Remitter Web Services API Specification

RemitONE Money Transfer Management Solution

Version 3.5
November 2020

Table of Contents

1. Changelog

Version	Date	Notes
0.1	26/05/2016	Initial version
0.2	20/06/2016	Added getCarouselImages() + getMobileNetworkCreditTypes() + getTransactionPaymentInstructions() + Improved validate()
0.2.1	29/06/2016	Refactored the 2FA response element for login, loginPin and changePassword. It now includes the type and indications about code re-submission.
0.3	29/06/2016	Added getSeed and refactored login and loginPin
0.4	04/07/2016	Renamed calculateTransferBetweenWallets(), requestTransferBetweenWallets() and confirmTransferBetweenWallets() to xMoveFundsBetweenWallets() to avoid confusion with upcoming Wallet Transfers. Added requestRegistrationConfirmationCode() and requestTransactionConfirmationCode() methods.
0.5	15/07/2016 19/07/2016 22/07/2016 25/07/2016 26/07/2016 02/08/2016	Added parameters for getCharges(). Added destination_country_id parameter to getCollectionPointStates(), getCollectionPointCities() and getCollectionPoints() Refactored parameters for createTransaction() Added username parameter to nearly all methods + Added more optional parameter for listTransactions() Refactored parameters and response for getRates() Moved getSourceCountries(), getRemitterUISettings(), register(), confirmRegistration() and requestRegistrationConfirmationCode() to the remitterUser group Added more data to the responses of getProfile() and updateProfile() Made delivery_branch_state optional for getDeliverybranchCities() + Added searchUtilityBillCompanies() method + Added more fields in getUtilityBillCompanies() response + Added more information in getbeneficiary() and listBeneficiaries()
0.6	04/08/2016	changePIN() now has encrypted_data field with seed, new_pin, confirm_pin, current_password, and current_pin fields.
0.7	04/08/2016	Moved getXUISettings methods to the UI group + Renamed UI group to UISettings
0.8	10/08/2016	Added password and verify_password as encrypted data + toc for register() + Updated structure of error messages: The main error message is now a direct child of response + include field name causing the validation error + Added getTransactionUISettings() with SMS options
0.8.1	22/08/2016	Corrected loginPin() parameters to use the encrypted_data array.
0.8.2	26/08/2016	Refactored changePassword() parameters. Removed confirm_pin parameter from changePin()
0.8.3	02/09/2016	Refactored register() parameters.
0.8.4	06/09/2016	Added nationalities and “hear about us” options in getRemitterUISettings() + Added building_no and taxpayer_reg fields to register()
0.8.5	19/09/2016	Added remitter_username field to updateProfile() method
0.8.6	22/09/2016	Attempt to login the user on successful confirmRegistration() and non-authenticated changePassword() Renamed “email_address” parameter by “username” in forgotPassword()

0.9	03/10/2016	Refactored createBeneficiary() and updateBeneficiary() method parameters to reflect all information that can be used in order to create transactions. Added “account_item_number” parameter on createTransaction() method to specify which Beneficiary account to use for an Account Transfer. Added avatar field in getBeneficiary(), listBeneficiaries() and updateBeneficiary() responses. The parameter is not available yet. This is read-only at the moment.
0.9.1	06/10/2016	Renamed “accountX” children of getBeneficiary() and updateBeneficiary() to “account” so that it could be iterated over more easily.
0.9.2	17/10/2016	Reflected optional fields on updateBeneficiary(). Added avatar parameter on createBeneficiary() and updateBeneficiary(). Removed “verify_password” parameter from register().
0.9.3	20/10/2016	Added auth/logout() method. Corrected id1_x parameter names in updateProfile() method. Added country_id and country_iso_code data in the response of getProfile() and updateProfile().
0.9.4	20/10/2016	Added getLoadWalletDetails method. Unified requestLoadWallet() and confirmLoadWallet() methods into loadWallet(). “collection_point_state” is now optional for getCollectionPointCities(). Unified requestMoveFundsBetweenWallets() and confirmMoveFundsBetweenWallets() methods into moveFundsBetweenWallets().
0.10	25/10/2016	Refactored id document related fields in getProfile() and updateProfile() responses. Added several parameters to register() and updateProfile().
0.10.1	25/10/2016	Added “import_id” parameter to createBeneficiary(), updateBeneficiary(), getBeneficiary() and listBeneficiaries() methods. Added “payment_token” parameter to createTransaction(), paymentCleared(), getTransaction() and listTransactions() methods.
0.10.2	01/11/2016	Added “my_wallet_ref” to the response of loadWallet().
0.10.3	08/11/2016	Added mapping example in updateBeneficiary() in preparation for an Account transfer.
0.11	29/11/2016	Refactored responses of createTransaction(), confirmTransaction() and getTransaction() to include more details about the delivery entities. Refactored listTransaction() response to remove unnecessary fields.
0.11.1	02/12/2016	Added “type” in the response of getTransactionPaymentInstructions().
0.11.2	07/12/2016	Added “tax” in the response of getTransaction(), createTransaction() and confirmTransaction()
0.12.0	12/12/2016	Changed response of getBeneficiaryUISettings(), getRemitterUISettings(), getTransactionUISettings() to return “elements”. Updated transfer_type description in getTransactionUISettings(). Added field account_item_number in getTransactionUISettings() method.
0.12.1	13/12/2016	Added bank_routing_transit_number parameters to createBeneficiary() and updateBeneficiary() methods. Added bank_routing_transit_number fields in the responses of getBeneficiary() and updateBeneficiary() methods.

0.12.2	13/12/2016	Renamed fields “account_numberX” to “bank_account_numberX”, fields “account_typeX” to “bank_account_typeX” and field “bank_iban_code” to “bank_iban” in getBeneficiary(), createBeneficiary(), updateBeneficiary() and getTransactionUISettings() methods. Renamed field “benef_ac” to “benef_bank_account_number” in getTransaction(), createTransaction() and confirmTransaction() methods. Added field “benef_bank_account_type” in getTransaction(), createTransaction() and confirmTransaction() methods.
0.12.3	14/12/2016	Added “additional_attributes” examples in getRemitterUISettings(), getBeneficiaryUISettings() and getTransactionUISettings() methods. Added “avatar” parameter to register(), getProfile() and updateProfile() methods.
0.12.4	21/12/2016	Added “utility_bill_description” and “utility_bill_invoice” parameters to createTransaction() and the response of getTransactionUISettings().
0.12.5	08/02/2017	Added <password_validation> element to response of getRemitterUISettings() method. Removed “country_id” parameter from updateBeneficiary() method.
0.12.6	15/05/2017	Added “remitt_pay”, “commission_before_promotion”, “promotion_names” and “promotion_ids” elements to the response of getCharges() method.
0.12.7	22/05/2017	Added “payment_method_name” element to response of getTransactionPaymentInstructions() method. Added “remitt_benef_relation” element to response of getBeneficiary() method. Added “remitt_benef_relation” field to request of createBeneficiary() and updateBeneficiary() methods.
0.12.8	30/05/2017	Added “purpose” and “source_of_income” elements to the response of createTransaction(), confirmTransaction() and getTransaction() methods.
0.12.9	01/06/2017	Added “commission_before_promotion”, “promotion_names”, “promotion_ids”, “service_level” elements to the response of createTransaction(), confirmTransaction() and getTransaction() methods.
0.12.10	02/06/2017	Added “default_source_of_income” and “default_purpose” elements to the response of getTransactionUISettings(). Added “account_types” element to the response of getTransactionUISettings().
0.13.0	21/06/2017	Added getCountryPrefixes() method within UISettings.
0.13.1	23/06/2017	Added “source_country_iso_code” and “destination_country_iso_code” elements to the response of getCharges().
0.13.2	07/07/2017	Added “id_types” element to response of getRemitterUISettings() and getBeneficiaryUISettings().
0.13.3	08/09/2017	Added “promotion_code” field to request of getCharges(). Renamed “discount_code” field to “promotion_code” on createTransaction().
0.13.4	11/09/2017	Added getChangePasswordSettings() method to UISettings.
0.13.5	09/10/2017	Added “registration_type” field to getRemitterUISettings() and register() methods.
1.0	20/12/2017	Made it clear that the base url needs to be the client url.
1.1	05/02/2018	Add more information about the encryption public key in Appendix I.

1.2	01/03/2018	Added “current_password” field to changePassword() method.
1.3	20/03/2018	Fixed all appendices references
1.4	23/03/2018	Added payment_gateway_acknowledged to confirmTransaction() and getTransaction() methods.
1.5	03/05/2018	Added “receive_marketing” field to register() method.
1.6	08/08/2018	Added searchWalletBeneficiary() method
3.4	06/10/2020	Added referral_code into register() method

2. Introduction

The RemitONE Remitter WebServices Infrastructure uses the REST protocol. Our services use the POST method for invoking the service, and hence all data must be provided as POST variables. The response is provided as structured XML.

This document outlines the input and output parameters for each of the RemitONE Remitter WebServices. Yet, it is not the production version as the information are subject to further changes that may occur during the development. While most of the content should remain the same, this is provided at the moment for informational purpose only.

Connection Parameters

The BASE_URL is the URL at which the Remitter WebServices are hosted. This BASE_URL is then followed by the required GROUP name, and then the required METHOD name.

For example,
BASE_URL = https://<insert your system URL here>/remitterws/

GROUP = transaction
METHOD = createTransaction

This would give a complete URL of :
https://<insert your system URL here>/remitterws/transaction/createTransaction

The client application would need to make a POST to this URL to invoke this particular WebService method.
In the specification below, a * indicates a required input parameter.

Most of the invocations must contain the following parameter (as POST variable) :

Name	Type	Notes
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**

These will be provided to the remitter after logging in.

All other parameters will depend on the particular WebService being invoked.

Response Format

A Successful Response will always contain the following XML structure :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        [Specific response for the particular Web Service]
    </result>
</response>
```

An Error Response is as follows :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>FAIL</status>
    <message>[Error message]</message>
</response>
```

An Error Response with Validation errors is as follows :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>FAIL</status>
    <message>VALIDATION FAILED</message>
    <result>
        <errors>
            <error>
                  <field>[Field triggering the error(s)]</field>
                  <message>[Validation error message related to field]</message>
                  <message>[Validation error message related to field]</message>
            </error>
            <error>
                  <field>[Field triggering the error(s)]</field>
                  <message>[Validation error message related to field]</message>
            </error>
        </errors>
    </result>
</response>
```

3.Web Service Method Details

Authentication

getSeed

Group : **auth**
Method : **getSeed**
This method is used to get a unique Seed used by login and loginPin methods.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <seed>XXXXXXXXXXXXXXXXXXXXXXXXXXXX</seed>
    </result>
</response>
```

---

login

Group : **auth**
Method : **login**
This method is used to verify the Remitter_User login credentials.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
encrypted_data *	Text	Encrypted associative json array using the seed (generated by a call to **getSeed**) and the public OpenSSL key. The associative array should be: {“seed”: “XXXXXXXXX”, “password”: “YYYYYY”}

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <session_token>XXXXXXXXXXXXXXXXXXXXXXXXXXXX</session_token>
        <app_pin>YYYYYY</app_pin>
        <two_factor_authentication>
            <required>true</required>
            <type>SMS</type>
            <can_resend_code>true</can_resend_code>
        </two_factor_authentication>
    </result>
</response>
```

Notes: Please refer to appendix I for details about the encryption.

Notes 2: The app pin might be provided if this is the first time that the user is logging in via the remitter Web Services; so that it can be given to the user. Otherwise this information will not be included in the response.

The two factor authentication flag indicates if a 2FA code has been sent to the remitter. If so, this information must be provided using **confirmTwoFactorAuthentication**. Otherwise, the user will not be able to perform any further action.

The two factor authentication type can be: SMS, Entrust or GoogleAuthenticator. This list may grow in the future.

---

loginPin

Group : **auth**
Method : **loginPin**
This method verifies the Remitter_User login credentials using the app_pin.

Input fields :

Name	Type	Notes
username *	Text	The username of the remitter
encrypted_data *	Text	Encrypted associative json array using the seed (generated by a call to **getSeed**) and the public OpenSSL key. The associative array should be:

		{“seed”: “XXXXXXXXX”, “pin”: “YYYYYY”}

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <session_token>XXXXXXXXXXXXXXXXXXXXXXXXXXXX</session_token>
        <two_factor_authentication>
            <required>true</required>
            <type>SMS</type>
            <can_resend_code>true</can_resend_code>
        </two_factor_authentication>
    </result>
</response>
```

Notes: Please refer to appendix I for details about the encryption.

Notes 2: The two factor authentication flag indicates if a 2FA code has been sent to the remitter. If so, this information must be provided using **confirmTwoFactorAuthentication**. Otherwise, the user will not be able to perform any further action.

The two factor authentication type can be: SMS, Entrust or GoogleAuthenticator. This list may grow in the future.

---

confirmTwoFactorAuthentication

Group : **auth**
Method : **confirmTwoFactorAuthentication**
This method validates the provided two factor authentication code that the user has normally received by SMS (if enabled).

Input fields:

Name	**Type**	**Notes**
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**
code *	Text	The Two Factor Authentication code that has been sent by SMS

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

---

requestTwoFactorAuthentication

Group : **auth**
Method : **requestTwoFactorAuthentication**
This method re-sends another two factor authentication code by SMS.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

---

logout

Group : **auth**
Method : **logout**
This method logs out the user and terminates its session.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging

		in via **login** or **loginPin**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

---

forgotPassword

Group : **auth**
Method : **forgotPassword**
This method sends a password reset link by email.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter.
dob *	Date	YYYY-MM-DD

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

---

changePassword

Group : **auth**
Method : **changePassword**
This method updates the remitter password.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter.
encrypted_data *	Text	Encrypted associative json array using the

		seed (generated by a call to **getSeed**) and the public OpenSSL key. The associative array should be: {“seed”: “XXXXXXXXX”, “new_password”: “YYYYYY”, “current_password”: “YYYYYY”,} The current_password of the remitter is not required but is validated if provided.
session_token	Text	The Session Token provided when logging in via **login** or **loginPin**. This is required when the user is already logged in.
forgot_password_token	Text	The forgot_password password token can be generated by a call to **forgotPassword**. This is required when the user is not logged in.

Example Output XML If session_token was provided :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

Example Output XML If forgot_password_token was provided :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <session_token>XXXXXXXXXXXXXXXXXXXXXXXXXXX</session_token>
        <app_pin>YYYYYY</app_pin>
        <two_factor_authentication>
            <required>true</required>
            <type>SMS</type>
            <can_resend_code>true</can_resend_code>
        </two_factor_authentication>
    </result>
</response>
```

Notes: Please refer to appendix I for details about the encryption.

Notes2: The app pin might be provided if this is the first time that the user is logging in via the remitter Web Services; so that it can be given to the user. Otherwise this information will not be included in the response.

The two factor authentication flag indicates if a 2FA code has been sent to the remitter. If so, this information must be provided using **confirmTwoFactorAuthentication**. Otherwise, the user will not be able to perform any further action.

The two factor authentication type can be: SMS, Entrust or GoogleAuthenticator. This list may grow in the future.

---

changePin

Group : **auth**
Method : **changePin**
This method changes the pin of the user.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
encrypted_data *	Text	Encrypted associative json array using the seed (generated by a call to **getSeed**) and the public OpenSSL key. The associative array should be: {“seed”: “XXXXXXXXX”, “new_pin”: “YYYYYY”, “current_password”: “ZZZZZZ”} or {“seed”: “XXXXXXXXX”, “new_pin”: “YYYYYY”, “current_pin”: “ZZZZZZ”} The current_password of the remitter is required if there is no current_pin provided. The current_pin of the remitter is required if there is no current_password provided.
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

Notes: Please refer to appendix I for details about the encryption.

Beneficiary

getDestinationCountries

Group : **beneficiary**
Method : **getDestinationCountries**
This method is used to retrieve a list of destination countries available for the remitter's country.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <countries>
            <country>
                <id>001</id>
                <name>United Kingdom</name>
                <iso_code>UK</iso_code>
            </country>
            <country>
                <id>002</id>
                <name>United States</name>
                <iso_code>US</iso_code>
            </country>
        </countries>
    </result>
</response>
```

---

createBeneficiary

Group : **beneficiary**
Method : **createBeneficiary**
This method creates and links a new beneficiary to the remitter.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
fname *	Text	
mname	Text	
lname *	Text	
nickname	Text	
gender	Text	MALE / FEMALE
address1 *	Text	
address2	Text	
address3	Text	
city *	Text	
state	Text	
postcode	Text	
country_id *	Text	Country id, as per output of **getDestinationCountries**
nationality	Text	
dob	Date	YYYY-MM-DD
fathers_name	Text	
mothers_name	Text	
national_id_number	Text	
email	Text	
telephone	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.
mobile	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.
sms_payout_mobile	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.
remitt_benef_relation	Text	Beneficiary’s relation to sender.
avatar	Base64 encoded	png, gif, jpg and jpeg

	file	
import_id	Text	Optional third party id of the beneficiary. This is used for migration of data into RemitONE.
id_type	Code	
id_details	Text	
id_start	Date	
id_expiry	Date	
id_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id_issued_by	Text	
id_issue_place	Text	
id_issue_country	Text	
card_number	Text	Used for Card Transfers The actual number if known, or REQUESTED if the bank needs to create a new card
bank_account_number1	Text	Used for Account Transfers
bank_account_type1	Text	Used for Account Transfers
bank_account_name1	Text	Used for Account Transfers
bank1	Text	Used for Account Transfers
bank_iban1	Text	Used for Account Transfers
bank_swift_code1	Text	Used for Account Transfers
bank_ifsc_code1	Text	Used for Account Transfers
bank_bsb_code1	Text	Used for Account Transfers
bank_routing_transit_number1	Text	Used for Account Transfers
bank_branch1	Text	Used for Account Transfers
bank_branch_id1	Number	Used for Account Transfers
bank_branch_code1	Text	Used for Account Transfers
bank_branch_city1	Text	Used for Account Transfers
bank_branch_state1	Text	Used for Account Transfers
bank_branch_postcode1	Text	Used for Account Transfers
bank_branch_telephone1	Phone	Used for Account Transfers
bank_branch_manager1	Text	Used for Account Transfers
additional_bank1	Text	Used for Account Transfers
additional_bank_branch1	Text	Used for Account Transfers

intermediary_bank_account_number1	Text	Used for Account Transfers
intermediary_bank1	Text	Used for Account Transfers
intermediary_bank_address1	Text	Used for Account Transfers
intermediary_bank_swift_code1	Text	Used for Account Transfers
bank_account_number2	Text	Used for Account Transfers
bank_account_type2	Text	Used for Account Transfers
bank_account_name2	Text	Used for Account Transfers
bank2	Text	Used for Account Transfers
bank_iban2	Text	Used for Account Transfers
bank_swift_code2	Text	Used for Account Transfers
bank_ifsc_code2	Text	Used for Account Transfers
bank_bsb_code2	Text	Used for Account Transfers
bank_routing_transit_number2	Text	Used for Account Transfers
bank_branch2	Text	Used for Account Transfers
bank_branch_id2	Number	Used for Account Transfers
bank_branch_code2	Text	Used for Account Transfers
bank_branch_city2	Text	Used for Account Transfers
bank_branch_state2	Text	Used for Account Transfers
bank_branch_postcode2	Text	Used for Account Transfers
bank_branch_telephone2	Phone	Used for Account Transfers
bank_branch_manager2	Text	Used for Account Transfers
intermediary_bank_account_number2	Text	Used for Account Transfers
intermediary_bank2	Text	Used for Account Transfers
intermediary_bank_address2	Text	Used for Account Transfers
intermediary_bank_swift_code2	Text	Used for Account Transfers
bank_account_number3	Text	Used for Account Transfers
bank_account_type3	Text	Used for Account Transfers
bank_account_name3	Text	Used for Account Transfers
bank3	Text	Used for Account Transfers
bank_iban3	Text	Used for Account Transfers
bank_swift_code3	Text	Used for Account Transfers
bank_ifsc_code3	Text	Used for Account Transfers
bank_bsb_code3	Text	Used for Account Transfers
bank_routing_transit_number3	Text	Used for Account Transfers
bank_branch3	Text	Used for Account Transfers

bank_branch_id3	Number	Used for Account Transfers
bank_branch_code3	Text	Used for Account Transfers
bank_branch_city3	Text	Used for Account Transfers
bank_branch_state3	Text	Used for Account Transfers
bank_branch_postcode3	Text	Used for Account Transfers
bank_branch_telephone3	Phone	Used for Account Transfers
bank_branch_manager3	Text	Used for Account Transfers
intermediary_bank_account_number3	Text	Used for Account Transfers
intermediary_bank3	Text	Used for Account Transfers
intermediary_bank_address3	Text	Used for Account Transfers
intermediary_bank_swift_code3	Text	Used for Account Transfers
bank_account_number4	Text	Used for Account Transfers
bank_account_type4	Text	Used for Account Transfers
bank_account_name4	Text	Used for Account Transfers
bank4	Text	Used for Account Transfers
bank_iban4	Text	Used for Account Transfers
bank_swift_code4	Text	Used for Account Transfers
bank_ifsc_code4	Text	Used for Account Transfers
bank_bsb_code4	Text	Used for Account Transfers
bank_routing_transit_number4	Text	Used for Account Transfers
bank_branch4	Text	Used for Account Transfers
bank_branch_id4	Number	Used for Account Transfers
bank_branch_code4	Text	Used for Account Transfers
bank_branch_city4	Text	Used for Account Transfers
bank_branch_state4	Text	Used for Account Transfers
bank_branch_postcode4	Text	Used for Account Transfers
bank_branch_telephone4	Phone	Used for Account Transfers
bank_branch_manager4	Text	Used for Account Transfers
intermediary_bank_account_number4	Text	Used for Account Transfers
intermediary_bank4	Text	Used for Account Transfers
intermediary_bank_address4	Text	Used for Account Transfers
intermediary_bank_swift_code4	Text	Used for Account Transfers
collection_point_id	Number	Used for Cash Collection
collection_point_name	Text	Used for Cash Collection
collection_point_code	Text	Used for Cash Collection

collection_point_proc_bank	Text	Used for Cash Collection
collection_point_address	Text	Used for Cash Collection
collection_point_city	Text	Used for Cash Collection
collection_point_state	Text	Used for Cash Collection
collection_point_tel	Phone	Used for Cash Collection
utility_bill_company	Text	Used for Utility Bill Transfers
utility_bill_company_code	Text	Used for Utility Bill Transfers
utility_bill_address1	Text	Used for Utility Bill Transfers
utility_bill_address2	Text	Used for Utility Bill Transfers
utility_bill_address3	Text	Used for Utility Bill Transfers
utility_bill_city	Text	Used for Utility Bill Transfers
utility_bill_state	Text	Used for Utility Bill Transfers
utility_bill_postcode	Text	Used for Utility Bill Transfers
utility_bill_bank	Text	Used for Utility Bill Transfers
utility_bill_account_no	Text	Used for Utility Bill Transfers
utility_bill_bank_bic	Text	Used for Utility Bill Transfers
utility_bill_bank_swift_iban	Text	Used for Utility Bill Transfers
mobile_transfer_number	Phone	Used for Mobile Transfers
mobile_transfer_network	Text	Used for Mobile Transfers
mobile_transfer_network_id	Number	Used for Mobile Transfers
mobile_transfer_network_credit_type_id	Number	Used for Mobile Transfers
mobile_transfer_network_credit_type	Text	Used for Mobile Transfers
home_delivery_notes	Text	Used for Home Delivery

Notes: If the Bank needs to create a new Card for the beneficiary, then specify the Card Number as 'REQUESTED'. This data will then be fed to the Bank dealing with that country for creation of the ATM Card.

Other fields may be required depending on the actual configuration of the system. This list is therefore non-exhaustive.

The data saved against the beneficiary will be used when creating a transaction. As a result, make sure to save the relevant data against the beneficiary for the type of transaction that you will create.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
```

```
    <new_beneficiary_id>42</new_beneficiary_id>
</response>
```

---

updateBeneficiary

Group : **beneficiary**
Method : **updateBeneficiary**
This method updates a linked beneficiary data.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
beneficiary_id *	Number	
fname	Text	
mname	Text	
lname	Text	
nickname	Text	
gender	Text	MALE / FEMALE
address1	Text	
address2	Text	
address3	Text	
city	Text	
state	Text	
postcode	Text	
nationality	Text	
dob	Date	YYYY-MM-DD
fathers_name	Text	
mothers_name	Text	
national_id_number	Text	
email	Text	
telephone	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.

mobile	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.
sms_payout_mobile	Phone	Phone number starting with either “+” or “00” followed by the country iso code and then the number without the initial “0”.
remitt_benef_relation	Text	Beneficiary’s relation to sender.
avatar	Base64 encoded file	png, gif, jpg and jpeg
import_id	Text	Optional third party id of the beneficiary. This is used for migration of data into RemitONE.
id_type	Code	
id_details	Text	
id_start	Date	
id_expiry	Date	
id_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id_issued_by	Text	
id_issue_place	Text	
id_issue_country	Text	
card_number	Text	Used for Card Transfers The actual number if known, or REQUESTED if the bank needs to create a new card
bank_account_number1	Text	Used for Account Transfers
bank_account_type1	Text	Used for Account Transfers
bank_account_name1	Text	Used for Account Transfers
bank1	Text	Used for Account Transfers. The delivery bank name from **getDeliveryBanks()**
bank_iban1	Text	Used for Account Transfers
bank_swift_code1	Text	Used for Account Transfers
bank_ifsc_code1	Text	Used for Account Transfers
bank_bsb_code1	Text	Used for Account Transfers
bank_routing_transit_number1	Text	Used for Account Transfers

bank_branch1	Text	Used for Account Transfers. The delivery bank branch name from **getDeliveryBranches()**
bank_branch_id1	Number	Used for Account Transfers. The delivery bank branch id from **getDeliveryBranches()**
bank_branch_code1	Text	Used for Account Transfers. The delivery bank branch branch_code from **getDeliveryBranches()**
bank_branch_city1	Text	Used for Account Transfers. The delivery bank branch city from **getDeliveryBranches()**
bank_branch_state1	Text	Used for Account Transfers. The delivery bank branch state from **getDeliveryBranches()**
bank_branch_postcode1	Text	Used for Account Transfers. The delivery bank branch postcode from **getDeliveryBranches()**
bank_branch_telephone1	Phone	Used for Account Transfers. The delivery bank branch telephone from **getDeliveryBranches()**
bank_branch_manager1	Text	Used for Account Transfers. The delivery bank branch manager from **getDeliveryBranches()**
additional_bank1	Text	Used for Account Transfers
additional_bank_branch1	Text	Used for Account Transfers
intermediary_bank_account_number1	Text	Used for Account Transfers
intermediary_bank1	Text	Used for Account Transfers
intermediary_bank_address1	Text	Used for Account Transfers
intermediary_bank_swift_code1	Text	Used for Account Transfers
bank_account_number2	Text	Used for Account Transfers
bank_account_type2	Text	Used for Account Transfers
bank_account_name2	Text	Used for Account Transfers
bank2	Text	Used for Account Transfers. The delivery bank name from **getDeliveryBanks()**
bank_iban2	Text	Used for Account Transfers
bank_swift_code2	Text	Used for Account Transfers
bank_ifsc_code2	Text	Used for Account Transfers
bank_bsb_code2	Text	Used for Account Transfers
bank_routing_transit_number2	Text	Used for Account Transfers

bank_branch2	Text	Used for Account Transfers. The delivery bank branch name from **getDeliveryBranches()**
bank_branch_id2	Number	Used for Account Transfers. The delivery bank branch id from **getDeliveryBranches()**
bank_branch_code2	Text	Used for Account Transfers. The delivery bank branch branch_code from **getDeliveryBranches()**
bank_branch_city2	Text	Used for Account Transfers. The delivery bank branch city from **getDeliveryBranches()**
bank_branch_state2	Text	Used for Account Transfers. The delivery bank branch state from **getDeliveryBranches()**
bank_branch_postcode2	Text	Used for Account Transfers. The delivery bank branch postcode from **getDeliveryBranches()**
bank_branch_telephone2	Phone	Used for Account Transfers. The delivery bank branch telephone from **getDeliveryBranches()**
bank_branch_manager2	Text	Used for Account Transfers. The delivery bank branch manager from **getDeliveryBranches()**
intermediary_bank_account_number2	Text	Used for Account Transfers
intermediary_bank2	Text	Used for Account Transfers
intermediary_bank_address2	Text	Used for Account Transfers
intermediary_bank_swift_code2	Text	Used for Account Transfers
bank_account_number3	Text	Used for Account Transfers
bank_account_type3	Text	Used for Account Transfers
bank_account_name3	Text	Used for Account Transfers
bank3	Text	Used for Account Transfers. The delivery bank name from **getDeliveryBanks()**
bank_iban3	Text	Used for Account Transfers
bank_swift_code3	Text	Used for Account Transfers
bank_ifsc_code3	Text	Used for Account Transfers
bank_bsb_code3	Text	Used for Account Transfers
bank_routing_transit_number3	Text	Used for Account Transfers
bank_branch3	Text	Used for Account Transfers. The delivery bank branch name from **getDeliveryBranches()**

bank_branch_id3	Number	Used for Account Transfers. The delivery bank branch id from **getDeliveryBranches()**
bank_branch_code3	Text	Used for Account Transfers. The delivery bank branch branch_code from **getDeliveryBranches()**
bank_branch_city3	Text	Used for Account Transfers. The delivery bank branch city from **getDeliveryBranches()**
bank_branch_state3	Text	Used for Account Transfers. The delivery bank branch state from **getDeliveryBranches()**
bank_branch_postcode3	Text	Used for Account Transfers. The delivery bank branch postcode from **getDeliveryBranches()**
bank_branch_telephone3	Phone	Used for Account Transfers. The delivery bank branch telephone from **getDeliveryBranches()**
bank_branch_manager3	Text	Used for Account Transfers. The delivery bank branch manager from **getDeliveryBranches()**
intermediary_bank_account_number3	Text	Used for Account Transfers
intermediary_bank3	Text	Used for Account Transfers
intermediary_bank_address3	Text	Used for Account Transfers
intermediary_bank_swift_code3	Text	Used for Account Transfers
bank_account_number4	Text	Used for Account Transfers
bank_account_type4	Text	Used for Account Transfers
bank_account_name4	Text	Used for Account Transfers
bank4	Text	Used for Account Transfers. The delivery bank name from **getDeliveryBanks()**
bank_iban4	Text	Used for Account Transfers
bank_swift_code4	Text	Used for Account Transfers
bank_ifsc_code4	Text	Used for Account Transfers
bank_bsb_code4	Text	Used for Account Transfers
bank_routing_transit_number4	Text	Used for Account Transfers
bank_branch4	Text	Used for Account Transfers. The delivery bank branch name from **getDeliveryBranches()**
bank_branch_id4	Number	Used for Account Transfers. The delivery bank branch id from **getDeliveryBranches()**

bank_branch_code4	Text	Used for Account Transfers. The delivery bank branch branch_code from **getDeliveryBranches()**
bank_branch_city4	Text	Used for Account Transfers. The delivery bank branch city from **getDeliveryBranches()**
bank_branch_state4	Text	Used for Account Transfers. The delivery bank branch state from **getDeliveryBranches()**
bank_branch_postcode4	Text	Used for Account Transfers. The delivery bank branch postcode from **getDeliveryBranches()**
bank_branch_telephone4	Phone	Used for Account Transfers. The delivery bank branch telephone from **getDeliveryBranches()**
bank_branch_manager4	Text	Used for Account Transfers. The delivery bank branch manager from **getDeliveryBranches()**
intermediary_bank_account_number4	Text	Used for Account Transfers
intermediary_bank4	Text	Used for Account Transfers
intermediary_bank_address4	Text	Used for Account Transfers
intermediary_bank_swift_code4	Text	Used for Account Transfers
collection_point_id	Number	Used for Cash Collection
collection_point_name	Text	Used for Cash Collection
collection_point_code	Text	Used for Cash Collection
collection_point_proc_bank	Text	Used for Cash Collection
collection_point_address	Text	Used for Cash Collection
collection_point_city	Text	Used for Cash Collection
collection_point_state	Text	Used for Cash Collection
collection_point_tel	Phone	Used for Cash Collection
utility_bill_company	Text	Used for Utility Bill Transfers
utility_bill_company_code	Text	Used for Utility Bill Transfers
utility_bill_address1	Text	Used for Utility Bill Transfers
utility_bill_address2	Text	Used for Utility Bill Transfers
utility_bill_address3	Text	Used for Utility Bill Transfers
utility_bill_city	Text	Used for Utility Bill Transfers
utility_bill_state	Text	Used for Utility Bill Transfers
utility_bill_postcode	Text	Used for Utility Bill Transfers
utility_bill_bank	Text	Used for Utility Bill Transfers

utility_bill_account_no	Text	Used for Utility Bill Transfers
utility_bill_bank_bic	Text	Used for Utility Bill Transfers
utility_bill_bank_swift_iban	Text	Used for Utility Bill Transfers
mobile_transfer_number	Phone	Used for Mobile Transfers
mobile_transfer_network	Text	Used for Mobile Transfers
mobile_transfer_network_id	Number	Used for Mobile Transfers
mobile_transfer_network_credit_type_id	Number	Used for Mobile Transfers
mobile_transfer_network_credit_type	Text	Used for Mobile Transfers
home_delivery_notes	Text	Used for Home Delivery

Notes: Only provide the fields that need to be updated. If you do not send a field, the existing value will remain unchanged.

The data saved against the beneficiary will be used when creating a transaction. As a result, make sure to save the relevant data against the beneficiary for the type of transaction that you will create.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <beneficiary>
            <beneficiary_id>42</beneficiary_id>
            <name>John Doe</name>
            <fname>John</fname>
            <mname></mname>
            <lname>Doe</lname>
            <nickname>Johnny</nickname>
            <gender>MALE</gender>
            <avatar>http://remitone.com/benef_avatar/42.jpg</avatar>
            <import_id></import_id>
            <address1>123 Street</address1>
            <address2></address2>
            <address3></address3>
            <city>Beijing</city>
            <state></state>
            <postcode></postcode>
            <country>China</country>
            <country_id>999</country_id>
            <country_iso_code>CN</country_iso_code>
            <nationality>Chinese</nationality>
            <dob></dob>
            <fathers_name></fathers_name>
            <mothers_name></mothers_name>
            <national_id_number></national_id_number>
            <email></email>
            <telephone></telephone>
            <mobile></mobile>
            <sms_payout_mobile></sms_payout_mobile>
```

```
<id_type>CHEQUEBOOK</id_type>
<id_details>123456789</id_details>
<id_start></id_start>
<id_expiry></id_expiry>
<id_scan></id_scan>
<id_issued_by></id_issued_by>
<id_issue_place></id_issue_place>
<id_issue_country></id_issue_country>
<account_transfer>
    <account>
        <bank_account_number></bank_account_number>
        <bank_account_type></bank_account_type>
        <bank_account_name></bank_account_name>
        <bank></bank>
        <bank_iban></bank_iban>
        <bank_swift_code></bank_swift_code>
        <bank_ifsc_code></bank_ifsc_code>
        <bank_bsb_code></bank_bsb_code>
        <bank_routing_transit_number></bank_routing_transit_number>
        <bank_branch></bank_branch>
        <bank_branch_id></bank_branch_id>
        <bank_branch_code></bank_branch_code>
        <bank_branch_city></bank_branch_city>
        <bank_branch_state></bank_branch_state>
        <bank_branch_postcode></bank_branch_postcode>
        <bank_branch_telephone></bank_branch_telephone>
        <bank_branch_manager></bank_branch_manager>
        <additional_bank></additional_bank>
        <additional_bank_branch></additional_bank_branch>
        <intermediary_bank_account_number></intermediary_bank_account_number>
        <intermediary_bank></intermediary_bank>
        <intermediary_bank_address></intermediary_bank_address>
        <intermediary_bank_swift_code></intermediary_bank_swift_code>
    </account>
    <account>
        <bank_account_number></bank_account_number>
        <bank_account_type></bank_account_type>
        <bank_account_name></bank_account_name>
        <bank></bank>
        <bank_iban></bank_iban>
        <bank_swift_code></bank_swift_code>
        <bank_ifsc_code></bank_ifsc_code>
        <bank_bsb_code></bank_bsb_code>
        <bank_routing_transit_number></bank_routing_transit_number>
        <bank_branch></bank_branch>
        <bank_branch_id></bank_branch_id>
        <bank_branch_code></bank_branch_code>
        <bank_branch_city></bank_branch_city>
        <bank_branch_state></bank_branch_state>
        <bank_branch_postcode></bank_branch_postcode>
        <bank_branch_telephone></bank_branch_telephone>
        <bank_branch_manager></bank_branch_manager>
        <intermediary_bank_account_number></intermediary_bank_account_number>
        <intermediary_bank></intermediary_bank>
        <intermediary_bank_address></intermediary_bank_address>
        <intermediary_bank_swift_code></intermediary_bank_swift_code>
    </account>
    <account>
        <bank_account_number></bank_account_number>
        <bank_account_type></bank_account_type>
        <bank_account_name></bank_account_name>
        <bank></bank>
```

```
            <bank_iban></bank_iban>
            <bank_swift_code></bank_swift_code>
            <bank_ifsc_code></bank_ifsc_code>
            <bank_bsb_code></bank_bsb_code>
            <bank_routing_transit_number></bank_routing_transit_number>
            <bank_branch></bank_branch>
            <bank_branch_id></bank_branch_id>
            <bank_branch_code></bank_branch_code>
            <bank_branch_city></bank_branch_city>
            <bank_branch_state></bank_branch_state>
            <bank_branch_postcode></bank_branch_postcode>
            <bank_branch_telephone></bank_branch_telephone>
            <bank_branch_manager></bank_branch_manager>
            <intermediary_bank_account_number></intermediary_bank_account_number>
            <intermediary_bank></intermediary_bank>
            <intermediary_bank_address></intermediary_bank_address>
            <intermediary_bank_swift_code></intermediary_bank_swift_code>
        </account>
        <account>
            <bank_account_number></bank_account_number>
            <bank_account_type></bank_account_type>
            <bank_account_name></bank_account_name>
            <bank></bank>
            <bank_iban></bank_iban>
            <bank_swift_code></bank_swift_code>
            <bank_ifsc_code></bank_ifsc_code>
            <bank_bsb_code></bank_bsb_code>
            <bank_routing_transit_number></bank_routing_transit_number>
            <bank_branch></bank_branch>
            <bank_branch_id></bank_branch_id>
            <bank_branch_code></bank_branch_code>
            <bank_branch_city></bank_branch_city>
            <bank_branch_state></bank_branch_state>
            <bank_branch_postcode></bank_branch_postcode>
            <bank_branch_telephone></bank_branch_telephone>
            <bank_branch_manager></bank_branch_manager>
            <intermediary_bank_account_number></intermediary_bank_account_number>
            <intermediary_bank></intermediary_bank>
            <intermediary_bank_address></intermediary_bank_address>
            <intermediary_bank_swift_code></intermediary_bank_swift_code>
        </account>
    </account_transfer>
    <cash_collection>
        <collection_point_id>1</collection_point_id>
        <collection_point_name>Poste</collection_point_name>
        <collection_point_code>MCNCP</collection_point_code>
        <collection_point_proc_bank>Bank1</collection_point_proc_bank>
        <collection_point_address>Milkyway</collection_point_address>
        <collection_point_city>Beijing</collection_point_city>
        <collection_point_state>Beijing</collection_point_state>
        <collection_point_tel></collection_point_tel>
    </cash_collection>
    <utility_bill>
        <utility_bill_company></utility_bill_company>
        <utility_bill_company_code></utility_bill_company_code>
        <utility_bill_address1></utility_bill_address1>
        <utility_bill_address2></utility_bill_address2>
        <utility_bill_address3></utility_bill_address3>
        <utility_bill_city></utility_bill_city>
        <utility_bill_state></utility_bill_state>
        <utility_bill_postcode></utility_bill_postcode>
        <utility_bill_bank></utility_bill_bank>
```

```
                <utility_bill_account_no></utility_bill_account_no>
                <utility_bill_bank_bic></utility_bill_bank_bic>
                <utility_bill_bank_swift_iban></utility_bill_bank_swift_iban>
                <remitter_pay_own_bill>t</remitter_pay_own_bill>
            </utility_bill>
            <mobile_transfer>
                <mobile_transfer_number></mobile_transfer_number>
                <mobile_transfer_network></mobile_transfer_network>
                <mobile_transfer_network_id></mobile_transfer_network_id>
                <mobile_transfer_network_credit_type_id></mobile_transfer_network_credit_type_id>
                <mobile_transfer_network_credit_type></mobile_transfer_network_credit_type>
            </mobile_transfer>
            <card_transfer>
                <card_number></card_number>
            </card_transfer>
            <home_delivery>
                <home_delivery_notes></home_delivery_notes>
            </home_delivery>
        </beneficiary>
    </result>
</response>
```

---

listBeneficiaries

Group : **beneficiary**
Method : **listBeneficiaries**
This method provides a list of all beneficiaries linked to the remitter.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <beneficiaries>
            <beneficiary>
                <beneficiary_id>42</beneficiary_id>
                <name>John Doe</name>
                <fname>John</fname>
                <mname></mname>
                <lname>Doe</lname>
```

```
                <nickname>Johnny</nickname>
                <gender>MALE</gender>
                <avatar>http://remitone.com/benef_avatar/42.jpg</avatar>
                <country>France</country>
                <country_iso_code>FR</country_iso_code>
                <import_id></import_id>
            </beneficiary>
        </beneficiaries>
    </result>
</response>
```

---

getBeneficiary

Group : **beneficiary**
Method : **getBeneficiary**
This method provides the data of a beneficiary.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
beneficiary_id *	Text	The id of the beneficiary, as per the output of **listBeneficiaries**.

Example Output XML :

```
<?xml version="1.0" encodin/g="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <beneficiary>
            <beneficiary_id>42</beneficiary_id>
            <name>John Doe</name>
            <fname>John</fname>
            <mname></mname>
            <lname>Doe</lname>
            <nickname>Johnny</nickname>
            <gender>MALE</gender>
            <avatar>http://remitone.com/benef_avatar/42.jpg</avatar>
            <import_id></import_id>
            <remitt_benef_relation></remitt_benef_relation>
            <address1>123 Street</address1>
            <address2></address2>
            <address3></address3>
            <city>Beijing</city>
            <state></state>
            <postcode></postcode>
```

```
<country>China</country>
<country_id>999</country_id>
<country_iso_code>CN</country_iso_code>
<nationality>Chinese</nationality>
<dob></dob>
<fathers_name></fathers_name>
<mothers_name></mothers_name>
<national_id_number></national_id_number>
<email></email>
<telephone></telephone>
<mobile></mobile>
<sms_payout_mobile></sms_payout_mobile>
<id_type>CHEQUEBOOK</id_type>
<id_details>123456789</id_details>
<id_start></id_start>
<id_expiry></id_expiry>
<id_scan></id_scan>
<id_issued_by></id_issued_by>
<id_issue_place></id_issue_place>
<id_issue_country></id_issue_country>
<account_transfer>
    <account>
        <bank_account_number></bank_account_number>
        <bank_account_type></bank_account_type>
        <bank_account_name></bank_account_name>
        <bank></bank>
        <bank_iban></bank_iban>
        <bank_swift_code></bank_swift_code>
        <bank_ifsc_code></bank_ifsc_code>
        <bank_bsb_code></bank_bsb_code>
        <bank_routing_transit_number></bank_routing_transit_number>
        <bank_branch></bank_branch>
        <bank_branch_id></bank_branch_id>
        <bank_branch_code></bank_branch_code>
        <bank_branch_city></bank_branch_city>
        <bank_branch_state></bank_branch_state>
        <bank_branch_postcode></bank_branch_postcode>
        <bank_branch_telephone></bank_branch_telephone>
        <bank_branch_manager></bank_branch_manager>
        <additional_bank></additional_bank>
        <additional_bank_branch></additional_bank_branch>
        <intermediary_bank_account_number></intermediary_bank_account_number>
        <intermediary_bank></intermediary_bank>
        <intermediary_bank_address></intermediary_bank_address>
        <intermediary_bank_swift_code></intermediary_bank_swift_code>
    </account>
    <account>
        <bank_account_number></bank_account_number>
        <bank_account_type></bank_account_type>
        <bank_account_name></bank_account_name>
        <bank></bank>
        <bank_iban></bank_iban>
        <bank_swift_code></bank_swift_code>
        <bank_ifsc_code></bank_ifsc_code>
        <bank_bsb_code></bank_bsb_code>
        <bank_routing_transit_number></bank_routing_transit_number>
        <bank_branch></bank_branch>
        <bank_branch_id></bank_branch_id>
        <bank_branch_code></bank_branch_code>
        <bank_branch_city></bank_branch_city>
        <bank_branch_state></bank_branch_state>
        <bank_branch_postcode></bank_branch_postcode>
```

```
            <bank_branch_telephone></bank_branch_telephone>
            <bank_branch_manager></bank_branch_manager>
            <intermediary_bank_account_number></intermediary_bank_account_number>
            <intermediary_bank></intermediary_bank>
            <intermediary_bank_address></intermediary_bank_address>
            <intermediary_bank_swift_code></intermediary_bank_swift_code>
        </account>
        <account>
            <bank_account_number></bank_account_number>
            <bank_account_type></bank_account_type>
            <bank_account_name></bank_account_name>
            <bank></bank>
            <bank_iban></bank_iban>
            <bank_swift_code></bank_swift_code>
            <bank_ifsc_code></bank_ifsc_code>
            <bank_bsb_code></bank_bsb_code>
            <bank_routing_transit_number></bank_routing_transit_number>
            <bank_branch></bank_branch>
            <bank_branch_id></bank_branch_id>
            <bank_branch_code></bank_branch_code>
            <bank_branch_city></bank_branch_city>
            <bank_branch_state></bank_branch_state>
            <bank_branch_postcode></bank_branch_postcode>
            <bank_branch_telephone></bank_branch_telephone>
            <bank_branch_manager></bank_branch_manager>
            <intermediary_bank_account_number></intermediary_bank_account_number>
            <intermediary_bank></intermediary_bank>
            <intermediary_bank_address></intermediary_bank_address>
            <intermediary_bank_swift_code></intermediary_bank_swift_code>
        </account>
        <account>
            <bank_account_number></bank_account_number>
            <bank_account_type></bank_account_type>
            <bank_account_name></bank_account_name>
            <bank></bank>
            <bank_iban></bank_iban>
            <bank_swift_code></bank_swift_code>
            <bank_ifsc_code></bank_ifsc_code>
            <bank_bsb_code></bank_bsb_code>
            <bank_routing_transit_number></bank_routing_transit_number>
            <bank_branch></bank_branch>
            <bank_branch_id></bank_branch_id>
            <bank_branch_code></bank_branch_code>
            <bank_branch_city></bank_branch_city>
            <bank_branch_state></bank_branch_state>
            <bank_branch_postcode></bank_branch_postcode>
            <bank_branch_telephone></bank_branch_telephone>
            <bank_branch_manager></bank_branch_manager>
            <intermediary_bank_account_number></intermediary_bank_account_number>
            <intermediary_bank></intermediary_bank>
            <intermediary_bank_address></intermediary_bank_address>
            <intermediary_bank_swift_code></intermediary_bank_swift_code>
        </account>
    </account_transfer>
    <cash_collection>
        <collection_point_id>1</collection_point_id>
        <collection_point_name>Poste</collection_point_name>
        <collection_point_code>MCNCP</collection_point_code>
        <collection_point_proc_bank>Bank1</collection_point_proc_bank>
        <collection_point_address>Milkyway</collection_point_address>
        <collection_point_city>Beijing</collection_point_city>
        <collection_point_state>Beijing</collection_point_state>
        <collection_point_tel></collection_point_tel>
```

```
            </cash_collection>
            <utility_bill>
                <utility_bill_company></utility_bill_company>
                <utility_bill_company_code></utility_bill_company_code>
                <utility_bill_address1></utility_bill_address1>
                <utility_bill_address2></utility_bill_address2>
                <utility_bill_address3></utility_bill_address3>
                <utility_bill_city></utility_bill_city>
                <utility_bill_state></utility_bill_state>
                <utility_bill_postcode></utility_bill_postcode>
                <utility_bill_bank></utility_bill_bank>
                <utility_bill_account_no></utility_bill_account_no>
                <utility_bill_bank_bic></utility_bill_bank_bic>
                <utility_bill_bank_swift_iban></utility_bill_bank_swift_iban>
                <remitter_pay_own_bill>t</remitter_pay_own_bill>
            </utility_bill>
            <mobile_transfer>
                <mobile_transfer_number></mobile_transfer_number>
                <mobile_transfer_network></mobile_transfer_network>
                <mobile_transfer_network_id></mobile_transfer_network_id>
                <mobile_transfer_network_credit_type_id></mobile_transfer_network_credit_type_id>
                <mobile_transfer_network_credit_type></mobile_transfer_network_credit_type>
            </mobile_transfer>
            <card_transfer>
                <card_number></card_number>
            </card_transfer>
            <home_delivery>
                <home_delivery_notes></home_delivery_notes>
            </home_delivery>
            <transfer_methods>
                  <transfer_method></transfer_method>
            </transfer_methods>
        </beneficiary>
    </result>
</response>
```

---

searchWalletBeneficiary

Group : **beneficiary**
Method : **searchWalletBeneficiary**
This method is used to search for a beneficiary for Wallet Transfer. The beneficiary must already be a registered member on the system with an online login.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
benef_search_email *	Text	Email address of the beneficiary

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <beneficiaries>
            <beneficiary>
                <beneficiary_id>42</beneficiary_id>
                <email>user@online.com</email>
                <country_id>999</country_id>
            </beneficiary>
        </beneficiaries>
    </result>
</response>
```

Transactions

getMobileNetworkOperators

Group : **transaction**
Method : **getMobileNetworkOperators**
This method provides the list of mobile network operators.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <mnos>
            <mno>
                <mobile_network_id>1</mobile_network_id>
                <name>AT&T</name>
                <code>ATT</code>
                <country_id>999</country_id>
                <number_pattern></number_pattern>
                <regex_pattern></regex_pattern>
            </mno>
            ...
        </mnos>
    </result>
</response>
```

---

getMobileNetworkCreditTypes

Group : **transaction**
Method : **getMobileNetworkCreditTypes**
This method provides the list of mobile network credit types for a specified MNO.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
mobile_network_operator_id *	Text	The id of the Mobile Network Operator, as per the output of **getMobileNetworkOperators**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <mobile_network_credit_types>
            <mobile_network_credit_type>
                <credit_type_id>1</credit_type_id>
                <name>Top-up</name>
                <code>ATT</code>
                <country_id>999</country_id>
                <type>Open</type>
                <currency>USD</currency>
                <minimum>5.00</minimum>
                <maximum>1337.00</maximum>
                <amount></amount>
            </mobile_network_credit_type>
            ...
        </mobile_network_credit_types>
    </result>
</response>
```

Notes: The Type can be either Open or Fixed. If Open, you will make use of the minimum and maximum. If Fixed, only the amount will matter.

---

getUtilityBillCompanies

Group : **transaction**
Method : **getUtilityBillCompanies**
This method provides the list of utility bill companies.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging

		in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <utility_bill_companies>
            <company>
                  <utility_company_id>3</utility_company_id>
                  <company_name>British Telecom</company_name>
                  <company_code>BT</company_code>
                  <country_id>996</country_id>
                  <address1>123 Street</address1>
                  <address2></address2>
                  <address3></address3>
                  <city>London</city>
                  <state></state>
                  <postcode>E15 1BT</postcode>
                  <iban_no>IT40S0542811101000000123456</iban_no>
                  <bic_no>BC23456</bic_no>
            </company>
        </utility_bill_companies>
    </result>
</response>
```

---

searchUtilityBillCompanies

Group : **transaction**
Method : **searchUtilityBillCompanies**
This method provides the list of utility bill companies corresponding to the search.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.
search *	Text	The keyword search term.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <utility_bill_companies>
            <company>
                  <utility_company_id>3</utility_company_id>
                  <company_name>British Telecom</company_name>
                  <company_code>BT</company_code>
                  <country_id>996</country_id>
                  <address1>123 Street</address1>
                  <address2></address2>
                  <address3></address3>
                  <city>London</city>
                  <state></state>
                  <postcode>E15 1BT</postcode>
                  <iban_no>IT40S0542811101000000123456</iban_no>
                  <bic_no>BC23456</bic_no>
            </company>
        </utility_bill_companies>
    </result>
</response>
```

---

getDeliveryBanks

Group : **transaction**
Method : **getDeliveryBanks**
This method provides a list of delivery banks.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <delivery_banks>
            <delivery_bank>
```

```
                <bank_id>14</bank_id>
                <name>delivery bank</name>
                <country_id>999</country_id>
                <bank_code>del123</bank_code>
                <swift_code>sw123</swift_code>
                <address>address 123</address>
                <city>city123</city>
                <state>state123</state>
                <telephone>123</telephone>
                <account_number_mask>123</account_number_mask>
            </delivery_bank>
        </delivery_banks>
    </result>
</response>
```

---

getDeliveryBranchStates

Group : **transaction**
Method : **getDeliveryBranchStates**
This method provides a list of states that branches are located in. Data retrieved from here can be used in conjunction with getDeliveryBranchCities() and getDeliveryBranches() to get a list of delivery branches.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
delivery_bank_id *	Text	The id of the Delivery Bank, as per the output of **getDeliveryBanks**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <delivery_branch_states>
            <state>state123</state>
            <state>state456</state>
        </delivery_branch_states>
    </result>
</response>
```

---

getDeliverybranchCities

Group : **transaction**
Method : **getDeliveryBranchCities**
This method provides a list of cities that branches are located in. Data retrieved from here can be used in conjunction with getDeliveryBranches() to get a list of delivery branches.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
delivery_bank *	Text	The name of the Delivery Bank, as per the output of **getDeliveryBanks**.
delivery_branch_state	Text	The name of the Delivery Branch State, as per the output of **getDeliveryBranchStates**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <delivery_branch_cities>
            <city>city123</city>
            <city>city456</city>
        </delivery_branch_cities>
    </result>
</response>
```

---

getDeliveryBranches

Group : **transaction**
Method : **getDeliveryBranches**
This method provides a list of delivery branches.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter

session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
delivery_bank *	Text	The name of the Delivery Bank, as per the output of **getDeliveryBanks**.
delivery_branch_state	Text	The name of the Delivery Branch State, as per the output of **getDeliveryBranchStates**.
delivery_branch_city	Text	The name of the Delivery Branch City, as per the output of **getDeliveryBranchCities**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <delivery_bank_branches>
            <delivery_bank_branch>
                <id>1</id>
                <name>Branch123</name>
                <branch_code>BRANCH-123</branch_code>
                <delivery_bank>1</delivery_bank>
                <city>city123</city>
                <state>state123</state>
                <telephone>012345678</telephone>
                <manager>manager123</manager>
            </delivery_bank_branch>
        </delivery_bank_branches>
    </result>
</response>
```

---

getCollectionPointStates

Group : **transaction**
Method : **getCollectionPointStates**
This method provides a list of states that collection points are located in. Data retrieved from here can be used in conjunction with getCollectionPointCities() and getCollectionPoints() to get a list of collection points.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <collection_point_states>
            <state>state123</state>
            <state>state456</state>
            ...
        </collection_point_states>
    </result>
</response>
```

---

getCollectionPointCities

Group : **transaction**
Method : **getCollectionPointCities**
This method provides a list of cities that collection points are located in. Data retrieved from here can be used in conjunction with getCollectionPoints() to get a list of collection points.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.
collection_point_state	Text	The name of the Collection Point State, as per the output of **getCollectionPointStates**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <collection_point_cities>
            <city>city123</city>
```

```
            <city>city456</city>
            ...
        </collection_point_cities>
    </result>
</response>
```

---

getCollectionPoints

Group : **transaction**
Method : **getCollectionPoints**
This method provides a list of collection points. Data retrieved using the method getCollectionPointCities() can be used with this method to get a list of collection points.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.
collection_point_state	Text	The name of the Collection Point State, as per the output of **getCollectionPointStates**.
collection_point_city	Text	The name of the Collection Point City, as per the output of **getCollectionPointCities**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <collection_points>
            <collection_point>
                <collection_id>1</collection_id>
                <name>colltion123</name>
                <bank>Bank 123</bank>
                <delivery_bank>1</delivery_bank>
                <address>address123</address>
                <city>city123</city>
                <state>state123</state>
                <country_id>123</country_id>
                <code>COLLECTION-123</code>
                <telephone>012345678</telephone>
```

```
                <fax>fax123</fax>
                <email>email123</email>
                <working_hours>1</working_hours>
                <contact_person>person123</contact_person>
                <default_in_country>t</default_in_country>
                <enabled>t</enabled>
                <collection_pin_prefix>123</collection_pin_prefix>
                <rate_markup>1</rate_markup>
            </collection_point>
        </collection_points>
    </result>
</response>
```

---

getCharges

Group : **transaction**
Method : **getCharges**
This method provides the charges for transaction creation.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country *	Text	The name of the Destination Country, as per the output of **getDestinationCountries**.
trans_type *	Text	Account / Cash Collection / Card Transfer / Home Delivery / Mobile Transfer
payment_method *	Code	See **Appendix B** for codes.
service_level *	Code	See **Appendix C** for codes.
sms_confirmation *	Boolean	t / f
sms_notification *	Boolean	t / f
sms_benef_confirmation *	Boolean	t / f
amount_type *		Whether the amount is specified in the source currency or the destination currency. Possible values : SOURCE / DESTINATION.
amount_to_send *	Numeric	
source_currency	Text	The 3 letters ISO currency code, as listed in **Appendix A** or per the output of **getTransactionUISettings**.

destination_currency	Text	The 3 letters ISO currency code, as listed in **Appendix A** or per the output of **getTransactionUISettings**.
collection_point_id	Integer	The id of the Collection Point, as per the output of **getCollectionPoints**.
benef_branch_id	Integer	The id of the Delivery Bank Branch, as per the output of **getDeliveryBranches**.
benef_bank	Text	The name of the Delivery Bank, as per the output of **getDeliveryBanks**.
benef_mobiletransfer_network_credit_type_id	Integer	The id of the Mobile Network Credit Type, as per the output of **getMobileNetworkCreditTypes**.
utility_company	Text	The name of the Utility Bill Company, free text or as per the output of **getUtilityBillCompanies**.
promotion_code	Text	Discount code for promotions

Notes: If source and destination currencies are not entered,then the default source and destination currencies will be used instead.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <source_country_iso_code>GB</source_country_iso_code>
        <source_currency>GBP</source_currency>
        <source_amount>100</source_amount>
        <rate>37.53</rate>
        <destination_country_iso_code>PH</destination_country_iso_code>
        <destination_currency>PHP</destination_currency>
        <destination_amount>3753.00</destination_amount>
        <commission>10.50</commission>
        <agent_fee>0.00</agent_fee>
        <hq_fee>0.50</hq_fee>
        <total_charges>11.00</total_charges>
        <tax>3.40</tax>
        <remitt_pay>111.00</remitt_pay>
        <commission_before_promotion>10.50</commission_before_promotion>
        <promotion_names></promotion_names>
        <promotion_ids></promotion_ids>
    </result>
</response>
```

createTransaction

Group : **transaction**
Method : **createTransaction**
This method initiates the creation of a transaction.

Input fields:

Name	Type	Notes
username *	Text	The username of the current user
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
trans_type *	Text	Account / Cash Collection / Card Transfer / Home Delivery / Mobile Transfer
beneficiary_id *	Number	The id of the Beneficiary to send money to, as per the output of **listBeneficiaries** or **createBeneficiary**.
account_item_number	Number	Required when creating an Account Transfer, specify the number (from 1 to 4 included) of the Beneficiary Account to use.
source_currency *	Text	Source currency, as per the result of **getTransactionUISettings**.
destination_currency *	Text	Destination currency, as per the result of **getTransactionUISettings**.
amount_type *		Whether the amount is specified in the source currency or the destination currency. Possible values : SOURCE / DESTINATION.
amount *	Numeric	
purpose	Code	See **Appendix E** for codes, as per the output of **getTransactionUISettings**
source_of_income	Code	See **Appendix D** for codes, as per the output of **getTransactionUISettings**
payment_method *	Code	See **Appendix B** for codes, as per the output of **getTransactionUISettings**
payment_token	Text	Optional token provided by the payment gateway
remitter_wallet_currency	Text	Currency of the wallet used for the payment, as per the result of **getWallets**. Required if payment method is the Wallet.
service_level *	Code	See **Appendix C** for codes, as per the output of **getTransactionUISettings**

promotion_code	Text	
sms_confirmation	Boolean	t / f
sms_notification	Boolean	t / f
sms_mobile	Text	Mobile number for Remitter to send SMS to
sms_benef_confirmation	Boolean	t / f
sms_benef_mobile	Text	Mobile number for Beneficiary to send SMS to
utility_bill_invoice	Text	Used for Utility Bill transactions. Requirement is defined as per the output of **getTransactionUISettings**
utility_bill_description	Text	Used for Utility Bill transactions. Requirement is defined as per the output of **getTransactionUISettings**
comments_to_beneficiary	Text	

NOTE: You may need to call **updateBeneficiary** before invoking this method so that the information saved against that beneficiary are up to date.

NOTE 2: After invoking this method, **confirmTransaction** must be called to complete the creation of the transaction.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
	<status>SUCCESS</status>
	<result>
		<transaction>
			<trans_session_id>RA20079127</trans_session_id>
			<purpose></purpose>
			<source_of_income></source_of_income>
			<delivery_date>2010-05-18 00:00:00</delivery_date>
			<trans_type>Account</trans_type>
			<benef_id>42</benef_id>
			<benef_name>ZIAD HASSAN SYFULLAH</benef_name>
			<benef_tel>004412345678</benef_tel>
			<benef_mobile>+4412345678</benef_mobile>
			<benef_email></benef_email>
			<benef_id_type>NONE</benef_id_type>
			<benef_id_detail></benef_id_detail>
			<collection_point></collection_point>
			<collection_point_id></collection_point_id>
			<collection_point_bank></collection_point_bank>
			<collection_point_code></collection_point_code>
			<collection_point_address></collection_point_address>
			<collection_point_city></collection_point_city>
			<collection_point_state></collection_point_state>
			<collection_pin></collection_pin>
```

```
            <benef_bank_account_number>1234567890</benef_bank_account_number>
            <benef_bank_account_type></benef_bank_account_type>
            <benef_bank_account_name></benef_bank_account_name>
            <benef_bank_iban></benef_bank_iban>
            <benef_bank_swift_code>ABC123</benef_bank_swift_code>
            <benef_bank_bsb_code></benef_bank_bsb_code>
            <benef_bank_ifsc_code>12340678912</benef_bank_ifsc_code>
            <benef_bank>PRIME BANK LIMITED</benef_bank>
            <benef_bank_city>CHITTAGONG</benef_bank_city>
            <benef_bank_state>CHITTAGONG</benef_bank_state>
            <benef_branch>AGRABAD</benef_branch>
            <benef_branch_code></benef_branch_code>
            <benef_branch_telephone></benef_branch_telephone>
            <benef_branch_manager></benef_branch_manager>
            <benef_bank_routine_transit_number></benef_bank_routine_transit_number>
            <additional_benef_bank></additional_benef_bank>
            <additional_benef_bank_branch></additional_benef_bank_branch>
            <benef_card_number></benef_card_number>
            <benef_address1>BOGRA</benef_address1>
            <benef_address2></benef_address2>
            <benef_address3></benef_address3>
            <benef_city>BOGRA</benef_city>
            <benef_state></benef_state>
            <benef_postcode></benef_postcode>
            <benef_mobiletransfer_number>001234567</benef_mobiletransfer_number>
            <benef_mobiletransfer_network>mobileNetwork</benef_mobiletransfer_network>
            <benef_mobiletransfer_network_credit_type></benef_mobiletransfer_network_credit_type>
            <delivery_notes></delivery_notes>
            <utilitybill_company></utilitybill_company>
            <utilitybill_account_no></utilitybill_account_no>
            <utilitybill_invoice></utilitybill_invoice>
            <utilitybill_address1></utilitybill_address1>
            <utilitybill_address2></utilitybill_address2>
            <utilitybill_address3></utilitybill_address3>
            <utilitybill_city></utilitybill_city>
            <utilitybill_state></utilitybill_state>
            <utilitybill_postcode></utilitybill_postcode>
            <utilitybill_bank></utilitybill_bank>
            <utilitybill_bank_code></utilitybill_bank_code>
            <utilitybill_bank_bic></utilitybill_bank_bic>
            <utilitybill_description></utilitybill_description>
            <payment_method>1</payment_method>
            <payment_token></payment_token>
            <send_country>SG</send_country>
            <send_currency>SGD</send_currency>
            <send_amount>157.57</send_amount>
            <rate>49.2800</rate>
            <commission>5.00</commission>
            <commission_before_promotion>5.00</commission_before_promotion>
            <promotion_names></promotion_names>
            <promotion_ids></promotion_ids>
            <service_level>1</service_level>
            <fees>0</fees>
            <tax>0</tax>
            <remitter_pay>162.57</remitter_pay>
            <receive_country>BD</receive_country>
            <receive_currency>BDT</receive_currency>
            <receive_amount>7765.05</receive_amount>
        </transaction>
        <sms_confirmation_code>true</sms_confirmation_code>
        <email_confirmation_code>false</email_confirmation_code>
    </result>
```

```
</response>
```

Notes: The sms_confirmation_code and email_confirmation_code flags indicates if the remitter needs to send codes when calling **confirmTransaction**.

---

confirmTransaction

Group : **transaction**
Method : **confirmTransaction**
This method confirms the creation of a transaction.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
trans_session_id *	Numeric	Temporary ID generated by the system and returned as result of **createTransaction**
email_confirmation_code	Text	**DEPRECATED Please use confirmation_code.** The code that may have been sent to the remitter email. Required if indicated by the output of **createTransaction**
sms_confirmation_code	Text	**DEPRECATED Please use confirmation_code.** The code that may have been sent to the remitter by SMS. Required if indicated by the output of **createTransaction**
confirmation_code	Text	The confirmation code for both the email and sms is the same. Required if indicated by the output of **createTransaction**
confirmation_pin	Text	May be required based on the configuration of the system. Optional by default.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
```

```
<result>
        <transaction>
              <trans_ref>RA20079127</trans_ref>
              <status>HQ_OK</status>
              <creation_date>2010-05-11 10:28:02.244553</creation_date>
              <purpose></purpose>
              <source_of_income></source_of_income>
              <processing_bank></processing_bank>
              <bank_accept_date>2010-05-11 10:28:35.435216</bank_accept_date>
              <bank_branch_accept_date></bank_branch_accept_date>
              <delivery_date>2010-05-18 00:00:00</delivery_date>
              <processed_date></processed_date>
              <processed_by></processed_by>
              <payment_gateway_acknowledged>false</payment_gateway_acknowledged>
              <trans_type>Account</trans_type>
              <benef_id>42</benef_id>
              <benef_name>ZIAD HASSAN SYFULLAH</benef_name>
              <benef_tel>004412345678</benef_tel>
              <benef_mobile>+4412345678</benef_mobile>
              <benef_email></benef_email>
              <benef_id_type>NONE</benef_id_type>
              <benef_id_detail></benef_id_detail>
              <collection_point></collection_point>
              <collection_point_id></collection_point_id>
              <collection_point_bank></collection_point_bank>
              <collection_point_code></collection_point_code>
              <collection_point_address></collection_point_address>
              <collection_point_city></collection_point_city>
              <collection_point_state></collection_point_state>
              <collection_pin></collection_pin>
              <benef_bank_account_number>1234567890</benef_bank_account_number>
              <benef_bank_account_type></benef_bank_account_type>
              <benef_bank_account_name></benef_bank_account_name>
              <benef_bank_iban></benef_bank_iban>
              <benef_bank_swift_code>ABC123</benef_bank_swift_code>
              <benef_bank_bsb_code></benef_bank_bsb_code>
              <benef_bank_ifsc_code>12340678912</benef_bank_ifsc_code>
              <benef_bank>PRIME BANK LIMITED</benef_bank>
              <benef_bank_city>CHITTAGONG</benef_bank_city>
              <benef_bank_state>CHITTAGONG</benef_bank_state>
              <benef_branch>AGRABAD</benef_branch>
              <benef_branch_code></benef_branch_code>
              <benef_branch_telephone></benef_branch_telephone>
              <benef_branch_manager></benef_branch_manager>
              <benef_bank_routine_transit_number></benef_bank_routine_transit_number>
              <additional_benef_bank></additional_benef_bank>
              <additional_benef_bank_branch></additional_benef_bank_branch>
              <benef_card_number></benef_card_number>
              <benef_address1>BOGRA</benef_address1>
              <benef_address2></benef_address2>
              <benef_address3></benef_address3>
              <benef_city>BOGRA</benef_city>
              <benef_state></benef_state>
              <benef_postcode></benef_postcode>
              <benef_mobiletransfer_number>001234567</benef_mobiletransfer_number>
              <benef_mobiletransfer_network>mobileNetwork</benef_mobiletransfer_network>
              <benef_mobiletransfer_network_credit_type></benef_mobiletransfer_network_credit_type>
              <delivery_notes></delivery_notes>
              <utilitybill_company></utilitybill_company>
              <utilitybill_account_no></utilitybill_account_no>
              <utilitybill_invoice></utilitybill_invoice>
              <utilitybill_address1></utilitybill_address1>
```

```
                    <utilitybill_address2></utilitybill_address2>
                    <utilitybill_address3></utilitybill_address3>
                    <utilitybill_city></utilitybill_city>
                    <utilitybill_state></utilitybill_state>
                    <utilitybill_postcode></utilitybill_postcode>
                    <utilitybill_bank></utilitybill_bank>
                    <utilitybill_bank_code></utilitybill_bank_code>
                    <utilitybill_bank_bic></utilitybill_bank_bic>
                    <utilitybill_description></utilitybill_description>
                    <payment_method>1</payment_method>
                    <payment_token></payment_token>
                    <send_country>SG</send_country>
                    <send_currency>SGD</send_currency>
                    <send_amount>157.57</send_amount>
                    <rate>49.2800</rate>
                    <commission>5.00</commission>
                    <commission_before_promotion>5.00</commission_before_promotion>
                    <promotion_names></promotion_names>
                    <promotion_ids></promotion_ids>
                    <service_level>1</service_level>
                    <fees>0</fees>
                    <tax>0</tax>
                    <remitter_pay>162.57</remitter_pay>
                    <receive_country>BD</receive_country>
                    <receive_currency>BDT</receive_currency>
                    <receive_amount>7765.05</receive_amount>
                    <bank_sequence>2929</bank_sequence>
                    <pay_method></pay_method>
                    <issue_date></issue_date>
                    <bank_ref></bank_ref>
                    <bank_comments></bank_comments>
                    <bank_credit_date></bank_credit_date>
                    <bank_clear_date></bank_clear_date>
                    <benef_trans_ref>GHT33762634</benef_trans_ref>
            </transaction>
      </result>
</response>
```

---

requestTransactionConfirmationCode

Group : **transaction**
Method : **requestTransactionConfirmationCode**
This method re-send the confirmation codes that were eventually sent by email and/or SMS during the transaction creation process. This method is optional. The codes will be sent automatically after successfully calling createTransaction() anyway. This is only used if the user does not receive the first codes.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging

		in via **login** or **loginPin**.
trans_session_id *	Numeric	Temporary ID generated by the system and returned as result of **createTransaction**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <sms_confirmation_code>true</sms_confirmation_code>
        <email_confirmation_code>false</email_confirmation_code>
    </result>
</response>
```

---

paymentCleared

Group : **transaction**
Method : **paymentCleared**
This method marks transactions in PENDING_CLEARANCE status as payment cleared.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
trans_ref *	Text	The transaction reference, as per the output of **confirmTransaction**.
security_hash *	Text	The hash that should be generated to authenticate the action. RemitONE will provide the recipe during the setup.
payment_token	Text	Optional token provided by the payment gateway

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
       <trans_ref>RC12321321321</trans_ref>
```

```
        <status>SENT_FOR_DELIVERY</status>
    </result>
</response>
```

---

listTransactions

Group : **transaction**
Method : **listTransactions**
This method provides a list of transactions created by the remitter.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
from_date	Date	YYYY-MM-DD
to_date	Date	YYYY-MM-DD
trans_type	Text	Account / Cash Collection / Card Transfer / Home Delivery / Mobile Transfer / Utility Bill
status	Text	See **Appendix F**
destination_country	Text	Country Name, as per output of **getDestinationCountries**
source_currency	Text	Source currency, as per the result of **getTransactionUISettings**.
destination_currency	Text	Destination currency, as per the result of **getTransactionUISettings**.
beneficiary_id	Number	The id of the Beneficiary, as per the output of **listBeneficiaries** or **createBeneficiary**.
payment_method	Code	See **Appendix B** for codes, as per the output of **getTransactionUISettings**
remitter_wallet_currency	Text	Currency of the wallet used for the payment, as per the result of **getWallets**. Required if payment method is the Wallet.
order	Text	ASC / DESC. By default, transactions are listed in inverse chronological order (DESC)

Notes: The number of transactions returned will be limited to 200.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
            <transactions>
                  <transaction>
                        <trans_ref>RA20079112</trans_ref>
                        <trans_type>Account</trans_type>
                        <status>HQ_OK</status>
                        <creation_date>2010-05-07 16:25:12.26266</creation_date>
                        <processed_date></processed_date>
                        <originating_country>SG</originating_country>
                        <destination_country>BD</destination_country>
                        <source_currency>SGD</source_currency>
                        <source_amount>85.00</source_amount>
                        <dest_currency>BDT</dest_currency>
                        <dest_amount>4188.80</dest_amount>
                        <payment_method>1</payment_method>
                        <benef_id>42</benef_id>
                        <benef_name>Kevin Smith</benef_name>
                        <benef_mobile>0987654321</benef_mobile>
                        <compliance_needed>f</compliance_needed>
                        <compliance_checked>f</compliance_checked>
                  </transaction>
            </transactions>
      </result>
</response>
```

---

getTransaction

Group : **transaction**
Method : **getTransaction**
This method retrieves the details of a single transaction.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
trans_ref *	Text	The transaction reference

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<response>
    <status>SUCCESS</status>
    <result>
            <transaction>
                  <trans_ref>RA20079127</trans_ref>
                  <status>HQ_OK</status>
                  <creation_date>2010-05-11 10:28:02.244553</creation_date>
                  <purpose></purpose>
                  <source_of_income></source_of_income>
                  <processing_bank></processing_bank>
                  <bank_accept_date>2010-05-11 10:28:35.435216</bank_accept_date>
                  <bank_branch_accept_date></bank_branch_accept_date>
                  <delivery_date>2010-05-18 00:00:00</delivery_date>
                  <processed_date></processed_date>
                  <processed_by></processed_by>
                  <payment_gateway_acknowledged>false</payment_gateway_acknowledged>
                  <trans_type>Account</trans_type>
                  <benef_id>42</benef_id>
                  <benef_name>ZIAD HASSAN SYFULLAH</benef_name>
                  <benef_tel>004412345678</benef_tel>
                  <benef_mobile>+4412345678</benef_mobile>
                  <benef_email></benef_email>
                  <benef_id_type>NONE</benef_id_type>
                  <benef_id_detail></benef_id_detail>
                  <collection_point></collection_point>
                  <collection_point_id></collection_point_id>
                  <collection_point_bank></collection_point_bank>
                  <collection_point_code></collection_point_code>
                  <collection_point_address></collection_point_address>
                  <collection_point_city></collection_point_city>
                  <collection_point_state></collection_point_state>
                  <collection_pin></collection_pin>
                  <benef_bank_account_number>1234567890</benef_bank_account_number>
                  <benef_bank_account_type></benef_bank_account_type>
                  <benef_bank_account_name></benef_bank_account_name>
                  <benef_bank_iban></benef_bank_iban>
                  <benef_bank_swift_code>ABC123</benef_bank_swift_code>
                  <benef_bank_bsb_code></benef_bank_bsb_code>
                  <benef_bank_ifsc_code>12340678912</benef_bank_ifsc_code>
                  <benef_bank>PRIME BANK LIMITED</benef_bank>
                  <benef_bank_city>CHITTAGONG</benef_bank_city>
                  <benef_bank_state>CHITTAGONG</benef_bank_state>
                  <benef_branch>AGRABAD</benef_branch>
                  <benef_branch_code></benef_branch_code>
                  <benef_branch_telephone></benef_branch_telephone>
                  <benef_branch_manager></benef_branch_manager>
                  <benef_bank_routine_transit_number></benef_bank_routine_transit_number>
                  <additional_benef_bank></additional_benef_bank>
                  <additional_benef_bank_branch></additional_benef_bank_branch>
                  <benef_card_number></benef_card_number>
                  <benef_address1>BOGRA</benef_address1>
                  <benef_address2></benef_address2>
                  <benef_address3></benef_address3>
                  <benef_city>BOGRA</benef_city>
                  <benef_state></benef_state>
                  <benef_postcode></benef_postcode>
                  <benef_mobiletransfer_number>001234567</benef_mobiletransfer_number>
                  <benef_mobiletransfer_network>mobileNetwork</benef_mobiletransfer_network>
                  <benef_mobiletransfer_network_credit_type></benef_mobiletransfer_network_credit_type>
                  <delivery_notes></delivery_notes>
                  <utilitybill_company></utilitybill_company>
                  <utilitybill_account_no></utilitybill_account_no>
```

```
                    <utilitybill_invoice></utilitybill_invoice>
                    <utilitybill_address1></utilitybill_address1>
                    <utilitybill_address2></utilitybill_address2>
                    <utilitybill_address3></utilitybill_address3>
                    <utilitybill_city></utilitybill_city>
                    <utilitybill_state></utilitybill_state>
                    <utilitybill_postcode></utilitybill_postcode>
                    <utilitybill_bank></utilitybill_bank>
                    <utilitybill_bank_code></utilitybill_bank_code>
                    <utilitybill_bank_bic></utilitybill_bank_bic>
                    <utilitybill_description></utilitybill_description>
                    <payment_method>1</payment_method>
                    <payment_token></payment_token>
                    <send_country>SG</send_country>
                    <send_currency>SGD</send_currency>
                    <send_amount>157.57</send_amount>
                    <rate>49.2800</rate>
                    <commission>5.00</commission>
                    <commission_before_promotion>5.00</commission_before_promotion>
                    <promotion_names></promotion_names>
                    <promotion_ids></promotion_ids>
                    <service_level>1</service_level>
                    <fees>0</fees>
                    <tax>0</tax>
                    <remitter_pay>162.57</remitter_pay>
                    <receive_country>BD</receive_country>
                    <receive_currency>BDT</receive_currency>
                    <receive_amount>7765.05</receive_amount>
                    <bank_sequence>2929</bank_sequence>
                    <pay_method></pay_method>
                    <issue_date></issue_date>
                    <bank_ref></bank_ref>
                    <bank_comments></bank_comments>
                    <bank_credit_date></bank_credit_date>
                    <bank_clear_date></bank_clear_date>
                    <benef_trans_ref>GHT33762634</benef_trans_ref>
            </transaction>
      </result>
</response>
```

---

getTransactionPaymentInstructions

Group : **transaction**
Method : **getTransactionPaymentInstructions**
This method retrieves the instruction details of a transaction

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging

		in via **login** or **loginPin**.
trans_ref *	Text	The transaction reference

Payment by Bank Transfer Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
            <instructions>
                  <payment_method_name>ORMBankTransfer</payment_method_name>
                  <logo_url>https://yourmto.com/payment_method.jpg</logo_url>
                  <type>bank_transfer</type>
                  <message>Please make the payment by making an account transfer
to our bank account using the following details and use the reference of this
transfer as the reference for the payment</message>
                  <account_number>9876543210</account_number>
                  <sort_code>03-02-01</sort_code>
                  <bank_name>Barclays plc</bank_name>
            </instructions>
      </result>
</response>
```

Notes: The different “type” values are listed in **Appendix J**.

Payment by Credit/Debit Card Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
            <instructions>
                  <type>card</type>
                  <message>Make payment for your transfer using Secure Trading
128-bit encrypted secure checkout. Click the button below to make the payment
for your transfer now.</message>
                  <sitereference>123456789</sitereference>
                  <mainamount>1337</mainamount>
                  <currencyiso3a>GBP</currencyiso3a>
                  <version>1</version>
                  <orderreference>123456789</orderreference>
                  <trans_ref>RC123456789</trans_ref>
                  <sitesecurity>XXX</sitesecurity>
                  <accounttypedescription>YYY</accounttypedescription>
            </instructions>
      </result>
</response>
```

Notes: The different “type” values are listed in **Appendix J**.
The extra fields depends on the Payment Gateway. While this example illustrate an integration with Secure Trading, other payment gateways will feature

different fields.

getTempTransaction

Group : **transaction**
Method : **getTempTransaction**
This method retrieves the details of a single transaction.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
trans_ref *	Text	The transaction reference

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
	<status>SUCCESS</status>
	<result>
		<transaction>
			<trans_session_id>RA20079127</trans_session_id>
			<purpose></purpose>
			<source_of_income></source_of_income>
			<delivery_date>2010-05-18 00:00:00</delivery_date>
			<trans_type>Account</trans_type>
			<benef_id>42</benef_id>
			<benef_name>ZIAD HASSAN SYFULLAH</benef_name>
			<benef_tel>004412345678</benef_tel>
			<benef_mobile>+4412345678</benef_mobile>
			<benef_email></benef_email>
			<benef_id_type>NONE</benef_id_type>
			<benef_id_detail></benef_id_detail>
			<collection_point></collection_point>
			<collection_point_id></collection_point_id>
			<collection_point_bank></collection_point_bank>
			<collection_point_code></collection_point_code>
			<collection_point_address></collection_point_address>
			<collection_point_city></collection_point_city>
			<collection_point_state></collection_point_state>
			<collection_pin></collection_pin>
			<benef_bank_account_number>1234567890</benef_bank_account_number>
			<benef_bank_account_type></benef_bank_account_type>
			<benef_bank_account_name></benef_bank_account_name>
			<benef_bank_iban></benef_bank_iban>
			<benef_bank_swift_code>ABC123</benef_bank_swift_code>
			<benef_bank_bsb_code></benef_bank_bsb_code>
			<benef_bank_ifsc_code>12340678912</benef_bank_ifsc_code>
			<benef_bank>PRIME BANK LIMITED</benef_bank>
			<benef_bank_city>CHITTAGONG</benef_bank_city>
			<benef_bank_state>CHITTAGONG</benef_bank_state>
			<benef_branch>AGRABAD</benef_branch>
			<benef_branch_code></benef_branch_code>
			<benef_branch_telephone></benef_branch_telephone>
```

```
                    <benef_branch_manager></benef_branch_manager>
                    <benef_bank_routine_transit_number></benef_bank_routine_transit_number>
                    <additional_benef_bank></additional_benef_bank>
                    <additional_benef_bank_branch></additional_benef_bank_branch>
                    <benef_card_number></benef_card_number>
                    <benef_address1>BOGRA</benef_address1>
                    <benef_address2></benef_address2>
                    <benef_address3></benef_address3>
                    <benef_city>BOGRA</benef_city>
                    <benef_state></benef_state>
                    <benef_postcode></benef_postcode>
                    <benef_mobiletransfer_number>001234567</benef_mobiletransfer_number>
                    <benef_mobiletransfer_network>mobileNetwork</benef_mobiletransfer_network>
                    <benef_mobiletransfer_network_credit_type></benef_mobiletransfer_network_credit_type>
                    <delivery_notes></delivery_notes>
                    <utilitybill_company></utilitybill_company>
                    <utilitybill_account_no></utilitybill_account_no>
                    <utilitybill_invoice></utilitybill_invoice>
                    <utilitybill_address1></utilitybill_address1>
                    <utilitybill_address2></utilitybill_address2>
                    <utilitybill_address3></utilitybill_address3>
                    <utilitybill_city></utilitybill_city>
                    <utilitybill_state></utilitybill_state>
                    <utilitybill_postcode></utilitybill_postcode>
                    <utilitybill_bank></utilitybill_bank>
                    <utilitybill_bank_code></utilitybill_bank_code>
                    <utilitybill_bank_bic></utilitybill_bank_bic>
                    <utilitybill_description></utilitybill_description>
                    <payment_method>1</payment_method>
                    <payment_token></payment_token>
                    <send_country>SG</send_country>
                    <send_currency>SGD</send_currency>
                    <send_amount>157.57</send_amount>
                    <rate>49.2800</rate>
                    <commission>5.00</commission>
                    <commission_before_promotion>5.00</commission_before_promotion>
                    <promotion_names></promotion_names>
                    <promotion_ids></promotion_ids>
                    <service_level>1</service_level>
                    <fees>0</fees>
                    <tax>0</tax>
                    <remitter_pay>162.57</remitter_pay>
                    <receive_country>BD</receive_country>
                    <receive_currency>BDT</receive_currency>
                    <receive_amount>7765.05</receive_amount>
            </transaction>
            <sms_confirmation_code>true</sms_confirmation_code>
            <email_confirmation_code>false</email_confirmation_code>
      </result>
</response>
```

Notes: The sms_confirmation_code and email_confirmation_code flags indicates if the remitter needs to send codes when calling **confirmTransaction**.

Rates

getRates

Group : **rate**
Method : **getRates**
This method provides the rates commonly displayed within the dashboard.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country	Text	The name of the Destination Country, as per the output of **getDestinationCountries**.
source_currency	Text	The 3 letters ISO currency code, as listed in **Appendix A**, as per the output of **getTransactionUISettings**
destination_currency	Text	The 3 letters ISO currency code, as listed in **Appendix A**, as per the output of **getTransactionUISettings**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <rates>
            <rate>
                  <destination_country>Canada</destination_country>
                  <source_currency>EUR</source_currency>
                  <destination_currency>CAD</destination_currency>
                  <account>1.444907</account>
                  <cash_collection>1.444851</cash_collection>
                  <card>1.444767</card>
                  <home_delivery>1.444921</home_delivery>
                  <utility_bill>1.445002</utility_bill>
                  <mobile_transfer>1.444899</mobile_transfer>
                  <wallet_transfer>1.444950</wallet_transfer>
            </rate>
        </rates>
    </result>
</response>
```

Remitter

getProfile

Group : **remitterUser**
Method : **getProfile**
This method provides the remitter user's data.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <remitter>
            <remitter_id>10001</remitter_id>
            <status>enabled</status>
            <firstname>James</firstname>
            <middlename></middlename>
            <lastname>Smith</lastname>
            <gender>Male</gender>
            <avatar>42.jpg</avatar>
            <avatar_content></avatar_content>
            <building_no>119</building_no>
            <address1>Flat 4, Rue Bleue</address1>
            <address2></address2>
            <city>Ottawa</city>
            <state></state>
            <postcode>OW878</postcode>
            <country>Canada</country>
            <country_id>02</country_id>
            <country_iso_code>CA</country_iso_code>
            <telephone>54654675</telephone>
            <mobile></mobile>
            <fax></fax>
            <email></email>
            <dob>23-07-1979</dob>
            <place_of_birth></place_of_birth>
            <country_of_birth></country_of_birth>
            <nationality>CA</nationality>
            <place_of_birth></place_of_birth>
            <country_of_birth></country_of_birth>
            <fathers_name></fathers_name>
            <mothers_name></mothers_name>
```

```
            <national_id_number></national_id_number>
            <verified>t</verified>
            <id_documents>
                <id_document>
                    <id_type>passport</id_type>
                    <id_details>31337</id_details>
                    <id_issued_by>CAPA</id_issued_by>
                    <id_issue_place>Ottawa</id_issue_place>
                    <id_start>03-07-2005</id_start>
                    <id_expiry>03-07-2015</id_expiry>
                </id_document>
                <id_document>
                    <id_type>driving_license</id_type>
                    <id_details>31337</id_details>
                    <id_issued_by>CAPA</id_issued_by>
                    <id_issue_place>Ottawa</id_issue_place>
                    <id_start>06-09-2002</id_start>
                    <id_expiry>06-09-2012</id_expiry>
                </id_document>
                <id_document>
                    <id_type>NATIONAL_INSURANCE</id_type>
                    <id_details>31337</id_details>
                    <id_issued_by>CAPA</id_issued_by>
                    <id_issue_place>Ottawa</id_issue_place>
                    <id_start>06-09-2002</id_start>
                    <id_expiry>06-09-2012</id_expiry>
                </id_document>
                <id_document>
                    <id_type>NATIONAL_ID</id_type>
                    <id_details>31337</id_details>
                    <id_issued_by>CAPA</id_issued_by>
                    <id_issue_place>Ottawa</id_issue_place>
                    <id_start>06-09-2002</id_start>
                    <id_expiry>06-09-2012</id_expiry>
                </id_document>
            </id_documents>
            <kyc_video>https://.../video.mp4</kyc_video>
            <occupation></occupation>
            <purpose></purpose>
            <source_of_income></source_of_income>
            <employer></employer>
            <business_address></business_address>
            <account_number>123456789</account_number>
            <annual_income></annual_income>
            <annual_remittance></annual_remittance>
            <type>registered</type>
            <business_type></business_type>
            <orgtype></orgtype>
            <company_name></company_name>
            <company_type></company_type>
            <company_reg_no></company_reg_no>
            <hear_about_us></hear_about_us>
            <hear_about_us_other></hear_about_us_other>
            <cpf_number></cpf_number>
            <taxpayer_reg>123456789</taxpayer_reg>
            <card_issue_date></card_issue_date>
            <groups>Default,Group2,Group3</groups>
            <creation_date>2015-06-15 12:56:38.38181+01</creation_date>
        </remitter>
    </result>
</response>
```

updateProfile

Group : **remitterUser**
Method : **updateProfile**
This method updates the details of the remitter.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
mname	Text	The middle name
nationality	Text	2 letter country ISO code, as per the output of **getRemitterUISettings**.
gender	Text	Male / Female
building_no	Text	
address1	Text	
address2	Text	
city	Text	
state	Text	
postcode	Text	
email	Text	
telephone	Text	
mobile	Text	
fax	Text	
dob	Date	YYYY-MM-DD
place_of_birth	Text	
country_of_birth	Text	2 letter country ISO code, as per the output of **getRemitterUISettings**.
fathers_name	Text	
mothers_name	Text	
national_id_number	Text	
avatar	Base64 encoded file	png, gif, jpg and jpeg

id1_type	Code	
id1_details	Text	
id1_issued_by	Text	
id1_issue_place	Text	
id1_start	Date	YYYY-MM-DD
id1_expiry	Date	YYYY-MM-DD
id1_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id2_type	Code	
id2_details	Text	
id2_issued_by	Text	
id2_issue_place	Text	
id2_start	Date	YYYY-MM-DD
id2_expiry	Date	YYYY-MM-DD
id2_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id3_type	Code	
id3_details	Text	
id3_issued_by	Text	
id3_issue_place	Text	
id3_start	Date	YYYY-MM-DD
id3_expiry	Date	YYYY-MM-DD
id3_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id4_type	Code	
id4_details	Text	
id4_issued_by	Text	
id4_issue_place	Text	
id4_start	Date	YYYY-MM-DD
id4_expiry	Date	YYYY-MM-DD
id4_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
taxpayer_reg	Text	
account_number	Text	
sort_code	Text	
remitter_username	Text	Email address used for authentication

purpose	Text	
source_of_income	Text	
employer	Text	
business_address	Text	The address of the employer
annual_income	Text	
annual_remittance	Text	
hear_about_us	Text	as per the output of **getRemitterUISettings**.
hear_about_us_other	Text	If hear_about_us is “Other” then specify, otherwise it will be saved as empty anyway

Notes: All fields that are meant to be updated must be provided on update, otherwise the existing value will persist. Similarly, if you want to keep an existing value, do not send an empty field.

Some fields may be required as per the output of **getRemitterUISettings** with a *registration_type* set to “registered”. Please note that even though you registered as “quickregistered”, updating your profile will require you to use “registered” and provide the new list of parameters. Besides, this list shown above is non exhaustive as other fields might be available as per the output of **getRemitterUISettings** too.

It has to be reminded here that the email, name and country cannot be changed.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <remitter>
            <remitter_id>10001</remitter_id>
            <status>enabled</status>
            <firstname>James</firstname>
            <middlename></middlename>
            <lastname>Smith</lastname>
            <gender>Male</gender>
            <avatar>42.jpg</avatar>
            <avatar_content></avatar_content>
            <building_no>119</building_no>
            <address1>Flat 4, Rue Bleue</address1>
            <address2></address2>
            <city>Ottawa</city>
            <state></state>
            <postcode>OW878</postcode>
            <country>Canada</country>
            <country_id>02</country_id>
```

```
<country_iso_code>CA</country_iso_code>
<telephone>54654675</telephone>
<mobile></mobile>
<fax></fax>
<email></email>
<dob>23-07-1979</dob>
<nationality>CA</nationality>
<place_of_birth></place_of_birth>
<country_of_birth></country_of_birth>
<fathers_name></fathers_name>
<mothers_name></mothers_name>
<national_id_number></national_id_number>
<verified>t</verified>
<id_documents>
    <id_document>
        <id_type>passport</id_type>
        <id_details>31337</id_details>
        <id_issued_by>CAPA</id_issued_by>
        <id_issue_place>Ottawa</id_issue_place>
        <id_start>03-07-2005</id_start>
        <id_expiry>03-07-2015</id_expiry>
    </id_document>
    <id_document>
        <id_type>driving_license</id_type>
        <id_details>31337</id_details>
        <id_issued_by>CAPA</id_issued_by>
        <id_issue_place>Ottawa</id_issue_place>
        <id_start>06-09-2002</id_start>
        <id_expiry>06-09-2012</id_expiry>
    </id_document>
    <id_document>
        <id_type>NATIONAL_INSURANCE</id_type>
        <id_details>31337</id_details>
        <id_issued_by>CAPA</id_issued_by>
        <id_issue_place>Ottawa</id_issue_place>
        <id_start>06-09-2002</id_start>
        <id_expiry>06-09-2012</id_expiry>
    </id_document>
    <id_document>
        <id_type>NATIONAL_ID</id_type>
        <id_details>31337</id_details>
        <id_issued_by>CAPA</id_issued_by>
        <id_issue_place>Ottawa</id_issue_place>
        <id_start>06-09-2002</id_start>
        <id_expiry>06-09-2012</id_expiry>
    </id_document>
</id_documents>
<occupation></occupation>
<purpose></purpose>
<source_of_income></source_of_income>
<employer></employer>
<business_address></business_address>
<business_address></business_address>
<account_number>123456789</account_number>
<annual_income></annual_income>
<annual_remittance></annual_remittance>
<type>registered</type>
<business_type></business_type>
<orgtype></orgtype>
<company_name></company_name>
<company_type></company_type>
```

```
            <company_reg_no></company_reg_no>
            <hear_about_us></hear_about_us>
            <hear_about_us_other></hear_about_us_other>
            <cpf_number></cpf_number>
            <taxpayer_reg>123456789</taxpayer_reg>
            <card_issue_date></card_issue_date>
            <groups>Default,Group2,Group3</groups>
            <creation_date>2015-06-15 12:56:38.38181+01</creation_date>
        </remitter>
    </result>
</response>
```

Notes: If any value regarding the address, date of birth or id documents is changed, the remitter may be set to unverified. As a result the remitter will not be able to perform all usual actions as long as it has not been verified back.

---

uploadProfileKYCVideo

Group : **remitterUser**
Method : **uploadProfileKYCVideo**

This method is used to upload the content of a profile KYC Video to be available in the remitter profile as proof of ID.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
filename	Text	The filename to get the extension from (at least the basename)
kyc_video	Text	Base64 encoding of the video content

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <kyc_video>https://…/video.mp4</kyc_video>
    </result>
</response>
```

---

getProfileKYCVideo

Group : **remitterUser**
Method : **getProfileKYCVideo**

This method is used to retrieve the content of the profile KYC video of the remitter.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <kyc_video>
            <data>...base64...</data>
            <mimetype>video/mp4</mimetype>
        </kyc_video>
    </result>
</response>
```

---

getSourceCountries

Group : **remitterUser**
Method : **getSourceCountries**
This method is used to retrieve a list of source countries available for the ORM.

Input fields:
None.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <countries>
            <country>
                <id>001</id>
                <name>United Kingdom</name>
                <iso_code>UK</iso_code>
            </country>
```

```
            <country>
                <id>002</id>
                <name>United States</name>
                <iso_code>US</iso_code>
            </country>
        </countries>
    </result>
</response>
```

---

register

Group : **remitterUser**
Method : **register**
This method is used to initiate the registration of the remitter via the web service. This will involve creating a temporary member/online remitter with the data provided.

There are currently 2 types of registration that are available: “quickregistered” and “registered”. The first one allows you to capture a fixed and drastically reduced number of fields to ease/speed up the initial registration process. Upon confirmation, the remitter will be able to perform a limited number of actions such as managing its beneficiaries. However, you will need to call the **updateProfile** method in order to complete the registration, if additional fields are still required, to start remittance.

Please note that the list of fields required for “quickregistered” is definitive and cannot be customised. If you want to capture more fields directly during registration, then you will need to use “registered” as the registration_type.

Input fields:

Name	Type	Notes
registration_type	Text	“quickregistered” or “registered”. By default, “registered” will be used when not provided.
fname *	Text	
lname *	Text	
email *	Text	
encrypted_data *	Text	Encrypted associative json array using the public OpenSSL key. The associative array should be: {“password”: “YYYYYY”}
nationality	Text	2 letter country ISO code, as per the output of **getRemitterUISettings**.

gender	Text	Male / Female
building_no	Text	
address1	Text	
address2	Text	
city	Text	
postcode	Text	
state	Text	
source_country_id *	Text	The id of the Source Country, as per the output of **getSourceCountries**.
telephone	Text	
mobile	Text	
fax	Text	
dob *	Date	YYYY-MM-DD
place_of_birth	Text	
country_of_birth	Text	
avatar	Base64 encoded file	png, gif, jpg and jpeg
id1_type	Code	
id1_details	Text	
id1_issued_by	Text	
id1_issue_place	Text	
id1_start	Date	YYYY-MM-DD
id1_expiry	Date	YYYY-MM-DD
id1_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
id2_type	Code	
id2_details	Text	
id2_issued_by	Text	
id2_issue_place	Text	
id2_start	Date	YYYY-MM-DD
id2_expiry	Date	YYYY-MM-DD
id2_scan	Base64 encoded file	png, gif, jpg, jpeg, bmp and pdf
taxpayer_reg	Text	
account_number	Text	
sort_code	Text	

hear_about_us	Text	as per the output of **getRemitterUISettings**.
hear_about_us_other	Text	If hear_about_us is “Other” then specify, otherwise it will be saved as empty anyway
receive_marketing	Boolean	t / f
toc *	Text	Agreement to the Terms and Conditions. Set to “true” to accept them.
introducer_agent_code	Text	The Agent ID which you wish to link the remitter to.
referral_code	Text	Alphanumeric code.

NOTE: Other fields may be required depending on the actual configuration of the system. This list is therefore non-exhaustive.
Please call **getRemitterUISettings** method to get your list.

Notes: Please refer to appendix I for details about the encryption.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <email_verification_code>true</email_verification_code>
        <sms_verification_code>false</sms_verification_code>
    </result>
</response>
```

The email and sms verification code requirement flags indicates whether or not these codes will be required when calling **confirmRegistration**. If none of them are required, then you can directly call the **confirmRegistration** method.

---

confirmRegistration

Group : **remitterUser**
Method : **confirmRegistration**
This method verifies the confirmation codes that have been eventually sent by email and/or SMS during the registration process.

Input fields:

Name	Type	Notes
email_address *	Text	The email address of the newly registered remitter.
email_verification_code	Text	The code sent by email after the call to **register**. This may be required if the output of **register** call indicates so.
sms_verification_code	Text	The code sent by SMS after the call to **register**. This may be required if the output of **register** call indicates so.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <session_token>XXXXXXXXXXXXXXXXXXXXXXXXXXXX</session_token>
        <app_pin>YYYYYY</app_pin>
        <two_factor_authentication>
            <required>false</required>
            <type>SMS</type>
            <can_resend_code>true</can_resend_code>
        </two_factor_authentication>
    </result>
</response>
```

Notes: The app pin might be provided if this is the first time that the user is logging in via the remitter Web Services; so that it can be given to the user. Otherwise this information will not be included in the response.

The two factor authentication flag indicates if a 2FA code has been sent to the remitter. If so, this information must be provided using **confirmTwoFactorAuthentication**. Otherwise, the user will not be able to perform any further action.

The two factor authentication type can be: SMS, Entrust or GoogleAuthenticator. This list may grow in the future.

If you used “quickregistered” as the registration_type in **register**, you may still be required to call **updateProfile** in order to complete the registration and allow the remitter to create transactions.

requestRegistrationConfirmationCode

Group : **remitterUser**
Method : **requestRegistrationConfirmationCode**
This method re-send the confirmation codes that were eventually sent by email and/or SMS during the registration process. This method is optional. The codes will be sent automatically after successfully calling register() anyway. This is only used if the user does not receive the first codes.

Input fields:

Name	Type	Notes
email_address *	Text	The email address of the newly registered remitter.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
</response>
```

Wallet

getWallets

Group : **wallet**
Method : **getWallets**
This method provides the balances of the remitter wallets.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <wallets>
            <wallet>
                <currency>GBP</currency>
                <credit_balance>1337.00</credit_balance>
                <enabled>true</enabled>
            </wallet>
            <wallet>
                <currency>EUR</currency>
                <credit_balance>42.00</credit_balance>
                <enabled>false</enabled>
            </wallet>
        </wallets>
    </result>
</response>
```

Notes: The enabled flag indicates if the currency is still enabled within the system. If disabled, the funds attached to it can be transferred to one of the enabled wallets of the same remitter. However, the system will not accept any transaction payment from/to a disabled wallet.

---

getWalletActivity

Group : **wallet**
Method : **getWalletActivity**
This method provides the list of events linked to the remitter wallets.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
days	Numeric	Optional. Restrict the listed activities to the past X days. If null, the default value of 3 months will be used.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <activities>
            <activity>
                <date>2015-06-16 13:46:08+01</date>
                <currency>EUR</currency>
                <credit_added>-13.37</credit_added>
                <credit_before>100.00</credit_before>
                <credit_after>86.63</credit_after>
                <updated_by>John Doe</updated_by>
                <credit_type>TP</credit_type>
                <trans_ref>RC000321000213</trans_ref>
                <deposit_ref></deposit_ref>
                <trans_amount>11.55</trans_amount>
                <original_source_currency>GBP</original_source_currency>
                <original_remitt_pay>11.55</original_remitt_pay>
                <remitter_paid>11.55</remitter_paid>
                <conversion_rate>1.15757575</conversion_rate>
                <hqok_date></hqok_date>
                <account_ref></account_ref>
                <notes>Payment due for Transaction RC000321000213</notes>
            </activity>
        </activities>
    </result>
</response>
```

Notes: See **Appendix G** for all Remitter credit types.

---

getLoadWalletCharges

Group : **wallet**
Method : **getLoadWalletCharges**
This method calculates any charges that will apply to the wallet loading as well as the final amount to pay.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets**
amount *	Numeric	The amount to load into the wallet
payment_method_id *	Numeric	The Payment Method id, as per the output of **getLoadWalletPaymentMethods**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <currency>GBP</currency>
        <min_load_amount>1.00</min_load_amount>
        <max_load_amount>50.00</max_load_amount>
        <max_limit>150.00</max_limit>
        <amount_available_to_load>20.00</amount_available_to_load>
        <charges>2.00</charges>
        <total_to_pay>15.37</total_to_pay>
        <amount>13.37</amount>
        <payment_method>6</payment_method>
        <new_balance>100.00</new_balance>
    </result>
</response>
```

---

getLoadWalletPaymentMethods

Group : **wallet**
Method : **getLoadWalletPaymentMethods**
This method provides a list of the wallet loading payment methods available.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter

session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <payment_methods>
            <payment_method>
                <id>6</id>
                <name>Credit/Debit Card</name>
            </payment_method>
            <payment_method>
                <id>8</id>
                <name>Bank Transfer</name>
            </payment_method>
        ...
        </payment_methods>
    </result>
</response>
```

---

loadWallet

Group : **wallet**
Method : **loadWallet**
This method initiates the load of the wallet.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets**
amount *	Numeric	The amount to load into the wallet
payment_method_id *	Numeric	The Payment Method id, as per the output of **getLoadWalletPaymentMethods**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <my_wallet_ref>10000</my_wallet_ref>
        <currency>GBP</currency>
        <charges>2.00</charges>
        <total_to_pay>15.37</total_to_pay>
        <amount>13.37</amount>
        <payment_method>6</payment_method>
        <status>UNPAID</status>
        <creation_date>2016-06-26 16:52:08.892531+01</creation_date>
        <new_balance>100.00</new_balance>
    </result>
</response>
```

Notes: You will need to call **loadWalletPaymentCleared** method to clear the payment of this wallet load.

---

getLoadWalletDetails

Group : **wallet**
Method : **getLoadWalletDetails**
This method retrieves the details of a wallet load in order to clear the payment for it.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
my_wallet_ref *	Text	The wallet loading reference, as per the output of **loadWallet**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <my_wallet_ref>10000</my_wallet_ref>
        <currency>GBP</currency>
        <charges>2.00</charges>
        <total_to_pay>15.37</total_to_pay>
        <amount>13.37</amount>
        <payment_method>6</payment_method>
        <status>UNPAID</status>
```

```
        <creation_date>2016-06-26 16:52:08.892531+01</creation_date>
        <new_balance>100.00</new_balance>
    </result>
</response>
```

Notes: This response is the same as the one from **loadWallet**. It is only needed to compute the wallet security hash if you have not cached the data of the latter confirmation call.

---

loadWalletPaymentCleared

Group : **wallet**
Method : **loadWalletPaymentCleared**
This method clears the payment of a wallet load when the Card Processor (ex: Secure Trading) indicates that the payment was successful.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
my_wallet_ref *	Text	The wallet loading reference, as per the output of **loadWallet**.
security_hash *	Text	The hash that should be generated to authenticate the action. RemitONE will provide the recipe during the setup.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <my_wallet_ref>10000</my_wallet_ref>
        <currency>GBP</currency>
        <charges>2.00</charges>
        <total_to_pay>15.37</total_to_pay>
        <amount>13.37</amount>
        <payment_method>6</payment_method>
        <status>SENT_FOR_DELIVERY</status>
        <creation_date>2016-06-26 16:52:08.892531+01</creation_date>
        <new_balance>100.00</new_balance>
    </result>
</response>
```

---

calculateMoveFundsBetweenWallets

Group : **wallet**
Method : **calculateMoveFundsBetweenWallets**
This method provides the rate used in the conversion and calculate the final amount that will be moved between two of the remitter's wallets.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
from_currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets**
to_currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets.** This should not be a disabled wallet currency.
amount *	Numeric	The amount to move from the source currency wallet to the destination one.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <rate>1.245454</rate>
        <received_amount>13.37</received_amount>
    </result>
</response>
```

---

moveFundsBetweenWallets

Group : **wallet**
Method : **moveFundsBetweenWallets**
This method moves funds between two of the remitter's wallets.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
from_currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets**
to_currency *	Text	The 3 letters ISO currency code, as listed in **Appendix A,** as per the output of **getWallets.** This should not be a disabled wallet currency.
amount *	Numeric	The amount to move from the source currency wallet to the destination one.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <sending_wallet>
            <currency>GBP</currency>
            <previous_balance>9705.20</previous_balance>
            <transfer_amount>40.00</transfer_amount>
            <new_balance>9665.20</new_balance>
        </sending_wallet>
        <receiving_wallet>
            <currency>EUR</currency>
            <previous_balance>64.21</previous_balance>
            <transfer_amount>52.47</transfer_amount>
            <rate>1.311757</rate>
            <new_balance>116.68</new_balance>
        </receiving_wallet>
    </result>
</response>
```

UI Settings

getCarouselImages

Group : **UISettings**
Method : **getCarouselImages**
This method lists the urls to the images for the carousel.

Input fields:

Name	Type	Notes
type *	Text	The Type of images to get. The value can be either: **mobile** or **orm**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <images>
            <image>http://remitone.com/path/to/image1.png</image>
            <image>http://remitone.com/path/to/image2.jpg</image>
            ...
        </images>
    </result>
</response>
```

---

getCountryPrefixes

Group : **UISettings**
Method : **getCountryPrefixes**
This method is used to retrieve the telephone prefix for all countries.

Input fields:

Name	Type	Notes
iso_code	Text	Optional. 2 letter country ISO code, as listed in **Appendix A.**

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<response>
    <status>SUCCESS</status>
    <result>
        <countries>
            <country>
                <iso_code>GB</iso_code>
                <name>United Kingdom</name>
                <prefix>44</prefix>
                <main_prefix>false</main_prefix>
            </country>
            <country>
                <iso_code>US</iso_code>
                <name>United States</name>
                <prefix>1</prefix>
                <main_prefix>true</main_prefix>
            </country>
            ...
        </countries>
    </result>
</response>
```

Notes: The “main_prefix” element is used when countries share the same prefix (Ex: +1 is used in the US as well as Canada). This allows the system to promote one of them by default. If you would rather promote another one, please contact RemitONE support to swap it.

---

getChangePasswordSettings

Group : **UISettings**
Method : **getChangePasswordSettings**
This method is used to retrieve password validations for change/reset password.

Input fields:

None.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <password_validation>
            <min_length>8</min_length>
            <upper>true</upper>
            <lower>true</lower>
            <digit>true</digit>
            <special>false</special>
        </password_validation>
    </result>
</response>
```

---

getBeneficiaryUISettings

Group : **UISettings**
Method : **getBeneficiaryUISettings**
This method is used to retrieve all fields that need to be displayed as well as which ones are required in order to create a new beneficiary.

Note: remitt_benef_relation_types is only shown in result if its configuration has been set as a drop-down list.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
            <response>
                <status>SUCCESS</status>
                <result>
                    <id_types>
                           <id_type>
                           <code>PASSPORT</code>
                           <name>PASSPORT</name>
                           </id_type>
                           <id_type>
                           <code>DRIVINGLICENSE</code>
                           <name>DRIVINGLICENSE</name>
                           </id_type>
                          ...
                    </id_types>
                    <card_types>
                    </card_types>
                    <remitt_benef_relation_types>
                    </remitt_benef_relation_types>
                    <elements>
                        <element>
                            <name>id1_type</name>
                            <display>true</display>
                            <required>true</required>
                        </element>
                        <element>
                            <name>email</name>
                            <display>true</display>
                            <required>false</required>
                        </element>
                        ...
                    </elements>
                </result>
            </response>
```

Notes: Some elements may contain an extra parameter “additional_attributes”

to list extra information relevant to the field. Ex**:**

```
<element>
      <name>fname</name>
      <displayed>true</displayed>
      <required>true</required>
      <additional_attributes>
            <min_length>2</min_length>
      </additional_attributes>
</element>
```

---

getRemitterUISettings

Group : **UISettings**
Method : **getRemitterUISettings**
This method is used to retrieve all fields that need to be displayed as well as which ones are required in order to register a new remitter. Additionally, the list of nationalities and “hear about us” options are included if needed.

There are currently 2 types of registration that are available: “quickregistered” and “registered”. The first one allows you to capture a fixed and drastically reduced number of fields to ease/speed up the initial registration process. Upon confirmation, the remitter will be able to perform a limited number of actions such as managing its beneficiaries. However, you will need to use “registered” in the context of **updateProfile** in order to complete the registration and start remittance.

Please note that the list of fields required for “quickregistered” is definitive and cannot be customised. If you want to capture more fields directly during registration, then you will need to use “registered” as the registration_type.

Input fields:

Name	Type	Notes
source_country_id *	Text	The id of the Source Country, as per the output of **getSourceCountries**.
registration_type	Text	“quickregistered” or “registered”. By default, “registered” will be used when not provided.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
```

```
<hear_about_us_options>
    <hear_about_us_option>Family or Friend</hear_about_us_option>
    <hear_about_us_option>Advertisement</hear_about_us_option>
    ...
</hear_about_us_options>
<transfer_purposes>
    <transfer_purpose>Family Support</transfer_purpose>
    <transfer_purpose>Gift</transfer_purpose>
    ...
</transfer_purposes>
<sources_of_income>
    <source_of_income>Salary</source_of_income>
    <source_of_income>Pension</source_of_income>
    ...
</sources_of_income>
<nationalities>
    <nationality>
        <name>AFGHANISTAN</name>
        <value>AF</value>
    </nationality>
    <nationality>
        <name>ALBANIA</name>
        <value>AL</value>
    </nationality>
    ...
</nationalities>
<password_validation>
    <min_length>8</min_length>
    <upper>true</upper>
    <lower>true</lower>
    <digit>true</digit>
    <special>false</special>
</password_validation>
<id_types>
    <id_types_1>
          <id_type>
                <code>Passport</code>
                <name>Passport</name>
          </id_type>
          <id_type>
                <code>National_Insurance</code>
                <name>National Insurance No</name>
          </id_type>
          ...
    </id_types_1>
    <id_types_2>
          <id_type>
                <code>Passport</code>
                <name>Passport</name>
          </id_type>
          <id_type>
                <code>National_ID</code>
                <name>National ID</name>
          </id_type>
          ...
    </id_types_2>
</id_types>
<elements>
    <element>
        <name>id1_type</name>
        <required>true</required>
```

```
                <display>true</display>
            </element>
            <element>
                <name>email</name>
                <required>false</required>
                <display>true</display>
            </element>
            ...
        </elements>
    </result>
</response>
```

Notes: Some elements may contain an extra parameter “additional_attributes” to list extra information relevant to the field. Ex**:**

```
<element>
      <name>dob</name>
      <displayed>true</displayed>
      <required>true</required>
      <additional_attributes>
            <minimum_age>18</minimum_age>
            <maximum_age>150</maximum_age>
      </additional_attributes>
</element>
```

---

getTransactionUISettings

Group : **UISettings**
Method : **getTransactionUISettings**
This method is used to retrieve all fields that need to be displayed as well as which ones are required in order to create a new transaction. This will also include the remittance payment methods, transfer types and other data available.

Input fields:

Name	Type	Notes
username *	Text	The username of the remitter
session_token *	Text	The Session Token provided when logging in via **login** or **loginPin**.
destination_country_id *	Text	The id of the Destination Country, as per the output of **getDestinationCountries**.
transfer_type	Text	The Transfer Type as per the output of **getTransactionUISettings**. This is optional but can be used to refine the service levels and get the corresponding transaction fields used in **createBeneficiary**/**updateBeneficiary**

		methods.
account_item_number	Number	When transfer_type is “Account”, this field can be used to specify the number (from 1 to 4 included) of the Beneficiary Account to use.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <transfer_types>
            <transfer_type>Account</transfer_type>
            <transfer_type>Cash Collection</transfer_type>
            ...
        </transfer_types>
        <payment_methods>
            <payment_method>
                <id>6</id>
                <name>Credit/Debit Card</name>
            </payment_method>
            <payment_method>
                <id>8</id>
                <name>Bank Transfer</name>
            </payment_method>
        ...
        </payment_methods>
        <source_of_incomes>
            <source_of_income>Business</source_of_income>
            <source_of_income>Gift</source_of_income>
            ...
        </source_of_incomes>
        <default_source_of_income>Business</default_source_of_income>
        <purposes>
            <purpose>Education Support</purpose>
            <purpose>Investment</purpose>
            ...
        </purposes>
        <default_purpose>Education Support</default_purpose>
        <service_levels>
            <service_level>
                <id>2</id>
                <name>nextday</name>
                <display_name>Next Day</display_name>
                <rate_percentage>0.00</rate_percentage>
                <commission_percentage>0.00</commission_percentage>
                <delivery_period>1</delivery_period>
                <transfer_type>All</transfer_type>
                <default_service_level>f</default_service_level>
            </service_level>
            ...
        </service_levels>
        <account_types>
            <account_type>
                <id>SAVINGS</id>
                <name>Savings</name>
```

```
            </account_type>
            <account_type>
                <id>CHEQUING</id>
                <name>Chequing</name>
            </account_type>
            ...
        </account_types>
        <source_currencies>
            <currency>GBP</currency>
            <currency>EUR</currency>
            ...
        </source_currencies>
        <destination_currencies>
            <currency>CNY</currency>
            <currency>USD</currency>
            ...
        </destination_currencies>
        <sms_options>
            <sms_confirmation_display>f</sms_confirmation_display>
            <sms_notification_display>t</sms_notification_display>
            <sms_mobile_display>t</sms_mobile_display>
            <sms_benef_confirmation_display>t</sms_benef_confirmation_display>
            <sms_benef_mobile_display>t</sms_benef_mobile_display>
        </sms_options>
        <elements>
            <element>
                <name>source_currency</name>
                <required>true</required>
                <display>true</display>
            </element>
            <element>
                <name>purpose</name>
                <required>false</required>
                <display>true</display>
            </element>
            ...
        </elements>
    </result>
</response>
```

Notes: Some elements may contain an extra parameter “additional_attributes” to list extra information relevant to the field. Ex**:**

```
<element>
      <name>card_number</name>
      <displayed>true</displayed>
      <required>true</required>
      <additional_attributes>
            <min_length>4</min_length>
            <max_length>20</max_length>
      </additional_attributes>
</element>
```

Support

getSupportDetails

Group : **Support**
Method : **getSupportDetails**
This method is used to retrieve the company/MTO information.

Input fields:
This is a GET Request so there are no input fields.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
      <responseId>130531</responseId>
      <status>SUCCESS</status>
      <result>
            <company_name>abc</company_name>
            <company_tel>1234567</company_tel>
            <company_fax>1234567</company_fax>
<customer_service_email>customerservice@company.com</customer_service_email>
            <company_address>anywhere, anywhere</company_address>
</result>
</response>
```

---

sendEmailToSupport

Group : **Support**
Method : **sendEmailToSupport**
This method is used to send an email to the company customer service email. This method can be used to make a contact form.

Input fields:

Name	Type	Notes
name *	Text	The name of the person writing the message/email
email *	Text	The email of the person writing the message/email
reason *	Text	The subject of the message/email
Message *	Text	The contents of the message/email

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
```

```
      <responseId>130531</responseId>
      <status>SUCCESS</status>
</response>
```

4.Testing Web Services Functionality

It is possible to test the output of the WebServices using the Remitter WS SDK.

It is available in PHP and allows you to speed up the integration as you can import it directly into your project and use all the listed methods effortlessly. It will handle the encryption tasks as well as request forging. Ex:

```
$remitterWS = new RemitterWS;
$remitterWS->auth->setUsername('john@company.com');

// Retrieve a session token if the user has no active session
$token = $remitterWS->auth->login('password');

// Alternatively, you can set the session token directly if you have already
stored it
$remitterWS->auth->setSessionToken('1234567890');

// You can then retrieve all the user's transactions as an array
$transactions = $remitterWS->transaction->listTransactions();
```

Please contact RemitONE in order to get the download instructions.

Using Web Services in a Live Environment: Some suggestions

Although it is possible to use the WebServices in any way that seems fit, we would suggest the following procedure.

1. Obtain a list of the fields necessary to create Beneficiaries using **getBeneficiaryUISettings**.
2. Create a new beneficiary using **createBeneficiary**, or search for an existing one using **listBeneficiary**.
3. Create a transaction between the remitter and this beneficiary by calling **updatebeneficiary** with the transaction related details, then call **createTransaction** after using **getTransactionUISettings** as well as other methods such as **getDeliveryBanks, getCharges** etc...
4. Receive a transaction creation response containing the actual charges as calculated by the system, compliance check results, and a temporary transaction session id. It is the responsibility of the user to examine these results.
5. Once the user is satisfied with the results, call **confirmTransaction** and provide eventual confirmation codes to actually create the transaction in the system.
6. At some later time, **getTransaction** can be used to determine whether the transaction has been processed.

Appendix A : Destination Country ISO Codes

Name	ISO code	Currency
Afghanistan	AF	AFA
Albania	AL	ALL
Algeria	DZ	DZD
American Samoa	AS	
Andorra	AD	ADP
Angola	AO	AON
Anguilla	AI	
Antarctica	AQ	
Antigua and Barbuda	AG	
Argentina	AR	ARS
Armenia	AM	AMD
Aruba	AW	AWG
Australia	AU	AUD
Austria	AT	EUR
Azerbaijan	AZ	AZN
Bahamas	BS	BSD
Bahrain	BH	BHD
Bangladesh	BD	BDT
Barbados	BB	BBD
Belarus	BY	BYR
Belgium	BE	EUR
Belize	BZ	BZD
Benin	BJ	
Bermuda	BM	BMD
Bhutan	BT	BTN
Bolivia	BO	BOB
Bosnia and Herzegovina	BA	BAM
Botswana	BW	BWP
Bouvet Island	BV	
Brazil	BR	BRE
British Indian Ocean Territory	IO	
Brunei Darussalam	BN	BND
Bulgaria	BG	BGL

Burkina Faso	BF	
Burundi	BI	BIF
Cambodia	KH	KHR
Cameroon	CM	XAF
Canada	CA	CAD
Cape Verde	CV	CVE
Cayman Islands	KY	KYD
Central African Republic	CF	XAF
Chad	TD	XAF
Chile	CL	CLF
China	CN	CNY
Christmas Island	CX	
Cocos (Keeling) Islands	CC	
Colombia	CO	COP
Comoros	KM	
Congo	CG	
Congo, the Democratic Republic of the	CD	
Cook Islands	CK	
Costa Rica	CR	CRC
Cote D_Ivoire	CI	
Croatia	HR	HRK
Cuba	CU	CUP
Cyprus	CY	CYP
Czech Republic	CZ	CZK
Denmark	DK	DKK
Djibouti	DJ	DJF
Dominica	DM	
Dominican Republic	DO	DOP
Ecuador	EC	ECS
Egypt	EG	EGP
El Salvador	SV	SVC
Equatorial Guinea	GQ	
Eritrea	ER	
Estonia	EE	EEK
Ethiopia	ET	ETB
Falkland Islands (Malvinas)	FK	
Faroe Islands	FO	
Fiji	FJ	FJD

Finland	FI	EUR
France	FR	EUR
French Guiana	GF	
French Polynesia	PF	
French Southern Territories	TF	
Gabon	GA	
Gambia	GM	GMD
Georgia	GE	
Germany	DE	EUR
Ghana	GH	GHC
Gibraltar	GI	GIP
Greece	GR	GRD
Greenland	GL	
Grenada	GD	
Guadeloupe	GP	
Guam	GU	
Guatemala	GT	GTQ
Guinea	GN	GNF
Guinea-Bissau	GW	GWP
Guyana	GY	GYD
Haiti	HT	HTG
Heard Island and Mcdonald Islands	HM	
Holy See (Vatican City State)	VA	
Honduras	HN	HNL
Hong Kong	HK	HKD
Hungary	HU	HUF
Iceland	IS	ISK
India	IN	INR
Indonesia	ID	IDR
Iran, Islamic Republic of	IR	
Iraq	IQ	IQD
Ireland	IE	EUR
Israel	IL	ILS
Italy	IT	EUR
Jamaica	JM	JMD
Japan	JP	JPY
Jordan	JO	JOD

Kazakhstan	KZ	
Kenya	KE	KES
Kiribati	KI	
Korea, Democratic People's Republic of	KP	
Korea, Republic of	KR	
Kuwait	KW	KWD
Kyrgyzstan	KG	
Lao People's Democratic Republic	LA	
Latvia	LV	
Lebanon	LB	LBP
Lesotho	LS	LSL
Liberia	LR	LRD
Libyan Arab Jamahiriya	LY	LYD
Liechtenstein	LI	
Lithuania	LT	
Luxembourg	LU	EUR
Macao	MO	
Macedonia, the Former Yugoslav Republic of	MK	
Madagascar	MG	MGF
Malawi	MW	MWK
Malaysia	MY	MYR
Maldives	MV	
Mali	ML	
Malta	MT	MTL
Marshall Islands	MH	
Martinique	MQ	
Mauritania	MR	MRO
Mauritius	MU	MUR
Mayotte	YT	
Mexico	MX	MXP
Micronesia, Federated States of	FM	
Moldova, Republic of	MD	
Monaco	MC	
Mongolia	MN	MNT
Montserrat	MS	
Morocco	MA	MAD
Mozambique	MZ	MZN
Myanmar	MM	

Namibia	NA	
Nauru	NR	
Nepal	NP	NPR
Netherlands	NL	EUR
Netherlands Antilles	AN	ANG
New Caledonia	NC	
New Zealand	NZ	NZD
Nicaragua	NI	NIO
Niger	NE	
Nigeria	NG	NGN
Niue	NU	
Norfolk Island	NF	
Northern Mariana Islands	MP	
Norway	NO	NOK
Oman	OM	OMR
Pakistan	PK	PKR
Palau	PW	
Palestinian Territory, Occupied	PS	
Panama	PA	PAB
Papua New Guinea	PG	PGK
Paraguay	PY	PYG
Peru	PE	PEN
Philippines	PH	PHP
Pitcairn	PN	
Poland	PL	PLZ
Portugal	PT	EUR
Puerto Rico	PR	
Qatar	QA	QAR
Reunion	RE	
Romania	RO	ROL
Russian Federation	RU	
Rwanda	RW	RWF
Saint Helena	SH	
Saint Kitts and Nevis	KN	
Saint Lucia	LC	
Saint Pierre and Miquelon	PM	
Saint Vincent and the Grenadines	VC	

Samoa	WS	WST
San Marino	SM	
Sao Tome and Principe	ST	
Saudi Arabia	SA	SAR
Senegal	SN	
Serbia and Montenegro	CS	
Seychelles	SC	SCR
Sierra Leone	SL	SLL
Singapore	SG	SGD
Slovakia	SK	
Slovenia	SI	
Solomon Islands	SB	SBD
Somalia	SO	SOS
South Africa	ZA	ZAR
South Georgia and the South Sandwich Islands	GS	
Spain	ES	EUR
Sri Lanka	LK	LKR
Sudan	SD	SDP
Suriname	SR	
Svalbard and Jan Mayen	SJ	
Swaziland	SZ	SZL
Sweden	SE	SEK
Switzerland	CH	CHF
Syrian Arab Republic	SY	
Taiwan, Province of China	TW	
Tajikistan	TJ	
Tanzania, United Republic of	TZ	
Thailand	TH	THB
Timor-Leste	TL	
Togo	TG	
Tokelau	TK	
Tonga	TO	TOP
Trinidad and Tobago	TT	TTD
Tunisia	TN	TND
Turkey	TR	TRL
Turkmenistan	TM	
Turks and Caicos Islands	TC	
Tuvalu	TV	

Uganda	UG	UGX
Ukraine	UA	
United Arab Emirates	AE	AED
United Kingdom	GB	GBP
United States	US	USD
United States Minor Outlying Islands	UM	
Uruguay	UY	UYP
Uzbekistan	UZ	
Vanuatu	VU	VUV
Venezuela	VE	VEB
Vietnam	VN	
Virgin Islands, British	VG	
Virgin Islands, U.s.	VI	
Wallis and Futuna	WF	
Western Sahara	EH	
Yemen	YE	YER
Zambia	ZM	ZMK
Zimbabwe	ZW	ZWD

Appendix B : Payment Method Codes

Code	Description
1	Cash
2	Cheque
3	Bank Transfer

Note: These codes may vary – please confirm with your Administrator

Appendix C : Service Level (Delivery Speed) Codes

Code	Description
1	5 Days
2	Next Day
3	Immediate

Note: These codes may vary – please confirm with your Administrator

Appendix D : Source of Income Codes

Code	Description
1	Salary
2	Business
3	Return on Investment
4	Gift
5	Pension

Note: These codes may vary – please confirm with your Administrator

Appendix E : Remittance Purpose Codes

Code	Description
1	Family Support

2	Medical Support
3	Education Support
4	Investment
5	Charity

Appendix F : Transaction Statuses

Status	Description
PENDING_CLEARANCE	Transaction has been created in the system but the remitter has not paid for transaction. Awaiting payment before transaction will be ready for processing.
ENTERED	Transaction awaiting approval by sending agent before processing.
AGENT_OK	Transaction awaiting HQ approval or routing before processing.
HQ_READY	Transaction has been approved by HQ and will be sent to processing bank/agent in the next batch. This only applies if you are using batch processing.
HQ_OK	Transaction awaiting processing bank/agent approval before ready for processing.
HQ_OK_PAID	Transaction awaiting processing bank/agent approval before ready for processing but transaction has been credited to processing bank/agent already. This status only applies in error correction scenarios.
SENT_FOR_PAY	Transaction ready to be paid out by processing bank/agent.
SENT_FOR_DELIVERY	Transaction ready to be paid out by processing bank/agent branch.
PROCESSED	Transaction has been processed.
ABORTED	Transaction has been accepted by processing bank/agent but now it has been requested to be cancelled. Awaiting processing bank/agent approval for cancellation or rejection of cancellation request.
ERROR	Transaction has some problem with it and is going through error correction cycle.
DELETED	Transaction has been cancelled.

Appendix G : Remitter Credit Types

Type	Description
TP	Transaction payment

ADD	Wallet loading
REDUCE	Funds transferred from this wallet into another wallet of the same remitter.

Appendix H : Transaction Payment Instruction Types

Type	Description
card	This will require an integration with a payment gateway in order to send the credit/debit card details. Generally, a new screen will be needed to capture all these details.
bank_transfer	The details of the bank account are included in the response and can be displayed as is in the payment details section.
cheque	The details of the cheque to write are included in the response and can be displayed as is in the payment details section.
wallet	This will leverage the wallet API methods in order to list the available wallets and use the available funds to pay. Generally, this is achieved directly in the payment details section.
phone	The details of the phone number to call are included in the response and can be displayed as is in the payment details section.
custom	This payment is configurable by the Clients.

Appendix I : Encryption

Within this document, you have encountered a parameter called “encrypted_data”. The value of this parameter should follow a set of instructions in order to get it right.

RemitONE suspport should have already provided you a public OpenSSL key. This is a PKCS8 PEM formatted 2048 RSA public key encoded in UTF-8 and using a RSA/ECB/PKCS1Padding Cipher.

These key pairs have been generated using the following commands on Linux:

```
openssl genrsa  -out server_privatekey.tmp 2048
openssl pkcs8 -topk8 -inform pem -in server_privatekey.tmp -outform pem -nocrypt -out server_privatekey
openssl rsa -pubout -in server_privatekey.tmp -out server_publickey
rm -f server_privatekey.tmp
```

The public key to use here will be “server_publickey”.

Here is the complete process:

- Create a JSON string containing the password (and eventually the seed).
- Encrypt this string using the public key.
- Base64 encode the encrypted result to get a final string.
- Use that string in your call as the encrypted_data value.

PHP-to-Java and Java-to-PHP encryption/decryption examples can be provided if needed. Please contact Support should you need it.

The Seed needs to be requested by calling **auth/getSeed** method. It is then used within an associative json array that will be encrypted. The encrypted_data associative array should be of this form:

Ex: {“seed”: “XXXXXXXXXXXXXXXXXXXXXX”, “password”: “YYYYYYYYYYY”}

The result should be encoded using base64 encoding.