How to Use Scripting to Require Additional Fields on a Screen

Posted on **March 20, 2015** by **Alnoor Cassim**

Scripting Tips for Sage 100 ERP
by Alnoor Cassim, xkzero Technical Services

When creating a new Customer, Vendor, Sales Order, Purchase Order, Invoice, Receipt, or the like in Sage 100 ERP, a common business process requirement is to ensure specific standard fields or user defined fields (UDFs) are populated.

First of all, in Sage 100 ERP, the software itself has its own built-in requirements. As an example, Sage 100 ERP requires a user to have logged in with a salesperson code to to be able to create a new customer.

Your company may have set up other business requirements, such as this:
Before an order can be placed, your business rules may require the following fields to be populated: Price Level and Email Address.

Also, you may need to temporarily place the customer on credit hold and apply a $1 credit limit until the credit hold is cleared by the finance department.

Or, perhaps you have an imprecise, manual process in place that relies on all users to follow and remember these rules. You may have even defined tasks and security events in Role Maintenance to accomplish a portion of the automation, then thought, “I wish there were a clean way to enforce all these rules!” In fact, there is a way! Once again (like we showed you in the previous blog), ***scripting can save the day!*** The answer to your frustrations could be creating a fairly straightforward script.

What kind of script do you need to create?

Create an *event script* (a.k.a. *User Defined Script*) that runs in Customer Maintenance when the Accept button is pressed.

Note that you are not limited to requiring only the fields mentioned above. You are also not limited to Customer Maintenance. The event type you will create is called *Table Pre-Write*. All of the "Pre" events (Pre-Write, Pre-Validate, Pre-Delete) are typically used to add your own user defined validation and ***prevent an action from occurring***.

How do you create this script?

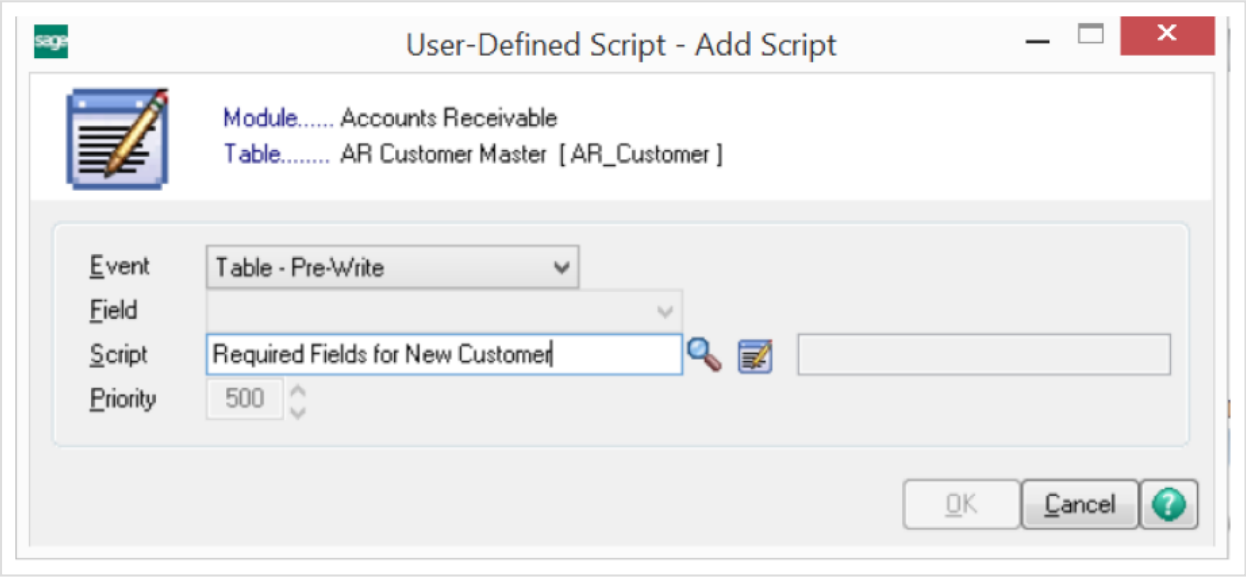
Follow these steps:

1. Go to the Custom Office/Main Menu.
Make these selections:
"User-Defined Field"
"Table Maintenance"

2. Go to the "Accounts Receivable" heading.
Choose: "AR Customer Master."
Right-click and choose: "User Defined Scripts."

3. You are now in the "User Defined Script" window.
Click: "Add"

4. This will open the "User-Defined Script – Add Script" window. (See image below.)
Choose: **Event:**"Table – Pre-Write"
Type a name for your script, such as this: "Required Fields for New Customer"



5. This will bring you to a message indicating that the script file does not exist.
It will be followed by a question, "Do you want to create it? Yes/No?"
Yes, you do want to create the script!
Click: "Yes."

6. Now the "Edit Script" window will appear.
Copy the following code and paste it into your script editor window:

NOTE: **It is important to use the code*exactly*as shown below, including line breaks and spaces.**

```
'Alnoor Cassim - xkzero - Require Additional Fields in Customer Maintenance

'Init VARs
sEmailAddress = "" : sPriceLevel = "" : sCustomerNo = "" : nCreditLimit = 1

'Check if current user is member of the Finance_Dept role.
In_Finance = oSession.AsObject(oSession.Security).IsMember("Finance_Dept")

'Enforce new customer rules for any non-Finance_Dept roles (In_Finance = 0)
'If we wanted to run only for Finance_Dept check for In_Finance > 0

If In_Finance = 0 and oBusObj.EditState = 2 Then

        retVal = oBusObj.GetValue("EmailAddress$", sEmailAddress)
        retVal = oBusObj.GetValue("PriceLevel$", sPriceLevel)
        retVal = oBusObj.GetValue("CustomerNo$", sCustomerNo)

        If sPriceLevel = "" Then

        sMsg =  "The Price Level is blank." & vbCrLf & _
                "Please enter a Price Level for customer " & sCustomerNo
        'retVal = oSession.WriteLog ("M", Replace(sMsg,vbCrLf,CHR(138))'Write to Activity Log
        retVal = oScript.SetError(sMsg) 'Prevent the Accept and show the message
        retVal = oScript.InvokeButton("fldr.pAddl") 'Click the Additional tab folder
        Exit Sub

        End If

        If sEmailAddress = "" Then

        sMsg =  "The Email Address is blank." & vbCrLf & _
                "Please enter an Email Address for customer " & sCustomerNo
        'retVal = oSession.WriteLog ("M", Replace(sMsg,vbCrLf,CHR(138)) 'Write to Activity Log
        retVal = oScript.SetError(sMsg) 'Prevent the Accept and show the message
        retVal = oScript.InvokeButton("fldr.pMain") 'Click the Main tab folder

        Exit Sub

        End If

        'Put the Customer on Credit Hold and set the Credit Limit = $1.
        'This will overwrite whatever was previously entered for these 2 fields.
        retVal = oBusObj.SetValue("CreditHold$", "Y")
        retVal = oBusObj.SetValue("CreditLimit", nCreditLimit)

End If
```

How does this work?

To better understand the logic behind the script's functionality, read this:
"InitVARs"
In the first section, we "initialize" (init) the variables to prepare for use later in the script.

"IF/THEN/ELSE"
Through these stated conditions, we run our main logic. In this particular script, we want the conditions to be as follows:

Check if we're a member of the role called "Finance_Dept."

Check if the customer on the screen is a "new customer."

How do we find out if the current user is a member of "Finance_Dept" role?

Use the special **IsMember()** security function you see in the code block. If the value to the left of the equals sign is more than **0** the current user is a member of that role. Otherwise, they are not a member.

How do we find out if the customer is new or existing?

> **We do this by checking the "edit state" as indicated by the following values:**
>
> 2 = new customer
> 1 = existing customer
> 0 = no customer on the screen

Get the values of the Price Level, Email Address, and Customer No fields.
If either of the first 2 fields are blank, take these steps to prevent the customer from being saved:

1. **Check for a blank price level.**
   Notice the ***If / End If*** block to check for a blank Price Level.
   Execute the **SetError(msg)** function.
   This is a 2-part function that will both prevent the Accept button from being clicked and show a message box to the user with your own user defined message. A good message could be this, "**Oops! The Price Level is blank**."
2. **Auto-click the Additional tab folder.**
   Run the **InvokeButton()** function as a way of auto-clicking the ***Additional*** tab folder.
3. **Exit the script.**
   To immediately exit the script, run the Exit Sub command This is not required for preventing the Accept.
4. **Check for a blank email address.**
   To check for a blank email address, the ***If / End If block*** runs similarly, except the *Main* tab is

auto-clicked.

5. **Write message to the Activity Log.**
   Do you want to write the message to the Activity Log? Then remove the single quote character from the line where you see the **WriteLog()** function.**Next, Put the Customer on Credit Hold and set the Credit Limit = $1**
   The two **SetValue()** lines at the end accomplish these tasks.
   Note: *You should **not** follow this up with a **Write()** command because the Pre-Write script runs before the standard Sage 100 **Write()** command runs, which will write / save the record for us. If we ran on Post-Write event then we would issue a **Write()**.*

You're almost done!

6. **Accept the script.**
   Now you can"Accept" the script.
   This will return you to the "Add Script" window.
7. **Return to the User Defined Scripts window.**
   Close this screen and return to the "User Defined Scripts" window.
   To do so, click: "OK."
8. **Go to the Script Compile window.**
   Next, go back to the main "User Defined Field" and "Table Maintenance" window.
   To do so, click: "Close."Here, you must also click "Close" **again** to see the "Script Compile" window.
   IMPORTANT: When the "Script Compile" window appears, ***click the"Compile" button***.
   You can further edit the script, if necessary. To do so, from the "Custom Office/Main" menu, choose the "User Defined Script Maintenance" task. When done editing the script, to syntax check the script, click "**Check Script,"** then to save the script, click "**Accept."**
9. **Compile. Close.**
   The next step is to click the "**Compile"** button in the lower-left of the screen. When the **Script Compile** window appears, click the **Compile button again** followed by the **Close button**.
   You're done!

This is the second of many scripting examples we will share through the xkzero blog. If you need help with scripting, programming or technical issues with your Sage ERP, no matter how complex, please feel free to contact us:

xkzeroTechnical Services
Email: info@xkzero.com
Call: 847-416-2009

This entry was posted in **Sage 100 ERP**, **xkzero Technical Services** and tagged **Alnoor Cassim**, **scripting** by **Alnoor Cassim**. Bookmark the **permalink [http://erpappsblog.com/?p=1435]** .

About Alnoor Cassim

Director, xkzero Technical Services for Sage 100 ERP. Providing companies solutions to utilize and enhance their ERP software to grow their business, identify and create solutions that streamline business processes, improve productivity, increase profitability, and provide win-win scenarios for all.

View all posts by Alnoor Cassim →