WELCORP Web Service API

Reference Manual

Date Commenced: 3 August 2006

Last Modified: 18 August 2022

Version: 2.42

Table of Contents

Overview

The WEL Web Service API allows a registered user to submit a variety of messages and retrieve reports using SOAP over HTTP.

The location of the Web Service, service description and WSDL can be found at

https://www.welcorp.com/webservice/service.asmx

This document is technical in nature and it is assumed that the reader is familiar with SOAP and the implementation of web services.

A note on code samples: These have been provided in .Net c# and Java using callable methods; however, any high level SOAP toolkit should provide the same functionality. The web service is cross platform and conforms to the web service standards.

Service Addresses

The original WEL API is located at:

https://www.welcorp.com/webservice/service.asmx

For those who require access to a version using the SOAP DOCUMENT-LITERAL encoding, the address is:

https://www.welcorp.com/webservice/webserviceDL.asmx

Web Methods

Authentication

All web methods require an authentication header consisting of username and password (and optionally usercode) to be submitted within the SOAP header.

Authentication Header User Credentials:

Property	Type	Required	Description
Username	String	Yes	Assigned username
Password	String	Yes	Assigned password
Usercode	String	No	In the case that the user credentials belong to an administrator, the usercode of the job to be submitted under or retrieved, or of the user to be modified or deleted must be provided.

SubmitJob

This method is used to submit a Fax, Email, SMS, Text to Speech or Voice job to the WEL Corp messaging system.

Expects:

Job entity conforming to one of the following types:- FaxJob, EmailJob, SmsJob, TtsJob or VoiceJob

Returns:

Integer representing the job id or an exception on error.

******Important Note***

When sending multiple Fax jobs or multiple Text-to-Speech jobs, there is a non-trivial performance and time cost in converting the document/message, so in the case of multiple identical messages, users are **strongly** recommended to batch these together into one job, rather than send many individual jobs. This also applies to Voice jobs as uploading multiple large files can also be a performance bottleneck.

Job

Job is an abstract class – submitted jobs must be one of FaxJob, EmailJob, SmsJob, TtsJob or VoiceJob.

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Recipients[]	Recipient array	No*	Array of message recipients.
AdditionalReportEmails	String	No	List of additional emails to be CC'd on the report email

* One or more ListIds OR one or more Recipient required. Note that you may not use a combination of both ListIds AND Recipient as the object model differs between the two.

FaxJob

Extends **Job**

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Files[]	File	Yes	One or more file objects
Recipients[]	FaxRecipient array	No	Required if no ListIds submitted. FaxRecipient derives from Recipient
IsHighRes	Boolean	No	If true, fax is sent in high resolution.
CallbackURL	String	No	If present, a HTML post request will be sent to the specified URL when the system receives a Fax report. Note, this overrides the user setting.
CutOffHour	String	No	The hour that the job will be stopped. (format: 13:00)
DeDupe	Boolean	No	Dedupe the recipient set

EmailJob

Extends **Job**

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Subject	String	Yes	Email subject
Text	String	No**	Text version of email (required if no HTML version exists)
Html	String	No**	HTML version of email (required if no Text version exists)
Files[]	File array	No	One or more file objects to be attached to email
Recipients[]	EmailRecipient array	No	Required if no ListIds submitted. EmailRecipient derives from Recipient
DeDupe	Boolean	No	Dedupe the recipient set

** Either one or both of Text, Html is required

SmsJob

Extends **Job**
Also sends as MMS if a file is attached.

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Text	String	Yes	SMS text. Restriction of 3060 characters.
IsTwoWay	Boolean	Yes	Is this a two way SMS job?
SMSResponseEmailAddress	String	No	Email address to send the text of the SMS replies to. If blank, the users main email address will be used.
ExpiryHours	Byte	No	If this is a two way SMS job, how many hours before job should expire?
BatchReplies	Boolean	Yes	Should replies be batched at expiry? If not, replies will be sent individually to Sender.ReplyTo value as they arrive. For Two Way Jobs only.
Recipients[]	SmsRecipient array	No	Required if no ListIds submitted. SmsRecipient derives from Recipient
CallbackURL	String	No	If present, a HTML post request will be sent to the specified URL when the system receives a 2 way SMS reply. (Also see below field)
CallbackOnSMSStatusUpdate	bool	No	If present, and CallbackURL is set, then callbacks will be made to the specified URL on SMS status updates as well as 2 way SMS replies.
UseTwoWayCustomSenderID	bool	No	If present, and the job is a 2 Way SMS, then the SMS sender id will be the ReplyTo field given in your sender object, and not the WEL automated reply handling numbers. This means WEL will not get replies and optouts. *(Not for general use)*
CutOffHour	String	No	The hour that sms's will not be delivered past. NOTE, some carriers will deliver up to an hour after this time, and some carriers might ignore it completely. (format: 13:00)

OptoutCode	String	No	If a reply to a 2 Way SMS contains this string, then the recipient is added to the users optout list.
Dedupe	Boolean	No	Dedupe the recipient set
MMSFile	File	No	Send job as MMS if included. Supported file types are: `.gif, .jpg, .jpeg, .mp3, .mp4, .3gpp`
LinkInSMSPage	String	No	A Page template that will be merged with the recipient fields to create a custom Link In SMS page. See LinkInSMS details below.
LinkInSMSecQuestion	String	No	When using LinkInSMS, this question must be answered by the recipient before viewing the linked content
LinkInSMSSecAnswer	String	No	When using LinkInSMS, this answer must be entered by the recipient before viewing the linked content

TtsJob

Extends **Job**

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Text	String	Yes	Text to speech text.
IsTwoWay	Boolean	Yes	Is this a two way TTS job?
KeypressOnly	Boolean	Yes	If the purpose of your job is to collect keypresses only and not transfer to a third party, set this to true (along with "IsTwoWay")
SuppressHeader	Boolean	Yes	Suppress the introductory header?
PreferredDestinationType	Enum	No	UseDefault = 0 UseLandline = 1 UseMobile = 2
TransferKeys[]	TransferKey array	No	Required for Two Way TTS jobs (not required where KeypressOnly = true)
KeyAckMessages []	KeyAckMessage array	No	Optional for Two Way TTS jobs.
Recipients[]	TtsRecipient array	No	Required if no ListIds submitted. TtsRecipient derives from Recipient
Voice	String	No	Language/Voice that text will be read in.
HeaderText	String	No	Optional Header Text. Default will be used if blank
SkipMsgRepeat	Boolean	No	Message will be repeated unless this field is present and set to "true"
VoiceMailMessage	String	No	The message that will be left if the call is picked up by voicemail.
CutOffHour	String	No	The hour that the job will be stopped. (format: 13:00)
CallbackURL	String	No	If present, a HTML post request will be sent to the specified URL when the system receives a TTS report. Note, this overrides the user setting.
Dedupe	Boolean	No	Dedupe the recipient set

VoiceJob

Extends **Job**

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	A future date and time for the job to be activated
RepeatInterval	Integer	No	Causes a scheduled job to repeat every **"RepeatInterval"** seconds
ListIds[]	Integer array	No*	An array of list id's if using lists already uploaded to the WEL website.
Sender	Sender	No	Complex type consisting of sender information.
Recording	File	Yes	File object – only WAV files are accepted.
Recipients[]	VoiceRecipient array	No	Required if no ListIds submitted. VoiceRecipient derives from Recipient
PreferredDestinationType	Enum	No	UseDefault = 0 UseLandline = 1 UseMobile = 2
TransferKeys[]	TransferKey array	No	Required for Two Way Voice jobs (not required where KeypressOnly = true)
KeyAckMessages []	KeyAckMessage array	No	Optional for Two Way TTS jobs
IsTwoWay	Boolean	Yes	Is this a two way Voice job?
KeypressOnly	Boolean	Yes	If the purpose of your job is to collect keypresses only and not transfer to a third party, set to true. **Note:** you must also set IsTwoWay to true for keypresses to be collected
CutOffHour	String	No	The hour that the job will be stopped. (format: 13:00)
CallbackURL	String	No	If present, a HTML post request will be sent to the specified URL when the system receives a Fax report. Note, this overrides the user setting.
DeDupe	Boolean	No	Dedupe the Recipient Set

TtsDirectJob

Extends **Job**

Properties:

Property	Type	Required	Description
Name	String	Yes	A user defined name for the job
Scheduled	DateTime	No	Inherited but not used
RepeatInterval	Integer	No	Inherited but not used
ListIds[]	Integer array	No*	Inherited but not used
Sender	Sender	No	Inherited but not used
Text	String	Yes	Text to speech text.
Voice	String	No	Language/Voice that text will be read in.

Sender

Properties:

Property	Type	Required	Description
Name	String	No	Contact name of user submitting job. If not provided, the company name from the stored user profile will be used.
Company	String	No	Company name. If not provided, the company name from the stored user profile will be used. Company name will be mentioned in TTS broadcasts as part of the introductory message
ReplyTo	String	No	Reply email (for email job) or number/name (for SMS job). If not provided, details in stored user profile will be used. The reply email will be displayed in email broadcasts; the reply mobile number will be displayed in one way SMS jobs. Note, for SMS jobs, an alphanumeric sender id must be 11 characters or less.

File

Properties:

Property	Type	Required	Description
Name	String	Yes	File name including extension
Priority	Integer	No	Priority order for multiple files
Content[]	Byte array	Yes	Byte array of file content

Recipient

An abstract class representing a message recipient.

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Destination – e.g. fax number, email address, phone number or mobile number.

PhoneRecipient

An abstract class representing a phone recipient. Extends **Recipient**

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid phone, fax or mobile number

Valid number construction for Destination:-

Local number with area code	e.g. 02 9123 4567
International number with "+"	e.g. +44 2 1234 5678
International number with "0011"	e.g. 0011 44 2 1234 5678

FaxRecipient

Extends PhoneRecipient.

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid phone, fax or mobile number

EmailRecipient

Extends **Recipient**

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid email address
MergeField[]	MergeField array	No	A list of merge key/value pairs

Valid construction for Destination:- any valid email address

SmsRecipient

Extends **PhoneRecipient**

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid phone style number
MergeField[]	MergeField array	No	A list of merge key/value pairs. This can include the LinkInSMS page content. See Link In SMS section.

TtsRecipient

Extends **PhoneRecipient**

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid phone style number
MergeField[]	MergeField array	No	A list of merge key/value pairs

VoiceRecipient

Extends **PhoneRecipient**

Properties:

Property	Type	Required	Description
Reference	String	No	User defined reference for this recipient
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	No	First name of recipient
LastName	String	No	Last name of recipient
Destination	String	Yes	Valid phone style number

MergeField

Properties:

Property	Type	Required	Description
Key	String	Yes	Key relating to the merge field in message to be replaced.
Value	String	Yes	Value to replace key with for specific recipient.

TransferKey

Properties:

Property	Type	Required	Description
Keypress	Byte	Yes	Number that customer can press.
TransferNumber	String	Yes	Phone number to be transferred to.
MaxSimultaneousCalls	Integer	Yes	Maximum number of simultaneous calls allowed to transfer number.

KeyAckMessage

Properties:

Property	Type	Required	Description
Keypress	Byte	Yes	Number that customer can press.
Message	String	Yes	TTS Message to read out when key is pressed.

RetrieveReport

Use this method to retrieve information about a specific WEL job by Job ID.

Expects:

Integer **jobId** representing the job to be retrieved.

Returns:

StatusReport entity

RetrieveDetailedReport

Use this method to retrieve a detailed job report by Job ID.

Expects:

Integer **jobId** representing the job to be retrieved.

Returns:

DetailedReport entity

StatusReport

Entity representing the status report of a job.

Properties:

Property	Type	Description
JobId	Integer	The job id to be retrieved.
JobName	String	The user assigned Job name
Submitted	Dateime	The datetime that the job was submitted to the network. This is the system time + System Timezone offset.
JobType	String	FAX, Email, Voice, SMS, Text-To-Speech
JobStatus	String	Scheduled, Pending, Broadcast Submitted, Complete, Closed, Queued
TotalRecipients	Integer	Total number of recipients submitted
SentRecipients	Integer	Total number successfully sent to
FailedRecipients	Integer	Total number of failed recipients
TotalCost	Double	Total cost of job (if complete or closed)
RelatedJobs	String	For repeat scheduled tasks, returns a comma separated string of job IDs that were started by the supplied JobID.
Pages	Integer	The number of Fax pages per recipient sent for fax job, or the number of SMS messages per recipient.
ResentAs	String	If a job has been resent to the failed or errored recipients, this will have the ID of the resend job.
ResendOriginalID	String	If this job is the resend of a previous job, it will have the id of the original job

DetailedReport

Extends **StatusReport**

Properties:

Property	Type	Description
JobId	Integer	The job id to be retrieved.
Submitted	Dateime	The datetime that the job was submitted to the network. This is the system time + System Timezone offset.
JobType	String	FAX, Email, Voice, SMS, Text-To-Speech
JobStatus	String	Scheduled, Pending, Broadcast Submitted, Complete, Closed, Queued
TotalRecipients	Integer	Total number of recipients submitted
SentRecipients	Integer	Total number successfully sent to
FailedRecipients	Integer	Total number of failed recipients
TotalCost	Double	Total cost of job (if complete or closed)
RelatedJobs	String	For repeat scheduled tasks, returns a comma separated string of job IDs that were started by the supplied JobID.
Pages	Integer	The number of Fax pages per recipient sent for fax job, or the number of SMS messages per recipient.
ReportRecipient[]	ReportRecipient array	Array of completed recipients for the job

ReportRecipient

Properties:

Property	Type	Description
Recipient	String	Recipient name
Reference	String	Recipient reference
Destination	String	For user lists only:- fax, phone or email
Status	String	SENT, ERR etc.
Duration	Integer	Fax/TTS/voice jobs only – duration of message
Cost	Double	Cost of message
Keypress	Integer	2 Way TTS jobs only – keypress user selected
Reply	String	2 Way SMS jobs only – user reply
Voicemail	Boolean	True if the voice message went to Voice Mail

AddUser(User)

Please note: this method is available for administrator and agent privileges only.

Use this method to add users to the broadcast system.

Expects:

User object.

Returns:

Boolean indicating success or fail.

ModifyUser(User)

Please note: this method is available for administrator and agent privileges only.

Use this method to modify users to the broadcast system.

Expects:

User object. You must also provide the usercode of the user to be modified in the authentication header.

Note, only the fields that are not blank or null will be changed.

Returns:

Boolean indicating success or fail.

Note: To update a users password, pass in a **user** object with all fields blanks except for the new password. Similarly, the other fields can be updated individually.

DeleteUser(User)

Please note: this method is available for administrator and agent privileges only.

Use this method to suspend users from the broadcast system.

Expects:

The usercode of the user to be suspended in the authentication header.

Returns:

Boolean indicating success or fail.

Note: This does not remove the user from the system, but simply sets them to disabled.
Alternatively, you can use ModifyUser with **user.IsLive = false;**
To reactivate a suspended user, user ModifyUser with **user.IsLive = true;**

User

Entity representing a user object.

Properties:

Property	Type	Required	Description
UserCode	String	Yes*	User code of the user. This will become the authentication username.
Password	String	Yes*	User password
Title	String	No	Recipient title (e.g. Mr, Ms etc)
FirstName	String	Yes*	First name of recipient
LastName	String	Yes*	Last name of recipient
Email	String	Yes*	Email address
Phone	String	No	Phone number
Fax	String	No	Fax number
Mobile	String	No	Mobile/Cell number
Company	Company	Yes*	Company object (See below)
MonthlyLimit	Double	No	What is this user's monthly spend limit? If not provided this value is set to the system default of 5,000
IsFax	Boolean	Yes*	Activate fax for this user
IsSms	Boolean	Yes*	Activate SMS for this user
IsTts	Boolean	Yes*	Activate TTS for this user
IsEmail	Boolean	Yes*	Activate Email for this user
IsTtsTwoWay	Boolean	Yes*	Activate 2 way TTS for this user
ReportType **	Enum	No	Detailed = 0 (default), Exception = 1, Summary = 2
ListCosts[]	ListCosts array	No	Array of costs for this user. If not provided, values are set to the reseller default.
CallBackURL	String	No	A path for a status update callback. Leave blank for no callback
FaxHeaderText	String	No	An optional value for the text to go in the Fax header in place of the company name.
IsLive	Boolean	No	On User-Add, a null value will be taken as TRUE.
IsAntiSpam	Boolean	No	Set to true if the user has agreed to the Anti Spam policy on your system.
EmailToBroadcastEmailAddress	String	No	Alternate email to broadcast email address or domain: Email to Broadcast emails from this address or domain will use this account. This can be an email address or a domain address.

BroadcastLimitWarnAddress	String	No	Alternate email to sent credit/spend limit warnings to
EmailToBroadcastAuthType	String	No	Email to broadcast authentication type: Choose between one of: ● ChResp ● Password ● None
EmailToBroadcastSubjectBehaviour	String	No	How the incoming Email subject is handled: Choose between one of: ● JobName ● MessageConcat
EmailToBroadcastBodyAsFaxCover	String	No	Does the email body of an Email to Fax get used as a cover page: Choose between one of: ● Yes ● No
EmailToBroadcastSignatureStart	String	No	Text to mark the beginning of the signature area of an Email to broadcast email.
SMSSenderID	String	No	The default sender ID to use when sending SMS's. This must be between 1 and 11 alphanumeric characters only.
DefaultTTSVoice	String	No	The default voice to use for TTS messages. This must be one of the voice codes from the "TTS Language Support" section of this document
TTSRepeatCount	String	No	The default number of times to repeat the body of a TTS message. The text should be one of: ● "Two" ● "Three" ● "Four" (excluding the quotes)
CreditLimitForAlert	String	No	Decimal amount below which a Low Credit alert is sent to the user.
CreditLimitAlertAddress	String	No	Email address to which the low credit alert is sent.
DefaultSMSReplyEmail	String	No	Default email address for SMS replies to be sent to.
EtoBXcode	String	No	Code embedded in the header of an email to broadcast to identify the account used for sending.
HideCostsInReport	String	No	Option to hide the final costs in job reports. "Yes" or "No"(default).

SMSFilterOutNonAscii	String	No	Filter out non-ascii characters from outbound SMSs. Value must be “Yes” or “No”(default).
TTSCharLimit	String	No	Maximum number of characters that will be sent as a TTS. Extra characters will be discarded.
TTSMsgPrefix	String	No	String to be added at the start of all outgoing TTS message. Eg “\Speed=30”
DocRetentionHours	String	No	Number of hours after which documents are deleted from the server. Leave blank to not delete documents (default)
InboundFaxEmail	String	No	Email address for inbound faxes (if configured) to be emailed to.
ShowRecipientsInOnlineReport	String	No	Show recipient details in the online report. “Yes” (default) or “No”.
ShowRecipientsInEmailReport	String	No	Show recipient details in the emailed report. “Yes” (default) or “No”.
IncludeCopyOfFaxInReport	String	No	Option to include a copy of the sent fax in the emailed fax report. “Yes” or “No”(default).
HideAccounts	String	No	Do not display the web portal accounts page to the user. “Yes” or “No”(default).
EmailToBroadcastIPWhiteList	String	No	Comma separated list of IP addresses allowed to send or relay email to broadcast jobs for the user. Leave blank for no restriction.
EmailToBroadcastIPWhiteListAlertEmail	String	No	If a non-authorized IP address tries to send an email to broadcast job, an alert will be sent to this address if set.
TimeZoneID	String	No	Numeric ID for the users Timezone. See Appendix.
UserReceiveReports	String	No	If this option is included, its values should be one of “true” or “false”, and controls whether job reports are sent out.

* Required only for the **AddUser** method. When updating a user, only fields with values will change the user profile.

**** Report Types:**

Detailed: The user will receive a report listing summary details plus all recipients attempted, success or fail.

Exception: The user will receive a report listing summary details plus recipients who have failed.

Summary: The user will receive a report listing only the basic numbers of total attempted, number successful and number failed.

Company

Entity representing a company object.

Properties:

Property	Type	Required	Description
CompanyName	String	Yes	Company name (if not provided, the users name is utilized)
Address	String	No	Address
City	String	No	City or suburb
State	String	No*	State if applicable
Postcode	String	No	Post or zip code
Country	String	Yes	Country of user. If not provided will be set to the resellers default.

ListCosts

Entity representing a billing object.

Properties:

Property	Type	Required	Description
ListType	Enum	Yes	AuMedia = 1, (Australian media) AuBusinesss = 2, (Australian business) AuGovernment = 4, (Australian government) Messaging = 3, (User's own lists, web service lists) USMedia = 7, (USA media) AsiaMedia = 8 (Asia Pacific media)
Service	Enum	Yes	Fax = 1, Email = 2, SMS = 3, TextToSpeech = 4, TextToSpeechMobile = 5, TwoWaySMS = 9, TwoWayTTS = 10, TwoWayTTSMobile = 11 ***(NOTE, TTS costs also apply to "Voice" calls)***
Cost	Double	Yes	Cost in dollars. E.g. 0.5 is equivalent to 50 cents ***(NOTE, for TTS/Voice, this amount is for Flagfall AND ongoing costs)***

AddPrePay(float, string)

Please note: this method is available for administrator privileges only.

Use this method to add a prepayment to a “credit” users account.

Expects:

Float/double: The amount to add to the users credit, eg. 99.99.
String: A payment message for tracking purposes. Eg. “Paid by Visa on 12/12/2020, Auth:ABC123”

Returns:

Boolean indicating success or an error.

Cancel Job

Use this method to cancel unsent scheduled jobs from the WEL system.

Expects:

An Integer containing the Job number of the broadcast to cancel.
In the case of a reoccurring scheduled task, the latest Job number, or the original job number will cancel the scheduled Job.

Returns:

Boolean indicating success or fail.

TtsDirectStatus

Use this method to retrieve the status or associated URL for a TTSDirect Job.

Expects:

Integer **jobId** representing the job to be retrieved.

Returns:

String containing the job status or a URL to the resulting WAV file.

Using Merge Fields

Merge fields may be used in the following jobs only:-

1. Email
2. SMS
3. Text-To-Speech

To indicate a merge field in your message, the field should be enclosed by double percentage characters – e.g. `%%MyField%%`

There are reserved merge fields which you may use without specifically creating using the merge key/value pairs. These are:

1. `%%Title%%` - recipient's title
2. `%%FirstName%%` - recipient's first name
3. `%%LastName%%` - recipient's last name
4. `%%Recipient%%` - will be replaced by recipient's full name
5. `%%Reference%%` - will be replaced by the recipient's reference
6. `%%Email%%` - will be replaced by the recipient's destination for Email type jobs
7. `%%Mobile%%` - will be replaced by the recipient's destination for SMS type jobs
8. `%%Phone%%` - will be replaced by the recipient's destination for Text-To-Speech type jobs

Sample message:

Attention: `%%Recipient%%`

Dear `%%FirstName%%`,

According to our records you are currently overdue on payment for `%%BillName%%`. The amount owing is `%%BillAmount%%`.

Please organize payment before the due date of `%%DueDate%%`.

Sample Construction

```
<SmsRecipient>
      <Title>Mr</Title>
      <FirstName>Andrew</FirstName>
      <LastName>Citizen</LastName>
      <Reference>19462</Reference>
      <Destination>0412 345 678</Destination>
      <MergeField>
            <Key>BillName</Key>
            <Value>June Electricity Bill</Value>
      </MergeField>
      <MergeField>
            <Key>BillAmount</Key>
            <Value>$78.32</Value>
      </MergeField>
      <MergeField>
            <Key>DueDate</Key>
            <Value>1 August 2007</Value>
      </MergeField>
</SmsRecipient>
```

Code Samples

C# Examples

Notes:

The Wel Webservice should be setup as a "**Web reference**" pointing to http://www.welcorp.com/webservice/service.asmx in your project. Note, in recent versions of Visual studio, this is found under the "Advanced…" button in the "Service Reference" page.

```
public void TestFax()
{
	List<FaxRecipient> recipientList = new List<FaxRecipient>();
	FaxRecipient recipient = new FaxRecipient();
	recipient.Title = "Ms";
	recipient.FirstName = "Annette";
	recipient.LastName = "Citizen";
	recipient.Reference = "Of the world";
	recipient.Destination = "02 9123 4567";

	recipientList.Add(recipient);

	List<File> fileList = new List<File>();

	File myFile = new File();
	myFile.Name = "test.doc";
	myFile.Priority = 1;
	System.IO.FileStream file = new
		System.IO.FileStream("files/test.doc",
		System.IO.FileMode.Open, System.IO.FileAccess.Read);
	System.IO.BinaryReader br = new System.IO.BinaryReader(file);
	byte[] buffer = new byte[(int)file.Length];
	br.Read(buffer, 0, (int)file.Length);
	br.Close();

	myFile.Content = buffer;

	fileList.Add(myFile);

	FaxJob faxJob = new FaxJob();
	faxJob.Name = "Test Fax";
	faxJob.Sender = new Sender();
	faxJob.Sender.Company = "My Company";
	faxJob.Sender.Name = "My Name";
	faxJob.Sender.ReplyTo = "02 9876 5432";

	faxJob.Recipients = recipientList.ToArray();
	faxJob.Files = fileList.ToArray();

	AuthenticationHeader authHeader = new AuthenticationHeader();
	authHeader.Username = "test";
	authHeader.Password = "test";

	WelWebService service = new WelWebService();
	service.AuthenticationHeaderValue = authHeader;
```

```
        int jobid = service.SubmitJob(faxJob);
}
```

```
public void TestEmail()
{
      List<EmailRecipient> recipientList = new List<EmailRecipient>();

      EmailRecipient recipient = new EmailRecipient();
      recipient.Title = "Ms";
      recipient.FirstName = "Annette";
      recipient.LastName = "Citizen";
      recipient.Reference = "Of the world";
      recipient.Destination = "a.citizen@theworld.com ";
      List<MergeField> fieldList = new List<MergeField>();
      MergeField mergeField = new MergeField();
      mergeField.Key = "FavouriteAnimal";
      mergeField.Value = "Cats";
      fieldList.Add(mergeField);

      recipient.MergeFields = fieldList.ToArray();

      recipientList.Add(recipient);

      List<File> myFiles = new List<File>();
      File myFile = new File();
      myFile.Name = "test.jpg";
      myFile.Priority = 1;
      System.IO.FileStream file = new
            System.IO.FileStream("files/test.jpg",
            System.IO.FileMode.Open, System.IO.FileAccess.Read);
      System.IO.BinaryReader br = new System.IO.BinaryReader(file);
      byte[] buffer = new byte[(int)file.Length];
      br.Read(buffer, 0, (int)file.Length);
      br.Close();

      myFile.Content = buffer;
      myFiles.Add(myFile);

      EmailJob emailJob = new EmailJob();
      emailJob.Name = "Test Email";
      emailJob.Sender = new Sender();
      emailJob.Sender.Company = "My Company";
      emailJob.Sender.Name = "Test Person";
      emailJob.Sender.ReplyTo = "test@test.com";
      emailJob.Subject = "My Test Email";
      emailJob.Recipients = recipientList.ToArray();
      emailJob.Text = "Hi %%FirstName%%, your favourite animal is
                  %%FavouriteAnimal%%.";
      emailJob.Files = myFiles.ToArray();

      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      int jobid = service.SubmitJob(emailJob);
}
```

```
public void TestSMS()
{
      List<SmsRecipient> recipientList = new List<SmsRecipient>();

      SmsRecipient recipient = new SmsRecipient();
      recipient.Title = "Ms";
      recipient.FirstName = "Annette";
      recipient.LastName = "Citizen";
      recipient.Reference = "Of the world";
      recipient.Destination = "+61 4 9876 5432";

      List<MergeField> fieldList = new List<MergeField>();
      MergeField mergeField = new MergeField();
      mergeField.Key = "FavouriteAnimal";
      mergeField.Value = "Cats";
      fieldList.Add(mergeField);

      recipient.MergeFields = fieldList.ToArray();
      recipientList.Add(recipient);

      SmsJob smsJob = new SmsJob();
      smsJob.Name = "SMS Test";
      smsJob.Sender = new Sender();
      smsJob.Sender.Company = "My Company";
      smsJob.Sender.Name = "Test Person";
      smsJob.Sender.ReplyTo = "+61 4 1234 5678";
      smsJob.IsTwoWay = false;
      smsJob.Recipients = recipientList.ToArray();
      smsJob.Text = "Test SMS message - your favourite animal is
                    %%FavouriteAnimal%%.";

      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      int jobid = service.SubmitJob(smsJob);

}
```

```
public void TestTwoWaySMS()
{
      List<SmsRecipient> recipientList = new List<SmsRecipient>();

      SmsRecipient recipient = new SmsRecipient();
      recipient.Title = "Ms";
      recipient.FirstName = "Annette";
      recipient.LastName = "Citizen";
      recipient.Reference = "Of the world";
      recipient.Destination = "+61 4 9876 5432";

      List<MergeField> fieldList = new List<MergeField>();
      MergeField mergeField = new MergeField();
      mergeField.Key = "FavouriteAnimal";
      mergeField.Value = "Cats";
      fieldList.Add(mergeField);

      recipient.MergeFields = fieldList.ToArray();
      recipientList.Add(recipient);

      SmsJob smsJob = new SmsJob();
      smsJob.Name = "2 Way SMS Test";
      smsJob.Sender = new Sender();
      smsJob.Sender.Company = "My Company";
      smsJob.Sender.Name = "Test Person";
      smsJob.Sender.ReplyTo = "smsreplies@mycompany.com";
      smsJob.IsTwoWay = true;
      smsJob.BatchReplies = false;
      smsJob.ExpiryHours = 24;
      smsJob.Recipients = recipientList.ToArray();
      smsJob.Text = "Test SMS message - your favourite animal is
                     %%FavouriteAnimal%%. Reply to this to get 25% off
                     your next visit to Pet Care.";

      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      int jobid = service.SubmitJob(smsJob);

}
```

```
public void TestTTS()
{
	List<TtsRecipient> recipientList = new List<TtsRecipient>();

	TtsRecipient recipient = new TtsRecipient();
	recipient.Title = "Ms";
	recipient.FirstName = "Annette";
	recipient.LastName = "Citizen";
	recipient.Reference = "Of the world";
	recipient.Destination = "+61 4 9876 5432";

	List<MergeField> fieldList = new List<MergeField>();
	MergeField mergeField = new MergeField();
	mergeField.Key = "FavouriteAnimal";
	mergeField.Value = "Cats";
	fieldList.Add(mergeField);

	recipient.MergeFields = fieldList.ToArray();

	recipientList.Add(recipient);

	TtsJob ttsJob = new TtsJob();
	ttsJob.Name = "Test TTS";
	ttsJob.Sender = new Sender();
	ttsJob.Sender.Company = "My Company";
	ttsJob.Sender.Name = "Test Person";
	ttsJob.Sender.ReplyTo = "+61 4 1234 5678";
	ttsJob.IsTwoWay = false;

	ttsJob.Recipients = recipientList.ToArray();
	ttsJob.Text = "Test TTS message - your favourite animal is
		%%FavouriteAnimal%%.";

	AuthenticationHeader authHeader = new AuthenticationHeader();
	authHeader.Username = "test";
	authHeader.Password = "test";

	WelWebService service = new WelWebService();
	service.AuthenticationHeaderValue = authHeader;
	int jobid = service.SubmitJob(ttsJob);

}
```

```
public void TestTTSTwoWay()
{
      List<TtsRecipient> recipientList = new List<TtsRecipient>();

      TtsRecipient recipient = new TtsRecipient();
      recipient.Title = "Ms";
      recipient.FirstName = "Annette";
      recipient.LastName = "Citizen";
      recipient.Reference = "Of the world";
      recipient.Destination = "02 9123 4567";

      List<MergeField> fieldList = new List<MergeField>();
      MergeField mergeField = new MergeField();
      mergeField.Key = "FavouriteAnimal";
      mergeField.Value = "Cats";
      fieldList.Add(mergeField);

      recipient.MergeFields = fieldList.ToArray();

      recipientList.Add(recipient);

      TtsJob ttsJob = new TtsJob();
      ttsJob.Name = "Test Two Way TTS";
      ttsJob.Sender = new Sender();
      ttsJob.Sender.Company = "My Company";
      ttsJob.Sender.Name = "Joe Blow";
      ttsJob.Sender.ReplyTo = "02 9123 4567";

      ttsJob.IsTwoWay = true;
      List<TransferKey> transKeys = new List<TransferKey>();
      TransferKey transKey = new TransferKey();
      transKey.Keypress = 1;
      transKey.MaxSimultaneousCalls = 15;
      transKey.TransferNumber = "03 9123 4567";
      transKeys.Add(transKey);

      ttsJob.TransferKeys = transKeys.ToArray();
      ttsJob.Recipients = recipientList.ToArray();
      ttsJob.Text = "Test TTS message - your favourite animal is
                     %%FavouriteAnimal%%. To speak to a vet, please
                     press 1 now.";

      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      int jobid = service.SubmitJob(ttsJob);

}
```

```
public void TestVoice()
{
      List<VoiceRecipient> recipientList = new List<VoiceRecipient>();

      VoiceRecipient recipient = new VoiceRecipient();
      recipient.Title = "Ms";
      recipient.FirstName = "Annette";
      recipient.LastName = "Citizen";
      recipient.Reference = "Of the world";
      recipient.Destination = "02 9123 4567";

      recipientList.Add(recipient);

      File myFile = new File();
      myFile.Name = "test.wav";
      myFile.Priority = 1;
      System.IO.FileStream file = new
            System.IO.FileStream("files/test.wav",
            System.IO.FileMode.Open, System.IO.FileAccess.Read);
      System.IO.BinaryReader br = new System.IO.BinaryReader(file);
      byte[] buffer = new byte[(int)file.Length];
      br.Read(buffer, 0, (int)file.Length);
      br.Close();

      myFile.Content = buffer;

      VoiceJob voiceJob = new VoiceJob();
      voiceJob.Name = "WAV File Test";
      voiceJob.Sender = new Sender();
      voiceJob.Sender.Company = "My Company";
      voiceJob.Sender.Name = "My Name";

      voiceJob.Recipients = recipientList.ToArray();
      voiceJob.Recording = myFile;

      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      int jobid = service.SubmitJob(voiceJob);

}
```

```
public void GetReport()
{
      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      StatusReport statusReport = service.RetrieveReport(1234);
}


public void GetDetailedReport()
{
      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      DetailedReport statusReport =
                  service.RetrieveDetailedReport(1234);
}
```

```
public void TestAddUser()
{
	AuthenticationHeader authHeader = new AuthenticationHeader();
	authHeader.Username = "testadmin";
	authHeader.Password = "test";

	WelWebService service = new WelWebService();
	service.AuthenticationHeaderValue = authHeader;
	User user = new User();
	user.UserCode = "au/wsnewuser";
	user.Title = "Ms";
	user.FirstName = "Rita";
	user.LastName = "Mae";
	user.Email = "rita@somewhere.com";
	user.Phone = "03 9123 4567";
	user.Mobile = "0412 345 678";
	user.Fax = "03 9123 4567";
	user.Password = "wsnewuser";
	user.ReportType = ReportType.Detailed;
	user.IsEmail = true;
	user.IsFax = true;
	user.IsSms = true;
	user.IsTts = false;
	user.IsTtsTwoWay = false;

	Company company = new Company();
	company.CompanyName = "My Company";
	company.Address = "1 Green St";
	company.City = "Melbourne";
	company.State = "VIC";
	company.Postcode = "3000";
	company.Country = "Australia";

	user.Company = company;

	List<ListCosts> listCosts = new List<ListCosts>();

	ListCosts listCost = new ListCosts();
	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.Fax;
	listCost.Cost = 0.25;
	listCosts.Add(listCost);

	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.SMS;
	listCost.Cost = 0.2;
	listCosts.Add(listCost);

	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.Email;
	listCost.Cost = 0.1;
	listCosts.Add(listCost);

	user.ListCosts = listCosts.ToArray();

	bool success = service.AddUser(user);
}
```

```
public void TestModifyUser()
{
	AuthenticationHeader authHeader = new AuthenticationHeader();
	authHeader.Username = "testadmin";
	authHeader.Password = "test";
	authHeader.Usercode = "au/wsnewuser";

	WelWebService service = new WelWebService();
	service.AuthenticationHeaderValue = authHeader;
	User user = new User();
	user.Title = "Ms";
	user.FirstName = "Rita";
	user.LastName = "Mae";
	user.Email = "rita@somewhere.com";
	user.Password = "wsnewuser";
	user.ReportType = ReportType.Detailed;
	user.IsEmail = true;
	user.IsFax = true;
	user.IsSms = true;
	user.IsTts = false;
	user.IsTtsTwoWay = false;

	Company company = new Company();
	company.CompanyName = "My Company";
	company.Address = "1 Green St";
	company.City = "Melbourne";
	company.State = "VIC";
	company.Postcode = "3000";
	company.Country = "Australia";

	user.Company = company;

	List<ListCosts> listCosts = new List<ListCosts>();

	ListCosts listCost = new ListCosts();
	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.Fax;
	listCost.Cost = 0.25;
	listCosts.Add(listCost);

	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.SMS;
	listCost.Cost = 0.2;
	listCosts.Add(listCost);

	listCost.ListType = ListType.Messaging;
	listCost.Service = Service.Email;
	listCost.Cost = 0.1;
	listCosts.Add(listCost);

	user.ListCosts = listCosts.ToArray();

	bool success = service.ModifyUser(user);
}
```

```
public void TestCancelJob ()
{
      AuthenticationHeader authHeader = new AuthenticationHeader();
      authHeader.Username = "test";
      authHeader.Password = "test";

      WelWebService service = new WelWebService();
      service.AuthenticationHeaderValue = authHeader;
      bool success = service.CancelJob(1234);
}
```

Java Axis Examples

Connecting to the Wel Web Service using Java and Axis 1.4

This document assumes that java and axis are setup appropriately, and the user has some basic familiarity with them.

1. Create the Wel Web Service interface classes:

   To create the WelService class files and the service Stub run:

   ```
   java org.apache.axis.wsdl.WSDL2Java (WSDL-file-URL)
   ```

   as per the axis documentation, then import the resultant classes into your java project.

2. Update Wel Web Service stubs to provide the required Authentication Header.

   In the two stub files:

   ```
   Wel_x0020_Web_x0020_ServiceSoapStub
   Wel_x0020_Web_x0020_ServiceSoap12Stub
   ```

   Insert the below code lines after the statement:

   ```
   setRequestHeaders(_call);
   ```

   in the **`submitJob`**, **`retrieveReport`** and **`retrieveDetailedReport`** methods

   ```
   org.apache.axis.message.SOAPHeaderElement auth = new
   SOAPHeaderElement("http://www.welcorp.com/webservice/","AuthenticationHeader")
   ;
   auth.addChildElement("Username","").addTextNode("test");
   auth.addChildElement("Password","").addTextNode("test");
   _call.addHeader(auth);
   ```

3. Main Sample Code

```
/*
 * Main.java
 *
 * Created on 24 November 2006, 20:51
 *
 */

package weltest;

import com.welcorp.www.webservice.*;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.Calendar;
import java.util.LinkedList;
import org.apache.axis.types.UnsignedByte;

public class Main {

    public Main() {
    }

    public static void main(String[] args) {
        try {
            TestFax();
            //TestEmail();
            //TestSMS();
            //TestTwoWaySMS();
            //testTTS();
            //TestTTSTwoWay();
            //TestVoice();
            //GetReport();
            //GetDetailsReport();
        } catch (Exception e){
            System.err.println(e.toString());
        }
    }
```

```
    static public void TestFax() throws IOException{
        // Add Fax Recipients
        LinkedList recipientList = new LinkedList();
        FaxRecipient recipient = new FaxRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("02 9123 4567");
        recipientList.add(recipient);
        LinkedList fileList = new LinkedList();

        // Add Files to Fax
        File myFile = new File();
        myFile.setName("test.doc");
        myFile.setPriority(1);

        java.io.File inFile = new java.io.File("c:/test.doc");
        InputStream is = new FileInputStream(inFile);
        long iFlength = inFile.length();
        byte[] buffer = new byte[(int)iFlength];
        int offset = 0;
        int numRead = 0;
        while (offset < buffer.length && (numRead=is.read(buffer, offset,
buffer.length-offset)) >= 0) offset += numRead;
        is.close();

        myFile.setContent(buffer);
        fileList.add(myFile);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setCompany("My Company");
        sender.setName("My Name");
        sender.setReplyTo("02 9876 5432");

        //Setup FaxJob and add sender, recipients and files
        FaxJob faxJob = new FaxJob();
        faxJob.setName("Test Fax");
        faxJob.setSender(sender);
        faxJob.setRecipients((FaxRecipient[])recipientList.toArray(new
FaxRecipient[1]));
        faxJob.setFiles((File[])fileList.toArray(new File[1]));

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        faxJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(faxJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void TestEmail() throws IOException{
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        EmailRecipient recipient = new EmailRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("a.citizen@theworld.com ");

        //Create and add merge fields for recipient, then record recipient
        LinkedList fieldList = new LinkedList();
        MergeField mergeField = new MergeField();
        mergeField.setKey("FavouriteAnimal");
        mergeField.setValue("Cats");
        fieldList.add(mergeField);
        recipient.setMergeFields((MergeField[])fieldList.toArray(new
MergeField[1]));
        recipientList.add(recipient);

        //Add attachment files
        LinkedList fileList = new LinkedList();
        File myFile = new File();
        myFile.setName("test.jpg");
        myFile.setPriority(1);

        java.io.File inFile = new java.io.File("c:/test.jpg");
        InputStream is = new FileInputStream(inFile);
        long iFlength = inFile.length();
        byte[] buffer = new byte[(int)iFlength];
        int offset = 0;
        int numRead = 0;
        while (offset < buffer.length && (numRead=is.read(buffer, offset,
buffer.length-offset)) >= 0) offset += numRead;
        is.close();

        myFile.setContent(buffer);
        fileList.add(myFile);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setName("Test Person");
        sender.setCompany("My Company");
        sender.setReplyTo("test@test.com");

        //Setup EmailJob and add sender, recipients and files
        EmailJob emailJob = new EmailJob();
        emailJob.setName("Test Email");
        emailJob.setSender(sender);
        emailJob.setSubject("My Test Email");
        emailJob.setRecipients((EmailRecipient[])recipientList.toArray(new
EmailRecipient[1]));
        emailJob.setText("Hi %%FirstName%%, your favourite animal is
%%FavouriteAnimal%%.");
        emailJob.setFiles((File[])fileList.toArray(new File[1]));

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        emailJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(emailJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }

    }
```

```
    static public void TestSMS(){
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        SmsRecipient recipient = new SmsRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("+61 4 9876 5432");

        //Create and add merge fields for recipient, then record recipient
        LinkedList fieldList = new LinkedList();
        MergeField mergeField = new MergeField();
        mergeField.setKey("FavouriteAnimal");
        mergeField.setValue("Cats");
        fieldList.add(mergeField);
        recipient.setMergeFields((MergeField[])fieldList.toArray(new
MergeField[1]));
        recipientList.add(recipient);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setName("Test Person");
        sender.setCompany("My Company");
        sender.setReplyTo("+61 4 1234 5678");

        //Setup SmsJob and add sender, recipients and files
        SmsJob smsJob = new SmsJob();
        smsJob.setName("SMS Test");
        smsJob.setSender(sender);
        smsJob.setIsTwoWay(false);
        smsJob.setRecipients((SmsRecipient[])recipientList.toArray(new
SmsRecipient[1]));
        smsJob.setText("Test SMS message - your favourite animal is
%%FavouriteAnimal%%.");

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        smsJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(smsJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void TestTwoWaySMS(){
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        SmsRecipient recipient = new SmsRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("+61 4 9876 5432");

        //Create and add merge fields for recipient, then record recipient
        LinkedList fieldList = new LinkedList();
        MergeField mergeField = new MergeField();
        mergeField.setKey("FavouriteAnimal");
        mergeField.setValue("Cats");
        fieldList.add(mergeField);
        recipient.setMergeFields((MergeField[])fieldList.toArray(new
MergeField[1]));
        recipientList.add(recipient);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setName("Test Person");
        sender.setCompany("My Company");
        sender.setReplyTo("smsreplies@mycompany.com");

        //Setup SmsJob and add sender, recipients and files
        SmsJob smsJob = new SmsJob();
        smsJob.setName("SMS Test");
        smsJob.setSender(sender);
        smsJob.setIsTwoWay(true);
        smsJob.setExpiryHours(new UnsignedByte(24));
        smsJob.setRecipients((SmsRecipient[])recipientList.toArray(new
SmsRecipient[1]));
        smsJob.setText("Test SMS message - your favourite animal is
%%FavouriteAnimal%%." +
                "Reply to this to get 25% off your next visit to Pet Care.");

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        smsJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(smsJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void testTTS(){
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        TtsRecipient recipient = new TtsRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("+61 4 9876 5432");

        //Create and add merge fields for recipient, then record recipient
        LinkedList fieldList = new LinkedList();
        MergeField mergeField = new MergeField();
        mergeField.setKey("FavouriteAnimal");
        mergeField.setValue("Cats");
        fieldList.add(mergeField);
        recipient.setMergeFields((MergeField[])fieldList.toArray(new
MergeField[1]));
        recipientList.add(recipient);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setName("Test Person");
        sender.setCompany("My Company");
        sender.setReplyTo("+61 4 1234 5678");

        //Setup TTSJob and add sender, recipients and files
        TtsJob ttsJob = new TtsJob();
        ttsJob.setName("TTS Test");
        ttsJob.setSender(sender);
        ttsJob.setIsTwoWay(false);
        ttsJob.setRecipients((TtsRecipient[])recipientList.toArray(new
TtsRecipient[1]));
        ttsJob.setText("Test TTS message - your favourite animal is
%%FavouriteAnimal%%.");
        ttsJob.setPreferredDestinationType(PreferredDestinationType.UseDefault);

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        ttsJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(ttsJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void TestTTSTwoWay(){
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        TtsRecipient recipient = new TtsRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("02 9123 4567");

        //Create and add merge fields for recipient, then record recipient
        LinkedList fieldList = new LinkedList();
        MergeField mergeField = new MergeField();
        mergeField.setKey("FavouriteAnimal");
        mergeField.setValue("Cats");
        fieldList.add(mergeField);
        recipient.setMergeFields((MergeField[])fieldList.toArray(new
MergeField[1]));
        recipientList.add(recipient);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setName("Test Person");
        sender.setCompany("My Company");
        sender.setReplyTo("02 9876 5432");

        //Setup TransferKeys
        LinkedList transKeys = new LinkedList();
        TransferKey transKey = new TransferKey();
        transKey.setKeypress(new UnsignedByte(1));
        transKey.setMaxSimultaneousCalls(15);
        transKey.setTransferNumber("03 9123 8765");
        transKeys.add(transKey);

        //Setup TTSJob and add sender, recipients and files
        TtsJob ttsJob = new TtsJob();
        ttsJob.setName("Test Two Wat TTS");
        ttsJob.setSender(sender);
        ttsJob.setIsTwoWay(true);
        ttsJob.setTransferKeys((TransferKey[])transKeys.toArray(new
TransferKey[1]));
        ttsJob.setRecipients((TtsRecipient[])recipientList.toArray(new
TtsRecipient[1]));
        ttsJob.setText("Test TTS message - your favourite animal is
%%FavouriteAnimal%%. " +
                "your favourite animal is %%FavouriteAnimal%%. To speak to a vet,
please press 1 now.");
        ttsJob.setPreferredDestinationType(PreferredDestinationType.UseDefault);

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        ttsJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(ttsJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void TestVoice() throws Exception{
        // Add Recipients
        LinkedList recipientList = new LinkedList();
        VoiceRecipient recipient = new VoiceRecipient();
        recipient.setTitle("Ms");
        recipient.setFirstName("Annette");
        recipient.setLastName("Citizen");
        recipient.setReference("Of the world");
        recipient.setDestination("02 9123 4567");
        recipientList.add(recipient);

        // Add Voice File
        File myFile = new File();
        myFile.setName("test.wav");
        myFile.setPriority(1);

        // Add Files to Fax
        java.io.File inFile = new java.io.File("c:/test.wav");
        InputStream is = new FileInputStream(inFile);
        long iFlength = inFile.length();
        byte[] buffer = new byte[(int)iFlength];
        int offset = 0;
        int numRead = 0;
        while (offset < buffer.length && (numRead=is.read(buffer, offset,
buffer.length-offset)) >= 0) offset += numRead;
        is.close();

        myFile.setContent(buffer);

        //Set Sender Details
        Sender sender = new Sender();
        sender.setCompany("My Company");
        sender.setName("My Name");

        //Setup VoiceJob and add sender, recipients and files
        VoiceJob voiceJob = new VoiceJob();
        voiceJob.setName("WAV File Test");
        voiceJob.setSender(sender);
        voiceJob.setRecipients((VoiceRecipient[])recipientList.toArray(new
VoiceRecipient[1]));
        voiceJob.setRecording(myFile);
        voiceJob.setPreferredDestinationType(PreferredDestinationType.UseDefault);

        Calendar nextWeek = Calendar.getInstance();
        nextWeek.add(Calendar.DAY_OF_WEEK, 7);
        voiceJob.setScheduled(nextWeek);

        try {
            int iret = SendJob(voiceJob);
            System.out.println(iret);
        } catch (Exception e) {
            System.err.println(e.toString());
        }
    }
```

```
    static public void GetReport() throws Exception{

        // Make a service
        Wel_x0020_Web_x0020_Service service = new
Wel_x0020_Web_x0020_ServiceLocator();
        // Now use the service to get a stub which implements the SDI.
        Wel_x0020_Web_x0020_ServiceSoap WelWebService =
service.getWel_x0020_Web_x0020_ServiceSoap();

        StatusReport report = WelWebService.retrieveReport(1234);

        System.out.println(report.getJobStatus());
    }



    static public void GetDetailsReport() throws Exception{
        // Make a service
        Wel_x0020_Web_x0020_Service service = new
Wel_x0020_Web_x0020_ServiceLocator();
        // Now use the service to get a stub which implements the SDI.
        Wel_x0020_Web_x0020_ServiceSoap WelWebService =
service.getWel_x0020_Web_x0020_ServiceSoap();

        StatusReport report = WelWebService.retrieveDetailedReport(1234);

        System.out.println(report.getJobStatus());

    }



    static public int SendJob(Job oJob) throws Exception{
        // Make a service
        Wel_x0020_Web_x0020_Service service = new
Wel_x0020_Web_x0020_ServiceLocator();
        // Now use the service to get a stub which implements the SDI.
        Wel_x0020_Web_x0020_ServiceSoap WelWebService =
service.getWel_x0020_Web_x0020_ServiceSoap();

        int iRes = WelWebService.submitJob(oJob);

        return iRes;
    }
}


    static public void CancelJob () throws Exception{
        // Make a service
        Wel_x0020_Web_x0020_Service service = new
Wel_x0020_Web_x0020_ServiceLocator();
        // Now use the service to get a stub which implements the SDI.
        Wel_x0020_Web_x0020_ServiceSoap WelWebService =
service.getWel_x0020_Web_x0020_ServiceSoap();

        Boolean success = WelWebService.canceljob(1234);

        System.out.println(success);

    }
```

PHP Code Example

Connecting to the Wel Web Service using PHP and NuSoap

This document assumes that PHP is installed and configured, and the newsoap library is downloaded and installed.

Change '`./nusoap/lib/nusoap.php`' to match the location of your nusoap library.

Basic SMSJob example:

```
<html><body>
<?php
	require_once('./nusoap/lib/nusoap.php');

	$namespace = "http://www.welcorp.com/webservice/";

	//Setup Header
	$hdrArr = array(
		'Username' => 'test',
		'Usercode' => 'test',
		'Password' => 'test');

	//Create properly typed header object
	$hdr = array('AuthenticationHeader' => new
soapval('AuthenticationHeader', 'AuthenticationHeader', $hdrArr, false,
$namespace));

	//Setup Job parameters
	$jobarr = array(
		'Name' => 'Test Job name',
		'Sender' => array(
			'ReplyTo' => '098765432'
		),
		'Recipients' => array(
			'Recip_1' => array(
				'Destination' => '+098765432',
				'Reference' => 'testing'
			)
		),
		'Text' => 'Test message goes here.',
		'IsTwoWay' => false,
		'BatchReplies' => false,
		'ExpiryHours' => new soapval('ExpiryHours', 'unsignedByte', 0,
false, 'http://www.w3.org/2001/XMLSchema')  //Need this soapval, otherwise it
goes out as an int, not an unsignedbyte
	);

	//Create properly typed job object
	$smsJob = array(new soapval('job', 'SmsJob', $jobarr, false,
$namespace));

	//create client and get wsdl
	$client = new
nusoap_client('http://www.welcorp.com/webservice/service.asmx?WSDL', 'wsdl');

	$err = $client->getError();
	if ($err) echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';

	//Do call
```

```
	$result = $client->call('SubmitJob', $smsJob, $namespace, false, $hdr);

	// Display the result
	print_r($result);
	// Display the request and response
	echo '<h2>Request</h2>';
	echo '<pre>' . htmlspecialchars($client->request, ENT_QUOTES) . '</pre>';
	echo '<h2>Response</h2>';
	echo '<pre>' . htmlspecialchars($client->response, ENT_QUOTES) .
'</pre>';

?>
</body></html>
```

Basic FAXJob example:

```
<html><body>
<?php
	require_once('nusoap.php');

	$namespace = "http://www.welcorp.com/webservice/";

	//setup Header
	$hdrArr = array(
		'Username' => 'test',
		'Password' => 'test');
	//create properly typed header object
	$hdr = array('AuthenticationHeader' => new
soapval('AuthenticationHeader', 'AuthenticationHeader', $hdrArr,
false,$namespace));

	//setup job parameters
	// Open File
	$file = "test.pdf";
	if( !($fp = fopen($file, "r")))
	{
	// Error opening file
	// Handle error as appropriate for your script
	exit;
	}

	// Read data from the file into $data
	$data = "";
	while (!feof($fp)) $data .= fread($fp,1024);


	$filearr = array("Test.pdf", false, base64_encode($data));


	$jobarr = array(
		'Name' => 'test api',
		'Sender' => array(
			'ReplyTo' => '+18178008516'
		),
		'Recipients' => array(
			'Recip_1' => array(
				'Destination' => '+33957029251',
				'Reference' => 'testAPI'
			)
		),
		'Files' => array(new soapval('TestFile', 'File', array($filearr),
false, $namespace),
```

```
            'ExpiryHours' => new soapval('ExpiryHours', 'unsignedByte', 0,
false, 'http://www.w3.org/2001/XMLSchema')     //Need this soapval, otherwise it
goes out as an int, not an unsigned byte
      );


      //create properly typed job object
      $smsJob = array(new soapval('job', 'FaxJob', $jobarr, false,
$namespace));

      //create client and get wsdl
      $client = new
nusoap_client('http://www.welcorp.com/webservice/service.asmx?WSDL', 'wsdl');

      $err = $client->getError();
      if ($err) echo '<h2>Construction error</h2><pre>' . $err . '</pre>';

      //Do call

      $result = $client->call('SubmitJob', $smsJob, $namespace, false, $hdr);

      //Display Result
      print_r($result);

      //Display the request and response
      echo '<h2>Request</h2>';
      echo '<pre>' . htmlspecialchars($client->request, ENT_QUOTES) . '</pre>';
      echo '<h2>Response</h2>';
      echo '<pre>' . htmlspecialchars($client->response, ENT_QUOTES) .
'</pre>';
      ?>
      </body>
</html>
```

Add User example:

```
<html><body>
<?php
      require_once('nusoap.php');

      $namespace = "http://www.welcorp.com/webservice/";

      //Setup Header
      $hdrArr = array(
            'Username' => 'agent_or_admin_username',
            'Password' => 'agent_or_admin_password');

      //Create properly typed header object
      $hdr = array('AuthenticationHeader' => new soapval('AuthenticationHeader',
'AuthenticationHeader', $hdrArr, false, $namespace));

      //Setup User
      $userarr = array(
      'UserCode' => 'au/apiuseraddtest2',
      'Password' => 'Testing123',
      'FirstName' => 'FN-APITest2',
      'LastName' => 'LN-APITest2',
```

```
        'Email' => 'apitest2@test.com',
        'IsFax' => false,
        'IsSms' => false,
        'IsTts' => false,
        'IsEmail' => true,
        'IsTtsTwoWay' => false);

        //create properly typed user object
        $userobj = array(new soapval('newuser', 'User', $userarr, false, $namespace));

        //create client and get wsdl
        $client = new nusoap_client('http://www.welcorp.com/webservice/service.asmx?WSDL', 'wsdl');

        $err = $client->getError();
        if ($err) echo '<h2>Constructor error</h2><pre>' . $err . '</pre>';

        //Do call
        $result = $client->call('AddUser', $userobj, $namespace, false, $hdr);

        // Display the result
        print_r($result);

        // Display the request and response
        echo '<h2>Request</h2>';
        echo '<pre>' . htmlspecialchars($client->request, ENT_QUOTES) . '</pre>';
        echo '<h2>Response</h2>';
        echo '<pre>' . htmlspecialchars($client->response, ENT_QUOTES) . '</pre>';

?>
        </body>
</html>
```

Appendix

Valid number formats

Valid phone, fax or mobile number construction:-

Local number with area code	e.g. 02 9123 4567
International number with "+"	e.g. +44 2 1234 5678
International number with "0011"	e.g. 0011 44 2 1234 5678

Other number constructions are invalid.

Broadcast Status Codes

List Created	User has created their job, but has not yet submitted it.
Pending	For SMS only:- job is in the pending queue.
Retry	For SMS only:- aggregator cannot be contacted, job has been re-queued.
Queued By WS	For Web Service submissions only:- job has been queued and awaiting processing.
Test Sent	For Fax and TTS only:- user has requested a test send and is awaiting results.
Scheduled	Job is scheduled for later date and time.
Broadcast Submitted	Job has been submitted and is currently in progress.
Complete	Job is complete.
Closed	Job has been closed (either completed or cancelled).
Cancelled	Scheduled job has been cancelled by user.

SMS Message Lengths

1 message	Up to 160 characters.
2+ messages	153 characters per SMS message. (so 400 characters would take 3 messages)

Current maximum is 3060 characters, for a 20 part SMS message

Send Codes

SENT	Message was successfully sent to the recipient.
Fax Specific Error Codes	
BUSY	Line gave busy signal with all retries. The receiving fax machine was busy.
NOAN	Phone keeps ringing. Nobody answers. If you are sure that the number is correct, please contact the recipient. There may be a problem with their fax machine (e.g. out of paper).
INVD	The number is not a valid number. Please check the number and dial again.
NOLN	Problem with phone line. Please check line.
BARR	Wrong number (number changed, call barred).
NSUP	Service or Function not supported by network or user.
COLL	At the moment we started trying to send a fax a call came in.
ERR	Procedural or unknown error.
FAIL	Document conversion failed. Please resubmit your document.
NOFX	No fax machine detected. Please check the number and try again.
PAGC	No confirmation received after last page transmitted. The fax may have been received by the recipient and the machine failed to confirm.
BADC	Bad connection or fax modem error at receiving end.
OPTO	Number in users optoutlist
REFU	Call refused by recipient or carrier
Text-To-Speech Specific Error Codes	
BUSY	Line busy.
BSIN	Incorrect number
NOAN	Phone keeps ringing. Nobody answers.
MGNC	Receipt of message not confirmed.
OPTO	Number in users optoutlist
REFU	Call refused by recipient or carrier
ERR	Procedural or unknown error
LDTE	Latest Delivery timeout expired
NOLN	Problem with phone line. Please check line.
HNUP	Call disconnected by the network or remote user (reason unspecified).
RJCT	The message has been rejected
SMS Specific Error Codes	
DERR	Aggregator error.
UERR	User validation error.
TRAN	Transaction limit reached.
PERR	Invalid password.
SVRE	SMS server error - please retry later.
SNDE	Error in sender number.
RECE	Error in receiver number.
MSGE	No SMS message.
LENE	SMS message too long.
NUME	Sending number invalid.
OUTE	SMS outgoing update error.

Notes:

For SMS: Status SENT indicates that the message has been successfully submitted to the network. In some cases the message may fail after if the number does not exist or for some other network related issue. In these cases, the status will be updated accordingly up to four hours after submission.

For Email: All emails are submitted and the system must wait to either receive bounced back emails, or to receive notification that an email has been read. The system cannot ascertain if all emails are read – if a recipient uses an HTML email package and does not block outgoing code then the "Read" status will be received. However, this is a minority of recipients. After 12 hours the job is closed and all emails that have not failed are updated to "Sent".

TTS Language Support.

The valid values for the TTSJob voice parameter are:

Voice	Comment
English-Allison	American English, female. **Default Voice**
English-Alan	Australian English, male
English-Grace	Australian English, female
English-Dave	American English, male
English-Steven	American English, male
English-Susan	American English, female
English-Kate	British English, female
English-Simon	British English, male
Chinese-Linlin	Chinese, female
Chinese-Lisheng	Chinese, female
French-Juliette	French, female
French-Charlotte	Canadian French, female
French-Olivier	Canadian French, male
German-Katrin	German, female
Luca-Italian	Italian, male
Mexican-Esperanza	Mexican, female
Portuguese-Felipe	Brazilian Portuguese, male
Portuguese-Fernanda	Brazilian Portuguese, female
Spanish-Carlos	American Spanish, male
Spanish-Diego	Argentine Spanish, male
Spanish-Francisca	Chilian Spanish, female
Spanish-Soledad	American Spanish, female
Samantha	British English, female
Daniel	British English, male
Sophie	Australian English, female
Chris	Australian English, male
Sarah	American English, female
Andrew	American English, male
Delphine	Canadian French, female
Leo	Canadian French, male
Marie	French, female
Jacques	French, male
Camila	Spanish, female
Pablo	Spanish, male
Isabella	American Spanish, female
Luis	American Spanish, male
Stefano	Italian, male
Ella	German, female
Lien	Mandarin Chinese, female
Wang	Mandarin Chinese, male
Benedita	Brazilian Portuguese, female
Francisco	Brazilian Portuguese, male
Ahmad	Arabic, male
Zara	Arabic, female

These values are case *insensitive.*

Timezone Numeric codes

Used in the User object to set the users timezone.

Albania +1 GMT	1
Algeria +1 GMT	2
American Samoa -11 GMT	195
Andorra +1 GMT	3
Angola +1 GMT	4
Anguilla -4 GMT	5
Antarctica -2 GMT	6
Antigua And Barbuda -4 GMT	7
Argentina -3 GMT	8
Argentina - Western Province -4 GMT	9
Armenia +4 GMT	10
Aruba -4 GMT	11
Ascension Island0 GMT	12
Australia - AEST +10 GMT	14
Australia - Lord Howe Island +10.5 GMT	13
Australia - Northern Territory +9.5 GMT	15
Australia - Queensland +10 GMT	16
Australia - South Australia +9.5 GMT	17
Australia - Western Australia +8 GMT	18
Austria +1 GMT	19
Azerbaijan +3 GMT	263
Bahamas -5 GMT	21
Bahrain +3 GMT	22
Bangladesh +6 GMT	23
Barbados -4 GMT	24
Belarus +2 GMT	25
Belgium +1 GMT	26
Belize -6 GMT	27
Benin +1 GMT	28
Bermuda -4 GMT	29
Bhutan +6 GMT	30
Bolivia -4 GMT	31
Botswana +2 GMT	32
Brazil - Acre -5 GMT	33
Brazil - Atlantic Islands -2 GMT	34
Brazil - East -3 GMT	35
Brazil - West -4 GMT	36
Brunei +8 GMT	38
Bulgaria +2 GMT	39
Burkina Faso0 GMT	40
Burundi +2 GMT	41
Cambodia +7 GMT	42
Cameroon +1 GMT	43
Canada - Atlantic -4 GMT	44
Canada - Central -6 GMT	45
Canada - Eastern -5 GMT	46
Canada - Mountain -7 GMT	47
Canada - Newfoundland -3.5 GMT	48
Canada - Yukon & Pacific -8 GMT	49
Cayman Islands -5 GMT	52
Chad +1 GMT	53
Chile -4 GMT	56
China +8 GMT	57
Colombia -5 GMT	58
Cook Islands -10 GMT	60
Costa Rica -6 GMT	61
Croatia +1 GMT	62
Cuba -5 GMT	63
Cyprus +2 GMT	64
Czech Republic +1 GMT	65
Denmark +1 GMT	66
Djibouti +3 GMT	67
Dominica -4 GMT	68
Ecuador -5 GMT	70
Egypt +2 GMT	71
El Salvador -6 GMT	72
Equatorial Guinea +1 GMT	73
Eritrea +3 GMT	74
Estonia +2 GMT	75
Ethiopia +3 GMT	76
Falkland Islands -4 GMT	77
Fiji +12 GMT	78
Finland +2 GMT	79
France +1 GMT	80
French Guiana -3 GMT	81
French Polynesia -10 GMT	82
Gabon +1 GMT	83

Gambia0 GMT	84
Georgia +4 GMT	85
Germany +1 GMT	86
Ghana0 GMT	87
Greece +2 GMT	88
Greenland -3 GMT	89
Greenland - Scoresbysun -1 GMT	90
Greenland - Thule -4 GMT	91
Grenada -4 GMT	92
Guadeloupe -4 GMT	93
Guam +10 GMT	94
Guatemala -6 GMT	95
Guyana -3 GMT	97
Haiti -5 GMT	98
Honduras -6 GMT	99
Hong Kong +8 GMT	100
Hungary +1 GMT	101
Iceland0 GMT	102
India +5.5 GMT	103
Indonesia - Central +8 GMT	104
Indonesia - East +9 GMT	105
Indonesia - West +7 GMT	106
Iran +3.5 GMT	107
Iraq +3 GMT	108
Ireland0 GMT	109
Israel +2 GMT	110
Italy +1 GMT	111
Ivory Coast0 GMT	112
Jamaica -5 GMT	113
Japan +9 GMT	114
Jordan +2 GMT	115
Kazakhstan +6 GMT	116
Kenya +3 GMT	117
Kiribati +12 GMT	118
Kuwait +3 GMT	121
Kyrgyzstan +5 GMT	122
Laos +7 GMT	123
Latvia +2 GMT	124
Lebanon +2 GMT	125
Lesotho +2 GMT	126
Liberia0 GMT	127

Libya +2 GMT	128
Lithuania +2 GMT	129
Luxembourg +1 GMT	130
Macedonia +1 GMT	131
Madagascar +3 GMT	132
Malawi +2 GMT	134
Malaysia +8 GMT	135
Maldives +5 GMT	136
Mali0 GMT	137
Malta +1 GMT	138
Mariana Island +10 GMT	165
Marshall Islands +12 GMT	140
Martinique -4 GMT	141
Mauritania0 GMT	142
Mauritius +4 GMT	143
Mayotte +3 GMT	144
Mexico -6 GMT	145
Mexico - Baja Calif Norte -8 GMT	146
Mexico - Nayarit Sinaloa Sonora -7 GMT	147
Moldova +2 GMT	148
Monaco +1 GMT	149
Mongolia +8 GMT	150
Morocco0 GMT	151
Mozambique +2 GMT	152
Namibia +1 GMT	153
Nepal +5.75 GMT	155
Netherlands +1 GMT	156
New Caledonia +11 GMT	158
New Zealand +12 GMT	159
Nicaragua -6 GMT	160
Niger +1 GMT	161
Niue -11 GMT	163
Norfolk Island +11.5 GMT	164
North Korea +9 GMT	119
Norway +1 GMT	167
Oman +4 GMT	168
Pakistan +5 GMT	169
Palau +9 GMT	170
Panama -5 GMT	171
Papua New Guinea +10 GMT	172
Paraguay -4 GMT	173

Peru -5 GMT	174
Philippines +8 GMT	175
Poland +1 GMT	176
Portugal +1 GMT	177
Puerto Rico -4 GMT	178
Qatar +3 GMT	179
Reunion +4 GMT	181
Romania +2 GMT	182
Russia - zone eight +9 GMT	183
Russia - zone eleven +12 GMT	184
Russia - zone five +6 GMT	185
Russia - zone four +5 GMT	186
Russia - zone nine +10 GMT	187
Russia - zone one +2 GMT	188
Russia - zone seven +8 GMT	189
Russia - zone six +7 GMT	190
Russia - zone ten +11 GMT	191
Russia - zone three +4 GMT	192
Russia - zone two +4 GMT	193
Rwanda +2 GMT	194
Saint Helena0 GMT	212
Saint Kitts and Nevis -4 GMT	213
Saint Lucia -4 GMT	214
Saint Vincent Grenadines -4 GMT	216
San Marino +1 GMT	197
Saudi Arabia +3 GMT	199
Senegal0 GMT	200
Seychelles +4 GMT	203
Sierra Leone0 GMT	204
Singapore +8 GMT	205
Slovenia +1 GMT	206
Solomon Islands +11 GMT	207
Somalia +3 GMT	208
South Africa +2 GMT	209
South Korea +9 GMT	120
Spain +1 GMT	210
Sri Lanka +5.5 GMT	211
Sudan +2 GMT	217
Swaziland +2 GMT	218
Sweden +1 GMT	219
Switzerland +1 GMT	220
Syria +2 GMT	221
Taiwan +8 GMT	222
Tanzania +3 GMT	224
Thailand +7 GMT	225
Togo0 GMT	226
Tonga +13 GMT	227
Trinidad and Tobago -4 GMT	228
Tunisia +1 GMT	229
Turkey +2 GMT	230
Turkmenistan +5 GMT	231
Tuvalu +12 GMT	233
Uganda +3 GMT	234
United Arab Emirates +4 GMT	235
United Kingdom0 GMT	236
United Kingdom - Channel Islands0 GMT	54
United Kingdom - New Hebrides +11 GMT	237
United Kingdom - Northern Ireland0 GMT	238
Uruguay -3 GMT	239
USA - Alaska -9 GMT	240
USA - Aleutian -10 GMT	241
USA - Arizona -7 GMT	242
USA - Central -6 GMT	243
USA - Eastern -5 GMT	244
USA - Hawaii -10 GMT	245
USA - Indiana East -5 GMT	246
USA - Mountain -7 GMT	247
USA - Pacific -8 GMT	248
Uzbekistan +5 GMT	249
Vanuatu +11 GMT	250
Venezuela -4 GMT	252
Vietnam +7 GMT	253
Wallis And Futuna Islands +12 GMT	256
Yemen +3 GMT	257
Zambia +2 GMT	261
Zimbabwe +2 GMT	262

Wel Callback System

Version 1.1

Intoduction

The Wel callback system returns a simple Post request to a user's chosen URL containing details about the success or failure of a chosen FAX, TTS or Voice message. For SMS, it will send a callback containing the reply details for a given 2 Way SMS reply, and optionally it can also do a callback when the SMS delivery/failure is confirmed. (See notes below)

There is one call back for each broadcast recipient, so a broadcast to 10 people should generate 10 status callback messages.

Configuration

The activation and setting of the callback URL is defined in two ways.

1. On a per user basis, and is set either via the user administration page by a Wel admin (field: "callbackURL"), or via the AddUser or ModifyUser method in the API (field "callbackURL").
2. Via the API, the callback can be set on a per broadcast basis using the "**CallbackURL"** parameter

If the callbackURL parameter is blank, no callback is attempted for that user/broadcast, otherwise a callback is sent to the given URL.
The Callback URL should be of the standard format. Eg: http://testdomain.com/testscript.php

For SMS, if a callback parameter is defined, then by default callbacks are only done for 2 way SMS replies. If you need callback on SMS status updates as well, then the additional Boolean parameter: **"CallbackOnSMSStatusUpdate"** can be set to true, either in the API SMSJob request, or in the user administration page.

Technical details

On receipt of a final message status report from the internal Telephony systems, an asynchronous Callback request is sent to the specified URL containing the following POST fields:

Fax/Voice:

BroadcastID: **1234567**
BroadcastName: **Test Broadcast**
Timestamp: **2010-09-20T23:04:56 10:00**
Cost: **10**
Reference: **TestCompany**
Recipient: **TestUser**
Destination: **001133123456789**
Status: **SENT**
Duration: **8**
Keypress: **2**

SMS Status Update:

BroadcastID: **1234567**
BroadcastName: **Test Broadcast**
Timestamp: **2010-09-20T23:04:56 10:00**
Reference: **TestCompany**
Recipient: **TestUser**
Destination: **001133123456789**
Status: **SENT**
HandsetDeliveryTimestamp: **2012-03-09 13:48:46**

SMS Reply Details:

BroadcastID: **1234567**
BroadcastName: **Test Broadcast**
Timestamp: **2010-09-20T23:04:56 10:00**
Reference: **TestCompany**
Recipient: **TestUser**
Destination: **001133123456789**
Response: **The msg returned by the recipient**

Callback Retry:

If the reply message fails to get a valid HTML response, or the words “error” or “fail” (case insensitive) are found in the response, then the callback is queued for retry.

The message is retried with intervals between retries of: 1, 5, 15 and 45 minutes.
If no valid response after this time, then the callback is dropped.

Link In SMS

Overview

Link In SMS is a feature where custom, user specific pages of html or text data can be presented through a simple SMS message. How it works is that for each SMS recipient a custom web page will be created and linked to in their SMS, allowing viewing of the page of html with a simple click. This allows a single SMS to communicate arbitrary large amounts of data. For example, a SMS broadcast could send out employee's individual upcoming monthly timetables, or their invested stock prices, or lists of overdue library books, etc. each accessed by a simple click in their SMS.

Implementation

There are two methods to use the Link In SMS System:

1. The Page data can be passed in full for each user.
2. A Link In SMS page template can be passed once for the job, and the pages generated by merging with the recipient fields.

Option 1:

With the first method of using Link In SMS through the API, the desired page data is added to the API request by adding the page data in a mergefield called "linkinsms" for each SMS recipient, and including the phrase %%linkinsms%% in your SMS message.

An example Link in SMS request would have recipient fields like:

```
Destination = '+098765432',
Reference = 'mytest',
FirstNane = 'John'
MergeField= [
       LinkInSMS = '<pageof html data for John's schedule…>'
]

Destination = '+098765431',
Reference = 'myothertest',
FirstNane = 'Jane'
MergeField= [
       LinkInSMS = '<pageof html data for Jane's schedule…>'
]
```

And the SMS message could be:

"Hi %%firstname%%. Please click on %%linkinsms%% to view your schedule for next week."

The resulting SMS received by the employee would look like:

"Hi Jane. Please click on http://lis.ms/3jKu738aU to view your schedule for next week."

And clicking this would take them to their schedule in the provided page data.

Option 2:

The second method of including a Link In SMS page is to populate the SMSJob field "LinkInSMSPage" with a template containing merge fields. This is appropriate for simple pages where there are minor differences between the recipients' pages.

For example:

```
SMSJob.LinkInSMSPage = "<p>Welcome %%firstname%%.</p><p>Please read the below
Terms and Conditions</p>
<p>… Long terms and conditions  …</p>"
```

Other details:

In both cases, if the "page content" is a just a URL, then when the recipient clicks on the link, they are redirected to the URL supplied

For example:

Page content = "http://HealthNews.com/latest_flu_info.html

Or the link could let them access custom pages just for them:

Page content = "http://myschedule.com/schedule.php?userid=123&auth=h6Fh89Uz

To reduce the amount of page data needed for each recipient, you can set Link In SMS header and footer values in your user profile. So you can have a Logo and other links automatically wrapped around any uploaded html content. This way you can just send the core content and all header, logo and styling are already done.

Document Revision History

Version 2.41 to Version 2.42

Page	Object	Description
Page 22	User	Added UserReceiveReports.

Version 2.40 to Version 2.41

Page	Object	Description
Page 55	-	Added additional TTS voices.

Version 2.39 to Version 2.40

Page	Object	Description
Page 10	SMSJob	Added LinkInSMSecQuestion
Page 10	SMSJob	Added LinkInSMSecAnswer

Version 2.38 to Version 2.39

Page	Object	Description
Page 53	-	Update Fax and Voice error codes

Version 2.37 to Version 2.38

Page	Object	Description
Page 10	SMSJob	Added LinkInSMSPage field
Page 60	-	Updated Link In SMS details to include the LinkInSPSPage template details.

Version 2.36 to Version 2.37

Page	Object	Description
Page 54	-	Added Luca (Italian) and Juliette (French) TTS voices.

Version 2.35 to Version 2.36

Page	Object	Description
Page 60	-	Added Link In SMS system.
Page 15	-	Added comment on Link In SMS system to SMS merge field.

Version 2.34 to Version 2.35

Page	Object	Description
Page 55	-	Removed Voice: “PORTUGUESE-GABRIELA”. Added Voice “GERMAN-KATRIN”.

Version 2.33 to Version 2.34

Page	Object	Description
Page 20	User	Added the fields:

		CreditLimitForAlert; CreditLimitAlertAddress; EtoBXcode; HideCostsInReport; SMSFilterOutNonAscii; TTSCharLimit; DefaultSMSReplyEmail; TTSMsgPrefix; DocRetentionHours; InboundFaxEmail; ShowRecipientsInOnlineReport; ShowRecipientsInEmailReport; IncludeCopyOfFaxInReport; HideAccounts; EmailToBroadcastIPWhiteList; EmailToBroadcastIPWhiteListAlertEmail; TimeZoneID;
Page 55		Added the list of timezone numeric codes appendix

Version 2.32 to Version 2.33

Page	Object	Description
Page 13	TTSDirectJob	Restored TTSDirectjob type.
Page 23		Restored TtsDirectStatus Method to Web Service

Version 2.31 to Version 2.32

Page	Object	Description
Page 17	StatusReport	Added ResentAs.
Page 17	StatusReport	Added ResendOriginalID.

Version 2.30 to Version 2.31

Page	Object	Description
Page 9	SMSJob	Added SMSResponseEmailAddress.

Version 2.29 to Version 2.30

Page	Object	Description
Page 20	User	Added EmailToBroadcastEmailAddress.
Page 20	User	Added BroadcastLimitWarnAddress.

Version 2.28 to Version 2.29

Page	Object	Description
		Cleanup

Version 2.27 to Version 2.28

Page	Object	Description
Page 20	User	Added EmailToBroadcastEmailAddress.
Page 20	User	Added BroadcastLimitWarnAddress.

Version 2.26 to Version 2.27

Page	Object	Description
Page 7	FaxJob	Added Dedupe.
Page 8	EmailJob	Added Dedupe.
Page 9	SMSJob	Added Dedupe.
Page 10	TTSJob	Added Dedupe.
Page 11	VoiceJob	Added Dedupe.

Version 2.25 to Version 2.26

Page	Object	Description
Page 21	ListCosts	Added notes regarding Voice/TTS billing.
Page 20	User	Added additional user option fields: • EmailToBroadcastSubjectBehaviour, • EmailToBroadcastBodyAsFaxCover, • EmailToBroadcastSignatureStart, • SMSSenderID, • DefaultTTSVoice, • TTSRepeatCount.

Version 2.24 to Version 2.25

Page	Object	Description
Page 7	Job	Added AdditionalReportEmails field

Version 2.23 to Version 2.24

Page	Object	Description
Page 9	SMSJob	Added OptoutCode String field

Version 2.22 to Version 2.23

Page	Object	Description
Page 19	User	Added IsAntiSpam Boolean field
Page 19	User	Added EmailToBroadcastAuthType String field

Version 2.21 to Version 2.22

Page	Object	Description
Page 17	ReportRecipient	Added Voicemail Boolean field

Version 2.20 to Version 2.21

Page	Object	Description
Page 19	User	Added FaxHeaderText field.

Version 2.19 to Version 2.20

Page	Object	Description

Page 18	NA	Updated description for AddUser, ModifyUser and DeleteUser.
Page 21	NA	Added “AddPrePay” function.

Version 2.18 to Version 2.19

Page	Object	Description
Page 9	SMSJob	Added “CutOffHour” field.
Page 47	NA	Added “AddUser” PHP sample code.

Version 2.17 to Version 2.18

Page	Object	Description
Page 18	NA	Add, Modify and Delete user can now be called from an Agent level account.

Version 2.16 to Version 2.17

Page	Object	Description
Page 17	STATUSREPORT	Added JobName field

Version 2.15 to Version 2.16

Page	Object	Description
Page 5	NA	Added the details of the document literal encoded version of the webservice
Page 9	SMSJob	Added UseTwoWayCustomSenderID field.
Page 47	NA	PHP Fax sample code

Version 2.14 to Version 2.15

Page	Object	Description
Page 11	NA	Removed obsolete TTS Direct Job specification.
Page 21	NA	Removed obsolete TTSDirectStatus specification.
Page 8	SMSJob	Added UseTwoWayCustomSenderID field.

Version 2.13 to Version 2.14

Page	Object	Description
Page 5	NA	Added recommendation regarding batching Fax, TTS and Voice jobs.
Page 23	NA	Added notes on Web Reference setup to C# code examples.

Version 2.12 to Version 2.13

Page	Object	Description
Page 15	STATUSREPORT	Fixed SUBMITTED field description.
Page 16	DETAILEDREPORT	Fixed SUBMITTED field description.

Version 2.11 to Version 2.12

Page	Object	Description
Page 45	n/a	Updated SMS length details.
Page 7	SMSJOB	Updated maximum SMS length.

Version 2.10 to Version 2.11

Page	Object	Description
Page 48	n/a	Expanded Callback documentation, including SMS callbacks and Callback Retry system.
Page 7	SMSJOB	Added CallbackOnSMSStatusUpdate field.

Version 2.9 to Version 2.10

Page	Object	Description
Page 47	n/a	Added new voice English-Grace.
Page 47	n/a	Removed voice Esperanto-Ludoviko.

Version 2.8 to Version 2.9

Page	Object	Description
Page 6	FAXJOB	Added CallBackURL field.
Page 6	FAXJOB	Added CutOffHour field.
Page 9	TTSJOB	Added CallbackURL field.
Page 9	VoiceJob	Added CallbackURL field..

Version 2.7 to Version 2.8

Page	Object	Description
Page 8	TTSJOB	Added KeyAckMessages array.
Page 9	VoiceJob	Added KeyAckMessages array.
Page 14	KeyAckMessage	KeyAckMessage definition.

Version 2.6 to Version 2.7

Page	Object	Description
Page 7	SMSJOB	Added SMS specific callback URL

Version 2.5 to Version 2.6

Page	Object	Description
Page 47	n/a	Added new voices to Voice list

Version 2.4 to Version 2.5

Page	Object	Description
Page 18	User	Added CallBackURL field
Page 48	n/a	Added outline of Callback system

Version 2.3 to Version 2.4

Page	Object	Description
Page 8	TTSJob	Added CutOffHour field

Version 2.2 to Version 2.3

Page	Object	Description
Page 14-15	StatusReport/ DetailedReport	Added Pages field

Version 2.1 to Version 2.2

Page	Object	Description
Page 9	VoiceJob	Added missing PreferredDestinationType

Version 2.0 to Version 2.1

Page	Object	Description
Page 42		Added PHP/NuSoap sample code

Version 1.9 to Version 2.0

Page	Object	Description
Page 6-9		Added integer parameter *RepeatInterval* for reoccurring scheduled tasks to all job types
Page 9	TTSDirectJob	Added TTSDirectjob type.
Page 19		Added TtsDirectStatus Method to Web Service

Version 1.82 to Version 1.9

Page	Object	Description
Page 5	Job	Added integer parameter *RepeatInterval* for reoccurring scheduled tasks.
Page 13	StatusReport	Added string parameter *Relatedjobs* for reoccurring scheduled tasks.
Page 18	CancelJob	Updated description

Version 1.8 to Version 1.82

Page	Object	Description
Page 41	-	Added SMS message lengths.

Version 1.7 to Version 1.8

Page	Object	Description
Page 8	TTSJob	Added String parameter *VoiceMailMessage*

Version 1.6 to Version 1.7

Page	Object	Description
Page 15		Description text fixed.
Page 18	Cancel Job	Cancel Job function added.

Version 1.5 to Version 1.6

Page	Object	Description
Page 7	TTSJob	Added Boolean parameter *SkipMsgRepeat*.

Version 1.4 to Version 1.5

Page	Object	Description
Page 4-13		Updated "Type" information to note Arrays and Object types.
Page 5	FaxJob	Added Boolean parameter *IsHighRes*.
Page 7	TTSJob	Added String parameter *Voice*.
Page 7	TTSJob	Added String parameter *HeaderText*.
Page 7-8	VoiceJob	Added parameter *TransferKeys*.
Page 7-8	VoiceJob	Added Boolean parameter *IsTwoWay*.
Page 7-8	VoiceJob	Added Boolean parameter *KeypressOnly*.
Page 41		Added list of valid "TTSJob.Voice" parameters.

Version 1.3 to Version 1.4

Page	Object	Description
Page 14-16	User, Company, ListCosts	Added Web methods *AddUser, ModifyUser* and *DeleteUser*
Page 26	n/a	Added c# examples for new methods above

Version 1.2 to Version 1.3

Page	Object	Description
Page 4	Authentication Header	Added *Usercode* parameter to allow administrators to submit jobs on behalf of their customers.

Version 1.1 to Version 1.2

Page	Object	Description
Page 6	TtsJob	Added Boolean property *SuppressHeader*
Page 6	TtsJob	Added Boolean property *KeypressOnly*
Page 12	ReportRecipient	Added integer property *Keypress*
Page 12	ReportRecipient	Added string property *Reply*
Page 34	n/a	Added revision history

Version 1 to Version 1.1

Page	Object	Description
Page 22	n/a	Added Java Axis code samples