

**elastic cms**  
easy to generate xml

## Requirements

1. A server with a **MySQL 5** database
2. A server with **PHP 5**.
3. Enabled **mod\_rewrite** module on the server

## Installation

1. Unpack the cms files from the .rar file
2. Create a database on the server
3. Import the **dump.sql** file from the main cms catalog to the database
4. In **cfg/conn.database.php**, change the database access parameters to your own. The file contains these parameters:

```
host = server name, usually localhost
user = database user name
pass = database password
database = database name
```

5. Upload the cms files to the server.
6. Grant write permission (**chmod 777**) to these catalogs
  - **access\_path/**
  - **external/libs/cache/**
  - **external/libs/compiled/**
  
  - **store/images/**
  - **store/images/thumbnail**
  - **store/audio/**
  - **store/video/**

Try to run the cms from the browser. In case of any error check whether you have followed all the steps exactly as described and whether your server meets the cms requirements.

**After an error, before running the cms again, delete the contents of the first three catalogs mentioned in point 6.**

If no error occurs, you can log in to the cms panel typing **demo** as the login and password.

# General configuration

## 1. Changing the login and password

In `cfg/cfg.php` find the code presented below and change the login and password to your own:

```
define("LOGIN", "demo");  
define("PASS", "demo");
```

## 2. Default picture thumbnail generation.

A thumbnail is not created in this case, it is used only as a preview in the cms. In `cfg/cfg.php` find the code shown below:

```
define('THUMB_HEIGHT', 40);  
define('THUMB_WIDTH', auto);
```

The first value is, of course, the height, while the second is the width. You can use one out of three types of values:

- a number of pixels, e.g. `40`
- `auto`, which enables size constraints to preserve proportions. One of the parameters must be a numerical value.
- `false`, which disables thumbnail generation. Must be used for both values.

## Detailed configuration - (xml generation)

### 1. Configuration files

The files responsible for generating xmls are in the `config/` catalog. All the files contained there are responsible for xmls, except `cfg.php` - this file is of no interest.

Every file from which an xml is generated is named `cfg.some_name.php`. The second part, i.e. `some_name` will also be the name of the generated xml file.

### 2. Configuration file description

Let's look at an example file from the `config/` directory, e.g. `cfg.texts.php`

```
menu_name = Texts
input[] = title,Title
text[] = des,Description with html tags
link[] = url,Url link
```

The first three lines are of no interest, so we'll start from the 4th line. The `menu_name` parameter is responsible for the name displayed in the cms - so this will be just `Texts`. The rest of the 3 lines are fields found in the cms and will be generated in the xml.

Each of these fields consists of 3 parts:

- `field[]` (e.g. `input[]`) - parameter name. There's a couple of them, each responsible for a different kind of field existing in the cms (listed later)
- the parameter after the `=` sign, e.g. `title` - the name of an attribute that will exist in the generated xml file.
- the parameter after the `,` e.g. `URL link` - the name that will show up in the cms. It can be used as info for a concrete field.

### 3. Configuration file parameters (types of fields that can be generated)

You can put the following fields in the configuration file:

- `input[]` - single-line text field, without html encoding
- `text[]` - multiline text field with html coding (textarea)
- `link[]` - link-type text field, the 'http://' prefix is added, the field is clickable in the cms
- `image[]` - image field, clickable with preview and upload to server. Handles `jpg`, `gif` and `png` (with alpha) formats.

This field can possess additional information about generating a picture thumbnail, as shown in the following code:

```
image[] = pic,Picture file with thumbnail
thumbnail = true
thumbnail_width = 100
thumbnail_height =
```

If we want to save the aspect ratio of the generated thumbnail, one of the parameters describing the size of the thumbnail are to be left empty after the `=` sign.

- `audio[]` - audio field, clickable with preview and server upload, `mp3` format
- `video[]` - video field, clickable with preview and server upload, `flv` format

These are all the fields, however there is nothing stopping you from keeping `pdf` or `doc` files on the server. All you need to do is create an `image[]` field and use it to upload entirely different files. A disadvantage if this is that there won't be a thumbnail displayed, but the file path will be generated to the xml file just fine.

#### 4. Combining configuration files (unlimited xml tree)

The true power of the cms lies in the option to combine one configuration file with others, while the amount of these connections is meaningless.

- Joining two files (`cfg.gallery.php`):

```
menu_name = Pictures gallery with categories
input[] = cat,Pictures category name
subcategory = pictures_2
```

As you can see in the above code, all you need to do is add `subcategory`, and the name of the configuration file to be added after the `=` sign. In this case it's `cfg.pictures_2.php`. We don't use the `cfg` prefix or the `php` extension.

- Joining multiple files simultaneously:

```
menu_name = Unlimited xml tree
input[] = title,Title
text[] = des,Description with html tags
link[] = url,Url link
subcategory = "pictures_1|music|films"
```

Like in the previous example, except all the subcategories are listed in `"",` and the files are separated by a `|`.

Of course, it is possible to generate a multilevel tree due to the possibility that the joined files already contain their own joins.

#### 5. Generated xml names

The names of the xml files we want to import, i.e. for flash, correspond to the configuration file names, which means that e.g. for `cfg.pictures.php` the xml file will be `pictures.xml`. If we have a tree as a result of joining configuration files, we only import the top-level filename.

If we want to have everything generated in one file, we reference `feed.xml`.

## 6. Blocking certain cms functions

As you may have noticed, there are some more lines of code in each configuration file:

```
arr = true  
add = true  
remove = true  
edit = true
```

These are, respectively, the up and down arrow buttons in the cms, the add category button, the remove and edit buttons. This, as suspected, allows enabling (button visible) or disabling (button invisible) individual functions of the cms. Naturally, each parameter can only take one of 2 possible values: **true** or **false**.

This configuration allows you to, for example, prevent deleting a given category after creating it, thanks to which the client won't accidentally remove an important higher-level category, which would in turn force the deletion of the elements it contains.

## Summary

As you can see, the cms is able to generate every type of xml tree. Configuration is fairly easy and takes a small amount of time. The cms can manage every flash webpage based on xml and storing data in attributes.

I hope that this instruction was understandable. In case of any questions and problems, contact me at [miro24lp@gmail.com](mailto:miro24lp@gmail.com)