How to read and write files using JES application

JES is a programming environment that you can use to write, test, and run code locally on your computer. JES offers many features such as photo, video and other media editing.

JES is a programming environment that you can use to write, test, and run code locally on your computer. JES offers many features such as photo, video and other media editing.

You can also complete other tasks in JES like reading or writing files. You can read data from different file types, including text and CSV files.

How to read a file using JES

To read a file in JES, first open it, then read the contents and save it as a variable for further processing.

If you're reading a text file, you can save each file line individually as an element in an array. If you are reading a CSV file, you can also store the value of each cell separately.

1. Open the JES software on the computer. In the programming window, create a new function:

```
def readFromFile():
```

2. Use the pickAFile() function to ask the user to select a file.

```
file = pickAFile()
```

3. File validation to ensure users only select TXT or CSV files:

```
if not file.endswith(".txt") and not file.endswith(".csv"): ??
print("Error: Only .txt and .csv files are supported.") ??return
```

4. Open the selected file with the open() function. The first parameter is the file that you are reading. The second parameter defines the mode used to open the file. For example, rt means read file - read file.

```
openedFile = open(file, "rt")
```

5. While opening the file, read all the contents inside. Save the file contents in a variable:

```
allContent = openedFile.read()
```

6. Close the file:

```
openedFile.close()
```

7. The variable 'allContent' contains a single string that stores the entire contents of the file. A newline character (**n**) separates each line in a file. If ball wants to access individual lines, split a string and store each line as an element in an array:

```
if file.endswith(".txt"): ??rows = allContent.split("n") ??print(rows)
```

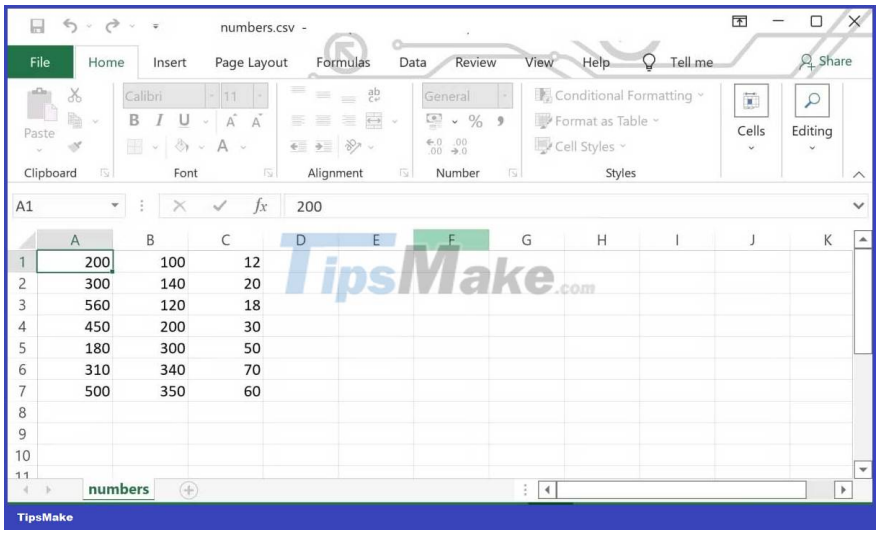
8. If you are reading a CSV file, you can separate each line to get the value from each individual cell. For each row in the CSV file, separate the values ??by commas, and store the values ??in an array of doubles. The structure of the array will look similar to this: [[1,2,3], [4,5,6], [7,8,9]].

```
if file.endswith(".csv"): ??rows = allContent.split("n") ??csvData = [] ??
for row in rows: ????if (row != ''): ??????cells = row.split(",") ??????
csvData.append([float(cell) for cell in cells]) ??print(csvData)
```

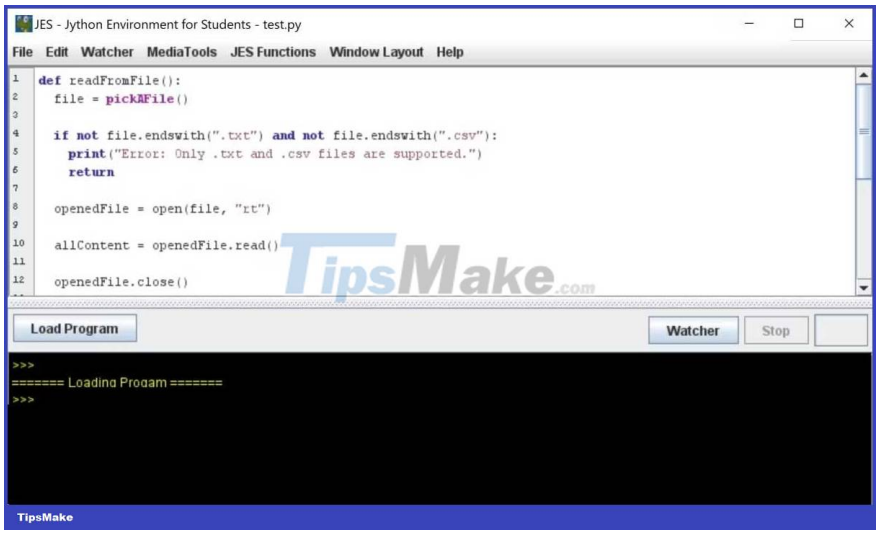
9. To test this program, create a new text file named sample.txt, and fill it with some text:

```
This is the start of the file This is another line This is the third line This is
```

10. Alternatively, create a CSV file, name it **numbers.csv** and fill it with some data.



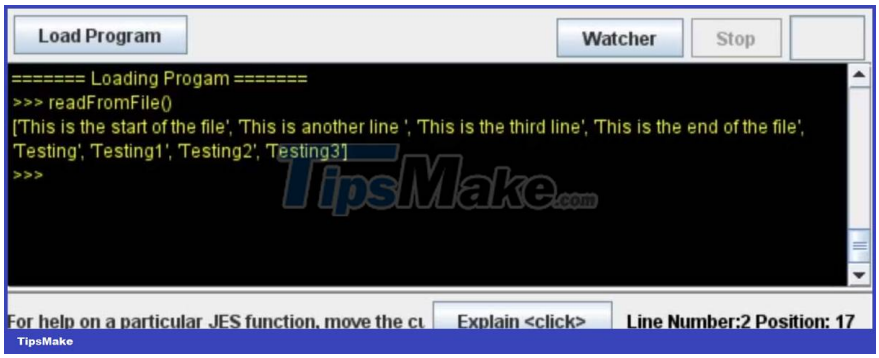
11. On the JES interface, click the **Load Program** button , located between the programming window and the command line:



12. Run the readFromFile() function in the command line:

```
readFromFile()
```

13. Using the file prompt, navigate to where you saved the **sample.txt** file . Select the file to open it and see what's printed to the console:



14. Run the readFromFile() function again in the command prompt. Select the numbers.csv file to see what's printed to the console, with each cell separated and stored in an array:

How to write a file in JES

You can write a text or CSV file using the write() function. You can open the file for appending or burning. Additional data will add to existing content, while writing will overwrite any existing content in the file.

Create a new function and use it to write a text and CSV file.

1. Create a new function named writeToFile():

```
def writeToFile():
```

2. Use the pickAFile() function to prompt the user to select a file:

```
file = pickAFile()
```

3. Open the file to add data:

```
openedFile = open(file, "at")
```

4. Alternatively, if you want to overwrite all the contents of the file, enter w as the second argument:

```
openedFile = open(file, "w")
```

5. Write files. To add multiple lines, use 'n" to split the content into lines, or reuse the write() function:

```
if file.endswith(".txt"): ??openedFile.write("nTesting") ??
openedFile.write("nTesting1nTesting2") ??openedFile.write("nTesting3")
```

6. To write a CSV file, write all the data for a row using the write() function, and separate the values ??for each cell with a comma:

```
if file.endswith(".csv"): ??openedFile.write("n12,34,56")
```

7. Close the file after writing it:

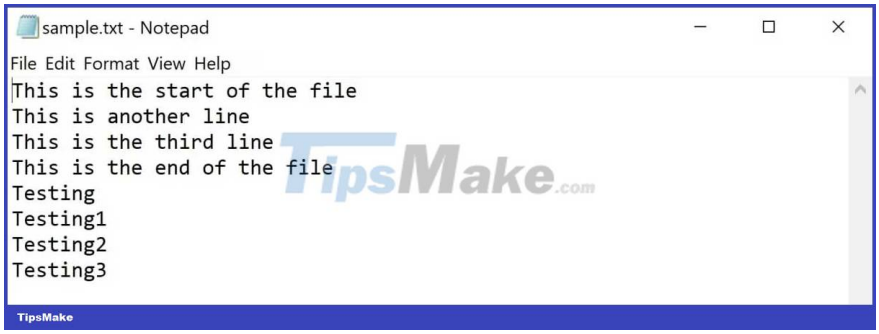
```
openedFile.close() print("Wrote to file successfully")
```

8. Click the Load Program button, located between the programming window and the command line.

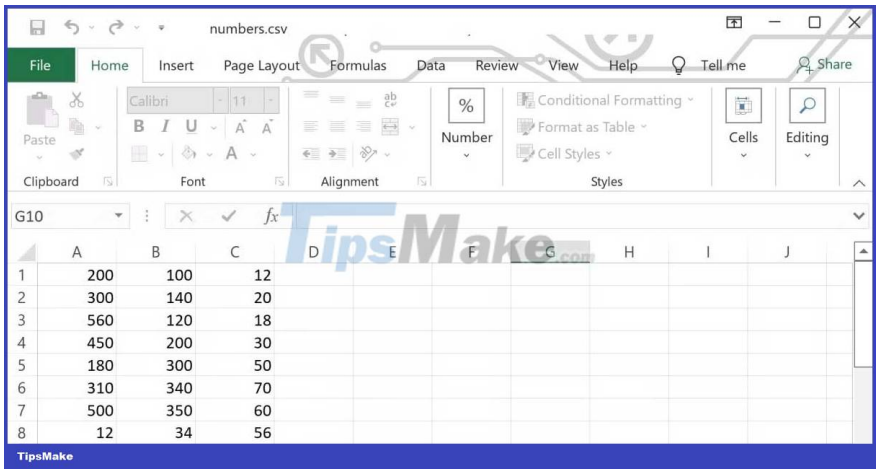
9. Run **the writeToFile()** function in the command prompt:

```
writeToFile()
```

10. Using the file prompt, select the file 'sample.txt.' After JES finishes writing the file, open 'sample.txt' to see the new lines added to the end of the file:



11. Run the writeToFile() function again in the command line. Open the file ' **numbers.csv** ' to see the new cell values ??added at the end of the file.



Writing data to a file is an extremely useful function that you can use if you need to save any data inside a program. Above is how to read and write roles into a JES app. Hope the article is useful to you.

You finished reading the article "**How to read and write files using JES application**" edited by the **TipsMake** team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.