PHP – XML

Tartalomjegyzék

PHP – DINAMIKUS MULTIMÉDIÁS TARTALOM 1

TARTALOMJEGYZÉK 1
EDDIG VOLT 1
MAI ANYAG 1
XML feldolgozása általában 2
A PHP és XML története 2
SAX (Simple API for XML) 2
1. példa: 2
2. példa: 4
3. példa: 5
SimpleXML 7
1. példa 7
2. példa 7
3. példa 8
DOM 9
1. példa 9
2. példa 9
3. példa 10
4. példa 11
XSLT 12
1. példa: 12
Az alkalmazás fejlesztése – Szűrés 13
1. példa: SAX 13
2. példa: SimpleXML 15
3. példa: DOM 17
Az alkalmazás bővítése – Insert, Delete, Update – DOM 19
edit.php 19
insert_dom.php 21
update_dom.php 22
delete_dom.php 23
MA MEGTANULTUK 23
EZUTÁN JÖN 24

Eddig volt

Az előző órákon megismerkedtünk

- multimédiás tartalmaknak adatbázisban való tárolásával és keresésével,
- valamint metainformációknak XML-ben történő leírásával,
- ezekben való kereséssel (XPATH)
- és ezeknek módosításával (XUPDATE)

Mai anyag

A mai órán azt nézzük meg, hogy PHP-ban milyen lehetőségeink vannak XML file-okat feldolgozni. Szó lesz:

- A különböző feldolgozási filozófiákról és lehetőségekről általában és PHP5-ben
  - SAX
  - SimpleXML
  - DOM
  - XSLT
- Egy példa XML dokumentum megjelenítéséről a fenti módszerekkel
- Az adatok szűréséről

- XML dokumentum módosításáról

XML feldolgozása általában

Az XML feldolgozásának alapvetően kétféle filozófiája van. Az egyik az XML dokumentumon szekvenciálisan végighalad az elejétől a végéig, és lehetőséget teremt bizonyos műveletek elvégzésére kezdőtag-ek, zárótag-ek, adatok, megjegyzések, stb. esetén. Ebben az esetben nincsen szükség az egész XML dokumentumot beolvasni a memóriába, elég csak egy kis szeletét, viszonylag gyors módszer, viszont jellegéből adódóan számos, bonyolultabb esetben nem lehet használni. Nincs visszalépés, a feldolgozó program az XML dokumentum struktúrájához igazodik, így elég speciális.
A másik megközelítés az XML dokumentum hierarchikus mivoltát használja ki, és egy fát épít fel belőle. Ezáltal lehetővé válik a struktúra bármelyik részének elérése, általánosabb feldolgozó rutinok írhatók. Hátránya viszont, hogy az egész dokumentum beolvasásra kerül a memóriában, így nagy XML-ek esetében lassú és erőforrás-igényes lehet egy ilyen feldolgozás.

A PHP és XML története

- SAX már volt a PHP3-astól.
- PHP4-ben emellé jött a DOM és az XSLT, eléggé kiforratlanul.
- PHP5-ben mindegyiket teljesen újraírták, és új lehetőségekkel bővítették (SimpleXML).
- W3C standard
- Interoperabilitás az egyes bővítmények között.

SAX (Simple API for XML)

Az első filozófia szerint működik, azaz szekvenciálisan halad végig az XML dokumentumokon, és az egyes elemek elérésekor egy eseményt generál, amit a megfelelő eseménykezelővel dolgozhatunk fel. Az események a következők:

- elem kezdete
- elem vége
- karakteres adat
- feldolgozási utasítás (processing instruction)
- külső dokumentumra való hivatkozás

A feldolgozás módja: egy parser létrehozása (*xml_parser_create* metódus), eseménykezelők hozzárendelése (*xml_set_element_handler*, *xml_set_character_data_handler*, *xml_set_default_handler*, stb.) és megírása, majd a dokumentum beolvasása, és feldolgozása (*xml_parse*).
Nagyon kicsi a memóriaigénye, gyors és évek óta stabil, viszont nekünk kell felépíteni a feldolgozó modellt.

1. példa:

A példa XML dokumentum egyszerű kiírása a képernyőre:

```
<?php
header("Content-Type: text/html;charset=UTF-8");
?>

<html>
   <head>
      <title>SAX 1.</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
```

```
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
   <body>
<?php

$parser_object = xml_parser_create("UTF-8");
xml_parser_set_option($parser_object, XML_OPTION_TARGET_ENCODING, "UTF-8");

//Ha azt akarjuk, hogy ne legyenek nagy betusek az elemek
//xml_parser_set_option($parser_object, XML_OPTION_CASE_FOLDING, false);

xml_set_element_handler($parser_object,"startElement","endElement");
xml_set_character_data_handler($parser_object, "contentHandler");
xml_set_default_handler ($parser_object, "DefaultHandler");

//kezdo tagek esemenykezeloje
function startElement($parser_object, $elementname, $attribute) {
  print "\n<ul>";
  print "$elementname ";
  foreach ($attribute as $key => $value)
  {
    print "$key => $value; ";
  }
}
//zarotag-ek esemenykezeloje
function endElement($parser_object, $elementname) {
  print "</ul>\n";
}

//szoveg esemenykezeloje
function contentHandler($parser_object,$data)
{
  if (trim($data) != "") {
      //print utf8_decode("'<font color='green'>$data</font>'");
      print "'<font color='green'>$data</font>'";
  }
}

//default esemenykezelo
function DefaultHandler($parser_object, $data) {
    print "<font color='blue'>$data</font>";
}

$fp = fopen("album1.xml", "rb");
while ( $data = fread($fp, 4096) ) {
    if ( !xml_parse($parser_object, $data, feof($fp)) ) {
      die (sprintf("<br>XML error: %s %d sor, %d oszlop",
           xml_error_string(xml_get_error_code($parser_object)),
           xml_get_current_line_number($parser_object),
           xml_get_current_column_number($parser_object)));
    }
}
xml_parser_free($parser_object);

?>

   </body>
</html>
```

2. példa:

Az album képeinek megjelenítése. Hibakezelés: *xml_error_string*, *xml_get_error_code*.

```
<?php
   //Ez azert kell, mert a META tag nem csinalta ezt
   header("Content-Type: text/html;charset=UTF-8");
?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>

<body>
<?php

$kepenbelul = false;
$tag = "";
$fileNev = "";
$mimeType = "";
$datum = "";
$szerzo = "";
$leiras = "";

function ElemIras() {
   printf("<img src='%s'><br>\n", $GLOBALS['fileNev']);
   printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
               <li><b>Mimetype:</b> %s</li>
               <li><b>DÃ¡tum:</b> %s</li>
               <li><b>LeÃrÃ¡s:</b> %s</li>
          </ul>
          <hr>",
            $GLOBALS['szerzo'],
            $GLOBALS['mimeType'],
            $GLOBALS['datum'],
            $GLOBALS['leiras']);
}

function startElement($parser, $tagName, $attrs) {
   if ($GLOBALS['kepenbelul']) {
            $GLOBALS['tag'] = $tagName;
      } elseif ($tagName == "KEP") {
            $GLOBALS['kepenbelul'] = true;
      }
}

function endElement($parser, $tagName) {
   if ($tagName == "KEP") {
      ElemIras();
      $GLOBALS['fileNev'] = "";
      $GLOBALS['mimeType'] = "";
      $GLOBALS['datum'] = "";
      $GLOBALS['szerzo'] = "";
      $GLOBALS['leiras'] = "";
            $GLOBALS['kepenbelul'] = false;
      }
}

function characterData($parser, $data) {
```

```
   if ($GLOBALS['kepenbelul']) {
      switch ($GLOBALS['tag']) {
            case "FILENEV":
                  $GLOBALS['fileNev'] .= $data;
                  break;
            case "MIMETYPE":
                  $GLOBALS['mimeType'] .= $data;
                  break;
            case "DATUM":
                  $GLOBALS['datum'] .= $data;
                  break;
            case "SZERZO":
                  $GLOBALS['szerzo'] .= $data;
                  break;
         case "LEIRAS":
                  $GLOBALS['leiras'] .= $data;
                  break;
      }
      }
}

$xml_parser = xml_parser_create("UTF-8");
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
$fp = fopen("album1.xml","rb")
      or die("Hiba a file megnyitasa kozben.");
while ($data = fread($fp, 4096)) {
      xml_parse($xml_parser, $data, feof($fp))
            or die (sprintf("<br>XML error: %s %d sor, %d oszlop",
           xml_error_string(xml_get_error_code($parser_object)),
           xml_get_current_line_number($parser_object),
           xml_get_current_column_number($parser_object)));
}
fclose($fp);
xml_parser_free($xml_parser);

?>
</body>
</html>
```

3. példa:

Ha valaki a feldolgozó részt külön osztályba szeretné rakni, akkor a függvényeket az osztály metódusaiként kell elhelyezni, és jelezni kell az XML parsernek, hogy az eseménykezelők mely objektum metódusai (*xml_set_object*(xml_parser, parser_objektum)):

```
<?php
   //Ez azert kell, mert a META tag nem csinalta ezt
   header("Content-Type: text/html;charset=UTF-8");
?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>

<body>
<?php

class AlbumParser {
```

```
var $kepenbelul = false;
var $tag = "";
var $fileNev = "";
var $mimeType = "";
var $datum = "";
var $szerzo = "";
var $leiras = "";

function ElemIras() {
   printf("<img src='%s'><br>\n", $this->fileNev);
   printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
               <li><b>Mimetype:</b> %s</li>
               <li><b>DÃ¡tum:</b> %s</li>
               <li><b>LeÃrÃ¡s:</b> %s</li>
          </ul>
          <hr>",
            $this->szerzo,
            $this->mimeType,
            $this->datum,
            $this->leiras);
}

function startElement($parser, $tagName, $attrs) {
   if ($this->kepenbelul) {
         $this->tag = $tagName;
   } elseif ($tagName == "KEP") {
         $this->kepenbelul = true;
   }
}

function endElement($parser, $tagName) {
   if ($tagName == "KEP") {
      $this->ElemIras();
      $this->fileNev = "";
      $this->mimeType = "";
      $this->datum = "";
      $this->szerzo = "";
      $this->leiras = "";
         $this->kepenbelul = false;
   }
}

function characterData($parser, $data) {
   if ($this->kepenbelul) {
         switch ($this->tag) {
               case "FILENEV":
               $this->fileNev .= $data;
               break;
               case "MIMETYPE":
               $this->mimeType .= $data;
               break;
               case "DATUM":
               $this->datum .= $data;
               break;
         case "SZERZO":
               $this->szerzo .= $data;
               break;
         case "LEIRAS":
               $this->leiras .= $data;
               break;
```

```
            }
      }
   }
}

$xml_parser = xml_parser_create("UTF-8");
$album_parser = new AlbumParser();
xml_set_object($xml_parser, $album_parser);
xml_set_element_handler($xml_parser, "startElement", "endElement");
xml_set_character_data_handler($xml_parser, "characterData");
$fp = fopen("album1.xml","rb")
      or die("Hiba a file megnyitasa kozben.");
while ($data = fread($fp, 4096)) {
      xml_parse($xml_parser, $data, feof($fp))
            or die (sprintf("<br>XML error: %s %d sor, %d oszlop",
           xml_error_string(xml_get_error_code($parser_object)),
           xml_get_current_line_number($parser_object),
           xml_get_current_column_number($parser_object)));
}
fclose($fp);
xml_parser_free($xml_parser);

?>
</body>
</html>
```

SimpleXML

Egy új lehetőség PHP-ben, ami nagyon egyszerűvé teszi XML dokumentumok feldolgozását. Az XML dokumentum teljes egészében beolvasásra kerül, és belőle egy objektumhierarchia épül fel, ezen a szokásos iterátorokkal és tömbműveletekkel haladhatunk végig. Ez tehát egy objektumorientált megközelítése az XML feldolgozásának. A DOMhoz képest kevesebbet tud, de általában bőven elég arra, ami általában XML feldolgozásához kell. Van benne XPath támogatás. Valószínűleg kicsit kevesebb memóriát foglal, mint a DOM. A dokumentum meglévő elemei módosíthatóak.
A feldolgozáshoz csak kevés utasítás szükséges. Be kell tölteni a dokumentumot (*simplexml_load_file*), és onnantól az objektumhierarchiában mozogni. Az egyes objektumoknak négy metódusa van: asXML (az adott node tartalmát jól formázott XML-ként adja vissza), attribute (az adott elem attribútumait lehet elérni), children (adott elem gyerekeit lehet elérni), xpath (a megadott XPath kifejezést futtatja le).

1. példa

```
<?php

header("Content-Type:text/html;charset=UTF-8");

$xml = simplexml_load_file("album1.xml");

print_r($xml);
echo "<hr>";

echo $xml->kepek[0]->kep[0]->fileNev."<hr>";

?>
```

2. példa

Az album megjelenítése:

```
<?php
   //Ez azert kell, mert a META tag nem csinalta ezt
   header("Content-Type: text/html;charset=UTF-8");
?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<?php

//xml betoltese
$album = simplexml_load_file('album1.xml');

//vegigmenni a kepeken
foreach ($album->kepek[0]->kep as $kep) {
        printf("<img src='%s'><br>\n", $kep->fileNev);
        printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
                    <li><b>Mimetype:</b> %s</li>
                    <li><b>DÃ¡tum:</b> %s</li>
                    <li><b>LeÃrÃ¡s:</b> %s</li>
                </ul>
                <hr>",
                  $kep->szerzo,
                  $kep->mimeType,
                  $kep->datum,
                  $kep->leiras);
}

?>
</body>
</html>
```

3. példa

XPath használata:

```
<?php
   //Ez azert kell, mert a META tag nem csinalta ezt
   header("Content-Type: text/html;charset=UTF-8");
?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<pre>
<?php

//xml betoltese
$album = simplexml_load_file('album1.xml');

//XPath megadasa
$ered = $album->xpath("//kep[@azon<4]");

foreach ($ered as $node) {
```

```
    echo htmlspecialchars($node->asXML())."\n";
}

?>
</pre>
</body>
</html>
```

DOM

A DOM bővítmény teljesen újra lett írva a PHP5-ben, visszafele nem kompatibilis, szabványosabb és majdnem teljesen implementált a szabványnak megfelelően.
A DOM az XML dokumentum fa reprezentációja, azaz minden node-nak egy szülője van, 0 vagy több gyereke, és lehetnek testvérei. Minden node: az elemek, az attribútumok, a szövegek.
A következő fontosabb osztályok vannak: DomDocument, DomElement, DomAttribute, mindegyik a DomNode-ból származik.
DomNode: A fa struktúrában a node-ok között a megfelelő tulajdonságok segítségével mehetünk egy szinttel lejjebb, a következő testvérre, vagy egy szinttel feljebb. Az ide tartozó tulajdonságok: firstChild, lastChild, nextSibling, previousSibling, parentNode. Egy node nevét, értékét, típusát a nodeName, nodeValue és nodeType tulajdonságok adják meg. A metódusok segítségével egy node illeszthető be vagy távolítható el a struktúrába/ból. Fontosabb metódusok: appendChild, replaceChild, removeChild.
A DomDocumenten keresztül van lehetőségünk XML betöltésére, mentésére, validálására. Ezzel hozunk létre új elemeket (createElement), attribútumokat (createAttribute), szövegeket (createTextNode). Ezzel érhetjük el a struktúra bizonyos részeit id vagy név alapján: getElementById, getElementsByTagName.
Egy DomElement alapvetően az attribútumait tudja elérni, változtatni: getAttribute, setAttribute.
Ezeken keresztül tehát lehetőségünk van az XML dokumentum tetszőleges részét elérni és feldolgozni, és XML dokumentumot felépíteni változtatni.
A DOM támogatja az XPath-t, a validálást és az XSLT feldolgozást is.

1. példa

Az XML dokumentum kiírása a képernyőre

```
<?php

//makes a new DomDocument Object with version 1.0
$dom = new DomDocument();

//loads an XML Document from the filesystem
$dom->load("album1.xml");

//sets output encoding
$dom->encoding="ISO-8859-2";

//returns the XML Document as a string
print "<pre>".htmlspecialchars($dom->saveXML())."</pre>";

?>
```

2. példa

Az egyes részek elérése

```
<?php
```

```
header("Content-Type: text/plain;charset=UTF-8");

$dom = new DomDocument();
$dom->load("album1.xml");
//$dom->encoding="ISO-8859-2";

//A gyokerelem elerese
$root = $dom->documentElement;

print "--- get all childnodes of the root nodes --- \n";
foreach ($root->childNodes as $child) {
    print $child->nodeName ."\n";
}

print "--- get firstChild of root Node ---\n";
print $root->firstChild->nodeName ."\n";

print "--- get lastChild of rootNode --- \n";
print $root->lastChild->nodeName ."\n";

print "--- get nextSibling of firstChild of rootNode --- \n";
print $root->firstChild->nextSibling->nodeName ."\n";

print "--- get parentNode of firstChild of rootNode --- \n";
print $root->firstChild->parentNode->nodeName ."\n";

print "--- get all attribute of the first book node --- \n";
$kepek = $root->firstChild->nextSibling->nextSibling->nextSibling-
>nextSibling->nextSibling;
foreach ($kepek->childNodes as $child) {
    print $child->nodeName ." => " . $child->nodeValue ."\n";
}

print "--- get value of attribute id ---\n";
print $kepek->firstChild->nextSibling->getAttribute("azon")."\n";

print "--- get ownerDocument of rootNode ---\n";
var_dump( $root->ownerDocument);

print "--- get getElementsByTagName(kep) ---\n";
foreach ($dom->getElementsByTagName("kep") as $node) {
    print htmlspecialchars($node->nodeValue)."\n";
}

?>
```

3. példa

XPath használata

```
<?php

header("Content-Type: text/plain; charset=UTF-8");

//load xml document
$dom = new DomDocument();
$dom->load("album1.xml");

//create DOMXPath object
$xpath = new Domxpath($dom);

//osszes szerzo
```

```
$result = $xpath->query("//szerzo");
//elso megtalalt szerzo
$elsoSzerzo = $result->item(0);
print $elsoSzerzo->nodeValue."\n\n";

//$query the document
$result = $xpath->query("//kep[contains(szerzo,'Péter')]");
//loop through the result (it's a NamedNodemap..)
foreach ($result as $fileNev) {
    print $fileNev->nodeValue. "\n";
}

?>
```

4. példa

Az album megjelenítése

```
<?php
   //Ez azert kell, mert a META tag nem csinalta ezt
   header("Content-Type: text/html;charset=UTF-8");
?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<?php

$dom = new DomDocument();
$dom->load("album1.xml");
$album = $dom->documentElement;
$kepek = $album->firstChild->nextSibling->nextSibling->nextSibling-
>nextSibling->nextSibling->childNodes;
//$kepek = $dom->getElementsByTagName("kep");

foreach($kepek as $kep) {
   if ($kep instanceof domElement) {
      ElemIras($kep);
   }
}

function ElemIras($kep) {
   $fileNev = "";
   $mimeType = "";
   $datum = "";
   $szerzo = "";
   $leiras = "";

   foreach($kep->childNodes as $child) {
      if ($child instanceof domElement) {
         switch ($child->tagName) {
            case "fileNev": {
               $fileNev = $child->nodeValue;
               break;
            }
            case "szerzo": {
               $szerzo = $child->nodeValue;
               break;
```

```
            }
            case "mimeType": {
               $mimeType = $child->nodeValue;
               break;
            }
            case "datum": {
               $datum = $child->nodeValue;
               break;
            }
            case "leiras": {
               $leiras = $child->nodeValue;
               break;
            }
         }
      }
   }
   /*
     Itt lehetne szurni az egyes adatokra: fileNev, szerzo, stb.,
     es csak akkor kiiratni.
   */
   printf("<img src='%s'><br>\n", $fileNev);
         printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
                     <li><b>Mimetype:</b> %s</li>
                     <li><b>DÃ¡tum:</b> %s</li>
                     <li><b>LeÃrÃ¡s:</b> %s</li>
                </ul>
                <hr>",
                  $szerzo,
                  $mimeType,
                  $datum,
                  $leiras);
}

?>
</body>
</html>
```

XSLT

Egy funkcionális nyelv XML dokumentumok transzformálására. Bővebben: www.w3schools.com
PHP5-ben nagyon gyors lett, viszonylag kevés PHP utasítás kell hozzá. DomDocumentként kell betölteni, létre kell hozni egy *XSLTProcessor*t, ebbe kell importálni az XSL dokumentumunkat (*importStyleSheet*), és ezzel kell feldolgozni az XML dokumentumot (*transformToXML*, *transformToDoc*, *transformToUri*).
Lehetőség van PHP függvények hívására XSLT-ből:

```
<xsl:value-of select="php:function('fuggvenynev', XPath)"/>
```

1. példa:

Az album megjelenítése:

```
<?php

header("Content-Type: text/html;charset=UTF-8");

// Load the XML source
$xml = new DOMDocument;
$xml->load('album1.xml');

$xsl = new DOMDocument;
```

```
$xsl->load('album.xsl');

// Configure the transformer
$proc = new XSLTProcessor;
$proc->importStyleSheet($xsl); // attach the xsl rules

echo $proc->transformToXML($xml);

?>
```

Az alkalmazás fejlesztése – Szűrés

1. példa: SAX

```
<?php

header("Content-Type: text/html;charset=UTF-8");

$mimeType="";
$szerzo="";
$leiras="";
$datum="";
$szelesseg="";
$magassag="";

if (isset($_POST["szuresgomb"])) {
   $mimeType=$_POST["mimeType"];
   $szerzo=$_POST["szerzo"];
   $leiras=$_POST["leiras"];
   $datum=$_POST["datum"];
   $szelesseg=$_POST["szelesseg"];
   $magassag=$_POST["magassag"];
}

?>
<html>
   <head>
      <title>Album</title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>

<body>

<h2>SzÅ±rÃ©s</h2>
<form name="form_szures" action="<?=$_SERVER["PHP_SELF"]?>" method="post">
<table>
   <tr><td>Mimetype:</td>  <td><input type="text" name="mimeType"
value="<?=$mimeType?>"></td></tr>
   <tr><td>SzerzÅ‘:</td>    <td><input type="text" name="szerzo"
value="<?=$szerzo?>"></td></tr>
   <tr><td>LeÃrÃ¡s:</td>    <td><input type="text" name="leiras"
value="<?=$leiras?>"></td></tr>
   <tr><td>DÃ¡tum:</td>     <td><input type="text" name="datum"
value="<?=$datum?>"></td></tr>
   <tr><td>SzÃ©lessÃ©g:</td> <td><input type="text" name="szelesseg"
value="<?=$szelesseg?>"></td></tr>
   <tr><td>MagassÃ¡g:</td>  <td><input type="text" name="magassag"
value="<?=$magassag?>"></td></tr>
```

```
   <tr><td colspan="2"><input type="submit" name="szuresgomb"
value="SzÅ±res"></td></tr>
</table>
</form>
<?php

class AlbumParser {

   var $kepenbelul = false;
   var $tag = "";
   var $fileNev = "";
   var $mimeType = "";
   var $datum = "";
   var $szerzo = "";
   var $leiras = "";

   function ElemIras() {
      printf("<img src='%s'><br>\n", $this->fileNev);
      printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
                  <li><b>Mimetype:</b> %s</li>
                  <li><b>DÃ¡tum:</b> %s</li>
                  <li><b>LeÃrÃ¡s:</b> %s</li>
             </ul>
             <hr>",
               $this->szerzo,
               $this->mimeType,
               $this->datum,
               $this->leiras);
   }

   function startElement($parser, $tagName, $attrs) {
      if ($this->kepenbelul) {
            $this->tag = $tagName;
      } elseif ($tagName == "KEP") {
            $this->kepenbelul = true;
      }
   }

   function endElement($parser, $tagName) {
      if ($tagName == "KEP") {
         if ( ($GLOBALS["mimeType"]=="" || stripos($this->mimeType,
$GLOBALS["mimeType"]) !== false) &&
              ($GLOBALS["szerzo"]=="" || stripos($this->szerzo,
$GLOBALS["szerzo"]) !== false) &&
              ($GLOBALS["leiras"]=="" || stripos($this->leiras,
$GLOBALS["leiras"]) !== false) &&
              ($GLOBALS["datum"]=="" || stripos($this->datum,
$GLOBALS["datum"]) !== false) &&
              ($GLOBALS["szelesseg"]=="" || stripos($this->szelesseg,
$GLOBALS["szelesseg"]) !== false) &&
              ($GLOBALS["magassag"]=="" || stripos($this->magassag,
$GLOBALS["magassag"]) !== false) ) {

            $this->ElemIras();
         }
         $this->fileNev = "";
         $this->mimeType = "";
         $this->datum = "";
         $this->szerzo = "";
         $this->leiras = "";
            $this->kepenbelul = false;
```

```
      }
   }

   function characterData($parser, $data) {
      if ($this->kepenbelul) {
            switch ($this->tag) {
                  case "FILENEV":
                  $this->fileNev .= $data;
                  break;
                  case "MIMETYPE":
                  $this->mimeType .= $data;
                  break;
                  case "DATUM":
                  $this->datum .= $data;
                  break;
            case "SZERZO":
                  $this->szerzo .= $data;
                  break;
             case "LEIRAS":
                  $this->leiras .= $data;
                  break;
            }
      }
   }
}

if (isset($_POST["szuresgomb"])) {
   $xml_parser = xml_parser_create("UTF-8");
   $album_parser = new AlbumParser();
   xml_set_object($xml_parser, $album_parser);
   xml_set_element_handler($xml_parser, "startElement", "endElement");
   xml_set_character_data_handler($xml_parser, "characterData");
   $fp = fopen("album1.xml","rb")
      or die("Hiba a file megnyitasa kozben.");
   while ($data = fread($fp, 4096)) {
      xml_parse($xml_parser, $data, feof($fp))
            or die (sprintf("<br>XML error: %s %d sor, %d oszlop",
              xml_error_string(xml_get_error_code($parser_object)),
              xml_get_current_line_number($parser_object),
              xml_get_current_column_number($parser_object)));
   }
   fclose($fp);
   xml_parser_free($xml_parser);
}

?>
</body>
</html>
```

2. példa: SimpleXML

```
<?php

header("Content-Type: text/html;charset=UTF-8");

$mimeType="";
$szerzo="";
$leiras="";
$datum="";
$szelesseg="";
$magassag="";
```

```
if (isset($_POST["szuresgomb"])) {
   $mimeType=$_POST["mimeType"];
   $szerzo=$_POST["szerzo"];
   $leiras=$_POST["leiras"];
   $datum=$_POST["datum"];
   $szelesseg=$_POST["szelesseg"];
   $magassag=$_POST["magassag"];

   $xpath="/album/kepek/kep[contains(mimeType,'$mimeType') and
                           contains(szerzo,'$szerzo') and
                           contains(leiras,'$leiras') and
                           contains(datum,'$datum') and
                           contains(szelesseg,'$szelesseg') and
                           contains(magassag,'$magassag')]";
   echo $xpath;
}

?>
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>

<h2>SzÅ±rÃ©s</h2>
<form name="form_szures" action="<?=$_SERVER["PHP_SELF"]?>" method="post">
<table>
   <tr><td>Mimetype:</td>  <td><input type="text" name="mimeType"
value="<?=$mimeType?>"></td></tr>
   <tr><td>SzerzÅ‘:</td>    <td><input type="text" name="szerzo"
value="<?=$szerzo?>"></td></tr>
   <tr><td>LeÃrÃ¡s:</td>    <td><input type="text" name="leiras"
value="<?=$leiras?>"></td></tr>
   <tr><td>DÃ¡tum:</td>     <td><input type="text" name="datum"
value="<?=$datum?>"></td></tr>
   <tr><td>SzÃ©lessÃ©g:</td> <td><input type="text" name="szelesseg"
value="<?=$szelesseg?>"></td></tr>
   <tr><td>MagassÃ¡g:</td>  <td><input type="text" name="magassag"
value="<?=$magassag?>"></td></tr>
   <tr><td colspan="2"><input type="submit" name="szuresgomb"
value="SzÅ±res"></td></tr>
</table>
</form>
<?php

if (isset($_POST["szuresgomb"])) {

   //xml betoltese
   $album = simplexml_load_file('album1.xml');

   //XPath megadasa
   $ered = $album->xpath($xpath);

   foreach ($ered as $kep) {
           printf("<img src='%s'><br>\n", $kep->fileNev);
           printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
                       <li><b>Mimetype:</b> %s</li>
                       <li><b>DÃ¡tum:</b> %s</li>
```

```
                    <li><b>LeÃ–rÃ¡s:</b> %s</li>
                </ul>
                <hr>",
                  $kep->szerzo,
                  $kep->mimeType,
                  $kep->datum,
                  $kep->leiras);
   }
}
?>
</body>
</html>
```

3. példa: DOM

```
<?php

header("Content-Type: text/html;charset=UTF-8");

$mimeType="";
$szerzo="";
$leiras="";
$datum="";
$szelesseg="";
$magassag="";

if (isset($_POST["szuresgomb"])) {
   $mimeType=$_POST["mimeType"];
   $szerzo=$_POST["szerzo"];
   $leiras=$_POST["leiras"];
   $datum=$_POST["datum"];
   $szelesseg=$_POST["szelesseg"];
   $magassag=$_POST["magassag"];

   $xpath="/album/kepek/kep[contains(mimeType,'$mimeType') and
                            contains(szerzo,'$szerzo') and
                            contains(leiras,'$leiras') and
                            contains(datum,'$datum') and
                            contains(szelesseg,'$szelesseg') and
                            contains(magassag,'$magassag')]";
   echo $xpath;
}

?>
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>

<h2>SzÅ±rÃ©s</h2>
<form name="form_szures" action="<?=$_SERVER["PHP_SELF"]?>" method="post">
<table>
   <tr><td>Mimetype:</td>  <td><input type="text" name="mimeType"
value="<?=$mimeType?>"></td></tr>
   <tr><td>SzerzÅ‘:</td>    <td><input type="text" name="szerzo"
value="<?=$szerzo?>"></td></tr>
   <tr><td>LeÃ–rÃ¡s:</td>    <td><input type="text" name="leiras"
value="<?=$leiras?>"></td></tr>
```

```
   <tr><td>DÃ¡tum:</td>     <td><input type="text" name="datum"
value="<?=$datum?>"></td></tr>
   <tr><td>SzÃ©lessÃ©g:</td> <td><input type="text" name="szelesseg"
value="<?=$szelesseg?>"></td></tr>
   <tr><td>MagassÃ¡g:</td>  <td><input type="text" name="magassag"
value="<?=$magassag?>"></td></tr>
   <tr><td colspan="2"><input type="submit" name="szuresgomb"
value="SzÅ±res"></td></tr>
   <tr><td colspan="2"><a href="edit.php">Uj felvitele</a></td></tr>
</table>
</form>
<hr>
<?php

if (isset($_POST["szuresgomb"])) {
   //load xml document
   $dom = new DomDocument();
   $dom->load("album1.xml");
   //create DOMXPath object
   $xp = new Domxpath($dom);
   //$query the document
   $result = $xp->query($xpath);
   //loop through the result (it's a NamedNodemap..)
   foreach ($result as $kep) {
       ElemIras($kep);
   }
}

function ElemIras($kep) {
   $fileNev = "";
   $mimeType = "";
   $datum = "";
   $szerzo = "";
   $leiras = "";
   $azon = $kep->getAttribute("azon");

   foreach($kep->childNodes as $child) {
      if ($child instanceof domElement) {
         switch ($child->tagName) {
            case "fileNev": {
               $fileNev = $child->nodeValue;
               break;
            }
            case "szerzo": {
               $szerzo = $child->nodeValue;
               break;
            }
            case "mimeType": {
               $mimeType = $child->nodeValue;
               break;
            }
            case "datum": {
               $datum = $child->nodeValue;
               break;
            }
            case "leiras": {
               $leiras = $child->nodeValue;
               break;
            }
         }
      }
```

```
   }
   printf("<img src='%s'><br>\n", $fileNev);
         printf("<ul><li><b>SzerzÅ‘:</b> %s</li>
                     <li><b>Mimetype:</b> %s</li>
                     <li><b>DÃ¡tum:</b> %s</li>
                     <li><b>LeÃrÃ¡s:</b> %s</li>
                </ul>
                <a href='edit.php?azon=%s'>Modosit</a> 
                <a href='delete_dom.php?azon=%s'>Torol</a>
                <hr>",
                  $szerzo,
                  $mimeType,
                  $datum,
                  $leiras,
                  $azon,
                  $azon);
}

?>
</body>
</html>
```

Az alkalmazás bővítése – Insert, Delete, Update – DOM

Egy XML dokumentum létrehozására és módosítására is alapvetően kétféle filozófia van: vagy beolvassuk az egész dokumentumot a memóriába (DOM vagy objektumhierarchia), és abban változtatunk, és kiírjuk, vagy pedig egyszerű szövegműveleteket alkalmazunk. Az előbbi memóriaigényes, viszont egyszerűbb és bonyolultabb dokumentumok esetén szinte az egyetlen út. Az utóbbi gyors és kevés memóriát igényel, egyszerűbb dokumentumok készítésénél javallott.
Most DOM-mal nézzük meg, hogyan lehet az albumunkhoz új elemet hozzávenni, módosítani és törölni egy létezőt.

edit.php

```
<?php
header("Content-Type: text/html;charset=UTF-8");

$fileNev="";
$mimeType="";
$szerzo="";
$leiras="";
$datum="";
$szelesseg="";
$magassag="";
$azon="";

if (isset($_GET["azon"])) {
   $azon=$_GET["azon"];

   $dom = new DomDocument();
   $dom->load("album1.xml");

   $xpath = new Domxpath($dom);
   $result = $xpath->query("/album/kepek/kep[@azon='$azon']");
   $kep = $result->item(0);

   foreach($kep->childNodes as $child) {
      if ($child instanceof domElement) {
         switch ($child->tagName) {
```

```
            case "fileNev": {
               $fileNev = $child->nodeValue;
               break;
            }
            case "szerzo": {
               $szerzo = $child->nodeValue;
               break;
            }
            case "mimeType": {
               $mimeType = $child->nodeValue;
               break;
            }
            case "datum": {
               $datum = $child->nodeValue;
               break;
            }
            case "leiras": {
               $leiras = $child->nodeValue;
               break;
            }
            case "szelesseg": {
               $szelesseg = $child->nodeValue;
               break;
            }
            case "magassag": {
               $magassag = $child->nodeValue;
               break;
            }
         }
      }
   }
}

?>
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-
2">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<h2><?php echo $azon!==""?"Modositas":"Uj elem felvitele"; ?></h2>
<form name="form_edit" action="<?php echo
$azon!==""?"update_dom.php?azon=".$azon:"insert_dom.php"; ?>"
method="post">
<table>
   <tr><td>Filenev:</td>  <td><input type="text" name="fileNev"
value="<?=$fileNev?>"></td></tr>
   <tr><td>Mimetype:</td>  <td><input type="text" name="mimeType"
value="<?=$mimeType?>"></td></tr>
   <tr><td>SzerzÅ‘:</td>    <td><input type="text" name="szerzo"
value="<?=$szerzo?>"></td></tr>
   <tr><td>LeÃ–rÃ¡s:</td>    <td><input type="text" name="leiras"
value="<?=$leiras?>"></td></tr>
   <tr><td>DÃ¡tum:</td>     <td><input type="text" name="datum"
value="<?=$datum?>"></td></tr>
   <tr><td>SzÃ©lessÃ©g:</td> <td><input type="text" name="szelesseg"
value="<?=$szelesseg?>"></td></tr>
```

```
   <tr><td>MagassÃ¡g:</td>  <td><input type="text" name="magassag"
value="<?=$magassag?>"></td></tr>
   <tr><td colspan="2"><input type="submit" name="editgomb" value="<?php
echo $azon!==""?"Modositas":"Uj elem felvitele"; ?>"></td></tr>
</table>
</form>
</body>
</html>
```

insert_dom.php

```
<?php
header("Content-Type: text/html;charset=UTF-8");
$dom = new DomDocument();
$dom->load("album1.xml");
$album = $dom->documentElement;
$kepekelem = $album->firstChild->nextSibling->nextSibling->nextSibling-
>nextSibling->nextSibling;
$kepek = $kepekelem->childNodes;

/*
$xpath = new Domxpath($dom);
$result = $xpath->query("/album/kepek/kep[last()]");
$ujazon = $result->item(0)->getAttribute("azon");
$ujazon++;
*/
foreach($kepek as $kep) {
   if ($kep instanceof domElement) {
      $ujazon = $kep->getAttribute("azon");
   }
}
$ujazon++;


$kep = $dom->createElement("kep");
$kep->setAttribute("azon", $ujazon);
$elem = $dom->createElement("fileNev", $_POST["fileNev"]);
$kep->appendChild($elem);
$elem = $dom->createElement("mimeType", $_POST["mimeType"]);
$kep->appendChild($elem);
$elem = $dom->createElement("szerzo", $_POST["szerzo"]);
$kep->appendChild($elem);
$elem = $dom->createElement("leiras", $_POST["leiras"]);
$kep->appendChild($elem);
$elem = $dom->createElement("datum", $_POST["datum"]);
$kep->appendChild($elem);
$elem = $dom->createElement("szelesseg", $_POST["szelesseg"]);
$kep->appendChild($elem);
$elem = $dom->createElement("magassag", $_POST["magassag"]);
$kep->appendChild($elem);

$kepekelem->appendChild($kep);

$dom->save("album1.xml");

?>
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-
2">
      <META http-equiv="Pragma" content="no-cache">
```

```
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<a href="szures_dom.php">Szures</a>
</body>
</html>
```

update_dom.php

```
<?php
header("Content-Type: text/html;charset=UTF-8");

$azon="";
if (isset($_GET["azon"])) {
   $azon=$_GET["azon"];
}

$dom = new DomDocument();
$dom->load("album1.xml");
$album = $dom->documentElement;
$kepekelem = $album->firstChild->nextSibling->nextSibling->nextSibling-
>nextSibling->nextSibling;
$kepek = $kepekelem->childNodes;

$xpath = new Domxpath($dom);
$result = $xpath->query("/album/kepek/kep[@azon='$azon']");
$kep = $result->item(0);
/*
foreach($kepek as $kep) {
   if ($kep instanceof DomElement && $kep->getAttribute("azon")==$azon) {
      break;
   }
}
*/
//echo $kep->nodeValue;

$ujkep = $dom->createElement("kep");
$ujkep->setAttribute("azon", $azon);
$elem = $dom->createElement("fileNev", $_POST["fileNev"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("mimeType", $_POST["mimeType"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("szerzo", $_POST["szerzo"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("leiras", $_POST["leiras"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("datum", $_POST["datum"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("szelesseg", $_POST["szelesseg"]);
$ujkep->appendChild($elem);
$elem = $dom->createElement("magassag", $_POST["magassag"]);
$ujkep->appendChild($elem);

$kepekelem->replaceChild($ujkep, $kep);

$dom->save("album1.xml");

?>
<html>
   <head>
      <title></title>
```

```
      <meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-
2">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<a href="szures_dom.php">Szures</a>
</body>
</html>
```

delete_dom.php

```
<?php
header("Content-Type: text/html;charset=UTF-8");

$azon="";
if (isset($_GET["azon"])) {
   $azon=$_GET["azon"];
}

$dom = new DomDocument();
$dom->load("album1.xml");
$album = $dom->documentElement;
$kepekelem = $album->firstChild->nextSibling->nextSibling->nextSibling-
>nextSibling->nextSibling;
$kepek = $kepekelem->childNodes;

$xpath = new Domxpath($dom);
$result = $xpath->query("/album/kepek/kep[@azon='$azon']");
$kep = $result->item(0);
/*
foreach($kepek as $kep) {
   if ($kep instanceof DomElement && $kep->getAttribute("azon")==$azon) {
      break;
   }
}
*/
//echo $kep->nodeValue;

$kepekelem->removeChild($kep);

$dom->save("album1.xml");

?>
<html>
   <head>
      <title></title>
      <meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-
2">
      <META http-equiv="Pragma" content="no-cache">
      <meta http-equiv="Cache-Control" content="no-cache,must-revalidate">
   </head>
<body>
<a href="szures_dom.php">Szures</a>
</body>
</html>
```

Ma megtanultuk

- XML feldolgozási filozófiákat
- PHP specifikus megoldásokat

- SAX
- SimpleXML
- DOM
- XSLT

Ezután jön

- Oracle és XML ???
- XML dinamikus XML generálás ???