Java FileOutputStream Class

# Java FileOutputStream Class

Java FileOutputStream is an output stream used for writing data to a file.

If you have to write primitive values into a file, use FileOutputStream class. You can write byte-oriented as well as character-oriented data through FileOutputStream class. But, for character-oriented data, it is preferred to use FileWriter than FileOutputStream.

## **FileOutputStream class declaration**

Let's see the declaration for Java.io.FileOutputStream class:

1. **public** **class** FileOutputStream **extends** OutputStream

## **FileOutputStream class methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| protected void finalize() | It is sued to clean up the connection with the file output stream. |
| void write(byte[] ary) | It is used to write **ary.length** bytes from the byte array to the file output stream. |
| void write(byte[] ary, int off, int len) | It is used to write **len** bytes from the byte array starting at offset **off** to the file output stream. |
| void write(int b) | It is used to write the specified byte to the file output stream. |
| FileChannel getChannel() | It is used to return the file channel object associated with the file output stream. |
| FileDescriptor getFD() | It is used to return the file descriptor associated with the stream. |
| void close() | It is used to closes the file output stream. |

## **Java FileOutputStream Example 1: write byte**

|  |  |
| --- | --- |
| **Package** com.company;  **import** java.io.FileOutputStream;  **public class** Main {   **public static void** main(String[] args) {  **try** {  FileOutputStream fileOutputStream = **new** FileOutputStream(**"D:\\Cat.txt"**);  fileOutputStream.write(65);  fileOutputStream.close();  System.***out***.println(**"success.........."**);  } **catch** (Exception e){  System.***out***.println(**"error : "** + e);  }  } } | success..........  text file op:  A |

## **Java FileOutputStream Example 1: write Strings**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.FileOutputStream;  **public class** Main {   **public static void** main(String[] args) {  **try** {  FileOutputStream fileOutputStream = **new** FileOutputStream(**"D:\\Cat.txt"**);  String x = **" Java, C#, ASP.NET, SQL, Javascript"**;  **byte** b[] = x.getBytes(); *//converting string into byte array* fileOutputStream.write(b);  fileOutputStream.close();  System.***out***.println(**"Success........"**);  } **catch** (Exception e){  System.***out***.println(**"error : "** + e);  }  } } | Success........ |

**FileDescriptor getFD()**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.FileDescriptor; **import** java.io.FileOutputStream; **import** java.io.IOException; **import** java.io.FileInputStream; **import** java.nio.channels.FileChannel;  **public class** Main {   **public static void** main(String[] args) {  **try** {  **boolean** ifValid = **false**;  **int** i = 0;  FileOutputStream fileOutputStream = **new** FileOutputStream(**"D:\\Cat.txt"**);  String x = **" Java, C#, ASP.NET, SQL, Javascript"**;  **byte** b[] = x.getBytes(); *//converting string into byte array* fileOutputStream.write(b); *// fileOutputStream.close();* System.***out***.println(**"Success........"**);   *//FileDescriptor getFD()* FileDescriptor x1 = fileOutputStream.getFD();  ifValid = x1.valid();  System.***out***.println(**" FileDescriptor : "** + x1);  System.***out***.println(**" isValid or not : "** + ifValid);  fileOutputStream.close();  } **catch** (Exception e){  System.***out***.println(**"error : "** + e);  }  } } | Success........  FileDescriptor : java.io.FileDescriptor@2ac1fdc4  isValid or not : true |

# Java.io.FileInputStream.getChannel()

|  |  |
| --- | --- |
| **package** com.company; **import** java.io.IOException; **import** java.io.FileInputStream; **import** java.nio.channels.FileChannel;  **public class** Main {   **public static void** main(String[] args) **throws** IOException {  FileChannel fc = **null**;  FileInputStream fis = **null**;  **int** i = 0;  **long** pos;  **char** c;   **try** {   *// create new file input stream* fis = **new** FileInputStream(**"D:\\Cat.txt"**);   *// read till the end of the file* **while**((i = fis.read())!=-1) {   *// get file channel* fc = fis.getChannel();   *// get channel position* pos = fc.position();   *// integer to character* c = (**char**)i;   *// prints* System.***out***.print(**"No of bytes read: "**+pos);  System.***out***.println(**"; Char read: "**+c);  }   } **catch**(Exception ex) {   *// if an I/O error occurs* System.***out***.println(**"IOException: close called before read()"**);  }   } } | No of bytes read: 1; Char read:  No of bytes read: 2; Char read: J  No of bytes read: 3; Char read: a  No of bytes read: 4; Char read: v  No of bytes read: 5; Char read: a  No of bytes read: 6; Char read: ,  No of bytes read: 7; Char read:  No of bytes read: 8; Char read: C  No of bytes read: 9; Char read: #  No of bytes read: 10; Char read: ,  No of bytes read: 11; Char read:  No of bytes read: 12; Char read: A  No of bytes read: 13; Char read: S  No of bytes read: 14; Char read: P  No of bytes read: 15; Char read: .  No of bytes read: 16; Char read: N  No of bytes read: 17; Char read: E  No of bytes read: 18; Char read: T  No of bytes read: 19; Char read: ,  No of bytes read: 20; Char read:  No of bytes read: 21; Char read: S  No of bytes read: 22; Char read: Q  No of bytes read: 23; Char read: L  No of bytes read: 24; Char read: ,  No of bytes read: 25; Char read:  No of bytes read: 26; Char read: J  No of bytes read: 27; Char read: a  No of bytes read: 28; Char read: v  No of bytes read: 29; Char read: a  No of bytes read: 30; Char read: s  No of bytes read: 31; Char read: c  No of bytes read: 32; Char read: r  No of bytes read: 33; Char read: i  No of bytes read: 34; Char read: p  No of bytes read: 35; Char read: t |