COMPSCI 340/ SOFTENG 370

PYTHON TUTORIAL

About US

- **VASANTH BORAIYAN**

  Qualifications: Master of Engineering Studies(Software Engineering) 2015 – present.

  10+ years of experience in IT field(Micro Soft Business Intelligence, MS SQL Server)

  3+ years of experience as Lecturer at SNR Sons College Coimbatore

  Master of Computer Applications at Bharathiar University Coimbatore – 2000

  Bachelor of Science in Computer Science – 1996

  Email: vbor051@aucklanduni.ac.nz

- **MELWYN ‘MEL’ PEREIRA**

  Email – mper930@aucklanduni.ac.nz

  Twitter - @CRUDlyf

  Softeng Slack - @mel

  Qualifications: MASTER OF ENGINEERING STUDIES (SOFTWARE)

How can you contact me?

- Email: vbor051@aucklanduni.ac.nz
- How can you meet me?

  Weekly office hours: Wednesday 12:00 PM to 01:00 PM

  Location: in the area across from Room 303.488

  By appointment: please, email me to set time and place

Agenda

- Introduction to Multithreading
- Why Thread stop is not recommended in programming
- Solutions to Thread Stop
- Threading.Event
- Threading.Lock
- Threading.Rlock
- Useful Hints

1) Introduction to Multithreading:

i) What is multithreading?

- Multithreading extends the idea of multitasking into applications, so you can subdivide specific operations within a single application into individual threads. Each of the threads can run in parallel. The OS divides processing time not only among different applications, but also among each thread within an application.
- operating system process to manage its use by more than one user at a time .
- multiple threads to exist within the context of a process.
- status of the work on that thread is kept track of until the work is completed.

ii) Why do we use multithreading?

Multiple programs executed at same time(waiting time reduced)

Optimum usability of the resource(CPU)

Using multiple processors(dual core)

iii) How to do multi threading in Python?

a python program

2) Why should Thread Stop, Thread Suspend, Thread Resume – not recommended in our programming.

- It is unsafe practice
- thread unlock all the monitors.
- inconsistent state.
- Damaged objects can cause unknown results.
- Corruption occurs

3) Solution to Thread Stop

- Introduce a variable – the target should check the variable regularly

  for example the variable name can be Thread_stop_indicator

  if Thread_stop_indicator = 0, then operation should be still executing.

  if the Thread_stop_indicator = 1, then operation should be stop running.

Or

In python you can use setDaemon property to True, and based on that manipulation can be executed.

4) threading.**Event**()

- **What**: Returns a New Event Object
- **Why**: Useful to control the process.
- **How**: An event manages a flag that can be set to true with the set() method and reset to false with the **clear()** method. The **wait()** method blocks until the flag is true

isSet() - Return true if and only if the internal flag is true.

set() – Set the internal flag to true.

clear() – Set the internal flag to false –thread is blocked until it is set to True

wait() – Return if internal flag is true, else it will block.

threading.Lock

- **What**: returns a new primitive lock object.
- **Why**: Once a thread has acquired it, subsequent attempts to acquire it block, until it is released; any thread may release it.
- **How**: Lock.acquire – if true then thread blocked, if false the released.

  Lock.release – To release a lock, When the lock is locked, reset it to unlocked, and return. If any other threads are blocked waiting for the lock to become unlocked, allow exactly one of them to proceed.

threading.RLock

- **What**: function that returns a new reentrant lock object, A reentrant lock must be released by the thread that acquired it. Once a thread has acquired a reentrant lock, the same thread may acquire it again without blocking;