Python sleep()

The sleep() function suspends (waits) execution of the current thread for a given number of seconds.

Python has a module named time which provides several useful functions to handle time-related tasks. One of the popular functions among them is `sleep()` .

The `sleep()` function suspends execution of the current thread for a given number of seconds.

Example 1: Python sleep()

```
import time

print("Printed immediately.")
time.sleep(2.4)
print("Printed after 2.4 seconds.")
```

Here's how this program works:

- `"Printed immediately"` is printed
- Suspends (Delays) execution for 2.4 seconds.
- `"Printed after 2.4 seconds"` is printed.

As you can see from the above example, `sleep()` takes a floating-point number as an argument.

Before Python 3.5, the actual suspension time may be less than the argument specified to the `time()` function.

Since Python 3.5, the suspension time will be at least the seconds specified.

Example 2: Python create a digital clock

```
import time

while True:
  localtime = time.localtime()
  result = time.strftime("%I:%M:%S %p", localtime)
  print(result)
  time.sleep(1)
```

In the above program, we computed and printed the current local time inside the infinite while loop. Then, the program waits for 1 second. Again, the current local time is computed and printed. This process goes on.

When you run the program, the output will be something like:

```
02:10:50 PM
02:10:51 PM
02:10:52 PM
02:10:53 PM
02:10:54 PM
... .. ...
```

Here is a slightly modified better version of the above program.

```
import time

while True:
  localtime = time.localtime()
  result = time.strftime("%I:%M:%S %p", localtime)
  print(result, end="", flush=True)
  print("\r", end="", flush=True)
  time.sleep(1)
```

To learn more, visit digital clock in Python shell.

Multithreading in Python

Before talking about `sleep()` in multithreaded programs, let's talk about processes and threads.

A computer program is a collection of instructions. A process is the execution of those instructions.

A thread is a subset of the process. A process can have one or more threads.

Example 3: Python multithreading

All the programs above in this article are single-threaded programs. Here's an example of a multithreaded Python program.

```
import threading

def print_hello_three_times():
  for i in range(3):
    print("Hello")

def print_hi_three_times():
    for i in range(3):
      print("Hi")

t1 = threading.Thread(target=print_hello_three_times)
t2 = threading.Thread(target=print_hi_three_times)

t1.start()
t2.start()
```

When you run the program, the output will be something like:

```
Hello
Hello
Hi
Hello
Hi
Hi
```

The above program has two threads *t1* and *t2*. These threads are run using `t1.start()` and `t2.start()` statements.

Note that, *t1* and *t2* run concurrently and you might get different output.

Visit this page to learn more about Multithreading in Python.

time.sleep() in multithreaded programs

The `sleep()` function suspends execution of the current thread for a given number of seconds.

In case of single-threaded programs, `sleep()` suspends execution of the thread and process. However, the function suspends a thread rather than the whole process in multithreaded programs.

Example 4: sleep() in a multithreaded program

```
import threading
import time

def print_hello():
  for i in range(4):
    time.sleep(0.5)
    print("Hello")

def print_hi():
    for i in range(4):
      time.sleep(0.7)
      print("Hi")

t1 = threading.Thread(target=print_hello)
t2 = threading.Thread(target=print_hi)
t1.start()
t2.start()
```

The above program has two threads. We have used `time.sleep(0.5)` and `time.sleep(0.75)` to suspend execution of these two threads for 0.5 seconds and 0.7 seconds respectively.

Recommended Reading: Python time.sleep() sleeps thread