Theory: Writing files

21 minutes 7 / 7 problems solved

Start practicing

3851 users solved this topic. Latest completion was 39 minutes ago.

One very important skill for programmers is knowing how to create new files or add information to existing ones. In this topic, you'll learn to write and append to files in Python.

§1. Write to file

Well, the first step of writing to a file in Python is, of course, opening said file for writing. The basic mode for writing is `'w'` which allows us to write text to the file. There are a few things we need to pay attention to. First, this mode allows us to create new files. This happens when the file we're trying to open doesn't exist yet. Second, if the file already exists, its contents will be overwritten when we open it for writing.

Now that the file is open, we can use the `write()` method. `file.write()` allows us to write strings to the file, other types of data need to be converted to a string beforehand. Let's see it in action.

```
file = open('test_file.txt', 'w', encoding='utf-8')
file.write('This is a line in a test file!')
file.close()
```

If we read the same file and print its contents, we'll get this:

```
# the output:
# This is a line in a test file!
```

Note that the strings are written to the file exactly as they are. When we call the `write()` method several times, the passed strings get written without any separators: no spaces, no newlines, just strings combined together into one.

§2. Writing multiple lines

If we want the file to have multiple lines, we need to specify where the ends of the lines should be. Lines in files are separated by **newline** escape sequences: `'\n'`, `'\r'` or `'\r\n'`. We'll settle on `'\n'` in this topic. Yet, keep in mind that this escape sequence depends on your operating system.

Suppose, we have a list of names and we want to write them to a file, each on a new line. This is how it can be done:

```
names = ['Kate', 'Alexander', 'Oscar', 'Mary']

name_file = open('names.txt', 'w', encoding='utf-8')

# write the names on separate lines
for name in names:
    name_file.write(name + '\n')

name_file.close()
```

If we print the lines of the file as a list, this is what we'll get:

```
['Kate\n', 'Alexander\n', 'Oscar\n', 'Mary\n']
```

As you can see, our file has four lines with the four names from the list. If we wanted the names to be on the same line separated by whitespace, we would do it similarly, but instead of `\n`, we would add a space.

Another method for writing the files is `file.writelines()`. `writelines()` takes an iterable sequence of strings and writes them to the file. Just like with `write()`, we need to specify the line separators ourselves. This is how we could've written the **names.txt** file using this method:

Current topic:

✓ Writing files Stage 2

Topic depends on:

✓ Files in Python Stage 2

Topic is required for:

✓ Context manager Stage 2

✓ Flush and file arguments of print

os module

Table of contents:

↑ Writing files

§1. Write to file

§2. Writing multiple lines

§3. Append to file

§4. Summary

Feedback & Comments

```python
names = ['Kate\n', 'Alexander\n', 'Oscar\n', 'Mary\n']

name_file = open('names.txt', 'w', encoding='utf-8')

name_file.writelines(names)

name_file.close()
```

In the end, we got the same file as before, the only difference is that the original strings had to come with the separator.

§3. Append to file

The `'w'` mode works perfectly fine if we don't care if anything gets deleted from the existing file. However, in many cases, we want to add some lines to the file, not overwrite it completely. How can we do that?

Well, we can use the `'a'` mode which stands for **append**. As you might have guessed, this allows us to write new strings to the file while keeping the existing ones.

Suppose, we want to add the name Rachel to the **names.txt.**

Here's how we can do that:

```python
name_file = open('names.txt', 'a', encoding='utf-8')

name_file.write('Rachel\n')

name_file.close()
```

Now if we print the lines of the file, the output will look like this:

```
['Kate\n', 'Alexander\n', 'Oscar\n', 'Mary\n', 'Rachel\n']
```

§4. Summary

In this topic, we've looked at a basic file operation — writing to files. Depending on whether we want to keep the original contents of the file or not, we can use the mode `'a'` or `'w'` respectively. The actual writing can be done with `write()` or `writelines()` methods.

Report a typo

Thanks for your feedback!

Write here how we could improve this theory

Start practicing

Comments (7) Hints (0) Useful links (0) Show discussion