Лабораторна робота №12

Тема: Застосування класів Java для роботи з файловою системою. Основні операції над файлами та каталогами.

Мета: Навчитися створювати та реалізовувати додатки з використанням класів Java для виконання операцій над файлами та каталогами.

Обладнання: ОС Windows, інтегроване середовище JetBrains IntelliJ IDEA.

1. Теоретичні відомості

Для роботи з фізичними файлами та каталогами (директоріями), розміщеними на зовнішніх носіях, у додатках Java використовуються класи File (пакет java.io) та Files (пакет java.nio.file).

Клас File існує з найпершої версії мови. Він містить базові методи доступу до файлів і каталогів, але не містить методів для читання та запису файлів. Клас Files доступний, починаючи з версії Java 7. Він, як і клас File, містить методи доступу до файлів і каталогів, але додатково дозволяє писати та читати файли, працювати з символічними посиланнями, а також копіювати та переміщати файли. Крім того, клас Files містить методи для обходу дерева каталогів, дозволяючи видаляти чи копіювати цілу структуру каталогів.

Класи File і Files використовують безліч допоміжних класів та інтерфейсів. Щоб не було проблем з імпортом, потрібно додати на початок програми наступні інструкції імпорту всіх класів із пакетів (залежно від використовуваного класу):

```
import java.io.*;
```

для класу File або:

```
import java.nio.file.*;

import java.nio.file.attribute.*;
```

для класу Files.

Класи активно використовують контрольовані винятки (особливо клас Files), тому необхідно або обробити ці винятки за допомогою інструкції `try...catch`, або додати клас виключення після ключового слова `throws` в заголовок методу, наприклад:

```
public static void main(String[] args) throws Exception {

// ... }
```

1.1 Клас File

Об'єкт класу File створюється одним з наведених нижче конструкторів:

- `File(String path)` - вказується шлях до файлу;
- `File(File dir, String name)` - вказується об'єкт класу File (каталог) та ім'я файлу;

– `File(String dirPath, Sring name)` – вказується шлях до файлу та ім'я файлу;

– `File(URI uri)` – вказується об'єкт URI, який відповідає адресі в Інтернеті.

При створенні об'єкта класу File будь-яким із конструкторів компілятор не виконує перевірку існування фізичного файла із заданим шляхом.

Методи класу File наведені в таблиці 1.1.

Таблиця 1.1.

Метод	**Опис**
boolean isDirectory()	Чи є об'єкт файлу директорією
boolean isFile()	Чи є об'єкт файлом
long length()	Повертає розмір/довжину файлу в байтах.
boolean createNewFile()	Створює файл. Якщо файл вже був, повертає false.
boolean mkdir()	Створює директорію. Назва mkdir походить від make directory.
boolean mkdirs()	Створює директорію та всі піддиректорії.
boolean delete()	Видаляє файл об'єкта на диску. Якщо об'єкт - директорія, то тільки якщо в ній немає файлів.
void deleteOnExit()	Додає файл до спеціального списку файлів, які будуть автоматично видалені під час закриття програми.
File createTempFile(String prefix, String suffix, File directory)	Створює "тимчасовий файл" - файл з випадково згенерованим унікальним ім'ям - щось типу "dasd4d53sd". Додаткові параметри – префікс до імені, суфікс (закінчення). Якщо директорія не вказана, файл створюється в спеціальній директорії ОС для тимчасових файлів

Метод	Опис
boolean exists()	Повертає true, якщо файл з такою назвою існує на диску комп'ютера.
String getAbsolutePath().	Повертає повний шлях файлу з усіма піддиректоріями
String getCanonicalPath()	Повертає канонічний шлях файлу. Наприклад, перетворює шлях "c:/dir/dir2/../a.txt" до шляху "c:/dir/a.txt"
String[] list()	Повертає масив імен файлів, що містяться в директорії, якою є поточний об'єкт-файл.
File[] listFiles()	Повертає масив файлів, які містяться в директорії, якою є поточний об'єкт-файл.
long getTotalSpace()	Повертає розмір диска (кількість байт), на якому розташований файл.
long getFreeSpace()	Повертає кількість вільного місця (кількість байт) на диску, на якому розташований файл.
boolean renameTo(File)	Перейменовує файл - вміст файлу фактично отримує нове ім'я. Тобто. можна перейменувати файл "c:/dir/a.txt" на "d:/out/text/b.doc".
String getName()	Повертає лише ім'я файлу, без шляху.
String getParent()	Повертає тільки шлях (директорію) до поточного файлу, без імені.
Path toPath()	Повертає об'єкт Path, який відповідає поточному об'єкту File
boolean canRead()	файл доступний для читання
boolean canWrite()	файл доступний для запису

Розгляньмо приклад отримання інформації про файл:

```
import java.io.File;

public class FileExample {
 public static void main(String[] args) {
 File file = new File("src/io");

 System.out.println("Ім'я файлу: " + file.getName());
 System.out.println("Шлях: " + file.getPath());
 System.out.println("Абсолютний шлях: " +
file.getAbsolutePath());
 System.out.println("Батьківський каталог: " +
file.getParent());
 System.out.println(file.exists() ? "Файл/каталог існує." :
"Файл/каталог не існує.");
 System.out.println(file.canWrite() ? "Файл/каталог доступний
для редагування." : "Файл/каталог недоступний для редагування.");
 System.out.println(file.canRead() ? "Файл/каталог доступний
для читання." : "Файл/каталог не доступний для читання.");
 System.out.println((file.isDirectory() ? "Каталог." : "Не
каталог."));
 System.out.println(file.isFile() ? "Файл." : "Не файл.");
 System.out.println(file.isAbsolute() ? "Абсолютний шлях." :
"Не абсолютний шлях.");
 System.out.println("Дата останнього редагування: " +
file.lastModified());
 System.out.println("Розмір: " + file.length() + " байт.");
 }
}
```

Каталог - це також об'єкт класу File, який містить список інших файлів та каталогів. Після створення об'єкта класу File, що є каталогом, його метод isDirectory() поверне значення true.

Для створення каталогу можна використовувати метод mkdir(), який поверне true у успішному випадку. Якщо цей шлях вже існує або каталог не можна створити через відсутність повного шляху до нього, то повернеться false.

Метод mkdirs() створює як каталог, так і всіх його батьків.

Якщо потрібно отримати вміст каталогу, можна викликати метод list() без аргументів. Він поверне повний список (масив) імен файлів та каталогів, що містяться у цьому каталозі.

Є ще схожий метод listFiles(), який повертає масив файлів (об'єктів, а не їхніх імен).

Приклад отримання вмісту каталога:

```
import java.io.File;
public class DirList {
    public static void main(String[] args) {
```

```
        String catalogName = "src";
        File catalog = new File(catalogName);

         if (catalog.isDirectory()) {
         System.out.println("Папка" + catalogName);
         String[] list = catalog.list();
         if (list! = null) {
            for (String fileName : list) {
            File file = new File(catalogName + "/" + fileName);
         if (file.isDirectory()) {
            System.out.printf("\t%s каталог%n", fileName);
         } else {
            System.out.printf("\t%s файл%n", fileName);
              }
            }
          }
         } else {
            System.out.println(catalogName + "не є каталогом");
         }
      }
   }
```

1.2 Клас Files

Усі методи цього класу статичні та працюють з об'єктами типу Path. Методів дуже багато, тому в таблиці 1.2 наведені лише основні:

Таблиця1.2.

Метод	**Опис**
Path createFile(Path path)	Створює новий файл за допомогою path
Path createDirectory(Path path)	Створює нову директорію
Path createDirectories(Path path)	Створює кілька директорій
Path createTempFile(prefix, suffix)	Створює тимчасовий файл
Path createTempDirectory(prefix)	Створює тимчасову директорію
void delete (Path path)	Видаляє файл або директорію, якщо вона порожня
Path copy(Path src, Path dest)	Копіює файл
Path move (Path src, Path dest)	Переміщує файл

Метод	Опис
boolean isDirectory(Path path)	Перевіряє, що шлях – це директорія, а не файл
boolean isRegularFile(Path path)	Перевіряє, що шлях – це файл, а не директорія
boolean exists(Path path)	Перевіряє, що об'єкт по заданому шляху існує
long size(Path path)	Повертає розмір файлу
byte[] readAllBytes(Path path)	Повертає весь вміст файлу у вигляді масиву байт
String readString(Path path)	Повертає весь вміст файлу у вигляді рядка
List<String> readAllLines(Path path)	Повертає весь вміст файлу у вигляді списку рядків
Path write(Path path, byte[])	Записує у файл масив байт
Path writeString(Path path, String str)	Записує у файл рядок
DirectoryStream<Path> newDirectoryStream(Path dir)	Повертає колекцію файлів (і піддиректорій) із заданої директорії

Приклади створення файлів та каталогів методами класу Files.
Створити файл:

```
Files.createFile(Path.of("c:\readme.txt"));
```

Створити директорію:

```
Files.createDirectory(Path.of("c:\\test"));
```

Створити директорію і всі необхідні піддиректорії, якщо їх не існує:

```
Files.createDirectories(Path.of("c:\\test\\1\\2\3"));
```

Приклади копіювання, переміщення та видалення файлів методами класу Files.
Копіювання файлу:

```
Path path1 = Path.of("c:\readme.txt");
Path path2 = Path.of("c:\readme-copy.txt");
Files.copy(path1, path2);
```

Переміщення та перейменування файлу:

```
Path path1 = Path.of("c:\readme.txt");
Path path2 = Path.of("d:\readme-new.txt");
```

```
Files.move(path1, path2);
```

Видалення файлу:

```
Path path = Path.of("d:\readme-new.txt");
Files.delete(path);
```

Методами класу Files можливо перевірити тип файлу та факт його існування, а також визначити розмір файлу.

Приклад читання вмісту файлу у вигляді списку рядків:

```
Path path = Path.of("c:\\readme.txt");
List<String> list = Files.readAllLines(path);

for (String str : list)
 System.out.println(str);
```

Для отримання файлів та піддиректорій у заданій директорії є спеціальний метод - `newDirectoryStream()`, який повертає спеціальний об'єкт типу `DirectoryStream<Path>`, наприклад:

```
Path path = Path.of("c:\\windows");
try (DirectoryStream<Path> files =
Files.newDirectoryStream(path)) {
 for (Path path : files)
 System.out.println(path);
}
```

Об'єкт `DirectoryStream<Path>` має дві властивості. По-перше, він має ітератор, який повертає шляхи до файлів, і цей об'єкт можливо використовувати всередині циклу `for-each`.

А по-друге, цей об'єкт є потоком даних, і його потрібно закривати за допомогою методу `close()`, або використовувати всередині `try-with-resources`.

2. Порядок виконання роботи

2.1 Для здобуття мінімально необхідного похідного балу за виконання лабораторної роботи з таблиці 2.1 обрати завдання до виконання. Номер варіанту за списком для кожної підгрупи.

2.3 Для здобуття більш високого балу за виконання лабораторної роботи виконати додатково одне завдання (рівень 1) або два завдання (рівень 2) з таблиці 2.2.

2.3 Створити та надати викладачеві звіт з виконання лабораторної роботи. Звіт повинен містити тему, мету та обладнання для виконання лабораторної роботи, короткий опис постанови задачі для кожного з завдань, лістинг програми, скріншоти результатів тестування програми.

3. Варіанти індивідуальних завдань

Таблиця 2.1

Варіант	**Завдання**
1	Написати програму, яка зчитуватиме з клавіатури рядки, і якщо цей рядок - це шлях до існуючого файлу, виводити в консолі "<введений рядок> - це файл". Якщо шлях до існуючої директорії, виводити в консолі "введений рядок - це директорія".
2	Створити файл, що містить список директорій (кожен рядок файла містить повний шлях до певної директорії). Написати програму, яка знаходить у цьому списку найбільшу за об’ємом директорію та виводить її вміст на екран.
3	Написати програму, яка зчитуватиме з клавіатури два шляхи до файлу. Якщо файлу по першому шляхом не існує, його потрібно створити. Якщо файл по першому шляху існує, потрібно перемістити цей файл по другому шляху, але тільки в тому випадку, якщо по другому шляху такого файлу немає.
4	Написати програму, яка вводить ім’я директорії та підраховує в цій директорії кількість файлів кожного типу. Тип файла визначається за його розширенням (наприклад, .java, .class, .txt і т. д.). Вивести кількість файлів кожного типу на екран і в текстовий файл.
5	Написати програму, яка буде зчитувати з клавіатури шлях до директорії, отримувати список файлів і директорій у заданій директорії та виводити в консолі інформацію про них у вигляді: "<шлях до файлу> - це файл", якщо це файл, "<шлях до директорії> - це директорія", якщо це директорія.
6	Написати програму, яка вводить ім’я директорії, перейменовує всі файли з розширенням .txt із заданої директорії, даючи їм імена 1.txt, 2.txt, 3.txt і так далі по порядку. Вивести старі імена файлів на екран і в текстовий файл.
7	Написати програму, яка буде зчитувати з клавіатури шляхи до двох директорій і копіювати файли з однієї директорії до іншої (тільки файли, піддиректорії ігноруються). Вивести кількість скопійованих файлів і імена піддиректорій на екран і в текстовий файл.

Варіант	Завдання
8	Написати програму, яка зчитуватиме з клавіатури шляхи до двох директорій, копіюватиме з першої заданої директорії до другої всі файли, змінені за останні 3 дні. Вивести імена файлів, що копіюються, на екран і в текстовий файл.
9	Написати програму, яка зчитуватиме з клавіатури шляхи до двох директорій і переміщатиме файли з однієї директорії в іншу (тільки файли, піддиректорії ігноруються). Вивести кількість переміщених файлів і імена піддиректорій на екран і в текстовий файл.
10	Написати програму, яка вводить ім’я директорії та знаходить кількість рядків у кожному .txt-файлі заданої директорії. Результати вивести на екран і в текстовий файл

Таблиця 2.2.

Варіант	Завдання
1	Написати програму, яка вводить назву директорії і маску імені файла (наприклад, масці «*a.?xt» відповідають імена файлів a.txt і bba.xxt), а потім перейменовує всі файли директорії, імена яких відповідають масці. При перейменуванні розширення файла зберігається, а ім’я змінюється на номер файла в алфавітному порядку.
2	Написати програму, яка вводить імена двох директорій і регулярний вираз, а потім копіює з першої директорії до другої всі файли, імена яких відповідають регулярному виразу. Програма повинна обробляти всі вкладені директорії. При копіюванні структура вкладених директорій має зберігатися.
3	Написати програму, яка порівнює вміст двох заданих директорій і виводить список файлів, які відрізняються. Для порівняння файлів використовуйте ім’я, розмір і дату останньої зміни файла. Програма має обробляти всі вкладені директорії.
4	Напишіть програму, яка для заданої директорії визначає, яку частину загального обсягу (у відсотках) займають файли кожного типу. Тип файла визначається його розширенням. Для зберігання даних використовуйте колекцію HashMap.

Варіант	Завдання
5	Напишіть програму, яка для заданої директорії видаляє директорії найнижчого рівня вкладеності, а файли з них переміщує на один рівень вище.

4. Питання для самоконтролю

1. Які класи забезпечують роботу з текстовими файлами й бінарними файлами?
2. У чому сенс відкриття та закриття файлів?
3. Як відкрити текстовий файл для додавання записів у кінець файла?
4. Яку інформацію про фото можна отримати з об’єкта File?
5. Які операції над файлами і директоріями реалізує клас File?
6. Якими способами можна скопіювати файл у Java?
7. Чи може об'єкт Fileвідповідати файлу, якого ще немає?
8. Як перетворити об'єкт File до типу Path?