Python-File Handling

File handling is an important part of any web application.

Python has several functions for creating, reading, updating, and deleting files.

File Handling

The key function for working with files in Python is the open() function.

The open() function takes two parameters; *filename*, and *mode*.

There are four different methods (modes) for opening a file:

"r" - Read - Default value. Opens a file for reading, error if the file does not exist

"**a" - Append** - Opens a file for appending, creates the file if it does not exist

"w" - Write - Opens a file for writing, creates the file if it does not exist

"**x" - Create** - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode

"**t" - Text** - Default value. Text mode

"b" - Binary - Binary mode (e.g. images)

Syntax

To open a file for reading it is enough to specify the name of the file:

f = open("demofile.txt")

The code above is the same as:

f = open("demofile.txt", "rt")

Because "r" for read, and "t" for text are the default values, you do not need to specify them.

Read Files:

demofile.txt

Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!

To open the file, use the built-in open() function.

Python-File Handling

The open() function returns a file object, which has a read() method for reading the content of the file:

Example:

f = open("demofile.txt", "r")
print(f.read())

```
f = open("demofile.txt", "r")

print(f.read())
```

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
```

If the file is located in a different location, you will have to specify the file path, like this:

Example

Open a file on a different location:

f = open("D:\\myfiles\welcome.txt", "r")
print(f.read())

Read Only Parts of the File

By default the read() method returns the whole text, but you can also specify how many characters you want to return:

Example

Return the 5 first characters of the file:

f = open("demofile.txt", "r")
print(f.read(**5**))

```
f = open("demofile.txt", "r")

print(f.read(5))
```

```
Hello
```

Read Lines

You can return one line by using the readline() method:

```
f = open("demofile.txt", "r")

print(f.readline())
```

```
Hello! Welcome to demofile.txt
```

Python-File Handling

```
f = open("demofile.txt", "r")

print(f.readline())
print(f.readline())
```

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
```

By looping through the lines of the file, you can read the whole file, line by line:

```
f = open("demofile.txt", "r")
for x in f:
  print(x)
```

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
```

Close Files

It is a good practice to always close the file when you are done with it.

```
f = open("demofile.txt", "r")

print(f.readline())

f.close()
```

```
Hello! Welcome to demofile.txt
```

Python File Write

Write to an Existing File

To write to an existing file, you must add a parameter to the open() function:

"a" - Append - will append to the end of the file

"w" - Write - will overwrite any existing content

#Append

```
f = open("demofile2.txt", "a")
f.write("Now the file has more content!")
f.close()

#open and read the file after the appending:
f = open("demofile2.txt", "r")
print(f.read())
```

```
Hello! Welcome to demofile2.txt
This file is for testing purposes.
Good Luck!Now the file has more content!
```

#overwrite:

```
f = open("demofile3.txt", "w")
f.write("Woops! I have deleted the content!")
f.close()

#open and read the file after the overwriting:
f = open("demofile3.txt", "r")
print(f.read())
```

```
Woops! I have deleted the content!
```

Create a New File

Python-File Handling

To create a new file in Python, use the open() method, with one of the following parameters:

"x" - Create - will create a file, returns an error if the file exist

"a" - Append - will create a file if the specified file does not exist

"w" - Write - will create a file if the specified file does not exist

Create a file called "myfile.txt":

f = open("myfile.txt", "x")

Result: a new empty file is created!

Delete a File

To delete a file, you must import the OS module, and run its os.remove() function

Remove the file "demofile.txt":

```
import os
os.remove("demofile.txt")
```

Check if File exist:

To avoid getting an error, you might want to check if the file exists before you try to delete it:

Example

Check if file exists, *then* delete it:

```
import os
if os.path.exists("demofile.txt"):
  os.remove("demofile.txt")
else:
  print("The file does not exist")
```

Delete Folder

To delete an entire folder, use the os.rmdir() method:

Example

Remove the folder "myfolder":

```
import os
os.rmdir("myfolder")
```

Python-File Handling

Thank you