

## Re: Unchecked\_Conversion and task pointer.

---

*Source:* <http://coding.derkeiler.com/Archive/Ada/comp.lang.ada/2005-07/msg00193.html>

---

- *From:* "Dmitry A. Kazakov" <[mailbox@xxxxxxxxxxxxxxxxxxxxx](mailto:mailbox@xxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Sun, 10 Jul 2005 10:26:09 +0200
- 

On Sat, 9 Jul 2005 20:42:33 -0500, Randy Brukardt wrote:

> "Dmitry A. Kazakov" <[mailbox@xxxxxxxxxxxxxxxxxxxxx](mailto:mailbox@xxxxxxxxxxxxxxxxxxxxx)> wrote in message  
> [news:28m5yx3jhzvu\\$.16u6va8093srl.dlg@xxxxxxxxxxxxxxxxx](mailto:news:28m5yx3jhzvu$.16u6va8093srl.dlg@xxxxxxxxxxxxxxxxx)  
> ...  
>>> No, you should definitely use an interface.  
>>  
>> It depends. Tasks are still non-tagged in the sense that you cannot have a  
>> "task" interface.  
>  
> You can declare a task interface in Ada 200Y:  
>  
> type T\_Int is task interface;  
>  
> I probably should have done that in my example.

Can a task interface have entries?

>> In an interface Who\_Am\_I can only be a *\*procedure\**, which  
>> could be then implemented by an entry. But it should an *\*entry\** from the  
>> beginning if the intent is to have a dispatching entry Who\_Am\_I called  
>> using, say, timed entry call.  
>  
> You are allowed to use the primitive procedures of an interface in a timed  
> entry call, and they will work as an entry in that case.

Does it mean that in Ada 200Y any procedure of an interface can be used as  
if it were an entry?

> So while the  
> *\*syntax\** is that of a procedure, they do not lose their entry  
> characteristics and still act as an entry.  
>  
> I wasn't (and am still not) convinced that this is the right design, but it  
> certainly works and has the needed effects. I think there will be some  
> confusion from a readability standpoint, but otherwise you can do everything  
> with such a procedure that you can do with an entry (other than requeue it;  
> no one can figure out how a dispatching requeue could work - it couldn't  
> have been allowed if dispatching entries existed, either).

Re: Unchecked\_Conversion and task pointer.

Why? Ignoring misleading prefix notation can be ignored, to have a dispatching requeue, the target task or protected object of the entry must be class-wide:

```
task type A is -- This is not Ada
entry Foo;
end A;
function Factory (...) return A'Class;

Object : X'Class := Factory (...);
....
requeue Object.Foo; -- Dispatches to Foo
....
```

BTW, similarly to class-wide subroutines there could be class-wide entries. A requeue from a class-wide entry to a "primitive" entry of the same object would be dispatching:

```
protected type A is -- This is not Ada
entry Foo'Class; -- A class-wide entry
entry Bar; -- A "primitive" entry
end A;
```

>> In that case mix-in will be the only workaround.  
>  
> Not at all. See above.

It seems so.

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

- 
- *Follow-Ups:*
    - ◆ **Re: Unchecked\_Conversion and task pointer.**  
◇ From: Randy Brukardt
  - *References:*
    - ◆ **Unchecked\_Conversion and task pointer.**  
◇ From: e.coli
    - ◆ **Re: Unchecked\_Conversion and task pointer.**  
◇ From: Dmitry A. Kazakov
    - ◆ **Re: Unchecked\_Conversion and task pointer.**  
◇ From: Randy Brukardt
    - ◆ **Re: Unchecked\_Conversion and task pointer.**  
◇ From: Dmitry A. Kazakov

Re: Unchecked\_Conversion and task pointer.

Re: Unchecked\_Conversion and task pointer.

◆ **Re: Unchecked\_Conversion and task pointer.**

◇ From: Randy Brukardt

- Prev by Date: **Re: GCC 4.0 Ada.Containers Cursor danger.**
- Next by Date: **Re: GCC 4.0 Ada.Containers Cursor danger.**
- Previous by thread: **Re: Unchecked\_Conversion and task pointer.**
- Next by thread: **Re: Unchecked\_Conversion and task pointer.**
- Index(es):
  - ◆ **Date**
  - ◆ **Thread**