The Java™ Tutorials

Trail: Essential Classes
Lesson: Basic I/O
Section: File I/O (Featuring NIO.2)

The Java Tutorials have been written for JDK 8. Examples and practices described in this page don't take advantage of improvements introduced in later releases.

Copying a File or Directory

You can copy a file or directory by using the `copy(Path, Path, CopyOption...)` method. The copy fails if the target file exists, unless the `REPLACE_EXISTING` option is specified.

Directories can be copied. However, files inside the directory are not copied, so the new directory is empty even when the original directory contains files.

When copying a symbolic link, the target of the link is copied. If you want to copy the link itself, and not the contents of the link, specify either the `NOFOLLOW_LINKS` or `REPLACE_EXISTING` option.

This method takes a varargs argument. The following `StandardCopyOption` and `LinkOption` enums are supported:

- `REPLACE_EXISTING` – Performs the copy even when the target file already exists. If the target is a symbolic link, the link itself is copied (and not the target of the link). If the target is a non-empty directory, the copy fails with the `FileAlreadyExistsException` exception.
- `COPY_ATTRIBUTES` – Copies the file attributes associated with the file to the target file. The exact file attributes supported are file system and platform dependent, but `last-modified-time` is supported across platforms and is copied to the target file.
- `NOFOLLOW_LINKS` – Indicates that symbolic links should not be followed. If the file to be copied is a symbolic link, the link is copied (and not the target of the link).

If you are not familiar with `enums`, see Enum Types.

The following shows how to use the `copy` method:

```
import static java.nio.file.StandardCopyOption.*;
...
Files.copy(source, target, REPLACE_EXISTING);
```

In addition to file copy, the `Files` class also defines methods that may be used to copy between a file and a stream. The `copy(InputStream, Path, CopyOptions...)` method may be used to copy all bytes from an input stream to a file. The `copy(Path, OutputStream)` method may be used to copy all bytes from a file to an output stream.

The `Copy` example uses the `copy` and `Files.walkFileTree` methods to support a recursive copy. See Walking the File Tree for more information.

Previous page: Deleting a File or Directory
Next page: Moving a File or Directory