Threading

Jan Popko

Python Advanced

Business Trends Academy (BTA) GmbH

bta
BUSINESS TRENDS ACADEMY

Nestorstraße 36
D-10709 Berlin

Tel.: +49 (0) 30 894 087 57
Fax: +49 (0) 30 895 429 94

Geschäftsführer:
Gabriele Fleischmann-Hahn
Maxi-Marlen Fleischmann
Hauptsitz des Unternehmens:
Nestorstraße 36, D-10709 Berlin
HRB 115251 B / Amtsgericht Berlin Charlottenburg
Steuer-Nr. 27/248/31179

Threads

- kein echtes Multitasking, da nur ein Prozessor genutzt wird
- CPU springt schnell zwischen den Threads und suggeriert es
- sinnvoll für parallele Programmierung, hilft down-Zeiten zu minimieren
- Threads, die in anderen Threads erstellt werden, erben deren Eigenschaften

```
import threading

def func(x):
        # does something

# Thread Konstruktor – startet noch keinen Thread
t = threading.Thread(target = func, args = [x])
# startet Thread
t.start()
# wartet bis der Thread terminiert
t.join()
```

Methoden für Thread

.name - gibt Namen des Threads wieder(Standard: Thread-N)
- ersetzt .getName() und setName()

.ident – ist nicht Null Integer,
- None, wenn der Thread noch nicht gestartet wurde
- kann wieder verwendet werden, wenn Thread beendet ist

.is_alive() - gibt an ob der Thread noch nicht beendet ist

.daemon – gibt an ob der Thread ein Daemon Thread ist
- muß vor .start() gesetzt werden
- ersetzt isDaemon() und setDaemon

Methoden für threading

threading.active_count() - gibt anzahl der aktiven Threads wieder
- ist gleich der Länge enumerate() Liste

threading.current_thread() - gibt den gerade aktiven Thread wieder
- wenn kein Thread mit threading erstellt wurde gibt die Methode ein Dummy Thread Objekt(mainthread)

threading.get_ident() - gibt einen identifier zurück
- ist ein Integer != 0, hat keine direkte Bedeutung

threading.enumerate() - gibt eine Liste aller aktiven Threads wieder
- inclusive dummy und daemonic Threads
- keine Threads, die beendet sind oder noch nicht gestartet haben

threading.main_thread() - gibt den Hauptthread wieder

Daemon Thread

- Thread der automatisch terminiert, sobald der Hauptthread beendet ist

- Initialisierung:

Entweder:

```
t = threading.Thread(target=func,name=None,args=(x,),daemon=True)
t.start()
```

Oder:

```
t = threading.Thread(target=func,name=None,args=(x,))
t.daemon = True
t.start()
```

Modul Queue

- produziert multi-producer, multi-consumer Schlangen
- kann Maximalwert der Elemente in der Schlange bekommen (maxsize)
- bei maxsize <=0 ist die Schlange unendlich, Standard: maxsize = 0
- Drei Varianten:

Queue.Queue()- Konstruktor für “first in first out“ Schlange

Queue.LifoQueue()- Konstruktor für “last in first out“ Schlange

Queue.PriorityQueue()- gibt zuerst kleinstes Element aus

- kleinstes Element ist: sorted(list(entries))[0])
- typischer Eintrag als Tupel: (priority_number,data)

Methoden für Queue

.put(item) – hängt Element an die Schlange, wenn sie nicht voll ist

.get(item) – gibt Element aus der Schlange und entfernt es daraus

- Die folgenden Methoden sind keine Garantie dafür, daß .put() und .get() funktionieren:

  .qsize() - gibt die geschätzte länge der Schlange wieder

  .empty() - True wenn die Schlange leer ist, sonst False

  .full() - True wenn die Schlange voll ist, sonst False

Business Trends Academy (BTA) GmbH
Nestorstraße 36
D-10709 Berlin
Tel.: +49 (0) 30 894 087 57
Fax: +49 (0) 30 895 429 94
Geschäftsführer:
Gabriele Fleischmann-Hahn
Maxi-Marlen Fleischmann
Hauptsitz des Unternehmens:
Nestorstraße 36, D-10709 Berlin
HRB 115251 B / Amtsgericht Berlin Charlottenburg
Steuer-Nr. 27/248/31179