Notification Providers Developer Guide

iAnywhere® Mobile Office 5.7

SYBASE® iAnywhere

Document ID: DC01173-01-0570-01
LAST REVISED: *September 2009*

Copyright and Trademarks

Disclaimer

This documentation, as well as the software described in it, is furnished under a license agreement. The content of this documentation is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by iAnywhere Solutions, Inc. iAnywhere Solutions, Inc. assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation. Any changes in the programs will be incorporated in a future edition of this publication.

Technical Support

For product-specific technical information, visit the iAnywhere Technical Support site at http://frontline.sybase.com/support/.

NOTE: Register at our technical support site for the latest information for your product. This site is available only to customers with a valid maintenance contract.

United States
+1 800 235 7576, menu options 2, 1
+1 208 322 7575, menu options 2, 1
6:00 a.m. to 6:00 p.m. Mountain Time (GMT -7)

United Kingdom
+44 (0) 117 333 9032
8:00 a.m. to 6:00 p.m. (GMT+1)

Germany
+49 (0) 7032 798-555
8:00 a.m. to 6:00 p.m. (GMT+1)

France
+33 (0) 825 826 835
8:00 a.m. to 6:00 p.m. (GMT+1)

Benelux
+31 (0) 302 478 455
8:00 a.m. to 6:00 p.m. (GMT+1)

Table of Contents

Copyright and Trademarks ii
Disclaimer ii
Technical Support ii
Table of Contents iii
Welcome 1
Solution Overview 1
Custom Notification Providers 2
ISMSProvider Interface 2
Syntax 2
ISMSProvider.Send Method 2
Syntax 2
NotificationRetryException Class 3
Enum LogLevel 5
NotificationProviders.xml Configuration File 6
NotificationProviders.xml File Reference 6
Provider 6
Sample NotificationProviders.xml File 7
Sybase 365 SMS Sample Provider 9
Appendix A: Registry Values for Notification Retry Times 10

Welcome

Welcome to iAnywhere® Mobile Office, a component of the Sybase® Information Anywhere® Suite. iAnywhere Mobile Office is specifically designed for today's mobile business workforce. It combines fully integrated wireless email and PIM with on-device security and business process mobilization.

iAnywhere Mobile Office ships with "out of the box" notification mechanisms for SMS provisioning and an IP-based push mechanism for email and PIM notifications. This document will introduce you to developing custom Notification Providers to replace the built-in notification mechanisms with your own.

Solution Overview

iAnywhere Mobile Office supports customization of the notification mechanisms used to provision a user and to notify users of changed PIM items. The customizations are implemented as a .NET assembly and installed on the Mobile Office server.

When an administrator performs the task of registering a user in the Mobile Office administration console, by default the system will send an e-mail to the user's Inbox and optionally an SMS message to a user's device. If the SMS option is chosen, the Administrator is required to enter the SMS email address (usually in the form <mobile_phone_number>@<carrier_domain>) for the user.

By using a custom notification provider, the administrator can enter only the mobile phone number and have the phone number passed into a .NET assembly where any mechanism can be used to send the registration info to the user's device.

A custom notification provider can also be used for SMS push notifications (in place of an always connected IP connection). When new PIM items are available for a device to retrieve, if the device is configured to accept SMS push notifications, a SMS message will be sent to device to automatically trigger the device to download the new item. This SMS notification message will not be visible in the device's SMS inbox.

Custom Notification Providers

As described above, the system may invoke a custom notification provider during the user registration process, or if configured, for SMS Push notifications.

To implement a Custom Notification Provider for either scenario, a custom .NET class that implements the **iAnywhere.MobileOffice.Notifications.ISMSProvider** interface must be supplied and configured in the system. Your .NET project will need to add a reference to "\iAnywhere Mobile Office\Bin\INotifications.dll" for the **iAnywhere.MobileOffice.Notifications.ISMSProvider** interface.

ISMSProvider Interface

Syntax

Visual Basic (Declaration)

```
Public Interface ISMSProvider
```

C#

```
public interface ISMSProvider
```

ISMSProvider.Send Method

Syntax

Visual Basic (Declaration)

```
Function Send( _
   ByVal strUsername As String,  _
   ByVal strDeviceId As String,  _
   ByVal strDeviceType As String,  _
   ByVal strPhoneNumber As String,  _
   ByVal strIMSI As String,  _
   ByVal strSMSAddress As String,  _

   ByVal strConfig As String,  _
   ByVal strNotification As String _
) As Boolean
```

C#

```
bool Send(
   string strUsername,
   string strDeviceId,
   string strDeviceType,
   string strPhoneNumber,
   string strIMSI,
   string strSMSAddress,
   string strConfig,
   string strNotification
)
```

Parameters

strUsername
Mobile Office user name

strDeviceId
Device Id that the notification will be delivered to – only available if known at time of invocation

strDeviceType
Type of device – only available if known at time of invocation

The following table shows the currently supported device type strings:

Device Type	Description
ce	Windows Mobile Professional (WM6) or Pocket PC (WM5) device
smartphone	Windows Mobile Standard (WM6) or Smartphone (WM5) device
epoc	Symbian device
iphone	iPhone

strPhoneNumber
Mobile Phone Number retrieved from the device.

strIMSI
International Mobile Subscriber Identity linked to the SIM card in the device. Only available if successfully retrieved and available at time of invocation.

strSMSAddress

SMS Address used in the Mobile Office Admin application when registering the device (if entered).

strConfig
Entire Config section in NotificationProviders.xml

strNotification
Notification message to send to device

Return Value

True if the Send was successful, otherwise, false.

NotificationRetryException Class

NotificationRetryException can be thrown by custom provider to tell the notification engine to resend a failed notification after a specified period.

By default, if a custom provider fails to send a notification, it should throw an exception other than NotificationRetryException to indicate the failure to notification engine. Notification engine

will log the exception as an error in Mobile Office log viewer, and also delete the notification item from the notification queue.

The custom provider can have more control over how the notification engine should handle the error by throwing NotificationRetryException. The NotificationRetryException class lets the custom provider tell the notification engine to keep the failed notification item in the queue and resend it to the custom provider after a specified time. The class can also specify a log level to be used by notification engine when it writes the error message into Mobile Office log viewer.

Two constructors are provided for NotificationRetryException class:

Syntax

Visual Basic (Declaration)

```
Public Sub New(ByVal iDelaySeconds As Integer)
```

C#

```
public NotificationRetryException(int iDelaySeconds)
```

Parameters

iDelaySeconds

The seconds to wait before sending the item again to provider by notification engine

This constructor specifies how many seconds the notification engine should wait before sending the item again to the provider. No log will be added into the Mobile Office log viewer when this exception is thrown.

Syntax

Visual Basic (Declaration)

```
Public Sub New(ByVal iDelaySeconds As Integer, ByVal logLevel As
iAnywhere.MobileOffice.Notifications.LogLevel, ByVal strLogMessage As String)
```

C#

```
public NotificationRetryException(int iDelaySeconds,
iAnywhere.MobileOffice.Notifications.LogLevel logLevel, string strLogMessage)
```

Parameters

iDelaySeconds

The seconds to wait before sending the item again by notification engine

logLevel

The log level used by notification engine when logging the exception into the Mobile Office log viewer.

strErrorMessage

The error message logged into the Mobile Office log viewer.

Enum LogLevel

LogLevel can be used by NotificationRetryException to specify what log level should be used when notification engine writing the log entry into Mobile Office log viewer.

LogLevel.None

Do not log the item

LogLevel.Error

Log an error

LogLevel.Warning

Log as a warning

LogLevel.Information

Log as an information

NotificationProviders.xml Configuration File

Custom Notification Providers are configured via an XML configuration file which is installed by default to \Program Files\ iAnywhere Mobile Office \Bin\NotificationProviders.xml.

NotificationProviders.xml File Reference

Provider

```
      <Provider EventType = “SMSActivation”|“SMSPush”
PreventDuplicates=”true”|”false”>
            <AssemblyName/>
            <ClassName/>
            <Config/>
      </Provider>
```

Describes a .NET Assembly and Type which will be used to notify a user of Mobile Office registration in place of the existing SMS e-mail mechanism. or send a SMS push notification.

If the AssemblyName and/or ClassName fields are invalid or contain errors, the notification provider will be disabled.

Attributes

EventType

The EventType attribute indicates whether the configured provider is for sending device activation messages (SMSActivation), or SMS push notification messages (SMSPush).

PreventDuplicates

The PreventDuplicates attribute indicates whether to allow notification provider queues multiple SMS notifications of this event type for the same device. For SMSPush event provider, this attribute should be set to “true”. For SMSActivation event provider, this attribute should be set to “false”.

Innertags

<AssemblyName>

…

</AssemblyName>

The .NET assembly which contains the custom Notification Provider. Assembly name can be the fully qualified name if the assembly is deployed in the GAC or the filename if the assembly is deployed in the Mobile Office ..\Bin folder.

<ClassName>

…

</ClassName>

The .NET class which implements the iAnywhere.MobileOffice.Notifications.ISMSProvider interface.

```
<Config>
      …
</Config>
```

This section is passed in its entirety (including the start and end <Config> tags) as the *strConfig* parameter of the ISMSProvider.Send method. Developers can place additional information here required to connect to their particular notification backend such as server name, port, etc.

Sample NotificationProviders.xml File

The following is a sample of the most recent NotificationProviders.xml file.
Note that if an old formatted NotificationProviders.xml already exists in server's bin folder, then it will not be automatically updated to the new version when installing a server patch installation. The notification configuration file is backward compatible, so all functions in the old configuration file still work. However, in order to use the features available only in the new format of the notification configuration file, the administrator needs to manually update the NotificationProviders.xml to the new format as shown below.

```
- <NotificationProviders>
      <!--  Uncomment the SMSPush Provider's start and end tags to configure a SMS
      push provider. The specified assembly needs to be put in server's Bin
      directory. The Config section is specific to each provider type, and will be
      assigned to strConfig parameter when IProviderEx.Send() method is called.  -
      ->
<!-- Provider EventType="SMSPush" PreventDuplicates="true">
            <ClassName>iAnywhere.MobileOffice.Notifications.SMSGatewayNotification
            </ClassName>        <AssemblyName>Sy365Notification.dll</AssemblyName>
                  <Config>
                  <Scheme>http</Scheme>
            <ServerUrl>localhost</ServerUrl>
      <ServerPath>/subdir/filename.sms</ServerPath>
      <ServerPort>80</ServerPort>
      <Username>Somebody</Username>
      <Password>SomePassword</Password>
                  <CustomData>Some string</CustomData>
                  <SMSAddressPriority>0</SMSAddressPriority>
            </Config>
</Provider  -->
<!-- Uncomment the SMSActivation Provider's start and end tags to configure a SMS
activation provider. The specified assembly needs to be put in server's Bin
directory. The Config section is specific to each provider type, and will be
assigned to strConfig parameter when  IProviderEx.Send() method is called.  -->
<!-- Provider EventType="SMSActivation" PreventDuplicates="false">
            <ClassName>iAnywhere.MobileOffice.Notifications.SMSGatewayNotification
            </ClassName>
            <AssemblyName>Sy365Notification.dll</AssemblyName>
            <Config>
                  <Scheme>http</Scheme>
                  <ServerUrl>localhost</ServerUrl>
                  <ServerPath>/subdir/filename.sms</ServerPath>
                  <ServerPort>80</ServerPort>
```

```
                <Username>Somebody</Username>
                <Password>SomePassword</Password>
                <CustomData>Some string</CustomData>
                <SMSAddressPriority>0</SMSAddressPriority>
            </Config>
</Provider  -->

</NotificationProviders>
```

Sybase 365 SMS Sample Provider

A sample notification provider project is installed with the Mobile Office server under "\iAnywhere Mobile Office\Tools\Sybase 365 SMS" folder. The sample project shows how to send SMS activation messages or SMS push messages through Sybase 365 gateway server. For more information on Sybase 365 products and services, please see http://www.sybase.com/mobileservices.

In order to use the Sybase 365 gateway server to send the notification message, the output of the project needs to be copied into iAnywhere Mobile Office\Bin folder. In addition, you will need to update NotificationProviders.xml with your Sybase 365 account information.

Appendix: Registry Values for Notification Retry Times

The timeout, retry, and polling times for SMS push notifications are configurable in the registry.

The values described here are found in the following key:

HKEY_LOCAL_MACHINE\SOFTWARE\Extended Systems\OneBridge Sync Server\Server

Value: NotificationEngineTimeoutSeconds

Type: DWORD

Default: 600 seconds

Description: Controls how long the engine waits for a response from notification providers before it aborts the current notification.

Value: NotificationEngineRetrySeconds

Type: DWORD

Default: 2* NotificationEngineTimeoutSeconds

Description: Controls how long the engine waits before retrying the notification that has been timed out.

Value: NotificationEnginePollSeconds

Type: DWORD

Default: 10 seconds

Description: Controls how long the engine waits before polling for a notification.