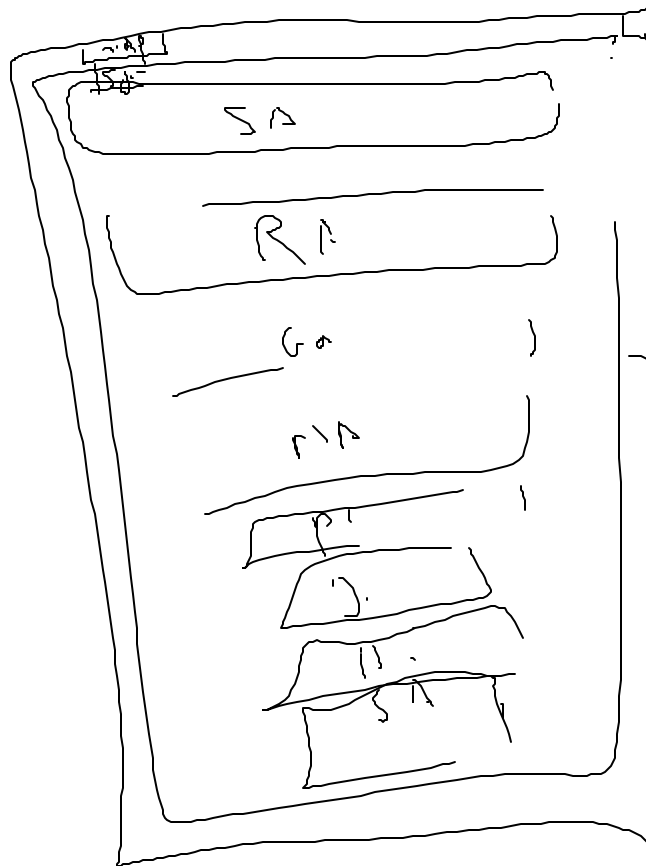
System.exit(int status);

status = 0

Stops the JVM

Not For Applets

String Tokenizers
Public -



Main MenuBar

- add menus
- add new items

Lect. 8

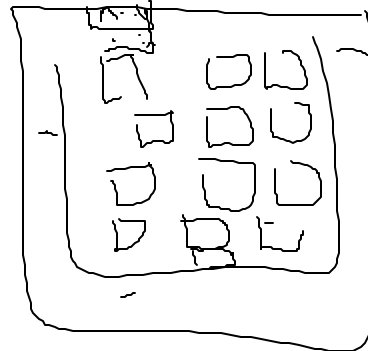
Free Books

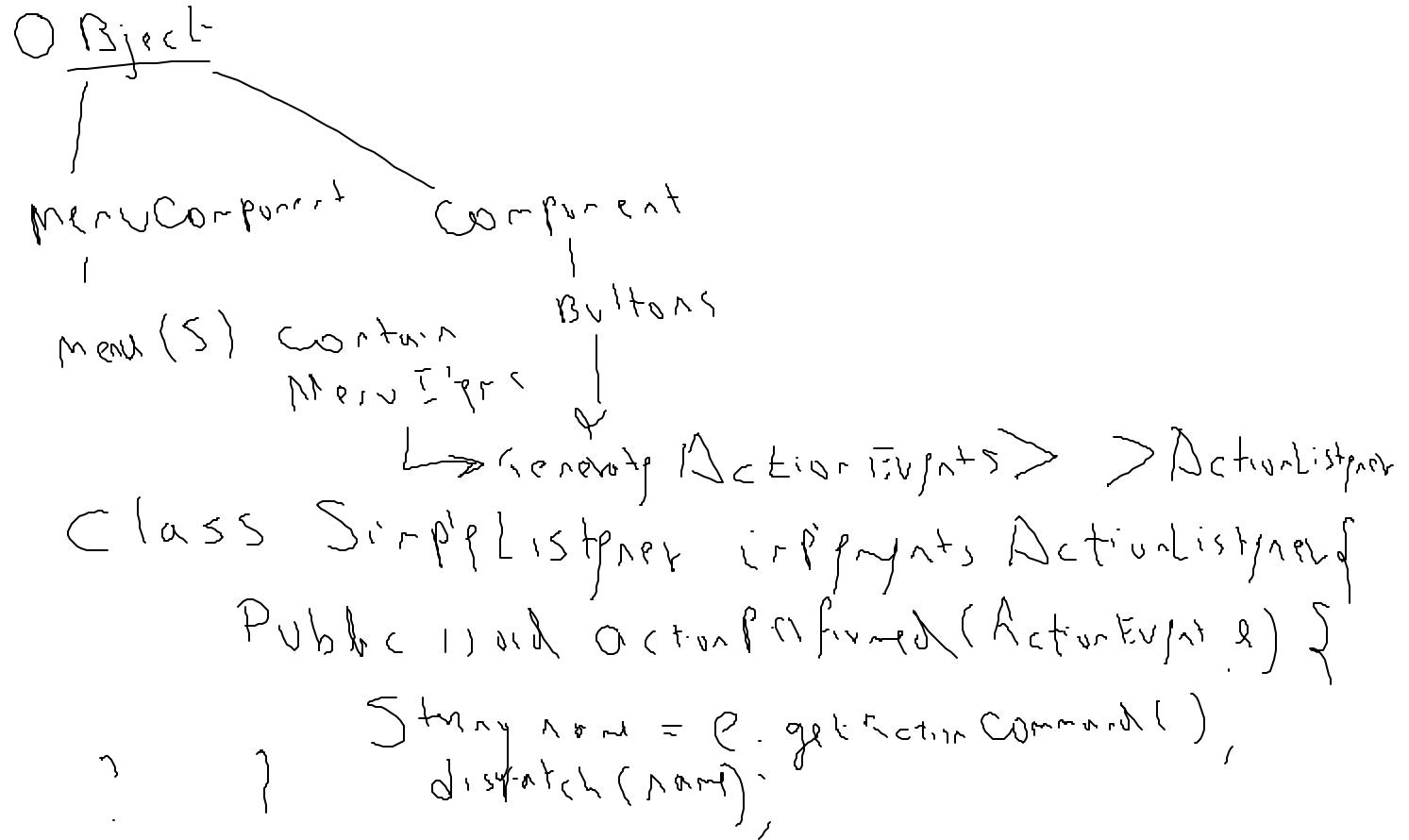
Pizza
Tool...

Continued

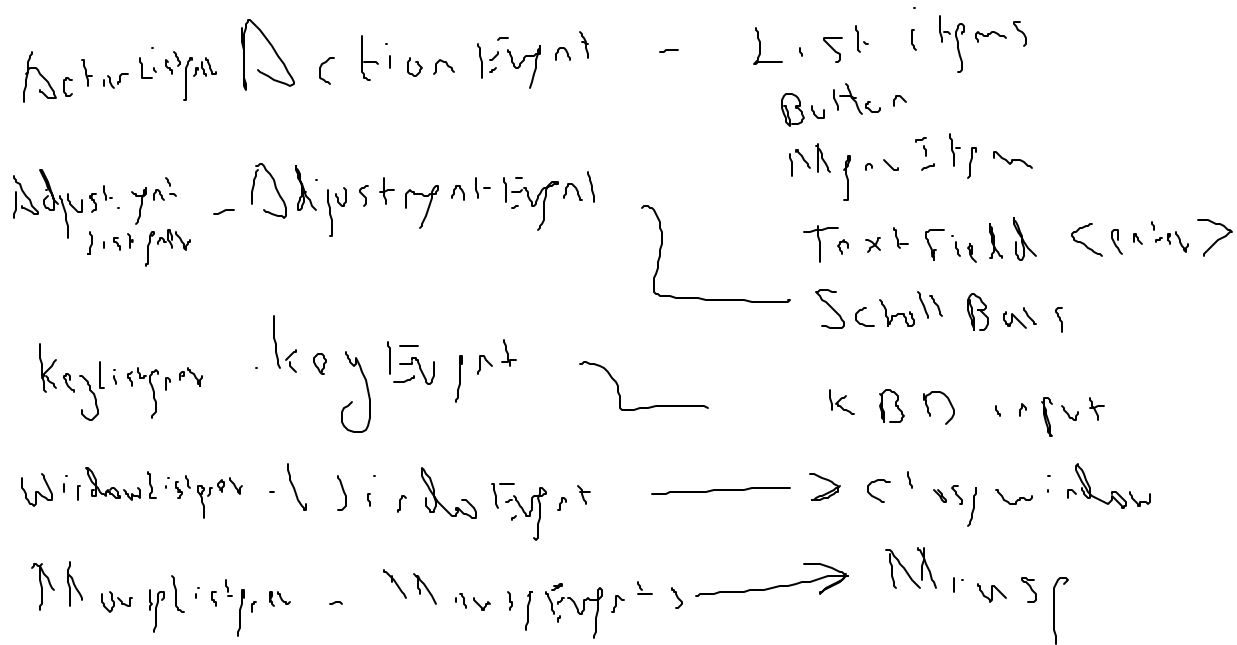
GL(0,1)

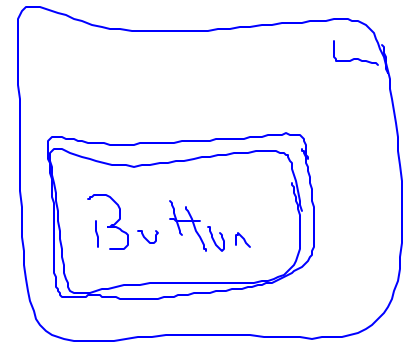
GL(0,1,20,20) RAD





AW Event classes





Bordyr layout. west
Bordyr layout. east
South
North

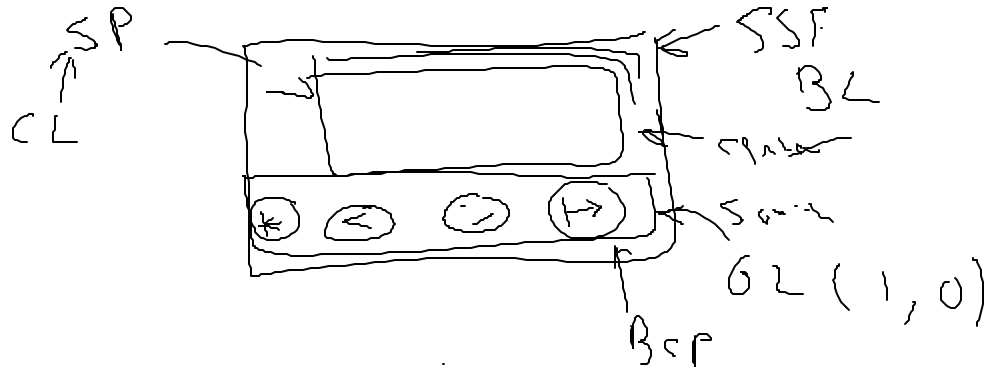
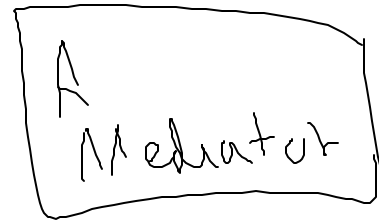
CARD Layout

first (container)

last (container)

next (container)

Previous (container)



Applets

1. Init
2. Start / Start Running
3. Stop
4. Final

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

public class Foo extends java.applet.Applet {
    public void paint(Graphics g) {
        g.drawString("Hello", 5, 30);
    }
}
```

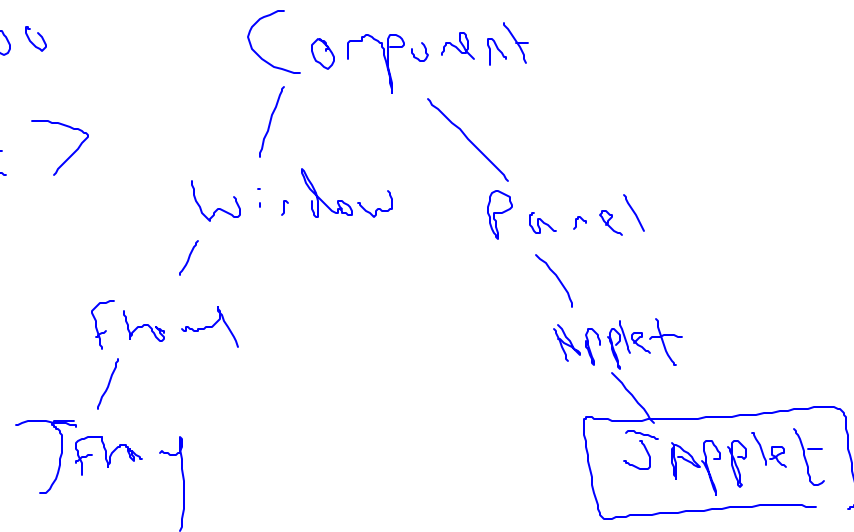
< Applet

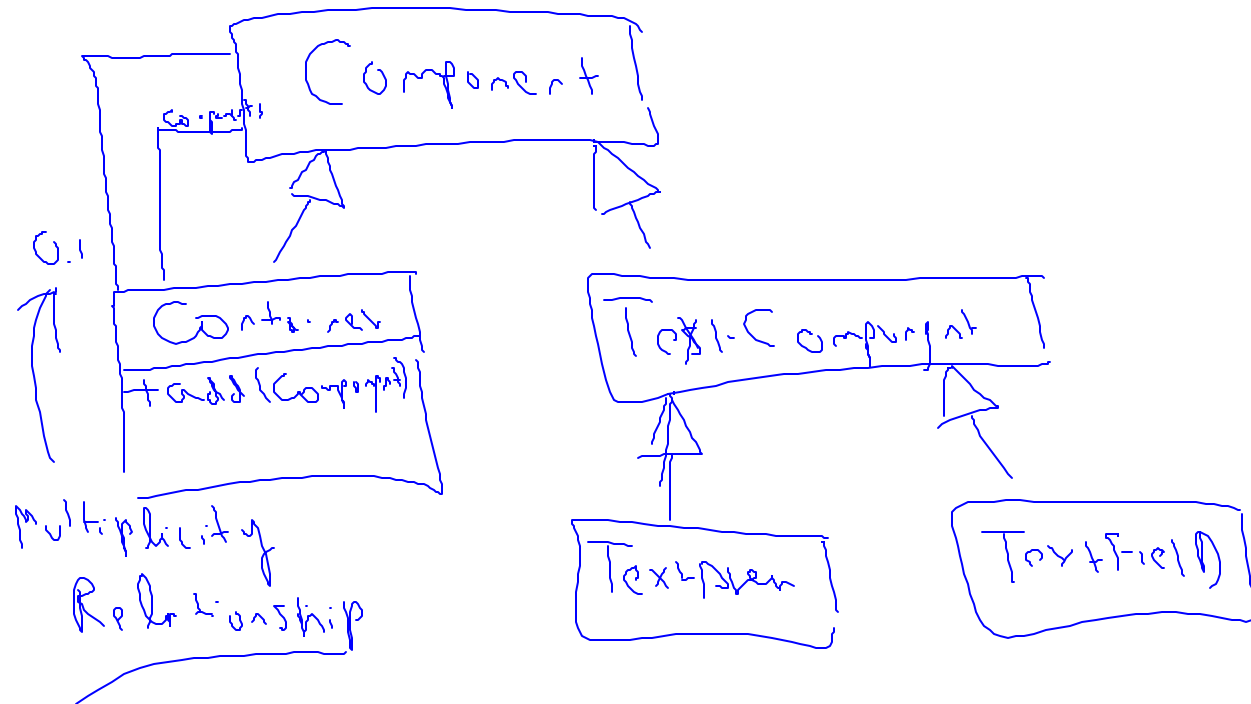
CODE = "foo.class"

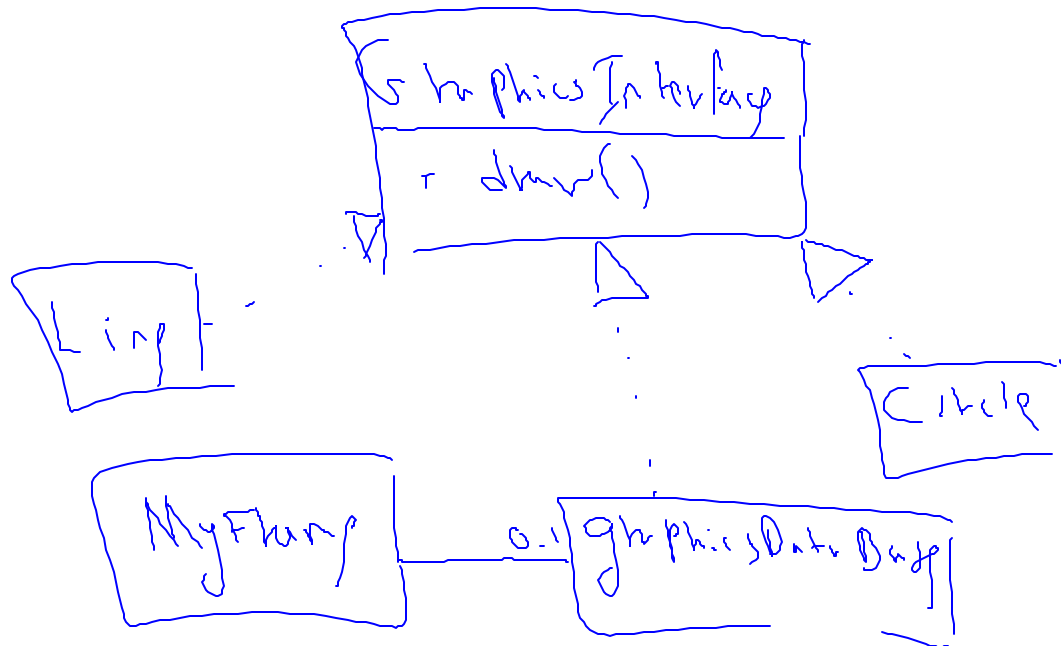
width = 300

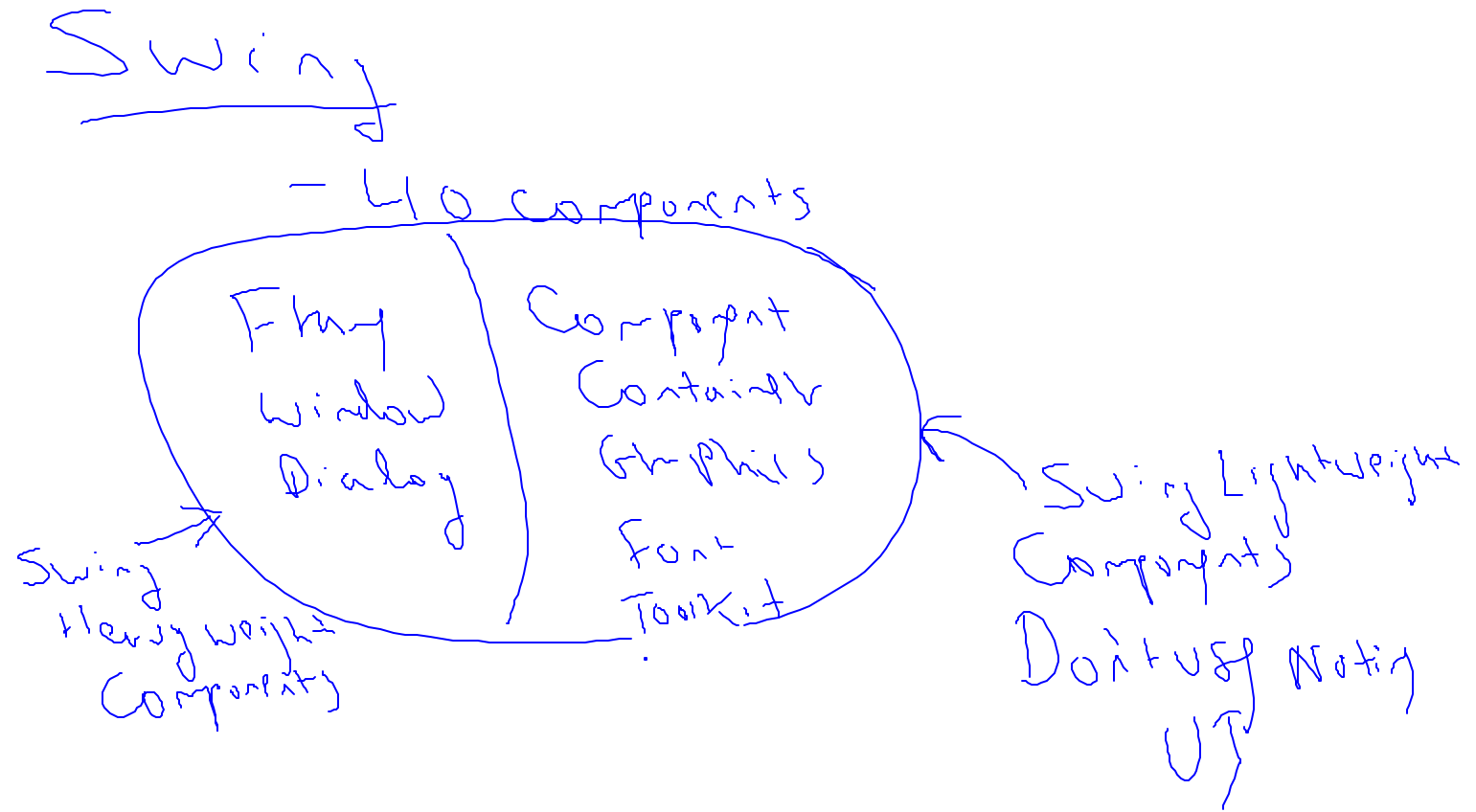
height = 300

< /Applet >



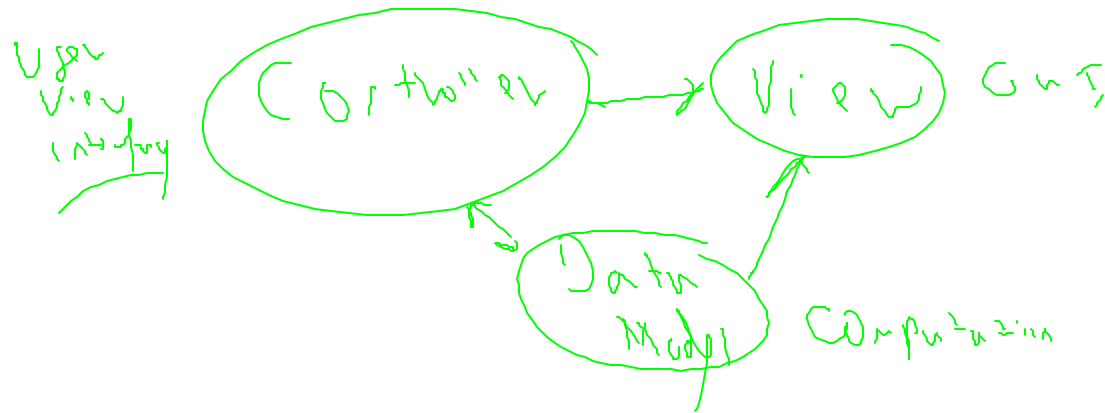






```
Import javax.swing.*;  
import javax.swing.border.*;  
import javax.swing.text.*;
```

Uses MVC Framework: - Design pattern



In JFrame you must

```
JPanel jp = getContentPane();
```

```
jp.setLayout(new BorderLayout());
```

```
JButton b = new JButton("Hi");
```

J Components

- Auto scrolling
- Tooltips
- Borders
- KeyStrokes

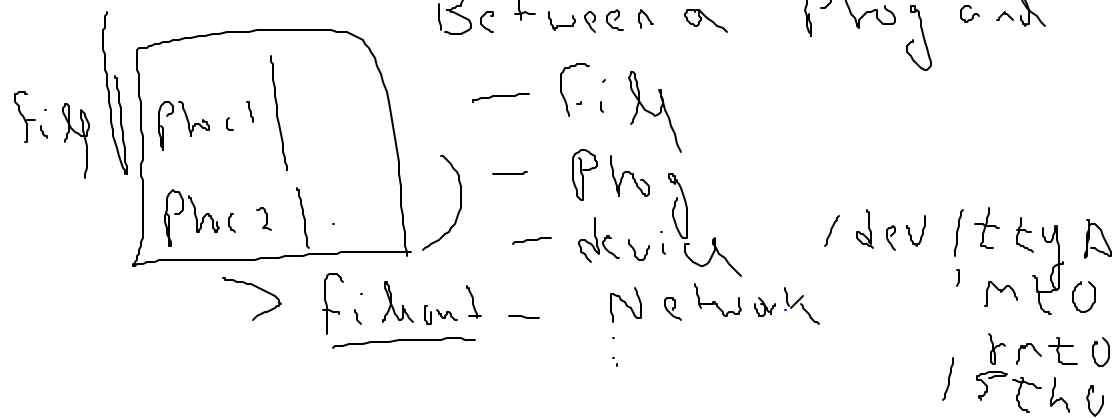
- Settable LAF!

```
public void  
    setLookAndFeel() {  
        String laf  
            = UIManager.getSystemLookAndFeelClassName();  
        try {  
            UIManager.setLookAndFeel(laf);  
        }  
        catch (...) {}  
    }
```


Streams & File I/O

Stream = Unformatted
Sequence of Bytes

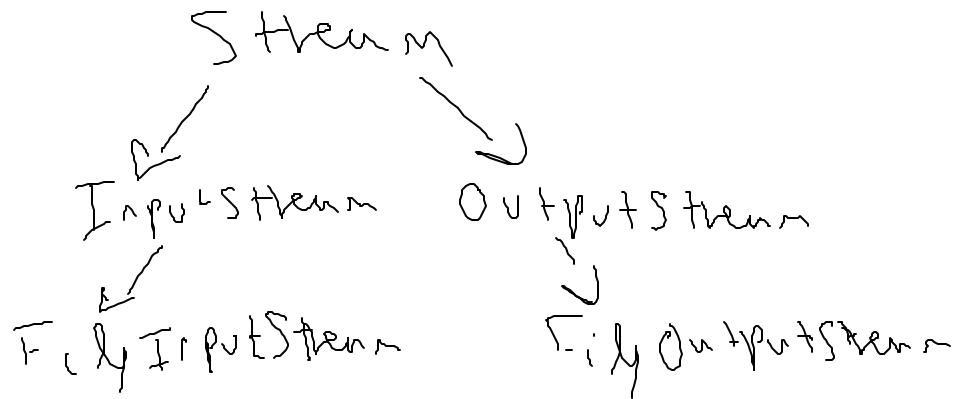
Pipe = stream used
Between a Prog and



Lecture 9

Files

- File Names
- Batch Files
- Streams
- Readers
- Writers
- Tokenizers
- Filters



Import

java.io.*

java.awt.*

File Dialog

fd = new FileDialog

new Frame(),

String Prompt

(int Mode)

```
Public static String  
    getFileName () {
```

```
    Return  
    getFileName(FileDialog LOAD);  
}
```

```

Public Static String
  getFileName(int mode) {
File Dialog fd =      String fn
  new FileDialog(
    new Frame(),
    "Select File",
    fd.show(), mode);
    =
    fd.getDirectory() +
    fd.getFile();
    fd.dispose();
    Return fn;
}

```


How Do I write

dir * java

* wild card

public static String [] getWildNames(

File dir,

String wild) {

absPath = dir.getAbsolutePath();

return

dir.listFiles(new
? WildFilter(wild));

```
String[] frames = dir.listFiles(new FileFilter() {
    for (int i = 0; i < frames.length; i++) {
        frames[i] = absPath + frames[i];
    }
}
return frames;
```

```

Public class WildFilter
    {
        Private String suffix;
        WildFilter (String - suffix) {
            suffix = suffix;
        }
        Public boolean
            accept ( File dir,
                    String fn ) {
            Return
                fn.endsWith(suffix);
        }
    }

```

```
Static Public void  
deletep ( String wild ) {  
  
    String fracs[] =  
    for ( int i = 0; i < fracs.length; i++) {  
        fracs[i] = getWildNews(wild);  
        File f = new File( fracs[i] );  
        deletep( f );  
    }  
}
```

Word Print merge

<< Include FQN >>

Recurring Processing

Fig. Lower File Names

```

public void lowerFileNames(String
    filePath) {
    String filename = filePath.split("\\.");
    String pathStr = filePath.substring(0,
        filename.length - 1);
    for (int i = 0; i < filename.length; i++) {
        String a = filename[i];
        String b = a.toLowerCase();
    }
}

```

try { For Binary
FileOutputStream only

fos =
new FileOutputStream(fn);

fos.write(bytes);

fos.close();

catch (IOException e) { ... }

Make Test HTML

↳ frames [] - getFrames...

PrintWriter pw

= new PrintWriter (fos)

Pw.println("<HTML>")
+ "<body>"
+ "<X1>"

```
for (int i = 0; i < front.length(); ++i) {
```

```
    PrintWriter("<LI>")
```

```
        + "<a href = \"
```

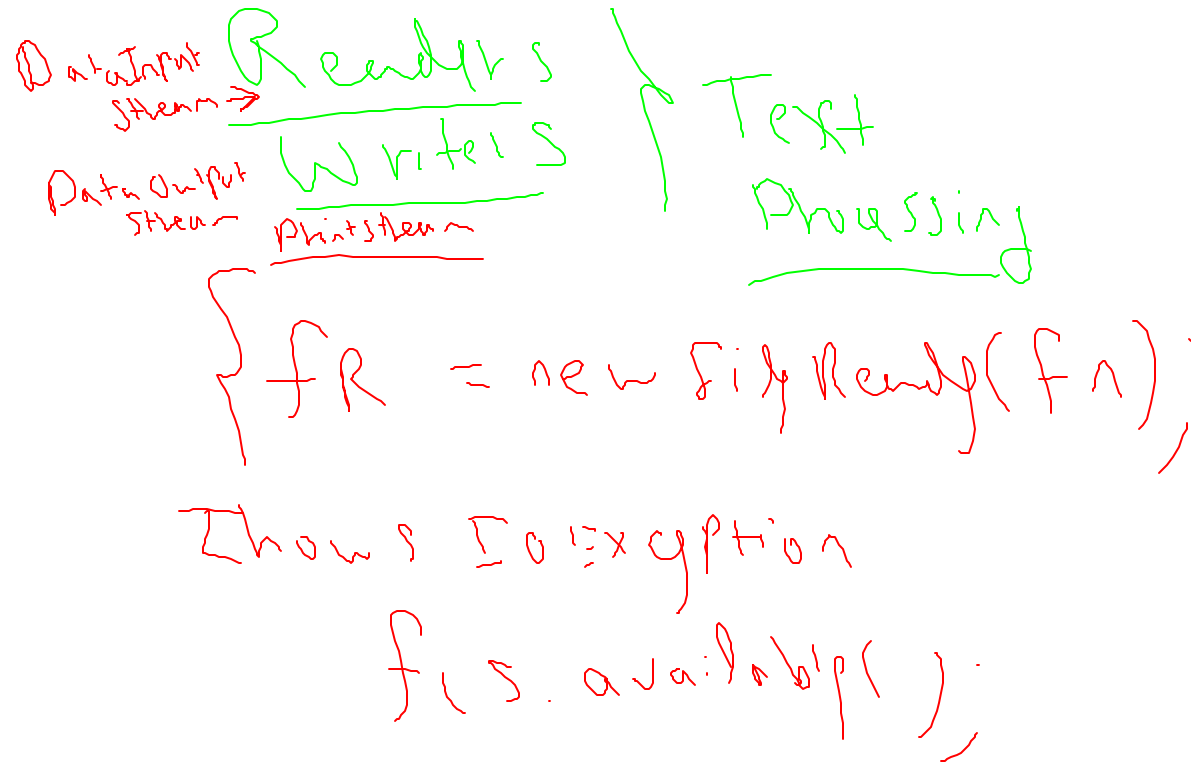
```
        + front[i]
```

```
        + \">\" +
```

```
        front[i]
```

```
    + \"</a></li>\"
```

```
    + \"</ul></body></html>\"
```



StreamTokenizer

≠ StringTokenizer !

↓ TT_EOF ✓

↓ TT_EOL ✓

↓ TT_NUMBER ✓

↓ TT_WORD ✓

SE resetSyntax()

SE whileScan(chars (low, hi) → ✓

SW wordChars (low, hi, ✓

```

fr
while ((next = st.nextToken()) !=
      StreamTokenizer.TT_EOF) {
    switch (next) {

```

```

        case:

```

```

            st.TT_NUMBER, {

```

```

                double d = st.nval;

```

```

                ...
            }

```

```

        case

```

```

            st.TT_EOF:

```

```

            case st.TT_WORD:

```

```

fr.close();

```


The
Servlet API

— No GUI

Client
HTML

{put
get
post}

Web

Java
Servlet
JSPDK

call
Servlet(S)
fss

Shell. Startup

Lecture 10

Servlets!

Sessions

Survey

Object
Serialization

doGet

doPost

doPrint

— Mime Types —

Binary Responses

Simple Servlet

Import

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*
```

```
Public class  
    HTTPGetServlet  
    extends  
        HTTPServlet {  
  
    Public void  
        doGet(  
            HTTP Servlet Request req,  
            HTTP Servlet Response res) {  
    }  
}
```

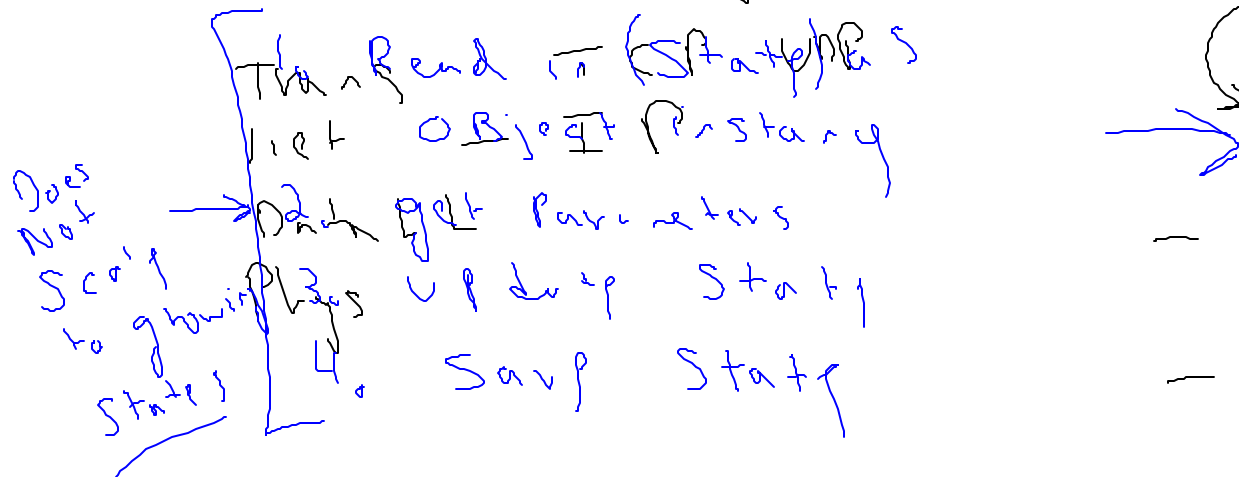
$P_{\omega} = \text{res. gew.}(\dots)$
 $P_{\omega} \text{ Min.}(\dots)$

$\left. \begin{array}{l} P_{\omega} \text{ class}(\dots) \\ \dots \end{array} \right\}$

PHP class

PHP session post script P Servlet

Session extend by HTTP Service EGI



1. S
1. Read Statz

Object Input Stream

Object
Instance
S1

```
ois =  
new ObjectInputStream  
(ObjectInputStream  
(f));
```

Statz S = (Statz) ois.readObject();

2. Get Param
3. Modify Statz
4. Save Statz

f(Statp)

f(Object)

f(new Statp / Obj

f((Statp) Object)

```

File f = new File("S.ruby.exe"),
Object o = OLS readObj(ck),
    OLS.close();
// get + stop, S
Object output = Stream
    OLS =
        new ObjectOutputStream(
            new FileOutputStream(f));
OOLS.writeObj(o);
OOLS.flush();
OOLS.close();
    
```

f(o) →
 ← oo = f(o)
 RMI
 IIOP

Get VS Post

Get

- Static Data

- Shows on URL

Post:

- Dynamic Data

- HTTP:// . 8080 / cookie

Post - Better for passing user input

- Params stored in Body of Request

Put - File uploads

delete - File Deletes

do Delete

Forms and Input to Servlets

```
<Form  
  Method = "Post"  
  Action = "mailto: ..."
```

```
→ <Input type =  
  Text value = 'name'  
  size = "25">
```

Text - Size & maxlength

Checkbox - checked

Radio -

Hidden

Password - Like text No echo

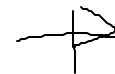
Submit - Submit Form

Reset - Clear Form

Image - Image Button

File - File selection

~~00~~
PW Println...
Logic HTML
<a href = ...
DL ANT
P = & P2:



HTTP / 1.0

MIME - Multipurpose
Internet Mail Ext.

Id's of File Type

Text/html

/ Plain

audio/basic ... au

x-lav

Image

/ gif

/ jpeg

HTTP Servlet Response
H SR

→ "image/gif"

```
{ H SR.setContentType("Text/plain");  
  PrintWriter pw =  
    H SR.getWriter();
```

ServletOutputStream

```
SOS = H SR.getOutputStream()  
SOS.write(bytes);  
SOS.close();
```

1. - - - ↗