

## **Java Create and Write To Files**

**Topics**: JAVA

Written on April 11, 2023

In Java, you can create and write to files using the FileWriter and BufferedWriter classes.

Here's an example:

```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
public class WriteToFileExample {
    public static void main(String[] args)
        try {
            FileWriter fileWriter = new FileWriter("output.txt");
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
            bufferedWriter.write("Hello, world!");
            bufferedWriter.newLine();
            bufferedWriter.write("This is an example of writing to a file in
Java."):
            bufferedWriter.close();
            System.out.println("Data has been written to the file.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

In this example, we create a FileWriter object for a file named "output.txt". We also create a BufferedWriter object, which allows us to write to the file in a buffered manner, improving performance.

We then use the write() method to write data to the file, and the newLine() method to add a new line after the first line. Finally, we close the BufferedWriter object to flush any buffered data to the file and release any resources held by the object.

If an IOException occurs while writing to the file, we catch it and print the stack trace.

Note that if the file does not exist, it will be created automatically by the FileWriter class. If the file already exists, its contents will be overwritten by the new data.

You can also append data to an existing file using the FileWriter constructor that takes a boolean

parameter, as shown in the following example:

```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
public class AppendToFileExample {
    public static void main(String[] args) {
        try {
            FileWriter fileWriter = new FileWriter("output.txt", true); //
true means append to file
            BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
            bufferedWriter.newLine();
            bufferedWriter.write("This line will be appended to the file.");
            bufferedWriter.close();
            System.out.println("Data has been appended to the file.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

In this example, we pass true as the second argument to the FileWriter constructor, which tells it to append data to the end of the file instead of overwriting its contents. We then write a new line and some text to the file using the BufferedWriter object.

© Copyright **Aryatechno**. All Rights Reserved. Written tutorials and materials by <u>Aryatechno</u>