Sending emails in FileMaker with MBS Plugin

Sending emails in FileMaker with MBS Plugin

FileMaker has a built in feature for sending email. While this is easy to use and convenient for a lot of users, some need more: So we have an email sending feature in our MBS Plugin with the following features:

- Send emails with plain text or html
- Several attachments from files, text or containers
- As many TO, CC, BCC, Sender and ReplyTo addresses.
- Subject, HTML and text properly encoded.
- Any SMTP Server
- Custom Port
- Customize SSL options
- Proxy options
- Authentication options

So in order to send an email via MBS Plugin, you need a SMTP server and the credentials. For the server, we need a domain name or an IP, either IPv4 or IPv6. Than we need credentials, typical name and password. Next we need to have an idea about SSL. Your server should require SSL and you need to know if it's SSLv3, TLS 1.0 or best TLS 1.2. You can tell our plugin to use a specific TLS/SSL version and require SSL. Finally you can provide a SSL certificate chain for your server to actually check certificates from server.

To create the email, you use the SendMail functions like the following snippet. You pass HTML text, plain text, subject, from address and smtp details. For our functions we pass email and name separated. The plugin will encode the name properly, so non-ASCII characters survive. Same for subject, plain text and html text.

```
Set Variable [$EmailID; Value:MBS("SendMail.CreateEmail")]
Set Variable [$r; Value:MBS("SendMail.SetFrom"; $EmailID; EMail::FromEmail;
EMail::FromName)]
Set Variable [$r; Value:MBS("SendMail.SetHTMLText"; $EmailID; EMail::HTMLText)]
Set Variable [$r; Value:MBS("SendMail.SetPlainText"; $EmailID; EMail::PlainText)]
Set Variable [$r; Value:MBS("SendMail.SetSubject"; $EmailID; EMail::Subject)]
```

Add the SMTP settings here or later directly on the CURL functions:

```
Set Variable [$r; Value:MBS("SendMail.SetSMTPServer"; $EmailID; EMail::SMTP Server)]
Set Variable [$r; Value:MBS("SendMail.SetSMTPUsername"; $EmailID; EMail::SMTP
Username)]
Set Variable [$r; Value:MBS("SendMail.SetSMTPPassword"; $EmailID; EMail::SMTP
Password)]
```

The next snippet adds addresses. We take them from a separate table and add them depending of the type. We support here TO, CC and BCC recipients. You can also add Sender and ReplyTo addresses. If you need you can also add other email headers like Company Name, Priority or read receipt request.

```
Go to Related Record [Show only related records; From table: “Recipient”; Using layout: 
“Recipient” (Recipient)]
Go to Record/Request/Page [First]
Loop
    If [Recipient::Type = "To"]
        Set Variable [$r; Value:MBS("SendMail.AddTO"; $EmailID; Recipient::Email; 
Recipient::Name)]
    Else If [Recipient::Type = "CC"]
        Set Variable [$r; Value:MBS("SendMail.AddCC"; $EmailID; Recipient::Email; 
Recipient::Name)]
    Else If [Recipient::Type = "BCC"]
        Set Variable [$r; Value:MBS("SendMail.AddBCC"; $EmailID; 
Recipient::Email; Recipient::Name)]
    End If
    Go to Record/Request/Page [Next; Exit after last]
End Loop
Go to Related Record [From table: “EMail”; Using layout: “EMail” (EMail)]
```

From another table we pick attachments. You can pass attachments as containers, files or plain text. This way you are flexible to add attachments as you need. For each attachment you can provide a file name and a mime type. If no mime type is specified, our plugin can guess it from the file extensions.

```
Go to Related Record [Show only related records; From table: “Attachment”; Using layout: 
“Attachment” (Attachment)]
Go to Record/Request/Page [First]
Loop
    If [not IsEmpty(Attachment::Container)]
        Set Variable [$r; Value:MBS("SendMail.AddAttachmentContainer"; $EmailID; 
Attachment::Container; Attachment::Name; Attachment::Type)]
    End If
    If [not IsEmpty(Attachment::Path)]
        Set Variable [$r; Value:MBS("SendMail.AddAttachmentFile"; $EmailID; 
Attachment::Path; Attachment::Name; Attachment::Type)]
    End If
    If [not IsEmpty(Attachment::Text)]
        Set Variable [$r; Value:MBS("SendMail.AddAttachmentText"; $EmailID; 
Attachment::Path; Attachment::Name; Attachment::Type)]
    End If
    Go to Record/Request/Page [Next; Exit after last]
End Loop
Go to Related Record [From table: “EMail”; Using layout: “EMail” (EMail)]
```

If you like to review the email in source code, you can use the GetSource function and display the email in a text field:

```
Set Field [EMail::EmailSource; MBS( "String.ReplaceNewline";
MBS("SendMail.GetSource"; $EmailID); 1)]
```

To send we need a CURL transfer. So you create a new CURL session and pass the email sending settings. Next you can change some CURL settings like SSL settings or switch to a different port. So for this example we do not check SSL certificates, but prefer TLS v1.2.

```
Set Variable [$curl; Value:MBS("CURL.New")]
Set Variable [$r; Value:MBS("SendMail.PrepareCURL"; $EmailID; $curl)]
#Maybe use alternative SMTP port?
//  Set Variable [$r; Value:MBS("CURL.SetOptionPort"; $curl; 587)]
#This turns TLS on and requires connection to be encrypted
Set Variable [$r; Value:MBS("CURL.SetOptionUseSSL"; $curl; 3)]
#force TLS v1.2
Set Variable [$r; Value:MBS("CURL.SetOptionSSLVersion"; $curl; 6)]
#This disables certificate verification, so we accept any:
Set Variable [$r; Value:MBS("CURL.SetOptionSSLVerifyHost"; $curl; 0)]
Set Variable [$r; Value:MBS("CURL.SetOptionSSLVerifyPeer"; $curl; 0)]
#Better with certificates if you have some:
//  Set Variable [$r; Value:MBS( "CURL.SetOptionCAInfo"; $curl; "/Library/FileMaker
Server/certificates.pem")]
//  Set Variable [$r; Value:MBS("CURL.SetOptionSSLVerifyHost"; $curl; 2)]
//  Set Variable [$r; Value:MBS("CURL.SetOptionSSLVerifyPeer"; $curl; 1)]
```

The perform call will actually do the transfer. Our plugin can do transfer synchronously, in background threaded or asynchronously on main thread. This example uses synchronous transfer, so the script waits for the transfer to finish. But you can also run in background or asynchronously and later trigger a script to inform you about the result of the transfer.

```
Set Variable [$r; Value:MBS("CURL.Perform"; $curl)]
```

You can show debug log and input in fields if you like:

```
Set Field [EMail::DebugInput; MBS("CURL.GetInputAsText"; $curl)]
Set Field [EMail::DebugMessages; MBS("CURL.GetDebugAsText"; $curl)]
```

Finally you can do the cleanup:

```
Set Variable [$r; Value:MBS("CURL.Cleanup"; $curl)]
Set Variable [$r; Value:MBS("SendMail.Release"; $EmailID)]
```

If you have trouble with setting this up, please contact us. Our plugin works fine with a lot of smtp servers including google mail and outlook 365.