symfony

Open-Source PHP5 MVC Framework
Agile Development

Helpers

PART 1 - JAVASCRIPT(*JS*) and AJAX (remote calls)

<?php echo use_helper('Javascript') ?>

JAVASCRIPT AND AJAX HELPERS

JavaScript Helpers

link_to_function ($name, $function, $html_options=array())

```
<?php echo link_to_function("Click me!", "alert('foobar')") ?> // will generate:
<a href="#" onClick="alert('foobar');return none;">Click me!</a>
```

javascript_tag ($content)

```
<?php echo javascript_tag("document.getElementById('indicator').innerHTML=
 '<strong>Data processing complete</strong>';") ?>
```

update_element_function ($element_id, $options=array())

```
<?php echo javascript_tag( update_element_function( 'indicator', array(
"position"=>"after", "content" =>"<strong>Data processing complete</strong>" )))?>
```

Ajax Helpers

An AJAX interaction is made up of three parts:
* **a caller** (a link, button or any control that the user manipulates to launch the action)
* **a server action**
* **a zone in the page to display the result of the action to the user.**

Symfony provides multiple helpers to insert AJAX interaction in your templates by putting the **caller** in a link, a button, a form, or a clock. These helpers output HTML code, not JavaScript.

link_to_remote ($name, $options=array(), $html_options=array())

```
<?php echo link_to_remote('Delete this post', array(
    'update' => 'indicator',   'url'  => 'post/delete?id='.$post->getId()  )) ?>
```

remote_function ($options=array())

```
<?php echo javascript_tag(remote_function(array(
    'update'  => 'myzone',
    'url'      => 'mymodule/myaction'
 ))) ?>
```

change part of the page according to a server response

form_remote_tag ($options=array(), $options_html=array())

```
<?php echo form_remote_tag(array(
    'update'   => 'item_list',   'url'  => '@item_add'  )) ?>
  <label for="item">Item:</label>
  <?php echo input_tag('item') ?>
  <?php echo submit_tag('Add') ?>
</form>
```

*opens a <form>, just like the **form_tag()** helper does.*

observe_field ($field_id, $options=array())

```
<?php echo form_tag('@item_add_regular') ?>
  <label for="item">Item:</label>
  <?php echo input_tag('item') ?>
  <?php echo submit_tag('Add') ?>
  <?php echo observe_field('item', array(
     'update'   => 'item_suggestion',
     'url'         => '@item_being_typed'
  )) ?>
</form>
```

the module/action written in the @item_being_typed rule will be called each time the item field changes, and the action will be able to get the current item value from the value request parameter. If you want to pass something else than the value of the observed field, you can specify it as a JavaScript expression in the 'with' parameter

periodically_call_remote ($options=array())

```
<?php echo periodically_call_remote(array(
    'frequency' => 60,
    'update'    => 'notification',
    'url'       => '@watch',
    'with'      => "'param=' + $('mycontent').value"
)) ?>
```

this helper is an AJAX interaction triggered every x seconds. It is not attached to a HTML control, but runs transparently in the background, as a behaviour of the whole page

Another functions

- submit_to_remote ($name, $value, $options=array())
- evaluate_remote_response ()
- observe_form ($form_id, $options=array())
- visual_effect ($name, $element_id=false, $js_options=array())
- sortable_element ($element_id, $options=array())
- draggable_element ($element_id, $options=array())
- drop_receiving_element ($element_id, $options=array())
- javascript_cdata_section ($content)
- input_auto_complete_tag ($name, $value, $url, $tag_options=array(), $completion_options=array())
- input_in_place_editor_tag ($name, $url, $editor_options=array())

REMOTE CALL PARAMETERS

All the AJAX helpers can take other parameters, in addition to the **update** and **url** parameters:

position

The position parameter can be defined as:

Value	Position
before	before the element
after	after the element
top	at the top of the content of element
bottom	bottom of the content of element

conditions

confirm

'confirm' => 'Are you sure?'
A ***JS*** dialog box showing '*Are you sure?*' will pop-up when the user clicks on the caller, and the ***module/action*** will be called only if the user confirms his choice by clicking 'Ok'.

condition

'condition' => "$('elementID') == true",
The remote call can also be conditioned by a test performed on the browser side (in JavaScript).

script execution

'script' => true
If the response code of the AJAX call (the code sent by the server, inserted in the update element) contains ***JS***, these scripts are not executed by default. This is to prevent remote attack risks.The ability to execute scripts in remote responses explicitly with the script option.

callbacks

Callback	Event
before	Before request is initiated
after	Immediately after request is initiated and before loading
loading	When the remote response is being loaded
loaded	When the browser has finished loading the remote response
interactive	When the user can interact with the remote response, even though it has not finished loading
success	When the XMLHttpRequest is completed, and the HTTP status code is in the 2XX range
failure	When the XMLHttpRequest is completed, and the HTTP status code is not in the 2XX range
404	When the request returns a 404 status
complete	When the XMLHttpRequest is complete (fires after **success** or **failure**, if present)

```
e.g.: <?php echo link_to_remote('Delete this post', array(
          'update'     => 'indicator',
          'url'        => 'post/delete?id='.$post->getId(),
          'position'   => 'after',
          'confirm'    => 'Are you sure?',
          'script'     => true
          'loading'    => "Element.show('indicator')",
          'complete'   => "Element.hide('indicator')"  )) ?>
```

NOTES

* **Actions** called as **remote functions** know that they are in an AJAX transaction, and therefore automatically **don't include** the **web debug toolbar** in development. Also, they skip the decoration process (their template is not included in a layout by default). If you want an Ajax view to be decorated, you need to specify explicitly **has_layout: true** for this view in the module **view.yml** file. Actions called through Ajax, return **true** to the following call:

```
$isAjax = $this->isXmlHttpRequest();
```

* The AJAX helpers won't work if the URL of the remote action doesn't belong to the same domain as the current page. This restriction exists for security reasons, and relies on browsers limitations that cannot be bypassed.

The Prototype and Script.aculo.us JavaScript libraries are bundled with the symfony.