Computer System Lab.

CPython

순천향대학교
컴퓨터시스템연구실

이인규

23.02.20

SCH University

목차

- Python3 dev 설치
- .cpp 파일 생성
- .cpp 파일로 동적 라이브러리 파일 생성
- Python 파일 실행

사전 시도

- Numpy 라이브러리 열어보기
  - 너무 방대한 파일에 해석 실패
  - Python과 c가 섞인 문법에 혼란

- Extending Python with C or C++ 공식 문서 참고
  - Python의 C와 C++ 확장 공식문서 해석 실패

- Windows에서는 python3-dev 개발 툴을 지원하지 않음

개발 환경

- 220.69.209.126 CS LAB 서버에서 진행
- Anaconda 가상환경 22.9.0 버전 사용
- Python 3.9.12 버전 사용
- Gcc 컴파일러 11.3.0 버전 사용

Python3-dev 설치

sudo apt-get install python3-dev 명령어 사용

```
(base) hadoop@seolark:~/notebooks/이인규$ sudo apt-get install python3-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-dev is already the newest version (3.10.6-1~22.04).
```

.cpp 파일 작성

```
// test.cpp
#define EXPORT

extern "C"
{
    EXPORT int add(int a, int b){
        return a + b;
    }
}
```

1. define EXPORT

Marks a declaration, a group of declarations, or another module as exported by the current module.
(in cppreference.com)

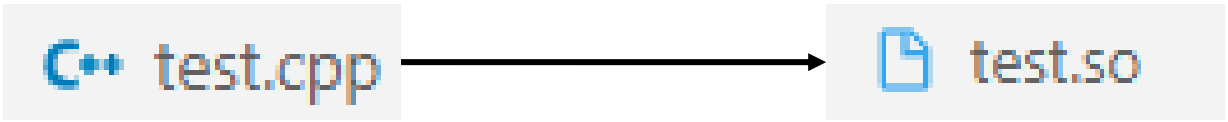
선언, 선언 그룹 또는 다른 모듈을 현재 모듈에서 내보낸 것으로 표시합니다.

2. extern "C"

함수가 다른 곳에서 정의되고 C 언어 호출 규칙을 사용한다는 것을 지정합니다.

.cpp 파일로 동적 라이브러리 파일 생성

```
g++ shared -fPIC -o test.so test.cpp
```



1. shared
- static과 shared 두 가지 옵션 중 선택 가능하다.
- static 옵션은 정적 라이브러리 파일을 의미함
- 동적 라이브러리 파일은 실행파일과 구분하는 것

2. -fPIC
- 재배치 기법을 사용하지 않도록 하는 것
- 재배치 기법은 라이브러리의 주소를 현재 사용하려는 프로세스 주소에서 접근 가능하게 재배치 하는 것
- 프로세스마다 특정 변수(A)의 주소 값을 상대 주소를 사용하여 재배치의 단점을 보완하는 것

3. -o
- 아무런 최적화 작업을 하지 않음

Python 파일 실행

```python
# test.py
import time
import ctypes

start = time.time()
path = "./test.so"
c_module = ctypes.cdll.LoadLibrary(path)

add = c_module.add
add.argtypes = (ctypes.c_int, ctypes.c_int)
add.restype = ctypes.c_int

res = add(1,2)
end = time.time()

print(res)
print(f"Time: {end - start:.15f}")
```

```
3
Time: 0.000043630599976
```

- 잘 실행이 되는 것을 볼 수 있지만, 넘파이 보다 속도 차이가 많이 나는 문제가 발생함.
- 해결 방안으로 ctypes 라이브러리를 통한 파일 로드 과정을 init 함수에 작성

```python
np_a = np.array([[1,2,3],[4,5,6],[7,8,9]])
np_b = np.array([[9,8,7],[6,5,4],[3,2,1]])
```

```
numpy + Time: 0.000003576278687
[[10 10 10]
 [10 10 10]
 [10 10 10]]
```

Python 파일 실행

```
np_a = np.array([[1,2,3],[4,5,6],[7,8,9]])
np_b = np.array([[9,8,7],[6,5,4],[3,2,1]])

numpy + Time: 0.000003576278687
[[10 10 10]
 [10 10 10]
 [10 10 10]]

3
dll add Time: 0.0000545978546l4
dll add Time: 0.000004053115845
```

할 일

- 개념 부족으로 인한 잘 적용이 되고 있는지 검증이 되지 않음
- 컴파일러 옵션에 대한 이해가 모두 되어있지 않음
- init 함수 생성 후 import 시 사전 준비 작업이 되도록 코드 설계
- .cpp를 python에 사용할 수 있는 cpp 코드 작성법 학습
- .cpp 파일을 python에서 사용할 수 있는 python 코드 작성법 학습

Question?

Please contact :

이인규
순천향대학교 컴퓨터학부
멀티미디어관 M606

Email : dldlsrb1414@naver.com