

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 programs are matched by file-closing operations.

String manipulation and search

The following program illustrates the use of loops and the standard class *String*, inputting a line of text, searching it for the first blank space, and extracting the part of the string up to and including the space.

```
/*
srchspc.java:
Prompts for a line of text, displays the first word.

D. Keil 4/08
*/
import java.util.Scanner;

public class srchspc
{
    public static void main(String[] args)
    {
        // Prompt for line:
        System.out.print("Enter a string: ");
        Scanner cin = new Scanner(System.in);
        String buf = cin.nextLine();

        // Find first space:
        int i = 0, spaceloc = 0;
        boolean found = false;
        while (i < buf.length() && !found)

        {
            if (buf.charAt(i) == ' ')
            {
                spaceloc = i;
                found = true;
            }
            i = i + 1;
        }

        // Display the part of the string up to and including the space:
        System.out.println("Hello " + buf.substring(0, spaceloc));
    }
}
```

The *while* loop executes until either the counter (*i*) reaches the end of the string, or until a flag, *found*, is set. The data type of *found* is *boolean*, and it is set to *true* when and if a space is found.

The methods *length*, *charAt*, and *substring*, of the class *String* are called in this program.

Input of strings is illustrated. A scanner (input) stream object, *cin*, is created. The scanner method *nextLine* fetches a series of characters from the keyboard terminated by *Enter*.

Defining and using methods

The following two programs work exactly like *srchspc.java* and *update.java*, but define methods (*first_word*, *read_file*, and *write_file*) to perform some of the operations. The methods are called from *main*. The examples illustrate the concepts of modular design and procedural abstraction. Program code is often easier to write and understand if it is divided into short modules. Even a two-line method, like *main* in *update2.java*, is not too short to be well designed.

Note that the method *first_word* in *srchspc2.java* returns a *String* object, while *read_file* and *write_file* in *update2.java* return no values and are used for their side effects. All three methods happen to take *String* parameters.

```
/*
srchspc2.java:
Prompts for a line of text, displays
the first word.

D. Keil 4/08
*/
import java.util.Scanner;

public class srchspc2
{
    private static String first_word(String s)
    // Returns characters of s up to 1st space
    {
        // Find first space:
        int i = 0, spaceloc = 0;
        boolean found = false;
        while (i < s.length() && !found)
        {
            if (s.charAt(i) == ' ')
            {
                spaceloc = i;
                found = true;
            }
            i = i + 1;
        }
        return s.substring(0, spaceloc);
    }

    public static void main(String[] args)
    {
        // Prompt for line:
        System.out.print("Enter a string: ");
        Scanner cin = new Scanner(System.in);
        String buf = cin.nextLine();

        // Select first word:
        buf = first_word(buf);

        // Display the part of the string up to
        // and including the space:
        System.out.println("Hello " + buf);
    }
}
```

```
/* update2.java:
Reads a string from file 'update.txt',
prompts for update of 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 update2
{
    private static void read_file(String name)
    throws FileNotFoundException
    // Open input file, read and display string.
    {
        System.out.print("Reading "+name);
        FileReader reader = new FileReader(name);
        Scanner fin = new Scanner(reader);
        String buf = fin.nextLine();
        fin.close();
        System.out.println("File contents:\n"+buf);
    }

    private static void write_file(String name)
    throws FileNotFoundException
    // Prompt for new string, write to file
    {
        // Prompt for new string:
        System.out.print("Enter new text to store: ");
        Scanner cin = new Scanner(System.in);
        String buf = cin.nextLine();

        // Write new string to file:
        PrintWriter fout = new PrintWriter(name);
        System.out.print("Saving "+buf+" to "+name);
        fout.println(buf);
        fout.close();
    }

    public static void main(String[] args)
    throws FileNotFoundException
    {
        read_file("update.txt");
        write_file("update.txt");
    }
}
```

