UVS
UNIVERSITE VIRTUELLE DU SENEGAL

# Méthodes d'identification d'un utilisateur

## Programmation Web dynamique

El hadji Mamadou NGUER  Enseignant chercheur en Informatique  à l'UVS

# Chapitre 1 : Méthodes d'identification d'un utilisateur

- **Objectifs spécifiques :** A la suite de ce chapitre, l'étudiant doit être capable de :

1. Décrire les méthodes d'identification d'un utilisateur en PHP
2. D'identifier un utilisateur en utilisant les Cookies
3. D'identifier un utilisateur en utilisant les Sessions

# Chapitre 1 : Méthodes d'identification d'un utilisateur

**Plan de la séquence** :

1. Introduction
2. Les cookies
3. Les sessions
4. Conclusion

# Introduction

- Il existe en PHP deux moyens pour identifier un utilisateur : les Cookies et les Sessions

- Les cookies identifient l'utilisateur en stockant ses informations dans son ordinateur (client).

- Une session par contre identifie un utilisateur en stockant ses informations (sous forme de variables) au niveau du serveur.

- L'inconvénient avec les Cookies est que l'utilisateur doit d'abord les accepter.

# Les cookies

## Qu'est-ce qu'un cookie?

- Un cookie est un petit fichier que le serveur stocke sur l'ordinateur client.
- Le cookie est alors envoyé à chaque fois que le même client demande une page via un navigateur.
- Avec PHP, on peut à la fois créer et récupérer des valeurs de cookie.

## Créer un cookies avec PHP

Un cookie est créé avec la fonction **setcookie ()** .

**Syntaxe**

**setcookie(*name, value, expire, path, domain, secure, httponly* );**

Seul le paramètre *nom* est requis. Les autres paramètres sont facultatifs.

- **name** est le nom du cookie.

Pour les autres paramètres voir

http://php.net/manual/fr/function.setcookie.php

# Les cookies

## Récupérer un cookie

On récupère la valeur d'un cookie en utilisant la variable globale **$_COOKIE**.
La fonction **isset ()** peut être utilisée pour savoir si le cookie est défini:

## Exemple :

```
<?php
    $nom_cookie = "user";
    $valeur_cookie = "alaxam";
    setcookie($nom_cookie, $valeur_cookie, time() + (86400 * 30), "/"); // 86400 = 1 jour
?>
    <html>
    <body>

<?php
    if(!isset($_COOKIE[$nom_cookie])) {
       echo "Le cookie nommé '" . $nom_cookie . "' n'est pas défini!";
    } else {
       echo "Le cookie '" . $nom_cookie . "' est défini!<br>";
       echo "Sa valeur est: " . $_COOKIE[$nom_cookie];
    }
?>
```

# Les cookies

## Récupérer un cookie

**Remarques:**

- La fonction setcookie () doit apparaître AVANT la balise <html>.
- La valeur du cookie est automatiquement codée par URL lors de l'envoi du cookie et automatiquement décodée lors de la réception (pour éviter l'encodage d'URL, utilisez plutôt setrawcookie ()).

## Modifier la valeur d'un cookie

- Pour modifier un cookie, il suffit de redéfinir le cookie à l'aide de la fonction setcookie ():

## Supprimer un cookie

- Pour supprimer un cookie, on le redéfinie avec la fonction setcookie () en utilisant une date d'expiration antérieur.

**Exemple:**
```
setcookie("user", "", time() - 3600);
```

## Vérifiez si les cookies sont activés

- Pour vérifier si les cookies sont activés, on fait un test sur la taille des cookies :

```
if(count($_COOKIE) > 0)
```

# Les sessions en PHP

**Définition:**

- Une session est un moyen de stocker des informations (sous forme de variables) à utiliser sur plusieurs pages.
- Contrairement à un cookie, les informations ne sont pas stockées sur l'ordinateur de l'utilisateur mais sur le serveur.

**Les variables de session PHP**

- Lorsque vous travaillez avec une application, vous l'ouvrez, faire quelques modifications, puis vous la fermez. Cela ressemble beaucoup à une session. L'ordinateur sait qui vous êtes. Il sait quand vous démarrez l'application et lorsque vous terminez.
- Mais sur internet il y'a un problème: le serveur web ne sait pas qui vous êtes et ce que vous faites parce que l'adresse HTTP ne conservent pas l'état.

# Les sessions en PHP

## Les variables de session PHP

- Une session PHP résout ce problème en vous permettant de stocker des informations utilisateur sur le serveur pour une utilisation ultérieure (c.-à-d. identifiant, articles commerciaux...).

- Toutefois, les informations de session sont temporaires et seront supprimées après que l'utilisateur ait quitté le site. Si vous avez besoin d'un stockage permanent, vous pouvez utiliser une base de données.

- Une session crée un identifiant unique (UID) pour chaque visiteur et les variables stockées sont basées sur cette UID. L'UID est soit stocké dans un cookie, soit propagé dans l'URL.

- Une fois connecté, l'IUD est chaque fois envoyé au serveur quand on clique sur un lien.

# Les sessions en PHP

## Démarrer une session PHP

- Une session est lancée avec la fonction **session_start ()**.
- Elle est définie avec la variable globale **$ _SESSION**.

Exemple: (demo_session1.php)

```php
<?php session_start(); // Démarrer la session
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Définir des variables de session
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Les variables sont définies.";
?>
</body>
</html>
```

**Remarque:** La fonction session_start () doit être placé dans le document avant toute balise HTML.

# Les sessions en PHP

## Accès aux variables de session

- On crée une autre page appelée "demo_session2.php" à partir de laquelle on accède aux informations de session que nous avons définies sur la première page ("demo_session1.php").

- Notez que les variables de session ne sont pas transmises individuellement à chaque nouvelle page, mais qu'elles sont extraites de la session que nous ouvrons au début de chaque page (session_start ()).

- Notez également que toutes les valeurs des variables de session sont stockées dans la variable globale $ _SESSION :

# Les sessions en PHP

## Accès aux variables de session

**Exemple: (**demo_session2.php)

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Affichage des valeurs des variables de session
echo "Ma couleur fav est
" . $_SESSION["favcolor"] . ".<br>";
echo "Mon animal fav est
" . $_SESSION["favanimal"] . ".";
?>

</body>
</html>
```

# Les sessions en PHP

## Accès aux variables de session

On peut aussi utiliser la fonction print_f pour afficher toutes les valeurs de la variable de session d'une session utilisateur :

**Exemple**

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>
</body>
</html>
```

# Les sessions en PHP

**Comment ça marche? Comment sait il que c'est moi?**

- La plupart des sessions définissent une clé utilisateur sur l'ordinateur de l'utilisateur qui ressemble à ceci: 765487cf34ert8dede5a562e4f3a7e12.

- Ensuite, lorsqu'une session est ouverte sur une autre page, elle scanne l'ordinateur pour avoir une clé d'utilisateur.

- S'il y’a une correspondance, il accède à cette session, sinon, il démarre une nouvelle session.

# Les sessions en PHP

## Modifier une variable de session PHP

Pour modifier une variable de session, il suffit de remplacer sa valeur:

## Exemple

```php
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Changer la valeur d'une variable de session
$_SESSION["favcolor"] = "yellow";
print_r($_SESSION);
?>

</body>
</html>
```

# Les sessions en PHP

## Détruire une session PHP

- **session_unset ()** supprime toutes les variables de session
- **session_destroy ()** détruit la session
- **unset($_SESSION['var'])** détruit la variable $var de la session en cours

## Exemple

```
<?php
session_start();
?>
...
<?php
// Supprime toutes les variables de session
session_unset();

// Détruit la session
session_destroy();
?>
...
```

# Conclusion

- Au cours de cette séquence, nous avons :
  - décrit les méthodes d'identification d'un utilisateur en PHP
  - comment identifier un utilisateur en utilisant les Cookies
  - et comment identifier un utilisateur en utilisant les Sessions

- Pour consolider les connaissances acquises dans cette séquence, veuillez effectuer :
  - les tests de connaissance de la séquence
  - et les exercices de la fiche de TP