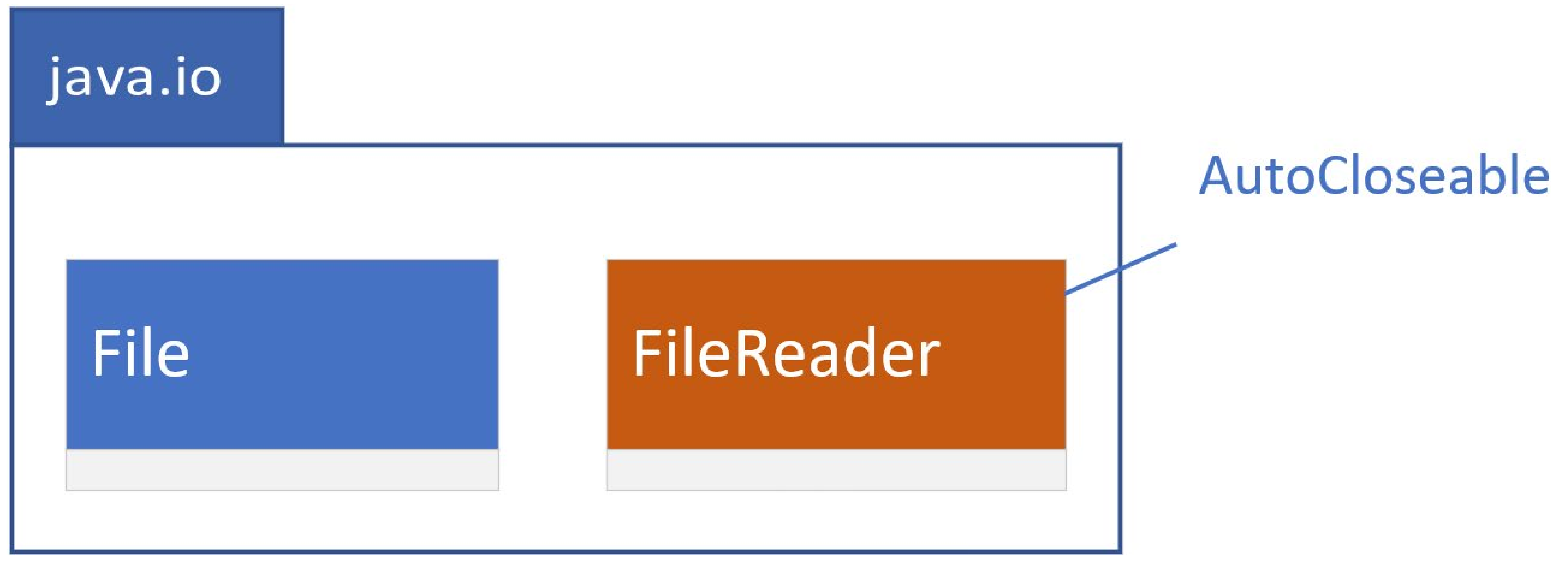
Legacy (io)



The File class and the FileReader class have been part of Java, since version 1.

The FileReader class implements the AutoCloseable interface, through it's parent class, Reader.

This class opens a file resource implicitly.

In contrast, when you create an instance of a File, you aren't actually opening that file.

Instead, you're working with something called a file handler, that lets you perform OS-like operations.

File Handle vs. File Resource

A file handle is a reference to a file that is used by the operating system to keep track of the file.

It is an abstract representation of the file, and it does not contain any of the actual data from the file.

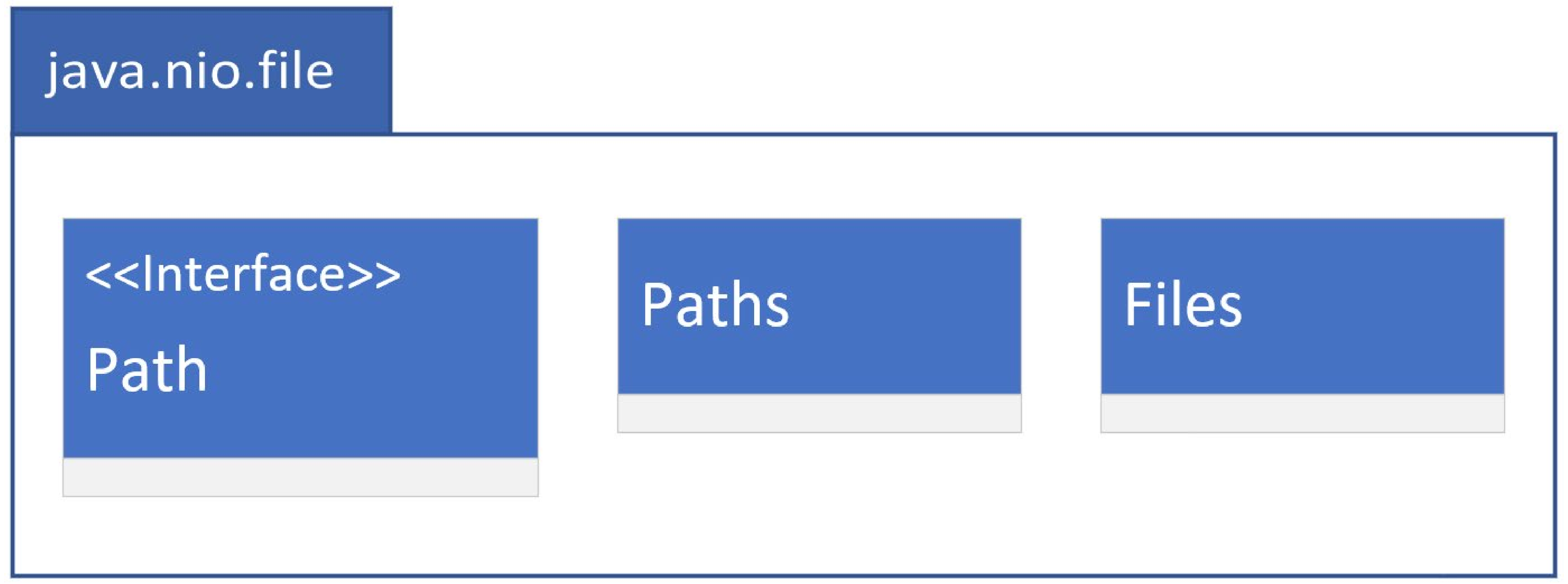
A file resource, on the other hand, is the actual data from the file.

It is stored on the disk, and it can be accessed by the operating system, and by applications.

File System Concepts

- A **directory** (or folder) is a file system container for other directories or files.
- A **path** is either a directory or a filename, and may include information about the parent directories or folders.
- A **root directory** is the top-level directory in a file system.
- The **current working directory** is the directory that the current process is working in or running from.
- An **absolute path** Includes the root (by either starting with / or optionally, C:\ in windows, where c is the root identifier.
- A **relative path** defines a path relative to the current working directory, and therefore would not start with /, but may optionally start with a dot **.** then a file separator character.

Updated (jdk7, nio2)



Path is an interface, and not a class, like the File class was.

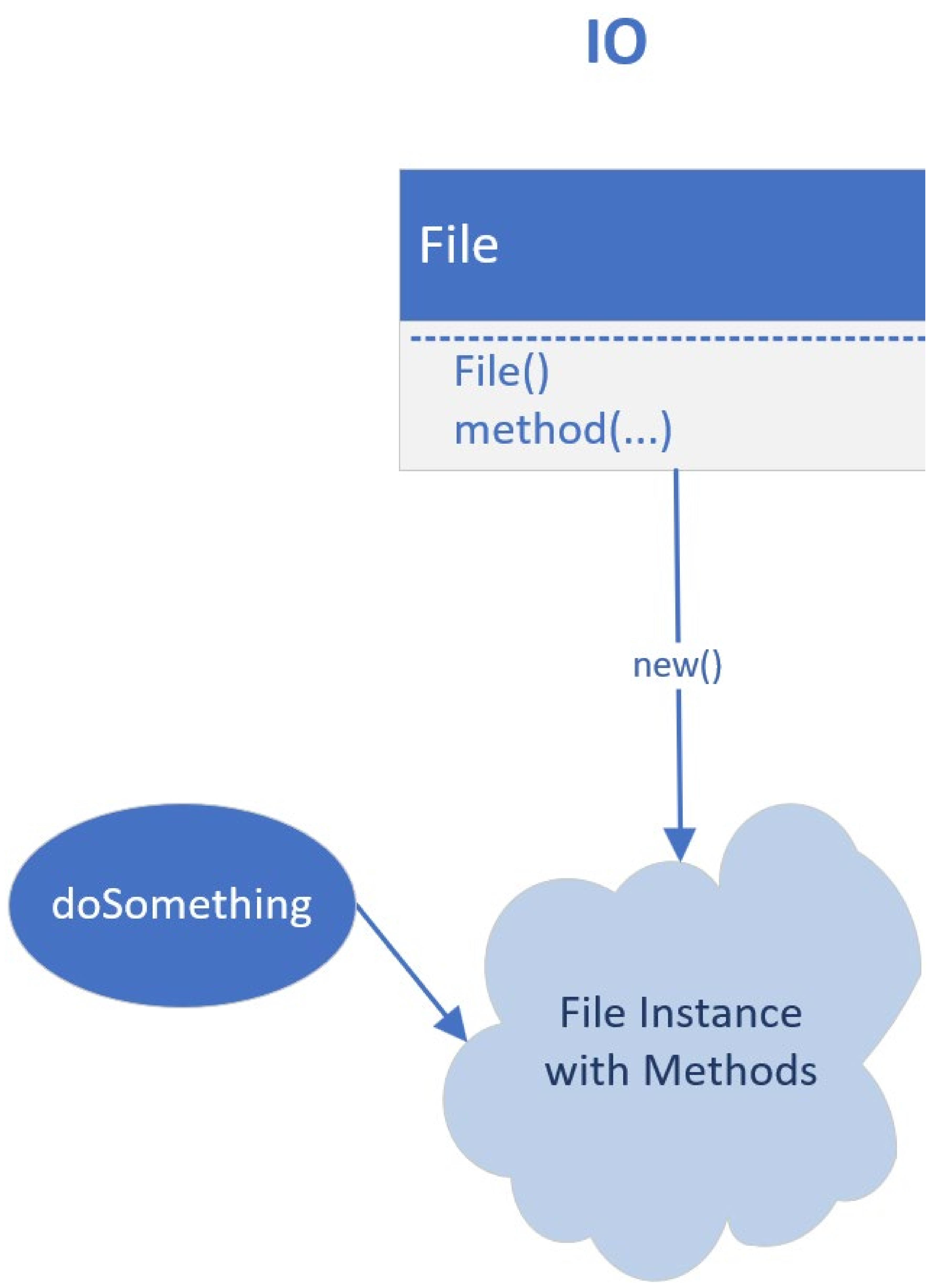
The Paths class consists exclusively of static methods that return a Path instance.

The Files class, on the other hand, has many static methods that perform operations on files and directories.

IO - Using File

When you use the File class, you get an instance, with a File constructor, and then you execute a method on that instance.

Here, behavior is a member of the instance itself.

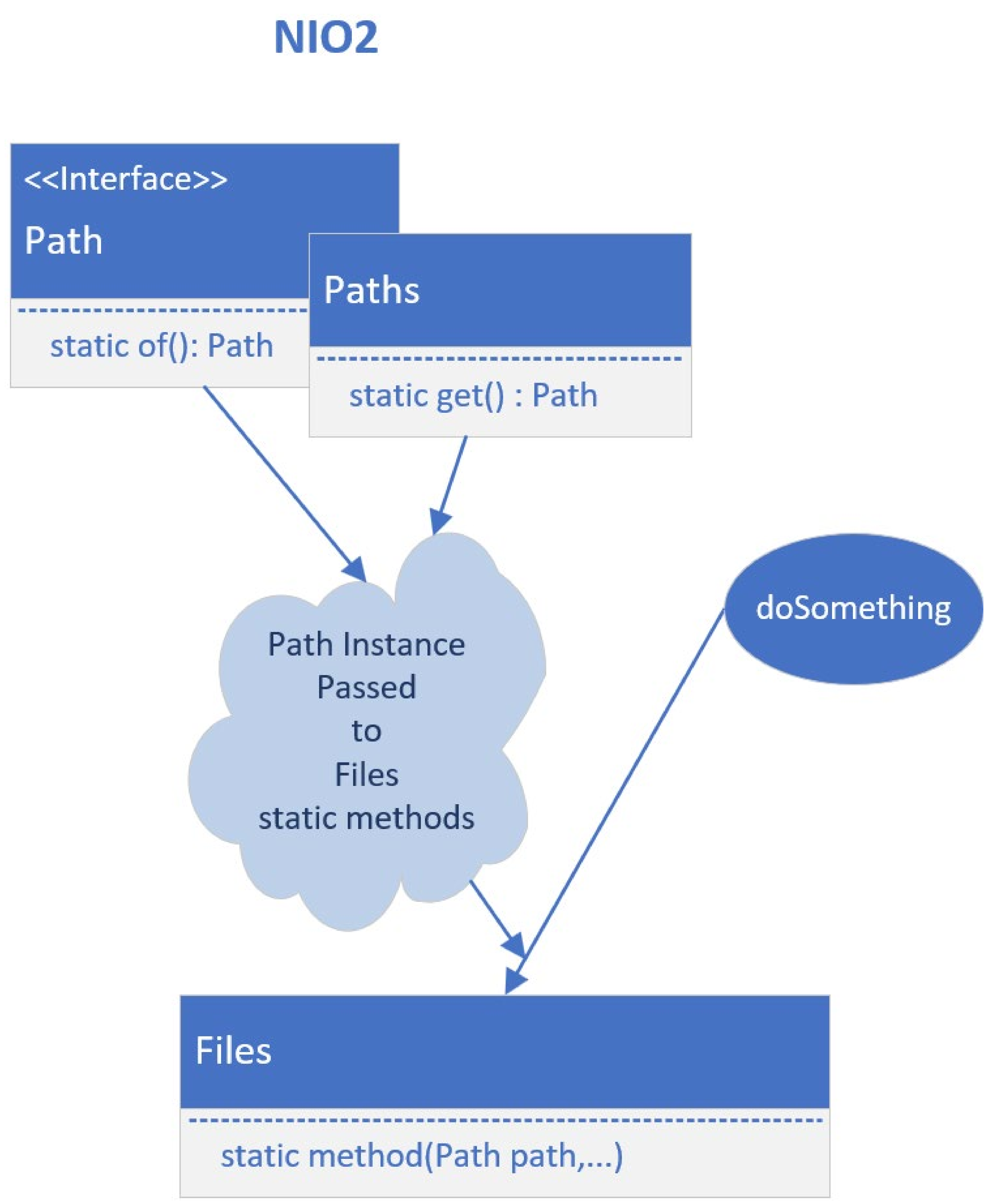


NIO2 - Using Files and Path (instead of File)

Now, if we look at how to use the NIO2 types, to do something, we first have to get an instance of Path.

We use factory methods on the Path class, or the Paths class, or other types I'll talk about later.

We then call the static method on the Files class, to do something, on the path instance passed as an argument.


NIO2
<<Interface>>
Path
static of(): Path
Paths
static get() : Path
Path Instance Passed to Files static methods
doSomething
Files
static method(Path path,...)