Superlists for osTicket

by Cartmega

05/06/2022

Aristotelis Joannou

Table of Contents

Open a New Ticket

Please fill in the form below to open a new ticket.

Contact Information

Email Address *: john@doe.com

Full Name *: John Doe

Phone Number: Ext:

Help Topic: SuperLists

SuperLists Form

Topic: SuperLists

Form: SuperLists Form

Field: -- SELECT TOPIC FORM FIELD --

-- SELECT TOPIC FORM FIELD --
sup_Topic
sup_Form
sup_Field

INTRODUCTION

Superlists is a plugin that allows for smart, dynamic, interlinked and grouped ajax custom lists in forms that appear on the ‘new ticket’ pages, both for clients and agents and are updated with ajax from any database.

Superlists are interlinked such that each list is fed the user selection of the previous list using ajax. Superlists can even be tailored to individual clients!

Every superlist is bound to a ticket field which collects the user selection that will eventually end up in the ticket created. Each superlist ticket field can individually collect either the key or value of the superlist selection as required.

- Requires 'osTicket Motherload' (a plugin controller for osTicket) which is included in this software bundle.
- Requires PHP version 7.2.x or higher. Proper operation cannot be guaranteed with earlier versions.
- No core file modifications necessary!

ONLINE DEMO

Did you know that there is an online demo of Superlists?
If you have not had a chance to try Superlists, you can check it out right now!

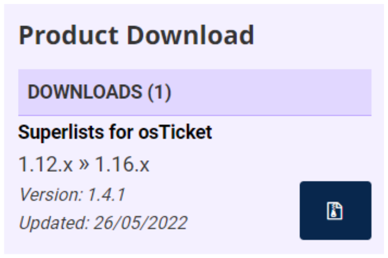
When you click the button below you will be taken
to the Cartmega demo website, where you can test drive Superlists.

INSTALLATION

Click the play button below to watch the installation video
or follow the instructions on the following page.

cartmega
AJAX SUPERLISTS
Superlists plugin for osTicket
(includes osTicket Motherload)
INTERLINKED, GROUPED, DYNAMIC, AJAX LISTS
FULLY PROGRAMMABLE!
E-COMMERCE SOLUTIONS
INSTALLATION ✓
CONFIGURATION ✓
TESTING ✓
Visit ostdemo.cartmega.com for a test drive.
www.cartmega.com

1. Superlists is available online here:
   https://www.cartmega.com/superlists-for-osticket.html

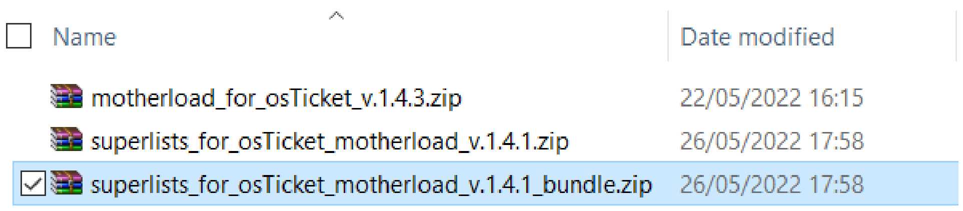


Product Download

DOWNLOADS (1)

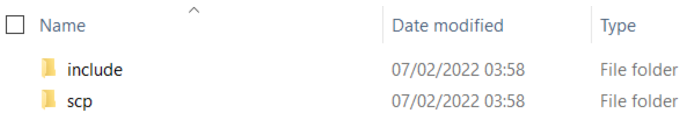
Superlists for osTicket
1.12.x » 1.16.x
Version: 1.4.1
Updated: 26/05/2022

2. Download & Unzip.
   a. Download our plugin. (superlists_for_osTicket_motherload_v.x.x.x_bundle.zip)
   b. Extract its contents onto your hard drive.

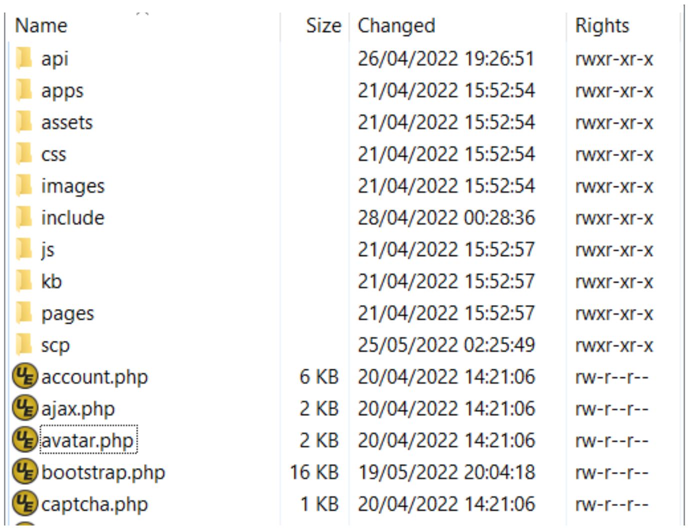


Name	Date modified
motherload_for_osTicket_v.1.4.3.zip	22/05/2022 16:15
superlists_for_osTicket_motherload_v.1.4.1.zip	26/05/2022 17:58
superlists_for_osTicket_motherload_v.1.4.1_bundle.zip	26/05/2022 17:58

3. Upload Motherload first.
   a. Extract the contents of motherload_for_osTicket_v.x.x.x.zip, onto your hard drive.
   b. Open the folder 'UPLOAD' where you just extracted the zip file.
   c. FTP into your osTicket installation & find the osTicket root directory.
   d. Select all files & folders within the ‘UPLOAD’ folder (on your computer) and upload all to the osTicket root directory of your FTP server.

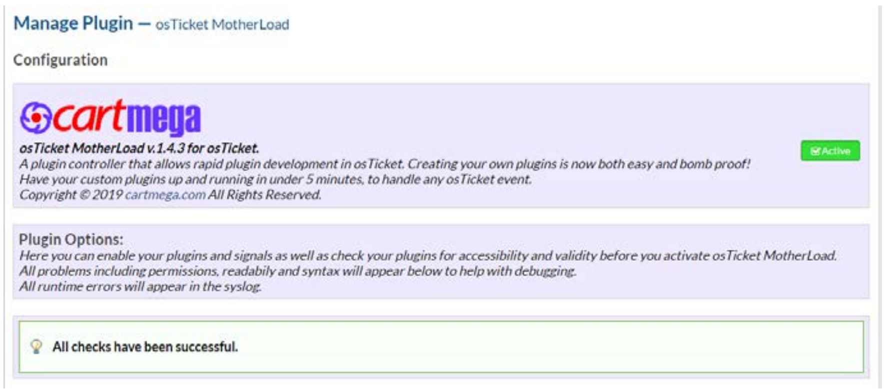


Name	Date modified	Type
include	07/02/2022 03:58	File folder
scp	07/02/2022 03:58	File folder



Name	Size	Changed	Rights
api		26/04/2022 19:26:51	rwxr-xr-x
apps		21/04/2022 15:52:54	rwxr-xr-x
assets		21/04/2022 15:52:54	rwxr-xr-x
css		21/04/2022 15:52:54	rwxr-xr-x
images		21/04/2022 15:52:54	rwxr-xr-x
include		28/04/2022 00:28:36	rwxr-xr-x
js		21/04/2022 15:52:57	rwxr-xr-x
kb		21/04/2022 15:52:57	rwxr-xr-x
pages		21/04/2022 15:52:57	rwxr-xr-x
scp		25/05/2022 02:25:49	rwxr-xr-x
account.php	6 KB	20/04/2022 14:21:06	rw-r--r--
ajax.php	2 KB	20/04/2022 14:21:06	rw-r--r--
avatar.php	2 KB	20/04/2022 14:21:06	rw-r--r--
bootstrap.php	16 KB	19/05/2022 20:04:18	rw-r--r--
captcha.php	1 KB	20/04/2022 14:21:06	rw-r--r--

4. Install Motherload in osTicket.
   a. Login into osTicket.
   b. Go to Admin panel > Manage > Plugins.
   c. Click ‘Add New Plugin’ & click ‘Install’ next to ‘osTicket Motherload’.
   d. To enable the plugin, place a check mark next to ‘osTicket Motherload’. Click the ‘More’ button and click ‘Enable’.
   e. Click on ‘osTicket Motherload’ to enter the plugin settings. Just make sure that there are no errors reported.



Manage Plugin — osTicket MotherLoad

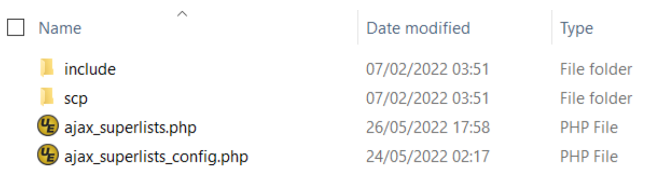
Configuration

osTicket MotherLoad v.1.4.3 for osTicket.
A plugin controller that allows rapid plugin development in osTicket. Creating your own plugins is now both easy and bomb proof!
Have your custom plugins up and running in under 5 minutes, to handle any osTicket event.
Copyright © 2019 cartmega.com All Rights Reserved.

Plugin Options:
Here you can enable your plugins and signals as well as check your plugins for accessibility and validity before you activate osTicket MotherLoad.
All problems including permissions, readabily and syntax will appear below to help with debugging.
All runtime errors will appear in the syslog.

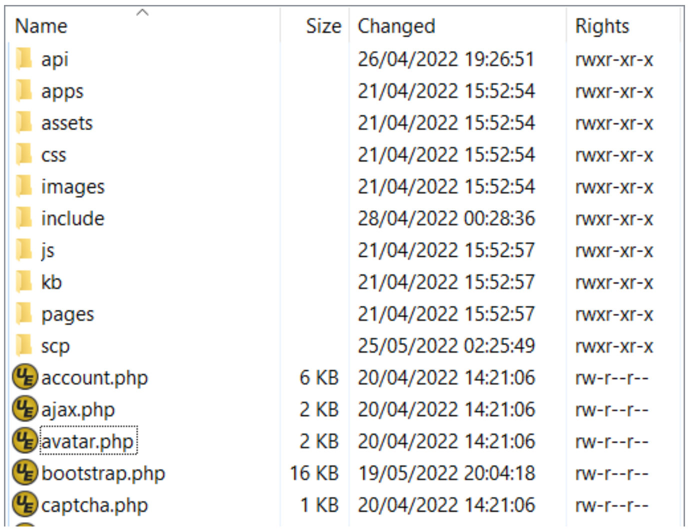
All checks have been successful.

5. Upload Superlists.
   a. Extract the contents of superlists_for_osTicket_motherload_v.x.x.x.zip, onto your hard drive.
   b. Open the folder 'UPLOAD' where you just extracted the zip file.



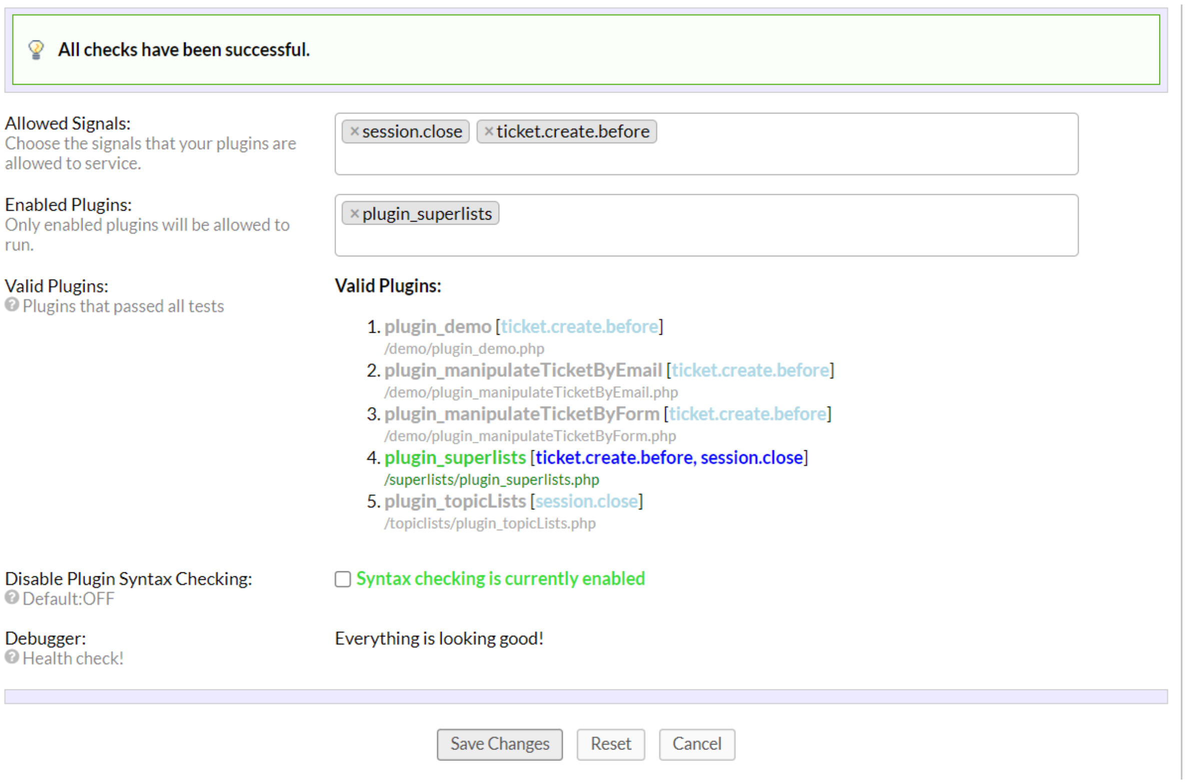
Name	Date modified	Type
include	07/02/2022 03:51	File folder
scp	07/02/2022 03:51	File folder
ajax_superlists.php	26/05/2022 17:58	PHP File
ajax_superlists_config.php	24/05/2022 02:17	PHP File

   c. FTP into your osTicket installation & find the osTicket root directory.
   d. Select all files & folders within the ‘UPLOAD’ folder (on your computer) and upload all to the osTicket root directory of your FTP server.



Name	Size	Changed	Rights
api		26/04/2022 19:26:51	rwxr-xr-x
apps		21/04/2022 15:52:54	rwxr-xr-x
assets		21/04/2022 15:52:54	rwxr-xr-x
css		21/04/2022 15:52:54	rwxr-xr-x
images		21/04/2022 15:52:54	rwxr-xr-x
include		28/04/2022 00:28:36	rwxr-xr-x
js		21/04/2022 15:52:57	rwxr-xr-x
kb		21/04/2022 15:52:57	rwxr-xr-x
pages		21/04/2022 15:52:57	rwxr-xr-x
scp		25/05/2022 02:25:49	rwxr-xr-x
account.php	6 KB	20/04/2022 14:21:06	rw-r--r--
ajax.php	2 KB	20/04/2022 14:21:06	rw-r--r--
avatar.php	2 KB	20/04/2022 14:21:06	rw-r--r--
bootstrap.php	16 KB	19/05/2022 20:04:18	rw-r--r--
captcha.php	1 KB	20/04/2022 14:21:06	rw-r--r--

6. Install Superlists in osTicket.
   a. Login into osTicket.
   b. Go to Admin panel > Manage > Plugins > osTicket MotherLoad.
   c. In Allowed Signals: add 'session.close' & 'ticket.create.before'.
   d. In Enabled Plugins: add 'plugin_superlists'.
   e. Click [Save]. You should see no errors being reported in the debugger.



All checks have been successful.

Allowed Signals:
Choose the signals that your plugins are allowed to service.
×session.close ×ticket.create.before

Enabled Plugins:
Only enabled plugins will be allowed to run.
×plugin_superlists

Valid Plugins:
Plugins that passed all tests

Valid Plugins:

1. plugin_demo [ticket.create.before]
   /demo/plugin_demo.php
2. plugin_manipulateTicketByEmail [ticket.create.before]
   /demo/plugin_manipulateTicketByEmail.php
3. plugin_manipulateTicketByForm [ticket.create.before]
   /demo/plugin_manipulateTicketByForm.php
4. plugin_superlists [ticket.create.before, session.close]
   /superlists/plugin_superlists.php
5. plugin_topicLists [session.close]
   /topiclists/plugin_topicLists.php

Disable Plugin Syntax Checking:
Default:OFF
☐ Syntax checking is currently enabled

Debugger:
Health check!
Everything is looking good!

7. Configure Superlists database connection.
   a. FTP into your osTicket installation & find the osTicket root directory.
   b. Edit the file ‘ajax_superlists_config.php’.
   c. Find the following line…

```
$this->__db = mysqli_connect("localhost", "username", "password", "database");
```

… and modify it to reflect your own database connection.

Congratulations. You have completed the installation of Motherload & Superlists. You can also uninstall it by just deleting the '/motherload/plugins/superlists' folder, without any issues whatsoever.

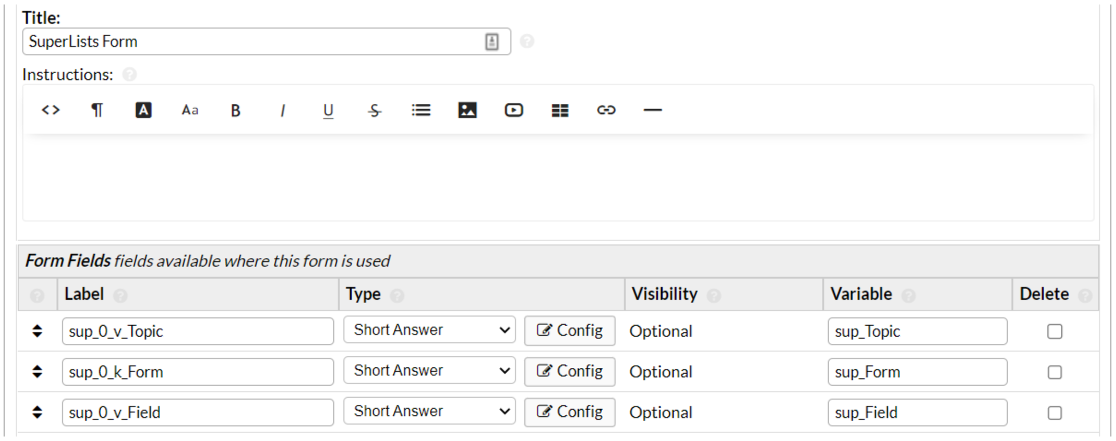
You can now proceed to test the Superlist demo.

Prepare osTicket for the Superlist Demo

osTicket configuration

1. In osTicket, create a New Custom Form called 'SuperLists Form'.
2. Create the following 3 fields on the form:

Label	*Variable (only required for this demo)*	
sup_0_v_Topic	sup_Topic	'Short Answer Field'
sup_0_k_Form	sup_Form	'Short Answer Field'
sup_0_v_Field	sup_Field	'Short Answer Field'



Title: SuperLists Form

Instructions:

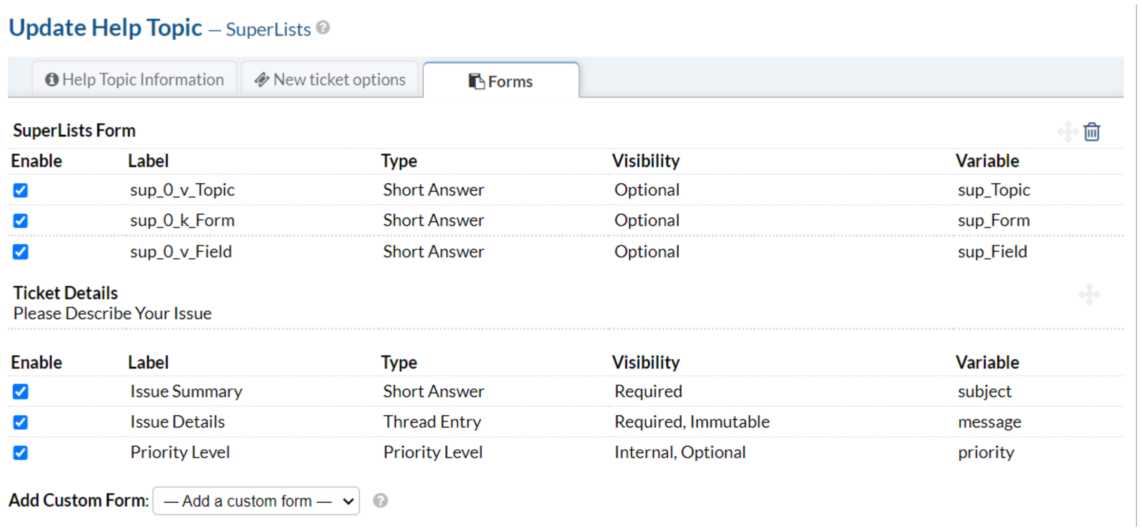
Form Fields fields available where this form is used

Label	Type		Visibility	Variable	Delete
sup_0_v_Topic	Short Answer	Config	Optional	sup_Topic	☐
sup_0_k_Form	Short Answer	Config	Optional	sup_Form	☐
sup_0_v_Field	Short Answer	Config	Optional	sup_Field	☐

PLEASE NOTE: Superlists work with the basic "Short Answer" Field only!

PLEASE NOTE: Superlists will not obey any configuration done via the 'Config' button on the 'Update form section'! Leave those settings at their defaults.

3. Create a New Help Topic 'SuperLists' and assign the 'SuperLists Form' to it.



Update Help Topic — SuperLists

SuperLists Form

Enable	Label	Type	Visibility	Variable
✓	sup_0_v_Topic	Short Answer	Optional	sup_Topic
✓	sup_0_k_Form	Short Answer	Optional	sup_Form
✓	sup_0_v_Field	Short Answer	Optional	sup_Field

Ticket Details
Please Describe Your Issue

Enable	Label	Type	Visibility	Variable
✓	Issue Summary	Short Answer	Required	subject
✓	Issue Details	Thread Entry	Required, Immutable	message
✓	Priority Level	Priority Level	Internal, Optional	priority

Add Custom Form: — Add a custom form —

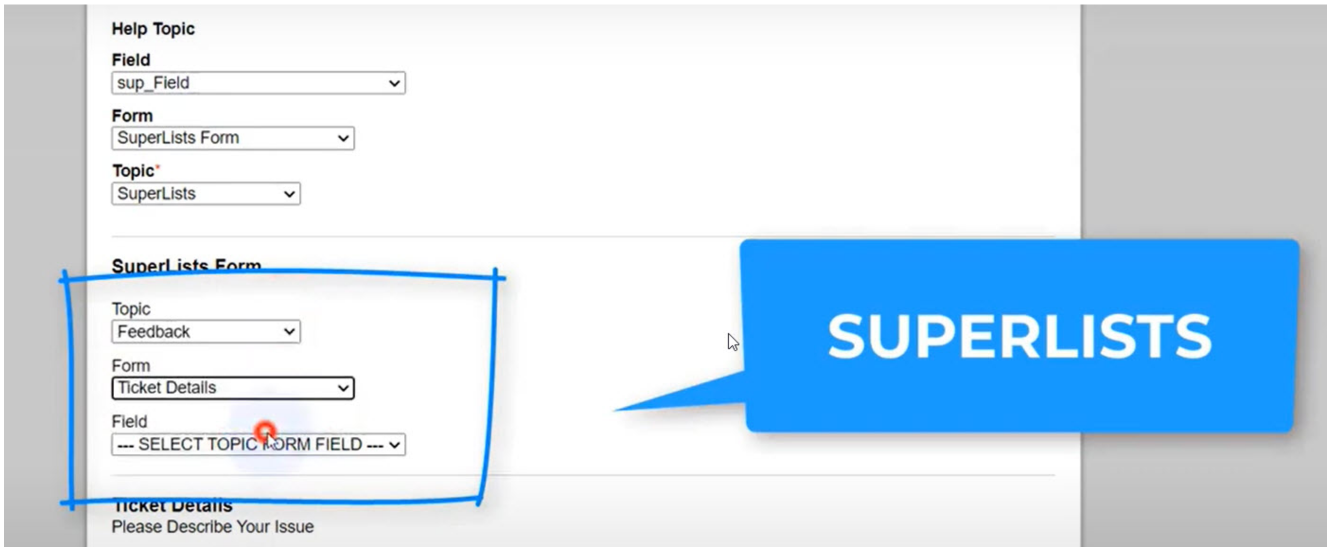
Now, we can test the demo.

TESTING

Client Side Demo

Visit the client side of your osTicket installation and click ‘Open a New Ticket’.

Choose the Help Topic 'SuperLists' which will load the 'SuperLists Form'.



Help Topic

Field: sup_Field

Form: SuperLists Form

Topic*: SuperLists

SuperLists Form

Topic: Feedback

Form: Ticket Details

Field: --- SELECT TOPIC FORM FIELD ---

Ticket Details

Please Describe Your Issue

You should see 3 Superlists shown. If not, then please check your database connection in ‘ajax_superlists_config.php’.

In the demo,

- choosing a Topic yields all the Forms contained in that Topic,
- choosing a Form, yields all Fields contained in that Form,
- choosing a Form-Field name, concludes the user choices.

NOTE: You will notice that Superlists always retain their values between page posts (when a page is submitted). This is also true with all fields on the page such as the ‘Details’ text editor field, which is quite useful.

Logic behind the demo

We needed a demo that would be available to all osTicket installations and we needed to display 3 interlinked lists for this purpose.

We went looking within the osTicket installation for data which was already linked to other data 3-levels deep. We chose the following:

a) **Topics**, each contains…

b) Multiple **forms**, each containing…

c) Multiple **form-fields**.

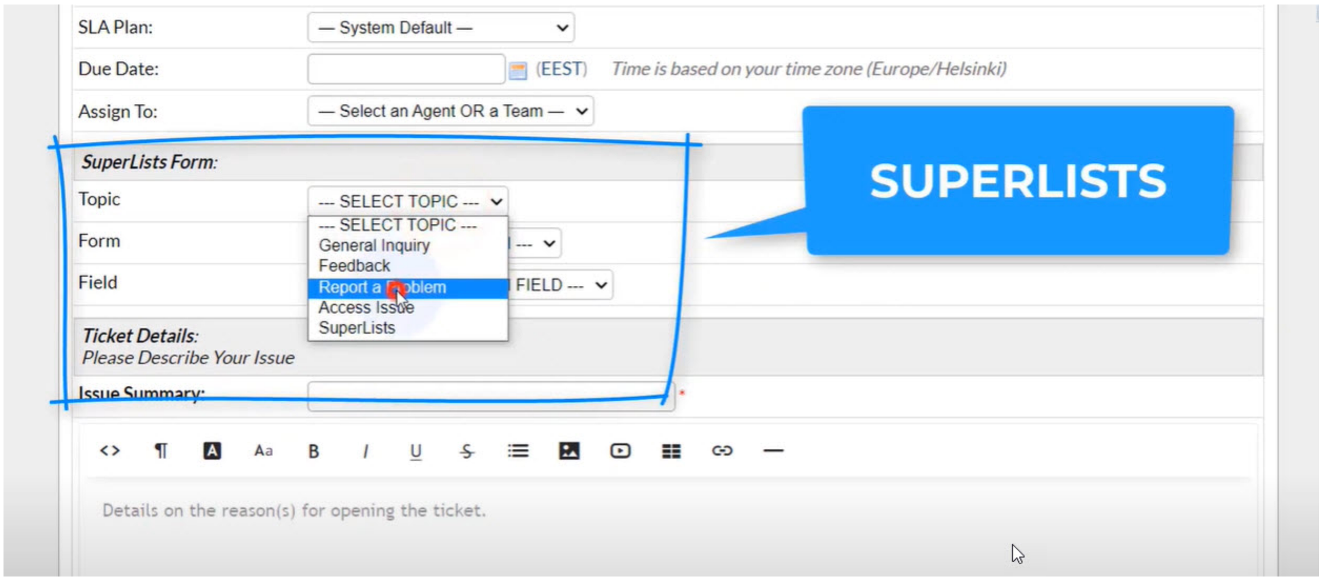
This is an ideal (1 -> many -> many) relationship in relational databases.

So from a single Topic we eventually filter down to one specific Form Field ie. **Topics -> Forms -> Form-fields**.

We chose these fields because they were a convenient way to show 3 interlinked pieces of information which already exist within all osticket installations.

Agent Panel Demo

Visit the agent panel of your osTicket installation and click ‘New Ticket’.



SLA Plan: — System Default —

Due Date: (EEST) *Time is based on your time zone (Europe/Helsinki)*

Assign To: — Select an Agent OR a Team —

SuperLists Form:

Topic: --- SELECT TOPIC ---

Form

Field: --- FIELD ---

Ticket Details:
Please Describe Your Issue

Issue Summary:

Details on the reason(s) for opening the ticket.

As in the client demo, you should see 3 Superlists shown.

Everything should function exactly the same as in the client demo.

CUSTOMIZATION

All customization is done inside the file ‘ajax_superlists_config.php’.

NOTE: If the data used to populate your lists contains **Unicode characters**, make sure that your ‘ajax_superlists_config.php’ file is saved in UTF-8 encoding.

A. Configuration Functions (ajax_superlists_config)

You will notice that there exist 3 functions whose names begin with ‘getList_’.

1. getList_Topic
2. getList_Form
3. getList_Field

These are responsible for generating the 3 Superlists in the demo.

NOTE: It is important to note that within this file, they are in the same sequence as they appear in the browser however this is of no consequence. These functions can appear in any order you prefer. Their order in the browser is only determined by the order they appear on your osTicket form!

In the demo, we only make use of 3 Superlists, however you may declare as many lists as your solution requires. The only limit is the power and capabilities of your server and the power of the client’s PC which will have to render the results (as well as the connection between the two).

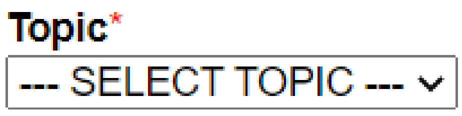
Function naming convention

You can define your functions like this:

protected function [getList]_[field name]

[getList]	: the superlist function prefix (do not change).
[field name *]	: The field name & the label that will be shown to the user.

e.g. The ‘**Topic’** Superlist must share its name with the ‘**getList_Topic**’ function.



* (if your field name contains spaces, replace ' ' with '_' in field name)

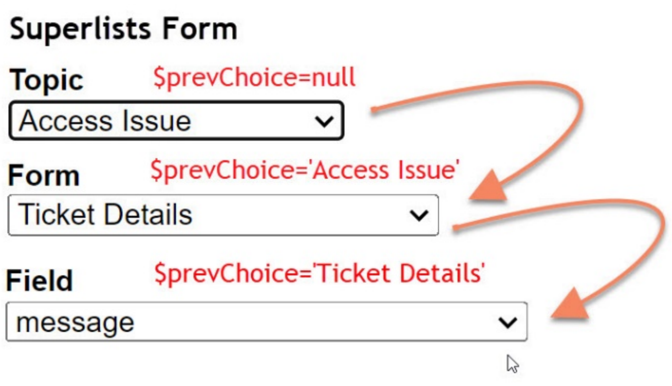
Function arguments

You will notice that all functions take only one argument:

[$prevChoice] : When the user makes a selection in the previous Superlist, the selected key will be held in this variable and used as a function argument for the next Superlist.

Basically, `$prevChoice` is the glue that binds the Superlists together in an unbreakable chain.

Whenever a user makes a selection on a Superlist, his choice is loaded into `$prevChoice` and it is carried over and fed into the next Superlist, in order to prepare its own entries.



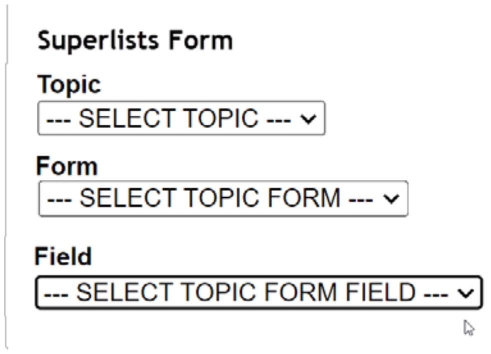
All this is done behind the scenes inside the browser, while the user is making choices – this communication between the browser and the server uses a particular technology called AJAX. Whenever a new choice is made, the browser asks the server and the functions within 'ajax_superlists_config.php' answer.

Superlist header

You will notice that one of the very first lines within each function is this:

```
$title='TOPIC FORM FIELD';
```

This corresponds to the header of each Topic list i.e. what the user sees when no selection has yet been made.



NOTE: To further customize your headers, you can modify variable `$headerFormat` in file '/include/plugins/motherload/plugins/superlists/plugin_superLists.php'.

Populating Superlists

Get your fields with SQL statements from a database or define them manually.

NOTE: Knowledge of MySQL & PHP is needed to pull data from a database!

1. **Populate lists with MySQL**
   You will notice that each `getList_` function contains an SQL section like so:

```
a. $sql = " SELECT topic_id, topic FROM ost_help_topic ORDER BY
   topic_id;";
b. $stmt = $this->__db->prepare($sql);
c. //$stmt->bind_param('i', $prevChoice);
d. if (!$stmt->execute()) {return array($title);}
e. $recs=array();
f. if (($res=$stmt->get_result()) && ($res->num_rows)) {
g. while ($rec=$res->fetch_assoc()) {$recs[$rec[topic_id]]=$rec[topic];}
h. }
```

   a. This is the SQL statement in the `getList_Topic` function.

   b. Prepare the statement for execution (do not change this).

   c. If you will use the variable `$prevChoice` in your SQL statement, uncomment this line. The ‘i’ here states that `$prevChoice` will contain an integer (‘s’ corresponds to ‘string’ etc). You can pass more than one variable to your SQL statement. click here for more info on `bind_param`…

      In the `getList_Form` function, we use `$prevChoice` in the `WHERE` clause (`where topic_id=?`). The value of `$prevChoice` is substituted into the ‘`?`’ at runtime. In this case `$prevChoice` is actually an ‘integer’ (`$stmt->bind_param('i', $prevChoice);`) because it comes from the key of the array in the previous function (`getList_Topic`) which selects `$rec['topic_id']` as the key for each array record returned.
      NOTE: `$prevChoice` is always the key of the previously selected entry. The first list always receives a `$prevChoice=null`.

   d. Here the SQL statement is executed (do not change this).

   e. We prepare an array to hold the records we got from DB (do not change this).

   f. We check if we got any data back from the DB (do not change this).

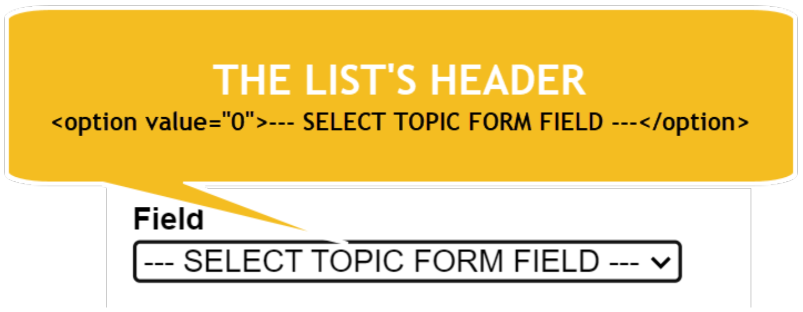
   g. This line loops through the returned data and adds it to our array.
      NOTE: `topic_id & topic` are the fields we are pulling from the DB using our SQL statement (see a.). Here, we assign the `topic` as the value for the particular array key. e.g. `recs[2]='Feedback'`.

In practice, this results in the following HTML code in the browser:

```
<option value="0">--- SELECT TOPIC ---</option>
<option value="1">General Inquiry</option>
<option value="2">Feedback</option>
<option value="10">Report a Problem</option>
<option value="11">Access Issue</option>
<option value="12">SuperLists</option>
```

h. The end (do not change this).

WARNING: Each Superlist includes a **header** which has a key of ‘0’ (zero). It is important to avoid adding entries to your lists which have a key of ‘0’ (zero). If this happens, your header may be replaced with your own entry and the list will not function as expected.


THE LIST'S HEADER
<option value="0">--- SELECT TOPIC FORM FIELD ---</option>
Field
--- SELECT TOPIC FORM FIELD ---

TIP: It may be desirable at an early stage of the process, to build one SQL statement to include all data you will need for all of your Superlists. ie. Build a query to return all Topics, joined to all Forms, joined to all Fields. Use this query as a template in all your Superlists, making slight modifications each time to filter out what you do not want, based on `$prevChoice`.

2. Populate lists manually

IMPORTANT: *Since we will not be using the DB, make sure you disable the 2 lines referring to* `$this->__db` *in your config file otherwise you will get 500 errors.*

It is quite easy to just use plain PHP arrays to populate your superlists.

In this example we will create a set of Superlists 3-levels deep showing car makes, models and years of manufacture for each major release.

NOTE: For the sake of simplicity we will only use a few records.

Here is a table of the data:

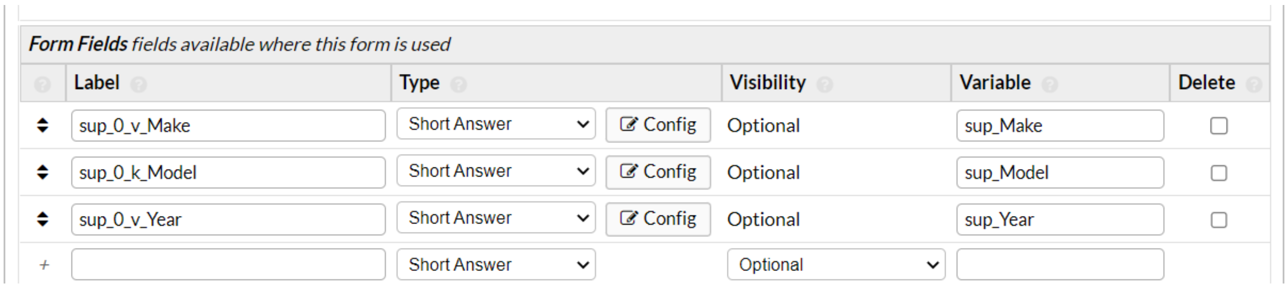
id	*make*	*model_id*	*model*	*year*
1001	BMW	201	1 serie	2015
1002	BMW	202	2 serie Active Tourer	2014
1003	BMW	203	3 serie	2015
1004	Kia	301	Quoris	2015
1005	Kia	301	Quoris	2014
1006	Kia	302	Soul	2014
1007	Kia	302	Soul	2016
1008	Kia	303	Venga	2014

We will name our 3 Superlists as follows: make, model, year.

Our functions will be declared as follows:

a. getList_make
b. getList_model
c. getList_year

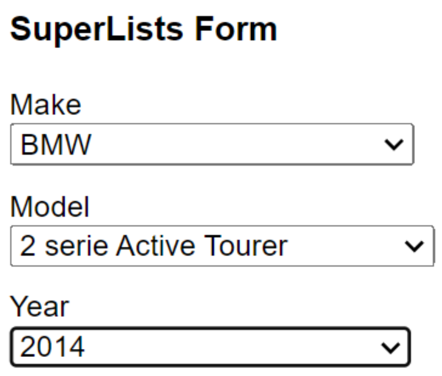
Our first task is to declare these names as our labels and variables in our osTicket form.



Form Fields *fields available where this form is used*

	Label	Type		Visibility	Variable	Delete
⇕	sup_0_v_Make	Short Answer	Config	Optional	sup_Make	☐
⇕	sup_0_k_Model	Short Answer	Config	Optional	sup_Model	☐
⇕	sup_0_v_Year	Short Answer	Config	Optional	sup_Year	☐
+		Short Answer		Optional		

```
protected function getList_Make($prevChoice) {
      global $ost,$cfg;

      $title='CAR MAKE';

      $recs=array(
            'bmw' =>'BMW',
            'kia' =>'KIA'
      );

      return array($title,$recs);
}

protected function getList_Model($prevChoice) {
      global $ost,$cfg;

      $title='CAR MODEL';

      switch($prevChoice) {

            case 'bmw':
                  $recs=array(
                        '201' =>'1 serie',
                        '202' =>'2 serie Active Tourer',
                        '203' =>'3 serie',
                  );
                  break;

            case 'kia':
                  $recs=array(
                        '301' =>'Quoris',
                        '302' =>'Soul',
                        '303' =>'Venga',
                  );
                  break;
      }

      return array($title,$recs);
}

protected function getList_Year($prevChoice) {
      global $ost,$cfg;

      $title='CAR YEAR';

      switch($prevChoice) {

            case '201':
                  $recs=array(
                        '1001'     =>'2015',
                  );
                  break;

            case '202':
                  $recs=array(
                        '1002'     =>'2014',
                  );
                  break;

            case '203':
                  $recs=array(
                    '1003'      =>'2015',
                );
                break;

            case '301':
                $recs=array(
                    '1005'      =>'2014',
                    '1004'      =>'2015',
                );
                break;

            case '302':
                $recs=array(
                    '1006'      =>'2014',
                    '1007'      =>'2016',
                );
                break;

            case '303':
                $recs=array(
                    '1008'      =>'2014',
                );
                break;

        }

        return array($title,$recs);
}
```



Make	BMW
Model	2 serie Active Tourer
Year	2014

The method couldn't be simpler. We are just working with arrays here. Depending on what `$prevChoice` we receive, we return a particular array. Superlists takes care of the rest.

NOTE: If the data used to populate your lists contains Unicode characters, make sure that you 'ajax_superlists_config.php' file is saved in UTF-8 encoding.

Returning Superlist arrays

Every function we have studied previously always returns an array back to the user browser. This is the last line of each function.

```
return array($title,$recs);
```

The $title is used to complete the header of the Superlists.

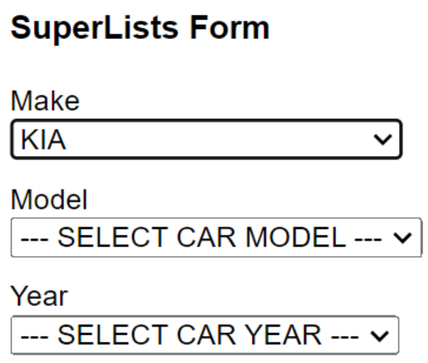
The $recs are of course our array of data for the Superlist's content.

There is however a third option you can use here.

Default selection

```
return array($title,$recs,'kia');
```

This third option affects the default selection of our Superlist when the page is first loaded. 'Kia' will therefore always be the default selection.



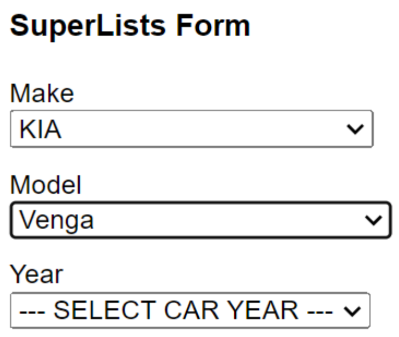
SuperLists Form

Make
KIA

Model
--- SELECT CAR MODEL ---

Year
--- SELECT CAR YEAR ---

Furthermore if we also specify `return array($title,$recs,303);` in function `getList_Model`, we will also have a default value for the model Superlist.



SuperLists Form

Make
KIA

Model
Venga

Year
--- SELECT CAR YEAR ---

NOTE: Specifying a default option will not allow Superlists to retain their selected values across page submissions. Without specifying this option, when the user submits the page, the Superlists will retain their selected values from the previous page.

NOTE: An easy way of returning an empty list is by `return array($title);` without an array specified. This may be useful when some error will not allow to return any records.

Further Customization / Personalized Lists

Each function begins with:

```
global $ost,$cfg;
```

These are osTicket global variables and are being pulled into your function in case you may decide to use them. A description of each variable is beyond the scope of this manual.

Just like these variables, osTicket maintains many others. One such useful variable is `$thisclient`, which contains information on the currently active client.

In order to use this variable in your function, you should first allow exposure to it:

1. uncomment this command in your 'ajax_superlists_config.php' file:

```
require('client.inc.php');
```

2. change your global declaration within your function to include it.

```
global $ost,$cfg,$thisclient;
```

You can now use something like this to extract information from the active logged-in client who is interacting with your application.

```
if ($thisclient && $thisclient->isValid()) {$clientLoggedIn=true;}
// Get logged in user info
$org=false;
if ($clientLoggedIn) {$userId=$thisclient->getId();} else {$userId=false;}
if ($clientLoggedIn) {$userName=$thisclient->getUsername();} else {$userName=false;}
if ($clientLoggedIn) {$userType=$thisclient->getUserType();} else {$userType=false;}
if ($clientLoggedIn) {$userEmail=$thisclient->getEmail();} else {$userEmail=false;}
if ($clientLoggedIn) {$org=$thisclient->getOrganization();} else {$org=false;}
if ($clientLoggedIn && $org) {$orgName=$org->getName();} else {$orgName=false;}
if ($clientLoggedIn) {$orgId=$thisclient->getOrgId();} else {$orgId=false;}
```

You can then use some of this information in your SQL statements to present your client with personalized Superlists.

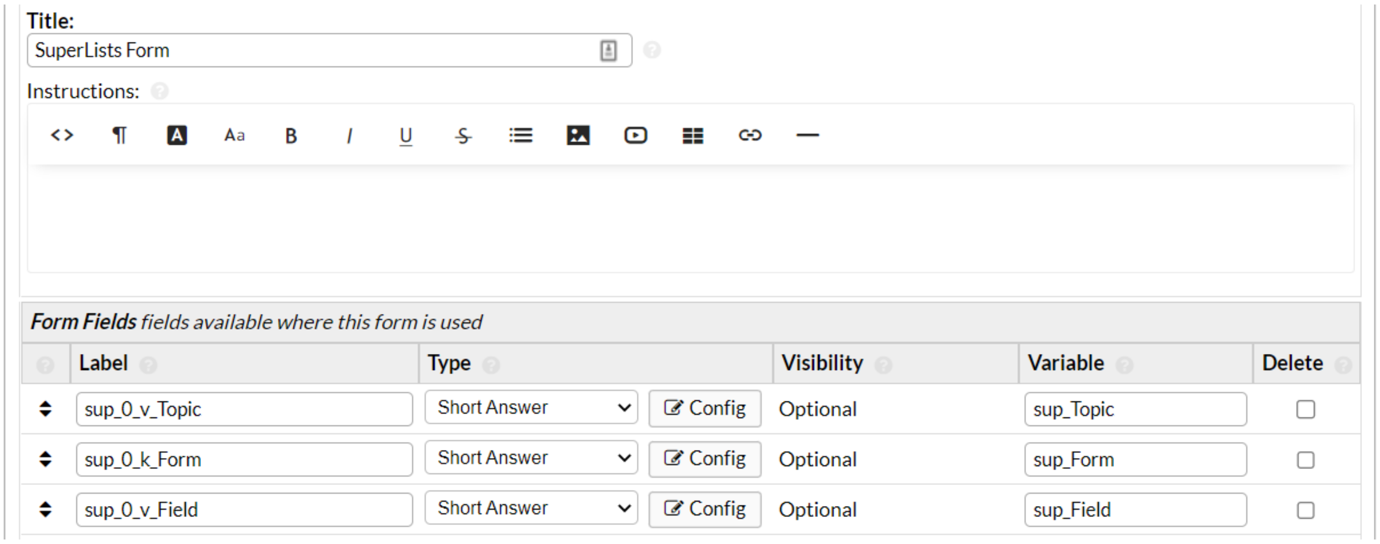
e.g. you can present your client with a list of order numbers based on his email address registered in osTicket.

```
$sql = "SELECT `order_id` FROM `oc_order` WHERE `email`=? ORDER BY `order_id`;";
$stmt = $this->__db->prepare($sql);
$stmt->bind_param('s', $userEmail);
```

B. osTicket Form Configuration

PLEASE NOTE: Superlists work with the basic "Short Answer" Field only!

PLEASE NOTE: Superlists will not obey any configuration done via the 'Config' button on the 'Update form section'! Leave those settings at their defaults.



Title: SuperLists Form

Instructions:

Form Fields fields available where this form is used

	Label	Type		Visibility	Variable	Delete
	sup_0_v_Topic	Short Answer	Config	Optional	sup_Topic	☐
	sup_0_k_Form	Short Answer	Config	Optional	sup_Form	☐
	sup_0_v_Field	Short Answer	Config	Optional	sup_Field	☐

For the demo we used the above configuration. Of course the name of the form is immaterial, just as long as you add it to one of your Topics so that the Superlists get generated.

Labels

The form fields must be named according to the following naming convention:

`[sup]_[group number]_[collector]_[field name]` `e.g.sup_0_v_Order ID`

Part	*Options*	*Description*
[sup]		superlist indicator (do not change)
[0]	int	the group number for a group of interdependent superlists
[v]	k or v	collect the key or value of the superlist in this field
[field name]		the name of the field

Groups (integer)

In our demo, we make use of 3 Superlists which belong to the same group (0). All 3 work together, one feeding the other. You could however define as many Superlist groups as you like, just by changing the `[group number]` and of course give each of them a different `[field name]` so that there is no clash with other Superlists. Each group is independent and can take any number of Superlists.

Field Names

Used as field label on the webpage and used in the field names of the HTML elements themselves. Must be used in the function names that populate the fields.

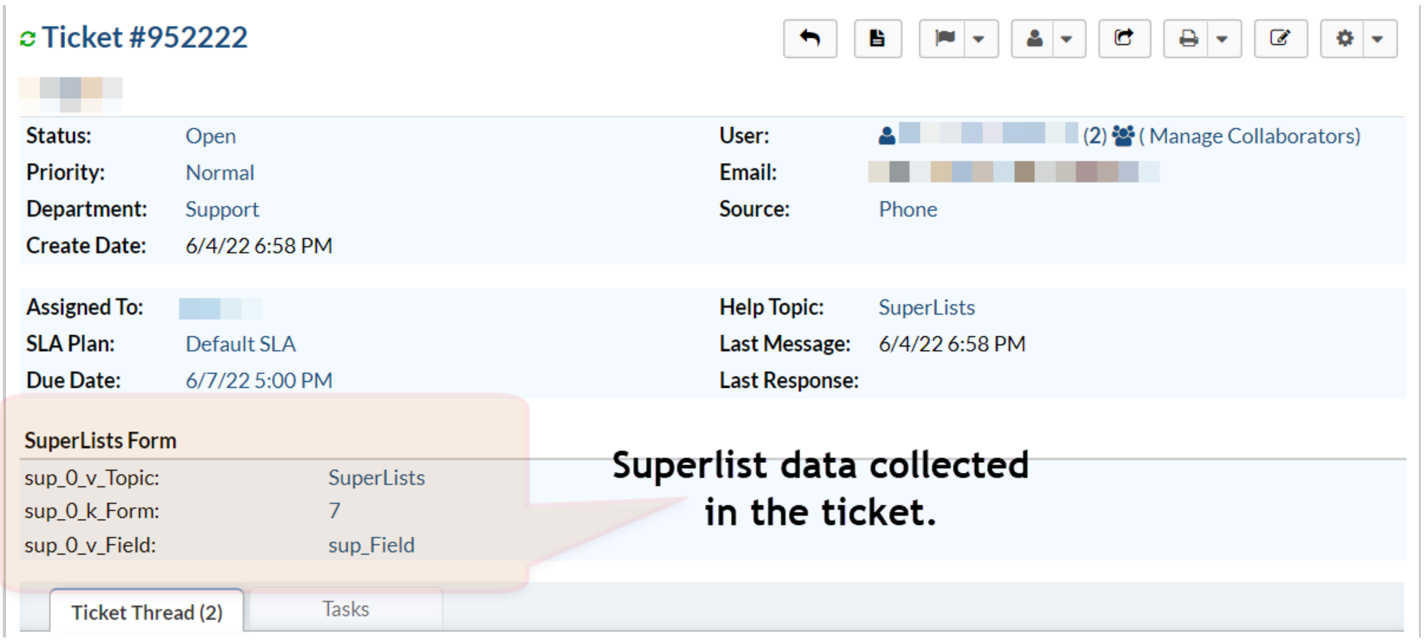
Collector (k or v)

This flag can take either one of two letters ‘k’ or ‘v’.

The purpose of the collector is to decide which one of the 2 possible values to pass over to osTicket for the purpose of saving it in the ticket.

As we are well aware, we have arranged to populate each Superlist with key/value pairs from our arrays. osTicket cannot keep both of these values in a single field (even though it is possible it is generally undesirable).

With this flag, we tell Superlists which one of these 2 options we would prefer to keep in our ticket – the key or the value.



Ticket #952222

Status:	Open	User:	(2) (Manage Collaborators)
Priority:	Normal	Email:	
Department:	Support	Source:	Phone
Create Date:	6/4/22 6:58 PM		

Assigned To:		Help Topic:	SuperLists
SLA Plan:	Default SLA	Last Message:	6/4/22 6:58 PM
Due Date:	6/7/22 5:00 PM	Last Response:	

SuperLists Form

sup_0_v_Topic:	SuperLists
sup_0_k_Form:	7
sup_0_v_Field:	sup_Field

Variables

The variable names can of course be used within osTicket itself but also are accessible within the Superlists plugin as will be revealed in the next section.

C. plugin_superLists.php

Below features are essential to Superlists functionality:

signalConnect

The array of signals we are subscribed to (do not modify).

$headerFormat

Modify the wording of your Superlist headers.

Below features are provided to demo the Superlist capabilities available:

if ($_POST) {…}

Execute code only after the ticket is submitted.

The whole of this code block can be removed if needed.

$enabledTopics

Can be `true` or `false` or `array('Topic A', 'Topic B')`;

Whether to proceed based on the Help Topic chosen by the user.

$enabledGuests

Whether to proceed if the user is a guest (not a logged in client).

$enabledClients

Whether to proceed if the user is a logged in client.

D. Motherload Capabilities (plugin_superLists.php)

In the previous section we saw how we can create and populate our Superlists.

What if we wanted to react to what the user has selected in a list?

Motherload which is the plugin controller for Superlists gives us such capabilities.

```
$this->dbgecho(‘message’)
```

Useful when you are forced to debug your plugin in a live environment without any clients seeing your debug messages.

The `$debug_ip` array controls who can see the messages.

- True (everyone can see)
- False (no one can see)
- array('xxx.xxx.xxx.xxx') (IP restricted)

```
$this->ostlog(LOG_WARN,'Title','Message');
```

You can easily write to the osTicket syslog.

```
$this->addErr('Title','Message');
```

Add an osTicket error to its error array. The errors are shown by osTicket near the top of the page only if you disrupt a process like ticket.

e.g. when you attach your plugin to ticket.create.before, you can use the following code to induce an error in osTicket.

```
if ($_POST) {
  $this->addErr('Title','Message');
  return false;
}
```

```
$this->addAlert('Title','Message');
```

Show a JQuery skinned modal pop-up alert to the user.

```
$this->dbgecho('<pre>'.print_r($this->ml,2).'</pre>');
```

Show all data in the Motherload object.

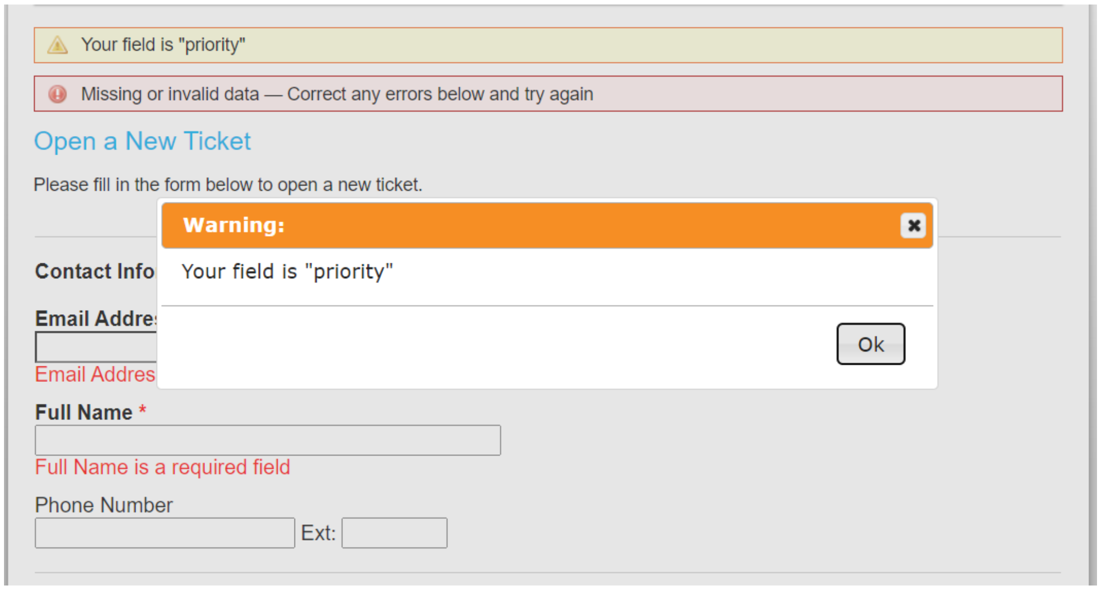
The above commands can only be used inside the file ‘plugin_superLists.php’.

Example: We can use the following script to pop an alert showing the value selected in the first Superlist after submitting it.

We paste our code after `$this->injectJSListGenerator();`.

We will be looking for our Superlist value in the `cdata` array. The name of the field is the variable name we specified on our osTicket form ie. `sup_Field` etc.

```
// ---------------------------------------------------------------------
// FIELD VALIDATION AFTER POST
// ---------------------------------------------------------------------
if ($_POST) {
      $field=isset($this->ml->cdata['sup_Field']['value']) ? $this->ml-
>cdata['sup_Field']['value'] : '';
      if ($field == 'priority') {
            $this->addAlert('Warning:','Your field is "'.$field.'"');
      }
}
```



Your field is "priority"

Missing or invalid data — Correct any errors below and try again

Open a New Ticket

Please fill in the form below to open a new ticket.

Warning:

Your field is "priority"

Ok

Contact Info

Email Addre

Email Addres

Full Name *

Full Name is a required field

Phone Number

Ext:

Much useful information is also contained in the `$this->ml->_POST` array and of course in `$this->ml`.

They can be echoed on screen for scrutiny like this:
`$this->dbgecho('<pre>'.print_r($this->ml,2).'</pre>');`

You will also notice the osTicket global variables are available in `function run()` and this means your options are endless. You can even customize the behavior of your Superlists based on URL, client, staff and much more!

DATABASE SAFETY

WARNING:

`$prevChoice` is coming from the wild and could be easily manipulated.

STAY SAFE & AVOID SQL INJECTIONS.

Use `prepare` **/** `bind_param` **/** `execute` in your functions.

Keep to the examples in 'ajax_superlists_config'!

CSS STYLING

You can style your superlists with CSS by addressing any one of five available classes.

You can use the following classes as follows:

Class	*Application*
.superlist	Apply style to all superlists
.tl_group_x	Apply style to a whole group of superlists
.tl_grp_index_x	Apply style to a particular level in every group of superlists
.tl_index_x	Apply style to just a single superlist
.form-control	Accommodates styling in osTicket Responsive Theme & osTicket Bootstrap Theme

THEME COMPATIBILITY

Theme	*Compatibility*
1. osTicket Awesome Theme	✓
2. osTicket Responsive Theme	✓
3. osTicket Bootstrap Theme *Depending on the version, this theme may need an updated bootstrap & JQuery.* *e.g. If you have osTicket 1.14 with this theme, and your Superlists are not working, you probably need some updates:* *1. Make sure you have jquery-3.4.0.min.js in /js/ (the version that came with osTicket 1.14)* *2. Make sure you have jquery-ui-1.12.1.custom.min.js in /js/ (the version that came with osTicket 1.14)* *3. Make sure you have the right paths to the above files in:* *a. /include/client/header.inc.php* *b. /include/staff/footer.inc.php* *c. /include/staff/login.tpl.php* *4. Make sure you have bootstrap.min.js (v3.3.7 or higher) in /assets/default/js/ (the version that can support Jquery 3.0+).* *5. Superlists should work fine now.*	✓

The Ultimate in
Help Topic Categorization
Topiclists can be interlinked & nested & populated with AJAX
and even tailored to individual clients!
for osTicket
includes motherload
'It is finally here... the holy grail in the ticket creation process!'
cartmega
TOPICLISTS
Topiclists
plugin for osTicket
(includes osTicket Motherload)
HELP TOPIC, INTERLINKED, DYNAMIC, AJAX LISTS
FULLY PROGRAMMABLE!
MOTHER LOAD
SUPPORT CENTER
Support Ticket System
Open a New Ticket

TOPICLISTS

What Superlists can do for Forms, Topiclists can do for Help Topics.

Enhance your Help Topics with multiple, custom, smart, dynamic, interlinked and nested drop-down lists which are updated with ajax from any database.

Check out the video and online demo below:

cartmega demo

SOLUTIONS TO PROBLEMS

PLEASE NOTE: Superlists is a Motherload plugin. As such, you should also check the 'Common Problems' section in the Motherload readme file for possible errors and solutions.

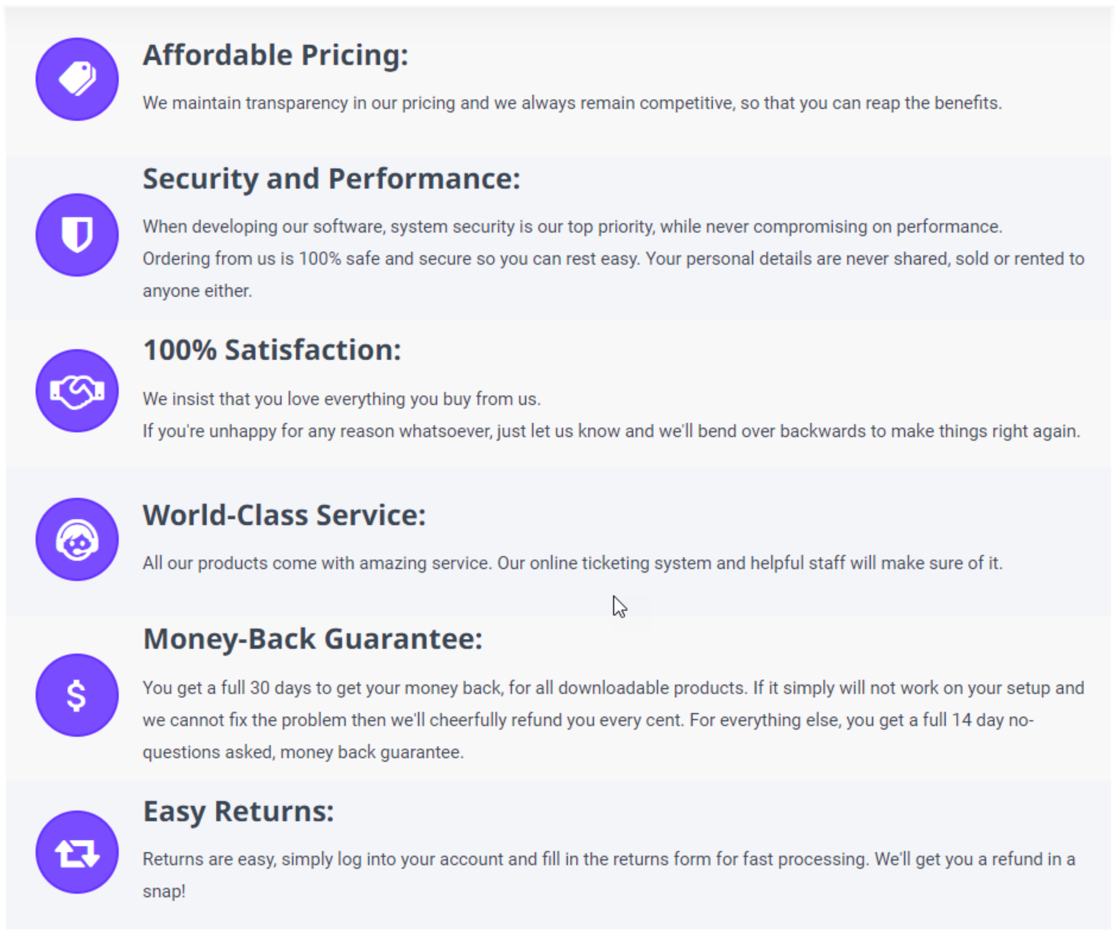
Problem:	Not working?
Description:	Superlists need Ajax to work.
.	
Solution:	Check your `Admin Panel > Dashboard > Information > PHP Settings > cgi.fix_pathinfo` and make sure `"1" is recommended if AJAX is not working` is checked. Always check the motherload plugin page and make sure there are no errors being reported.

Problem:	A superlist gets populated correctly but does not contain a header "---SELECT blah blah ---". Instead, the first entry in the list is shown immediately after the page loads.
Description:	Your list contains an entry that has a '0' (zero) as its key. e.g. 0=>'BMW'. This overrides the list's header which by design has also a '0' as its key. Your entry therefore takes the place of the header and breaks the header functionality.
.	
Solution:	Change your entry's key to some other value which is not '0'. You can use any other alphanumeric value.
Problem:	You have followed the installation/configuration video, but the demo Superlists are not appearing or loading correctly.
Description:	You need to do some basic digging to find the cause of the problem.
.	
Solution:	1. Visit the Motherload plugin page and see if any errors are being reported. They could be due to some server misconfiguration or even a mistake in your coding. There should be no errors reported for normal function of Superlists. Be sure that these errors are eliminated before going further. 2. When you click F12 in your browser you can see error messages in red from the browser. Normally you should only be able to see the below messages from Superlists and no errors: `SL: Initialized!          superlist_generator.js:130` If you are seeing errors, disable Motherload and reload the page, and see if the errors reappear. If they reappear, then you must check to see if they can be fixed before running Motherload again. If the red error messages are affecting the functionality of JQuery, this may be impacting Superlists. 3. Are you using any special characters in your Superlists (e.g. Spanish/French/Greek)? Make sure the configuration file

ajax_superlists_config.php at the root of your osTicket installation is saved in UTF-8 format. If not, then some special characters in your arrays may be breaking the JSON functionality.

4. If the previous steps are OK but still Superlists are not loading correctly then this may be an indication that another addon/theme you have installed on osTicket may be affecting Superlists. In this case you should open a support ticket so we can investigate further.

Why choose Cartmega?



Affordable Pricing:

We maintain transparency in our pricing and we always remain competitive, so that you can reap the benefits.

Security and Performance:

When developing our software, system security is our top priority, while never compromising on performance. Ordering from us is 100% safe and secure so you can rest easy. Your personal details are never shared, sold or rented to anyone either.

100% Satisfaction:

We insist that you love everything you buy from us.
If you're unhappy for any reason whatsoever, just let us know and we'll bend over backwards to make things right again.

World-Class Service:

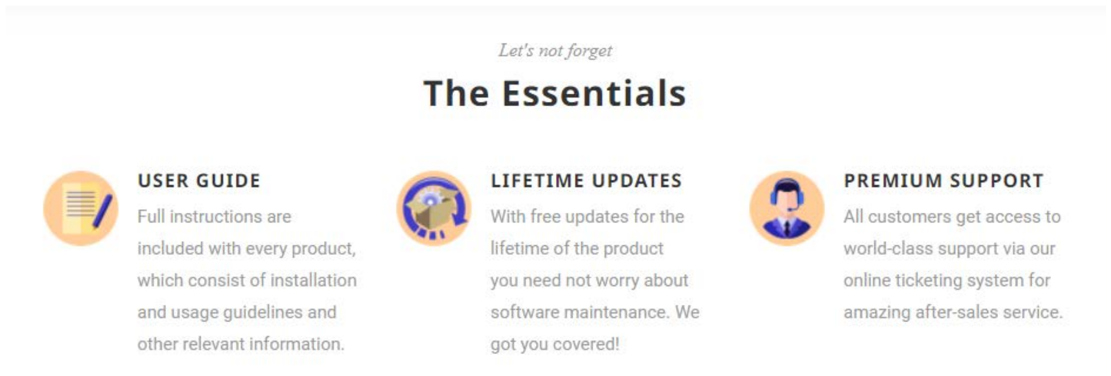
All our products come with amazing service. Our online ticketing system and helpful staff will make sure of it.

Money-Back Guarantee:

You get a full 30 days to get your money back, for all downloadable products. If it simply will not work on your setup and we cannot fix the problem then we'll cheerfully refund you every cent. For everything else, you get a full 14 day no-questions asked, money back guarantee.

Easy Returns:

Returns are easy, simply log into your account and fill in the returns form for fast processing. We'll get you a refund in a snap!



Let's not forget

The Essentials

USER GUIDE

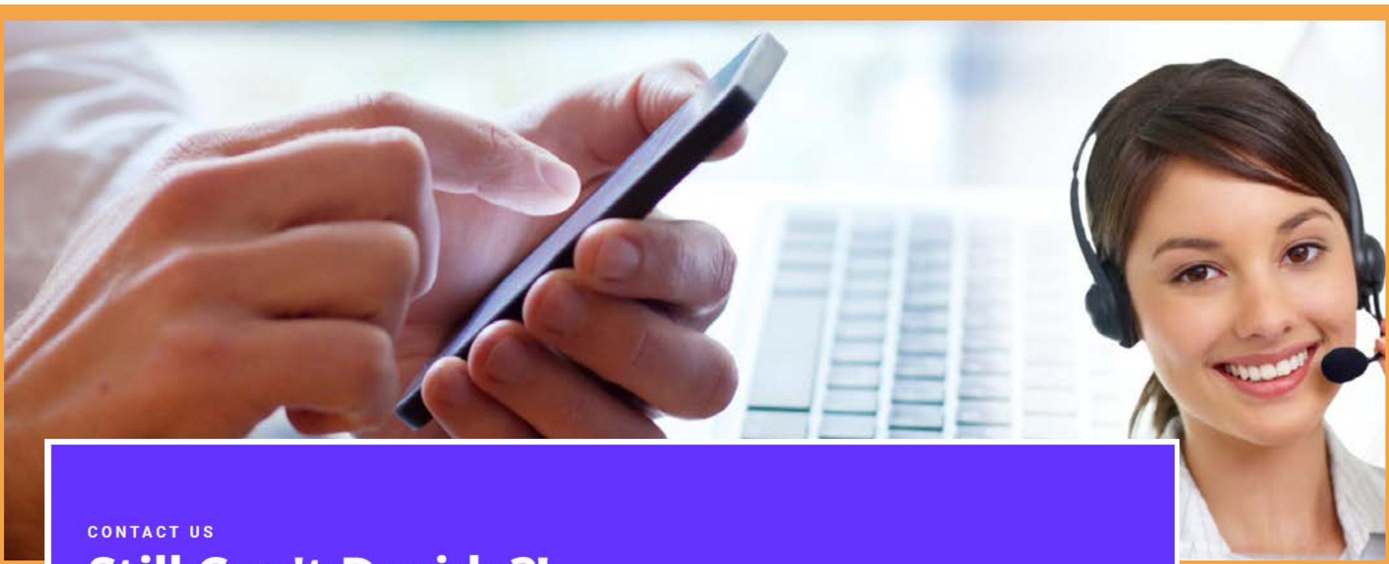
Full instructions are included with every product, which consist of installation and usage guidelines and other relevant information.

LIFETIME UPDATES

With free updates for the lifetime of the product you need not worry about software maintenance. We got you covered!

PREMIUM SUPPORT

All customers get access to world-class support via our online ticketing system for amazing after-sales service.



CONTACT US

Still Can't Decide?!

Just in case you still have questions or not sure that what you've chosen is suitable, no worries. Contact us, we are here to help.

Get In Touch

Software disclaimer

Software developed by cartmega.com is provided 'as is' without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of fitness for a purpose, or the warranty of non-infringement.

Without limiting the foregoing, cartmega.com makes no warranty that:

- the software will meet your requirements;
- the software will be uninterrupted, timely, secure or error-free;
- the results that may be obtained from the use of the software will be effective, accurate or reliable;
- the quality of the software will meet your expectations;
- any errors in the software obtained from the cartmega.com web site will be corrected.

Software and its documentation:

- could include technical or other mistakes, inaccuracies or typographical errors. cartmega.com may make changes to the software or documentation made available on its web site;
- may be out of date, and cartmega.com makes no commitment to update such materials;
- cartmega.com assumes no responsibility for errors or omissions in the software or documentation available.

In no event shall cartmega.com be liable to you or any third parties for any special, punitive, incidental, indirect or consequential damages of any kind, or any damages whatsoever, including, without limitation, those resulting from loss of use, data or profits, whether or not cartmega.com has been advised of the possibility of such damages, and on any theory of liability, arising out of or in connection with the use of this software.

The use of the software is at your own discretion and risk and with agreement that you will be solely responsible for any damage to your computer system or loss of data that results from such activities. No advice or information, whether oral or written, obtained by you from cartmega.com or from the cartmega.com web site shall create any warranty for the software.