

Lecture-20

The *close()* Method:

The `close()` method of a *file* object flushes any unwritten information and closes the file object, after which no more writing can be done.

Python automatically closes a file when the reference object of a file is reassigned to another file. It is a good practice to use the `close()` method to close a file.

Syntax

```
fileObject.close()
```

GALGOTIAS
UNIVERSITY

Example

```
# Open a file
fo = open("foo.txt", "wb")
print "Name of the file: ", fo.name
# Close opened file
fo.close()
```

This produces the following result –

```
Name of the file: foo.txt
```

GALGOTIAS
UNIVERSITY

Reading and Writing Files:

The *file* object provides a set of access methods to make our lives easier. We would see how to use *read()* and *write()* methods to read and write files.

The *write()* Method

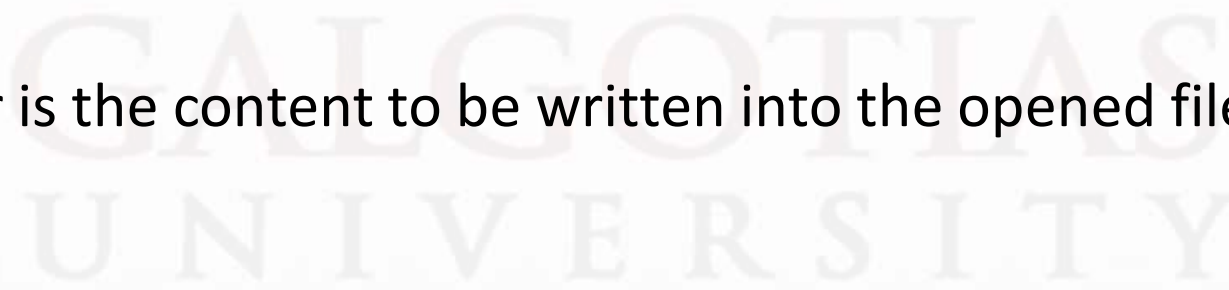
The *write()* method writes any string to an open file. It is important to note that Python strings can have binary data and not just text.

The *write()* method does not add a newline character ('\n') to the end of the string –

Syntax

```
fileObject.write(string)
```

Here, passed parameter is the content to be written into the opened file.



Example:

```
# Open a file
fo = open("foo.txt", "wb")
fo.write( "Python is a great language.\nYeah its great!!\n")
# Close opened file
fo.close()
```

The above method would create *foo.txt* file and would write given content in that file and finally it would close that file. If you would open this file, it would have following content.

```
Python is a great language.
Yeah its great!!
```

GALGOTIAS
UNIVERSITY

Example:

```
# Open a file
```

```
fd = open("ram.txt", "r+")
```

```
for i in range(10):
```

```
    fd.write( "This is line no. %d\n" %(i+1))
```

```
    print(fd.read())
```

```
# Close opened file
```

```
fd.close()
```



GALGOTIAS
UNIVERSITY

School of Basic and Applied Sciences

Course Code : BSCM 304

Course Name: Programming Using Python

OUTPUT:

This is line no. 0

This is line no. 1

This is line no. 2

This is line no. 3

This is line no. 4

This is line no. 5

This is line no. 6

This is line no. 7

This is line no. 8

This is line no. 9



GALGOTIAS
UNIVERSITY

Append (a):

Open a file

```
fd = open("ram.txt", "a")
```

```
for i in range(10):
```

```
    fd.write( "This is line no. %d\n" %(i+1))
```

```
    print(fd.read())
```

Close opened file

```
fd.close()
```

GALGOTIAS
UNIVERSITY

School of Basic and Applied Sciences

Course Code : BSCM 304

Course Name: Programming Using Python

Output:

This is line no. 0

This is line no. 1

This is line no. 2

This is line no. 3

This is line no. 4

This is line no. 5

This is line no. 6

This is line no. 7

This is line no. 8

This is line no. 9

This is line no. 1

This is line no. 2

This is line no. 3

This is line no. 4

This is line no. 5

This is line no. 6

This is line no. 7

This is line no. 8

This is line no. 9

This is line no. 10

GALGOTIAS
UNIVERSITY

The *read()* Method:

The *read()* method reads a string from an open file. It is important to note that Python strings can have binary data. apart from text data.

Syntax

`fileObject.read([count])` Here, passed parameter is the number of bytes to be read from the opened file. This method starts reading from the beginning of the file and if *count* is missing, then it tries to read as much as possible, maybe until the end of file.

GALGOTIAS
UNIVERSITY

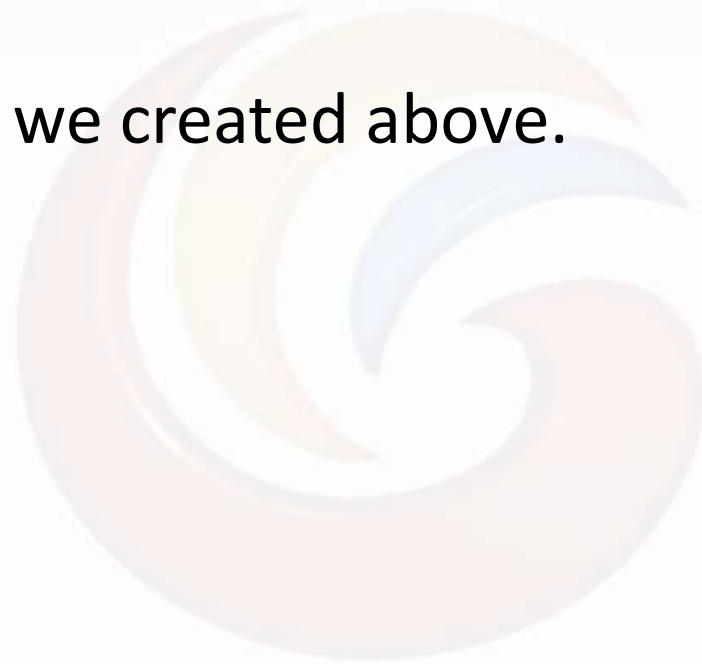
Example:

Let's take a file *foo.txt*, which we created above.

```
# Open a file
fo = open("foo.txt", "r+")
str = fo.read(10);
print "Read String is : ", str
# Close opened file
fo.close()
```

This produces the following result –

Read String is : Python is



GALGOTIAS
UNIVERSITY

File Positions:

The *tell()* method tells you the current position within the file; in other words, the next read or write will occur at that many bytes from the beginning of the file.

The *seek(offset[, from])* method changes the current file position. The *offset* argument indicates the number of bytes to be moved. The *from* argument specifies the reference position from where the bytes are to be moved.

If *from* is set to 0, it means use the beginning of the file as the reference position and 1 means use the current position as the reference position and if it is set to 2 then the end of the file would be taken as the reference position.

GALGOTIAS
UNIVERSITY

Example

Let us take a file *foo.txt*, which we created above.

```
# Open a file
```

```
fo = open("foo.txt", "r+")
```

```
str = fo.read(10)print "Read String is : ", str
```

```
# Check current position
```

```
position = fo.tell()
```

```
print "Current file position : ", position
```

GALGOTIAS
UNIVERSITY

Reposition pointer at the beginning once again

```
position = fo.seek(0, 0);
```

```
str = fo.read(10)
```

```
print "Again read String is : ", str
```

Close opened file

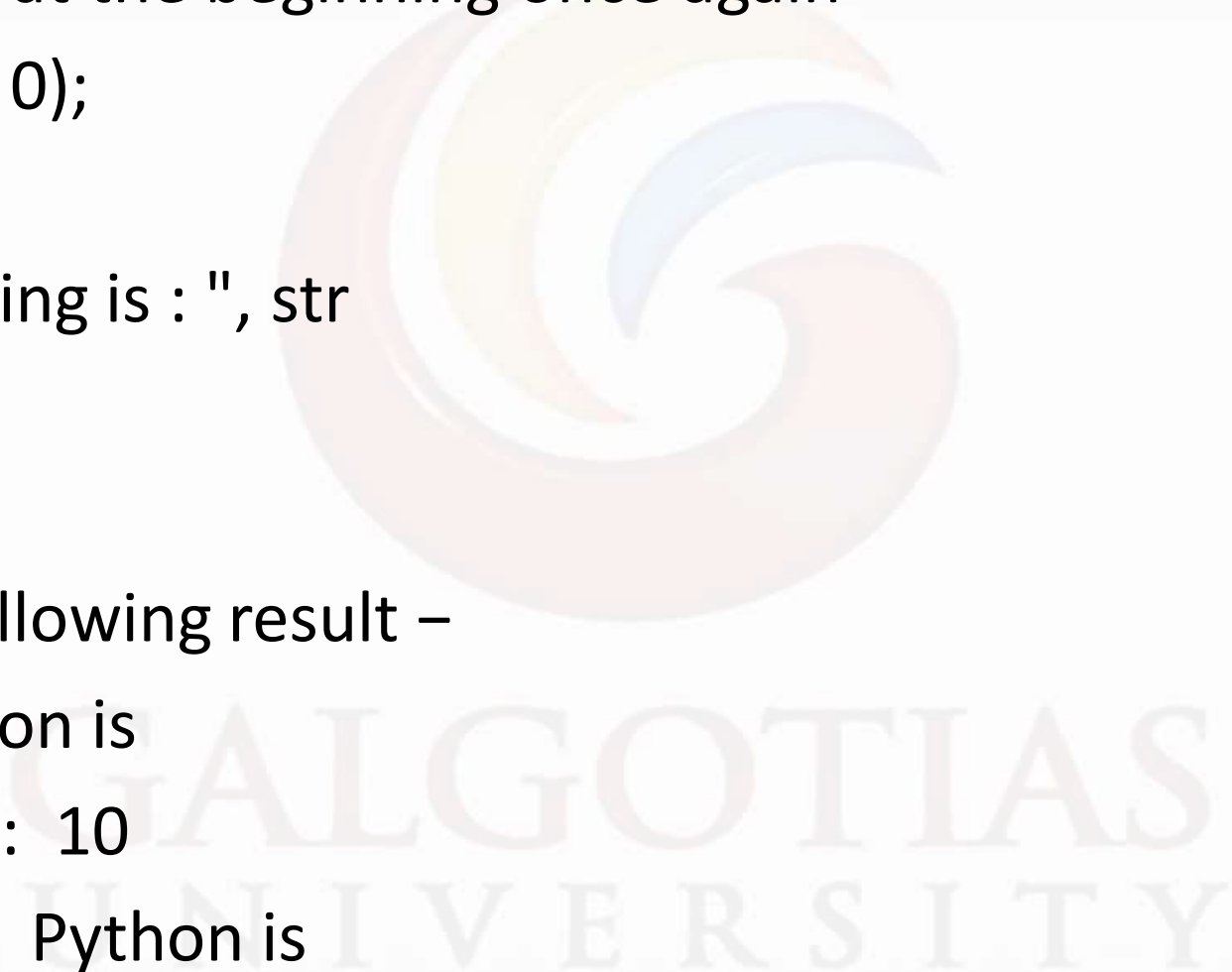
```
fo.close()
```

This produces the following result –

Read String is : Python is

Current file position : 10

Again read String is : Python is



References:

1. Introduction to Computation and Programming using Python, by John Guttag, PHI Publisher
2. Fundamentals of Python first Programmes by Kenneth A Lambert, Copyrighted material Course Technology Inc. 1 st edition (6th February 2009)
3. <https://www.geeksforgeeks.org/python-programming-language>

GALGOTIAS
UNIVERSITY

*****END OF THE LECTURE*****

*****THANK YOU*****

GALGOTIAS
UNIVERSITY