Grandstream Networks, Inc.

XML Based Idle Screen Customization Guide

GXP16xx IP Phone

GXP16xx XML Idle Screen Customization Guide

Index

Table of Figures

GXP16xx XML Idle Screen Customization Guide

Table of Tables

GXP16xx XML Idle Screen Customization Guide

INTRODUCTION

The Grandstream GXP16xx supports XML based idle screen customization. This XML API allows users to customize the idle screen layout as well as the logo, text or system variables to be displayed. The design of the display and layout depends highly upon personal preferences and requirements.

This document specifies the Grandstream XML Customizable Screen API design that will be used on GXP16xx.

WHAT IS XML

XML (eXtensible Markup Language) is a markup language* for documents and applications containing structured information. This information contains both content (text, pictures, input box and etc.) and an indication of what role that content plays (e.g. content in a section header is different from content in a footnote, or content in a figure caption, or content in a database table, and etc.). Almost all documents have certain kind of structure.

*Note: A markup language is a mechanism to identify structures in a document. The XML specification defines a standard way to add markup to documents.

WHY XML

What benefits does XML provide to SIP endpoints? XML enables our SIP phones to serve as output devices where the phones could interact with external applications in a flexible and programmable manner. Two specific XML APIs supported by GXP16xx are XML Custom Screen and XML Phonebook.

XML API ARCHITECTURE

The XML idle screen customization API on GXP16xx could use HTTP/HTTPS or TFTP as the transport protocol. The following figure shows how it works via HTTP as an example. Basically, GXP16xx initiates the HTTP GET Request to the HTTP server and waits for the response. Once the phone receives the response with XML content in BODY, it displays the information.

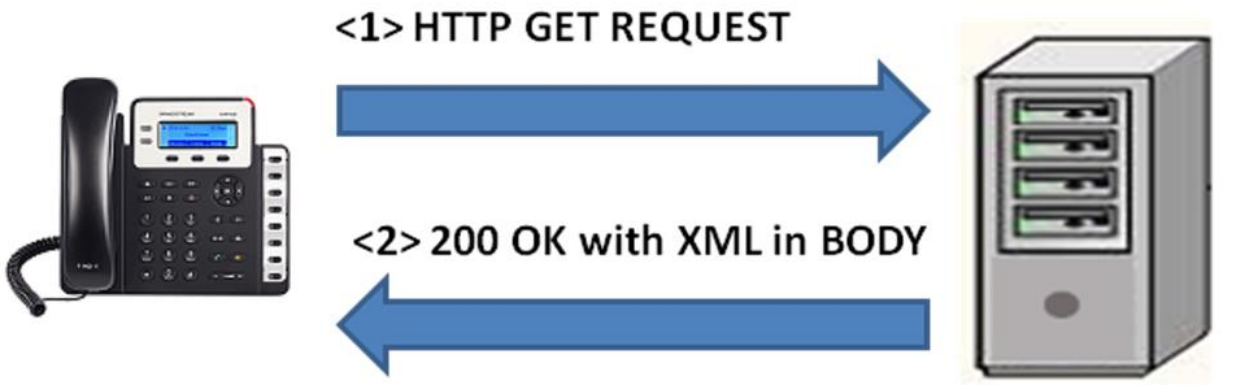

Figure 1: GXP16xx XML API via HTTP

Two types of XML API architectures are introduced below, depending on whether the transaction is within a LAN or accessed via the Internet.

1. A transaction in LAN area may exchange information in the following manner. GXP16xx sends request and accepts XML contents via HTTP/HTTPS/TFTP, directly communicating with the HTTP/HTTPS/TFTP Server. The Server will then handle the request and response via any protocols with the other application server to get the expected information for the XML idle screen display. The following figure shows downloading XML idle screen via HTTP within LAN.

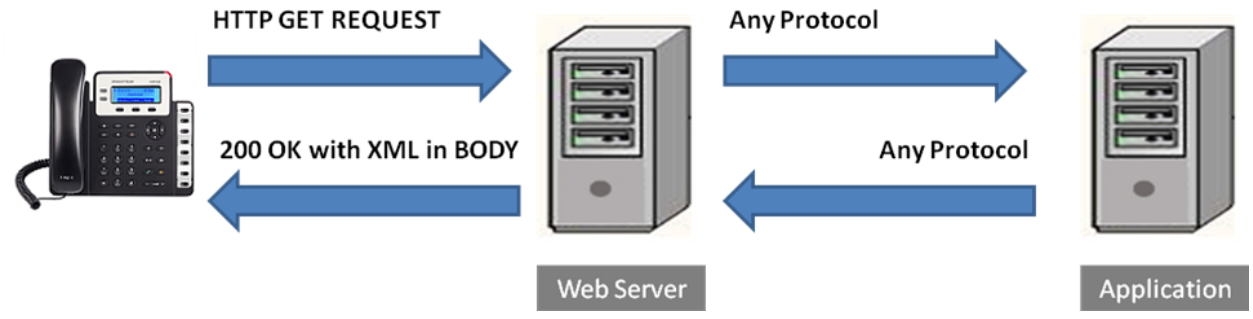

Figure 2: GXP16xx XML API Structure - 1

2. If the above Web Server accesses Internet, it could interact with outside web server and respond real-time content to GXP16xx.

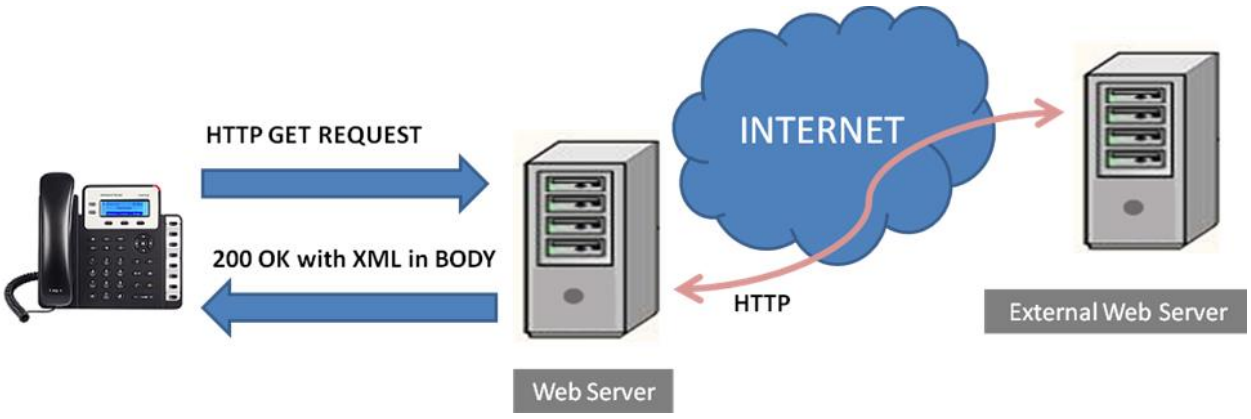

Figure 3: GXP16xx XML API Structure - 2

As illustrated above, all the logic lies within the server side of the architecture. The GXP16xx phone receives XML document and displays it accordingly.

XML IDLE SCREEN DOWNLOADING CONFIGURATION

FIRMWARE

Before the XML idle screen customization is used on GXP16xx, please make sure the firmware on the phone is upgraded to the latest version. Please refer to the following link for firmware upgrading information:

http://www.grandstream.com/support/firmware

WEB GUI CONFIGURATION

To download the XML idle screen file to GXP16xx, firstly enable the Idle Screen XML Download and configure the Server Path under Web GUI->Settings->XML Applications page.

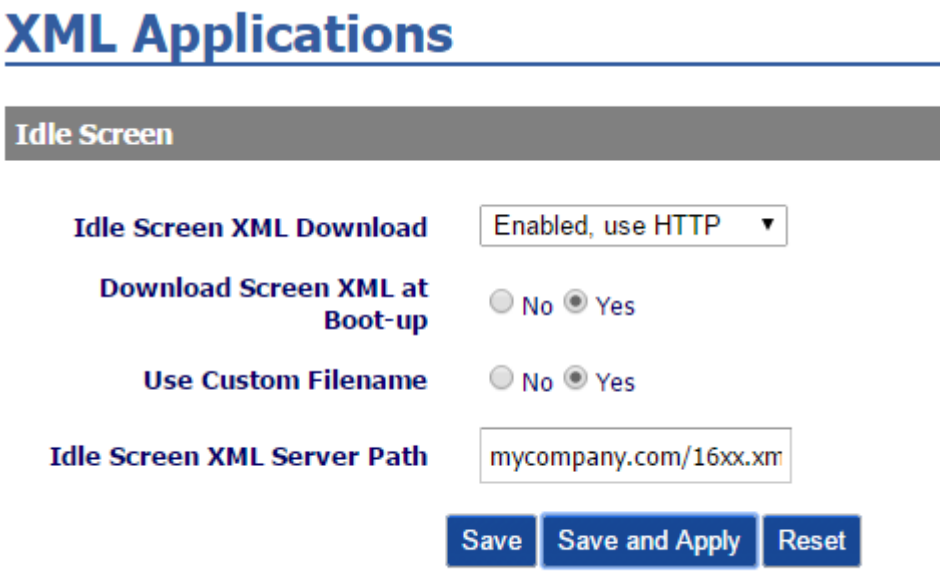

Figure 4: Web GUI Configuration

- **Idle Screen XML Download**
  The idle screen XML file could be downloaded via HTTP, HTTPS or TFTP. By default it is "Disabled".

- **Download Screen XML at Boot-up**
  If it's set to "Yes", when the phone boots up, it will send out request to download the XML idle screen file automatically.

If it's set to "No", users would need go to LCD MENU->Preference->Download SCR XML to download the idle screen manually. Users could also erase the current downloaded idle screen by pressing MENU->Preference->Erase Custom SCR. By default, this option is set to "No".

- **Use Custom Filename**
  If "Use custom filename" is set to "No", the XML file name must be **idle_screen.xml**. In this case, users only need to specify the folder path in "Idle Screen XML Server Path" where the idle_screen.xml is located (For example, 192.168.40.10/XMLscreendir).

  If "Use custom filename" is set to "Yes", users could name the file as preferred. In this case, the "Idle Screen XML Server Path" has to be specified to the name of the XML file (For example, 192.168.40.10/XMLscreendir/welcome.xml).

- **Idle Screen XML Server Path**
  This specifies the path where the phone is going to download the XML file. Since downloading method is selected as "HTTP", "TFTP", or "HTTPS" already when enabling the XML idle screen download, users shall not specify the protocol in this path anymore. The accepted format are as follows:

  IP_address[:port]/dir
  IP_address[:port]/dir/filename
  Hostname[:port]/dir
  Hostname[:port]/dir/filename

  Examples:

  192.168.40.10/XMLscreendir
  192.168.40.10/XMLscreendir/welcome.xml
  192.168.40.10:443/XMLscreendir
  192.168.40.10:443/XMLscreendir/welcome.xml
  mycompany.com/gs_screen_dir
  mycompany.com:8080/idlescreen
  service.mycompany.com/XML/gxp16xxidle.xml
  service.mycompany.com:8080/XML/gxp16xxidle.xml

Note:

- If "[:port]" is not specified, port 80 will be used as default for HTTP; port 443 will be used as default for HTTPS; port 69 will be used as default for TFTP;
- If "Use custom filename" is set to "No", the Server Path does not necessarily need to contain the destination file name. Users only need specify the directory path where the file is located.

After the above configuration, click on "Save and Apply" in the web GUI page. The phone will apply and display the XML idle screen upon pressing MENU->Preference->Download XML SCR in phone's LCD or once the phone restarted if the option "Download Screen XML at Boot-up" is selected.

Users may also use the XML configuration file to provision the phone with the above XML idle screen downloading options. In this case, GXP16xx needs to be rebooted and provisioned. The corresponding P values are as below.

- **P340**: Idle Screen XML Download. Possible values: 0 (Disabled) / 1 (HTTP) / 2 (TFTP) / 3 (HTTPS), other values are ignored;
- **P1349**: Download Screen XML at Boot-up. Possible values: 0 (No) / 1 (Yes), other values are ignored;
- **P1343**: Use custom filename. Possible values: 0 (No) / 1 (Yes), other values are ignored;
- **P341**: Idle Screen XML Server Path. This is a string up to 128 characters.

Note:

- As you may know, it is also possible to enter the idle screen file server path into a web browser. In this way you'll be able to see the exact XML document on your PC that your phone will be receiving;
- The downloaded XML idle screen file can be displayed but won't be saved internally on the phone. It is recommended to save the XML idle screen file externally on your PC or server for your reference and future use.

GXP16xx IDLE SCREEN

OVERVIEW

Without XML idle screen customization or other configurations, a GXP16xx's default idle screen is like below. Press the **NextScr** softkey will toggle among the following different idle screens (vary among models):

- Default idle screen when the phone boots up
- Weather Information
- IP Address and Extension Number

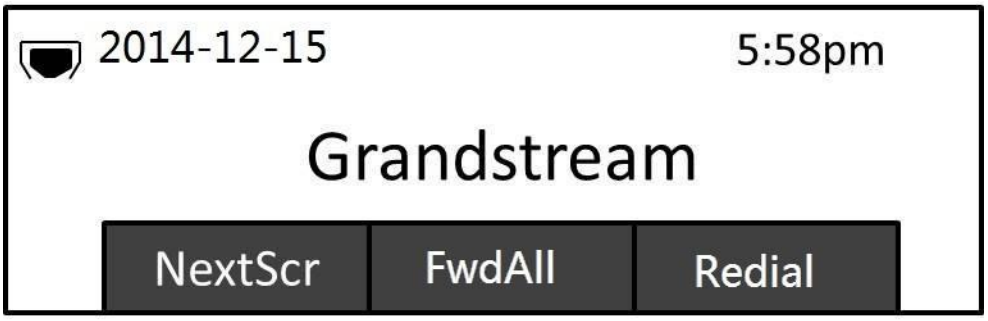

Figure 5: GXP16xx Default Idle Screen

The idle screens for all models are specified in the table blow. Although all the supported idle screens could be modified via the XML idle screen file, normally the idle screen 1 (default idle screen) is the only one users would need to customize for logo display, text display and etc.

Table 1: GXP16xx Idle Screens Overview

Model	Idle Screen 1	Idle Screen 2	Idle Screen 3
GXP1610	Default	Weather	IP Address and Extension Number
GXP1620	Default	Weather	IP Address and Extension Number
GXP1625	Default	Weather	IP Address and Extension Number
GXP1628	Default	Weather	IP Address and Extension Number

XML DOCUMENT FORMAT

HEADER

In the first line of the XML document, the following header can be set as XML declaration. It defines the XML version and encoding. On GXP16xx, UTF-8 is used as encoding method for correct display.

```
<?xml version="1.0" encoding="UTF-8"?>
```

SPECIAL CHARACTERS

As followed by the standard XML recommendation, some characters need to be escaped. The following table lists the characters with their escape sequence.

Table 2: Special Characters in XML Document

Characters	Name	Escape Sequence
&	Ampersand	&amp
“	Quote	&quot
‘	Apostrophe	&apos
<	Left Angle Bracket	<
>	Right Angle Bracket	>

GXP16xx SCREEN XML STRUCTURE

The main structure of GXP16xx idle screen template could be represented as below. This provides users an overview of the XML element and screen layout. For more details on element attribute and text information, please refer to section [GXP16xx IDLE SCREEN XML ELEMENT].

GXP16xx SCREEN XML MAIN STRUCTURE

```
<Screen>

    <SoftkeyBar>
            <!-- ON THE BOTTOM OF SCREEN. FOR SOFTKEY DISPLAY-->
    </SoftkeyBar>


    <IdleScreen>
            <!--MAIN IDLE SCREEN DISPLAY INFORMATION (LOGO, TEXT) HERE-->
    </IdleScreen>


    <IdleScreen>
                    <!-- DISPLAY WEATHER SCREEN-->
    </IdleScreen>


    <IdleScreen>
                <!-- DISPLAY IP ADDRESS AND EXTENSIONS->
    </IdleScreen>


</Screen>
```

GXP16xx IDLE SCREEN XML ELEMENT

This section describes details of the XML element used in GXP16xx XML idle screen customization. Please note that the element name is case-sensitive when being used in XML document.

ROOT ELEMENT <Screen>

<Screen> is the root element of the XML document. This element is mandatory.

```
<Screen>
All the information for screen display is here
</Screen>
```

<Screen> ELEMENT DETAILS

The following tables shows child element and attribute for <Screen> element.

Table 3: <Screen> Element

Object	Position	Type	Values	Comments
Screen	**Root element**	**Mandatory**	**-**	**Root element of the XML document**
LeftStatusBar	Child element	Optional	-	Defines account bar display
SoftkeyBar	Child element	Mandatory	-	Defines softkey bar display
IdleScreen	Child element	Mandatory	-	Main customization area. Normally, only the 1st <IdleScreen> element needs customization for logo, text or variable display. Users usually could leave the other IdleScreen elements as default.

<SoftkeyBar> ELEMENT

The <SoftkeyBar> section defines how the softkey layout is, e.g, softkey bar height, position and button shape. This element is mandatory and it's recommended to keep it as default. The following figure shows the <SoftkeyBar> area for GXP16xx.

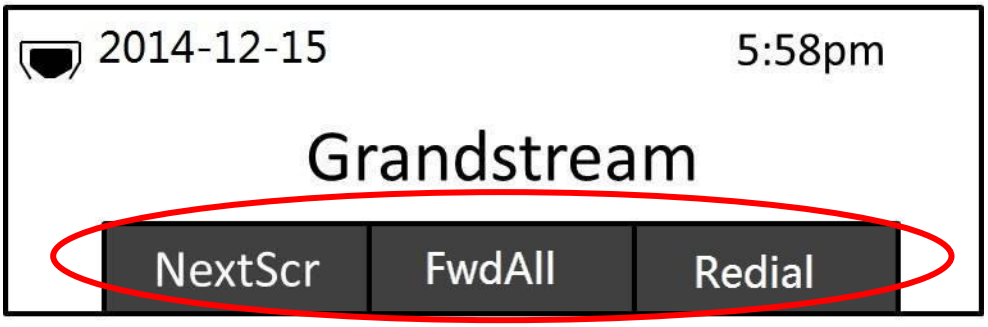

Figure 6: GXP16xx SoftkeyBar Area

```
<SoftkeyBar>
    <Layout height="Height of the softkeyBar" >
        <DisplayBitmap>
            Image information here
        </DisplayBitmap>
        <DisplayList>
            <X>X location</X>
            <Y>Y location </Y>
        </DisplayList>
    </Layout>
    <ButtonShape id="Id number" width="Width of the button" height="Height of the button">
        <DisplayElement>
            DisplayElement information here
        </DisplayElement>
    </ButtonShape>
</SoftkeyBar>
```

<SoftkeyBar> ELEMENT DETAILS

Table 4: <SoftkeyBar> Element

Object	Position	Type	Values	Comments
SoftkkeyBar	**Element**	**Mandatory**	**-**	**Defines softkey bar display**
Layout	Child element	Mandatory	-	Defines softkey bar layout
height	<Layout> Attribute	Mandatory	int	
Buttonshape	Child element	Mandatory	-	Defines display per softkey and it applies the same display to each key

id	<Buttonshape> Attribute	Mandatory	int	A unique ID for softkey template
width	<Buttonshape> Attribute	Mandatory	int	
height	<Buttonshape> Attribute	Mandatory	int	

<IdleScreen> ELEMENT

This is the main customization section for the XML idle screen.

```
<IdleScreen>
    <ScreenShow>weatherShow </ScreenShow>
    <ShowStatusLine>true/false</ShowStatusLine>
    <DisplayElement>
        Element display information here
    </DisplayElement>
    <DisplayBitmap isfile="true/false">
    <DisplayString font="unifont">
    </DisplayString>
    <SoftKeys>
        Softkey
    </SoftKeys>
</IdleScreen>
```

<IdleScreen> ELEMENET DETAILS

Table 5: <IdleScreen> Element

Object	Position	Type	Values	Comments
IdleScreen	**Element**	**Mandatory**	**-**	
ScreenShow	Child element	Optional	-	For weather/stock/currency display only. The text could be the "weatherShow", "stockShow" or "currencyShow".
ShowStatusLine	Child element	Mandatory	-	It could use "true" or "false" as its text. "true": the line label on the left side will always display. "false": the line label on the left side will not display. For the 1st <IdleScreen> section (default screen), it should be set to "true".
DisplayElement	Child element	Optional	-	

DisplayBitmap	Child element	Optional	-	
DisplayString	Child element	Optional	-	
DisplaySet	Child element	Optional	-	For weather/stock/currency only by default
Softkeys	Child element	Mandatory	-	

<DisplayElement> ELEMENT

This element contains all the contents to be displayed, i.e., string, picture and rectangle.

```
<DisplayElement>
    <DisplayString>String information</DisplayString>
    <DisplayBitmap>Image information</DisplayBitmap>
    <DisplayRectangle x="X location" y="Y location" width="Width" height="Height"
bgcolor="Background color"/>
</DisplayElement>
```

<DisplayElement> ELEMENET DETAILS

Table 6: <DisplayElement> Element

Object	Position	Type	Values	Comments
DisplayElement	**Element**	**Mandatory**	**-**	
DisplayString	Child element	Optional	-	Displays string
DisplayBitmap	Child element	Optional	-	Displays bitmap picture
DisplayRectangle	Child element	Optional	-	Displays rectangle

<DisplayString> ELEMENT

This element is used for displaying string information on the screen.

```
<DisplayString font ="unifont/bold" width="width of the string" height="height of the string"
halign=" center/left/right" color="color of the string" bgcolor="color of the background" >
    <X>X location</X>
    <Y>Y location </Y>
    <DisplayStr>Display String</DisplayStr>
</DisplayString>
```

<DisplayString> ELEMENET DETAILS

Table 7: <DisplayString> Element

Object	Position	Type	Values	Comments
DisplayString	**Element**	**Optional**	**-**	
font	<DisplayString> Attribute	Optional	"unifont" /"bold"	Default font type is "unifont"
width	<DisplayString> Attribute	Optional	int	
height	<DisplayString> Attribute	Optional	int	
halign	<DisplayString> Attribute	Optional	string	Default value is "left"
color	<DisplayString> Attribute	Optional	string	Default value is "Black"
bgcolor	<DisplayString> Attribute	Optional	string	Default value is "White"
X	Child element	Mandatory	int	Displays the string from X
Y	Child element	Mandatory	int	Displays the string from Y
DisplayStr	Child element	Mandatory	string	The string to be displayed. System variables can be used here
displayCondition	Child element	Optional	-	The string will be displayed under certain condition. If not specified it's always displayed

<DisplayBitmap> ELEMENT

This element is to display a bitmap picture in the screen, for example, to customize the logo. Inside the <Bitmap> tag, the picture must be encoded in base 64 format already. If you search "Base 64 Encoder" online, there are plenty of online tools as base 64 encoder to encode the .bmp picture.

```
<DisplayBitmap isfile="true/false" isflash="true/false">
    <Bitmap>Bitmap file encoded in base64 format</Bitmap>
    <X>X location</X>
    <Y>Y location</Y>
 </DisplayBitmap>
```

<DisplayBitmap> ELEMENET DETAILS

Table 8: <DisplayBitmap> Element

Object	Position	Type	Values	Comments
DisplayBitmap	**Element**	**Optional**	**-**	
isfile	<DisplayBitmap> Attribute	Optional	"true"/ "false"	"true": to display the picture embedded in the firmware. Users won't be able to directly use it for customized pictures. "false": to display the picture customized in <Bitmap> tag. Default is false.
isflash	<DisplayBitmap> Attribute	Optional	"true"/ "false"	Default value is "false".
X	Child element	Mandatory	int	Displays the picture from X
Y	Child element	Mandatory	int	Displays the picture from Y
Bitmap	Child element	Mandatory	string	The base-64 encoded .bmp file

To create .bmp and display it on the phone:

- Firstly, make sure the picture is in .bmp format and not exceed the LCD size of the phone.
- Use a base-64 encoder to encode the picture.
- Copy and paste and encoded result inside <Bitmap> tag.
- Make sure the isfile attribute is set to "false" in <DisplayBitmap> element.

Example:

```
<DisplayBitmap isfile="false">
        <Bitmap>Qk3GAgAAAAAAD4AAAAoAAAAqgAAABsAAAABAAEAAAAAAIgCAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAP///wD/////////////////////////////AAAD/////////////////////////////AAAD/////////////////////////////
AAAD+D/wf/wAAwAfgAB8B/wPw/wA+AAHAAAD+B/wH8AAAAAAAAAAAH4DwPgAAAABAAAD+A/w
D4H//wPwH//g4D4BwDgfgf//AAAD+A/wB4H//wHwP//B8D4AwBgPgf//AAAD+AfwB4H//wHwP//B8B8AY
AwPgf//AAAD/APwB4D4HwHwHx/A8B8CAEAPgPAfAAAD/AHwA8D4H4D4Hgfg8B+BgCAHgPAfAAA
D/ADwA8B8H4D4DgfgeD+BwDgHgHgfAAAD/ABwEfgAP4B/AAfwAAAA8D4H8AA/AAAD/AgwEPwA/8
B/wB/+AAAA/D8H/AD/AAAD/AwQGP///8B/////////////////AAAD/A4QGH///8B/////////////////AAAD/AcAGD///+B
/////////////////AAAD/geAHB///+B/////////////////AAAD/geAHh///+A/////////////////AAAD/gfgHw////A/////////////////AAA
D/gfgHwf///A/////////////////AAAAAAfwH4f///gf/////////////////AAACAAfwH4P///gf/////////////////AAADAA/4H8P////////
//////////////AAADgB/8H+P/////////////////AAAD/////////////////////////////AAAD/////////////////////////////
/////////AAAA= </Bitmap>
        <X>0</X>
        <Y>7</Y>
</DisplayBitmap>
```

<DisplayRectangle> ELEMENT

This element is to render rectangle display. It could be used as a frame or background bar.

```
<DisplayRectangle x="X location" y="Y location" width="Width" height="Height"
bgcolor="Background color" border-color="Rectangle border color"/>
```

<DisplayRectangle> ELEMENET DETAILS

Table 9: <DisplayRectangle> Element

Object	Position	Type	Values	Comments
DisplayRectangle	**Element**	**Optional**	**-**	**Displays rectangle**
x	<DisplayRectangle> Attribute	Optional	int	Default value is 0
y	<DisplayRectangle> Attribute	Optional	int	Default value is 0
width	<DisplayRectangle> Attribute	Mandatory	int	
height	<DisplayRectangle> Attribute	Mandatory	int	
bgcolor	<DisplayRectangle> Attribute	Optional	string	Default value is Black
border-color	<DisplayRectangle> Attribute	Optional	string	Default value is None

<SoftKeys> ELEMENT

This element is the parent element for <SoftKey> element. The purpose is to set up the softkey display and action. This element is mandatory.

```
<SoftKeys>
    <SoftKey useshapeid="buttonshapeid here">
        Softkey information here
    </SoftKey>
 </SoftKeys>
```

<SoftKeys> ELEMENET DETAILS

Table 10: <Softkeys> Element

Object	Position	Type	Values	Comments
SoftKeys	**Element**	**Mandatory**	**-**	
softkey	Child element	Mandatory	-	Defines each softkey' s display and action

<SoftKey> ELEMENT

This element defines each softkey's label and action. This element is mandatory. The text for <Action> and <conditionType> are pre-defined in the firmware already so it's recommended to keep this section by default.

```
<SoftKey useshapeid="buttonshapeid here">
        <Icon>Icon information here</Icon>
        <Action>
          Pre-defined softkey actions here
        </Action>
        <displayCondition>
            <conditionType>Pre-defined condition Type here</conditionType>
        </displayCondition>
</SoftKey>
```

<SoftKey> ELEMENET DETAILS

Table 11: <SoftKey> Element

Object	Position	Type	Values	Comments
SoftKey	**Element**	**Mandatory**	**-**	
useshapeid	<SoftKey>Attribute	Optional	int	By default, it will use the first defined <Buttonshape>. Otherwise, specify id
Icon	Child Element	Optional	-	By default it's only for SwitchSCR
Action	Child Element	Mandatory	-	Softkey Action (pre-defined)
displayCondition	Child Element	Mandatory	-	Softkey display condition (pre-defined)

Lists of the pre-defined softkey <Action> and <ConditionType> contents are described below. Please refer to full ConditionType listed in section "XML Idle Screen Display Condition Type".

Table 12: Pre-defined Softkey <Action> and <ConditionType>

Softkey	Action	ConditionType	Description
SwitchSCR	<SwitchSCR/>	SubScreen	To switch among default idle screen, weather, stock, currency screen
XML Service	<XmlService/>	XmlApp	For XML Application softkey
SignIn	<SignIn/>	signIn	Displayed when call queue feature is used with GXE5028
SignOut	<SignOut/>	signOut	
BackSpace	<BackSpace/>	backSpace	Displayed in onhook dialing state when number is entered
Cancel	<CANCEL/>	backSpace	
MissedCalls	<MissedCalls/>	missCall	Displayed when there is new missed call
FwdAll	<FwdedCalls/>	hasFowardedCallLog	Displayed when account1 is registered and "Enable Call Feature" is set to "Yes"
CnclFw	<CancelFwd/>	callFwded	Displayed when account1 has Call Forward All activated
Redial	<Redial/>	hasDialedCalllog	Displayed when there is dialed call
RefreshStock	<RefreshStock/>	By default it's not specified and it will be always displayed.	Displayed in stock idle screen
RefreshCurrency	<RefreshCurrency/>		Displayed in currency idle screen
ReverseCurrency	<ReverseCurrency/>		

VMsg	<VoiceMail/>	By default it's not specified and it will be always displayed. It could use "hasVoiceMail" so it will be displayed only when there is new voicemail.	To display Voicemail softkey.
Headset	<Headset/>	It's always displayed	To toggle to headset.
Phonebook	<PhoneBook/>	If not specified, it will be always displayed	To bring up phonebook entries

XML IDLE SCREEN ELEMENT ATTRIBUTE

The following tables list the values for frequently used element attribute.

ATTRIBUTE color/bgcolor/border-color

- For "color" attribute, the default value is "Black";
- For "bgcolor" attribute, the default value is "White";
- For "border-color" attribute, the default value is "None".

Table 13: Attribute color/bgcolor/border-color

color/bgcolor/border-color	Details
None	
Black	
Dark6	
Dark5	
Dark4	
Dark3	
Dark2	
Dark1	
Gray	
LightGray	
Light1	
Light2	
Light3	
Light4	
Light5	
Light6	
White	

SYSTEM VARIABLES IN STRING DISPLAY

In <DisplayString> element, the following system variables could be used to display the pre-defined values in XML customized idle screen.

Table 14: System Variables for XML Idle Screen

$String			
$a	This variable is replaced with the configured account name	$A	This variable is replaced with configured softkey label
$b	N/A	$B	This variable is replaced with the current day of month with leading zero, possible values: 01, 02, …, 31
$c	This variable is replaced with Missed Call string along with missed call count.	$C	This variable is replaced with DND (Do-Not-Disturb) label when DND is enabled
$d	This variable is replaced with the current day of month with leading zero, possible values: 1, 2, …, 31	$D	This variable is replaced with the current day of month with leading zero, possible values: 01, 02, …, 31
$e	This variable is replaced with the onhook dialing number	$E	N/A
$f	This variable is replaced with the Month-week-date format based on the configuration	$F	N/A
$g	This variable is replaced with the country name for weather information	$G	This variable is replaced with the number of the Missed Call
$h	This variable is replaced with the current hour of day in 12-hour format with leading zero, possible values: 01, 02, …, 12	$H	This variable is replaced with the current hour of day in 24-hour representation with leading zero, possible values: 00, 02, …, 23
$i	This variable is replaced with the system IPV6 Address	$I	This variable is replaced with the system IPV4 Address
$j	This variable is replaced with Forwarded Call string along with forwarded calls count	$J	N/A
$l	N/A	$L	This variable is replaced with the city name for weather information
$m	This variable is replaced with the current minute of hour with leading zero, possible	$M	This variable is replaced with the current month in English, possible values: January,

	values: 01, 02, …, 59		February, …, December
$n	This variable is replaced with the current month in number with leading zero, possible values: 1, 2, …, 12	$N	This variable is replaced with the configured SIP Display Name or account name
$o	This variable is replaced with the current month in number with leading zero, possible values: 01, 02, …, 12	$O	N/A
$p	N/A	$P	This variable is replaced with the current AM/PM status in upper case, possible values: AM, PM
$r	This variable is replaced with the volume level	$R	N/A
$s	This variable is replaced with the current second of minute with leading zero, possible values: 01, 02, …, 59	$S	This variable is replaced with the state name of the weather information
$t	N/A	$T	This variable is replaced with the current hour:minute (am/pm) of the day, in which ":" will flash per second. Depending on user's configuration, it will be displayed as 12 hour or 24 hour format. Possible values: 1:00pm, 13:00
$v	This variable is replaced with 5V power usage alert message when incorrect power is used	$V	This variable is replaced with the configured Account SIP Server host
$w	This variable is replaced with the temperature of the weather information	$W	This variable is replaced with the current day of week and has the following possible values: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday
$x	This variable is replaced with the humidity of the weather information	$X	This variable is replaced with the configured Account SIP User ID
$y	This variable is replaced with the current year in 2-digit number, for example: 06, 07	$Y	This variable is replaced with the current year in 4-digit number, for example: 2006, 2007 …

Note:

To display "$", please use "$$" escape sequence.

XML IDLE SCREEN DISPLAY CONDITION TYPE

The following tables list all the available <ConditionType> in XML idle screen. They could be used for softkey display or string display.

Table 15: ConditionType for XML Idle Screen

ConditionType	Description for softkey/string display
SubScreen	Displayed on idle screen, weather, and IP address screen.
signIn	Displayed when call queue feature is used with GXE5028
signOut	
backSpace	Displayed in onhook dialing state when number is entered
missCall	Displayed when there is new missed call
hasFowardedCallLog	Displayed when account1 is registered and "Enable Call Feature" is set to "Yes"
callFwded	Displayed when account1 has Call Forward All activated
hasDialedCalllog	Displayed when there is dialed call
hasVoiceMail	Displayed when there is new voicemail.
alwaysDisplay	Default display if not specified

GXP16xx XML IDLE SCREEN CUSTOMIZATION FILE EXAMPLE

This is the XML idle screen customization file example, which is corresponding to the LCD examples above:

```
<?xml version="1.0" encoding="UTF-8"?>
<Screen>
  <SoftkeyBar>
    <Layout height="13" >
      <DisplayList>
        <X>0</X>
        <Y>0</Y>
      </DisplayList>
    </Layout>
    <ButtonShape id="0" width="43" height="13">
      <DisplayElement>
        <DisplayBitmap isfile="true">
          <Bitmap>/app/resource/etc/softkey_button_b.bmp</Bitmap>
          <X>0</X>
          <Y>0</Y>
        </DisplayBitmap>
      </DisplayElement>
      <DisplayElement>
        <DisplayString font="unifont" halign="center" color="White" bgcolor="Black" width="39" height="11">
          <DisplayStr>$A</DisplayStr>
          <X>1</X>
          <Y>1</Y>
        </DisplayString>
      </DisplayElement>
    </ButtonShape>
  </SoftkeyBar>
  <IdleScreen>
    <ShowStatusLine>false</ShowStatusLine>
    <DisplayElement>
      <DisplayString font="unifont" width="70" height="12">
        <DisplayStr>$f</DisplayStr>
        <X>0</X>
        <Y>0</Y>
      </DisplayString>
```

```
    <DisplayString font="unifont" halign="right" width="50" height="12">
      <DisplayStr>$T</DisplayStr>
      <X>77</X>
      <Y>0</Y>
    </DisplayString>
  </DisplayElement>
  <DisplayElement>
  <!-- COMPANY NAME    -->
    <DisplayString font="unifont" halign="center" width="128" bgcolor="White">
      <DisplayStr>XML EXAMPLE</DisplayStr> <!-- default is $b -->
      <X>0</X>
      <Y>12</Y>
    </DisplayString>
  </DisplayElement>
  <!-- forwarded call msg -->
  <DisplayElement>
    <DisplayString font="unifont" halign="center" width="128" bgcolor="White">
      <DisplayStr>$j</DisplayStr>
      <X>0</X>
      <Y>12</Y>
      <displayCondition>
        <conditionType>hasFowardedCallLog</conditionType>
      </displayCondition>
    </DisplayString>
  </DisplayElement>
  <DisplayElement>
    <DisplayBitmap isfile="true">
      <Bitmap>/app/resource/icon/misscall_13.bmp</Bitmap>
      <X>6</X>
      <Y>13</Y>
      <displayCondition>
        <conditionType>missCall</conditionType>
      </displayCondition>
    </DisplayBitmap>
    <!-- TEXT CONTENTS -->
    <DisplayString font="unifont" width="102" bgcolor="White">
      <DisplayStr>$c</DisplayStr>
      <X>25</X>
      <Y>12</Y>
      <displayCondition>
```

```
        <conditionType>missCall</conditionType>
      </displayCondition>
    </DisplayString>
  </DisplayElement>
  <SoftKeys>
    <SoftKey>
      <Action>
        <SwitchSCR/>
      </Action>
      <displayCondition>
        <conditionType>SubScreen</conditionType>
      </displayCondition>
    </SoftKey>
    <SoftKey>
      <Action>
        <SignIn/>
      </Action>
      <displayCondition>
        <conditionType>signIn</conditionType>
      </displayCondition>
    </SoftKey>
    <SoftKey>
      <Action>
        <SignOut/>
      </Action>
      <displayCondition>
        <conditionType>signOut</conditionType>
      </displayCondition>
    </SoftKey>
    <SoftKey>
      <Action>
        <BackSpace/>
      </Action>
      <displayCondition>
        <conditionType>backSpace</conditionType>
      </displayCondition>
    </SoftKey>
    <SoftKey>
      <Action>
        <CANCEL/>
```

```
        </Action>
        <displayCondition>
          <conditionType>backSpace</conditionType>
        </displayCondition>
      </SoftKey>
      <SoftKey>
        <Action>
          <MissedCalls/>
        </Action>
        <displayCondition>
          <conditionType>missCall</conditionType>
        </displayCondition>
      </SoftKey>
      <SoftKey>
        <Action>
          <FwdedCalls/>
        </Action>
        <displayCondition>
          <conditionType>hasFowardedCallLog</conditionType>
        </displayCondition>
      </SoftKey>
      <SoftKey>
        <Action>
          <CancelFwd/>
        </Action>
        <displayCondition>
          <conditionType>callFwded</conditionType>
        </displayCondition>
      </SoftKey>
    </SoftKeys>
  </IdleScreen>
  <IdleScreen>
    <ScreenShow>weatherShow</ScreenShow>
    <ShowStatusLine>false</ShowStatusLine>
    <!-- LINE -->
    <DisplayString font="unifont">
      <DisplayStr>$L, $S, $g</DisplayStr>
      <X>2</X>
      <Y>-2</Y>
    </DisplayString>
```

```
    <DisplayString font="unifont">
      <DisplayStr>$w, $0t</DisplayStr>
      <X>2</X>
      <Y>13</Y>
    </DisplayString>
    <SoftKeys>
      <SoftKey>
        <Action>
          <SwitchSCR/>
        </Action>
        <displayCondition>
          <conditionType>SubScreen</conditionType>
        </displayCondition>
      </SoftKey>
    </SoftKeys>
  </IdleScreen>
</Screen>
```