Geek University

Evolua seu lado geek!

www.geekuniversity.com.br

Compartilhando estados entre processos

Compartilhando estados entre processos

Antes de partimos para detalhes sobre este tema, gostaria de relembrar que o recomendado é compartilhar nada ou o mínimo possível de recursos/memória/dados entre threads/processos.

Isso porque a partir do momento que recursos/memória/dados são compartilhados, tudo se torna mais complexo e o gerenciamento das threads/processos deixará seu código mais complicado.

Compartilhando estados entre processos

Quando falamos em compartilhar “estados” você deve lembrar de conceitos da orientação a objetos onde classes possuem estados (atributos) e métodos.

A ideia aqui não é trabalhar com objetos, mas com com valores simples realizando o **Compartilhamento de Memória**.

Compartilhando estados entre processos

Compartilhamento de Memória

Recomendado caso você precise compartilhar uma variável de valor único ou array entre seus processos:

multiprocessing.Value
multiprocessing.Array

Compartilhando estados entre processos

Compartilhamento de Memória

multiprocessing.**Value**
multiprocessing.**Array**

Estas classes utilizam de ctypes (código em linguagem C interagindo com Python) e desta forma é necessário que especifiquemos o tipo de dado a ser utilizado de acordo com a tabela ao lado:

ctypes

ctypes type	C type	Python type	Type code
c_bool	_Bool	bool(1)	
c_char	char	1-character bytes object	'c'
c_wchar	wchar_t	1-character string	'u'
c_int	int	int	'i'
c_long	long	int	'l'
c_float	float	float	'f'
c_char_p	char * (NUL terminated)	bytes object or None	
c_wchar_p	wchar_t * (NUL terminated)	string object or None	
c_void_p	void *	int or None	

Compartilhando estados entre processos

Compartilhamento de Memória

multiprocessing.**Value**
multiprocessing.**Array**

Estas classes utilizam de ctypes (código em linguagem C interagindo com Python) e desta forma é necessário que especifiquemos o tipo de dado a ser utilizado de acordo com a tabela ao lado:

```python
import multiprocessing
import ctypes


# Valor compartilhado do tipo int com valor default 0
contador = multiprocessing.Value('i')

# Valor compartilhado do tipo boolean com valor default False
ativo = multiprocessing.Value(ctypes.c_bool, False)


lock = multiprocessing.RLock()


# Valor compartilhado do tipo long, com valor default 0 e com o lock especificado
quantidade = multiprocessing.Value('I', 0, lock=lock)


# Array compartilhado do tipo int com os valores iniciais: 1, 2, 3, 4, 5
lista = multiprocessing.Array('i', [1, 2, 3, 4, 5])
```

ctypes

ctypes type	C type	Python type	Type code
c_bool	_Bool	bool(1)	
c_char	char	1-character bytes object	'c'
c_wchar	wchar_t	1-character string	'u'
c_int	int	int	'i'
c_long	long	int	'l'
c_float	float	float	'f'
c_char_p	char * (NUL terminated)	bytes object or None	
c_wchar_p	wchar_t * (NUL terminated)	string object or None	
c_void_p	void *	int or None	

Compartilhando estados entre processos

Compartilhamento de Memória

multiprocessing.**Value**
multiprocessing.**Array**

Estas classes utilizam de ctypes (código em linguagem C interagindo com Python) e desta forma é necessário que especifiquemos o tipo de dado a ser utilizado de acordo com a tabela ao lado:

```python
import multiprocessing
import ctypes


# Valor compartilhado do tipo int com valor default 0
contador = multiprocessing.Value('i')

# Valor compartilhado do tipo boolean com valor default False
ativo = multiprocessing.Value(ctypes.c_bool, False)


lock = multiprocessing.RLock()


# Valor compartilhado do tipo long, com valor default 0 e com o lock especificado
quantidade = multiprocessing.Value('I', 0, lock=lock)


# Array compartilhado do tipo int com os valores iniciais: 1, 2, 3, 4, 5
lista = multiprocessing.Array('i', [1, 2, 3, 4, 5])
```

ctypes

ctypes type	C type	Python type	Type code
c_bool	_Bool	bool(1)	
c_char	char	1-character bytes object	'c'
c_wchar	wchar_t	1-character string	'u'
c_int	int	int	'i'
c_long	long	int	'l'
c_float	float	float	'f'
c_char_p	char * (NUL terminated)	bytes object or None	
c_wchar_p	wchar_t * (NUL terminated)	string object or None	
c_void_p	void *	int or None	

Vamos ao código ver isso na prática…

Geek University

Evolua seu lado geek!

www.geekuniversity.com.br