Java IO

Problem

Problem

- Как читать/писать данные из/в файла?
- Как принимать/отправлять данные из/в сети?
- How is **data reading** from **Input** and **writing** to **Output**?

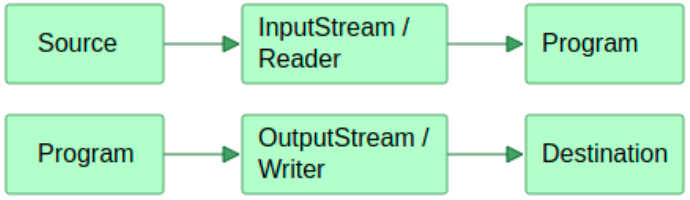
Solution

Java IO API

- package `java.io`
- reading and writing data (input and output)

Concept IO

Concept Java IO



Streams

Types

- Byte Based
- Character Based

Types (target)

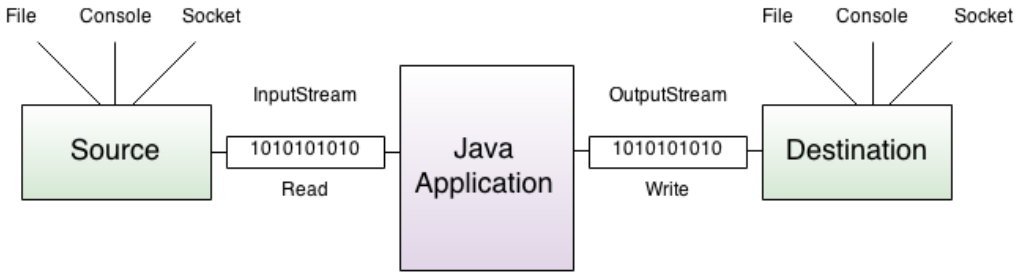
- Basic
- Arrays
- Files
- Pipes
- Buffering
- Filtering

Types (target)

- Parsing
- Strings
- Data
- Data-Formatted
- Objects
- Utilities

Byte-based streams

Byte-based streams



Byte-based streams

	Byte Based	
	Input	Output
Basic	`InputStream`	`OutputStream`
Arrays	`ByteArrayInputStream`	`ByteArrayOutputStream`
Files	`FileInputStream`	`FileOutputStream`
Buffering	`BufferedInputStream`	`BufferedOutputStream`
Filtering	`FilterInputStream`	`FilterOutputStream`
Data	`DataInputStream`	`DataOutputStream`
Objects	`ObjectInputStream`	`ObjectOutputStream`

InputStream

- `available(): int`
- `close(): void`
- `read(): int`
- `read(byte[] buffer): int`
- `read(byte[] buffer, int offset, int length): int`
- `skip(long number): long`

OutputStream

- close(): void
- flush(): void
- write(int b): void
- write(byte[] buffer): void
- write(byte[] buffer, int offset, int length): void

Closing Streams

Closeable

```
public interface Closeable extends AutoCloseable {
    public void close() throws IOException;
}
```

Ways to Close Streams

- `try` ... `catch` ... `finally`
- try-with-resource

try … catch … finally

```
import java.io.FileInputStream;
import java.io.IOException;

public class Program {
    public static void main(String[] args) {
        FileInputStream fin = null;
        try {
            fin = new FileInputStream("C://SomeDir//notes.txt");
            int i = -1;
            while ((i = fin.read()) != -1) {
                System.out.print((char) i);
            }
        } catch (IOException e) {
            System.out.println(e.getMessage());
        } finally {
            try {
                if (fin != null) {
                    fin.close();
                }
            } catch (IOException e) {
```

try-with-resources

```
import java.io.FileInputStream;
import java.io.IOException;

public class Program {
    public static void main(String[] args) {
        try (FileInputStream fin = new FileInputStream("C://SomeDir//notes.txt")) {
            int i = -1;
            while ((i = fin.read()) != -1) {
                System.out.print((char) i);
            }
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

ByteArrayInputStream

Constructors

- `ByteArrayInputStream(byte[] buf)`
- `ByteArrayInputStream(byte[] buf, int offset, int length)`

Example

```
import java.io.ByteArrayInputStream;
import java.io.IOException;

public class ReadExample {
    public static void main(String[] args) throws IOException {
        byte[] buf = {35, 36, 37, 38};
        ByteArrayInputStream byt = new ByteArrayInputStream(buf);
        int k = 0;
        while ((k = byt.read()) != -1) {
            char ch = (char) k;
            System.out.println("ASCII value of Character is:"
                + k + "; Special character is: " + ch);
        }
    }
}
```

ByteArrayOutputStream

Constructors

- `ByteArrayOutputStream()`
- `ByteArrayOutputStream(int size)`

Example

```
import java.io.ByteArrayOutputStream;
import java.io.FileOutputStream;

public class DataStreamExample {
    public static void main(String[] args) throws Exception {
        FileOutputStream fout1 = new FileOutputStream("D:\\f1.txt");
        FileOutputStream fout2 = new FileOutputStream("D:\\f2.txt");

        ByteArrayOutputStream bout = new ByteArrayOutputStream();
        bout.write(65);
        bout.writeTo(fout1);
        bout.writeTo(fout2);

        bout.flush();
        bout.close();
        System.out.println("Success...");
    }
}
```

Example `writeTo()`

```
import java.io.FileOutputStream;
import java.io.IOException;

public class DataStreamExample {
    public static void main(String[] args) throws Exception {
        ByteArrayOutputStream baos = new ByteArrayOutputStream();
        String text = "Hello Wolrd!";
        byte[] buffer = text.getBytes();
        try {
            baos.write(buffer);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        try (FileOutputStream fos = new FileOutputStream("hello.txt")) {
            baos.writeTo(fos);
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Class `File`

Fields

- `public static final char separatorChar = fs.getSeparator();`
- `public static final String separator = "" + separatorChar;`
- `public static final char pathSeparatorChar = fs.getPathSeparator();`
- `public static final String pathSeparator = "" + pathSeparatorChar;`

Constructors

- `File(File parent, String child)`
- `File(String pathname)`
- `File(String parent, String child)`
- `File(URI uri)`

Methods

- `createNewFile(): boolean`
- `delete(): boolean`
- `exists(): boolean`
- `getAbsolutePath(): String`
- `getName(): String`
- `getParent(): String`
- `isDirectory(): boolean`
- `isFile(): boolean`

Methods

- `isHidden(): boolean`
- `length(): long`
- `lastModified(): long`
- `list(): String[]`
- `listFiles(): File[]`
- `mkdir(): boolean`
- `renameTo(File dest): boolean`

Example

```
import java.io.File;
import java.io.IOException;

public class FileDemo {
    public static void main(String[] args) {
        try {
            File file = new File("javaFile123.txt");
            if (file.createNewFile()) {
                System.out.println("New File is created!");
            } else {
                System.out.println("File already exists.");
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Example

```
    public static void main(String[] args) {
        String path = "";
        boolean bool = false;
        try {
            File file = new File("testFile1.txt");
            file.createNewFile();
            System.out.println(file);
            File file2 = file.getCanonicalFile();
            System.out.println(file2);
            bool = file2.exists();
            path = file2.getAbsolutePath();
            System.out.println(bool);
            if (bool) {
                System.out.print(path + " Exists? " + bool);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Example

```
import java.io.File;

public class FileExample {
    public static void main(String[] args) {
        File f = new File("/Users/sonoojaiswal/Documents");
        String filenames[] = f.list();
        for (String filename : filenames) {
            System.out.println(filename);
        }
    }
}
```

Example

```
import java.io.File;

public class FileExample {
    public static void main(String[] args) {
        File dir = new File("/Users/sonoojaiswal/Documents");
        File files[] = dir.listFiles();
        for (File file : files) {
            System.out.println(file.getName() + " Can Write: "
                + file.canWrite() + "Is Hidden:"
                + file.isHidden() + " Length:"
                + file.length() + " bytes ");
        }
    }
}
```

FileInputStream

Constructors

- `FileOutputStream(String filePath)`
- `FileOutputStream(File fileObj)`
- `FileOutputStream(String filePath, boolean append)`
- `FileOutputStream(File fileObj, boolean append)`

Example

```
import java.io.FileOutputStream;

public class FileOutputStreamExample {
    public static void main(String[] args) {
        try {
            FileOutputStream fout = new FileOutputStream("D:\\testout.txt");
            String s = "Welcome to party!";
            byte b[] = s.getBytes();
            fout.write(b);
            fout.close();
            System.out.println("Success...");
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

FileOutputStream

Constructors

- `FileInputStream(File file)`
- `FileInputStream(FileDescriptor fdObj)`
- `FileInputStream(String name)`

Example

```
import java.io.FileInputStream;

public class DataStreamExample {
    public static void main(String[] args) {
        try {
            FileInputStream fis = new FileInputStream("D:\\testout.txt");
            int i = fis.read();
            System.out.print((char) i);
            fis.close();
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

BufferedInputStream

Constructors

- `BufferedInputStream(InputStream inputStream)`
- `BufferedInputStream(InputStream inputStream, int bufSize)`

Example

```
import java.io.BufferedInputStream;
import java.io.FileInputStream;

public class BufferedInputStreamExample {
    public static void main(String[] args) {
        try {
            FileInputStream fin = new FileInputStream("D:\\testout.txt");
            BufferedInputStream bin = new BufferedInputStream(fin);
            int i;
            while ((i = bin.read()) != -1) {
                System.out.print((char) i);
            }
            bin.close();
            fin.close();
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

BufferedOutputStream

Constructors

- `BufferedOutputStream(OutputStream outputStream)`
- `BufferedOutputStream(OutputStream outputStream, int bufSize)`

Example

```
import java.io.BufferedOutputStream;
import java.io.FileOutputStream;

public class BufferedOutputStreamExample {
    public static void main(String[] args) throws Exception {
        FileOutputStream fout = new FileOutputStream("D:\\testout.txt");
        BufferedOutputStream bout = new BufferedOutputStream(fout);
        String s = "Welcome to javaTpoint.";
        byte b[] = s.getBytes();
        bout.write(b);
        bout.flush();
        bout.close();
        fout.close();
        System.out.println("success");
    }
}
```

DataOutputStream

Methods

- `writeBoolean(boolean v): void`
- `writeByte(int v): void`
- `writeChar(int v): void`
- `writeDouble(double v): void`
- `writeFloat(float v): void`
- `writeInt(int v): void`
- `writeLong(long v): void`
- `writeShort(int v): void`
- `writeUTF(String str): void`

Example

```
import java.io.DataOutputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class OutputExample {
    public static void main(String[] args) throws IOException {
        FileOutputStream file = new FileOutputStream(D:\\testout.txt);
        DataOutputStream data = new DataOutputStream(file);
        data.writeInt(65);
        data.flush();
        data.close();
        System.out.println("Succcess...");
    }
}
```

DataInputStream

Methods

- `readBoolean(): boolean`
- `readByte(): byte`
- `readChar(): char`
- `readDouble(): double`
- `readFloat(): float`

Methods

- `readInt(): int`
- `readLong(): long`
- `readShort(): short`
- `readUTF(): String`
- `skipBytes(int n): int`

Example

```
import java.io.DataInputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;

public class DataStreamExample {
    public static void main(String[] args) throws IOException {
        InputStream input = new FileInputStream("D:\\testout.txt");
        DataInputStream inst = new DataInputStream(input);
        int count = input.available();
        byte[] ary = new byte[count];
        inst.read(ary);
        for (byte bt : ary) {
            char k = (char) bt;
            System.out.print(k + "-");
        }
    }
}
```

ZipOutputStream

Constructor

- `ZipOutputStream(OutputStream out)`

Example

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class Program {
    public static void main(String[] args) {
        String filename = "C:\\SomeDir\\notes.txt";
        try (ZipOutputStream zout = new ZipOutputStream(new FileOutputStream("C:\\SomeDi
             FileInputStream fis = new FileInputStream(filename);) {
            ZipEntry entry1 = new ZipEntry("notes.txt");
            zout.putNextEntry(entry1);
            byte[] buffer = new byte[fis.available()];
            fis.read(buffer);
            zout.write(buffer);
            zout.closeEntry();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
```

ZipInputStream

Constructor

- `ZipInputStream(InputStream in)`

Example

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

public class Program {
    public static void main(String[] args) {
        try (ZipInputStream zin = new ZipInputStream(new FileInputStream("C:\\SomeDir\\ou
            ZipEntry entry;
            String name;
            long size;
            while ((entry = zin.getNextEntry()) != null) {
                name = entry.getName();
                size = entry.getSize();
                System.out.printf("File name: %s \t File size: %d \n", name, size);
                FileOutputStream fout = new FileOutputStream("C:\\somedir\\new" + name);
                for (int c = zin.read(); c != -1; c = zin.read()) {
                    fout.write(c);
                }
```

Char-based streams

Char-based streams

	Character Based	
	Input	Output
Basic	`Reader`, `InputStreamReader`	`Writer`, `OutputStreamWriter`
Arrays	`CharArrayReader`	`CharArrayWriter`
Files	`FileReader`	`FileWriter`
Strings	`StringReader`	`StringWriter`
Buffering	`BufferedReader`	`BufferedWriter`
Filtering	`FilterReader`	`FilterWriter`

Reader

- absract void close()
- absract int read(char[] buffer, int offset, int count)
- read(): int
- read(char[] buffer): int
- read(CharBuffer buffer): int
- skip(long count): long

Writer

- abstract void close()
- abstract void flush()
- abstract void write(char[] buffer, int off, int len)
- append(char c): Writer
- append(CharSequence chars): Writer
- write(int c): void
- write(char[] buffer): void
- write(String str): void
- write(String str, int off, int len): void

PrintStream

Constructors

- `PrintStream(OutputStream outputStream)`
- `PrintStream(OutputStream outputStream, boolean autoFlushingOn)`
- `PrintStream(OutputStream outputStream, boolean autoFlushingOn, String charSet) throws UnsupportedEncodingException`
- `PrintStream(File outputFile) throws FileNotFoundException`

Constructors

- `PrintStream(File outputFile, String charSet) throws FileNotFoundException, UnsupportedEncodingException`
- `PrintStream(String outputFileName) throws FileNotFoundException`
- `PrintStream(String outputFileName, String charSet) throws FileNotFoundException, UnsupportedEncodingException`

Example

```
import java.io.FileOutputStream;
import java.io.PrintStream;

public class PrintStreamTest {
    public static void main(String[] args) throws Exception {
        FileOutputStream fout = new FileOutputStream("D:\\testout.txt ");
        PrintStream pout = new PrintStream(fout);
        pout.println(2016);
        pout.println("Hello Java");
        pout.println("Welcome to Java");
        pout.close();
        fout.close();
        System.out.println("Success?");
    }
}
```

PrintWriter

Constructors

- `PrintWriter(File file)`
- `PrintWriter(File file, String csn)`
- `PrintWriter(OutputStream out)`
- `PrintWriter(OutputStream out, boolean autoFlush)`
- `PrintWriter(String fileName)`
- `PrintWriter(String fileName, String csn)`
- `PrintWriter(Writer out)`
- `PrintWriter(Writer out, boolean autoFlush)`

Example

```
import java.io.File;
import java.io.PrintWriter;

public class PrintWriterExample {
    public static void main(String[] args) throws Exception {
        PrintWriter writer = new PrintWriter(System.out);
        writer.write("Java is popular programming language.It content many technologies.")
        writer.flush();
        writer.close();
        PrintWriter writer1 = null;
        writer1 = new PrintWriter(new File("D:\\testout.txt"));
        writer1.write("Like Spring, Hibernate, etc.");
        writer1.flush();
        writer1.close();
    }
}
```

BufferedWriter

Constructors

- `BufferedWriter(Writer out)`
- `BufferedWriter(Writer out, int sz)`

Example

```
import java.io.BufferedWriter;
import java.io.FileWriter;

public class BufferedWriterExample {
    public static void main(String[] args) throws Exception {
        FileWriter writer = new FileWriter("D:\\testout.txt");
        BufferedWriter buffer = new BufferedWriter(writer);
        buffer.write("Welcome to the party!");
        buffer.close();
        System.out.println("Success");
    }
}
```

BufferedReader

Constructors

- `BufferedReader(Reader in)`
- `BufferedReader(Reader in, int sz)`

Example

```
import java.io.BufferedReader;
import java.io.FileReader;

public class BufferedReaderExample {
    public static void main(String[] args) throws Exception {
        FileReader fr = new FileReader("D:\\testout.txt");
        BufferedReader br = new BufferedReader(fr);

        int i;
        while ((i = br.read()) != -1) {
            System.out.print((char) i);
        }
        br.close();
        fr.close();
    }
}
```

FileWriter

Constructors

- `FileWriter(File file)`
- `FileWriter(File file, boolean append)`
- `FileWriter(FileDescriptor fd)`
- `FileWriter(String fileName)`
- `FileWriter(String fileName, boolean append)`

Example

```
import java.io.FileWriter;

public class FileWriterExample {
    public static void main(String[] args) {
        try {
            FileWriter fw = new FileWriter("D:\\testout.txt");
            fw.write("Welcome to the party!");
            fw.close();
        } catch (Exception e) {
            System.out.println(e);
        }
        System.out.println("Success...");
    }
}
```

FileReader

Constructors

- `FileReader(String fileName)`
- `FileReader(File file)`
- `FileReader(FileDescriptor fd)`

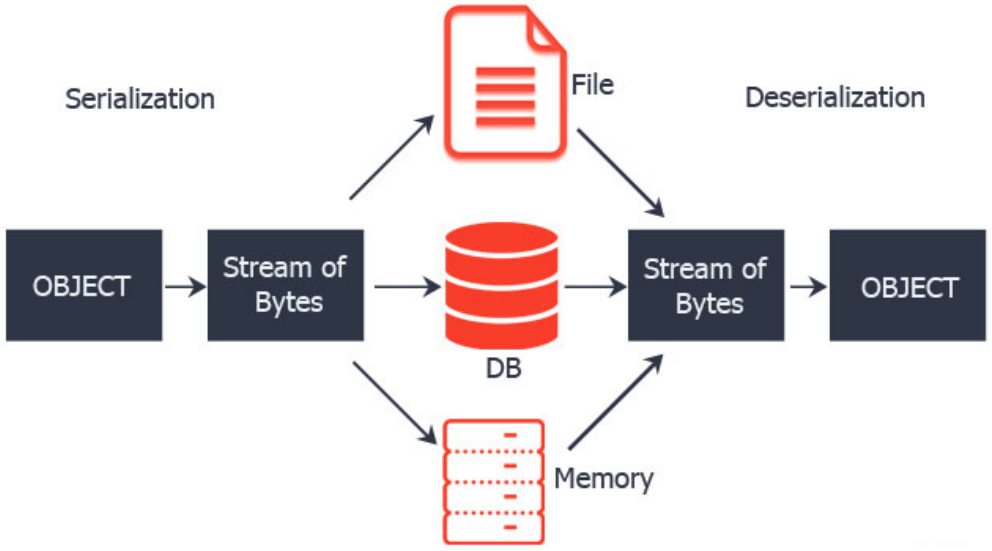
Example

```
import java.io.FileReader;

public class FileReaderExample {
    public static void main(String[] args) throws Exception {
        FileReader fr = new FileReader("D:\\testout.txt");
        int i;
        while ((i = fr.read()) != -1)
            System.out.print((char) i);
        fr.close();
    }
}
```

Object serialization

Serialization and Deserialization



Interface Serializable

```
public interface Serializable {
}
```

ObjectOutputStream

Methods

- `close(): void`
- `flush(): void`
- `write(byte[] buf): void`
- `write(int val): void`
- `writeBoolean(boolean val): void`
- `writeByte(int val): void`
- `writeChar(int val): void`

Methods

- `writeDouble(double val): void`
- `writeFloat(float val): void`
- `writeInt(int val): void`
- `writeLong(long val): void`
- `writeShort(int val): void`
- `writeUTF(String str): void`
- `writeObject(Object obj): void`

Example

```
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

class Persist {
    public static void main(String[] args) throws Exception {
        Student s1 = new Student(211, "ravi");

        FileOutputStream fout = new FileOutputStream("f.txt");
        ObjectOutputStream out = new ObjectOutputStream(fout);

        out.writeObject(s1);
        out.flush();
        System.out.println("success");
    }
}
```

ObjectInputStream

Methods

- `close(): void`
- `skipBytes(int len): int`
- `available(): int`
- `read(): int`
- `readBoolean(): boolean`
- `readByte(): byte`
- `readChar(): char`

Methods

- `readDouble(): double`
- `readFloat(): float`
- `readInt(): int`
- `readLong(): long`
- `readShort(): short`
- `readUTF(): String`
- `readObject(): Object`

Example

```
import java.io.FileInputStream;
import java.io.ObjectInputStream;

class Depersist {
    public static void main(String[] args) throws Exception {
        ObjectInputStream in =
            new ObjectInputStream(new FileInputStream("f.txt"));
        Student s = (Student) in.readObject();
        System.out.println(s.id + " " + s.name);
        in.close();
    }
}
```

Example

```
import java.io.Serializable;

class Person implements Serializable {
    private String name;
    private transient double height;

    Person(String name, double height) {
        this.name = name;
        this.height = height;
    }

    String getName() {
        return this.name;
    }

    double getHeight() {
        return this.height;
    }
}
```

Console

Methods

- `flush(): void`
- `format(String, Object…): Console`
- `printf(String, Object…): Console`
- `readLine(): String`
- `readLine(String, Object…): String`
- `readPassword(): char[]`

Example

```
import java.io.Console;

class ReadStringTest {
    public static void main(String[] args) {
        Console c = System.console();
        System.out.println("Enter your name: ");
        String n = c.readLine();
        System.out.println("Welcome " + n);
    }
}
```