BEA WebLogic Network Gatekeeper™

API Description for Parlay X

Version 1.0
Revised: March 14, 2005

Contents

Introduction and Roadmap

The following sections describe the audience for and organization of this document:

- "Document Scope and Audience" on page 1-2
- "Guide to this Document" on page 1-2

Document Scope and Audience

The purpose of this guide is to describe the APIs for Parlay X Web services exposed by the WebLogic Network Gatekeeper.

For basic knowledge about WebLogic Network Gatekeeper and the functions it provides, refer to the Product Description..

For basic knowledge about how to use Parlay X Web services, refer to the Developer's Guide.

Guide to this Document

- Chapter 1, "Introduction and Roadmap," informs you about the structure and contents of this document, the used writing conventions, and related documentation.
- Chapter 2, "Parlay X Web Services Interface," gives general information about the Parlay X API. Definitions for datatypes and exceptions general for all modules.
- Chapter 3, "Access," gives information on the Access API used for user login and logout.
- Chapter 4, "Third Party Call," gives information on the Call API used for application-initiated calls.
- Chapter 5, "Network-Initiated Third Party Call," gives information on the Network-initiated third party call API used for handling calls initiated from the telephone network.
- Chapter 6, "SMS," gives information on the SMS API used for sending and receiving SMS:es.
- Chapter 7, "Multimedia Message," gives information on the Multimedia message API used for sending and receiving multimedia messages such as MMS:es.
- Chapter 8, "Payment," gives information on the Payment API used for handling payment based on content for both pre- and post-paid services.
- Chapter 9, "Terminal Location," gives information on the Terminal location API used for retrieving the location of an end user's terminal.
- Chapter 10, "User Status," gives information on the User Status API used for retrieving the status of an end user's terminal.

Parlay X Web Services Interface

The following sections provide general information about the Parlay X API:

- "About Parlay X web services interface" on page 2-2
- "Common Datatypes" on page 2-3
- "Common Exceptions" on page 2-3

About Parlay X web services interface

The Parlay X web services API offers a standardized high level interface towards WebLogic Network Gatekeeper.

To run a Parlay X web services application, It is necesssary to have a connection to a WebLogic Network Gatekeeper, together with a set of WSDL files describing the API.

In some cases, there is also a need for some additional runtime libraries depending on which language the application is implemented in.

Each main component is contained in a specific module.

Module	**Description**
Access	Contains web services methods for handling sessions towards the WebLogic Network Gatekeeper. All Web services applications use this module as an entry point.
Third party call	Contains web services methods for handling application initiated calls.
Network-initiated third party call	Contains web services methods for handling network initiated calls.
SMS	Contains web services methods for handling sending and reception of SMS:es.
Multimedia Message	Contains web services methods for handling sending and reception of MMS:es.
Payment	Contains web services methods for handling charging based on content.
User status	Contains web services methods for getting information on the status of mobile terminals.
Terminal location	Contains web services methods for getting the geographical position of a mobile terminal.
Types	Contains datatyses used.

Common Datatypes

All datatypes are defined as xsd (XML Schema) types for example, when a type is denoted String, the exact definition is xsd:string.

Below is a list of the datatypes common for all sub-APIs. The datatypes specific for each sub-API are described in their respective chapter.

Table 2-1 General data types

Data type	Description
EndUserIdentifier	Identifies an end user. Specified as a URI: [scheme]:[schemeSpecificPart] (RFC 2396, amended by RFC 2732). Example schemes are tel (RFC 2806) and sip (RFC 3261).
ArrayOfEndUserIdentifier	A collection of elements where each element is of data type EndUserIdentifier.

Common Exceptions

Below is a list of the exceptions common for all sub-APIs. The exceptions specific for each sub-API are described in their respective chapter.

All of these exceptions are common to multiple Parlay X Web Services. Each exception is assigned an eight character identifier, where:

The first 3 characters “GEN” identify the exception as generic: that is, common to multiple Parlay X services. The “GEN” string shall not be assigned to any Parlay X Web Service-specific exception.

The next 4 digits “1xxx” uniquely identifies the exception within the set of common exceptions. The “1xxx” string may be re-used by any Parlay X Web Service-specific exception defined in this specification.

The last character identifies the severity of the exception condition, as follows:

- “F”: fatal error, typically indicating an infrastructure problem; the operation triggering the exception should not be retried

- "E": error, typically indicating an application or user error; the operation triggering the exception has not completed and may be retried
- "W": warning, typically indicating that an operation has completed, but there are cautions or other caveats.

Table 2-2 General exceptions

ID	Text string	
GEN1000E	UnknownEndUserException	This fault occurs if the end user identification that is passed is unknown.
GEN1001E	InvalidArgumentException	This fault occurs if an argument passed is semantically incorrect or when the parameter does not conform to the limits specified in the Parlay X specification: e.g. when passing the end user identification: "tel:www.parlay.org".
GEN1002F	ServiceException	This fault is caused by generic platform or network errors.
GEN1003E	PolicyException	This fault is caused by a violation of a policy of the Parlay X Web Service: e.g., when parameter values are used that are outside the scope of the service level agreement.
GEN1004E	ApplicationException	This fault is caused by a generic error in an application web service when processing a message invocation from a Parlay X Web Service. The Parlay X Web Service can log this information and possibly raise an alarm when the number of exceptions reaches a pre-defined threshold.
GEN1005W	MessageTooLongException	This fault is caused if a message to be sent exceeds the maximum length supported by the Web Service; e.g. the message may be too long for a destination terminal device.

Access

A normal Parlay X web services user (without the right to use the administrative parts) can perform the following tasks:

- Login
- Log out
- Change password

The methods described below are not a part of the Parlay X standard, since Parlay X does not deal with users and user administration but leaves this to each implementation:

- “Access API” on page 3-2
- “Exceptions” on page 3-4

Access API

applicationLogin

Used by an application to login and retrieve a loginTicket.

Table 3-1 applicationLogin(serviceProvider, application, applicationInstanceGroup, password)

Parameter Name	Type	Description
Input		
serviceProvider	String	ID of the service provider as given by the operator or the service provider.
application	String	ID of the application as given by the operator or the service provider.
applicationInstanceGrou p	String	ID of the application instance group as given by the operator or the service provider.
password	String	Password for the application as given by the operator or the service provider. Note that this may also have been changed by the by the application provider.
Returns		
	String	ID of the login-session. This ID is used for each request towards WebLogic Network Gatekeeper. Shall be used to access other Parlay X services. The ID shall be sent in the SOAP header of every SOAP operation.

Possible exceptions
AccessException
GeneralException

applicationLogout

Used to logout an application from the underlying system. Destroys the login session and the corresponding loginTicket.

Table 3-2 applicationLogout(loginTicket)

Parameter Name	Type	Description
Input		
loginTicket	String	ID of the login-session. The loginTicket retrieved when logging in.
Returns		
	-	

Possible exceptions
AccessException
GeneralException

changeApplicationPassword

Used to change the password for an application.

Table 3-3 changeApplicationPassword(loginTicket, oldPassword, newPassword)

Parameter Name	Type	Description
Input		
loginTicket	String	ID of the login-session. The loginTicket retrieved when logging in.
oldPassword	String	The current password.
newPassword	String	The new password.

Table 3-3 changeApplicationPassword(loginTicket, oldPassword, newPassword)

Parameter Name	Type	Description
Returns		
	-	
Possible exceptions		
AccessException		
GeneralException		

Exceptions

Table 3-4 Exceptions specific for Access

Exception	Description
AccessException	Exceptions of this type are raised when there are error conditions related to the Access Web Service. Other error conditions are reported using the exception GeneralException.
GeneralException	This exception is raised when the applications session has expired or there are communication problems with the underlying platform.

CHAPTER 4

Third Party Call

The following sections provide information about the Call API used for application-initiated calls:

Call API

makeACall

Setup a call between two parties.

Table 4-1 makeACall(callingParty, calledParty, charging)

Parameter Name	Type	Description
Input		
callingParty	EndUserIdentifier	URI of the calling party.
calledParty	EndUserIdentifier	URI of the called party.
charging	String	Optional. If present, represents the name of an operator-specific charging plan that defines who to charge for the call and how much. If the named charge plan does not exist, InvalidArgumentException is thrown. If no charge plan is specified, charging occurs in accordance with an operator-specific charging policy.
Returns		
	String	Identifier for the call.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException

getCallInformation

Get information on an ongoing call.

Table 4-2 getCallInformation(callIdentifier)

Parameter Name	Type	Description
Input		
callIdentifier	String	ID of the call to get information on.
Returns		
	CallInformationType	Status of the call.

Possible exceptions
UnknownEndUserException
ServiceException

endCall

Terminate an ongoing call.

Table 4-3 endCall(callIdentifier)

Parameter Name	Type	Description
Input		
callIdentifier	String	ID of the call to get information on.
Returns		
Void		

Possible exceptions
CallTerminatedException
ServiceException
UnknownEndUserException

cancelCallRequest

Cancels a request to start a Call. This method can only be used to cancel Calls that has not started.

Table 4-4 cancelCallRequest(callIdentifier)

Parameter Name	Type	Description
Input		
callIdentifier	String	ID of the call to get information on.
Returns		
Void		

Possible exceptions
CallConnectedException
ServiceException
UnknownCallIdentifierException

Data Types

Table 4-5 Data types specific for call

Data type	Description	
CallInformationType	Structure where:	
	[0]	Indicates the current status of the call. Datatype is CallStatus.
	[1]	The time of the beginning of the call. Datatype is DateTime. Valid when callStatus <> CallInitial.
	[2]	Duration of the call expressed in seconds. Datatype is int. Valid when callStatus == CallTerminated.
	[3]	Cause of the termination of the call. Datatype is CallTerminationCause. Valid when callStatus == CallTerminated.
CallStatus	Enumeration with the following values:	
	CallInitial	call is being established
	CallConnected	call is active
	CallTerminated	call was terminated
CallTerminationCaus e	Enumeration with the following values:	
	CallingPartyNoAnswer	Calling Party did not answer
	CalledPartyNoAnswer	Called Party did not answer
	CallingPartyBusy	Calling Party was busy
	CalledPartyBusy	Called Party was busy

Table 4-5 Data types specific for call

Data type	Description	
	CallingPartyNotReachable	Calling Party was not reachable
	CalledPartyNotReachable	Called Party was not reachable
	CallHangUp	The call was terminated by either party hanging up
	CallAborted	The call was aborted (any other termination cause)

Exceptions

Table 4-6 Exceptions specific for call

ID	Exception	Description
3PC1000W	CallConnectedException	The call was already active
3PC1001W	CallTerminatedException	The call is already terminated
3PC1002E	UnknownCallIdentifierException	The callIdentifier supplied does not relate to any known call request or has expired.

Network-Initiated Third Party Call

The following sections describe the Network-initiated third party call API used for handling calls initiated from the telephone network:

Call API

handleBusy

Invoked from the network to an application. Informs the Parlay X gateway how to handle incoming calls from the callingParty when trying to set up a call and the calledParty's call status is busy. The return value is decided by the application.

Table 5-1 handleBusy(callingParty, calledParty)

Parameter Name	Type	Description
Input		
callingParty	EndUserIdentifier	URI of the calling party.
calledParty	EndUserIdentifier	URI of the called party.
Returns		
	Action	Indicates the action to be performed by the gateway, see "Action" on page 5-8. Continue: results in normal handling of the busy event in the network, for example, playing of a busy tone to the callingParty. EndCall: results in the call being terminated; the exact tone or announcement that will be played to the callingParty is operator-specific. Route: results in the call being re-routed to a calledParty specified by the application.

Possible exceptions
UnknownEndUserException

Possible exceptions
ApplicationException
InvalidArgumentException

handleNotReachable

Invoked from the network to an application. Informs the Parlay X gateway how to handle incoming calls from the callingParty when trying to set up a call and the calledParty's call status is not reachable. The return value is decided by the application

Table 5-2 handleNotReachable(callingParty, calledParty)

Parameter Name	Type	Description
Input		
callingParty	EndUserIdentifier	URI of the calling party.
calledParty	EndUserIdentifier	URI of the called party.
Returns		
	Action	Indicates the action to be performed by the gateway, see "Action" on page 5-8. Continue: results in normal handling of the 'not reachable' event in the network, for example, playing of a busy tone to the callingParty. EndCall: results in the call being terminated; the exact tone or announcement that will be played to the callingParty is operator-specific. Route: results in the call being re-routed to a calledParty specified by the application.

Possible exceptions
ApplicationException
InvalidArgumentException
UnknownEndUserException

handleNoAnswer

Invoked from the network to an application. Informs the Parlay X gateway how to handle incoming calls from the callingParty when trying to set up a call and the called parties does not answer. The return value is decided by the application

Table 5-3 handleNoAnswer(callingParty, calledParty)

Parameter Name	**Type**	**Description**
Input		
callingParty	EndUserIdentifier	URI of the calling party.
calledParty	EndUserIdentifier	URI of the called party.
Returns		
	Action	Indicates the action to be performed by the gateway, see "Action" on page 5-8. Continue results in normal handling of the no answer event in the network, for example, playing of a busy tone to the callingParty. EndCall results in the call being terminated; the exact tone or announcement that will be played to the callingParty is operator-specific. Route, results in the call being re-routed to a calledParty specified by the application.

Possible exceptions
ApplicationException
InvalidArgumentException
UnknownEndUserException

handleCalledNumber

Invoked from the network to an application. Informs the Parlay X gateway how to handle incoming calls from the callingParty to the calledParty before the actual call-setup is performed.

This is can be used when using calledParty as a common telephone number, and depending on the callingParty's telephone number, the call is routed to a destination number located geographically closest to the callingParty. The return value is decided by the application

Table 5-4 handleCalledNumber(callingParty, calledParty)

Parameter Name	Type	Description
Input		
callingParty	EndUserIdentifier	URI of the calling party.
calledParty	EndUserIdentifier	URI of the called party.

Table 5-4 handleCalledNumber(callingParty, calledParty)

Parameter Name	Type	Description
Returns		
	Action	Indicates the action to be performed by the gateway, see “Action” on page 5-8. Continue: results in normal handling, that is the call will be routed to the calledParty number, as originally dialled. EndCall: results in the call being terminated; the exact tone or announcement that will be played to the callingParty is operator-specific. Route: results in the call being re-routed to a calledParty specified by the application.

Possible exceptions
ApplicationException
InvalidArgumentException
UnknownEndUserException

handleOffHook

Invoked from the network to an application. Informs the Parlay X gateway how to handle incoming calls from the callingParty when trying to set up a call and the called party is off hook. The return value is decided by the application

Table 5-5 handleOffHook(callingParty)

Parameter Name	Type	Description
Input		
callingParty	EndUserIdentifier	URI of the calling party.
Returns		
	Action	Indicates the action to be performed by the gateway, see “Action” on page 5-8. Continue: results in normal handling of the off hook event in the network, for example, the calling party can enter digits and, when enough digits are entered, the call is routed based on this information. EndCall: results in the call being terminated; the exact tone or announcement that will be played to the callingParty is operator-specific. Route: results in the call being re-routed to a calledParty specified by the application.

Possible exceptions
ApplicationException
InvalidArgumentException
UnknownEndUserException

Data Types

Table 5-6 Data types specific for network initiated calls

Data type	Description	
Action	Structure containing the following parameters:	
	actionToPerform	Indicates the action as described in ActionValues. Datatype: ActionValues, see below.
	routingAddress	The address to be used in case the action indicates 'Route' Datatype: EndUserIdentifier
	charging	OPTIONAL. If present, represents the name of an operator-specific charging plan that defines who to charge for the call and how much. If no charge plan is specified, the callingParty will be charged, based on an operator-specific charging policy. Datatype: String
ActionValues	The ActionValues data type is an enumeration with the following values:	
	Route	Request to (re-)route the call to the address indicated with routingAddress.
	Continue	Request to continue the call without any changes. This will result in normal handling of the event in the network
	EndCall	Request to end the call. This will result in termination of the call. The callingParty will receive a tone or announcement.

Exceptions

No exceptions defined specifically for network initiated calls.

SMS

The following sections describe the SMS API used for sending and receiving SMS:es:

Send SMS API

sendSms

Send an SMS to one or several destinations.

Table 6-1 sendSms (destAddressSet, senderName, charging, message)

Parameter Name	Type	Description
Input		
destAddressSet	EndUserIdentifier[ ]	Array of EndUserIdentifier. MSISDN of the destination terminal(s).
senderName	String	Format is tel:<mailbox ID>\<mailbox password>\tel:<originator address> Mailbox ID and password are supplied by the service provider. Example: "tel:50000\apassword\tel:+46541768700"
charging	String	Optional.
message	String	Message. For GSM systems, if containing characters not in the GSM 7-bit character set, the SMS is sent as a Unicode SMS. If longer than the maximum supported length (for example. for GSM, 160 GSM 7-bit characters or 70 Unicode characters), the message will be sent as several concatenated short messages.
Returns		
	String	ID of the SMS delivery request.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException
MessageTooLongException
PolicyException

sendSmsLogo

Send an SMS logo to one or several destinations.

Table 6-2 sendSmsLogo (destAddressSet, senderName, charging, image, smsFormat, message)

Parameter Name	Type	Description
Input		
destAddressSet	EndUserIdentifier[ ]	Array of EndUserIdentifier. MSISDN of the destination terminal(s).
senderName	String	Format is tel:<mailbox ID>\<mailbox password>\tel:<originator address> Mailbox ID and password are supplied by the service provider. Example: "tel:50000\apassword\tel:+46541768700"
charging	String	Optional.
image	Base64Binary	The image/logo to send. See "Handling of SMS Logos" on page 6-10 for a description of the different formats supported. Note that the image encoding must be according to the one stated in the parameter smsFormat.

Table 6-2 sendSmsLogo (destAddressSet, senderName, charging, image, smsFormat, message)

Parameter Name	Type	Description
smsFormat	smsFormat	Possible values are: [EMS | SmartMessaging] EMS for EMS message SmartMessaging for Smart Messaging messages.
Returns		
	String	ID of the SMS delivery request.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException
MessageTooLongException
PolicyException
UnsupportedFormatException

sendSmsRingtone

Send an SMS ringtone to one or several destinations.

Table 6-3 sendSmsRingtone (destAddressSet, senderName, charging, ringtone, smsFormat, message)

Parameter Name	Type	Description
Input		
destAddressSet	EndUserIdentifier[ ]	Array of EndUserIdentifier. MSISDN of the destination terminal(s).
senderName	String	Format is tel:<mailbox ID>\<mailbox password>\tel:<originator address> Mailbox ID and password are supplied by the service provider. Example: "tel:50000\apassword\tel:+46541768700"
charging	String	Optional.
ringtone	String	See "Handling of SMS Ringtones" on page 6-12 for a description of the different formats supported. Note that the ringtone encoding must be according to the one stated in the parameter smsFormat.
smsFormat	smsFormat	Possible values are: [EMS | SmartMessaging] EMS for EMS message SmartMessaging for Smart Messaging messages.
Returns		
	String	ID of the SMS delivery request.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException
MessageTooLongException
UnsupportedFormatException

getSmsDeliveryStatus

Requests the status of a previous SMS delivery request identified by requestIdentifier.

Table 6-4 getSmsDeliveryStatus(requestIdentifier)

Parameter Name	Type	Description
Input		
requestIdentifier	String	ID of the SMS delivery request.
Returns		
	DeliveryStatusType[]	Array of DeliveryStatusType. Each entry in the array corresponds to a destination address as given when the message was sent. See "DeliveryStatusType" on page 6-9.

Possible exceptions
UnknownRequestIdentifierException
ServiceException

SMS notification API

notifySmsReception

When an SMS arrives to WebLogic Network Gatekeeper, a SOAP request with this operation, will be invoked to the server-side of the Parlay X client.

This method must be implemented on the server-side of the Parlay X client.

WebLogic Network Gatekeeper must be aware of the location of the Web Service implementing this interface. The location of the web service is defined using OAM in WebLogic Network Gatekeeper.

Table 6-5 notifySmsReception(registrationIdentifier, smsServiceActivationNumber, senderAddress, message)

Parameter Name	Type	Description
Input		
registrationIdentifier	String	ID of the mailbox.
smsServiceActivationNumber	String	Address to service.
senderAddress	EndUserIdentifier	Sender address.
message	String	Message. Up to 160 characters.
Returns		
Void		

Possible exceptions
ApplicationException

Receive SMS API

getReceivedSms

Gets all SMS:es received to a mailbox since last invocation of the methods on the mailbox.

Table 6-6 getReceivedSms(registrationIdentifier)

Parameter Name	Type	Description
Input		
registrationIdentifier	String	The mailbox. Format is <mailbox ID>\<mailbox password> Mailbox ID and password are supplied by the service provider. Example: "tel:50000\apassword"
Returns		
	SmsType[]	Array of SMSType. Lists of SMS:es received SMS since last invocation.

Possible exceptions
UnknownRegistrationIdentifierException
ServiceException

Data Types

Table 6-7 Data types specific for SMS

Data type	Description	
DeliveryStatusType	Structure containing the following parameters:	
	destinationAddress	It indicates the destination address to which the notification is related Datatype: EndUserIdentifier.
	deliveryStatus	Indicates the delivery result for destinationAddress. Possible values are: [Delivered |DeliveryUncertain | DeliveryImpossible] Datatype: DeliveryStatus
DeliveryStatus	The DeliveryStatus data type is an enumeration with the following values:	
	Delivered	Successful delivery. In case of concatenated messages, only when all the SMS-parts have been successfully delivered.
	DeliveryUncertain	Cannot resolve status. For example, because it was handed off to another network.
	DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
	MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.
SmsType	Structure containing the following parameters:	
	message	Text received in SMS. Datatype: String

Table 6-7 Data types specific for SMS

Data type	Description	
	senderAddress	It indicates address sending the SMS. Datatype: EndUserIdentifier
SmsFormat	Enumeration with the following values:	
	Ems	Enhanced Messaging Service, standardized in 3GPP TS 23.040, which defines a logo/ringtone format.
	SmartMessaging	Nokia's de facto standard, which defines a logo/ringtone format.

Exceptions

Table 6-8 Exceptions specific for SMS

ID	Exception	Description
SMS1000E	UnsupportedFormatException	The smsFormat supplied is not one of the permitted values of the SmsFormat data type.
SMS1001E	UnknownRegistrationIdentifierException	The registrationIdentifier supplied is not known by the server
SMS1002E	UnknownRequestIdentifierException	The requestIdentifier supplied does not relate to any known SMS request or has expired.

Handling of SMS Logos

Smartmessaging

Logos sent in Smartmessaging format will result in Nokia operator logos being sent to the handset. The subscriber must be registered in it's home network.

Formats supported:

- Standard: 72x14 pixels, monochrome bitmap
- Bigger: 78x21, monochrome bitmap

The image is sent as an hexadecimal string with a leading Mobile Country Code (MCC) and the Mobile Network Code (MNC):

`<MCC><NCC><image data>`

`<MCC>` is a 3-digit decimal value, that should be encoded as little endian BCD, resulting in two octets with “f” as filler.

`<NCC>` is a 2-digit decimal value, that should be encoded as little endian BCD, resulting in one octet.

For example, if the MCC is 240 and NCC is 01, `<MCC>` will be 42F0 and `<NCC>` will be 10. The resulting hexadecimal string will be `42F010<image data>`.

Below is an example of tools and the procedure to code the hexadecimal string to be used when sanding the logos.

1. Start with a monochrome bitmap image in the correct dimensions.
2. Convert it to a wireless bitmap image (WBMP). A converter can be found at http://elvis.teraflops.com/wbmp
3. The WBMP a should be converted to a hexadecimal string. A converter can be found at http://www.clickatell.com/central/skins/converters/convert_bitmaps.php
4. Add the Mobile Country Code (MCC) and the Mobile Network Code (MNC) first in the string as described above. Values for MCC and MNC can be found at http://www.gsmworld.com/roaming/gsminfo

EMS

Formats supported:

- Small Picture
- Large Picture

Only pictures that can be fitted into one SMS are supported.

The picture should be encoded as a hexadecimal string according to format specified in the Ericsson document *Developer's Guidelines, Enhanced Messaging Service (EMS), EN/LZT 103 5256 R6A*.

Handling of SMS Ringtones

Smartmessaging

Hexadecimal string representing a ringtone.

A tool for converting midi-files to Nokia ringtones is found at
http://www.clickatell.com/central/skins/converters/convert_mid.php

EMS

Hexadecimal string representation of an iMelody.
See http://www.irda.org/standards/specifications.asp
for a specification of the format.

CHAPTER 7

Multimedia Message

The following sections describe the Multimedia message API used for sending and receiving multimedia messages such as MMS:es:

Send Message API

sendMessage

Send an MMS or e-mail to one or several destinations.

Table 7-1 sendMessage(destAddressset, senderAddress, subject, priority, charging) Attachment[content]

Parameter Name	Type	Description
Input		
destAddressset	EndUserIdentifier[ ]	Array of EndUserIdentifier. MSISDN of the destination terminal(s).
senderAddress	String	Format is <mailbox ID>\<mailbox password>\tel:<originator address> Mailbox ID and password are supplied by the service provider. Example: "50000\apassword\tel:+46541768700"
subject	String	OPTIONAL. Subject of the message.
priority	MessagePriority	OPTIONAL. Represents the priority of the message. If not defined, the network will assign a priority based on an operator policy.
charging	String	OPTIONAL. Represents the name of an operator-specific charging plan that defines who to charge for the message and how much. If the named charge plan does not exist, the InvalidArgumentException is thrown. If no charge plan is specified, the sending service/application will be charged, based on operator-specific charging policy.

Table 7-1 sendMessage(destAddressset, senderAddress, subject, priority, charging) Attachment[content]

Parameter Name	Type	Description
Attachment[content]	MIME or DIME	Data to be sent with the message, that is MIME or DIME format. **Note:** Sent as a SOAP-Attachment. Not as a parameter in the SOAP envelope. When sending a SMIL message, the MIME content ID in the attachment must be the same as the SMIL Content ID defined in the SMIL document. For example, if a text is defined as below in the SMIL document: `<par dur="2000ms"> <text src="Cid:MyText.txt" region="Image"> <param name="foreground-color" value="#000000"/> <param name="textsize" value="large"/> </text> </par>` The corresponding MIME Content ID (`MyText.txt`) shall be defined in the SOAP attachment.
Returns		
	String	ID of the MMS delivery request.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException

Possible exceptions
MessageTooLongException
PolicyException

getMessageDeliveryStatus

Requests the status of a previous MMS delivery request identified by requestIdentifier.

Table 7-2 getMessageDeliveryStatus(requestIdentifier)

Parameter Name	Type	Description
Input		
requestIdentifier	String	ID of the SMS delivery request.
Returns		
	DeliveryStatusType[]	Array of DeliveryStatusType. Each entry in the array corresponds to a destination address as given when the message was sent. See "DeliveryStatusType" on page 7-8.

Possible exceptions
UnknownRequestIdentifierException
ServiceException

Receive message API

getReceivedMessages

Poll for new messages.

Table 7-3 getReceivedMessages(registrationIdentifier, priority)

Parameter Name	Type	Description
Input		
registrationIdentifier	String	The mailbox. Format is <mailbox ID>\<mailbox password> Mailbox ID and password are supplied by the service provider. Example: "tel:50000\apassword"
priority	MessagePriority	OPTIONAL. The priority of the messages to poll from the Parlay X gateway. All messages of the specified priority and higher will be retrieved. If not specified, all messages are returned, that is, the same as specifying priority Low.
Returns		
	MessageRef[]	Array of messages received.

Possible exceptions
InvalidArgumentException
ServiceException

Possible exceptions
PolicyException
UnknownRegistrationIdentifierException

getMessage

Reads a message. The data is returned as a SOAP-Attachment in the return message, not as a return parameter. The attachment is encoded according to MIME or DIME format.

Table 7-4 getMessage(messageRefIdentifier) Attachment[content]

Parameter Name	Type	Description
Input		
messageRefIdentifier	String	ID of the message.
Attachment[content]		Data returned in the SOAP Header of the response to the SOAP/HTTP request. **Note:** Sent as a SOAP-Attachment in the response. Not as a parameter in the SOAP envelope.
Returns		
Void		

Possible exceptions
InvalidArgumentException
ServiceException

Possible exceptions
PolicyException
UnknownMessageException

Message notification API

notifyMessageReception

Notifies an application of an incoming MMS. This method must be implemented on the server-side of the Parlay X client.

WebLogic Network Gatekeeper must be aware of the location of the Web Service implementing this interface. The location of the web service is defined using OAM in WebLogic Network Gatekeeper.

Table 7-5 notifyMessageReception (registrationIdentifier, messageRef)

Parameter Name	Type	Description
Input		
registrationIdentifier	String	ID of the mailbox.
messageRef	Message Ref	Contains information associated with the received message.
Returns		
Void		

Possible exceptions
InvalidArgumentException
ApplicationException
UnknownRegistrationIdentifierException

Data Types

Table 7-6 Data types specific for Messaging

Data type	Description	
MessagePriority	The MessagePriority data type is an enumeration with the following values:	
	Low	Low message priority.
	Normal	Normal message priority.
	High	High message priority.
DeliveryStatus	The DeliveryStatus data type is an enumeration with the following values:	
	Delivered	Successful delivery
	DeliveryUncertai n	Delivery status unknown: e.g. because it was handed off to another network.
	DeliveryImpossib le	Unsuccessful delivery; the message could not be delivered before it expired.
	MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.
DeliveryStatusType	Structure containing the following parameters:	

Table 7-6 Data types specific for Messaging

Data type	Description	
	destinationAddress	Address associated with the delivery status. The address field is coded as a URI. Datatype: EndUserIdentifier.
	deliveryStatus	Parameter indicating the delivery status. Datatype: DeliveryStatus
MessageRef	Structure containing the following parameters:	
	messageRefIdentifier	OPTIONAL. Contains a reference to a message stored in the Parlay X gateway. If the message is pure text, this parameter is not present. Datatype: String.
	messageServiceActivationNumber	Number associated with the invoked Message service, that is the destination address used by the terminal to send the message. Datatype: String.
	senderAddress	Indicates message sender address. Datatype: EndUserIdentifier.
	subject	OPTIONAL. Indicates the subject of the received message. This parameter will not be used for SMS services. Datatype: String.
	priority	The priority of the message: default is Normal. Datatype: MessagePriority.

Table 7-6 Data types specific for Messaging

Data type	Description	
	message	OPTIONAL. If present, the messageRefIdentifier is not present and this parameter contains the whole message. The type of the message is always pure ASCII text in this case. The message will not be stored in the Parlay X gateway. Datatype: String.
MessageURI	Structure containing the following parameters:	
	bodyText	Contains the message body if it is encoded as ASCII text. Datatype: String.
	fileReferences	Array of URL references to all the attachments in the Multimedia message. These are URLs to different files, e.g. GIF pictures or pure text files. Datatype: URL[].

Exceptions

Table 7-7 Exceptions specific for Messaging

ID	Exception	Description
MSG1000E	UnknownRequestIdentifierException	The Parlay X gateway did not recognize the requestIdentifier parameter. The message may have timed out or may have never been sent. This fault includes a string that provides additional information
MSG1001E	UnknownRegistrationIdentifierException	The provided registration identifier does not exist. This fault includes a string that provides additional information.
MSG1002E	UnknownMessageException	The provided messageRefIdentifier was not found in the Parlay X gateway. The message may have been timed out or it may never have been received by the gateway. This fault includes a string that provides additional information

Payment

The following sections describe the Payment API used for handling payment based on content for both pre- and post-paid services:

Amount charging API

chargeAmount

Charge an amount directly from an end user account.

Table 8-1 chargeAmount(endUserIdentifier, amount, billingText, referenceCode)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be charged.
amount	Decimal	The currency amount of the charge
billingText	String	Textual information to appear on the bill
referenceCode	String	Textual information to uniquely identify the request.
Returns		
Void		

Possible exceptions
ChargeFailureException
UnknownEndUserException
InvalidArgumentException

refundAmount

Refunds an amount directly from an end user account.

Table 8-2 refundAmount(endUserIdentifier, amount, billingText, referenceCode)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be refunded.
amount	Decimal	The currency amount of the charge
billingText	String	Textual information to appear on the bill
referenceCode	String	Textual information to uniquely identify the request.
Returns		
Void		

Possible exceptions
ChargeFailureException
UnknownEndUserException
InvalidArgumentException

Volume charging API

chargeVolume

Charges a volume directly from an end user account.

Table 8-3 chargeVolume(endUserIdentifier, volume, billingText, referenceCode)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be charged.
volume	Long	The volume to charge.
billingText	String	Textual information to appear on the bill.
referenceCode	String	Textual information to uniquely identify the request.
Returns		
Void		

Possible exceptions
ChargeFailureException
UnknownEndUserException
InvalidArgumentException

getAmount

Converts the given volume to a currency amount.

Table 8-4 getAmount(endUserIdentifier, volume)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be charged.
volume	Long	The volume to be converted.
Returns		
	Decimal	Currency amount resulting from the conversion process.

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

refundVolume

Refunds a volume to an account.

Table 8-5 refundVolume(endUserIdentifier, volume, billingText, referenceCode)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be charged.

Table 8-5 refundVolume(endUserIdentifier, volume, billingText, referenceCode)

Parameter Name	Type	Description
volume	Long	The volume to be refunded.
billingText	String	Textual information to appear on the bill.
referenceCode	String	Textual information to uniquely identify the request.
Returns		
	Decimal	Currency amount resulting from the conversion process.

Possible exceptions
ChargeFailureException
UnknownEndUserException
InvalidArgumentException

Reserved amount charging API

reserveAmount

Reserve an amount directly from an end user account.

Table 8-6 reserveAmount(endUserIdentifier, amount, billingText)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user’s account subject to the reservation.
amount	Decimal	The currency amount of the reservation.

Table 8-6 reserveAmount(endUserIdentifier, amount, billingText)

Parameter Name	Type	Description
billingText	String	Textual information to appear on the bill
Returns		
	String	Reservation Identifier. Textual information to uniquely identify the reservation.

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

reserveAdditionalAmount

Add an amount to a previously made reservation.

Table 8-7 reserveAdditionalAmount(reservationIdentifier, amount, billingText)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation to be amended.
amount	Decimal	The currency amount of the reservation.
billingText	String	Textual information to appear on the bill
Returns		
Void		

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

chargeReservation

Charge a previously made reservation.

Table 8-8 chargeReservation(reservationIdentifier, amount, billingText, referenceCode)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation to be charged.
amount	Decimal	The currency amount of the reservation.
billingText	String	Textual information to appear on the bill
referenceCode	String	Textual information to uniquely identify the request.
Returns		
Void		

Possible exceptions
ChargeFailureException

Possible exceptions
UnknownReservationException
InvalidArgumentException

releaseReservation

Release a previously made reservation. Returns the funds left in the reservation.

Table 8-9 releaseReservation(reservationIdentifier)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation.
Returns		
Void		

Possible exceptions
UnknownReservationException
ServiceException

Reserved volume charging API

getAmount

Converts the given volume to a currency amount.

Table 8-10 getAmount(endUserIdentifier, volume)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account to be charged.
volume	Long	The volume to be converted.
Returns		
	Decimal	Currency amount resulting from the conversion process.

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

reserveVolume

Reserve a volume directly from an end user account.

Table 8-11 reserveAmount(endUserIdentifier, amount, billingText)

Parameter Name	Type	Description
Input		
endUserIdentifier	EndUserIdentifier	The end user's account subject to the reservation.
volume	Long	The volume of the reservation.
billingText	String	Textual information to appear on the bill
Returns		
	String.	Reservation identifier. Textual information to uniquely identify the reservation.

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

reserveAdditionalVolume

Add a volume to a previously made reservation.

Table 8-12 reserveAdditionalVolume(reservationIdentifier, volume, billingText)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation to be amended.
volume	Long	The volume of the reservation.
billingText	String	Textual information to appear on the bill
Returns		
Void		

Possible exceptions
ServiceException
UnknownEndUserException
InvalidArgumentException

chargeReservation

Charge a previously made reservation.

Table 8-13 chargeReservation(reservationIdentifier, volume, billingText, referenceCode)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation to be charged.

Table 8-13 chargeReservation(reservationIdentifier, volume, billingText, referenceCode)

Parameter Name	Type	Description
volume	Long	The volume of the reservation.
billingText	String	Textual information to appear on the bill
referenceCode	String	Textual information to uniquely identify the request.
Returns		
Void		

Possible exceptions
ChargeFailureException
UnknownReservationException
InvalidArgumentException

releaseReservation

Release a previously made reservation. Returns the funds left in the reservation.

Table 8-14 releaseReservation(reservationIdentifier)

Parameter Name	Type	Description
Input		
reservationIdentifier	String	ID of the reservation.
Returns		
Void		

Possible exceptions
UnknownReservationException
ServiceException

Data Types

Table 8-15 Data types specific for Payment

Data type	Description	
DatedTransaction	Structure containing the following parameters:	
	transactionDate	Date the transaction occurred. Datatype: Date.
	transactionDetails	The transaction details. Datatype: String.

Exceptions

Table 8-16 Exceptions specific for Payment

ID	Exception	Description
ACM1000E	UnknownVoucherException	Indicates that the voucher identification that is passed is unknown.
ACM1001E	InadequateCredentialsException	Indicates that one of the credentials was left nil where that was not expected.

CHAPTER 9

Terminal Location

The following sections describe the Terminal location API used for retrieving the location of an end user's terminal:

Terminal location API

getLocation

Get the location of a mobile terminal.

Table 9-1 getLocation(endUser, requester, accuracy)

Parameter Name	Type	Description
Input		
endUser	EndUserIdentifier	End user for whom location information is being requested.
requester	EndUserIdentifier	OPTIONAL. The address of the requester.
accuracy	LocationAccuracy	The desired accuracy. Possible values are: [Low | Medium | High] Each operator must assign a “radius of uncertainty” to each value (e.g. < 3km, < 1km, < 100m, respectively)
Returns		
	LocationInfo	Location of the mobile terminal.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException
PolicyException

Data Types

Table 9-2 Data types specific for User Location

Data type	Description	
LocationInfo	Structure containing the following parameters:	
	longitude	Indicates the status of the end user. Datatype: Float
	latitude	Additional information if the userStatusIndicator is 'Other'. Datatype: Float
	accuracy	How accurate the Location information is. If the degree of accuracy wanted is available this should be given. If not, the best possible accuracy is returned. Possible values: [Low | Medium | High'].
	dateTime	Date and time the location information was obtained. Datatype: DateTime
LocationAccuracy	Enumeration with the following values:	
	Low	Low accuracy. That is, < 3 km radius of uncertainty.
	Medium	Medium accuracy. That is, < 1 km radius of uncertainty.
	High	High accuracy. That is, < 100 m radius of uncertainty.

Exceptions

No exceptions specific for User Location.

CHAPTER 10

User Status

The following sections describe the User Status API used for retrieving the status of an end user's terminal:

User status API

getUserStatus

Get the status of a mobile terminal.

Table 10-1 getUserStatus(endUser, requester)

Parameter Name	Type	Description
Input		
endUser	EndUserIdentifier	End user for whom user status information is being requested.
requester	EndUserIdentifier	OPTIONAL. The address of the requester.
Returns		
	UserStatusData	User status of the end user.

Possible exceptions
UnknownEndUserException
ServiceException
InvalidArgumentException
PolicyException

Data Types

Table 10-2 Data types specific for User Status

Data type	Description	
UserStatusData	Structure containing the following parameters:	
	userStatusIndicator	Indicates the status of the end user. Datatype: UserStatusIndicator
	additionalUserStatusInformation	Additional information if the userStatusIndicator is 'Other'. Datatype: String
UserStatusIndicator	Enumeration with the following values:	
	Online	User is online.
	Offline	User is offline (in case of a mobile terminal: switched off, in case of other terminal: not connected to the service), or wants to appear to be offline.
	Busy	User is busy.
	Other	Custom user status information can be retrieved from additionalUserStatusInformation.

Exceptions

No exceptions specific for User status.