Java Input - Output

Java Basic

Based On Java 21

Contents

- Overview
- Java Legacy IO
- Java New IO (Nio2)

Overview

- Java ဟာ စတင် Release လုပ်စဉ်ကထဲက File System တွေကို ကိုယ်စားပြုဖို့နဲ့ Input Output တွေကို ဆောင်ရွက်နိုင်ဖို့အတွက် java.io package ကို Support လုပ်ထားခဲ့ပါတယ်

- ဒါပေမဲ့လဲ java.io မှာပါတဲ့ API တွေမှာ OS တွေအားလုံးမှာပါတဲ့ Attribute တွေကို ကိုယ်စားမပြုနိုင်တာတို့၊ Fails ဖြစ်ရင်တောင် ဘာကြောင့်ဆိုတာမသိနိုင်တာတို့ File Operation တွေအတွက် အသုံးများတဲ့ Operation တွေမပါဝင်တာကြောင့် Java 7 အ ရောက်မှာ Nio2 ဆိုတဲ့ API အသစ်ကို Release လုပ်လာခဲ့တာဖြစ်ပါတယ်

Java Legacy IO

- Legacy Java IO
- File Class
- I / O Streams
- Serializables

Legacy Java IO

- Java စတင်ပေါ်ပေါက်ခဲ့စဉ်ကထဲကပါဝင်ခဲ့တဲ့ API တစ်ခုဖြစ်ပြီး File System ကို အသုံးပြုတဲ့အခါ တွေမှာနဲ့ Input / Output တွေကို ဆောင်ရွက်လိုတဲ့ အခါမျိုးတွေမှာ အသုံးပြုနိုင်ပါတယ်

- File System ထဲမှာရှိတဲ့ Directory တွေ File တွေကို ကိုယ်စားပြုတဲ့ Class အနေနှင့် File Class ကို ပြင်ပေးထားပြီး File ရဲ့ Properties တွေကို သိလိုတဲ့ အခါတွေမှာနှင့် File တွေ Directory တွေကို Create လုပ်တယ် Delete လုပ်တယ်၊ တန်ဖိုးတွေကို ပြောင်းလိုတဲ့ အခါတွေမှာ အသုံးပြုနိုင်ပါတယ်

- File တွေထဲက Data တွေကို Read, Write လုပ်ဖို့အတွက်ကတော့ I / O Stream တွေကို ပြင်ဆင် ပေးထားပါတယ်

File Class

- File System မှာရှိတဲ့ File တွေ Directory တွေကို ကိုယ်စားပြုတဲ့ Class ဖြစ်ပါတယ်
- File Object ကို တည်ဆောက်တဲ့အခါမှာ File System ပေါ်မှာရှိတဲ့ Path ကို အသုံးပြုပြီး တည်ဆောက်ရမှာ ဖြစ်ပြီး Path အနေနှင့် Absolute Path ကော Relative Path ကိုပါ အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်
- File Object ကို တည်ဆောက်တဲ့အခါမှာ အသုံးပြုတဲ့ Path ဟာ File System ပေါ်မှာ ရှိနေမှရယ်မဟုတ်၊ မရှိရင်လဲ တည်ဆောက်လို့ရပါတယ်
- File Object ကနေတဆင့် File တွေကို Manipulate လုပ်နိုင်သလို၊ Physical File ရဲ့ Information တွေကိုလဲ သိရှိဖို့အတွက် Method တွေကို ပြင်ပေးထားပါတယ်

Constructors of File Class

Constructors	Description
File(String path)	Absolute Path ဒါမှမဟုတ် Relative Path ကို အသုံးပြုပြီး တည်ဆောက်နိုင်
File(String parent, String child)	Parent Directory Path နှင့် Child Path Name တို့ဖြင့် တည်ဆောက်နိုင်
File(File parent, String child)	Parent Directory File နှင် Child Path Name တို့ဖြင့် တည်ဆောက်နိုင်
File(URI uri)	File တည်ရှိရာ URI ကို အသုံးပြုပြီး File Object ကို တည်ဆောက်နိုင်

Methods to check state

Methods	Description
isAbsolute():boolean	Absolute Path နှင့် တည်ဆောက်ထားတဲ့ File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
isDirectory():boolean	Directory ဟုတ်မဟုတ် စစ်ဆေးနိုင်
isFile():boolean	File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
isHidden():boolean	Hidden File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
canExecute():boolean	Execute လုပ်လို့ရတဲ့ File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
canRead():boolean	Read လုပ်လို့ရတဲ့ File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
canWrite():boolean	Write လုပ်လို့ရတဲ့ File ဟုတ်မဟုတ် စစ်ဆေးနိုင်
exists():boolean	File System မှာ တကယ်ရှိမရှိ စစ်ဆေးနိုင်

Methods to manipulate

Methods	Description
createNewFile():boolean	New File တစ်ခုကို တည်ဆောက်နိုင်ပါတယ်
mkdir():boolean	Directory တစ်ခုကို တည်ဆောက်နိုင်ပါတယ်
mkdirs():boolean	Directory တစ်ခုကို Parent နှင့်အတူ တည်ဆောက်နိုင်ပါတယ်
delete():boolean	File Object ကို Delete လုပ်နိုင်ပါတယ်
deleteOnExit():void	Application ကို Exit လုပ်တဲ့အခါမဟာ Delete လုပ်နိုင်ပါတယ်
renameTo(File dest):boolean	File Name ကို ပြောင်းလဲ သတ်မှတ်နိုင်ပါတယ်

Methods to get path information

Methods	Description
getName():String	File Name ကို ရယူနိုင်ပါတယ်
getPath():String	File Path ကို ရယူနိုင်ပါတယ်
getAbsoluteFile():File	Absolute Path ဖြင့်တည်ဆောက်ထားသော File Object ကို ရယူနိုင်ပါတယ်
getAbsolutePath():String	Absolute Path ဖြင့်တည်ဆောက်ထားသော Path ကို ရယူနိုင်ပါတယ်
getCanonicalFile():File	Canonical File Object ကို ရယူနိုင်ပါတယ်
getCanonicalPath():String	Canonical File Path တန်ဖိုးကို ရယူနိုင်ပါတယ်
getParent():String	Parent File name ကို ရယူနိုင်ပါတယ်
getParentFile():File	Parent File Object ကို ရယူနိုင်ပါတယ်

Methods to get state

Methods	Description
getFreeSpace():long	Free Space တန်ဖိုးကို ရယူနိုင်ပါတယ်
getTotalSpace():long	Total Space တန်ဖိုးကို ရယူနိုင်ပါတယ်
getUsableSpace():long	Usable Space တန်ဖိုးကို ရယူနိုင်ပါတယ်
lastModified():long	Last modified date တန်ဖိုးကို epoch millisecond ဖြင့်ရယူနိုင်ပါတယ်
length():long	File size တန်ဖိုးကို ရယူနိုင်ပါတယ်

Methods to change state

Methods	Description
setExecutable(boolean exec, boolean owner):boolean	Execute လုပ်လို့ရမရ Owner Only State ဖြင့် သတ်မှတ်
setExecutable(boolean exed):boolean	Execute လုပ်လို့ရမရသတ်မှတ်နိုင်
setReadable(boolean read, boolean owner):boolean	Read လုပ်လို့ရမရ Owner Only State ဖြင့် သတ်မှတ်နိုင်
setReadable(boolean read):boolean	Read လုပ်လို့ရမရသတ်မှတ်နိုင်
setWritable(boolean write, boolean owner):boolean	Write လုပ်လို့ရမရ Owner Only State ဖြင့် သတ်မှတ်နိုင်
setWritable(boolean write):boolean	Write လုပ်လို့ရမရသတ်မှတ်နိုင်
setReadOnly():boolean	Readonly File အဖြစ် သတ်မှတ်နိုင်
setLastModified(long time):boolean	Last Modified Properties ကို ပြောင်းလဲ သတ်မှတ်နိုင်

Methods to list children

Methods	Description
list():String[]	Child File Name တွေကို Array အဖြစ်ရယူနိုင်ပါတယ်
list(FilenameFilter filter):String[]	Child File Name တွေကို File Name filter ဖြစ်စစ်ပြီး ရယူနိုင်ပါတယ်
listFiles():File[]	Child File တွေကို Array အဖြစ်ရယူနိုင်ပါတယ်
listFiles(FilenameFilter filter):File[]	Child File တွေကို File Name filter ဖြစ်စစ်ပြီး ရယူနိုင်ပါတယ်
listFiles(FileFilter filter):File[]	Child File တွေကို File filter ဖြစ်စစ်ပြီး ရယူနိုင်ပါတယ်

Methods to get other representations

Methods	Description
toString():String	String တန်ဖိုးကို ပြောင်းပြီး ရယူနိုင်ပါတယ်
toPath():Path	NiO2 ရဲ့ Path အဖြစ် ပြောင်းပြီး ရယူနိုင်ပါတယ်
toURI():URI	URI အဖြစ် ပြောင်းပြီး ရယူနိုင်ပါတယ်

I/O Streams

- File Object ဟာ File တွေ Directory တွေကို ကိုယ်စားပြုတဲ့ Object သာဖြစ်ပြီး File တွေကို Read Write လုပ်တာတွေကိုတော့ လုပ်လို့မရနိုင်ပါဘူး

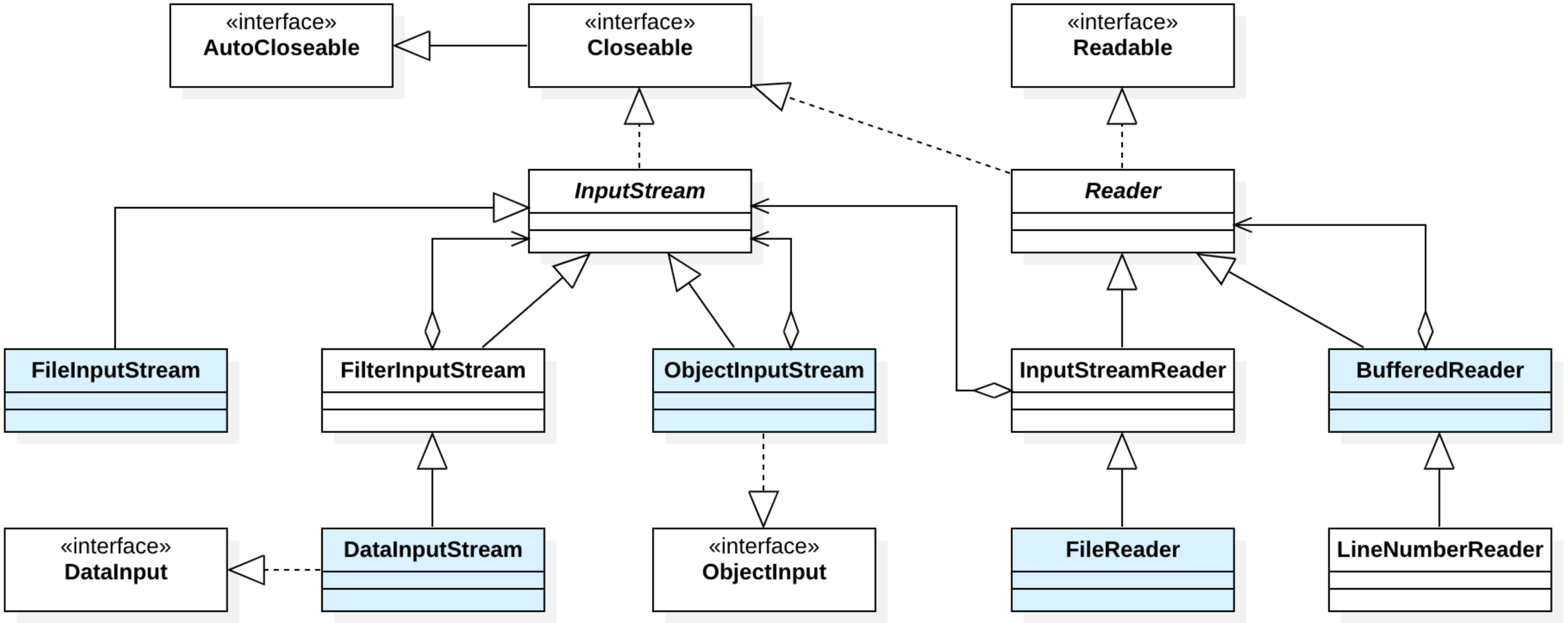
- Java Program တွေကနေ Input Output တွေကို ဆောင်ရွက်နိုင်ဖို့အတွက် I/O Streams တွေကို ပြင်ဆင်ပေးထားပါတယ်

- မိမိအသုံးပြုလိုတဲ့ Data အမျိုးအစားအလိုက် အလွယ်တကူ အသုံးပြုနိုင်အောင် I/O Stream တွေကို ပြင်ဆင်ပေးထားပါတယ်

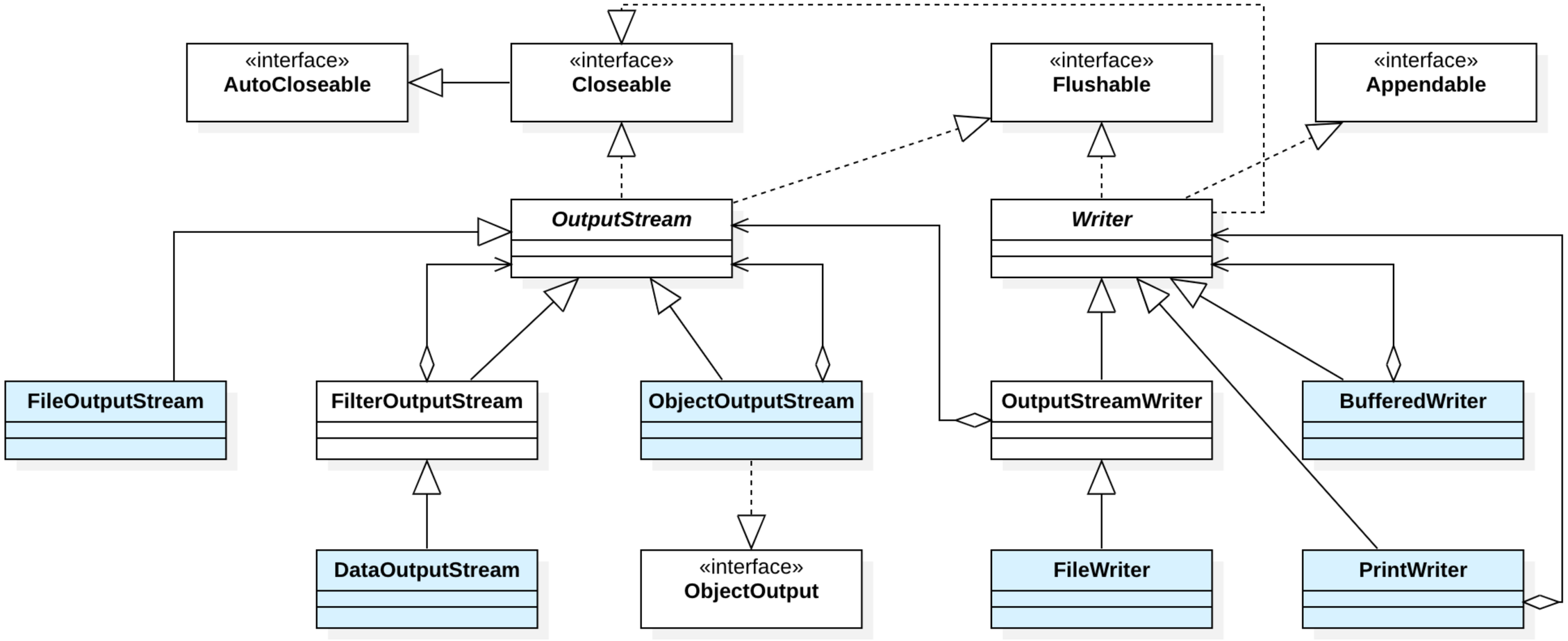
Type of Streams

Data Type	Input Streams	Output Streams
Byte Streams	FileInputStream	FileOutputStream
Character Streams	FileReader	FileWriter
Buffered Streams	BufferedReader	BufferedWriter
Data Streams	DataInputStream	DataOutputStream
Object Streams	ObjectInputStream	ObjectOutputStream

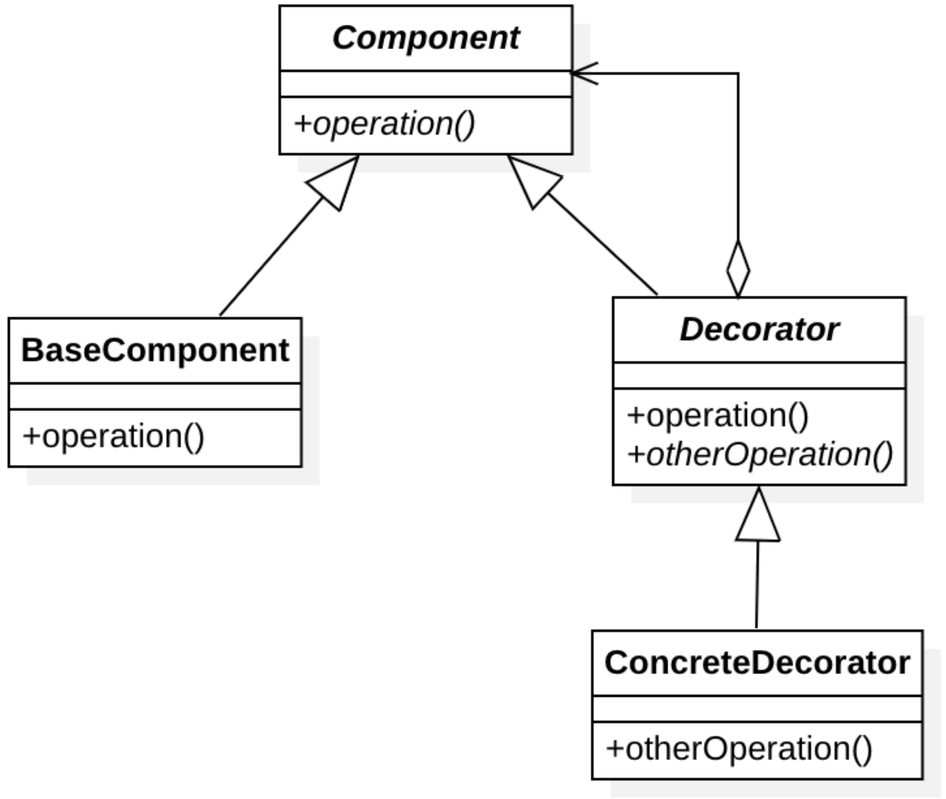
Input Streams



Output Streams



Decorator Pattern



- Decorator Pattern ဟာ behavioral design patterns တစ်ခုဖြစ်ပြီး ရှိပြီးသား Component တစ်ခုရဲ့ Feature တွေကို အသုံးပြုပြီး မတူညီတဲ့ Feature တွေကို အသစ်ဖြည့်စွက်လိုတဲ့ အခါမှာ အသုံးပြုနိုင်ပါတယ်

Base Input Methods

Method	Description
read():int	တစ်ခါ Read လုပ်ရင် 1 Byte ကို Read လုပ်နိုင်ပါတယ်
read(byte[]):int	တစ်ခါ Read လုပ်ရင် သတ်မှတ်ထားတဲ့ byte[] ဖြင့် Read လုပ်နိုင်ပါတယ်
read(byte[], int, int):int	Offset နှင့် Length ကို သတ်မှတ်ပြီး byte[] ဖြင့် Read လုပ်နိုင်ပါတယ်
readAllBytes():byte[]	Input Stream ထဲက ကျန်သမျှ byte ကို Read လုပ်နိုင်ပါတယ်
readNBytes(byte[], int, int):int	Offset နှင့် Length ကို သတ်မှတ်ပြီး byte[] ဖြင့် Read လုပ်နိုင်ပါတယ်
readNBytes(int):byte[]	Length ကို သတ်မှတ်ပြီး byte[] ဖြင့် Read လုပ်နိုင်ပါတယ်
transferTo(OutputStream):long	Input Stream ထဲက Data တွေကို Output Stream ကို Transfer လုပ်ပေးနိုင်

AutoCloseable and Try-with resources

- I / O Stream တွေဟာ External Resources တွေကို အသုံးပြုနေတဲ့အတွက် အသုံးပြုပြီးပါက သေချာပေါက် Close လုပ်ပေးနိုင်ဖို့လိုအပ်ပါတယ်

- Java 7 ကနေစပြီး AutoCloseable Interface ကို Implement လုပ်ထားတဲ့ Object တွေကို Try-With Resources Statement နှင့် တွဲဖက်ရေးသားပါက နောက်ဆုံးမှာ အလိုအလျောက် Close လုပ်ပေးလာနိုင်ပါတယ်

- Close လုပ်ဖို့လိုတဲ့ Object တွေကို try clause ထဲမှာ ရေးသားထားပါက အလိုအလျောက် Close လုပ်ပေးနိုင်မှာ ဖြစ်ပါတယ်

Try-with resources

```
try(var input = new FileInputStream(original);
    var output = new FileOutputStream(copy)) {

    byte [] array = new byte[1024];

    while(input.read(array) != -1) {
        output.write(array);
    }

    return true;

} catch (IOException e) {
    e.printStackTrace();
    return false;
}
```

Data Specific Input Methods

Method	Description
readBoolean():boolean	Can read boolean value
readChar():char	Can read character value
readByte():byte	Can read byte value
readShort():short	Can read short value
readInt():int	Can read int value
readLong():long	Can read long value
readDouble():double	Can read double value
readFloat():load	Can read float value
readUnsignedByte():int	Can read unsigned byte value
readUnsignedShort():int	Can read unsigned short value
readUTF():String	Can read UTF String value
readObject():Object	Can read object value

Text Based Input Methods

Method	Description
read():int	Character တစ်လုံးကို Read လုပ်နိုင်ပါတယ်
read(char[]):int	Character Array ကို Read လုပ်နိုင်ပါတယ်
read(char[], int, int):int	Character Array ကို Offset နှင့် Length ကို သတ်မှတ်ပြီး Read လုပ်နိုင်ပါတယ်
read(CharBuffer):int	CharBuffer အနေနှင့် Read လုပ်နိုင်ပါတယ်
readLine():String	တစ်ကြိမ်ကို စာကြောင်းတစ်ကြောင်း Read လုပ်ပေးနိုင်ပါတယ်
lines():Strem<String>	စာကြောင်းတွေကို Stream Object အနေနှင့် ရယူနိုင်ပါတယ်

Base Output Methods

Method	Description
write(int):void	တစ်ခါ Write လုပ်ရင် 1 Byte ကို Write လုပ်နိုင်ပါတယ်
write(byte[]):void	တစ်ခါ Write လုပ်ရင် သတ်မှတ်ထားတဲ့ byte[] ဖြင့် Write လုပ်နိုင်ပါတယ်
write(byte[], int, int):void	Offset နှင့် Length ကို သတ်မှတ်ပြီး byte[] ဖြင့် Write လုပ်နိုင်ပါတယ်

Data Specific Output Methods

Method	Description
writeBoolean(boolean):void	Can write boolean value
writeByte(int):void	Can write single byte value
writeBytes(String):void	Can write String as byte values
writeChar(int):void	Can write char value
writeChars(String):void	Can write String as char values
writeDouble(double):void	Can write double value
writeFloat(float):void	Can write float value
writeInt(int):void	Can write int value
writeLong(long):void	Can write long value
writeShort(int):void	Can write short value
writeUTF(String):void	Can write unicode string value

Text Based Output Methods

Method	Description
write(char[]):void	Character Array ကို Write လုပ်နိုင်ပါတယ်
write(char[], int, int):void	Character Array ကို Offset နှင့် Length ကို သတ်မှတ်ပြီး Write လုပ်နိုင်ပါတယ်
write(int):void	Single Character ကို Write လုပ်နိုင်ပါတယ်
write(String):void	String တစ်ခုအနေနှင့် Write လုပ်နိုင်ပါတယ်
write(String, int, int):void	String ကို Offset နှင့် Length ကို သတ်မှတ်ပြီး Write လုပ်နိုင်ပါတယ်
newLine():void	တစ်ကြိမ်ကို စာကြောင်းတစ်ကြောင်း Read လုပ်ပေးနိုင်ပါတယ်

Serialization

- JVM Memory ပေါ်မှာရှိတဲ့ Java Object တွေကို အခြားနေရာတစ်နေနေရာကို Output လုပ်ပြီး သိမ်းပေးနိုင်ပြီး Serialize လုပ်တယ်လို့ ခေါ်ဆိုလေ့ရှိပါတယ်

- Serializable Interface ကို Implement လုပ်ထားတဲ့ Object တွေကိုသာ Serialize လုပ်နိုင်မှာ ဖြစ်ပြီး Serialize လုပ်တဲ့အခါမှာ Type Information တွေနှင့် State တွေကိုပဲ အသုံးပြုတာဖြစ်ပါတယ်

- တကယ်လို့ Serialize လုပ်တဲ့အခါမှာ မပါစေလိုတဲ့ Instance Variable တွေကို transient modifier ဖြင့် သတ်မှတ်နိုင်ပါတယ်

- Serialize လုပ်ထားတဲ့ Data တွေကို JVM Memory ပေါ်ကိုပြန်ပြီး Read လုပ်တာကိုတော့ De-Serialize လုပ်တယ်လို့ ခေါ်ဆိုကြပါတယ်

Serialize Version ID

- Serializable Interface ကို Implement လုပ်တဲ့ အခါမှာ Serialize Version ID မပါဝင်ပါက Compiler က Generate လုပ်ပြီး သိမ်းပေးပါတယ်

- Serialize လုပ်တဲ့ အခါမှာ Serialize Version ID ကိုပါ Object ထဲကို ထည့်သိမ်းပေးပြီး၊ ပြန်ပြီး Deserialize လုပ်တဲ့အခါမှာ သိမ်းထားတဲ့ Version ID နှင့် Class ထဲက Version ID ကို တိုက်ပြီးစစ်ပါတယ်

- တကယ်လို့ Serializable Class တွေမှာ Serialize Version ID ကို ရေးမထားပါက Compile လုပ်တိုင်း ပြောင်းနေနိုင်တာမို့ Deserialize လုပ်မည့် Object ရဲ့ Version မတူတဲ့ပြဿနာကို ဖြစ်စေနိုင်မှာ ဖြစ်ပါတယ်

Java New IO (Nio2)