Qualiente

Le Web dans tous ses médias

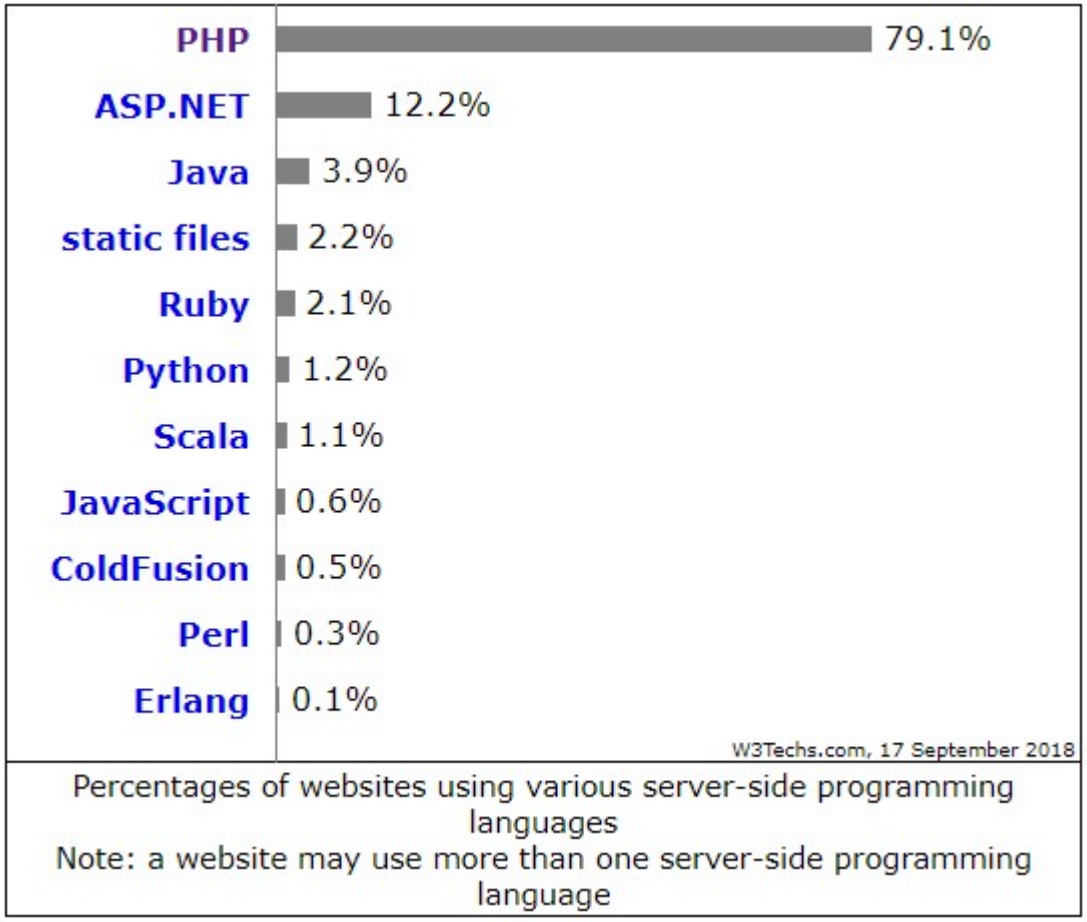
Le langage PHP

Formateur | Pierre Vélon

Introduction

- En HTML, le contenu est dit statique
  - Exemple : <h1>Nous sommes le 1er janvier 1970</h1>
    - Cette ligne donnera toujours la même date, quel que soit le jour d'ouverture du fichier
- Pour répondre aux besoins des entreprises, il est indispensable de concevoir des sites Web dynamiques
  - A partir d'un même code source, une page doit pouvoir s'afficher différemment, selon le contexte utilisateur
    - Age de l'internaute, profession, droits sur l'application, jour de connexion, mot-clé saisi dans un moteur de recherche…
  - Exemple : rubrique « Mon compte » d'un site Web
- Il existe différents langages de programmation pouvant permettre de développer ce type de système d'information
  - PHP, .net, Java, Python, Ruby, Perl…

Statistiques des différents langages



Percentages of websites using various server-side programming languages
Note: a website may use more than one server-side programming language

Le langage PHP

- Le langage PHP dispose de plus de 4000 instructions standards (fonctions) pour développer des applications Web dynamiques
  - Affichage, opérations mathématiques, manipulations de chaînes de caractères, validation de formats, stockage de données…
- PHP est un langage opensource
  - Gratuit, modifiable, redistribuable
- L'acronyme PHP signifie Hypertext PreProcessor
- La première version du PHP a été créé en 1994 par Rasmus Lerdorf (programmeur groenlandais et canadien)
- La version actuellement utilisée est la 7.2
- Le langage PHP s'appuie sur les langages HTML et CSS pour la mise en forme des pages

Premier exemple concret

- Affichage de la date du jour en PHP
  - Utilisation des fonctions echo (affichage) et date

```
<?php
        echo(date("d/m/Y"));
?>
```

- Utilisation du HTML/CSS pour la mise en forme

```
<?php
        echo '
        <div style="color: red;"><strong>
        '. date("d/m/Y"). '
        </strong></div>';
?>
```

Métiers associés aux langages Web

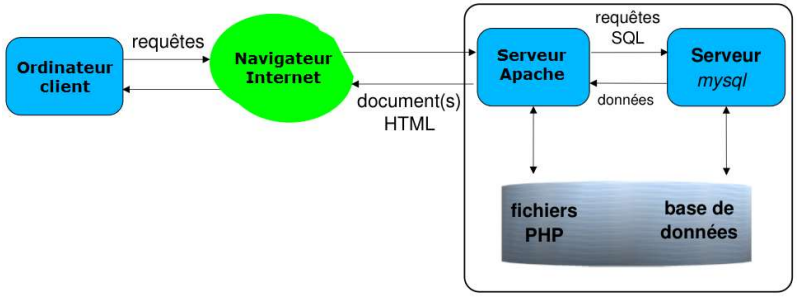
- Développeur front-end : en charge de l'aspect graphique, de l'ergonomie et de la navigation d'un site Web
  - Langages associés : HTML, CSS, JavaScript
- Développeur back-end : en charge des fonctionnalités du site Web, de l'architecture des fichiers et de l'aspect technique en général
  - Langages associés : PHP, SQL, JavaScript, XML, Python, Ruby…
- Développeur full-stack : assure les missions du front-end et du back-end
- Chef de projet digital : coordination, planification, relation avec le client, étude de faisabilité, budget, animation et organisation d'une équipe de développeurs
  - Nécessite une connaissance à la fois technique et fonctionnelle

Architecture technique

- Contrairement au HTML, CSS et JavaScript (langages dits « côté client »), un navigateur Web ne peut pas interpréter seul les lignes de code PHP
  - Un autre programme, appelé serveur Web, doit être utilisé pour convertir à la volée le PHP en HTML
    - Le serveur Web le plus utilisé avec PHP est Apache, devant NGINX
- Il n'existe pas d'étape de compilation
  - Le langage est traduit à chaque demande utilisateur
  - Un internaute ignore par conséquent tout du code initial
    - Aspirer un site Web développé en PHP n'est pas possible !
- Les fichiers PHP doivent être rangés dans un répertoire particulier pour pouvoir être interprétés par le serveur Web
    - Souvent nommé « www » sur Windows ou « htdocs » sur Mac

Architecture technique

- PHP traite des données variables
  - Ces informations sont stockées dans une base de données
    - La base de données la plus utilisée avec PHP est MySQL
  - La liaison entre le code PHP et la base de données se fait par des requêtes SQL (Structured Query Language)
  - La représentation des données est faite avec le HTML/CSS
- Les plates-formes en production reposent en majorité sur le quatuor Linux, Apache, MySQL et PHP (LAMP)
  - Cette combinaison de logiciels est très utilisée par les entreprises compte-tenu du faible coût d'acquisition et d'installation



Structure de base

- Un code PHP est inséré entre les instructions <?php et ?>
- Chaque instruction doit se terminer par un « ; »
  - En cas d'oubli, tout le script est inopérant
- Les retours à la ligne, les espaces et les tabulations n'ont aucun impact sur le résultat et sont mêmes souhaités pour la lisibilité du code
- Un commentaire PHP est introduit par :
  - // pour une seule ligne
  - /* au début, */ à la fin pour un ensemble de lignes
- Exemple de fichier PHP

```
<?php
// Affichage du mot « Bonjour » grâce à l'instruction « echo »
echo("Bonjour");
?>
```

Préparer son environnement en local

- Téléchargement d'un éditeur de texte
  - Propositions : SublimeText, Notepad++, Brackets, Coda
- Téléchargement de la suite logicielle gratuite comprenant Apache, MySQL et PHP
  - Propositions : EasyPHP DevServer, WampServer, Mamp
- Identification du répertoire utilisé par Apache pour exécuter correctement les fichiers PHP
  - « www », « eds-www » ou « htdocs » selon les configurations
  - Tous les scripts PHP devront être rangés dans ce répertoire
- Lancement des services Apache et MySQL
- Saisie d'un code PHP qui affiche « Ceci est mon premier code PHP » en italique et visualisation dans le navigateur Web de votre choix
  - URL possible : http://127.0.0.1, http://localhost, http://localhost:8888...

Préparer son environnement distant

- Téléchargement de FileZilla Client, logiciel FTP gratuit
- Récupération des paramètres SFTP communiqués
  - Nom d'hôte, identifiant, mot de passe
- Saisie de ces paramètres dans le gestionnaire de sites, disponible en haut à gauche
- Connexion au serveur distant
- Copie du script PHP dans le répertoire « public_html »
- Visualisation de sa page personnelle dans un navigateur
- Mise en favori de liens utiles
  - php.net : site de référence pour la syntaxe de toutes les fonctions PHP et des exemples d'application
  - phpfrance.com : tutoriels et forum d'entraide très actif

Introduction à l'algorithmie

- Comme tout langage de programmation, le PHP s'appuie sur des algorithmes pour répondre à un besoin
- Un algorithme est une suite d'instructions dont l'exécution permettra d'atteindre un résultat précis
  - Exemple : indiquer un itinéraire à un touriste
    - Tourner à gauche, tout droit pendant 200 m, prendre à droite, arrivée
- L'algorithmie est indépendante d'un langage donné, elle représente l'ossature d'un programme
  - La maîtrise de l'algorithmie permet ainsi de passer plus facilement d'un langage à un autre car seule la syntaxe diffère
  - La rédaction d'un algorithme sans langage associé se fait grâce à un style d'écriture appelé « pseudo-code »
- Pour définir une bonne structure logique, il faut de la méthode et de la rigueur
  - Pas besoin d'être doué(e) en mathématiques pour réussir !

Exemple d'algorithmes en pseudo-code

- Début
- A <- 1
- A <- A * 3
- A <- A + 4
- A <- (A * 7) – 1
- Afficher A
- Fin

- Début
- Si le prof est de mauvaise humeur Alors
  - Interrogation surprise
    - Si la note est inférieure à 5 Alors
      - Mieux réviser la prochaine fois
    - Sinon Si la note est supérieure à 17 Alors
      - Envisager la filière « développement » l'année prochaine
    - Finsi
- Finsi
- Fin

Variables

- Les variables sont les éléments de base de tout langage de programmation
  - Elles permettent de stocker provisoirement des valeurs
  - En fonction d'un contexte, la valeur d'une variable sera différente et pourra influer sur les instructions suivantes
- Une variable peut être représentée par une boîte
  - Pour lire le contenu de cette boîte, il faut la désigner par son nom
  - Il est parfois utile de connaître le type des informations stockées
- Exemple : création d'une variable « météo »
  - En fonction du temps qu'il fait, « météo » prendra la valeur « soleil » ou « pluie » (information textuelle)
  - Conséquences sur les instructions suivantes de l'algorithme
    - Si « météo » vaut « pluie », il faudra penser à prendre un parapluie
    - Par contre, si « météo » vaut « soleil », il faudra prendre des lunettes

Variables en PHP

- Les variables PHP sont nommées par le symbole $ suivi de lettres, de chiffres et/ou du caractère « _ »
  - Le premier caractère après le $ ne peut pas être un chiffre
  - Les noms de variables sont sensibles à la casse
    - Exemple : « $test1 » est différent de « $Test1 »
- Pour ranger une valeur à l'intérieur d'une variable, le PHP utilise le signe « = » (principe d'affectation)
  - Les valeurs texte sont encadrées par des guillemets " ou des simples quotes '
  - Exemple : $nom = "Durand"; ou $nom = 'Durand';
    - Equivalent à : nom <- "Durand" en pseudo-code
- Contrairement à d'autres langages, PHP n'impose pas de déclarer le type d'information stockée
  - L'utilisation en lecture d'une variable non initialisée ne posera pas non plus de problème

Variables en PHP

- Une variable PHP est par défaut une variable locale
  - Elle n'est lisible que dans le contexte où elle a été créée
  - Une fois la page affichée, elle cesse d'exister
- L'instruction « isset($var); » est vraie si et seulement si la variable existe
- L'instruction « unset($var); » permet de détruire la variable créée
- Si l'entité définie ne varie jamais quel que soit le contexte, il est préférable de déclarer une constante plutôt qu'une variable grâce à l'instruction define
  - Exemple de déclaration et d'affichage d'une constante

```
define("NOM","Durand");
// Par convention, les constantes sont en majuscules
echo(NOM);
// Pas de « $ » devant la constante
```

Ex 1 à 4

Types de données

- PHP dispose de 4 types de données simples
  - Les booléens
    - boolean : 1 pour TRUE ou 0 pour FALSE
  - Les entiers
    - integer
  - Les nombres à virgule
    - double
  - Les chaînes de caractères
    - string
- L'instruction « gettype($var) » renvoie le type alloué par PHP en fonction de la valeur de $var
  - Exemple : gettype("test") affichera la valeur « string »
- Le type d'une variable peut changer au cours de l'exécution
  - Exemple de modification d'un type

```
$var1 = "Ceci est le numéro"; // Type string
$var2 = 3; // Type integer
$var2 = $var1; // Type string
```

Concaténation

- Opérateur de concaténation : « . »
  - Exemple :

    ```
    $a = "Bonjour";
    $b = "à tous";
    echo($a." ".$b); // Affiche « Bonjour à tous »
    ```
- Opérateur d'affectation concaténant : « .= »
  - Exemple :

    ```
    $a = "Bonjour";
    $a .= " ";
    $a .= "à tous";
    echo($a); // Affiche « Bonjour à tous »
    ```
- Opérateurs arithmétiques concaténants : « += » , « -= », « *= », « /= »
  - Exemple :

    ```
    $a = 5; $b = 3; $b *= $a; echo($b); // Affiche 15
    ```
- La concaténation permet également d'associer des variables et des chaînes de caractères dans une même instruction
  - Cas le plus fréquent : utilisation de balises HTML, équivalentes à des chaînes de caractères
    - Exemple : echo("".$a."");

Ex 5 à 8

Opérateurs de base

- Opérateurs arithmétiques
  - Incrémentation : « $i = $i + 1; » équivaut à « $i++; »
    - Très utilisé dans les boucles : changement de valeur à chaque passage
  - Décrémentation : « $i = $i – 1; » équivaut à « $i--; »
- Opérateurs de comparaison
  - Chaque test retourne un booléen : 1 si vrai, 0 si faux
  - Egal à : « == »
    - Attention : ne pas confondre le test d'égalité avec « = » qui symbolise l'affectation d'une valeur à une variable (instruction toujours vraie)
  - Différent de : « != »
  - Egal en valeur et en type : « === »
    - Exemple : $a = 1; $b = "1";
      - L'instruction « $a == $b » est vraie (même valeur)
      - L'instruction « $a === $b » est fausse car la variable $a est de type integer et la variable $b est de type string (même valeur mais type différent)
  - Inférieur / supérieur : « < », « > », « <= », « >= »

Opérateurs logiques

- « && » ou « AND »
  - « $a && $b » renvoie TRUE (1) si $a et $b renvoient TRUE (1)
- « || » ou « OR »
  - « $a || $b » renvoie TRUE (1) si $a ou $b renvoient TRUE (1)
- « XOR »
  - « $a || $b » renvoie TRUE (1) si $a ou $b renvoient TRUE (1) mais pas les deux en même temps
- « ! »
  - « !$a » renvoie TRUE (1) si $a renvoie FALSE (0)

ET	Vrai	Faux
Vrai	Vrai	Faux
Faux	Faux	Faux

OR	Vrai	Faux
Vrai	Vrai	Vrai
Faux	Vrai	Faux

XOR	Vrai	Faux
Vrai	Faux	Vrai
Faux	Vrai	Faux

Ex 9

Manipulations d'une chaîne de caractères

- Modification de la casse d'une chaîne de caractères
  - strtolower($chaine) : renvoie la chaîne en minuscules
  - strtoupper($chaine) : renvoie la chaîne en majuscules
  - ucfirst($chaine) : renvoie la chaîne avec la première lettre en majuscule
- Remplacement d'une suite de caractères
  - str_replace($anciens,$nouveau,$chaine);
    - Exemple : str_replace("a","b","abc"); renvoie « bbc »
- Découpage d'une chaîne de caractères
  - substr($chaine,$debut,$longueur);
    - Exemple : substr("démo",1,2); renvoie « ém » (le 1er caractère correspond à 0)
- Récupération de la première position d'une suite de caractères
  - strpos($chaine,$recherche);
    - Exemple : strpos("apprendre","e"); renvoie « 4 » (position du premier e)
- Longueur d'une chaîne de caractères
  - strlen($chaine)
    - Exemple : strlen("module"); renvoie « 6 »
- Suppression des balises HTML
  - strip_tags($chaine)
    - Exemple : strip_tags("Gras") renvoie « Gras »

Ex 10 à 13

Tableaux PHP

- PHP propose 3 types de tableaux grâce à la fonction array
  - Le tableau indexé numériquement
  - Le tableau associatif
  - Le tableau multidimensionnel
- La taille d'un tableau PHP est gérée dynamiquement
  - La déclaration n'est donc pas nécessaire
- Un tableau indexé numériquement est une liste d'éléments accessibles par leur position
  - La position, également appelée index, est unique et commence à 0
  - La lecture d'un élément se fait grâce au nom du tableau, suivi de la position correspondante entre crochets
  - Exemple de déclaration d'un tableau indexé numériquement

```
$tableau = array(1,2,3,4,5,6,7,8,9);
echo($tableau[0]); // Affiche « 1 »
echo($tableau[1]); // Affiche « 2 »
```

Tableaux PHP

- Le tableau associatif associe une chaîne de caractères à un élément sous la forme clé => valeur
  - La lecture d'un élément se fait grâce au nom du tableau, suivi de la clé correspondante entre crochets
  - Exemple de déclaration d'un tableau associatif

```
$tableau = array(
"filiere" => "Informatique",
"cours" => "PHP",
"niveau" => "Débutant");
echo($tableau["cours"]); // Affiche « PHP »
```

- Le tableau multidimensionnel est un tableau de tableaux
  - Notion de matrice
  - Exemple de déclaration d'une matrice 3*2

```
$matrice = array(array(5,4), array(3,2), array(8,6));
echo($matrice[2][0]); // Affiche « 8 »
```

Manipulations de tableaux PHP

- Récupération du nombre d'éléments dans un tableau
  - count
    - Exemple : $size = count($tableau);
- Création d'un tableau à partir d'une chaîne de caractères
  - explode
    - Exemple : $string = "a,b,c,d"; $tableau = explode(",";$string);
- Transformation d'un tableau en chaîne de caractères
  - implode
    - Exemple : $tableau = array(1,2,3); $string = implode(",";$tableau);
- Test de présence d'une chaîne de caractères dans un tableau
  - in_array
    - Exemple : in_array(3,$tableau) renvoie 1 si une des valeurs du tableau est 3
- Affichage non formaté de toutes les valeurs d'un tableau
  - print_r
  - var_dump
- Tri des valeurs d'un tableau
  - sort
    - Exemple : $tableau = array("c", "z", "a")
      print_r(sort($tableau)) donne "a", "c", "z"

Ex 14

Fonctions PHP

- Il existe 2 types de fonctions en PHP
  - Les fonctions dites natives : print, echo, isset, define, include…
  - Les fonctions définies par le développeur
    - Une fonction utilisateur est un regroupement d'instructions visant à être exécutées plusieurs fois sans recopie du code
    - Une fonction utilisateur peut prendre des paramètres en entrée (arguments), renvoyer une valeur après traitement ou afficher du contenu
    - Les variables extérieures à une fonction ne sont pas disponibles dans cette fonction et vice-versa
- La déclaration d'une fonction s'appelle déclaration et peut se faire à tout moment dans le code source grâce au mot-clé function
  - Il est toutefois conseillé de regrouper toutes les fonctions dans un seul fichier et d'appeler ce dernier grâce à la fonction include
  - Syntaxe

```
function nom_fonction($var1, $var 2)
{
    // Ensemble d'instructions basées sur $var1 et $var2
}
```

Exemples de fonctions PHP

- Multiplication de 2 nombres

```
function multiplication($num1,$num2)
{
        $resultat = $num1 * $num2;

        return($resultat);
}

echo(multiplication(4,5)); // Affiche « 20 »
```

- Codification d'un pays sur 2 caractères

```
function code_pays($pays)
{
        $temp = substr($pays,0,2);
        $temp = strtoupper($temp);

        echo($temp);
}

code_pays("France"); // Affiche « FR »
```

Ex 15 à 19

Structures de contrôle

- Les structures de contrôle permettent de répéter certaines actions ou de soumettre certaines exécutions à des conditions
- Les conditions : instructions « if », « elseif », « else »
  - Syntaxe

    if (condition) { instructions } // Condition de départ

    elseif (condition) { instructions } // Ligne répétée si nécessaire

    else { instructions } // Exécution des instructions si aucun des cas précédents n'a été détecté
  - Exemple :

```
function is_mineur($age)
{
    if ($age < 18)
    {
        return(true);
    }

    return(false);
}
```

Structures de contrôle

- Les tests multiples sur une variable : instruction « switch »
  - Exemple

```
$nombre = mt_rand(0,4);
// Affecte une valeur aléatoire entre 0 et 4 à la variable $nombre
switch ($nombre) {
case 4 : echo("Excellent travail"); break;
case 3 : echo("Travail satisfaisant"); break;
case 2 : echo("Travail insuffisant"); break;
case 1 : echo("Travail médicore"); break;
default : echo("Changez de filière ☹ ! "); // Cas où $nombre = 0
}
```

- Attention, même si la correspondance a été trouvée, toutes les instructions suivantes sont exécutées
  - Il faut obligatoirement ajouter l'instruction « break; » dans chaque case pour forcer le programme à sortir du switch

Structures de boucles

- Une boucle se caractérise par l'exécution répétée d'une même série d'instructions
  - L'instruction while (« tant que »)
    - Syntaxe : while (condition) { instructions }
    - Exemple

```
$i = 1;
while ($i <= 10) {
echo($i);
$i++; }
```

  - L'instruction for (« pour »)
    - Syntaxe : for (expression1; condition; expression2) { instructions }

L'expression1 est exécutée une fois pour initialiser la boucle

La condition est testée à chaque passage

L'expression2 est exécutée à la fin d'un passage dans la boucle (en général, incrémentation d'une variable utilisée dans la condition)

- Exemple : for ($i=1; $i < 10; $i++) { echo($i."
"); }

Structures de boucles

- L'instruction foreach (« pour chaque ») permet de récupérer toutes les valeurs d'un tableau de façon séquentielle

  - Syntaxe pour un tableau indexé numériquement

```
$tableau = array(2,7,9);
foreach($tableau as $valeur)
{
   echo("<b>".$valeur."</b><br>");
}
```

  - Syntaxe pour un tableau associatif

```
$tableau = array("nom"=>"Dupont", "prenom"=>"Marc");
foreach($tableau as $key=>$valeur)
{
   echo($key." : ".$valeur."<br>");
}
```

Ex 20 à 25

Récupération des données de formulaires

- Toutes les données saisies par un utilisateur sont stockées dans un tableau associatif
  - Ce tableau existe au sein de la page indiquée dans l'attribut « action » de la balise « form »
    - Nom du tableau en méthode « GET » : « $_GET »
    - Nom du tableau en méthode « POST » : « $_POST »
    - Nom du tableau en méthode « GET » ou en méthode « POST » : « $_REQUEST »
  - Il est toujours défini mais il est vide si aucune donnée n'a été saisie
  - Clés du tableau associatif : valeurs de l'attribut « name » de chaque objet de formulaire
  - Valeurs du tableau associatif : données saisies par l'utilisateur
- Exemple :

page1.php

```
<form name="test" method="get" action="page2.php">
        <input type="text" name="saisie" />
</form>
```

page2.php

```
<?php
        echo($_GET["saisie"]);
?>
```

Ex 26 à 28

Téléchargement d'images ou de fichiers

- Les fichiers envoyés avec un formulaire se traitent différemment des autres données
  - La balise « form » doit contenir un attribut HTML supplémentaire
    - <form method="post" enctype="multipart/form-data">
  - La balise à utiliser est « input » avec l'attribut « type="file" »
    - <input type="file" name="fichier" />
  - L'accès aux fichiers téléchargés se fait ensuite grâce au tableau associatif « $_FILES »
    - $_FILES["fichier"]["name"] : nom et adresse du fichier à l'origine
    - $_FILES["fichier"]["tmp_name"] : nom et adresse temporaires
    - $_FILES["fichier"]["type"] : type du fichier
    - $_FILES["fichier"]["size"] : taille du fichier en octets
    - $_FILES["fichier"]["error"] : code erreur

Ex 29