

Acesso ao sistema de arquivos

Bibliografia

ANDRADE, A. G. P. D. **Java IO, Java NIO e NIO.2: Quando Utilizar?**, 2017. Disponível em:
<<https://medium.com/@antonio.gabriel/java-io-java-nio-e-nio-2-quando-utilizar-8c900b1c57a1>>.

Sistema de arquivos

- O **sistema de arquivos** é o sistema utilizado pelo sistema operacional para armazenar, organizar e acessar os dados que estão armazenados no computador.
- Utilizam o sistema de arquivos os discos (rígidos ou SSD), mídias óticas, pendrive, etc.

Categorias das operações de E/S em arquivos

- As operações de E/S em arquivos são divididas em duas categorias:

Operações para acesso ao sistema de arquivos

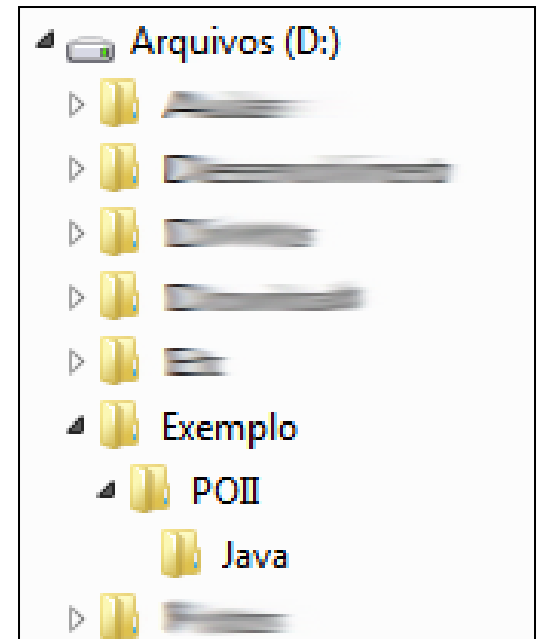
- Obter a relação de arquivos e subdiretórios de um diretório
- Ler propriedades de um arquivo ou diretório
- Criar diretórios, apagar diretórios, renomear arquivos, etc.

Operações de leitura e edição de conteúdo de arquivos

- Ler o conteúdo de arquivos
- Alterar (gravar) dados em arquivos

Diretórios

- **Caminho:** local (diretório) de um arquivo
 - Utiliza-se um caractere separador para expressar o caminho de um arquivo
 - barra (/) ou
 - barra invertida (\)
- **Caminho absoluto**
 - Contém todos os diretórios desde a raiz
 - Exemplo: D:\Exemplo\POII\Java\Slide1.pdf
- **Caminho relativo**
 - O caminho é relativo ao diretório atual
 - Exemplo: Java\Slide1.pdf



Acesso ao sistema de arquivos

A classe **File** representa um arquivo ou um diretório.

File
+ separatorChar : char = fs.getSeparator() + separator : String = ""+separatorChar + pathSeparatorChar : char = fs.getPathSeparator() + pathSeparator : String = ""+pathSeparatorChar
+ File(pathname : String) + getName() : String + getParent() : String + getParentFile() : File + getPath() : String + isAbsolute() : boolean + getAbsolutePath() : String + getAbsoluteFile() : File + getCanonicalPath() : String + getCanonicalFile() : File + canRead() : boolean + canWrite() : boolean + exists() : boolean + isDirectory() : boolean + isFile() : boolean + isHidden() : boolean + lastModified() : long + length() : long + createNewFile() : boolean + delete() : boolean + list() : String[] + listFiles() : File[] + mkdir() : boolean + mkdirs() : boolean + renameTo(dest : File) : boolean + setLastModified(time : long) : boolean + setReadOnly() : boolean + setWritable(writable : boolean) : boolean + setReadable(readable : boolean) : boolean + setExecutable(executable : boolean) : boolean + canExecute() : boolean + listRoots() : File[] + getTotalSpace() : long + getFreeSpace() : long + getUsableSpace() : long + createTempFile(prefix : String, suffix : String) : File

Membro	Descrição
File ()	Cria um objeto que representa um arquivo ou um diretório, que pode ou não existir
getName ()	Retorna o nome do arquivo ou diretório
exists ()	Retorna true se o arquivo/diretório existe
isDirectory ()	Retorna true se o objeto representar um diretório
isFile ()	Retorna true se o objeto representar um arquivo
length ()	Retorna o tamanho, em bytes, do arquivo (válido apenas para arquivos)
listFiles ()	Retorna um array com arquivos e diretórios contidos no diretório representado pelo objeto
createTempFile ()	Cria um arquivo no diretório temporário

Exemplo

```
File diretorio = new File("C:\\Windows");

File[] conteudoDiretorio = diretorio.listFiles();
for (File item : conteudoDiretorio) {
    if (item.isDirectory()) {
        System.out.println("Diretório: " + item.getName());
    } else {
        System.out.println("Arquivo " + item.getName() +
                           "tem " + item.length() + " bytes");
    }
}
```