CSCI 2120:
Software Design & Development II

UNIT3: I/O management

io api

OutputStreamWriter

Overview

1. Introduction
2. OutputStreamWriter class declaration
3. OutputStreamWriter constructors
4. OutputStreamWriter methods
5. OutputStreamWriter examples

Introduction

- **OutputStreamWriter in Java** is a character output stream that translates characters into bytes. It uses a stream of bytes as its data for output.

- `OutputStreamWriter` acts as a bridge between an incoming sequence of characters and an outgoing stream of bytes and converts a character stream into a byte stream.

- When characters are written to an `OutputStreamWriter`, it converts them into bytes according to the default or specified character encoding and writes those bytes to the specified `OutputStream`.

- In other words, data (characters) written to this writer are encoded into bytes using the specified character set and then writes those bytes to the specified `OutputStream`.

- The character set guarantees that any Unicode characters that we write to the `OutputStreamWriter` will be encoded into the correct byte representation.

OutputStreamWriter class declaration

An `OutputStreamWriter` is a concrete subclass of `Writer` class that extends `Object` class. It is also a superclass of `FileWriter` class that provides a convenient way to write characters to a file. `OutputStreamWriter` class implements `Closeable, Flushable, Appendable`, and `AutoCloseable`.

The general syntax to declare `OutputStreamWriter` class in Java is as follows:

```
public class OutputStreamWriter
       extends Writer
       implements Closeable, Flushable, Appendable, AutoCloseable
```

The inheritance diagram for `OutputStreamWriter` class is as follows:

```
java.lang.Object
   java.io.Writer
       java.io.OutputStreamWriter
```

OutputStreamWriter Constructors

1. OutputStreamWriter(OutputStream out):
This constructor creates `OutputStreamWriter` object that uses the default character encoding to convert characters into bytes.

The general syntax to create an object of `OutputStreamWriter` class in java is as follows:

```
OutputStreamWriter osw = new OutputStreamWriter(OutputStream out);
```

Here, the parameter to the constructor of `OutputStreamWriter` is of type `OutputStream`, so we can pass an object of any class derived from `OutputStream` to it.

OutputStreamWriter Constructors

2. OutputStreamWriter(OutputStream out, String charsetName):
This constructor creates an `OutputStreamWriter` object that uses the named character encoding.

`charsetName` specifies the character encoding that is used to encode characters into bytes. This constructor throws an exception named `UnsupportedEncodingException` when named character encoding is not supported.

---

3. OutputStreamWriter(OutputStream out, Charset cs):
This constructor creates an `OutputStreamWriter` object that uses the specified `charset` to encode characters into bytes.

---

4. OutputStreamWriter(OutputStream out, CharsetEncoder enc)
This constructor constructs an `OutputStreamWriter` object that uses the specified charset encoder.

OutputStreamWriter Methods

In addition to methods inherited from `Writer` class, Java `OutputStreamWriter` class also defines some useful methods. They are as follows:

Note:
All these methods provided by Java OutputStreamWriter class throws an exception named IOException if any error occurs

OutputStreamWriter Methods

Method	Description
void close()	This method closes the stream, flushing it first.
void flush()	This method flushes the stream.
String getEncoding()	This method returns the name of the character encoding being used by this stream.
void write(char[ ] cbuf, int offset, int length)	This method writes a segment of an array of characters. The first parameter cbuf specifies the buffer of characters, second parameter offset specifies from where to start writing characters, and third parameter length specifies the number of characters to write into the stream.

OutputStreamWriter Methods

Method	Description
void write(int c)	This method writes a single character. The parameter defined in this method c of type int specifies a character to be written.
void write(String str, int offset, int length)	This method writes a segment of a string. The parameter defined in method str represents a String, second parameter Offset specifies from where to start writing characters, and third parameter length specifies the number of characters to write into the stream..

Example 1: Write String to File

1. Let’s take a simple example program where we will write a string data into the specified file using `OutputStreamWriter` class. Look at the source code to understand better.

Example 1: Write String to File

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

public class OutputStreamWriterTester1 {
   public static void main(String[] args) throws IOException {
       String data = "I love Java Programming.";

       // Create FileOutputStream using filepath
       FileOutputStream file = new FileOutputStream("./src/outputfile.txt");

       // Create OutputStreamWriter by passing file to its constructor.
       OutputStreamWriter osw = new OutputStreamWriter(file);

       // Write string to the file.
       osw.write(data);

       // Close the writer.
       osw.close();
       System.out.println("Successfully written...");
   }
}
```

Example 1: Write String to File

Output:

```
Successfully written...
```

outputfile.txt:

```
I love Java Programming.
```

Explanation:

In this program, we have created an output stream using the `FileOutputStream` that is used to store the data in the form of individual bytes.

The `FileOutputStream` has been connected with the output.txt file where data is to be stored. Connecting the file `outputfile.txt` with `FileOutputStream` is done as follows:

```
FileOutputStream file = new FileOutputStream("./src/outputfile.txt");
```

To write string data to the file, we have created an object of `OutputStreamWriter` class with passing reference variable file to its constructor. Then we called `write()` method provided by `OutputStreamWriter` class to write data to the file.

When we execute the above program, the `outputfile.txt` file is filled with the following content “`I Love Java Programming`”

Example 2: Write with different encodings

2. Let’s create a program to get the type of encoding that is used to write data to the output stream.

Example 2: Write with different encodings

```
import java.io.FileOutputStream;
import java.io.OutputStreamWriter;
import java.io.IOException;
import java.nio.charset.Charset;

public class OutputStreamWriterTester2 {
   public static void main(String[] args) throws IOException {
       // Create FileOutputStream using filepath.
       FileOutputStream file = new FileOutputStream("./src/outputfile.txt");

       // Create OutputStreamWriter object with default encoding.
       OutputStreamWriter osw = new OutputStreamWriter(file);

       // Create OutputStreamWriter object with specified the encoding.
       OutputStreamWriter osw2 = new OutputStreamWriter(file, Charset.forName("UTF8"));

       // Return the name of character encoding used by output streams.
       System.out.println("Name of character encoding used by ows: " + osw.getEncoding());
       System.out.println("Name of character encoding used by ows2: " + osw2.getEncoding());

       // Closing the output stream writers.
       osw.close();
       osw2.close();
   }
}
```

Example 2: Write with different encodings

Output:

```
Name of character encoding used by ows: UTF8
Name of character encoding used by ows2: UTF8
```

outputfile.txt:

Explanation:

In this example we created two output stream writer named `osw` and `osw2`.

a. `osw` uses the default the character encoding. Therefore, the `getEncoding()` method returns the default character encoding.

b. `osw2` specifies the character encoding, `UTF8`. Therefore, the `getEncoding()` method returns the specified character encoding.

Note:

We have called static method named `forName()` of Charset class to specify the type of character encoding. Charset class is used to map characters and bytes.

END