

Data Formats for Persistent Storage

Note Title

2/8/2007

	Human readable/modifiable can manually read/edit in files	Machine readable only can be more compact & faster
Customized programmer defines what is saved & how it is saved		
Standardized provided algorithms automatically save & load data		
Managed separate subsystem maintains the data		

	Human readable/modifiable can manually read/edit in files	Machine readable only can be more compact + faster
Customized programmer defines what is saved + how it is saved	character files in a customized format Ex: Print Stream (out) Scanner (in)	application uses system-provided formats to read/write data Ex: DataOutputStream, DataInputStream
Standardized provided algorithms automatically save + load data		
Managed separate subsystem maintains the data		

character files in a customized format

Ex: PrintStream (out)
Scanner (in)

```
ps = new PrintStream(new  
    FileOutputStream(f))  
ps.println(---);
```

Parsing — process of reading/interpreting characters
Often looking for patterns

a common type of pattern: Regular expression over some alphabet

b a a a b a b a a a a b \Rightarrow a^*b any # of a's + then b

a a a b a b a a a a b \Rightarrow a^+b 1 or more a's + then b

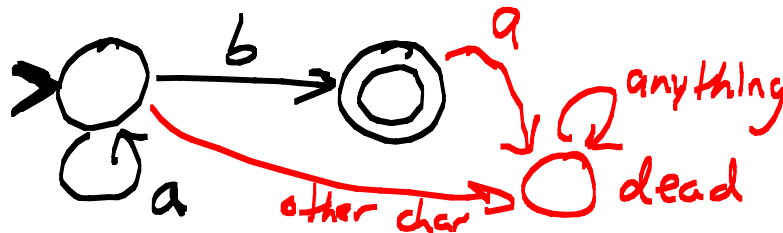
$(a \vee b)^*$ \Rightarrow a string containing any # of a's + b's

Parser can take a pattern + look for matches.

It compiles pattern into a finite state machine:

states + transitions
← incl. start + end states

a^*b



see InteractiveScanner.java on the examples page for a demo

application uses system-provided
formats to read/write data

Ex: DataOutputStream,
DataInputStream

```
File f = new File(filename);  
OutputStream out = new FileOutputStream(f);  
DataOutputStream dataOut =  
    new DataOutputStream(out);  
dataOut.writeInt(myNumber);  
dataOut.writeChars(myString);  
out.close();
```

```
File f = new File(filename);  
InputStream in = new FileInputStream(f);  
DataInputStream dataIn =  
    new DataInputStream(in);  
myNumber = dataIn.readInt();  
myString = dataIn.readChars();  
in.close();
```

	Human readable/modifiable can manually read/edit in files	Machine readable only can be more compact & faster
Customized programmer defines what is saved & how it is saved	character files in a customized format Ex: PrintStream (out) Scanner (in)	application uses system-provided formats to read/write data Ex: DataOutputStream, DataInputStream
Standardized provided algorithms automatically save & load data	high-level semantics is captured in a standard hierarchical format Ex: XML encoder XML decoder	low-level traversal of & restoration of heap structures Ex: Java serialization w/ ObjectOutputStream & ObjectInputStream
Managed separate subsystem maintains the data		

high-level semantics is captured in a standard hierarchical format

Ex: XML encoder
XML decoder

Restrictions: all objects must have

- a no-argument constructor
- getters/setters for properties

Person

String name

int ssn

Person mom

get/set for each

no-arg constr. w/
default values

Compares object being saved to a "default" object & saves the differences

See Person.java and Test.xml on the examples page.

low-level traversal of +
restoration of heap structures

Ex: Java serialization w/
ObjectOutputStream + ObjectInputStream

```
ObjectOutputStream oos = new ObjectOutputStream(new  
FileOutputStream(f));
```

```
oos.writeObject(myObject);
```

implements Serializable

similarly for ObjectInputStream
myObject = (↑) ois.readObject()
cast here

ois

↑
no methods!!

Issues: what if you change class definition

Solution:

```
static long serialVersionUID = 1L;
```

← default is a hash
of the "signature"
of the class defn.

	Human readable/modifiable can manually read/edit in files	Machine readable only can be more compact & faster
Customized programmer defines what is saved & how it is saved	character files in a customized format Ex: PrintStream (out) Scanner (in)	application uses system-provided formats to read/write data Ex: DataOutputStream, DataInputStream
Standardized provided algorithms automatically save & load data	high-level semantics is captured in a standard hierarchical format Ex: XML encoder XML decoder	low-level traversal of & restoration of heap structures Ex: Java serialization w/ ObjectOutputStream & ObjectInputStream
Managed separate subsystem maintains the data	data model & access available in database UI Ex: relational database, JDBC	database system manages app's persistent heap objects EX: JDO

data model & access
available in database UI

Ex: relational database,
JDBC

Relation = Table \approx Class

Person			
ID	Name	SSN	MoMID
1	Eve	0	null
4	Abel	4	1

Departments	
Name	ManagerID (Person)
Eden Council	1

DB ← ODBC

JDBC
Java Database
Connectivity

Java
Application
(objects)

ResultSet →
provides iteration
over a table

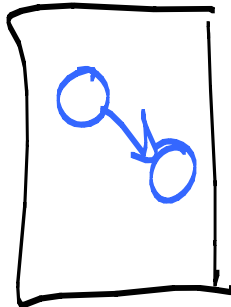
- ① Pose a ^{SQL} query to the database
"SELECT Person WHERE
Name = Eve"
- ② Iterate over the
result set

database system manages app's
persistent heap objects

EX: JDO

Application

Save objects to
the persistent
heap &
load
incrementally



JDO
resolve
disk/
memory
refs.

Persistent Heap on Disk

