FILESYSTEM RESEARCH DOCUMENTATION

# File vs. Path in Java

Java provides two main classes for working with file systems: File and Path.

# java.io.File

- Part of the older IO API.

- Represents files and directory pathnames in an abstract manner.

- Can be used for file creation, deletion, and metadata (like last modified date).

Example:

File file = new File("example.txt");

if (!file.exists()) {

file.createNewFile();

}

Reference: https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/io/File.html

# java.nio.file.Path

- Introduced in Java 7 (NIO.2 API).

- More modern and flexible.

- Represents a path in the file system.

- Works well with streams and supports advanced file operations.

Example:

Path path = Paths.get("example.txt");

if (!Files.exists(path)) {

Files.createFile(path);

}

References:

- https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/nio/file/Path.html

- https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/nio/file/Files.html

# Common Operations

# Changing Directory

Using File:

File currentDirectory = new File(".");

currentDirectory = new File(currentDirectory, "subdir");

Using Path:

Path currentDirectory = Paths.get(".");

currentDirectory = currentDirectory.resolve("subdir");

# Listing Directory Contents

Using File:

File dir = new File(".");

String[] files = dir.list();

Using Path:

Files.list(Paths.get(".")).forEach(System.out::println);

# Creating Files and Directories

Using File:

new File("newfile.txt").createNewFile();

new File("newfolder").mkdir();

Using NIO:

Files.createFile(Paths.get("newfile.txt"));

Files.createDirectory(Paths.get("newfolder"));

# Symbolic Links

Java NIO supports symbolic links:

Path link = Paths.get("link.txt");

Path target = Paths.get("target.txt");

Files.createSymbolicLink(link, target);

Note: Requires elevated permissions on some systems.

More on symbolic links: https://www.baeldung.com/java-symbolic-links

# Absolute vs. Relative Paths

- Relative Path: relative to the current working directory.

- Absolute Path: the full path from the root directory.

Example:

Path relative = Paths.get("subdir/file.txt");

Path absolute = relative.toAbsolutePath();

# File Attributes

Reading file attributes using NIO:

BasicFileAttributes attrs = Files.readAttributes(path, BasicFileAttributes.class);

System.out.println("Size: " + attrs.size());

System.out.println("Created: " + attrs.creationTime());

Reference: https://www.baeldung.com/java-file-attributes

# Useful Libraries and Resources

- Baeldung Java IO Guide: https://www.baeldung.com/java-io

- GeeksforGeeks Java File Handling: https://www.geeksforgeeks.org/file-handling-in-java/

- W3Schools Java Files: https://www.w3schools.com/java/java\_files.asp

- Official Java Tutorials: https://docs.oracle.com/javase/tutorial/essential/io/