

Module Configuration

Modules in version 3.1+ support the use of a **config.json** file, that defines basic configuration settings for the module.

 Due to backwards compatibility with version 3.0, config.json files are not required but are **strongly recommended** as they may be required in future versions.

Configuration Format

Example

/components/modules/my_module/config.json

```
{
  "version": "1.0.0",
  "name": "My Module Name",
  "description": "A module like no other!",
  "type": "generic"
  "authors": [
    {
      "name": "Phillips Data, Inc.",
      "url": "http://www.blesta.com"
    }
  ],
  "package": {
    "name_key": "input_name_key"
  },
  "service": {
    "name_key": "service_field_key"
  },
  "module": {
    "row": "Signular Form of Module Row",
    "rows": "Plural Form of Module Row",
    "group": "Module Group Name",
    "row_key": "module_row_field_key"
  },
  "email_tags": {
    "module": ["tag_one", "tag_two"],
    "package": ["type", "package"],
    "service": ["service_field_key", "service_field_username"],
  }
}
```

Definition

Option	Description						
version	The version of the module. This should be a semantic version number.						
name	The name of the module. This may also be a language definition.						
description	A brief description of the module. This may also be a language definition in Blesta v4.9.0+.						
type	Added in Blesta v5. A tag defining the module type. Currently the only type relevant to Blesta's code base is "registrar" as these will be used by the Domain Manager plugin.						
authors	An array of author objects.<table><tr><th>Option</th><th>Description</th></tr><tr><td>name</td><td>The name of the author. This may be a language definition</td></tr><tr><td>url</td><td>The URL to link to for the author.</td></tr></table>	Option	Description	name	The name of the author. This may be a language definition	url	The URL to link to for the author.
Option	Description						
name	The name of the author. This may be a language definition						
url	The URL to link to for the author.						

logo	The path within the module to the logo file. If not set, defaults to views/default/images/logo.png .										
package	An object of service settings. <table> <tr> <th>Option</th><th>Description</th></tr> <tr> <td>name_key</td><td>The value of the input key in \$vars that represents the package's label before it becomes a service (e.g. if your module registers domains, this might be domain).</td></tr> </table>	Option	Description	name_key	The value of the input key in \$vars that represents the package's label before it becomes a service (e.g. if your module registers domains, this might be domain).						
Option	Description										
name_key	The value of the input key in \$vars that represents the package's label before it becomes a service (e.g. if your module registers domains, this might be domain).										
service	An object of service settings. <table> <tr> <th>Option</th><th>Description</th></tr> <tr> <td>name_key</td><td>The value of service_fields.key used to identify a service's label (e.g. if your module registers domains, this might be domain).</td></tr> </table>	Option	Description	name_key	The value of service_fields.key used to identify a service's label (e.g. if your module registers domains, this might be domain).						
Option	Description										
name_key	The value of service_fields.key used to identify a service's label (e.g. if your module registers domains, this might be domain).										
module	An object of module settings. <table> <tr> <th>Option</th><th>Description</th></tr> <tr> <td>row</td><td>The singular form of the noun used to describe a module row (e.g. if your module registers domain, this might be Registrar Account). This may also be a language definition.</td></tr> <tr> <td>rows</td><td>The plural form of the noun used to describe the module row (e.g. Registrar Accounts). This may also be a language definition.</td></tr> <tr> <td>group</td><td>The noun used to describe a collection of module rows (e.g. Registrar Accounts). This may also be a language definition.</td></tr> <tr> <td>row_key</td><td>The value of module_row_meta.key used to identify a module row.</td></tr> </table>	Option	Description	row	The singular form of the noun used to describe a module row (e.g. if your module registers domain, this might be Registrar Account). This may also be a language definition.	rows	The plural form of the noun used to describe the module row (e.g. Registrar Accounts). This may also be a language definition.	group	The noun used to describe a collection of module rows (e.g. Registrar Accounts). This may also be a language definition.	row_key	The value of module_row_meta.key used to identify a module row.
Option	Description										
row	The singular form of the noun used to describe a module row (e.g. if your module registers domain, this might be Registrar Account). This may also be a language definition.										
rows	The plural form of the noun used to describe the module row (e.g. Registrar Accounts). This may also be a language definition.										
group	The noun used to describe a collection of module rows (e.g. Registrar Accounts). This may also be a language definition.										
row_key	The value of module_row_meta.key used to identify a module row.										
email_tags	An object containing tags for the module, service, package, and any other tag groups, each of which contain an array of strings representing each tag in that group. These tags will be displayed when an admin configures the welcome email template.										

Using the Configuration

To use the configuration file you must include it within the constructor of your module using the [loadConfig\(\)](#) method.

/components/modules/my_module/my_module.php

```
<?php
class MyModule extends Module {
    public function __construct() {
        $this->loadConfig(dirname(__FILE__) . DS . "config.json");
    }
}
```

If your configuration uses any language definitions, be sure to include those before loading the config.

/components/modules/my_module/my_module.php

```
<?php
class MyModule extends Module {
    public function __construct() {
        Language::loadLang("my_module", null, dirname(__FILE__) . DS . "language" . DS);
        $this->loadConfig(dirname(__FILE__) . DS . "config.json");
    }
}
```