

Клиент-серверные приложения на Python

Урок 8

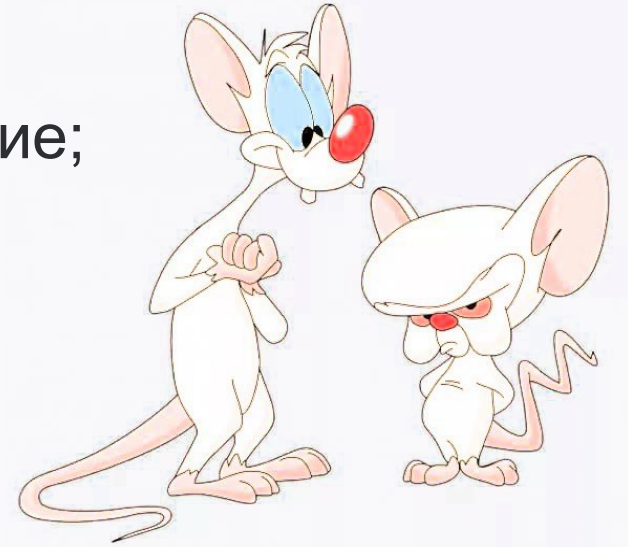


Потоки

Введение в потоки. Введение в многопоточное программирование.
Модуль `threading`. Прimitives синхронизации. Модуль `Queue`.
Модуль `multiprocessing`

Цели урока

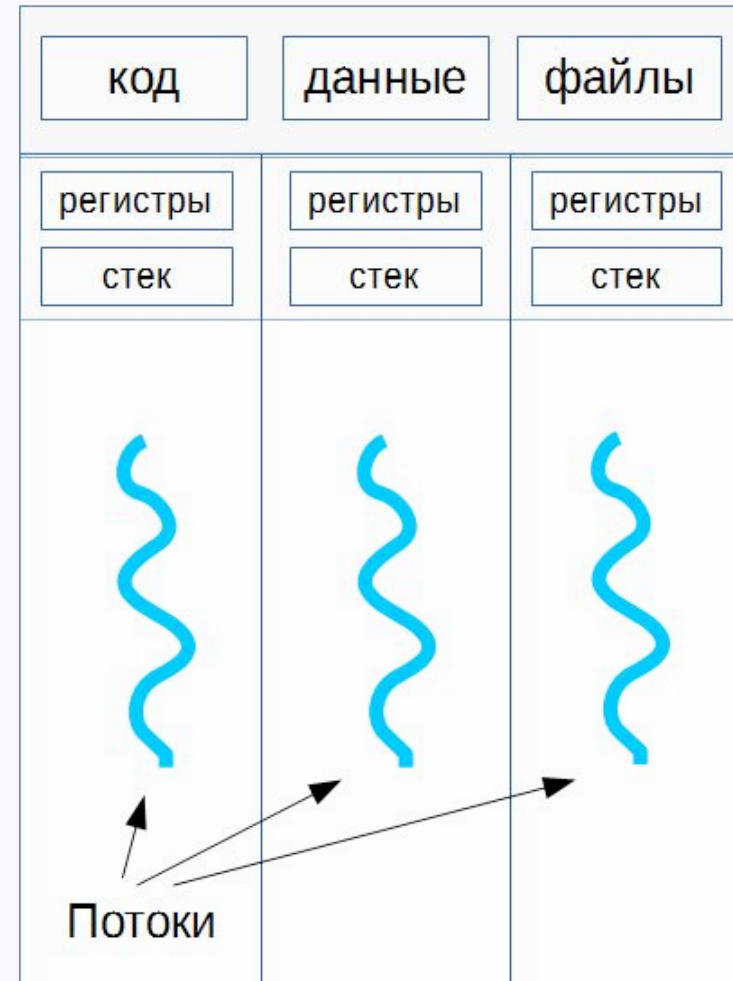
1. Окунуться в параллельное программирование;
2. Не испугаться GIL;
3. Вспомнить про сокеты.



Введение в параллельный мир



Однопоточный процесс



Многопоточный процесс

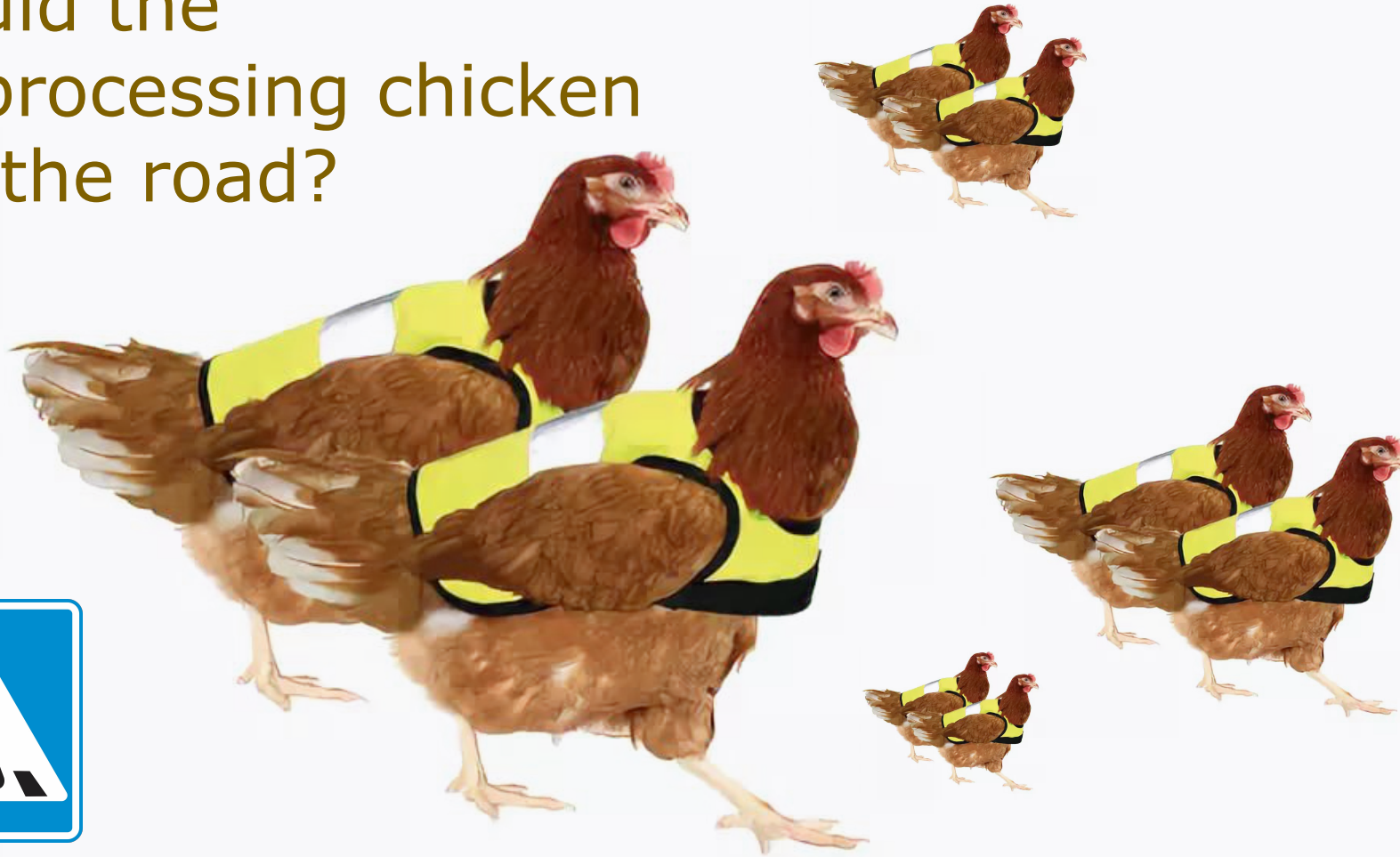


Кратко о многопоточности



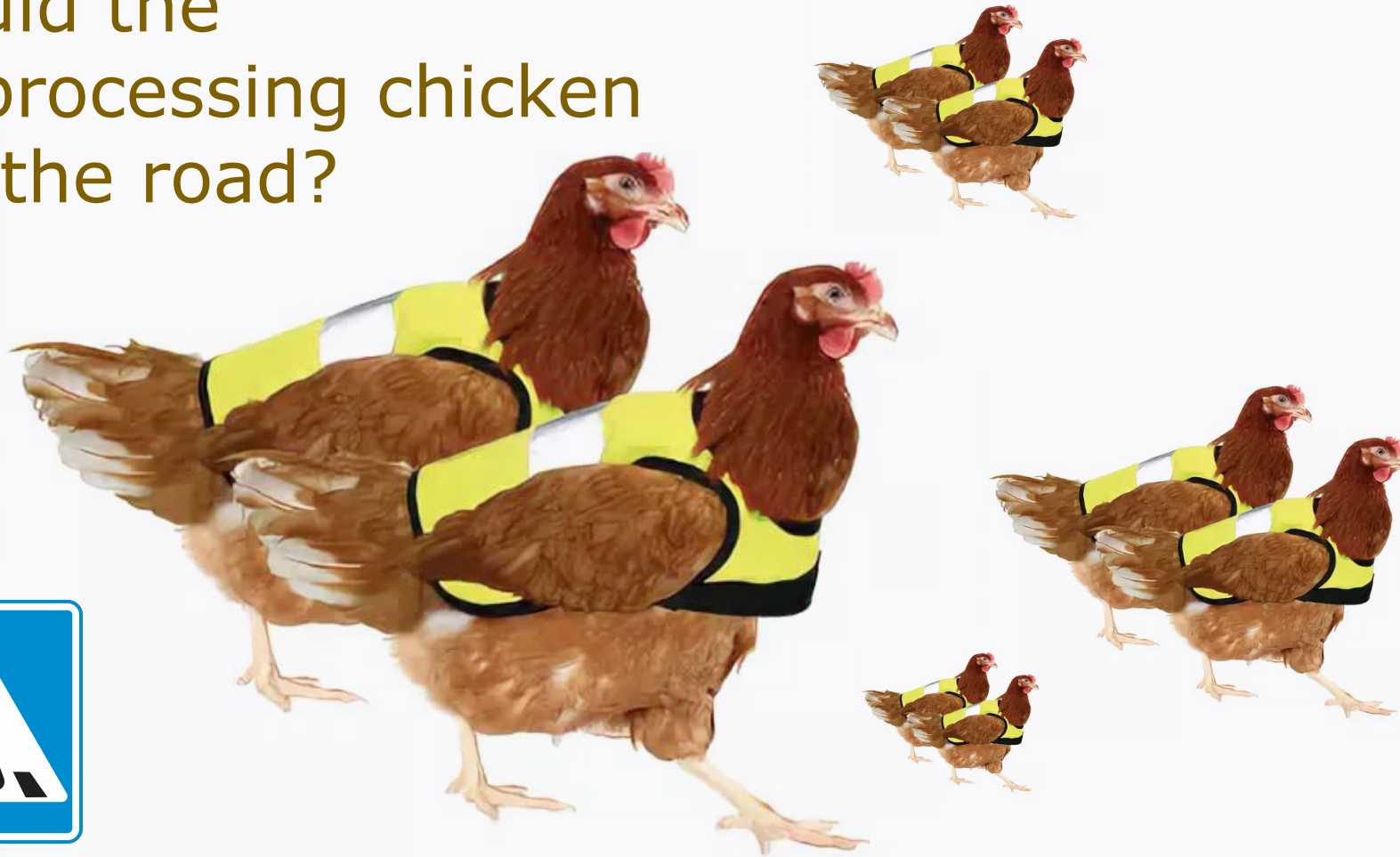
Кратко о многопоточности...

Why did the
multiprocessing chicken
cross the road?



Кратко о многопоточности

Why did the
multiprocessing chicken
cross the road?



to To other side. get the
Jason Whittington

Модуль threading

```
import threading
import time

def clock(interval):
    while True:
        print("The time is %s" % time.ctime())
        time.sleep(interval)

if __name__ == "__main__":
    p = threading.Thread(target=clock, args=(15, ))
    p.start()
```



Модуль threading

```
import threading
import time
```

```
def clock(interval):
    while True:
        print("The time is %s" % time.ctime())
        time.sleep(interval)
```

```
if __name__ == "__main__":
    p = threading.Thread(target=clock, args=(15, ))
    p.start()
```

Создать новый поток

Функция для потока

Аргументы для функции



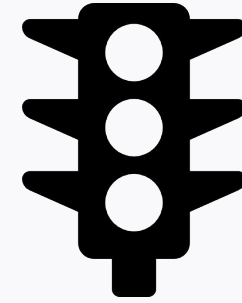
Взаимодействие потоков



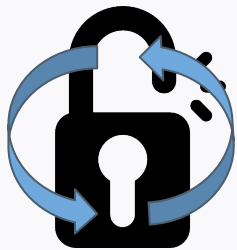
Примитивы синхронизации



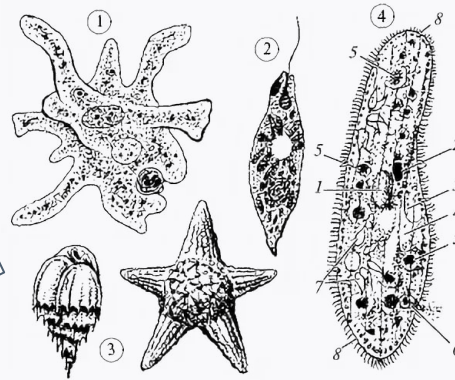
Lock



Semaphore



RLock



BoundedSemaphore



Основные методы



`.acquire()` - **Захватить**



`.release()` - **Отпустить**



Модуль queue



Queue (FIFO)

LifoQueue (LIFO)

PriorityQueue
(Очередь
с приоритетами)



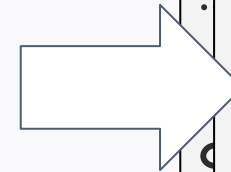
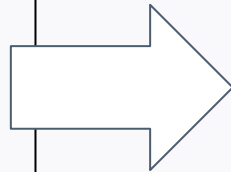
Модуль queue

Producer

```
q.put(data)
```

... что-то ещё

```
q.join()
```



Consumer

Consumer

Consumer

```
data = q.get()
```

... Обработать данные

```
q.task_done()
```

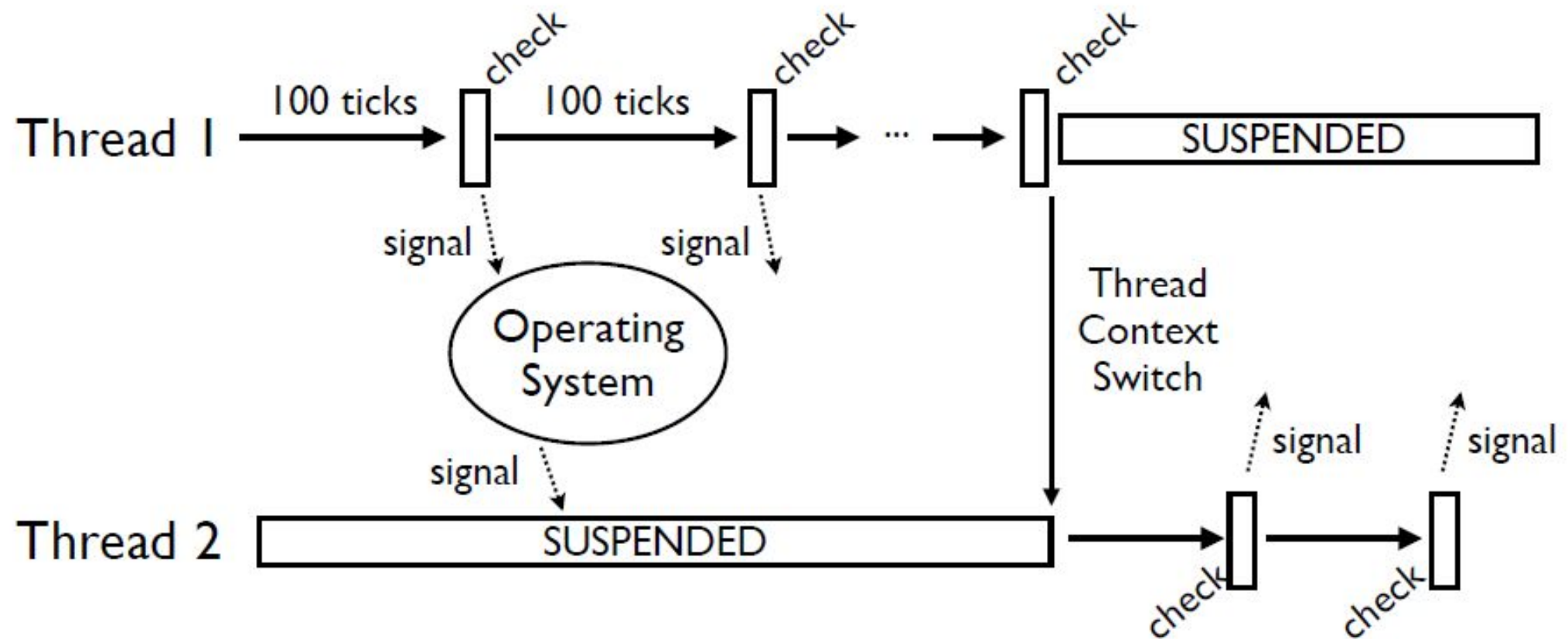


GIL (Global Interpreter Lock)

- Глобальный мьютекс.
- Только один поток имеет доступ к внутренностям интерпретатора.
- Несколько потоков могут замедлить работу.
- GIL не мешает при работе с вводом/выводом.



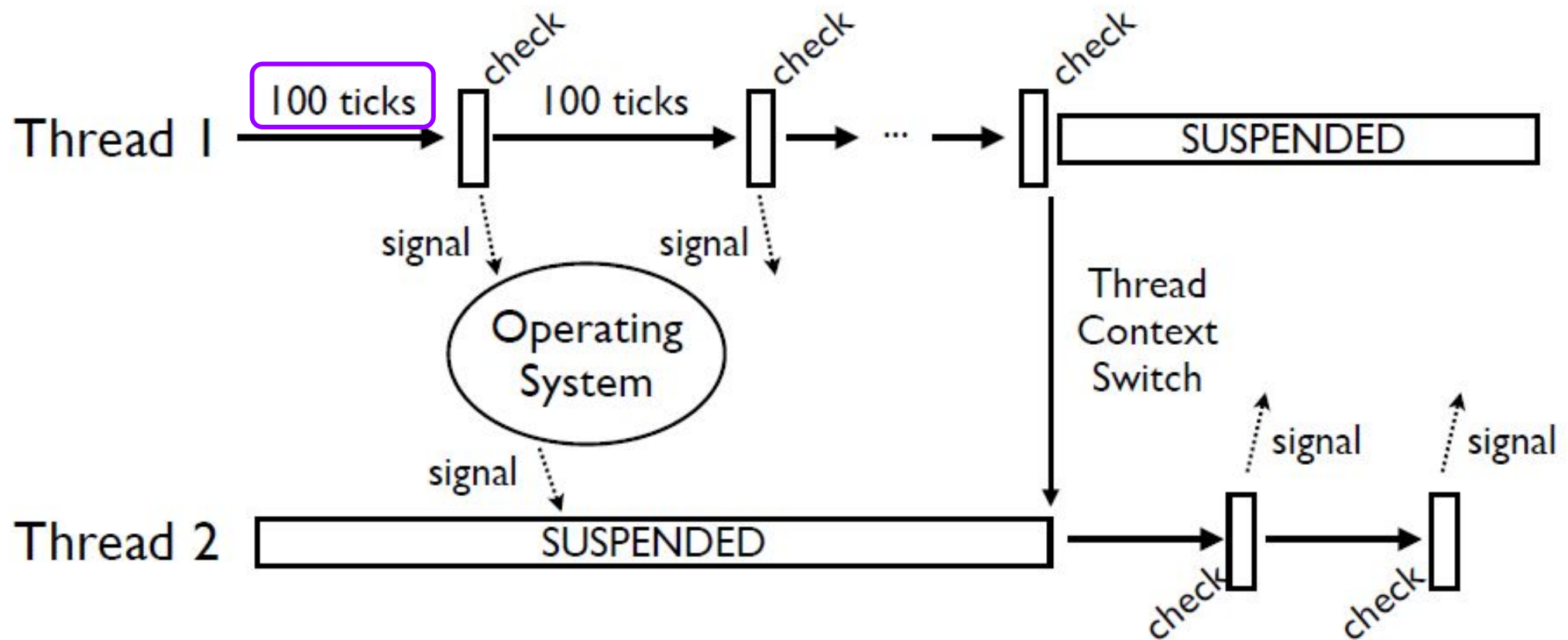
GIL (Global Interpreter Lock)



Старая версия GIL



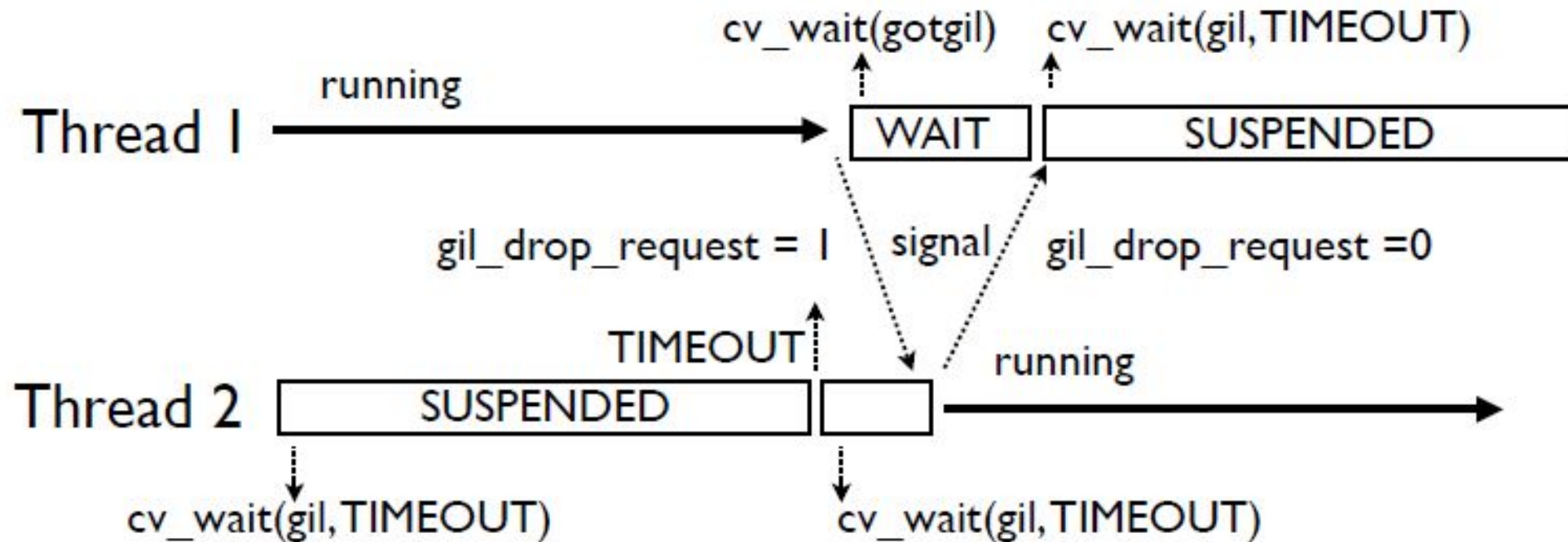
GIL (Global Interpreter Lock)



Старая версия GIL



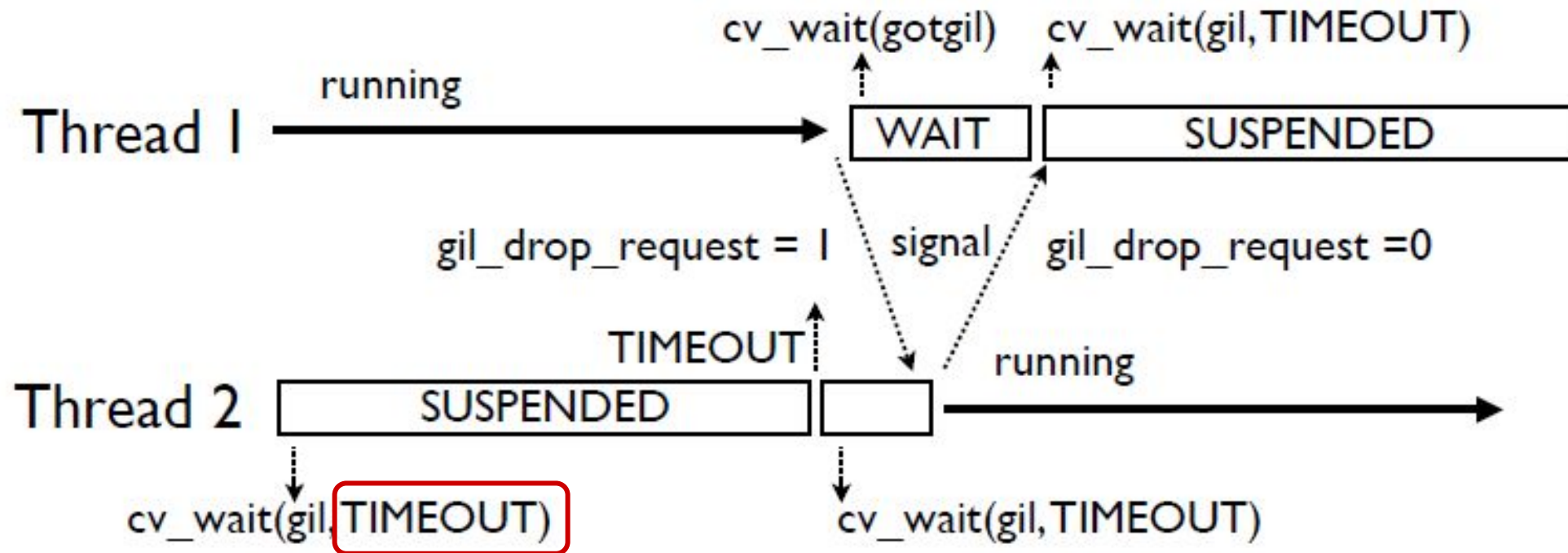
GIL (Global Interpreter Lock)



Новая версия GIL



GIL (Global Interpreter Lock)



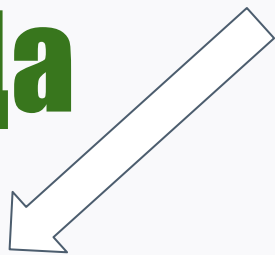
Новая версия GIL



Python и потоки

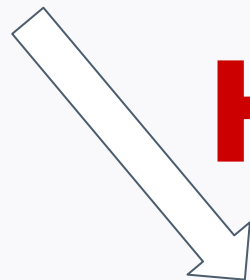
Thread

Да



Ввод-вывод
(файлы, сокеты, БД, ...)

Нет



CPU-зависимые операции
(вычисления)



Модуль multiprocessing



Модуль multiprocessing

Многие методы и интерфейсы идентичны модулю `threading`



Модуль multiprocessing

Многие методы и интерфейсы идентичны модулю `threading`

```
import multiprocessing
import time

def clock(interval):
    while True:
        print("The time is %s" % time.ctime())
        time.sleep(interval)

if __name__ == "__main__":
    p = multiprocessing.Process(target=clock, args=(15, ))
    p.start()
```



Модуль multiprocessing

Многие методы и интерфейсы идентичны модулю `threading`

```
import multiprocessing
import time
```

```
def clock(interval):
    while True:
        print("The time is %s" % time.ctime())
        time.sleep(interval)
```

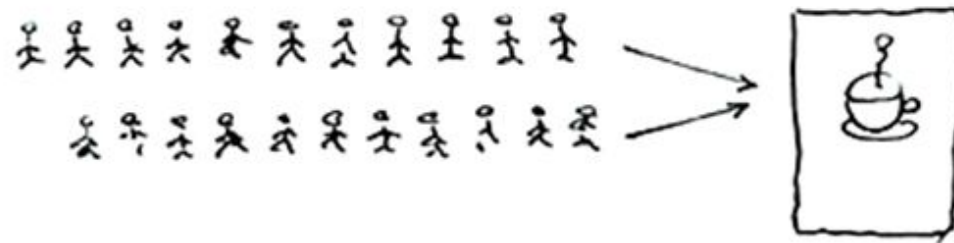
```
if __name__ == "__main__":
    p = multiprocessing.Process(target=clock, args=(15, ))
    p.start()
```

Обязательно, т.к. новый процесс загрузит скрипт полностью

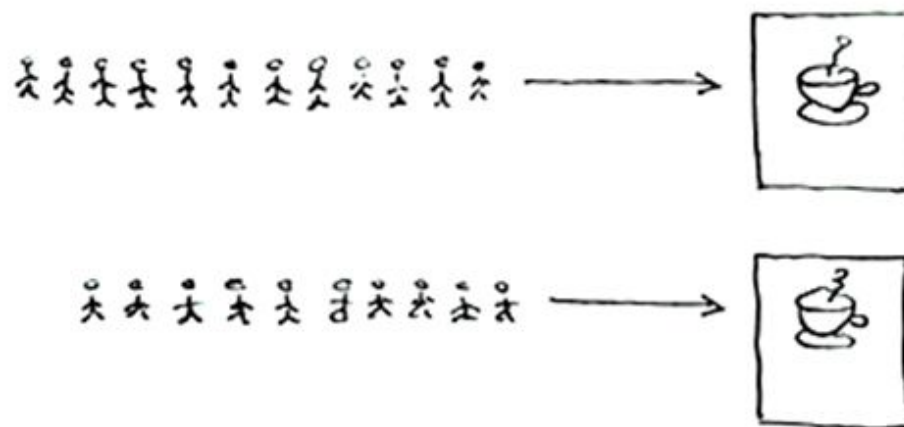


Такая разная параллельность

Concurrent = Two Queues One Coffee Machine



Parallel = Two Queues Two Coffee Machines



© Joe Armstrong 2013



Практическое задание



Практическое задание

1. На клиентской стороне реализовать прием и отправку сообщений с помощью потоков в P2P-формате (обмен сообщениями между двумя пользователями).

Итогом выполнения практических заданий первой части продвинутого курса Python стал консольный мессенджер. Усовершенствуем его во второй части: реализуем взаимосвязь мессенджера с базами данных и создадим для него графический пользовательский интерфейс.



Дополнительные материалы

1. Андрей Светлов. Загадочный GIL:
<http://asvetlov.blogspot.ru/2011/07/gil.html>;
2. Хабрахабр. Правильное использование Qthread:
<https://habrahabr.ru/post/150274/>;
3. Python Concurrency Cheatsheet:
<https://www.pythonsheets.com/notes/python-concurrency.html>;
4. Синхронизация потоков в Python:
<http://python-3.ru/page/sinhronizacija-potokov-v-python>.

