Introducción a XML (Extensible Markup Language)

- Introducción a los lenguajes de marcado (o de marcas)
- Desarrollados inicialmente para tratar con documentos (información predominantemente textual), hoy también para gestión de datos
- Describen las partes lógicas (estructura lógica y semántica) de un tipo de documento (o conjunto de datos) particular
- NO son lenguajes de formato o descripción de páginas (RTF, PostScript, PDF...); representación encomendada a hojas de estilo
- NO son lenguajes de programación (el marcado es interpretado por un programa)
- Marcan un documento por medio de etiquetas asignadas a las partes: `<TITULO>El mundo y sus demonios</TITULO>`
- Se crean a partir de metalenguajes (lenguajes para construir lenguajes específicos), como
    - **SGML** (*Standard Generalized Markup Language*) (1986)
- **XML** (Extensible Markup Language) (1998)
- **HTML** (*HyperText Mark-up* Language) es uno de estos lenguajes, basado primero en SGML y luego también en XML, con el nombre **XHTML** (eXtensible *HyperText Mark-up Language*), para preparar los documentos (o páginas) difundidos en la WWW (*World Wide Web*)

SGML

- Metalenguaje normalizado para el tratamiento de documentos digitales extensos, complejos y de tipos muy diferentes (transcripciones de rollos sumerios, documentación técnica de aviones, historiales médicos, notaciones musicales..., páginas web, obras literarias, registros MARC, instrumentos de descripción archivística...), facilitando su intercambio y reutilización
- **Metalenguaje**: lenguaje que permite crear lenguajes de marcado para definir la estructura y contenido de un tipo concreto de documentos; la especificación de un lenguaje concreto se recoge en una **DTD** (*Document Type Definition*, definición de tipo de documento)
- **Normalizado**: SGML: ISO (ISO-8879:1986), sobre la base de GML (1969): creado por Ch. F. Goldfarb (Mosher y Lorie), de IBM, para intercambiar documentos electrónicos entre plataformas informáticas
- Lenguaje robusto y potente / complejo y caro de implantar y mantener

- Algunos lenguajes basados en SGML:

- HTML (*HyperText Markup Language*, 1991-1993)

    - TEI P3 (*Text Encoding Initiative*, 1990-1994)

- MARC SGML (*MAchine Readable Cataloguing*, 1995)
- EAD 1.0 (*Encoded Archival Description*, 1995-1998)

    - ...

XML

- Versión abreviada de SGML
    - omite las partes más complejas y menos usadas de SGML
    - incide en aspectos de distribución e interoperatividad en la Web
- Permite definir la estructura de tipos específicos de documentos (o conjuntos de datos) por medio de **DTD** (como en SGML) o de **XML Schemas**, creando así lenguajes, vocabularios o aplicaciones, que definen sus elementos y los atributos de éstos

- Desarrollado (1996-1998) en el W3C (*World Wide Web Consortium*, Consorcio de la Web) por Jon Bosak (Sun) y otros

    - Rec. 1.0, 1998-02-10, Rec. 1.0, 2ª ed. 2000-10-06, 3ª ed. 2004-02-04
    - Rec. 1.1, 2004-02-04
- Apto para cualquier plataforma (codificación ISO-8859 de 8 bits, o ISO 10646 o Unicode, con UTF-8, 16 o 32)
- Algunos lenguajes basados en XML:
    - XHTML (*HyperText Markup Language*, 2000)
    - TEI P4 (*Text Encoding Initiative*, 2002)
    - MARC XML (2001); MARCXML Schema (2002)
    - EAD Version 2002 (*Encoded Archival Description*, 2002)
    - ...

El documento XML

- Los archivos de ordenador que contienen el documento SGML o XML (y también las DTD o Esquemas XML) son ficheros de texto (lo que les hace transportables de una a otra plataforma), codificados con un juego de caracteres basado en
  - ISO 8859 de 8 bits, como ISO 8859-1 o ISO Latin-1, o bien en
  - ISO 10646 (o Unicode, con UTF-8, 16 o 32, sólo en XML; el primitivo HTML sólo usaba ASCII, 7 bits), para poder codificar cualquier alfabeto
- En general los documentos XML llevan la extensión "`.xml`" (si bien los documentos XHTML suelen llevar la extensión "`.html`" o "`.htm`"), las DTD llevan "`.dtd`" y los Esquemas XML "`.xsd`".
- Como tales ficheros de texto, pueden ser creados por medio de:
  - editor de textos (como el Bloc de notas de Windows...)
  - editor específico de XML (XMetaL, XMLWriter, XML Spy...), lo que simplifica la tarea y ayuda en la validación
  - otras aplicaciones que exporten en formato XML
- Su tratamiento puede realizarse por medio de:
  - procesador XML genérico de los navegadores: Firefox 1.0, Netscape 6, MS Internet Explorer 5, Opera 5, Mozilla 1.0, Doczilla...
  - procesadores específicos de gestores de XML
  - aplicaciones especializadas según el tipo de documento

La estructura básica de un documento XML

- Examine el ejemplo sencillo de documento XML que se incluye al final

- Un documento XML está formado por:
  - el **prólogo**, que incluye:
    - una **declaración XML** (*instrucción de procesamiento*): `<?xml version="1.0" encoding="ISO-8859-1"?>`
    - generalmente (no necesariamente en XML) una **declaración de tipo de documento** que especifica una DTD (*definición de tipo de documento*), interna, externa o mixta, para la que hayan sido construidos: `<!DOCTYPE MENSAJE SYSTEM "mensaje.dtd">`. [En general las declaraciones toman la forma `<!...>`]

        - la referencia a una hoja de estilo externa que indica cómo representar el documento, en su caso (instrucción de procesamiento): `<?xml-stylesheet href="mensaje.css" type="text/css"?>`. [En general las instrucciones de procesamiento toman la forma `<?...?>`]
        - ... [Otras instrucciones de procesamiento, en su caso]
    - la **instancia del documento**, encerrada en un **elemento raíz** o **elemento de documento**, que contiene el resto de los constructores de un documento XML
        - **elementos**, que se corresponden con cada parte distinguible en la estructura del documento; son los constructores básicos, y están marcados por **etiquetas**,
        - **atributos**, que pueden acompañar a un elemento, estando expresados en la etiqueta inicial (o única de elemento vacío), para dar información adicional sobre dicho elemento por medio del **valor** que llevan asociado,
        - **entidades**, o variables que almacenan texto que ha de insertarse en el documento y que se referencian por medio de una **referencia de entidad**, que comienza por "&" y termina por ";",
        - **texto analizable** (o datos de carácter analizables, *Parsed Character DATA*, o **PCDATA**), que será fundamentalmente el texto del documento (el procesador de XML busca marcado dentro de él),
        - **texto no analizable** (o datos de carácter, *Character DATA*, o **CDATA**), que aparecerá principalmente como valores de ciertos atributos (el procesador de XML NO busca marcado dentro de él)

Fundamentos del marcado

- El **marcado** es el texto incluido entre **delimitadores**:
    - "<" y ">", que encierran **etiquetas**, o bien
    - "&" y ";", que encierran una **referencia de entidad**
- Las **etiquetas** delimitan **elementos** (partes estructurales del documento -párrafos, encabezados, etc.-, que pueden anidarse, esto es, contener otros elementos), y pueden ser
    - Dos: una inicial (en la forma `<elemento>`) y otra final (`</elemento>`), encerrando el contenido del elemento. Es el caso más frecuente. Ej.: `<title>Cosmos</title>`. Ej.: `<ASUNTO>Reunión proyecto</ASUNTO>`. En SGML, alguna (o ambas) podían ser opcionales; en XML no hay tal opcionalidad.
    - Una: etiqueta inicial o única de elementos vacíos (en la forma `<elemento>` bajo SGML, pero `<elemento/>` bajo XML); suelen insertar algo en el documento (una imagen, un salto de línea o de página...). Ej.: `<br/>`. Ej.: `<FECHA fd="09" fm="01" fa="2007" />`

- Los **elementos** son los constructores básicos en SGML y XML
- El *elemento de documento* (o *elemento raíz*)
  - encierra toda la instancia del documento
  - se corresponde con el tipo expresado en la *declaración de tipo de documento* `<!DOCTYPE...>` (que se tratará más adelante)
  - Ej.: `<html>[...]</html>`. Ej.: `<MENSAJE>[...]</MENSAJE>`
- Un elemento puede contener (según se especifique en la DTD -en XML sólo si ésta existe-):
  - **Otros elementos** (subelementos o elementos hijos, lo que origina conceptos como elementos padres, ancestros, descendientes, herencia de determinadas características o propiedades...). Ej.: `<html><head> [...] </head><body> [...] </body></html>`. Ej.: `<MENSAJE> <CABECERA> [...] </CABECERA> <CUERPO> [...] </CUERPO> </MENSAJE>`
  - **Texto** (normalmente el espaciado no tiene importancia: cualquier número de espacios o saltos de línea equivale a un espacio). Ej.: `<title>Cosmos</title>`. Ej.: `<DESPEDIDA>Saludos.</DESPEDIDA>`
  - Una mezcla de elementos o texto (**contenido mixto**). Ej.: `<p>En <strong>XHTML</strong> es obligatorio entrecomillar los valores de los atributos.</p>`. Ej.: `<P>Yo sólo podría el <ENFASIS>jueves o el viernes</ENFASIS>.</P>`
  - **Nada**, si es un elemento vacío. Ej.: `<hr>` (en SGML, y por tanto en HTML) o `<hr/>` (en XML, y por tanto en XHTML). Ej.: `<FECHA fd="09" fm="01" fa="2007" />`
- Un elemento puede llevar uno o más **atributos** (según se especifique en la DTD)
- Los **atributos**
  - especifican alguna propiedad del elemento en cuestión, por medio de un valor asociado, en la forma *`atributo="valor"`* (en XML el valor del atributo va necesariamente entrecomillado; en SGML a veces es opcional)
  - se expresan en la etiqueta inicial del elemento. Ej.: `<MENSAJE prioridad="maxima">[...]</MENSAJE>`.
  - si existen varios pueden expresarse en cualquier orden, p.ej.: `<FECHA fd="09" fm="01" fa="2007" />`, o bien `<FECHA fa="2007" fm="01" fd="09" />`.
  - dan sentido a la mayoría de los elementos vacíos. Ej.: `<img src="logouc3m.jpg" alt="Logo de la UC3M"/>`. Ej.: `<FECHA fd="09" fm="01" fa="2007" />`
  - la lista de atributos que puede llevar un elemento, su obligatoriedad, el tipo de valor que pueden llevar asociado, así como un posible valor por defecto, se especifican en la DTD
- Los **nombres** de elementos y atributos
  - comienzan por una letra (o "_"), seguida de letras o números o ".", "-", "_"
  - en XML, y por tanto en XHTML, son **sensibles a la caja de letra** (mayúsculas o minúsculas); en SGML, y por tanto en HTML, no

La definición del tipo de documento (DTD)

- Contiene la definición formal de un tipo de documento particular, definiendo:
  - elementos de su estructura que conforman el *vocabulario*
  - atributos que esos elementos pueden tomar para enriquecer su semántica
  - reglas que rigen las interacciones entre esos elementos (*gramática o sintaxis*), y
  - entidades que se pueden incluir en los documentos
- Así, todos los aspectos del marcado se especifican en la DTD a base de **declaraciones** (el orden en que se expresen es indiferente)
  - **de elementos**: tantas como elementos distintos (contenedores o vacíos) contenga la DTD; incluyen un **modelo de contenido**, que describe qué puede contener el elemento: nada, texto (datos de carácter), otros elementos hijos (especificando cuáles, su orden, su obligatoriedad y su repetibilidad) o contenido mixto (texto mezclado con elementos)
  - **de listas de atributos** que puede adoptar un elemento específico (en su caso); se detalla el tipo de valor que pueden tomar, su obligatoriedad y la existencia de un valor por defecto
  - **de entidades**: sobrenombre asociado a un grupo de datos (interna, de texto externa, binaria externa, de carácter o numérica)
  - **de notaciones**: (no las tratamos en este curso)
- En el documento XML se "declara", opcionalmente, la DTD a la que el documento se ajusta ("Declaración de Tipo de Documento": interna, externa, o combinación de ambas)
- **Validación** del marcado de un documento: un documento es
  - "bien formado": si está bien construido según las normas de XML
  - "válido": si además se ajusta a la DTD declarada [En SGML es obligatoria la existencia de una DTD, y por tanto su declaración. En XML, un documento puede no ajustarse a ninguna DTD.]

- Hay miles de DTD, accesibles y de uso público, tanto en SGML (que pueden ser convertidas a XML), como en XML (a las que hay que sumar Esquemas XML -XML Schemas-), pero a veces será necesario escribir DTD nuevas dar solución a necesidades particulares de una organización. [Como, por ejemplo, EAG (*Encoded Archival Guide*, Guía de Archivo Codificada), DTD de XML elaborada por la Subdirección General de los Archivos Estatales de España para la descripción de centros de archivo.]
- Al tratar las declaraciones que siguen, examine el ejemplo sencillo de DTD que se incluye al final

Declaración de elementos (o de tipos de elemento)

- Bloques básicos de una DTD o documento
- Incluye el modelo de contenido para el elemento
- Hay dos tipos de elementos:
    - **Contenedores** (el caso más habitual):
        - pueden incluir otros elementos, texto, o contenido mixto
        - DTD: Declarado en la forma `<!ELEMENT elemento (modelo_de_contenido)>`. Ej.: `<!ELEMENT SALUDO (#PCDATA)>`
        - Documento: marcado con una etiqueta inicial y otra final. Ej.: `<SALUDO>Hola de nuevo.</SALUDO>`
    - **Vacíos**:
        - no pueden contener texto ni ningún otro elemento
        - normalmente transmiten información mediante los valores de sus atributos
        - DTD: Declarado en la forma `<!ELEMENT elemento EMPTY>`. Ej.: `<!ELEMENT FECHA EMPTY>`
        - Documento: marcado con una etiqueta única de elemento vacío. Ej.: `<FECHA fd="09" fm="01" fa="2007" />`

Modelos de contenido

- Se refiere sólo al primer nivel de elementos anidados en su interior; los elementos hijos tienen sus propios modelos de contenido
- Tipos de contenido de un elemento:
    - **Nada** (elementos vacíos, como se ha visto), expresado en la forma `<!ELEMENT elemento EMPTY>`. Ej.: `<!ELEMENT FECHA EMPTY>`.
    - **Cualquier tipo de contenido** (infrecuente y desaconsejado, útil durante el desarrollo de la DTD), en la forma `<!ELEMENT elemento ANY>`
    - **Sólo texto**, en la forma `<!ELEMENT elemento (#PCDATA)>` (*parsed character data*, datos de carácter analizables). Ej.: `<!ELEMENT FIRMANTE (#PCDATA)>`
    - **Sólo otros elementos**. Pueden usarse **partículas de contenido**: bloques encerrando entre paréntesis elementos o nuevas partículas de contenido). La combinación de elementos, listas de opciones, secuencias y partículas de contenido (que además pueden anidarse) permiten expresiones muy complejas). Ej.: `<!ELEMENT MENSAJE (CABECERA, CUERPO)>`. Ej.: `<!ELEMENT CUERPO (SALUDO?, (P | CITA)+, DESPEDIDA?, FIRMANTE?)>`. Se detalla a continuación.
    - Otros elementos y texto (contenido mixto), en la forma `<!ELEMENT elemento (#PCDATA | elemento_1 | elemento_n)*>` ("*" es obligatorio). Ej.: `<!ELEMENT P (#PCDATA | ENFASIS)*>`

- En un modelo de contenido (o partícula de contenido) puede usarse uno de dos *separadores* de elementos (o partículas de contenido) para indicar:
  - ",": **secuencia**; los elementos deben aparecer (al margen de que sean opcionales o repetibles) en el orden expresado. Ej.: `<!ELEMENT MENSAJE (CABECERA, CUERPO)>`
  - "|": **lista de alternativas** u opciones; se podrá elegir uno de los elementos expresados en el modelo de contenido o partícula de contenido: `<!ELEMENT FIGURA (GRAFICO | FOTO)>`
- La **Repetibilidad** de un elemento (o partícula de contenido) se expresa colocando, detrás del elemento (o partícula de contenido) un *indicador de frecuencia*:
  - "": 1 vez (obligatorio, no repetible). Ej.: `<!ELEMENT MENSAJE (CABECERA, CUERPO)>`
  - "+": 1 o más (obligatorio, repetible). Ej.: `<!ELEMENT CABECERA (FECHA, DE, A+, CC*, ASUNTO?)>`
  - "?": 0 o 1 (opcional, no repetible). (V. ej. anterior)
  - "*": 0 o más (opcional, repetible). (V. ej. anterior)

Declaración de listas de atributos:

- Permite definir la lista de uno o más atributos que puede llevar un elemento
- DTD: Declarado en la forma `<!ATTLIST` *`elemento atributo_1 tipo_atributo_1 valor_por_defecto_1 atributo_n tipo_atributo_n valor_por_defecto_n`* `>`. Ej.: `<!ATTLIST FECHA fd CDATA #REQUIRED fm CDATA #REQUIRED fa CDATA #REQUIRED>`
- Documento: Ej.: `<FECHA fd="09" fm="01" fa="2007" />`
- **Tipos de atributos**:
  - **CDATA**: texto (datos de carácter no analizables)
  - de señalización:
    - **ID**: identificador que deberá ser único en el documento (no se podrá señalar un valor por defecto)
    - **IDREF** o **IDREFS**: referencia a un identificador único (o una lista de ellos) asociado como valor a un atributo de tipo ID de algún elemento del documento

- **ENTITY** o **ENTITIES**: entidad (o lista de ellas)
- **NMTOKEN** o **NMTOKENS**: como CDATA, pero restringido a una serie de caracteres que puedan formar un nombre XML válido (o una lista de ellos)

    - enumerados:
        - lista de valores posibles de los que el atributo podrá tomar uno
        - DTD: declarados en la forma `<!ATTLIST elemento atributo (valor_1 | valor_n) valor_por_defecto>`. Ej.: `<!ATTLIST MENSAJE prioridad (maxima | alta | normal | baja | minima) "normal">`
        - Documento: Ej.: `<!MENSAJE prioridad="maxima">[...]<MENSAJE>`
    - **NOTATION**: de notación (no los tratamos en este curso)
- Existencia de **valores por defecto** y obligatoriedad del atributo:
    - **#REQUIRED**: el atributo debe expresarse necesariamente en el documento, y no toma ningún valor por defecto
    - **#IMPLIED**: el atributo puede expresarse o no en el documento, pero si no se expresa no toma ningún valor por defecto
    - **#FIXED "*valor*"**: el autor del documento no puede modificar el *valor* expresado, que es fijo. Declarados en la forma `<!ATTLIST elemento atributo FIXED "valor">`, donde es obligatorio expresar el *valor* que toma el atributo
    - **valor por defecto**, expresado en la forma `<!ATTLIST elemento atributo tipo_atributo "valor">`. Si el atributo no se expresa en el documento toma ese *valor* por defecto. Ej.: `<!ATTLIST MENSAJE prioridad (maxima | alta | normal | baja | minima) "normal">` (en este ejemplo, si en el documento XML no se expresa el atributo, el procesador XML debe entender que toma el valor "normal"; si en la DTD en lugar de `"normal"` apareciera `#REQUIRED` el atributo tendría que expresarse necesariamente en el documento, y no habría valor por defecto; si apareciera `#IMPLIED`, el atributo podría expresarse o no, pero en este último caso no tomaría ningún valor por defecto)

Declaración de entidades:

- Unidad virtual de almacenamiento (cadena de caracteres, fichero gráfico...)
- Tipos:
    - **De texto**:
        - Pueden ser internas, esto es, declaradas en el documento en la parte interna de la DTD, o externas, esto es, declaradas en la DTD externa, el caso más habitual)
        - pueden contener grupos de palabras, cadenas de caracteres o bloques de texto usados con frecuencia
        - cada vez que en el documento XML se haga referencia a la entidad, la cadena de caracteres la reemplazará
    - **De carácter y numéricas**:
        - Facilitan un mecanismo de descripción de caracteres no ASCII, como letras acentuadas o símbolos especiales

        - XML posee un pequeño conjunto de entidades de carácter predefinidas que se pueden utilizar cuando se necesite incluir caracteres especiales en los documentos: `&` (&), `&apos;` ('), `>` (>), `<` (<), `&quot;` (")
    - **Binarias**:
        - contienen datos que no sean texto o marca (gráfico, fichero multimedia...)
    - **Paramétricas**:
        - tipo especial de uso reservado a las DTDs
        - diseñadas para contener listas de atributos y modelos de contenido que se vayan a utilizar con frecuencia
- Hay que declararlas en la DTD antes de poder utilizarlas
- Se declaran en la forma `<!ENTITY nombre "contenido">` (las paramétricas como `<!ENTITY %nombre "contenido">`)
- Se referencia a una entidad en la DTD o en el documento como `&nombre;` (las paramétricas como `%nombre;`)