XML & PHP

Виолета Божикова

Езикът на XML

- Езикът на XML е начин да се структурират данни, с цел - споделяне между уебсайтовете.
- XML е лесно да се създаде. Той много прилича на HTML, освен че вие правите свои собствени тагове. Още, всеки XML таг трябва да има затварящ таг (в HTML това не е задължително).
- По аналогия с HTML в XML – в таговете може да се използват атрибути.

Езикът на XML

- Всеки XML-документ започва с декларативен ред, например:

<?xml version='1.0' encoding='UTF-16', standalone='yes'?>

Указват се:

- версията на езика XML (атрибут **version**, например: **version='1.0'**),
- кодировката на текста в този документ (атрибут encoding, например: **encoding='UTF-16'**) и
- атрибут standalone - дали документа съществува сам за себе си или зависи от други файлове (например: **standalone='yes'**).

пример за XML-документ

- Нека имаме писмо, чието съдържание се съхранява в следния вид: note.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
   <to>Маша Петрова</to>
   <body>Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.</body>
   <from>Васил Иванов</from>
</note>
```

note.xml (версия 1)

Отношения между елементите на XML –документите

- Елементите в XML - документите могат да бъдат свързани помежду си чрез отношения **родител/потомък** или **родственик/родственик**.
- В нашия пример <note> се явява родител на <to>, който от своя страна е потомък на **<note>**, а **<to>, <body>** и <from> се явяват родственици.
- От друга страна, текстът също се явява потомък на елемента, например, текстът "Васил Иванов" се явява потомък на елемента <from>.
- Такава структура се нарича дърво; частите, които имат потомци се наричат клони, а които нямат - листа.

```
<?xml version="1.0"
encoding="UTF-8"?>
<note>
  <to>Маша Петрова</to>
  <body>Привет, Маша!
  Как си?
  Аз съм добре.
  Планирам да дойда
  на гости у вас.</body>
  <from>Васил
Иванов</from>
</note>
```

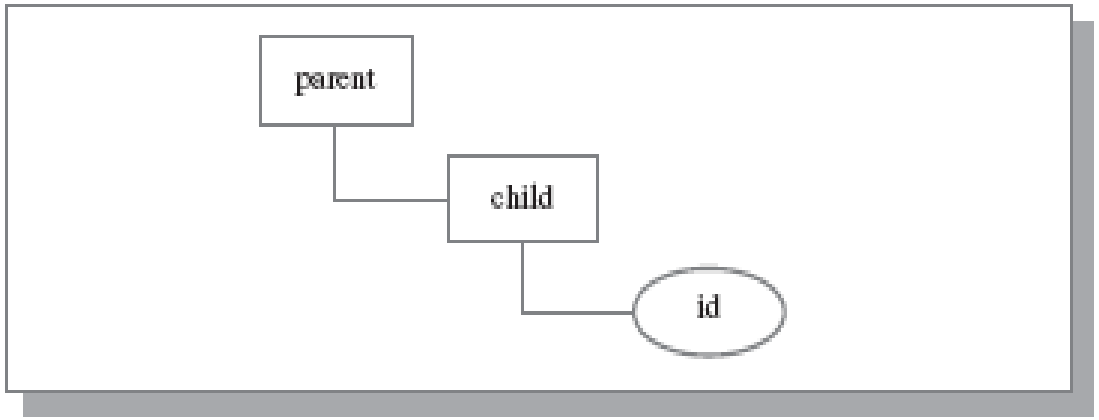
note.xml (версия 1)

Обектен модел на XML-документа

- Структурата на XML-документа много напомня на обектния модел: тя е **йерархична**, едни елементи могат да са потомци на други.
- Всеки XML -документ може да се представи в качество на обектен модел. При това едни елементи (например, тагове) са „обекти“, а други (атрибути и текстови елементи) – техни „свойства“.
- Например, за документа:

<parent> <child id="text">Добър ден!</child></parent>

обектният модел може да изглежда както е показано: правоъгълниците са обекти, а овалите – свойства на обектите.



Какво е XML Parser?

- За да прочитане и актуализация, за създаване и манипулиране на XML документ, ще ви трябва XML парсер (т.е анализатор).
- В PHP има два основни типа XML парсери:
  - Tree-Based Parsers
  - Event-Based Parsers

Tree-Based Parsers (Парсери, базирани на дървета)

- Tree базирани парсъри държат целия документ в паметта и превръщат XML документа в дървовидна структура.
- Те анализират целия документ и осигуряват достъп до елементите на дървото.
- Този тип анализатори са по-добрия избор за по-малки XML документи, но не и за голями XML документи, поради проблеми с производителността (във втория случай).
- Примери за tree-based parsers:
  - SimpleXML
  - DOM

Event-Based Parsers (Парсери, базирани на събития)

- Event-based парсърите не държат целия документ в паметта, а в даден момент четат един възел и ви позволяват да взаимодействате с него в реално време.
- След като преминете към следващия възел, старият се изхвърля.
- Този тип анализатори е много подходящ за големи XML документи. Той анализира бързо и консумира по-малко памет.
- Пример за парсъри, основани на събития:
  - XMLReader
  - XML Expat Parser

SimpleXML Parser

- SimpleXML това е tree-based parser. Представлява вградено в PHP5 (и нагоре) разширение (extension) за работа с XML данни.
- Осигурява лесен начин за извличане името на елементите, атрибутите и текстовото съдържание, ако знаете структурата или оформлението на XML документа.
- Обръща един XML документ в даннова структура, която може да обхождате, като колекция от масиви или обекти.
- Сравнен с DOM или XML Expat parser, SimpleXML изисква писане на по-малко редове код при четене на данните на един елемент.

PHP SimpleXML – Четене от String

- PHP функция **simplexml_load_string()** се използва за четене на XML данни от един string. Връща object от class SimpleXMLElement със свойства, съдържащи данните на xml документа, или **FALSE -** при неуспех.
- Да предположим, че имаме променлива $myXMLData, която съдържа следните XML данни от note.xml:

```
$myXMLData = "<?xml version='1.0' encoding='UTF-8'?>
                <note>
                  <to>Маша Петрова</to>
                  <body>Привет, Маша!
                  Как си?
                  Аз съм добре.
                  Планирам да дойда
                  на гости у вас.</body>
                  <from>Васил Иванов</from>
                </note>";
```

- Примерът по долу показва как да използваме функцията **simplexml_load_string() function** за прочитане на XML данните, съхранени в string $myXMLData:

```
<pre>
<?php
$myXMLData ="<?xml version='1.0' encoding='UTF-8'?>
<note>
   <to>Маша Петрова</to>
   <body>Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.</body>
   <from>Васил Иванов</from>
</note>";
$xml=simplexml_load_string($myXMLData) or die("Error: Cannot create object");
print_r($xml);
?>
</pre>
```

Изход:

```
SimpleXMLElement Object
(
   [to] => Маша Петрова
   [body] => Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.
   [from] => Васил Иванов
)
```

PHP SimpleXML – четене от файл

- Използва се **simplexml_load_file()** функция на PHP за четене на **XML данни от** XML **файл**.
- Да предположим, че имаме един XML файл, наречен "note.xml", който изглежда по познатия начин:

```
<?xml version="1.0" encoding="UTF-8" standalone='yes' ?>
<note title="Писмо">
   <to>Маша Петрова</to>
   <body>Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.</body>
   <from>Васил Иванов</from>
</note>
```

note.xml

Index.php

```
<pre>
<?php
$xml=simplexml_load_file("note.xml")
or die("Error: Cannot create object");
print_r($xml);
?>
</pre>
```

Изход:

```
SimpleXMLElement Object
(    [@attributes] => Array
        (
            [title] => Писмо
        )
    [to] => Маша Петрова
    [body] => Привет, Маша!
    Как си?
    Аз съм добре.
    Планирам да дойда
    на гости у вас.
    [from] => Васил Иванов
)
```

Извличане на стойности на възел (елемент)

- Извличане на стойностите на елемент от "note.xml" файл:

```
<pre>
<?php
$xml=simplexml_load_file("note.xml")
or die("Error: Cannot create
object");
echo $xml->to . "<br>";
echo $xml->body . "<br>";
echo $xml->from . "<br>";
?>
</pre>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Маша Петрова</to>
  <body>Привет, Маша!
  Как си?
  Аз съм добре.
  Планирам да дойда
  на гости у вас.</body>
  <from>Васил Иванов</from>
</note>
```

note.xml

изход:

```
Маша Петрова
Привет, Маша!
     Как си?
     Аз съм добре.
     Планирам да дойда
     на гости у вас.
Васил Иванов
```

Извличане на стойности на възел (елемент)

Още един XML File:

- Да разгледаме колекция, съдържаща описание на личности.
- Всяка личност се описва с такива характеристики като:
  - фамилия,
  - име,
  - дата на раждане и
  - електронен адрес.
- Така стуктурата на колекцията "Личности ", където се съхраняват данните за всички известни за нас личности може да се представи по следния начин:

```
<?xml version="1.0"?>
<collection>
    <person id="10">
        <name>
            <first>Nick</first>
            <last>Petrov</last>
        </name>
        <birth>
            <day>23</day>
            <month>12</month>
            <year>89</year>
        </birth>
        <email> nick@abv.bg
        </email>
    </person>
    <person id="20">
        <name>
            <first>Boris</first>
            <last>Ivanov</last>
        </name>
        <birth>
            <day>03</day>
            <month>05</month>
            <year>90</year>
        </birth>
        <email> bois@yahoo.com
        </email>
    </person>
</collection>
```

persons.xml

Извличане на стойности на възел (елемент)

- Извличане (Get) на стойностите на елементите email, name->last и birth->day на първия и втория възли „person” от " persons.xml" файл:

```
Изход:
Име:Petrov
Рождена дата:23
email: nick@abv.bg

Име:Ivanov
Рождена дата:03
email: nick@abv.bg
```

Index.php

```
<pre>
<?php
$xml=simplexml_load_file("persons.xml") or die("Error: Cannot
create object");
echo "Име:".$xml->person[0]->name->last. "<br>";
echo "Рождена дата:".$xml->person[0]->birth->day."<br>";
echo "email:".$xml->person[0]->email."<br>";

echo "Име:".$xml->person[1]->name->last."<br>";
echo "Рождена дата:".$xml->person[1]->birth->day."<br>";;
echo "email:".$xml->person[0]->email."<br>";
?>
</pre>
```

Още един XML File

Пример с PHP SimpleXML:
Извличане на стойността на елемент на даден възел

- Следният код извлича стойността на елемент <title> в първия и втория възли <book> в "books.xml" файл:

Example

```
<?php
$xml=simplexml_load_file("books.xml") or die("Error: Cannot create object");
echo $xml->book[0]->title."<br>";
echo $xml->book[1]->title;
?>
```

```
<?xml version="1.0" encoding="utf-8"?>
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">XQuery Kick Start</title>
    <author>James McGovern</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

books.xml

Изход:

Everyday Italian
Harry Potter

Извличане на стойностите на елементите на възлите, чрез цикъл

Например, следния пример обхожда всички <book> елементи в "books.xml" файл, и извлича стойностите на възловите елементи <title>, <author>, <year>, и <price>:

Изход:

Everyday Italian, Giada De Laurentiis, 2005, 30.00
Harry Potter, J K. Rowling, 2005, 29.99
XQuery Kick Start, James McGovern, 2003, 49.99
Learning XML, Erik T. Ray, 2003, 39.95

```php
<?php
$xml=simplexml_load_file("books.xml")
or die("Error: Cannot create object");
foreach($xml->children() as $books) {
   echo $books->title . ", ";
   echo $books->author . ", ";
   echo $books->year . ", ";
   echo $books->price . "<br>";
}
?>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">XQuery Kick Start</title>
    <author>James McGovern</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

books.xml

Извличане на стойностите на елементите на възлите, чрез цикъл

Например, следния пример обхожда всички <person> елементи в „persons.xml" файл, и извлича стойностите на възловите елементи:

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <person id="10">
    <name>
      <first>Nick</first>
      <last>Petrov</last>
    </name>
    <birth>
      <day>23</day>
      <month>12</month>
      <year>89</year>
    </birth>
    <email> nick@abv.bg
    </email>
  </person>
  <person id="20">
    <name>
      <first>Boris</first>
      <last>Ivanov</last>
    </name>
    <birth>
      <day>03</day>
      <month>05</month>
      <year>90</year>
    </birth>
    <email> bois@yahoo.com
    </email>
  </person>
</collection>
```

Изход:

```
Petrov, Nick, 23,  nick@abv.bg
Ivanov, Boris, 03,  bois@yahoo.com
```

Index.php

```
<pre>
<?php
$xml=simplexml_load_file("persons.xml")
or die("Error: Cannot create object");
foreach($xml->children() as $persons) {
   echo $persons->name->last. ", ";
   echo $persons->name->first. ", ";
   echo $persons->birth->day . ", ";
   echo $persons->email . "<br>";
}
?>
</pre>
```

За persons.xml – Извличане на стойностите на атрибута id на person:

```
<?php
$xml=simplexml_load_file("persons.xml")
or die("Error: Cannot create object");
echo $xml->person[0]['id'] . "<br>";
echo $xml->person[1]['id'];
?>
```

Изход:

10

20

```
<?xml version="1.0" encoding="utf-8"?>
<collection>
  <person id="10">
    <name>
      <first>Nick</first>
      <last>Petrov</last>
    </name>
    <birth>
      <day>23</day>
      <month>12</month>
      <year>89</year>
    </birth>
    <email> nick@abv.bg
    </email>
  </person>
  <person id="20">
    <name>
      <first>Boris</first>
      <last>Ivanov</last>
    </name>
    <birth>
      <day>03</day>
      <month>05</month>
      <year>90</year>
    </birth>
    <email> bois@yahoo.com
    </email>
  </person>
</collection>
```

За book.xml – Извличане на стойностите на атрибута category на book:

Следващият пример получава:

- стойността на атрибута на атрибут "категория" на първия <book> елемент и
- стойността на атрибута на атрибут "Lang" на елемент <заглавие> във втория < title> във втория елемент <book >:

```
<?php
    $xml=simplexml_load_file("books.xml
    ") or die("Error: Cannot create
    object");
    echo $xml->book[0]['category'] .
    "<br>";
    echo $xml->book[1]->title['lang'];
?>
```

Изход:

COOKING
en

```
<?xml version="1.0" encoding="utf-8"?>
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">XQuery Kick Start</title>
    <author>James McGovern</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

books.xml

PHP SimpleXML – Извличане на атрибутите на елементите в цикъл

- Примерът извлича стойностите на атрибутит ите на елемент <title> в"books.xml" файл:

```
<?php
    $xml=simplexml_load_file("books.x
    ml") or die("Error: Cannot create
    object");
    foreach($xml->children() as $books) {
       echo $books->title['lang'];
     echo "<br>";
    }
?>
```

Изход:

en
en
en-us
en-us

```
<?xml version="1.0" encoding="utf-8"?>
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">XQuery Kick Start</title>
    <author>James McGovern</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="WEB">
    <title lang="en-us">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

books.xml

XML Expat Parser - това е event-based parser

- Вграденият в PHP **XML Expat Parser** (той е част от ядрото на PHP) дава възможност за обработваме XML документи с PHP.
- Да видим следните XML фрагмент:

<from>Jani</from>

- Един event-based parser се отнася към този елемент като към серия от 3 събития:
  - **Стартиране на елемент**: **from**
  - **Стартиране на CDATA секция, стойност**: **Jani**
  - **Затваряне на елемент**: **from**

"note.xml" ще бъде използван в примерите, които следват

```
<?xml version="1.0"
encoding="UTF-8"
standalone='yes' ?>
<note title="Писмо">
   <to>Маша Петрова</to>
   <body>Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.</body>
   <from>Васил
Иванов</from>
</note>
```

note.xml

- **Ще инициализираме XML Expat Parser в PHP**
- **Ще дефинираме няколко обработчици (handlers) на различни XML събития, и тогава**
- **Ще парсваме XML файла.**

```
<pre>
<?php
// Инициализация на XML parser
$parser=xml_parser_create();
// Функция start за използване в началото на
      елемент
function  start($parser,$element_name,
      $element_attrs) {
  switch($element_name) {
    case "NOTE":
    echo "-- Note --<br>";
    break;
    case "TO":
    echo "To: ";
    break;
    case "FROM":
    echo "From: ";
    break;
    case "BODY":
    echo "Message: ";
  }
} // Функция stop за използване в края на
      елемента
function stop($parser,$element_name) {
  echo "<br>";
}
```

Резултат:

```
-- Note --

     To: Маша Петрова

     Message: Привет, Маша!
     Как си?
     Аз съм добре.
     Планирам да дойда
     на гости у вас.

     From: Васил Иванов
```

```
// Функция за използване когато се срещнат
символни данни
function char($parser,$data) {
  echo $data;
}

// Специфицира се манипулатор на елемент (element
handler)
xml_set_element_handler($parser,"start","stop");

// Специфицира се даннов манипулатор (data
handler)
xml_set_character_data_handler($parser,"char");

// Отваряне на XML файла
$fp=fopen("note.xml","r");

// Четене на данните от файла
while ($data=fread($fp,4096)) {
  xml_parse($parser,$data,feof($fp)) or
  die (sprintf("XML Error: %s at line %d",
  xml_error_string(xml_get_error_code($parser)),
  xml_get_current_line_number($parser)));
} // Освобождаване на XML parser
xml_parser_free($parser);
?>
</pre>
```

Обяснение на примера:

- Инициализация на XML parser с xml_parser_create() function
- Създаване на функции, които да се използват с различни обработчици на събития (event handlers)
- Добавяне на функцията xml_set_element_handler(), за да се определи кои функции ще се изпълняват, когато парсера срещне отварящия и затварящ тагове
- Добавяне на функцията xml_set_character_data_handler(), за да се определи кои функции ще се изпълняват, когато парсера срещне символни данни
- Разбор на "note.xml" файла с xml_parse () функция
- В случай на грешка, добавете xml_error_string() функция за конвертиране на XML грешка към текстово описание
- Извикване на функцията xml_parser_free(), за да се освободи паметта, заета в резултат на xml_parser_create() функция

The XML DOM Parser

- Вграденият DOM parser дава възможност да се обработват XML документи с PHP. Част е от ядрото на PHP и не изисква инсталация.
- Това е също **tree-based parser**.
- Нека да вземем следния фрагмент от XML документ:

<?xml version="1.0" encoding="UTF-8"?>
<from>Jani</from>

- DOM вижда XML фрагмента като дървовидна структура:
- Ниво 1: **XML Document**
- Ниво 2: **Root елемент: <from>**
- Ниво 3: **Текстов елемент: "Jani"**

Нека да използваме като пример писмо файл note.xml

Зареждане и извежане на XML файла

- Трябва да инициализираме XML parser, да заредим (load) xml документа и да го изведем:

```
<?php
    $xmlDoc = new DOMDocument();
    $xmlDoc->load("note.xml");
    print $xmlDoc->saveXML();
?>
```

Изходът от нашия пример е:

Маша Петрова
Привет, Маша!
Как си?
Аз съм добре.
Планирам да дойда на гости у вас.
Васил Иванов

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Маша Петрова</to>
  <body>Привет, Маша!
  Как си?
  Аз съм добре.
  Планирам да дойда
  на гости у вас.</body>
  <from>Васил Иванов</from>
</note>
```

note.xml

- Примерът създава един **DOMDocument-Object** и зарежда XML от "note.xml" в него.
- След което функция **saveXML()** вкарва **DOMDocument-Object** в един string, който може да изведем.
- Ако изберете "View source“ в прозореца на браузъра си, ще видите следния HTML:

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
   <to>Маша Петрова</to>
   <body>Привет, Маша!
   Как си?
   Аз съм добре.
   Планирам да дойда
   на гости у вас.</body>
   <from>Васил Иванов</from>
</note>
```

Обхождане в XML документа (Looping through XML)

Ние трябва да:

- инициализираме XML parser,
- заредим XML,
- обходим всички елементи на <note> елемента:

```
<?php
    $xmlDoc = new DOMDocument();
    $xmlDoc->load("note.xml");

    $x = $xmlDoc->documentElement;

foreach ($x->childNodes AS $item)
   {print $item->nodeName . " = " .
   $item->nodeValue . "<br>“;  }
?>
```

- Вие виждате, че има празни text възли между елементите в резултата от изпълнение.
- Когато се генерира XML, той често съдържа white-spaces между възлите. XML DOM parser третира тези пространства като обикновени елементи, и ако не ги зачитате, това понякога ще ви създаде проблем

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Маша Петрова</to>
  <body>Привет, Маша!
  Как си?
  Аз съм добре.
  Планирам да дойда
  на гости у вас.</body>
  <from>Васил Иванов</from>
</note>
```

note.xml

```
Изходът от нашия файл е:
#text =

to = Маша Петрова
#text =

body = Привет, Маша!
     Как си?
     Аз съм добре.
     Планирам да дойда
     на гости у вас.
#text =

from = Васил Иванов
#text =
```

Използвана литература

- **http://www.w3schools.com/php/php_xml_parsers.asp**