Introduction to SWIG

Main Program in C++, Tool in C#

# What is SWIG?

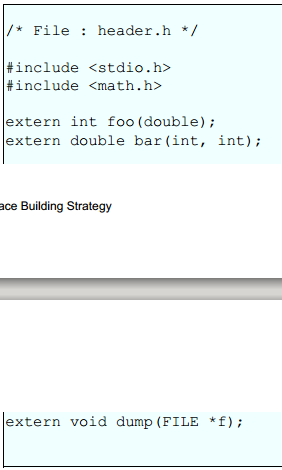
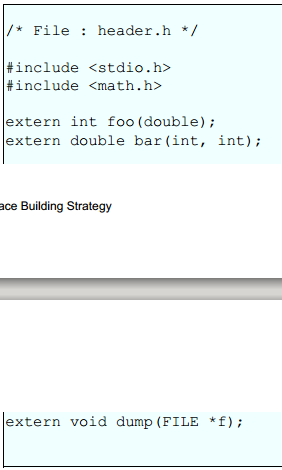
Simply creates the wrappers needed to access C/C++ declarations from other languages

SWIG normally requires “no modifications” to existing code

기존 **코드를 수정하지 않고,** 다른 언어가 C/ C++ 선언들로 **접근**할 수 있도록 **wrapper**를 쉽게 만들어 주는 IDL.

# A SWIG exAmple (C)

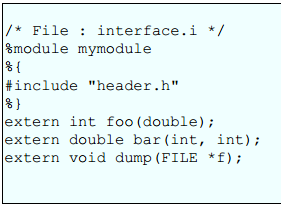
## C input

우리는 여기 있는 function과 변수를 다른 언어에서 사용할 것이다.

그렇다면, SWIG interface file(.i)을 만드는 것부터 시작해보자.

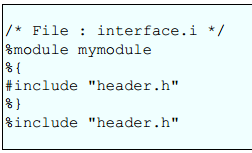
## SWIG interface file



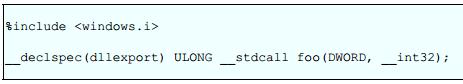
**%module** 지시자:  
-SWIG로 생성될 module의 이름

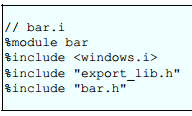
**%{ … %}** 블록:  
-생성될 wrapper code에 들어 갈 header나 declaration을 추가(ANSI C/C++ 규격을 따를 것)

물론 아래와 같이도 가능



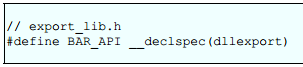
## Microsoft extension

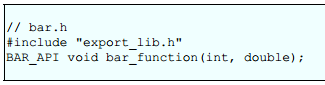


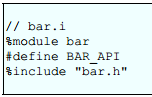


C++ standard에는 윈도우에서 정한 “\_\_declspec(dllimport)”, “\_\_stdcall”등이 존재 하지 않는다.

windows.i에 전처리 문 처리를 구현했기 때문에, 전처리 정의가 포함 되어 있다면, 인터페이스 파일 안에 “%include <windows.i>”를 기재 해야한다.

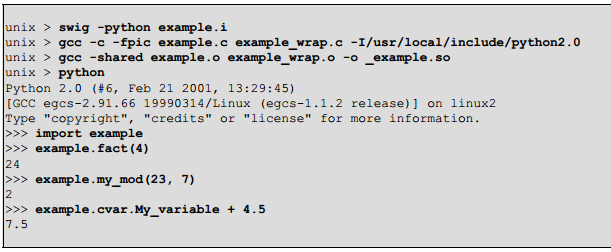




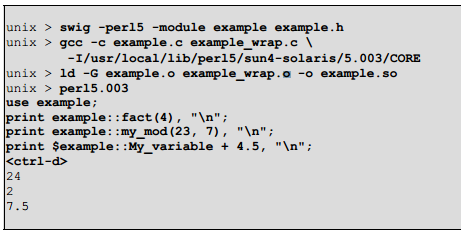


또한, “BAR\_API”와 같은 매크로가 사용되었을 때, 지우고 싶다면, 인터페이스 파일 안에서 전처리기를 사용하는 것도 한 방법이다.  
“#define BAR\_API” in bar.i

## SWIG command



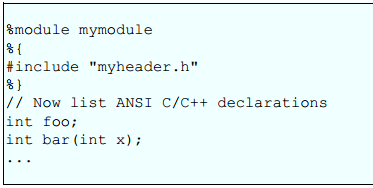
귀찮다면,



# A SWIG EXAMPLE (C++)

## C++ input

## SWIG interface file



## SWIG command





|  |  |
| --- | --- |
| 명령 | 설명 |
| swig | 시작 |
| -c++ | input header: C++ |
| -python | target language: python |
| -outdir | output 파일 저장 경로 설정 |
| pyfiles | 저장 경로 |
| -o | wrapper파일 이름 변경 |
| cppfiles/example\_wrap.cpp | wrapper 경로 및 이름 |
| example.i | interface 이름 |

# SWIG 구동의 주요 키워드

* Unmanaged code
* Scripting languages
* Proxy classes
* Nested structures
* Typemaps
* Preprocessor