PipeCommunication

Esta classe é responsável por implementar comunicação entre duas entidades.

Esta classe necessita de receber pelo menos as duas entidades (por conveniência, vamos assumir que cada entidade é uma função), sendo opcionalmente o fornecimento de um timeout para a comunicação. A cada entidade é lhe atribuída uma extremidade do Pipe. Depois vamos criar um processo para cada entidade, processo este que terá como target (ou seja, vai realizar essa função) a entidade em si e cujo argumento será a extremidade anteriormente atribuída (a função vai ser aplicada àquela extremidade).

Esta classe tem um método `run()` que é responsável por fazer correr os processos criados.

In [1]:
```
from multiprocessing import Pipe, Process
```

In [2]:
```
class PipeCommunication():
    def __init__(self,leftE, rigthE, timeout=None):
        '''
            Classe responsável por ligar 2 entidades através de um Pipe para poderem comunicar entre si.
            A cada entidade será atribuída uma extremidado do pipe.
            Será criado um processo para cada entidade onde o processo terá como alvo a entidade respetiva e
                passar-lhe-á como argumento a extremidade da conexão que lhe é correspondente.
        '''
        left_end, right_end = Pipe()
        if (timeout == None):
            self.timeout = 30
        else:
            self.timeout = timeout
        self.left_process = Process(target = leftE, args=(left_end,))
        self.right_process = Process(target = rightE , args=(right_end,))

    def run(self):
        self.left_process.start()
        self.right_process.start()
        self.left_process.join(self.timeout)
        self.right_process.join(self.timeout)
```

A célula seguinte é uma célula de teste.

In [3]:
```
def leftE(conn):
    print('LeftE: I am leftE! Sending message!')
    conn.send(b'Ola eu sou a entidade da Esquerda!')
    conn.close()
```

A célula seguinte é célula de teste.

In [4]:
```
def rightE(conn):
    print('RightE: Eu sou a RightE! Receiveng messages!')
    msg = conn.recv()
    print('RightE leu: ' + msg.decode())
    conn.close()
    try:
        print(conn.recv())
    except:
        print('Conexão já foi fechada! Não há nada para ler')
```

In [5]:
```
def teste():
    PipeCommunication(leftE, rightE, timeout=30).run()
```

In [6]:
```
teste()
```

```
LeftE: I am leftE! Sending message!
RightE: Eu sou a RightE! Receiveng messages!
RightE leu: Ola eu sou a entidade da Esquerda!
Conexão já foi fechada! Não há nada para ler
```