Manipulação de arquivos XML com PHP

Combinação perfeita

Bernhard Aichinger – www.sxc.hu

A manipulação de arquivos XML é de suma importância no desenvolvimento de aplicações, seja para o armazenamento de arquivos de configuração, de dados reais, ou mesmo para o intercâmbio de informações entre diferentes aplicações. Use as funções SimpleXML do PHP 5 para tratar arquivos XML.

por Pablo Dall'Oglio

XML é, sem dúvida, uma tecnologia inovadora. Arquivos XML são usados, hoje em dia, para os mais diversos fins, desde a definição de interfaces gráficas para aplicativos até o envio de dados de uma aplicação para outra.

A linguagem de programação *PHP*, em sua versão 5, possui a extensão *SimpleXML*. Esse conjunto de funções facilita bastante a interpretação de arquivos XML, permitindo assim seu carregamento, alteração e utilização para os mais diversos fins.

Em nosso primeiro exemplo, vamos realizar uma simples leitura de um documento XML e analisar seu retorno. Aqui, tomamos como base o arquivo XML do **exemplo 1**. No pequeno programa mostrado no **exemplo 2**, carregamos o documento XML `paises.xml` através da função `simplexml_load_file()`, e analisamos o resultado desse procedimento, através da função `var_dump()`. É possível ver claramente cada *tag* do XML sendo transformada em propriedades do objeto resultante.

A função `simplexml_load_file()` realiza a leitura de um documento XML, criando ao final um objeto do tipo `SimpleXmlElement`. Caso o documento esteja formatado incorretamente, ou mesmo que não seja um documento XML, a função retornará `FALSE`.

O resultado da execução do código do **exemplo 2**, nesse caso, é:

```
object(SimpleXMLElement)#1 (5) {
  ["nome"]=>      string(8)  " Brasil "
  ["idioma"]=>    string(11) " portugues "
  ["religiao"]=>  string(10) " catolica "
  ["moeda"]=>     string(11) " Real (R$) "
  ["populacao"]=> string(13) " 190 milhoes "
}
```

Um objeto `SimpleXMLElement` representa um elemento de um documento XML. Quando interpretado através das funções `simplexml_load_file()` ou `simplexml_load_string()`, o documento XML resulta em um objeto do tipo `SimpleXMLElement`, contendo seus atributos e valores, bem como outros objetos `SimpleXMLElement` internos, representando subelementos (nodos). Abaixo, estão alguns dos métodos oferecidos por um objeto do tipo `SimpleXmlElement`:

- `asXML ()`: Retorna uma cadeia de caracteres formatada em XML representando o objeto, bem como seus subelementos.
- `attributes ()`: Lista os atributos definidos dentro da tag XML do objeto.
- `children ()`: Retorna os elementos filhos do objeto (sub-nodos), bem como seus valores.
- `addChild(string nome, string valor, string namespace)`: Adiciona um elemento ao nodo especificado e retorna um objeto do tipo `SimpleXmlElement`.

O **exemplo 3** demonstra como acessar diretamente as propriedades do objeto `SimpleXMLElement` resultante da leitura do documento XML. Para tal, utilizamos o mesmo arquivo XML do exemplo anterior, imprimindo na tela as propriedades do objeto resultante com seu respectivo valor.

O resultado da execução do **exemplo 3** é:

```
Nome :  Brasil
Idioma :  portugues
Religiao :  catolica
Moeda :  Real (R$)
População :  190 milhoes
```

Já vimos como acessar diretamente as propriedades do XML sabendo os seus nomes. No **exemplo 4**, vamos percorrer o mesmo documento XML e imprimir na tela suas propriedades (tags), mesmo sem saber seus nomes. Isso é possível através da utilização do método `children()`, que recebe um objeto `SimpleXMLElement` e retorna todos os seus elementos filhos na forma de um vetor contendo a chave e o valor, que pode ser iterado por um laço `foreach`.

A execução do **exemplo 4** gera a seguinte saída:

Exemplo 1: paises.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pais>
        <nome> Brasil </nome>
        <idioma> portugues </idioma>
        <religiao> catolica </religiao>
        <moeda> Real (R$) </moeda>
        <populacao> 190 milhoes </populacao>
</pais>
```

Exemplo 2: Processamento do arquivo paises.xml

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises.xml');

// exibe as informaçoes do objeto criado
var_dump($xml);
?>
```

Exemplo 3: Acesso às propriedades do objeto SimpleXMLElement

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises.xml');

// imprime os atributos do objeto criado
echo 'Nome : '     . $xml->nome      . "\n";
echo 'Idioma : '   . $xml->idioma    . "\n";
echo 'Religiao : ' . $xml->religiao  . "\n";
echo 'Moeda : '    . $xml->moeda     . "\n";
echo 'Populaçao : '. $xml->populacao . "\n";
?>
```

Exemplo 4: Tags de paises.xml

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises.xml');

foreach ($xml->children() as $elemento => $valor)
{
  echo "$elemento -> $valor\n";
}
?>
```

Exemplo 5: paises2.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pais>
  <nome> Brasil </nome>
  <idioma> portugues </idioma>
  <capital> Brasilia </capital>
  <religiao> catolica </religiao>
  <moeda> Real (R$) </moeda>
  <populacao> 190 milhoes </populacao>
  <geografia>
          <clima> tropical </clima>
          <costa> 7367 km </costa>
          <pico> Neblina (3014 m) </pico>
  </geografia>
</pais>
```

Exemplo 6: Interpretação de paises2.xml

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises2.xml');

echo 'Nome : '     . $xml->nome      . "\n";
echo 'Idioma : '   . $xml->idioma    . "\n";

echo "\n";
echo "*** Informaçoes Geograficas ***\n";
echo 'Clima : '    . $xml->geografia->clima . "\n";
echo 'Costa : '    . $xml->geografia->costa . "\n";
echo 'Pico  : '    . $xml->geografia->pico . "\n";
?>
```

Exemplo 7: Alteração de propriedades

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises2.xml');

// alteraçao de propriedades
$xml->populacao = '220 milhoes';
$xml->religiao  = 'cristianismo';
$xml->geografia->clima = 'temperado';
// adiciona novo nodo
$xml->addChild('presidente', 'Chapolin');

// exibindo o novo XML
echo $xml->asXML();
// grava no arquivo paises2.xml
file_put_contents('paises2.xml', $xml->asXML());
?>
```

```
nome ->  Brasil
idioma ->  portugues
religiao ->  catolica
moeda ->  Real (R$)
populacao ->  190 milhoes
```

Aperfeiçoamos um pouco nosso documento XML, adicionando uma seção `geografia`, que agrupa informações como `clima`, `costa` e `pico`. Nesse caso, o objeto resultante (`$xml`) irá conter uma propriedade chamada `geografia`, que é também um objeto do tipo `SimpleXMLElement` com suas respectivas propriedades, sendo passíveis da utilização dos mesmos métodos listados no início deste artigo (`asXML()`, `children()`, `attributes()` etc...)

A saída da execução do **exemplo 6** é:

```
Nome :  Brasil
Idioma :  portugues

*** Informaçoes Geograficas ***
Clima :  tropical
Costa :  7367 km
Pico  :  Neblina (3014 m)
```

Já realizamos o acesso ao documento XML de diversas formas; agora veremos como alterar o seu conteúdo, no **exemplo 7**. Nesse caso, após a carga do documento, atribuímos novos valores aos campos, acessando diretamente cada uma das propriedades que se deseja alterar no objeto. Também adicionamos um novo nodo, chamado `presidente`, através do método `addChild()`. Após as atribuições de novos valores, utilizamos o método `asXML()` para retornar o novo documento XML formatado e atualizado. Caso queiramos sobrescrever o arquivo original, basta utilizarmos a função `file_put_contents()`.

O resultado da execução do **exemplo 7** é:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pais>
   <nome> Brasil </nome>
   <idioma> portugues </idioma>
   <capital> Brasília </capital>
   <religiao>cristianismo</religiao>
   <moeda> Real (R$) </moeda>
   <populacao>220 milhoes</populacao>
   <geografia>
      <clima>temperado</clima>
      <costa> 7367 km </costa>
      <pico> Neblina (3014 m)</pico>
   </geografia>
   <presidente>Chapolin</presidente>
</pais>
```

Agora que já vimos como alterar o conteúdo do XML, vamos ver como se dá o acesso a elementos repetitivos. No **exemplo 8**, adicionamos uma seção chamada `<estados>`, que contém uma listagem com nomes de Estados. O código a seguir imprime na tela algumas informações do documento XML, como nos exemplos anteriores, e também mostra como exibir esses elementos repetitivos, através do laço de repetições `foreach()`.

O resultado da execução do código do **exemplo 9** é:

```
Nome :  Brasil
Idioma :  portugues

*** Estados ***
Estado :  Rio Grande do Sul
Estado :  Sao Paulo
Estado :  Minas Gerais
Estado :  Rio de Janeiro
Estado :  Parana
Estado :  Mato Grosso
```

Agora vamos complicar um pouco mais nosso arquivo XML. O arquivo do **exemplo 10** contém elementos com atributos na definição da própria tag. O **exemplo 11** ilustra como trabalhar com elementos que contêm atributos na definição da própria tag (`nome="Minas Gerais" capital="Belo Horizonte"`), como o arquivo do **exemplo 10**. Para tal, alteramos a lista de Estados para conter o `nome` e a `capital`. Nesse caso, as informações são retornadas em forma de vetor. Utilizando estruturas como o `foreach()` para percorrer a lista de Estados, acessamos cada atributo do vetor de forma indexada, sendo o índice o próprio nome do atributo que desejamos acessar. Veja o **exemplo 11**.

A execução do **exemplo 11** gera a seguinte saída:

```
*** Estados ***
Estado : Rio Grande do Sul
➥ Capital: Porto Alegre
Estado : São Paulo Capital: São Paulo
Estado : Minas Gerais
➥ Capital: Belo Horizonte
Estado : Rio de Janeiro
➥ Capital: Rio de Janeiro
Estado : Paraná Capital: Curitiba
Estado : Mato Grosso Capital: Cuiabá
```

Veja que no **exemplo 11** é necessário saber exatamente o nome dos atributos que desejamos acessar, de forma indexada. No **exemplo 12**, percorremos a lista de Estados através de um `foreach()`, como no exemplo anterior; a diferença é que, dentro desse laço de repetição, podemos percorrer os atributos de cada `$estado` (que é, na verdade, um objeto `SimpleXMLElement`), através do método `attributes()`, que retorna a chave e o valor de cada atributo de um elemento `nodo`.

A execução do **exemplo 12** leva à seguinte saída:

Exemplo 8: paises3.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pais>
  <nome> Brasil </nome>
  <idioma> portugues </idioma>
  <religiao> catolica </religiao>
  <moeda> Real (R$) </moeda>
  <populacao> 190 milhoes </populacao>
  <geografia>
          <clima> tropical </clima>
          <costa> 7367 km </costa>
          <pico> Neblina (3014 m) </pico>
  </geografia>
  <estados>
          <nome> Rio Grande do Sul </nome>
          <nome> Sao Paulo </nome >
          <nome> Minas Gerais </nome>
          <nome> Rio de Janeiro </nome>
          <nome> Parana </nome>
          <nome> Mato Grosso </nome>
  </estados>
</pais>
```

Exemplo 10: paises4.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<pais>
  <nome> Brasil </nome>
  <idioma> portugues </idioma>
  <religiao> catolica </religiao>
  <moeda nome="Real" simbolo="R$"/>
  <populacao> 190 milhoes </populacao>
  <geografia>
          <clima> tropical </clima>
          <costa> 7367 km </costa>
          <pico nome="Neblina" altitude="3014"/>
  </geografia>
  <estados>
          <estado nome="Rio Grande do Sul" capital="Porto Alegre"/>
          <estado nome="Sao Paulo" capital="Sao Paulo"/>
          <estado nome="Minas Gerais" capital="Belo Horizonte"/>
          <estado nome="Rio de Janeiro" capital="Rio de Janeiro"/>
          <estado nome="Parana" capital="Curitiba"/>
          <estado nome="Mato Grosso" capital="Cuiabá"/>
  </estados>
</pais>
```

```
*** Estados ***
nome=>Rio Grande do Sul
capital=>Porto Alegre
nome=>São Paulo
capital=>São Paulo
nome=>Minas Gerais
capital=>Belo Horizonte
nome=>Rio de Janeiro
capital=>Rio de Janeiro
nome=>Paraná
capital=>Curitiba
nome=>Mato Grosso
capital=>Cuiabá
```

Exemplo 9: Interpretação de elementos repetitivos

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises3.xml');

echo 'Nome : '     . $xml->nome      . "\n";
echo 'Idioma : '   . $xml->idioma    . "\n";

echo "\n";
echo "*** Estados ***\n";

/*        Voce pode acessar um estado diretamente pelo seu indice:
  echo $xml->estados->nome[0];
*/

foreach ($xml->estados->nome as $estado)
{
  echo 'Estado : '    . $estado . "\n";
}
?>
```

Exemplo 11: Interpretação de paises4.xml

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises4.xml');

echo "*** Estados ***\n";

// percorre a lista de estados
foreach ($xml->estados->estado as $estado)
{
  // imprime o estado e a capital
  echo str_pad('Estado : '    . $estado['nome'], 30) .
               'Capital: '    . $estado['capital']   . "\n";
}

?>
```

Exemplo 12: Uso do foreach

```
<?php
// interpreta o documento XML
$xml = simplexml_load_file('paises4.xml');

echo "*** Estados ***\n";

// percorre os estados
foreach ($xml->estados->estado as $estado)
{
  // percorre os atributos de cada estado
  foreach ($estado->attributes() as $key => $value)
  {
          echo "$key=>$value\n";
  }
}
?>
```

Conclusão

Como pudemos ver neste artigo, a manipulação de arquivos XML no PHP é extremamente simples (afinal, a extensão se chama *SimpleXML*), e pode ser realizada com apenas algumas linhas de código. Esses recursos provam mais uma vez os grandes avanços proporcionados pela quinta versão dessa linguagem de programação que cresce diariamente em popularidade por sua facilidade de uso, flexibilidade e também pelo crescente número de recursos criados pela sua grande comunidade de desenvolvedores. ■

O autor

Pablo Dall'Oglio é autor de projetos em Software Livre reconhecidos, como o *Agata Report* (http://agata.dalloglio.net) e o editor *Tulip* (http://tulip.dalloglio.net). É autor do primeiro livro sobre PHP-GTK no mundo (http://www.php-gtk.com.br/). É especialista em orientação a objetos, *PHP* e *PHP-GTK*. Pode ser contatado pelo email pablo@php.net ou pablo@dalloglio.net.