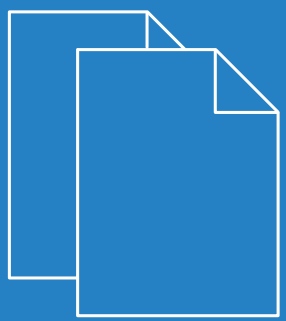
SECURE. NETWORKS.

LANCOM Wireless ePaper Server

Integration IBM Notes



04/2019

LANCOM Systems

Contents

Copyright

While the information in this manual has been compiled with great care, it may not be deemed an assurance of product characteristics. LANCOM Systems shall be liable only to the degree specified in the terms of sale and delivery.

The reproduction and distribution of the documentation and software supplied with this product and the use of its contents is subject to written authorization from LANCOM Systems. We reserve the right to make any alterations that arise as the result of technical development.

Windows® and Microsoft® are registered trademarks of Microsoft, Corp.

LANCOM, LANCOM Systems, LCOS, LANcommunity and Hyper Integration are registered trademarks. All other names or descriptions used may be trademarks or registered trademarks of their owners. This document contains statements relating to future products and their attributes. LANCOM Systems reserves the right to change these without notice. No liability for technical errors and/or omissions.

This product contains separate open-source software components which are subject to their own licenses, in particular the General Public License (GPL). The license information for the device firmware (LCOS) is available on the device's WEBconfig interface under "Extras > License information". If the respective license demands, the source files for the corresponding software components will be made available on a download server upon request.

Products from include software developed by the "OpenSSL Project" for use in the "OpenSSL Toolkit" (*www.openssl.org*).

Products from include cryptographic software written by Eric Young (*eay@cryptsoft.com*).

Products from LANCOM Systems include software developed by the NetBSD Foundation, Inc. and its contributors.

Products from LANCOM Systems contain the LZMA SDK developed by Igor Pavlov.

LANCOM Systems GmbH

Adenauerstr. 20/B2

52146 Wuerselen

Germany

www.lancom-systems.com

1 Objectives and requirements

The instructions listed here are intended exclusively for experts! They also describe systems of a software version and are therefore for illustration purposes only.

1.1 Objectives

The object of this document is to facilitate the successful implementation of the LANCOM Wireless ePaper solutions in IBM Notes. This is achieved with an IBM Notes script library and an XSL templates for the room signage. The code in the script library and the template are examples designed to simplify the task xml:lang="en"of creating your own files. You need to adjust the code to meet with the local conditions in your operating environment.

1.2 Requirements

The successful implementation of the LANCOM Wireless ePaper solutions in an IBM Notes environment depends upon the availability of certain software programs and their settings.

1. Software

   The following software is required:

   - LANCOM Wireless ePaper Server
   - IBM Notes 8.5 or higher
   - IBM Notes Designer 8.53 or higher

2. IBM Notes

   Notes must already be set up to handle the reservation of the resource (meeting rooms). Additionally, the following items must be satisfied by the Notes database for resource reservation:

   - Script library

     A script library as outlined in this document (see chapter 2).

   - Mask

     A mask in the Notes format "profile document". For each meeting room, the mask must contain the constant ROOMNR<Raum> from the script library in the form of a text field. The name needs to be adjusted in the script library accordingly (see the notes on the script library in chapter 2).

   - View

     Each meeting room needs a view that displays current and future room reservations for the current day (ROOMVIEW<Raum> from the script library). In the script library, the name of the view must be adapted for each meeting room (see the notes on the script library in chapter 2).

   - Agent

     An agent that invokes the CheckForUpdate function from the script library. The operating times and update interval should be defined as required (see the notes on the script library in chapter 2).

3. LANCOM Wireless ePaper Server

The LANCOM Wireless ePaper Server needs to meet the following requirements:

- The template must be stored under \\<installation-directory>\data\templates\.
- The address (name or IP) of the Server must be entered into the script library as the constant UPDATEURLEPAPERSERVER (see the notes on the script library in chapter 2).
- Images referenced in the template must be stored under \\<installation-directory>\data\images\.

2 IBM Notes script library

The script library contains the programming code that queries the relevant information from the resource management system, processes it and, if necessary, sends an update to the LANCOM Wireless ePaper Server.

2.1 Example of a script library, IBM notes

The complete code in the script file available for download from the LANCOM website is provided below as an overview. An explanation of the individual sections of code follows. Line numbering is not a part of the code and is for purposes of clarity only.

```
001 %REM
002  Script library for resource management in Notes
003  Created 08.10.2014 by LANCOM Systems GmbH
004 %END REM
005
006 Option Public
007
008 '# Declaration of constants
009 Private Const ROOMIDAACHEN = "D1001BF6"
010 Private Const ROOMNAMEAACHEN = "Aachen B2.100"
011 Private Const ROOMNRAACHEN = "B2100"
012 Private Const ROOMVIEWAACHEN = "CurResForB2100"
013 Private Const UPDATEURLEPAPERSERVER = "http://epaper-server:8001/service/task"
014 Private Const DEBUGEMAIL = "debugging@yourcompany.com"
015 Private Const PROFILEDOC = "yourProfileDoc"
016 Private Const NOMOREMEETINGS = "No more meetings"
017 Private Const TEMPLATE = "lcsconference_landscape.xsl"
018
019
020 '# Create a Unique-ID
021 Function createGuid()
022  Dim object
023  Set object = CreateObject("Scriptlet.TypeLib")
024  guid = object.Guid
025  Set object = Nothing
026
027  createGuid = Left(guid, Len(guid)-2)
028 End Function
029
030
031 '# Handling XML control characters in strings
032 Function EscapeXML(newStr As String) As String
033
034 Dim replaceFrom (1 To 5) As String
035 Dim replaceTo (1 To 5) As String
036 Dim escapedString As String
037
038  replaceFrom(1) = |&|
039  replaceFrom(2) = |"|
040  replaceFrom(3) = |'|
041  replaceFrom(4) = |<|
042  replaceFrom(5) = |>|
043
044  replaceTo(1) = |&amp;|
045  replaceTo(2) = |&quot;|
```

```
046  replaceTo(3) = |&apos;|
047  replaceTo(4) = |&lt;|
048  replaceTo(5) = |&gt;|
049
050  EscapeXML = Replace(newStr,replaceFrom,replaceTo)
051
052 End Function
053
054
055 '# Function for Display updates
056 Function UpdateDisplay(meetingUNID As String, meetingUNID2 As String, displayID As
String, resourceName As String, roomNr As String) As String
057
058  Dim session As New NotesSession
059  Dim db As NotesDatabase
060  Set db = session.CurrentDatabase
061
062  Dim profDoc As NotesDocument
063  Set profDoc = db.Getprofiledocument(PROFILEDOC)
064  Dim mailDoc As NotesDocument
065  Dim rtItem As NotesRichTextItem
066
067  Dim objHttp As Variant
068  Dim reqHttp As String
069  Dim url As String
070  Dim response As String
071
072  Dim meetingDoc1 As NotesDocument
073  Dim meetingDoc2 As NotesDocument
074  Dim meetingChair1 As NotesName
075  Dim meetingChair2 As NotesName
076  Dim roomName As NotesName
077
078  Dim dispDate As String
079  Dim dispRoom As String
080  Dim dispTime1 As String
081  Dim dispPurpose1 As String
082  Dim dispChair1 As String
083  Dim dispTime2 As String
084  Dim dispPurpose2 As String
085  Dim dispChair2 As String
086
087  DebugMode = False
088
089  Set roomName = New NotesName(resourceName)
090  dispRoom = roomName.Common
091  dispDate = Format$(Today,"dd.mm.yyyy")
092
093  '# Requesting the meetings
094  If meetingUNID <> "" Then
095   Set meetingDoc1 = db.Getdocumentbyunid(meetingUNID)
096   Set meetingChair1 = New NotesName(meetingDoc1.Chair(0))
097        dispTime1   =   Format$(meetingDoc1.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc1.EndDateTime(0),"hh:mm")
098   If meetingDoc1.Hasitem("txtPurpose") Then
099    dispPurpose1 = EscapeXML(FullTrim(meetingDoc1.txtPurpose(0)))
100   Else
101    dispPurpose1 = EscapeXML(Trim(meetingDoc1.Purpose(0)))
102   End If
103
104   dispChair1 = meetingChair1.Common
105
106   If meetingUNID2 <> "" Then
107    Set meetingDoc2 = db.Getdocumentbyunid(meetingUNID2)
```

```
108    Set meetingChair2 = New NotesName(meetingDoc2.Chair(0))
109          dispTime2   =   Format$(meetingDoc2.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc2.EndDateTime(0),"hh:mm")
110    If meetingDoc2.Hasitem("txtPurpose") Then
111     dispPurpose2 = EscapeXML(FullTrim(meetingDoc2.txtPurpose(0)))
112    Else
113     dispPurpose2 = EscapeXML(Trim(meetingDoc2.Purpose(0)))
114    End If
115
116    dispChair2 = meetingChair2.Common
117   End If
118  Else
119   dispPurpose1 = NOMOREMEETINGS
120  End If
121
122  Set mailDoc = db.CreateDocument
123  mailDoc.Form = "Memo"
124  mailDoc.Subject = "Report UpdateDisplay for "+ resourceName +" in Ressource.nsf vom
" + Format$(Now)
125  mailDoc.SendTo = DEBUGEMAIL
126  Set rtItem = mailDoc.CreateRichTextItem("Body")
127
128  '# XML-Erstellung
129  reqHttp = |<?xml version="1.0" encoding="UTF-8" standalone="yes"?>|+_
130      |<TaskOrder title="Refresh label |+displayID +| for room |+dispRoom+|">|+_
131         |<TemplateTask    xml:lang="en"labelId="|+displayID  +|"  externalId="4711"
template=|+TEMPLATE|>|+_
132      |<room roomName="|+dispRoom+|">|+_
133       |<field key="date" value="|+dispDate+|"/>|+_
134       |<field key="time1" value="|+dispTime1 +|"/>|+_
135       |<field key="purpose1" value="|+dispPurpose1 +|"/>|+_
136       |<field key="chair1" value="|+dispChair1 +|"/>|+_
137       |<field key="time2" value="|+dispTime2 +|"/>|+_
138       |<field key="purpose2" value="|+dispPurpose2 +|"/>|+_
139       |<field key="chair2" value="|+dispChair2 +|"/>|+_
140      |</room>|+_
141     |</TemplateTask>|+_
142      |</TaskOrder>|
143
144
145  Dim oldVal As Variant
146  oldval = profDoc.Getitemvalue(roomNr)
147
148  '# Check if update is necessary.
149  If StrComp(reqHttp,oldVal(0)) Then
150
151   Call profdoc.Replaceitemvalue(roomNr, Nothing)
152   Call profDoc.Replaceitemvalue(roomNr, reqHttp)
153   Call profDoc.Save(True, False)
154
155   Set objHttp = CreateObject("Microsoft.XMLHTTP")
156   url = UPDATEURLEPAPERSERVER
157   objHttp.Open "POST", url, False, "", ""
158   objHttp.setRequestHeader "Content-Type:", "application/xml"
159   objHttp.setRequestHeader "Content-Lenght:", Len(reqHttp)
160   objHttp.setRequestHeader "User-Agent:", "LCS_NTS_AGNT"
161   objHttp.Send(reqHttp)
162
163   If (objHttp.readyState <> 4) Or (objHttp.status <> 200) Then
164    Call rtItem.AppendText ("The request could not be executed."+Chr(13)+Chr(10)+_
165    "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
166    "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
167    "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
168    "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
```

```
169    "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
170    "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
171    Call mailDoc.Send(False)
172   Else
173    Call rtItem.AppendText ("Works fine! "+Chr(13)+Chr(10)+_
174    "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
175    "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
176    "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
177    "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
178    "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
179    "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
180    Call mailDoc.Send(False)
181   End If
182
183   response = objHttp.responseText
184   Set objHttp=Nothing
185
186  ElseIf DebugMode Then
187   Call rtItem.AppendText ("Same request. No update. "+Chr(13)+Chr(10) +"Request: " +
reqHttp)
188   Call mailDoc.Send(False)
189   'MsgBox("same as last")
190  End If
191
192  UpdateDisplay = response
193
194 End Function
195
196
197 '# Function to trigger the update process
198 Sub CheckForUpdate
199
200  Dim session As New NotesSession
201  Dim db As NotesDatabase
202  Set db = session.Currentdatabase
203
204  Dim meetingView As NotesView
205
206  Dim response As String
207  Dim meetingUNID1 As String
208  Dim meetingUNID2 As String
209
210
211  '# Example for room AACHEN, required for each room
212  Set meetingView = db.Getview(ROOMVIEWAACHEN)
213
214  If meetingView.Allentries.Count > 1 Then
215   meetingUNID1 = meetingView.Getfirstdocument().Universalid
216   meetingUNID2 = meetingView.Getnthdocument(2).Universalid
217  ElseIf meetingView.Allentries.Count = 1 Then
218   meetingUNID1 = meetingView.Getfirstdocument().Universalid
219   meetingUNID2 = ""
220  Else
221   meetingUNID1 = ""
222   meetingUNID2 = ""
223  End If
224
225   response = UpdateDisplay(meetingUNID1, meetingUNID2,ROOMIDAACHEN, ROOMNAMEAACHEN,
ROOMNRAACHEN)
226
227 End Sub
```

2.2 Explanations of individual sections of code

This chapter provides an explanation of the individual sections of code to help you to adapt your script library.

Declaration of constants

This section of code is used to declare the constants. This makes it easier to change their values and offers a clear overview.

```
008 '# Declaration of constants
009 Private Const ROOMIDAACHEN = "D1001BF6"
010 Private Const ROOMNAMEAACHEN = "Aachen B2.100"
011 Private Const ROOMNRAACHEN = "B2100"
012 Private Const ROOMVIEWAACHEN = "CurResForB2100"
013 Private Const UPDATEURLEPAPERSERVER = "http://epaper-server:8001/service/task"
014 Private Const DEBUGEMAIL = "debugging@yourcompany.com"
015 Private Const PROFILEDOC = "yourProfileDoc"
016 Private Const NOMOREMEETINGS = "No more meetings"
017 Private Const TEMPLATE = "lcsconference_landscape.xsl"
```

```
Private Const ROOMIDAACHEN
```

The ID of the Wireless ePaper Display.

```
Private Const ROOMNAMEAACHEN
```

The designation of the Wireless ePaper Display.

```
Die Private Const ROOMNRAACHEN
```

ID of the room. This is used in the profile document to designate the text box where the last transmitted data set is to be stored.

```
Private Const ROOMVIEWAACHEN
```

The name of the Notes view that displays the current and future room reservations.

```
Private Const UPDATEURLEPAPERSERVER
```

The update URL used to contact the LANCOM Wireless ePaper Server. Notation: "http://<name or IP>:<port>/service/task"

```
Private Const DEBUGEMAIL
```

An e-mail address used for debugging (updates performed, etc.).

```
Private Const PROFILEDOC
```

The name of the profile document that stores the last transmitted data records for all of the rooms.

```
Private Const NOMOREMEETINGS
```

The text to be rendered on the display if there are no further reservations for the day.

```
Private Const TEMPLATE
```

The file name of the template used to show the reservations on the Wireless ePaper Displays.

Function: Create a unique ID

```
020 '# Create a Unique-ID
021 Function createGuid()
022  Dim object
023  Set object = CreateObject("Scriptlet.TypeLib")
024  guid = object.Guid
025  Set object = Nothing
026
027  createGuid = Left(guid, Len(guid)-2)
028 End Function
```

This function generates a unique ID that is required in the course of the processing.

Function: Handling XML control characters in strings

```
031 '# Handling XML control characters in strings
032 Function EscapeXML(newStr As String) As String
033
034 Dim replaceFrom (1 To 5) As String
035 Dim replaceTo (1 To 5) As String
036 Dim escapedString As String
037
038  replaceFrom(1) = |&|
039  replaceFrom(2) = |"|
040  replaceFrom(3) = |'|
041  replaceFrom(4) = |<|
042  replaceFrom(5) = |>|
043
044  replaceTo(1) = |&amp;|
045  replaceTo(2) = |&quot;|
046  replaceTo(3) = |&apos;|
047  replaceTo(4) = |&lt;|
048  replaceTo(5) = |&gt;|
049
050  EscapeXML = Replace(newStr,replaceFrom,replaceTo)
051
052 End Function
```

This function ensures that any XML control characters contained in the text strings for the Display (e.g., the topic xml:lang="en"of the reservation) are rendered properly and do not cause problems with the XML.

Function: UpdateDisplay

Due to its scope, this function is shown here in its entirety and explained below.

```
055 '# Function for Display updates
056 Function UpdateDisplay(meetingUNID As String, meetingUNID2 As String, displayID As
String, resourceName As String, roomNr As String) As String
057
058  Dim session As New NotesSession
059  Dim db As NotesDatabase
060  Set db = session.CurrentDatabase
061
062  Dim profDoc As NotesDocument
063  Set profDoc = db.Getprofiledocument(PROFILEDOC)
064  Dim mailDoc As NotesDocument
065  Dim rtItem As NotesRichTextItem
066
067  Dim objHttp As Variant
068  Dim reqHttp As String
069  Dim url As String
070  Dim response As String
071
072  Dim meetingDoc1 As NotesDocument
073  Dim meetingDoc2 As NotesDocument
074  Dim meetingChair1 As NotesName
075  Dim meetingChair2 As NotesName
076  Dim roomName As NotesName
077
078  Dim dispDate As String
079  Dim dispRoom As String
080  Dim dispTime1 As String
081  Dim dispPurpose1 As String
082  Dim dispChair1 As String
083  Dim dispTime2 As String
```

```
084  Dim dispPurpose2 As String
085  Dim dispChair2 As String
086
087  DebugMode = False
088
089  Set roomName = New NotesName(resourceName)
090  dispRoom = roomName.Common
091  dispDate = Format$(Today,"dd.mm.yyyy")
092
093  '# Request the meetings
094  If meetingUNID <> "" Then
095   Set meetingDoc1 = db.Getdocumentbyunid(meetingUNID)
096   Set meetingChair1 = New NotesName(meetingDoc1.Chair(0))
097        dispTime1   =   Format$(meetingDoc1.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc1.EndDateTime(0),"hh:mm")
098   If meetingDoc1.Hasitem("txtPurpose") Then
099    dispPurpose1 = EscapeXML(FullTrim(meetingDoc1.txtPurpose(0)))
100   Else
101    dispPurpose1 = EscapeXML(Trim(meetingDoc1.Purpose(0)))
102   End If
103
104   dispChair1 = meetingChair1.Common
105
106   If meetingUNID2 <> "" Then
107    Set meetingDoc2 = db.Getdocumentbyunid(meetingUNID2)
108    Set meetingChair2 = New NotesName(meetingDoc2.Chair(0))
109          dispTime2   =   Format$(meetingDoc2.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc2.EndDateTime(0),"hh:mm")
110    If meetingDoc2.Hasitem("txtPurpose") Then
111     dispPurpose2 = EscapeXML(FullTrim(meetingDoc2.txtPurpose(0)))
112    Else
113     dispPurpose2 = EscapeXML(Trim(meetingDoc2.Purpose(0)))
114    End If
115
116    dispChair2 = meetingChair2.Common
117   End If
118  Else
119   dispPurpose1 = NOMOREMEETINGS
120  End If
121
122  Set mailDoc = db.CreateDocument
123  mailDoc.Form = "Memo"
124  mailDoc.Subject = "Report UpdateDisplay for "+ resourceName +" in Ressource.nsf vom
" + Format$(Now)
125  mailDoc.SendTo = DEBUGEMAIL
126  Set rtItem = mailDoc.CreateRichTextItem("Body")
127
128  '# XML creation
129  reqHttp = |<?xml version="1.0" encoding="UTF-8" standalone="yes"?>|+_
130      |<TaskOrder title="Refresh label |+displayID +| for room |+dispRoom+|">|+_
131         |<TemplateTask    xml:lang="en"labelId="|+displayID  +|"  externalId="4711"
template=|+TEMPLATE|>|+_
132      |<room roomName="|+dispRoom+|">|+_
133       |<field key="date" value="|+dispDate+|"/>|+_
134       |<field key="time1" value="|+dispTime1 +|"/>|+_
135       |<field key="purpose1" value="|+dispPurpose1 +|"/>|+_
136       |<field key="chair1" value="|+dispChair1 +|"/>|+_
137       |<field key="time2" value="|+dispTime2 +|"/>|+_
138       |<field key="purpose2" value="|+dispPurpose2 +|"/>|+_
139       |<field key="chair2" value="|+dispChair2 +|"/>|+_
140      |</room>|+_
141     |</TemplateTask>|+_
142      |</TaskOrder>|
143
```

```
144
145  Dim oldVal As Variant
146  oldval = profDoc.Getitemvalue(roomNr)
147
148  '# Check if update is necessary.
149  If StrComp(reqHttp,oldVal(0)) Then
150
151   Call profdoc.Replaceitemvalue(roomNr, Nothing)
152   Call profDoc.Replaceitemvalue(roomNr, reqHttp)
153   Call profDoc.Save(True, False)
154
155   Set objHttp = CreateObject("Microsoft.XMLHTTP")
156   url = UPDATEURLEPAPERSERVER
157   objHttp.Open "POST", url, False, "", ""
158   objHttp.setRequestHeader "Content type:", "application/xml"
159   objHttp.setRequestHeader "Content-Length:", Len(reqHttp)
160   objHttp.setRequestHeader "User agent:", "LCS_NTS_AGNT"
161   objHttp.Send(reqHttp)
162
163   If (objHttp.readyState <> 4) Or (objHttp.status <> 200) Then
164    Call rtItem.AppendText ("The request did not execute."+Chr(13)+Chr(10)+_
165    "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
166    "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
167    "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
168    "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
169    "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
170    "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
171    Call mailDoc.Send(False)
172   Else
173    Call rtItem.AppendText ("works fine! "+Chr(13)+Chr(10)+_
174    "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
175    "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
176    "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
177    "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
178    "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
179    "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
180    Call mailDoc.Send(False)
181   End If
182
183   response = objHttp.responseText
184   Set objHttp=Nothing
185
186  ElseIf DebugMode Then
187   Call rtItem.AppendText ("Same request. No update. "+Chr(13)+Chr(10) +"Request: " +
reqHttp)
188   Call mailDoc.Send(False)
189   'MsgBox("same as last")
190  End If
191
192  UpdateDisplay = response
193
194 End Function
```

Individual code sections of the function: UpdateDisplay

Declaration of variables

```
058  Dim session As New NotesSession
059  Dim db As NotesDatabase
060  Set db = session.CurrentDatabase
061
062  Dim profDoc As NotesDocument
063  Set profDoc = db.Getprofiledocument(PROFILEDOC)
```

```
064  Dim mailDoc As NotesDocument
065  Dim rtItem As NotesRichTextItem
066
067  Dim objHttp As Variant
068  Dim reqHttp As String
069  Dim url As String
070  Dim response As String
071
072  Dim meetingDoc1 As NotesDocument
073  Dim meetingDoc2 As NotesDocument
074  Dim meetingChair1 As NotesName
075  Dim meetingChair2 As NotesName
076  Dim roomName As NotesName
077
078  Dim dispDate As String
079  Dim dispRoom As String
080  Dim dispTime1 As String
081  Dim dispPurpose1 As String
082  Dim dispChair1 As String
083  Dim dispTime2 As String
084  Dim dispPurpose2 As String
085  Dim dispChair2 As String
```

This section of the code declares the different variables used by the function. This includes the variables used to store the information about the individual meetings, along with the profile document and various other necessary variables.

Querying the reservations on the basis of UNIDs

```
093  '# Request the meetings
094  If meetingUNID <> "" Then
095   Set meetingDoc1 = db.Getdocumentbyunid(meetingUNID)
096   Set meetingChair1 = New NotesName(meetingDoc1.Chair(0))
097        dispTime1   =   Format$(meetingDoc1.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc1.EndDateTime(0),"hh:mm")
098   If meetingDoc1.Hasitem("txtPurpose") Then
099    dispPurpose1 = EscapeXML(FullTrim(meetingDoc1.txtPurpose(0)))
100   Else
101    dispPurpose1 = EscapeXML(Trim(meetingDoc1.Purpose(0)))
102   End If
103
104   dispChair1 = meetingChair1.Common
105
106   If meetingUNID2 <> "" Then
107    Set meetingDoc2 = db.Getdocumentbyunid(meetingUNID2)
108    Set meetingChair2 = New NotesName(meetingDoc2.Chair(0))
109          dispTime2   =   Format$(meetingDoc2.StartDateTime(0),"hh:mm")+"   -   "   +
Format$(meetingDoc2.EndDateTime(0),"hh:mm")
110    If meetingDoc2.Hasitem("txtPurpose") Then
111     dispPurpose2 = EscapeXML(FullTrim(meetingDoc2.txtPurpose(0)))
112    Else
113     dispPurpose2 = EscapeXML(Trim(meetingDoc2.Purpose(0)))
114    End If
115
116    dispChair2 = meetingChair2.Common
117   End If
118  Else
119   dispPurpose1 = NOMOREMEETINGS
120  End If
```

In this section of code, the details about the reservations are retrieved from the Notes server and the variables are set for the current and next reservation. If there is no current or future reservation, then the text for display is taken from the constant NOMOREMEETINGS. The text entered for the topic xml:lang="en"is processed by the `EscapeXML` function to ensure that the variables are free of XML control characters.

E-mail definition

```
122  Set mailDoc = db.CreateDocument
123  mailDoc.Form = "Memo"
124  mailDoc.Subject = "Report UpdateDisplay for "+ resourceName +" in Ressource.nsf from
 " + Format$(Now)
125  mailDoc.SendTo = DEBUGEMAIL
126  Set rtItem = mailDoc.CreateRichTextItem("Body")
```

This section of code defines the structure of an e-mail that can be created and sent when debugging or updating.

XML creation

```
128 '# XML creation
129 reqHttp = |<?xml version="1.0" encoding="UTF-8" standalone="yes"?>|+_
130     |<TaskOrder title="Refresh label |+displayID +| for room |+dispRoom+|">|+_
131        |<TemplateTask    xml:lang="en"labelId="|+displayID  +|"  externalId="4711"
template=|+TEMPLATE|>|+_
132     |<room roomName="|+dispRoom+|">|+_
133      |<field key="date" value="|+dispDate+|"/>|+_
134      |<field key="time1" value="|+dispTime1 +|"/>|+_
135      |<field key="purpose1" value="|+dispPurpose1 +|"/>|+_
136      |<field key="chair1" value="|+dispChair1 +|"/>|+_
137      |<field key="time2" value="|+dispTime2 +|"/>|+_
138      |<field key="purpose2" value="|+dispPurpose2 +|"/>|+_
139      |<field key="chair2" value="|+dispChair2 +|"/>|+_
140     |</room>|+_
141    |</TemplateTask>|+_
142     |</TaskOrder>|
```

The HTTP request containing the information for the fields in the template is created here for transmission to the Wireless ePaper Server. The value for `externalId`, in this example `4711`, can be chosen freely. This is required by the Wireless ePaper Server to identify the source of the update and to record it to a log.

Comparison of the current and last records

```
145 Dim oldVal As Variant
146 oldval = profDoc.Getitemvalue(roomNr)
147
148 '# Check to see if update is necessary.
149 If StrComp(reqHttp,oldVal(0)) Then
150
151  Call profdoc.Replaceitemvalue(roomNr, Nothing)
152  Call profDoc.Replaceitemvalue(roomNr, reqHttp)
153  Call profDoc.Save(True, False)
154
155  Set objHttp = CreateObject("Microsoft.XMLHTTP")
156  url = UPDATEURLEPAPERSERVER
157  objHttp.Open "POST", url, False, "", ""
158  objHttp.setRequestHeader "Content type:", "application/xml"
159  objHttp.setRequestHeader "Content length:", Len(reqHttp)
160  objHttp.setRequestHeader "User agent:", "LCS_NTS_AGNT"
161  objHttp.Send(reqHttp)
162
163  If (objHttp.readyState <> 4) Or (objHttp.status <> 200) Then
164   Call rtItem.AppendText ("The request did not execute."+Chr(13)+Chr(10)+_
165   "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
166   "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
167   "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
168   "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
169   "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
170   "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
171   Call mailDoc.Send(False)
172  Else
173   Call rtItem.AppendText ("works fine! "+Chr(13)+Chr(10)+_
```

```
174   "for Meeting UNID 1: " +meetingUNID +Chr(13)+Chr(10)+_
175   "for Meeting UNID 2: " +meetingUNID2 +Chr(13)+Chr(10)+_
176   "readyState: " +Format$(objHttp.readyState)+Chr(13)+Chr(10)+_
177   "HTTP Status: " +Format$(objHttp.status)+Chr(13)+Chr(10)+_
178   "Request: " +Chr(13)+Chr(10)+ reqHttp +Chr(13)+Chr(10)+Chr(13)+Chr(10)+_
179   "Response: " +Chr(13)+Chr(10)+ objHttp.responseText)
180   Call mailDoc.Send(False)
181  End If
182
183  response = objHttp.responseText
184  Set objHttp=Nothing
185
186 ElseIf DebugMode Then
187  Call rtItem.AppendText ("Same request. No update. "+Chr(13)+Chr(10) +"Request: " +
reqHttp)
188  Call mailDoc.Send(False)
189  'MsgBox("same as last")
190 End If
191
192 UpdateDisplay = response
```

To ensure that only updates with new information are sent to the Wireless ePaper Server and thus to the Displays, a comparison is made between the last sent data record and the new one. If there are differences, the update will be sent to the server; otherwise the record will be discarded. If an update is sent, the response from the server is taken as the return value for the function. This can be useful for monitoring and troubleshooting.

Function: Starting the script library update process

```
197 '# Function to trigger the update process
198 Sub CheckForUpdate
199
200  Dim session As New NotesSession
201  Dim db As NotesDatabase
202  Set db = session.Currentdatabase
203
204  Dim meetingView As NotesView
205
206  Dim response As String
207  Dim meetingUNID1 As String
208  Dim meetingUNID2 As String
209
210
211  '# Example for room AACHEN, required for each room
212  Set meetingView = db.Getview(ROOMVIEWAACHEN)
213
214  If meetingView.Allentries.Count > 1 Then
215   meetingUNID1 = meetingView.Getfirstdocument().Universalid
216   meetingUNID2 = meetingView.Getnthdocument(2).Universalid
217  ElseIf meetingView.Allentries.Count = 1 Then
218   meetingUNID1 = meetingView.Getfirstdocument().Universalid
219   meetingUNID2 = ""
220  Else
221   meetingUNID1 = ""
222   meetingUNID2 = ""
223  End If
224
225  response = UpdateDisplay(meetingUNID1, meetingUNID2,ROOMIDAACHEN, ROOMNAMEAACHEN,
ROOMNRAACHEN)
226
227 End Sub
```

This function is invoked by the Notes agent, which specifies the operating times and intervals for the checks on whether the displays need updating.

If several meeting rooms (Displays) need updating, then this function needs to be supplemented with the following code for each meeting room. The constants ROOMVIEWAACHEN, ROOMIDAACHEN, ROOMNAMEAACHEN and ROOMNRAACHEN all need to be supplemented with the constants corresponding to the additional room. These constants also need to be declared at the beginning of the script library.

```
211  '# Example for room AACHEN, required for each room
212  Set meetingView = db.Getview(ROOMVIEWAACHEN)
213
214  If meetingView.Allentries.Count > 1 Then
215   meetingUNID1 = meetingView.Getfirstdocument().Universalid
216   meetingUNID2 = meetingView.Getnthdocument(2).Universalid
217  ElseIf meetingView.Allentries.Count = 1 Then
218   meetingUNID1 = meetingView.Getfirstdocument().Universalid
219   meetingUNID2 = ""
220  Else
221   meetingUNID1 = ""
222   meetingUNID2 = ""
223  End If
224
225   response = UpdateDisplay(meetingUNID1, meetingUNID2,ROOMIDAACHEN, ROOMNAMEAACHEN,
ROOMNRAACHEN)
```

3 Template for room signage

The template is stored on the Wireless ePaper Server as an XSL file. It specifies the format in which the Wireless ePaper Server renders the updates sent by the script library. The resulting image is then delivered to the corresponding Wireless ePaper Displays.

3.1 Example of a template

This XSL template is based on a potential meeting-room signage concept xml:lang="en"for the LANCOM Systems GmbH. Line numbering is not a part of the code and is for purposes of clarity only.

```
001 <?xml version="1.0" encoding="UTF-8"?>
002 <!—The template is provided for 7.4" Displays mounted in the landscape orientation.-->
003 <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
004
005 <xsl:template match="Record">
006
007  <!-- Rendering information for 7.4” Displays -->
008  <image height="480" width="800" rotation="90" font-family="Verdana">
009
010   <!-- Room -->
011   <field height="108" width="780" x="10" y="20">
012    <text align="center" font-size="40" font-weight="bold">
013     <utils method="toUpperCase">
014      <xsl:value-of select="room/@roomName"/>
015     </utils>
016    </text>
017
018     <!-- Date -->
019    <text align="center" font-size="35" font-weight="bold" padding-top="10">
020     <xsl:value-of select="room/field[@key='date']/@value"/>
021    </text>
022   </field>
023
024   <line thickness="2" x-from="0" x-to="800" y-from="130" y-to="130"/>
025
026   <!-- Time1 -->
027   <field height="50" width="780" x="20" y="150">
028    <text align="left" font-weight="bold" font-size="40">
029     <xsl:value-of select="room/field[@key='time1']/@value"/>
030    </text>
031   </field>
032
033   <!-- Purpose1 -->
034   <field height="50" width="780" x="20" y="200">
035    <text align="left" font-size="40" condense="1, 0.8, 0.6, 0.5">
036     <xsl:value-of select="room/field[@key='purpose1']/@value"/>
037    </text>
038   </field>
039
040   <!-- Chair1 -->
041   <field height="40" width="770" x="20" y="250">
042    <text align="left" font-weight="bold" font-size="30">
043     <xsl:value-of select="room/field[@key='chair1']/@value"/>
044    </text>
045   </field>
```

```
046
047    <!-- Time2 -->
048   <field height="35" width="780" x="20" y="320">
049    <text align="left" font-size="28">
050     <xsl:value-of select="room/field[@key='time2']/@value"/>
051    </text>
052   </field>
053
054   <!-- Purpose2 -->
055   <field height="35" width="780" x="20" y="355">
056    <text align="left" font-size="28" condense="1, 0.8, 0.6, 0.5">
057     <xsl:value-of select="room/field[@key='purpose2']/@value"/>
058    </text>
059   </field>
060
061    <!-- Chair2 -->
062   <field height="30" width="770" x="20" y="390">
063    <text align="left" font-size="20">
064     <xsl:value-of select="room/field[@key='chair2']/@value"/>
065    </text>
066   </field>
067
068   <!-- LANCOM Logo -->
069   <field align="right" height="60" width="780" x="10" y="410">
070    <img src="../images/LANCOM_Logo_200.png"></img>
071   </field>
072  </image>
073 </xsl:template>
074 </xsl:stylesheet>
```

3.2 Explanations of individual sections of code

This chapter explains the various sections of code in the example XSL template.

Display information

```
007   <!-- Rendering information for 7.4” Displays -->
008   <image height="480" width="800" rotation="90" font-family="Verdana">
```

This section specifies which Wireless ePaper Display the following code applies to. In this case, it is a 7.4" Display with a resolution of 800 x 480 pixels.

`height` specifies the height of the image in pixels.

`width` specifies the width of the image in pixels.

`rotation` specifies the rotation of the image in degrees. 0 corresponds to vertically mounted Displays and 90 to horizontally mounted Displays.

`font-family` specifies the font to be used.

Definition of text fields

```
010   <!-- Room -->
011   <field height="108" width="780" x="10" y="20">
012    <text align="center" font-size="40" font-weight="bold">
013     <utils method="toUpperCase">
014      <xsl:value-of select="room/@roomName"/>
015     </utils>
016    </text>
017
```

```
018    <!-- Date -->
019    <text align="center" font-size="35" font-weight="bold" padding-top="10">
020     <xsl:value-of select="room/field[@key='date']/@value"/>
021    </text>
022   </field>
```

This area specifies the position of the room name and date.

First, a field is defined that specifies the height, width and position on the Display:

```
<field height="108" width="780" x="10" y="20">.
```

`height` specifies the height of the field in pixels.

`width` specifies the width of the field in pixels.

`x` specifies the distance from the left edge of the Display in pixels.

`y` specifies the distance from the top edge of the Display in pixels.

The layout of the various texts is defined in this field:

```
<text align="center" font-size="40" font-weight="bold">
```

`align` specifies how the text is justified (e.g., center, left, right).

`font-size` specifies the font size.

`font-weight` specifies the font style (e.g., bold, italic).

```
<utils method="toUpperCase">
```

Invoking this method forces the text to appear in the uppercase.

```
<xsl:value-of select="room/@roomName"/>
```

Here, the XML information provided by the script library is accessed and the text is inserted at the appropriate position. So far, the definitions have only set out the layout.

The next step uses the same principle to specify the text that shows the transmitted date. Additional fields contain the layout for the rest of the information that relates to the individual meetings.

Integration of graphics

```
068    <!-- LANCOM Logo -->
069    <field align="right" height="60" width="780" x="10" y="410">
070     <img src="../images/LANCOM_Logo_200.png"></img>
071    </field>
```

In addition to text fields, images can be integrated too. Similar to the above, a field is specified that sets the position of the image.

```
<img src="../images/LANCOM_Logo_200.png" ></img>
```

The storage location and the name of the image file is specified here.

3.3 Interaction between template, XML information and Wireless ePaper Displays

This chapter uses an example to describe the interaction between the template, XML information and the actual representation on the Wireless ePaper Display.

Step 1:

When an update is triggered by the Notes agent, the Wireless ePaper Server receives XML information with the following content: (Line numbering is not a part of the code and is for purposes of clarity only.)

```
001 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
002 <TaskOrder title="Refresh D1001BF6 for Aachen B2.100">
003   <TemplateTask  xml:lang="en"labelId="D1001BF6" externalId="4711"
        template="lcsconference_landscape.xsl">
004     <room roomName="Aachen B2.100">
005       <field key="date" value="16.09.2014"/>
006       <field key="time1" value="10:00 - 11:30"/>
007       <field key="purpose1" value="Marketing Project Coordination"/>
008       <field key="chair1" value="Alec Coad"/>
009       <field key="time2" value="11:30 - 13.00"/>
010       <field key="purpose2" value="Team Meeting Controlling"/>
011       <field key="chair2" value="Emily Kirby"/>
012     </room>
013   </TemplateTask>
014 </TaskOrder>
```

Step 2:

The Wireless ePaper Server processes the received information line by line. This sets the relevant Display and the template required. Based on the template, the Server now creates the data set used to render the image. The layout is set according to the template and then the information from the XML is inserted.

Two content elements are described here for illustrative purposes: The chairperson of the next meeting and the company logo.

The chairperson for the subsequent meeting is formatted according to the template as follows:

```
061   <!-- Chair2 -->
062   <field height="30" width="770" x="20" y="390">
063    <text align="left" font-size="20">
064     <xsl:value-of select="room/field[@key='chair2']/@value"/>
065    </text>
066   </field>
```

The information necessary for the representation can be seen in the XML below:

```
011    <field key="chair2" value="Emily Kirby"/>
```

Access to the image specified in the template does not require access to the XML information. The path is evaluated instead.

```
068    <!-- LANCOM Logo -->
069    <field align="right" height="60" width="780" x="10" y="410">
070     <img src="../images/LANCOM_Logo_200.png"></img>
071    </field>
```

Step 3:

The image is rendered and sent to the Wireless ePaper Display. The Display appears as shown below. The chairperson of the subsequent meeting **(1)** and the image **(2)** are highlighted in color.