DIABD. Módulo de sindicación de contenidos

Tarea 2: Lectura de canales de sindicación

Profesor Manuel Blázquez Ochando

Introducción:

Los canales de sindicación pueden ser aprovechados para su lectura y explotación. De hecho la información contenida en un canal de sindicación se encuentra delimitada, etiquetada y clasificada, por lo que su recuperación e interpretación puede ser llevada a cabo mediante programas de análisis de la codificación de los archivos XML. Tales programas se llaman PARSERS y permiten elaborar un mapa jerárquico de los elementos y nodos que constituyen los archivos XML de los canales de sindicación. Este proceso permite seleccionar y recuperar la información contenida en dichos archivos para su impresión en pantalla, ingreso en base de datos o transformación en terceros archivos.

El objetivo de la tarea2 es aprender a desarrollar un programa Parser y probar su funcionamiento con los canales de sindicación. Para ello se proporciona un código parser que deberá ser ampliado, probado e introducido en Joomla.

Explicación:

1. El parser new simpleXMLElement()

Como se acaba de introducir un programa PARSER es aquel que analiza la estructura de un archivo en este caso de tipo XML, para elaborar un mapa de los nodos, elementos y atributos de que está formado el canal de sindicación.

Existen muchos programas destinados a tal fin, pero en esta tarea2 se propone uno de los más utilizados por su sencillez y facilidad para ser adaptado a cualquier caso y formato. Se trata del parser **new simpleXMLElement()**.

Se trata de una función PHP que incorpora las funcionalidades de análisis de estructuras XML de DOM (Document Object Model), lenguaje utilizado para el modelado, selección y acceso a los lenguajes de marcado. Dicho de otra forma, permite elaborar un mapa completo de la estructura del archivo XML para el acceso y selección de la información que contiene.

No obstante, como se comprobará, el método de selección y acceso a los contenidos en el lenguaje PHP se lleva a cabo por medio de XPath y no mediante DOM (DOM sólo afecta al proceso de análisis y mapeado del archivo y XPath al de selección y acceso a los datos propiamente dichos).

Un ejemplo sencillo de PARSER podría ser el que se muestra en la siguiente tabla.

DIABD. Módulo de sindicación de contenidos

Tarea 2: Lectura de canales de sindicación

Profesor Manuel Blázquez Ochando

Ejemplo de programa PARSER RSS2.0 para la obtención del título del canal

```php
<?php

// PASO 1: La variable $feed contiene la URL del canal de sindicación
$feed = "URL";

// PASO 2: La variable $mapa contiene la función de parser del canal de sindicación
$mapa = new SimpleXMLElement($feed, NULL, TRUE);

// PASO 3: La variable $title alberga la selección del título del canal
$title = $mapa->xpath("//channel/title");

// PASO 4: Imprimir en pantalla la variable $title
echo "$title";

?>
```

El método de aplicación de un parser como el del ejemplo, se lleva a cabo en tres pasos.

PASO 1:

En primer lugar es necesario determinar la URL del canal de sindicación que se pretende leer. Para ello se crea una variable **$feed** que contendrá el valor de la URL completa. Es necesario cerciorarse de que la URL del canal de sindicación haga referencia al archivo XML en formato RSS, Atom o RDF.

PASO 2:

En segundo lugar se tiene que ejecutar el proceso de mapeado y análisis de la estructura del canal de sindicación. Ello se consigue mediante la variable **$mapa** en la que se almacenarán los resultados del proceso de parseo. De hecho la variable **$mapa** contiene la función parser mencionada anteriormente **new simpleXMLElement()** que hace referencia al canal de sindicación referido en el paso1.

PASO 3:

Una vez ejecutada la función de parseo, se pueden seleccionar los contenidos del canal de sindicación, mediante el lenguaje XPath a partir del mapa elaborador por el parser, que facilita el marcado de la ruta de acceso a los nodos, elementos y atributos del archivo XML. Para ello es necesario conocer cuál es la estructura básica del formato de sindicación. De esta forma el contenido del elemento seleccionado será almacenado en la variable **$title** para su posterior reaprovechamiento.

PASO 4:

La variable **$title** contiene el título del canal de sindicación. Para visualizarlo en pantalla, no hay más que utilizar la función **echo "$title";**

2. ¿Qué contiene la variable $mapa?

Para conocer realmente el contenido resultante del proceso de parseo de la función **new simpleXMLElement()** sobre el canal de sindicación lo mejor es efectuar una prueba de impresión en pantalla. Dado que la variable **$mapa**, se convierte en una matriz de matrices o lo que es lo mismo un array de arrays, para imprimirlo en pantalla adecuadamente será necesario utilizar la función **print_r($mapa)** que nos permitirá imprimir toda la cadena de variables y matrices contenida como resultado del proceso de parseo.

Para no perder la perspectiva de lo que es capaz de reconocer el parser, el canal de sindicación que se utilizará para la siguiente prueba será el proporcionado en la página http://www.mblazquez.es/docs/atom.xml Se trata de un canal de sindicación de prueba en formato Atom cuyo único contenido es la descripción de los elementos y etiquetas del formato Atom, de forma tal que cuando se imprima la variable $mapa, podamos leer fácilmente qué contenidos han sido reconocidos.

A continuación se muestra el código del archivo XML que conforma el canal de sindicación de prueba que se va a utilizar, disponible en http://www.mblazquez.es/docs/atom.xml:

Canal de sindicación de prueba que se utilizará para comprobar la función de parseo

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">

    <id>identificador del canal</id>
    <updated>fecha de actualización</updated>
    <title type="text">título</title>
    <subtitle type="html">subtítulo o descripción del canal</subtitle>
    <link href="url del canal"/>

    <author>
        <name>nombre y apellidos del autor del canal</name>
        <uri>url del perfil del autor del canal</uri>
        <email>correo electrónico del autor del canal</email>
    </author>

    <contributor>
        <name>nombre y apellidos del colaborador del canal</name>
        <uri>url del perfil del colaborador del canal</uri>
        <email>correo electrónico del colaborador del canal</email>
    </contributor>

    <generator uri="url del programa" version="version del programa">N. Gen</generator>
    <rights>derechos del canal</rights>

    <entry>
        <id>identificador de la entrada</id>
        <published>fecha de publicación</published>
        <updated>fecha de actualización</updated>
        <category term="categoría temática"/>
```

```
        <link href="url de la entrada"/>
        <title type="text">título de la entrada</title>
        <content type="html">contenido de la entrada</content>
        <summary>resumen o sumario de la entrada</summary>

        <author>
            <name>nombre y apellidos del autor</name>
            <uri>url del perfil del autor</uri>
            <email>correo electrónico del autor</email>
        </author>

    </entry>

</feed>
```

Como se puede comprobar cada etiqueta contiene la descripción de su contenido de forma tal que pueda ser fácilmente reconocible en los resultados de la prueba.

A continuación se muestra el código que se va a emplear en la prueba:

Ejemplo de impresión de la variable $mapa

```php
<?php

$feed = "http://www.mblazquez.es/docs/atom.xml";
$mapa = new SimpleXMLElement(file_get_contents($feed));
print_r($mapa);

?>
```

El resultado que se obtiene en su ejecución es el siguiente:

Resultados de la prueba de impresión de la variable $mapa

```
SimpleXMLElement Object (
    [id] => identificador del canal
    [updated] => fecha de actualización
    [title] => título
    [subtitle] => subtítulo o descripción del canal

    [link] => SimpleXMLElement Object (
            [@attributes] => Array (
                    [href] => url del canal ) )

    [author] => SimpleXMLElement Object (
            [name] => nombre y apellidos del autor del canal
            [uri] => url del perfil del autor del canal
            [email] => correo electrónico del autor del canal )

    [contributor] => SimpleXMLElement Object (
            [name] => nombre y apellidos del colaborador del canal
            [uri] => url del perfil del colaborador del canal
            [email] => correo electrónico del colaborador del canal )
```

```
[generator] => nombre del programa generador del canal
[icon] => url del icono del canal
[logo] => url del logo del canal
[rights] => derechos del canal

[entry] => SimpleXMLElement Object (
        [id] => identificador de la entrada
        [published] => fecha de publicación
        [updated] => fecha de actualización

        [category] => SimpleXMLElement Object (
                [@attributes] => Array (
                        [term] => categoría temática ) )

        [link] => SimpleXMLElement Object (
                [@attributes] => Array (
                        [href] => url de la entrada ) )

        [title] => título de la entrada
        [content] => contenido de la entrada
        [summary] => resumen o sumario de la entrada

        [author] => SimpleXMLElement Object (
                [name] => nombre y apellidos del autor
                [uri] => url del perfil del autor
                [email] => correo electrónico del autor )

        [contributor] => SimpleXMLElement Object (
                [name] => nombre y apellidos del colaborador
                [uri] => url del perfil del colaborador
                [email] => correo electrónico del colaborador ) ) )
```

Como puede observarse, la función **new simpleXMLElement()** reconoce la estructura del canal de sindicación siempre y cuando esté basado en XML, tal y como es el caso, generando un array de arrays con todos los elementos y etiquetas utilizadas en dicho canal de sindicación de forma tal que puedan ser manejadas como objetos desde PHP. Esto significa acceder a ellas mediante el lenguaje de selección XPath.

3. Aprendiendo a seleccionar los elementos con XPath

XPath es un lenguaje de selección de nodos y elementos de las etiquetas de los archivos XML. No obstante también tiene sus aplicaciones en materia de recuperación de información por cuanto nos permite filtrar, seleccionar y extraer la información y contenidos de los elementos que constituyen el canal de sindicación.

DIABD. Módulo de sindicación de contenidos

Tarea 2: Lectura de canales de sindicación

Profesor Manuel Blázquez Ochando

¿Cómo seleccionar y acceder a los elementos de un canal de sindicación?

Lo primero a tener en cuenta en XPath es la sintaxis básica de selección. Todos los formatos de sindicación anidan las etiquetas utilizadas para describir tanto el canal de sindicación como las entradas disponibles en el mismo. Tal jerarquización se refleja en XPath mediante el empleo de barras oblicuas, también denominadas slash (**/**), tal y como se muestra en la siguiente tabla:

Ejemplo de selección del título del canal

Estructura XML

```
<feed>
    <title></title>
</feed>
```

Selección XPath del título del canal

```
//feed/title
```

La doble barra oblicua (**//**) se utiliza para efectuar selecciones absolutas dentro del archivo XML. De esta forma, solo seleccionará aquella estructura que contenga la etiqueta title dentro de la etiqueta feed.

En el caso de un título contenido en una entrada del canal de sindicación, esta estructura se complica:

Ejemplo de selección del título de una entrada del canal

Estructura XML

```
<feed>
    <title></title>
    <entry>
        <title></title>
    </entry>
</feed>
```

Selección XPath del título del canal

```
//feed/entry/title
ó
//entry/title
```

Como se muestra en la tabla anterior la sentencia XPath toma una ruta diferente para seleccionar el elemento **title** dentro de **entry** y a su vez dentro de **feed**. También es posible su selección marcando una ruta absoluta desde **entry**, dado que es la única ruta de acceso, obviando la etiqueta **feed**.

Pero también puede darse el caso de seleccionar el atributo de un elemento determinado. Para ello el lenguaje XPath emplea el signo reservado @ que precede al nombre del atributo para designarlo correctamente.

Ejemplo de selección de un atributo term

Estructura XML

```
<feed>
    <title></title>
    <entry>
        <title></title>
        <category term='prueba'>
    </entry>
</feed>
```

Selección XPath del título del canal

```
//feed/entry/title/@term
```

4. Aplicando XPath al PARSER

Si bien las apreciaciones señaladas para seleccionar los elementos de un archivo XML mediante XPath son correctas, para aplicarlas correctamente sobre un parser **new simpleXMLElement()** se producen ciertos cambios que hay que tener en cuenta.

Por un lado la selección de los elementos se efectúa principalmente mediante sintaxis DOM de selección de los canales, pero también pueden utilizarse las consultas XPath anteriormente mencionadas. Por ejemplo:

DOM y XPath para seleccionar elementos y atributos

Estructura XML

```
<feed>
    <title></title>
    <entry>
        <link href='http://www.mblazquez.es'/>
        <title></title>
    </entry>
</feed>
```

Selección DOM y XPath de la URL de una entrada

```
$link = $mapa->entry->link->xpath('@href');
```

- En **azul** instrucción DOM
- En **verde** instrucción XPath

Como se puede apreciar la variable **$link** es igual a la variable **$mapa** seguida de flechas que indican los nodos correspondientes a la estructura, a saber **entry->link**, **(En XPath esto se expresaría como entry/link)**. Pero para poder acceder al contenido del elemento **link**, es decir, el atributo **href**, sí se requiere el empleo de **xpath('@href')**.

Este ejemplo debe alertar de que pueden combinarse los métodos de selección propios del parser mediante DOM y la sintaxis XPath, logrando una instrucción de acceso a los contenidos mucho más precisa e intuitiva.

De hecho el mapeado efectuado sobre el canal de sindicación, tal y como se ha comentado se efectúa mediante las funciones de DOM integradas por PHP y que parten necesariamente de la variable **$mapa**, por lo que resulta necesario seleccionar a partir de este punto los nodos tal y como se acaba de mostrar.

El resultado de aplicar todos estos conceptos es el siguiente programa PARSER:

Ejemplo de PARSER ATOM

```
$feed = "URL";
$mapa = new SimpleXMLElement(file_get_contents($feed));

$title = $mapa->title;
$author = $mapa->author->name;
$contributor = $mapa->contributor->name;
$updated = $mapa->updated;
$link = $mapa->link->xpath("@href");
$summary = $mapa->summary;
$category = $mapa->category->xpath("@term");
```

En esta tabla puede comprobarse como el proceso de selección de los nodos corre a cargo del método de DOM, pero los atributos de los elementos son seleccionados mediante XPath. En instrucciones mucho más complejas, XPath se emplea con mayor preponderancia.

Tarea y actividades:

El objetivo de la presente tarea es probar los siguientes programas, disponibles en el campus virtual en el archivo comprimido: **parser.zip**

- prueba-mapa.php
- prueba-parser-atom.php
- prueba-parser-rss2.php

DIABD. Módulo de sindicación de contenidos

Tarea 2: Lectura de canales de sindicación

Profesor Manuel Blázquez Ochando

Tarea en el archivo: prueba-mapa.php
1. Crear una carpeta **parser** en el directorio raíz de joomla y asignar permisos 777. 2. Subir a dicha carpeta el archivo **prueba-mapa.php** 3. Ejecutar el programa en el navegador 4. Pegar el resultado en el siguiente espacio en blanco

Tarea en el archivo: prueba-parser-atom.php
1. Subir a la carpeta **parser** el archivo **prueba-parser-atom.php** 2. Ejecutar el programa en el navegador. 3. Buscar 1 canal de sindicación en formato ATOM anotarlo en esta tabla y probarlo en el programa. 4. Hacer impresión de pantalla del resultado obtenido y pegarlo en el siguiente espacio en blanco.

DIABD. Módulo de sindicación de contenidos

Tarea 2: Lectura de canales de sindicación

Profesor Manuel Blázquez Ochando

Tarea en el archivo: prueba-parser-rss2.php

1. Subir a la carpeta **parser** el archivo **prueba-parser-rss2.php**
2. Ejecutar el programa en el navegador.
3. Buscar 1 canal de sindicación en formato RSS2.0 anotarlo en la tabla y probarlo en el programa.
4. Hacer impresión de pantalla del resultado obtenido y pegarlo en el siguiente espacio en blanco.

5. Modificar el programa prueba-parser-rss2.php, para que sea capaz de seleccionar e imprimir en pantalla los siguientes elementos correspondientes a <item>:

- <link></link>
- <guid></guid>
- <category></category>
- <pubDate></pubDate>
- <description></description>

6. Escribir en el siguiente espacio en blanco las modificaciones efectuadas en el archivo.

7. Probar el PARSER con un canal de sindicación RSS2.0, anotar su URL e imprimir pantalla a continuación.