XML

XML

XML, ou Extensible Markup Language, é um padrão da W3C (World Wide Web Consortium) para definição de documentos estruturados.

É um subconjunto de uma especificação bem mais abrangente chamada de SGML – Standard Generalized Markup Language que foi desenvolvida ao final da década de 70, hoje é padrão ISO8879 (1986).

XML

XML foi proposto como uma versão light de SGML, com o objetivo de ser aplicada na Web.

Apenas descreve a estrutura dos dados, não interessando como estes serão apresentados

XML

Um documento XML é formado por uma coleção de elementos XML, organizados de uma maneira hierárquica.

Cada elemento pode conter texto ou outros elementos.

Os elementos são representados por etiquetas.

Ex.:

```
<texto></texto>
```

Os atributos são nomes contidos nos elementos.

Ex.:

```
<texto valor="teste"></texto>
```

A especificação XML não define um vocabulário de nomes para elementos ou atributos.

Apenas define regras sobre como os elementos devem ser representados.

Fica a critério do autor das os nomes aos elementos e atributos.

```
<bilhete codigo=”XPTO80”>
    <voo transportador=”JH” numero=”3241” de=”GUA” para=”PEC” />
   <passageiro>
    <sobrenome>Silva</sobrenome>
    <prenome>José</prenome>
   </passageiro>
</bilhete>
```

Documentos válidos e bem formatados

Uma aplicação XML requer um esquema que descreva todos os seus atributos, valores permitidos, e outras regras de construção.

- DTD – Document Type Definition
- XML Schema

Um DTD permite validar um documento XML.

No DTD são definidos todos os elementos legais, atributos, regras de uso, etc.

Documentos XML que combinam com o DTD são documentos válidos.

XML

```
<!ELEMENT bilhete (voo, passageiro) >
<!ATTLIST bilhete codigo NMTOKEN #REQUIRED >
<!ELEMENT passageiro (sobrenome, prenome) >
<!ELEMENT prenome (#PCDATA) >
<!ELEMENT sobrenome (#PCDATA) >
<!ELEMENT voo EMPTY >
<!ATTLIST voo de NMTOKEN #REQUIRED >
<!ATTLIST voo numero NMTOKEN #REQUIRED >
<!ATTLIST voo para NMTOKEN #REQUIRED >
<!ATTLIST voo transportador NMTOKEN #REQUIRED >
```

A sintaxe usada no DTD não é XML, é SGML.

A validade de um documento depende do DTD.

Um documento bem-formatado que é valido de acordo com um DTD pode não ser válido de acordo com outro DTD.

XML

A unidade de informação dentro de um documento XML é o elemento.

Um elemento é formado por duas etiquetas (tags) que atribuem significado ao conteúdo.

Ex.:

<vazio/> <log>erro 32</log>

<host name=”xp32”/>

O conteúdo de um elemento pode ser composto de texto , outros elementos ou de texto e elementos misturados.

```
<contato>
  <nome>Fulano de Tal</nome>
  <email>fulano@terra.com.br</email>
  <telefone>
    <ddd>11</ddd>
    <numero>5234-0394</numero>
  </telefone>
</contato>
```

Um documento XML pode ser dividido em duas partes.

- Inicial ou prolog – É tudo que aparece antes do elemento raiz
- O corpo do documento – Começa no seu elemento raiz.

XML

```
<?xml version=”1.0” encoding=”iso-8859-1” ?>
<!-- Isto é um comentário -->
<?comando tipo=”simples” parametro ?>
<!DOCTYPE cartao-simples SYSTEM “cartoes.dtd”>
<cartao-simples>
  <logotipo href=”/imagens/logo14bins.gif” />
  <nome>Fulano de Tal</nome>
  <endereco>Rua Bela Vista, 45 – fundos Centro -
  04938-201 – São Paulo SP</endereco>
  <email>fulano@terra.com.br</email>
  ...
</cartao-simples>
```

O prolog pode consistir de:

- Uma Instrução de processamento padrão, chamada de declaração XML
- Uma declaração <!DOCTYPE> no início do documento
- Comentários
- Zero ou mais instruções de processamento

O corpo do documento começa no seu elemento raiz e também pode conter comentários.

Sua estrutura representa uma árvore.

Cada elemento é um nó.

Os elementos contidos dentro de elementos (parent nodes) são os nós filhos (child nodes).

Nós que não tem filhos são nós folha (leafs).

Documentos XML também podem conter entidades de substituição.

Entidades de substituição são símbolos do tipo **&nome**; onde "***nome***" é usado para identificar a entidade.

Entidades representam texto ou fragmentos de XML que serão embutidos no documento quando ele for processado.

Elementos e atributos

- A etiqueta inicial de um elemento é identificada pelo nome do elemento e pode ter zero ou mais atributos.
- Não deve haver espaço entre o < e o nome do elemento na etiqueta inicial nem entre o </ e o nome na etiqueta final.

Atributos tem sempre a forma *nome*=”*valor*” e só podem aparecer na etiqueta inicia

Os nomes dos elementos nas etiquetas inicial e final têm que ser iguais.

Formato errado:

```
<Profissão>Analista</profissão>
<tamanho>123.45</TAMANHO>
```

Bem formatados:

```
<data>15/04/1945</data>
<data dia=”15” mes=”04” ano=”1945” />
```

Identificadores

- Qualquer caracter alfanumérico
- Símbolos “_”, “-” e “.”
- Não podem começar com número, ponto ou hífen
- Não podem ter espaço
- Não existe limite para o tamanho do nome

Referências de entidades

- < corresponde a <
- > corresponde a >
- & corresponde a &
- " corresponde a “
- ' corresponde a '

Referência de caracteres

- Através de uma sintaxe semelhante à usada para incluir entidades é possível incluir caracteres Unicode.
- A sintaxe é &#xNNNN; Ex.: &#000D;
- Há uma tabela com todos os caracteres Unicode em www.unicode.org/charts

Seções CDATA

- Servem para embutir texto que deve ser tratado como caractere e que não pode ser tratado como XML
- Seu conteúdo não é interpretado
- É ótimo para incluir textos que contém sinais **<** e **>**
- Inicia com **<![CDATA[** e termina com **]]>**
- A seqüência **]]>** não deve aparecer dentro do bloco **CDATA**

```
<titulo>Curso de XML</titulo>
<exemplo>Considere o seguinte trecho de
   XML: <![CDATA[
   <empresa>
     <nome>José & Silva S/A</nome>
   <empresa>
]]></exemplo>
```

Instruções de processamento

- Servem para embutir instruções que serão utilizadas por algum programa que irá interpretar a página.
- Se um programa não reconhecer a instrução, esta será ignorada.
- Tem duas partes:
  - nome que identifica a instrução
  - dados ocupam o resto da instrução

<?nome-do-alvo área de dados ?>

<?query-sql select nome, email from agenda where id=25 ?>

Declaração XML

- É opcional
- Define o alfabeto a ser utilizado no documento XML

```
<?xml encoding=”iso-8859-5” ?>

<!-- a partir daqui, texto em cirílico -->
```

Documento Bem-formatado

- Ter um único elemento raiz
- Etiquetas iniciais e finais combinam
- Elementos bem aninhados
- Valores de atributos entre aspas ou apóstrofes
- Atributos não repetidos
- Identificadores válidos para elementos e atributos
- Comentários não devem aparecer dentro das etiquetas
- Sinais < ou & nunca devem ocorrer dentro dos valores dos atributos ou nos nós de texto do documento

Namespaces

- Um identificador unívoco para cada elemento e atributo
- Este identificados se soma ao nome do elemento formando um nome maior que jamais se repete
- O identificador é associado ao nome através de um prefixo usando o sinal de “:” como separador

Criar um Namespace significa declarar, dentro do escopo de um elemento uma URI como identificador de um espaço onde certos nomes serão reconhecidos

O atributo "**xmlns**" é utilizado para realizar a declaração de namespace

```
<raiz>
  <empresa xmlns=”urn:cgc:01.234.567/0001-89/acme”>
    <nome>Acme S.A.</nome>
    <endereco>Rua Boa Vista, 22 – Centro</endereco>
  </empresa>
  <cartao>
    <nome>Beltrano do Prado</nome>
    <endereco>Rua Bela Vista, 45 – Bom Retiro
      02837-102 – São Paulo – SP</endereco>
  </cartao>
</raiz>
```

No caso de termos elementos misturados podemos:

```
<cartao xmlns:empresa=”urn:cgc:01.234.567/0001-89/acme”>
  <nome>Beltrano do Prado</nome>
  <endereco>Rua Bela Vista, 45 – Bom Retiro
    02837-102 – São Paulo – SP</endereco>
  <empresa:nome>Acme S.A.</empresa:nome>
  <empresa:endereco>Rua Boa Vista, 22 –
    Centro</empresa:endereco>
</cartao>
```

Pode-se declarar um namespace em qualquer elemento

Ele se aplica ao próprio elemento e aos descendentes

Se a declaração foi feita com prefixo, o próprio elemento que declara o namespace também precisa ter o prefixo

XML

```
<car:cartao
   xmlns:empresa=”urn:cgc:01.234.567/0001-89/acme”
   xmlns:car=”http://www.provedor.com/belfin”>
  <car:nome>Beltrano do Prado</car:nome>
  <car:endereco>Rua Bela Vista, 45 – Bom Retiro
   02837-102 – São Paulo – SP</car:endereco>
  <empresa:nome>Acme S.A.</empresa:nome>
  <empresa:endereco>Rua Boa Vista, 22 –
   Centro</empresa:endereco>
</car:cartao>
```

Com namespaces um nome de elemento agora tem duas partes:

- A parte local
- O prefixo

O nome formado pela parte local e pelo prefixo é o nome qualificado

Os DTDs não reconhecem a parte local e o prefixo, mas reconhecem o nome qualificado

Na validação é necessário levar em conta o prefixo

O namespace também pode ser declarado no DTD, para que os autores das páginas não precisem fazê-lo

Validação

- Para criar uma aplicação XML é preciso definir um vocabulário limitado de elementos e atributos e regras de formatação que sejam comuns a todos os documentos escritos na nova linguagem
- Em outras palavras, é preciso definir uma classe de documentos
- Uma das formas de fazer essa especificação é através da construção de um DTD ou Document Type Definition

Validação com DTD

- Um documento XML válido inclui uma declaração de tipo de documento (document type declaration) que identifica o seu DTD
- O DTD pode ser um arquivo localizado no sistema do usuário ou em algum lugar da rede ou estar embutido em algum lugar do programa
- O DTD utiliza uma sintaxe SGML, diferente da sintaxe XML e DTD representa uma classe de documentos XML

O DTD

- É a especificação da linguagem usada por uma determinada aplicação do XML
- Pode conter itens que o documento não está usando, mas o documento não pode conter itens que não estão definida no DTD
- Representa o universo de todos os documentos que podem ser criados
- Um documento pode, no máximo, ter uma estrutura idêntica à definida no DTD

Considere o documento XML:

```
<pessoa>
  <nome>
    <prenome>Fulano</prenome>
    <sobrenome>de Tal</sobrenome>
  </nome>
  <profissao>Programador</profissao>
  <profissao>Analista de Suporte</profissao>
</pessoa>
```

O DTD abaixo descreve o documento XML anterior

```
<!ELEMENT pessoa (nome, profissao*)>
<!ELEMENT nome (prenome, sobrenome)>
<!ELEMENT prenome (#PCDATA)>
<!ELEMENT sobrenome (#PCDATA)>
<!ELEMENT profissao (#PCDATA)>
```

O DTD pode ser armazenado em um arquivo separado, ou pode ser embutido no próprio arquivo .xml

No DTD a declaração de elemento informa o nome do elemento e, entre parênteses, o que ele pode conter

O elemento raiz não é identificado dentro do DTD mas na declaração **<!DOCTYPE>** de cada documento que implementa o DTD

No prolog do documento que contém o XML de pessoa deve haver a seguinte declaração:

```
<!DOCTYPE pessoa SYSTEM
                "http://pessoas.com/pessoa.dtd">
```

Documentos válidos sempre contêm uma referência para o DTD

Esta referência pode ser do tipo **SYSTEM**, e apontar para um arquivo localizado em algum lugar do sistema ou da rede, ou do tipo **PUBLIC**

Declarações do tipo **PUBLIC** possuem uma identificação que associa o documento a uma DTD conhecida

É comum oferecer também uma URL alternativa para o caso do software não reconhecer o identificador:

```
<!DOCTYPE HTML PUBLIC
        "-//WC3//DTD HTML 4.0//EN"
        "http://www.w3.org/TR/REC-html40/strict.dtd">
```

O exemplo acima é o **DOCTYPE** usado nos documentos HTML 4.0

Um DTD pode ser embutido em um documento

```
<?xml version=”1.0” encoding=”iso-8859-1” ?>
<!DOCTYPE pessoa [
  <!ELEMENT pessoa (nome, profissao*)>
  <!ELEMENT nome (prenome, sobrenome)>
  <!ELEMENT prenome (#PCDATA)>
  <!ELEMENT sobrenome (#PCDATA)>
  <!ELEMENT profissao (#PCDATA)> ]>
<pessoa>
  <nome>
    <prenome>Fulano</prenome>
    <sobrenome>de Tal</sobrenome>
  </nome>
</pessoa>
```

Para embutir parcialmente:

```
<!DOCTYPE pessoa SYSTEM “pessoa.dtd” [
  <!ELEMENT nome (#PCDATA)>
]>
```

O DTD local não pode sobrepor declarações de elementos feitas em documentos externos mas pode redefinir entidades

O código acima só seria aceito se o DTD externo não declarasse **<nome>**

Elementos do DTD

O **<!ELEMENT>** é utilizado na declaração de cada elemento da página

```
<!ELEMENT trem (locomotiva, vagoes*) >
<!ELEMENT trecho (ferroviario | aéreo | rodoviário) >
<!ELEMENT circulo (centro, ( raio | diâmetro) >
<!ELEMENT ponto ((x, y) | ( r, d)) >
<!ELEMENT nome (prenome | ( prenome | inicial*, sobrenome)) >
<!ELEMENT nome (#PCDATA | enfase)* >
<!ELEMENT nome EMPTY >
```

O **<!ATTLIST>** é utilizado na declaração de atributos declarados nos elementos da página

```
<!ATTLIST voo de NMTOKEN #IMPLIED >
<!ATTLIST voo numero CDATA #FIXED “12/3283-1” >
<!ATTLIST voo para ( REC | CGH | GRU ) #REQUIRED  >
<!ATTLIST voo transp ( RG | JH ) “RG”  >

<!ATTLIST voo de NMTOKEN #IMPLIED
                numero CDATA #FIXED “12/3283-1”
                para ( REC | CGH | GRU ) #REQUIRED
                transp ( RG | JH ) “RG” >
```

Tipos de dados de **<!ATTLIST>**

- **CDATA** caracteres de qualquer tipo;
- **NMTOKEN** mesmas restrições para nomes de elementos e atributos e não pode conter números;
- **ID** nome unívoco;
- **IDREF** referência a um **ID**;
- **ENTITY** referência a entidade externa (arquivo, imagem, etc)

Valores default para **<!ATTLIST>**

- **#REQUIRED** força o autor do documento a definir o atributo
- **#IMPLIED** o atributo é opcional
- **#FIXED** o atributo tem um valor fixo

A **<!ENTITY>** é utilizado para definir novas entidades

- Processadas internamente - A referência deve ser um nome e o valor pode ser qualquer texto inclusive trechos bem formados de XML
- Processadas externamente – Define uma referência a um documento XML externo
- Parâmetros – Utilizadas onde as declarações são muito repetitivas

<!ENTITY> Processadas internamente

```
<!ENTITY empresa
    “ACME Indústria &amp;amp; Comércio de Coisas S.A.” >
<!ENTITY copyright “<table>
  <tr>
    <td>Copyright &#x00A( 2006</td>
  </tr>
</table>” >
```

O documento pode conter referências à entidades

```
<texto>Visitante da &empresa; ainda hoje!.</texto>
<div>&copyright;</div>
```

- **<!ENTITY>** Processadas externamente

```
<!ENTITY menu_sup SYSTEM “/fragmentos/menu_superior.xml”>
```

A entidade sendo utilizada dentro do documento

```
<texto>Qualquer texto</texto>
&menu_sup;
<elemento>...
```

Entidades de Parâmetros

```
<!ENTITY voo de (REC | CGH | GRU | GIG | SDU) >
<!ENTITY voo para (REC | CGH | GRU | GIG | SDU) >
```

Utilizando entidades como parâmetros em documentos com o objetivo de otimização e simplificação

```
<!ENTITY % aerosp “CGH | GRU” >
<!ENTITY % aerorio “GIG | SDU” >
<!ENTITY % aeroportos “REC | %aerorio; | %aerosp;” >
<!ATTLIST voo de (%aeroportos;) #REQUIRED >
<!ATTLIST voo para (%aeroportos;) #REQUIRED >
```

Namespaces

- DTDs não suportam namespaces
- Para declarar um elemento não basta usar apelas o nome local. É preciso usar o nome qualificado

```
<!ENTITY prefixo:elemento (#PCDATA) >
```

FIM