File   Edit   View   Selection   Find   Packages   Help

Project | WebCrawler.java   Cha12_2.java   Cha12_4.java   Cha12_10.java   Cha12_22.java   test2   test1

```java
import java.io.*;
import java.util.*;

public class Cha12_22 {
  public static void main(String[] args) throws Exception {
    // Check command line parameter usage
    if (args.length != 3) {
      System.out.println(
        "Usage: java directory targetFile oldStr newStr");
      System.exit(1);
    }

    // Check if source file exists
    File dir = new File(args[0]);
    if (!dir.exists()) {
      System.out.println("Source directory " + args[0] + " does not exist");
      System.exit(2);
    }

    // get directory files
    for(File file : dir.listFiles() ) {
      String s2 = "";
      try (
        // Create input files
        Scanner input = new Scanner(file);
      ) {
        while (input.hasNext()) {
          String s1 = input.nextLine();
          s2 += s1.replaceAll(args[1], args[2]) + "\n";
        }
      }

      //delete old files and create new blank file for writing
      //File newFile = new File(file.getName());
      file.delete();
      try (
        // Create output files
        PrintWriter output = new PrintWriter(file);
      ) {
        output.println(s2);
      }
      catch(IOException e) {
```

File   Edit   View   Selection   Find   Packages   Help

Project | WebCrawler.java   Cha12_2.java   Cha12_4.java   Cha12_10.java   Cha12_22.java   test2   test1

```
Creates a File object for the specified path name. The path name may be a
directory or a file.
Creates a File object for the child under the directory parent. The child may be
a file name or a subdirectory.
Creates a File object for the child under the directory parent. The parent is a
File object. In the preceding constructor, the parent is a string.
Returns true if the file or the directory represented by the File object exists.
Returns true if the file represented by the File object exists and can be read.
Returns true if the file represented by the File object exists and can be written.
Returns true if the File object represents a directory.
Returns true if the File object represents a file.
Returns true if the File object is created using an absolute path name.
Returns true if the file represented in the File object is hidden. The exact
definition of hidden is system dependent. On Windows, you can mark a file
hidden in the File Properties dialog box. On Unix systems, a file is hidden if
its name begins with a period (.) character.
Returns the complete absolute file or directory name represented by the File
object.
Returns the same as getAbsolutePath() except that it removes redundant
names, such as "." and "..", from the path name, resolves symbolic links (on
Unix), and converts drive letters to standard uppercase (on Windows).
Returns the last name of the complete directory and file name represented by
the File object. For example, new File("c:\\book\\test.dat").getName() returns
test.dat.
Returns the complete directory and file name represented by the File object.
For example, new File("c:\\book\\test.dat").getPath() returns c:\book\test.dat.
Returns the complete parent directory of the current directory or the file
represented by the File object. For example, new
File("c:\\book\\test.dat").getParent() returns c:\book.
Returns the time that the file was last modified.
Returns the size of the file, or 0 if it does not exist or if it is a directory.
Returns the files under the directory for a directory File object.
Deletes the file or directory represented by this File object. The method returns
true if the deletion succeeds.
Renames the file or directory represented by this File object to the specified name
represented in dest. The method returns true if the operation succeeds.
Creates a directory represented in this File object. Returns true if the the directory is
created successfully.
Same as mkdir() except that it creates directory along with its parent directories if
the parent directories do not exist.
```
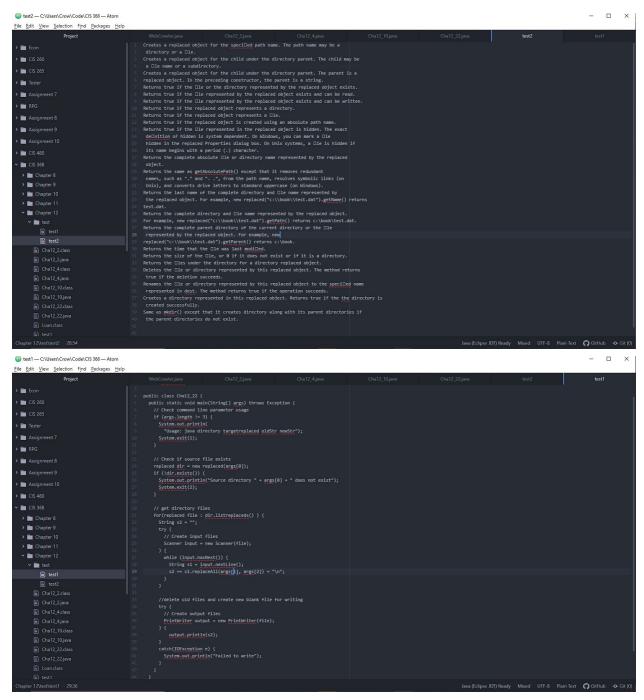
Before alteration

```
Select Windows PowerShell                                                    —   □   ✕

You are about to embark on insert great crusade for which we have striven insertse many monthly.

You are about to embark on insert great crusade for which we have striven insertse many monthly.
The eye of insert world are upon you

You are about to embark on insert great crusade for which we have striven insertse many monthly.
The eye of insert world are upon you
The hopes and prayers of liberty loving people everywhere march whith you

You are about to embark on insert great crusade for which we have striven insertse many monthly.
The eye of insert world are upon you
The hopes and prayers of liberty loving people everywhere march whith you
In company with our brave allies and broinsertr in arms on oinsertr fronts.

PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
Cha12_22.java:43: error: ';' expected
        System.out.println("Failed to write")
                                             ^
1 error
PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
Cha12_22.java:42: error: cannot find symbol
        catch(ImagingOpException e) {
             ^
  symbol:   class ImagingOpException
  location: class Cha12_22
1 error
PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test the insert
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test the insert
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test the insert
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test the insert
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test System insert
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test File replaced
true
true
PS C:\Users\Crow\code\CIS 368\Chapter 12> javac Cha12_22.java
PS C:\Users\Crow\code\CIS 368\Chapter 12> java Cha12_22 test File replaced
PS C:\Users\Crow\code\CIS 368\Chapter 12>
```

File Edit View Selection Find Packages Help

| Project | WebCrawler.java | Cha12_2.java | Cha12_4.java | Cha12_10.java | Cha12_22.java | test2 | test1 |

```
1   Creates a replaced object for the specified path name. The path name may be a
2    directory or a file.
3   Creates a replaced object for the child under the directory parent. The child may be
4    a file name or a subdirectory.
5   Creates a replaced object for the child under the directory parent. The parent is a
6   replaced object. In the preceding constructor, the parent is a string.
7   Returns true if the file or the directory represented by the replaced object exists.
8   Returns true if the file represented by the replaced object exists and can be read.
9   Returns true if the file represented by the replaced object exists and can be written.
10  Returns true if the replaced object represents a directory.
11  Returns true if the replaced object represents a file.
12  Returns true if the replaced object is created using an absolute path name.
13  Returns true if the file represented in the replaced object is hidden. The exact
14   definition of hidden is system dependent. On Windows, you can mark a file
15   hidden in the replaced Properties dialog box. On Unix systems, a file is hidden if
16   its name begins with a period (.) character.
17  Returns the complete absolute file or directory name represented by the replaced
18   object.
19  Returns the same as getAbsolutePath() except that it removes redundant
20   names, such as "." and ". .", from the path name, resolves symbolic links (on
21   Unix), and converts drive letters to standard uppercase (on Windows).
22  Returns the last name of the complete directory and file name represented by
23   the replaced object. For example, new replaced("c:\\book\\test.dat").getName() returns
24  test.dat.
25  Returns the complete directory and file name represented by the replaced object.
26  For example, new replaced("c:\\book\\test.dat").getPath() returns c:\book\test.dat.
27  Returns the complete parent directory of the current directory or the file
28   represented by the replaced object. For example, new
29  replaced("c:\\book\\test.dat").getParent() returns c:\book.
30  Returns the time that the file was last modified.
31  Returns the size of the file, or 0 if it does not exist or if it is a directory.
32  Returns the files under the directory for a directory replaced object.
33  Deletes the file or directory represented by this replaced object. The method returns
34   true if the deletion succeeds.
35  Renames the file or directory represented by this replaced object to the specified name
36   represented in dest. The method returns true if the operation succeeds.
37  Creates a directory represented in this replaced object. Returns true if the the directory is
38   created successfully.
39  Same as mkdir() except that it creates directory along with its parent directories if
40   the parent directories do not exist.
41
42
```

Chapter 12\test\test2   28:54   Java (Eclipse JDT) Ready   Mixed   UTF-8   Plain Text   GitHub   Git (0)

File Edit View Selection Find Packages Help

| Project | WebCrawler.java | Cha12_2.java | Cha12_4.java | Cha12_10.java | Cha12_22.java | test2 | test1 |

```java
3
4   public class Cha12_22 {
5     public static void main(String[] args) throws Exception {
6       // Check command line parameter usage
7       if (args.length != 3) {
8         System.out.println(
9           "Usage: java directory targetreplaced oldStr newStr");
10        System.exit(1);
11      }
12
13      // Check if source file exists
14      replaced dir = new replaced(args[0]);
15      if (!dir.exists()) {
16        System.out.println("Source directory " + args[0] + " does not exist");
17        System.exit(2);
18      }
19
20      // get directory files
21      for(replaced file : dir.listreplaceds() ) {
22        String s2 = "";
23        try {
24          // Create input files
25          Scanner input = new Scanner(file);
26        ) {
27          while (input.hasNext()) {
28            String s1 = input.nextLine();
29            s2 += s1.replaceAll(args[1], args[2]) + "\n";
30          }
31        }
32
33        //delete old files and create new blank file for writing
34        try {
35          // Create output files
36          PrintWriter output = new PrintWriter(file);
37        ) {
38            output.println(s2);
39        }
40        catch(IOException e) {
41            System.out.println("Failed to write");
42        }
43      }
44    }
```

Chapter 12\test\test1   29:36   Java (Eclipse JDT) Ready   Mixed   UTF-8   Plain Text   GitHub   Git (0)

After program runs