

UD07 – Generación de interfaces a partir de documentos XML

Desarrollo de interfaces

2º DAM

Objetivos

- Reconocer las ventajas de generar interfaces de usuario a partir de su descripción XML.
- Generar la descripción del interfaz en XML usando un editor gráfico.
- Analizar el documento XML generado.
- Modificar el documento XML.
- Generar el código correspondiente al interfaz a partir del documento XML.
- Programar una aplicación que incluya el interfaz generado.

Lenguajes de descripción de interfaces basados en XML

Este lenguaje en la actualidad es muy importante puesto que presenta diversos usos entre los que destacan:

- **Intercambio de información entre aplicaciones:** al almacenar información mediante documentos de texto plano no se requiere software especial.
- **Computación distribuida:** se trata de la posibilidad de utilizar XML para intercambiar información entre diferentes ordenadores a través de redes.
- **Información empresarial:** este lenguaje tiene cada vez más importancia para generar interfaces empresariales gracias a la facilidad para estructurar los datos de la forma más apropiada para cada empresa.

Lenguajes de descripción de interfaces basados en XML

- XML es un lenguaje de marcado que permite estructurar datos mediante etiquetas y atributos.
- Aunque ha surgido competencia (como JSON), XML sigue siendo utilizado para almacenar y transportar información estructurada.
- Se emplea en bases de datos, archivos de configuración, intercambio de datos entre sistemas y más.
- Muchos estándares y protocolos utilizan XML. Por ejemplo:
 - ✓ SOAP (Simple Object Access Protocol) para servicios web.
 - ✓ RSS (Really Simple Syndication) para sindicación de contenidos.
 - ✓ SVG (Scalable Vector Graphics) para gráficos vectoriales.
- Estos estándares garantizan la interoperabilidad y la consistencia en diferentes sistemas.

Lenguajes de descripción de interfaces basados en XML

- Algunos competidores de XML son:
 - ✓ **JSON (JavaScript Object Notation):** Más ligero y fácil de leer, se utiliza ampliamente en aplicaciones web y APIs.
 - ✓ **YAML (Yet Another Markup Language):** Diseñado para ser legible por humanos, se usa en archivos de configuración.
- La elección entre XML y sus alternativas depende del contexto y las necesidades específicas.
- A pesar de sus competidores, sigue siendo importante en diferentes ámbitos:
 - ✓ **Intercambio de Datos:** Muchos sistemas aún utilizan XML para compartir información.
 - ✓ **Documentación Técnica:** Escribir manuales, especificaciones y documentación técnica.
 - ✓ **Lenguajes de Marcado Personalizados:** Empresas y organizaciones crean sus propios lenguajes basados en XML para necesidades específicas.
- Al ser un metalenguaje, puede ser empleado para definir otros lenguajes.

Lenguaje de descripción de interfaces basado en XML



XML: eXtensible Markup Language



XHTML: eXtensible HyperText Markup Language



GML: Geography Markup Language



MathML: Mathematical Markup Language



RSS: Really Simple Syndication



XSLT: eXtensible Stylesheet Language for Transformations




SVG: Scalable Vector Graphics

XHTML

- eXtensible HyperText Markup Language (lenguaje de marcado de hipertexto extensible). Es un lenguaje derivado de XML similar a HTML, pero con algunas diferencias que lo hacen más robusto y aconsejable para la modelación de páginas web.
- Los documentos XHTML tienen que cumplir:
 - ✓ `<!DOCTYPE>` es obligatorio.
 - ✓ El atributo `xmlns` en `<html>` es obligatorio.
 - ✓ `<html>`, `<head>`, `<title>` y `<body>` son obligatorios.
 - ✓ Los elementos siempre deben estar correctamente anidados.
 - ✓ Los elementos siempre deben estar cerrados.
 - ✓ Los elementos siempre deben estar en minúsculas.
 - ✓ Los nombres de los atributos siempre deben estar en minúsculas.
 - ✓ Los valores de los atributos siempre se deben citar.
 - ✓ La minimización de atributos está prohibida.



Ejemplo de código XHTML




```
<!DOCTYPE html>
<html xmlns="https://www.w3.org/1999/xhtml">
<head>
    <title>Nuevo documento XHTML</title>
</head>
<body>
    <h1>Cabecera principal del documento</h1>
    <p>Primer párrafo</p>
    <h2>Cabecera secundaria</h2>
    <p>Otro párrafo con contenido</p>
</body>
</html>
```


GML (Geography Markup Language)

Un documento GML puede recibir un formato que define el tipo de texto que es (títulos, párrafos, listas...). Este tipo de documentos están compuestos de marcas precedidas de doble punto (:).

- :h1. Generación de interfaces a partir de documentos XML
- :p. Lenguajes de descripción de interfaces basados en XML
- :ol
- :li.GML
- :li.MathML
- :li. RSS
- :li.SVG
- :li.XHTML
- :eol.

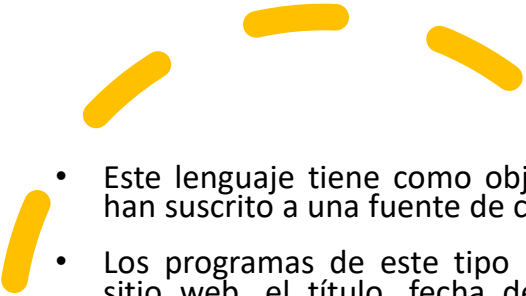


MathML (Mathematical Markup Language)

Este lenguaje se usa junto con el lenguaje XHTML con el objetivo de intercambiar información entre programas de tipo matemático.

En el siguiente ejemplo se muestra la fórmula matemática $a^2 + b^2 = c^2$ escrita en lenguaje MathML.

```
<math>
  <mrow>
    <mi>a</mi>
    <mn>2</mn>
    <mo>+</mo>
    <mi>b</mi>
    <mn>2</mn>
    <mo>=</mo>
    <mi>c</mi>
    <mn>2</mn>
  </mrow>
</math>
```

- 
- Este lenguaje tiene como objetivo difundir información a usuarios que se han suscrito a una fuente de contenidos actualizada frecuentemente.
 - Los programas de este tipo suelen estar compuestos por novedades del sitio web, el título, fecha de publicación o descripción. En el siguiente ejemplo se muestra una noticia publicada en lenguaje RSS.

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<rss version="2.0">
```

```
<channel>
```

```
<title>Última hora</title>
```

```
<description>Noticia importante </description>
```

```
<link>https://elpais.com/ultimas noticias/</link>
```

```
<lastBuildDate>Mon, 06 Jan 2022 00:10:00 </lastBuildDate>
```

```
<pubDate>Mon, 06, Jan 2020 16:20:00 +0000
```

```
</pubDate>
```

```
</channel>
```


```
</rss>
```



RSS (Really Simple Syndication)

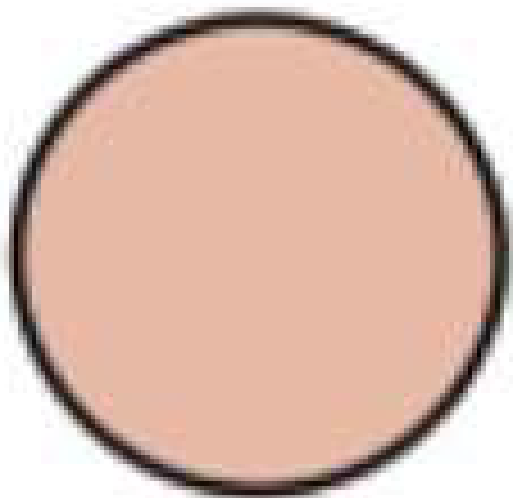
XSLT (eXtensible Stylesheet Language for Transformations)

- Es un estándar del W3C que permite transformar documentos XML en otros formatos, incluso aquellos que no son XML.
- Las hojas de estilo XSLT aplican reglas de plantilla al documento fuente para realizar la transformación.
- Con XSLT se pueden agregar o eliminar elementos y atributos desde o hacia el archivo de salida.



SVG (Scalable Vector Graphics)

- Este lenguaje permite representar elementos geométricos vectoriales, imágenes de mapa de bits y texto.
- El lenguaje SVG es la especificación que define cómo se deben crear y representar las imágenes SVG. Mientras que SVG es el formato de archivo en sí mismo.
- Está basado en XML, que es un lenguaje de marcado utilizado para estructurar datos en forma de etiquetas y atributos.
- El lenguaje SVG describe los elementos gráficos (líneas, círculos, rectángulos, texto, etc.) y sus propiedades (color, tamaño, posición, etc.) utilizando sintaxis XML.



Resultado del ejemplo

Ejemplo de SVG

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<svg height="100" width="100">
```

```
  <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3"  
  fill="darksalmon" />
```

Lo siento, tu navegador no es compatible con svg.

```
</svg>
```

```
</body>
```

```
</html>
```

Ejemplo SVG

```
<svg width="100" height="100">  
  <ellipse cx="60" cy="60" rx="40" ry="20" fill="yellowgreen"/>  
</svg>
```



Ejemplo elipse

```
<svg width="120" height="128">  
  <rect x="0" y="0" width="120" height="60" fill="lightseagreen"/>  
</svg>
```

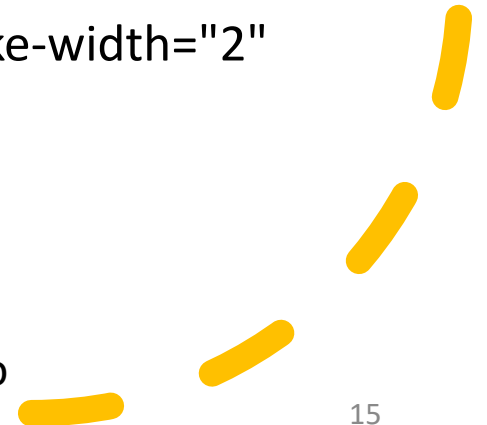


Ejemplo rectángulo

```
<svg width="120" height="128">  
  <polygon fill="SteelBlue" stroke="black" stroke-width="2" points="15,90 45,30 75,90"/>  
</svg>
```



Ejemplo triángulo



XAML (Extensible Application Markup Language)

- Este lenguaje está basado en XML y permite realizar una descripción gráfica de las interfaces de los distintos usuarios (desde el punto de vista gráfico).
- XAML es un lenguaje que se puede aplicar al desarrollo de interfaces para escritorio y, además, suele utilizarse para web. Existen una serie de editores que permiten incorporar herramientas de edición y analizadores sintácticos, como pueden ser Visual Studio y Blend, entre otros.
- Uno de los principales objetivos que se pretende en el diseño de interfaces que están basadas en XAML es separar totalmente las capas de presentación de la capa lógica para conseguir evitar que se mezclen aquellos elementos que pertenezcan a distintas capas. Esto podría afectar a la distribución modular de la aplicación y al acoplamiento de esta.

XAML (Extensible Application Markup Language)

- XAML es un lenguaje basado en XML para cerrar e inicializar objetos .NET con relaciones jerárquicas.
- Entre las principales características de XAML se puede señalar que cada elemento gráfico se define mediante una etiqueta de apertura y otra de cierre, además de por un conjunto de atributos que definirán el aspecto y comportamiento de este.
- XAML cuenta con bastantes ventajas respecto a sus competidores, sobre todo al permitir desarrollar distintas interfaces mediante su asociación con .Net. En este lenguaje, tanto las etiquetas como los atributos se corresponden de forma directa con otros elementos que pertenecen al lenguaje .Net.

Descripción de la sintaxis en XAML

Las etiquetas XAML definen los distintos elementos pertenecientes a la interfaz y cuentan con una serie de atributos:

- ✓ **Atributo Name:** único identificador del elemento. Bastante útil cuando se precisa hacer referencias en el código (x:Name).
- ✓ **Atributo Key:** puede contar con un identificador para los distintos elementos definidos en el diccionario (x:Key).
- ✓ **{Binding}:** se refiere al elemento que está definido dentro de un valor de un atributo, permitiendo definir un enlace a una fuente de datos, un fichero, una base de datos, etcétera.
- ✓ **{StaticResource}:** similar al anterior, pero, en este caso, hace referencia a un elemento que está definido en el diccionario de recursos.
- ✓ **{Null}:** representa el valor nulo.

El documento XML.

Análisis y edición

- Un archivo XML es un fichero en formato texto, que contiene etiquetas para identificar los elementos y datos que componen el documento. La primera línea del fichero debe ser:

`<?xml version="1.0" encoding="UTF-8"?>`

- ✓ Versión: indica la versión de XML que se está utilizando.
 - ✓ Encoding: especifica el juego de caracteres con el que se ha codificado el documento.
- El resto del documento XML se escribirá con etiquetas, y siempre hay que abrirlas y cerrarlas. La estructura de etiquetas de apertura y cierre siempre sigue un patrón como este:
 - `<etiqueta> </etiqueta>`

El documento XML.

Análisis y edición

- Un documento XML tiene una estructura anidada de manera jerárquica.
- El conjunto formado por las etiquetas (apertura y finalización) y el contenido se conoce como elemento.
- Es obligatorio cerrar todas las etiquetas, y si un elemento tiene vinculados otros elementos (hijos), elementos que descienden de él, se deben cerrar las etiquetas de los hijos antes de poder cerrar la etiqueta del padre.



Etiquetas

- Las etiquetas XML son marcas que sirven para separar un contenido de otro en el documento.
- Una etiqueta empieza con el carácter “<”, continúa un nombre identificativo, y termina con el carácter “>”.
- El nombre de la etiqueta debe empezar por una letra, y continuar con letras, dígitos, guiones, rayas, punto o dos puntos.
- Existen tres tipos de etiquetas:
 - ✓ **Start-tag:** etiquetas de apertura.
`<etiqueta>`
 - ✓ **End-Tag:** etiquetas de cierre, similar a las de apertura, pero comienzan por “/”.
`</etiqueta>`
 - ✓ **Empty-Tag:** etiquetas vacías, que terminan por “/”.
`<etiqueta_vacia />`

Atributos

- Un atributo es un componente de las etiquetas estructurado de manera nombre=valor.
- Se puede encontrar en las etiquetas de apertura o en las etiquetas vacías, pero no en las de cierre.
- Hay que tener en cuenta que en una misma etiqueta no pueden existir dos atributos con el mismo nombre, siendo siempre todos los atributos de un elemento únicos.

- Por ejemplo:

```
<cliente nombre="Fran" apellido1="Lifante" apellido2="Rivera" />
```

- Ejemplo de uso incorrecto:

```
<cliente nombre="Fran" apellido="Lifante" apellido="Rivera">
```

- El error radica en que tenemos el atributo apellido repetido.

Valores

- El atributo de un elemento XML proporciona información acerca del elemento, es decir, sirven para definir las propiedades de los elementos.
- La estructura de un atributo XML es siempre un par de nombre-valor.

```
<biblioteca>  
  <texto tipo_texto="libro" titulo="El libro de las ilusiones" editorial= "Anagrama">  
    <tipo>  
      <libro isbn="8433969978" edicion="1" paginas="347"/>  
    </tipo>  
    <autor nombre="Paul Auster"/>  
  </texto>  
</biblioteca >
```

En el ejemplo observamos que los elementos aparecen coloreados en rojo, los nombres de los atributos en negro y sus valores en azul.

Eventos

- Los eventos proporcionan un mecanismo adecuado para tratar las diferentes formas de interacción entre el usuario y la aplicación, por ejemplo, cuando el usuario presiona una tecla o pulsa con el puntero del ratón.
- En algunas ocasiones, ante la llegada de un evento, nos interesará tratarlo.
- En cambio, otras veces no será necesario el tratamiento del evento con ninguna acción.

Eventos

Algunos de los eventos más comunes que se pueden producir en una aplicación como interacción con las interfaces gráficas son:

- ✓ **MouseMove:** evento producido al mover el ratón por encima de un control.
- ✓ **MouseDown:** este evento se produce al pulsar cualquier botón del ratón.
- ✓ **Change:** se produce al cambiar el contenido del control.
- ✓ **Click:** uno de los eventos más comunes se produce al hacer clic con el botón izquierdo del ratón sobre el control.
- ✓ **GetFocus:** este evento se activa cuando el control recibe el enfoque, es decir, cuando se activa el control en tiempo de ejecución para introducir datos o realizar alguna operación.
- ✓ **LostFocus:** es lo contrario del evento anterior, se activa cuando el control pierde el enfoque, es decir, se pasa a otro control para seguir introduciendo datos.

Herramientas libres y propietarias para la creación de interfaces de usuario multiplataforma

- Una de las características fundamentales que presentan los editores de XML es la sencillez para escribir código resaltando la sintaxis, insertando elementos y estructuras de XML de uso común a través de la función de autocompletado.
- A continuación, podréis ver las principales características de los editores más utilizados para el lenguaje XML.

Notepad++

- Este editor reconoce la sintaxis de múltiples lenguajes de programación.
- Es gratuito, está disponible para Linux y Windows y su código fuente se puede descargar.
- Ha tenido un gran éxito entre los desarrolladores por las características que ofrece, lo ligero que es y su rapidez.
- Su interfaz puede ser personalizada y posee diversos plugins entre los que destaca el llamado XML Tools que añade un menú con opciones específicas como validar un documento XML con su DTD (Document Type Definition).

Visual Studio Code

- Es uno de los editores de código fuente más utilizados.
- Es compatible con varios lenguajes de programación y está disponible para Windows, Linux y macOS.
- Permite el resaltado de sintaxis, la finalización inteligente de código, tiene interfaz personalizable y es gratuito.
- Ofrece el servicio Live Share, extensión que permite compartir el código con otro programador, de forma que se puede colaborar en tiempo real.

Generación de código para diferentes plataformas

- Uno de los retos más importantes a los que se enfrentan los desarrolladores es el intercambio de datos entre sistemas incompatibles.
- El intercambio de datos con XML reduce en gran medida la dificultad ante la que se enfrentan los desarrolladores en cuanto al intercambio de datos entre sistemas incompatibles.
- Gracias a su portabilidad, XML se ha convertido en una de las tecnologías más utilizadas como base para el almacenaje de contenidos, como modelo de representación de metadatos y como medio de intercambio de contenidos.
- Esto es debido a que XML es un lenguaje independiente de la plataforma, lo que significa que cualquier programa diseñado para lenguaje XML puede leer y procesar los datos XML independientemente del hardware o del sistema operativo

La tecnología XML se basa en tres pilares fundamentales que permiten generar código para diferentes plataformas

- ✓ **Representación de metadatos:** XML es ampliamente utilizado para representar metadatos debido a su estructura jerárquica y su capacidad para definir etiquetas personalizadas y atributos. Esta capacidad es esencial para sistemas de indexación y recuperación de información, ya que permite etiquetar y organizar datos de manera estructurada, facilitando la búsqueda y clasificación de información dentro de grandes conjuntos de datos.
- ✓ **Medio de intercambio de contenidos:** XML es un formato de intercambio de datos comúnmente utilizado debido a su legibilidad, flexibilidad y facilidad para ser procesado por diferentes sistemas y plataformas. Los documentos XML pueden ser fácilmente transformados y procesados para diversos fines, como integración en sistemas de bases de datos, visualización en sitios web, intercambio de mensajes entre aplicaciones a través de servicios web y muchas otras aplicaciones.
- ✓ **Almacenamiento de contenidos:** La demanda de bases de datos XML nativas está en aumento debido a la necesidad de almacenar y gestionar documentos XML de manera eficiente y sin necesidad de transformaciones previas. Estas bases de datos permiten el almacenamiento directo de documentos XML y proporcionan capacidades avanzadas para consultar y manipular datos estructurados.