Creating forms with input validation the easy way, with this PHP library

One of the most time consuming and hated jobs in webdevelopment is without doubt, creating forms.

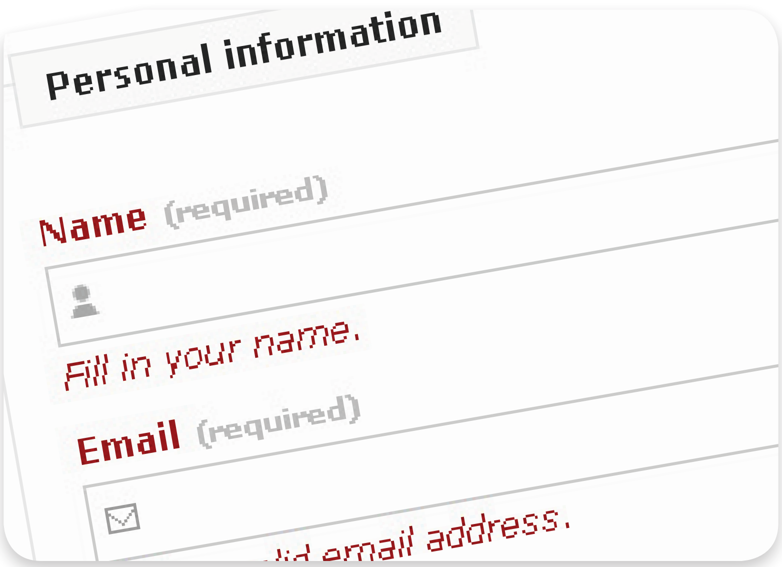
Think about the many hours you spend creating contact forms that validated the email address or that upload form that only needs to accept jpg images that are smaller then 300 by 300 pixels.

With EasyForms we make an end to this. EasyForms is a PHP library that helps you create forms on a fast and easy way including input validation.

There is a form builder included so the you can create forms by clicking and filling in some fields.

Togheter with the EasyForm library, you get 5 example forms that are created using this library:

· Contact form
· Image upload with extension and image size validation
· Quote request form
· Subscribe form that adds the email address and name to a CSV file
· Simple login script



Thank you for purchasing my script. If you have any questions that are beyond the scope of this help file, please feel free to email me at *wim@sitebase.be*. Thanks so much!

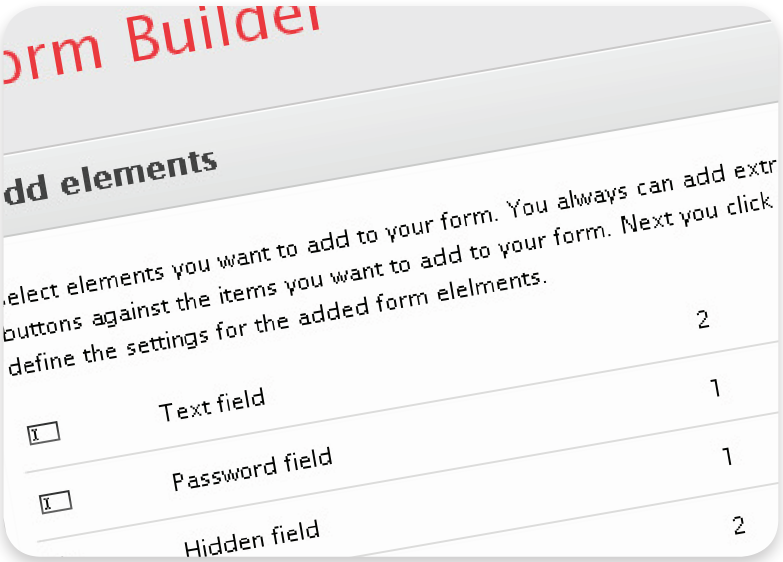
PHP Version
5.x

MySQL
Not needed

Compatible browsers
FireFox 2, FireFox 3, IE6, IE7, IE8, Safari

Using the EasyForms form builder tool

With this tool you can compose your forms by clicking an filling in some forms.



orm Builder

dd elements

elect elements you want to add to your form. You always can add extr
buttons against the items you want to add to your form. Next you click
define the settings for the added form elelments.

		2
	Text field	1
	Password field	1
	Hidden field	2

Upload all the files in the zip to your server and point your browser to the **Composer** directory.

Now you will see the Form builder. First you click on the **Add to form** buttons against the elements you want to add to your form.

Next you scroll down and edit the settings of the elements you added to your form. After your done changing the settings you click on the **save button**.

Scroll down to the form settings. Fill in a **title** (for example My mail form) and **name** (for example frmmail) and hit the generate button.

Now you will get a preview of your form. Click on the **Get source** button to get the PHP source for this form or click on the **Back** button to make changes to your form.

The different form elements and their parameters

Each type of form element has is own class in the library. Below you find a list of the different elements with a list of parameters that you can configure on that element.

Text field

Class: Base_Text

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example txtemail.
- **value:** Default value that will be shown
- **required:** If this is set to true the field must be filled in
- **validators:** Read the page about validators.
- **description:** An extra description for this field. For example: fill in your email address.

```
$Text = new Base_Text("Email", "txtemail", "", true, null, "What's your email address?");
```

Password field

Class: Base_Password

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example txtpass.
- **value:** Default value that will be shown
- **required:** If this is set to true the field must be filled in
- **validators:** Read the page about validators.

```
$Password = new Base_Password("Password", "txtpass", "", true, null);
```

Hidden field

Class: Base_Hidden

Parameters:

- **name:** A name for this field. For example txthidden.
- **value:** Default value that will be shown
- **validators:** Read the page about validators.

```
$Hidden = new Base_Hidden("txthidden", "default value", null);
```

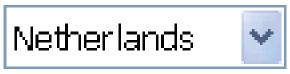
Textarea

Class: Base_Textarea

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example txtmessage.
- **value:** Default value that will be shown
- **cols:** Number of columns the textarea is width
- **rows:** Number of rows the textarea is high.
- **required:** If this is set to true the field must be filled in
- **validators:** Read the page about validators.
- **description:** An extra description for this field. For example: fill in your email address.

```
$Textarea = new Base_Textarea("Message", "txtmessage", "", 40, 5, true,
null, "Tell me a story.");
```



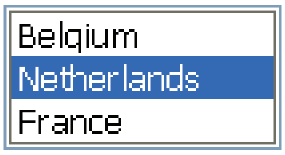
Select menu

Class: Base_Select

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example sltcountry.
- **value:** Key value array that represents the content
- **required:** If this is set to true the field must be filled in
- **default:** The default selected item
- **description:** An extra description for this field. For example: Select your country from the list.

```
$Select = new Base_Select(“Country”, “sltcountry”, array(“be” => “Belgium”, “nl” => “Netherlands”, “fr” => “France”), true, “nl”, “Select your country form the list.”);
```



Select list

Class: Base_List

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example lstcountry.
- **value:** Key value array that represents the content
- **required:** If this is set to true the field must be filled in
- **size:** Number of items visible.
- **default:** The default selected item
- **description:** An extra description for this field. For example: Select your country from the list.

```
$List = new Base_List(“Country”, “lstcountry”, array(“be” => “Belgium”, “nl” => “Netherlands”, “fr” => “France”), true, 3, “nl”, “Select your country form the list.”);
```

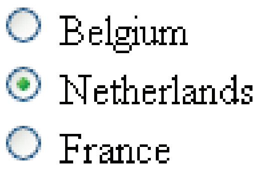
Checkbox

Class: Base_Checkbox

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example chkaccept.
- **value:** The value that you receive when the checked checkbox is submitted
- **checked:** If true the checkbox is checked
- **required:** If this is set to true the checkbox must be checked
- **description:** An extra description for this field. For example: Do you accept the terms.

```
$Checkbox = new Base_Checkbox(“Accept”, “chkaccept”, “accept“, true,
true, “Do you accept this?”);
```



Belgium
[✓ Netherlands]
France

Radiogroup

Class: Base_List

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this radio group. For example rndcountry.
- **value:** Key value array that represents the content
- **required:** If this is set to true there must be one radio button selected
- **default:** The default selected radio button
- **description:** An extra description for this field. For example: What’s your country?

```
$Radiogroup = new Base_Radio(“Country”, “rndcountry”, array(“be” =>
“Belgium”, “nl” => “Netherlands”, “fr” => “France”), true, “nl”, “What’s
your country?”);
```

File Field

Class: Base_File

Parameters:

- **label:** The label text that is shown besides the field
- **name:** A name for this field. For example fileupload.
- **required:** If this is set to true the field must be filled in
- **validators:** Read the page about validators.
- **description:** An extra description for this field. For example: Choose an avatar from your PC.

```
$Fileupload = new Base_File(“File”, “fileupload”, true, null, “Choose an
avatar from your PC.”);
```

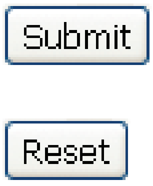
Fieldset

Class: Base_Field

Parameters:

- **label:** The label text that is shown.

```
$Fieldset = new Base_Field(“Personal information”);
```



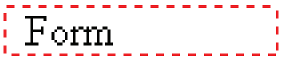
Button (submit and reset)

Class: Base_Submit and Base_Reset

Parameters:

- **name:** The label text that is shown.
- **Value:** The label on the button

```
$Submit = new Base_Submit(“btnsubmit”, “Send”);
```



Form

Class: Base_Form

Parameters:

- **label:** The label text for this form
- **name:** A name for this form. For example frmemail.
- **action:** The file that processes the form.

```
$Form = new Base_Form(“Email”, “frmemail”, “email.php”);
```

This form object is a bit different from the other EasyForm element (Text, List, ..). That’s because the form element has an **Add Method**. With this method you can add other elements to it.

For example if we want to make an email form we first create an email, subject and message field. Then we create a Form element and we add the email, subject and message field to the form. In code this will look like:

```
<?php
include(“lib/init.php”);

$Email = new Base_Text(“Email”, “txtemail”, “”, true, null, “What’s your
email address?”);
$Subject = new Base_Text(“Subject”, “txtsubject”, “”, true, null, “What is
the subject of your email?”);
$Message = new Base_Textarea(“Message”, “txtmessage”, “”, 40, 5, true,
null, “Tell me your question.”);
$Submit = new Base_Submit(“btnsubmit”, “Send”);

$Form = new Base_Form(“Email Form”, “frmemail”);
$Form->Add($Email);
$Form->Add($Subject);
$Form->Add($Message);
$Form->Add($Submit);

echo $Form;
```

Validating the form input with validators

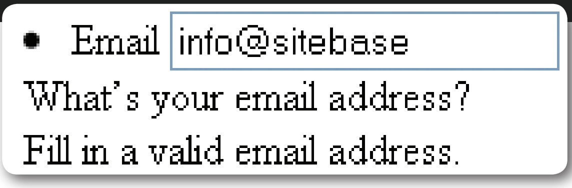
The previous email form was a good example of how you can create a simple email form that holds the values of the field after submitting and doing some required field validation. But of course in some cases we want some advanced input validation. For example, in case of the email form we want to validate the email input to see if it is indeed an email address.

Therefore we can use validators. This objects you can add to an element in the validators parameter. Below you'le find the code to add an email validator to the email field.

```
$EmailValidator = new Base_Validators_Email("Fill in a valid email address.");

$Email = new Base_Text("Email", "txtemail", "", true, $EmailValidator, "What's your email address?");
```

You can see that we create a new email validator with as parameter the error that should be shown to the user when it's not a valid email address.



- Email info@sitebase

What's your email address?

Fill in a valid email address.

It's also possible to add more than one validator to an input field. For example if we want to validate if it's an email address and we want that the email address contains the string "hotmail.com" we can use this code.

```
$EmailValidator = new Base_Validators_Email("Fill in a valid email address.");
$ContainsValidator = new Base_Validators_Contains("It must be an hotmail address.", "hotmail.com");

$Email = new Base_Text("Email", "txtemail", "", true, array($EmailValidator, $ContainsValidator), "What's your email address?");
```

You can see that we created two validators and added them as an array to the email field.

An overview of the validators and their parameters

Email validation

Class: Base_Validators_Email

Parameters:

- **message:** The message to show when invalid

Number validation

Class: Base_Validators_Integer

Parameters:

- **message:** The message to show when invalid
- **min:** The minimum value of the number
- **max:** The maximum value of the number

If you don’t want to use a parameter, set it to null.

Contains validation

Class: Base_Validators_Contains

Parameters:

- **message:** The message to show when invalid
- **contains:** The string that the input must contain to be a valid input.

File validation

This validator can be used to validate a file from an filefield. You can limit the size or the type of file. This type of validator can be added to a Base_File element.

Class: Base_Validators_File

Parameters:

- **message:** The message to show when invalid
- **maxsize:** The maximum size of the file in bytes
- **extensions:** An array of allowed extensioins. For example you can use array(“jpg”,”png”, “gif”);
- **mimes:** An array of allowed mime types.

If you don’t want to use a parameter, set it to null.

Image validation

This validator can be used to validate the height and width of an image.

Class: Base_Validators_Image

Parameters:

- **message:** The message to show when invalid
- **width:** The maximum width
- **height:** The maximum height

If you don’t want to use a parameter, set it to null.

Processing the form data when it is valid

The last thing to do is processing the form data when it’s valid. Therefore we create a function somewhere in the PHP file that receives one parameter. This parameter contains the valid form data.

```
<?php
include(“lib/init.php”);
$EmailValidator = new Base_Validators_Email(“Fill in a valid email address.”);
$ContainsValidator = new Base_Validators_Contains(“It must be an hotmail address.”, “hotmail.com”);
$Email = new Base_Text(“Email”, “txtemail”, “”, true, array($EmailValidator, $ContainsValidator), “What’s your email
address?”);
$Subject = new Base_Text(“Subject”, “txtsubject”, “”, true, null, “What is the subject of your email?”);
$Message = new Base_Textarea(“Message”, “txtmessage”, “”, 40, 5, true, null, “Tell me your question.”);
$Submit = new Base_Submit(“btnsubmit”, “Send”);

$Form = new Base_Form(“Email Form”, “frmemail”);
$Form->Add($Email);
$Form->Add($Subject);
$Form->Add($Message);
$Form->Add($Submit);
$Form->SetPhpCallback(“formHandle”);

if(!$Form->Processed()){
        echo $Form;
}else{
        echo “<p>Thank you</p>”;
}
function formHandle($data){
        echo $data[‘txtemail’] . “<br />”;
        echo $data[‘txtsubject’] . “<br />”;
        echo $data[‘txtmessage’] . “<br />”;
}
```

Notice the $Form->SetPhpCallback(“formHandle”); line. This tells the form that when it’s submitted and all values are valid. That it must execute the formHandle function with as first parameter an array of all input data.

The regular echo $Form is also replaced with an if else construction to hide the form and show a message to the user when the form is valid submitted.

To test this script you can test the file /Examples/DocumentationExample.php.

Overview of the CSS classes to style your form.

In the image you see an example of the code that is generated by EasyForm. This is the form we created in previous page.

```
⊟ <form id="frmemail" enctype="multipart/form-data" method="post" action="" name="frmemail">
    ⊟ <fieldset>
          <legend>Email Form</legend>
        ⊟ <li id="li_txtemail" class="error">
              <label id="label_txtemail" class="required" for="txtemail">Email</label>
              <input id="txtemail" class="required" type="text" value="" name="txtemail"/>
              <div id="description_txtemail" class="description">What's your email address?</div>
              <div id="error_txtemail" class="errormessage">Fill in a valid email address.</div>
          </li>
        ⊞ <li id="li_txtsubject">
        ⊞ <li id="li_txtmessage" class="error">
        ⊞ <li id="li_btnsubmit">
          <input id="formtag" type="hidden" value="frmemail" name="formtag"/>
      </fieldset>
  </form>
```

If an input field is submitted and its not valid the parent li element will get an error class and the error message will show up. Like you can see on the li_txtemail.

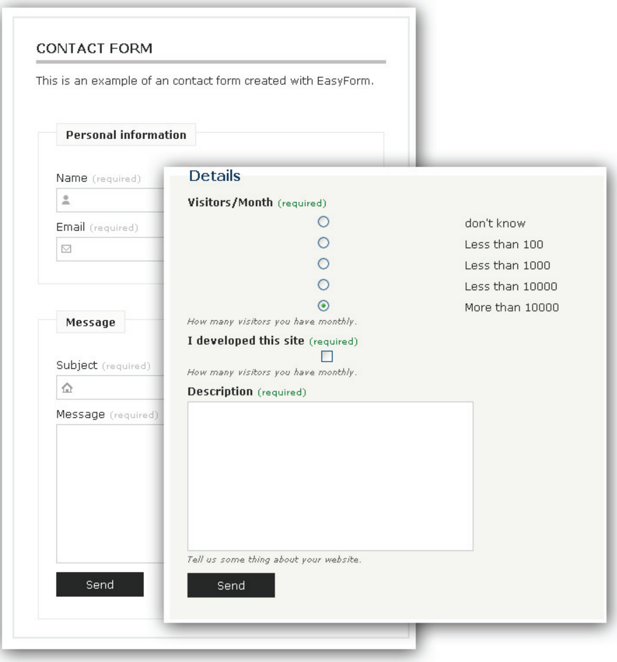
As you can see the li_txtsubject is valid because it hasent got a class="error".

If the input field is required the label and the input contains a class required.

If you filled in the description for your element, the li item contains a div with class description.

With a bit of imagination and CSS knowledge you will create beautifull forms in no time.

Look at the examples for some inspiration.



CONTACT FORM

This is an example of an contact form created with EasyForm.

Personal information

Name (required)

Email (required)

Message

Subject (required)

Message (required)

Send

Details

Visitors/Month (required)

○	don't know
○	Less than 100
○	Less than 1000
○	Less than 10000
◉	More than 10000

How many visitors you have monthly.

I developed this site (required)

☐

How many visitors you have monthly.

Description (required)

Tell us some thing about your website.

Send