File Management

Monday, December 13, 2021 3:20 PM

FILES

Much like python files (.py), all files are saved in a file. Computer operating systems consists of a mountain of files.

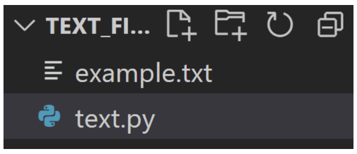
As a computer scientist or software engineer, working with files is an essential skill that you will learn.

This section will demonstrate how to read and write text (.txt) files.

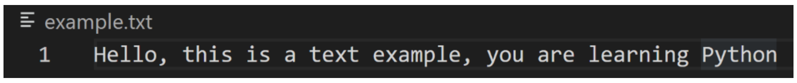
There are Binary and text files. Text files contain data readable to a Human beings, for example JSON files and plain text files. Binary files are used for storing complied code and audio and video files.

To open a file in Python we can use the `open()` function. When using the `help()` function it proves pages of information showing all the different ways to use it.

To make this easier, we can create a new directory and add a .txt file called example.txt. Next add a Python file named text.py.



Inside example.txt, we can insert some dummy data for testing purposes:

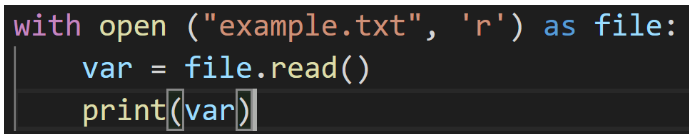


```
example.txt
Hello, this is a text example, you are learning Python
```

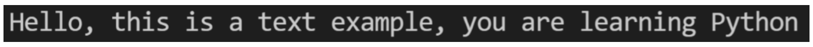
One way to open a file is to call the `open()` function with the name or path to the file in the parenthesis. Note that if you create a .txt file in a different directory, you will have to write it's entire path in the parenthesis.

In this case we will use a with `open()` block with the file name followed by the mode, in this case we want to read the files contents so we use r `'r'`.
After we specify the open, the file name and the mode as file we can store it's content in a variable called var, which we can then use the assignment operator to say var is equal to `file.read()`.

We can `print()` the variable var to see its contents. See the example.



```python
with open ("example.txt", 'r') as file:
    var = file.read()
    print(var)
```



```
Hello, this is a text example, you are learning Python
```

To close a file we can just say `file.close()` with the pass condition to insure the file is closed and cannot be read or edited, see example:

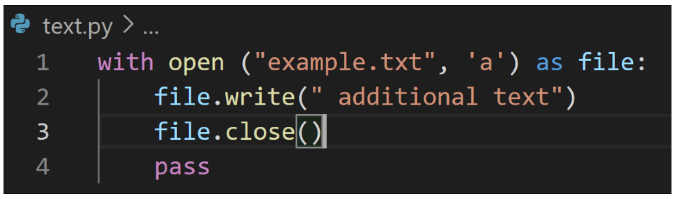
```python
file.close()
    pass
```

Additionally we look at say the first 20 characters of the text file we can further modify the `print()` function by using square brackets and specifying the number of string positions, see the next example:

```python
with open ("example.txt", 'r') as file:
    var = file.read()
    print(var[0:20])
    file.close()
    pass
```

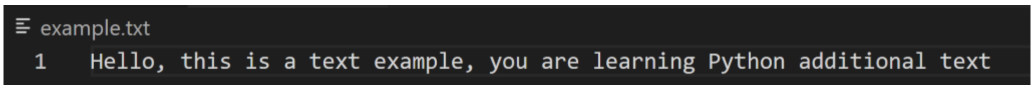
0:20 indicates that we are displaying the string positions from 0 to 20, both values are separated by a colon.

We can also use the append mode which just like in lists, will add on text to the end of a file. We can do this by using an a `'a'` for the mode. We can then remove the `file.read()`and add `file.write()` with our added string within the parenthesis, see the example:



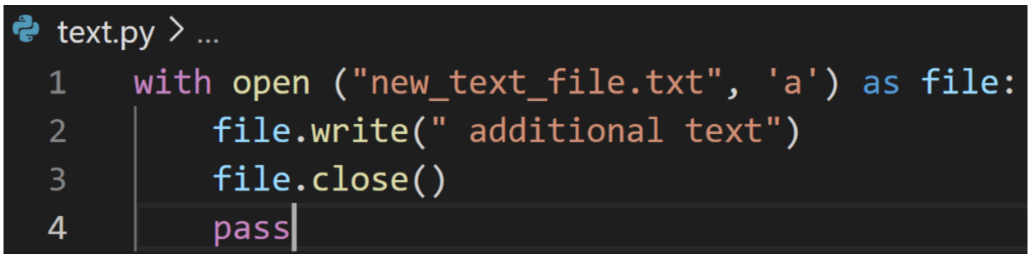
```
text.py > ...
with open ("example.txt", 'a') as file:
    file.write(" additional text")
    file.close()
    pass
```

Running this code doesn't seem to do anything but it in fact does. If we go to the example.txt file, we can see that the string " additional text" has been added to the end of it, see the example:



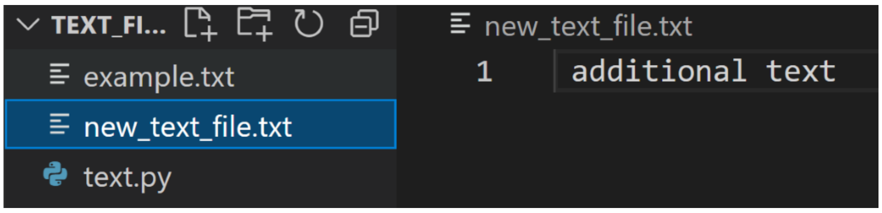
```
example.txt
Hello, this is a text example, you are learning Python additional text
```

The final mode is the write mode, using this we can actually create new files. We can do this by simply changing the file name.



```
text.py > ...
with open ("new_text_file.txt", 'a') as file:
    file.write(" additional text")
    file.close()
    pass
```

This will create a new file with the contents in the`file.write()` code block, below is the result:



```
new_text_file.txt
additional text
```

Finally it is important to understand that all the text saved is being saved as a string. You can confirm this by adding the `type()` function to a print statement with the variable name in the parenthesis.