JAVAWORLD JAVA

AUG 29, 2019 11:40 AM PDT

About
A beginner's library for learning about essential Java programming concepts, syntax, APIs, and packages.

ADVANCED JAVA LANGUAGE FEATURES

Exceptions in Java, Part 1: Exception handling basics

Throwing, trying, catching, and cleaning up after Java exceptions

Page 3 of 3

```
public static void main(String[] args)
{
   if (args.length != 2)
   {
      System.err.println("usage: java Copy srcfile dstfile");
      return;
   }

   try
   {
      copy(args[0], args[1]);
   }
   catch (IOException ioe)
   {
      System.err.println("I/O error: " + ioe.getMessage());
   }
}

static void copy(String srcFile, String dstFile) throws IOException
{
   FileInputStream fis = null;
   FileOutputStream fos = null;
   try
   {
      fis = new FileInputStream(srcFile);
      fos = new FileOutputStream(dstFile);
      int c;
      while ((c = fis.read()) != -1)
         fos.write(c);
   }
   finally
   {
      if (fis != null)
         try
         {
            fis.close();
         }
         catch (IOException ioe)
         {
            System.err.println(ioe.getMessage());
         }

      if (fos != null)
         try
         {
            fos.close();
         }
         catch (IOException ioe)
         {
            System.err.println(ioe.getMessage());
         }
   }
}
```

The file-copying logic has been moved into a `copy()` method. This method is designed to report an exception to the caller, but it first closes each open file.

This method's `throws` clause only lists `IOException`. It isn't necessary to include `FileNotFoundException` because `FileNotFoundException` subclasses `IOException`.

Once again, the `finally` clause presents a lot of code just to close two files. In the second part of this series, you will learn about the `try-with-resources` statement, which obviates the need to explicitly close these files.

In conclusion

In this article we've focused on the basics of Java's exception-oriented framework, but there is much more to grasp, including the aforementioned `try-with-resources` statement. The second half of this tutorial introduces Java's more advanced exception-oriented language features and library types.

Jeff Friesen teaches Java technology (including Android) to everyone.

Follow