Java I/O

Streams, Readers and Writers

- A stream is a sequence of bytes.
  - Input streams obtain bytes from an external source
  - Output streams move bytes to an external target
  - Streams can be used to communicate between Java threads
- Reader obtains character data
  - Convert bytes to characters using appropriate conversion
- Writer outputs character data
  - Convert Unicode characters to bytes according to local conversion rules

Standard Stream

- Java provides 3 standard I/O streams:
  - System.in InputStream for standard input
  - System.out PrintStream for standard output
  - System.err PrintStream for standard error output
- Use buffered I/O for efficiency:
  - BufferedReader
  - BufferedWriter
  - BufferedInputStream
  - BufferedOutputStream

File Input and Output

- Read files with FileReader or FileInputStream
  - Constructor uses filename or File object
  - Can throw FileNotFoundException
- Reading methods return -1 or null on EOF

- Write to files using FileWriter or FileOutputStream
    - Constructor uses filename or File object
    - Can throw IOException when creating file (e.g. disk full)
- Use close() method to end an I/O stream
- File class encapsulates file information
    - Think of File objects as holding file path names, not files
    - Hides details of directory and file separator characters
- Contains useful file and directory methods
    - canRead() file is readable
    - canWrite() file is writeable
    - canExecute() file is executable
    - exists() file exists
    - delete() delete file
    - isDirectory() tests if a name is a directory
    - length() size of file
    - list() list all files in a directory as String[]
    - mkdir() creates a directory
    - lastModified() time&date of last change as a Date