

---

# XML & PHP

---

Mohammad Ghoniem  
Université de Bretagne Sud

---

# Plan

- XML
  - Support XML dans PHP
-

---

# XML

- Extensible Markup Language
    - Langage de balisage extensible
    - Permet de définir des documents structurés
  - Quelques caractéristiques
    - Plus flexible que HTML
    - Extensible (balises personnalisées)
    - Plus pointilleux sur le validité de forme
-

---

# Terminologie

- Un document XML est composé d'un ou plusieurs éléments :
    - Exemple : `<Titre>Le tour du monde en 80 jours</Titre>`
    - Un élément peut posséder des attributs
      - `<Prix monnaie="euro">25.43</Prix>`
    - Un élément peut être vide ou non.
      - `<Image src="arbre.gif"></Image>`
      - `<Image src="arbre.gif" />`
-

---

# Contraintes de forme

- Les valeurs d'attributs doivent être mises entre guillemets
  - Les éléments doivent avoir une balise ouvrante et fermante
  - Les balises doivent s'emboîter correctement
  - Un document XML doit contenir un élément racine englobant tous les autres éléments
-

---

# Structure d'un document XML

- Une ligne de déclaration
    - `<?xml version="1.0" standalone="no" ?>`
  - Des éléments XML
  - Des instructions de traitement
    - `<?AppCible instructions ?>`
    - Exemple : `<?php print "Date de création 01/02/2005"; ?>`
  - Des références d'entité
    - Les entités évitent de retaper de longs passages plusieurs fois
    - Par exemple : `<auteur>&mon_nom;</auteur>`
  - Des commentaires
    - `<!-- Ceci est un commentaire →`
-

---

# Définition de type de document

- Une DTD (*Document Type Definition*) fournit
    - une description des éléments que peut contenir un document XML de ce type
    - des attributs des éléments
    - des entités XML
    - elle peut être interne ou externe
    - Exemple :

```
<!DOCTYPE nomelementracine [  
  
...  
>
```
-

---

# Déclaration de type d'élément

- Elle indique si l'élément déclaré contient d'autres éléments, du texte ou s'il est vide.
  - Elle indique la nature des éléments (obligatoire ou facultative) et leur fréquence dans le document.
  - Les opérateurs \*, + et ? Permettent d'indiquer le nombre d'instances admises de l'élément
-

---

# Exemples

- `<!ELEMENT titre (#PCDATA)>`  
→ cet élément contient du texte
  - `<!ELEMENT image EMPTY>`  
→ cet élément est vide
  - `<!ELEMENT livre (titre, auteurs, isbn, prix)>`  
→ cet élément contient uniquement d'autres éléments
  - `<!ELEMENT nom_element ANY>`  
→ cet élément peut contenir du texte ou d'autres éléments
  - `<!ELEMENT auteurs (auteur+)>`
  - `<!ELEMENT tdm (chapitres, annexes?)>`
-

---

# Déclaration de liste d'attributs

- Un élément peut admettre un ou plusieurs attributs qui peuvent être obligatoires, facultatifs ou implicites.
    - ❑ REQUIRED (obligatoire)
    - ❑ IMPLIED (valeur par défaut si non spécifiée)
    - ❑ FIXED (une seule valeur pour toutes les instances)
-

---

# Types d'attributs

- <!ATTLIST prix devise CDATA #REQUIRED>
- <!ATTLIST auteur genre (masculin|feminin) #IMPLIED>
- <!ATTLIST employe
  - idemploye ID #REQUIRED
  - idmanager IDREF #IMPLIED
  - idsubbrodones IDREFS #IMPLIED>



---

# Déclaration d'entités

- `<!ENTITY mon_nom "Pierre Richard">`
  - `<!ENTITY description1 SYSTEM "description.xml">`
  - `<!ENTITY description1 SYSTEM "description.xml"  
PUBLIC "http://www.domaine.com/description1.xml">`
-

---

# Support XML de PHP

- PHP inclut des fonctions permettant d'analyser un document XML.
  - L'analyse de documents se base sur la définition de fonctions de rappels (callbacks).
  - PHP ne se charge pas de la validation des documents XML.
  - Le support XML est fourni par un module facultatif de PHP.
-

---

# L'API XML de PHP

- Étapes principales d'un script d'analyse
    - Créer un analyseur XML
    - Enregistrer les fonctions de rappel auprès de l'analyseur
    - Lire les données à partir du fichier XML et les transmettre à l'analyseur
    - Libérer la mémoire occupée par l'analyseur
-

---

# Les fonctions de l'API XML

- `int xml_parser_create(string [param_codage])`
  - `int xml_set_element_handler(int analyseur, string gestionnaire_debut, string gestionnaire_fin)`
  - `int xml_set_character_data_handler(int analyseur, string gestionnaire)`
  - `int xml_set_processing_instruction_handler (int analyseur, string cible, string données)`
  - ...
  - `int xml_parse(int analyseur, string données, int [taille]);`
  - `int xml_parser_free(int analyseur)`
-

---

# Exemple 1 / 2

```
// The handler for element opening tags
function startElementHandler($parser, $name, $attribs) {
    echo("<$name<BR>");
}
// The handler for element closing tags
function endElementHandler($parser, $name) {
    echo("</$name<BR>");
}

// The handler for character data
function cdataHandler($parser, $data) {
    echo("$data<BR>");
}

// Now we create the parser
$parser=xml_parser_create();
```

---

---

# Exemple 2/2

```
// Register the start and end element handlers
xml_set_element_handler($parser, "startElementHandler", "endElementHandler");

// Register the character data parser
xml_set_character_data_handler($parser, "cdataHandler");

// Open the XML file
$file="exemple1.xml";
if (!$fp = fopen($file, "r")) {
    die("could not open $file for reading");
}

// Read chunks of 4K from the file, and pass it to the parser
while ($data = fread($fp, 4096)) {
    if (!xml_parse($parser, $data, feof($fp))) {
        die(sprintf("XML error %d %d", xml_get_current_line_number($parser),
            xml_get_current_column_number($parser)));
    }
}
```

---

---

# Fonctions utilitaires

- `xml_get_error_code(int analyseur)`
  - `xml_error_string(int code)`
  - `xml_get_current_line_number(int analyseur)`
  - `xml_get_current_column_number(int analyseur)`
  - `utf8_decode(string données)`
  - `Utf8_encode(string données)`
-

---

# Applications XML en PHP

- Applications web d'entreprise
    - XML = format d'échange de données entre applications métier
    - Une DTD décrit les commandes, transactions, stocks, factures etc.
  - Recherches intelligentes sur les éléments et attributs des documents XML définis dans une DTD
  - Conversion XML → HTML
  - Vues différentes des mêmes données (filtrage/modification de certains nœuds etc.)
-