Billing Configuration API 1.0

Integration Guide

CONTENTS

CONTENTS 2

DOCUMENT CHANGE LOG 3

VERSIONING 4

Billing Configuration Web Service 4

INTRODUCTION 5

WEB SERVICES API 7

Security 7
Message Signing 7
Client Certificates 7
WSDL Location 8
Requesting Web Services access 8
Methods and Interfaces 9
CreateBillingConfiguration 9
API Usage Examples 11

Document change log

Date	Version	Change detail
11th June 2012	1.0.0	Created
04th June 2012	1.0.1	Added Error Codes

Versioning

Billing Configuration Web Service

Version 1.0.0

https://webservices.bango.com/billingconfiguration/?WSDL

Introduction

The Bango Billing Configuration API allows you to programmatically configure a billing request ahead of the actual transaction request taking place. This API is designed to be used in conjunction with the DirectBilling API.

The API allows you to configure all of the available options for a transaction to be completed in a “just-in-time” manner, and allows you to override any particular configured options for a Bango Number to be used only for that transaction combined with a billingConfigurationId.

This means there is no requirement to configure a collection of Bango Numbers ahead of time to satisfy all variances for each transaction request.

Configuration Options

Configuration options can be used to set extra configuration options, when creating the billing request. The possible types are:

- APPLICATION_CATEGORY_ID – the category id mapping to the available category
- APPLICATION_SIZE_KB – the size of the application specified in KB
- BILLING_CONFIGURATION_TIME_OUT – the time in seconds that a billing configuration request remains valid to be used on a subsequent payment request before becoming invalidated

Not all configuration options are required, depending on the situation in which the API is used, however it is recommended that where available, all options are set and passed to the method.

Types of Payment Method

For many users, a choice of payment methods is available when collecting payment. The possible types are:

- OPERATOR – on-bill payment
- PSMS – on-bill payment via premium text message
- CARD – credit/debit card
- INTERNET – other Internet payment types (PayPal etc)

In some situations, you might want to prevent certain types of payment method from appearing, or even force just a single type to be used (for example, to force a payment via credit card).

The CreateBillingConfiguration method used for billing configuration allows for optional list of the above payment method types to be supplied, filtering the list of payment methods to only those matching the requested types.

If you want to process a payment by credit card, you should first check the UserInformation.HasCreditCard method to check the user has a card already on file. If they don’t, your application needs to collect the card details and pass to ValidateCardForUser before attempting to collect a payment.

External Transaction IDs and Idempotency

All transactional methods now support an optionally supplied external transaction ID which can be generated by your system and added alongside payment and refund requests within the Bango system. This external

transaction ID can then be used in reports generated by Bango for reconciliation purposes within your own systems.

If you wish to enforce idempotency on transaction requests, you must ensure an external transaction ID is supplied on each request.

Idempotency refers to a method whose response is always the same for given input parameters whilst those parameters remain the same. That is, if you are processing a credit card payment, subsequent calls to the associated method DirectBilling.DoPaymentWithBillingConfiguration with the same parameters would always return you the response which was given on the first call, however, subsequent calls can not effect a state change.

This can be useful to avoid scenarios such as double billing – if you were to call the associated method DirectBilling.DoPaymentWithBillingConfiguration twice, the first call may result in a successful purchase; the second call would not request a new payment to be processed, but would return the same result as the initial call, with exactly the same parameters such as response codes, transaction ID, etc.

Idempotency is not enabled by default. To enforce this on all transactional methods contact support@bango.com to request this feature..

Web Services API

The Web Services API is a secure server-to-server API (SSL only), offering ahead-of-transaction billing configuration.

Security

The Bango Web Services API is protected to ensure that only authorized clients use it. There are three primary levels of security:

The Web Services API allows your server to send requests to the Bango server. The Bango server will only accept requests from authorized IP addresses. Requests directly from an on-handset application are not allowed – the application must communicate with your server, which must then communicate with the Bango Web Services API.

- API username/password and third-party account authentication
- Client IP address validation
- Secure Sockets Layer (SSL) data transport

A failure at any one of these security levels denies access to the Bango Web Services API.

Please note: the certificate used by the Bango Web Services API is issued by Thawte. Please make sure Thawte is configured on your servers as a Trusted Root Certification Authority (CA).

Message Signing

We now optionally support message signing of the parameters within the SOAP body to add a layer of security for in-transit messages between your server and the Bango WebServices.

TODO: Add detail and worked examples

Client Certificates

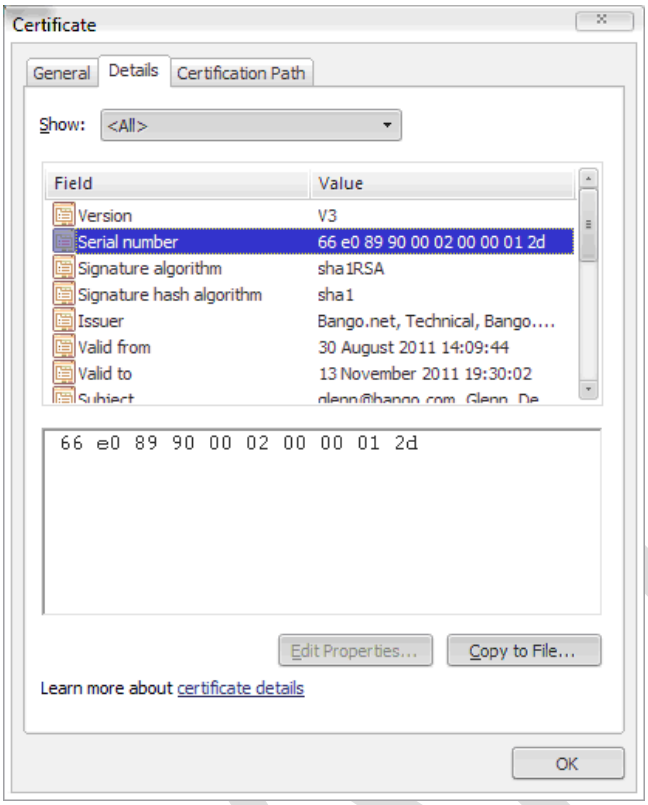
There is also an optional level of security utilizing Digital ID certificates. Certificates can be attached to each request, and Bango will validate your certificate using the Serial Number, Thumbnail and Expiry Date.

If you wish to use this additional security, please note the port number for connectivity should be changed from the default 443 to **8443**. For details on how to supply your certificate details, please see below.

To begin, you will need to purchase a suitable client certificate – Bango recommend a Verisign Digital ID certificate.

Once you have purchased and configured your certificate, you will need to contact Bango customer services and supply them the **Serial Number**, **Thumbnail** and **Expiry Date** for the certificate you wish to use.

An example of how to find these details using Microsoft Windows is shown overleaf:



These details will be added to your WebServices credentials and configured to be required on every request.

Once this is done, you should ensure your integration end point uses the new 8443 port when connecting, and that the certificate is attached to each request. Once enabled, failure to provide a valid certificate will deny access to the Bango Web Services API.

WSDL Location

The Billing Configuration WSDL can be located at the following URL:

https://webservices.bango.com/billingconfiguration/?wsdl

For those wishing to use Digital ID certificates, the WSDL can be located at:

https://webservices.bango.com:8443/billingconfiguration/?wsdl

Requesting Web Services access

Before you can use the Web Services API, you need to first request access credentials. Login to the Management tools on bango.com and go to the Setup and Config tab, then select 'Web Service API's' on the left.

Access to the API is dependent upon package level. If you are unsure whether the API is available to you please contact Customer Services.

Method: **CreateBillingConfiguration**

Description: Creates and configures billing ahead of a transaction request for the specified Bango Number. The Bango Number must be pre-configured via the Bango.com Management Tools with the appropriate access model and default prices. The values supplied can be used to override the pre-configured values in the database.

On successful configuration of a number, a billingConfigurationId will be returned which should be used in a subsequent request to the DirectBilling.DoPaymentWithBillingConfiguration method call.

Inputs:

username	Your username
password	Your password
bango	Bango number
typeFilter	String array of allowed payment method types
priceList	Array of price entities, each comprising the following: amount, int, no nulls currency, char(3), no nulls
externalTransactionId	An optional transaction ID generated by your system which can be logged against a billing configuration request within the Bango system.
pageTitle	The name of the content being purchased
configurationOptions	Array of BillingRequestConfigurationOption entities, each comprising the following: configurationOptionName, string, no nulls configurationOptionValue, string, no nulls Currently supported Configuration Options are: • **APPLICATION_CATEGORY_ID** • **APPLICATION_SIZE_KB** • **BILLING_CONFIGURATION_TIME_OUT** Unsupported values which are passed on a request will result in an INVALID_CONFIGURATION_NAME / _VALUE response.

Outputs:

responseCode	Result of the payment request (see below)
responseMessage	Any additional information to clarify the payment status code
billingConfigurationId	Billing Configuration ID returned when billing was successfully configured – this ID should be used when calling the relevant methods in the DirectBilling API.

Response Codes:

OK	Success
ACCESS_DENIED	Invalid username/password
ACCESS_DENIED	Invalid credentials. You are not authorized to call this method
ACCESS_DENIED	Invalid client IP address
ACCESS_DENIED	Invalid certificate
ACCESS_DENIED	Invalid access to specified Bango number
SERVICE_UNAVAILABLE	The service is currently unavailable and your

	request could not be processed
INTERNAL_ERROR	A problem on the server meant that the request could not be processed
INVALID_BANGO	Invalid Bango number
INVALID_ACCESSMODEL	Bango number does not have the correct access model
INVALID_CONFIGURATION_NAME	A supplied configurationOptionName is invalid
INVALID_CONFIGURATION_VALUE	A supplied configurationOptionValue is invalid
INVALID_STRING_LENGTH	A supplied configurationOptionValue length is invalid
REQUIRED_CONFIGURATION_MISSING	A required configurationOptionName is not supplied

Notes

API Usage Examples

The following examples are pseudo-code for illustration purposes only. Refer to your appropriate technology documentation for details of how to use web services in your chosen development environment.

Creating a Billing Configuration request:

```
String myBango = "12345"
String myPageTitle = "TEST"
String myExtId = "MYEXTID"

CreateBillingConfigurationRequest req
req.username = myUsername
req.password = myPassword

req.bango = myBango
req.pageTitle = myPageTitle
req.externalTransactionId = myExtId

req.priceList = new PriceList[]  { new Price(1.00, "GBP"), new Price(1.75, "USD"), new Price(2.50,
"EUR") }
req.configurationOptions = new configurationOptions[]
{
new BillingConfigurationOption ("APPLICATION_CATEGORY_ID", "1"),
new BillingConfigurationOption ("APPLICATION_SIZE_KB", "575")
}

CreateBillingConfigurationResponse res =
BILLINGCONFIGURATIONAPI.CreateBillingConfiguration(req)

If res.responseCode = "OK" Then
        // Successful billing configuration – can now use within Direct Billing API
        billingConfigID = res.billingConfigurationId
Else
        // Deal with error
        responseCode = res.responseCode
        responseMessage = res.responseMessage
End If
```