The dBASE Book Plus – Errata

By Ken Mayer

September, 2016

As with any book of the nature of this one, sometimes a reader will find an issue that either needs more explanation than was provided, or a possible problem with some of the example code.

This errata “sheet” will contain any corrections that I feel need to be added (and if I create a second edition of the book, I intend to include them in that).

In February of 2016, Peter Schneider noted a couple of things.

Chapters 4-6:

DataLayer.dmd – The Data Module used in Chapters 4-6:

In the *AddRow* method, I use the following:

```
if type( "xValue" ) == "C" or type( "xValue" ) == "V"
```

The type() function as explained in the OLH does not return a letter “V”, but I tripped across it, perhaps as an undocumented feature, when working with the Firebird tables for the sample application when working with *varchar* field types. In theory the value of “xValue” if character should return a “C”, but I was being very cautious at the time and added the second comparison above. It may not be necessary.

In the *Common_OnOpen* method of the data module, I made a mistake that actually ended up not being caught in the sample code for a couple of reasons.

The problem is that the code assigning the PrimaryKey property is assigning it to the data module rather than to the query. This means that if you activate a query, this property gets set, but if you activate a second query, the property will be overwritten. For proper OOP *encapsulation*, this property belongs in the query. This line of code should be modified:

```
this.parent.PrimaryKey = this.rowset.fields[1].fieldName
```

For this to be properly encapsulated you would want to take out “.parent” in the statement above, which would give this statement:

```
this.PrimaryKey = this.rowset.fields[1].fieldName
```

However, as it turns out, this is the only place in the application this property is referenced. This property does not actually need to be there *(having examined all the rest of the code in the application, searching for references to the property)*. Most of the sample code works with the same assumption here that the first field in the table is the primary key and acts accordingly. The command shown above should be removed from the data module will no ill effects. I believe I put it there thinking I would use it but then ended up doing something else in the application. Interesting that while the code is in the data module, it is not referenced in the text of the book.

dBASE Plus 10.3

ADOQuery Object and Parameters

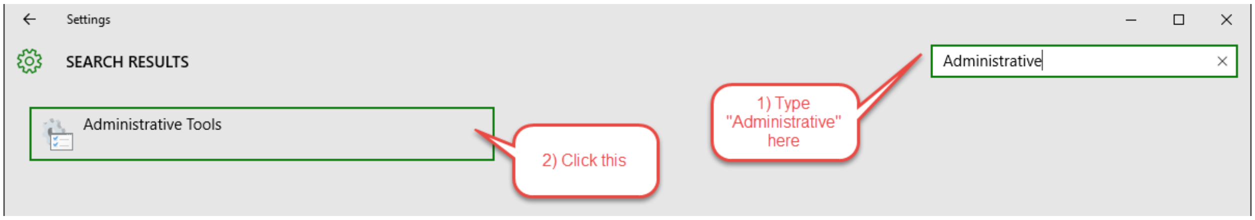
The *params* or *parameters* array used to pass items to a query sometimes didn't always work. Using the *value* property appears to be more consistent and resolve issues in this area.

```
        qInvoices.parameters["InvNum"].value := nRows
```

It is strongly recommended that you use the *value* property with your parameters.

Windows 10 – Administrative Tools (Page 36)

When helping someone out with my book, I needed to turn Firebird's database engine back on, and was having a hard time finding Administrative Tools. I found when you go to the Start menu in Windows, and select "Settings" the easiest thing to do is go to the search box in the upper right corner and type "Administrative", which will bring up "Administrative Tools". Clicking on that will bring up the screen shown on page 36.



1) Type "Administrative" here

2) Click this

Chapter 4 – CreateDatabase.prg

There appears to be a problem with this code that I never saw when I built the book. This appears to have changed in dBASE Plus 10.3.1, in February 2016.

At statements 458, 491, 505, 519, and 533 the command:

```
        qCustomers.parameters[ "Addr2"].value     := ""
```

is causing a problem, returning an error code of 239 (generic "Database Engine Error"), and not adding the record to the table. The solution, it appears, is to add a space between the quotes:

```
        qCustomers.parameters[ "Addr2"].value     := " "
```

The error disappears. You can also replace the "" (or '' [single quotes]) with the keyword *null* and the error will also disappear.

You can fix it, or download the source code again. I have uploaded a fixed version.

Chapter 12 – Catalyst ActiveX Controls

The first issue to point out is that when shipped with dBASE Plus 9 we had a “free” license to use them. However, while they still ship with dBASE Plus 10 (and presumably later versions), the license has expired. If you still have dBASE Plus 9 installed on your computer, you may be able to use these controls with dBASE Plus 10 (or later). However, deploying an application with them may be problematic. I have not tried this.

Second, the form for the file transfer protocol control has a couple of bugs in it that somehow slipped through:

Line 642: *lastErrorMessage* should be *lastErrorString*

Line 1145: oIni.setValue("FTP", "Password", cPass)

The variable should be *cPassword* not *cPass*.

These last two issues were found by Mervyn Bick and myself, and the issue with licensing we found out after a bug report was posted by Mervyn about an error: “Connection error: 12002”, noting “This product is not licensed to perform this operation.”

Misc. Typos

Page 48, top of the page:

```
      sql := "select * from "Invoices"
```

There is an extra quote in front of the word “Invoices”, it should read:

```
      sql := "select * from Invoices"
```

This was found by Luca Bertoni.