PageMate®

Developer's Guide

Version 3.3-1

Systemetrics, Inc.
153 Lexington Avenue
Cambridge, MA 02138 USA
Phone 617.868.8308
http://www.system.com

Printed January 20, 2014

TABLE OF CONTENTS

Foreword

PageMate is a software product for Windows, Unix and OpenVMS that provides support for electronic messaging to pagers, cell phones, radios, PDAs and other portable electronic communication and display devices.

PageMate® has evolved since 1984 from a utility for electronic paging to an integrated messaging application that now provides capabilities to track and deliver messages using a wide variety of media and technologies. In the 1980's, paging was confined to sending short numeric messages to devices commonly called "beepers". We think of this as being the first generation of electronic messaging to personal portable devices.

In the 1990's, the second generation of electronic messaging extended paging capabilities to include support for alphanumeric messages. Users were freed from the burden of being forced to try to interpret short, cryptic numeric messages. PageMate Version 2, introduced in 1993, provided early support for users and applications that required robust and reliable messaging to the new breed of alphanumeric pagers. By the end of the decade, PageMate's support for alphanumeric messaging had been extended to include client-server implementations with an interface to electronic mail and multi-threaded message queuing services in heterogeneous networks of Windows, UNIX and OpenVMS systems.

At the turn of the century, the development of new technologies for electronic messaging and digital communications, together with the explosive growth and popularity of the Internet and web-based applications, demanded that we take a fresh look at the services and messaging support provided by PageMate. From its inception, PageMate has provided unique capabilities to support messaging from other computer processes, including help desk, dispatch, process control and SCADA applications, that can impose variable and unpredictable demands in message volume. PageMate's client-server architecture, support for automatic server fail-over, and administrative capabilities for auditing and resource management have long made it the solution of choice for electronic messaging in mission-critical applications.

While continuing to support all of the existing services and capabilities of PageMate Version 2, Version 3 supports a wide variety of new features and functionality, including new options for integrating PageMate functionality in site-specific and web-based applications. Options ranging from simple command line scripts and procedures to more sophisticated XML (Extensible Markup Language) and COM (Component Object Model) functionality include:

- A command line interface (CLI) providing capabilities to submit messages and inquire about the status of messages previously submitted via scripts and command procedures that execute on a PageMate Client.

- A native client application program interface (API) providing capabilities to submit messages and inquire about the status of messages previously submitted via function calls embedded in site-specific applications that execute on a PageMate Client.

- A native administrator application program interface (API) providing capabilities to manage the PageMate catalog and perform other administrative operations via function calls embedded in site-specific applications that execute on a PageMate Server.

- An XML RPC (Extensible Markup Language Remote Procedure Call) interface providing capabilities to submit messages and inquire about the status of messages previously submitted via TCP stream socket operations embedded in site-specific applications that execute on any system in a network.

- A COM+ (Component Object Model) interface providing support for a wide array of both client and server/administrator capabilities in site-specific applications that execute on a PageMate Automated Messenger (PAM) Server system.

PageMate is a client-server software product. In a standalone configuration, both PageMate Server and PageMate Client reside on the same system. A local instance of PageMate Client is automatically included and installed with PageMate Server. PageMate Client can be installed separately on systems other than the Server to provide native PageMate interactive GUI, API and command line interface components on the client. XML RPC and COM+ interfaces are server components that can be accessed by site-specific applications running either on the PageMate Server or, through suitable network or web-based services, from any system in a network.

This Developer's Guide provides information about features and functions in PageMate that can be used to incorporate PageMate functionality in site-specific applications. Additional PageMate documentation, application notes, product news and technical support resources, can be found on Systemetrics' web site at `http://www.system.com` and on a site dedicated to PageMate support at `http://www.pagemate.com`.

Systemetrics welcomes and encourages your feedback and suggestions about our software, documentation, services and support. You are invited to call us with personal suggestions or send e-mail to `support@pagemate.com.`

Chapter

1

Client CLI

The PageMate Client Command Line Interface provides support for client functionality from scripts and command procedures on Windows, UNIX and OpenVMS.

PageMate Client Command Line Interface (CLI) is an application that provides capabilities to submit messages and track message delivery from the command line level. PageMate Client CLI is the interface of choice for submitting messages from scripts, command procedures and third-party applications that do not provide source-level control. Many network monitors and plant automation systems, for example, provide an option to execute a script or command procedure in response to an alarm, trap or event. Scripts and command procedures are commonly used with these applications to send messages via PageMate to provide notice of alarms, traps and events to technical support and administrative personnel.

Command line functionality is provided on Windows platforms via the `page` command. The command in UNIX environments is `pagecmd`. OpenVMS provides support for both `page/cmd` and `page/dcl`. Apart from differences in the spelling of the command verb, however, functionality of command line mode operation on all platforms is identical.

Command Syntax

Common command line syntax for PageMate client on all platforms is:

`command [options] [objects]`

where

`command` is the platform-specific command `page` (Windows), `pagecmd` (UNIX) or `page/cmd` (OpenVMS);

`[options]` is a series of one or more command options prefixed with minus sign (-); and

`[objects]` is a series of one or more command objects as required by `options.`

Options include

`-s` `send`. Send a message to a subscriber, group or profile. Must be immediately followed by a subscriber, group or profile name or a PIN and a message. If used together with other options, -s must be the last option in the command line.
Example: pagecmd –p2 –s WATSON "Come here, I need you."

`-p` `priority`. Specifies message priority. Must be conjoined (sans space) with a digit in the numeric range 0 through 3, specifying message priority. Valid for use in conjunction with -s (send) only, otherwise ignored.
See example above.

`-n` `network`. Specifies the network (paging service provider) to which a direct page is to be submitted. Must be conjoined (sans space) with a network name specification. Valid for use in conjunction with –s (send) when direct paging is enabled and a numeric PIN is specified as the addressee.
Example: page -p1 -nSkyTel -s 8001234567 "Call home"

`-r` `profile`. Specifies a profile to be used in processing replies to a two-way message submitted to a subscriber. Must be conjoined (sans space) with a profile name specification. Valid for use in conjunction with –s (send) on clients served by a PAM server. Addressee must be a listed subscriber. Message is sent as a page to the specified subscriber with response required pursuant to the profile.
Example: page -rSLA60 -s FRED "Please confirm ticket 46117 asap"

`-l` `list`. Requests a listing of subscriber and/or group records from the PageMate catalog. Must be followed by a catalog name specification (optionally including wildcard). May be optionally conjoined (sans space) with one or more of the following:

`b` – brief format (one line per catalog name)
`c` – list records from common catalog only
`u` – list records from user private catalog only
`s` – list subscriber records only
`r` – list group records only
`v` – verify name (returns "OK" if name is valid)

Example: page -lbc S*
lists all records in the system common catalog having names beginning

with the letter S. Listing is in brief format (one line per name).

`-i` `import`. Import to subscriber catalog from file. Must be immediately followed by a filename specification.
Example: pagecmd –i /usr/temp/pagers.txt

`-x` `export`. Export subscriber catalog to file. Must be immediately followed by a filename specification.
Example: page/cmd -x sys$common:[pagemate.site]pagers.txt

`-v` `version`. Display software version and build date for currently installed PageMate software.
Example: page -v

`-h` `help`. Display help for command line commands and syntax. May optionally be followed by an option (with minus sign introducer) or suboption (sans minus sign).
Example: page -h -l
displays help for –l (list) option
Example: page -h b
displays help for b (brief) suboption for list

Legacy Functionality

In comparison to earlier versions, PageMate Version 3 provides considerably broader functionality under the command line interface. It also standardizes command line syntax among different platforms. All new scripts and command procedures should be written to use the command line syntax described above. For those users who have existing scripts and command procedures that use deprecated "send" or "show" command verb syntax, legacy support for this syntax is provided in PageMate Version 3. Scripts and command procedures that use "send" or "show" command verb syntax must strictly adhere to PageMate Version 2 command line syntax and functionality (*i.e.*, you cannot mix Version 2 and Version 3 syntax in a single statement).

Chapter

2

Client API

PageMate's Client API provides support for submitting messages and inquiring about the status of messages previously submitted from site-specific applications.

The PageMate Client Application Program Interface (API) provides source-level capabilities to submit messages and track message delivery. The PageMate Client API is the interface of choice for custom, site-specific applications and third-party applications that need a direct, highly efficient pipeline to PageMate from site-specific applications that support unmanaged C language call structures. To use the PageMate API, you must modify application source code to call PageMate functions (*e.g.*, `pagemate_send`). Some third-party help desk, plant automation and supervisory control and data acquisition (SCADA) software products provide integrated support for the PageMate API.

PageMate Client API functions include `pagemate_send`, `pagemate_psend`, `pagemate_direct` and `pagemate_status`. The PageMate Client API has been written primarily in and for applications written in C/C++. If your application is written in C/C++, you can use the C header files provided in the PageMate include directory (*e.g.*, ~\Program Files\PageMate\Include) on your system to provide prototype definitions for PageMate API functions.

If your application is written in Visual Basic, you will need to provide declarations for the PageMate API functions you want to use. Your declarations will need to specify arguments as in the example below. In addition, if the PageMate Library directory will not be specified by the Path environment variable when your application executes, you will need to use the `Lib` clause in your `Declare` statement to specify where the applicable PageMate DLL is located on your system. An example declaration for `pagemate_send` is shown immediately below.

```
Declare Function pagemate_send _
Lib "c:\Program Files\PageMate\Lib\PGRPCDLL.dll" _
ByVal PagerName As String, ByVal Message as String, _
ByVal ErrorBuffer as String) As Long
```

If you place the Declare statement in a `Form` or `Class` module, you should precede it with the `Private` keyword. After the API function is declared, you can call it just as you would any Visual Basic function. An example (excerpt) from Visual Basic code to call `pagemate_send` is shown below.

```
Dim MessageSeqn As Long Dim PagerID As String
Dim Message As String
Dim ErrorBuffer As String * 160
PagerID = "TOM_JONES"
Message = "Test Message"
MessageSeqn = pagemate_send(PagerID, Message, ErrorBuffer)
```

PageMate Client API functions are listed and described in alphabetical order by function name on the following pages in this section.

pagemate_direct

FORMAT

pagemate_direct (networkname, pagertype, pin, message, reference, priority, error_buffer)

DESCRIPTION

pagemate_direct is an API function that submits a message for delivery to a pager, cell phone or similar mobile device that is served by a listed network but not listed in a subscriber record in the PageMate system common catalog. pagemate_direct provides functionality for unlisted subscribers similar to that provided by pagemate_psend for listed subscribers, but absent benefits of history and audit trail provided for messages addressed to listed subscribers.

ARGUMENTS

networkname

address of a null-terminated string or character array, 32 characters or less in length, specifying a network (electronic messaging system or service provider) name as defined in the system networks catalog (networks.dat) on the PageMate Server

pagertype

an integer value in the range 0 to 1, specifying the type of the recipient device as follows: 0 (zero) for pagers and equivalent devices capable of displaying alphanumeric text; 1 (one) for pagers (*e.g.*, beepers) that can display only numeric characters.

pin

address of a null-terminated string or character array, 16 characters or less in length, specifying the network-specific PIN (pager identification number) for the recipient pager

message

address of a null-terminated string or character array, 500 characters or less in length, specifying the text or numeric message to send

reference

address of a null-terminated string or character array, 16 characters or less in length, specifying a user or application-specific reference (e.g.,

trouble ticket or case number, application-specific sequence or message reference number)

priority

an integer value in the range 1 to 3, specifying message priority for the message send request.

error_buffer

address of a 160-character array into which pagemate_direct will write detailed error message information.

RETURNS

message sequence number or longword condition value, as follows:

+n positive integer sequence number of message successfully enqueued; or

-1 error condition value. Text description of error or exception condition is provided via error_buffer parameter.

NOTES

pagemate_direct is an extended parameters version of pagemate_send that can be called from any application program written in any language that supports the C calling standard.

Application programs that call pagemate_direct on Windows platforms must include /PageMate/Include/PageDLL.h and be linked with /PageMate/Lib/PGRPCDLL.lib and /PageMate/Lib/PGRPCDLL.dll.

Application programs that call pagemate_direct on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_direct on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

DIRECT_ENABLE must be set to TRUE on the PageMate Server to enable support for pagemate_direct functionality on clients. The default value for DIRECT_ENABLE is FALSE. Setting DIRECT_ENABLE to TRUE on a PageMate Server enables direct messaging functionality on the server and on all clients served by the server. Direct messaging functionality cannot be locally enabled or disabled on a client.

If the priority value specified in a call to pagemate_direct is anything other than 0, 1, 2 or 3, the site-specific default priority value will be used instead.

If the message is successfully enqueued, pagemate_direct returns a positive integer message sequence number as the value of the function. This sequence number can subsequently be passed to pagemate_status to obtain completion status for the message. If the message cannot be enqueued for transmission, pagemate_direct returns a negative integer function value and a text description of the error or exception condition.

pagemate_direct_reply

FORMAT

```
pagemate_direct_reply  (networkname, pagertype,
pin, message, reference, priority, reply_to,
error_buffer)
```

DESCRIPTION

pagemate_direct_reply is an API function that submits a message for delivery to a pager, cell phone or similar mobile device that is served by a listed network but not listed in a subscriber record in the PageMate system common catalog. pagemate_direct_reply provides functionality equivalent to pagemate_direct with the addition of reply_to.

ARGUMENTS

networkname

address of a null-terminated string or character array, 32 characters or less in length, specifying a network (electronic messaging system or service provider) name as defined in the system networks catalog (networks.dat) on the PageMate Server

pagertype

an integer value in the range 0 to 1, specifying the type of the recipient device as follows: 0 (zero) for pagers and equivalent devices capable of displaying alphanumeric text; 1 (one) for pagers (*e.g.*, beepers) that can display only numeric characters.

pin

address of a null-terminated string or character array, 16 characters or less in length, specifying the network-specific PIN (pager identification number) for the recipient pager

message

address of a null-terminated string or character array, 500 characters or less in length, specifying the text or numeric message to send

reference

address of a null-terminated string or character array, 16 characters or less in length, specifying a user or application-specific reference (e.g., trouble ticket or case number, application-specific sequence or message reference number)

priority

an integer value in the range 1 to 3, specifying message priority for the message send request.

reply_to

address of a null-terminated string or character array, 128 characters or less in length, specifying an address to which replies from the message recipient should be directed by the messaging service. Functionality of reply_to varies by protocol and paging service provider and is supported in conjunction with PageMate Automated Messenger (PAM) Server only. This parameter is required but may be a null or empty string.

error_buffer

address of a 160-character array into which pagemate_direct_reply will write detailed error message information.

RETURNS

message sequence number or longword condition value, as follows:

+n positive integer sequence number of message successfully enqueued; or

-1 error condition value. Text description of error or exception condition is provided via error_buffer parameter.

NOTES

pagemate_direct_reply is an extended parameters version of pagemate_send that can be called from any application program written in any language that supports the C calling standard.

Application programs that call pagemate_direct_reply on Windows platforms must include /PageMate/Include/PageDLL.h and be linked with /PageMate/Lib/PGRPCDLL.lib and /PageMate/Lib/PGRPCDLL.dll.

Application programs that call pagemate_direct_reply on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_direct_reply on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

DIRECT_ENABLE must be set to TRUE on the PageMate Server to enable support for pagemate_direct functionality. The default value for DIRECT_ENABLE is FALSE. Setting DIRECT_ENABLE to TRUE

on a PageMate Server enables direct messaging functionality on the server and on all clients served by the server. Direct messaging functionality cannot be locally enabled or disabled on a client.

If the priority value specified in a call to pagemate_direct_reply is anything other than 0, 1, 2 or 3, the site-specific default priority value will be used instead.

If the message is successfully enqueued, pagemate_direct_reply returns a positive integer message sequence number as the value of the function. This sequence number can subsequently be passed to pagemate_status to obtain completion status for the message. If the message cannot be enqueued for transmission, pagemate_direct_reply returns a negative integer function value and a text description of the error or exception condition.

pagemate_psend

FORMAT

pagemate_psend (pagername, message, reference, priority, error_buffer)

DESCRIPTION

pagemate_psend is an API function that submits a message for delivery to a subscriber, group or profile listed in the PageMate system common catalog. If blind paging is enabled, pagemate_send may also be used to submit a message directly to a PIN on the default network. pagemate_psend provides functionality equivalent to pagemate_send with the addition of reference and priority parameters.

ARGUMENTS

pagername

address of a null-terminated string or character array, 64 characters or less in length, specifying an individual subscriber, group or profile name, a compound subscriber/profile combination, or, if blind paging is enabled, a valid PIN for a pager registered with the default network. When an individual subscriber, group or profile name is specified, pagername is limited to 32 characters (including null termination). When a PIN is specified for blind paging, pagername is limited to 16 characters. In configurations served by a PAM Server, pagername can be a compound subscriber/profile specification, up to 64 characters in length, in the format "subscribername>profilename". When a compound subscriber/profile is specified, the message is submitted to PAM Server with instructions to deliver to the subscriber's pager (or similar portable message display device, including digital telephones) with response required pursuant to the profile.

message

address of a null-terminated string or character array, 500 characters or less in length, specifying the text or numeric message to send.

reference

address of a null-terminated string or character array, 16 characters or less in length, specifying a user or application-specific reference (e.g., trouble ticket or case number, application-specific sequence or message reference number).

priority

an integer value in the range 0 to 3, specifying message priority for page request. Priority 1 is high (most urgent), 2 is medium, 3 is low, and 0 (zero) means "use site-specific default priority".

error_buffer

address of a 160-character array into which pagemate_psend will write detailed error message information

RETURNS

message sequence number or longword condition value, as follows:

+n	positive integer sequence number of page successfully enqueued
0	failure; no such pagername/recipient
-1	failure; explanation or reason provided in error_buffer

NOTES

pagemate_psend is an extended parameters version of pagemate_send that can be called from any application program written in any language that supports the C calling standard.

Application programs that call pagemate_psend on Windows platforms must include /PageMate/Include/PageDLL.h and be linked with /PageMate/Lib/PGRPCDLL.lib and /PageMate/Lib/PGRPCDLL.dll.

Application programs that call pagemate_psend on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_psend on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

If the priority value specified in a call to pagemate_psend is anything other than 0, 1, 2 or 3, the site-specific default priority value will be used instead. In this case, message processing will proceed normally and no error message or status will be returned as a result of the invalid priority specification.

If the message is successfully enqueued, pagemate_psend returns a positive integer message sequence number as the value of the function. This sequence number can subsequently be passed to pagemate_status to obtain completion status for the page. If the message cannot be

enqueued for transmission, pagemate_psend returns a negative integer function value and a text description of the error or exception condition.

pagemate_send

FORMAT

pagemate_send (pagername, message, error_buffer)

DESCRIPTION

pagemate_send is an API function that submits a message for delivery to a subscriber, group or profile listed in the PageMate system common catalog. If blind paging is enabled, pagemate_send may also be used to submit a message directly to a PIN on the default network.

PARAMETERS

pagername

address of a null-terminated string or character array, 32 characters or less in length, specifying an individual subscriber, group or profile name, defined in the PageMate system common catalog, or, if blind paging is enabled, a valid PIN for a pager registered with the default network

message

address of a null-terminated string or character array, 500 characters or less in length, specifying the text or numeric message to send

error_buffer

address of a 160-character array into which pagemate_send will write detailed error message information

RETURNS

message sequence number or longword condition value, as follows:

+n	positive integer sequence number of page successfully enqueued
0	failure; no such pagername/recipient
-1	failure; explanation or reason provided in error_buffer

NOTES

Application programs that call pagemate_send on Windows platforms must include /PageMate/Include/PAGEDLL.h and be linked with /PageMate/Lib/PGRPCDLL.lib and /PageMate/Lib/PGRPCDLL.dll.

Application programs that call pagemate_send on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a.

On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_send on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

If the message is successfully enqueued, pagemate_send returns a positive integer message sequence number as the value of the function. This sequence number can subsequently be passed to pagemate_status to obtain completion status for the page. If the message cannot be enqueued for transmission, pagemate_send returns a negative integer function value and a text description of the error or exception condition.

pagemate_send_reply

FORMAT

```
pagemate_send_reply  (pagername, message,
reference, priority, reply_to, error_buffer)
```

DESCRIPTION

pagemate_send_reply is an API function that submits a message for delivery to a subscriber, group or profile listed in the PageMate system common catalog. If blind paging is enabled, pagemate_send_reply may also be used to submit a message directly to a PIN on the default network. pagemate_send_reply provides functionality equivalent to pagemate_send with the addition of reference, priority and reply_to parameters.

ARGUMENTS

pagername

address of a null-terminated string or character array, 64 characters or less in length, specifying an individual subscriber, group or profile name, a compound subscriber/profile combination, or, if blind paging is enabled, a valid PIN for a pager registered with the default network. When an individual subscriber, group or profile name is specified, pagername is limited to 32 characters (including null termination). When a PIN is specified for blind paging, pagername is limited to 16 characters. In configurations served by a PAM Server, pagername can be a compound subscriber/profile specification, up to 64 characters in length, in the format "subscribername>profilename". When a compound subscriber/profile is specified, the message is submitted to PAM Server with instructions to deliver to the subscriber's pager (or similar portable message display device, including digital telephones) with response required pursuant to the profile.

message

address of a null-terminated string or character array, 500 characters or less in length, specifying the text or numeric message to send.

reference

address of a null-terminated string or character array, 16 characters or less in length, specifying a user or application-specific reference (e.g., trouble ticket or case number, application-specific sequence or message reference number).

priority

an integer value in the range 0 to 3, specifying message priority for page request. Priority 1 is high (most urgent), 2 is medium, 3 is low, and 0 (zero) means "use site-specific default priority".

reply_to

address of a null-terminated string or character array, 128 characters or less in length, specifying an address to which replies from the message recipient should be directed by the messaging service. Functionality of reply_to varies by protocol and paging service provider and is supported in conjunction with PageMate Automated Messenger (PAM) Server only. This parameter is required but may be a null or empty string.

error_buffer

address of a 160-character array into which pagemate_psend will write detailed error message information

RETURNS

message sequence number or longword condition value, as follows:

+n	positive integer sequence number of page successfully enqueued
0	failure; no such pagername/recipient
-1	failure; explanation or reason provided in error_buffer

NOTES

pagemate_send-reply is an extended parameters version of pagemate_send that can be called from any application program written in any language that supports the C calling standard.

Application programs that call pagemate_send_reply on Windows platforms must include /PageMate/Include/PageDLL.h and be linked with /PageMate/Lib/PGRPCDLL.lib and /PageMate/Lib/PGRPCDLL.dll.

Application programs that call pagemate_send_reply on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_send_reply on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

If the priority value specified in a call to pagemate_send_reply is anything other than 0, 1, 2 or 3, the site-specific default priority value will be used instead. In this case, message processing will proceed normally

and no error message or status will be returned as a result of the invalid priority specification.

If the message is successfully enqueued, pagemate_send_reply returns a positive integer message sequence number as the value of the function. This sequence number can subsequently be passed to pagemate_status to obtain completion status for the page. If the message cannot be enqueued for transmission, pagemate_send_reply returns a negative integer function value and a text description of the error or exception condition.

pagemate_status

FORMAT

pagemate_status (sequence_number, error_buffer)

DESCRIPTION

pagemate_status returns processing status information for a previously submitted message.

ARGUMENTS

sequence_number

32-bit integer message sequence number as returned by pagemate_send, pagemate_psend, pagemate_send_reply, pagemate_direct or pagemate_direct_reply.

error_buffer

address of a 160-character array into which pagemate_status will write detailed error message information

RETURNS

longword status value, as follows:

-1	unable to complete request (e.g., communications failure)
0	invalid or unknown sequence number
+1	transmission complete; success
+2	message is enqueued
+3	message in process; starting
+4	message deleted before transmission
+5	message transmission failed or rejected

NOTES

Application programs that call pagemate_status on Windows platforms must include /PageMate/Include/PAGEDLL.h and be linked with /PageMate/Lib/Pgrpcdll.lib and /PageMate/Lib/Pgrpcdll.dll.

Application programs that call pagemate_status on UNIX platforms must include /usr/include/pgmapi.h and be linked with

/usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_status on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

pagemate_version

FORMAT

`pagemate_version (buffer)`

DESCRIPTION

pagemate_version returns PageMate software version information.

ARGUMENTS

`buffer`

address of a 40-character array into which pagemate_version will write text describing the current software version and build date

RETURNS

longword status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

Application programs that call `pagemate_version` on Windows platforms must include /PageMate/Include/PAGEDLL.h and be linked with /PageMate/Lib/Pgrpcdll.lib and /PageMate/Lib/Pgrpcdll.dll.

Application programs that call pagemate_status on UNIX platforms must include /usr/include/pgmapi.h and be linked with /usr/lib/libPage.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_status on OpenVMS platforms must include pagemate$examples:pgmapi.h and be linked with the shareable image pagemate$library:pagemateapi.exe.

Chapter

3

Administrator CLI

The PageMate Administrator Command Line Interface provides support for administrator functionality from scripts and command procedures on Windows, UNIX and OpenVMS.

PageMate Administrator Command Line Interface (CLI) is an application that provides capabilities to automate frequently-performed software product management operations through use of scripts and command procedures. Command line functionality is provided via the `admcmd` command. Command syntax is

`admcmd` ***`operation object [qualifiers]`***

where

`operation` is one of: `import, export, print` or `show`

`object` is specific to ***`operation`*** and is one of: `history, statistics, subscribers, groups, clients, license` or `version`

`qualifiers` is specific to ***`operation`*** and ***`object`*** as further described below.

Each element (***`operation`***, ***`object`*** and ***`qualifiers`***) in a command must be delimited from the preceding element by a space (blank). Qualifiers must be prefixed with forward slash.

import

Subscriber and group catalog records can be imported from a plain text (flat) file via the Administrator CLI. Required format for the plain text file for subscriber records is specified in Table 3-3 in the PageMate User's Guide. Required format for group records is specified in Table 3-4 in the PageMate User's Guide. An easy way to generate an example of the format for either subscriber or group records is to export a catalog created via the PageMate Administrator GUI.

Syntax for import subscribers and groups is

```
admcmd import subscribers /input=<filename>
```

and

```
admcmd import groups /input=<filename>
```

where

`<filename>` is a file name, optionally prefixed with a path specification. The `/input` qualifier is required.

Additional information about import functionality, including options to add, delete and update/replace records during import, is provided under Importing Subscriber Catalog Records and Importing Group Catalog Records in Chapter 3 of the PageMate User's Guide.

export

Subscriber and group catalog records can be exported via the Administrator CLI as described under Exporting Subscriber Catalog Records and Exporting Group Catalog Records in Chapter 3 of the PageMate User's Guide.

Syntax for export subscribers and groups is

```
admcmd export subscribers /output=<filename>
```

and

```
admcmd export groups /output=<filename>
```

where

`<filename>` is a file name, optionally prefixed with a path specification. The `/output` qualifier is optional. In the absence of specification, `<filename>` will default to `pagers.txt` and `groups.txt` (for subscribers and groups, respectively) in the current process default directory.

The Administrator CLI also provides capabilities for exporting message history and statistics.

export history

Syntax for export history is

```
admcmd export history /queue=<queue_name> /since=<tstamp>
/before=<tstamp>    /output=<filename>    /from=<username>
/to=<subscriber>
```

where

<*queue_name*> must be the name of a PageMate message queue. At the present time, the only supported queue name is `page`. The `/queue` qualifier is required.

<*tstamp*> is a date/time specification. When a <*tstamp*> value is provided, it must, at a minimum, be a date specification of the form dd-mmm-yyyy (*e.g.*, 05-JAN-2012). A <*tstamp*> specification may optionally include hours, minutes and seconds (*e.g.*, 05-JAN-2012:05:14:30). Qualifiers `/since` and `/before` are optional. When not specified, `/since` defaults to midnight (00:00:00) of the current date, and `/before` defaults to the end of the current day.

<*filename*> is a file name, optionally prefixed with a path specification. When a file name is specified with the `/output` qualifier, PageMate Administrator CLI will write its report output to the specified file. `/output` is an optional qualifier. In the absence of specification, PageMate Administrator CLI will write its report output to a file named `report.txt` in the current process default directory. Data provided in the report file are as described in `export.h` in the PageMate Include directory on the PageMate Server.

<*username*> is a user or PageMate subscriber name. When <*username*> is specified with the `/from` qualifier, PageMate Administrator CLI will report only messages submitted by (sent from) the specified username. `/from` is an optional qualifier. In the absence of specification, PageMate Administrator CLI will report messages from all sources.

<*subscriber*> is a PageMate subscriber name. When <*subscriber*> is specified with the `/to` qualifier, PageMate Administrator CLI will report only messages addressed to (sent to) the specified subscriber. `/to` is an optional qualifier. In the absence of specification, PageMate Administrator CLI will report messages to all subscribers.

export statistics

Syntax for export statistics is

```
admcmd export statistics /queue=<queue_name>
/since=<tstamp> /before=<tstamp> /output=<filename>
```

where

<*queue_name*> must be the name of a PageMate message queue. At the present time, the only supported queue name is `page`. The `/queue` qualifier is required. In conjunction with `/queue=page`, the qualifier `/sort=network` is supported to generate a report of page queue statistics by network. In the absence of `/sort=network`, the Administrator CLI will report page queue statistics organized by subscriber. In either case, only those subscribers or networks for which there was activity during the report period will be listed in the report.

<*tstamp*> is a date/time specification. When a <*tstamp*> value is provided, it must, at a minimum, be a date specification of the form dd-mmm-yyyy (*e.g.*, 05-JAN-2012). A <*tstamp*> specification may optionally include hours, minutes and seconds (*e.g.*, 05-JAN-2012:05:14:30). Qualifiers `/since` and `/before` are optional. When not specified, `/since` defaults to midnight (00:00:00) of the current date, and `/before` defaults to the end of the current day.

<*filename*> is a file name, optionally prefixed with a path specification. When a file name is specified with the `/output` qualifier, PageMate Administrator CLI will write its report output to the specified file. `/output` is an optional qualifier. In the absence of specification, PageMate Administrator CLI will write its report output to a file named `report.txt` in the current process default directory. Data provided in the report file are as described in `export.h` in the PageMate Include directory on the PageMate Server.

print

PageMate Administrator CLI provides capabilities to print subscriber and group catalog records, message history and message statistics to a file. Command syntax for print subscriber or group is

```
admcmd print subscriber /name=<string> /output=<filename>
```

and

```
admcmd print group /name=<string> /output=<filename>
```

where

<*string*> is a complete or wildcard catalog name. A complete catalog name is an alphanumeric string of up to 31 contiguous visible characters matching the name parameter in an existing PageMate catalog record. A wildcard catalog name is any alphanumeric string of up to 31 characters terminated with asterisk. The `/name` qualifier is required.

<*filename*> is the name of the file to which the CLI will write its output. The /output qualifier is optional. In the absence of specification, <*filename*> will default to report.txt in the current process default directory.

print history

Command syntax for print history is

```
admcmd print history /queue=<queue_name> /since=<tstamp>
/before=<tstamp>    /output=<filename>    /from=<username>
/to=<subscriber>
```

where

<*queue_name*> must be the name of a PageMate message queue. At the present time the only supported queue_name is page. The /queue qualifier is required.

<*tstamp*> is a date/time specification. When a <*tstamp*> value is provided, it must, at a minimum, be a date specification of the form dd-mmm-yyyy (*e.g.*, 05-JAN-2012). A tstamp specification may optionally include hours, minutes and seconds (*e.g.*, 05-JAN-2012:05:14:30). Qualifiers /since and /before are optional. When not specified, /since defaults to midnight (00:00:00) of the current date, and /before defaults to the end of the current day.

<*filename*> is a file name, optionally prefixed with a path specification. When a file name is specified with the /output qualifier, PageMate Administrator CLI will write its report output to the specified file. /output is an optional qualifier. In the absence of specification, PageMate Administrator CLI will write its report output to a file named report.txt in the current process default directory.

<*username*> is a user or PageMate subscriber name. When a username is specified with the /from qualifier, PageMate Administrator CLI will report only messages submitted by (sent from) the specified username. /from is an optional qualifier. In the absence of specification, PageMate Administrator CLI will report all messages.

<*subscriber*> is a PageMate subscriber name. When <*subscriber*> is specified with the /to qualifier, PageMate Administrator CLI will report only messages addressed to (sent to) the specified username. /to is an optional qualifier. In the absence of specification, PageMate Administrator CLI will report all messages.

print statistics

Command syntax for print statistics is

```
admcmd print statistics /queue=<queue_name>
  /since=<tstamp> before=<tstamp> /output=<filename>
```

where

`<queue_name>` must be the name of a PageMate message queue. At the present time, the only supported queue name is `page`. The `/queue` qualifier is required. In conjunction with `/queue=page`, the qualifier `/sort=network` is supported to generate a report of page queue statistics by network. In the absence of `/sort=network`, the Administrator CLI will report page queue statistics organized by subscriber. In either case, only those subscribers or networks for which there was activity during the report period will be listed in the report.

`<tstamp>` is a date/time specification. When a `<tstamp>` value is provided, it must, at a minimum, be a date specification of the form dd-mmm-yyyy (*e.g.*, 05-JAN-2012). A `<tstamp>` specification may optionally include hours, minutes and seconds (*e.g.*, 05-JAN-2012:05:14:30). Qualifiers `/since` and `/before` are optional. When not specified, `/since` defaults to midnight (00:00:00) of the current date, and `/before` defaults to the end of the current day.

`<filename>` is a file name, optionally prefixed with a path specification. When a filename is specified with the `/output` qualifier, PageMate Administrator CLI will write its report output to the specified file. `/output` is an optional qualifier. In the absence of specification, PageMate Administrator CLI will write its report output to a file named `report.txt` in the current process default directory.

show

PageMate Administrator CLI provides capabilities to display information about registered clients, software license and software version. Command syntax for show is

```
admcmd show <object>
```

where

object is one of: `clients`, `license` or `version`

show clients

Command syntax for show clients is

```
admcmd show clients
```

On servers that operate with a limited number of clients, show clients will display a list of registered PageMate clients. On servers that authorize unlimited clients, show clients will report "xxnnnnnn authorizes unlimited clients" where xxnnnnnn is the PageMate software license number registered on the server.

show license

Command syntax for show license is

```
admcmd show license
```

Show clients will display PageMate software license information, including license number, registered site name, and the expiration date for software maintenance and technical support as reflected in the currently installed software license.

show version

Command syntax for show version is

```
admcmd show version
```

Show version will display PageMate software version and build information for the installed PageMate Server.

Chapter

4

Administrator API

PageMate's Administrator API provides support for catalog management and reporting from site-specific applications.

The PageMate Administrator (API) provides capabilities to maintain system catalogs and report message queue statistics from site-specific applications running on a PageMate Server. There are two sets of functions supported under the PageMate Administrator API. Those referred to as common administrator functions are supported in both PageMate Classic Server and PageMate Automated Messenger Server environments. A second set of functions, referred to as web administrator functions, are supported only in PageMate Automated Messenger Server environments that include license for the PageMate Web Connector.

- ✓ Common administrator functions provide capabilities to maintain the PageMate subscriber catalog (*i.e.,* `pagers.dat`) and report message statistics and software version. Common administrator functions are supported in all server environments (on both PageMate Classic Server and PageMate Automated Messenger Server). Common administrator functions include `pagemate_add`, `pagemate_delete`, `pagemate_edit`, `pagemate_get`, `pagemate_qstats` and `pagemate_version`.

- ✓ Web administrator functions provide both client and administrative functionality derived from the PageMate Web Connector on PageMate Automated Messenger Servers. Web administrator functions include `pam_login`, `pam_logout`, `pam_catalog_add`, `pam_catalog_delete`, `pam_catalog_edit`, `pam_catalog_list`, `pam_catalog_find`, `pam_message_send`, `pam_message_direct`, `pam_message_list`, `pam_message_get`, `pam_message_status` and `pam_version`.

In PAM server environments, web administrator functions duplicate and extend the functionality provided by both the client API and common administrator functions in PageMate Classic server environments. The client function `pagemate_send`, for example, may be called from an application program to enqueue a text or numeric message for transmission. The client function `pagemate_psend` is an extended parameters version of `pagemate_send`, including support for message priority and a user-defined message reference. In

PAM server environments, the web administrator function `pam_message_send` provides the functionality of `pagemate_send` and `pagemate_psend` combined in a single function.

Similarly, the client API function `pagemate_direct` provides a capability to send messages to unlisted recipients (*i.e.*, to recipients not listed in the PageMate subscriber catalog). In PAM server environments, the web administrator function `pam_message_direct` provides equivalent functionality. The client function `pagemate_status` may be called to obtain the completion status of a message. In PAM server environments that include the Web Connector, the web administrator function `pam_message_status` provides equivalent functionality.

Administrator functions provide capabilities to maintain the PageMate subscriber catalog. Web administrator functions provide this capability as well, together with capabilities to maintain group catalogs and perform other functions derived from the PageMate Web Connector.

The PageMate API has been written primarily in and for applications written in C/C++. If your application is written in C/C++, you can use the C header files provided in the PageMate include directory (*e.g.*, ~\Program Files\PageMate\Include) on your system to provide prototype definitions for PageMate API functions.

If your application is written in Visual Basic, you will need to provide declarations for the PageMate API functions you want to use. Your declarations will need to specify arguments as in the example below. In addition, if the PageMate Library directory will not be specified by the Path environment variable when your application executes, you will need to use the `Lib` clause in your `Declare` statement to specify where the applicable PageMate DLL is located on your system. An example declaration for `pagemate_send` is shown immediately below.

```
Declare Function pagemate_send _
Lib "c:\Program Files\PageMate\Lib\PGRPCDLL.dll" _
ByVal PagerName As String, ByVal Message as String, _
ByVal ErrorBuffer as String) As Long
```

If you place the Declare statement in a `Form` or `Class` module, you should precede it with the `Private` keyword. After the API function is declared, you can call it just as you would any Visual Basic function. An example (excerpt) from Visual Basic code to call `pagemate_send` is shown below.

```
Dim MessageSeqn As Long Dim PagerID As String
Dim Message As String
Dim ErrorBuffer As String * 160
PagerID = "TOM_JONES"
```

```
Message = "Test Message"
MessageSeqn = pagemate_send(PagerID, Message, ErrorBuffer)
```

PageMate API functions are listed and described in alphabetical order by function name on the following pages in this section. Functions that are supported in both PageMate Classic and PAM server environments have names that begin with "pagemate", while those that are specific to PAM Web Server environments have names that begin with "pam".

pagemate_add

FORMAT

`pagemate_add (pager_record, error_buffer)`

DESCRIPTION

pagemate_add adds a subscriber record to the PageMate system common catalog.

ARGUMENTS

`pager_record`

address of a pager record structure, as defined by pager_record in pgmapi.h

`error_buffer`

address of a 160-character array into which pagemate_add will write detailed error message information as required

RETURNS

longword integer status value, as follows:

+1 success; record has been added to system common catalog

-1 failure; explanation or reason provided in error_buffer

NOTES

pagemate_add is supported on PageMate server systems only.

Application programs that call pagemate_add on Windows platforms must include /PageMate/Include/AdmAPI.h and be linked with /PageMate/Lib/AdmAPI.lib and /PageMate/Lib/AdmAPI.dll.

Application programs that call pagemate_add on UNIX platforms must include /usr/include/admapi.h and be linked with /usr/lib/libPageAdm.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_add on OpenVMS platforms must include pagemate$examples:admapi.h and be linked with the shareable image pagemate$library:pagemateadm.exe.

pagemate_delete

FORMAT

`pagemate_delete  (pagername, error_buffer)`

DESCRIPTION

pagemate_delete deletes a subscriber record from the PageMate system common catalog.

ARGUMENTS

`pagername`

address of a null-terminated string or character array, 32 characters or less in length, specifying the name (pager_id) of the pager record to delete from the system common catalog

`error_buffer`

address of a 160-character array into which pagemate_delete will write detailed error message information as required

RETURNS

longword integer status value, as follows:

+1 success; record has been deleted from the system common catalog

-1 failure; explanation or reason provided in error_buffer

NOTES

pagemate_delete is supported on PageMate server systems only.

Application programs that call pagemate_delete on Windows platforms must include /PageMate/Include/AdmAPI.h and be linked with /PageMate/Lib/AdmAPI.lib and /PageMate/Lib/AdmAPI.dll.

Application programs that call pagemate_delete on UNIX platforms must include /usr/include/admapi.h and be linked with /usr/lib/libPageAdm.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_delete on OpenVMS platforms must include pagemate$examples:admapi.h and be linked with the shareable image pagemate$library:pagemateadm.exe.

pagemate_edit

FORMAT

`pagemate_edit  (pager_record, error_buffer)`

DESCRIPTION

pagemate_edit updates (replaces) a subscriber record in the PageMate system common catalog.

ARGUMENTS

`pager_record`

address of a pager record structure, as defined by pager_record in pgmapi.h

`error_buffer`

address of a 160-character array into which pagemate_edit will write detailed error message information as required

RETURNS

longword integer status value, as follows:

+1 success; record has been updated in the system common catalog

-1 failure; explanation or reason provided in error_buffer

NOTES

pagemate_edit is supported on PageMate server systems only.

Application programs that call pagemate_edit on Windows platforms must include /PageMate/Include/AdmAPI.h and be linked with /PageMate/Lib/AdmAPI.lib and /PageMate/Lib/AdmAPI.dll.

Application programs that call pagemate_edit on UNIX platforms must include /usr/include/admapi.h and be linked with /usr/lib/libPageAdm.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_edit on OpenVMS platforms must include pagemate$examples:admapi.h and be linked with the shareable image pagemate$library:pagemateadm.exe.

pagemate_get

FORMAT

```
pagemate_get  (context, pagername,
pager_record, error_buffer)
```

DESCRIPTION

pagemate_get finds and returns to the caller a subscriber record from the PageMate system common catalog.

ARGUMENTS

context

address of longword integer in which pagemate_get will return an opaque value providing context for the next record in the pager catalog (see NOTES)

pagername

address of a null-terminated string or character array, 32 characters or less in length, specifying the name (pager_id) of the pager record to get from the system common catalog (see NOTES)

pager_record

address of a pager record structure, as defined by pager_record in pgmapi.h

error_buffer

address of a 160-character array into which pagemate_get will write detailed error message information as required

RETURNS

longword integer status value, as follows:

+1	success; record data is returned in pager_record
0	failure; no such record, end of file or no more records
-1	failure; explanation or reason provided in error_buffer

NOTES

pagemate_get is supported on PageMate server systems only.

Application programs that call pagemate_get on Windows platforms must include /PageMate/Include/AdmAPI.h and be linked with /PageMate/Lib/AdmAPI.lib and /PageMate/Lib/AdmAPI.dll.

Application programs that call pagemate_get on UNIX platforms must include /usr/include/admapi.h and be linked with /usr/lib/libPageAdm.a. On some UNIX platforms, one or more of the library archives /usr/lib/libdce.a and /usr/lib/libm.a may be required.

Application programs that call pagemate_get on OpenVMS platforms must include pagemate$examples:admapi.h and be linked with the shareable image pagemate$library:pagemateadm.exe.

To obtain the catalog record for a specific (known) pagername, set context=0 and specify the name (pager_id) of the pager record in pagername. The catalog record, if found, will be returned in pager_record.To retrieve all records in the system common pager catalog, first call pagemate_get with context=0 and pagername=* (a null-terminated string containing asterisk alone). pagemate_get will return the first record in the pager catalog and provide a value in context that may be used to obtain the next record. On each successive call, pass to pagemate_get the values in context and pagername that were returned in the previous call.

pagemate_version

FORMAT

`pagemate_version (buffer)`

DESCRIPTION

pagemate_version returns PageMate software version information.

ARGUMENTS

`buffer`

address of a 40-character array into which pagemate_version will write text describing the current software version and build date

RETURNS

longword status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

Support for `pagemate_version` is provided in the PageMate Client API and in both variants of Administrator API.

Application programs that call `pagemate_version` from within the Administrator API on Windows platforms should include /PageMate/Include/AdmAPI.h and be linked with /PageMate/Lib/AdmAPI.lib and /PageMate/Lib/AdmAPI.dll.

Application programs that call pagemate_version on UNIX platforms must include /usr/include/admapi.h and be linked with /usr/lib/libPageAdm.a.

Application programs that call pagemate_version on OpenVMS platforms must include pagemate$examples:admapi.h and be linked with the shareable image pagemate$library:pagemateadm.exe.

pam_catalog_add

FORMAT

pam_catalog_add (objectname, recordtype, token, record, buffer)

DESCRIPTION

pam_catalog_add adds a subscriber or group record to the PageMate system common catalog on a PAM server system.

ARGUMENTS

objectname

address of character array containing a null-terminated string, 32 characters or less in length, specifying a subscriber or group name to be added to the PageMate catalog

recordtype

address of character array containing one of: "subscriber" or "group" (sans quotation marks)

token

address of character string returned by `pam_login`

record

address of character array containing the record to be added to the catalog, as appropriate per recordtype (see NOTES)

buffer

address of a 160-character array into which `pam_catalog_add` will write a text message describing the result of the operation

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `buffer`

NOTES

`pam_login` must be called to obtain token before calling this function.

If recordtype=subscriber, record must specify the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

user_registration_name

employee_number

company_department

home_server_name

pager_network_name

pager_type

PIN

voice_phone_1 (primary voice phone)

voice_phone_2 (1st alternate voice phone)

voice_phone_3 (2nd alternate voice phone)

fax_phone

forward_to (subscriber name)

email_address

authority_name

password

priority

comments

If recordtype=group, record must specify the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

description (63 characters)

comments (599 characters)

number_of_members

member_name_1 (1st subscriber_name)

member_name_2 (2nd subscriber_name)

(*etc.* until number_of_members have been specified)

See example program (`pamapi_test.c`) provided in PageMate Examples directory for an example API call to `pam_catalog_find` to display a sample catalog record. Application programs that call `pam_catalog_add` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_catalog_delete

FORMAT

`pam_catalog_delete (objectname, recordtype, token, buffer)`

DESCRIPTION

pam_catalog_delete deletes a subscriber or group record from the PageMate system common catalog on a PAM server system.

ARGUMENTS

`objectname`

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying the name to be deleted from the PageMate catalog

`recordtype`

address of character array containing one of: "subscriber" or "group" (sans quotation marks)

`token`

address of character string returned by `pam_login`

`buffer`

address of a 160-character array into which `pam_catalog_add` will write a text message describing the result of the operation

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `buffer`

NOTES

`pam_login` must be called to obtain token before calling this function.

Application programs that call `pam_catalog_delete` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_catalog_edit

FORMAT

pam_catalog_edit (objectname, recordtype, token, record, buffer)

DESCRIPTION

pam_catalog_edit replaces a subscriber or group record in the PageMate system common catalog on a PAM server system.

ARGUMENTS

objectname

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying the name of the record to be edited in the PageMate catalog

recordtype

address of character array containing one of: "subscriber" or "group" (sans quotation marks)

token

address of character string returned by `pam_login`

record

address of character array containing the record to be edited in the catalog, as appropriate per recordtype (see NOTES)

buffer

address of a 160-character array into which `pam_catalog_edit` will write a text message describing the result of the operation

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `buffer`

NOTES

`pam_login` must be called to obtain token before calling this function.

If recordtype=subscriber, record must specify the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

record_key (from prior call to `pam_catalog_find`)

user_registration_name

employee_number

company_department

home_server_name

pager_network_name

pager_type

PIN

voice_phone_1 (primary voice phone)

voice_phone_2 (1st alternate voice phone)

voice_phone_3 (2nd alternate voice phone)

fax_phone

forward_to (subscriber name)

email_address

authority_name

password

priority

comments

If recordtype=group, record must specify the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

record_key (from prior call to `pam_catalog_find`)

description

owner_employee_number

company_department

forward_to (group name)

comments

number_of_members

member_name_1 (1st subscriber_name)

member_name_2 (2nd subscriber_name)

(*etc.* until number_of_members have been specified)

See example program (`pamapi_test.c`) provided in PageMate Examples directory for an example API call to `pam_catalog_find` to display a sample catalog record. Application programs that call `pam_catalog_edit` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_catalog_find

FORMAT

`pam_catalog_find (objectname, recordtype, token, record)`

DESCRIPTION

pam_catalog_find locates and returns a subscriber or group record from the PageMate system common catalog on a PAM server system.

ARGUMENTS

`objectname`

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying the name to find in the PageMate catalog

`recordtype`

address of null-terminated character array containing one of: "subscriber" or "group" (sans quotation marks)

`token`

address of character string returned by `pam_login`

`record`

address of character pointer to be returned by `pam_catalog_find` pointing to memory allocated by `pam_catalog_find` per `recordtype` (see NOTES). Caller must free this memory.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

`pam_login` must be called to obtain token before calling this function.

If recordtype=subscriber, memory pointed to by record will contain the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

subscriber_name

recordtype ("subscriber")

user_registration_name
employee_number
company_department
home_server_name
pager_network_name
pager_type
pager_pin
voice_phone_1 (primary voice phone)
voice_phone_2 (1st alternate voice phone)
voice_phone_3 (2nd alternate voice phone)
fax_phone
forward_to (subscriber name)
email_address
authority_name
password
priority
comments
authority_token
pager_status
access_code
pin_type
function_code
send_type
page_class
rf_channel
rf_zone
record_key

If recordtype=group, memory pointed to by record will contain the following parameters delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124):

group_name
recordtype ("group")
description
owner_employee_number
company_department

forward_to

comments

number_of_members

authority_mask

record_key

member_name_1 (1st subscriber_name)

member_name_2 (2nd subscriber_name)

(*etc.* until number_of_members have been listed)

An application that calls `pam_catalog_find` is responsible for freeing the memory returned via `record`. See example program (`pamapi_test.c`) in the PageMate Examples directory for an example API call to `pam_catalog_find`.

Application programs that call `pam_catalog_find` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_catalog_list

FORMAT

pam_catalog_list (objectname, recordtype, token, nameslist)

DESCRIPTION

pam_catalog_list returns a list of subscriber, group and profile records with names matching a specified wildcard lookup name string.

ARGUMENTS

objectname

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying a wildcard name for lookup in the PageMate catalog. The last visible character in objectname must be asterisk (*).

recordtype

address of character array containing one of: “subscriber”, “group”, “profile” or “any” (sans quotation marks). Recordtype specifies the type of records to return. Recordtype “any” indicates a request to return all subscriber, group and profile names matching the wildcard specification in objectname.

token

address of character string returned by `pam_login`

nameslist

address of character pointer to be returned by `pam_catalog_list` pointing to memory allocated by `pam_catalog_list` containing results of the catalog lookup (see NOTES). Caller must free this memory.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

`pam_login` must be called to obtain token before calling this function.

On return from `pam_catalog_list`, nameslist will contain a string of value pairs delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124) as follows:

objectname1 | description1 | objectname2 | description 2 | …

The caller of `pam_message_list` must free nameslist. An example of a command line application that calls PamAPI functions is provided in the /PageMate/Examples directory on every PAM server system.

Application programs that call `pam_catalog_list` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_login

FORMAT

`pam_login (subscribername, password, token)`

DESCRIPTION

pam_login obtains authorization for an application program to call other PAM API functions (API functions with names of the form pam_*xxxx*). pam_login validates subscribername and password parameters and returns a token that the application program must pass to other PAM API functions.

ARGUMENTS

`subscribername`

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying a subscribername from the PageMate catalog

`password`

address of character array containing a null-terminated string, 16 characters or less in length (including null terminator), specifying the password for subscribername

`token`

address of 32-character array into which `pam_login` will write a token (opaque string) value. This value must be provided as input to most other PamAPI functions to authenticate the user.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

Application programs that call `pam_login` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_logout

FORMAT

`pam_logout (token)`

DESCRIPTION

pam_logout invalidates the token passed as an input parameter, thereby terminating the authorization for the calling application to make further PAM API calls.

ARGUMENTS

`token`

address of character string returned by `pam_login`

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

`pam_login` must be called to obtain token before calling this function. Calling `pam_logout` invalidates the token provided as input.

Application programs that call `pam_logout` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_message_direct

FORMAT

```
pam_message_direct  (sender, network,
pagertype, pin, message, priority, reference,
reply_to, token, buffer)
```

DESCRIPTION

pam_message_direct submits a message for delivery to a pager, cell phone or similar mobile device that is served by a listed network but not listed in a subscriber record in the PageMate system common catalog.

ARGUMENTS

sender

address of character array containing a null-terminated string, 32 characters or less in length (including null terminator), specifying the name of the subscriber submitting (sending) the message. If sender is null or if the current user process does not have OPERATOR or higher authority, the message will be sent from the subscriber name associated with the current user process.

network

address of null-terminated character array specifying the name of the network to which the message should be submitted. If network is null, the message will be submitted to the default network (the first network in networks.dat).

pagertype

address of character array containing one of: "text" or "numeric" (sans quotation marks), specifying the pagertype of the recipient device.

pin

address of null-terminated character array specifying the PIN (pager identification number, a.k.a. personal identification number) of the recipient device.

message

address of null-terminated character array, 500 characters or less in length, containing the message to be sent

priority

address of null-terminated character array, 2 characters in length (including null terminator), specifying message priority in the range 0

through 3. Priority 0 specifies default (typically 3). Other priorities are 1 (highest), 2 (medium) and 3 (low).

`reference`

address of null-terminated character array, 16 characters or less in length, specifying a text reference to be associated with the message.

`reply_to`

address of a string or null-terminated character array, up to 128 characters in length, specifying the network address to which replies and responses to the message should be sent (see NOTES)

`token`

address of character string returned by `pam_login`

`buffer`

address of a 160-character array into which `pam_message_direct` will, in the event it cannot complete the operation successfully, write a text message describing the error or problem that prevented successful completion.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `buffer`

NOTES

`pam_login` must be called to obtain token before calling this function.

The `reply_to` parameter is protocol-specific, and the functionality it provides can vary among different paging service providers. For messages sent via a WCTP network, `reply_to` is equivalent to the WCTP `sendResponsesToID` parameter.

In the absence of a specification for `reply_to`, a WCTP paging service will send replies and responses to PageMate at the address specified by WCTP_PATH on the PAM Server that submitted the original message. Replies that are returned to PageMate will be interpreted as positive or negative acknowledgment of the original message, and can trigger execution of other action records under a profile.

If you want a WCTP or other compatible two-way paging service to send replies and responses directly to a network address other than PageMate on the PAM Server that submitted the original message, you should specify via `reply_to` the target network address. If the `reply_to` character string contains rate sign (e.g., 8001234567@mytelco.com), it will be assumed to be an e-mail address.

Application programs that call `pam_message_direct` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_message_get

FORMAT

pam_message_get (queuename, sequence_number, token, result, buffer)

DESCRIPTION

pam_message_get locates and returns information about a message previously submitted to a PAM server system.

ARGUMENTS

queuename

address of null-terminated character array specifying the name of the queue to be queried. The name of the message queue is "page" (sans quotation marks).

sequence_number

address of null-terminated character array specifying the sequence number of the message to be retrieved

token

address of character string returned by `pam_login`

result

address of character pointer to be returned by `pam_message_get` pointing to memory allocated by `pam_message_get` containing message data (see NOTES). Caller must free this memory.

buffer

address of a 160-character array into which `pam_message_get` will, in the event it cannot complete the operation successfully, write a text message describing the error or problem that prevented successful completion.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `result`

NOTES

`pam_login` must be called to obtain token before calling this function.

`pam_message_get` returns message data via result in a delimited string formatted as follows:

from (sender) | to (recipient) | timestamp | status | priority | reference | message_body

`pam_message_get` returns via result a string delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124). The caller of `pam_message_get` must free this memory. An example of a command line application that calls PamAPI functions is provided in the /PageMate/Examples directory on every PAM server system.

Application programs that call `pam_message_get` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_message_list

FORMAT

pam_message_list (queuename, sender, recipient, report_type, reference, startstamp, endstamp, token, report, buffer)

DESCRIPTION

pam_message_list compiles and returns from message history a list of messages that satisfy user-specified criteria. Any combination of sender, recipient, report_type, reference and time window (from startstamp to endstamp) can be specified as filter criteria. If not otherwise specified, sender, recipient and reference will default to wildcard (asterisk). In the absence of specification, startstamp will default to 00:00 of the current day and endstamp will default to the current system time. report_type and token are required input parameters. The compiled list is returned via report.

ARGUMENTS

queuename

address of null-terminated character array specifying the name of the message history queue to be queried. The name of the message queue is "page" (sans quotation marks).

sender

address of null-terminated character array specifying a PageMate subscriber name, if any, to be applied as a sender filter parameter in the selection of messages to include in the list. In the absence of specification, sender will default to wildcard (asterisk).

recipient

address of null-terminated character array specifying a PageMate subscriber name, if any, to be applied as a recipient filter parameter in the selection of messages to include in the list. In the absence of specification, recipient will default to wildcard (asterisk).

report_type

address of null-terminated character array having syntax in the form "<parameter_name>=<value>" (no embedded spaces). To compile a report of message history, <parameter_name> must be "status" and <value> must be a comma-delimited list of one or more of the status values "enqueued", "starting", "completed", "failed", "deleted",

"requeued" and "all". To obtain a list of all messages that have completed with any status, for example, specify "status=completed,failed,deleted,requeued".

An alternate use of report_type is provided when <parameter_name> is "statistics" and <value> is one of "totals", "network", "port" or "subscriber". This alternative provides a report of message statistics for the specified time period sorted on the basis of the <value> specification.

reference

address of null-terminated character array specifying a reference parameter, if any, to be applied as a filter parameter in the selection of messages to report.

startstamp

address of null-terminated character array specifying a start time for the time window from which to select messages for reporting. startstamp can be a calendar date, a time (based on 24-hour clock time), or a date and time.

endstamp

address of null-terminated character array specifying a start time for the time window from which to select messages for reporting. endstamp can be a calendar date, a time (based on 24-hour clock time), or a date and time.

token

address of character string returned by `pam_login`

report

address of character pointer to be returned by `pam_message_list` pointing to memory allocated by `pam_message_list` containing message data (see NOTES). Caller must free this memory.

buffer

address of a 160-character array into which `pam_message_get` will, in the event it cannot complete the operation successfully, write a text message describing the error or problem that prevented successful completion.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `result`

NOTES

`pam_login` must be called to obtain token before calling this function.

When report_type is status, `pam_message_list` returns via report a list of messages in a delimited string formatted as follows:

from (sender) | to (recipient) | timestamp | status | priority | reference | message_body

`pam_message_list` returns via result a string delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124). The caller of `pam_message_list` must free this memory. An example of a command line application that calls PamAPI functions is provided in the /PageMate/Examples directory on every PAM server system.

Application programs that call `pam_message_list` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_message_send

FORMAT

pam_message_send (objectname, recordtype, sender, message, priority, reference, reply_to, token, buffer)

DESCRIPTION

pam_message_send submits a message for delivery to a subscriber, group or profile listed in the system common catalog on a PAM server system.

ARGUMENTS

objectname

address of null-terminated character array specifying a subscriber, group or profile name listed in the PageMate catalog

recordtype

address of null-terminated character array containing one of: "subscriber", "group" or "profile" (lowercase, sans quotation marks), specifying the record type for `objectname`

sender

address of null-terminated character array, 32 characters or less in length, specifying the name of the subscriber submitting (sending) the message. If sender is null or if the current user process does not have OPERATOR or higher authority, the message will be sent from the subscriber name associated with the current user process.

message

address of null-terminated character array, 500 characters or less in length, containing the message to be sent

priority

address of null-terminated character array, 2 characters in length (including null terminator), specifying message priority in the range 0 through 3. Priority 0 specifies default (typically 3). Other valid priorities are 1 (highest), 2 (medium) and 3 (low).

reference

address of null-terminated character array, 16 characters or less in length, specifying a text reference to be associated with the message.

reply_to

address of a string or null-terminated character array, up to 128 characters in length, specifying the network address to which replies and responses to the message should be sent (see NOTES)

token

address of character string returned by pam_login

buffer

address of a 160-character array into which pam_message_send will write a text message describing the result of the operation

RETURNS

integer status value, as follows:

+n	sequence number of message successfully enqueued
-1	requested operation could not be completed; see buffer

NOTES

pam_login must be called to obtain token before calling this function.

The reply_to parameter is protocol-specific, and the functionality it provides can vary among different paging service providers. For messages sent via a WCTP network, reply_to is equivalent to the WCTP sendResponsesToID parameter.

In the absence of a specification for reply_to, a WCTP paging service will send replies and responses to PageMate at the address specified by WCTP_PATH on the PAM Server that submitted the original message. Replies that are returned to PageMate will be interpreted as positive or negative acknowledgment of the original message, and can trigger execution of other action records under a profile (see p.**Error! Bookmark not defined.**).

If you want a WCTP or other compatible two-way paging service to send replies and responses directly to a network address other than PageMate on the PAM Server that submitted the original message, you should specify via reply_to the target network address. If the reply_to character string contains rate sign (e.g., 8001234567@mytelco.com), it will be assumed to be an e-mail address.

Application programs that call pam_message_send on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_message_status

FORMAT

```
pam_message_status  (queuename,
sequence_number, token, result)
```

DESCRIPTION

pam_message_status returns status information for a message previously submitted on a PAM server system.

ARGUMENTS

queuename

address of null-terminated character array specifying the name of the queue to be queried. The name of the message queue is "page" (sans quotation marks).

sequence_number

address of null-terminated character array specifying the sequence number of the message whose status is to be returned

token

address of character string returned by `pam_login`

result

address of character pointer to be returned by `pam_message_status` pointing to memory allocated by `pam_message_status` containing message data (see NOTES). Caller must free this memory.

RETURNS

integer status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed; see `result`

NOTES

`pam_login` must be called to obtain token before calling this function.

If `pam_message_status` completes successfully, it will return via result a delimited string formatted as follows:

from (sender) | to (recipient) | timestamp | status | description |

In the result string, status (the 4th parameter in the result string) is a single word or term describing completion status (*e.g.*, "Success",

"Failed" or "Deleted"). Description is a phrase or sentence that provides additional detail about the completion status (*e.g.*, "Busy retry limit exceeded").

`pam_message_status` returns via result a string delimited by vertical bar (a.k.a. pipe character, ASCII decimal value 124). The caller of `pam_message_status` must free this memory. An example of a command line application that calls PamAPI functions is provided in the /PageMate/Examples directory on every PAM server system.

Application programs that call `pam_message_status` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

pam_version

FORMAT

`pam_version  (buffer)`

DESCRIPTION

pam_version returns PageMate software version information on a PAM server system.

ARGUMENTS

`buffer`

address of a 40-character array into which pam_version will write text describing the current software version and build date

RETURNS

longword status value, as follows:

+1 requested operation completed successfully

-1 requested operation could not be completed

NOTES

Application programs that call `pam_version` on Windows platforms must include /PageMate/Include/PamAPI.h and be linked with /PageMate/Lib/PamAPI.lib and /PageMate/Lib/PamAPI.dll.

Chapter

5

XML RPC Interface

XML RPC is one of the Remote Procedure Call variants that PageMate uses internally to support client-server communications. In lieu of using PageMate clients, a site that is licensed for unlimited clients or the PageMate Web Connector and wants to implement its own remote clients may call PageMate XML RPC functions directly from its own applications.

Extensible Markup Language, commonly known as XML, has been used since circa 1998 as an alternative to DCE RPC and ONC RPC to implement remote procedure calls over TCP/IP. A remote procedure call is a component of client-server architecture that allows a program or procedure running on one system, referred to as a client, to execute a function contained in a program or procedure on another system, referred to as a server, and obtain from the server results from the function execution returned to the client system.

In the context of PageMate as a client-server application, a program or procedure running on a system other than the PageMate Server can use a remote procedure call to execute a function on the server to retrieve catalog records, submit a message to the PageMate message queue, or inquire about the status of a message submitted at some earlier time.

A program or procedure running on a remote client, for example, can call an XML RPC function named SendPage, providing with the call a variety of data including a subscriber name and message text. When the SendPage function executes on the PageMate Server, the message text provided by the client will be enqueued for delivery to the named subscriber, and a message queue sequence number will be returned to the client. A message queue sequence number is a type of receipt or reference parameter that the client can subsequently use to inquire about the delivery status of the message.

When XML RPC is implemented over HTTP, XML payloads are typically wrapped in HTTP headers that are used to route the XML RPC traffic to appropriate processing code. PageMate uses an implementation of XML RPC that uses a dedicated port (a TCP stream socket) on the PageMate Server in lieu of HTTP-

based routing. The dedicated port used for XML RPC on a PageMate Server is specified via the XML_PORT site-specific parameter on the PageMate Server.

In PageMate XML RPC, a call or request transaction from a client to the PageMate Server is comprised of an 8-byte packet specifying the length of the XML request packet to follow (expressed as a decimal number in plain text, null-filled to a length of 8 bytes), followed by the request packet. In response, the PageMate Server sends two packets, the first being an 8-byte packet specifying the length of the XML response packet to follow, followed by the XML response (payload) packet.

PageMate XML RPC functions that can be called from site-specific clients are described in summary on following pages in this chapter. By way of example, operation of one of the most common and frequently called XML RPC functions, the SendPage function, is illustrated in detail below. In this example, `<nul>` represents a single null (binary zero) byte.

Packet #1: Client sends

```
835<nul><nul><nul><nul><nul>
```

Packet #2: Client sends

```
<?xml version="1.0"?>
<methodCall>
  <methodName>SendPage</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>Pager_Name</name>
            <value><string>SAMMY</string></value>
          </member>
          <member>
            <name>Network_Name</name>
            <value>
              <string>USA_Mobility</string>
            </value>
          </member>
          <member>
            <name>Pager_Type</name>
            <value><string>Text</string></value>
          </member>
          <member>
            <name>Pin</name>
```

```
            <value><string>1234567</string></value>
          </member>
          <member>
            <name>Message_Text</name>
            <value>
              <string>Hello World </string>
            </value>
          </member>
          <member>
            <name>Sender_Name</name>
            <value><string>Frank</string></value>
          </member>
          <member>
            <name>Sender_Node</name>
            <value><string>FIREBOLT</string></value>
          </member>
          <member>
            <name>msg_reference</name>
            <value><string/></value>
          </member>
          <member>
            <name>msg_priority</name>
            <value><int>3</int></value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

Packet #3: Server sends

```
375<nul><nul><nul><nul><nul>
```

Packet #4: Server sends

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>page_sequence_number</name>
```

```
            <value><int>2</int></value>
          </member>
          <member>
            <name>Error_Message</name>
            <value>
              <string>Message seqn 2 enqueued at
              09:02:56</string>
            </value>
          </member>
          <member>
            <name>return_value</name>
            <value><int>1</int></value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodResponse>
```

GetCatalogRecord

DESCRIPTION

GetCatalogRecord returns to an XML RPC client a subscriber record from the PageMate catalog on the server. Except as otherwise specified, all string parameters must be null-terminated single-byte (16 bits per character) ASCII character arrays.

REQUEST PARAMETERS

`context`

A numeric string value providing a positional reference for the record in the PageMate catalog. Context should be initialized to zero when requesting a specific record or when initiating a wildcard request for a series of records. In its response, the server will return a non-zero context value. When requesting a series of records, each request from the client should provide to the server the value returned in response to the previous call in the sequence.

`catalog`

A numeric string value specifying the catalog from which the client is requesting a record. Specify SYSTEM_CATALOG value as defined in XMLRPC.h in the PageMate\Include directory on a PageMate Server.

`Pager_Name`

An alphanumeric string, up to 31 characters in length, specifying the complete or partial (wildcard-terminated) subscriber name of the record to be returned. To request a specific record, specify the complete subscriber name. To request a series of records with names matching a wildcard-terminated string, specify a partial subscriber name terminating with the asterisk wildcard character (*e.g.*, AB* to request all subscriber names beginning with AB, or asterisk alone to request all subscriber records). A single record will be returned on each call until there are no more records matching the wildcard-terminated search string. When requesting a record to be returned based on Pager_Name specification, the User_Name parameter should be null (*i.e.*, an empty string).

`User_Name`

An alphanumeric string, up to 63 characters in length, specifying the complete or partial (wildcard-terminated) user registration name of the record to be returned. To request a specific record, specify the complete user registration name. To request a series of records with names

matching a wildcard-terminated string, specify a partial user registration name terminating with the asterisk wildcard character (*e.g.*, AB* to request all records with user registration names beginning with AB, or asterisk alone to request all subscriber records). A single record will be returned on each call until there are no more records matching the wildcard-terminated search string. When requesting a record to be returned based on User_Name specification, the Pager_Name parameter should be null (*i.e.*, an empty string).

RESPONSE PARAMETERS

`return_value`

A numeric string completion status value for the request which, if positive, indicates successful completion and, if negative, indicates failure. Failure indicates that all other response parameters except Error_Message should be ignored. In the case of failure, if the request was for a specific record, the record could not be found, and if a series of records was being requested through successive wildcard lookup requests, there are no more records available satisfying the request criteria.

`context`

A numeric value providing a positional reference for the record in the PageMate catalog. For successful completions, the server will return a non-zero value. When requesting a series of records, each request from the client should provide to the server the value returned in response to the previous call in the sequence.

`catalog`

A numeric value specifying the catalog from which the client is requesting a record. When requesting a series of records, each request from the client should provide to the server the value returned in response to the previous call in the sequence.

`Pager_Record`

A single catalog record is returned in a structure as specified by the definition of pager_record in XMLRPC.h in the PageMate\Include directory on a PageMate Server. The structure is returned as a base64-encoded string. Users who are unfamiliar with base64 encoding may wish to refer to RFC 1341, Section 5.2, and additional information provided at http://www.fourmilab.ch/webtools/base64 .

`Error_Message`

An ASCII string, suitable for display, explaining completion status in English.

GetCatalogGroup

DESCRIPTION

GetCatalogGroup returns to an XML RPC client a group record from the PageMate catalog on the server. Except as otherwise specified, all string parameters must be null-terminated single-byte (16 bits per character) ASCII character arrays.

REQUEST PARAMETERS

group_context

A numeric string value providing a positional reference for the record in the PageMate catalog. group_context should be initialized to zero when requesting a specific record or when initiating a wildcard request for a series of records. In its response, the server will return a non-zero context value. When requesting a series of records, each request from the client should provide to the server the value returned in response to the previous call in the sequence.

Group_Name

An alphanumeric string, up to 31 characters in length, specifying the complete or partial (wildcard-terminated) name of the group record to be returned. To request a specific record, specify the complete group name. To request a series of records with names matching a wildcard-terminated string, specify a partial group name terminated with the asterisk wildcard character (*e.g.*, AB* to request all group names beginning with AB, or asterisk alone to request all group records). A single record will be returned on each call until there are no more records matching the wildcard-terminated search string.

Description

An alphanumeric string, up to 63 characters in length, specifying text that should be matched in the Description field of the group record to be returned. Any string value specified here will treated as part of the search criteria in addition to any value specified for Group_Name. Description parameter is required, but a value is optional and may be null..

RESPONSE PARAMETERS

return_value

A numeric string completion status value for the request which, if positive, indicates successful completion and, if negative, indicates failure. Failure indicates that all other response parameters except

Error_Message should be ignored. In the case of failure, if the request was for a specific record, the record could not be found, and if a series of records was being requested through successive wildcard lookup requests, there are no more records available satisfying the request criteria.

group_context

A numeric value providing a positional reference for the record in the PageMate catalog. For successful completions, the server will return a non-zero value. When requesting a series of records, each request from the client should provide to the server the value returned in response to the previous call in the sequence.

Group_Record

A single catalog record is returned in a structure as specified by the definition of group_record in XMLRPC.h in the PageMate\Include directory on a PageMate Server. The structure is returned as a base64-encoded string. Users who are unfamiliar with base64 encoding may wish to refer to RFC 1341, Section 5.2, and additional information provided at http://www.fourmilab.ch/webtools/base64 .

Error_Message

An ASCII string, suitable for display, explaining completion status in English.

SendPage

DESCRIPTION

SendPage submits a message to be enqueued for delivery by the PageMate Server, and returns a message sequence number and status information. Except as otherwise specified, all string parameters must be null-terminated single-byte (16 bits per character) ASCII character arrays.

REQUEST PARAMETERS

Pager_Name

An uppercase alphanumeric string, up to 31 characters in length, matching the name of a subscriber listed in the PageMate catalog, specifying the recipient for the message to be enqueued.

Network_Name

A mixed-case and case-sensitive ASCII string, up to 31 characters in length, exactly matching the name of a network listed in the PageMate catalog, specifying the network to which the message should be submitted.

Pager_Type

A mixed-case and case-sensitive alphabetic ASCII string which must be one of either "Text" or "Numeric".

Pin

A numeric ASCII string, up to 15 characters in length, specifying the pager identification number or equivalent numeric address of the recipient device on the network specified by Network_Name.

Message_Text

A mixed-case ASCII string, up to 599 characters in length, specifying the message to be enqueued for delivery.

Sender_Name

A mixed-case ASCII string, up to 31 characters in length, specifying the username of the user submitting the message on the client system.

Sender_Node

A mixed-case ASCII string, up to 63 characters in length, specifying the computername or hostname of the client system.

msg_reference

A mixed-case ASCII string, up to 15 characters in length, specifying an optional reference parameter.

msg_priority

A single numeric ASCII character specifying the dispatch priority for the message being submitted (0=default, 1=high, 2=medium, 3=low).

RESPONSE PARAMETERS

return_value

A numeric string completion status value for the request which, if positive, indicates successful completion and, if negative, indicates failure.

page_sequence_number

A numeric string value providing a sequence number reference for the message in the PageMate Server message queue. If the message was enqueued successfully, page_sequence number will be a positive integer value. If the message could not be enqueued, page_sequence_number will be zero.

Error_Message

An ASCII string, suitable for display, explaining completion status in English.

SendGroupPage

DESCRIPTION

SendGroupPage submits a message to be enqueued by the PageMate server for delivery to a group. SendGroupPage returns message sequence numbers and status information. Except as otherwise specified, all string parameters must be null-terminated single-byte (16 bits per character) ASCII character arrays.

REQUEST PARAMETERS

`Group_Name`

An uppercase alphanumeric string, up to 31 characters in length, matching the name of a group listed in the PageMate catalog.

`Message_Text`

A mixed-case ASCII string, up to 599 characters in length, specifying the message to be enqueued for delivery.

`Sender_Name`

A mixed-case ASCII string, up to 31 characters in length, specifying the username of the user submitting the message on the client system.

`Sender_Node`

A mixed-case ASCII string, up to 63 characters in length, specifying the computername or hostname of the client system.

`msg_reference`

A mixed-case ASCII string, up to 15 characters in length, specifying an optional reference parameter.

`msg_priority`

A single numeric ASCII character specifying the dispatch priority for the message being submitted (0=default, 1=high, 2=medium, 3=low).

RESPONSE PARAMETERS

`return_value`

A numeric string completion status value for the request which, if positive, indicates successful completion and, if negative, indicates failure.

first_page_sequence_number

A numeric string value providing the sequence number reference for the first message enqueued under the request. If the message was successfully enqueued to at least one member of the group, first_page_sequence_number will be a positive integer value. If the message could not be enqueued to any group members, first_page_sequence_number will be zero.

last_page_sequence_number

A numeric string value providing the sequence number reference for the last message enqueued under the request. If the message was successfully enqueued to at least one member of the group, last_page_sequence_number will be a positive integer value, and the number of group members to which the message was successfully enqueued can be determined as (last_page_sequence_number – first_page_sequence_number + 1). If the message could not be enqueued, last_page_sequence_number will be zero.

Error_Message

An ASCII string, suitable for display, explaining completion status in English.

GetPageStatus

DESCRIPTION

GetPageStatus returns to an XML RPC client the current completion status associated with a message. Except as otherwise specified, all string parameters must be null-terminated single-byte (16 bits per character) ASCII character arrays.

REQUEST PARAMETERS

page_sequence_number

A numeric string value providing a sequence number reference for a message previously enqueued on the server.

RESPONSE PARAMETERS

return_value

A numeric string completion status value for the request which, if positive, indicates successful completion and, if negative, indicates failure. Failure indicates that all other response parameters except Error_Message should be ignored.

page_status

A numeric value indicating current completion status for the original message send request as defined by Status Codes listed in export.h in the PageMate Include directory on a PageMate Server.

extended_status

For messages that fail delivery, a numeric value indicating error status for the original message send request as defined by Extended Completion Codes listed in export.h in the PageMate Include directory on a PageMate Server.

Error_Message

An ASCII string, suitable for display, explaining completion status in English.

Chapter

6

COM+ Interface

PageMate's COM+ Interface was developed to support operation of the PageMate Web Connector. It is an "under the hood" component of PageMate that, subject to certain caveats and constraints, can be used to implement PageMate functionality in site-specific applications on PAM Servers.

Component Object Model (COM) is Microsoft's name for a language-neutral, building-block approach to developing object-oriented application programs. COM+ is an extension of the original COM interface standard introduced by Microsoft in 1993. PageMate's COM+ Interface provides support for programmed control and manipulation of a broad array of PageMate operations including functionality not exposed through any other interface. Effective use of this "under the hood" component of PAM Server assumes detailed familiarity with PageMate software architecture and data structures.

One of the caveats associated with use of PageMate's COM+ interface is that methods, parameter lists and data structures used in PageMate's COM+ interface can change from version to version. Whereas it is a goal and intent of Systemetrics to provide and maintain backward compatibility in other APIs and program interfaces, users should be aware that a goal to maintain backward compatibility in the COM+ interface is much less strict. Sites that use the COM+ interface to embed PageMate functionality in site-specific applications should take care to ensure that they have access to their original source code and be prepared to make changes if and as required to support operation with new versions of PageMate.

One of the relatively straightforward ways that the COM+ interface can be used is for management of named subscriber, group and profile records in the PageMate catalog. A subscriber, in PageMate terminology, is a user that is associated with a named subscriber record that includes specifications for things like employee name, employee number, department name, cell phone, pager or PIN, voice telephone number and a variety of other parameters.

Subscribername identifies a subscriber record in the catalog, and is also used as a login name to identify a PageMate user under the COM+ interface. Subscriber records have associated passwords and authority names. Passwords provide user

authentication and authority names specify what authority or privileges the user has to effect changes in the PageMate catalog and/or to control operation of the PageMate Server.

In addition to subscriber records, other named records in the catalog include group records and profile records. The names of each of these records are required to be unique with respect to all other names in the catalog. That is, subscribernames, groupnames and profilenames must be unique with respect to other subscribernames, groupnames and profilenames.

The PageMate V3 catalog also provides support for ownership of subscriber and group records. Ownership is based on employee number (enum) string values defined in subscriber records. Subscribers, therefore, may own their own or other subscriber records, as well as group records. Ownership of records, under some conditions, provides authority to edit or delete records.

Finally, rules that apply to all catalog records specify that no text field in any catalog record may contain a pipe or vertical bar character (|) or asterisk (*) character, no records may be named "UNDEFINED" or "UNKNOWN", and no subscriber record can be defined to have the reserved employee number "000000000". The pipe or vertical bar character (also called "vbar") is reserved for use as a delimiter in method parameters. In this document, spaces may be shown preceding and following each vertical bar for readability only. Space padding around vertical bars should not be used in method calls.

Authority

Authority.Login (subscribername, password)

The Login method of the Authority object will validate the password provided for the subscribername. If the username/password duple is valid, the Login method will return a character string token. The client should save the token and submit as an input parameter with other methods that may require it.

Authority.Logout (token)

The Logout method of the Authority object will invalidate the specified token. The Logout method returns the string "OK", which indicates only that the method has executed. The return string does not have any meaning with regard to success or failure of the invalidate operation.

Authority.Param (objectname)

The Param method of the Authority object will return the string value of the configuration parameter specified in objectname. Objectname may be any registry configuration parameter defined for use with PageMate (*e.g.*, PATH, LICENSE, VERSION, ANYPIN, MIN_PWD_SIZE, MIN_ENUM_SIZE, BLIND_ENABLE, *etc*.).

Objectname may also be a token obtained from Authority.Login. If objectname is a string token and if the token is valid, the Param method will return a string providing information about the associated subscriber, including the subscribername and employee number, delimited by vertical bar.

Finally, if objectname is the string "_ABOUT", the Param method will return a series of substrings, delimited by vertical bar, that provide PageMate software version and license information.

If the Param method cannot qualify or match objectname, it will return a null (empty) string.

Catalog

Catalog.Add (objectname, recordtype, token, record)

The Add method of the Catalog object will attempt to add the specified record to the PageMate catalog. If the record is added successfully, the Add method will return the string

"Added <catalogname> record <objectname>".

If the Add operation fails, the method will return a string describing the failure or problem. The return string is suitable and intended for use for display for an interactive user.

Record will vary depending on recordtype, as follows:

recordtype=subscriber

user_registration_name | employee_number | company_department | server_name | pager_network | pager_type | pager_pin | voice_phone_1 | voice_phone_2 | voice_phone_3 | fax_phone | forward_to | email_address | authority_name | password | priority | comments |

recordtype=group

description | owner_employee_number | company_department | comments | num_members | member1 | member2 | ..

recordtype=profile

description | fwd_name | ackp_name | nakp_name | rtop_name |

enable_flag | priority | rto_interval |

recordtype=action | n (where n = one-based index of the position the new record is to occupy in the linked list of ARs for objectname)

action_mask | flags | qcp_name | email_list | pagers_list | voice_list | fax_list | npf_filename | proc_filename |

recordtype=site

first_server_name | rpc_type | network_address | server_type |

recordtype=server

site_name | rpc_type | network_address | server_type |

Notes: For recordtype=subscriber, authority_name can be null (an empty string) or any one of: READONLY, OWN, CREATE or ADMINISTRATOR. Any other value will be treated as READONLY.

ADMINISTRATOR will be honored if and only if the submitting user has ADMINISTRATOR authority.

For recordtype=group, owner_employee_number and company_department may be empty, but must be present. If provided, owner_employee_number must match the employee_number of an existing subscriber record.

For recordtype=profile, the profile will be enabled if enable_flag is Y, T or 1; otherwise disabled.

For recordtype=action, qcp_name, npf_filename and proc_filename are filenames (path names) up to 511 characters in length. email_list, pagers_list, voice_list and fax_list are comma-delimited lists of "any" names up to 79 characters in length. npf_filename is not used (will always be the empty string) in this version of the software.

For recordtype=site and recordtype=server, rpc_type must be one of: "DCE (TCP/IP)", "DCE (DECnet)", "DCE (NetBEUI)", "ONC (TCP/IP)" or "XML (TCP/IP)". These are the display strings that are used in GUI dialogs and returned for rpc_type by Catalog.Find when recordtype=server. server_type must be a comma-delimited list containing one or more of: MASTER, SLAVE, and SERVER or NOSERVER. For recordtype=server, site_name must be an existing site name (including blank). A new server will be added as the last (highest index) server in the site.

Catalog.Delete (objectname, recordtype, token)

The Delete method of the Catalog object will attempt to delete the record for objectname from the PageMate catalog. Objectname must be a complete and exact name of a catalog record. Recordtype must be one of the following:

subscriber

group

profile

action | n (where n = one-based index of the record to delete)

site

server

The Delete method will return a string describing the success or failure result of the delete request. If the delete request succeeds, the method will return a string of the form

"Deleted <catalogname> record <objectname>".

Any other return will indicate and describe a failure. The return string is suitable and intended for use as a status (result) display for an interactive user. Uniqueness of site and server names within the catalog is a user responsibility. Delete will act upon the first matching site or server name that it finds.

Catalog.Find (objectname, recordtype, token)

The Find method of the Catalog object will attempt to locate and return the catalog record for the specified objectname. Recordtype may be subscriber, group, profile, action, site, server or any. Token should be the user login token (if logged in); otherwise the null string or a string containing a single zero (0). If a record matching objectname is found in the PageMate catalog, the record will be returned as a delimited string with the general form

objectname | recordtype | content |

where content will vary depending on recordtype, as follows:

recordtype=subscriber

user_registration_name | employee_number | company_department | server_name | pager_network | pager_type | pager_pin | voice_phone_1 | voice_phone_2 | voice_phone_3 | fax_phone | forward_to | email_address | authority_name | password | priority | comments | authority_mask | pager_status | access_code | pin_type | function_code | send_type | page_class | rf_channel | rf_zone | record_key |

recordtype=group

description | owner_employee_number | company_department | forward_to | comments | num_members | authority_mask | record_key | member1 | member2 | ...

reccordtype=profile

description | fwd_name | ackp_name | nakp_name | rtop_name | enable_flag | priority | rto_interval | authority_mask | comments |

recordtype=action | n (where n = one-based index of requested record)

index | total_records | action_mask | flags | qcp_name | email_list | pagers_list | voice_list | fax_list | npf_filename | proc_filename | authority_mask |

recordtype=any

(content appropriate for returned subscriber, group or profile recordtype)

recordtype=site

num_servers | server1 | server2 | …

recordtype=server

site_name | index | rpc_type | network_address | server_type |

recordtype=speedpager

recipient_name | (content as for recordtype=any_

recordtype=speedmsg

message_text (up to 500 characters)

Notes:

Recordtypes subscriber and group return new parameters beginning with PageMate V3.3-0. Recordtype subscriber now returns pager_status, a character string parameter that is one of the words: Active, Forward, Copy, Inactive and Disabled. access_code is a character string parameter that is reserved for future use. pin_type is a character string parameter that is one of the words Pager or Capcode. The next five parameters, function_code through rf_zone, are two-character parameters reserved for use when pin_type is Capcode. record_key is a record locator parameter, up to 15 characters in length, that must be provided as input to Catalog.Update for corresponding subscriber records.

Beginning with PageMate V3.3-0, groups can have other groups as members. Group records now have an owner_employee_number parameter that can be set to assign group record ownership to a subscriber. In the absence of assignment, group records will be owned by PageMate Administrator (anyone with Administrator authority). Group records also now have a forward_to parameter. When forwarding is enabled (as it is by default in PAM Server) and the objectname of another group is specified via forward_to, messages submitted under the current record will be redirected for delivery to the forward_to recipient group in lieu of delivery under the current group record. Group records also now have a record_key parameter, an opaque string parameter up to 15 characters in length that must be passed as input on calls to Catalog.Update for corresponding group records. Finally, when the COMPANY_VIEW Registry option is enabled, group records can be associated with a Company_Department name, with members limited to subscribers and other groups having a matching company_department record parameter value.

For recordtypes "speedpager" and "speedmsg", objectname must be one of "F1", "F2", F3", ... "F10". Recordtype "speedpager" functions as recordtype "any" would if the recipient name had been provided in objectname. Recordtypes speedpager and speedmsg are not supported in conjunction with COMPANY_VIEW (Registry option).

recordtype=any is provided to permit lookup of a name that might be a subscriber, group or profile name. If the name is found, the actual type will be returned in the second parameter in the return string. Any return parameters (*e.g.*, email_list) that have more than one value will be comma-delimited lists.

When an action record is returned, the string returned via recordtype will be simply "action". The index of the record being returned will be provided by "index" (a part of content), and the total number of records defined for the associated profile will be provided by "total_records" (the second parameter in content). On input, recordtype must be of the form "action|*n*", where *n* is the one-based index of the record to be returned.

If no record matching the requested objectname/recordtype is found, Catalog.Find will return a null (empty) string.

The authority_mask parameter is a single numeric digit (character) that indicates what authority the requesting user has with respect to editing or deleting the record returned by Catalog.Find. Values that may be returned via authority_mask include

0 = the requesting user cannot edit or delete the record

1 = the requesting user can edit most parameters in the record, with the exception of ownership and authority name fields, but may not delete the record

2 = the requesting user can edit most parameters in the record, with the exception of ownership, and may delete the record

8 = the requesting user can edit any parameter in the record (including ownership) and may delete the record.

Catalog.List (objectname, recordtype, token)

The List method of the Catalog object will attempt to locate and return an array of catalog names or name/description pairs for a specified wildcard objectname. Objectname must be a string of one or more characters, usually including a required terminating asterisk (*). Recordtype "any" indicates a request for all subscribernames,

groupnames and profilenames that match the wildcard objectname specification. Recordtype may be any one of the following:

subscriber	subscriber names based on subscriber name
registration	subscriber names based on registration name
group	group names
profile	profile names
any	subscriber + group + profile names
site	site names
server	server names (excluding blank names, if any)
network	network names
company	company names
pagertype	pager types
speedgroup	speed pager/message group names
speedmsg	speed (canned) messages
gac	global action conditions
queue q	ueue names
status	message status names

For recordtypes subscriber, group, profile and any, Catalog.List will return a delimited string of value pairs, as follows:

objectname1 | description1 | objectname2 | description2 | ...

For recordtype speedgroup, Catalog.List will return a delimited string of group names, as follows:

name1 | name2 | name 3 | …

For recordtypes speedpager and speedmsg, the format of the string returned by Catalog.List will depend on the objectname value provided by the caller. If objectname is asterisk, Catalog.List will return a delimited string of value pairs in which the first member of each pair will be one of "F1", F2", etc, and the second member will be either the corresponding recipient name (speedpager) or up to 40 characters of the corresponding message text. For example:

F1 | FRED_FLINTSTONE | F3 | YOGI_BEAR | …

For recordtypes speedpager and speedmsg, if objectname is a speedgroup name, Catalog.List will return a delimited string of value pairs, as follows:

objectname1 | description1 | objectname2 | description2 | ...

where each objectname is a button label (maximum 31 characters) and each description is a text string that describes or is represented by the associated button label. The function and purpose of Catalog.List in this

case is to support the operation of a SpeedPage dialog in which both recipient names and canned messages are represented on the screen by colored buttons arranged in a grid. (Note: How we specify the colors for buttons is yet to be determined. For now, the color of speed pager buttons might be green and the color of speeedmsg buttons might be blue or yellow).

For recordtype speedpager, each objectname will be a pager (subscriber) name and each description will be the associated user registration or other description text (up to 63 characters). For recordtype speedmsg, each objectname will be a button label and each description will be the message text (up to 499 characters) to be added to the Message text box when the user presses the speedmsg button. Each press of a speedmsg button will add (append) the text associated with the button to the contents of the Message text box. The text associated with a single speedmsg button can be up to 499 characters, but the Message text box also has a limit of 499 characters.

For recordtypes server, network, company, pagertype, gac, queue and status, the List method will return a delimited list of object names only, as follows:

objectname1 | objectname2 | objectname3 | ...

For recordtypes server, network, company, pagertype, gac and queue, the objectname parameter passed as input to Catalog.List will be ignored and treated as if it were a single asterisk character (requesting a list of all objects of the specified recordtype). For recordtype status, the objectname parameter must specify a queuename (e.g., "page" or "response"). A terminating vertical bar will follow each value in the list, including the last value. If no records matching the requested objectname/recordtype are found, Catalog.List will return a null (empty) string. For recordtype gac (global action conditions), the number of objectnames returned will always be exactly eight (8), any or all of which may be null (empty) strings.

Catalog.Update (objectname, recordtype, token, record)

The Update method of the Catalog object will attempt to update the specified record in the PageMate catalog. If the record is updated successfully, the Update method will return the string "Updated <catalogname> record <objectname>". If the Update operation fails, the method will return a string describing the failure or problem. The

return string is suitable and intended for use as a status (result) display for an interactive user.

When records are retrieved for display via Catalog.Find, password will be sent as a string of asterisks. If the same string of asterisks is returned via Catalog.Update, the password will not be changed by Update. If a null string is returned for password, the password will be changed to the null string. If a string of pure alphanumeric characters is returned via Catalog.Update, the password will be changed to the specified string. Passwords must be pure alphanumerics (i.e., comprised of the digits 0-9 and the alphabetic characters a-z and A-Z only; no spaces or special characters allowed).

Record will vary depending on recordtype, as follows:

recordtype=subscriber

record_key | user_registration_name | employee_number | company_department | server_name | pager_network | pager_type | pager_pin | voice_phone_1 | voice_phone_2 | voice_phone_3 | fax_phone | forward_to | email_address | authority_name | password | priority | comments | pager_status | access_code | pin_type | function_code | send_type | page_class | rf_channel | rf_zone |

recordtype=group

record_key | description | owner_employee_number | company_department | forward_to | comments | num_members | member1 | member2 | ...

recordtype=profile

description | fwd_name | ackp_name | nakp_name | rtop_name | enable_flag | priority | rto_interval

recordtype = action | n (where n = one-based index of record to update)

action_mask | flags | qcp_name | email_list | pagers_list | voice_list | fax_list | npf_filename | proc_filename

recordtype=server

site_name | index | rpc_type | network_address | server_type |

Notes:

Beginning with PageMate V3.3-0, the record parameter for recordtypes subscriber and group require record_key as the first item. record_key is a record locator parameter that must be obtained by locating the target record with a call to Catalog.Find before calling Catalog.Update. record_key is a string parameter that can be up to 15 characters in length.

Calls to Catalog.Update for recordtype subscriber must provide specifications for record_key through pager_pin. Although many may be empty, they must be present. All other parameters for recordtype subscriber are optional. Other parameters are as described in Notes for Catalog.Find.

All parameters in calls to Catalog.Update for recordtype group must be present, although all except record_key, num_members and member names (member1, member2, *etc.*) may be empty.

Message

Message.Send (objectname, recordtype, sender, message, priority, reference, reply_to, token)

The Send method of the Message object will attempt to enqueue message for delivery to objectname. Recordtype must be one of subscriber, group or profile. If the login username of the client user is known, it should be provided in sender; otherwise, sender may be the null string. Priority value zero ("0") specifies default priority. Reference may be any ASCII string up to 16 characters in length, including the null (empty) string. Reply_to can be any ASCII string up to 127 characters in length, including the null string. Support for and interpretation of reply_to is protocol-specific, but will most commonly be interpreted as an e-mail address to which the paging service provider will deliver replies and responses associated with the message being submitted. If the Send method successfully enqueues the message, it will return a string of the form "Enqueued message seqn <nnn> for <objectname>". Any other return string will indicate and describe an error. The return string is suitable and intended for use as a status (result) display for an interactive user. The numeric value of the message sequence number may be extracted from this message and saved for future reference in a call to the Status method of the Message object.

Message.Direct (sender, network, pagertype, pin, message, priority, reference, reply_to, token)

The Direct method of the Message object will attempt to enqueue message for delivery to pin on network. If the login username of the client user is known, it should be provided in sender; otherwise, sender may be the null string. Other parameters operate as described for the Send method (above). If network is the null (empty) string, network will default to the system default network (the first network listed in the networks catalog). If the Direct method successfully enqueues the message, it will return a string of the form "Enqueued message seqn <nnn> for <PIN> on <networkname>". Any other return will signal and describe an error. The return string is suitable and intended for use as a status (result) display for an interactive user. The numeric value of the message sequence number may be extracted from this message and saved for future reference in a call to the Status method of the Message object.

Note: The Direct method provides a capability to submit a message to an individual text or numeric pager that is not listed in the PageMate catalog. Messages submitted via this method cannot be addressed to groups or profiles.

Message.Status (queuename, seqn, token)

The Status method of the Message object will return a string describing the status of the message specified by queuename (the name of a message queue) and seqn (a message sequence number in the specified queue). The pager message queue (the most commonly used queue) is named "page". Message.Status returns a delimited string in the following form:

from (sender) | to (recipient) | tstamp | status | status_text |

Status will be one of the valid status codewords (e.g., Enqueued, Starting, Completed, Failed, etc.). Status_text is a more complete English phrase or sentence providing detailed information about the status for the specified message. A null string will be returned for any parameters that have unknown values.

Message.Get (queuename, seqn, token)

The Get method of the Message object returns the body of a specified message in a delimited string as follows:

from | to | tstamp | status | priority | reference | message_body

The Get method is similar to Status, the difference being that Status is provided for the purpose of obtaining detailed status information about a message, while Get returns detail of a different kind, including the message priority, reference and the complete message body (up to 500 characters in length).

When queuename is "response", seqn must be a vbar-delimited list specifying sequence number, time stamp, (optional) statustype and (optional) subscribername. If statustype is not specified, it will default to "any". If subscribername is specified, it can be either a subscribername or an employee number. If subscribername is not specified, it will default to asterisk (wildcard) for users with operator or administrator authority; otherwise, it will default to the current user (based on token).

Finally, a couple of useful things to know about the information returned by Message.Get when queuename is response:

- Messages in the response queue are stored and indexed by employee number, rather than by subscribername, so the data returned by

Message.Get will actually reflect messages associated with the subscriber's employee number, rather than subscribername.

- A message in the reponse queue has no associated priority value. Message.Get will always return zero priority for response queue messages, so this return parameter is meaningless for reporting purposes when queuetype is response.

Message.List (queuename, sender, recipient, statustype, reference, startstamp, endstamp, token)

The List method of the Message object provides a capability to generate a list of messages that meet user-specified criteria, such as messages sent to a specified recipient within a certain time window. Any combination of sender, recipient, statustype, reference and time window can be specified. If not specified, sender and reference will default to asterisk (wildcard). If recipient is not specified, it will default to wildcard when queuename is "page", and default to the subscribername of the current user when queuename is "response". When queuename is "response", recipient, if specified, can be either a subscribername or an employee number. Messages in the response queue are associated with employee numbers, rather than subscriber names, so although a request to Message.List may specify a subscribername, the data returned will be a list of messages associated with the subscriber's employee number. Startstamp, if not specified, will default to 00:00 of the current day. Endstamp, if not specified, will default to the current system time stamp. Queuename, statustype and token have no default values and must be specified.

Statustype must be a string of the form "statspec=<list>, where <list> is a comma-delimited list of queue-specific status values as defined under Catalog.List. For example: "statspec=enqueued,starting". statustype may not contain embedded blanks. The specification "statspec=all" requests a report for all status values.

Message.List will return a vbar-delimited string providing four (4) substring parameters for each message that qualifies. The four parameters (which will be repeated for each message in the list) are:

seqn | tstamp | status | reference |

One use for the Message.List method is to provide a list of messages that satisfy criteria of interest, based upon which list the user can subsequently select one or more from the list and use the Get method (and possibly the Reply method) for further processing of the messages.

Messages that are listed from the "response" queue will (after being retrieved for display via the Get method) be eligible for reply. Messages listed from the "page" queue are never eligible for reply.

Message.Reply (seqn, sender, token, acknak, reply)

The Reply method of the Message object submits a reply to the message having the sequence number and reference specified by seqn (a composite parameter specified in the form seqn | reference).

If the user on behalf of whom Message.Reply is called has OPERATOR or ADMINISTRATOR authority, any subscribername may be specified in the sender parameter; otherwise, the server will ignore the value provided in sender and automatically replace sender with the current user's subscribername (based on token). Token is required input. Acknak is required to be one of either "ACK" or "NAK" (upper or lower case, doesn't matter). Reply is an optional reply message (text string, up to 500 characters in length).

Queue

Queue.Current (queuename, token)

The Current method of the Queue object will return a vbar-delimited list of entries in the specified queue. OPERATOR or ADMINISTRATOR authority is required to obtain data from this method. The data returned for each item in the queue will include:

- sequence number (1-999999999)
- date_stamp (*e.g.*, 14-DEC-2009 12:31)
- sender (subscriber or profile name)
- recipient (subscriber name)
- reference (0-15 ASCII characters)
- status (*e.g.*, "Enqueued", "Starting", *etc.*)

Each item in the list will be terminated by vertical bar. There are no markers for end-of-record; therefore record delimiters must be determined by counting vbars.

Queue.Get (resource_type, resource_id, token)

The Get method of the Queue object returns the current value of the specified resource. Resource_type can be the name of a queue (e.g., page, email, voice, response or profile) or any queue-related resource (port, gac or action). If resource_type is a queuename, resource_id must be the sequence number of an item in the queue. In this case, Queue.Get will return the complete queue entry record in a vbar-delimited list, including:

sequence_number | tstamp | sender | recipient | reference | priority | message_body | status |

Queue.Get provides a capability to get/display all information for a message that is selected by the user from a list generated by either Queue.Current or Queue.List (*e.g.*, when the user asks for detail for a particular item in a list generated by one of these other methods).

Queue.Statistics (queuename, startstamp, endstamp, statusspec, token)

The Statistics method of the Queue object returns a report of message statistics for the specified time interval and queue. The method provides a special capability supporting a series or sequence of calls that "drill down" to provide increasing levels of detail for report items. All input parameters are required.

startstamp, endstamp and token have customary meanings. startstamp and endstamp specify the time window for data to be reported. startstamp and endstampt are expressed as dd-mmm-yyyy:hh:mm. token is a string obtained from Authority.Login that establishes a user's authority to request information or perform functions via the method.

queuename specifies the queue for which data is requested. The form for statusspec will vary based on queue. statusspec has the general form <parameter_name>=<value>.

Whe queuename is "page", <parameter_name> can be any one of "statistics", "depth" or "volume", and the format of <value> will vary depending on <parameter_name>.

<parameter_name>=statistics

When queuename is "page" and <parameter_name> is "statistics", <value> must be one of "Totals", "Network", "Port" or "Subscriber". When Queue.Statistics is first called (from the menubar), the user must specify a queuename and select a category from the list of valid categories for the queuename. When queuename is "page", the selected category name must be passed to Queue.Statistics in the statuspec parameter in the form "statistics=<categoryname>". For example: "statistics=Network". If Queue.Statistics is later called to obtain more detail about a selected line in any report, statuspec will have a special value that will be based on parameters returned in the prior call.

Queue.Statistics returns a report in four sections, as follows:

1. Report header. The report header is a series of six parameters in a vbar-delimited list, as follows:

queuename | formatype | numcols | title | startstamp | endstamp |

where

queuename = the queuename requested in the call

formatype = a format keyword (e.g., "summary" or "detail")

numcols = the number of columns in each row of report data

title = a title to be displayed at the top of the report (80 chars max)

startstamp and endstamp = 17-character time stamps

Title is a string that should be displayed as the first line in the title (centered at the top of the display). Startstamp and endstamp reformat and echo the time stamps specified as input to Queue.Statistics. Title, startstamp and endstamp parameters are provided as strings to be "plugged into" title lines in the display.

2. Column widths. A series of numcols numeric parameter specifying the maximum number of characters to be displayed from each column of report data. The first column width may be zero if the item in this column is not to be displayed (*i.e.*, a hidden parameter). For example: 0|32|10|10|10|10|

3. Column names. A series of numcols names to be displayed at the top of each column. For hidden columns, the column name will be a name that must be saved for return in a subsequent call for detail about any row in the report. For example:

network|Network Name|Success|Failed|Other|Total|

4. Report rows. A series of numcols parameters repeated for each row of report data. For example:

SkyWord|SkyWord|5273|12|2|5287|

If a user selects any line in the report to request detail about the line item, a call should be made to Queue.Statistics or Queue.Get to request a subsequent report. The choice of which method to call (Queue.Statistics or Queue.Get) will be based on whether the current report is a summary or detail report. For the example described above, if the user clicked in the SkyWord row of the summary report (to request a list of all 5287 individual messages that had been sent to SkyWord), a second call to Queue.Statistics would be made with statuspec specifying "network=SkyWord" (the hidden column name, followed by equal sign and the hidden value from the row data).

<parameter_name>=depth

When queuename is "page" and <parameter_name> is "depth", <value> must specify port type as one of “modem”, “direct”, “socket” or “all”. For example: “depth=direct”. This parameter specifies whether the data should be queue depth for modem (messages enqueued for delivery via modems), direct (messages enqueued for delivery via direct-connect serial ports), socket (messages enqueued for delivery via sockets), or all (total messages enqueued).

<parameter_name>=volume

When queuename is "page" and <parameter_name> is "volume", <value> must specify a port number. PageMate supports up to ten (10) serial ports for dispatching messages. If port number zero is specified, the data reported will be total for all ports; otherwise, port number must be an integer in the range 1 through 10.

When queuename is “page” and <parameter_name> is either "depth" or "volume", data reported will always begin an end on an even hour, day or month and be reported in intervals of minutes, hours or days. Report data will be delivered as a set of x-y pairs, where values of x will be integers representing minutes (0-60), hours (0-24) or days (0-31). Values of y will be positive integer values representing number of messages.

Index