**Потоки ввода-вывода (I/O) и работа с файлами в Java**

**Введение в систему ввода-вывода**

Java предоставляет мощный и гибкий механизм работы с потоками ввода-вывода (I/O), позволяющий взаимодействовать с файлами, сетями, консолью и другими источниками данных. Основой системы I/O в Java является абстракция потоков, которая позволяет читать и записывать данные в линейной последовательности.

Потоки делятся на два типа:

1. **Байтовые потоки** — работают с данными в виде байтов (например, для работы с бинарными файлами).
2. **Символьные потоки** — предназначены для работы с текстовыми данными.

**Основные классы и интерфейсы ввода-вывода**

Работа с I/O в Java осуществляется через классы из пакета java.io. Наиболее важные из них:

* **InputStream** и **OutputStream** — базовые классы для байтовых потоков.
* **Reader** и **Writer** — базовые классы для символьных потоков.
* **File** — класс для работы с файловой системой.

**Работа с файлами с использованием класса File**

Класс File предоставляет интерфейс для работы с файловой системой: создания, удаления и проверки существования файлов или каталогов.

Пример:

import java.io.File;

public class Main {

    public static void main(String[] args) {

        File file = new File("example.txt");

        // Проверка существования файла

        if (file.exists()) {

            System.out.println("Файл существует: " + file.getAbsolutePath());

        } else {

            System.out.println("Файл не найден");

        }

    }

}

**Чтение и запись байтовых данных**

Байтовые потоки используются для работы с файлами, содержащими бинарные данные (например, изображения).

**Запись данных с использованием FileOutputStream:**

import java.io.FileOutputStream;

import java.io.IOException;

public class Main {

    public static void main(String[] args) {

        try (FileOutputStream fos = new FileOutputStream("output.bin")) {

            fos.write(new byte[]{65, 66, 67}); // Записываем байты

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

//Чтение данных с использованием FileInputStream:

import java.io.FileInputStream;

import java.io.IOException;

public class Main {

    public static void main(String[] args) {

        try (FileInputStream fis = new FileInputStream("output.bin")) {

            int data;

            while ((data = fis.read()) != -1) {

                System.out.print((char) data); // Преобразуем байты в символы

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

**Работа с потоками в NIO**

С появлением Java NIO (New I/O) была добавлена улучшенная система работы с файлами и потоками, представленная в пакете java.nio.file. Например, чтение всего содержимого файла:

import java.nio.file.Files;

import java.nio.file.Paths;

import java.io.IOException;

public class Main {

    public static void main(String[] args) {

        try {

            String content = Files.readString(Paths.get("example.txt"));

            System.out.println(content);

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

}