
Subject: Referencing structure inline

Posted by [TimB](#) on Tue, 09 Dec 2014 04:57:36 GMT

[View Forum Message](#) <> [Reply to Message](#)

Perhaps a fundamental misunderstanding on my part but I'm wondering if there is a way of making this work:

```
IDL> z = {x:1,y:2}
```

```
IDL> z.x
```

```
1
```

```
IDL> ({x:1,y:2}).x
```

% Object reference type required in this context: <STRUCT Array[1]>.

I have a function which returns a structure and I would like to use it in this manner:

```
z = (myfunc(input)).x
```

However I get the same error as the example above. The following works

```
struct = myfunc(input)
```

```
z = struct.x
```

Can it be done in one line?

Subject: Re: Referencing structure inline

Posted by [Heinz Stege](#) on Tue, 09 Dec 2014 08:42:50 GMT

[View Forum Message](#) <> [Reply to Message](#)

Seems to be a bug in your IDL version. The following works in IDL 8.0:

```
IDL> help,file_info(!dir)
```

```
** Structure FILE_INFO, 21 tags, length=64, data length=63:
```

```
NAME      STRING  'C:\idl\idl80'
```

```
EXISTS     BYTE    1
```

```
READ       BYTE    1
```

```
WRITE      BYTE    1
```

```
EXECUTE     BYTE    1
```

```
REGULAR     BYTE    0
```

```
DIRECTORY   BYTE    1
```

```
BLOCK_SPECIAL BYTE    0
```

```
CHARACTER_SPECIAL
```

```
      BYTE    0
```

```
NAMED_PIPE  BYTE    0
```

```
SETUID       BYTE    0
```

```
SETGID       BYTE    0
```

```
SOCKET       BYTE    0
```

```
STICKY_BIT   BYTE    0
```

```

SYMLINK      BYTE      0
DANGLING_SYMLINK
      BYTE      0
MODE          LONG      511
ATIME         LONG64     1418112222
CTIME         LONG64     1299161262
MTIME         LONG64     1349551389
SIZE          LONG64      0
IDL> print,(file_info(!dir)).name
C:\idl\idl80
IDL> print,(file_info(!dir)).ctime
1299161262
IDL> print,!version
{ x86 Win32 Windows Microsoft Windows 8.0.1 Oct  5 2010    32
64}

```

Subject: Re: Referencing structure inline
 Posted by [Helder Marchetto](#) on Tue, 09 Dec 2014 10:55:01 GMT
[View Forum Message](#) <> [Reply to Message](#)

On Tuesday, December 9, 2014 9:42:57 AM UTC+1, Heinz Stege wrote:

> Seems to be a bug in your IDL version. The following works in IDL 8.0:

>

> IDL> help,file_info(!dir)

> ** Structure FILE_INFO, 21 tags, length=64, data length=63:

```

> NAME          STRING   'C:\idl\idl80'
> EXISTS        BYTE     1
> READ          BYTE     1
> WRITE         BYTE     1
> EXECUTE       BYTE     1
> REGULAR       BYTE     0
> DIRECTORY     BYTE     1
> BLOCK_SPECIAL BYTE     0
> CHARACTER_SPECIAL
>          BYTE     0
> NAMED_PIPE    BYTE     0
> SETUID        BYTE     0
> SETGID        BYTE     0
> SOCKET        BYTE     0
> STICKY_BIT    BYTE     0
> SYMLINK       BYTE     0
> DANGLING_SYMLINK
>          BYTE     0
> MODE          LONG      511
> ATIME         LONG64     1418112222
> CTIME         LONG64     1299161262
> MTIME         LONG64     1349551389

```

```

> SIZE LONG64 0
> IDL> print,(file_info(!dir)).name
> C:\idl\idl80
> IDL> print,(file_info(!dir)).ctime
> 1299161262
> IDL> print,!version
> { x86 Win32 Windows Microsoft Windows 8.0.1 Oct 5 2010 32
> 64}

```

Hi,

If this is a bug then it's still out there. Implied print seems to be the problem:

```

IDL> ({x:1,y:2}).x
% Object reference type required in this context: <STRUCT Array[1]>.
% Execution halted at: $MAIN$
IDL> print, ({x:1,y:2}).x
1
IDL> !version
{
  "ARCH": "x86_64",
  "OS": "Win32",
  "OS_FAMILY": "Windows",
  "OS_NAME": "MicrosoftWindows",
  "RELEASE": "8.4",
  "BUILD_DATE": "Sep272014",
  "MEMORY_BITS": 64,
  "FILE_OFFSET_BITS": 64
}

```

In other words, such indexing does not work with implied print.

I also tried this with a function:

```

function testPrintFunction
return, {x:1,y:2}
end

```

```

IDL> (testPrintFunction()).x
% Object reference type required in this context: <STRUCT Array[1]>.
% Execution halted at: $MAIN$
IDL> print, (testPrintFunction()).x
1

```

Cheers,
Helder

Subject: Re: Referencing structure inline

Posted by [Helder Marchetto](#) on Tue, 09 Dec 2014 10:56:55 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Tuesday, December 9, 2014 11:55:02 AM UTC+1, Helder wrote:

> On Tuesday, December 9, 2014 9:42:57 AM UTC+1, Heinz Stege wrote:

>> Seems to be a bug in your IDL version. The following works in IDL 8.0:

>>

>> IDL> help,file_info(!dir)

>> ** Structure FILE_INFO, 21 tags, length=64, data length=63:

>> NAME STRING 'C:\idl\idl80'

>> EXISTS BYTE 1

>> READ BYTE 1

>> WRITE BYTE 1

>> EXECUTE BYTE 1

>> REGULAR BYTE 0

>> DIRECTORY BYTE 1

>> BLOCK_SPECIAL BYTE 0

>> CHARACTER_SPECIAL

>> BYTE 0

>> NAMED_PIPE BYTE 0

>> SETUID BYTE 0

>> SETGID BYTE 0

>> SOCKET BYTE 0

>> STICKY_BIT BYTE 0

>> SYMLINK BYTE 0

>> DANGLING_SYMLINK

>> BYTE 0

>> MODE LONG 511

>> ATIME LONG64 1418112222

>> CTIME LONG64 1299161262

>> MTIME LONG64 1349551389

>> SIZE LONG64 0

>> IDL> print,(file_info(!dir)).name

>> C:\idl\idl80

>> IDL> print,(file_info(!dir)).ctime

>> 1299161262

>> IDL> print,!version

>> { x86 Win32 Windows Microsoft Windows 8.0.1 Oct 5 2010 32

>> 64}

>

>

> Hi,

> If this is a bug then it's still out there. Implied print seems to be the problem:

>

> IDL> ({x:1,y:2}).x

> % Object reference type required in this context: <STRUCT Array[1]>.

> % Execution halted at: \$MAIN\$

> IDL> print, ({x:1,y:2}).x

```
> 1
> IDL> !version
> {
>   "ARCH": "x86_64",
>   "OS": "Win32",
>   "OS_FAMILY": "Windows",
>   "OS_NAME": "MicrosoftWindows",
>   "RELEASE": "8.4",
>   "BUILD_DATE": "Sep272014",
>   "MEMORY_BITS": 64,
>   "FILE_OFFSET_BITS": 64
> }
>
> In other words, such indexing does not work with implied print.
>
> I also tried this with a function:
>
> function testPrintFunction
> return, {x:1,y:2}
> end
>
> IDL> (testPrintFunction()).x
> % Object reference type required in this context: <STRUCT   Array[1]>.
> % Execution halted at: $MAIN$
> IDL> print, (testPrintFunction()).x
> 1
>
> Cheers,
> Helder
```

Here is a link with some explanation. I'm sure it falls under the same problem pointed out by Chris T.

https://groups.google.com/d/msg/comp.lang.idl-pvwave/rbleuyI_6GfA/vli09AwC0rEJ

Cheers,
Helder
