

# Sessions CodeIgniter

&lt;

[Validation des formulaires >>](#)

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

- [• Aide codeigniter sur les sessions](#)

Les sessions codeigniter fonctionnent de la même façon que celles de PHP, excepté qu'elles ne nécessitent pas d'accéder à la configuration du serveur web pour être elles-mêmes configurées. CodeIgniter utilise son propre système de gestion des sessions, soit via les cookies, soit via la base de données.

La bibliothèque **session** doit être chargée :

- soit automatiquement avec autoload.php :

```
$autoload['libraries'] = array('session');
```

- Soit dans le code d'un contrôleur par exemple :

```
$this->load->library('session');
```

## Identifiant de session

```
$session_id = $this->session->userdata('session_id');
```

CodeIgniter sauvegarde en session des informations complémentaires sur le visiteur :

```
<?php
$adresse_ip           = $this->session->userdata('ip_address');
$user_agent_navigateur = $this->session->userdata('user_agent');
$derniere_visite      = $this->session->userdata('last_activity');
?>
```

## Ajout et récupération de variables

Pour une variable :

```
$this->set_userdata('nom', 'value');
```

**Exemple :**

```
<?php
```

```
$this->session->set_userdata('utilisateur', $user);

// .....

$aUser= $this->session->userdata('utilisateur');
echo(aUser->getNom());
?>
```

Pour plusieurs variables :

```
$newdata = array(
    'username' => 'johndoe',
    'email'    => 'johndoe@some-site.com',
    'logged_in' => TRUE
);

$this->session->set_userdata("user",$newdata);
```

## Suppression de variables

Une variable :

```
$this->session->unset_userdata('nom');
```

Plusieurs variables :

```
$array_items = array('username' => '', 'email' => '');
$this->session->unset_userdata($array_items);
```

## Suppression de la session

```
$this->session->sess_destroy();
```

## FlashSession

CodeIgniter offre la possibilité d'utiliser des variables de session mémorisées uniquement d'une requête sur l'autre puis ensuite automatiquement détruites.

```
$this->session->set_flashdata('item', 'value');
```

## Sessions et objets métier

Pour assurer la persistance des objet en session, il est parfois préférable de sur-définir la méthode **\_sleep** de chacun d'entre eux, de façon à définir les membres à sérialiser :

```
/**
```

```
* Utilisateur
*/
class Utilisateur
{
    ...
    public function __sleep(){
        return array('id','login','password','nom','prenom','mail');
    }
}
```



La limite de taille de la session codelgniter étant de **4ko**, il peut être nécessaire pour dépasser cette limite d'utiliser la base de données pour les sessions.

## Utilisation de la base de données

Dans le fichier config.php, modifier la ligne suivante :

```
$config['sess_cookie_name']      = 'ci_session';
$config['sess_expiration']       = 7200;
$config['sess_expire_on_close']  = FALSE;
$config['sess_encrypt_cookie']   = FALSE;
$config['sess_use_database']     = TRUE;
$config['sess_table_name']       = 'ci_sessions';
$config['sess_match_ip']        = FALSE;
$config['sess_match_useragent']  = TRUE;
$config['sess_time_to_update']   = 300;
```

Créer la table nécessaire dans votre BDD sur ce modèle :

```
CREATE TABLE IF NOT EXISTS `ci_sessions` (
  session_id varchar(40) DEFAULT '0' NOT NULL,
  ip_address varchar(45) DEFAULT '0' NOT NULL,
  user_agent varchar(120) NOT NULL,
  last_activity int(10) unsigned DEFAULT 0 NOT NULL,
  user_data text NOT NULL,
  PRIMARY KEY (session_id),
  KEY `last_activity_idx` (`last_activity`)
);
```

<

[Validation des formulaires >>](#)

## Utilisation des sessions natives PHP

Librairie phpSession : [phpsession.php](http://phpsession.php)

## Démarrage

Soit dans autoload.php :

```
$autoload['libraries'] = array('phpsession');
```

Soit dans un contrôleur :

```
class UnControleur extends \CI_Controller{  
function __construct()  
{  
    parent::__construct();  
    $this->load->library('phpsession');  
}  
}
```



Attention, en cas de sérialisation/désérialisation d'objets en session, pour éviter le message d'erreur **\_php\_incomplete\_class object** il est nécessaire de charger les modèles concernés avant le démarrage de la session, la session ne doit dans ce cas démarrer ou reprendre que dans un contrôleur, et non dans l'autoload !

From:

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

Permanent link:

<http://slamwiki2.kobject.net/slam4/php/codeigniter/session?rev=1418893152>

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

