Entrada/salida con java nio

Paquete java.nio.files

- Nuevo paquete de entrada/salida para lectura y escritura en ficheros
- Principales clases e interfaces de este paquete:
  - Path. Representa una ruta a un fichero o directorio
  - Paths. Clase utilizada para crear instancias de Path
  - Files. Dispone de diversos métodos estáticos para operar contra un fichero o directorio

Interfaz Path

- Representa una ruta a un fichero o directorio.
- Para crear una implementación:
  - Método *of* de Path:

```
Path pt=Path.of("/users/mydata.txt");
```

  - Método *get* de Paths:

```
Path pt=Paths.get("/users/mydata.txt");
```

Métodos de Path I

- Proporciona métodos para obtener información sobre la ruta:
  - Path getFileName(). Nombre del fichero o último elemento del Path

```
Path p1=Path.of("/user/mydata.txt");
Path p2=Path.of("/a/b");
System.out.println(p1.getFileName()); //mydata.txt
System.out.println(p2.getFileName()); //b
```

  - Path toAbsolutePath(). Ruta completa del fichero o directorio

```
Path p1=Path.of("c:\\user\\mydata.txt");
Path p2=Path.of("datos.txt");
System.out.println(p1.toAbsolutePath ()); // c:\user\mydata.txt
System.out.println(p2. toAbsolutePath()); //c:\ejercicios\...\datos.txt
```

ruta completa desde el directorio actual

Métodos de Path II

- **Path normalize(). Resuelve las rutas relativas y devuelve el path normalizado**

```
String url="c:\\temp\\..\\data.txt";
Path p1=Paths.get(url);
System.out.println(p1.normalize()); //c:\data.txt
```

- **Path relativize(Path other). Devuelve la ruta relativa de other respecto al path principal:**

```
Path p1=Path.of("c:\\temp\\mydata.txt");
Path p2=Paths.get("c:\\temp\\..\\data.txt");
System.out.println(p1.relativize(p2)); //..\..\data.txt
```

•Si uno es ruta absoluta y el otro no, se producirá una excepción

Al aplicar relativize, las dos rutas son internamente normalizadas primero

Revisión conceptos

Indica que se mostrará por pantalla tras la ejecución del siguiente código:

```
Path p1=Path.of("c:\\user\\.\\..\\texto.txt");
Path p2=Path.of("c:\\user\\..\\datos.txt");
System.out.print (p1.normalize().toAbsolutePath ());
System.out.println(p1. relativize(p2));
```

A. texto.txt – datos.txt
B. c:\texto.txt – ..\datos.txt
C. c:\texto.txt – c:\..\datos.txt
D. c:\user\texto.txt – ..\datos.txt

Respuesta

La respuesta correcta es la B. Dado que ambos ficheros cuelgan de la misma carpeta, la dirección relativa de uno a otro será ..\\fichero

Métodos de Path III

- Path resolve (Path other). Resuelve la ruta de other frente a la principal

```
Path p1=Paths.get(" c:\\temp\\..\\data.txt");
Path p2=Paths.get("new.txt");
System.out.println(p1.resolve(p2)); //c:\temp\..\data.txt\new.txt
```

- Si other es ruta absoluta se devuelve esa misma ruta

- int getNameCount(). Devuelve el número de elementos del path, sin incluir el directorio raíz

```
Path p1=Path.of("c:\\temp\\..\\mydata.txt");
System.out.println(p1.getNameCount()); //3
```

- Path getName(int index). Devuelve la parte del path que ocupa la posición indicada. El primer elemento (sin incluir la raíz) tiene índice 0

```
Path p1=Path.of("c:\\temp\\..\\mydata.txt");
System.out.println(p1.getName(2)); //mydata.txt
```

Revisión conceptos

Indica que se mostrará por pantalla tras la ejecución del siguiente código:

```
Path p1=Path.of("c:\\user\\.\\..\\texto.txt");
Path p2=Path.of("..\\datos.txt");
System.out.print(p1.resolve(p2).getName(4));
```

A. .
B. datos.txt
C. ..
D. Excepción

Respuesta

La respuesta correcta es la C. Dado que p2 es relativa, el resultado es la unión de las dos rutas:
c:\user\.\..\texto.txt\..\datos.txt
y en la posición número 4 nos encontramos ..

Lectura de un fichero con Files

- La clase Files proporciona los siguientes métodos estáticos para leer el contenido de un fichero:

  - Stream <String> lines(Path path). Devuelve un Stream con todas las líneas del fichero. A partir de ahí, se pueden aplicar los métodos de streams para realizar búsquedas, transformaciones, filtrados, etc.

  - List<String> readAllLines(Path path). Devuelve una lista con las cadenas del fichero, donde cada elemento corresponde con una línea.

```
Path p1=Path.of("c:\\user\\mydata.txt");
List<String> datos=Files.readAllLines(p1);
datos.forEach(s->System.out.println(s));
```

Imprime todas las líneas del fichero

  - BufferedReader newBufferedReader(Path pt). Devuelve un objeto BufferedReader para realizar la lectura de forma clásica.

Existen variantes de estos tres métodos que reciben como segundo parámetro un objeto Charset

Escritura en ficheros con Files I

➢Para realizar la escritura en ficheros, la clase Files proporciona los siguientes métodos:

- writeString(Path path, CharSequence csq, Charset cs, OpenOption... options). Escribe en el fichero indicado como primer parámetro, la cadena especificada en el segundo, utilizando el juego de caracteres del tercero y las opciones de escritura del cuarto:

```
try{
    Path p1=Path.of("c:\\user\\mydata.txt");
    Files.writeString(p1, "nuevo texto",
                                Charset.forName("UTF-8"),
                                 StandardOpenOption.APPEND);
 }
 catch(IOException ex){
          ex.printStackTrace();
 }
```

Escritura en modo append

Escritura en ficheros con Files II

- write(Path path, Iterable<? extends CharSequence>, Charset cs, OpenOption... options). Escribe en el fichero indicado como primer parámetro, la colección de cadenas especificada en el segundo, utilizando el juego de caracteres del tercero y las opciones de escritura del cuarto:

```
List<String> cadenas=List.of("lunes","martes","miércoles","jueves","viernes");
try{
    Path p1=Path.of("c:\\user\\mydata.txt");
    Files.write(p1, cadenas,
                                Charset.forName("UTF-8"),
                                StandardOpenOption.APPEND);
 }
 catch(IOException ex){
          ex.printStackTrace();
 }
```

Revisión conceptos

Dada la variable String FICHERO, que contiene la ruta de un archivo de texto, escribe dos bloques de código diferentes que impriman todas las líneas de fichero que tengan más de 10 caracteres

Respuesta

Opción 1:

```
Path pt=Path.of(FICHERO);
Files.lines(pt)
    .filter(s->s.length()>10)
    .forEach(System.out::println);
```

Opción 2:

```
Path pt=Path.of(FICHERO);
Files.readAllLines(pt)
    .stream()
    .filter(s->s.length()>10)
    .forEach(System.out::println);
```

Otros métodos de Files I

- static Path copy(Path source, Path target, CopyOption... options). Copia el contenido de un fichero en otro:
  - Si el fichero target ya existe y no se indica opción, se produce una excepción. Aunque si ambas rutas son iguales, el método se ejecuta sin cambios
  - Si source es un directorio, se creará en target un directorio vacío
  - Si target es un directorio, FileAlreadyExistsException
  - Si el tercer parámetro es StandardCopyOption.REPLACE_EXISTING, en fichero target será sustituido en caso de que exista
  - Si el tercer parámetro incluye StandardCopyOption.COPY_ATTRIBUTES, se copiarán también las propiedades del fichero origen en el destino

Otros métodos de Files II

- static Path move(Path source, Path target, CopyOption... options). Mueve un fichero origen a otro destino:
  - Si el fichero target ya existe y no se indica opción, se produce una excepción. Aunque si ambas rutas son iguales, el método se ejecuta sin cambios
  - Si source es un directorio, se creará en target un directorio vacío
  - Si target es un directorio, FileAlreadyExistsException
  - Si el tercer parámetro es StandardCopyOption.REPLACE_EXISTING, en fichero target será sustituido en caso de que exista

Otros métodos de Files III

- static void delete(Path path). Elimina el fichero si existe, si no se produce una excepción. Si es un directorio, deberá estar vacío

- static void deleteIfExists(Path path). Elimina el fichero si existe, sino no hace nada. Si es un directorio, deberá estar vacío

- static Path createFile(Path path, FileAttribute<?>... attrs). Crea el fichero indicado vacío. Si el fichero ya existe, se produce una excepción

Revisión conceptos

Los ficheros "test1.txt" y "test2.txt" contienen las cadenas de texto "hello" y "by", respectivamente. Indica que ocurrirá al ejecutar el siguiente código

```
Path p1=Path.of("c:\\test1.txt");
Path p2=Path.of("c:\\test2.txt");
Files.copy(p1,p2,StandardCopyOption.REPLACE_EXISTING);
Files.lines(p2).forEach(System.out::println);
```

A. Se imprime "hello"
B. Se imprime "by"
C. Se imprime "helloby"
D. Se produce una excepción

Respuesta

La respuesta correcta es la A, ya que el contenido del fichero "test1.txt" se mueve a "test2.txt", eliminando el existente.