mvQB – Multi-Value API for QuickBooks

Introduction 4
mvQB Server 4
Testing QuickBooks connection 5
Multi-Value Subroutines 7
Installing the mvQB MultiValue Subroutines 7
Accuterm Install 7
After Programs Transfers 11
Customer information 13
MVQB.CUSTOMER.BUILD.XREF 13
MVQB.CUSTOMER.READ 13
MVQB.CUSTOMER.WRITE 14
MVQB.CUSTOMER.AGING 15
Vendor information 16
MVQB.VENDOR.BUILD.XREF 16
MVQB.VENDOR.READ 16
MVQB.VENDOR.WRITE 17
GL Account Number information 19
MVQB.ACCOUNT.BUILD.XREF 19
MVQB.ACCOUNT.READ 19
MVQB.ACCOUNT.WRITE 20
Journal Entries 22
MVQB.JE.BUILD.XREF 22
MVQB.JE.READ 22
MVQB.JE.WRITE 23
Inventory Items 25
MVQB.ITEM.BUILD.XREF 25
MVQB.ITEM.READ 25
MVQB.ITEM.WRITE 27
Invoices 28
MVQB.INVOICE.BUILD.XREF 28
MVQB.INVOICE.READ 28
MVQB.INVOICE.WRITE 30
Credit Memos 30
MVQB.CREDITMEMO.BUILD.XREF 30
MVQB.CREDITMEMO.READ 31
MVQB.CREDITMEMO.WRITE 32
Purchase Orders 34
MVQB.PO.BUILD.XREF 34
MVQB.PO.READ 34
MVQB.PO.WRITE 35
BILLS 37
MVQB.BILL.BUILD.XREF 37
MVQB.BILL.READ 37
MVQB.BILL.WRITE 38
QuickBooks Information 40

MVQB.QUICKBOOKS.INFO....40
MVQB.SET.LICENSE....40
MVQB.CHANGE.DATABASE....40
MVQB.TEST.CONNECTION....41
MVQB.QUICKBOOKS.OPEN....41
MVQB.QUICKBOOKS.CLOSE....41
Initialization of the MVQB routines....42
MVQB.INIT....42
D3NT – Communication Setup....42
D3/ProPlus – Communication Setup....42
D3/Linux – Communication Setup....43
Universe – Communication Setup....43
ERROR variable....44
Troubleshooting....44
Registration and Licensing....45
Quickbooks Limitations....45
Maximum number of Transactions allowed in Quickbooks:....45

Introduction

The mvQB allows you to reach millions of customers who trust and use QuickBooks, the leading small business accounting software.

- The API enables your application to share data directly with QuickBooks: no need to enter the same data twice!
- Integration is transparent: the communication between your application and QuickBooks takes place behind the scenes.
- Your customer gets the best of both worlds: the power and efficiency of QuickBooks, combined with the advanced tailoring of your application to specific business needs
- mvQB API is designed for the Multi-Value program to use. No need to learn the internal for QuickBooks.
- Reach the Broad Base of QuickBooks Customers. By adding QuickBooks support to your application, you now have a large market to sell your application.

MvQB comes in 2 parts: Multi-Value subroutines and (mvQB server) a Windows program that translates the information from the Multi-Value subroutines and QuickBooks.

You can use mvQB with the following version of QuickBooks:

QuickBooks Pro 2004
QuickBooks Premier 2004
QuickBooks Enterprise 2004

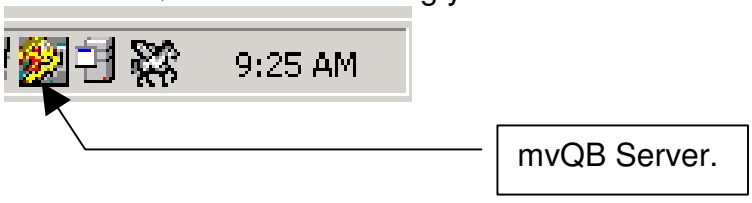
QuickBooks Pro 2005
QuickBooks Premier 2005
QuickBooks Enterprise 2005

mvQB Server

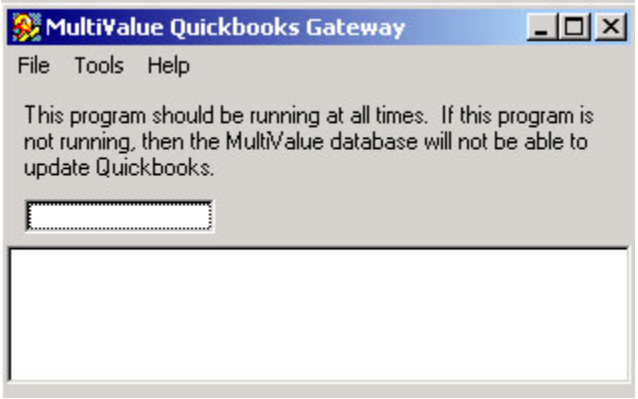
The mvQB Server must be installed on the same machine as QuickBooks. It is used to translate the requests from the Multi-Value subroutines into a format for QuickBooks to process.

The mvQB Server also controls the licensing.

When mvQB Server is running you will see it in the System Tray:

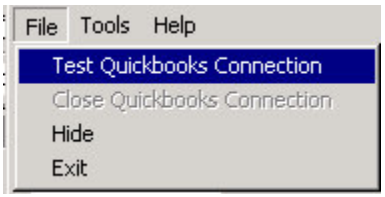


Double-click on the Icon to show the main screen.



This screen allows you to update settings, and run some tests to make sure QuickBooks and the mvQB server are communicating correctly. This program will also show general status information.

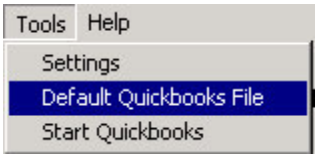
Testing QuickBooks connection



Select File | Test QuickBooks Connection to test to make sure that mvQB Server and QuickBooks are communicating correctly. This option will verify that the program can start QuickBooks, and verify version and other registration information.

If this fails, then you will need to verify that the QuickBooks is installed correctly and make sure QuickBooks is Registered. If the error message you are getting refers to the mvQB Registration and Licensing, then see “Registration and Licensing”.

The Tools menu provides a few options that help you set up mvQB and how it will interact with QuickBooks.



“Settings” will allow you to edit the Settings .INI file.

“Default QuickBooks Database” will allow you to set the default QuickBooks database file name. This should always be done. If you do not set the Default QuickBooks Database, then the QuickBooks UI will need to be open and running the database you want to update for mvQB to work.

“Start QuickBooks” will just start the QuickBooks UI.

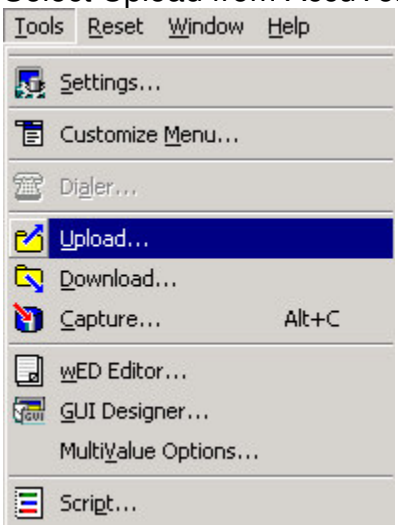
Multi-Value Subroutines

Installing the mvQB MultiValue Subroutines

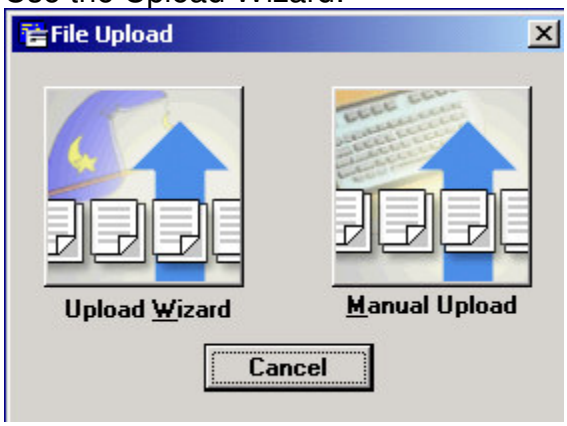
Create a file called MVQB.BP and a file called MVQB.ERRMSG. It is recommended that you place these file into a separate account, but is not required.

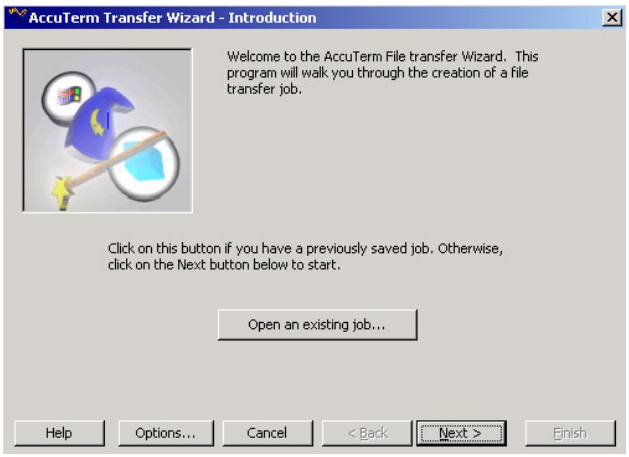
Accuterm Install

Select Upload from AccuTerm's Tool's menu:

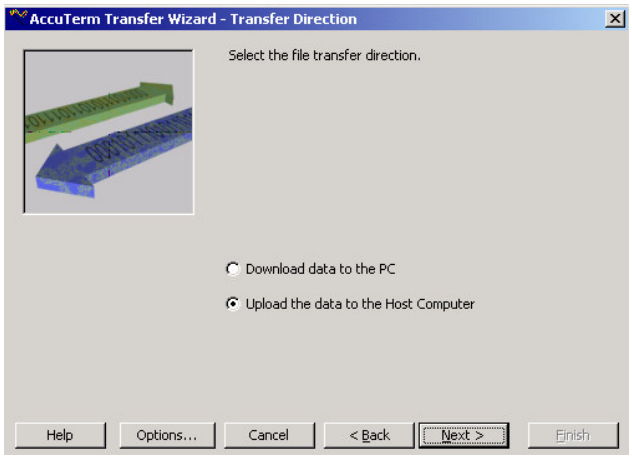


Use the Upload Wizard:

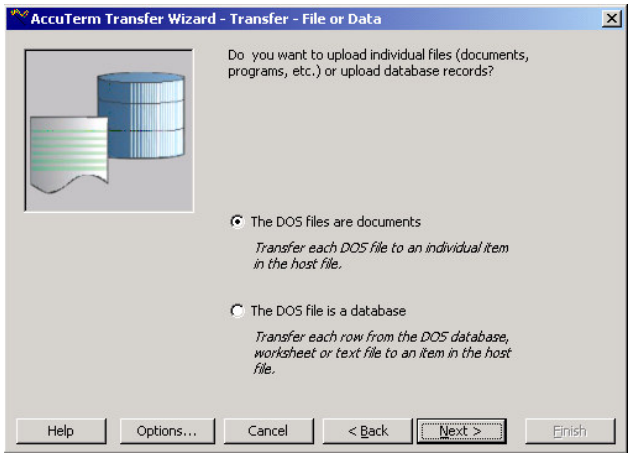




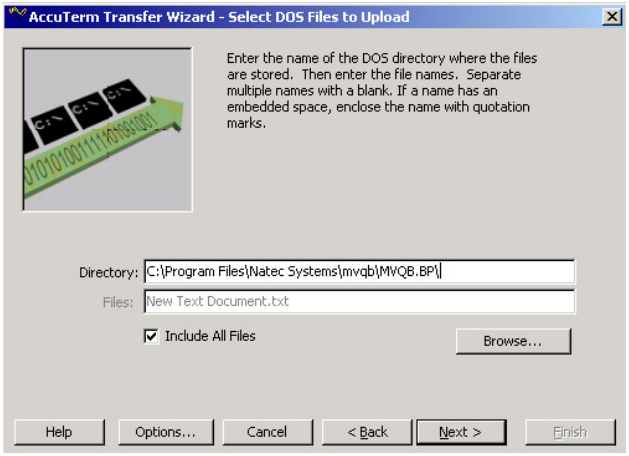
Select 'Next' on the first screen of the wizard.



Make sure "Upload the data to the Host Computer" is selected, and then click "Next"



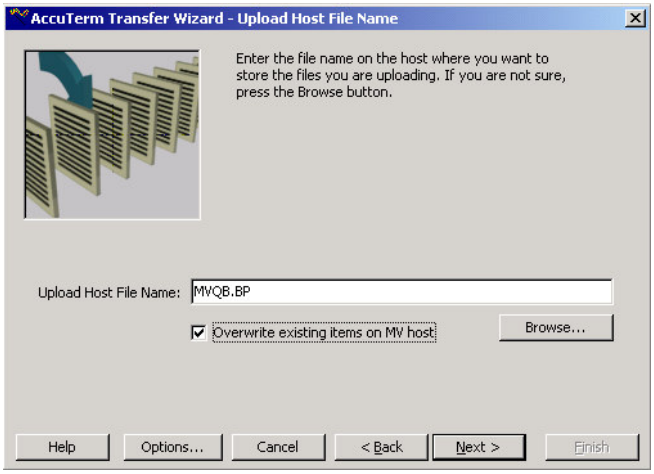
Make sure "The DOS files are Documents" is selected, and click 'Next"



Enter the Directory that contains the MVQB.BP programs. This generally found in the following directory:

```
C:\Program Files\Natec Systems\mvqb\MVQB.BP\
```

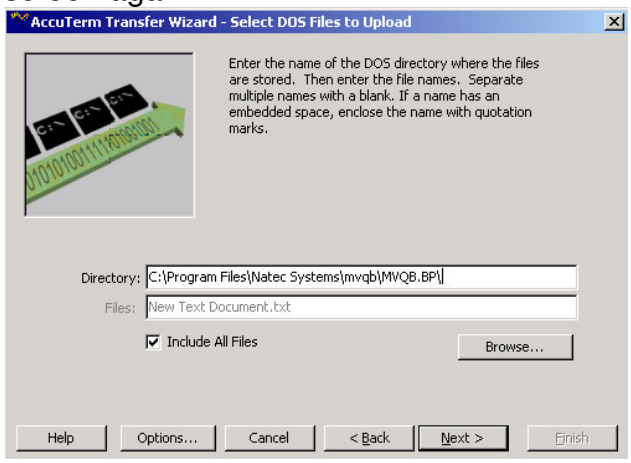
Make sure “Include All files” is checked, and then click ‘Next”



Enter “MVQB.BP” as the file you are placing the files into, and click “Next”

Press the “Run this Job” to transfer the programs into MVQB.BP.

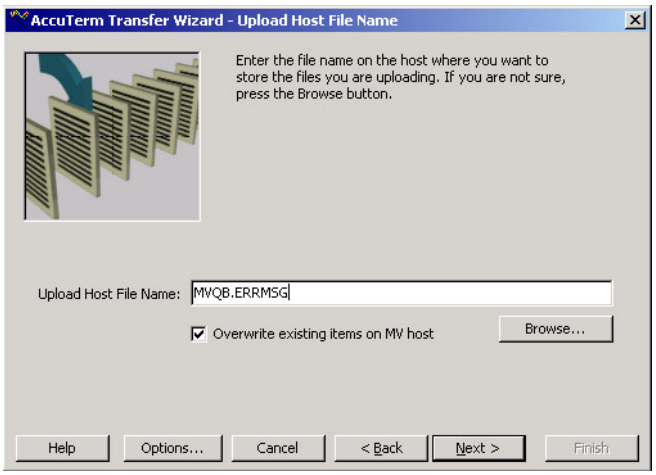
One that is done, you can install the Error Message file. This is optional, but recommended. To do this, press the Back button until you are at the Directory screen again:



Enter the Directory that contains the MVQB.ERRMSGprograms. This generally found in the following directory:

```
C:\Program Files\Natec Systems\mvqb\MVQB.ERRMSG\
```

Make sure “Include All files” is checked, and then click ‘Next”



Enter “MVQB.ERRMSG” as the file you are placing the files into, and click “Next”.

Press the “Run this Job” to transfer the Error Messages into MVQB.ERRMSG.

After Programs Transfers

When you are done transferring the programs to the MultiValue system, Compile and catalog MVQB.INIT in the MVQB.BP file.

Run MVQB.INIT:

```
 1) Setup mvQB - Programs/Communications
 2) Build all Cross-Reference Files
 3) Build Customer xRef
 4) Build Vendor xRef
 5) Build Chart of Accounts xRef
 6) Build Inventory xRef
 7) Build Invoice xRef
 8) Build Journal Entry xRef
 9) Build Purchase Order xRef


                Select Option, or TCL to exit?_
```

Select “Setup mvQB”. It will ask you the database you are using:

```
Choose a Database (1-D3/NT, 2-D3, 3- UniVerse)?
```

Select the Database you have these programs loaded on. It will then ask you questions needed to access the computer that Quickbooks is on, and then compile all the programs for you.

When the Setup is done, then you will see the following.

```
Run MVQB.TEST.CONNECTION to test your changes.  Press RETURN
?
```

Press RETURN to return to the menu, and type TCL to return to TCL. Then run MVQB.TEST.CONNECTION to verify that the Mulitvalue system can connect to the machine that the mvQB server is running on. (Please Note: Make sure the mvQB server is running.)

```
:MVQB.TEST.CONNECTION
Quickbooks Version:  QuickBooks Pro 2004
Quickbooks Database: C:\Program Files\Intuit\QuickBooks\sample_product-based bus
iness.qbw
```

If you get something other than this, then something is not setup correctly.

Customer information

MVQB.CUSTOMER.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.CUSTOMER programs will use.

SUBROUTINE MVQB.CUSTOMER.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

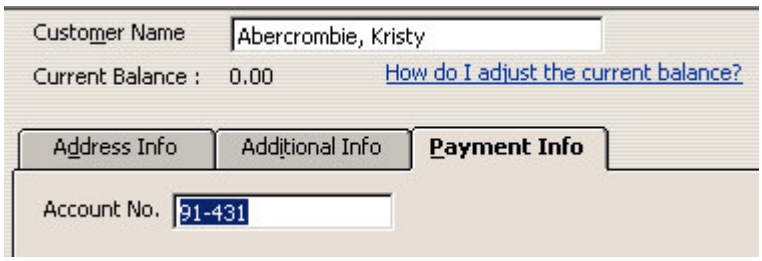
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CUSTOMER.READ

This program is used to retrieve a dynamic array of customer information from QuickBooks.

SUBROUTINE
MVQB.CUSTOMER.READ(REC.NO,CUST.ITEM,XREF.ITEM,ERROR)

REC.NO = This is the Multi-Value Record Id for the customer information you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Account Number Field.



CUST.ITEM = This is the customer information from QuickBooks.
<1> = QuickBooks Name Field
<2> = Company Name
<3> = Salutation
<4> = First Name
<5> = Middle Name
<6> = Last Name
<7> = Billing Address 1

<8> = Billing Address 2
<9> = Billing Address 3
<10> = Billing City
<11> = Billing State
<12> = Billing Zip
<13> = Billing Country
<14> = Shipping Address 1
<15> = Shipping Address 2
<16> = Shipping Address 3
<17> = Shipping City
<18> = Shipping State
<19> = Shipping Zip
<20> = Shipping Country
<21> = Phone
<22> = Alternate Phone
<23> = Resale Tax Number
<24> = Credit Limit
<25> = QuickBooks ListID

XREF.FLG = 0 – Fast Read. Using this option, the customer information is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.CUSTOMER.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CUSTOMER.WRITE

This program is used to write information to QuickBooks. This will take the information from the CUST.ITEM array and update QuickBooks Customer information with it. If the record does not exist, then the program will create a new Customer item.

SUBROUTINE
MVQB.CUSTOMER.WRITE(REC.NO,CUST.ITEM,XREF.FLG,ERROR)

REC.NO = This is the Multi-Value Record Id for the customer information you want to update in QuickBooks. You can find this information in the QuickBooks UI as the Account Number Field.

CUST.ITEM = This is the customer information from QuickBooks. See MVQB.CUSTOMER.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CUSTOMER.AGING
This program is used to return the aging information for a customer.

SUBROUTINE
MVQB.CUSTOMER.AGING(ACCT.NO,AGING.ITEM,CTRL.ITEM,ERROR)

ACCT.NO – The Customer account number to seach for. This is the account number from the MV Database. This account number must exist in Quickbooks. See MVQB.CUSTOMER.READ for more information on Account Numbers in Quickbooks.

CTRL.ITEM = report generation control information.
<1> = 0 - Read from xRef, if not found then rebuild xRef
1 - Read from QB, ignore the XREF information
<2> = 1 - Include Quickbooks Job Names as well as Main Customer. (Default)
0 - Main Customer only.

AGING.ITEM - The Aging information for the customer
<1,n> = InvoiceNo
<2,n> = DueDate
<3,n> = Billed Date
<4,n> = Aging – Actual number of Days the invoice has been outstanding.
<5,n> = Balance Due. The remaining amount due on the invoice.
<6,n> = Invoice Amount. The amount of the invoice.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Vendor information

MVQB.VENDOR.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.VENDOR programs will use.

SUBROUTINE MVQB.VENDOR.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.VENDOR.READ

This program is used to retrieve a dynamic array of vendor information from QuickBooks.

SUBROUTINE
MVQB.VENDOR.READ(REC.NO,VEND.ITEM,XREF.ITEM,ERROR)

REC.NO = This is the Multi-Value Record Id for the vendor information you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Account Number Field.

VEND.ITEM = This is the vendor information from QuickBooks.
<1> = Quickbooks Name
<2> = Company Name
<3> = Salutation
<4> = First Name
<5> = Middle Name
<6> = Last Name
<7> = Address 1

<8> = Address 2
<9> = Address 3
<10> = City
<11> = State
<12> = Zip Code
<13> = Country
<14> = Phone
<15> = Alternate Phone
<16> = Fax
<17> = Email
<18> = Contact
<19> = Alternate Contact
<20> = Name on Check
<21> = Credit Limit
<22> = Is Vendor Eligible for 1099(1-Eligible,0-Not)
<23> = Quickbooks List ID

XREF.FLG = 0 – Fast Read. Using this option, the vendor information is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.VENDOR.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.VENDOR.WRITE

This program is used to write information to QuickBooks. This will take the information from the VEND.ITEM array and update QuickBooks Vendor information with it. If the record does not exist, then the program will create a new Vendor item.

SUBROUTINE
MVQB.VENDOR.WRITE(REC.NO,VEND.ITEM,XREF.FLG,ERROR)

REC.NO = This is the Multi-Value Record Id for the vendor information you want to update in QuickBooks. You can find this information in the QuickBooks UI as the Account Number Field.

VEND.ITEM = This is the vendor information from QuickBooks. See MVQB.VEND.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

GL Account Number information

MVQB.ACCOUNT.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.ACCOUNT programs will use.

SUBROUTINE MVQB.VENDOR.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.ACCOUNT.READ

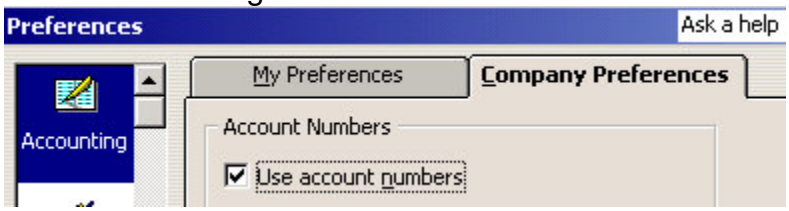
This program is used to retrieve a dynamic array of GL Account Number information from QuickBooks.

SUBROUTINE
MVQB.ACCOUNT.READ(ACCT.NO,ACCT.ITEM,XREF.ITEM,ERROR)

ACCT.NO = This is the Multi-Value Record Id for the GL Account number you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Number Field.

By Default, QuickBooks does not have “Use Account Numbers” turned on. When this option is not turned on, you are unable to see or update the Account number information.

To turn this feature on, go to “Edit | Preferences”, then the company information under “Accounting”.



ACCT.ITEM = This is the Account Number information from QuickBooks.
<1> = QuickBooks ListID
<2> = Account Name
ie: Banking: Wells Fargo
<3> = Description Of this Account
<4> = Bank Account Number, If one
<5> = Account Type
0 - Unknown
1 – Accounts Payable
2 – Accounts Receivable
3 – Bank
4 – Cost Of Goods Sold
5 – Credit Card
6 – Equity
7 – Expense
8 – Fixed Asset
9 – Income
10 – Long Term Liability
11 – Other Asset
12 – Other Current Asset
13 – Other Current Liability
14 – Other Expense
15 – Other Income
16 – Non-Posting

XREF.FLG = 0 – Fast Read. Using this option, the Account Number information is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.ACCOUNT.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.ACCOUNT.WRITE

This program is used to write GL Account numbers to QuickBooks. This will take the information from the ACCT.ITEM array and update QuickBooks Chart of Accounts with it.

If the Account exists, then this program will generate an error. QuickBooks will only allow you to create new GL Account number. It will not allow you to modify existing accounts unless you are using the QuickBooks UI.

SUBROUTINE
MVQB.ACCOUNT.WRITE(ACCT.NO,ACCT.ITEM,XREF.FLG,ERROR)

ACCT.NO = This is the Multi-Value Record Id for the GL Account number you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Number Field.

ACCT.ITEM = This is the GL Account Number information from QuickBooks. See MVQB.ACCOUNT.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Journal Entries

MVQB.JE.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.JE programs will use.

SUBROUTINE MVQB.JE.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

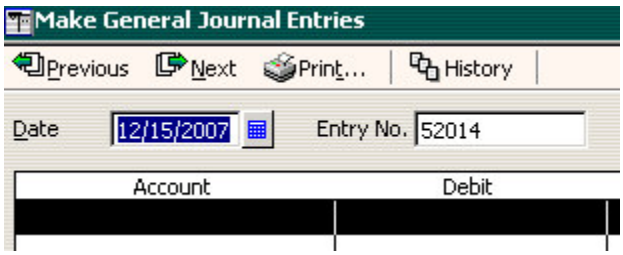
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.JE.READ

This program is used to retrieve a dynamic array of a Journal Entry in QuickBooks.

SUBROUTINE MVQB.JE.READ(REC.NO,JE.ITEM,XREF.ITEM,ERROR)

REC.NO = This is the Multi-Value Record Id for the Journal Entry you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Entry No Field.



JE.ITEM = This is the Journal Entry information from QuickBooks.
<1> = QuickBooks Transaction ID
<2> = Transaction Date
<3> = Is this an Adjustment. (Information only)
<4,d> = Debit Line ID. This is a unique id used to identify this line of information in QuickBooks.
<5,d> = Multi-Value GL Account Number.

This field many contain a value that starts with “QB:”. See Troubleshooting for more information on this error.

<6,d> = Dollar Amount

<7,d> = Memo for this Line

<8,c> = Credit Line ID. This is a unique id used to identify this line of information in QuickBooks

<9,c> = Multi-Value GL Account Number

<10,c> = Dollar Amount

<11,c> = Memo for this line

XREF.FLG = 0 – Fast Read. Using this option, the Journal Entry is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.JE.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.JE.WRITE

This program is used to write Journal Entries to QuickBooks. This will take the information from the JE.ITEM array and update QuickBooks.

If the Journal Entry exists, then this program will generate an error. QuickBooks will only allow you to create new Journal Entries. It will not allow you to modify existing records unless you are using the QuickBooks UI.

SUBROUTINE MVQB.JE.WRITE(REC.NO,JE.ITEM,XREF.FLG,ERROR)

REC.NO = This is the Multi-Value Record Id for the Journal Entry you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Entry No Field.

JE.ITEM = This is the Journal Entry information to update in QuickBooks. See MVQB.JE.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Inventory Items

Inventory Items in QuickBooks come in 4 different types:

Service Items - A service item refers to services that a business charges for or purchases. Examples include specialized labor, consulting hours, and professional fees.

Non-Inventory Items - A non-inventory item is any material or part that a business buys but does not keep on hand as inventory. There are two types of non-inventory items:

1. Materials or parts that are part of the business's overhead (for example, office supplies).

2. Materials or parts that the business buys to finish a specific job and then charges back to the customer.

Other Charge Items - "Other charge" items are miscellaneous charges that do not fall into the categories of service, labor, materials, or parts. Examples include delivery charges, setup fees, and service charges.

Inventory Items - An inventory item is any merchandise or part that a business purchases, tracks as inventory, and then resells.

MVQB.ITEM.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.ITEM programs will use.

SUBROUTINE MVQB.ITEM.BUILD.XREF(CONTROL.ITEM,ERROR)

CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

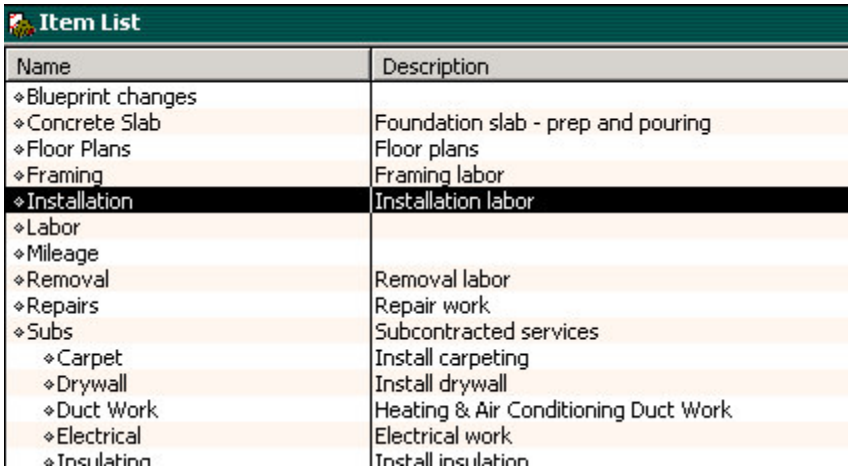
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.ITEM.READ

This program is used to retrieve a dynamic array of an Inventory item in QuickBooks.

SUBROUTINE MVQB.ITEM.READ(REC.NO,INV.ITEM,XREF.ITEM,ERROR)

REC.NO = This is the item Name/Number in QuickBooks. QuickBooks items can be either a name or a number, which is used by QuickBooks Invoice and PO systems.



Unlike other QuickBooks records, Items do not have a separate field for a record number, so the Item name will become the Multi-Value Record number.

INV.ITEM = This is a dynamic array for a QuickBooks Inventory Item.

- <1> = QuickBooks Transaction ID
- <2> = Short Description
- <3> = Is Taxable? (1-yes/0-no)
- <4> = Primary/Sales Description
- <5> = Sale Price
- <6> = Sale Price Percentage
- <7> = Income Account number
- <8> = Purchase Description
- <9> = Purchase Cost
- <10> = Expense/COGS Account
- <11> = Preferred Vendor Number
- <12> = Item type
  - 1- Service Item
  - 2- NonInventory Item
  - 3- Other Charge Item
  - 4- Inventory Item
- <13> = Reimbursable? (1-yes/0-no)
- <15> = Asset Account
- <16> = Reorder Point

XREF.FLG = 0 – Fast Read. Using this option, the Item is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.ITEM.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.ITEM.WRITE

This program is used to write Inventory Items to QuickBooks. This will take the information from the INV.ITEM array and update QuickBooks.

SUBROUTINE MVQB.ITEM.WRITE(REC.NO,INV.ITEM,XREF.FLG,ERROR)

REC.NO = This is the item Name/Number in QuickBooks. QuickBooks items can be either a name or a number, which is used by QuickBooks Invoice and PO systems.

INV.ITEM = This is the Inventory Item information to update in QuickBooks. See MVQB.ITEM.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Invoices

MVQB.INVOICE.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.INVOICE programs will use.

SUBROUTINE MVQB.INVOICE.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

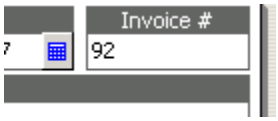
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.INVOICE.READ

This program is used to retrieve a dynamic array of an Invoice in QuickBooks.

SUBROUTINE
MVQB.INVOICE.READ(REC.NO,INVOICE.ITEM,XREF.ITEM,ERROR)

REC.NO = T his is the Multi-Value Record Id for the invoice you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Invoice # Field.



INVOICE.ITEM = This is a dynamic array for a QuickBooks Invoice Item.
<1> = QuickBooks Transaction ID
<2> = Customer Record Number
<3> = Invoice Date
<4> = Bill Address 1
<5> = Bill Address 2
<6> = Bill Address 3
<7> = Bill City
<8> = Bill State
<9> = Bill Zip
<10> = Ship Address 1
<11> = Ship Address 2

<12> = Ship Address 3
<13> = Ship City
<14> = Ship State
<15> = Ship Zip
<16> = Po Number
<17> = Due Date
<18> = Ship Date
<19> = Memo
<20> = Is Pending? (1-yes/0-no)
<21> = Sales Tax Percentage
<22> = Sales Tax Dollar. (read only)
Quickbooks requires you to use its sales tax calculation routines based on specific lines flagged as taxable.
<23> = Applied Payments Amount
<24> = Balance Remaining
<25> = To Be Printed? (1-yes/0-no)
<26,n> = Line ID – New lines, use –1
<27,n> = Item Record Number
<28,n> = Description
<29,n> = Quantity
<30,n> = Rate
<31,n> = Rate Percentage
<32,n> = Total Amount
<33,n> = Service Date
<34,n> = Is Taxable? (1-yes/0-no)
<35> = 'Ship Via' information. This is the name of the Shipping type found in quickbooks. Ie: UPS, Fedx It has to be an exact match to the description or the write will fail

This can be found at: List | Customer & Vendor Profile List | ShipVia List
<36> = Terms. This is the Terms Description found in Quickbooks. It has to be an exact match to the description or the write will fail
This can be found at: List | Customer & Vendor Profile List | Terms LIst

XREF.FLG = 0 – Fast Read. Using this option, the Item is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.INVOICE.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.INVOICE.WRITE

This program is used to write Invoices to QuickBooks. This will take the information from the INVOICE.ITEM array and update QuickBooks.

SUBROUTINE
MVQB.INVOICE.WRITE(REC.NO,INVOICE.ITEM,XREF.FLG,ERROR)

REC.NO = T his is the Multi-Value Record Id for the invoice you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Invoice # Field.
INVOICE.ITEM = This is the invoice information to update in QuickBooks. See MVQB.INVOICE.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Credit Memos

MVQB.CREDITMEMO.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.CREDITMEMO programs will use.

SUBROUTINE MVQB.CREDITMEMO.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

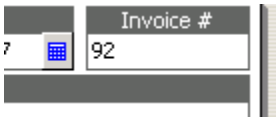
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CREDITMEMO.READ

This program is used to retrieve a dynamic array of an Invoice in QuickBooks.

SUBROUTINE MVQB.CREDITMEMO.READ(REC.NO,CREDITMEMO.ITEM,XREF.ITEM,ERROR)

REC.NO = T his is the Multi-Value Record Id for the invoice you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Invoice # Field.



CREDITMEMO.ITEM = This is a dynamic array for a QuickBooks Invoice Item.

<1> = QuickBooks Transaction ID
<2> = Customer Record Number
<3> = Date
<4> = Bill Address 1
<5> = Bill Address 2
<6> = Bill Address 3
<7> = Bill City
<8> = Bill State
<9> = Bill Zip
<10> = Ship Address 1
<11> = Ship Address 2
<12> = Ship Address 3
<13> = Ship City
<14> = Ship State
<15> = Ship Zip
<16> = Po Number
<17> = Due Date
<18> = Ship Date
<19> = Memo
<20> = Is Pending? (1-yes/0-no)
<21> = Sales Tax Percentage
<22> = Sales Tax Dollar
<23> = Applied Payments Amount
<24> = Balance Remaining

<25> = To Be Printed? (1-yes/0-no)
<26,n> = Line ID – New lines, use –1
<27,n> = Item Record Number
<28,n> = Description
<29,n> = Quantity
<30,n> = Rate
<31,n> = Rate Percentage
<32,n> = Total Amount
<33,n> = Service Date
<34,n> = Is Taxable? (1-yes/0-no)

XREF.FLG = 0 – Fast Read. Using this option, the Item is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.CREDITMEMO.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CREDITMEMO.WRITE

This program is used to write Invoices to QuickBooks. This will take the information from the CREDITMEMO.ITEM array and update QuickBooks.

SUBROUTINE
MVQB.CREDITMEMO.WRITE(REC.NO,CREDITMEMO.ITEM,XREF.FLG,ERRO
R)

REC.NO = T his is the Multi-Value Record Id for the invoice you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Invoice # Field.
CREDITMEMO.ITEM = This is the invoice information to update in QuickBooks. See MVQB.CREDITMEMO.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items

added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Purchase Orders

MVQB.PO.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.PO programs will use.

SUBROUTINE MVQB.PO.BUILD.XREF(CONTROL.ITEM,ERROR)
CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

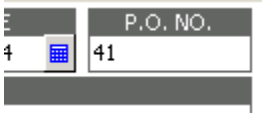
Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.PO.READ

This program is used to retrieve a dynamic array of a Purchase Order in QuickBooks.

SUBROUTINE MVQB.PO.READ(REC.NO,PO.ITEM,XREF.ITEM,ERROR)

REC.NO = T his is the Multi-Value Record Id for the Purchase Order you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the P.O. # Field.



PO.ITEM = This is a dynamic array for a QuickBooks Purchase order Item.
- <1> = QuickBooks Transaction ID
- <2> = Vendor Record Number
- <3> = Ship To Reference Number. This can be either a Vendor Number, or Customer Number
- <4> = Date Entry
- <5> = Vendor Address 1
- <6> = Vendor Address 2
- <7> = Vendor Address 3
- <8> = Vendor City
- <9> = Vendor State
- <10> = Vendor Zip
- <11> = Ship Address 1

<12> = Ship Address 2
<13> = Ship Address 3
<14> = Ship City
<15> = Ship State
<16> = Ship Zip
<17> = Due Date
<18> = Expected Date
<19> = Memo
<20> = Message to be printed on a purchase order for the vendor to read.
<21> = To Be Printed? (1-yes/0-no)
<22,n> = Line ID – New lines, use –1
<23,n> = Item Record Number
<24,n> = Description
<25,n> = Quantity
<26,n> = Rate
<27,n> = Total Amount
<28,n> = Service Date
<29,n> = Quantity Received

XREF.FLG = 0 – Fast Read. Using this option, the Item is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.PO.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.PO.WRITE

This program is used to write Purchase Orders to QuickBooks. This will take the information from the PO.ITEM array and update QuickBooks.

SUBROUTINE MVQB.PO.WRITE(REC.NO,PO.ITEM,XREF.FLG,ERROR)

REC.NO = T his is the Multi-Value Record Id for the Purchase Order you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the P.O. # Field.

PO.ITEM = this is the Purchase Order information to update in QuickBooks. See MVQB.PO.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

BILLS

MVQB.BILL.BUILD.XREF

This program is used to create the cross reference needed to translate between Multi-Value Record Ids and QuickBooks ListIDs. This program will create the file needed to store the information, and will create the needed records that the other MVQB.BILL programs will use.

SUBROUTINE MVQB.BILL.BUILD.XREF(CONTROL.ITEM,ERROR)

CONTROL.ITEM<1> = 0 – Rebuild complete cross-reference information. This will clear the cross-reference file and recreate the information.

1 – Rebuild changes only. Look for changes only to update the cross-reference file with.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.BILL.READ

This program is used to retrieve a dynamic array of a bill in QuickBooks.

SUBROUTINE MVQB.BILL.READ(REC.NO,BILL.ITEM,XREF.ITEM,ERROR)

REC.NO = T his is the Multi-Value Record Id for the Bill you want to retrieve from QuickBooks. You can find this information in the QuickBooks UI as the Ref No Field

BILL.ITEM = This is a dynamic array for a QuickBooks Purchase order Item.

- <1> = QuickBooks Transaction ID
- <2> = Vendor Record Number
- <3> = Ship To Reference Number. This can be either a Vendor Number, or Customer Number
- <4> = Date Entry
- <5> = Vendor Address 1
- <6> = Vendor Address 2
- <7> = Vendor Address 3
- <8> = Vendor City
- <9> = Vendor State
- <10> = Vendor Zip
- <11> = Ship Address 1
- <12> = Ship Address 2
- <13> = Ship Address 3
- <14> = Ship City
- <15> = Ship State
- <16> = Ship Zip

<17> = Due Date
<18> = Expected Date
<19> = Memo
<20> = Message to be printed on a purchase order for the vendor to read.
<21> = To Be Printed? (1-yes/0-no)
<22,n> = Line ID – New lines, use –1
<23,n> = Item Record Number
<24,n> = Description
<25,n> = Quantity
<26,n> = Rate
<27,n> = Total Amount
<28,n> = Service Date
<29,n> = Quantity Received

XREF.FLG = 0 – Fast Read. Using this option, the Item is read from the cross-reference file. If the record is not in the cross-reference file, then the program will read the information from QuickBooks.

This option will be faster when accessing the information, but will only be as current as the last time you ran MVQB.BILL.BUILD.XREF.

1 – Normal Read. This option will always read the information from QuickBooks. This process can be slower, but will always have the most current information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.BILL.WRITE

This program is used to write Bills to QuickBooks. This will take the information from the BILL.ITEM array and update QuickBooks.

SUBROUTINE MVQB.BILL.WRITE(REC.NO,BILL.ITEM,XREF.FLG,ERROR)

REC.NO = This is the Multi-Value Record Id for the Bill you want to write to QuickBooks.

BILL.ITEM = this is the Purchase Order information to update in QuickBooks. See MVQB.BILL.READ for more information.

XREF.FLG = 0 – Fast Write. Use this option if you know that the record you are updating does not exist in QuickBooks. If the program cannot find this information in the cross-reference file, it will send the information to QuickBooks as a new item.

1 – Normal Write. If the item does not exists in the cross-reference file, the write program will check to see if there are any new items added in QuickBooks. This is done to make sure that the information really is a new item and not just out of date cross-reference information.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

QuickBooks Information

There are several program you can use to get information about QuickBooks.

MVQB.QUICKBOOKS.INFO

This program is used to get the QuickBooks Version information.

SUBROUTINE MVQB.QUICKBOOKS.INFO(QUICKBOOKS.INFO,ERROR)

QUICKBOOKS.INFO = this is the information returned about QuickBooks
- <1> = QuickBooks Database Filename and Path
- <2> = Product Name. Ie: QuickBooks Pro 2004
- <3> = Major Version Number
- <4> = Minor Version Number
- <5> = Country
- <6> = mvQB Server License Key
- <7> = mvQB Server Product Id
- <8> = Quickbooks Registration Code/Installer ID
- <9> = mvQB Server Product Code
- <10> = mvQB Server Product Year

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.SET.LICENSE

This program is used to setup the mvQB Server license from your Pick systems.

SUBROUTIEN MVQB.SET.LICENSE(LICENSEINFO,ERROR)

LICENSEINFO<1> = License Key to set in mvQB server. This can also be done from the mvQB server under Help | About

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.CHANGE.DATABASE

This program will allow you set the QuickBooks Database, or change it from one Database to another. This will not change the default QuickBooks Database, just the currently active database.

Please Note: Calling this routine can take up to 1 minute to process depending on the speed of the machine that QuickBooks is loaded on.

SUBROUTINE MVQB.CHANGE.DATABASE(FILENAME,ERROR)

FILENAME = this is the QuickBooks Database file name you want to make active. 'DEFAULT' will change the database back to the Default QuickBooks Database set in the mvQB Server.

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.TEST.CONNECTION

This program is used to test to see if the MultiValue system can communicate with the mvQB Server and Quickbooks.

MVQB.TEST.CONNECTION

MVQB.QUICKBOOKS.OPEN

This program will open the Quickbooks file and leave it open until MVQB.QUICKBOOKS.CLOSE is called. This is differnet from the default action that mvQB server takes. By default, the mvQB server will open the connection when required and leave the connection open for 5 mins unless additional requests are made.

SUBROUTINE MVQB.QUICKBOOKS.OPEN(ERROR)

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

MVQB.QUICKBOOKS.CLOSE

This program will close an open quickbooks connection. Mainly used when MVQB.QUICKBOOKS.OPEN is called.

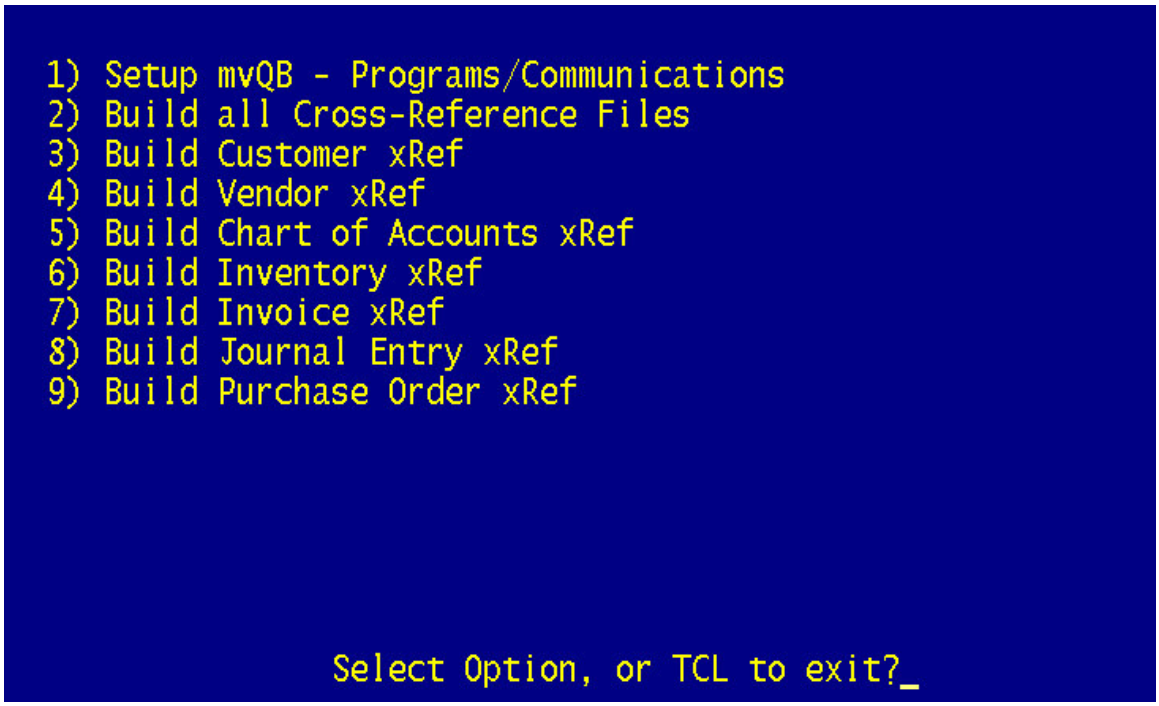
SUBROUTINE MVQB.QUICKBOOKS.CLOSE(ERROR)

Error = 0 –Successful, >0 is an error. See ERROR section for more information.

Initialization of the MVQB routines

MVQB.INIT

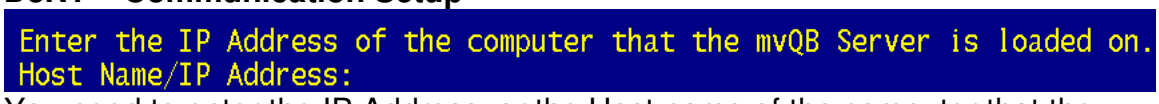
This program is used to initialize the cross-reference files needed for the these programs. This program will create and populate all the cross-reference files needed.



"Setup mvQB" must be done if you have not run it before. It will make sure that the communication routine needed for the MulitValue database to communicate with the mvQB Server are setup correctly.

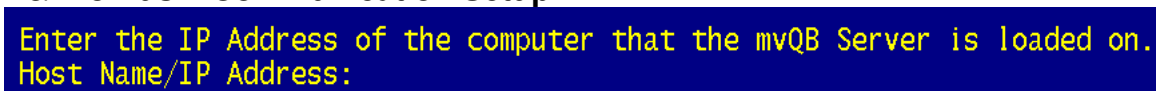
Depending on the MultiValue system installed on, there are a few changes that need to be made to the MVQB subroutines. MVQB.INIT will ask you some questions based on the type of database you select:

D3NT – Communication Setup



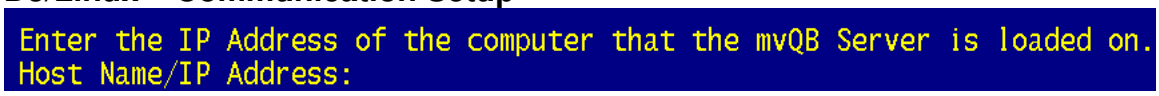
You need to enter the IP Address, or the Host name of the computer that the mvQB server is running on.

D3/ProPlus – Communication Setup



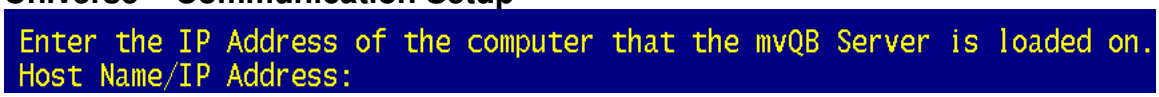
You need to enter the IP Address, or the Host name of the computer that the mvQB server is running on.

D3/Linux – Communication Setup



You need to enter the IP Address, or the Host name of the computer that the mvQB server is running on.

Universe – Communication Setup



You need to enter the IP Address, or the Host name of the computer that the mvQB server is running on.

ERROR variable

This variable is used to tell you of any errors that need to be handled. If ERROR = 0, then the process was successful.

If ERROR > 0, then there was an error of some kind. You can get a full description of the error by calling MVQB.ERROR.MSG.

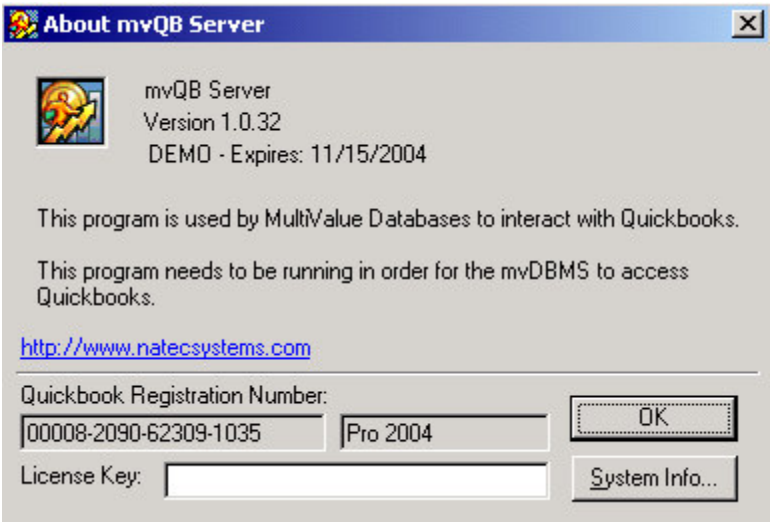
SUBROUTINE MVQB.ERROR.MSG(ERROR,ERRMSG.ITEM)
ERRMSG.ITEM<1> = Short description of the error
ERRMSG.ITEM<2> = Detail of the error and how to fix.

Troubleshooting

Q. I have values that start with "QB:" in my records.
A. When these values exist, it due to the QuickBooks ListID not being found in the cross-reference files. There can be several reasons for this to happen:

1) cross-reference Files are not current. Run the MVQB.BUILD program for the need file.
2) The information in QuickBooks does not have enough information in it to create the cross-reference record. This is generally caused when the field in QuickBooks that is supposed to contain the Multi-Value record number is left blank.

Registration and Licensing



To get to the Licensing screen, click on Help | About. Place your mvQB License Key into the block labeled “License Key”. The program will validate the license key you entered against the QuickBooks Registration Number and the Version of QuickBooks listed on the same screen.

Quickbooks Limitations

Maximum number of Transactions allowed in Quickbooks:

Pro and Premier Editions
QuickBooks can handle a maximum of 2 billion transactions. The maximum number of transactions is limited more by your computers disk space and memory than by QuickBooks.
Each list in QuickBooks has a maximum number of items it can contain, as shown in the following table:

List	Maximum number of items
Chart of accounts	10,000
Items, including inventory items (Group items can contain only 20 individual items.)	14,500
Price levels	20
Job types	10,000
Vendor types	10,000
Customer types	10,000
Payroll items	10,000

Classes	10,000
A/R terms and A/P terms total	10,000
Payment methods	10,000
Shipping methods	10,000
Customer messages	10,000
Memorized reports	14,500
Memorized transactions	14,500
To Do notes	10,000
Total names: employees, customers, vendors, and other names combined **Note:** Any one name list can contain up to 10,000 names, but the name lists combined cannot exceed 14,500 names.	14,500

Do you need more list capacity? Add up to twice as many customers, vendors, and inventory items when you upgrade to QuickBooks Enterprise Solutions, our top-of-the-line QuickBooks product. For more information, call (866) 379-6635 ext. 1005 or visit the QuickBooks Enterprise Solutions Web site (www.usequickbooks.com/qbes5).
Note: Some of the items mentioned in the chart above are not available in earlier versions of QuickBooks.
Note: You can see the total number of customers, items, and names in your company data file by pressing Ctrl+1 in QuickBooks to open the **Product Information** screen.
Practical limitations
The typical QuickBooks customer is a small business with 20 or fewer employees and annual revenue of less than $2 million. The ideal use of QuickBooks is to keep at least 2 years of detailed transactions in a data file so that you can run comparative reports and have prior-year project information.

Enterprise
QuickBooks Enterprise Solutions can handle a maximum of 2 billion transactions. The maximum number of transactions is limited more by your computer's disk space and memory than by QuickBooks.
Each list in QuickBooks Enterprise Solutions has a maximum number of items it can contain, as shown in the following table:

List	Maximum number of items
Chart of accounts	10,000
Items, including inventory items (Group items can contain only 20 individual items.)	29,000

Price levels	20
Job types	10,000
Vendor types	10,000
Customer types	10,000
Payroll items	10,000
Classes	10,000
A/R terms and A/P terms total	10,000
Sales reps	29,000
Sales tax codes	10,000
States	10,000
Payment methods	10,000
Shipping methods	10,000
Customer messages	10,000
Memorized reports	29.000
Memorized transactions	29,000
To Do notes	10,000
Total names: employees, customers, vendors, and other names combined	29,000

Note: Some of the items mentioned in the chart above are not available in earlier versions of QuickBooks.