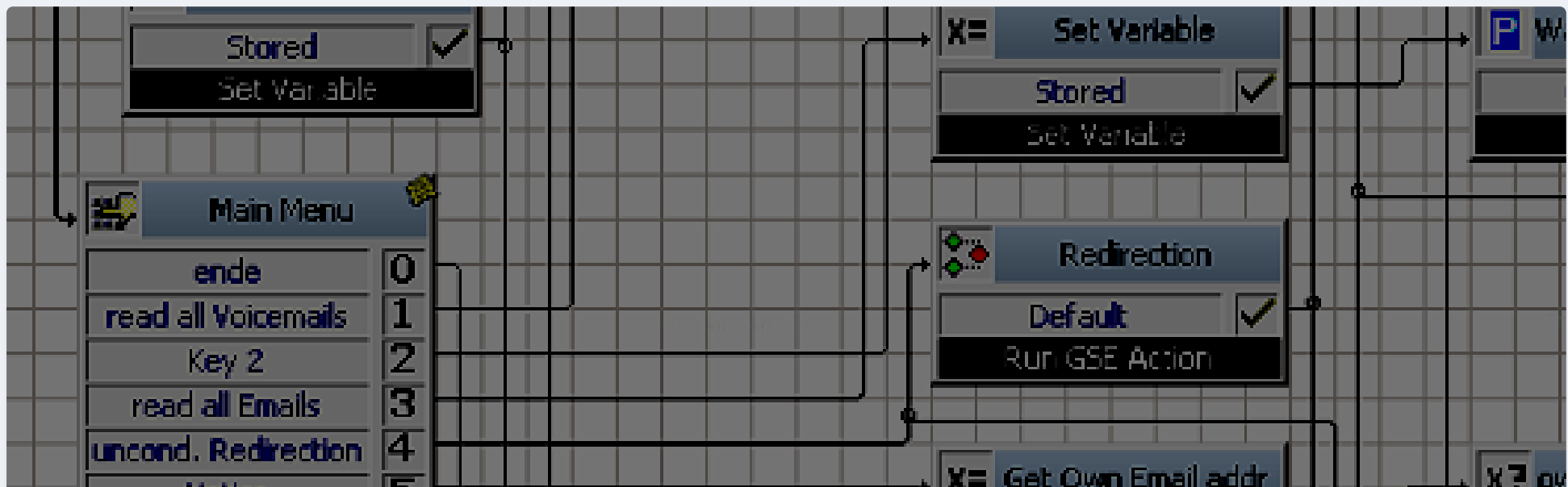


## The Call Routing Guy

A blog by Tom Wellige in General

Followers

1



## #20: How to place your own VBScript/Lua code into separate files and include them into your call routing





Entry posted by Tom Wellige in General September 27, 2024

1,334 views

Share

Followers

0

It is possible to extend functionality of the Graphical Script Editor (GSE) with some own custom VBScript/Lua code.

This is best done by placing the own code into a VBScript/Lua function and place the code of this function into the **Start** block of a GSE rule. Instructions on how this is done in detail and a collection of usefull VBScript/Lua functions can be found [here](#) (for VBScript based call routing) or [here](#) (for Lua based call routing).

There might be situations, where you would prefer (or where it simply is necessary) to place your own code into your own .vbs or .lua file and get this file somehow included into the SwyxWare call routing.

I wouldn't write this blog article if this wouldn't be possible. You just need to follow a few steps.

1. Place your code into your own **.vbs** or **.lua** file

You need to prepare that file a little bit more, but I will come back to that later.

2. Upload this file into either the **USER** scope of your call routing user, or into the **GLOBAL** scope to make it available/accessible for all call routings of all SwyxWare users.

There are a couple of detailed instructions available on how to do that, one of them can be found [here](#) (within the "[Jira Service Integration](#)" Open ECR Extension here on Swyx Forum).

In step 4.1/4.2 you simply have to select your own .vbs/.lua file. Ignore step 5 as you file will only be visible within the "File List" window (step3).

3. **Include** your file into your GSE rule. This is done within the **Start** block of the rule..

VBscript based

```
'#include "MyOwnCodeFile.vbs"
```

Lua based

```
--#include "MyOwnCodeFile.lua"
```

**i** The include statement can be used recursively. So you can use it also within your own code file to include other code files. Which might also include other files. And so on...

#### 4. That's it!

You can call your function now from everywhere within your GSE rule.

If you have multiple **GSE rules** on your script user and need your function in all of these rules, you can do above step 3 in all of the rules. The GSE will make sure that only one of them will be loaded, and not all of them (which would cause serious trouble by having your code defined multiples times in the resulting call routing rule).

For multiple GSE Rules it would also be possible to place the include statement only in one of the rule, to make it available in all rules. But I wouldn't recommend doing so! Over time you will forget in which rule you have included it and might delete/disable that rule later on and let all other rules fail in that moment (as your code will not be included anymore). Just include it into every rule where you need it and let the GSE just load it once.

This is also true for **GSE Actions**. If you need your code in several actions, just include it into every one of them. The already above mentioned "[Jira Service Integration](#)" makes it for example exactly like that as well.

#### Mandatory preparations of your own code file

As already mentioned in above step 1 there is something you need to do with your own code file in order to have the SwyxWare loading it into your call routing.

## 1. Digitally sign your file

The SwyxServer only loads correctly digitally signed .vbs/.lua files into the call routing. So you need to get your own file signed before uploading it into the SwyxWare database (above step 2) and include it.

This can be done with the **SignScript** tool, which as Enreach Partner you are able to download from the Enreach Partner Net following [this link](#). That link also explains the usage of this tool.

**i** Your code file must be **UTF-8 BOM** encoded in order to get it signed correctly. If you use for example the Notepad++ text editor you can easily check and set the file encoding within its "Encoding" program menu.

After signing your code file it is a good idea to verify the signature with the **-v** parameter. This will ensure that you do not miss the above UTF-8 BOM hint.

Also, after signing your code file every modification of it will render its signature invalid. So after every change of your file you need to sign it again.

## 2. Versionize your file

This will become mandatory in the future so it is good practise to start doing it already now.

The **first line** of your code needs to be comment formatted as follows. The given version number (in format nn.nn.n.n) should be the current GSE version you are using to create and test your call routing with.


VBScript based

```
'GSEVersion: 14.10.0.0
```

Lua based

```
--GSEVersion: 14.10.0.0
```

This versioning wasn't necessary in the past, isn't necessary at the time of writing this blog post, but will become necessary/mandatory in future SwyxWare versions.

 For the time being not versionized files will still be accepted by the SwyxServer, but this might change in the future, so better be prepared.

The version number you are giving should include the current major and minor version of your GSE (i.e. installed SwyxWare version). The following build and language numbers can be given as "0", as seen in the above example.

From SwyxWare 14.10 on the SwyxServer will check given version numbers. The **minimum** accepted version number is **10.00.0.0**. Any file with a lesser version number will be refused by the SwyxServer.

After you have signed your own file as described in step 1, you will find the version number in line 2. Don't worry, that's ok.

Enjoy!

PS: don't miss to take a look into the [ECR Useful Link Collection](#).

< Previous entry  
#19: We are many

Next entry >  
#21: The world isn't black & white, or is it?

**0 Comments**

There are no comments to display.

## Create an account or sign in to comment

You need to be a member in order to leave a comment

### Create an account

Sign up for a new account in our community. It's easy!

Register a new account

### Sign in

Already have an account? Sign in here.

Sign In Now

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community