





Select lesson text to explore the Explain with AI Premium Offering

Daemon Thread

This lesson describes how daemon threads exit as soon as the main program thread exits.

Daemon Thread

In Greek mythology, a daemon was a supernatural being of a nature between gods and humans whereas in the computer science realm a daemon is a computer program that runs as a background process rather than being under the direct control of an interactive user. A daemon thread in Python runs in the background. The difference between a regular thread and a daemon thread is that a Python program will not exit until all regular/user threads terminate. However, a program may exit if the daemon thread is still not finished.

For folks coming from Java background, this is similar to how JVM treats daemon threads. Daemon threads don't hold up a Java program from terminating.

Marking a thread Daemon

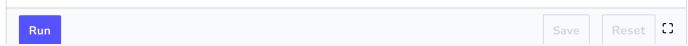
We can mark a thread daemon by passing in true for the daemon field in the Thread class's constructor.

Creating a daemon thread

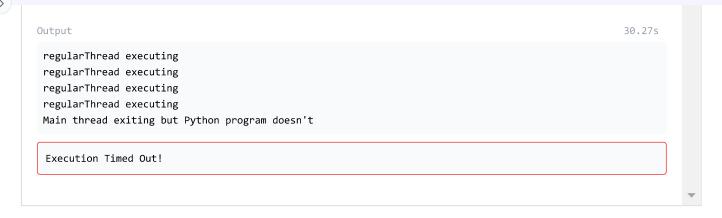
```
daemonThread = Thread(target=daemon_thread_task, daemon=True)
```

In the below snippet the main thread creates a **non-daemonic** thread and exits. However, the program continues to run. The code widget throws an error because the execution times out.

```
from · threading · import · Thread
    from · threading · import · current_thread
 3
    import·time
 4
 5
 6 def·regular_thread_task():
 7
    ····while·True:
    .....print("{0}.executing".format(current_thread().getName()))
 8
 9
     ·····time.sleep(1)
10
 11 \quad \texttt{regularThread} \cdot \texttt{-} \cdot \texttt{Thread} (\texttt{target=regular\_thread\_task}, \cdot \texttt{name="regularThread"}, \cdot \texttt{daemon=False}) 
12
    regularThread.start()··#·start·the·regular·thread
13
14
    time.sleep(3)
15
16
    print("Main·thread·exiting·but·Python·program·doesn't")
```



Select lesson text to explore the Explain with AI Premium Offering



In contrast, the runnable snippet below will exit even though the daemon thread is still executing.

```
1 from threading import Thread
 2
    from threading import current_thread
 3
    import time
 4
 5
    def daemon_thread_task():
 6
        while True:
 7
            print("{0} executing".format(current_thread().getName()))
 8
            time.sleep(1)
 9
   regularThread = Thread(target=daemon_thread_task, name="daemonThread", daemon=True)
10
    regularThread.start() # start the daemon thread
11
12
13
    time.sleep(3)
14
   print("Main thread exiting and Python program too")
 Run
                                                                                                         X
Output
                                                                                                      3.7s
 daemonThread executing
 daemonThread executing
 daemonThread executing
 Main thread exiting and Python program too
```

Daemon threads are shut-down abruptly. Resources such as open files and database connections may not shut-down properly and daemon threads are not a good choice for such tasks. One final caveat to

remember is that if you don't specify the daemon parameter in the constructor then the daemonic property is inherited from the current thread.

Select lesson text to explore the Explain with AI Premium Offering

>



Subclassing Thread





Lock





