Introduction à PHP

Cookies et Sessions

monnerat@u-pec.fr

9 avril 2026

IUT de Fontainebleau

Sommaire

Cookies

Cookies

Le principe

Introduction

Cookie (RFC 6265)

Les cookies permettent de stocker du coté client des données associées à un visiteur particulier.

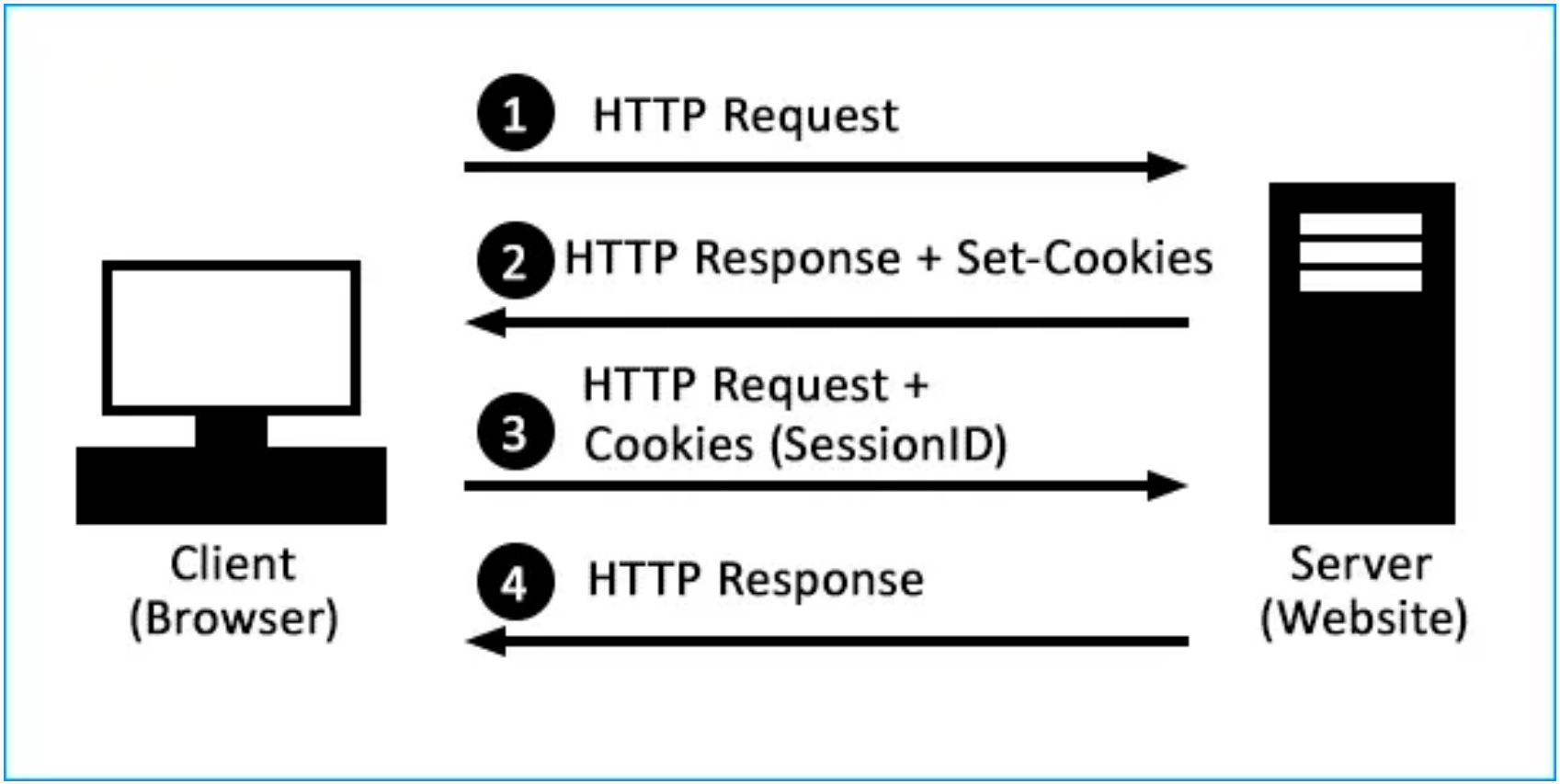
Pour permettre par exemple :

- d'identifier de manière unique un visiteur.
- de préremplir les données d'un formulaire le concernant pour lui éviter de les resaisir.
- de connaître ses derniers achats.

D'abord conçus par Netscape, ils sont maintenant pris en charge par tous les navigateurs, et intégré au protocole HTTP.

- Petit fichier texte stocké par le navigateur sur l'ordinateur du visiteur, à la demande du serveur.
- Le serveur envoie un cookie avec la page, stocké coté client.
- Il demande ainsi au navigateur du client de le lui renvoyer à chaque échange ; (le serveur peut ainsi accéder à l'information du cookie, la modifier...)

Les navigateurs peuvent refuser.


1 HTTP Request
2 HTTP Response + Set-Cookies
3 HTTP Request + Cookies (SessionID)
4 HTTP Response
Client (Browser)
Server (Website)

Cookies et HTTP

La demande de cookie (serveur $\rightsquigarrow$ client) se fait dans les en-têtes HTTP :

```
Set-Cookie : NOM=VALEUR; domain=NOM DE DOMAINE; expires=DATE
```

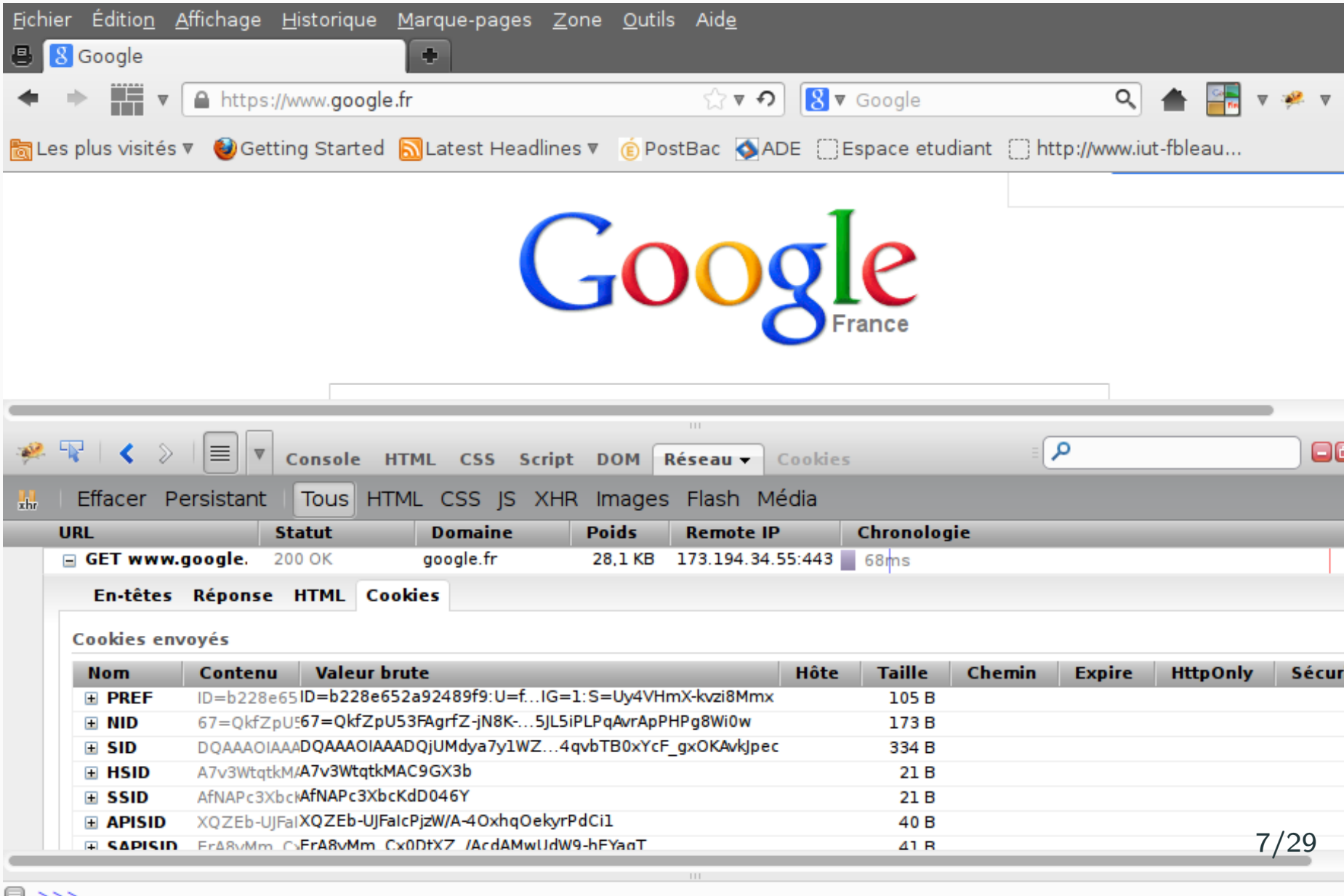
Envoi des cookies au serveur (client $\rightsquigarrow$ serveur) lorsque le client fait une requête à un serveur, les cookies pour le domaine et le chemin sont envoyé dans les en-têtes HTTP sous la forme

```
Cookie : NOM1=VALEUR1; NOM2=VALEUR2; ...
```

Remarques

- La taille maximale d'un cookie est de 4 ko.
- PHP permet de les gérer.

Exemple avec google



Cookies

Mise en oeuvre en PHP

La fonction `setcookie`

La fonction

```
bool setcookie ( string $name [, string $value
  [, int $expire = 0 [, string $path [, string $domain
  [, bool $secure = false  [, bool $httponly = false ]]]]]] )
```

est la fonction qui permet de gèrer les cookies. `setcookie()` définit un cookie qui sera envoyé avec le reste des en-têtes. Les cookies doivent passer avant le corps de la requête (c'est une restriction HTTP, pas de PHP). Cela vous impose d'appeler cette fonction avant toute balise html ou écriture depuis php dans le "flux html".

En cas d'echec, setcookie() retournera FALSE. Si setcookie() réussit, elle retournera TRUE. Cela n'indique pas si le client accepte ou pas le cookie. La RFC 6265 est la référence pour l'interprétation des paramètres.

Signature alternative depuis PHP 7.3.0

```
setcookie ( string $name ,
        string $value = "" ,
        array $options = [] ) : bool
```

Paramètres de `setcookie`

- `name` : le nom du cookie. 'cookiename' est appelé via `$_COOKIE['cookiename']`
- `value` : la valeur du cookie cette valeur est retrouvé en utilisant `$_COOKIE['cookiename']`
- `expire` : le temps après lequel le cookie expire. timestamp Unix, donc, ce sera un nombre de secondes depuis l'époque Unix (1 Janvier 1970). En d'autres mots, vous devriez fixer cette valeur à l'aide de la fonction time() et en y ajoutant le nombre de secondes après lequel on veut que le cookie expire. Vous pouvez utiliser aussi mktime()

Exemple

time()+60*60*24*30 fera expirer le cookie dans 30 jours. Si vous ne spécifiez pas ce paramètre, le cookie expirera à la fin de la session (lorsque le navigateur sera fermé).

Paramètres de `setcookie`

- `path` : le chemin sur le serveur sur lequel le cookie sera disponible

Exemple

Si la valeur est '/', le cookie sera disponible sur l'ensemble du domaine domain. Si la valeur est '/foo/', le cookie sera uniquement disponible dans le répertoire /foo/ ainsi que tous ces sous-répertoires comme /foo/bar/ du domaine domain. La valeur par défaut est le répertoire courant où le cookie a été défini.

- `domain` : le domaine où le cookie est disponible.

Exemple

Pour rendre le cookie disponible sur tous les sous-domaines de example.com, vous devez mettre la valeur '.example.com'. Le point (.) n'est pas requis mais est nécessaire pour la compatibilité avec encore plus de navigateurs.

Paramètres de `setcookie`

- `secure` : indique si le cookie doit uniquement être transmis à travers une connexion sécurisée HTTPS. Lorsqu'il est positionné à TRUE, le cookie ne sera positionné uniquement si la connexion sécurisée existe. La valeur par défaut est FALSE.
- `httponly` : lorsque ce paramètre vaut TRUE, le cookie n'est pas accessible depuis les langages de scripts coté client, comme Javascript.

Tous les arguments sauf name (nom) sont optionnels.

Remarque

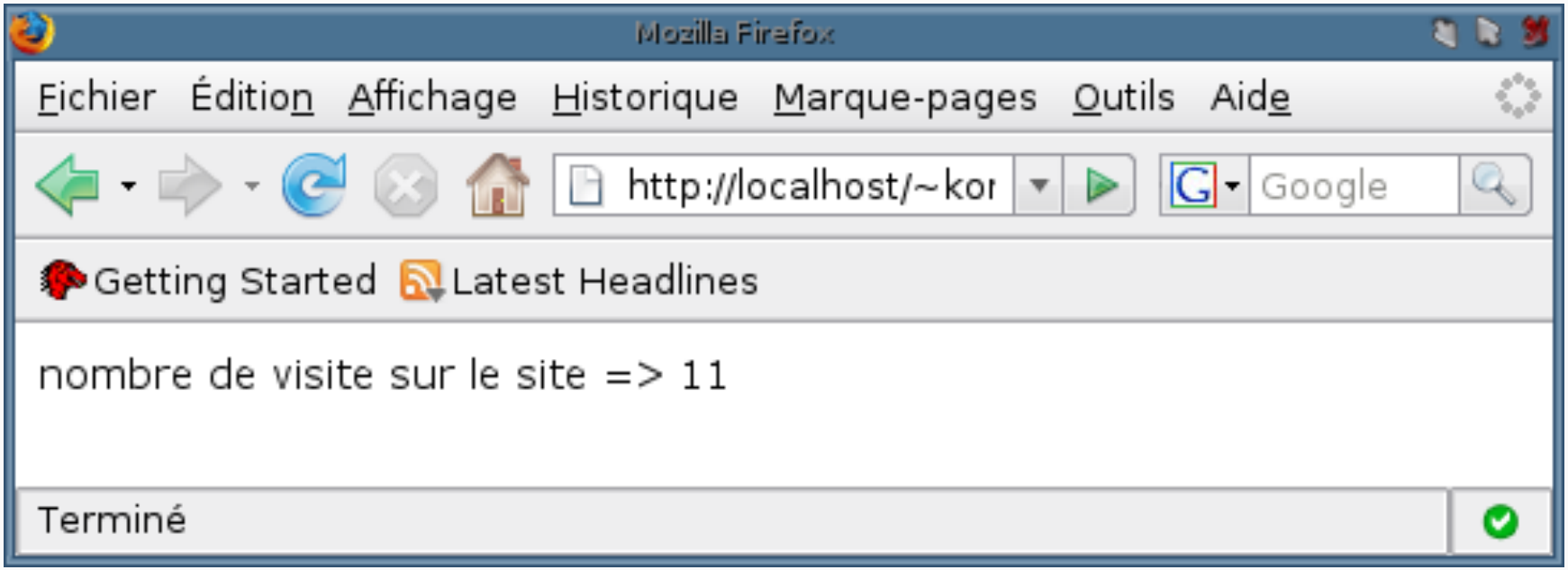
Pour effacer un cookie, il faut utiliser `setcookie` avec une date antérieure à la date courante. Le plus simple :

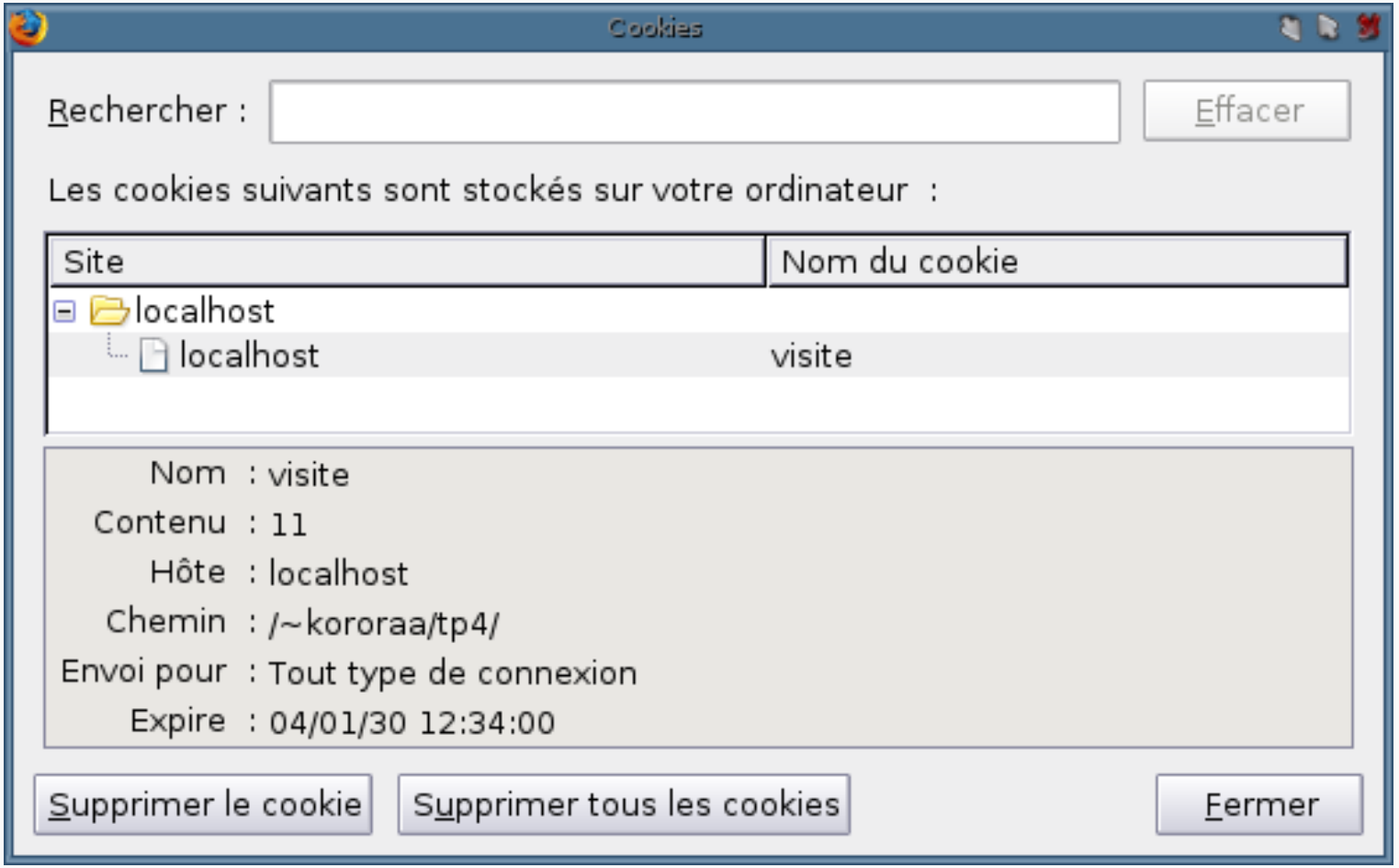
```
setcookie("moncookie","",1);
```

Exemple

```php
<?php
$visite = filter_input(INPUT_COOKIE, 'visite',
  FILTER_VALIDATE_INT
);
if (in_array($visite, [null,false], true))
  $visite = 1;
else
  $visite ++;
setcookie('visite',$visite,mktime(12,34,00,04,01,2030));
$message="nombre de visite sur le site => $visite";
?>
<html>
  <head>
  </head>
  <body>
    <?php echo $message;?>
  </body>
</html>
```

- La première fois, le cookie est créé, avec la valeur 1.
- Par la suite, il est envoyé au serveur, récupéré , incrémenté, et renvoyé au client.




Cookies
Rechercher :
Effacer
Les cookies suivants sont stockés sur votre ordinateur :
Site
Nom du cookie
localhost
localhost
visite
Nom : visite
Contenu : 11
Hôte : localhost
Chemin : /~kororaa/tp4/
Envoi pour : Tout type de connexion
Expire : 04/01/30 12:34:00
Supprimer le cookie
Supprimer tous les cookies
Fermer

Remarque

Pour stocker une information plus complexe qu'un nombre ou une chaîne de caractères, par exemple un tableau, utilisez les fonctions :

- `serialize` pour transformer le tableau en une chaîne de caractères, stockable dans un cookie.
- `unserialize` pour réaliser l'opération inverse.

```php
$moi=array (
  "nom"=>"monnerat",
  "bureau"=>114
);
echo serialize($moi);
```

affiche la chaîne :

```
a:2:{s:3:"nom";s:8:"monnerat";s:6:"bureau";i:114;}
```

Sessions

Dans la partie précédente, nous avons vu que les cookies permettent de stocker de l'information coté client.

Inconvénients

- taille des données est limitée.
- le client peut directement les modifier.

Pour les données "sensibles" $\Rightarrow$ Sessions

Définition

Session

Une session est un ensemble d'informations associé avec un utilisateur particulier et conservé (sur le serveur) tout au long de son interaction avec les différentes pages d'un site Web.

- Une session attribue à un visiteur un identifiant unique, stocké dans un cookie.
- A chaque fois qu'il revient en donnant cette identifiant, PHP peut récupérer toutes les informations sauvegardées pour cette utilisateur.
- Les données sont sauvegardées sur le serveur.
- Contrairement aux cookies, les sessions ont une durée de vie limitée, en particulier à la fermeture du navigateur.

Les sessions sont un peu plus adaptées au stockage de données confidentielles, par exemple :

- Contrôle d'accès après authentification.
- Gestion d'un panier d'achat.
- Formulaire en plusieurs parties. (on stocke les données déjà saisies)
- Disposer d'un cache pour certaines ressources très coûteuses. (requête sur une base)

Les sessions s'appuient sur la notion de cookie, donc ne résolvent pas complétement le problème de la sécurité.

Utilisation

1. `session_start() /* initialisation*/`
   - Cette fonction initialise la session. PHP essai de lire l'identifiant fourni par l'utilisateur, va chercher le fichier correspondant et met à disposition l'information correspondante dans la variable (super globale) `$_SESSION`
   - Si aucun identifiant de session n'est reçu, PHP en crée un unique aléatoire, l'envoie au visiteur et crée le fichier pour stocker l'information de session.
2. Par la suite, l'information est disponible en lecture ou écriture dans le tableau associatif `$_SESSION`.
3. `session_destroy() /* suppression de la session */`
   - La fonction détruit le fichier de données associé sur le serveur.
   - Après `session_destroy`, dans le script en cours, on peut effacer le tableau `$_SESSION=[]`.
4. Utiliser `setcookie` pour détruire le cookie de session du client. `setcookie(session_name(),'',0)`

Remarque : Le numéro de session est généré dans l'entête http. Il ne faut donc mettre aucun code html (pas même une ligne blanche) avant l'appel à la fonction `session_start()`

Mise en oeuvre

- L'identifiant de session est stocké dans un cookie nommé par défaut PHPSESSID
- Ce cookie contient l'identifiant assigné à la session.
- S'il n'existe pas, il est crée de manière aléatoire puis envoyé au navigateur par cookie (sans date d'expiration).

C'est le fichier php.ini qui gère les sessions

```
; Whether to use cookies.
session.use_cookies = 1

;session.cookie_secure =

; This option enables administrators to make their users invulnerable t
; attacks which involve passing session ids in URLs; defaults to 0.
 session.use_only_cookies = 0

; Name of the session (used as cookie name).
session.name = PHPSESSID

; Initialize session on request startup.
session.auto_start = 0

; Lifetime in seconds of cookie or, if 0, until browser is restarted.
session.cookie_lifetime = 0
```

Contrôle d'accès avec une session

1. Un formulaire HTML demande un nom utilisateur et un mot de passe.
2. Le script qui traite le formulaire vérifie que nom d'utilisateur et mot de passe fournis sont corrects (en les comparant à des valeurs stockées dans une base de données, par exemple) :

si c'est le cas, il crée et affecte une nouvelle variable de session :

```
$_SESSION['valid_user']=1;
```

ensuite il redirige l'utilisateur vers une des pages protégées.

Si le login n'est pas correct le script redirige vers le formulaire de login.

Toutes les pages protégées commencent par vérifier que l'utilisateur a bien été authentifié :

```php
<?php
include dirname(__FILE__).'/verification.php';
?>
```

en incluant le fichier

```php
if(!isset($_SESSION['valid_user'])
  || $_SESSION['valid_user']!=1) {
  header('Location:authentification.html');
  exit;
}
```

Le site doit prévoir au moins une page de déconnexion qui fait appel à `session_destroy`

login/password

- Dans la base de données, on ne stocke jamais les mots de passes en clair.
- On stocke une empreinte (`hash`) du mot de passe.

Il est recommandé d'utiliser la fontion `password_hash`

```
echo password_hash("toto",PASSWORD_DEFAULT);
```

donne

```
$2y$10$Ymz.xrvVrKqgGQWm6r.ux.e9rBafOsChRjGLDinAoW6UnePP0T/Tu
```

La fonction utilise (entre autres) des paramètres par défaut (on peut les fixer)

- un coût de calcul
- un sel

Pour vérifier l'empreinte d'un mot de passe, on utilisera la fonction `password_verify`

```php
<?php
$hash = '$2y$10$Ymz.xrvVrKqgGQWm6r.ux.e9rBafOsChRjGLDinAoW6UnePP0T/Tu

if (password_verify('toto', $hash)) {
    echo 'Le mot de passe est valide !';
} else {
    echo 'Le mot de passe est invalide.';
}
?>
```

Vérification

Le mot de passe est considéré comme correct si son empreinte (calculée) est identique à celle stockée dans la base de données.

Un exemple

- Le jeu de la devinette : le retour. Il s'agit d'essayer de deviner un nombre entre 0 et 100.

- Variables de sessions `essais` (historique des coups) et `nb_mystere`.

```
[root@portabledenis tmp]# cat sess_s4bktd5gl2dlmrqaan8gr27hub
essais|a:0:{}nb_mystere|i:82;nb_coups|i:0;[
```

```php
<?php
session_start(); //On initialise la session
include 'vues/header.php';
include 'vues/form.php';

if (!isset($_SESSION['nb_mystere'])){
  $_SESSION['essais'] = array();
  $_SESSION['nb_mystere'] = mt_rand(0, 100);
  $_SESSION['nb_coups'] = 0;
} else {
  $nombre_entre = filter_input(INPUT_POST,'nombre',FILTER_VALIDATE_INT);
  if ($nombre_entre !== null && $nombre_entre !== false){
    $_SESSION['essais'][] = $nombre_entre;
    $_SESSION['nb_coups']++;

    if ($_SESSION['nb_mystere'] > $nombre_entre){
      $message = "C'est plus !";
    }
    elseif ($nombre_entre > $_SESSION['nb_mystere']){
      $message = "C'est moins !";
    } else {
      session_destroy();
      $message =  'Bravo, <b>vous avez trouvé le nombre mystère ! </b>';
      $message .=  'Vous avez réussi en '.$_SESSION['nb_coups'].' coups';
    }
  }
  if (isset($message))
    include 'vues/message.php';
  include 'vues/histo.php';
  include 'vues/footer.php';
}
?>
```