Accés a dades

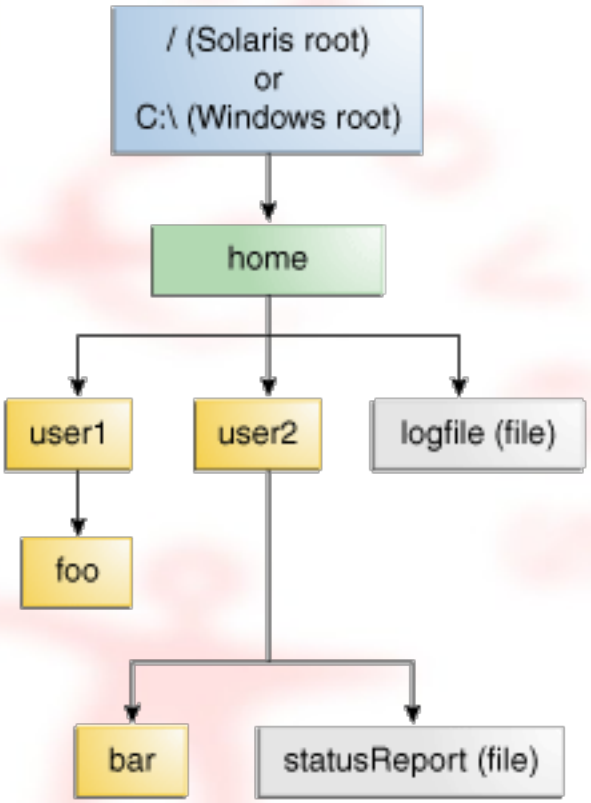
Sistema de fitxers

Sistemes de fitxers

El sistema de fitxers és un mètode per emmagatzemar i organitzar fitxers d'ordinador i les dades que contenen per tal de facilitar-ne la localització i accés.

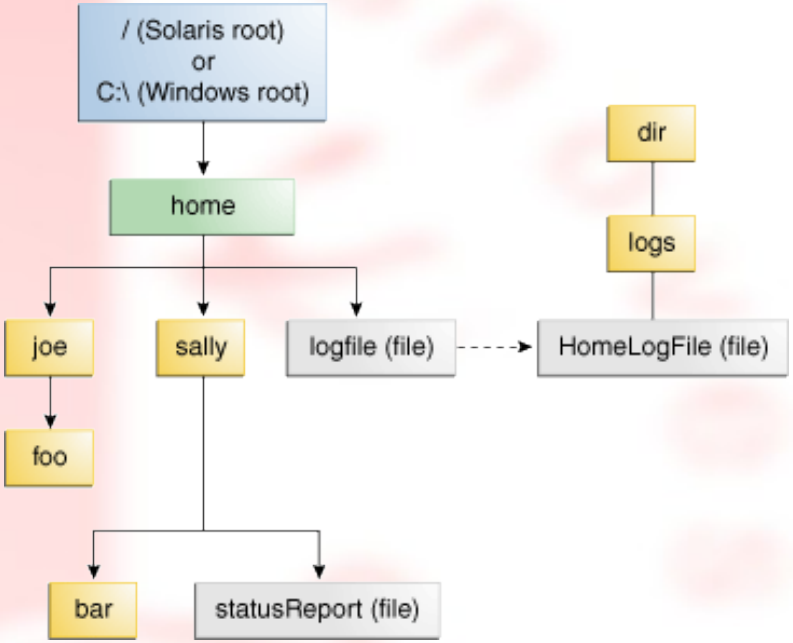
Generalment un sistema d'arxius té directoris que associen noms d'arxius amb arxius, normalment connectant el nom d'arxiu a un índex en una taula d'assignació d'arxius d'algun tipus, com ara FAT en sistemes d'arxius MS-DOS o els inodes dels sistemes Unix.

En alguns sistemes de fitxers els noms d'arxius són estructurats, amb sintaxis especials per a extensions d'arxius i números de versió. En altres, els noms d'arxius són simplement cadenes de text i les metadades de cada arxiu són allotjades separadament.



Rutes

En sistemes d'arxius jeràrquics, normalment, es declara la ubicació precisa d'un arxiu amb una cadena de text anomenada "ruta". La nomenclatura per a rutes varia lleugerament de sistema en sistema, però mantenen en general una mateixa estructura. Una ruta ve donada per una successió de noms de directoris i subdirectoris, ordenats jeràrquicament d'esquerra a dreta i separats per algun caràcter especial que sol ser una barra ('/') o barra invertida ('\') i pot acabar amb el nom d'un arxiu present en l'última branca de directoris especificada.



Un fitxer s'identifica de forma única amb la seva ruta. Una ruta pot ser **absoluta**, quan conté l'arrel del sistema, o **relativa** quan conté només part de la ruta, o l'inici de la ruta no és l'arrel.

Al sistema de fitxers podem trobar arxius especials que serveixen de referència a altres arxius, els **enllaços simbòlics**. Qualsevol operació feta sobre l'enllaç es porta a terme amb el fitxer o directori que enllaça, excepte l'esborrat o el canvi de nom. Normalment són transparents a l'usuari.

La classe *Path*

Aquesta classe representa una ruta dins del sistema de fitxers. Pot fer referència a un arxiu, a un directori o no existir. Utilitza la notació pròpia del sistema de fitxers.

Crear un objecte de la classe Path

La forma més senzilla és utilitzant la classe auxiliar Paths. Per a cada nivell de directoris guarda un element que el representa.

```
Path p1 = Paths.get("/tmp/foo");
Path p2 = Paths.get(args[0]);
Path p3 = Paths.get(URI.create("file:///Users/joe/FileTest.java"));
```

Recuperar informació del Path

Si tenim el path */home/joe/foo* llavors

- .toString torna "*/home/joe/foo*"
- .getFileName torna *foo*
- .getName(index) torna l'element de la ruta d'aquesta posició. getName(0) torna *home*.
- .getNameCount torna quants elements hi ha al path.
- subpath(inici, fi) torna el path entre la posició inici i la fi. SubPath(0,2) torna home/joe

La classe *Files*

És una classe d'utilitat que ens permet llegir, escriure i manipular fitxers i directoris.

Per exemple:

- Consultes
  - .isRegularFile: true si el path que rep com a argument és un fitxer, false si és un directori o un enllaç.
  - .isReadable
  - .isWriteable
  - .isExecutable
  - .isSameFile: true si els dos paths que rep com a arguments representen el mateix objecte.
- Esborrar fitxers o directoris:
  - Files.delete(Path): Esborra l'objecte representat per el path.

- Esborrar fitxers o directoris:
  - Files.delete(Path): Esborra l'objecte representat per el path.
- Copiar fitxers o directoris:
  - Files.copy(origen, destí, opcions): Origen i destí són Paths i opcions és un vararg per a les constants REPLACE_EXISTING, COPY_ATTRIBUTES, NOFOLLOW_LINKS.
- Moure fitxers o directoris:
  - Files.move(origen, destí, opcions): Origen i destí són Paths i opcions és un vararg per a les constants REPLACE_EXISTING, ATOMIC_MOVE.
- Recuperar atributs del Path:
  - .size en bytes
  - .isDirectory
  - .isRegularFile
  - .isHidden
  - .get/setLastModifiedTime
  - .get/setOwner

- Llegir i escriure continguts del fitxer:
  - Files.write(Path, bytes[], opcions): Escriu el contingut de l'array de bytes al fitxer. Dins les opcions podem decidir com s'obre el fitxer:
    - WRITE
    - CREATE: L'obre si existeix i si no el crea.
    - APPEND: Juntament amb els anteriors l'obre al final per afegir dades.
- Mètodes amb buffer per fitxers de text:
  - .newBufferedReader
  - .newBufferedWriter
- Exemple

```
String s = "Hello World! ";
byte data[] = s.getBytes();
Path p = Paths.get("./logfile.txt");
try (OutputStream out = new BufferedOutputStream(
  Files.newOutputStream(p, CREATE, APPEND))) {
  out.write(data, 0, data.length);
} catch (IOException x) {
  System.err.println(x);
}
```

- Crear i llegir directoris:
  - Files.createDirectory(Path): Crea el nou directori.
  - Files.newDirectoryStream: Torna un stream on cada element és un path de dins del directory.

```
Path dir = ...;
try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir)) {
    for (Path file: stream) {
        System.out.println(file.getFileName());
    }
}
```