



Blesta Import Documentation

Note: Last Updated April 28, 2008. This document intended for use with Blesta v. 1.3.0+.

Introduction:

The import library is a set of methods that allow data to be directly and easily imported into a Blesta database. It is recommended to use this library **only** when importing data from another system. For adding non-import data, be sure to **always** use the Blesta API.

To access the import library use the following syntax (assuming this file is located above the /inc/ directory of your Blesta installation):

```
<?php
require("../inc/config.php");
require("../inc/import.class.php");

final class myImport extends AbstractImport {
    public function __construct() {
    }
}
?>
```

Both "config.php" and "import.class.php" are required in order to successfully use the import library. The import library requires the \$mysql db object created in config.php in order to access the database, this requirement means that you must have successfully installed Blesta before using the import library.

Any AbstractImport function may now be called from within the myImport class using the follow syntax:

```
myImport::function();
```

Where function() is the name of the library function (below) you are calling. For more examples, see the open source import scripts available at <http://www.blesta.com/download/>.

Below is a description of all available methods that may be used when extending the AbstractImport class.

Available Methods:

- importAdmin
- importClient
- importContact
- importCoupon
- importCreditCard
- importCron
- importCurrency
- importDeferredService
- importGateway
- importInvoice
- importMaillog

```
importMessage
importPackage
importPendingService
importSetting
importService
importSession
importTemplate
importTicket
importTransaction
importUpdate
getCountry
getState
getUsers
```

Method Descriptions:

* denotes required field (Note: all fields should be supplied, whether required or not).

importAdmin(&\$admin) – Import an Admin account.

Input:

```
Array (
  *[id] => 1
  [pass] => 5f4dcc3b5aa765d61d8327deb882cf99
  [fname] => First Name
  [lname] => Last Name
  [email] => address@domain.com
  [email2] => address@mobile.phone.com
  [notify] => 1
  [xslvl] => 0
  [lastseen] => 2008-01-01 12:59:59
  [lastlogged] => 2008-01-01 12:00:00
  [lastip] => 192.168.0.3
  [lastloggedip] => 192.168.0.3
  [emailarr] => 1
  [bccnotices] => 1
  [active] => 1
)
```

Returns:

NULL

Definition:

[id] – Admin ID.

[pass] – Admin MD5 password.

[fname] –First Name.

[lname] – Last Name.

[email] – E-mail address.

[email2] – Mobile/Cell Phone e-mail address.

[notify] – Notify Admin during certain events (1 – yes, 0 – no).

[xslvl] – Access Level (0 – root, 1 – full, 2 – billing/support, 3 – support).

[lastseen] – Date/time last seen.

[lastlogged] – Date/time last logged in.

[lastip] – Last IP admin used.

[lastloggedip] – Last IP admin logged in from.
[emailarr] – Subscribed to AR Reports (1 – yes, 0 – no).
[bccnotices] – Subscribed to BCC notices (1 – yes, 0 – no).
[active] – Active status (1 – active, 0 – inactive).

importClient(&\$user) – Import a Client account.

Input:

```
Array (
  * [id] => 1000
  [pass] => password
  [cname] => Company
  [fname] => Firstname
  [lname] => Lastname
  [address1] => Address 1
  [address2] => Address 2
  [city] => City
  [state] => CA
  [zip] => 90000
  [country] => USA
  [phone] => 123.123.1234
  [fax] => 123.123.1235
  [email] => email@address.com
  [signdate] => 2008-01-01 12:00:00
  [lastseen] => 2008-01-01 12:00:00
  [lastlogged] => 2008-01-01 12:00:00
  [lastip] => 192.168.0.1
  [lastloggedip] => 192.168.0.1
  [referredby] => My Referral
  [imethod] => paper
  [autod] => 1
  [cid] => 100
  [type] => billing
  [suspendable] => 1
  [active] => 1
  [type] => billing
  [ccno] => 5454545454545454
  [ccexp] => 0809
  [date] => 2008-01-01 12:00:00
  [active] => 1
  [nextattempt] => 2008-01-04
)
```

Returns:

NULL

Definition:

[id] – The client ID (int), required and must be unique.
[pass] – Client password in plain-text.
[cname] – Company Name.
[fname] – First Name.
[lname] – Last Name.
[address1] – Address 1.
[address2] – Address 2.

[city] – City.

[state] – State (2-character). Use **getState()** to obtain the 2-character state code for any state name.

[zip] – Zip/Postal Code.

[country] – Country (3-character). Use **getCountry()** to obtain the 3-character country code for any country name, or 2-character country code.

[phone] – Phone Number.

[fax] – Fax Number.

[email] – E-mail Address.

[signdate] – Date/time client was created.

[lastseen] – Date/time client was last seen.

[lastlogged] – Date/time client last logged in.

[lastip] – Last IP client used.

[lastloggedip] – Last IP client logged in with.

[referredby] – Client Reference.

[imethod] – Invoice method (paper/email).

[autod] – Auto Debit (1 – on, 0 – off).

[cid] – Contact ID (int). If given a new contact will be created with the this contact id, else a new contact will be created with the next auto increment ID.

[type] – Type of contact (billing/technical).

[ccno] – Credit card number (numbers only).

[ccexp] – Credit card expiration date (mmyy format).

[date] – Date contact was created.

[suspendable] – Suspendable (1 – yes, 0 – no).

[active] – Active (1 – active, 0 – inactive, 2 – fraud)

[nextattempt] – Date to next attempt a previously failed card payment.

importContact(&\$contact) – Import a Client Contact.

Input:

```
Array (  
  [id] => 100  
  [uid] => true  
  [fname] => Firstname  
  [lname] => Lastname  
  [address] => Address  
  [city] => City  
  [state] => CA  
  [zip] => 90000  
  [country] => USA  
  [phone] => 123.123.1234  
  [ccno] => 5454545454545454  
  [ccexp] => 0809  
  [date] => 2008-01-01 12:00:00  
  [active] => 1  
  [nextattempt] => 2008-01-04  
)
```

Returns:

NULL

Definition:

[id] – Contact ID (optional).

[uid] – Client ID that this contact belongs to.

[fname] – First Name.

[lname] – Last Name.

[address] – Address.

[city] – City.

[state] – State (2-character). Use **getState()** to obtain the 2-character state code for any state name.

[zip] – Zip/Postal Code.

[country] – Country (3-character). Use **getCountry()** to obtain the 3-character country code for any country name, or 2-character country code.

[phone] – Phone Number.

[ccno] – Suspendable (1 – yes, 0 – no).

[ccexp] – Active (1 – active, 0 – inactive, 2 – fraud)

[date] – Date to next attempt a previously failed card payment.

[active] – Active (1 – active, 0 – inactive).

[nextattempt] - Date to next attempt a previously failed card payment.

importCoupon(&\$coupon) – Import a Coupon.

Input:

```
Array (  
  *[id] => 1  
  [code] => DISCOUNT  
  [discount] => 5.00  
  [used] => 0  
  [max] => 10  
  [start] => 2008-01-01 12:59:59  
  [end] => 2008-02-01 12:59:59  
  [active] => 1  
)
```

Returns:

NULL

Definition:

[id] – Coupon ID.

[code] – Coupon code.

[discount] – Discount percentage.

[used] – Count of how many of these coupons have been used.

[max] – Maximum number of these coupons to allow.

[start] – Date/time to begin allowing this coupon.

[end] – Date/time to stop allowing this coupon.

[active] – Active status (1 – active, 0 – inactive).

importCreditCard (\$uid, \$cid, \$ccno, \$ccexp=false) – Import a Credit Card.

Input:

uid – Client ID

cid – Contact ID

ccno – Credit Card number (numbers only).

ccexp – Credit Card expiration date (mmyy format).

Returns:

NULL

importCron(&\$cron) – Import Cron job log data.

Input:

```
Array (  
  [name] => renewdate  
  [date] => 2008-01-01 12:59:59  
)
```

Returns:

NULL

Definition:

[name] – Name of cron task.

[date] – Date cron task last ran.

importCurrency(&\$currency) – Import a Currency.

Input:

```
Array (  
  [code] => USD  
  [format] => 1  
  [prefix] => $  
  [suffix] =>  
)
```

Returns:

NULL

Definition:

[code] – ISO 4217 Currency Code

[format] - Numerical format for currency (1 - 1,234.56; 2 - 1.234,56; 3 - 1 234.56; 4 - 1 234,56; 5 - 1,23,456.78; 6 - 1 234; 7 - 1.234; 8 - 1,234;).

[prefix] – Prefix symbol.

[suffix] – Suffix symbol.

importDeferredService(&\$deferred) – Import a service scheduled for suspension.

Input:

```
Array (
    [sid] => 1
    [aid] => 1
)
```

Returns:

NULL

Definition:

[sid] – Service ID to suspend.

[aid] – Admin ID that suspended the service, else 0.

importGateway(&\$gateway) – Import Payment Gateway settings.

Input:

```
Array (
    [filename] => paypal.class.php
    [name] => PayPal
    [username] => paypal123
    [key] => 123blkasdu
    [testmode] => True
    [opt1] =>
    [opt2]
    [merchant] => 0
    [active] => 1
)
```

Returns:

NULL

Definition:

[filename] – Filename of the payment gateway.

[name] – Friendly name of the payment gateway.

[username] – Gateway specific.

[key] – Gateway specific.

[testmode] – Defines whether this gateway is in test mode (True/False).

[opt1] – Gateway specific.

[opt2] – Gateway specific.

[merchant] – Defines whether this is a merchant gateway or not (1 – yes, 0 – no).

[active] – Active status (1 – active, 0 – inactive).

importInvoice (&\$invoice) – Import an Invoice.

Input:

```
Array (
  *[id] => 1234
  [uid] => 10000
  [dateb] => 2008-01-01
  [dated] => 2008-02-01
  [dater] => 2008-01-14
  [previous] => 0.00
  [applied] => 5.99
  [type] => email
  [status] => sent
  [notes] =>
  [biller] => 0
  [lineitems] => Array (
    [0] => Array (
      [id] =>
      [iid] => 1234
      [sid] => 1
      [name] => Hosting
      [price] => 5.99
    )
  )
)
```

Returns:

NULL

Definition:

[id] – Invoice ID.

[uid] – Client ID this invoice is for.

[dateb] – Date the invoice was created.

[dated] – Date the invoice is due.

[dater] – Date the invoice was paid in full.

[previous] – Previous amount due on invoice.

[applied] – Amount applied to this invoice (deprecated).

[type] – Type of invoice (paper/email).

[notes] – Admin notes on invoice.

[biller] – 0 if system, else Admin ID that created the invoice.

[lineitems] – Array of line items

[lineitems][i][id] – (optional) line item of this id.

[lineitems][i][iid] – Invoice ID.

[lineitems][i][sid] – Service ID this line item is attached to.

[lineitems][i][name] – Line item description.

[lineitems][i][price] – Price of line item.

importMaillog (&\$maillog) – Import a Mail log entry.

Input:

```
Array (  
  *[id] => 100  
  [uid] => 1000  
  [aid] => 0  
  [recipient] => user@domain.com  
  [subject] => Invoice Due  
  [body] => Your invoice is due.  
  [date] => 2008-01-01 12:59:59  
)
```

Returns:

NULL

Definition:

[id] – Mail log ID.

[uid] – Client ID this e-mail was sent to.

[aid] – Admin ID that send this e-mail, else 0 if automatically sent by the system.

[recipient] – E-mail address this message was sent to.

[subject] – Subject of this message.

[body] – Body of the e-mail.

[date] – Date/time this message was sent.

importMessage (&\$message) – Import a message from an Admin to Clients.

Input:

```
Array (  
  *[id] => 2  
  [aid] => 1  
  [text] => Welcome!  
  [datetime] => 2008-01-01 12:59:59  
)
```

Returns:

NULL

Definition:

[id] – ID of message.

[aid] – ID of admin that posted this message.

[text] – Text of the message.

[datetime] – Date/time stamp of when this message was added.

importPackage (&\$package) – Import a Package.

Input:

```
Array (  
  *[id] => 20  
  [name] => Package Name  
  [prices] => 0-1.99, 3-4.99  
  [instantact] => true  
  [notes] => This package is the best!  
  [welcome] => Thank you for signing update for this package.  
  [status] => 1  
)
```

Returns:

NULL

Definition:

[id] – Package ID.

[name] – Package Name.

[prices] – Package prices formatted as follows: term-price.

[instantact] – Instant Activation (true or false).

[notes] – Notes/details about this package.

[welcome] – Welcome e-mail body.

[status] – Status of the package (1 – Active, 0 – Inactive, 2 – Restricted).

importPendingService (&\$service) – Import a Pending Service.

Input:

```
Array (  
  *[id] => 10  
  [uid] => 1000  
  [pid] => 20  
  [ip] => 192.168.0.3  
  [order] =>  
  [datetime] => 2008-01-01 12:59:59  
  [paid] => 0  
)
```

Returns:

NULL

Definition:

[id] – Unique pending ID (8 char).

[uid] – Client ID associated with this pending service.

[pid] – Package ID for this service.

[ip] – IP address of the user that ordered this service.

[order] – Base 64 encoded serialized order array info (module specific).

[datetime] – Date/time this pending service was added

[paid] – Status of payment for this pending service (1 – paid, 0 – unpaid).

importSetting (&\$setting) – Import a Blesta System Setting.

Input:

```
Array (  
    *[i d] => 1  
    [name] => template  
    [value] => default  
    [comment] => Template Directory Name  
)
```

Returns:

NULL

Definition:

[id] – Setting ID.

[name] – Name of setting

[value] – Value of setting

[comment] – A comment for this setting

importService(&\$service) – Import a Service.

Input:

```
Array (  
    *[i d] => 10  
    [p i d] => 20  
    [m i d] => 1  
    [u i d] => 1000  
    [user1] =>  
    [user2] =>  
    [pass] =>  
    [opt1] =>  
    [opt2] =>  
    [notes] =>  
    [dates] => 2008-01-01  
    [dater] => 2008-01-02  
    [dated] => 0000-00-00  
    [datep] => 0000-00-00  
    [term] => 1  
    [ai d] => 0  
)
```

Returns:

NULL

Definition:

[id] – Service ID.

[pid] – Package ID this service uses.

[mid] – Module row ID this service uses.
[uid] – Client ID this service is for.
[user1] – Module dependant.
[user2] – Module dependant.
[pass] – Module dependant.
[opt1] – Module dependant.
[opt2] – Module dependant.
[dates] – Date service was created.
[dater] – Date service renews.
[dated] – Date service was deleted.
[datep] – Date service was suspended.
[term] – Term (in months)
[aid] – Admin ID that added the service, else 0.

importSession (&\$session) – Import a Session.

Input:

```
Array (
  *[id] => 05h1i dr24r1982j 6cm162ug1h2
  [expire] => 1212421012
  [value] =>
  sess_uid|s: 4: "1000"; sess_fname|s: 8: "Fi rstName"; sess_l name|s: 8: "LastName";
)
```

Returns:

NULL

Definition:

[id] – Session ID value.
[expire] – Unix timestamp when this session should expire.
[value] – Serialized array of [sess_uid], [sess_fname], [sess_lname].

importTemplate (&\$template) – Import an e-mail template.

Input:

```
Array (  
  *[id] => 1  
  [name] => Invoice Delivery (Unpaid)  
  [from] => billing@domain.com  
  [subject] => Invoice Due  
  [message] => [fname], An invoice...  
  [tags] => [fname], [lname], [invoice]  
)
```

Returns:

NULL

Definition:

[id] – E-mail template ID.
[name] – Name of the e-mail template.
[from] – From address.
[subject] – E-mail subject.
[message] – Body of E-mail.
[tags] – All tags allowed in this e-mail template (comma separated).

importTicket (&\$ticket) – Import a Trouble Ticket.

Input:

```
Array (  
  *[id] => 100  
  [uid] => 1000  
  [closed] => 0000-00-00 00:00:00  
  [priority] => 1  
  [area] => Billing  
  [tickets] => Array (  
    [0] => Array (  
      [uid] => 1000  
      [aid] => 0  
      [updated] => 2008-01-01 12:59:59  
      [body] => Hi, just updating this ticket.  
    )  
    ...  
  )  
)
```

Returns:

NULL

Definition:

[id] – ID of the ticket.
[uid] – Client ID this ticket is for.

[closed] – Date/time this ticket was closed.
[priority] – Priority of this ticket (0 – high, 1 – medium, 2 – low)
[area] – The subject of this ticket.
[ltickets] – Array of responses for this ticket.
[ltickets][i][uid] – Client ID that updated this ticket, else 0 if not updated by client.
[ltickets][i][aid] – Admin ID that updated this ticket, else 0 if not updated by an admin.
[ltickets][i][updated] – Date/time this ticket was updated.
[ltickets][i][body] – Body of this ticket update.

importTransaction (&\$transaction) – Import a Payment Transaction.

Input:

```

Array (
  [id] => 1234
  [uid] => 1000
  [cid] => 10
  [amount] => 5.99
  [credited] => 0.00
  [applied] => 5.99
  [type] => check
  [transid] => 123456
  [status] => approved
  [message] =>
  [date] => 2008-01-01 12:59:59
  [app] => Array (
    [0] => Array (
      [rid] => 1234
      [iid] => 100
      [amount] => 5.99
      [datetime] => 2008-01-01 12:59:59
    )
    ...
  )
)
  
```

Returns:

NULL

Definition:

[id] – Received Payment ID.
[uid] – ID of client that payment was received for.
[cid] – Contact ID for the client that payment was made under.
[amount] – Amount of payment.
[credited] – Amount of payment that was credited.
[applied] – Amount of payment that was applied.
[type] – Payment type.
[transid] – Payment transaction ID received from the payment processor or check number.
[status] – Status of payment (approved/error/declined/pending/voided).
[message] – Notes about this transaction
[date] – Date/time stamp when this payment was received.
[app] – Applied transaction array.
[app][i][rid] – Received Payment ID.
[app][i][iid] – Invoice ID this applies to.

[app][i][amount] – Amount to apply.

[app][i][datetime] – Date/time stamp of when this payment was applied.

importUpdate(\$table, \$field, \$value, \$wherefield, \$wherevalue) – Update an entry.

Input:

table – MySQL table name to update.

field – MySQL table field name to update.

value – Value to assign to **field**.

wherefield – Update **field** with **value** when **wherefield** equals **wherevalue**.

Wherevalue – Value of where field that triggers an update of **field** with **value**.

Returns:

true on success, false on failure.

getCountry(\$country) – Fetches the appropriate 3-character code for a given country name or 2-character country code.

Input: String

Returns:

3-char code on success, NULL on failure.

getState(\$state) – Fetches the appropriate 2-character code for a given state name.

Input: String

Returns:

2-char code on success, NULL on failure.

getUsers() – Fetches client and contact id for all clients.

Returns:

```
Array (  
  [0] => Array (  
    [uid] => 1000  
    [cid] => 100  
  )  
  ...  
)
```

Definition:

[uid] – Client ID

[cid] – Contact ID associated with this Client.