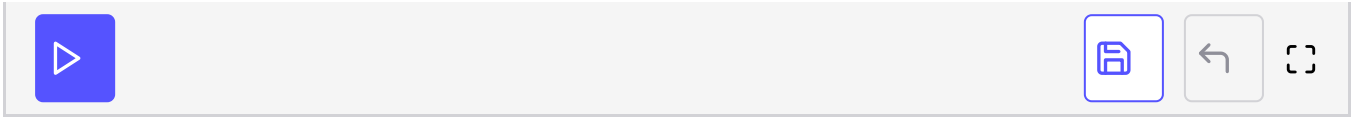
Locks

The multiprocessing module supports locks in much the same way as the threading module does. All you need to do is import **Lock**, acquire it, do something and release it. Let's take a look:

```
from multiprocessing import Process, Lock


def printer(item, lock):
    """
    Prints out the item that was passed in
    """
    lock.acquire()
    try:
        print(item)
    finally:
        lock.release()

if __name__ == '__main__':
    lock = Lock()
    items = ['tango', 'foxtrot', 10]
    for item in items:
        p = Process(target=printer, args=(item, lock))
        p.start()
```



Here we create a simple printing function that prints whatever you pass to it. To prevent the proceses from interfering with each other, we use a Lock object. This code will loop over our list of three items and create a process for each of them. Each process will call our function and pass it one of the items from the iterable. Because we're using locks, the next process in line will wait for the lock to release before it can continue.