Mid() vs Mid$()

Asked 15 years, 10 months ago Modified 6 years, 10 months ago

Viewed 25k times

17

According to the documentation in VB6 the Mid() function returns a variant, but Mid$() returns a string and apparently this is more efficient.

My questions are :

1. *What simple test can I use to discern the difference in performance ?* I tried monitoring a simple app performing a few string operations with Perfmon, but there was no discernible difference.
2. *Is it worth worrying about?* I've gotten into the habit of using the $-ized functions, but should I recommend everybody on my team to use it as well?

vb6

Share

Improve this question

Follow

edited Feb 9, 2018 at 2:32

asked Feb 6, 2009 at 8:46

jakdep

872 ● 1 ● 11 ● 28

3 Answers

Sorted by: Highest score (default)

25

Isn't worth worrying about. It's a remnant from Microsoft Basic of 15-20 years ago when a fast processor was orders of magnitude slower than anything today.

It has a certain esthetic appeal to use Mid$ rather than let VB determine what your datatypes are, though. And if you have any loops that are executing it, say, thousands of tiems a second, then your curiosity factor might increase. Otherwise, neh.

Here's a link to someone who measured the difference. Mid$ was about 2.5 times as fast as Mid. Including tests going back to VB4.

Share Improve this answer Follow

edited Feb 6, 2009 at 8:55

answered Feb 6, 2009 at 8:49

dkretz

37.6k ● 13 ● 83 ● 140

Thanks. Interesting link, exactly what I was looking for. – jakdep Feb 6, 2009 at 10:33

1 Might want to consider including a 3rd party library like Stamina. Includes many string handling routines written in C that are much faster than VB6. hallogram.com/stamina/routines.html – Gary Kindel Feb 9, 2009 at 14:37

4

Whilst performance between them is negligible its not really a differentiator as to which to use anyway.

There can be some nuances when using a variant when a strong type is required. For example what happens when you pass a variant to a parameter expecting a ByRef string? Nothing bad but something a little more than passing an address happens.

Hence if you know that you want to work with strings then go ahead and use the $ versions of these functions the behaviour of them and their use in other expressions is simpler and better understood. If you know you need a variant and your inputs are variants then sure use the non $ versions.

Share Improve this answer Follow

edited Feb 6, 2009 at 9:34

answered Feb 6, 2009 at 8:59

AnthonyWJones

189k ● 35 ● 235 ● 307

> Exactly! Variants should be avoided whenever possible.
> – mafu Feb 6, 2009 at 9:29

3

Honestly, I think it's negligible.

Maybe you can try something like this. Download the "High-Performance Timer Object" from

http://ccrp.mvps.org/, do a long loop (1.000.000 iterations or so) of string operations, and measure the run time difference. By "operations" I mean: Concatenation of `Variant`s as opposed to concatenation of `String`s. `Mid()` and `Mid$()` will very likely perform the same. OTOH - you can test *that* as well.

If you did, I'd be interested if you posted the results.

Share Improve this answer

Follow

answered Feb 6, 2009 at 8:53

Tomalak

338k ● 68 ● 545 ● 635

3 +1 for the Timer Object link. I compared Mid("ABC") with Mid$("ABC"), as in the link provided by le dorfier, over 100,000,000 iterations and measured the duration with High-Performance stopwatch. Mid() took 35.364 seconds and Mid$() took 13.56 seconds. So, it matches the results shown in the link. – jakdep Feb 6, 2009 at 11:34

I am surprised. This really *is* a mentionable difference. – Tomalak Feb 6, 2009 at 12:45

2 the difference is definetly measurable, and not negligible when processing a lot of incoming data ... changing all mid/left/right/etc function into their $ equivalent (as well as optimizing some loops and other processes reduced the refreshrate in one of my application from 3 seconds to 1/10 of a second) ... the $ made up for about half a second of this increase in speed – Hrqls Nov 15, 2012 at 8:21