Building Enterprise Database Applications

using Rational Rose, Java and Mr Architecture

A short course by Kade Hansson

Course Contents

- **UML, object-orientation and design patterns**
- **Java language and essential APIs**
- **Java GUI components and event model**
- **Java I/O and TCP/IP sockets**
- **JDBC, Servlets and JSPs**
- **Mr Architecture**

Input Streams

- InputStream is a primitive input token queue
  - *FileInputStream*– file implementation
  - *ByteArrayInputStream*, *StringBufferInputStream* – memory implementations
  - *FilterInputStream* wraps another (usually more primitive) *InputStream*
    - *BufferedInputStream*– uses a buffer to read the data
    - *DataInputStream*– allows reading of binary data
    - *PushbackInputStream*– provides unread single byte operation
  - *ObjectInputStream*– for deserializing objects

Output Streams

- *OutputStream* is a primitive queue receiving output tokens
  - *FileOutputStream*– file implementation
  - *ByteArrayOutputStream* – memory implementation
  - *FilterOutputStream* wraps another (usually more primitive) *OutputStream*
    - *BufferedOutputStream*– buffers the data before writing
    - *DataOutputStream*– allows writing of binary data
    - *PrintStream*- for writing character output
  - *ObjectOutputStream*– for serializing objects

Reading and Writing Character Streams

- Subclasses of the *Reader* class are used for reading character streams
  - *Reader*s provide character set decoding, file pointer marking and data skipping functionality over *InputStream*s
  - *Reader* hierarchy largely mirrors that for *InputStream*
- Subclasses of the *Writer* class are used for writing character streams
  - *Writer*s provide character set encoding functionality for *OutputStream*s
  - *Writer* hierarchy largely mirrors that for *OutputStream*

I/O Exceptions

- *IOException*- general I/O failure
  - *FileNotFoundException*– invalid filing system entry
  - *SocketException*– network failure
  - *EOFException*– read beyond end-of-stream
  - *WriteAbortedException* and *ObjectStreamException*– occur during serialization

Other Classes Relevant to I/O

- *Serializable* and *Externalizable* interfaces- used in writing/reading objects
- *StringTokenizer* and *StreamTokenizer*- for building simplest and simpler parsers
- *RandomAccessFile*– for doing random-access I/O to files
- *File* and *FileSystem*– filing system and filing system entry abstraction
- *URL* and *URLConnection*– for identifiying and connecting to network resources
- *Socket* and *ServerSocket*– for low-level network communications