1 Curso POO PHP XMLParser

1.1 XML Parser

Os analizadores de documentos XML SimpleXML e DOM non son funcionais cando se trata de procesar documentos XML de gran lonxitude, debido principalmente ao seu consumo de memoria. Nestes casos teremos que empregar algunha alternativa, como é o caso da extensión **XMLParser**, que emprega unha aproximación SAX e ven habilitada por defecto con PHP.

1.1.1 Procesamento de documentos XML

Os pasos que debemos seguir para procesar un documento XML empregando XMLParser son:

- Crear un recurso *intérprete* mediante unha chamada á función **xml_parser_create**.

```
$parser = xml_parser_create();
```

- Establecer, se é necesario, as opcións de procesamento empregando a función **xml_parser_set_option**. Unha das accións máis utilizadas é desactivar a opción **XML_OPTION_CASE_FOLDING**, que por defecto ven activada e provoca que se convertan a maiúsculas as letras minúsculas.

```
xml_parser_set_option($parser, XML_OPTION_CASE_FOLDING, 0);
```

- Configurar os manexadores de eventos que serán chamados cando se atopen no documento XML elementos (**xml_set_element_handler**), textos (**xml_set_character_data_handler**) ou instruccións de procesamento (**xml_set_processing_instruction_handler**), entre outros.

```
xml_set_element_handler($parser, "inicio", "fin");
xml_set_character_data_handler($parser, "texto");
```

  No caso do manexador de elementos, defínense dúas funcións que se chamarán cando se atope a etiqueta de apertura e a de peche do elemento respectivamente.

- Abrir o ficheiro co documento XML, e ir lendo e procesando anacos do mesmo coa función **xml_parse**.

```
$ficheiro = fopen("exemplo.xml", "r");
while ($data = fread($ficheiro, 4096)) {
   if(!xml_parse($parser, $data, feof($ficheiro))) exit("Erro ao analizar o documento XML");
}
```

- Unha vez rematado o procesamento do documento, débese liberar o recurso chamando á función **xml_parser_free**.

```
xml_parser_free($parser);
```

1.1.2 Creando unha clase que empregue XMLParse

Podemos encapsular o comportamento anterior empregando unha clase que conteña as funcións manexadoras dos eventos, e que se encargue de crear e inicializar o manexador. Neste caso é preciso empregar **xml_set_object** para indicar a clase que contén os manexadores.

O código seguinte aglutina o anterior empregando XMLParser para crear unha clase que procese o noso documento de exemplo e cree unha táboa HTML por cada película que conteña os seus datos principais.

```
class Analizador {
private $elemento_actual;

function __construct($nome_ficheiro) {
$parser = xml_parser_create();
xml_parser_set_option($parser, XML_OPTION_CASE_FOLDING, 0);

xml_set_object($parser, $this);
xml_set_element_handler($parser, "inicio", "fin");
xml_set_character_data_handler($parser, "texto");

$ficheiro = fopen($nome_ficheiro, "r");
while ($data = fread($ficheiro, 4096)) {
```

```
  if(!xml_parse($parser, $data, feof($ficheiro))) exit("Erro ao analizar o documento XML");
}
xml_parser_free($parser);
}

function inicio($parser, $nome_elemento, $atributos) {
$this->elemento_actual = $nome_elemento;
switch($nome_elemento) {
case "película":
echo '<table>';
break;
case "título":
echo '<tr> <th colspan="2">';
break;
case "títuloorixinal":
echo ' (';
break;
case "director":
case "xénero":
case "duración":
echo '<tr><td>' . $nome_elemento . '</td><td>';
}
}

function fin($parser, $nome_elemento) {
$this->elemento_actual = "";
switch($nome_elemento) {
case "película":
echo '</table>';
break;
case "títuloorixinal":
echo ')</th></tr>';
break;
case "director":
echo '</td></tr>';
}
}

function texto($parser, $data) {
switch($this->elemento_actual) {
case "película":
case "título":
case "títuloorixinal":
case "director":
case "xénero":
case "duración":
echo $data;
}
}
}

$a = new Analizador('exemplo.xml');
```

O resultado obtido será algo como o seguinte.

El santo (The Saint)	
director	Phillip Noyce
xénero	Acción
duración	111

--Víctor Lourido 23:35 24 jul 2013 (CEST)