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.

The Path Class

The `Path` class, introduced in the Java SE 7 release, is one of the primary entrypoints of the `java.nio.file` package. If your application uses file I/O, you will want to learn about the powerful features of this class.

Version Note: If you have pre-JDK7 code that uses `java.io.File`, you can still take advantage of the `Path` class functionality by using the `File.toPath` method. See Legacy File I/O Code for more information.

As its name implies, the `Path` class is a programmatic representation of a path in the file system. A `Path` object contains the file name and directory list used to construct the path, and is used to examine, locate, and manipulate files.

A `Path` instance reflects the underlying platform. In the Solaris OS, a `Path` uses the Solaris syntax (`/home/joe/foo`) and in Microsoft Windows, a `Path` uses the Windows syntax (`C:\home\joe\foo`). A `Path` is not system independent. You cannot compare a `Path` from a Solaris file system and expect it to match a `Path` from a Windows file system, even if the directory structure is identical and both instances locate the same relative file.

The file or directory corresponding to the `Path` might not exist. You can create a `Path` instance and manipulate it in various ways: you can append to it, extract pieces of it, compare it to another path. At the appropriate time, you can use the methods in the `Files` class to check the existence of the file corresponding to the `Path`, create the file, open it, delete it, change its permissions, and so on.

The next page examines the `Path` class in detail.

Previous page: What Is a Path? (And Other File System Facts)
Next page: Path Operations