

Helpers

<

[Bibliothèques >>](#)

- [1. Introduction à CodeIgniter](#)
- [2. Helpers](#)
- [3. Bibliothèques](#)
- [4. Sessions CodeIgniter](#)
- [5. Validation des formulaires](#)
- [6. ORM IgnitedRecord](#)
- [7. Doctrine](#)
- [8. Javascript et codeigniter](#)

helper url

- [1. Aide url helper](#)
- [2. URLs codeigniter](#)

Le helper **url** permet de gérer plus facilement les urls codeigniter :

Fonction	Paramètres	Rôle
site_url	[array or String]	Retourne une url valide, exemples : site_url();→ http://localhost/testPhp/ site_url('test');→ http://localhost/testPhp/test/ site_url(array('test','accueil'));→ http://localhost/testPhp/test/accueil/
current_url		Retourne l'url de la page en cours
base_url		Retourne l'url de base de l'application web
redirect	String or Array	Fait une redirection vers une url valide, exemples : redirect('test');→ http://localhost/testPhp/test/ redirect(array('test','accueil'));→ http://localhost/testPhp/test/accueil/

Création d'un Helper personnalisé

Nous allons créer un helper permettant d'accéder plus facilement aux ressources du dossier **assets**, qui contient les css, les images, les fichiers javascript...

[\[h application/helpers/assets_helper.php](#)

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

if (!defined('ASSETS_HELPER')){
    define('ASSETS_HELPER',true);
    function css_url($nom)
    {
        return base_url() . 'assets/css/' . $nom . '.css';
    }
    function js_url($nom)
    {
        return base_url() . 'assets/javascript/' . $nom . '.js';
    }
    function img_url($nom)
```

```
{
    return base_url() . 'assets/images/' . $nom;
}
function img($nom, $alt = '')
{
    return '';
}
?>
```

Il suffit ensuite d'activer le helper dans autoload.php :

```
$autoload['helper'] = array('url','assets');
```

Puis de l'utiliser :

```
echo(css_url('style'));
```

Le fichier helper doit être enregistré dans le dossier **helpers** de l'application, et se terminer par **_helper**.

Surdéfinition d'un Helper codeIgniter

Vérifier la valeur de subclass_prefix dans le fichier config/config.php :

```
$config['subclass_prefix'] = 'MY_';
```

La surdéfinition d'un helper existant permet de modifier le comportement des fonctions du helper prédéfinies.

Exemple

Surdéfinition du helper **url** et modification de la fonction **site_url** :

Le fichier helper surdéfini doit être enregistré dans le dossier **helpers** de l'application, et son nom doit commencer par le **subclass_prefix** devant le nom du helper à modifier.

La surdéfinition du helper url sera donc enregistrée sous **application/helper/MY_url_helper.php**

Surdéfinition de la fonction site_url :

Cette fonction prend en paramètre soit un String, soit un tableau :

Exemples :

- site_url("test");→<http://localhost/testPhp/test>
- site_url(array("test","page"));→<http://localhost/testPhp/test/page/>

Nous voulons qu'elle puisse prendre plusieurs arguments de type String, pour obtenir le résultat suivant :

- site_url("test","page");→<http://localhost/testPhp/test/page/>

|h MY_url_helper.php

```
function site_url($uri = '')
```

```
{
    if( ! is_array($uri))
    {
        //paramètres de la fonction mis dans un tableau
        $uri = func_get_args();
    }

    // fonction d'origine
    $CI =& get_instance();
    return $CI->config->site_url($uri);
}
```

<
Bibliothèques >>

From:
<http://slamwiki2.kobject.net/> - **Broken SlamWiki 2.0**

Permanent link:
<http://slamwiki2.kobject.net/slam4/php/codeigniter/helpers?rev=1355792244>

Last update: **2019/08/31 14:41**

