I/O - Files Methods

delete files

deleteIfExists()

It throws a checked exception. Mind the `IOException`!

```
  public static boolean deleteIfExists(Path path)
                                  throws IOException
```

delete

delete()

It throws a checked exception. Mind the `IOException`!

```
  Files.delete(path);
```

delete

Browsing files

browsing

list

It provides a simple and flat listing of the files and directories within the specified directory.
It throws a checked exception. Mind the `IOException`!

```
  try (Stream<Path> stream = Files.list(tmp)) {...}
```

find

Mind the `maxDepth` parameter!

```
public static Stream<Path> find(Path start,
 int maxDepth,
 BiPredicate<Path,BasicFileAttributes> matcher,
 FileVisitOption... options)
```

walk

This method returns a Stream that is a recursive listing of all paths in the directory and its subdirectories. It traverses the directory tree recursively, including all levels of subdirectories.

```
public static Stream<Path> walk(Path start,
   FileVisitOption… options) throws IOException

public static Stream<Path> walk(Path start, int maxDepth,
   FileVisitOption… options) throws IOException
```

walk

createDirectories

```
var dir = Path.of("/flip");
dir = Files.createDirectories(dir);
```

Unlike the `createDirectory()` method, an exception is not thrown if the directory could not be created because it already exists.
It is *createDirectories()* and **NOT** *mkdir()*.
createDirectories

Read lines

read lines

readAllLines

Mind the `IOException` !

```
//this returns a list, everything is in memory
List<String> listOfLines = Files.readAllLines(path);
```

lines

Mind the `IOException` !

```
//this returns a Stream (not in memory)
Stream<String> stream = Files.lines(pom)
```

isSameFile

IsSameFile

```
boolean result = Files.isSameFile(p1, p2);
```

The system might check if the files really exist if the `p1.equals(p2)` returns false.
If check is done and one of these two do not exist, after check:

```
Exception in thread "main" java.nio.file.NoSuchFileException: src/a
```

Mismatch

Finds and returns the position of the first mismatched byte in the content of two files, or -1L if there is no mismatch.

```
public static long mismatch(Path path,
 Path path2) throws IOException
```

```
Path hello = Path.of("/tmp/hello.txt");
long mismatch = Files.mismatch(hello, hello);  //-1
```

File Attributes

- BasicFileAttributes
- BasicFileAttributeView
  File Attributes