Yo! payments

API Specification
v3.21

COPYRIGHT NOTICE & DISCLAIMERS

Yo! Payments: API Specification

Changes may be made periodically to the information in this publication. Such changes will be incorporated in new editions of the specification. The service described in this document is made available under a license agreement, and may be used only in accordance with the terms thereof. It is against the law to use the service except as specifically provided in the license agreement. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopied, recorded or otherwise, without the prior written permission of Yo Uganda Limited.

The service license is hereby incorporated herein by this reference.

All product names mentioned in this manual are for identification purposes only, and are either trademarks or registered trademarks of their respective owners.

TABLE OF CONTENTS

1 DOCUMENT HISTORY

Revision Date	Comments	Reviewer
January 2011 – Date	Miscellaneous Revisions	Miscellaneous Reviewers
July 2013	Added status error codes -39 and -40 related to PULL and PUSH deposit transactions. Advanced document to version 3.13	Grace
September 2013	Add InstantNotificationUrl for pull deposit transactions. Advanced document to version 3.14	Eric
September 2013	Extend `InstantNotificationUrl` documentation for pull deposit transactions. Advanced document to version 3.15	Eric
November 2013	Added details in IPN section for triggering SMS response. Advanced document to version 3.16	Gerald
January 2014	Added Transaction Failure Notification API Advanced document to version 3.17	Eric
February 2014	Added general response XML parameter `ErrorMessageCode` for unsuccessful requests. Advanced document to version 3.18	Eric
February 2014	Added general response XML parameter `MNOTransactionReferenceId` for successful requests. Advanced document to version 3.19	Eric
April 2014	Deprecated the Push Deposit Method in favor of the Instant Payment Notification API Advanced document to version 3.20	Gerald
May 2014	Added `PrivateTransactionReference` parameter to `actransactioncheckstatus` API. Advanced document to version 3.21	Johnson

2 INTRODUCTION

2.1 About Yo! Payments

Yo! Payments is a revolutionary mobile payments gateway service. Yo! Payments enables businesses to receive payments from their customers via mobile money, as well as make mobile money payments to any mobile money account holder. Yo! Payments also has the capability to send mobile calling credit (“airtime”) directly to users. Yo! Payments offers a rich API which enables seamless integration with websites, IVR services, SMS services and any other medium through which businesses interact with their customers. Yo! Payments also offers an “internal transfer” service which enables account holders to cheaply transfer funds amongst each other.

Yo! Payments essentially opens the door for all types of businesses to benefit from the highly successful mobile money transfer phenomenon.

2.2 Yo! Payments API

Yo! Payments offers an Application Programming Interface (API) for businesses who wish to customize their customers’ payment experience. The API is only available for Business Account holders.

2.3 Supported Operations

The API provides support for both transaction-oriented operations and non transaction-oriented operations. A transaction-oriented operation is an operation intended to result in the transfer of funds from one account to another account.

2.3.1 Transaction-oriented Operations

The API supports the following transaction-oriented operations. Note that these operations are also available from the Web Interface.

- Withdraw Funds
- Deposit Funds
- Internal Transfer
- Airtime Transfer

2.3.2 Non Transaction-oriented Operations

Following are the non transaction-oriented operations supported by the API:

- Check Transaction Status
- Check Account Balance
- Get Mini-Statement

All the above operations are fully described in the later sections of this document.

3 USING THE API

3.1 PREREQUISITES

To use the API, you must, first of all, have a Yo! Payments Business Account[1]. The API is not available for Personal Accounts.

Once you have a Business Account, you must obtain an **API Username** and **API Password**. You may obtain the API Username and API Password from the web interface of your Payment Account. The API Username and API Password are used to map all your API requests to your Payment Account, and for security purposes.

You may change your API Username and API Password at any time from the web interface using the “Regenerate API Credentials” feature in your web account.

3.2 REQUEST FORMAT

3.2.1 HTTP Headers

All requests must be submitted in plain XML. Therefore, the following HTTP headers are **required** in all XML requests. If these headers are not present, the request may fail.

```
Content-Type: text/xml
Content-transfer-encoding: text
```

3.2.2 General XML Format

Following is the general format of the XML request. The parameters in the request below are required for all API requests.

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method></Method>
      [ other parameters ... ]
   </Request>
</AutoCreate>
```

The table below describes the parameters above:

Parameter Name	**Type**	**Opt**	**Description**
`APIUsername`	String	Mandatory	This is the API Username which, together with the API Password below, maps your API request to your Yo! Payments account. Obtain this parameter from the web interface. Note that if you do not have a Business Account, you cannot use the API.

[1] When using the Sandbox server (see section 15, a Business Account is created for you automatically).

APIPassword	String	Mandatory	This is the API Password which, together with the API Username above, maps your API request to your Yo! Payments account. Obtain this parameter from the web interface. Note that if you do not have a Business Account, you cannot use the API.
Method	String	Mandatory	This parameter identifies the type of request you are making.

3.3 General Response Format for a Successful Request

Below is a sample of the general response format.

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>OK</Status>
      <StatusCode>0</StatusCode>
      <TransactionStatus>SUCCEEDED</TransactionStatus>
      <TransactionReference></TransactionReference>
      <MNOTransactionReferenceId></MNOTransactionReferenceId>
      [ ... other fields here ... ]
   </Response>
</AutoCreate>
```

The table below provides a description of the fields.

Parameter Name	Type	Presence	Description
Status	String	Always Present	If there is no error, this is set to “OK”.
StatusCode	Integer	Always Present	This field is set to 0 (zero)
TransactionStatus	String	Present only for transaction-oriented API requests.	This field is present for transaction-oriented API requests such as withdrawals, deposits and internal transfers – in this case this field is set to SUCCEEDED.
TransactionReference	String	Present only for transaction-oriented API requests.	This field is present for transaction-oriented API requests such as withdrawals, deposits and internal transfers – in this case this field contains a value which uniquely identifies this transaction in your Yo! Payments account.
MNOTransactionReferenceId	String	Present only for transaction-oriented API requests.	Mobile Network Operator (MNO) Transaction Reference Identifier. This field is present for transaction-oriented withdrawals and deposits requests

			only. The value uniquely identifies this transaction on the relevant Mobile Network Operator's system.

3.4 General Response Format for an Unsuccessful Request

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>ERROR</Status>
      <StatusCode></StatusCode>
      <StatusMessage></StatusMessage>
      <ErrorMessageCode></ErrorMessageCode>
      <ErrorMessage></ErrorMessage>
      <TransactionStatus></TransactionStatus>
      <TransactionReference></TransactionReference>
   </Response>
</AutoCreate>
```

The table below describes the fields above:

Parameter Name	Type	Presence	Description
`Status`	String	Always Present	If there is an error, this parameter is set to "`ERROR`". In this case, check the `Message` parameter for a description of the error which occurred.
`StatusCode`	Integer	Always Present	This is an integral value which uniquely identifies the status of the transaction. To obtain the possible values of this field and their respective descriptions, refer to **Section 11: API Status Codes**.
`StatusMessage`	String	Always Present	Textual description of the status code above.
`ErrorMessageCode`	Integer	Present if `ErrorMessage` is available.	This field may or may not be present in the response XML. The field will only be present and set if `ErrorMessage` is present and set. If present this is set to an integral value which uniquely identifies the message in `ErrorMessage`. The error message code is useful in resolving transaction errors.
`ErrorMessage`	String	Present if additional information is available.	This field may or may not be present in the response XML. If present, it provides you with additional information about any error which may have occurred. The information contained in this field is highly useful in resolving any internal errors with the Yo! Payments service. If this information is provided in the response

			XML, provide it to your Yo! Payments contact, should you opt to submit an error report.
`TransactionStatus`	String	Present only for transaction-oriented API requests.	This field is present for transaction-oriented API requests such as withdrawals, deposits and internal transfers. This field may take any of two possible values namely `FAILED` and `INDETERMINATE`. Below is the meaning of the respective values: • **FAILED**. This means that your request was not successful. You may re-submit your request for processing if there was an error on your part. • **INDETERMINATE**. This means that your request is pending resolution by the Yo! Payments team. This normally happens if there was a delay in processing of mobile money transactions. Typicially requests which result in this status are resolved within 24 hours of your initiating the request. Contact your Yo! Payments representative for earlier resolution.
`TransactionReference`	String	Sometimes present but only for transaction-oriented API requests.	This value may or may not be present. If present, this value uniquely identifies your transaction in your Yo! Payments account.

3.5 General Response Format for a Pending Request

Below is a sample of the general response format for a PENDING request. All XML responses from Yo! Payments shall have the fields in the sample XML below, if the status is PENDING. A PENDING transaction is one which has not yet been processed to a conclusion. Therefore, this is a request which is neither successful nor unsuccessful but is awaiting execution. Note that not all transaction types can enter into the PENDING state. Review the specific transaction documentation to determine whether the PENDING state is valid for the transaction.

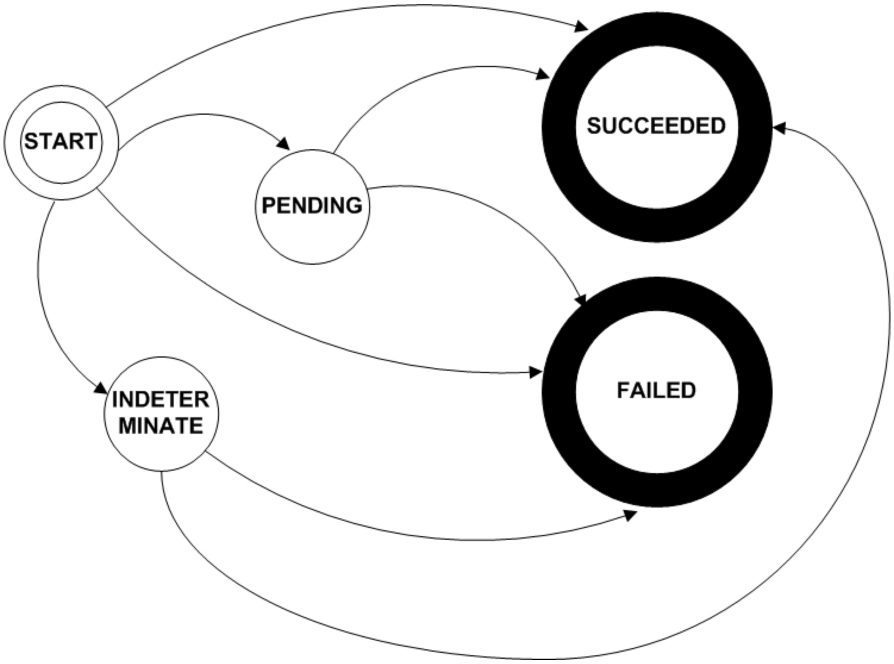
```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>OK</Status>
      <StatusCode>1</StatusCode>
      <TransactionStatus>PENDING</TransactionStatus>
      <TransactionReference></TransactionReference>
      [ ... other fields here ... ]
   </Response>
</AutoCreate>
```

The table below provides a description of the fields.

Parameter Name	Type	Description
`Status`	String	If there is no error, this is set to "`OK`".
`StatusCode`	Integer	This field is set to 1 (one)
`TransactionStatus`	String	This field is present for transaction-oriented API requests such as withdrawals, deposits and internal transfers – in this case this field is set to `PENDING`.
`TransactionReference`	String	This field is present for transaction-oriented API requests such as withdrawals, deposits and internal transfers – in this case this field contains a value which uniquely identifies this transaction in your Yo! Payments account.

3.6 State Diagram for Transaction Statuses

The diagram below attempts to provide a visual explanation of the transaction statuses supported in Yo! Payments.



3.7 Request Submission URL

All requests to the API system must be submitted to any of the following URLs:

`https://paymentsapi1.yo.co.ug/ybs/task.php`

`https://paymentsapi2.yo.co.ug/ybs/task.php`

4 WITHDRAW FUNDS TRANSACTION

This transaction enables you to withdraw funds from your account and deposit the funds into the account of a mobile money user. You shall be required to provide the telephone number of the mobile money user in order to complete this transaction. Note that this transaction will only succeed for supported mobile money network providers. To get a current list of supported mobile money network providers, visit our website. An example of where this transaction is useful is when making a payment to a mobile money user, or cashing out your balance. If you are cashing out your balance then you shall need to perform two steps namely: (a) Perform this transaction to transfer the money to a mobile money account; (b) Obtain the cash from the mobile money provider.

4.1 REQUEST FORMAT

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acwithdrawfunds</Method>
      <NonBlocking></NonBlocking>
      <Amount></Amount>
      <Account></Account>
      <AccountProviderCode></AccountProviderCode>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
      <ProviderReferenceText></ProviderReferenceText>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acwithdrawfunds`.
`NonBlocking`	String	Optional	If set to “TRUE”, you will follow up on the status of the transaction as described in section 0. If this parameter is empty or set to “FALSE” by default your connection to the system is maintained until your request is fulfilled.
`Amount`	Float	Mandatory	This is the amount to be withdrawn. Must be set to a value greater than zero. Fractional amounts may not be

			supported by certain mobile money providers.
`Account`	Numeric	Mandatory	This is a numerical value representing the account number of the mobile money account where you wish to transfer the funds to. This is typically the telephone number of the mobile phone receiving the amount. Telephone numbers MUST have the international code prepended, without the “+” sign. An example of a mobile money account number which would be valid for the MTN Uganda network is `256771234567`.
`AccountProviderCode`	String	Optional	Provide here the account provider code of the institution holding the account indicated in the `Account` parameter. See section 12 for a list of all supported account provider codes.
`Narrative`	String	Mandatory	Textual narrative about the transaction. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the `NarrativeFileBase64` parameter (see below).
`NarrativeFileName`	FileNameString[2]	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example “receipt.doc” or “receipt.pdf”. Note that the contents of this parameter are ignored if you have not provided the contents of the file using `NarrativeFileBase64` below.
`NarrativeFileBase64`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for

[2] A FileNameString is a case-insensitive string comprising of one or more of the following valid characters: {a through z, 0 through 9, period character, underscore character, whitespace, hyphen}. The FileNameString is validated with the following regular expression: `^[a-zA-Z0-9\._\s-]+$`

			example, in the case where you may want to attached a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the benecifiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.
`ProviderReferenceText`	String	Optional	In this field, enter text you wish to be present in any confirmation message which the mobile money provider network sends to the subscriber upon successful completion of the transaction. Some mobile money providers automatically send a confirmatory text message to the subscriber upon completion of transactions. This parameter allows you to provide some text which will be appended to any such confirmatory message sent to the subscriber.

4.2 Success Response Format

If your request is successful, you shall get a response in the format described in section 3.3 above.

4.3 Error Response Format

If your request is not successful, you shall get a response in the format described in section 3.4 above.

4.4 Pending Response Format

If your request is pending processing, for example in the case where you set `NonBlocking` to `TRUE`, then you shall get a response in the format described in section 3.5 above.

5 DEPOSIT FUNDS TRANSACTION

This transaction enables you to deposit funds into your account by transferring the said funds from a mobile money account holder. An example of where this transaction is useful is when receiving payment from a mobile money user for services you are rendering to them.

There are two ways in which this transaction is implemented namely:

- “Pull Method”
- “Push Method”

5.1 PULL METHOD

With the Pull Method, shortly after you submit your request, the mobile money user receives an on-screen notification on their mobile phone. This notification informs the mobile money user about your request to transfer funds out of their account, and requests them to authorize the request to complete the transaction. The transaction will not succeed unless the mobile money user authorizes it through the on-screen notification.

Note that this type of request is not supported by all mobile money operator networks. If unsure, consider the alternative push method documented in section 5.2 or refer to our website for more information.

5.1.1 Request Format

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acdepositfunds</Method>
      <NonBlocking></NonBlocking>
      <Amount></Amount>
      <Account></Account>
      <AccountProviderCode></AccountProviderCode>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
      <ProviderReferenceText></ProviderReferenceText>
      <InstantNotificationUrl></InstantNotificationUrl>
      <FailureNotificationUrl></FailureNotificationUrl>
      <AuthenticationSignatureBase64></AuthenticationSignatureBase64>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandat	Must be set to the string `acdepositfunds`.

		ory	
NonBlocking	String	Optional	If set to “TRUE”, you will follow up on the status of the transaction as described in section 0. If this parameter is empty or set to “FALSE” by default your connection to the system is maintained until your request is fulfilled.
Amount	Float	Mandatory	This is the amount to be deducted from the mobile money account and deposited into your Payment Account. Must be set to a value greater than zero. Fractional amounts may not be supported by certain mobile money providers.
Account	Numeric	Mandatory	This is a numerical value representing the account number of the mobile money account from where you are deducting the funds to be deposited into your Payment Account. This is typically the telephone number of the mobile phone sending the amount. Telephone numbers MUST have the international code pre-pended, without the “+” sign. An example of a mobile money account number which would be valid for the MTN Uganda network is 256771234567.
AccountProviderCode	String	Optional	Provide here the account provider code of the institution holding the account indicated in the Account parameter. See section 12 for a list of all supported account provider codes.
Narrative	String	Mandatory	Textual narrative about the transaction. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the NarrativeFileBase64 parameter (see below).
NarrativeFileName	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example “receipt.doc” or “receipt.pdf”. Note that the contents of this parameter are ignored if you have not provided the contents of the file using NarrativeFileBase64 below.
NarrativeFileBase64	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attached a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have

			not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the beneficiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.
`ProviderReferenceText`	String	Optional	In this field, enter text you wish to be present in any confirmation message which the mobile money provider network sends to the subscriber upon successful completion of the transaction. Some mobile money providers automatically send a confirmatory text message to the subscriber upon completion of transactions. This parameter allows you to provide some text which will be appended to any such confirmatory message sent to the subscriber.

`InstantNotificationUrl`	String	Optional	In this field provide a valid URL if you wish to be notified as soon as funds are successfully deposited into your account. The URL is where the payment notification will be sent. Please refer to the "Instant Payment Notification API" section of the Yo Payments API documentation for details on how to process the notification.
			Note:
			Any variables in the URL query string will be sent as HTTP POST variables in the IPN request.
			Notification URL format:
			Your URL must be properly URL encoded, for example:
			`http://ipnurl?key1=This+value+has+encoded+white+spaces&key2=value`
			In addition any special XML characters must be escaped or your request will fail. For example in the above URL, the ampersand character is replaced by the appropriate XML escape sequence like so:
			`http://ipnurl?key1=This+value+has+encoded+white+spaces&amp;key2=value`

`FailureNotificationUrl`	String	Optional	In this field provide a valid URL if you wish to be automatically notified when and if your deposit request fails.
			The URL is where the failure notification will be sent. Please refer to section 5.4 for more information on the Transaction Failure Notification API.
			URL format:
			Your URL must be properly URL encoded, for example:
			`http://failureurl?key1=This+value+has+encoded+white+spaces&key2=value`
			Note: Any variables in the URL query string will be sent as HTTP POST body variables in the failure notification request.
			In the above example the variables in the URL query string are sent as part of the POST body to `http://failureurl`.
			Any special XML characters must be escaped or your request will fail.
			For example in the above URL, the ampersand character is replaced by the appropriate XML escape sequence like so:
			`http://failureurl?key1= This+value+has+encoded+white+spaces &amp;key2=value`

`AuthenticationSignatureBase64`	String	Optional	This field may be required to authenticate certain deposit requests. Please contact our support services for clarification on the cases where this parameter is required. When this field is used, set it to a base-64 encoded digital signature for the particular request. To generate the signature concatenate the following parameters in order: 1. `APIUsername` 2. `APIPassword` 3. `Amount` 4. `Account` 5. `Narrative` 6. `ExternalReference` 7. the source IP address where the request originates Next, obtain an SHA1 hash of the above string. Then, encrypt the SHA1 hash you have obtained using your

			private encryption key. Finally, obtain the base-64 representation of the above encrypted data and store it in the `AuthenticationSignatureBase64` field. Ensure that you have shared with us your public key to be able to verify this signature.

5.1.2 Success Response Format

If your request is successful, you shall get a response in the format described in section 3.3 above.

5.1.3 Error Response Format

If your request is not successful, you shall get a response in the format described in section 3.4 above.

5.1.4 Pending Response Format

If your request is pending processing, for example in the case where you set `NonBlocking` to `TRUE`, then you shall get a response in the format described in section 3.5 above.

5.2 PUSH METHOD (DEPRECATED)

IMPORTANT: *This API Method is Deprecated and should not be used for new implementations – it will soon be discontinued. This functionality has been superseded by the Instant Payment Notification API in Section 5.3*

With the Push Method, after submitting your request, you are required to instruct the mobile money user to manually transfer the funds to a designated mobile money account. After the mobile money user has transferred the funds, the transaction will be automatically completed and will reflect on your account statement.

The key difference between the Push Method and Pull Method is that with the Push Method, the mobile money user takes the initiative to transfer funds from their account into the designated account, whereas with the Pull Method, the mobile money user only needs to authorize the transaction after they receive the on-screen notification on their mobile phone.

Note that you must instruct the mobile money user to transfer the funds within 10 minutes of your sending the push request. If they do not transfer the funds within this window, the transaction will be aborted, and if any funds are received after this window, they will be refunded within at most 1 hour of their sending the funds.

The Push Method is supported by virtually all mobile money network operators.

5.2.1 Request Format

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method> acrequestfundspush</Method>
      <Amount></Amount>
      <Account></Account>
      <AccountProviderCode></AccountProviderCode>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
      <ProviderReferenceText></ProviderReferenceText>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acrequestfundspush`.

`Amount`	Float	Mandatory	This is the amount to be deducted from the mobile money account and deposited into your Payment Account. Must be set to a value greater than zero. Fractional amounts may not be supported by certain mobile money providers.
`Account`	Numeric	Mandatory	This is a numerical value representing the account number of the mobile money account from where funds will be pushed into your account. This is typically the telephone number of the mobile phone sending the amount. Telephone numbers MUST have the international code pre-pended, without the “+” sign. An example of a mobile money account number which would be valid for the MTN Uganda network is `256771234567`.
`AccountProviderCode`	String	Optional	Provide here the account provider code of the institution holding the account indicated in the `Account` parameter. See section 12 for a list of all supported account provider codes.
`Narrative`	String	Mandatory	Textual narrative about the transaction. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the `NarrativeFileBase64` parameter (see below).
`NarrativeFileName`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example “receipt.doc” or “receipt.pdf”. Note that the contents of this parameter are ignored if you have not provided the contents of the file using `NarrativeFileBase64` below.
`NarrativeFileBase64`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want

			to attach a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the beneficiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.
`ProviderReferenceText`	String	Optional	In this field, enter text you wish to be present in any confirmation message which the mobile money provider network sends to the subscriber upon successful completion of the transaction. Some mobile money providers automatically send a confirmatory text message to the subscriber upon completion of transactions. This parameter allows you to provide some text which will be appended to any such confirmatory message sent to the subscriber.

5.2.2 Pending Response Format

Pending receipt of funds into the system you shall get a response in the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>OK</Status>
      <StatusCode>1</StatusCode>
      <TransactionStatus>PENDING</TransactionStatus>
      <TransactionReference></TransactionReference>
      <ReceivingPushAccount></ReceivingPushAccount>
      <SamplePushInstructions></SamplePushInstructions>
      <RequestExpirySeconds></RequestExpirySeconds>
      <RequestMandatoryReference></RequestMandatoryReference>
   </Response>
</AutoCreate>
```

The table below describes the parameters which are unique to this response. For a description of the standard parameters for pending responses, refer to section 3.5.

Parameter Name	Type	Description
`ReceivingPushAccount`	Numeric	This is the designated mobile money account code, or number to which the sender of the funds should be instructed to send (or "push") the funds to.
`SamplePushInstructions`	String	This is a sample of the push instructions which should be given to the sender of the funds, provided in a way that the sender understands, using the language of their mobile operator. A sample instruction might be "To complete this transaction, now send UGX400,000 to 0771234565"
`RequestExpirySeconds`	Numeric	This is the number of seconds within which the request will expire.
`RequestMandatoryReference`	String	Some mobile money network operators require the mobile phone user to provide a very specific “payment reference” or “payment reason”, as some operators call it, in order for the incoming funds to be properly routed. If this is the case for the mobile money network from which you wish to receive funds, then this parameter will be present and non-empty. If the mobile money network doesn’t require this, the parameter will be present but empty.

5.2.3 Error Response Format

If your request is not successful, you shall get a response in the format described in section 3.4 above.

5.3 Instant Payment Notification API

Instant Payment Notification (IPN) is an API service that sends a notification when a transaction is completed. The IPN is a HTTP POST notification with information related to the transaction and is sent when a transaction completes successfully. When you enable IPN, the notification is sent to a default URL that you specify to handle IPN requests in your Payments account settings.

By setting a special parameter in your response to the IPN request, you also have opportunity to trigger an SMS to be automatically sent to the payer's mobile phone.

5.3.1 Receiving Instant Payment Notifications

IPN is designed to be secure, reliable, and asynchronous. To meet these requirements, the protocol requires you to acknowledge receipt of IPN messages by a simple HTTP header status code: '200 OK ' response. The IPN service also provides a retry mechanism to handle cases in which HTTP status '200 OK' is not returned; for example, when a transmission or receipt failure occurs.

Below is a simple setup process for an application to receive IPN requests:

1. Set up your server to receive the HTTP POST IPN notifications
2. Write a program that parses the IPN parameters. For a list of the parameters, see section 5.3.3 below.
3. Write your program so that it verifies the `signature` value sent in the IPN to make sure Yo! Payments sent the IPN. For more information, see section 5.3.4 below.
4. Write your program to use the returned transaction parameters to notify you of the IPN-related transactions.
5. If you would like an SMS to be sent to the payer, configure your system to respond to the IPN request and set the special `narrative` parameter as described below.

You should periodically test your URL for Instant Payment Notification. Check that the processing application is running on the server and that you can reach the server from the internet.

Your application must respond to each IPN message with HTTP header status code 200, whether or not you intend to do anything with it. If you do not respond, we assume that the message was not received and the message is resent. The message is resent periodically until your application responds back with the HTTP status code 200.

Note: Avoid duplicate IPN messages. Check that you have not already processed the transaction identified by the same `network_ref` and `msisdn` parameters returned in the IPN message (See section 5.3.3IPN Transaction below). You may need to store the `network_ref` and `msisdn` parameters returned in IPN messages in a file or database so that you can check for duplicates. If the combination of `network_ref` and `msisdn` parameters sent is a duplicate, you should not process it again.

5.3.2 Triggering an SMS to the Payer's Mobile Phone

You may trigger an SMS to the payer's mobile phone by responding with a URL-encoded body as part of your response to the IPN. Below is an example of a valid body that will trigger an SMS:

```
narrative=
Dear+John%2C+we+have+received+your+payment+of+UGX2%2C500+%28ref%3A+71299191%29.+The+s
ame+has+been+successfully+posted+to+your+account.+Thank+you+for+your+business.
```

The above response will trigger the following SMS to the mobile phone of the payer: "`Dear John, we have received your payment of UGX2,500 (ref: 71299191). The same has been successfully posted to your account. Thank you for your business.`"

5.3.3 IPN Transaction Parameters

The table below describes the transaction information sent as HTTP POST variables or parameters with IPN requests:

Parameter Name	Type	Description
`date_time`	String	This parameter is set to the date and time the transaction or payment was affected. Note: This parameter is set to the network provider date time formatting and may be different for the various network providers. Also note that some network providers only supply a date for the transaction without specifying the time.
`amount`	Numeric	This parameter is set to the transaction amount.
`narrative`	String	This parameter is set to the reason for the payment.
`network_ref`	String	This parameter is set to the network provider reference
`external_ref`	String	This parameter is set to an external reference for this transaction.
`msisdn`	Numeric	This is parameter set to the phone number associated with the transaction.
`signature`	String	This parameter is used to authenticate that the request originated from Yo! Payments. The parameter is set to a base-64 encoded RSA signature. The signature is generated by concatenating the following request parameters in order: 1. `date_time`

		2. `amount` 3. `narrative` 4. `network_ref` 5. `external_ref` 6. `msisdn` The resulting string is signed and base-64 encoded.

Below is a sample IPN message:

```
date_time=2011-03-
13+00%3A00%3A00&amount=100000&msisdn=256700000000&network_ref=1234&external_ref=5678&
narrative=payment+reason&signature=9msIVVfjUiZob4SHOSuIxQ==
```

5.3.4 Verifying IPN Notifications

You must ensure that the IPN indeed came from Yo! Payments. You can do this by verifying the value of the `signature` parameter contained in the IPN message request. The IPN message contains the components or parameters (see section 5.3.3 above) you need to validate with server-side signature verification.

To verify the signature, follow the simple procedure below:

1. Decode the base-64 representation of the signature as received in the IPN request message.
2. Verify the decoded signature using the public key or certificate provided in your Payments account and the IPN message parameters as described in section 5.3.3 above for the `signature` parameter.

Essentially, the program you write to verify the signature, base-64 decodes the signature; decrypts the signature using a public key and then compares the resulting message digest or hash (SHA1) with a hash of a string that concatenates IPN message parameter values for: `date_time, amount, narrative, network_ref, external_ref,` and `msisdn` in order. If the message digests or hashes are the same, you can confidently assert that the request originated from Yo! Payments.

5.4 TRANSACTION FAILURE NOTIFICATION API

The Transaction Failure Notification API sends a notification to your system when a transaction is unsuccessful. The notification is HTTP POST, URL encoded form data with a transaction reference identifier related to the failed transaction. Notifications are sent to a default URL that you specify to capture and process failure notification requests.

5.4.1 Processing Failure Notifications

Receipt of a notification must be acknowledged by a simple HTTP header status code: '200 OK ' response, otherwise the notification is resubmitted to your system at a later time.

Below is a simple setup process for an application to receive failure notification requests:

1. Set up your server to receive and process the HTTP POST, URL encoded form data.
2. Write your program to verify the signature value of the `verification` parameter sent in the notification to make sure Yo! Payments sent the notification. See 5.4.3 below for more information.
3. Write your program to use the reference value of the `failed_transaction_reference` parameter to move the related transaction on your system from a PENDING to a FAILED status.

5.4.2 Notification Parameters

The table below describes the HTTP POST variables or parameters with failure notification requests:

Parameter Name	Type	Description
`failed_transaction_reference`	String	This parameter is set to the reference that maps the failure notification to a transaction on your system.
`transaction_init_date`	String	This parameter is set to the date and time the transaction was initiated.
`verification`	String	This parameter is used to verify that the request originated from Yo! Payments. The parameter is set to a base-64 encoded RSA signature. The signature is generated by concatenating the following request parameters in order: 1. `failed_transaction_reference` 2. `transaction_init_date` The resulting string is signed and base-64 encoded.

Below is a sample failure notification message:

```
transaction_date=2012-02-
13+00%3A00%3A00&failed_transaction_reference=12345678&verification=yGtrCY0gb1EaTu6ugj
G6+DQxrmxy7LAX0O9yD2EygZmjx
```

5.4.3 Verifying Failure Notifications

To verify the signature value of the `verification` parameter contained in the failure notification request, follow the simple procedure below:

1. Decode the base-64 representation of the signature as received in the notification.

2. Verify the decoded signature using the public key or certificate provided in your Payments account and the notification message parameters as described in section 5.4.2 above for the `verification` parameter.

Essentially, the program you write to verify the signature, base-64 decodes the signature; decrypts the signature using a public key and then compares the resulting message digest or hash (SHA1) with a hash of a string that concatenates the failure notification message parameter values for: `failed_transaction_reference` and `transaction_init_date` in order.

6 CHECK TRANSACTION STATUS

This facility enables you to check the status of a transaction that was earlier submitted for processing. This is useful particularly in the case where the “`NonBlocking`” field (see section 5.1.1 above) was set to `TRUE`. When you make a transaction request with the “`NonBlocking`” field set to `TRUE`, you immediately receive a response containing a transaction reference, as documented in section 3.5 above. You then use this transaction reference to follow up on your request.

Apart from the case where the “`NonBlocking`” field (see section 5.1.1 above) was set to `TRUE`, you may also use this API call to check the status of any transaction on the system.

6.1 REQUEST FORMAT

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>actransactioncheckstatus</Method>
      <TransactionReference></TransactionReference>
      <PrivateTransactionReference></PrivateTransactionReference>
      <DepositTransactionType></DepositTransactionType>
    </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string: `actransactioncheckstatus`
`TransactionReference`	String	Mandatory on Condition	Enter here the reference to the transaction whose status you would like to follow up on. This is typically the reference which came through with an earlier transaction response (see sections 3.3, 3.4 and 3.5 above). This field is optional if `PrivateTransactionReference` (see below) has been specified.
`PrivateTransactionReference`	String	Optional	This is the private transaction reference which was supplied in the `ExternalReference` parameter of a previously submitted Deposit, Withdrawal or Internal Transfer API request. If both `TransactionReference` and `PrivateTransactionReference` are

			specified, TransactionReference takes precedence and PrivateTransactionReference will be ignored. In the event that multiple transactions are associated with the specified PrivateTransactionReference, the most recent transaction will be returned.
DepositTransactionType	String	Optional	Set to "PUSH" if following up on the status of a push deposit funds transaction (see section 5.2). Set to "PULL" if following up on the status of a pull deposit funds transaction (see section 5.1). When not set or present for deposit funds transactions the default is pull deposit funds transactions.

6.2 SUCCESSFUL TRANSACTION RESPONSE FORMAT

If your request has been fully processed, and was successful, you shall get a response in the format described in section 3.3. In addition to the parameters in the response in section 3.3, the following parameters will also be present:

Parameter Name	Type	Description
Amount	Integer	The amount involved in the transaction.
AmountFormatted	String	The formatted amount. This is the same as Amount above but has the currency prepended. E.g UGX 20,000/=
CurrencyCode	String	The currency code. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter.
TransactionInitiationDate	String	The transaction initiation date. This is of the format YYYY-MM-DD HH:mm:ss, E.g 2011-07-18 14:12:20
TransactionCompletionDate	String	The transaction completion date. This is of the format YYYY-MM-DD HH:mm:ss, E.g 2011-07-18 14:12:21. If the transaction is not yet complete, this will be set to 0000-00-00 00:00:00

6.3 ERRONEOUS TRANSACTION RESPONSE FORMAT

If your request has been fully processed, but encountered an error, you shall get the response in the format described in section 3.4.

6.4 Pending Response Format

If your request has not yet been fully processed, you shall get the response documented in section 3.5.

7 INTERNAL TRANSFER TRANSACTION

An Internal Transfer is a transfer of funds from your Payment Account to the Payment Account of another Yo! Payments user. You can perform the transfer in any currency where you have a balance above zero. Verify your account balance in the currency you are interested in transferring before attempting to execute the Internal Transfer. You shall also need to have with you the account number of the other user to whom you wish to transfer funds, as well as their email address.

7.1 REQUEST FORMAT

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acinternaltransfer</Method>
      <CurrencyCode></CurrencyCode>
      <Amount></Amount>
      <BeneficiaryAccount></BeneficiaryAccount>
      <BeneficiaryEmail></BeneficiaryEmail>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acinternaltransfer`.
`CurrencyCode`	String	Mandatory	Specify here the standard code of the currency in which you wish to perform the transfer. Note that (a) you must have a positive balance in the currency you specify; and (b) you must specify a valid currency code. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter.
`Amount`	Float	Mandatory	This is the amount to be transferred. This amount will be debited from your Payment Account (plus applicable fees) and credited to the Payment Account of the beneficiary. Fractional amounts are

			supported.
`BeneficiaryAccount`	Numeric	Mandatory	This is a numerical value representing the account number of the Payment Account to which you are transferring the funds, i.e the “beneficiary” account number. Obtain this account number from the user to whom you are transferring the funds. You must provide a valid account number for the transaction to succeed.
`BeneficiaryEmail`	String	Mandatory	Provide here the email address of the recipient of the funds. You must provide a valid email address for the transaction to succeed. The Yo! Payments transaction processor will attempt to match the values you provide in the `BeneficiaryAccount` and `BeneficiaryEmail` with the values stored in the database. If they do not match, the transaction will not succeed.
`Narrative`	String	Mandatory	Textual narrative about the transaction. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the `NarrativeFileBase64` parameter (see below).
`NarrativeFileName`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example “receipt.doc” or “receipt.pdf”. Note that the contents of this parameter are ignored if you have not provided the contents of the file using `NarrativeFileBase64` below.
`NarrativeFileBase64`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attached a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the

			contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the benecifiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.

7.2 SUCCESS RESPONSE FORMAT

If your request is successful, you shall get a response in the format described in section 3.3 above.

7.3 ERROR RESPONSE FORMAT

If your request is not successful, you shall get a response in the format described in section 3.4 above.

7.4 PENDING RESPONSE FORMAT

Not valid for this transaction. You will not get the PENDING response for this type of transaction.

8 BALANCE CHECK REQUEST

The Balance Check Request enables you to get the current balance of your account. **NOTE:** If your account has more than one currency, this request will provide you the balance in all available currencies on your account, including any airtime balances[3] you may have. See Section 13 for a list of all supported currencies.

8.1 REQUEST FORMAT

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acacctbalance</Method>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acacctbalance`.

[3] See Section 10 for a description of Airtime Balances

8.2 Success Response Format

If your request is successful, you shall get a response in the format below:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>OK</Status>
      <StatusCode>0</StatusCode>
      <Balance>
         <Currency>
            <Code></Code>
            <Balance></Balance>
         </Currency>
         <Currency>
            <Code></Code>
            <Balance></Balance>
         </Currency>
      </Balance>
   </Response>
</AutoCreate>
```

The table below provides a description of the fields.

<table>
<tr><th>Parameter Name</th><th>Type</th><th>Presence</th><th>Description</th></tr>
<tr><td>Status</td><td>String</td><td>Always Present</td><td>If there is no error, this is set to “OK”.</td></tr>
<tr><td>StatusCode</td><td>Integer</td><td>Always Present</td><td>This field is set to 0 (zero)</td></tr>
<tr><td colspan="4">The Balance Section
The Balance section, if present in the response, contains of one or more Currency sub-sections. Each Currency sub-section contains balance information for that specific currency. Note that the Balance section is not always present in the response to the acacctbalance request. Rather, it is only present if there has been at least one transaction carried out on your account. You should interpret the absence of the Balance section as an indication that no transaction has yet been carried out on your account.
The information in the rows below pertains to the contents of each of the Currency sub-section. This information is only relevant if the Balance section is present in the response that you got.</td></tr>
<tr><td>Code</td><td>String</td><td>Always Present</td><td>The currency code for the currency. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter.</td></tr>
<tr><td>Balance</td><td>Float</td><td>Always Present</td><td>The balance in the specific currency.</td></tr>
</table>

8.3 ERROR RESPONSE FORMAT

If your request is not successful, you shall get a response in the format described in section 3.4 above, with the exception of the `TransactionStatus` and `TransactionReference` fields, which do not apply for the `acacctbalance` request.

8.4 PENDING RESPONSE FORMAT

Not valid for this transaction. You will never get the PENDING response for this type of transaction.

9 MINI STATEMENT REQUEST

The MINI statement request API call enables you to get a list of transactions which were carried out on your account during a certain period of time.

9.1 REQUEST FORMAT

Below is the XML request format:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acgetministatement</Method>
      <StartDate></StartDate>
      <EndDate></EndDate>
      <TransactionStatus></TransactionStatus>
      <CurrencyCode></CurrencyCode>
      <ResultSetLimit></ResultSetLimit>
      <TransactionEntryDesignation></TransactionEntryDesignation>
      <ExternalReference></ExternalReference>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acgetministatement`.
`StartDate`	String	Optional	The date and time from which transactions should be queried. If this is specified, it must be of the format `YYYY-MM-DD HM:MM:SS` for example `2011-03-13 00:00:00`. If this is not specified, the most recent 5 transactions will be returned
`EndDate`	String	Optional	The date and time up to which transactions should be queried. If this is specified, it should be of the format `YYYY-MM-DD HM:MM:SS` for example `2011-04-27 22:19:50`. This field must be present if `StartDate` above is specified
`TransactionStatus`	String	Optional	The transaction status. If specified, the only valid values are: `FAILED`, `PENDING`, `SUCCEEDED` and `INDETERMINATE`. You can specify more than one transaction statuses by using commas for example `PENDING,SUCCEEDED` or

			`INDETERMINATE,PENDING`. For an explanation of these transaction statuses, please refer to sections 3.3, 3.4 and 3.5
`CurrencyCode`	String	Optional	The currency code. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter. If you specify this parameter, the response will contain only transactions in the specified currency.
`ResultSetLimit`	Integer	Optional	If you wish to limit the maximum number of results that can be returned by this API call, specify here the maximum number of results that should be allowed. For example, if you specify 10 here, then a maximum of 10 results will be returned by this API call, even if more results could have been returned. **IMPORTANT NOTE 1**: Use a special value of 0 to mean "*do not limit the number of returned results, return everything*". **IMPORTANT NOTE 2**: By default, you can only receive a maximum of 15 results, regardless of what you specify here. However, by sending a special request to lift this limit to your account manager, this limit can be lifted. If you do not specify this parameter in your API call, a default of 15 will be used.
`TransactionEntryDesignation`	String	Optional	There are two broad categories of transactions which will be present on your statement. These are: (a) Ordinary Transactions; and (b) Transaction Fees. Transaction Fees are the charges which the Yo! Payments service levies on your account for your usage of the service (visit https://payments.yo.co.ug/ for an up-to-date list of charges). Ordinary Transactions are those types of transactions which are not fees levied by the Yo! Payments service. This parameter, therefore can take any of three possible values namely: (a) "TRANSACTION"; (b) "CHARGES"; or (c) "ANY". When you set TransactionEntryDesignation to "TRANSACTION", the response to your API call will only include Ordinary Transactions. When you set

			TransactionEntryDesignation to "CHARGES', the response to your API call will only include Transaction Fees. When you set TransactionEntryDesignation to "ANY", the response to your API call will include both Ordinary Transactions and Transaction Fees. If you do not set this parameter, it will default to "ANY".
`ExternalReference`	String	Optional	Set this field to an external reference if you wish to filter your results by the `ExternalReference` parameter that you provided with your withdraw or deposit request. See sections 4.1, 5.1.1, and 5.2.1 on the `ExternalReference` request parameter in withdraw and deposit transactions respectively.

9.2 Success Response Format

If your request is successful, you shall get a response in the format below. The `Transactions` section will only be present if there's at least one transaction for the specified period. If you did not specify the start date and end date, the absence of the `Transactions` section will mean that no transactions have ever been carried out on your account. The transactions are arranged in ascending order, i.e, the most recent transactions will appear at the end in the `Transactions` section.

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Response>
      <Status>OK</Status>
      <StatusCode>0</StatusCode>
      <TotalTransactions></TotalTransactions>
      <ReturnedTransactions></ReturnedTransactions>
      <Transactions>
         <Transaction>
            <TransactionSystemId></TransactionSystemId>
            <TransactionReference></TransactionReference>
            <TransactionStatus></TransactionStatus>
            <InitiationDate></InitiationDate>
            <CompletionDate></CompletionDate>
            <NarrativeBase64></NarrativeBase64>
            <Currency></Currency>
            <Amount></Amount>
            <Balance></Balance>
            <GeneralType></GeneralType>
            <DetailedType></DetailedType>
            <BeneficiaryBase64></BeneficiaryBase64>
            <SenderBase64></SenderBase64>
            <Base64TransactionExternalReference></Base64TransactionExternalReference>
            <TransactionEntryDesignation></TransactionEntryDesignation>
         </Transaction>
         <Transaction>
            <TransactionSystemId></TransactionSystemId>
            <TransactionReference></TransactionReference>
            <TransactionStatus></TransactionStatus>
            <InitiationDate></InitiationDate>
            <CompletionDate></CompletionDate>
            <NarrativeBase64></NarrativeBase64>
            <Currency></Currency>
            <Amount></Amount>
            <Balance></Balance>
            <GeneralType></GeneralType>
            <DetailedType></DetailedType>
            <BeneficiaryBase64></BeneficiaryBase64>
            <SenderBase64></SenderBase64>
            <Base64TransactionExternalReference></Base64TransactionExternalReference>
            <TransactionEntryDesignation></TransactionEntryDesignation>
         </Transaction>
      </Transactions>
   </Response>
</AutoCreate>
```

The table below provides a description of the fields in the XML response above:

Parameter Name	**Type**	**Presence**	**Description**
`Status`	String	Always Present	If there's no error, this is set to "`OK`"
`StatusCode`	Integer	Always Present	This field is set to 0 (zero)
`TotalTransactions`	Integer	Always Present	The total number of transactions that has ever been carried out on your account
`ReturnedTransactions`	Integer	Always Present	The total number of transactions that this request has returned
`Transactions` (Not always present)			
`TransactionSystemId`	Integer	Always Present	The unique identifier of the transaction. Use this parameter to sort transactions in the order in which they occurred. Sorting transactions in descending order of this parameter means that the most recent transactions will come first.
`TransactionReference`	String	Always Present	This field contains a value which uniquely identifies this transaction in your Yo!

			Payments account. This is typically the reference which comes through with an earlier transaction response (see sections 3.3, 3.4 and 3.5 above). You can use this reference to check the status of the transaction using the `actransactioncheckstatus` API.
`TransactionStatus`	String	Always Present	The transaction status. This is always set to any of the following: `FAILED, PENDING, SUCCEEDED, INDETERMINATE`. For an explanation of these transaction statuses, please refer to sections 3.3, 3.4 and 3.5 .
`InitiationDate`	String	Always Present	The date and time when this transaction was initiated. This is of the format `YYYY-MM-DD HH:MM:SS` for example `2011-03-23 13:23:10`
`CompletionDate`	String	Always Present	The date and time when this transaction was completed. This is of the format `YYYY-MM-DD HH:MM:SS` for example `2011-03-23 13:23:15`
`NarrativeBase64`	String	Always Present	A base64 encoded string representing notes about this transaction
`Currency`	String	Always Present	The transaction currency. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter.
`Amount`	Integer	Always Present	Amount involved in this transaction
`Balance`	String	Always Present	The account balance after the transaction was completed
`GeneralType`	String	Always Present	The Transaction general type for example `CREDIT`

`DetailedType`	String	Always Present	The detailed transaction type code. For a list of all possible values of this field, refer to section 14.
`BeneficiaryBase64`	String	Always Present	A base64 encoded string representing the beneficiary identifier. This could be a phone number, or an account number or any other information. The type of information stored in this field depends on the value of the `DetailedType` field above. To know what type of information is stored in this field for each possible value of the `DetailedType`, refer to section 14.
`SenderBase64`	String	Always Present	A base64 encoded string representing the sender identifier. This could be a phone number, or an account number or any other information. The type of information stored in this field depends on the value of the `DetailedType` field above. To know what type of information is stored in this field for each possible value of the `DetailedType`, refer to section 14.
`Base64TransactionExternalReference`	String	Always Present	If you provided the "External Reference" parameter at the time you performed the transaction, this field will be set to the base-64 encoded value of the “External Reference” parameter you provided. This field is always present but may be empty if you did not provide an "External Reference" parameter
`TransactionEntryDesignation`	String	Always Present	This parameter tells you the category of the transaction. There are two broad categories

			of transactions which will be present on your statement. These are: (a) Ordinary Transactions; and (b) Transaction Fees. Transaction Fees are the charges which the Yo! Payments service levies on your account for your usage of the service (visit https://payments.yo.co.ug/ for an up-to-date list of charges). Ordinary Transactions are those types of transactions which are not fees levied by the Yo! Payments service. This parameter, therefore can take any of two possible values namely: (a) "TRANSACTION"; (b) "CHARGES". When set to "TRANSACTION", this means the transaction is of type "Ordinary Transactions". When set to "CHARGES", this means that the transaction is of type "Transaction Fees"

9.3 ERROR RESPONSE FORMAT

If your request is not successful, you shall get a response in the format described in section 3.4 above with the exception of the `TransactionStatus` and `TransactionReference` fields, which do not apply for the `acgetministatement` request.

9.4 PENDING RESPONSE FORMAT

Not valid for this transaction. You will never get the PENDING response for this type of transaction.

10 AIRTIME TRANSFER

10.1 Overview of Airtime Transfer with Yo! Payments

Yo! Payments has the capability to send mobile calling credit (“airtime”) directly to users. Being designed to work with multiple mobile network operators simultaneously, Yo! Payments gives you a convenient central system through which you can send airtime to users on different mobile networks. This is useful in a variety of scenarios such as:

- Airtime Vending. If you are in the business of selling airtime, you can use Yo! Payments to send the purchased airtime to the mobile phone of the user. Our API can seamlessly be connected to airtime vending systems;
- Rewards. If you are carrying out a promotion where you wish to reward participants in your promotion with airtime, you can use Yo! Payments for this.
- Management of Phone Expenditure. Use this system to manage your mobile communication costs by sending your staff specific amounts of airtime every month.

10.2 What You Need in Place to Use the Airtime Transfer System

All you need is an airtime balance on your Payment Account. Contact your account representative to discuss how to load airtime onto your Payment Account.

10.3 Available APIs for Airtime Transfer

Yo! Payments offers you the following APIs for Airtime Transfer.

1. Send Airtime to Mobile API
2. Send Internal Airtime API
3. Check Balance API
4. Statement Request API
5. Check Status API

Note that the APIs for Airtime Transfer are very similar to those for Mobile Money. In some cases, such as the “Check Balance”, “Statement Request” and “Check Status” APIs, there is actually no difference between the Airtime Transfer API and the Mobile Money API.

In the sections which follow, we will describe in more detail how to use these APIs.

10.4 SEND AIRTIME TO MOBILE

This API call enables you to send airtime to a mobile phone user. Note that you must have an airtime balance on your Payment Account for you to successfully use this API call (see section 10.2 for more information on how to get an airtime balance).

NOTE: This request is largely similar to the “Withdraw Funds Transaction” request described in Section 4. Therefore, you will find it relatively easy to use this API, if you have used that API before.

10.4.1 Request Format

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acsendairtimemobile</Method>
      <NonBlocking></NonBlocking>
      <Amount></Amount>
      <Account></Account>
      <InternetBundle></InternetBundle>
      <AccountProviderCode></AccountProviderCode>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
      <ProviderReferenceText></ProviderReferenceText>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acsendairtimemobile`.
`NonBlocking`	String	Optional	If set to “TRUE”, you will follow up on the status of the transaction as described in section 10.6. If this parameter is empty or set to “FALSE” by default your connection to the system is maintained until your request is fulfilled.
`Amount`	Float	Mandatory	This is the amount of airtime to be sent to the mobile user. Must be set to a value greater than zero. Fractional amounts may not be supported by certain mobile

			money providers. This amount will be deducted from your airtime balance.
`Account`	Numeric	Mandatory	This is a numerical value representing the telephone number of the user who is to receive the airtime. Telephone numbers MUST have the international code prepended, without the "+" sign. An example of a mobile money account number which would be valid for the MTN Uganda network is `256771234567`.
`InternetBundle`	String	Optional	Set this field to the string `true` if the `Amount` parameter value is to be considered as an internet bundle. By default this value is `false`. Note that the transaction will fail if there is no internet bundle corresponding to the exact value of the `Amount` parameter indicated.
`AccountProviderCode`	String	Optional	Provide here the account provider code of the institution holding the account indicated in the `Account` parameter. See section 12 for a list of all supported account provider codes.
`Narrative`	String	Mandatory	Textual narrative about the transfer. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the `NarrativeFileBase64` parameter (see below).
`NarrativeFileName`	FileNameString[4]	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example "receipt.doc" or "receipt.pdf". Note that

[4] A FileNameString is a case-insensitive string comprising of one or more of the following valid characters: {a through z, 0 through 9, period character, underscore character, whitespace, hyphen}. The FileNameString is validated with the following regular expression: `^[a-zA-Z0-9\._\s-]+$`

			the contents of this parameter are ignored if you have not provided the contents of the file using `NarrativeFileBase64` below.
`NarrativeFileBase64`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attached a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the benecifiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.
`ProviderReferenceText`	String	Optional	In this field, enter text you wish to be present in any confirmation message which the mobile money provider network sends to the subscriber upon successful completion of the transaction. Some mobile money providers automatically send a confirmatory text message to the subscriber upon completion of transactions. This parameter allows you to provide some text which will be appended to any such

			confirmatory message sent to the subscriber.

10.4.2 Success Response Format

If your request is successful, you shall get a response in the format described in section 3.3 above.

10.4.3 Error Response Format

If your request is not successful, you shall get a response in the format described in section 3.4 above.

10.4.4 Pending Response Format

If your request is pending processing, for example in the case where you set `NonBlocking` to `TRUE`, then you shall get a response in the format described in section 3.5 above.

10.5 Send Internal Airtime

This API call enables you to send airtime from your Yo! Payments account to the account of another Yo! Payments user. Note that you must have an airtime balance on your Payment Account for you to successfully use this API call (see section 10.2 for more information on how to get an airtime balance).

NOTE: This request is largely similar to the "Internal Transfer Transaction" request described in Section 4. Therefore, you will find it relatively easy to use this API, if you have used that API before.

10.5.1 Request Format

Below is the format of the XML request:

```
<?xml version="1.0" encoding="UTF-8"?>
<AutoCreate>
   <Request>
      <APIUsername></APIUsername>
      <APIPassword></APIPassword>
      <Method>acsendairtimeinternal</Method>
      <CurrencyCode></CurrencyCode>
      <Amount></Amount>
      <BeneficiaryAccount></BeneficiaryAccount>
      <BeneficiaryEmail></BeneficiaryEmail>
      <Narrative></Narrative>
      <NarrativeFileName></NarrativeFileName>
      <NarrativeFileBase64></NarrativeFileBase64>
      <InternalReference></InternalReference>
      <ExternalReference></ExternalReference>
   </Request>
</AutoCreate>
```

The table below describes the parameters which are unique to this request. For a description of the standard parameters, refer to section 3.2.2.

Parameter Name	Type	Opt	Description
`Method`	String	Mandatory	Must be set to the string `acsendairtimeinternal`.
`CurrencyCode`	String	Mandatory	Specify here the standard code of the currency in which you wish to perform the transfer. Note that (a) you must have a positive balance in the currency you specify; and (b) you must specify a valid currency code. Please see section 13 for a complete list of currencies supported in the system, and the corresponding codes that are applicable for this parameter. **NOTE:** You must specify an airtime currency code here, if you try to specify a currency code used for mobile money, your request will fail.

`Amount`	Float	Mandatory	This is the amount of airtime to be sent to the beneficiary Yo! Payments user. Must be set to a value greater than zero. Fractional airtime amounts may not be supported by certain mobile networks. This amount will be deducted from your airtime balance (plus any applicable fees) and credited to the airtime balance of the Payment Account of the beneficiary. Fractional amounts may not be supported.
`BeneficiaryAccount`	Numeric	Mandatory	This is a numerical value representing the account number of the Payment Account to which you are transferring the funds, i.e the “beneficiary” account number. Obtain this account number from the user to whom you are transferring the airtime. You must provide a valid account number for the transaction to succeed.
`BeneficiaryEmail`	String	Mandatory	Provide here the email address of the recipient of the airtime. You must provide a valid email address for the transaction to succeed. The Yo! Payments transaction processor will attempt to match the values you provide in the `BeneficiaryAccount` and `BeneficiaryEmail` with the values stored in the database. If they do not match, the transaction will not succeed.
`Narrative`	String	Mandatory	Textual narrative about the transaction. Enter here a sentence describing the transaction. Provide a maximum of 4096 characters here. If you wish to provide more information, consider using the `NarrativeFileBase64` parameter (see below).
`NarrativeFileName`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attach a scanned receipt, or a scanned payment authorization, depending on your internally established business rules. This parameter requires you to provide the name of the file you are attaching, as a string, for example “receipt.doc” or “receipt.pdf”. Note that the contents of this parameter are ignored if you have not provided the contents of the file using

			`NarrativeFileBase64` below.
`NarrativeFileBase64`	String	Optional	This parameter enables you to attach a file to the transaction. This is useful, for example, in the case where you may want to attached a scanned receipt, or a scanned payment authorization, depending on your business rules. This parameter requires you to provide the contents of the file you are attaching, encoded in base-64 encoding. Note that the contents of this parameter are ignored if you have not provided a file name using `NarrativeFileName` above.
`InternalReference`	String	Optional	In this field, provide an internal transaction reference. If this transfer is related to another system transaction, enter its reference code in this field. If you are unsure about the meaning of this field, do not include it in your request. This field is useful in linking this request to another existing transaction which is already in the system.
`ExternalReference`	String	Optional	In this field, enter an external transaction reference. An external transaction reference is something which yourself and the benecifiary agree upon. For example, this may be an invoice number, or a phrase describing the purpose of this transaction in a way that the beneficiary would understand. This field is optional and you may omit it in your request.

10.5.2 Success Response Format

If your request is successful, you shall get a response in the format described in section 3.3 above.

10.5.3 Error Response Format

If your request is not successful, you shall get a response in the format described in section 3.4 above.

10.5.4 Pending Response Format

Not valid for this transaction. You will not get the PENDING response for this type of transaction.

10.6 CHECK STATUS

This facility enables you to check the status of a “Send Airtime” operation that was earlier submitted for processing. This is useful particularly in the case where the “`NonBlocking`” field (see section 10.4.1 above) was set to `TRUE`. When you make a transaction request with the “`NonBlocking`” field set to `TRUE`, you immediately receive a response containing a transaction reference, as documented in section 3.5 above. You then use this transaction reference to follow up on your request.

This API is exactly the same as the Mobile Money equivalent documented in Section 0. Please refer to Section 0 for the details on how to use this API call.

10.7 Check Balance

This API enables you to obtain your balance programmatically. This API is exactly the same as the Mobile Money equivalent documented in Section 8 of this document. **NOTE:** This request will give you your balance in all available currencies, including airtime balances. See Section 13 for a list of all supported currencies.

10.8 Statement Request

The Statement Request API call enables you to get a list of transactions which were carried out on your account during a certain period of time. This API is exactly the same as the Mobile Money equivalent documented in Section 9 of this document. **NOTE:** If you wish to only obtain airtime transfer transactions, make use of the `CurrencyCode` parameter. Specify there the particular currency code which corresponds to the airtime transfer transactions you wish to filter out. See Section 13 for a list of all supported currencies. See Section 9.1 for a description of how to use the `CurrencyCode` parameter.

11 API STATUS CODES

This section provides the different values of the status codes which you may receive in the `StatusCode` field of the XML response. Note that there are three distinct types of status codes:

- Status Codes Less than Zero
- Status Code Equal to Zero
- Status Codes Greater than Zero

11.1 Status Codes Less than Zero

If you get a status code less than zero then your API request failed.

NOTE: If you submit a transaction-oriented API request, such as withdrawal, deposit or internal transfer and you get a status code less than zero, then the `TransactionReference` field shall not be present in the XML Response and you shall not be able to view the transaction details in the “View Statement” section of your web account.

11.2 Status Code Equal to Zero

This means your API request was successful.

11.3 Status Codes Greater than Zero

If you get a status code greater than zero then your API request may have failed or may have succeeded. To determine whether your request failed or succeeded, check the `TransactionStatus` field in the XML response (for transaction-oriented API requests) or refer to the table in section 11.4 below (for other API requests).

Note that for transaction-oriented API requests, if the `TransactionStatus` field is set to `INDETERMINATE`, checking the table in section 11.4 below may not provide you with sufficient information on the status of your transaction. In such a case, contact your Yo! Payments representative for resolution if your transaction remains in the `INDETERMINATE` state for more than 24 hours.

11.4 Status Codes and their Meanings

Status Code Value	Meaning
-9999	You have submitted invalid XML, or one or more fields in the XML request you have submitted is missing or invalid. If you get this error code, check the `StatusMessage` parameter for more detailed information.
-40	Your deposit request failed. The PULL inbound deposit method is not supported by the network.
-39	Your deposit request failed. The PUSH inbound deposit method is not supported by the network.
-38	Failed to verify authentication signature. Please confirm that the signature is correct and is RSA with SHA1. Also verify that the public key provided corresponds to the private key that was used to generate the signature. (POSDEVICE)
-37	The specified purchase amount is not available for the requested product or service. Please verify the value for the amount(s) accepted for the given product or service. (POSDEVICE)
-36	You requested for a product or service that is currently not available for this POS device. (POSDEVICE)
-35	POS device is not linked to any account in the system. (POSDEVICE)
-34	A request was received from an unknown device or a POS device that has been disabled in the system. Please verify that the device exists in the system and is enabled. (POSDEVICE)
-33	Your `'acconfirmcustomer'` request failed. There was an error contacting the internal processing gateway. Please try your request again later, or contact support services if this error persists. (BILLPAY)
-32	Failed to confirm customer details. Bill-pay operator responded: Unknown customer. (BILLPAY)
-31	There was an internal error processing your request to perform a non-blocking transaction. Please submit your transaction again later.
-30	The transaction was not found in the system. Please verify your transaction reference.
-29	In the `acgetministatement` request, you have specified a currency code that does not exist or is in DRAFT state or is DISABLED
-28	In the `acgetministatement` request, you have specified an invalid start date from which to obtain a statement.
-27	In the `acgetministatement` request, you have specified an invalid end date up to which to obtain a statement.
-26	In the `acgetministatement` request, you have specified a start date without specifying the end date.

-25	In the `acgetministatement` request, you have specified an end date without specifying the start date.
-24	In the `acgetministatement` request, you have specified an invalid date range, E.g if the start date is after the end date
-23	This transaction has been declined because you have reached or exceeded one or more of your withdrawal limits.
-22	This transaction requires extra authorization before it may be completed. Requests for authorization have been duly sent. Upon successful authorization by all parties, the transaction shall be processed. DO NOT RE-SUBMIT THIS TRANSACTION UNLESS YOU ARE SURE IT HAS FAILED OR WAS REJECTED.
-21	Your IP address is not permitted to carry out transactions on this account.
-20	Your account was CANCELLED. Please contact your account representative for further advice.
-19	Your account was TERMINATED. Please contact your account representative for further advice.
-18	Your account is SUSPENDED. Please contact your account representative for further advice.
-13	You do not have sufficient funds to complete this transaction.
-12	There was a problem initiating this transaction. This transaction has failed. Please try again later. If this problem persists, contact support services.
-11	Invalid or Unsupported Currency
-10	The transaction has not been processed. This is the default transaction status for a transaction whose details have been successfully validated but it has not yet been submitted to the transaction processing system.
-9	Error committing initial statement entry. The transaction was not processed, and will not appear in the web interface.
-8	This is likley a duplicate transaction. Please vary your submission parameters such as references, if this is not in error.
-7	Invalid internal reference parameter. If you are sure that the internal reference you have provided is correct then the transaction was archived, or may have been deleted as a result of its old age.
-6	Duplicate transaction code (try again later)
-5	The file specified in 'file_narrative' was not found
-4	Invalid 'amount' parameter
-3	Unsupported transaction type parameter
-2	Unsupported MSISDN Network or Currency

-1	An internal error occurred. More information in 'ErrorMessage'
0	The Transaction was Successful
1	The transaction has been successfully recorded but is pending at the low-level. Use the `actransactioncheckstatus` API call to poll this transaction.
2	The transaction failed -- see 'ErrorMessage' for more information.
3	The transaction failed but we encountered an error updating the transaction state to mark the transaction as FAILED. The transaction is still in the INDETERMINATE state and will need to be manually marked as FAILED, and for a completion time to be added. If you get this error code, consider your transaction as FAILED. The transaction state will be updated to FAILED within 24 hours. Contact support services if the transaction state is not yet updated within 24 hours of your receiving this message.
4	The transaction succeeded but we encountered an error updating the transaction state to mark the transaction as successful. The transaction is still in the INDETERMINATE state and will need to be manually marked as SUCCEEDED, and for a completion time to be added. Your account may or may not have been credited with the funds. Check your statement on the online interface to verify whether your account was credited or not. If you get this error code, consider your transaction SUCCEEDED but ensure to contact support services if your account statement is not updated within 24 hours.
5	The transaction was rendered indeterminate in as far as our interaction with the network goes, however we failed to update the transaction state to add a completion date/time. The transaction will need to be manually completed and changed to FAILED or SUCCEEDED based on investigations with the mobile network. If you get this error code, contact support services if the transaction state is not definitively resolved within 1 hour. Your transaction may have succeeded or may have failed.
6	The transaction succeeded. However, because of an internal problem, your balance has not yet been updated to reflect the transaction. You shall notice that this transaction is still in the INDETERMINATE state. We shall mark the transaction as SUCCEEDED shortly and update your account balance. If you get this error code, consider your transaction successful. Contact support services if your account statement / balance is not updated to reflect this transaction within 24 hours.
7	Unsupported transaction type ''.The transaction was not processed and will appear in the web interface as a FAILED transaction. If you get this error code, consider your transaction as FAILED.
8	Unsupported transaction type ''.processed but there was a problem marking the transaction as FAILED. The transaction will be manually marked as failed shortly. If this transaction is still in the INDETERMINATE state within 24 hours of receiving this message, please contact support services. If you get this error code, consider your transaction as FAILED.
9	Indeterminate Transaction. The response we got from the mobile network

	mobile money system was inconclusive. It is not clear whether this transaction succeeded or failed. This situation shall be resolved within 1 hour and your transaction shall be moved to either the SUCCEEDED state or FAILED state, depending on the information from the mobile network mobile money system. If this situation is unresolved within 1 hour of your receipt of this message, kindly contact support services.
10	The transaction failed. There was an error contacting the internal processing gateway. Please try your transaction again later, or contact support services if this error persists.
11	The transaction failed. There was an error contacting the internal processing gateway. However, there was, also, a problem marking this transaction as FAILED. Therefore, you shall notice that this transaction is in the INDETERMINATE state. This shall be resolved within 1 hour of receiving this message. If you get this error code, your transaction failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour or if this error persists.
12	Indeterminate Transaction. There was an error communicating with the internal processing gateway. Therefore, you shall notice that this transaction is in the INDETERMINATE state. This shall be resolved within 1 hour of your receiving this message. If you get this error code, your transaction may have succeeded or may have failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour of if this error persists.
13	Indeterminate Transaction. There was an error communicating with the internal processing gateway. However, there was, also, a problem completing this transaction, therefore you shall notice that there is no completion date for this transaction. This situation shall be resolved within 1 hour of your receiving this message. If you get this error code, your transaction may have succeeded or may have failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour of if this error persists.
14	The OUTBOUND_MSISDN_DEBIT transation failed. Your account balance is unaffected. See 'ErrorMessage' for more information on why the transaction failed.
15	The OUTBOUND_MSISDN_DEBIT transaction failed. However, there was a problem restoring your account balance. Therefore, this transaction is still in the INDETERMINATE state. Your account balance will be restored within 1 hour of your receiving this message. If your account balance is still unrestored after 1 hour, contact support services and provide this reference code: ''
16	The OUTBOUND_MSISDN_DEBIT transaction could not be completed because of a problem posting the initial transaction statement. Therefore, this transaction is in the INDETERMINATE state. Your account balance will be restored within 1 hour of your receiving this message. If your account balance is still unrestored after 1 hour, contact support services and provide this reference code: ''. Please note that while your web interface will indicate SUCCEEDED, the transaction did not completely succeed.

17	The OUTBOUND_MSISDN_DEBIT succeeded, however you need to manually do the following: (a) Mark the transaction with code " in the Suspense Account " as SUCCEEDED; and (b) Debit the network tracking account " for network " with the amount: '
18	The transaction failed. There was an error contacting the internal processing gateway. Please try your transaction again later, or contact support services if this error persists.
19	The transaction failed. There was an error contacting the internal processing gateway. However, there was, also, a problem marking this transaction as FAILED. Therefore, you shall notice that this transaction is in the INDETERMINATE state. This shall be resolved within 1 hour of receiving this message. If you get this error code, your transaction failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour or if this error persists. If you contact support services, provide this reference code: ".
20	Indeterminate Transaction. There was an error communicating with the internal processing gateway. Therefore, you shall notice that this transaction is in the INDETERMINATE state. This shall be resolved within 1 hour of your receiving this message. If you get this error code, your transaction may have succeeded or may have failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour of if this error persists. If you contact support services, provide this reference code: ".
21	Indeterminate Transaction. There was an error communicating with the internal processing gateway. However, there was, also, a problem completing this transaction, therefore you shall notice that there is no completion date for this transaction. This situation shall be resolved within 1 hour of your receiving this message. If you get this error code, your transaction may have succeeded or may have failed. Contact support services if this transaction remains in the INDETERMINATE state for more than 1 hour of if this error persists. If you contact support services, provide this reference code: ".
22	The transaction failed but we encountered an error updating the transaction state to mark the transaction as FAILED. The transaction is still in the INDETERMINATE state and will need to be manually marked as FAILED, and for a completion time to be added. If you get this error code, consider your transaction as FAILED. The transaction will state will be updated to FAILED within 24 hours. Contact support services if the transaction state is not yet updated within 24 hours of your receiving this message. If you contact support services, provide this reference code: ".
23	The transaction was rendered indeterminate in as far as our interaction with the network goes, however we failed to update the transaction state to add a completion date/time. The transaction will need to be manually completed and changed to FAILED or SUCCEEDED based on investigations with the mobile network. If you get this error code, contact support services if the transaction state is not definitively resolved within 1 hour. Your transaction may have succeeded or may have failed. If you contact support services, provide this

	reference code: ''.
24	The transaction failed -- see 'ErrorMessage' for more information. Your balance remains unaffected.
25	Indeterminate Transaction. The response we got from the mobile network mobile money system was inconclusive. It is not clear whether this transaction succeeded or failed. This situation shall be resolved within 1 hour and your transaction shall be moved to either the SUCCEEDED state or FAILED state, depending on the information from the mobile network mobile money system. If this situation is unresolved within 1 hour of your receipt of this message, kindly contacts support services. If you contact support services, provide this reference code: ''.
26	The transaction was declined by the user.
27	The transaction is in a PENDING state but we encountered an error updating the transaction state to mark the transaction as PENDING. The transaction is still in the INDETERMINATE state and will need to be manually marked as PENDING, and for a completion time to be added. If you get this error code, consider your transaction as PENDING. The transaction state will be updated within 24 hours. Contact support services if the transaction state is not yet updated within 24 hours of your receiving this message.
28	Your `‘acrequestfundspush’` deposit request has expired. No funds related to the request were transferred to the designated mobile money account within 10 minutes from when the request was initiated. This means your push deposit request has subsequently FAILED.

12 ACCOUNT PROVIDER CODES

This section provides you with the mobile money provider codes for use in your withdraw / deposit transactions (`AccountProviderCode` parameter). Note that by default, Yo! Payments will attempt to derive the provider from the `Account` parameter. Nonetheless, there are providers who do not conform to any internationally-recognized account numbering.

Account Provider	Service Name	Account Provider Code
MTN Uganda Limited	Marketed as “MTN Mobile Money”	`MTN_UGANDA`
Airtel Uganda Limited	Marketed as "Airtel Money"	`AIRTEL_UGANDA`
Warid Telecom Uganda Limited	Marketed as "Warid Pesa"	`WARID_UGANDAPESA`

13 SUPPORTED CURRENCIES

This section provides you with an up-to-date list of currencies which are currently supported on the system. Use this list a reference point for your interpretation or use of the `Currency` and `CurrencyCode` XML parameters used in various requests and responses of Sections 7, 8 and 9.

Country	Currency Name	Provider	Currency Code[5]
Uganda	Uganda Shillings – MTN Mobile Money	MTN Uganda Limited	`UGX-MTNMM`
Uganda	Uganda Shillings – MTN Airtime	MTN Uganda Limited	`UGX-MTNAT`
Uganda	Uganda Shillings – Warid Airtime	Warid Telecom Limited	`UGX-WTLAT`
Uganda	Uganda Shillings – Orange Airtime	Orange Uganda Limited	`UGX-OULAT`
Uganda	Uganda Shillings – Airtel Airtime	Airtel Uganda Limited	`UGX-AIRAT`

[5] This is the value used in the `Currency` and `CurrencyCode` parameters in Sections 7, 8 and 9.

14 DETAILED TRANSACTION TYPE CODES

Some API calls may return a “detailed transaction type” field, such as the `DetailedType` field in the Mini Statement Request API call of section 9. This section provides you all possible detailed transaction type codes, and what they mean.

In addition, for the various detailed transaction type codes, the meaning of “`Beneficiary`” and “`Sender`” fields vary. This table also tells you the meaning of the `Beneficiary` and `Sender` for the various detailed transaction types. Use this information to interpret the `Beneficiary` and `Sender` fields in API calls which return this information, such as the Mini Statement Request API call of section 9.

Detailed Transaction Type Code	**Meaning**	**Beneficiary Field**	**Sender Field**
`INBOUND_MSISDN_CREDIT`	Addition of funds to the Yo! Payments account as a result of a deduction of mobile money from a mobile subscribers mobile money account. This is a CREDIT transaction and therefore results in increase of the Yo! Payments account balance if the transaction was successful	This is the account number of the Yo! Payments user receiving the funds.	This is the mobile money account number (normally telephone number) of the user who sent the funds.
`OUTBOUND_MSISDN_DEBIT`	Deduction of funds from the Yo! Payments account as a result of transfer of mobile money to the mobile money account of a mobile subscriber. This is a DEBIT transaction and therefore results in decrease of the Yo! Payments account balance if the transaction was successful	This is the mobile money account number (normally telephone number) of the user who is receiving the funds.	This is the account number of the Yo! Payments user sending the funds.
`INBOUND_YBSACCT_CREDIT`	Addition of funds to the Yo! Payments account as a result of the transfer of funds from another user account to the Yo! Payments account. This is a	This is the account number of the Yo! Payments user receiving the	This is the account number of the Yo! Payments user who sent the funds.

	CREDIT transaction, and therefore results in increase of the Yo! Payments account balance if the transaction was successful	funds.	
OUTBOUND_YBSACCT_DEBIT	Deduction of funds from the Yo! Payments account as a result of transfer of funds from the Yo! Payments account to the account of another Yo! Payment Services user	This is the account number of the Yo! Payments user receiving the funds.	This is the account number of the Yo! Payments user who sent the funds.
INBOUND_CUSTOM_CREDIT	Addition of funds to the Yo! Payments account as a result of a manual adjustment to the Yo! Payments account. Such a manual adjustment is only performed in special circumstances. In the event that this is done, the reason for the transaction is clearly documented in the 'Transaction Narrative' field	This is the account number of the Yo! Payments user receiving the funds.	This may take on custom information.
OUTBOUND_CUSTOM_DEBIT	Deduction of funds from the Yo! Payments account as a result of a manual adjustment to the Yo! Payments account. Such a manual adjustment is only performed in special circumstances and with clearly documented reasons, viewable by you in the 'Transaction Narrative' field.	This may take on custom information.	This is the account number of the Yo! Payments user sending the funds.

15 SANDBOX SERVER

A sandbox server has been provided to enable you to test out all the API operations above before finally launching your service. The sandbox server behaves exactly like the production server, except that there are no real transactions carried out.

15.1 Creating a Sandbox Account

To sign up for a Sandbox account, go to the URL below:

https://41.220.12.206/services/yopaymentsdev/signup/start/?sid=1

15.2 Using your Sandbox Account

To log into your Sandbox Account, go to the URL below:

https://41.220.12.206/services/yopaymentsdev/portal/

15.3 Submitting API Requests to the Sandbox Server

All API requests to the Sandbox server must be submitted to the URL below:

https://41.220.12.206/services/yopaymentsdev/task.php

15.4 Triggering Special Responses

In order for you to test out the various possible responses for transaction-oriented API operations[6], special parameters have been programmed into the Sandbox server to trigger the various responses, as documented here.

15.4.1 Withdraw Funds Transaction

The `Amount` parameter of the "Withdraw Funds" transaction (see section 4) may be set to the following special values to trigger the indicated corresponding response.

Special Value of `Amount` Parameter	**Response**
2111	This will trigger an unsuccessful response (see section 3.4), specifically with the `TransactionStatus` parameter set to "`FAILED`".
3991	This will trigger an unsuccessful response (see section 3.4), specifically with the `TransactionStatus` parameter set to "`INDETERMINATE`".
Any other value	This will trigger a successful response (see section 3.3).

[6] For the meaning of a "transaction-oriented" API operation, refer to section 2.3

15.4.2 Deposit Funds Transaction

The `Amount` parameter of the "Deposit Funds" transaction (see section 5) may be set to the following special values to trigger the indicated corresponding response.

Special Value of `Amount` Parameter	**Response**
2944	This will trigger an unsuccessful response (see section 3.4), specifically with the `TransactionStatus` parameter set to "`FAILED`".
8390	This will trigger an unsuccessful response (see section 3.4), specifically with the `TransactionStatus` parameter set to "`INDETERMINATE`".
Any other value	This will trigger a successful response (see section 3.3).