Subject: Re: V8.2 compilation error, works in V8.1
Posted by Michael Galloy on Fri, 01 Jun 2012 16:43:03 GMT
View Forum Message <> Reply to Message

On 6/1/12 9:51 AM, wlandsman wrote:
> We are getting many compilation errors in V8.2 for code that worked
> in V8.1. Apparently, there was a loophole in V8.1 that allowed one to
> directly access an object's variables in certain case, but now a line like
>
> self.obj2.index
>
> gives an error % Expression must be a structure in this
> context:<OBJREF
(<ObjHeapVar2(OBJ2)>)>
>
> One can always access the variable with a GETPROPERTY, and I suppose
> this might have been a "bug fix". Is there any place where one can get a
> list of the V8.2 bug fixes?

Yes, I confirm your code seems to indicate that the loophole allowing
member variable access of an object from another object's method has
been eliminated. I would count that as a bug fix, but I wish it had been
mentioned in the release notes. I think there is going to be some broken
code that will need to be updated for IDL 8.2.

Also, note that by making obj2 inherit from IDL_Object will make the
code work again on IDL 8.2, i.e., update obj2__define to:

```
pro obj2__define
  compile_opt idl2
  obj2 = { obj2, inherits IDL_Object, index:0L }
end
```

Mike
--
Michael Galloy
www.michaelgalloy.com
Modern IDL: A Guide to IDL Programming (http://modernidl.idldev.com)
Research Mathematician
Tech-X Corporation

Subject: Re: V8.2 compilation error, works in V8.1
Posted by Lajos Foldy on Fri, 01 Jun 2012 16:57:40 GMT
View Forum Message <> Reply to Message

On Friday, June 1, 2012 5:51:31 PM UTC+2, wlandsman wrote:
> We are getting many compilation errors in V8.2 for code that worked in V8.1. Apparently, there

was a loophole in V8.1 that allowed one to directly access an object's variables in certain case, but now a line like
>
> self.obj2.index
>
> gives an error
> % Expression must be a structure in this context: <OBJREF (<ObjHeapVar2(OBJ2)>)>
>
> One can always access the variable with a GETPROPERTY, and I suppose this might have been a "bug fix". Is there any place where one can get a list of the V8.2 bug fixes?
>
> Below I give an example with two simple objects.
>
> IDL> t = obj_new('obj1')
> IDL> t.printvalue
>
> gives an error in V8.2 but not V8.1
>
> ;obj1__define.pro
> pro obj1::printvalue,index=index
> print,self.obj2.index
> end
> pro obj1::getproperty,obj2=obj2
> if arg_present(obj2) then obj2 = self.obj2
> end
> function obj1::init,obj2=obj2
> if ~isa(obj2,'obj2') then self.obj2 = obj_new('obj2')
> return,1
> end
> pro obj1__define
> compile_opt idl2
> obj1 = {obj1, $
> obj2:obj_new()}
> end
>
> ;obj2__define.pro
> pro obj2::getproperty,index=index
> if arg_present(index) then index = self.index
> end
> pro obj2__define
> compile_opt idl2
> obj2 = {obj2, $
> index:0L}
> end

Add 'inherits IDL_Object' to your definitions.

("If your object inherits from the IDL_Object class, you can retrieve property values by calling the

property directly with the dot operator.")

regards,
Lajos

Subject: Re: V8.2 compilation error, works in V8.1
Posted by Lajos Foldy on Fri, 01 Jun 2012 17:23:54 GMT
View Forum Message <> Reply to Message

On Friday, June 1, 2012 5:51:31 PM UTC+2, wlandsman wrote:
> We are getting many compilation errors in V8.2 for code that worked in V8.1. Apparently, there was a loophole in V8.1 that allowed one to directly access an object's variables in certain case, but now a line like
>
> self.obj2.index
>
> gives an error
> % Expression must be a structure in this context: <OBJREF (<ObjHeapVar2(OBJ2)>)>
>
> One can always access the variable with a GETPROPERTY, and I suppose this might have been a "bug fix". Is there any place where one can get a list of the V8.2 bug fixes?
>
> Below I give an example with two simple objects.
>
> IDL> t = obj_new('obj1')
> IDL> t.printvalue
>
> gives an error in V8.2 but not V8.1
>
> ;obj1__define.pro
> pro obj1::printvalue,index=index
> print,self.obj2.index
> end
> pro obj1::getproperty,obj2=obj2
> if arg_present(obj2) then obj2 = self.obj2
> end
> function obj1::init,obj2=obj2
> if ~isa(obj2,'obj2') then self.obj2 = obj_new('obj2')
> return,1
> end
> pro obj1__define
> compile_opt idl2
> obj1 = {obj1, $
> obj2:obj_new()}
> end
>
> ;obj2__define.pro

```
>  pro obj2::getproperty,index=index
>  if arg_present(index) then index = self.index
>  end
>  pro obj2__define
>  compile_opt idl2
>   obj2 = {obj2, $
>           index:0L}
>  end
```

After replacing 'print,self.obj2.index' with 'print,(self.obj2).index', it works. So the loophole is still there :-)

regards,
Lajos

Subject: Re: V8.2 compilation error, works in V8.1
Posted by wlandsman on Fri, 01 Jun 2012 17:57:08 GMT
View Forum Message <> Reply to Message

Lajos, Mike

Thanks for the IDL_Object suggestion -- that seems like the simplest fix.

Here's another IDL 8.2 change that is giving us compilation errors:

```
IDL> print,!version
{ x86_64 linux unix linux 8.1 Mar  9 2011      64      64}
IDL> x = HASH('key1',1,'key2',2)
IDL> y = ['key1']
IDL> help,x[y]
<Expression>    LONG      =           1

IDL> print,!version
{ x86_64 linux unix linux 8.2 Apr 10 2012      64      64}
IDL> x = HASH('key1',1,'key2',2)
IDL> y = ['key1']
IDL> help,x[y]
<Expression>    HASH  <ID=8  NELEMENTS=1>
```

In this case, I think the V8.1 interpretation makes more sense. --Wayne

---

Subject: Re: V8.2 compilation error, works in V8.1
Posted by Lajos Foldy on Fri, 01 Jun 2012 18:18:56 GMT
View Forum Message <> Reply to Message

On Friday, June 1, 2012 7:57:08 PM UTC+2, wlandsman wrote:
> Lajos, Mike
>
> Thanks for the IDL_Object suggestion -- that seems like the simplest fix.
>
> Here's another IDL 8.2 change that is giving us compilation errors:
>
> IDL> print,!version
> { x86_64 linux unix linux 8.1 Mar 9 2011 64 64}
> IDL> x = HASH('key1',1,'key2',2)
> IDL> y = ['key1']
> IDL> help,x[y]
> <Expression> LONG = 1
>
> IDL> print,!version
> { x86_64 linux unix linux 8.2 Apr 10 2012 64 64}
> IDL> x = HASH('key1',1,'key2',2)
> IDL> y = ['key1']
> IDL> help,x[y]
> <Expression> HASH <ID=8 NELEMENTS=1>
>
> In this case, I think the V8.1 interpretation makes more sense. --Wayne

The IDL 8.2 behaviour is consistent with the array subscripting version:

IDL> x=[1,2,3] & y=[1] & help, x[y]
<Expression> INT = Array[1]

I think this was the motivation for the change.

regards,
Lajos

Subject: Re: V8.2 compilation error, works in V8.1
Posted by b_gom on Fri, 01 Jun 2012 18:54:32 GMT
View Forum Message <> Reply to Message

I ran into the same problem with some older object code that
mysteriously stopped working. Instead of spending a few hours tracking
the problem down to the missing INHERITS IDL_OBJECT line, I should
have just waited for the group to solve it for me!

Subject: Re: V8.2 compilation error, works in V8.1
Posted by wlandsman on Thu, 07 Jun 2012 17:19:19 GMT
View Forum Message <> Reply to Message

For completeness, I note that the differences between V8.2 and V8.1 also occur (not unexpectedly) with lists.

```
IDL> b = list('header',0)
IDL> n = [1]
IDL> t = b.get(position=n)
IDL> help,t
In V8.1
T               LONG      =           0
But in V8.2
T               LIST  <ID=6  NELEMENTS=1>
```

Lajos pointed out that this change made the behavior consistent with subscripting an array with a one element array vs. with a scalar. But whereas it is often doesn't matter whether one has a scalar or a one element array, it will always make a difference when code that used to return a scalar integer now returns a list data type.

On Friday, June 1, 2012 1:57:08 PM UTC-4, wlandsman wrote:

```
>  Lajos, Mike
>
>      Thanks for the IDL_Object suggestion -- that seems like the simplest fix.
>
>      Here's another IDL 8.2 change that is giving us compilation errors:
>
>  IDL> print,!version
>  { x86_64 linux unix linux 8.1 Mar  9 2011      64      64}
>  IDL> x = HASH('key1',1,'key2',2)
>  IDL> y = ['key1']
>  IDL> help,x[y]
>  <Expression>    LONG      =           1
>
>  IDL> print,!version
>  { x86_64 linux unix linux 8.2 Apr 10 2012      64      64}
>  IDL> x = HASH('key1',1,'key2',2)
>  IDL> y = ['key1']
>  IDL> help,x[y]
>  <Expression>    HASH  <ID=8  NELEMENTS=1>
>
>  In this case, I think the V8.1 interpretation makes more sense.  --Wayne
```