Live de Python #182

Roteiro

1. Afinal, o que é Cython?

Explicando conceitos

2. Compilar ou usar os módulos?

Dá pra fazer os dois

3. Pequenos exemplos

Vendo isso em runtime

4. Trabalhando com C

Bibliotecas externas

Roteiro

1. Afinal, o que é Cython?

Explicando conceitos

2. Compilar ou usar os módulos?

Dá pra fazer os dois

3. Pequenos exemplos

Vendo isso em runtime

4. Trabalhando com C

Bibliotecas externas

Se der tempo, ou outra live xD

picpay.me/dunossauro

apoia.se/livedepython

PIX

Ajude o projeto <3

Acássio Anjos, Ademar Peixoto, Alex Lima, Alexandre Harano, Alexandre Santos, Alexandre Tsuno, Alexandre Villares, Alynne Ferreira, Alysson Oliveira, Amaziles Carvalho, Andre Azevedo, André Rocha, Antonio Lins, Arnaldo Turque, Artur Zalewska, Bruno Barcellos, Bruno Batista, Bruno Freitas, Bruno Guizi, Bruno Oliveira, Caio Nascimento, Carlos Chiarelli, Cleber Santos, César Almeida, Davi Ramos, David Kwast, Diego Guimarães, Diego Ubirajara, Dilenon Delfino, Dino Aguilar, Donivaldo Sarzi, Elias Soares, Emerson Rafael, Eric Niens, Eugenio Mazzini, Euripedes Borges, Everton Alves, Fabiano Gomes, Fabio Barros, Fabio Castro, Flavkaze Flavkaze, Flávio Meira, Francisco Alencar, Franklin Silva, Fábio Barros, Gabriel Simonetto, Gabriel Soares, Gabriela Santiago, Geandreson Costa, Guilherme Castro, Guilherme Felitti, Guilherme Gall, Guilherme Ostrock, Gustavo Suto, Henrique Junqueira, Henrique Machado, Ismael Ventura, Israel Fabiano, Israel Gomes, Italo Silva, Jair Andrade, Jairo Rocha, Johnny Tardin, Jonatas Leon, Jonatas Oliveira, Jorge Plautz, Jose Mazolini, José Gomes, José Prado, João Lugão, Juan Gutierrez, Jônatas Silva, Kaio Peixoto, Kaneson Alves, Leonardo Cruz, Leonardo Mello, Leonardo Nascimento, Lidiane Monteiro, Lorena Ribeiro, Lucas Barros, Lucas Mello, Lucas Mendes, Lucas Oliveira, Lucas Polo, Lucas Teixeira, Lucas Valino, Luciano Ratamero, Luciano Silva, Maiquel Leonel, Marcela Campos, Marcelino Pinheiro, Marcos Ferreira, Maria Clara, Marina Passos, Matheus Vian, Murilo Cunha, Natan Cervinski, Nicolas Teodosio, Osvaldo Neto, Patric Lacouth, Patricia Minamizawa, Patrick Brito, Patrick Gomes, Paulo Tadei, Pedro Pereira, Peterson Santos, Rafael Lino, Reinaldo Silva, Renan Moura, Revton Silva, Richard Nixon, Riverfount Riverfount, Robson Maciel, Rodrigo Ferreira, Rodrigo Mende, Rodrigo Vaccari, Rodrigo Vieira, Ronaldo Silva, Rui Jr, Samanta Cicilia, Sandro Mio, Sara Selis, Silvio Xm, Thiago Araujo, Thiago Borges, Thiago Bueno, Thiago Moraes, Tony Dias, Tony Santos, Tyrone Damasceno, Vinícius Bastos, Vlademir Souza, Vítor Gomes, Wellington Abreu, Wendel Rios, Wesley Mendes, Willian Lopes, Willian Rosa, Wilson Duarte, Yuri Fialho, Yury Barros, Érico Andrei

Obrigado você

Afinal, o que é?

Cython

Cython é de comer?

Cython pode ser considerado duas coisas:

1. Uma linguagem de programação que **mistura** Python e o **sistema de tipos do C/C++**
2. Um **compilador** que traduz código Cython ou Python em **código eficiente em C/C++**
    a. Esse código por ser um executável; ou
    b. Um módulo python compilado

Sistema de tipos do C/C++?

Embora Python tenha tipos, python é uma linguagem dinamicamente tipada. O que isso quer dizer?

```
1   minha_variável = 1     # Inteiro
2   minha_variável = 'a'   # String
```

Ah, mas python agora tem dicas de tipos (3.6+)

Isso não muda nada em tempo de execução. Ou seja, isso é só uma dica. Ajuda as IDEs e ferramentas externas, mas não faz checagem em tempo de execução

```
1   minha_variável: int = 1     # Inteiro
2   minha_variável = 'a'        # String
```

Já, C...

C é uma linguagem estaticamente tipada, quando um tipo nasce, ele morre com o mesmo tipo

```c
int minha_variavel = 1;        // 1
minha_variavel = 'eduardo';    // 1634886767
```

pip install cython easycython

Bora começar a conhecer

Começar com um problema simples

Sim, um bem simples. Uma sequência de fibonacci.

```python
def fib(n):
    a = 0
    b = 1

    for i in range(n):
        a, b = a + b, a

    return a
```

Agora vamos medir o tempo de execução

Vamos usar o módulo nativo timeit para fazer isso

```python
from timeit import timeit

py = timeit(
    'fib(93)',
    number=1_000_000,
    setup='from fib_py import fib'
)

print(f'Time {py}')
```

Vamos inspecionar mais a fundo o que acontece

Para conseguir enxergar o problema mais a fundo, vamos profilar o código de verdade usando cProfile e pstats. Com um código do "mundo real".

```python
# run2.py
from random import randint
from fib_py import fib


numeros = [randint(0, 93) for x in range(100_000)]

for x in numeros:
    fib(x)
```

Vamos inspecionar mais a fundo o que acontece

Para conseguir enxergar o problema mais a fundo, vamos profilar o código de verdade usando cProfile e pstats. Com um código do "mundo real".

```
1   python -m cProfile -o prof run2.py
2   python -m pstats prof
```

Agora sim!

Compilando

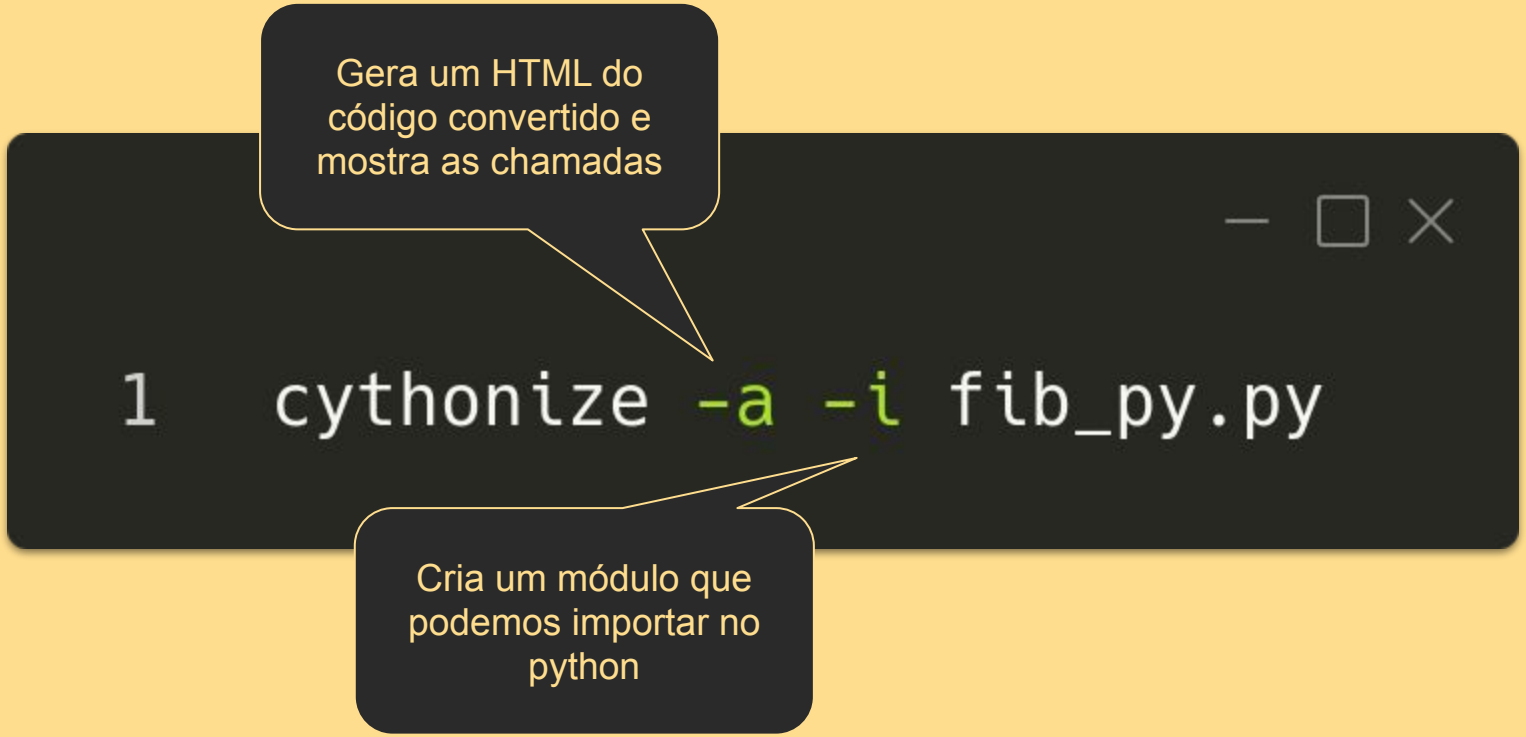
Chegou a hora de compilar

Hora do Cython brilhar!

```
cythonize -a -i fib_py.py
```

Chegou a hora de compilar

Hora do Cython brilhar!



Relatório do cython

Generated by Cython 3.0.0a9

Yellow lines hint at Python interaction.
Click on a line that starts with a "+" to see the C code that Cython generated for it.

Raw output: fib_py.c

```
+1: def fib(n):
+2:     a = 0
+3:     b = 1
 4:
+5:     for i in range(n):
+6:         a, b = a + b, a
 7:
+8:     return a
```

Agora vamos medir o tempo de execução de novo!

```python
from timeit import timeit

py = timeit(
    'fib(93)',
    number=1_000_000,
    setup='from fib_py import fib'
)

print(f'Time {py}')
```


UAU!

Profilar de novo para saber o que rola

```
1   python -m cProfile -o prof run2.py
2   python -m pstats prof
```

Pausa para passar a vassoura aqui (Makefile)

```
build:
  easycython *.pyx
  easycython fib_cy.py

clean:
    @echo "Passando a vassoura no pó"
    rm -rf __pycache__
    rm -f *.so
    rm -f *.c
    rm -rf build
    rm -f *.html
    rm -f prof*
```

Mas e os tipos em C?

Parou por aí?

Calma, vamos entender um pouco sobre números

O C, para otimizar o compilador, usa diversos tipos de inteiros. Para que seja possível alocar menos memória.

Tipo	Tamanho
char	-128 - 127
short	-32,768 - 32,767
int*	-32,768 - 32,767
long	-9223372036854775808 - 9223372036854775807
long long	BEM GRANDE.

Vamos começar um novo arquivo, com tipos

Podemos usar as dicas de tipo para dizer ao Cython qual o melhor tipo para usar na compilação. (Live de python #84)

Mas vamos por partes

```
1    def fib(n: int):
2        a: int = 0
3        b: int = 1
4
5        for i in range(n):
6            a, b = a + b, a
7
8        return a
```

```python
# cython: profile=True
import cython


def fib(n: cython.int):
    a: cython.ulonglongint = 0
    b: cython.ulonglongint = 1

    for i in range(n):
        a, b = a + b, a

    return a
```

```python
# cython: profile=True
import cython


def fib(n: cython.int):
    a: cython.ulonglongint = 0
    b: cython.ulonglongint = 1

    for i in range(n):
        a, b = a + b, a

    return a
```

unsigned long long

```python
# cython: profile=True
import cython


def fib(n: cython.int):
    a: cython.ulonglongint = 0
    b: cython.ulonglongint = 1

    for i in range(n):
        a, b = a + b, a

    return a
```

Para poder usar o cProfile.

NÃO USE EM PROD

Super set

Olhando por outra perspectiva

A linguagem Cython

Além de podermos usar python puro com anotações, o Cython tem a sua linguagem. Que é um misto de python e C/C++

```
1   cdef unsigned long long int fib(int n):
2       cdef int i
3       cdef unsigned long long int a = 0
4       cdef unsigned long long int b = 1
5
6       for i in range(n):
7           a, b = a + b, a
8
9       return a
```

Criando código em C puro

```c
#include "c_fib.h"
// c_fib.c

unsigned long long c_fib(int n) {
    int i;
    unsigned long long a=0.0, b=1.0, tmp;
    for (i=0; i<n; ++i) {
        tmp = a; a = a + b; b = tmp;
    }
    return a;
}
```

Criando código em C puro

```c
#ifndef __CFIB_H__
#define __CFIB_H__
// c_fib.h

unsigned long long cfib(int n);

#endif

```

Fazendo um wrapper em Cython

```
1   cdef extern from 'c_fib.h':
2       unsigned long long c_fib(int n)
3
4
5   def fib(n):
6       return c_fib(n)
```

Compilando o C e o Cython

```
# setup.py
# python setup.py build_ext -if

from distutils.core import setup, Extension
from Cython.Build import cythonize

exts = cythonize(
    [Extension(
        "c_fib_import", sources=["c_fib.c", "c_fib_import.pyx"]
     )
    ]
)

setup(
    ext_modules=exts,
)
```

picpay.me/dunossauro

apoia.se/livedepython

PIX

Ajude o projeto <3