Compare ByteStream : Character Stream

These two kinds of streams behave in different ways when you read and write data.

|  |  |
| --- | --- |
| **Character Stream** | **Byte Stream** |
| Character Streams were introduced in JDK 1.1 as part of java.io package, prior to that i/o facilities were supported only by byte streams | Byte streams originated with Java I/O to read raw binary data |
| I/O stream manages 16 bit Unicode characters | I/O stream manager manages 8 bit bytes of raw binary data |
| Character stream is essentially a byte stream fronted by a reader or writer |  |
| They are implemented by **Reader** and **Writer** classes | They are implemented by **Inputstream** and **Outputstream** class |
| Character streams are used for storing and retrieving text When the strings are written to a stream as character data, by default the Unicode characters are automatically converted to the local representation of the characters in the host machine, and these are then written  to the stream. | Binary streams are sometimes referred to as byte streams.  When you write data to a binary stream, the data is written to the stream as a series of bytes, exactly as it  appears in memory. No transformation of the data takes place. Binary numerical values are just written  as a series of bytes, 4 bytes for each value of type int, 8 bytes for each value of type long, 8 bytes for  each value of type double, and so on. |
| Java allows named mappings  between Unicode characters and sets of bytes to be defined, called charsets, and you can select an available charset that should apply when data is transferred to, or from |  |

**References:**

For more information refer pg 374 , chapter 8 ivor Horton and

Oracle docs :

<https://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html>

<https://docs.oracle.com/javase/8/docs/technotes/guides/io/io.html>