sfSmartyView, using Smarty with the symfony project...

Smarty(http://smarty.php.net) is a well known template engine.

This is an alpha version of a smarty plugin for the symfony project. I would recommend at the moment that you install it only in the sandbox, as it has not been tested, and it is my first plugin, so it may leave undesired remains when installed globally.

How to install it in the sandbox

```
$ cd sf_sandbox
$ symfony plugin-install http://plugins.symfony-project.com/sfSmartyViewPlugin
$ symfony cc
```

Create a file called `module.yml` in the main or in the application's config directory, which looks like this:

```
all:
  view_class: sfSmarty
```

Creating the templates for Smarty

For the Smarty templates you have to use the same filenames like for the normal symfony templates, but with the extension **.tpl**. This means we need a layout.tpl in application/templates and a indexSuccess.tpl (or whatever the action is called) in application/module/templates.

layout.tpl:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>

{include_http_metas}
{include_metas}

{include_title}

<link rel="shortcut icon" href="/favicon.ico" />
</head>
<body>
{$sf_data->getRaw('sf_content')}
</body>
</html>
```

indexSucess.tpl

```
<div>
<h2>You did it…</h2>
<p>this is the first Smarty template</p>
</div>
```

You will find those templates in the templates directory in the sfSmartyView plugin directory.

How can I use it in the sandbox?

First create the file module.yml in sf_sandbox/config like mentioned above. If you try to run it now with http://server/sf_sandbox/web/frontend_dev.php, you will receive an error:

```
The template "/indexSuccess.tpl" does not exist in:
```

That's easy, the next step is to put the file indexSuccess.tpl from above in the directory sf_sandbox/apps/frontend/modules/default/templates. Run it again. Now it complains about

```
The decorator template
"/var/www/localhost/htdocs/sf_sandbox/config/../data/symfony/m
odules/default/templates/defaultLayout.tpl" does not exist or
is unreadable“
```

Ok this is a problem, because the path is hardcoded in the default action :(. Edit the file sf_sandbox/data/symfony/modules/default/actions/actions.class.php and delete line 23. That's the line that starts with $this->setLayout(..., where the template is hardcoded.
Run it again, this time we get another error:

```
The decorator template
"/var/www/localhost/htdocs/sf_sandbox/apps/frontend/templates/
layout.tpl" does not exist or is unreadable
```

Much better, because we have only to put the file layout.tpl in the directory mentioned in the error message: sf_sandbox/apps/frontend/templates.
Run it again and hurray, it is working!
Maybe you can't read the text, because it brown text on brown background, but I am pretty sure that you know how to change that.

Ok, I have installed it...

This is an alpha version, this means I am not sure that the interface will stay the same. But for the moment it works like this:
To use a helper, you have to declare it like this:

```
{use helper=”helpername”}
```

this provides the same functionality like use_helper(‘helpername’);

Let’s say you action looks like this:

```
<?php
class testActions extends sfActions {
      public function indexAction(){
            $this->id = 100;
            $this->html = “<p>this is html</p>”;
            $this->text = “Hello”;
      }
}
```

```
?>
```

then you could use a indexSuccess.tpl file like this:

```
<div>
Id: {$id}<br>

{link_to name=”show index” internal_uri=”test/index” options=array('id' =>
'myid', 'confirm' => 'Are you sure?', 'absolute' => true)}

{$html|esc_entities}

{use helper=”javascript”}
{javascript}
alert(“{$text}”);
{/javascript}
</div>
```

Let’s go through it line by line :

1. <div> - normal html
2. {$id} - this is normal smarty functionality, $id was set in the action and is simply displayed by the template
3. empty
4. link_to… - this is a symfony helper. All symfony helpers can be accessed like this: the helper function is the smarty tag, and then you can use all arguments as field names. In this case the function definition look likes this:
   function link_to ($name = '', $internal_uri = '', $options = array())
   which means that the field names available are name, internal_uri and options. You need only the fields where there is no default value.
5. empty
6. This is the normal smarty variable that was set in the action, but the symfony helper is used as a modifier. This doesn’t work for every helper, but only for those where this functionality is implemented. See further below.
7. empty
8. use_helper(‘javascript’) in “symfony”
9. this is the javascript_tag() helper. This shows that a helper can also be implemented as a block function, where the contents is between the opening and the closing tag. This doesn’t work for every helper, but only for those where this functionality is implemented. See further below.
10. normal javascript, but the text is exchanged by smarty
11. closing tag for the javascipt_tag helper
12. </div>

the output would be

```
<div>
Id: 100<br>

<a href="http://myapp.example.com/test/index" id="myid" onclick="return
confirm('Are you sure?');">show index</a>

&lt;p&gt;this is html&lt;/p&gt;

<script type="text/javascript">
//<![CDATA[
  alert(“Hello”);
//]]>
```

```
</script>
</div>
```

How does it work?

SfSmartyView extend sfPHPView and overloads the classes where Smarty functionality is needed. Each time a template is changed Smarty compiles this template so that the symfony helpers are called directly, so using Smarty doesn't create a lot of overhead (I think, somebody should test this).
There is a helpers directory where sfSmartyView looks for custom implementations for symfony helpers, so if you think a certain symfony helper is better implemented as a modifier, you just have to change the class SmartyHelper.class.php, where Helper has to be exchanged with the actual helper name that you want to change. I have created three custom function, one for a modifier function, a normal function and a block function, so extending these classes should not be too complicated.

Configuration

You can put some configuration settings in the file

```
myproject/apps/myapp/config/app.yml
```

The Smarty library is expected in a directory called Smarty somewehre in the include_path. To define the path, put this in app.yml:

```
all:
  sfSmartyView:
    class_path: /path/to/the/file/smarty.clas.php
```

To define the extension of you template files use (include the dot for the extension!) (default: .tpl):

```
all:
  sfSmartyView:
    template_extension: .tpl
```

For the Smarty cache directory (default: default symfony cache directory + /Smarty)

```
all:
  sfSmartyView:
    cache_dir: /path/to/the/cache
```

If you want to switch on the template security, so that the template designers cannot use PHP in the templates use (default: false)

```
all:
  sfSmartyView:
    template_security: true
```

What now?

Please test this plugin and give me your feed back. I would like to have this plugin in beta status as soon as possible. Please put your ideas in this thread:

http://www.symfonyproject.com/forum/index.php?t=rview&goto=1454

As mentioned before, use it only on a test system, this is alpha code and I have no idea what it might break, use it at your own risk ;)

Georg