

Documentazione Autorender

Indice

- 1.Introduzione**
- 2.Struttura delle cartelle**
- 3.Algoritmo di esecuzione**
- 4.Plugin**
- 5.Modulo d'esempio**
- 6.Dettagli sui Template**
- 7.NOTE**

1. Introduzione

Il modulo Autorender è un motore grafico di templating basato sia sul popolare motore AutoTheme di Shawn McKenzie e che sul motore Smarty di smarty.php.net Autorender vuole coniugare la potenza dei motori AutoTheme e Smarty in un unico modulo. Permette il supporto dei temi AutoTheme senza significative modifiche e tuttemlemcaratteristiche di Smarty.

Il supporto per Smarty permette inoltre la piena portatilità dei moduli basati su pnRender di PostNuke.

2. Struttura delle Cartelle

Autorender ha la seguente struttura delle cartelle:

Root MdPro

--libs	
--render_smarty	contiene il codice base di Smarty senza modifiche
--plugins	contiene i plugin di Smarty per l'intero sito
--smarty_core	contiene plugin base di smarty plugins degli sviluppatori
--api	contiene plugin dipendenti dalle API di PostNuke e MDPro
--modules	
--Autorender	Contiene il codice AutoTheme modificato per supportare Smarty.
--Tuo modulo personalizzato	
--templates	Contiene i template Autorender del tuo modulo personalizzato
--plugins	Contiene i plugins perr Autorender del tuo modulo
--cache	
--_arcompile	Contiene i template compilati per Autorender, deve essere 777
--_cache	Contiene pagine cache di smarty, deve avere permessi 777

3. Algoritmi di esecuzione.

Fondamentalmente, Autorender può lavoroqare in due modi:

Primo modo – riceve il template, esegue il parsing, cioè lo elabora con smarty, lo interpreta come un template AutoTheme e lo elabora sotto AutoTheme, poi lo ritorna all'utente. Questo metodo supporta tutte le costanti inserite nel template, siano esse di Smarty e/o di Autotheme, ma è più lento nell'elaborazione che se si usasse solo Smarty o solo AutoTheme.

Secondo Modo – riceve il template, lo elabora con Smarty, poi lo ritorna all'utente. Questo metodo supporta solo i tag di Smarty, ma lavora più velocemente che il primo metodo. Se ne raccomanda l'uso nei moduli o nei blocchi sensibili alla velocità, che non richiedono tag di AutoTheme.

4. Plugins.

I plugin consentono di creare un comando o un modificatore personalizzato da usare all'interno del template.

I plugin si possono dividere fondamentalmente in

- ⊗ plugin per Smarty
- ⊗ plugin per le API
- ⊗ plugin per i Moduli.

A. I plugin di Smarty si trovano nella cartella `libs/render_smarty/plugins/smarty_core` e sono necessari per utilizzare le funzionalità base di Smarty.

a) Comandi

nome del file di plugin:
`function.NomeTuaFunzione.php`
uso nel template:
`<% NomeTaFunzione
 NomeParametro1=ValoreParametro1
 NomeParametro2=ValoreParametro2
%>`

b) Modificatori

nome del file di plugin:
`modifiers.NomeTuoModificatore.php`
uso nel template:
`<% $valore|NomeTuoModificatore %>`

a) Alcuni utili esempi di **sintassi Smarty forniti da un comando:**

```
<% section loop=$myarray name="myvar" %>  
    <% $myarray[myvar].array_element %> <br>
```

<%sectionelse%>

No data found

<%/section%>

Esegue un loop sulle voci di un vettore

<http://smarty.php.net/manual/en/language.function.section.php>

<%if \$condition ne "" %>

<% \$condition %> is not null

<%else%>

<% \$condition %> is null

<%/if%>

Controlla se la condizione è vera o falsa

Supporta le operazioni base (eq,ne,gt,lt,le,ge,not e analoghi matematici)

<http://smarty.php.net/manual/en/language.function.if.php>

<% include file="tuofile.tpl" %>

Includes another template into current one

<http://smarty.php.net/manual/en/language.function.include.php>

<%php%>

echo "php code works";

<%/php%>

Include codice php nel template. Non si raccomanda l'uso ;)

<http://smarty.php.net/manual/en/language.function.php.php>

Ecco alcune utili funzioni core supportati da Smarty (definite come plugins):

<% config_load file='dfs' section='dsfsd'

scope='fsd' global=true %>

è usata per caricare le variabili di configurazione dal file di configurazione al file del template.

<% counter name='dfs' start=1 skip=2

direction=up print=false assign=varname %>

Registra il contatore per ogni iterazione e viene usato per stampare tale indice.

<http://smarty.php.net/manual/en/language.function.counter.php>

<% cycle name="gfdg" values="blue,green,red" print="false"

delimiter=", " assign="varname" reset="true" %>

Viene usato per far ruotare una serie di valori (funziona come il foreach del PHP).

<http://smarty.php.net/manual/en/language.function.cycle.php>

<% debug output="La tua frase qui" %>

inserisce la console di debug nella pagina.

<http://smarty.php.net/manual/en/language.function.debug.php>

<% eval var="\$x+1" assign="varname"%>

usato per valutare una variabile come un template smarty.

<http://smarty.php.net/manual/en/language.function.eval.php>

<% fetch file="http://posizione.file/file" assign="varname" %>
Usato per scaricare file da un file system locale, http oppure ftp e visualizza o ne assegna il contenuto.
<http://smarty.php.net/manual/en/language.function.fetch.php>

**<% html_checkboxes name="outputname"
options=\$associated_array_values
selected=\$string_or_array_of_values %>**

è una funzione personalizzata che crea gruppi di checkbox con valori forniti dal creatore.
<http://smarty.php.net/manual/en/language.function.html.checkboxes.php>

**<% html_options
name="outputname"
values=\$array_of_values
selected=\$string_or_array %>**
Funziona come html_checkboxes, ma per liste di tipo select
<http://smarty.php.net/manual/en/language.function.html.options.php>

**<% html_radios name="outputname"
values=\$array_of_values
selected=\$string_or_array %>**

Funziona come html_checkboxes, ma per radio buttons
<http://smarty.php.net/manual/en/language.function.html.radios.php>

**<% html_select_date prefix="mydate_"
time="2005-12-31"
start_year="1970"
end_year="2020" %>**
Crea menu tendine a discesa per scegliere tra date.
<http://smarty.php.net/manual/en/language.function.html.select.date.php>

**<% html_select_time
prefix="mytime_"
time=\$unixtimestamp
%>**
Crea menu tendine a discesa per scegliere tra date.
<http://smarty.php.net/manual/en/language.function.html.select.time.php>

<% html_table loop=\$array_of_data cols=10 rows=20 %>
Inserisce un vettore in una tabella HTML.
<http://smarty.php.net/manual/en/language.function.html.table.php>

**<% html_image file="image_path" height=10 width=20
alt="Alt name" href="http://link.here" %>**
Crea il tag immagine dai dati inseriti.
Non so se ciò possa essere utile nel nostro caso...
<http://smarty.php.net/manual/en/language.function.html.image.php>

**<% mailto address="me@maxdev.com" text="Contattaci"
subj="Mail from site" %>**
Crea il tag mailto per i dati inseriti.
<http://smarty.php.net/manual/en/language.function.mailto.php>

<% math equation="x*y" x=2 y=3 format="%d" %>
Calcola una equazione matematica.
<http://smarty.php.net/manual/en/language.function.math.php>

Tutte le informazioni dettagliate sulle sopra descritte funzioni possono essere trovate sul sito di Smarty:

<http://smarty.php.net/manual/en/>

b) Ed ecco alcuni **modificatori core per Smarty**:

truncate:20:"...":true

Limita la lunghezza dei byte stampati alla lunghezza di 20 caratteri e inserisce '...' in fondo.

<http://smarty.php.net/manual/en/language.modifier.truncate.php>

nl2br

Converte il line feeds nel tag

<http://smarty.php.net/manual/en/language.modifier.nl2br.php>

date_format:"%B %e, %Y %H:%M:%S"

Visualizza in output il unixtimestamp in formato stringa data.

<http://smarty.php.net/manual/en/language.modifier.date.format.php>

strip_tags

Rimuove tag HTML

<http://smarty.php.net/manual/en/language.modifier.strip.tags.php>

Tutte le informazioni dettagliate sui modificatori possono essere trovate sul sito Smarty:

<http://smarty.php.net/manual/en/>

B. Funzioni e modifcatori API per PostNuke/MdPro

Esistono diversi modificatori speciali e diverse funzioni in supporto alle funzionalità che dipendono dalle API di PosNuke e di MdPro. Sotto PostNuke essi vengono usate da pnRender.

a. Funzioni

<% opentable %>

<% closetable %>

<% opentable2 %>

<% closetable2 %>

Tag per aprire e chiudere due tipi standard di tabelle.

<% const name=_YOUR_CONSTANT_NAME%>

Visualizza una costante (dovrebbe essere dipendente dalla lingua)

```
<% pager %>
```

Mostra il pager con i parametri che vengono passati.

```
<% pblock id=45 %>
```

Il contenuto in output di un blocco di cui è specificato l'id (in un sistema che permette di aggiungere blocchi da admin->Blocks)

```
<% pnconfiggetvar name="param_name" %>
```

Stampa i valori di configurazione per un nome di parametro dato

```
<% pmodapifunc modname="YourModuleName" type="user"  
func="your_func_name" param1="val1"  
param2="val2" %>
```

Chiama una funzione API per un modulo, un tipo e un parametro inserito

```
<% pmodgetvar module="YourModuleName" name="ParamName" %>
```

Riceve una variabile di configurazione del modulo

```
<% pnvarcleanfrominput name="ParamName" %>
```

Riceve parametri di input (post/get)

```
<% pnusergetvar uid="2" name="ParamName" %>
```

Riceve parametri passati dall'utente

```
<% pnsecgenauthkey module="YourModuleName" assign=$variable_name %>
```

Genera AuthKey (richiesta in certe operazioni di modifica/cancella/aggiorna)

b. Modificatori:

pnvarpreforstore

Prepara le variabili alla memorizzazione

pnvarprefordisplay

Prepara le variabili alla visualizzazione

pnvarprehtmldisplay

Prepara le variabili per la visualizzazione come codice HTML