

## Re: Error handling in VB6

---

*Source:*

<http://www.tech-archive.net/Archive/VB/microsoft.public.vb.general.discussion/2008-03/msg00371.html>

---

- *From:* "Tony Proctor" <[tony\\_proctor@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:tony_proctor@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Tue, 4 Mar 2008 15:27:57 -0000
- 

The VB syntax for catching and handling errors is definitely not "structured", but the internal Windows support for exceptions (which isn't VB-specific) is structured

In principle, you could set a single exception handler (i.e. "On Error Goto label" in VB6 terms) in your root Main() procedure, and that would catch all exceptions thrown in Main(), and in anything called from Main().

Unfortunately, this ideal doesn't work for event-driven VB programs – which happens to be the vast majority. For some reason (possibly one of 'determinism'), exceptions are not thrown out of event handlers back to whatever is next on the run-time stack. Any exception in an event handler therefore ends up being fatal to the program. The only way to avoid this is to always have an exception error in each event handler.

Tony Proctor

"MikeD" <[nobody@xxxxxxxxxxxxx](mailto:nobody@xxxxxxxxxxxxx)> wrote in message [news:eLoZt4ffIHA.2268@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:eLoZt4ffIHA.2268@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

"Ronald Raygun" <[invalid@xxxxxxxxxxxxx](mailto:invalid@xxxxxxxxxxxxx)> wrote in message [news:bsCdnaCcXftSqVDanZ2dneKdnZydnZ2d@xxxxxxxxxxxxx](mailto:news:bsCdnaCcXftSqVDanZ2dneKdnZydnZ2d@xxxxxxxxxxxxx)

Currently, the error handling in my VB6 app is very unstructured – it could be that I still haven't got my head around VB6 way of handling errors. It just doesn't "feel right" at the moment – I keep feeling that an error can "slip by" somehow, and crash my application.

My routines generally look either like this:

```
Private Sub DoThis()  
'decl here  
OnError GoTo SomeErrHandler  
...  
Exit Sub
```

## Re: Error handling in VB6

```
SomeErrHandler:  
MsgBox err.Description, vbCritical, APP_NAME  
End sub
```

OR this

```
Private Sub DoThat()  
'decl here
```

```
OnError Resume Next  
...
```

```
if err.Number <> 0 GoTo SomeErrHandler  
Exit Sub
```

```
SomeErrHandler:  
MsgBox err.Description, vbCritical, APP_NAME  
End Sub
```

First thing I would like to do is to have a 'global' Error catcher in my application that provides a last resort error catcher, that catches any unhandled errors.

Secondly, is this (above), the recommended way of handling VB6 errors – or does someone know of a better (i.e. more robust, "failsafe") way?

2 things up front:

1. VB6 has no such thing as "truly" structured error handling.
2. VB6 has no such thing as a global error handler.

Best thing you can do to make sure EVERY error is handled in some manner is to put an error handler in EVERY event procedure which contains code. Keep in mind that if you have an active error handler in an event procedure, that error handler is active in any procedures/functions you call from that event procedure if the procedure/function does not have its own error handler.

Personally, I only use On Error Resume Next in "special circumstances". Usually, the special circumstance is when it's OK to simply ignore the error or if on the very next line of code, I need to check for a specific error number. After that special circumstance is over, I go back to the "normal" error handling. Something like this (air code):

```
Private Function MyFunc() As Long
```

```
On Error GoTo EH
```

## Re: Error handling in VB6

<code>

On Error Resume Next

(1 or 2 lines, at most, of code)

If Err.Number = <specific error number> Then

<code for special handling of this error number>

End If

On Error GoTo EH

<code>

Exit Function

EH:

<error handling code>

End Function

I would never use On Error Resume Next and then GoTo my error handler, but that's more personal preference than right or wrong (although IMO, it's extremely unstructured). If you're going to do as you show in your 2nd example, you might as well litter your code with other GoTos.

—

Mike

Microsoft MVP Visual Basic