ctypes

C언어의 자료형을 다루고 DLL을 호출할 수 있는 라이브러리. 파이썬 2.5버전부터 기본으로 포함되어 있다.

자료형

C Type	Python Type	ctypes Type
bool	bool(1)	c_bool
char	1-character string	c_char
wchar_t	1-character Unicode string	c_wchar
char	int/long	c_byte
unsigned char	int/long	c_ubyte
short	int/long	c_short
unsigned short	int/long	c_ushort
int	int/long	c_int
size_t	int/long	c_size_t
unsigned int	int/long	c_uint
long	int/long	c_long
unsigned long	int/long	c_ulong
long long	int/long	c_longlong
unsigned long long	int/long	c_ulonglong
float	float	c_float
double	float	c_double
long double	float	c_longdouble
char * (NULL terminated)	string or none	c_char_p
wchar_t * (NULL terminated)	unicode or none	c_wchar_p
void *	int/long or none	c_void_p

예제

```
from ctypes import *

print c_ushort(-5)

seitz = c_char_p("loves the python")
print seitz
print seitz.value
```

결과

```
c_ushort(65531)
c_char_p('loves the python')
loves the python
```

포인터, 참조자

포인터 타입의 내용에 접근하려면 value 속성을 이용한다.

```
seitz = c_char_p("loves the python")
print seitz.value
```

char *, wchar_t *, void *가 아닌 포인터를 선언하려면 POINTER를 이용한다.

다음 코드는 int * 형의 변수 x 를 선언하고 NULL로 초기화한다.

```
x = POINTER(c_int)()
```

POINTER로 선언한 변수의 내용은 contents 속성으로 확인할 수 있다.

C/C++의 함수 중 포인터를 넘겨야 하는 경우 사용하는 참조자는 아래와 같이 사용한다.

```
functionname( byref(parameter) )
```

구조체, 공용체

구조체, 공용체의 선언은 Structure, Union 클래스를 상속받아 한다.

C 구조체

```
struct beer_recipe
{
    int amt_barley;
    int amt_water;
};
```

ctypes 구조체

```
class beer_recipe(Structure):
    _fields_ = [
        ("amt_barley", c_int),
        ("amt_water", c_int),
    ]
```

C 공용체

```
union {
    long barley_long;
    int barley_int;
```

```
    char barley_char[8];
}barley_amount;
```

ctypes 공용체

```
class barley_amount(Union):
    _fields_ = [
        ("barley_long", c_long),
        ("barley_int", c_int),
        ("barley_char", c_char * 8),
    ]
```

구조체의 포인터

```
class test(Structure):
    _fields_ = [
        ("a",  c_ushort),
        ("b",  c_ushort)
    ]

s = POINTER(test)
s = s(test())

# s = POINTER(test)(test()) 로 선언하여도 된다.

s.contents.a = 1
s.contents.b = 2
```

DLL 호출

함수의 타입에 따라 다음 3가지 함수를 사용할 수 있다.

cdll()

cdecl 호출 규약을 따르는 함수를 export하는 라이브러리 로딩

windll()

stdcall 호출 규약을 따르는 함수를 export하는 라이브러리 로딩

oledll()

windll()과 동일하나 HRESULT 에러 코드를 호출하는 COM 함수를 export하는 라이브러리를 로딩한다.

예제

다음은 msvcrt.dll의 printf를 호출하는 예제이다.

```
from ctypes import *
msvcrt = cdll.msvcrt
message_string = "Hello world!\n"
msvcrt.printf("Testing: %s", message_string)
```

참고

- ctypes 공식 홈페이지