David Keil Fall 2008
63.152 Computer Science I
Framingham State College

Java file input/output

The following program reads the first line of a file, *update.txt*, from disk, prompts for a new line from the user, and write the new file contents to disk, replacing the old contents.

```
/* update.java:
   Reads a string from file 'update.txt', prompts for update of the string, updates file.
   D. Keil 4/08
*/
import java.util.Scanner;
import java.io.FileReader;
import java.io.PrintWriter;
import java.io.FileNotFoundException;

public class update
{
    public static void main(String[] args)
          throws FileNotFoundException
    {
          // Open input file, read a string and display it:
          System.out.print("Reading 'update.txt'");
          FileReader reader = new FileReader("update.txt");
          Scanner fin = new Scanner(reader);
          String buf = fin.nextLine();
          fin.close();
          System.out.println("File contents:\n"+buf);

          // Prompt for new string:
          System.out.print("Enter new text to store in file: ");
          Scanner cin = new Scanner(System.in);
          buf = cin.nextLine();

          // Write new string to file:
          PrintWriter fout = new PrintWriter("update.txt");
          System.out.print("Writing " + buf  + " to file 'update.txt'");
          fout.println(buf);
          fout.close();
    }
}
```

The packages listed at the top provide class definitions for the keyboard-input scanner object, the file-input object, the file-output object, and the exception (error-reporting) object used in the program.

A method, such as *main* here, that can encounter a file-opening error (exception), has a *throws* clause after its header.

The program is divided into three parts, using comments at the beginning of each part. The first part reads the contents of the file, the second prompts the user for new text to store in the file, and the third writes the new text to the file, replacing the old contents.

The predefined class *FileReader* enables declaration of a file-reading variable, *reader*, associated with the file *update.txt.* The object, *reader*, is used to declare a file input stream object, *fin*. The expression *fin.nextLine();* returns the string found in the file stream up to the end of file or the next end of line.

Similarly, the predefined class *Scanner* is used to declare a keyboard-input stream object, *cin*, which is used to retrieve a string from the user.

The standard class *PrintWriter* is used to declare a stream-output object, *fout*, associated with the file *update.txt*. The statement *fout.println(buf);* stores the string *buf* to the file.

Creation of the *reader* and *fout* objects opens the file for input and output, respectively. This means that a channel is created between the disk file and the memory. File-opening operations in Java prorams are matched by file-closing operations.

David Keil Fall 2008
63.152 Computer Science I
Framingham State College

Reading and displaying file records

The following program can help with the file-reading and output-display aspects of the semester project.

The *printf* method takes a string parameter, called *format string*, followed by a list of parameters that will be displayed as specified in the format string. The boxed program line

```
System.out.printf("%-10s %6d %5.1f %n",name,num_courses,qpa);
```

Displays a String left-justified across 10 horizontal spaces, an integer right-justified across six spaces, and a *float* value, formatted to one decimal place, across five spaces. The "%n" at the end generates a newline.

```
/*
   read_rec.java:
   Reads two data records from file,
   displays them formatted as table.

   D. Keil 11/08
*/
import java.util.Scanner;
import java.io.FileReader;
import java.io.FileNotFoundException;

public class read_rec
{
    public static void main(String[] args)
       throws FileNotFoundException
    {
       // Open input file:
       System.out.println("Reading 'read_rec.txt'");
       FileReader reader = new FileReader("read_rec.txt");
       Scanner fin = new Scanner(reader);

       // Read and display record:
       String name = fin.next();
       int num_courses = fin.nextInt();
       float qpa = fin.nextFloat();
       System.out.printf("%-10s %6d %5.1f %n",name,num_courses,qpa);

       // Read and display record:
       name = fin.next();
       num_courses = fin.nextInt();
       qpa = fin.nextFloat();
       System.out.printf("%-10s %6d %5.1f %n",name,num_courses,qpa);

       // Close file:
       fin.close();

    }
}
```

The input file

```
Smith 31 3.68
Johnson 5 2.39
```

Generates the following output:

```
Reading 'read_rec.txt'
Smith          31   3.7
Johnson         5   2.4
```