**SWIG(Simplified Wrapper and Interface Generator)**

- Là một developer tool. Đơn giản hóa việc tích hợp C/C++ với các ngôn ngữ lập trình khác.

- dùng để kết nối programs viết bằng C, C++ với ngôn ngữ lâp trình bậc cao.

- supported language :

+ Dùng cho các loại ngôn ngữ kịch bản :  Perl, PHP, Python, Tcl and Ruby, …

+ Ngôn ngữ không kịch bàn : C#, Common Lisp (CLISP, Allegro CL, CFFI, UFFI), D, Go language, Java including Android, Lua, Modula-3, OCAML, Octave and R.

- use:

+ Phân tích interface C/C++.

+ Tạo ‘glue code’ để ngôn ngữ mục tiêu call vào C/C++’.

+ Xuất cây phân tích cú pháp định dạng XML, Lisp s-expressions.

- Là phần mềm miễn phí tương thích với các dự án thương mại và phi thương mại.

**Phiên bản mới nhất:**

2014/03/16 - [SWIG-3.0.0 released](http://sourceforge.net/p/swig/news/2014/03/swig-300-released/) [Documentation](http://www.swig.org/Doc3.0/index.html)

**Phiên bản đang sử dụng:**

SWIG-2.0 [Documentation](http://www.swig.org/Doc2.0/index.html)

**SWIG ANDROID EXAMPLE**

**1. C example**

|  |  |
| --- | --- |
| **content** | **a. Tao project**  **b. tao jni va gen file** :  **c. Tao file Android.mk va thuc hien tao library**  **d. Su dung trong code** |

**a. Tao project**

prorect name : test

project package : com.example.test

**b. tao jni va gen file** :

jni :  create  example.c, example.i,Android.mk

|  |  |
| --- | --- |
| example.c | /\* File : example.c \*/  /\* A global variable \*/  double Foo = 3.0;  /\* Compute the greatest common divisor of positive integers \*/  int gcd(int x, int y) {     int g;     g = y;     while (x > 0) {     g = x;     x = y % x;     y = g;     }  return g;  } |
| example.i | %module example;    %inline %{     extern int gcd(int x, int y);     extern double Foo; // tao lenh gen  %} |

|  |
| --- |
|  |

Chay lenh :

|  |
| --- |
| swig -java -package com.example.test -outdir src/com/example/test -o jni/example\_wrap.c jni/example.i |
| Ket qua :         + jni/example\_wrap.c         + java file (com.example.test):               example.java               exampleJNI.java |

**c. Tao file Android.mk va thuc hien tao library**

|  |
| --- |
| # File: Android.mk  LOCAL\_PATH := $(call my-dir)  include $(CLEAR\_VARS)  LOCAL\_MODULE:= example  LOCAL\_SRC\_FILES := example\_wrap.c example.c  include $(BUILD\_SHARED\_LIBRARY) |

Chay lenh ndk-build gen ra cac file

|  |
| --- |
| libs/armeabi/libexample.so |

**d. Su dung trong code**

Su dung lib trong source code com.example.test   MainActivity.java

|  |
| --- |
| static {     System.loadLibrary("example");  }    public void onClick(View v) {     String data = "";     try{     example.setFoo(10.0);     data = example.getFoo() + " : ";     data = data + example.gcd(10, 10);     }catch(Exception ex){     data = ex.getMessage();     }       Toast.makeText(this, data, Toast.LENGTH\_SHORT).show();  } |

**2. C++ example**

**a.Tao file shape.h va shape.ccp**

|  |  |
| --- | --- |
| shape.h | class Shape {  public:  Shape() {  nshapes++;  }  virtual ~Shape() {  nshapes--;  };  double x, y;  void  move(double dx, double dy);  virtual double area(void) = 0;  virtual double perimeter(void) = 0;  static int nshapes;  };  class Circle : public Shape {  private:  double radius;  public:  Circle(double r) : radius(r) { };  virtual double area(void);  virtual double perimeter(void);  };  class Square : public Shape {  private:  double width;  public:  Square(double w) : width(w) { };  virtual double area(void);  virtual double perimeter(void);  }; |
| shape.ccp | /\* File : example.cpp \*/  #include "shape.h"  #define M\_PI 3.14159265358979323846  /\* Move the shape to a new location \*/  void Shape::move(double dx, double dy) {  x += dx;  y += dy;  }  int Shape::nshapes = 0;  double Circle::area(void) {  return M\_PI\*radius\*radius;  }  double Circle::perimeter(void) {  return 2\*M\_PI\*radius;  }  double Square::area(void) {  return width\*width;  }  double Square::perimeter(void) {  return 4\*width;  } |

**b. run lenh**

swig -c++ -java -package com.example.test -outdir src/com/example/test -o jni/shape\_wrap.cpp jni/shape.i

tao ra cac file sau

|  |  |
| --- | --- |
| jni | shape\_wrap.cpp |
| com.example.test | Circle.java  shape.java  Shape.java  shapeJNI.java  Square.java |

c. file Android.mk va chay lenh

|  |  |
| --- | --- |
| Android.mk | # File: Android.mk  LOCAL\_PATH := $(call my-dir)  include $(CLEAR\_VARS)  LOCAL\_MODULE:= example  LOCAL\_SRC\_FILES := example\_wrap.c example.c shape\_wrap.cpp shape.cpp  #for c++  LOCAL\_CFLAGS:= -frtti    include $(BUILD\_SHARED\_LIBRARY) |
| run lenh | ndk-build |
| ket qua | libs/armeabi/libexample.so  obj/local/armeabi/objs/example/                                                    example\_wrap.o                                                    example\_wrap.o.d                                                    example.o                                                    example.o.d  obj/local/armeabi/libexample.so |

**c.Chay thu**

|  |
| --- |
| static {     System.loadLibrary("example");  }    public void onClick(View v) {     String data = "";       try{     double area = shapeJNI.Circle\_area(10l, new Circle(10.0d));     data +="area : " + area ;  }catch(Exception ex){  }  } |

**3.C++ STL**

Should the C++ Standard Template Library (STL) be required, an Application.mk file needs to be created in the same

directory as the Android.mk directory containing information about the STL to use. See the NDK documentation in the

$NDKROOT/docs folder especially CPLUSPLUS-SUPPORT.html. Below is an example of the Application.mk file to make

the STLport static library available for use:

# File: Application.mk

APP\_STL := gnustl\_static

**\*Chu y**

**file warper voi ham   
example\_warp.cpp → exampleJNI.java**

#ifdef \_\_cplusplus

extern "C" {

SWIGEXPORT

example.i → ra file example

Neu la C, module trung voi file.c