Literals as sub parameters become...

Angus Monro

Literals as sub parameters become variables Jan 3, 2012 11:53 PM

Hi Folks,

I'm finding that when I use a literal value (like 31, for example) as the parameter to a user-defined subroutine, that literal becomes listed in the Variable Overview as a variable that has the literal as both the variable name an value.

Hence, in the attached sample, you'll find I've defined a trivial subroutine TraceValue and invoked it with the parameter value 31. On reloading (in QV Developer 10 SR3), Settings > Variable Overview show 31 as both Variable Name and Value.

Does anyone else get this behaviour? Is there something I'm not understanding about correct use of QlikView subroutines? I'd have thought calling a sub with a literal to be pretty elementary, and am surprised by this side-effect!

I have found a workaround: change the literal parameter into an expression (e.g. use "57+0" instead of "57" - again, see the attached QVW). But this is, of course, a bit clumsy!

Regards,

Angus.

- 120104 sub param vars Doc Support Info.txt 17.7 K
- 120104 sub param vars.qvw 127.0 K

Tags: literals, variable, variable_overview, script, subroutine

Rob Wunderlich *2,718 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 4, 2012 10:32 AM

Yes, it's kind of an oddity about subroutines and unquoted parms. I think Ralf has pointed it on the forums. You can avoid the implicit variable creation by quoting the literal.

call TraceValue('31');

vs

call TraceValue(31);

It's because everything is evaluated. Anything not recognized as a literal (quoted), function or operator is assumed to be a variable. Variables can be automatically created by reference in the CALL statement.

-Rob

Ralf Becher

Re: Literals as sub parameters become variables Jan 4, 2012 3:49 AM

in response to Rob Wunderlich

Yea, this is a kind of bug:

http://community.qlikview.com/message/145444

Sometimes I also get empty variables (no varable name). I presume this comes from empty evaluations: $(<nothing>)

- Ralf

- sleep800.qvw 141.8 K

Rob Wunderlich *2,718 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 4, 2012 1:30 PM

in response to Ralf Becher

I wouldn't agree that it's a bug or that it should be changed. It's actually a useful behavior.

31 is not a QV literal. '31' is a literal. Any parameters meant to be passed by value should be a proper string.

The unusual bit is how CALL creates implicit variables. Both of these CALLs will create a variable.

```
CALL mysub(31)

CALL mysub(abc)
```

Parameters are evaluated before being read by the sub. if you pass the parm today(1), the sub receives the number, not the function call. An implicitly created variable seems to use an intial value of the name. So abc will not persist in the UI because it's evaluated value is null -- but it was created.

I find the implicit creation useful and I rely on it. Consider a SUB that "returns" a result by setting a variable. The caller should be able to pass in the variable to be set. For example:

```
Sub ByRef (retvar)

        retvar = 10;

End Sub
```

CALL ByRef(Z);

After the CALL, variable Z=10. Z did not need to be created prior to the call.

P.S. I don't know if this has anything to do with the variables created by SLEEP. That one I would call a bug because:

SLEEP 200;

SLEEP '200';

Both create a variable "2000". There doesn't seem to be any way to avoid it.

-Rob

Ralf Becher

Re: Literals as sub parameters become variables Jan 4, 2012 4:11 PM

in response to Rob Wunderlich

Hi Rob,

I see no benefit on auto-creation of variables without declarations. Only confusion and unexpected behavior.

I don't know any programming or script language with this behavior. This is very unusual. Also I cannot understand why a variable name can consist of digits only or even is <nothing>. Nobody would expect this.

Btw. there is no equal treatment of numerical parameters at all: Why today(1) isn't creating a variable "1"?

Your example of passing today(1) as a parameter which results into a number (the date) because it's evaluated is indeed a normal behavior which I would expect. This is a normal functional nesting.

I think this concept would need more clarification from the creator.. ;-)

- Ralf

Rob Wunderlich *2,718 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 4, 2012 6:28 PM

in response to Ralf Becher

Hi Ralf,

At the risk of breaking my New Years resolution to stop trying to have the last word...

>I see no benefit on auto-creation of variables without declarations. Only confusion and unexpected behavior.

I thought I pointed out a great benefit - no need to predefine return variables. But what would you have them do if the variable wasn't predefined? Throw a runtime error?

>I don't know any programming or script language with this behavior. This is very unusual.

I think it's very common in scripting languages. For example, VBScript (w/o option explicit) and REXX to name two. QT actually copied the VBScript behavior for creating and copying out variables. For example, in VBScript:

Literals as sub parameters become...

```
Sub mysub (parm1)

        parm1 = "hello"

End Sub



Sub testit

        Call mysub(abc)

        msgbox abc     'abc will have the value "hello"

End Sub
```

REXX also uses the convention of initializing variables to the same value as the variable name.

>Also I cannot understand >why a variable name can consist of digits only or even is <nothing>. Nobody would >expect this.

I'l admit that "allowing" variable names with digits only is strange. I put allowing in quotes because the syntax editor does flag:

```
SET 27=68;
```

with a red line syntax error squiggle. But it will execute and there is nothing in the doc that says it's not allowed.

>Btw. there is no equal treatment of numerical parameters at all: Why today(1) isn't creating a variable "1"?

I may be misunderstanding your point, but I don't see the relationship between how native functions behave and script SUBs.

>Your example of passing today(1) as a parameter which results into a number (the date) because it's evaluated is >indeed a normal behavior which I would expect. This is a normal functional nesting.

I misspoke. It's actually the string form of the date that's passed, not the number.

>I think this concept would need more clarification from the creator.. ;-)

I think that would be great.

I contributed to this thread because I wanted to make a few points I thought were useful.

1. Don't change this behavior without serious consideration. A lot of script will break.

2. Variables can be passed by reference -- myvar -- or value -- '$(myvar)'. I find it handy to have both.

-Rob

http://robwunderlich.com

Angus Monro

Re: Literals as sub parameters become variables Jan 4, 2012 6:51 PM

Hi Rob,

I don't mind the QV behaviour of implicitly creating variables (Sorry Ralf!) - it's not an uncommon approach in a variety of languages, so I've become comfortable enough with it. (It is, however, an undocumented 'feature'). But I don't think it should occur when a literal is the parameter, and think it contravenes the Reference Manual (for QV10).

Before I justify that statement: I'm going to change terminology and now use the term "constants" rather than "literals", because I've just looked in the QV10 Reference Manual and found that this is the term it uses.

So, the commentary on Call (p294) states "Each item in the list [of the actual parametes] may be a field name, a variable name or an arbitrary expression.". (The Script interpreter is clearly treating a numeric value as the 2nd option - a variable name).

The commentary on Expression Syntax (p365 / s. 22.1) states that the general syntax for an expression is:

expression ::=(constant |

fieldref |

operator1 expression |

expression operator2 expression|

function |

(expression))

where

constant is a string ... enclosed by single straight quotation marks ... or a number. ...

So, a value such a 10 clearly qualifies as a *constant*, and a *constant* clearly qualifies as an *expression* in and of itself, without requiring there to be any operator. Finally, returning to the CALL commentary, "if the parameter in the call statement is a variable name, copied back out again upon exit from the subroutine." By implication, this behaviour shouldn't happen with parameters other than variable names. Consequently, IMO, the CALL statement should treat such a value as an expression, not a variable name.

Consequently, I'm expecting that the script interpreter should be able to tell the difference between a variable name and a numeric constant in the context of a call statement.

Finally, if it makes this distinction, then I don't think your code, Rob, would break, since the interpreter will (correctly) identify your variable names as variables rather than constants, and so do the correct passing-back behaviour.

Regards,

Angus

Rob Wunderlich *2,718 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 4, 2012 6:56 PM

in response to Angus Monro

Nice analysis Angus. I agree that numbers should be treated as expressions and therefore not generate a variable. And I don't think that change only would break any existing script.

It's implementation might be a bit awkward. While

SET 27 = 32

is currently allowed, what should be behavior of

CALL mysub(27)

Pass the number 27 or a pointer to the variable 27?

Thanks,

Rob

John Witherspoon *4,709 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 4, 2012 7:11 PM

in response to Rob Wunderlich

I'd say that the variable syntax shouldn't allow variables that can be evaluated as numbers, like 27. It should therefore error on the set, and the call would be passing a number. (Edit: For backwards compatibility purposes, I wouldn't actually make this change. This is just how I think it should have worked.)

We already have a mechanism for handling the situation, though. If I create a variable 27 with value 32, then 27 = 27, but [27] = 32. So just structure your call appropriately:

```
CALL mysub(27)   // This is passing the number 27
CALL mysub([27]) // This is passing the variable 27, with value 32
```

I have not actually tried these to see if they behave like I would expect. But that's what I'd expect, and if it doesn't work that way, that's how I'd suggest it *should* work.

Angus Monro

Re: Literals as sub parameters become variables Jan 4, 2012 7:23 PM

in response to Rob Wunderlich

Thanks Rob.

On one hand, the behaviour of the interpreter could go down the path of

having a disambiguation rule that keeps things as "obvious" as possible

for a reader of the code; viz.:

   if it looks like a number, then treat it as a number unless context

expects & requires otherwise.

Four such contexts are:

statements expecting a variable name, such as:

SET 27=abc

SUB MySub(27,var2) ... ENDSUB

expressions expecting a variable name:

$(27)

[27]

Downside: it becomes impossible to pass a variable that has a number as its name as the parameter to a subroutine. I can live with that.

On the other hand, it could go down the path of banning numbers as variable names altogether.

The language purist in me biases me towards the latter option. But the QlikView philosophy on naming in general feels to me like it would lean towards letting users have maximum flexibility to the point of self-harm. At the end of the day, I've learned to relax over the years, and am not really too fussed . So long as it's well-documented.

Regards,

Angus.

Ralf Becher

Re: Literals as sub parameters become variables Jan 5, 2012 3:11 AM

in response to Angus Monro

I've played around a bit. These variants are working w/o variable creation:

```
CALL mysub(27+0);
```

```
CALL mysub((27));
```

- Ralf

Angus Monro

Re: Literals as sub parameters become variables Jan 5, 2012 5:33 PM

in response to Ralf Becher

I think your 2nd variant - parenthesising the value - gets the prize for the most elegant workaround, Ralf, with the added bonus that it would also be the best-performing option, since it incurs no arithmetic or conversion operations. Very nice.

Angus Monro

Re: Literals as sub parameters become variables Jan 5, 2012 5:53 PM

I'm feeling that this discussion has run its course - and have thoroughly enjoyed interacting with you all, I might add - and would like to pull-together some conclusions.

A. I think we're all agreed that it's a bona fide defect that Qlikview is turning numeric constants into variables when provided as subroutine parameters, especially since the Reference Manual is pretty clear that this shouldn't be an expected result.

Am I right in reading we're agreeing on this?

B. The cleanest, best-performing workaround is to put such constants in parentheses.

C. It's up to QlikTech how they resolve this, of course, but it seems like the platform needs to have a token-interpretation rule that interprets tokens that look like numeric constants as numeric constants, unless the linguistic context requires a variable name. This should provide satisfactory backward-compatibility for coding techniques that use implicit variable creation in the context of value-passback in subroutine calls, since surely no-one in their right mind would want to use this technique using a sequence of only digits as the variable name.

D. We've discussed a few language philosophy issues, which I'll name for completeness, but these clearly won't be resolved in this thread: desireability of implicit variable creation; consistency of parameter treatment between user-defined subroutines, built-in subroutines and built-in functions; flexibility of variable-name rules.

Angus.

Rob Wunderlich *2,718 posts since Apr 15, 2009*

Re: Literals as sub parameters become variables Jan 5, 2012 11:48 PM

in response to Angus Monro

Nice summary Angus. Thanks for your contributions.

Now does someone want to open the ticket and see if we can get a bug?

-Rob

Ralf Becher

Literals as sub parameters become variables Jan 6, 2012 3:59 AM

in response to Rob Wunderlich
Just an additional note. In the case of sleep (maybe other statements too) parenthesising the value wouldn't be enough:

```
sleep (1200);       // will create variable "("
```

```
sleep (1200+0);  // will not create any variable
```

- Ralf

Angus Monro

Literals as sub parameters become variables Jan 11, 2012 6:09 PM

in response to Rob Wunderlich
Done. Case 00087203.

Angus Monro

Literals as sub parameters become variables Feb 12, 2012 8:40 PM

in response to Angus Monro
Confirmed by QlikTech Issue Analysis team as a bug, ID 44972.

"Due to uncertainties regarding potential related problems and program priorities we cannot guarantee a fix date at this time."

1 2 Previous Next