

Re: What is the best way to define the Imported C function

Source: <http://coding.derkeiler.com/Archive/Ada/comp.lang.ada/2008-01/msg00453.html>

- *From:* "Dmitry A. Kazakov" <mailbox@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 26 Jan 2008 11:56:09 +0100
-

On Fri, 25 Jan 2008 21:05:02 -0800 (PST), qunying wrote:

I am learning Ada and try to test the interface with C.

for this function, what is the best way to define the function in Ada?

```
int xcb_parse_display(const char *name, char **host, int *display, int
*screen);
```

```
function parse_display (Name : String; Host: ??; Display :
Integer_Ptr; Screen : Integer_Ptr) return Integer;
pragma Import (C, parse_display, "xcb_parse_display");
```

Where Integer_Ptr is access Integer;

Name : Interfaces.C.char_array

and use To_C in order to pass a String there. Ada's String is not C's char[]

Display : access Interfaces.C.int

Firstly Integer is not necessarily int, the built-in package Interfaces.C provides integer types compatible with C. Secondly in out mode will do the trick where int * is used as an in-out parameter rather than a pointer to an array. Unfortunately you have a function, so in out is illegal. Therefore access Interfaces.C.int instead.

How to define the type for char **? Sould I use "type char_ptr_ptr is access char_ptr;" where char_ptr is defined in Interfaces.C; or there is a better way to do it?

Like with int *, that depends on the semantics. If the idea is to return a pointer to a string then just tell so:

Re: What is the best way to define the Imported C function

Host : access Interfaces.C.Strings.chars_ptr
Screen : access Interfaces.C.int

Surely, you would like to make a wrapper around the mess to make it more Ada-friendly:

```
type Display_ID is private; -- An opaque handle to
type Screen_ID is private;
-- This either does the job or raises an exception
procedure Parse_Display
( Name : String;
  Host : out Unbounded_String;
  Display : out Display_ID;
  Screen : out Screen_ID
);
...
private
type Display_ID is new Interfaces.C.int;
type Screen_ID is new Interfaces.C.int;
...
procedure Parse_Display
( Name : String;
  Host : out Unbounded_String;
  Display : out Display_ID;
  Screen : out Screen_ID
) is
function Internal
( Name : Interfaces.C.char_array;
  Host : access Interfaces.C.Strings.chars_ptr;
  Display : access Display_ID;
  Screen : access Screen_ID
) return Interfaces.C.int;
pragma Import (C, Internal, "xcb_parse_display");

Display_Value : aliased Display_ID;
Screen_Value : aliased Screen_ID;
Host_Value : aliased Interfaces.C.Strings.chars_ptr;
Result : Interfaces.C.int :=
Internal
( Name => To_C (Name),
  Host => Host_Value'Access,
  Display => Display_Value'Access,
  Screen => Screen_Value'Access
);
begin
if Result = 0 then
Host := To_Unbounded_String (To_Ada (Host_Value));
Display := Display_Value;
Screen := Screen_Value;
```

Re: What is the best way to define the Imported C function

Re: What is the best way to define the Imported C function

```
else  
raise <Something Appropriate>;  
end if;  
end Parse_Display;
```

—
Regards,
Dmitry A. Kazakov
<http://www.dmitry-kazakov.de>