SBNDF&CMS

User Guide

Creating a Simple Contact Form

By SBND Technologies

fcms.sbnd.net

Description and Requirements

This example shows how to develop a "Contact Us" functionality. The functionality must be able to store user contacts in the database and send notification emails to the administrator.

Creating a Component for the Administration Panel

First, you must create a new component for the administration panel. By using this component, the administrator will manage the user contacts. New components can be created directly in the directory *root/cmp*. However, we recommend that you create sub-directories */back* and */front* in the */cmp* directory. In this example the *root/cmp/back* and *root/cmp/front* directories are used.

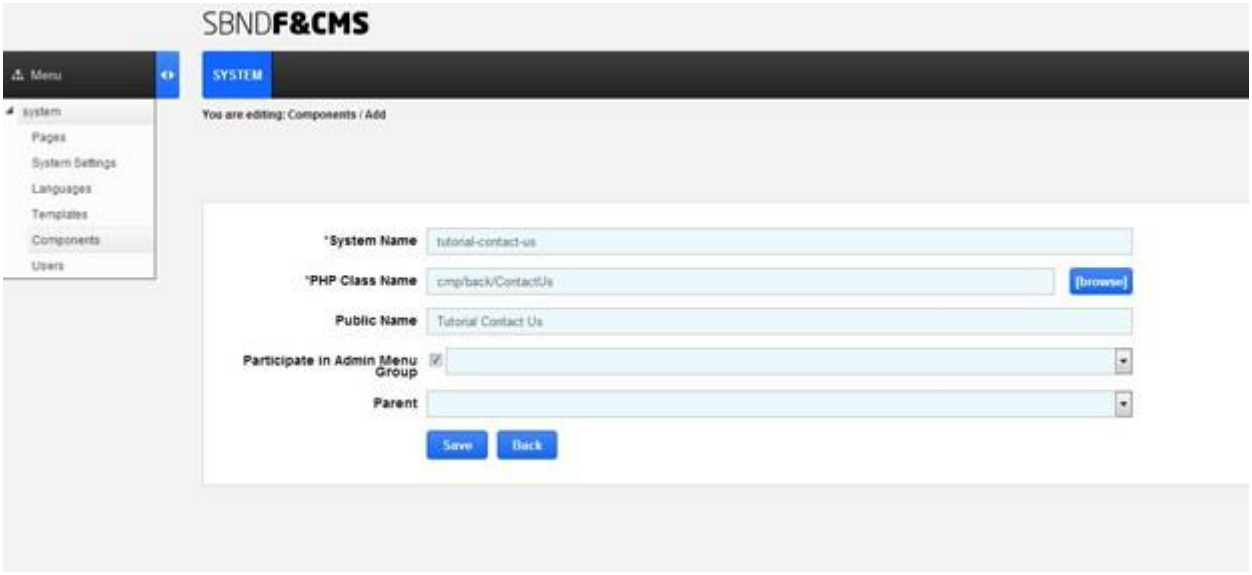
1. Create a PHP file *ContactUs.cmp.php* in the *root/cmp/back* directory. The sub-extension *cmp* is important, as it tells the system that this PHP file is a component.
2. In the file declare a class named *ContactUs* (the name of the class has to be the same as the file's name). That class extends the *CmsComponent* class:

```
1. class ContactUs extends CmsComponent{
2.   // define the name of the database table.
3.   public $base = 'tutorial_contact_us';
4.   }
```

3. Override the *main()* method and declare two component fields. For more information about component fields, see the *component-fields-options.txt* file in the documentation.

```
1. function main(){
2.   // call parent method for save main functionality.
3.   parent::main();
4.
5.   $this->setField('name', array(
6.   'text' => BASIC_LANGUAGE::init()->get('cu_name'),
7.   'perm' => '*'
8.   ));
9.   $this->setField('email', array(
10.     'text' => BASIC_LANGUAGE::init()->get('cu_email'),
11.     'perm' => '*'
12.     ));
13.     $this->setField('body', array(
14.     'text' => BASIC_LANGUAGE::init()->get('cu_body'),
15.     'perm' => '*',
16.     'formtype' => 'textarea',
17.     'dbtype' => 'text'
18.     ));
19.     }
```

4. Add the component in the administration panel. To do that, in the administration panel go to *System* → *Components*, click the *Add* button and enter the required information:



5. The default system *List* view shows columns for all components fields. You can change this as shown below:

```
1. function ActionList(){
2.   // do not need the admin to add new records
3. $this->delAction('add');
4.  // make the form only for read
5. $this->delAction('save');
6. // make sorting support
7.   $this->sorting = new BasicSorting('name', false, $this->prefix);
8.
9.   $this->map('name', BASIC_LANGUAGE::init()->get('cu_name'), 'formater');
10.     $this->map('email', BASIC_LANGUAGE::init()-
   >get('cu_email'), 'formater');
11.     $this->map('body', BASIC_LANGUAGE::init()-
   >get('cu_email'), 'cu_body');
12.
13.     return parent::ActionList();
14.     }
15.     function formater($val, $name, $row){
16.      // make click email write support
17.     if($name == 'email'){
18.     return '<a href="emilto:'.$val.'">'.$val.'</a>';
19.     }
20.     return $val;
21.     }
```

Creating a Component for the Client Side

First, you create a component that extends the *ContactUs* class (the previously created administrative component). After that you create a template file with specific formatting for the task. Finally, create a page where the *Contact Us* functionality will be presented.

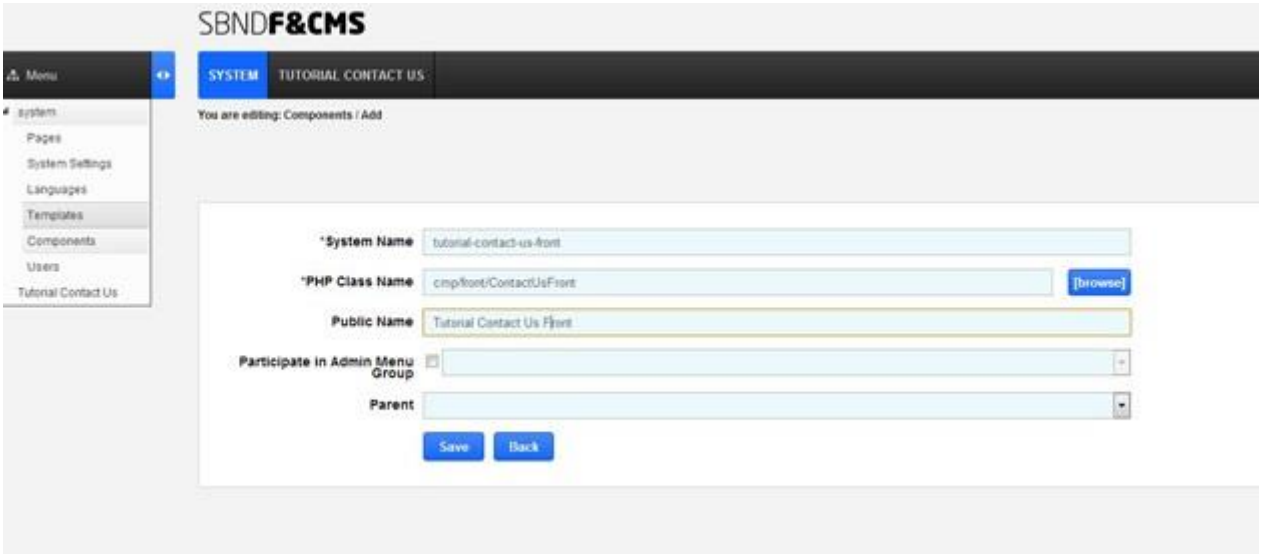
1. Create a PHP file *ContactUsFront.cmp.php* in the *root/cmp/front* directory:

```
1.   BASIC::init()->imported("ContactUs.cmp", "cmp/back")
2.   class ContactUsFront extends ContactUs {
3. }
```

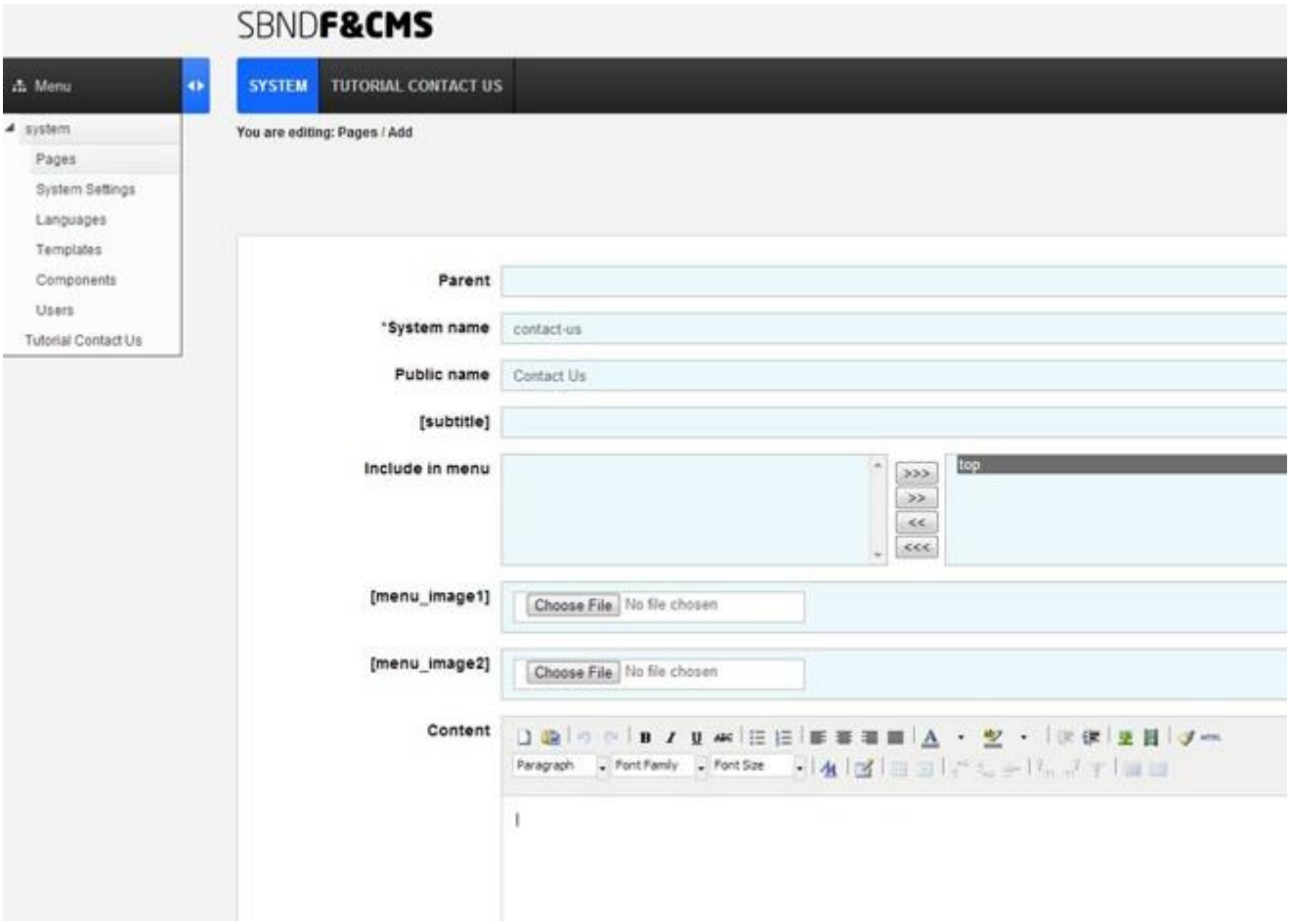
2. Configure the component to create a form by default:

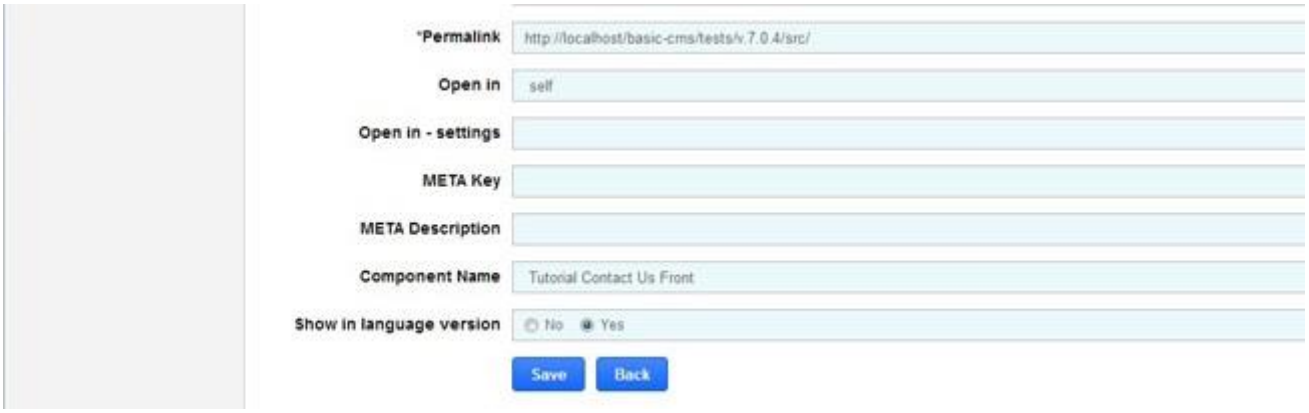
```
1. function main(){
2.   parent::main();
3.
4.   // make the form's name space. This will save form request if in the same pa
   ge have
5.   // more forms (login, search, other components form).
6.   $this->prefix = 'cu';
7.
8.   // redirect default action to form creator.
9.   $this->updateAction('list', 'ActionFormAdd');
10.
11.     // (optional) change text of the action cancel and use it like reset butt
   on.
12.     $this->updateAction('cancel', null, BASIC_LANGUAGE::init()-
   >get('cu_reset'));
13.
14.     // for security delete edit and delete actions
15.     $this->delAction('edit');
16.     $this->delAction('delete');
17.
18.     // by default the system use action edit when exist error. Because we sto
   p edit
19.     // action will change to action add.
20.     $this->errorAction = 'add';
21.     }
```

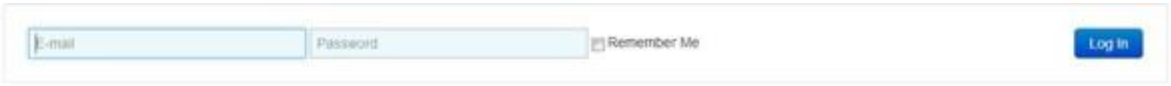
3. In the administration panel go to *System* → *Components*, click the *Add* button and enter the required information:

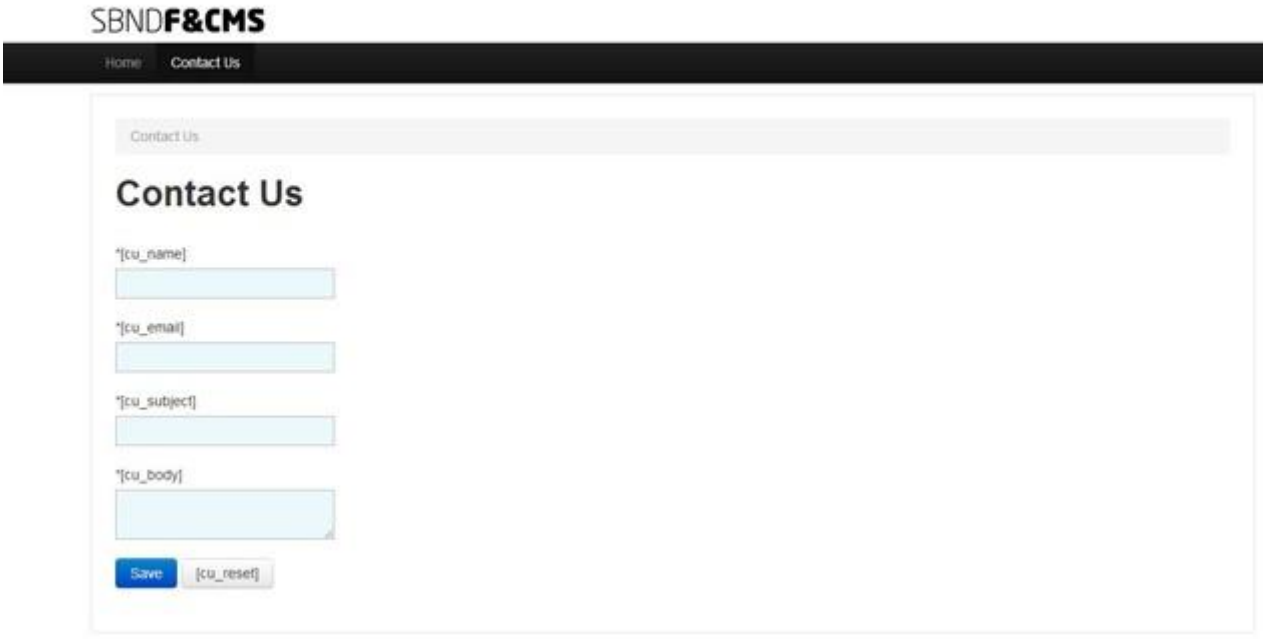


4. Create a new page for the *Contact Us* form. To do this, in the administration panel go to *System* → *Pages*, click the *Add* button and enter the required information for the new page:









Security Issues

There are two security issues that must be addressed - spam and email validation.

fcms.sbnd.net

Spam

To add captcha functionality in the form, add the following code in the *main()* method:

```
// make new field in the form.
  $this->setField('spam', array(
  'text' => BASIC_LANGUAGE::init()->get('cu_spam'),
  'formtype' => 'capcha',
  'dbtype' => 'none',
  'messages' => array(
  2 => BASIC_LANGUAGE::init()->get('invalid_sec_code')
  )
  ));

        // show method that will call after the standart validator.
        $this->specialTest = 'validator';

        // Make new method:
      function validator(){
        if(
        strtolower(BASIC_GENERATOR::init()->getControl('capcha')->
        code($this->prefix.'spam'))
        !=
        strtolower($this->getDataBuffer('spam'))
        ){
        $this->setMessage('spam', 2);
        }
        }
```

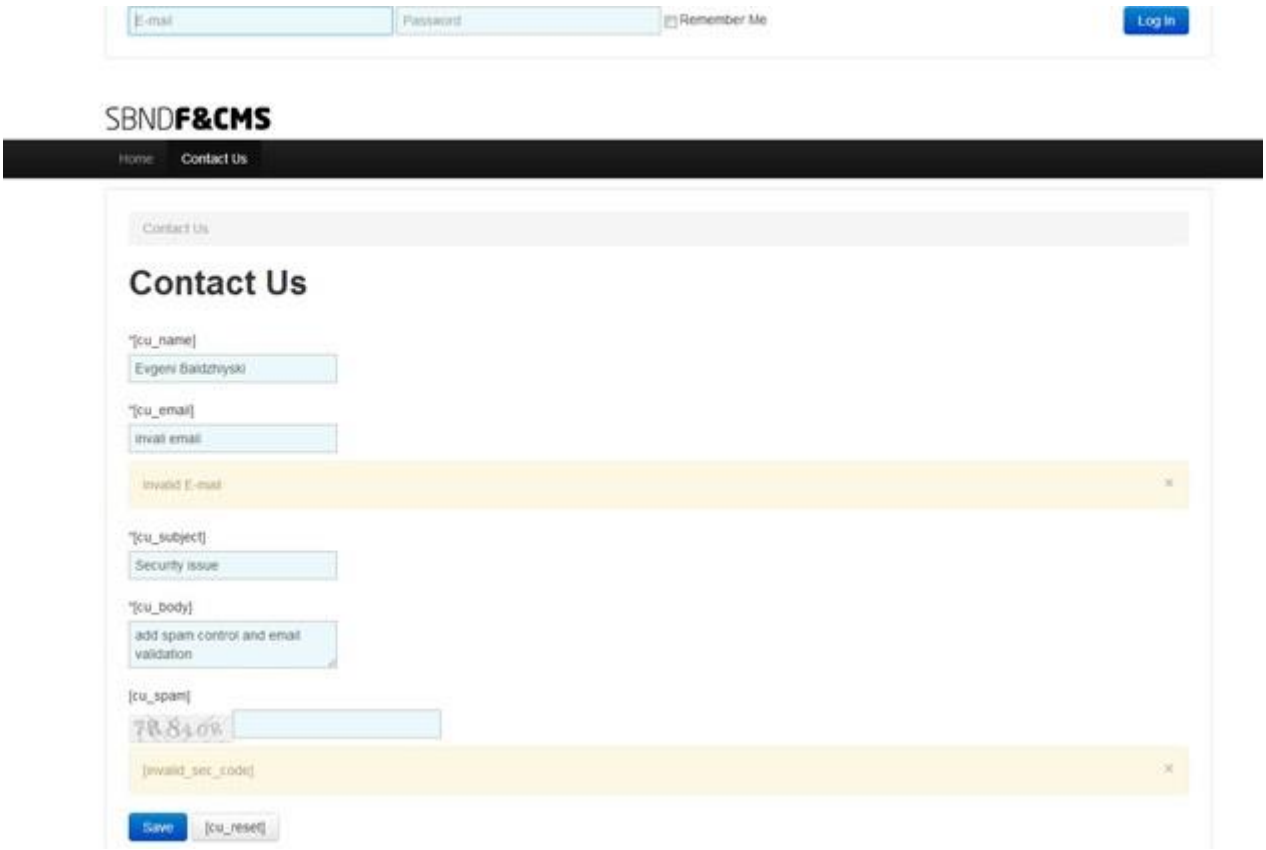
Email Validation

To add an error message for invalid emails, add the following code in the *main()* method:

```
// add error message for invalid email
  $this->updateField('email', array(
  'messages' => array(
  2 => BASIC_LANGUAGE::init()->get('invalid_email')
  )
  ));
```

In the method *validator()* add this code:

```
if(!BASIC::init()->validEmail($this->getDataBuffer('email'))){
$this->setMessage('email', 2);
}
```



Sending Notification Emails to the Administrator

```php
// extend the action save handler with send email functionality.
  function ActionSave(){
  // import spam mode.
  BASIC::init()->imported('spam.mod');

  // make email from form request
  $mail = new BasicMail(
  $this->getDataBuffer('email'),
  $this->getDataBuffer('name'),
        array(
        'subject' => $this->getDataBuffer('subject'),
        'body' => $this->getDataBuffer('body')
        )
        );
        // send to support email.
        $mail->send('support@sbnd.net');

        // make session flag thet will show message.
        BASIC_SESSION::init()->set('contact_sended', 1);

        return parent::ActionSave();
        }
        // extend the form creator to support show message after save success.
        function ActionFormAdd(){
        $this->delAction('cancel');
```

fcms.sbnd.net

```
26.
27.         if(BASIC_SESSION::init()->get('contact_sended')){
28.         BASIC_SESSION::init()->un('contact_sended');
29.
30.         BASIC_ERROR::init()->setMessage(
31.         BASIC_LANGUAGE::init()->get('contact_send_success'));
32.         }
33.         return parent::ActionFormAdd();
34.         }
35.   }
```