Liquid Documentation

Release 1.0.0

Ogi Sigit P

Jul 23, 2020

Contents

CHAPTER 1

Table of Contents

1.1 Reseller Account Startup

Upon activating your Reseller account with Liqu.id, you need to make the following settings within your Reseller Control Panel to begin selling Products to your Customers and Sub-Resellers:

1.1.1 Select your Selling Currency

At the time of sign-up, you must carefully select your desired Selling Currency. Your Selling Currency is the currency in which you wish to sell your Products and Services to your Customers and Sub-Resellers.

If the need arises, you may modify the Selling Currency as long as there are no transactions performed within your Reseller Account.

1.1.2 Add Funds in your Reseller Account

To let your Customers and Sub-Resellers buy Products and Services through you, you need to add and maintain sufficient funds in your Reseller Account with Liquid.

To add funds to your Account from the Control Panel:

1. Login to your Control Panel
2. Go to menu Billing -> Billing List
3. Click the Add Fund Button
4. There you would be presented with various methods available to Add Funds to your Account. You can choose one of the options and Add Funds to your Account.

1.1.3 Products and Services Signup and their Configuration

Following are the steps you need to take within your Control Panel to setup your Domain Registration Product for your Customers and Sub-Resellers:

- **Select the TLDs that you want to Sell**
    1. Login to your Reseller Control Panel
    2. Go to menu Settings -> Manage Products and Pricing -> Manage TLDs
    3. **Here, you may choose to either:**

        Sell To sell this TLD to your Customers and Sub-Resellers.

        No New Orders To stop accepting new Orders for this TLD from your Customers and Sub-Resellers. However, existing Orders can continue to be Managed and Renewed.

        Do Not Sell To entirely stop offering this TLD to your Customers and Sub-Resellers. However, you may set this status if there are no Orders of this TLD.

    4. Upon setting your TLD Signup Preferences, agree to the applicable Terms and Conditions and click the Submit button.
- **Set Selling Price for the Domain Registration Product**
    1. Login to your Reseller Control Panel
    2. Go to menu Settings -> Manage Products and Pricing -> Manage Prices
    3. On the succeeding Domains Pricing page, you will find a list of Top Level Domains (TLDs) that you have chosen to sell.
    4. Click the Edit button under the Manage Prices column, for the TLD whose pricing you need to set.
    5. **On the next page, set your selling prices as per your business requirements. You could:**
        - Set different prices for each action, i.e. Registration, Renewal, Transfer and Restore.
        - Set reducing price for each year to offer multi-year discounts. This way, you can offer increased discounts as and when your Customers register / renew domain names for more than the typical one year.
        - Set discounts for the minimum duration and nominal prices for every subsequent tenure, etc.

1.1.4 Configure your Website

You may use our free ready-made private labeled websites for all your retail business.

It is recommended that you undertake the following on your private labeled websites to improve your Customer's experience:

- Brand the URL
- Customize the Header and Footer
- Specify the Additional Payment Options for your Customers/Sub-Resellers

To manage your website:

1. Login to your Reseller Control Panel
2. Go to menu Settings -> Branding Settings
3. You could modify the logo, social media accounts and web contents from the branding settings page.

1.1.5 Use our REST APIs to integrate with your own website

If you are already selling several other Products and Services to your existing clients through another interface/Control Panel and do not want to add the burden of introducing the new Control Panels to them, you could use our REST APIs to integrate provisioning and management of these Products and Services in your existing interfaces and Control Panels.

See *REST APIs* page for more detail.

1.1.6 WHMCS Integrations

If you already using WHMCS billing software, you could easily install our registrar module to integrate liquid to your whmcs billing software.

See *WHMCS Integrations* page for more detail.

1.1.7 Configure your Payment Collection Options

Currently we provide 2 payment methods, wire transfer and paypal (including Credit card, amex, etc via paypal). Please setup the payment collection options:

1. Login to your Reseller Control Panel
2. Go to menu Settings -> Payment Gateway
3. Click Edit button on the right.
4. Complete the payment gateway configuration form, and click save.
5. Repeat number 3 and 4 to modify other payment gateway.

1.1.8 Create Company Users

Create independent Control Panel logins for your staff in various departments (like Sales, Billing, Support), to enable them to effectively manage your business.

Thus, you can enable the Billing User to List Transactions, Add Funds, etc., but not be able to changes the Name Servers of the domain name and so on. In effect, User creation would help you to assign fixed roles to persons of your company and assigning them different responsibilities to reduce your own workload.

You can add Company Users from within the Reseller Control Panel as mentioned below:

1. Login to your Reseller Control Panel
2. In the menu, click Settings -> User Management
3. Click Add User Button
4. Complete the form, and click Save button

1.2 Creating Demo Reseller Account

To understand and evaluate the interfaces and API we provide, you can use the Demo interface.

To create a fresh Demo Reseller account:

1. Visit https://reseller1.domainsas.com/manage/

2. Click Reseller Sign Up Tab, and fill up the form, and follow the steps thereafter to set up your Reseller account (make sure you provide an accurate email address).
3. Please wait for 24 hours for approval from our administrator. You will receive notifications email when your account activated.
4. Login to your Demo Reseller Account from https://reseller1.domainsas.com/manage/
5. You might wish to start with checking the flexibility and features of the system by referring to the Reseller Account Startup Guide.

Note: The demo account may not function at all times, since some beta testing is also done on the demo account. Also, the demo account is reset every day. All orders placed in the demo account would be automatically deleted every 24 hours.

1.3 REST APIs

1.3.1 Introductions

The Liquid API is organized around REST. Our API is designed to have predictable, resource-oriented URLs and to use HTTP response codes to indicate API errors. We use built-in HTTP features, like HTTP authentication and HTTP verbs, which can be understood by off-the-shelf HTTP clients, and we support cross-origin resource sharing to allow you to interact securely with our API from a client-side web application (though you should remember that you should never expose your secret API key in any public website's client-side code). JSON will be returned in all responses from the API, including errors.

To make the Liquid API as explorable as possible, we provide demo reseller account. Please create reseller demo account here to test your code or applications. Data created within demo reseller account will never cost any money and will not create real domain. Read the *Creating Demo Reseller Account* documentation.

1.3.2 Authentication

You authenticate to the Liquid API by providing one of your API keys in the request and your reseller ID. You can manage your API keys from your reseller control panel. You can have multiple API keys active at one time. Your API keys carry many privileges, so be sure to keep them secret! Make sure only use reseller demo account when developing your codes or applications.

Authentication to the API occurs via HTTP Basic Auth. Provide your reseller ID as the basic auth username and API key as password.

All API requests must be made over HTTPS. Calls made over plain HTTP will fail. You must authenticate for all requests.

1.3.3 Errors

Liquid uses conventional HTTP response codes to indicate success or failure of an API request. In general, codes in the 2xx range indicate success, codes in the 4xx range indicate an error that resulted from the provided information (e.g. a required parameter was missing, etc.), and codes in the 5xx range indicate an error with Liquid's servers.

Attributes

type The type of error returned. Can be unauthorized, invalid_request, or card_error.

message A human-readable message giving more details about the error.

code Programming readable error code, more details than type. For example: invalid_argument.

1.3.4 Pagination

All top-level Liquid API resources have support for bulk fetches — "list" API methods. For instance you can list domains, list contacts, list customers, list sub-resellers and list transactions. These list API methods share a common structure.

Liquid utilizes standard pagination, using the parameter limit and page_no, see below:

Arguments

limit A limit on the number of objects to be returned. Limit can range between 1 and 100 items.

page_no Page number, should be positive number.

For example, to retrieve items number 11 to 20, you would request with argument limit=10 and page_no=2

Response Format

body response

The body of the response will contain array of the data requested.

Full body response example:

```
[
  {
    "customer_id": "111111150843",
    "reseller_id": "111112",
    "status": "Active",
    "email": "example1@example1.com"
  },
  {
    "customer_id": "11111150842",
    "reseller_id": "11111112",
    "status": "Active",
    "email": "example2@example2.com"
  }
]
```

header response

- **link** link will contain list of paging urls (prev url, next url, first url, and last url). Example:

```
"Link": "<https://api.liqu.id/v1/customers?limit=10&page_no=3>; rel=\"next\",
<https://api.liqu.id/v1/customers?limit=10&page_no=5077>; rel=\"last\",
<https://api.liqu.id/v1/customers?limit=10&page_no=1>; rel=\"first\",
<https://api.liqu.id/v1/customers?limit=10&page_no=1>; rel=\"prev\""
```

- **X-Total-Count** total data available from the request. Example:

```
"X-Total-Count": "50763"
```

Full header response example:

```
{
 "Date": "Fri, 29 May 2015 16:32:12 GMT",
 "Content-Encoding": "gzip",
 "Server": "nginx",
 "Link": "<https://api.liqu.id/v1/customers?limit=10&page_no=3>; rel=\"next\",
↪<https://api.liqu.id/v1/customers?limit=10&page_no=5077>; rel=\"last\", <https://
↪api.liqu.id/v1/customers?limit=10&page_no=1>; rel=\"first\", <https://api.liqu.id/
↪v1/customers?limit=10&page_no=1>; rel=\"prev\"",
 "X-Frame-Options": "SAMEORIGIN",
 "Vary": "Accept-Encoding",
 "Content-Type": "application/json",
 "Transfer-Encoding": "chunked",
 "Connection": "keep-alive",
 "Keep-Alive": "timeout=5",
 "X-Xss-Protection": "1; mode=block",
 "X-Total-Count": "50763"
}
```

1.3.5 Rate Limiting

The API is provided to you to conduct normal business activities through your own interface(s). Any activity using Liquid API, that causes lossage or creates service degradation for other users, is constituted as abuse by Liquid. A few examples of API Abuse activities are stated below:

- Sending a huge number of Check Availability commands for already registered domain names, repeatedly.
- Adding a large number of Sub-Resellers and/or Customers who do not have any Orders.

Rate limiting in Liquid API is primarily considered on a per-reseller basis. Generally, you may make up to 100 API calls per 15 minutes. When you make more API calls than allowed, your reseller API Key will be temporary suspended, this suspension will not impact your reseller account.

Tips to avoid being Rate Limited

The tips below are there to help you code defensively and reduce the possibility of being rate limited.

- **Check Domain** Don't use Liquid API to check domain availability on your website interface.
- **Caching** Store API responses in your application or on your site if you expect a lot of use. For example, don't try to call the Liquid API on every page load of your website landing page. Instead, call the API infrequently and load the response into a local cache. When users hit your website load the cached version of the results.

1.3.6 REST API Requests

We provide API Console Tool for you to try out the API request and understand the responses.

- Demo Account API Requests
- Live Account Api Requests

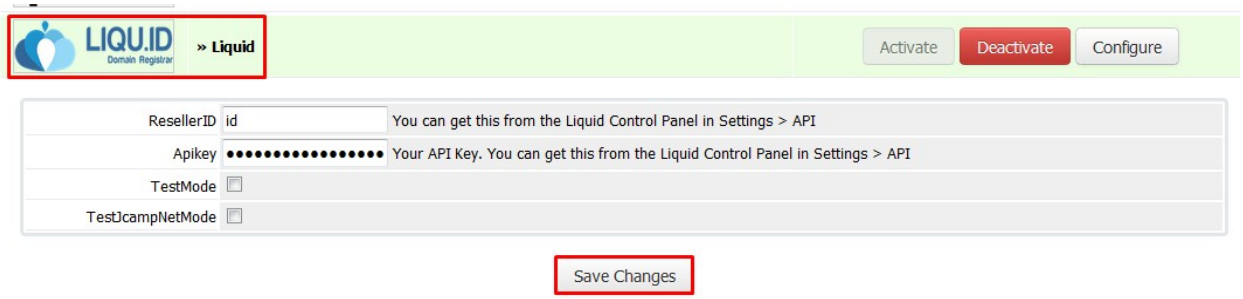
1.3.7 Feedback

If you find any issues with Liquid API, please use our ticketing support systems dedicated to Liquid API where we'll be available and actively listening to all of your feedback. We look forward to working with you and can't wait to see what everyone builds.

1.4 WHMCS Integrations

1.4.1 Liquid ResellerCamp WHMCS Registrar Module Installations

1. **Get the API key**
    a. Login to the ResellerCamp's reseller control panel (the url will be in the email you received when you signed up) and then go to Settings -> API.
    b. Please note your reseller ID at the bottom of the page.
    c. Click Add API Key button, enter the label and the IP address of the server where WHMCS is installed to authorize it for API access.
    d. On the same page, note down the API Key.
2. **Copy the whmcs module files**
    a. Download ResellerCamp's WHMCS Registrar Module
        - For PHP 7.2, Ioncube 10.2 (last updated 2020-06-02)
        - For PHP 7.1, Ioncube 10.2 (last updated 2018-07-20)
        - For PHP 5.6, Ioncube 10.2 (last updated 2018-05-17)
        - For PHP 7.2, Ioncube 5.6 (last updated 2017-10-05)
        - For PHP 5.6, Ioncube 5.5 (last updated 2017-10-05)
    b. Extract the zip files to /YourLocalPath/whmcs/modules/registrars
    c. Remember to replace "/YourLocalPath" with the actual location where you installed WHMCS.
3. **Setup WHMCS Configuration**
    a. Now, login to your WHMCS Administration Area
    b. Go to Setup > Products/Services > Domain Registrars
    c. Choose "Liquid" in the registrar dropdown menu and enter both the Reseller ID and API Key noted above.
    d. Then click Save Changes



And that's it, WHMCS will now be able to communicate with your ResellerCamp account to automate domain registration & management for your customers.

1.4.2 Demo Mode

To use the ResellerCamp demo mode or test mode, it's not as simple as ticking the demo mode option in the configuration area. You must setup an account separately on the dedicated resellercamp's demo system. Read the *Creating Demo Reseller Account* documentation.

Next enter your demo account details under Setup > Domain Registrars > ResellerCamp. With the Test Mode checkbox ticked you can now place domain registration orders in WHMCS, the domains will appear on your demo ResellerCamp account but no domain will actually be registered and you will not be charged.

Note: Live nameservers created at the Registry will return a Nameserver is not a valid Nameserver error unless they are created/registered in the demo environment.

The demo control panel will try to check the validity of the nameservers in the demo platform and not on the Registry, so you must register the nameservers first before attempting any domain registrations on the demo platform.

1.4.3 Synchron domain WHMCS with LIQUID

Synchron status Transfer in, expired date and status

This Feature is already included with WHMCS and it's disabled by default. By enabling this feature, any domain activity such as status, expiry, and transfer status will sync with data in liquid. The following step to enable this feature are :

- **Setup > General Settings > Domains**
  - Checklist domain Sync Enabled
  - Don't choose Sync Notify Only
- **Add cronjob** 0 0 * * * php -q /YourPathWHMCS/crons/domainsync.php that Cronjob will call skrip crons/domainsync.php once a day. each called will have 50 domains to be synced in scrolling, if all domains have been synced then it will start all over again.

Synchron status Transfer out

LIQUID provide a cron that allows the domain already transferred out to update the status to Expired, the steps :

- Download liquid cron here
  - For PHP 7, Ioncube 10.2 (last update 2018-05-17)
  - For PHP 5.6, Ioncube 10.2 (last update 2018-05-17)

  - For PHP 7, Ioncube 5.6 (last update 2017-10-17)
  - For PHP < 7, Ioncube 5.5 (last update 2017-10-17)

- Move the downloaded liquid folder to /YourPathWHMCS/crons/
- Rename /YourPathWHMCS/crons/liquid/config.sample.php become /YourPathWHMCS/crons/liquid/config.php and then set this part :
  - Database connection

```
$lq_cron_db = array(
        'host'     => 'localhost',
        'username' => 'username',
        'password' => 'password',
        'db'       => 'databaseName',
);
```

  - Set liqu.id account

```
$lq_cron_registrar = array(
    'liquid' => array(
        'api_url'     => 'https://api.liqu.id/v1/',
        'reseller_id' => '',
        'api_key'     => '',
    ),
    'resellercampid' => array(
        'api_url'     => 'https://api.liqu.id/v1/',
        'reseller_id' => '', // If you have account manage under␣
↪resellercamp.id
        'api_key'     => '',
    ),
);
```

  - Add Cronjob

    0 0 * * * php -q /YourPathWHMCS/crons/liquid/synctransferout.php Once a day call the script crons/liquid/synctransferout.php to chek poll message, if there is a domain transfer out from liqu.id the status will change become expired. You can view log syncron at /YourPathWHMCS/crons/liquid/report/synctransferout-Y-m-d.log.

1.4.4 WHMCS Addon - LIQUID PANDI Document Management Module

This module provides tools for registrar / domain resellers who use LIQUID software. This module is used for document management of domain registration requirements in PANDI. The required documents can be uploaded using the WHMCS member area and can be managed by registrar/reseller through the WHMCS admin page. Documents that are already uploaded will be automatically sent through URL api.liqu.id thus making the approval process easier.

WHMCS Addon Installation for Uploading Documents to Liquid

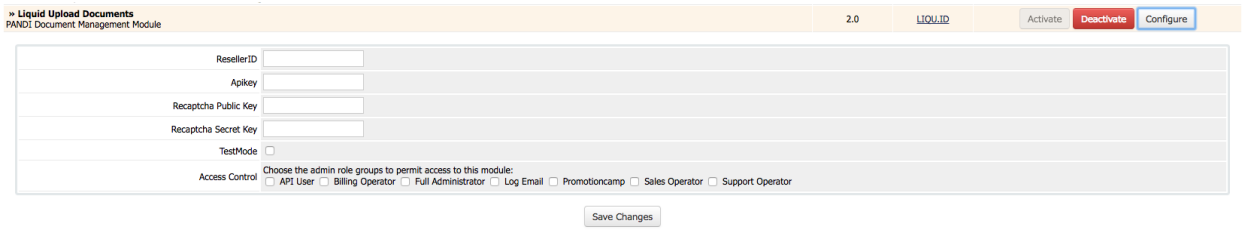
Before you start, please download WHMCS AddOn Module Document Upload below :

- For PHP 7.2, Ioncube 10.2 (updated at 2019-04-29)
- For PHP 7, Ioncube 5.6
- For PHP < 7, Ioncube 5.5

1. Create a new folder and name it documents in whmcs installed.

2. Copy the addon files to folder modules/addons.
3. Login.
4. Go to Setup menu > Addon Modules, then activate addon.

5. Configure addon, enter reseller id, apikey, and google recaptcha key (public & server). How to get a google recaptcha key can be seen here.



6. Modify template to add links to Document Upload page.

Note: Document Upload page can only be accessed in

http://domainname.com/index.php?m=liquid_upload_document&domain_id=ID_Domain

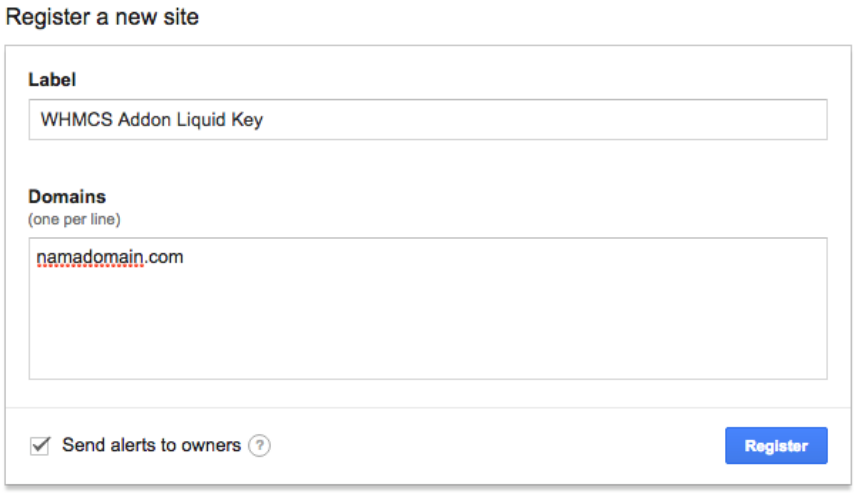
Link directing to the page can also be added in file clientareadomaindetails.tpl in folder templates/{active_template}/

Example of link code:

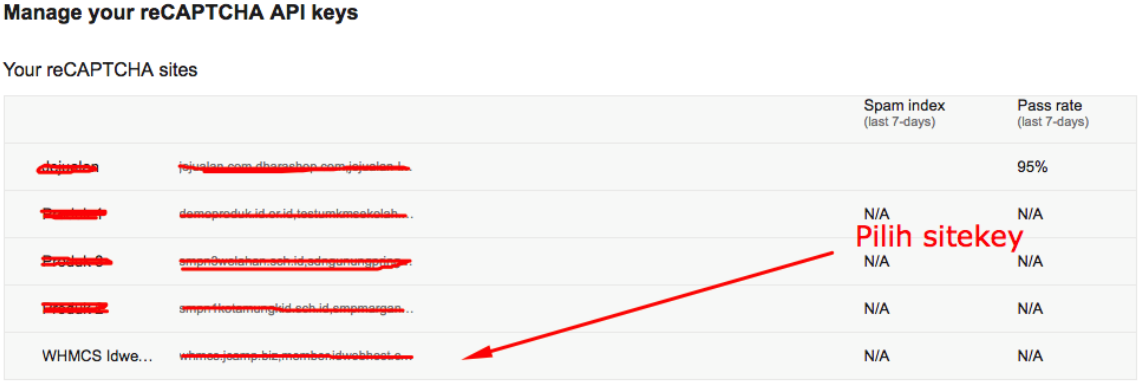
Upload Document

How to get Google Recaptcha Key

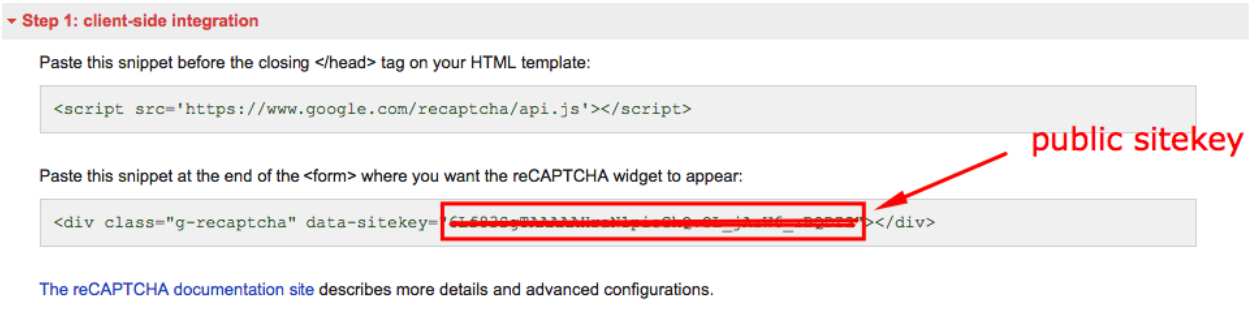
1. Go to https://www.google.com/recaptcha/admin#list
2. Create a new sitekey



3. Select a sitekey



4. Take public key and secret key



Note: Since this addon uses a recaptcha service from google, you will need a captcha key you can get after registering in recaptcha.

1.4.5 WHMCS Addon - LIQUID DNSSec Management Module

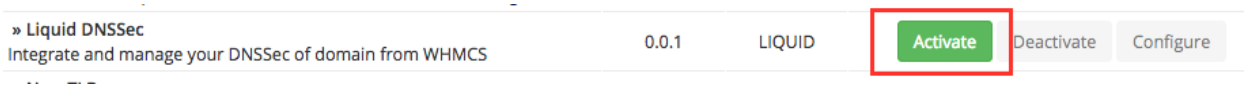
This module provides tools for registrars/domain resellers who use LIQUID software. This software is used for DNSSec data management with LIQUID module registrar. To use this module, make sure that you already use LIQUID module registrar on the domain you register. Only domains that are already registered in LIQUID can use this DNSSec module.

WHMCS Addon Installation for DNSSec Management

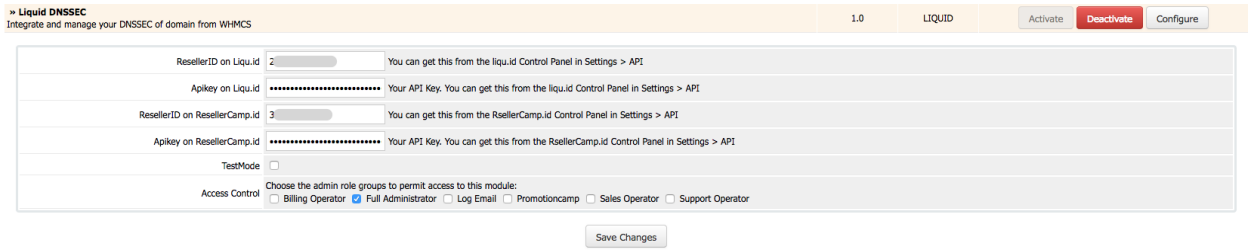
Before you start, please download WHMCS AddOn Module Liquid DNSSec Management:

- For PHP 7.2, Ioncube 10.2 (updated at 2018-08-13)
- For PHP 7.1, Ioncube 10.2 (updated at 2019-09-27)
- For PHP 7.2, Ioncube 5.6 (updated at 2018-08-13)
- For PHP 5.6, Ioncube 5.5 (updated at 2018-08-13)

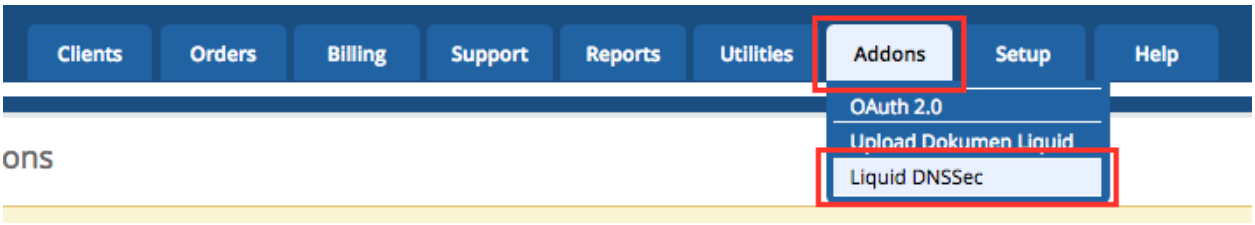
1. Extract the files you just downloaded.
2. Copy folder liquiddnssec from folder modules/addons to folder modules/addons in whmcs user directory.
3. Login as administrator.
4. Go to Setup menu > Addon Modules, and then activate addon.



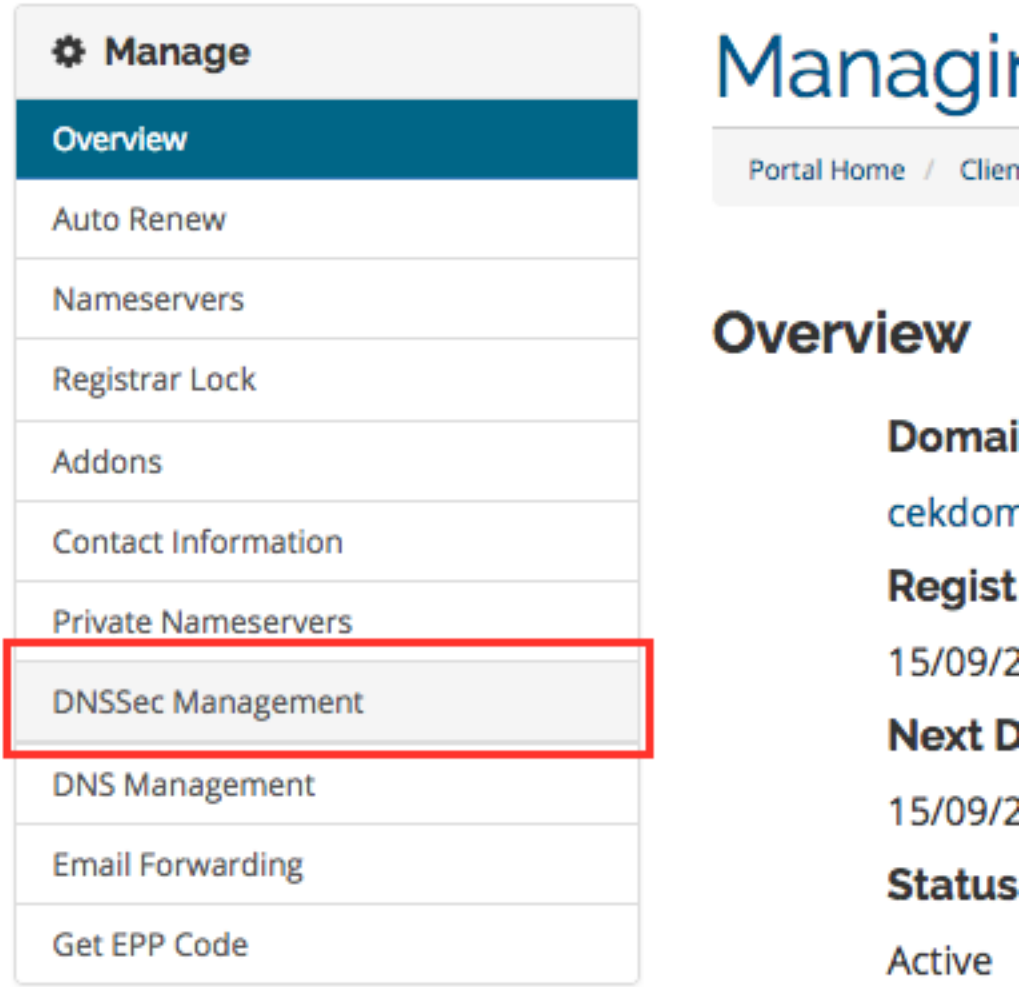
5. Configure addon, enter Reseller ID and Apikey



6. After the installation and configuration is done, Liquid DNSSec submenu will appear in Addons menu on the admin page.



7. DNSSec Management submenu will also appear on the sidebar of Manage menu on the client domain details page.



Demo Mode of DNSSec Management

Enter your demo account details under Setup menu > Addon Modules, in Liquid DNSSec. With the Test Mode checkbox ticked you can now manage your dnssec domain in WHMCS, the dnssec domain will appear on your demo ResellerCamp account but no domain will actually be registered and you will not be charged.

1.4.6 WHMCS Addon - LIQUID PANDI Premium Domain Management Module

This module provides tools for registrar / domain resellers who use LIQUID software. This module is used for management of PANDI Premium domain.

WHMCS Addon Installation for PANDI Premium Domain

Before you start, please download WHMCS AddOn Module PANDI Premium Domain below :

- For WHMCS 7.6 PHP 7.1 (updated at 2020-04-29)
- For WHMCS 7.6 PHP 5.6 (updated at 2020-04-29)

1. Set min length restriction domain and max length restriction domain according to tld premium domain which will be registered in configuration.php file.

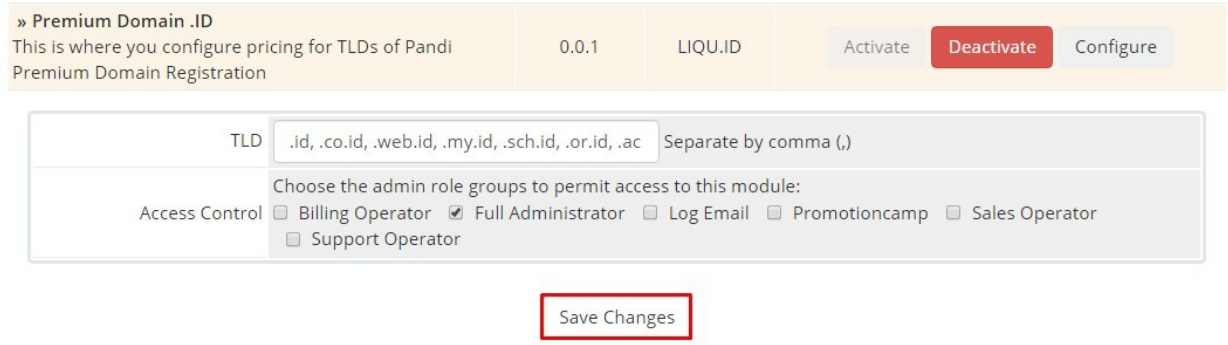
```
Example : $DomainMinLengthRestrictions[".co.id"] = 2;
          $DomainMaxLengthRestrictions[".co.id"] = 63;
```

2. Extract the files you just download.
3. Copy the addons file to folder modules/addons.
4. Copy this template file according folder structure. The original file should be backup first.

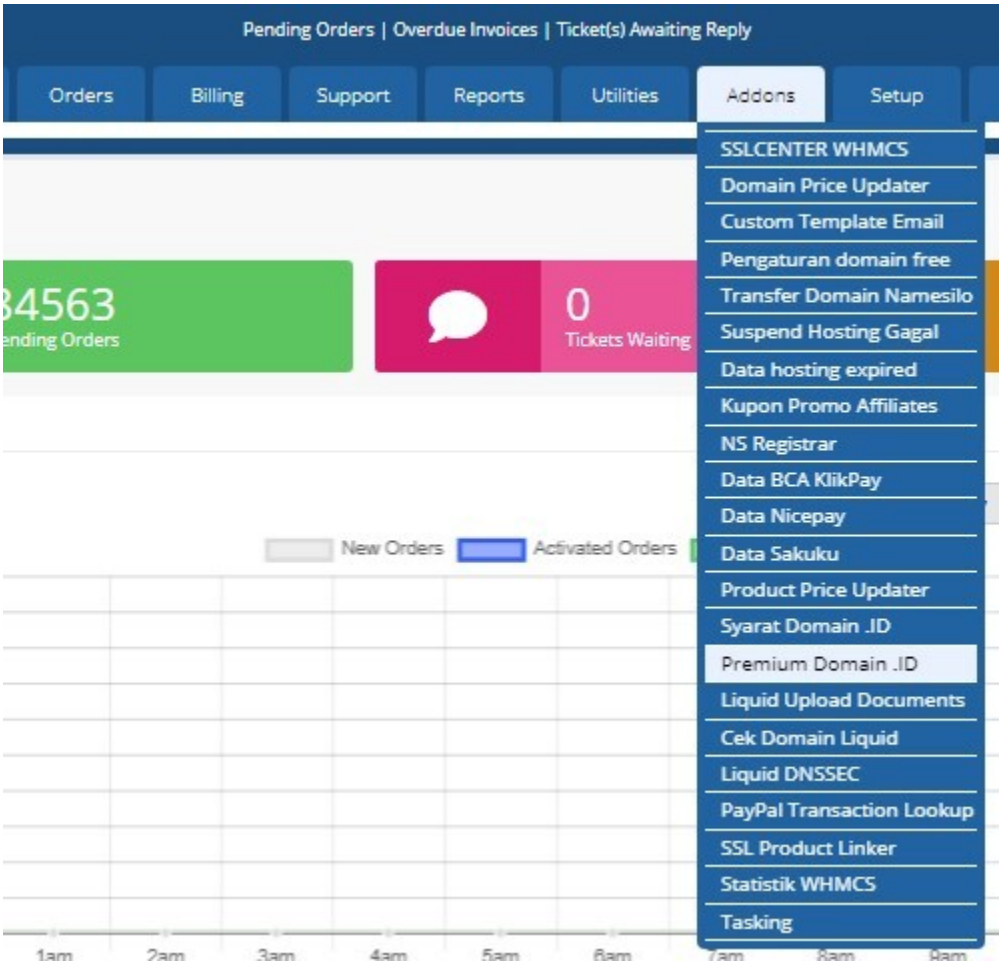
```
a. /templates/orderforms/standart_cart/configureproductdomain.tpl
b. /templates/orderforms/standart_cart/domainregister.tpl
c. /templates/orderforms/standart_cart/viewcart.tpl
d. /templates/orderforms/standart_cart/common.tpl
```

5. Login to your WHMCS Administration Area.
6. Go to Setup menu > Addons Modules, and then activate addons.

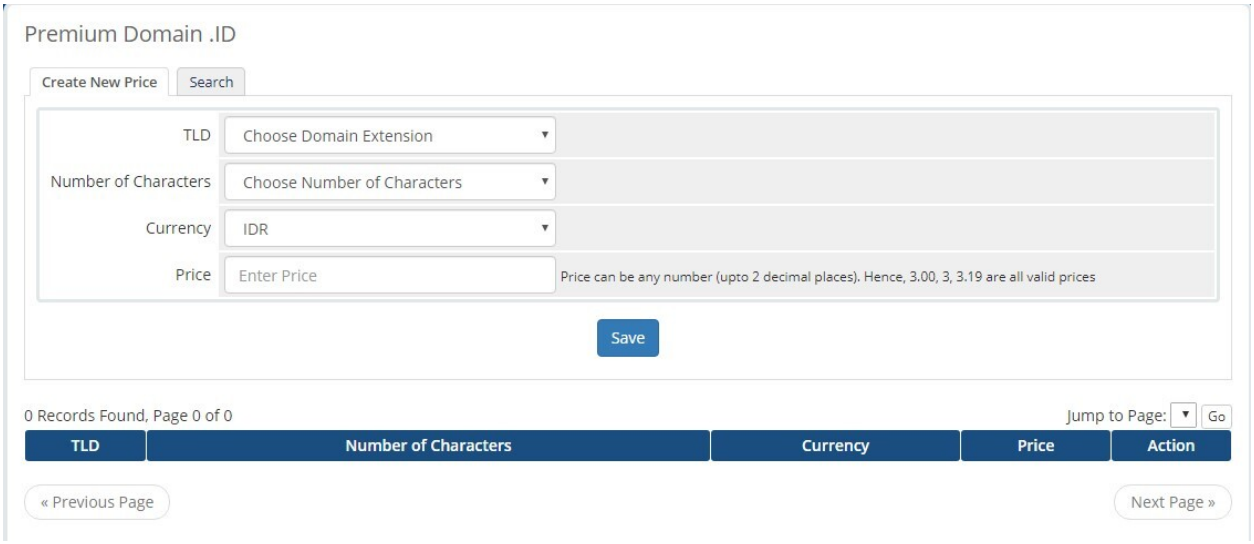
7. Configure addon, Enter TLD and choose role user who can access this addons.



8. After the installation and configuration is done, Premium Domain .ID submenu will appear in Addons menu on the admin page.



9. And then, set domain TLD, currency, and enter price.



Example :

10. When check premium domain availibilty at whmcs, it will show premium price.

1.4.7 Feedback

If you find any issues with Resellercamp's WHMCS registrar module, please use our ticketing support systems where we'll be available and actively listening to all of your feedback.

1.5 Blesta Integrations

1.5.1 Liquid ResellerCamp Blesta Registrar Module Installations

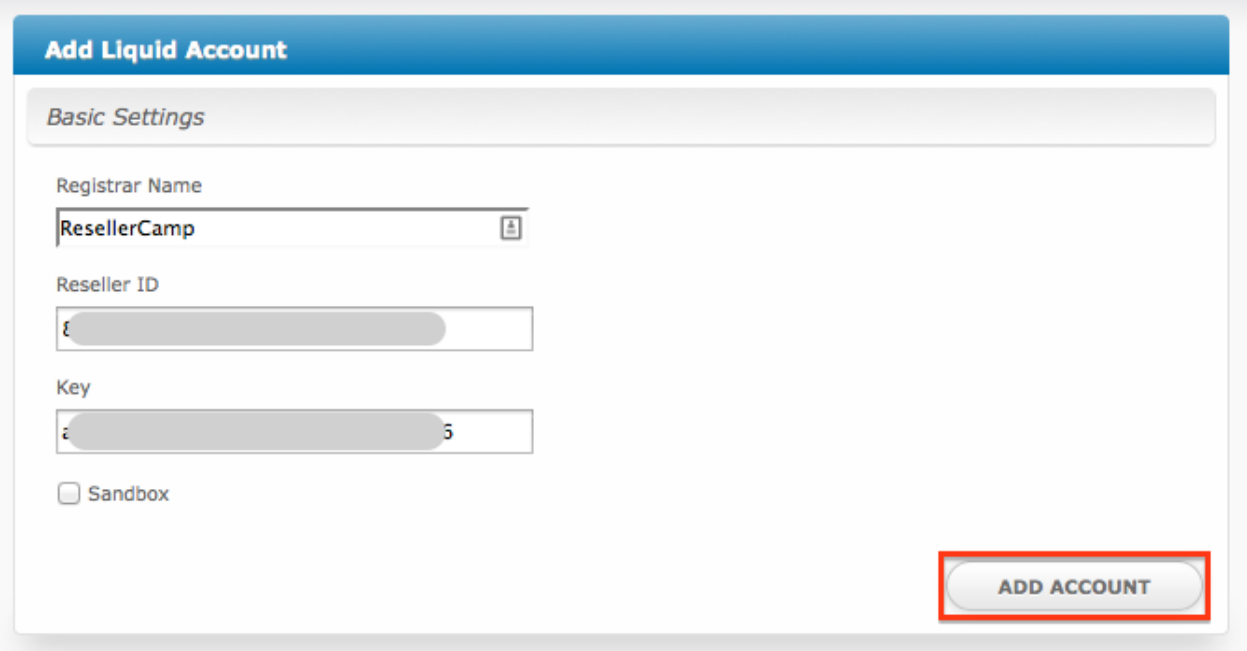
1. **Get the API key**
    a. Login to the ResellerCamp's reseller control panel (the url will be in the email you received when you signed up) and then go to Settings -> API.
    b. Please note your reseller ID at the bottom of the page.
    c. Click Add API Key button, enter the label and the IP address of the server where Blesta is installed to authorize it for API access.
    d. On the same page, note down the API Key.
2. **Copy the Blesta module files**
    a. Download ResellerCamp's Blesta Registrar Module here.
    b. Extract the zip files to /YourLocalPath/components/modules
    c. Remember to replace "/YourLocalPath" with the actual location where you installed Blesta.
3. **Setup Blesta Configuration**
    a. Now, login to your Blesta Administration Area
    b. Go to Setting and then choose menu Modules
    c. Choose Modules - Available

d. Choose "Liquid" in the Available Modules. Click Install

e. Go to menu Modules - Installed. Click Manage and then Add Account

f. Enter both the Reseller ID and API Key noted below

g. Then click Add Account



And that's it, Blesta will now be able to communicate with your ResellerCamp account to automate domain registration & management for your customers.

1.5.2 Demo Mode

To use the ResellerCamp demo mode or test mode, it's not as simple as ticking the demo mode option in the configuration area. You must setup an account separately on the dedicated resellercamp's demo system. Read the *Creating Demo Reseller Account* documentation.

Next enter your demo account details under Setting > Modules > Liquid. With the Sandbox checkbox ticked you can now place domain registration orders in Blesta, the domains will appear on your demo ResellerCamp account but no domain will actually be registered and you will not be charged.

Note: Live nameservers created at the Registry will return a Nameserver is not a valid Nameserver error unless they are created/registered in the demo environment.

The demo control panel will try to check the validity of the nameservers in the demo platform and not on the Registry, so you must register the nameservers first before attempting any domain registrations on the demo platform.

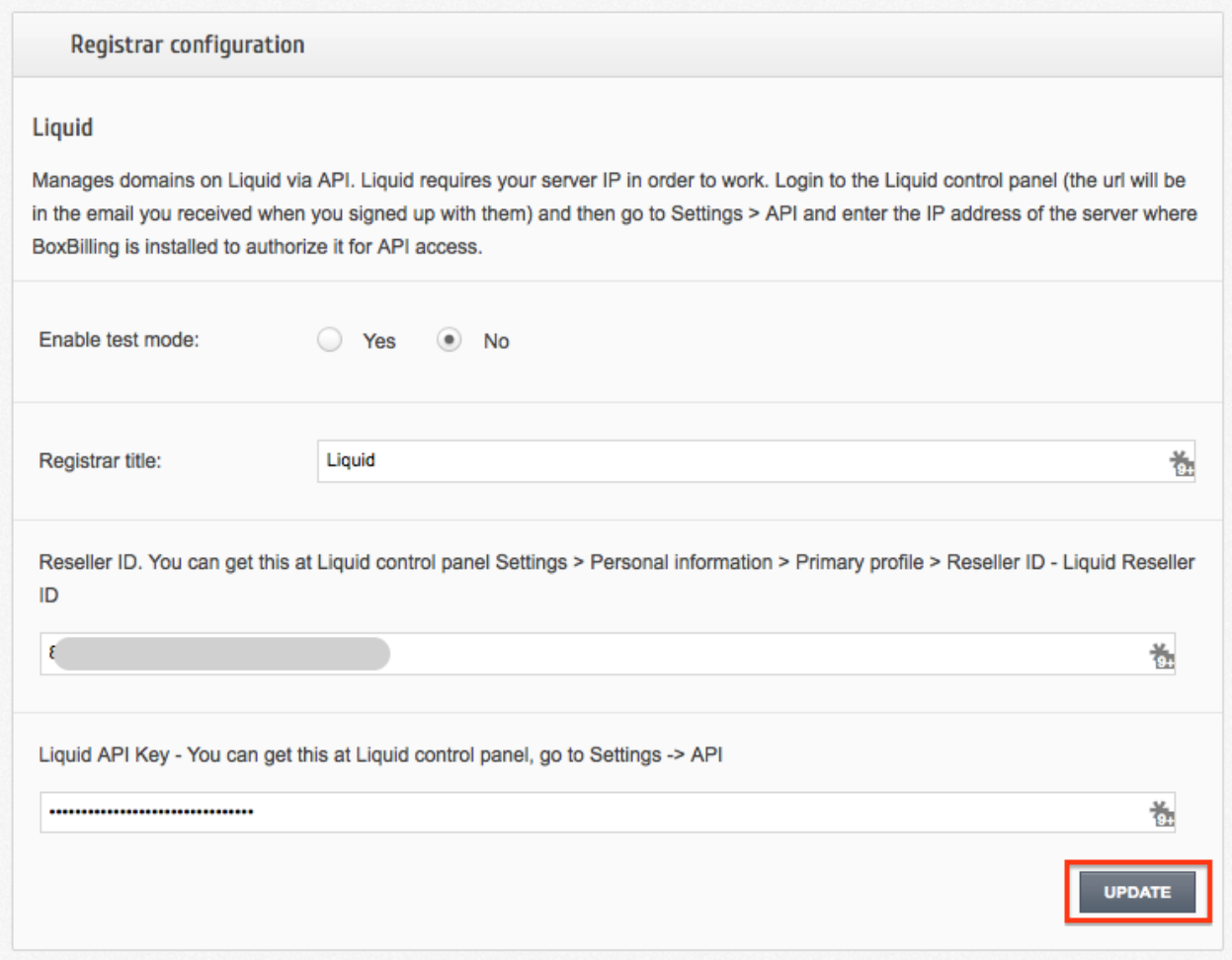
1.5.3 Feedback

If you find any issues with Resellercamp's Blesta registrar module, please use our ticketing support systems where we'll be available and actively listening to all of your feedback.

1.6 Boxbilling Integrations

1.6.1 Liquid ResellerCamp Boxbilling Registrar Module Installations

1. **Get the API key**
    a. Login to the ResellerCamp's reseller control panel (the url will be in the email you received when you signed up) and then go to Settings -> API.
    b. Please note your reseller ID at the bottom of the page.
    c. Click Add API Key button, enter the label and the IP address of the server where Boxbilling is installed to authorize it for API access.
    d. On the same page, note down the API Key.
2. **Copy the Boxbilling module files**
    a. Download ResellerCamp's Boxbilling Registrar Module here.
    b. Extract the zip files to /YourLocalPath/bb-library/Registrar/Adapter
    c. Remember to replace "/YourLocalPath" with the actual location where you installed Boxbilling.
3. **Setup Boxbilling Configuration**
    a. Now, login to your Boxbilling Administration Area
    b. Go to Configuration and then choose menu Domain Registration
    c. Choose New domain registrar
    d. Choose "Liquid" in the Available Modules. Click Install
    e. Go to menu Registrar. Click Edit with Pencil icon
    f. Enter both the Reseller ID and Liquid API Key noted below
    g. Then click Update



And that's it, Boxbilling will now be able to communicate with your ResellerCamp account to automate domain registration & management for your customers.

1.6.2 Demo Mode

To use the ResellerCamp demo mode or test mode, it's not as simple as ticking the demo mode option in the configuration area. You must setup an account separately on the dedicated resellercamp's demo system. Read the *Creating Demo Reseller Account* documentation.

Next enter your demo account details under Setting > Modules > Liquid. With the Sandbox checkbox ticked you can now place domain registration orders in Boxbilling, the domains will appear on your demo ResellerCamp account but no domain will actually be registered and you will not be charged.

Note: Live nameservers created at the Registry will return a Nameserver is not a valid Nameserver error unless they are created/registered in the demo environment.

The demo control panel will try to check the validity of the nameservers in the demo platform and not on the Registry, so you must register the nameservers first before attempting any domain registrations on the demo platform.

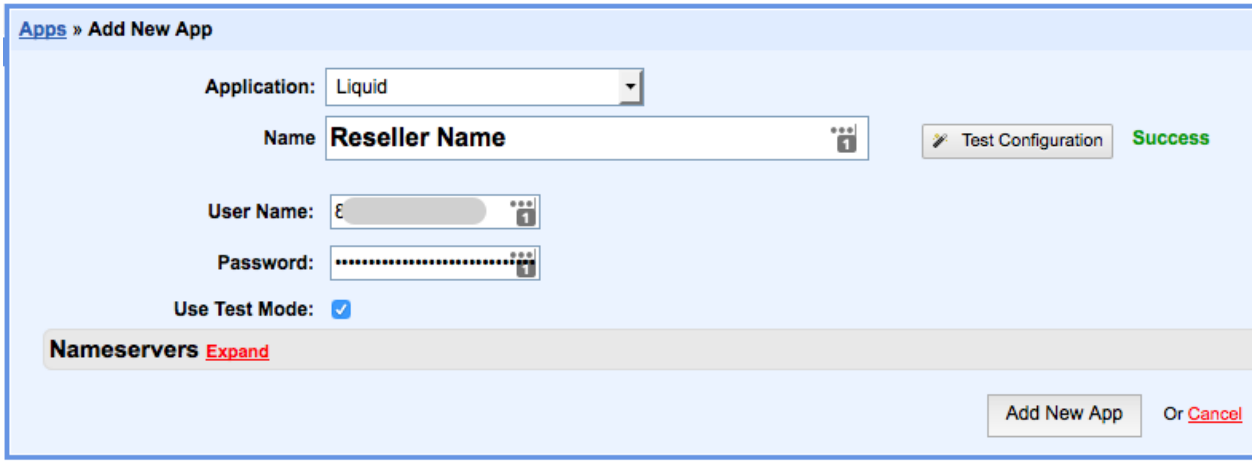
1.6.3 Feedback

If you find any issues with Resellercamp's Boxbilling registrar module, please use our ticketing support systems where we'll be available and actively listening to all of your feedback.

1.7 HostBill Integrations

1.7.1 Liquid ResellerCamp HostBill Registrar Module Installations

1. **Get the API key**
    a. Login to the ResellerCamp's reseller control panel (the url will be in the email you received when you signed up) and then go to Settings -> API.
    b. Please note your reseller ID at the bottom of the page.
    c. Click Add API Key button, enter the label and the IP address of the server where HostBill is installed to authorize it for API access.
    d. On the same page, note down the API Key.
2. **Copy the HostBill module files**
    a. Download 'ResellerCamp's HostBill Registrar Module:
        - For PHP 7.2 (last updated 2020-06-30)
        - For PHP 5.6 (last updated 2020-07-23)
        - For PHP 5.3
    b. Extract the zip files to /YourLocalPath/includes/modules/Domain
    c. Remember to replace "/YourLocalPath" with the actual location where you installed HostBill.
3. **Setup HostBill Configuration**
    a. Now, login to your HostBill Administration Area
    b. Go to Settings > Modules > Domain Modules
    c. Go to Inactive tab, then choose "Liquid" in the Modules. Click Activate
    d. Then will go to Add New App Menu.
    e. Make sure Application is selected Liquid, then enter Name and both the User Name dan Password with Reseller ID and API Key
    f. Then click Test Configuration, until display the success text at his side
    g. Then click Add New App



And that's it, HostBill will now be able to communicate with your ResellerCamp account to automate domain registration & management for your customers.

1.7.2 Demo Mode

To use the ResellerCamp demo mode or test mode, it's not as simple as ticking the demo mode option in the configuration area. You must setup an account separately on the dedicated resellercamp's demo system. Read the *Creating Demo Reseller Account* documentation.

Next enter your demo account details under Settings > Apps > Name of your Reseller. With the Test Mode checkbox ticked you can now place domain registration orders in HostBill, the domains will appear on your demo ResellerCamp account but no domain will actually be registered and you will not be charged.

Note: Live nameservers created at the Registry will return a Nameserver is not a valid Nameserver error unless they are created/registered in the demo environment.

The demo control panel will try to check the validity of the nameservers in the demo platform and not on the Registry, so you must register the nameservers first before attempting any domain registrations on the demo platform.

1.7.3 Feedback

If you find any issues with Resellercamp's HostBill registrar module, please use our ticketing support systems where we'll be available and actively listening to all of your feedback.

1.8 PHP Integrations

Liquid ResellerCamp PHP HTTP Integrations APIs

1.8.1 API library

Libraries for the Liquid API can be downloaded in here.

1.8.2 API Endpoint

```
https://api.liqu.id/v1
```

1.8.3 Example Request Authentication

```
$apiClient = new \Liquid\Client\ApiClient();
```

You can set the default API key, or you can always pass a key directly to an object's constructor. Authentication is transparently handled for you in subsequent method calls.

A sample test API key is included in all the examples on this page, so you can test any example right away. To test requests using your account, replace the sample API key with your actual API key.

1.8.4 HTTP status code summary

200 - OK	Everything worked as expected.
400 - Invalid Request	The request was unacceptable, often due to missing a required parameter.
401 - Unauthorized	No or invalid authentication details are provided.
402 - Failed Request	The parameters were valid but the request failed.
404 - Not Found	The requested resource doesn't exist.

1.8.5 Handling errors

Our API libraries raise exceptions for many reasons, such as a failed charge, invalid parameters, authentication errors, and network unavailability. We recommend writing code that gracefully handles all possible API exceptions.

```
try {
    // Use Liquid's library to make requests...
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
}
```

1.8.6 Expanding Objects

Many objects contain the ID of a related object in their response properties. For example, a Charge may have an associated Customer ID. Those objects can be expanded inline with the expand request parameter. Objects that can be expanded are noted in this documentation. This parameter is available on all API requests, and applies to the response of that request only.

Example Request

```
$apiClient = new \Liquid\Client\ApiClient();

// set Host to https://api.domainsas.com/v1 if you want to use Liquid Demo
$apiClient->getConfig()->setHost('https://api.liqu.id/v1');
```

(continues on next page)

(continued from previous page)

```
// set Reseller ID here
$apiClient->getConfig()->setUsername('##1#');
// set Apikey here
$apiClient->getConfig()->setPassword('###21cs##');
```

1.8.7 Retrieving data using callApi()

Example Request to retrieve all domains with params:

```
$resourcePath = '/domains';
$method       = 'GET';
$formParams   = array();
$headerParams = array();

$queryParams['limit'] = 100;
$queryParams['tld']   = 'com';

try {
    list($response, $header) = $apiClient->callApi(
        $resourcePath,
        $method,
        $queryParams,
        $formParams,
        $headerParams
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:


```
[
    {
        "domain_id":"##1##",
        "domain_name":"#####testxyz.com",
        "approval_status_id":"1",
        "start_date":"2015-07-27 00:00:00",
        "end_date":"2016-07-27 00:00:00",
        "reseller_id":"##1#",
        "customer_id":"#1##",
        "tld_id":"#1",
        "order_status_id":"1",
        "lb_order_id":null,
        "is_migrate_cust":"0",
        "is_privacy":"0",
        "reg_contact_id":"##1",
```


(continues on next page)

(continued from previous page)


```
        "adm_contact_id":"##1",
        "bill_contact_id":"##1",
        "tech_contact_id":"##1",
        "is_lock":"0",
        "is_suspend":"0",
        "is_theft_protection":"1",
        "raaVerificationStatus":null,
        "raaVerificationStartTime":null,
        "auth_code":null,
        "attr":"[]",
        "last_update":null,
        "ns1":"##demo1.#parkir##.net",
        "ns2":"##demo2.#parkir##.net",
        "ns3":"",
        "ns4":null,
        "ns5":null,
        "ns6":null,
        "ns7":null,
        "ns8":null,
        "ns9":null,
        "ns10":null,
        "ns11":null,
        "ns12":null,
        "ns13":null,
        "customer_name":"Customer Demo PHP",
        "expiry_date":"2016-07-27 00:00:00"
    },
    {
        "domain_id":"###1#",
        "domain_name":"domain#####123.com",
        "approval_status_id":"1",
        "start_date":"2015-07-27 00:00:00",
        "end_date":"2016-07-27 00:00:00",
        "reseller_id":"##1#",
        "customer_id":"#1##",
        "tld_id":"9#",
        "order_status_id":"1",
        "lb_order_id":null,
        "is_migrate_cust":"0",
        "is_privacy":"0",
        "reg_contact_id":"#1#",
        "adm_contact_id":"#1#",
        "bill_contact_id":"#1#",
        "tech_contact_id":"#1#",
        "is_lock":"0",
        "is_suspend":"0",
        "is_theft_protection":"1",
        "raaVerificationStatus":null,
        "raaVerificationStartTime":null,
        "auth_code":null,
        "attr":"[]",
        "last_update":null,
        "ns1":"##demo1.#parkir##.net",
        "ns2":"##demo2.#parkir##.net",
        "ns3":"",
        "ns4":null,
        "ns5":null,
```


(continues on next page)

(continued from previous page)

```
        "ns6":null,
        "ns7":null,
        "ns8":null,
        "ns9":null,
        "ns10":null,
        "ns11":null,
        "ns12":null,
        "ns13":null,
        "customer_name":"Customer Demo PHP",
        "expiry_date":"2016-07-27 00:00:00"
    }
]
```

1.8.8 Creating data using callApi()

Example Request to create a new customer:

```
$resourcePath   = '/customers';
$method         = 'POST';
$queryParams    = array();
$headerParams   = array();

$formParams['email']          = 'demo##php###@em####il.com';
$formParams['name']           = 'Customer Demo PHP';
$formParams['password']       = '##21&^9##fA';
$formParams['company']        = 'Customer Demo PHP';
$formParams['address_line_1'] = 'Pajangan';
$formParams['city']           = 'Bantul';
$formParams['state']          = 'Yogyakarta';
$formParams['country_code']   = 'ID';
$formParams['zipcode']        = '55321';
$formParams['tel_cc_no']      = 62;
$formParams['tel_no']         = 85732#####;

try {
    list($response, $header) = $apiClient->callApi(
        $resourcePath,
        $method,
        $queryParams,
        $formParams,
        $headerParams
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:

```
{
    "customer_id":"##1##",
    "reseller_id":"#1##+",
    "status":"Active",
    "email":"demo##php###@em####il.com",
    "name":"Customer Demo PHP",
    "company":"Customer Demo PHP",
    "creation_date":"2015-11-09 07:40:29",
    "total_receipts":"0.00",
    "address_line_1":"Pajangan",
    "address_line_2":"",
    "address_line_3":"",
    "city":"Bantul",
    "state":"Yogyakarta",
    "country_code":"ID",
    "country_name":"Indonesia",
    "zipcode":"55321",
    "tel_cc_no":"62",
    "tel_no":"85732#####",
    "alt_tel_cc_no":null,
    "alt_tel_no":null,
    "mobile_cc_no":null,
    "mobile_no":null,
    "fax_cc_no":null,
    "fax_no":null,
    "lang_id":"English"
}
```


1.8.9 Updating data using callApi()

Example Request to update a customer:

```
$customer_id    = ##1##;
$resourcePath   = '/customers/' . $customer_id;
$method         = 'PUT';
$queryParams    = array();
$headerParams   = array();

$formParams['email']          = 'demo##php###@em####il.com';
$formParams['name']           = 'Update Customer Demo PHP';
$formParams['company']        = 'Update Customer Demo PHP';
$formParams['address_line_1'] = 'Pajangan';
$formParams['city']           = 'Bantul';
$formParams['state']          = 'Yogyakarta';
$formParams['country_code']   = 'ID';
$formParams['zipcode']        = '55321';
$formParams['tel_cc_no']      = 62;
$formParams['tel_no']         = 85732#####;

try {
    list($response, $header) = $apiClient->callApi(
        $resourcePath,
        $method,
        $queryParams,
        $formParams,
        $headerParams
```

(continues on next page)

(continued from previous page)

```
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:


```
{
    "customer_id":"##1##",
    "reseller_id":"#1##",
    "status":"Active",
    "email":"demo##php###@em####il.com",
    "name":"Update Customer Demo PHP",
    "company":"Update Customer Demo PHP",
    "creation_date":"2015-07-27 02:18:42",
    "total_receipts":"80.00",
    "address_line_1":"Pajangan",
    "address_line_2":"",
    "address_line_3":"",
    "city":"Bantul",
    "state":"Yogyakarta",
    "country_code":"ID",
    "country_name":"Indonesia",
    "zipcode":"55321",
    "tel_cc_no":"62",
    "tel_no":"85732#####",
    "alt_tel_cc_no":null,
    "alt_tel_no":null,
    "mobile_cc_no":null,
    "mobile_no":null,
    "fax_cc_no":null,
    "fax_no":null,
    "lang_id":"English"
}
```


1.8.10 Deleting data using callApi()

Example Request to delete a customer:

```
$customer_id  = ##1##;
$resourcePath = '/customers/' . $customer_id;
$method       = 'DELETE';
$queryParams  = array();
$headerParams = array();
$formParams   = array();
```

(continues on next page)

(continued from previous page)

```
try {
    list($response, $header) = $apiClient->callApi(
        $resourcePath,
        $method,
        $queryParams,
        $formParams,
        $headerParams
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:

```
{
    "customer_id":"##1##",
    "deleted":true
}
```

1.8.11 Retrieving data using available class api

Using DomainsApi() to retrieve all domains:

```
$apiDomains = new \Liquid\Client\Api\DomainsApi($apiClient);

$limit                      = 2;
$page_no                    = null;
$domain_id                  = null;
$reseller_id                = null;
$customer_id                = null;
$show_child_orders          = null;
$tld                        = null;
$status                     = null;
$domain_name                = null;
$privacy_protection_enabled = null;
$creation_time_start        = null;
$creation_time_end          = null;
$expiry_date_start          = null;
$expiry_date_end            = null;
$reseller_email             = null;
$customer_email             = null;
$exact_domain_name          = null;

try {
    list($response, $header) = $apiDomains->retrieve(
        $limit,
```

(continues on next page)

(continued from previous page)

```
        $page_no,
        $domain_id,
        $reseller_id,
        $customer_id,
        $show_child_orders,
        $tld,
        $status,
        $domain_name,
        $privacy_protection_enabled,
        $creation_time_start,
        $creation_time_end,
        $expiry_date_start,
        $expiry_date_end,
        $reseller_email,
        $customer_email,
        $exact_domain_name
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:


```
[
    {
        "domain_id":"#1###",
        "domain_name":"test##domain#####.com",
        "approval_status_id":"1",
        "start_date":"2015-07-27 00:00:00",
        "end_date":"2016-07-27 00:00:00",
        "reseller_id":"#1###",
        "customer_id":"##1##",
        "tld_id":"#9",
        "order_status_id":"1",
        "lb_order_id":null,
        "is_migrate_cust":"0",
        "is_privacy":"0",
        "reg_contact_id":"1",
        "adm_contact_id":"1",
        "bill_contact_id":"1",
        "tech_contact_id":"1",
        "is_lock":"0",
        "is_suspend":"0",
        "is_theft_protection":"1",
        "raaVerificationStatus":null,
        "raaVerificationStartTime":null,
        "auth_code":null,
        "attr":"[]",
```


(continues on next page)

(continued from previous page)


```
        "last_update":null,
        "ns1":"##demo1.#parkir##.net",
        "ns2":"##demo2.#parkir##.net",
        "ns3":"",
        "ns4":null,
        "ns5":null,
        "ns6":null,
        "ns7":null,
        "ns8":null,
        "ns9":null,
        "ns10":null,
        "ns11":null,
        "ns12":null,
        "ns13":null,
        "customer_name":"Customer Demo PHP",
        "expiry_date":"2016-07-27 00:00:00"
    },
    {
        "domain_id":"#1###",
        "domain_name":"Tets###Io.com",
        "approval_status_id":"1",
        "start_date":"2015-07-27 00:00:00",
        "end_date":"2016-07-27 00:00:00",
        "reseller_id":"#1###",
        "customer_id":"##1##",
        "tld_id":"1",
        "order_status_id":"1",
        "lb_order_id":null,
        "is_migrate_cust":"0",
        "is_privacy":"0",
        "reg_contact_id":"1",
        "adm_contact_id":"1",
        "bill_contact_id":"1",
        "tech_contact_id":"1",
        "is_lock":"0",
        "is_suspend":"0",
        "is_theft_protection":"1",
        "raaVerificationStatus":null,
        "raaVerificationStartTime":null,
        "auth_code":null,
        "attr":"[]",
        "last_update":null,
        "ns1":"##demo1.#parkir##.net",
        "ns2":"##demo2.#parkir##.net",
        "ns3":"",
        "ns4":null,
        "ns5":null,
        "ns6":null,
        "ns7":null,
        "ns8":null,
        "ns9":null,
        "ns10":null,
        "ns11":null,
        "ns12":null,
        "ns13":null,
        "customer_name":"Customer Demo PHP",
        "expiry_date":"2016-07-27 00:00:00"
```


(continues on next page)

(continued from previous page)

```
    }
]
```

1.8.12 Creating data using available class api

Using BillingApi() to add fund a reseller:

```
$apiBilling = new \Liquid\Client\Api\BillingApi($apiClient);

$reseller_id = #1###;
$amount      = 150;
$description = 'add fund from API';

try {
    list($response, $header) = $apiBilling->addFundReseller(
        $reseller_id,
        $amount,
        $description
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:

```
{
    "transaction_id":"1",
    "reseller_id":"#1###",
    "transaction_type":"deposit",
    "amount":150,
    "balance":350,
    "date":"2015-11-09 08:15:32",
    "description":"add fund from API",
    "status":"pending"
}
```

1.8.13 Updating data using available class api

Using ResellersApi() to update a reseller:

```
$apiReseller = new \Liquid\Client\Api\ResellersApi($apiClient);

$reseller_id      = #1###;
$email            = 'demo##php###@em####il.com';
```

(continues on next page)

(continued from previous page)

```
$name             = 'Reseller Demo PHP';
$company          = 'Reseller Demo PHP';
$address_line_1   = 'Pajangan';
$city             = 'Bantul';
$state            = 'Yogyakarta';
$country_code     = 'ID';
$zipcode          = '55321';
$tel_cc_no        = 62;
$tel_no           = 857######;
$selling_currency = 'USD';
$address_line_2   = null;
$address_line_3   = null;
$alt_tel_cc_no    = null;
$alt_tel_no       = null;
$mobile_cc_no     = null;
$mobile_no        = null;
$fax_cc_no        = null;
$fax_no           = null;

try {
    list($response, $header) = $apiReseller->updateReseller(
        $reseller_id,
        $email,
        $name,
        $company,
        $address_line_1,
        $city,
        $state,
        $country_code,
        $zipcode,
        $tel_cc_no,
        $tel_no,
        $selling_currency,
        $address_line_2,
        $address_line_3,
        $alt_tel_cc_no,
        $alt_tel_no,
        $mobile_cc_no,
        $mobile_no,
        $fax_cc_no,
        $fax_no
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:

```
{
    "reseller_id":"#1###",
    "parent_reseller_id":"#1##",
    "status":"Active",
    "email":"demo##php###@em####il.com",
    "name":"Reseller Demo PHP",
    "brand_name":"Reseller Demo PHP",
    "company":"Reseller Demo PHP",
    "creation_date":"2014-10-27 03:06:38",
    "total_receipts":"159.00",
    "address_line_1":"Pajangan",
    "address_line_2":"",
    "address_line_3":"",
    "city":"Bantul",
    "state":"Yogyakarta",
    "country_code":"ID",
    "country_name":"Indonesia",
    "zipcode":"55321",
    "tel_cc_no":"62",
    "tel_no":"857######",
    "alt_tel_cc_no":null,
    "alt_tel_no":null,
    "mobile_cc_no":null,
    "mobile_no":null,
    "fax_cc_no":null,
    "fax_no":null,
    "selling_currency":"USD",
    "accounting_currency":"USD",
    "parentselling_currency":"USD",
    "lang_id":"English"
}
```


1.8.14 Deleting data using available class api

Using ResellersApi() to delete a reseller:

```
$apiReseller = new \Liquid\Client\Api\ResellersApi($apiClient);

$reseller_id = #1##;

try {
    list($response, $header) = $apiReseller->delete_(
        $reseller_id
    );
} catch (Liquid\Client\ApiException $e) {
    echo 'Caught exception: ', $e->getMessage(), "\n";
    echo '<br>HTTP response headers: ', $e->getResponseHeaders(), "\n";
    echo '<br>HTTP response body: ', $e->getResponseBody(), "\n";
    echo '<br>HTTP status code: ', $e->getCode(), "\n";
    die;
}

// convert obj to array
$response = json_decode(json_encode($response), true);

print_r($response);
```

Example response:

```
{
    "reseller_id":"#1##",
    "deleted":true
}
```

1.8.15 Available class api list

AccountApi()	CustomerApi()	EmailforwardingApi()
BillingApi()	DnsApi()	PrivacyprotectionApi()
CommonApi()	DomainforwardingApi()	ResellersApi()
ContactsApi()	DomainsApi()	

1.8.16 Feedback

If you find any issues with Liquid Resellercamp's PHP integration APIs, please use our ticketing support systems where we'll be available and actively listening to all of your feedback.

1.9 FAQ

isinya