# Programmation Web avec ***PHP***

bases de données, sessions

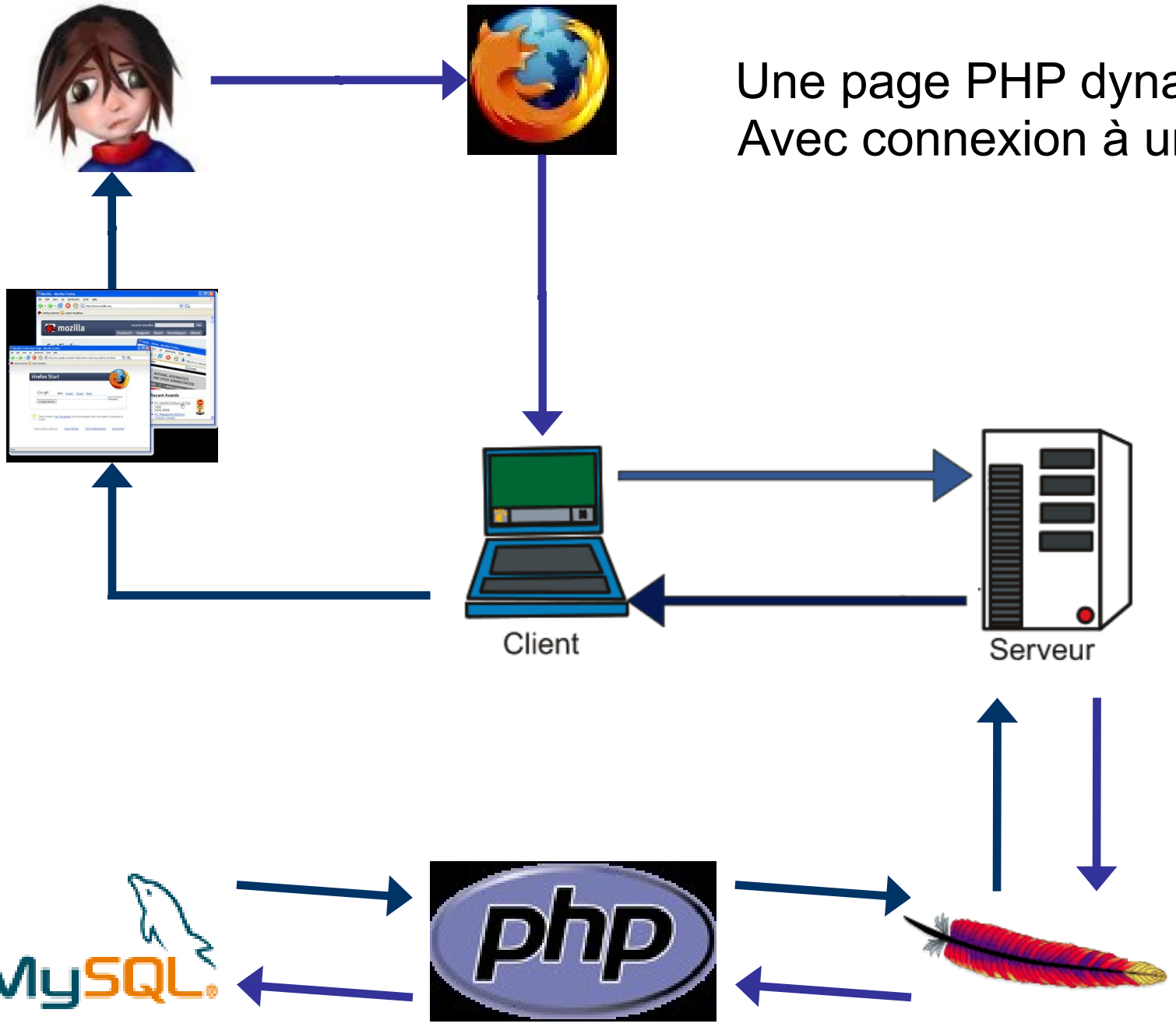
cours de M. Clergue
assaisonné par S. Verel

# Objectifs de la suite du cours

Maîtriser les techniques à la base de la création de sites web dynamiques côté serveur :

- génération automatique de pages HTML
- récupération des données utilisateur
- création et interrogation des bases de données

# Une page PHP dynamique
# Avec connexion à une BD



# pré-requis BD

1. Que signifie SQL ?

2. Quels sont les bases théoriques des BD ?

3. Dessiner un exemple de BD en anotant votre schéma

# pré-requis BD

4. Qu'est-ce qu'une clé primaire ?

5. Comment crée-t-on une table en sql ?

6. Comment efface-t-on une table ?

   A. CLEAR
   B. CLEARLY
   C. DROP
   D. DROPY

# Aide mySQL

Pour des compléments en SQL, et des compléments mySQL/PHP consulter par exemple :

http://cyberzoide.developpez.com/

# Quatre étapes pour questionner une BD

1. Connexion au serveur
2. Sélection de la base
3. Exécution de la requête
4. Traitement des données

# 1.Connexion au serveur

```
mysql_connect($server, $user, $pass) or
                        die('Erreur de connexion');
```

Ici :

```
$server = 'localhost:8889';
$user = 'etudiant';
$pass = 'toctoc';
```

# 2. Sélection de la base

```
mysql_select_db($db) or
                    die('Base inexistante');
```

Ici :

```
$db = 'test';
```

# 3. Exécution de la requête

```
$query = mysql_query($sql) or
                        die( 'Erreur' );
```

Ici :

```
$sql =  'select * from exemples;'
```

# 4.Traitement des données

```
while ( $list = mysql_fetch_array( $query ) ) {
…
}
```

# Exemple

```
$connexion = mysql_connect('localhost:8889', 'etudiant', 'toctoc');
if (! $connexion) echo "<div>pb de connexion</div>";
$bd = mysql_select_db( 'test', $connexion);
if (! $bd) echo "<div>pb de selection de bd</div>";
$requete = mysql_query("select * from exemple;", $connexion);
if ($requete) {
echo "<ul>";
while ($resultat = mysql_fetch_array($requete)) {
echo "<li>".$resultat["name"]." a pour numéro ".$resultat["tel"]."</li>"; }
echo "</ul>"; }
else echo "<div>pb de requete</div>";
```

# A Faire en TP

- Une BD répertoire téléphonique (ou une autre idée)
- Une page PHP pour l’afficher

# Le problème : une requête au lieu de…

```
<?php
// Demande à la base de vérifier si un utilisateur correspond
$query = "SELECT * FROM users WHERE user='{$_POST['username']}' AND
    password='{$_POST['password']}'";
mysql_query($query);

// Nous ne vérifions pas $_POST['password'], il peut contenir ce que l'utilisateur veut !
$_POST['username'] = 'aidan';
$_POST['password'] = "_' OR '_'='_"; // les _ sont là pour marquer les ' et les "

// Cela signifie que la requête envoyée à MySQL sera :
echo $query;
?>
```

Requête envoyée :

```
SELECT * FROM users WHERE name='aidan' AND password='_' OR '_'='_'
```

# Un peu de sécurité

- Principe pour forcer une bd : écrire des requêtes là où elles ne sont pas attendues
- Précautions :
  - Limiter les droits d'accès
  - Cryptage…
  - Vérification des champs (formulaire) avant insertion (requête)
    - is_bool(), is_null(), is_float(), is_int(), is_string(), is_object(), is_array()
    - Vérification de la longueur... Du contenu…
      - Avec du javascript
      - Lors de l'envoi

# Les sessions

Ou comment conserver des informations d'une page à une autre

# Les sessions

- Le transfert d'information d'une page (script) à un autre n'est pas simple en PHP :
  - Le protocole HTTP ne permet pas de conserver les connections
- Plusieurs solutions :
  - Par les cookies :  pas simple, non sécurisé
  - Par les sessions : mécanisme dédié interne à PHP

# Passage par cookies

- PHP propose des fonctions pour :

Poser un cookie :

```
<?
  $expire = 365*24*3600;// durée du cookie en sec, 1 an
  setcookie("lenom","lavaleur",time()+$expire);
?>
```

Récupérer un cookie :

```
<? // On affiche la valeur de nickname
  echo 'le cookie "lenom" a pour valeur : ';
  echo $_COOKIE["lenom"];
?>
```

# Les sessions PHP

- PHP propose un système intégré pour gérer le passage d'information
- Il est basé sur la notion de session :
  - Chaque visiteur accédant à un site se voit assigner un identifiant unique, appelé "identifiant de session".
  - Il est soit stocké dans un cookie, soit propagé dans l'URL.

# Les sessions PHP

En pratique :

2 fonctions :

```
session_start();
session_destroy();
```

1 variable super-globale :

```
$_SESSION
```

# Les sessions PHP

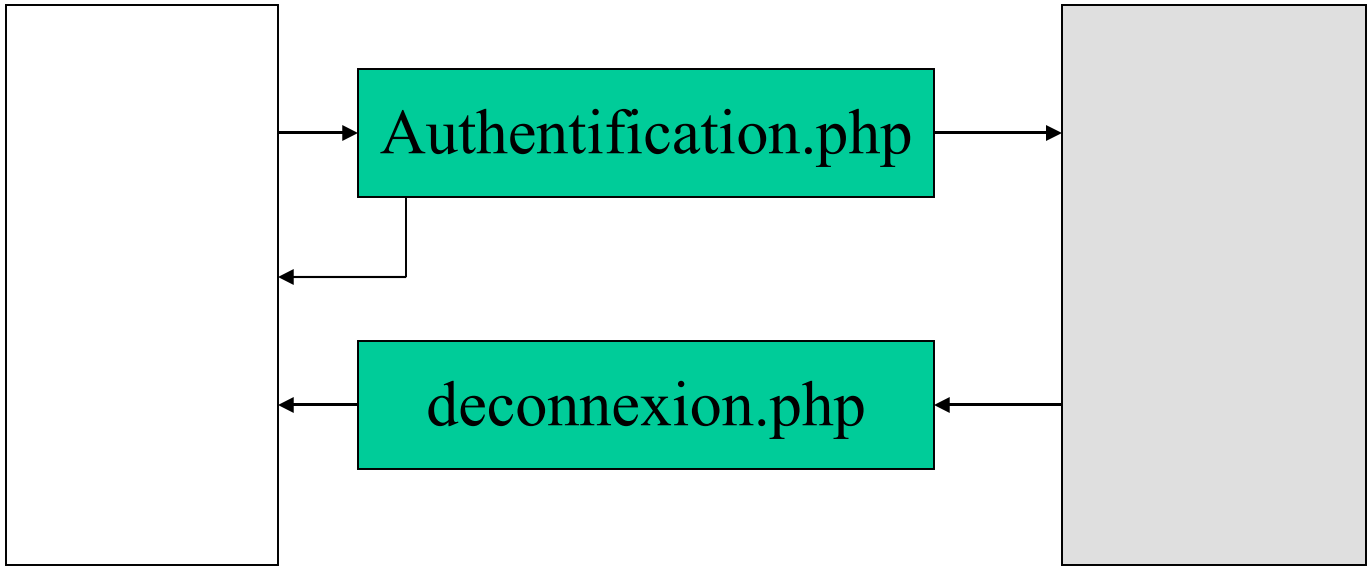
Exemple :

```php
<?php
  // On démarre la session
  if(!isset($_SESSION))session_start();

  if ( isset($_SESSION['toto'])) {
     $_SESSION['toto'] +=1;
  }
  else {
      $_SESSION['toto'] =1;
  }
?>
```

# Les sessions PHP

Application : accés authentifié à certaines pages



# (Eléments de) Programmation Objet

En PHP5

# Concepts

- Structuration
  - Objet, classe et instance
- Héritage et composition

# Objet, classe et instance

- Objet = état + opérations applicables

  un objet est une structure informatique caractérisée par un état et un ensemble d'opérations exécutables par cet objet qui caractérise son comportement.

- Classe = modèle d'objet

  une ***classe*** est un modèle de la structure statique (les **champs** ou **attributs** ou **membres**) et du comportement dynamique (les **opérations** ou **méthodes**) des objets associés à cette classe, que l'on appelle ses instances.

```php
// définition de la classe
class Chien {
  // déclaration d'un membre
  public $nom = 'brutus'; //une valeur par défaut
  // déclaration d'une méthode
  function aboie() {
     print "Grrrrrrrr...Ouaf...Ouaf!\n<br>";
  }
}
// création d'une instance de la classe Chien
$medor = new Chien();
// manipulation d'un attribut
$medor->nom = "médor";
// appel d'une methode
$medor->aboie();
```

# Portée des attributs et des méthodes

- Public

  L'attribut (la méthode) est accessible à l'intérieur et à l'extérieur de la classe
- Private

  L'attribut (la méthode) est accessible à l'intérieur de la classe et inaccessible à l'extérieur
- Protected

  …

# Portée des attributs (à l'intérieur d'une classe)

- Attention : la portée des variables ne passe pas "naturellement" les frontières des fonctions

```
var i = 5;

function foo(){echo i;} //erreur
```

- Quid des attributs et des méthodes ?

```
class Chien {
  public $nom = 'brutus';
  function aboie() {
     print $nom." dit:Grrr...Ouaf...Ouaf!\n<br>";
  }
}
$medor = new Chien(); $medor->aboie(); // erreur
```

# Portée des attributs (à l'intérieur d'une classe)

```
class Chien {
   public $nom = 'brutus';
   function aboie() {
       print $nom." dit:Grrr...Ouaf...Ouaf!\n<br>";
   }
}
$medor = new Chien(); $medor->aboie(); // erreur
```

***Pourquoi l'erreur ?***

A. aboie() n'est pas public

B. $nom est défini dans la classe

C. Brutus était pour Mac Cain

D. $nom est locale à aboie()

# Pseudo variable **this**

C'est une variable (pas tout à fait, on ne peut rien lui assigner) faisant référence à l'objet courant.

```
class Chien {
      public $nom = 'brutus';
      function aboie() {
            print $this->nom." dit:Ouaf!\n<br>";
      }
}
$medor = new Chien();
$medor->nom = "médor";
$medor->aboie(); // affiche médor dit:Ouaf!
```

```
class Chien {
  private $nom = 'brutus';
  function setNom($nom){
     $this->nom = $nom;
  }
  function aboie() {
          print $this->nom." dit:Ouaf!\n<br>";
     }
}
$medor = new Chien();
$medor->nom = "médor"; // erreur
$medor->setNom("médor"); // OK
```

# Constructeur

- Les classes sont munies d'un constructeur par défaut (sans paramètre)
- Il est possible de définir son propre constructeur

PHP4 style (PHP5 compatible):

```
function Chien($nom) {$this->nom = $nom;}
```

PHP5 style :

```
function __construct ($nom) {$this->nom = $nom;}
```

# Héritage

- *Copier/coller* conceptuel

```
class Chien {
  private $nom = 'brutus';
  function __construct($nom){
      $this->nom = $nom;
  }
  function setNom($nom){
      $this->nom = $nom;
  }
  function aboie() {
          print $this->nom.
          " dit:Ouaf!\n<br>";
      }
}
```

```
class Chiot {
…
}
```

```
class Chiot {
  private $nom = 'brutus';
  public $cri = 'ieeeek';
  function __construct($nom,$cri){
     $this->nom = $nom;
     $this->cri = $cri;
  }
  function setNom($nom){
     $this->nom = $nom;
  }
  function aboie() {
          print $this->nom.
          " dit:".$this->cri."\n<br>";
     }
}
```

```
class Chiot extends Chien{
  public $cri = 'ieeeek';
  function __construct($nom,$cri){
     parent::__construct($nom);
     $this->cri = $cri;
  }
  function aboie() {
          print $this->nom.
          " dit:".$this->cri."\n<br>";
     }
}
```

```
class Chiot extends Chien{
  public $cri = 'ieeeek';
  function __construct($nom,$cri){
     parent::__construct($nom);
     $this->cri = $cri;
  }
  function aboie() {
          print $this->nom.
          " dit:".$this->cri."\n<br>";
          Chien::aboie();
          }
}
```

# Abstraction

**Classes abstraites :**

- ```
  abstract class Chien {
      abstract function aboie();
  }
  ```

**Interfaces :**

```
interface Animal {
   public peutAvancer();
}

Class Chien implements Animal { ... }
```
```

# Upload de fichier

**formulaire.html**

```
<!-- Le type d'encodage des données, enctype, -->
<!-- DOIT être spécifié comme ce qui suit     -->
<form enctype="multipart/form-data"
action="telechargement.php" method="post">
  <!-- MAX_FILE_SIZE doit précéder le -->
  <!-- champ input de type file       -->
  <input type="hidden"
         name="MAX_FILE_SIZE" value="3000000" />
  <!-- Le nom de l'élément input détermine -->
  <!-- le nom dans le tableau $_FILES -->
  Envoyez ce fichier :
  <input name="userfile" type="file" />
  <input type="submit" value="Envoyer le fichier" />
</form>
```

```php
<?php
////////////////////////////////////////////////////////
// deux ou trois petites verifications avant tout !!
////////////////////////////////////////////////////////
// existence du telechargement
if(!isset($_FILES['userfile'])){
        // dans ce cas, soit le fichier depasse la limite
       // imposee par upload_max_filesize (par defaut 8M),
        // soit on ne vient pas a cette page
       //par formulaire.php et donc on sort
        exit("problème telechargement");
}
////////////////////////////////////////////////////////
// taille du telechargement
$fsize = $_FILES["userfile"]["size"];
if($fsize == 0 || $fsize > 2621000) {
        // la taille depasse le MAX_FILE_SIZE
        //du formulaire (==0)
        // ou depasse une limite ( > 2621000)
        ///idealement ldentique a MAX_FILE_SIZE
        exit("keep the filesize ($fsize) under 3MB!!");
}
```

telechargement.php

```php
////////////////////////////////////////////////////////
// l'extension du fichier :
$path_parts = pathinfo($_FILES['userfile']['name']);
// les echos juste pour montrer les resultats de pathinfo
echo $path_parts['dirname'], "<br />";
echo $path_parts['basename'], "<br />";
echo $path_parts['extension'], "<br />";

if(!preg_match('/(gif|jpg|jpeg|png|bmp)$/i',
                $path_parts['extension'])){
     // l'extension ne fait pas partie
     // des extensions acceptees
     exit("type de fichier non reconnu");
}
```

telechargement.php

```php
////////////////////////////////////////////////////////////
// debut de l'upload
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
// le repertoire dans lequel le fichier va être uploade
$uploaddir = './upload/' ;
// le chemin complet du fichier :
// $uploaddir : le repertoire
// basename($_FILES['userfile']['name']) :
// le nom du fichier entré par l'utilisateur,
// sans le repertoire
$uploadfile = $uploaddir .
basename($_FILES['userfile']['name']);
////////////////////////////////////////////////////////////
// verification de l'existence du fichier
// par defaut le fichier existant est ecrasé
if(file_exists($uploadfile)){
      exit("fichier deja telechargé");
}
```

telechargement.php

```
////////////////////////////////////////////////////////
// $_FILES['userfile']['tmp_name'] :
// le nom du fichier temporaire
// $uploadfile : le nom du fichier
// (Si le fichier de destination existe déjà,
// il sera écrasé)
if (move_uploaded_file(
            $_FILES['userfile']['tmp_name'],
            $uploadfile)) {
   echo "Le fichier est valide, et a été téléchargé
         avec succès. Voici plus d'informations :\n";
}
else {
   echo "Attaque potentielle
         par téléchargement de fichiers.
         Voici plus d'informations :\n";
}
```

telechargement.php

```
////////////////////////////////////////////////////////////
echo 'Voici quelques informations de débogage :';
print_r($_FILES);
////////////////////////////////////////////////////////////

if ($_FILES['nom_du_fichier']['error']) {
  switch ($_FILES['nom_du_fichier']['error']){
    case 1: // UPLOAD_ERR_INI_SIZE
    case 2: // UPLOAD_ERR_FORM_SIZE
    case 3: // UPLOAD_ERR_PARTIAL
    case 4: // UPLOAD_ERR_NO_FILE
  }
}
else {
 // $_FILES['nom_du_fichier']['error'] vaut 0
 //soit UPLOAD_ERR_OK
 // ce qui signifie qu'il n'y a eu aucune erreur }
```

telechargement.php

# Entrées / Sorties avec les fichiers

- $ressource = fopen ($filename, $mode)
  - crée une ressource nommée, spécifiée par le paramètre filename, sous la forme d'un flux
  - Retourne faux en cas de problème (et warning)
  - Mode : ‘r’, ‘r+’, ‘w’, ‘w+’, ‘a’, ‘a+’, ‘x’, ‘x+’
    - + : signifie : lecture et écriture
    - r : lecture à partir du début du fichier
    - w : écriture (et écrasement du fichier)
    - a : concaténation
    - Pour w, a : si le fichier n’existe pas, on le crée
    - x : création en lecture seule. Si  le fichier exite, retourne faux

# Entrées / Sorties avec les fichiers

- fclose ($ressource)
  - Ferme la ressource ouverte avec fopen.
  - Retourne vrai en cas de succès, faux sinon
- feof ($ressource)
  - feof( ) retourne TRUE si le pointeur handle est à la fin du fichier ou si une erreur survient, sinon, retourne FALSE.
  - Attention à ne pas se tromper de $ressource… boucle infinie…
- $msg = fread ($ressource, $length)
  - fread( ) lit jusqu'à $length octets dans le fichier référencé par $ ressource.
  - La lecture s'arrête lorsque length octets ont été lus ou que l'on a atteint la fin du fichier
  - Retourne la chaîne lue ou FALSE si une erreur survient.
- fwrite ($ressource, $chaine, $length) // $length optionel
  - fwrite( ) écrit le contenu de la chaîne $chaine dans le fichier pointé par $ressource.
  - Si la longueur $length est fournie, l'écriture s'arrêtera après $length octets ou à la fin de la chaîne (le premier des deux).
  - fwrite() retourne le nombre d'octets écrits ou FALSE en cas d'erreur.

# Entrées / Sorties avec les fichiers

- $ligne fgets( $ressource, $length ) // $length optionel depuis PHP 4.2.0
  - retourne la chaîne lue jusqu'à la longueur $length - 1 octet depuis le pointeur de fichier $ressource, ou bien la fin du fichier, ou une nouvelle ligne (qui est inclue dans la valeur retournée). Si aucune longueur n'est fournie, la longueur par défaut est de 1 ko ou 1024 octets.
- fgetss ( $ressource, $length, $tag_ok )
  - $length, $tag_ok optionnel  ($length depuis php 5)
  - Idem que fgets, mais en supprimant les tag html, sauf ceux dans $tag_ok
- fgetc ($ressource)
  - retourne une chaîne contenant un seul caractère, lu depuis le fichier pointé par handle.
  - fgetc( ) retourne FALSE à la fin du fichier
- inout.php

# Entrées / Sorties avec les fichiers

- flock ($ressource, $operation)
  - flock() agit sur le fichier ($ressource qui doit avoir été ouvert au préalable.
  - operation est une des valeurs suivantes :
    - Acquisition d'un verrou en lecture : operation = LOCK_SH
    - Acquisition d'un verrou exclusif en écriture : operation = LOCK_EX
    - Libération d'un verrou partagé ou exclusif, operation = LOCK_UN
    - Si vous voulez que flock( ) ne se bloque pas durant le verrouillage, ajoutez LOCK_NB à operation.
  - Cette fonction retourne TRUE en cas de succès, FALSE en cas d'échec.
  - Le verrou est également levé avec la fonction fclose()  (qui est également automatiquement appelée lors de la fin du script).

# Entrées / Sorties avec les fichiers

- $nb_octets_lus = readfile ($filename) -- Affiche un fichier (dans la page web)
- $chaine_lue = file_get_contents ($filename)
  - Comme readfile, sauf lit tout un fichier dans une chaîne
- $nb_octets_lus = fpassthru( $ressource) -- Affiche le reste du fichier

- $result = fscanf ($ressource, $format)
  - Analyse un fichier en fonction d'un format (comme en C)
  - Le résultat peut être une chaîne ou un tableau
- fstat -- Lit les informations sur un fichier à partir d'un pointeur de fichier
- ftruncate -- Tronque un fichier

# Entrées / Sorties avec les fichiers

- fseek($ressource, $offset [, $whence])
  - modifie le curseur de position dans le fichier $ressource. La nouvelle position mesurée en octets à partir du début du fichier est obtenue en additionnant la distance offset à la position $whence. Ce paramètre peut prendre les valeurs suivantes :
    - SEEK_SET - La position finale vaut offset octets.
    - SEEK_CUR - La position finale vaut la position courante ajoutée à offset octets.
    - SEEK_END - La position finale vaut la position courante par rapport à la fin du fichier, ajoutée de offset.
    - Si $whence n'est pas spécifiée, il vaut par défaut SEEK_SET.
  - **fseek( )** retourne 0 en cas de succès, et sinon -1. Notez que positionner le pointeur au-delà de la fin du fichier n'est pas une erreur.
- $position = ftell($ressource) -- Renvoie la position du pointeur du fichier
- rewind($ressource)
  - replace le pointeur du fichier $ressource au début.
  - Cette fonction retourne TRUE en cas de succès, FALSE en cas d'échec.
  - Si vous avez ouvert le fichier en mode d'ajout ("a" ou "a+"), toutes les données que vous écrirez dans ce fichier seront toujours ajoutées à la fin, sans se soucier de la position du pointeur de fichier.

# Entrées / Sorties avec les fichiers

- $nb_octets_ecrit = file_put_contents( $filename, $a_ecrire)
  - Ecrit une chaîne ou un tableau ($a_ecrire) dans un fichier de nom $filename
- fflush($ressource) -- Envoie tout le contenu généré dans un fichier
- tempnam ( $dir, $prefix)
  - crée un fichier temporaire unique dans le dossier $dir. Si le dossier n'existe pas, tempnam( ) va générer un nom de fichier dans le dossier temporaire du système. Le nom sera préfixé par le paramètre $prefix.
  - tempnam ( ) retourne le nom du fichier temporaire ou FALSE en cas d'échec.
  - À effacer « manuellement »
- $ressource = tmpfile ( )
  - tmpfile() crée un fichier temporaire avec un nom unique, ouvert en écriture et lecture (w+), et retourne un pointeur de fichier, identique à ceux retournés par fopen(). Ce fichier sera automatiquement effacé lorsqu'il sera fermé (avec fclose()), ou lorsque le script sera terminé.