

Re: STRING length

Source: <http://coding.derkeiler.com/Archive/Ada/comp.lang.ada/2006-11/msg00166.html>

- *From:* "markww" <markww@xxxxxxxxxx>
 - *Date:* 14 Nov 2006 17:09:06 -0800
-

Actually please disregard the previous post, I found I needed to with and use:

```
Ada.Text_IO.Unbounded_IO
```

So.. finally one compilation error remains, the actual call to my function:

```
begin
```

```
Add_Record("Mark", "555-555-5555", "123 main street");
```

```
end LinkList;
```

the error is:

```
expected private type "Ada.Strings.Unbounded.Unbounded_String"
```

what does that mean? The problem is with calling the procedure, if I comment the call out compilation is successful. The procedure looks like:

```
procedure Add_Record(strName : in UNBOUNDED_STRING;  
strPhone : in UNBOUNDED_STRING;  
strAddress : in UNBOUNDED_STRING)  
is  
Temp : CHAR_REC_POINT;  
begin  
Put(strName);  
New_Line;  
Put(strPhone);  
New_Line;  
Put(strAddress);  
New_Line;  
end Add_Record;
```

I guess the compiler doesn't interpret a literal string as an `unbounded_string`?

Re: STRING length

Thanks,
Mark

On Nov 14, 7:44 pm, "markww" <mar...@xxxxxxxxxx> wrote:

Thanks Georg, that looks to be exactly what I need. I do have a problem '#including', or, 'withing' rather unbounded string.
Now the head of my source file looks like:

```
with Ada.Text_IO, Ada.Integer_Text_IO, Ada.Strings.Unbounded;  
use Ada.Text_IO, Ada.Integer_Text_IO;
```

which is alright but as soon as I try:

```
use Ada.Strings.Unbounded;
```

the compiler gives me a bunch of errors, it seems to conflict with Text_IO / Integer_Text_IO? Seems like all my previous calls to Put() now became invalid. I'm not familiar with Ada but with C++ and understand namespace collisions, is the same thing going on here?

Thanks

On Nov 14, 5:24 pm, Georg Bauhaus <bauh...@xxxxxxxxxxxxxxxx> wrote:

On Tue, 2006-11-14 at 14:51 -0800, markww wrote:

Hi,

How does one use a variable length string in ada? You use variable length strings in Ada by declaring them

to be of type UNBOUNDED_STRING which is defined in Ada.Strings.Unbounded.

```
type MY_RECORD is  
record  
Name: UNBOUNDED_STRING;  
Phone: UNBOUNDED_STRING;  
Address: UNBOUNDED_STRING;  
end record;
```

Re: STRING length

Given that Phone is likely to be limited in length, you could consider declaring the Phone component to be of type BOUNDED_STRING, which is a string type with a maximum length. Unlike STRING, objects of this type can have any number of characters up to the maximum. See Ada.Strings.Bounded.

Yet another use of strings is in nested scopes: If you need a string in just one place, e.g. temporarily, you can use a plain STRING as in

```
declare
temp: constant STRING := some_string_returning_func(...);
begin
-- use temp
end;
```

The point here is that the `temp` string variable takes its bound from the initialization. You can also make it a variable, if you need to write to string components.

See http://en.wikibooks.org/wiki/Ada_Programming/Strings

-- Georg-- Hide quoted text -- Show quoted text --