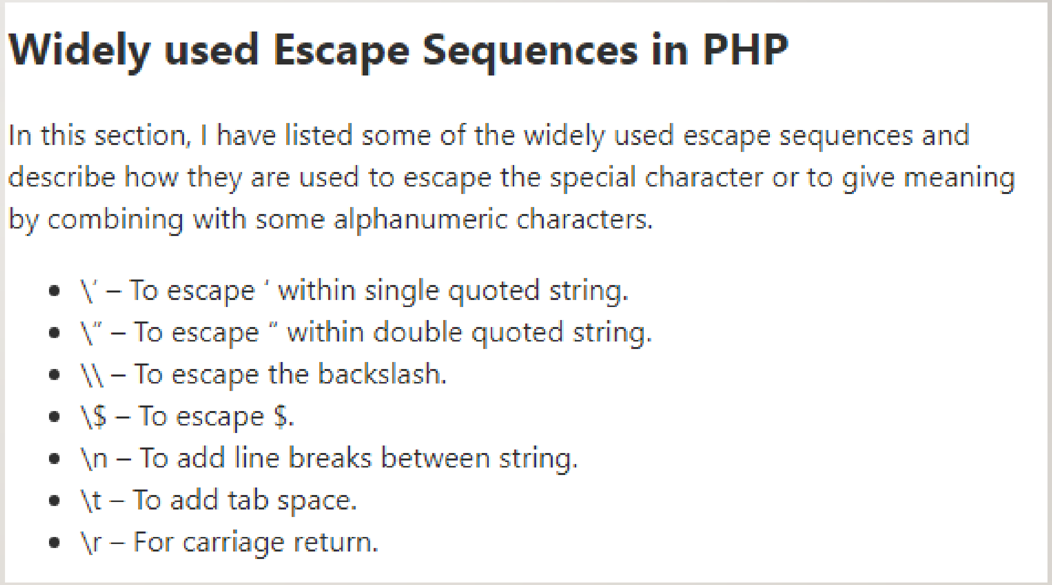
EXPLORATION OF SUPERGLOBALS IN SMARTY

WHAT IS ESCAPE?

- Escape is a variable modifier that is used to encode or escape a variable to for example *html, url, single quotes, hex, hexentity, javascript,* and *mail*. By default, the escape is used by *html*.
- We can escape all template variable output by wrapping it in *htmlspecialchars({$output}, ENT_QUOTES, SMARTY_RESOURCE_CHAR_SET)*;, which is the same as {*$variable|escape:"html"*}

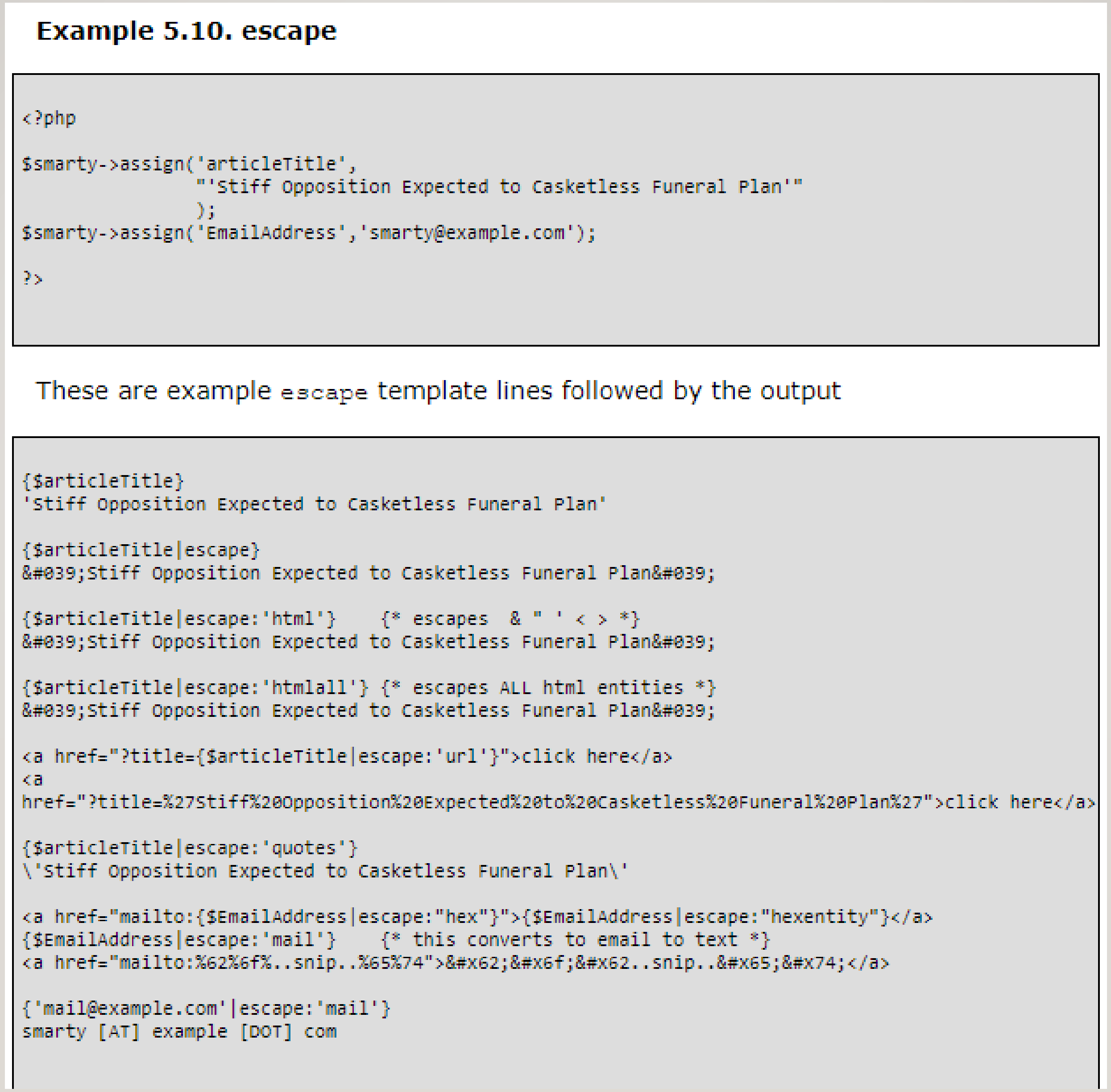


Widely used Escape Sequences in PHP

In this section, I have listed some of the widely used escape sequences and describe how they are used to escape the special character or to give meaning by combining with some alphanumeric characters.

- \' – To escape ' within single quoted string.
- \" – To escape " within double quoted string.
- \\ – To escape the backslash.
- \$ – To escape $.
- \n – To add line breaks between string.
- \t – To add tab space.
- \r – For carriage return.

&amp; becomes & (ampersand)
&quot; becomes " (double quote)
&#039; becomes ' (single quote)
&lt; becomes < (less than)
&gt; becomes > (greater than)



Example 5.10. escape

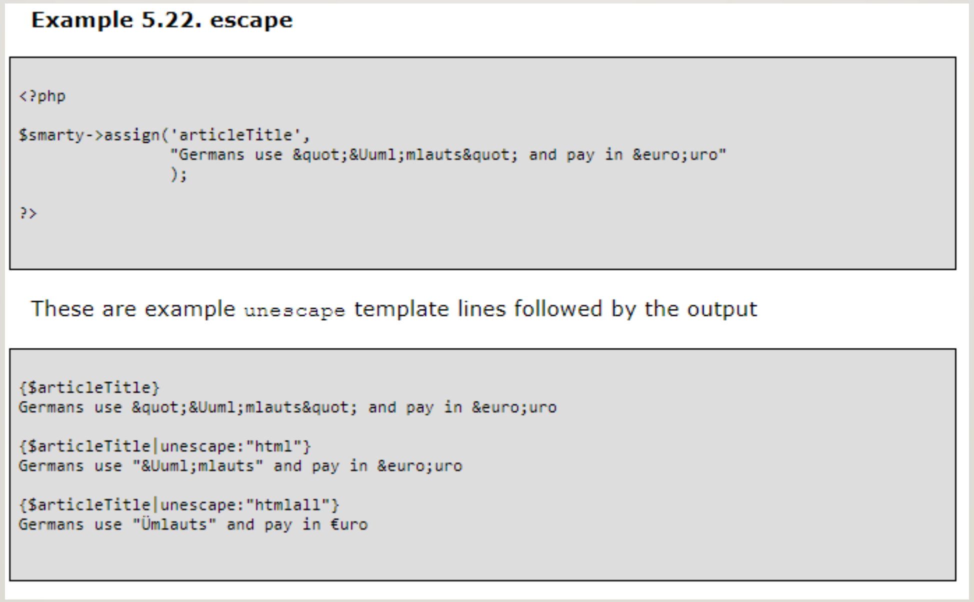
```
<?php

$smarty->assign('articleTitle',
                "'Stiff Opposition Expected to Casketless Funeral Plan'"
                );
$smarty->assign('EmailAddress','smarty@example.com');

?>
```

These are example `escape` template lines followed by the output

```
{$articleTitle}
'Stiff Opposition Expected to Casketless Funeral Plan'

{$articleTitle|escape}
&#039;Stiff Opposition Expected to Casketless Funeral Plan&#039;

{$articleTitle|escape:'html'}    {* escapes  & " ' < > *}
&#039;Stiff Opposition Expected to Casketless Funeral Plan&#039;

{$articleTitle|escape:'htmlall'} {* escapes ALL html entities *}
&#039;Stiff Opposition Expected to Casketless Funeral Plan&#039;

<a href="?title={$articleTitle|escape:'url'}">click here</a>
<a
href="?title=%27Stiff%20Opposition%20Expected%20to%20Casketless%20Funeral%20Plan%27">click here</a>

{$articleTitle|escape:'quotes'}
\'Stiff Opposition Expected to Casketless Funeral Plan\'

<a href="mailto:{$EmailAddress|escape:"hex"}">{$EmailAddress|escape:"hexentity"}</a>
{$EmailAddress|escape:'mail'}    {* this converts to email to text *}
<a href="mailto:%62%6f%..snip..%65%74">&#x62;&#x6f;&#x62..snip..&#x65;&#x74;</a>

{'mail@example.com'|escape:'mail'}
smarty [AT] example [DOT] com
```

WHAT IS UNESCAPED?

- *Unescape* is used to decode *entity, html, and htmlall*. It counters the effects of the escape modifier for the given types. By default, it uses html:



Example 5.22. escape

```
<?php

$smarty->assign('articleTitle',
                "Germans use &quot;&Uuml;mlauts&quot; and pay in &euro;uro"
                );

?>
```

These are example `unescape` template lines followed by the output

```
{$articleTitle}
Germans use &quot;&Uuml;mlauts&quot; and pay in &euro;uro

{$articleTitle|unescape:"html"}
Germans use "&Uuml;mlauts" and pay in &euro;uro

{$articleTitle|unescape:"htmlall"}
Germans use "Ümlauts" and pay in €uro
```

WHAT IS SANDBOXING?

- When PHP is mixed with templates, there are no restrictions on what type of logic can be injected into a template.
- Smarty insulates the templates from PHP, creating a controlled separation of presentation from business logic.
- Smarty also has security features that can further enforce granular restrictions on templates.
- It is a security measure that is used to test unverified programs that may contain a virus or other malicious code, without allowing the software to harm the host device.

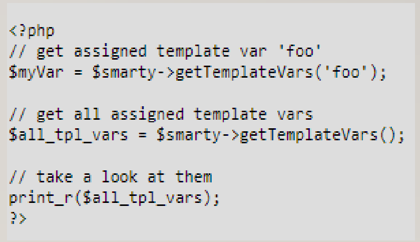
{ASSIGN} VS. ASSIGN()

{ASSIGN} IS A BUILT-IN FUNCTION

- Used for assigning template variables **during the execution of a template**.

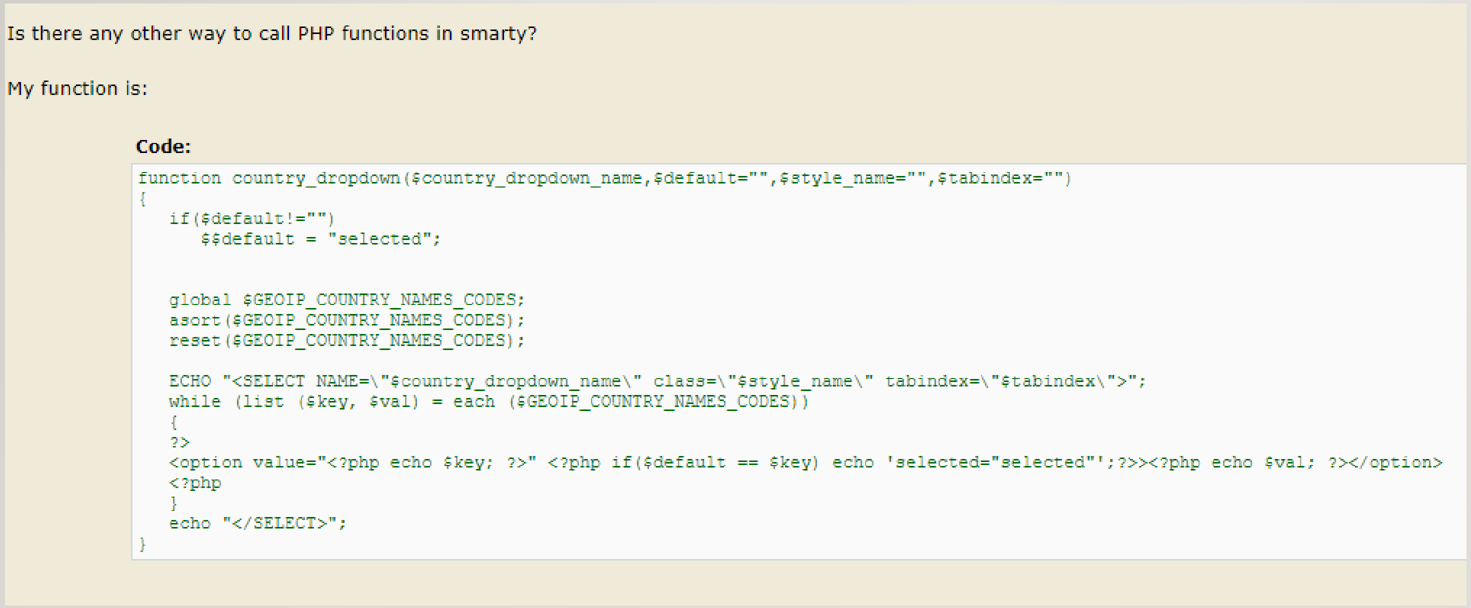
ASSIGN() IS A CLASS METHOD

- Used to assign variables/objects to the templates that were done by the front-end
- We can use getTemplateVars() to configure what type of assigned variable values are returned from an assign() call
- **If no parameter is given, an array of all assigned variables are returned.**
    - **i.e.** array getTemplateVars(string varname);



```php
<?php
// get assigned template var 'foo'
$myVar = $smarty->getTemplateVars('foo');

// get all assigned template vars
$all_tpl_vars = $smarty->getTemplateVars();

// take a look at them
print_r($all_tpl_vars);
?>
```

CAN YOU CALL FUNCTIONS IN CLASS METHODS?

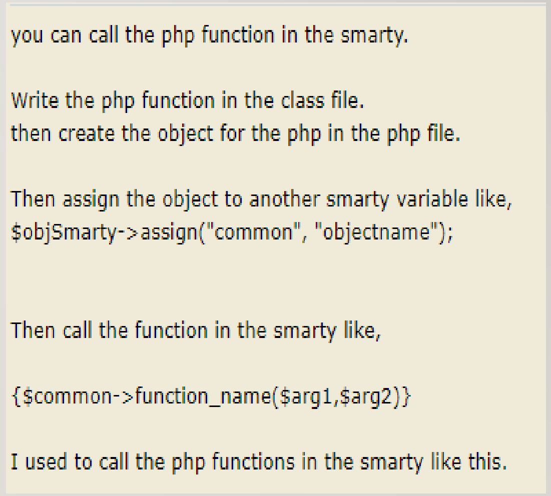


Is there any other way to call PHP functions in smarty?

My function is:

Code:

```
function country_dropdown($country_dropdown_name,$default="",$style_name="",$tabindex="")
{
   if($default!="")
      $$default = "selected";


   global $GEOIP_COUNTRY_NAMES_CODES;
   asort($GEOIP_COUNTRY_NAMES_CODES);
   reset($GEOIP_COUNTRY_NAMES_CODES);

   ECHO "<SELECT NAME=\"$country_dropdown_name\" class=\"$style_name\" tabindex=\"$tabindex\">";
   while (list ($key, $val) = each ($GEOIP_COUNTRY_NAMES_CODES))
   {
   ?>
   <option value="<?php echo $key; ?>" <?php if($default == $key) echo 'selected="selected"';?>><?php echo $val; ?></option>
   <?php
   }
   echo "</SELECT>";
}
```



you can call the php function in the smarty.

Write the php function in the class file.
then create the object for the php in the php file.

Then assign the object to another smarty variable like,
$objSmarty->assign("common", "objectname");

Then call the function in the smarty like,

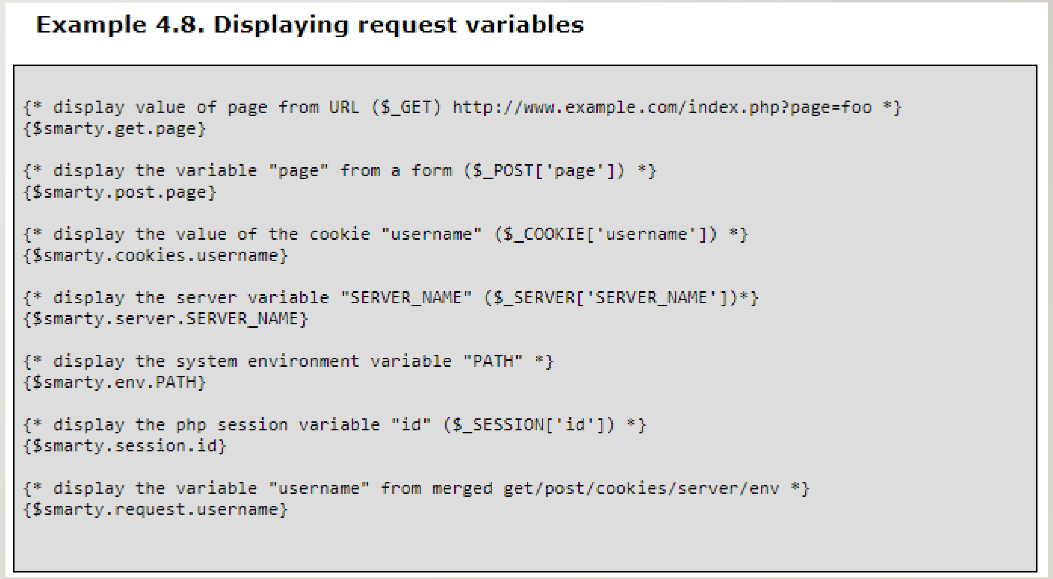
{$common->function_name($arg1,$arg2)}

I used to call the php functions in the smarty like this.

https://stackoverflow.com/questions/33097552/calling-static-methods-in-smarty-using-variable-as-class-name -> This post showed how Smarty allows functions to be used for a static class method.

$SMARTY.CONST

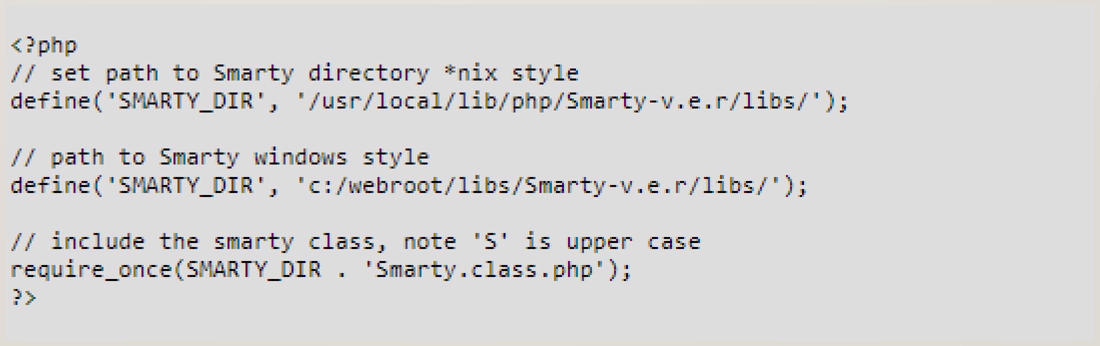
- Although Smarty provides direct access to PHP constants for convenience, it is typically avoided as this is mixing underlying application code structure into the templates.
- **A good practice is to assign specific needed values to template vars.**
  - Smarty suggests that we must provide a specific data type that a class method like assign must operate upon. In that case, there are possibilities that the class methods can access functions and classes from the same and/or different files in a PHP application


Example 4.8. Displaying request variables

```
{* display value of page from URL ($_GET) http://www.example.com/index.php?page=foo *}
{$smarty.get.page}

{* display the variable "page" from a form ($_POST['page']) *}
{$smarty.post.page}

{* display the value of the cookie "username" ($_COOKIE['username']) *}
{$smarty.cookies.username}

{* display the server variable "SERVER_NAME" ($_SERVER['SERVER_NAME'])*}
{$smarty.server.SERVER_NAME}

{* display the system environment variable "PATH" *}
{$smarty.env.PATH}

{* display the php session variable "id" ($_SESSION['id']) *}
{$smarty.session.id}

{* display the variable "username" from merged get/post/cookies/server/env *}
{$smarty.request.username}
```

SMARTY_DIR OPERATION

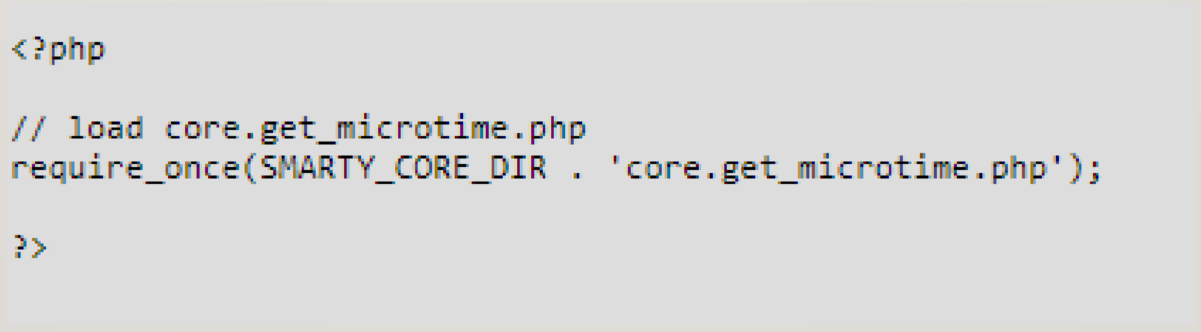
- This is the **full system path** to the location of the Smarty class files. If this is not defined in your script, then Smarty will attempt to determine the appropriate value automatically. If defined, the path **must end with a trailing slash/**.
  - Basically, Smarty will assign a specific location to store all your programs that include key data in the virtual machine that we work with:



```php
<?php
// set path to Smarty directory *nix style
define('SMARTY_DIR', '/usr/local/lib/php/Smarty-v.e.r/libs/');

// path to Smarty windows style
define('SMARTY_DIR', 'c:/webroot/libs/Smarty-v.e.r/libs/');

// include the smarty class, note 'S' is upper case
require_once(SMARTY_DIR . 'Smarty.class.php');
?>
```

SMARTY_CORE_DIR OPERATION

- Like SMARTY_DIR, the purpose of the SMARTY_CORE_DIR is:
  - This is the *full system path* to the location of the Smarty core files. If not defined, Smarty will default this constant to the internals/ sub-directory below SMARTY_DIR. If defined, the path must end with a slash/. Use this constant when manually including any of the core.* files.



```php
<?php

// load core.get_microtime.php
require_once(SMARTY_CORE_DIR . 'core.get_microtime.php');

?>
```