LETTERS

Letters to Visual Basic Programmer's Journal *are welcome. To be considered for publication, letters must include your name, address, and daytime telephone number. Letters may be edited for form, fit, and style. Please send them to Letters to the Editor, c/o Fawcette Technical Publications, 209 Hamilton Avenue, Palo Alto, CA 94301-2500; fax them to 415-853-0230; or send them electronically on CompuServe to 74774,305 or on the Internet at 74774.305@compuserve.com.*

CHAMPING AT THE 32 BITS

Ward Hitt's article "Tune for Blazing Speed" in the January issue of *VBPJ* was one of the most relevant and useful in the five-year history of the publication. I have ordered his book.

Now, his treatment of the 16-bit/32-bit VB 4.0 issue is also top notch! Superior! One thing, though. It's like saying, "Oh, and by the way, the world as we know it will end on March 7, 1996!" His numbers regarding speed, which I believe 100 percent because he did the legwork and actually tested, fly in the face of everything we've been led to believe regarding VB 4.0 and Windows 95 in general.

The ads for VB 4.0 pretty much equate "Blazing Speed" with 32 bits! Now, to find that we'd have to be nuts to compile in 32 bits (that's certainly the way I take this—the things that are faster in 32 bits amount to nothing and I can do without the Rich Text control) is just plain shocking. We're being lied to. His article is the only hint of truth I've seen.

Other people are getting hints of this. A few of us have noticed that VB 3.0 apps are running under Windows 95 faster than VB 4.0 apps (had one buddy say that his VB 4.0 app turned his Pentium into a 386!). That's because we always compile in 32 bits. I mean, why shouldn't we? Why are we running this neat 32-bit operating system anyway?

I have two VB machines right now: one a 486/66 with eight megs of RAM (like the machine in the article) running Win 3.1 and VB 3.0 and another, a P133 with 16 megs of RAM running Win 95 and VB 4.0.

I can tell you this. I'm gonna do everything on the old machine for now. To heck with VB 4.0! Who needs it? I've looked and looked. There's no reason... none... for me to build stuff in 4.0 and have to include a run time that's twice the size of VB 3.0's. Where's the "added value"?

I think Ward Hitt's article and discoveries need more attention. Is he up to being Microsoft Enemy Number One? Being a VB professional, this must be quite the personal dilemma for him (would be for me). He's done excellent work here. He'd be right to at least fear Microsoft to some measure. But he'd be more right to spread the word on this in some manner fitting its importance.

Jeff Morris
received by e-mail

Thanks, Jeff, for your kind words about the article. I agree with you that VB4 32-bit performance is, to say the least, disappointing. And as you point out, the areas where 32-bit code is superior, such as mathematics, are not critical in some real-world optimization situations.

I also agree that Microsoft blundered when they implemented an architecture that requires such huge distributables. With the world going on line, it's essential that download times be kept to a reasonable level. I simply don't buy the oft-heard argument that we can distribute EXEs without the run times, assuming that our users will already have the latest DLLs installed. This only worked when all that was needed was VBRUN100.DLL.

But don't throw out the baby with the bath water. There are plenty of reasons to use VB4, most of which never made the top-10 feature lists. The new Printer object finally provides decent printing support. MDI children can now be hidden. Named and Optional parameters make creating reusable subroutines even easier. The rational DAO model simplifies database programming. The new editor is far superior. And the list goes on.

Of course, no feature excuses performance slowdowns in a development tool, and you can rest assured that VBPJ *will stay on top of this issue, now and in the future. We're currently preparing a follow-up to the performance article, which will compare VB3, VB4-16, and VB4-32 performance on Windows 3.1, 95, and NT using the same scientific approach used in my* Optimizing Visual Basic 4 *book and the January article. Look for it in the next couple of months.—W.H.*

A MATTER OF MAINTENANCE

I think John Vaughn's editorial is very thought provoking ["The True Cost of Code Maintenance," *Visual Basic Programmer's Journal*, December 1995].

If his assumptions are correct, the entire RAD concept is going to fail miserably because *nobody* can afford to pay

VISUAL BASIC® PROGRAMMER'S JOURNAL

PUBLISHER & EDITOR
James E. Fawcette

ASSOCIATE PUBLISHER
ADVERTISING DIRECTOR
Ellen Gould

ART AND PRODUCTION DIRECTOR
Michael Hollister

EXECUTIVE EDITORS
Matthew Carter, Frank Moncrief, Lee Thé

MANAGING EDITOR
Christine McGeever

ASSOCIATE MANAGING EDITOR
Nina Goldschlager

ASSOCIATE EDITOR
Amy Little

SENIOR DESIGNERS
Kristen Avery, Margaret Horoszko, Kimberly Bell

ELECTRONIC MEDIA DESIGNER
Robert W. Scoble

CONTRIBUTING EDITORS
Daniel Appleman, Chris Barlow, Andrew Brust
Carl Franklin, Roger Jennings, Deborah Kurata
Craig Leach, Mark Novisoff, Sam Patterson
Karl E. Peterson, Keith Pleas, Richard Hale Shaw
Jonathan Wood, Jonathan Zuck

CONTRIBUTORS
Tom Campbell, Dana Cline, Steve Jackson,
L.J. Johnson, Paul Litwin, John Updike,
William R. Vaughn, Peter Vogel

TECHNICAL REVIEW BOARD
Daniel Appleman, Andrew Brust, Douglas Haynes
Ward Hitt, Deborah Kurata, Patrick O'Brien
Sam Patterson, Karl E. Peterson, William R. Vaughn
Phil Weber, Don Willits, Jonathan Wood, Jonathan Zuck

EDITORIAL ADVISORY BOARD
Richard Finkelstein, Craig Goren, David McCarter
Jim Lisiak, Bill Neville, Joe Robison

CIRCULATION
Tena Larson, *Circulation & Marketing Director*
Bill Hooven, *Circulation Manager*
Sherry Thomas-Zon, *Circulation & Marketing Associate*
Karen Koenen, *Associate Circulation Manager*
Shirley Modric, *Circulation Coordinator*

CONFERENCES
Elaine Kato, *Conference Director*
Michael Sturdivant, *Conference Coordinator*
Leslie Tighe, *Conference Coordinator*

ADVERTISING SALES
Robin Nakamura, *National Sales Manager*
Craig R. Altman, *Senior Regional Account Manager*
(Southwest, Southeast, South Central)
Cheri Poff, *Regional Account Manager*
(Northwest & North Central)
415-833-7100 or 800-848-5523
Carolyn Lighty & Andy Skurna, *Regional Account Managers* (Northeast & Mid-Atlantic)
201-712-0044 or 800-685-2405
Gwen Rambis, *Sales Associate*

TECHNICAL DIRECTOR
David Lynch

ANCILLARY PRODUCTS COORDINATOR
Andrew King

PROJECT MANAGER, ELECTRONIC PRODUCTS
Robyn K. Glennon

CUSTOMER SERVICE
Lara L. Ireland, Katie Hathaway, Regan Moore
(CompuServe: 75451,2343)

ASSISTANT TO THE PUBLISHER
Trisha Merriam

ACCOUNTANT
Jena Siress

RECEPTIONIST
Connie Pulmano

DESIGN TEAM
Enigma Inc., Michael Carr, Cynthia Engler
Linda Magyary, Kate Paddock, Lynn Sanford

FAWCETTE TECHNICAL PUBLICATIONS

James E. Fawcette
PRESIDENT

Wilson, Sonsini, Goodrich & Rosati
CORPORATE COUNSEL

HOW TO REACH US

SUBSCRIPTION SERVICES

For subscriber services, please send your magazine mailing label with your inquiry to *Visual Basic Programmer's Journal*, P.O. Box 58872, Boulder, CO 80322-8872; call us at 303-684-0365; or send us a fax at 303-661-1816.

CUSTOMER SERVICE

To subscribe to *Visual Basic Programmer's Journal* or the VBCD, or to register for VBITS, please call Customer Service at 800-848-5523 or 415-833-7100. You can also send e-mail to **75451.2343 @compuserve.com** or post a message in the Customer Service section of the *VBPJ* Forum on CompuServe.

EDITORIAL OFFICE

Fawcette Technical Publications
209 Hamilton Avenue
Palo Alto, CA, USA, 94301-2500
Phone: 415-833-7100 • Fax: 415-853-0230

LETTERS TO THE EDITOR, PRODUCT ANNOUNCEMENTS, AND BASIC HERO CANDIDATES

Please send or fax letters and product information to the Editorial Office listed above; address them to Letters to the Editor and New Products Editor, respectively. If you'd like us to consider your application for the Basic Heroes column of *Visual Basic Programmer's Journal*, please send us a brief description of the application along with your phone number and e-mail address; we will contact you if we decide to profile your application.

As an alternative to contacting us by mail or fax, you can also send letters, product information, suggestions for Basic Heroes, and other article ideas to **74774.305@compuserve.com** or post them in the Talk to Editors section of the *VBPJ* Forum on CompuServe.

VBPJ DEVELOPMENT EXCHANGE ON THE WORLD WIDE WEB

Through the *VBPJ* Development Exchange, located at **http://www.windx.com**, you can subscribe to *Visual Basic Programmer's Journal* or the VBCD; register for VBITS; order books; download source code that accompanies articles in *VBPJ* as well as sample applications; and browse resources for Windows developers, including listings of add-on tools for Windows developers and links to other software-industry Web sites.

VBPJ FORUM ON COMPUSERVE

Typing "**GO WINDX**" using WinCIM will bring you to the *VBPJ* Forum on CompuServe, where you can post questions about Visual Basic programming and get them answered by expert section leaders as well as other members of the Visual Basic community. You can also post messages for our customer service representatives, our conference staff, and the editors of *Visual Basic Programmer's Journal*. Code that accompanies articles in *VBPJ* is available for download in the Magazine Library.

VBPJ SITE ON THE MICROSOFT NETWORK

"**GO WINDX**" also takes you to our site on The Microsoft Network, where you can read about development tools that enhance Visual Basic and Visual C++, and send e-mail to the editors or customer service representatives.

REPRINTS AND PERMISSIONS

For all *Visual Basic Programmer's Journal* editorial and advertising reprints, contact Andrew King at our Editorial Office listed above or by e-mail at **74737.2202@compuserve.com**. If you would like to quote from an article, please specify the issue date and title of the article, the portion you would like to quote, and the purpose. You may also request permission to photocopy portions of *VBPJ* for internal or personal use.

LETTERS

that much for application program maintenance. I think that a new paradigm is being invented in our industry, and we don't even realize it.

The new paradigm is: "Application programs are disposable." Because the cost of development of an application program is, as he says, a small fraction of the maintenance of it, the tendency will be to write a new application program instead of maintaining the existing one.

This tendency is exaggerated by the "component model" of development, in which the major pieces of functionality are packaged as components which are purchased. In this case, the application program becomes a thin layer of GUI "glue" on top of the components.

The components will usually not require internal maintenance (if you buy them) once the application is fielded, so you only have to perform maintenance on the GUI glue part. If the application program is complicated or hard to maintain, then it may be cheaper to pitch the old and make a new one, rather than try to maintain it.

It is only feasible to discard a program that "works" in favor of development of a new one when the cost of development is much less than the cost of maintenance. Thus as we adopt cheaper development tools, it becomes more attractive to discard the old applications instead of maintaining them.

Also, because most of the "hard" code is now buried in purchased (or internally maintained) components, the maintenance of the thin-layer GUI application program becomes easier. This is especially true given that the "hellishly powerful programming environment" is also available to the maintenance programmers.

Now this idea of disposable application programs is as yet unproven in practice, so there is a lot of work in making the concept viable in production. I presume your magazine will be generating articles in the future about "design for maintainability" and other methods of improving the maintenance process.

We may be cursed, as in the Chinese proverb, to "live in interesting times"!

John Alderman
received on CompuServe

I must take exception to the Guest Opinion by John Vaughn ["The True Cost of Code Maintenance," *Visual Basic Programmer's Journal*, December 1995]. Mr. Vaughn argues that the ratio of maintenance time to development time for Visual Basic as compared to a traditional language will rise from about 4:1 (80-year maintenance to 20 years development in his example) to perhaps 49:1 (98 years maintenance to two years development in his example). Thus he argues that managers had better be very careful about the maintenance burden they are assuming in the brave new world of Visual Basic.

Fortunately his argument is specious. It is based on the assumption that maintenance tools will operate at the efficiency of traditional languages while only development will be done using efficient modern tools such as Visual Basic. By analogy (and on a down-to-earth level) one would argue that while a giant earthen dam might be built using power equipment such as bulldozers and backhoes, all maintenance must be done with hand shovels. Or on a programming level this dictates that a program written in C/C++ can only be maintained at the machine or assembly code level.

Why doesn't Mr. Vaughn believe maintenance programmers will use the same tools as the developers? Sure there will be startup problems of maintaining in a new paradigm, but these problems will be solved with tools as powerful as those used for development. There may be many reasons to fear venturing forth into the brave new world, but program maintenance isn't one of them.

Jacob L. Raab
Summit, New Jersey

Read the last paragraph of Mr. Raab's letter. In its first two sentences he encapsulates his argument and lays bare its flaws.

"Why," he asks, "doesn't Mr. Vaughn believe [VB] maintenance programmers will use the same tools as the developers?" In fact I believe it entirely possible that they will do just that. But if they do they will be ill served. Development and maintenance are distinct activities. The tool that works well in one arena may fail utterly in the other.

Consider the primitive COBOL generators that enjoyed a brief vogue in the early 1970s. For their time they were truly capable development tools, but the code they produced was virtually impossible to read. Here we had a development tool not merely worthless in the maintenance process, but actively deleterious to it.

In the next sentence Mr. Raab, perhaps sensing he is on shaky ground, abandons his assertion that VB maintenance programmers will use the same tools as developers. Instead they will use "[maintenance] tools as powerful as those used for development." Note the use of the future tense. These tools do not yet exist, but Mr. Raab displays a touching faith that the clever people who now write VB development tools will be able to produce them when the need arises. This assumes that success in one endeavor is a sure predictor of another, a common fallacy. In the 50s the world was certain that the scientists who had produced the first fission reactors *would have no trouble harnessing fusion. Forty years and many billions of dollars later, the world is still waiting. When Mr. Raab rests his argument on similar logic, he rests it on air.*

Mr. Raab's reasoning is easily refuted. Nonetheless his letter is troubling, because people will read it and believe. They will believe because they like Visual Basic and do not want to hear that there may be a serpent in the garden.

In a discipline as volatile and fast-changing as information technology this tendency to embrace the pleasant fiction and reject the hard reality is very dangerous. It would be useful if articulate and intelligent professionals like Jacob Raab worked to discourage rather than foster it.—J.V.

NOT CUT-AND-PASTE, BUT NOT BAD

While Carl Franklin's CopyTable function certainly works as advertised, there's a much simpler way to copy a table from one database to another [Q&A, "Tabular Genetic Engineering," *Visual Basic Programmer's Journal*, January 1996]. Use a SELECT...INTO...IN query:

```
Sub CopyTable (dbSrc As Database, _
  sDestDB As String, sSrcTbl As _
  String, sDestTbl As String)

  Dim dbTemp As Database
  Dim SQL As String
  Const DB_LANG_GENERAL = _
    ";LANGID=0x0809;_
    CP=1252;COUNTRY=0"

  ' Create destination database if
  ' it does not exist
  If Len(Dir$(sDestDB)) = 0 Then
    Set dbTemp = CreateDatabase_
      (sDestDB, _
      DB_LANG_GENERAL)
    dbTemp.Close
  End If

  ' Copy table into destination
  ' database
  SQL = "SELECT * INTO " & sDestTbl
  SQL = SQL & " IN '" & _
    sDestDB & "' "
  SQL = SQL & "FROM " & _
    sSrcTbl & ";"
  dbSrc.Execute SQL

End Sub
```

This method requires substantially less code and will likely run much faster than explicitly duplicating each field and inserting records individually into the new table. The downside is that the above procedure does not trap and log errors as Carl's function did.

Phil Weber
VBPJ Technical Review Board