Tableau comparatif des méthodes d'inclusion de scripts PHP dans un article SPIP

Comparaison de méthodes d'exécution de scripts PHP dans des articles sous SPIP. Les 3 méthodes décrites ont été testées avec SPIP 1.9.2d. Ma contribution consiste à mettre en oeuvre le passage des paramètes pour 2 d'entre elles.

---

1. Intro.

Afin d'éviter de paraphraser ce qui a déjà été écrit, j'ai décidé d'essayer de faire un résumé comparatif de ce qui permet d'exécuter des fonctions PHP dans des articles de SPIP.
En effet les forums des contributions sur le sujet permettent de s'en sortir pour pouvoir rendre les articles un peu plus dynamiques.
L'inconvénient est d'avoir le temps de se faire une synthèse car il faut souvent compulser de nombreuses pages avant d'avoir la solution.

J'ai relevé 3 méthodes qui fonctionnent avec la version 1.9.2d de SPIP, apparemment incompatibles avec la version 2. Pourtant j'ose espérer que la 3ème méthode (qui est en fait une fonctionnalité de SPIP) fonctionne toujours.

2. Tableau de comparaison.

Caracté-ristiques	Méthodes		
	1	2	3
2.1. Auteurs	GoUaRfIg ! , Michel Maillard, Jim WANDERSCHEID	Stephane Le Solliec	Créateurs de SPIP (...)
2.2. Liens des contributions utilisées	Sur SPIP Contrib pour l'article de fond ; et sur le site de l'auteur pour le passage des paramètres.	Sur le site de l'auteur	Sur le site de SPIP
2.3. Mes contributions	Implantation du passage de paramètres.	Implantation du passage de paramètres.	Aucune

Caracté-ristiques	Méthodes		
	1	2	3
2.4. Difficultés rencontrées	• Mise au point de **l'expression régulière**. • Prise en compte du caractère '*blanc*' introduit par SPIP version française précédant un point d'interrogation ' ?'. • Conversion de **charset** notamment pour les caractères ' ?' et '&'.	• Mise au point de **l'expression régulière**. • L'indispensabilité de la fonction **my_include**. En effet la fonction '**ob_start()**' ne supporte pas d'être dans une boucle. Je suppose que la fonction **my_include** créée et détruit l'objet '*buffer*' implicitement créé par '**ob_start()**' ce qui n'est pas le cas lorsqu'elle est placée dans la boucle principale du filtre '**recherche_php**'. • L'extrême difficulté de mesurer la longueur de la chaîne URL avec la fonction '**strlen**' notamment à cause de l'encodage des caractères des valeurs données aux paramètres passés (*voir les 'posts' du lien pour s'en convaincre)*. J'ai contourné le problème en récupérant la position ('**strpos**') du caractère '>' de fin d'URL.	• Synthèse difficile de la documentation officielle.
2.5. Notions SPIP sous-jacentes	Filtre appliqué à des balises. En l'occurence la balise #TEXTE.	Filtre appliqué à des balises. En l'occurence la balise #TEXTE.	Modèles avec la balise #ENV pour le passage des paramètres par variable d'environnement.
2.6. Syntaxe d'appel	**#INCLURE(ma_fonction.php)**	**<php_cache_ma_fonction>**	**<ma_fonctionNNN>** **NNN** représente un nombre ici **obligatoire**.
2.7. Syntaxe avec passage de paramètres	**#INCLURE(ma_fonction.php?par1=XXX&par2=YYY)** **par1** et **par2** sont 2 paramètres passés à la fonction. **XXX** et **YYY** sont respectivement les valeurs données à ces paramètres . **?, &** et **=** sont nécessaires à la syntaxe.	**<php_cache_ma_fonction|par1=XXX|par2=YYY>** **par1** et **par2** sont 2 paramètres passés à la fonction. **XXX** et **YYY** sont respectivement les valeurs données à ces paramètres . **|** et **=** sont nécessaires à la syntaxe.	**<ma_fonctionNNN|par1=XXX|par2=YYY>** **NNN** représente un nombre ici **facultatif.** **par1** et **par2** sont 2 paramètres passés à la fonction. **XXX** et **YYY** sont respectivement les valeurs données à ces paramètres . **|** et **=** sont nécessaires à la syntaxe.

<table>
<tr><th>Caracté-ristiques</th><th colspan="3">Méthodes</th></tr>
<tr><th></th><th>1</th><th>2</th><th>3</th></tr>
<tr><td>2.8. Exemples</td>
<td>

L'appel au niveau de la zone de saisie de l'article :

#INCLURE(bonjour.php?Param1=H2Fooko&Param2=Youkulélè)

L'aperçu dans l'espace privé (notez l'espace introduit par SPIP avant le point d'intérrogation ' ?'. Il s'agit en principe d'une particularité de la langue française. Qu'en est-il pour les autres langues avec SPIP ?) une fois enregistrée la saisie :

#INCLURE(bonjour.php ?Param1=H2Fooko&Param2=Youkulélè)

La source PHP de « bonjour.php » (notez que les fonctions de la librairie mbstring ne sont là que pour retranscrire les accents) montre l'utilisation des paramètres Param1 et Param2 passés au sein de la fonction via une URL et utilisés dans le script en faisant précéder ces noms de paramètres par le signe dollar '$' typique des variables PHP
(ici $Param1 et $Param2) :

<pre>
<?php
    $ligne1 = "Bien le Bonjour de ";
    echo mb_convert_encoding($ligne1,
'UTF-8',
mb_detect_encoding($ligne1)).$Param1.
'<br>';
    $ligne2 = "Signé : ";
    echo mb_convert_encoding($ligne2,
</pre>
</td>
<td>

L'appel au niveau de la zone de saisie de l'article :

<php_cache_bonjour|Param1=H2Fooko|Param2=Youkulélè>

L'aperçu dans l'espace privé est inexistant.
La source PHP de « bonjour.php » (identique à la méthode précédente, localisé dans le même répertoire) montre l'utilisation des paramètres Param1 et Param2 passés au sein de la fonction comme pour un modèle SPIP (voir méthode suivante) et utilisés dans le script en faisant précéder ces noms de paramètres par le signe dollar '$' typique des variables PHP (ici $Param1 et $Param2) :

<pre>
<?php
    $ligne1 = "Bien le Bonjour de ";
    echo mb_convert_encoding($ligne1,
'UTF-8',
mb_detect_encoding($ligne1)).$Param1.'<
br>';
    $ligne2 = "Signé : ";
    echo mb_convert_encoding($ligne2,
'UTF-8',
mb_detect_encoding($ligne2)).$Param2;
?>
</pre>

Enfin le rendu :

Bien le Bonjour de H2Fooko
Signé : Youkulélè
</td>
<td>

L'appel au niveau de la zone de saisie de l'article :

<bonjour|Param1=H2Fooko|Param2=Youkulélè>

L'aperçu dans l'espace privé montre déjà le résultat tel qu'il sera dans l'espace public :

Bien le Bonjour de H2Fooko
Signé : Youkulélè

La source de « bonjour.html » (notez que les fonctions de la librairie mbstring ne sont plus nécessaires) montre l'utilisation des paramètres Param1 et Param2 passés au sein de la fonction en séparant les couples (variable = valeur) par des barres verticales de séparation | et utilisés dans le script en enveloppant ces noms de paramètres par une balise SPIP "#ENV{ }"
(ici "#ENV{Param1}" et "#ENV{Param2}") :

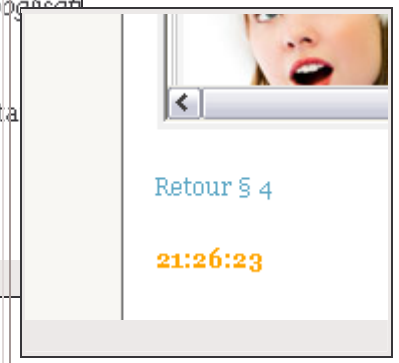
<pre>
<?php
    echo "Bien le Bonjour de
"."#ENV{Param1}".'
';
    echo "Signé :
"."#ENV{Param2}".'
';
?>
</pre>

Enfin le rendu :

Bien le Bonjour de H2Fooko
</td></tr>
</table>

Caracté-ristiques	Méthodes		
	1	2	3
	`'UTF-8', mb_detect_encoding($ligne2)).$Param2; ?>` • Enfin le rendu : Bien le Bonjour de H2Fooko Signé : Youkulélè		Signé : Youkulélè
2.9. Localisation des scripts	Précisé par la variable **$dossier_inclus** du filtre **autoriser_php**, il suffit d'indiquer le nom et le chemin du répertoire que vous aurez créé préalablement. En ce qui me concerne : **$dossier_inclus** = './my_script_php/' ; Il suffit d'uploader ensuite le script PHP à cet endroi.	Précisé par la variable **$ScriptDir** du filtre **recherche_php**, il suffit d'indiquer le nom et le chemin du répertoire que vous aurez créé préalablement. En ce qui me concerne : **$ScriptDir** = './my_script_php/' ; J'ai délibérémment pris le même répertoire que pour la première méthode afin d'utiliser indifféremment la première ou la seconde méthode.	Le répertoire **/modeles/** doit être créé à l'intérieur du répertoire **./squelettes/** s'il n'existe pas déjà. (*Il faudrait étudier les variables intrinsèques de SPIP pour éventuellement changer la localisation des scripts*). Ces scripts PHP sont '*enveloppés*' dans un fichier HTML.
2.10. Extension	***.php**	***.php**	***.html** Le contenu peut être du pur PHP à l'exception des variables passées en paramètre.

Caracté-ristiques	Méthodes		
	1	2	3
2.11. Fichiers de la distribution à modifier	• Le fichier **mes_fonctions.php** du répertoire **./squelettes/** est à modifier (ou à créeer de toutes pièces s'il n'existe pas). On y rajoute la fonction du filtre **autoriser_php**. • Le fichier du squelette **article.html** est à copier de la distribution dans le répertoire **./squelettes/** s'il n'existe pas déjà avant de le modifier pour y appliquer le nouveau filtre à la balise **#TEXTE**.	• Le fichier **mes_fonctions.php** du répertoire **./squelettes/** est à modifier (ou à créeer de toutes pièces s'il n'existe pas). On y rajoute la fonction du filtre **recherche_php** ainsi que la fonction **my_include**. • Le fichier du squelette **article.html** est à copier de la distribution dans le répertoire **./squelettes/** s'il n'existe pas déjà avant de le modifier pour y appliquer le nouveau filtre à la balise **#TEXTE**.	Pas de modification de fichier de la distribution de SPIP. Seul le script PHP est à déposer dans **./squelette/modeles/** en renommant l'extension du fichier de script PHP en HTML et en enveloppant les variables passées en paramètre par "**#ENV{ }**".

2.12. Comportement dans la zone privée de SPIP	L'image ci-dessous correspond à une copie de l'espace privé de cet article.
	Comme on peut le constater sur la copie d'écran ci-dessous on ne voit que l'appel dans l'espace privé. Et dans l'espace public l'heure est bien rendue.
	
	Ci dessous encore un test extra simple donnant l'
	penser que je vais y arriver :
	
	|#INCLURE (heure.php)|
	
	Et maintenant mon dernier test avec les fonctions
	
	Un examen de la source HTML de l'espace privé donne :
	
	`<p class="spip">Ci dessous encore un test extra simple donnant l’heure qui m’incite à penser que je vais y arriver :</p>`
	`<table class="spip">`
	`<tbody>`
		L'image ci-dessous correspond à une copie de l'espace privé de cet article.
		Comme on peut le constater sur la copie d'écran ci-dessous on ne voit rien dans l'espace privé. Par contre dans l'espace public les graphes sont bien rendus.
		
		Et finalement avec un peu de perspicacité et de *googlisation*,
		l'exemple N°0 inséré dans cette page :
		
		Reste plus qu'à vous expliquer comment et vu l'heure ta
		ce soir.
		
		Enfin l'exemple N°33
		
		Un examen de la source HTML de l'espace privé donne :
		
		`Et finalement avec un peu de perspicacité et de <i class="spip">googlisation,</i> voici ci-dessous l’exemple N°0 inséré dans cette page :</p>`
			L'image ci-dessous correspond à une copie de l'espace privé de cet article.
			Comme on peut le constater sur la copie d'écran ci-dessous on aperçoit bien l'heure dans l'espace privé de même que dans l'espace public. Seul diffère la police de caractères.
			
			 
			
			Un examen de la source HTML de l'espace privé donne :
			
			`<p class="spip"><a name="heuremodel"></a><span style="color:orange;font-`

Caracté-ristiques	Méthodes		
	1	2	3
	<tr class="row_even"><td>#INCLURE (heure.php)</td></tr> </tbody> </table>	<php_cache_example0> Reste plus qu’à vous expliquer comment et vu l’heure tardive ça ne sera pas pour ce soir. <php_cache_example33>	weight:bold">21:30:23 Notez qu'il s'est écoulé 4mn entre la copie d'écran et la génération de la source HTML de la page privée !
2.13. Mes suppositions.	Le squelette de la page privée doit être différent de celui de la page publique. Il n'a probablement pas reconnu le filtre appliqué : « **autoriser_php** » puisque l'on retrouve en clair « **#INCLURE(heure.php)** » dans cette page privée.	Le squelette de la page privée doit - là aussi - être différent de celui de la page publique. Il n'a probablement pas reconnu le filtre appliqué : « **recherche_php** » puisque l'on retrouve en clair « **<php_cache_example0>** » et un peu plus loin « **<php_cache_example33>** » dans cette page privée. De plus lors du rendu de cette page privée, du fait de la présence des symboles : « **<** » et « **>** » ouvrant et fermant normalement une balise, le contenu situé entre ces symboles n'a pas été reconnu comme une balise standard HTML. Si bien qu'il a été ignoré et rien n'est affiché.	Un modèle SPIP est par définition un mini squelette, il a donc été inclu dans le squelette de la page privée (pas besoin de reconnaissance), il a donc rendu tout naturellement un résultat au script PHP contenu dans ce modèle.

Caracté-ristiques	Méthodes		
	1	2	3
2.14. Sécurité	Voir le paragraphe consacré sur le site du contributeur.	Voir le paragraphe consacré sur le site du contributeur.	**ATTENTION** contrairement aux 2 autres méthodes il faut s'assurer que votre répertoire **/modeles/** soit protégé par un fichier **.htaccess**. Ce contenu suffit : `deny from all` quant au fichier .htaccess je l'ai mis dans le répertoire /modeles/ En effet si on essaye d'accéder au script directement en pointant son adresse http avec votre navigateur on peut récupérer la source de ce script dans la source de la page html rendue ! Du style : http:\\h2fooko.free.fr/squelettes/modeles/heure.html Très génant si votre script contient un mot de passe pour se connecter à votre base de données par exemple ! Le fait que le script soit enveloppé dans un fichier html, si ce script génère une erreur (il attend par exemple des paramètres non passés) la page pointée est quand même rendue avec du PHP en clair dans la source ! Avec les 2 premières méthodes, je n'ai pas observé ce problème.
2.15. Compatibilité SPIP	1.9.2d	1.9.2d	1.9.2d

Caracté-ristiques	Méthodes		
	1	2	3
2.16. Exemples sur ce site	• Génération d'un SiteMap Google	• Inclusion de fichier HTML sans *iframe* dans un article SPIP	• Affichage de courbes JpGraph

3. Modifications du code.

3.1. « mes_fonctions.php »

Mes contributions sont mises en évidence sur fond jaune dans les sources ci-dessous. Les sources des filtres « **autoriser_php** » et « **recherche_php** » et la fonction « **my_include** » sont à inclure dans le fichier « **mes_fonctions.php** » du répertoire squelette. Ce fichier est à créér ex-nihilo s'il n'existe pas déjà.

J'ai aussi mis des liens sur les fonctions php utilisées pointant vers une documentation en ligne. Celle ci s'ouvre dans le cadre (iframe), un simple clic droit permettrait de l'ouvrir dans une nouvelle fénêtre. Dans le cas d'une visualisation dans le cadre il suffit de cliquer sur le bouton « **précédente** » de votre navigateur pour revenir au code source.

« Filtre autoriser_php »

J'ai pu mettre au point l'expression régulière de recherche de l'URL de passage des paramètres tout en tenant compte de l'espace introduit par SPIP avant chaque point d'interrogation.
Je suis impressionné par la puissance des expressions régulières qui - bien employées - permettent de récupérer dans le tableau "résultat" ce que l'on souhaite.
Le test permet de détecter l'existence d'une URL et dans l'affirmative de préparer l'isolement de la chaine URL traitée ensuite avec la fonction "parse_str" localisée plus loin.
Le reste est décrit sur le site des contributions des auteurs.

```php
<?php
function autoriser_php( $texte) {

// Dossier où sont stockés les scripts
//(à partir du fichier spip.php et non à partir du répertoire squelettes)
    $dossier_inclus = './my_script_php/';

// Les fichiers inclus autorises sont :
// *.php, *.php3, *.php4 , *.php5, *.phtml, *.htm, *.html, *.inc
    $fichiers_autorises="(php|php3|php4|php5|phtml|htm|html|inc)";

// Recherche des fichiers inclus autorisés dans le dossier de stockage
    $dir=opendir("$dossier_inclus");
    while ($fichier=readdir ($dir))
    {
        if(( preg_match( $fichiers_autorises, $fichier)))
        {
            $scripts_autorises[]= $fichier;
        }
    }
```

```
    closedir($dir);
// Expression régulière de recherche de #INCLURE(script.php?var1=XXX&var2=YYY) avec une URL.

    $chaine_url = "(( \?(\w)+=(.)+){1}(&amp;(\w)+=(.)+)*)*";
    $chaine_recherche = "/#INCLURE( *)\(( *)([^\)]*\.".$fichiers_autorises.$chaine_url.")( *)\)/i";
    while( preg_match( $chaine_recherche, $texte, $resultats)) {
        $autorise = false;
        reset( $scripts_autorises);
        $nom_length = strlen($resultats[3]);
        // SPIP nous a introduit un caractère blanc ' ' avant le point d'interrogation '?' pour la version française.
        $pos_question = strpos($resultats[3],' ?');
        // Si la chaine recherchée n'existe pas on souhaite prendre la longueur max.
        // dans le cas où il n'y a pas d'URL.
        if ($pos_question === false)
        {
            $pos_question = $nom_length;
            $nom_url = "";
        } else {
            // le chiffre 7 correspond à la longueur de ' ?'. On récupère l'URL après le ?.
            $nom_url = substr($resultats[3],$pos_question+7,$nom_length);
            //SPIP convertit le caractère '&' en '&amp;' on refeait l'opération inverse !
            $nom_url = str_replace ("&amp;","&",$nom_url);
        };
        $nom_script = substr($resultats[3],0,$pos_question);
        // Verifie le droit d'inclure ce fichier script !
        while( $script = each( $scripts_autorises)) {
            if( strcasecmp( $script[value], $nom_script) == 0) $autorise = true;
        }
        if( $autorise == true) {
            // Verifie que le fichier existe
            $f=$dossier_inclus . $nom_script;
            if( file_exists ($f)) {
                // Vide le buffer de sortie
                $affichage_php = '';
                // On créée les variables on on les initialise grâce à l'URL.
                parse_str($nom_url);
                // Et lance le fichier inclu !!!
                ob_start();
                include ($f);
            } else {
                $affichage_php = "<b>#INCLURE: Le fichier de script $f n'existe pas !</b>";
            }
        } else {
            $affichage_php = "<b>#INCLURE: Script NON autoris&eacute; !</b>";
        }
```

```
        // Attention, n'effectue qu'un seul remplacement à la fois !
        $affichage_php = ob_get_contents();
        ob_end_clean();
        $texte = preg_replace( $chaine_recherche, $affichage_php, $texte, 1);
    }
    return( $texte);
}
?>
```

Fonction « my_include »

J'ai seulement rajouté à la fonction "my_include" le traitement d'une pseudo URL passée par variable et encadrée par les caractères "|" et ">".
Le test sur le premier caractère me permet de détecter ou non l'existence de cette pseudo URL. Ca peut paraître un peu léger, mais en fait si on arrive jusque là sans erreurs c'est que l'expression régulière a repéré une chaîne en principe conforme : un autre test est pour moi superflu.
Les 2 caractères extrêmes me permettent d'isoler le contenu de l'URL (Voir les difficultés rencontrées). Et pour être conforme à ce qu'attend la fonction "parse_str", je remplace tous les "|" par des "&".
Le reste des explications se trouve sur le site de l'auteur.

```
<?php
function my_include($fich, $url){
    if ($url[0] === '|'){
        $nom_url = substr($url,1,strpos($url,'>') - 1);
        // Pour être conforme à la syntaxe acceptée par la fonction 'parse_str'.
        $nom_url = str_replace ("|","&",$nom_url);
        parse_str ($nom_url);
    };
    // Débranche le retour vers le navigateur et renvoie dans un buffer
    ob_start();
    include($fich.'.php');
    // on recupère le buffer
    $retour = ob_get_contents();
    // on vide et ferme le buffer
    ob_end_clean();
    return $retour;
}

?>
```

Filtre « recherche_php »

Ici encore j'ai bénéficié du testeur d'expression régulière pour apprendre à isoler ma pseudo URL. J'ai aussi exploité au mieux l'utilisation du tableau "mymatches" :

- La séquence recherchée en entier (indice[0])
- Le nom du script PHP (indice[2])
- La pseudo-URL (indice[3]).

```php
<?php
function recherche_php($t) {
    $ScriptDir = './my_script_php/';
    $SearchedStrUrl = "/<php_cache_(([^>]+?)([|]{1}((\w)+=(.)+){1}([|]*(\w)+=(.)+)*)*)>/i";
    // on remplace les <php_cache_XX>
    if (preg_match_all($SearchedStrUrl,$t,$mymatches)) {
        for ($i=0;$i< sizeof ($mymatches[2]);$i++) {
            $tag = my_include($ScriptDir.$mymatches[2][$i], $mymatches[3][$i]);
            $t = str_replace($mymatches[0][$i],$tag,$t);
        }
    }
    return $t;
}
?>
```

3.2. Squelette « article.html »

Ci-dessous sur fond jaune la façon d'appliquer les 2 filtres "autoriser_php" et "recherche_php" à la balise #TEXTE dans le squelette "article.html".

```
#CACHE{86400} []
[(#REM) Entete de la page + titre du site ] [(#REM) Fil d'Ariane ]
<:accueil_site:> > [(#TITRE|couper{80})][ > (#TITRE|couper{80})]
[(#REM) Contenu principal : contenu de l'article ]
#DEBUT_SURLIGNE [(#LOGO_ARTICLE||image_reduire{200,200})] [

(#SURTITRE)

]
#TITRE
[
```

```
(#SOUSTITRE)

] #FIN_SURLIGNE

[(#DATE|nom_jour) ][(#DATE|affdate)][, <:par_auteur:> (#LESAUTEURS)]

[(#REM) Inclure le modele des liens de traductions ] #MODELE{article_traductions}
#DEBUT_SURLIGNE [
(#CHAPO)
]
[

<:voir_en_ligne:> : [(#NOM_SITE|sinon{[(#URL_SITE|couper{80})]})]

] [
(#TEXTE|image_reduire{520,0}|recherche_php|autoriser_php)
] #FIN_SURLIGNE [(#REM) Portfolio : album d'images ]
<:info_portfolio:>
[(#FICHIER|copie_locale|image_reduire{0,60}|inserer_attribut{alt,[(#TITRE|couper{80}|texte_backend)]})]
#EMBED_DOCUMENT [
(#TITRE)
] [
(#DESCRIPTIF)
]
[ <:info_ps:>
#DEBUT_SURLIGNE
(#PS)
#FIN_SURLIGNE
] [(#REM) Autres documents joints a l'article ]
<:titre_documents_joints:>

    [(#TITRE|sinon{<:info_document:>})] (#TYPE_DOCUMENT[ - (#TAILLE|taille_en_octets)])

        [
```

(#DESCRIPTIF)

]

[(#REM) Petition : La petition ayant une PAGINATION il faut absolument lui passer SELF] [(#PETITION|?{' '})] [

<:info_notes:>

#DEBUT_SURLIGNE(#NOTES)#FIN_SURLIGNE

] [(#REM) Forum de l'article] D:\Documents and Settings\FR028328\Mes Documents\Perso\SVG SPIP\docu\articlehtml.html - forum#forum[

<:repondre_article:>

]

[(#REM) Menu de navigation laterale]

[(#REM) Menu de navigation par rubriques] [(#REM) Articles dans la meme rubrique]

<:meme_rubrique:>

#TITRE

[(#REM) Menu de navigation mots-cles] #MODELE{article_mots}

[(#REM) Pied de page]