Guidelines:Documentation_es

Es muy importante que a la hora de documentar con DocBook sigais las normas de estilo de desarrollo, ya que al final documentar y desarrollar siguen una serie de normas basicas de estilo muy similares. En el caso del XML se refiere al uso de bloques de las marcas o tags mientras que en el desarrollo tradicional (C/C++, PHP, etc) está mas orientado a nomenclaturas de variables, etc.

Asegúrese de haber leido antes las normas de estilo

Introducción

DocBook es básicamente un XML con un DTD asociado y muy concreto que regula el formato. Ademas del XML donde vamos a “escribir” nuestro documento, hay asociadas unas plantillas de estilos que son las que finalmente dan el aspecto “visual” del texto que vamos a generar. Podriamos decir que hay dos procesos principales a la hora de trabajar con DocBook.

- Editar el XML.
- Generar el resultado final mediante docbook2pdf, docbook2html, docbook2man exportandolo a PDF, HTML o las paginas MAN.

DocBook tiene dos ventajas clave que son las que justifican que editar manualmente un XML con un juego de tags inmenso merezca la pena:

> 1. Es XML, texto ascii, por lo que podemos utilizar un sistema de control de version para llevar la documentación, algo esencial para poder trabajar en equipo.
> 2. Genera documentación en formato PDF, HTML y MAN a partir de la misma fuente. Esto muchas tiene ventajas y nos permitirá ahorrar tiempo.
> 3. Es un lenguaje estricto que produce documentación siempre de la misma manera. La gente está ya acostumbrada a leer documentos DocBook. Además al ser tan estricto obliga a estructurar los documentos desde un principio.
> 4. Es un estándar GNU usado en multitud de proyectos.

La curva de aprendizaje de DocBook es bastante costosa, pero los beneficios compensan los costes, de esta forma empezaremos a trabajar con DocBook - como norma general- , para cualquier documentación de proyecto en la que sea necesario colaborar con diferentes personas. A continuación se describirán una serie de consejos muy basicos para poder trabajar con él, editarlo y hacer textos basicos. Se han detallado enlaces con varios.

Editando DocBook

Utiliza un editor cualquiera, preferiblemente uno que tenga coloreado sintáctico. La estructura del XML inicial seria como sigue:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook V3.1//EN"
      "/usr/share/xml/docbook/schema/dtd/4.4/docbookx.dtd" >
<book>
    <chapter>
       <title>Capitulo 1</title>
       <sect1>
          <title>Seccion 1</title>
          <para>
          Erase una vez un lobo feroz que comía usuarios de Windows...
          </para>
       </sect1>
       <sect1>
          <title>Seccion 2</title>
          <para>
              Hace mucho, mucho tiempo, antes de que GNOME y KDE pusieran de
rodillas a CDE...
          </para>
       </sect1>
    </chapter>
</book>
```

Este es un pequeño ejemplo de un documento DocBook. En el lo primero que podemos apreciar es su cabecera XML. Esta define el DTD (usando una referencia a un fichero en el filesystem, previamente habria que comprobar si existe o ajustar el enlace a un DTD que exista.

Lo siguiente es el tag <book> que define que vamos a iniciar un “libro”. Hay otro tipo llamado <article> que funciona de una forma ligeramente diferente. Lo mejor es que cojas tu editor, cortes peges y pruebes a generar o “renderizar” el texto final.

Renderizando el texto DocBook

Te harán falta los siguientes paquetes debian (es probable que me deje alguno):

docbook docbook-dsssl docbook-utils docbook-xml docbook-xsl openjade

El paso para “generar” la documentacion sera simplemente usar el comando con el XML donde has generado tu texto en formato DocBook/XML:

```
$ docbook2pdf test.xml
Using catalogs: /etc/sgml/catalog
Using stylesheet: /usr/share/docbook-utils/docbook-utils.dsl#print
Working on: /tmp/test.xml
Done.
```

Si has cometido algun error en la edición del texto te lo indicará. DocBook es MUY estricto. Si el comando ha tenido éxito tendras un PDF con tu texto. Puedes hacer la prueba con docbook2html o docbook2man ya que funcionan de una forma muy similar.

Insertando contenido conflictivo con XML

Un problema típico a la hora de escribir XML es que el propio texto que queremos introducir DENTRO del XML contenga a su vez, tags similares a las del XML o incluso XML en si, por ejemplo, cuando queremos describir un ejemplo de uso. Para ello existe un tag especial que indica “este contenido se escribe tal cual, sin interpretar”. Este tag especial, se usa de la siguiente manera:

```
<![CDATA[__TEXTO__]]>
```

Donde TEXTO es el texto XML, con tags o caracteres “prohibidos”, como “&”, “>” ó “<”.

Un ejemplo de esto seria por ejemplo:

<![CDATA[

```
 El tag  sirve en HTML para indicar negrita.
 El equivalente en DocBook es la cursiva, que es <emphasis></emphasis>
```

]]> </code>

Haciendo una lista de elementos

Lista sencilla

Esto seria una lista sencilla

```
<simplelist type='inline'>
  <member>A</member>
  <member>B</member>
  <member>C</member>
</simplelist>
```

Lista con puntos negros o botones

Esto seria una lista con puntos negros típica. El punto negro se puede cambiar por otros tipos de grafico.

```
<para>
    La administración de Babel se divide en los siguientes apartados:
</para>
<itemizedlist>
    <listitem>
        <para>Usuarios de Babel</para>
    </listitem>
    <listitem>
        <para>Agentes</para>
    </listitem>
```

```
    <listitem>
        <para>Configuración del servidor</para>
    </listitem>
    <listitem>
        <para>Mantenimiento de la Base de Datos</para>
    </listitem>
</itemizedlist>
```

Lista numerada

Igual que la anterior pero en vez de itemizedlist sera orderedlist. Si queremos cambiar el tipo de numero (para numeros romanos, p.e) hay que anañdir el atributo numeration =“lowerroman” en el tag de orderedlist. Veamos el ejemplo:

```
<para>
    Los principales contribuyentes a dicho proyecto han sido:
</para>
<orderedlist numeration="lowerroman">
    <listitem>
        <para>Hal Computer Systems y O'Reilly & Associates, de
              1991 a 1994.
        </para>
    </listitem>
    <listitem>
        <para>El grupo Davenport, de 1994 a 1998.</para>
    </listitem>
        <listitem>
        <para>El grupo <acronym>OASIS</acronym> de 1998 hasta hoy.</para>
    </listitem>
</orderedlist>
```

Secciones, titulos, capitulos, subsecciones y demas

La estructura basica de un Book (libro) es diferente de la de un artículo, pero la gran mayoria de elementos son comunes. Vamos a describir la estructura básica:

- <chapter> - Capitulo de un libro, la sección mas importante.
- <title> - Titulo del capitulo.
- <para> - Parrafos de texto, generalmente se puede poner un parrafo de texto nada mas comenzar el capitulo para hacer una introduccion al mismo.
- <sect1> - Seccion primera. Debe contener un tag 'title' y contenidos dentro de un 'para'.
- <sect2> - Seccion segunda. Debe ir incluida dentro de una seccion primera. Contiene a su vez un 'title' y un 'para' y puede contener otras subsecciones, como sect3, sect4...

Veamos un ejemplo práctico sacado de la documentación de Babel Enterprise:

```
<chapter>
```

```
    <title>Administración de Babel</title>
    <para>La administración de babel se realiza desde la aplicación
Web.</para>
    <sect1>
        <title>Usuarios de Babel</title>
        <para>Algo que decir</para>
        <sect2>
            <title>Grupos de usuarios</title>
            <para>Mas quqe decir</para>
        </sect2>
    </sect1>
    <sect1>
        <title>Segundo título</title>
        <para>Más que decir</para>
    </sect1>
</chapter>
```

Macros e includes

Puedes importar un XML externo a modo de include con una sencilla sintaxis. Esto permite por ejemplo que secciones comunes de diferentes documentos compartan el mismo origen, por ejemplo, el apéndice de la licencia FDL. Basta con declarar en la cabecera una macro, definida por una palabra que luego sera referenciada como:

```
&macro;
```

Veamos un ejemplo:

Cabecera del XML, donde debemos definir la macro, en este caso se llamará *include_fdl*

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook V3.1//EN"
      "/usr/share/xml/docbook/schema/dtd/4.4/docbookx.dtd" [
<!ENTITY babel_version "v1.0">
<!ENTITY babel "Babel Enterprise">
<!ENTITY include_fdl SYSTEM "fdl.xml">
]>
```

Y para “ejecutar” esa macro, es decir, sustituir el texto, bastará con poner en el texto la llamada a la macro, esto es:

```
&include_fdl;
```

Para utilizar macros de sustitucion de texto es igual de sencillo. En el ejemplo anterior se han declarado dos variables, 'babel_version' y 'babel' que seran sustituidas por su valor cuando en el texto se escriba

&babel_version;

Cómo usar negrita en DocBook

Identifica la hoja de estilos que estás usando en el sistema. Docbook generalmetne te dice cual estás usando, por ejemplo:

```
docbook2pdf babel.xml
Using catalogs: /etc/sgml/catalog
Using stylesheet: /usr/share/docbook-utils/docbook-utils.dsl#print
Working on: /datos/babel/trunk/babel_doc/en/babel.xml
```

Edita tu hoja de estilo, es este caso “/usr/share/docbook-utils/docbook-utils.dsl”, y añade este código al final, justo debajo de otra definición de “element”:

```
(element emphasis
  (if (equal? (attribute-string "role") "bold")
      (make sequence
        font-weight: 'bold
        (process-children))
      (make sequence
        font-posture: 'italic
        (process-children))
  )
)
```

Ahora para usar negrita, solo tienes que usar el siguiente tag:

<emphasis role ='bold'>bold text</emphasis> in your Docbook code.

Estilo de documentación con DocBook

Como se ha dicho ya en la introducción es importante que sigamos una serie de recomendaciones para hacer mas fácil el trabajo en grupo, entre las que podemos destacar:

- Tabular cada tag hijo y su contenido. Ejemplo:

```
<chapter>
    <title>Administración de Babel</title>
    <para>
        La administración de babel se realiza desde la aplicación Web.
    </para>
    <sect1>
        <title>Usuarios de Babel</title>
        <para>
            Algo que decir
        </para>
        <sect2>
            <title>Grupos de usuarios</title>
            <para>
```

```
                Mas que decir
            </para>
        </sect2>
    </sect1>
    <sect1>
        <title>Segundo título</title>
        <para>
            Más que decir
        </para>
    </sect1>
</chapter>
```

- El contenido de los tags de párrafo (<para>) debe de estar tabulado y en una nueva linea. El ancho de su lineas no puede ser superior a 80 columnas, continuando en una nueva linea (alineada a la anterior) si fuese necesario. **No te fies de los editores de texto**. Ejemplo correcto:

```
<para>
    Este sería un párrafo bastante largo cuyo ancho será superior a 80
columnas. Si
    siguiesemos escribiendo en él, lo haríamos en una nueva linea, alineada
al comienzo
    de la linea anterior. Este ejemplo es prueba de ello.
</para>
```

Ejemplo **INCORRECTO**:

```
<para>Si este texto fuese para la documentación de los programas de Artica
no serviría porque está escrito en una linea muy larga y por tanto no sería
correcto.  Además, el <para> debería de estar cerrado en la linea inferior.
</para>
```

- Cada tag hijo debe de estar en una nueva linea. Lo siguiente es **INCORRECTO**

```
<chapter><title>Titulo</title>
    <para>parrafo que va en este capitulo</para>
</chapter>
```

- La excepción a la regla anterior son los tags *inline*, es decir, aquellos que afectan a partes de un texto o párrafo. El siguiente tag <acronym> es correcto:

```
<para>El grupo <acronym>OASIS</acronym> de 1998 hasta hoy.</para>
```

- Es preferible separar los textos de los tags en bloques medios y grandes, por ejemplo:

```
<para>
     gran bloque de texto... gran bloque de texto... gran bloque de texto...
     gran bloque de texto... gran bloque de texto... gran bloque de texto...
     gran bloque de texto...
</para>
```

- Sé práctico escribiendo los párrafos, se prefieren varios párrafos cortos a uno largo y

complicado.
- No crear el texto basado en plantillas propias, sino pensando en el output que generará en plantillas estandares. Cualquier mejora visual ha de ser hecha pensando en la compatibilidad primero y en la mejora visual después.

Reglas de oro al escribir documentación

Hay unas reglas de oro para que la documentación sea sencilla y fácil de leer.

Regla de oro 1

- Limita cada frase a menos de 25 palabras.

Ejemplo:

Una frase así:

> *Bajo condiciones normales, el kernel no siempre escribe los datos a disco inmediatamente, si no que los guarda en un buffer de memoria para periódicamente escribirlos a disco y así agilizar las operaciones*.

Se reescribiría a:

> *Generalmente, el kernel almacena el dato en memoria antes de escribirlo periódicamente a disco*.

Regla de oro 2

- Cada párrafo es un tema
- Cada frase una idea

Consejo: Planifica el orden de los párrafos antes de empezar a escribir. Decide de qué tema quieres hablar en cada párrafo.

Regla de oro 3

- Usa ejemplos explícitos para demostrar cómo funciona una aplicación. Proporciona instrucciones en vez de descripciones.

Ejemplo:

Un texto así:

> *Hay una caja de texto que se puede usar para buscar la definición de una palabra.*

Estaría mejor redactado así:

> *Para buscar la definición de una palabra, escribala en la caja de texto y pulse el botón "Buscar".*

Regla de oro 4

- Escribe de un modo neutral

Ejemplo:

> *La aplicación es un pequeño capturador de pantallas.*

Es mejor así:

> *Puede usar la aplicación para hacer una captura de pantalla.*

Referencias

Documentación útil de Docbook

- DocBook. Manual basico.
- DocBook. Manual completo.
- DocBook. Más manuales
- Mini-guia de referencia.
- Guia de referencia de hojas de estilo para DocBook