Minicurso de XML: Do Básico à Prática

Uma jornada pela tecnologia que estrutura dados

Gleryston Matos

Gleryston Matos:

Graduado em sistemas e informação

+ 12 anos atuando como desenvolvedor de software

Líder técnico | Consultor | Desenvolvedor sênior

Membro e organizador do DUG-CE

Escritor de artigos no medium.

Conteúdo Programático: Minicurso de XML

Este conteúdo foi pensado para oferecer uma visão geral e prática sobre a tecnologia XML (eXtensible Markup Language), abordando desde os conceitos fundamentais até as tecnologias associadas, ideal para iniciantes.

O que veremos hoje?

Módulo 1: Introdução ao XML

Módulo 2: Estrutura e Sintaxe

Módulo 3: Validação de Documentos

Módulo 4: Tecnologias Associadas e Encerramento

Módulo 1: Introdução ao XML

O que é XML?

XML significa **eXtensible Markup Language** (Linguagem de Marcação Extensível). Pense nela como um conjunto de regras para criar outras linguagens de marcação.

Propósito Principal: O XML não foi criado para apresentar dados, mas sim para descrever, armazenar e transportar dados de forma que seja legível tanto por humanos quanto por máquinas. Ele se concentra no significado dos dados.

Diferença Fundamental do HTML:

HTML (HyperText Markup Language): Focado na **apresentação** dos dados. As tags do HTML são predefinidas (<p>, <h1>, <img>) e dizem ao navegador como exibir o conteúdo.

XML: Focado na **descrição** dos dados. As tags no XML **não são predefinidas.** Você as cria para descrever a sua informação.

Principais Características e Vantagens

Legibilidade: Sua estrutura é simples e se assemelha a uma árvore, facilitando a compreensão.

Flexibilidade (Extensibilidade): Você inventa as tags que fazem sentido para os seus dados. Se você precisa descrever um carro, pode criar a tag <carro>.

Independência de Plataforma: Por ser um formato de texto simples, pode ser lido e processado por praticamente qualquer linguagem de programação ou sistema operacional.

Sugestão: Notepad++ com plugin XML Tools

Anatomia de um Documento XML

Vamos analisar a estrutura de um arquivo XML que descreve um livro.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<livro isbn="978-85-359-0277-3">
  <titulo>Dom Casmurro</titulo>
  <autor>Machado de Assis</autor>
</livro>
```

Anatomia de um Documento XML

Prólogo (<?xml ... ?>): A primeira linha. Declara a versão do XML e o encoding (padrão de caracteres) do arquivo. É opcional, mas uma boa prática.

Elemento Raiz (<livro>): Todo documento XML deve ter um, e apenas um, elemento que envolve todos os outros. Aqui, é o elemento <livro>.

Elementos Filhos (<titulo>, <autor>, etc.): São as tags aninhadas dentro do elemento raiz. Elas descrevem as partes da informação. Note que <titulo> é um filho de <livro>.

Atributo (isbn="..."): É uma informação adicional sobre um elemento, declarada dentro da tag de abertura. Atributos são pares nome="valor".

Comentário (``): Texto que será ignorado pelo processador XML, útil para anotações.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<livro isbn="978-85-359-0277-3">
  <titulo>Dom Casmurro</titulo>
  <autor>Machado de Assis</autor>
</livro>
```

Módulo 2: Estrutura e Sintaxe

Regras Essenciais de Sintaxe

O XML é rigoroso. Um erro pequeno pode invalidar todo o documento.

Tag de Fechamento é Obrigatória: Toda tag que é aberta (<autor>) deve ser fechada (</autor>).

Errado: <autor>Machado de Assis

Certo: <autor>Machado de Assis</autor>

Tags de **auto fechamento** são usadas para elementos que não contêm conteúdo interno.

Certo: <autor></autor>

Certo: <autor/>

Regras Essenciais de Sintaxe

O XML é rigoroso. Um erro pequeno pode invalidar todo o documento.

Case-Sensitive: As tags diferenciam maiúsculas de minúsculas.

Errado: <titulo>Dom Casmurro</Titulo>

Certo: <titulo>Dom Casmurro</titulo>

Regras Essenciais de Sintaxe

O XML é rigoroso. Um erro pequeno pode invalidar todo o documento.

Aninhamento Correto: As tags devem ser fechadas na ordem inversa em que foram abertas.

Errado: <autor><nome>Machado de Assis</autor></nome>

Certo: <autor><nome>Machado de Assis</nome></autor>

Regras Essenciais de Sintaxe

O XML é rigoroso. Um erro pequeno pode invalidar todo o documento.

Aspas em Atributos: Os valores dos atributos devem sempre estar entre aspas (simples ou duplas).

Errado: <livro isbn=9788535902773>

Certo: <livro isbn="978-85-359-0277-3">

Elementos vs. Atributos: Qual usar?

Não há uma regra de ouro, mas uma boa prática é:

Use **elementos** para representar os dados principais, a informação em si. (Ex: no segundo exemplo id e nome são elementos do produto).

Use **atributos** para representar metadados, ou seja, informações sobre o elemento. (Ex: no primeiro exemplo id é um atributo do produto, no segundo exemplo id é um elemento do produto).

```
<produto id="123">
  <nome>Cafeteira Elétrica</nome>
</produto>
```

```
<produto>
  <id>123</id>
  <nome>Cafeteira Elétrica</nome>
</produto>
```

Comentários e Entidades

Comentários: Já vimos, usam a sintaxe ``.

Entidades: Como o XML usa < e > para definir tags, você precisa de uma forma especial para usar esses caracteres no texto.

```
<calculo>A comparação 5 &lt; 10 é verdadeira.</calculo>
<empresa>M&amp;M Consultoria</empresa>
```

< para o sinal de menor que (<)

> para o sinal de maior que (>)

& para o "E" comercial (&)

' para apóstrofo (')

" para aspas duplas (")

Seções CDATA

Se você tem um bloco grande de texto que contém muitos caracteres especiais (como um código de programação), usar entidades para tudo seria inviável. Para isso, use uma seção **CDATA** (Character Data). O processador ignora tudo que está dentro dela.

```
<script>
  <![CDATA[
    if (a < b && b > c) {
      console.log("Exemplo de código com < e &");
    }
  ]]>
</script>
```

Módulo 3: Validação de Documentos

A Importância de Validar um XML

Validar um XML significa checar se ele segue um conjunto de regras predefinidas (um "contrato"). Isso garante que todos os documentos trocados entre sistemas tenham a mesma estrutura e tipos de dados, evitando erros de integração.

DTD (Document Type Definition)

O DTD é a forma mais antiga de definir a estrutura de um XML. Ele diz quais elementos são permitidos, em que ordem e quantas vezes podem aparecer.

O exemplo define que um <catalogo> deve ter um ou mais (+) elementos <livro>, e cada <livro> deve ter titulo, autor e ano_publicacao, nesta ordem.

```
<!DOCTYPE catalogo [
  <!ELEMENT catalogo (livro+)>
  <!ELEMENT livro (titulo, autor, ano_publicacao)>
  <!ATTLIST livro isbn CDATA #REQUIRED>
  <!ELEMENT titulo (#PCDATA)>
  <!ELEMENT autor (#PCDATA)>
  <!ELEMENT ano_publicacao (#PCDATA)>
]>
```

XSD (XML Schema Definition)

O XSD é a forma **moderna e mais poderosa** de validação. Ele supera as limitações do DTD.

Vantagens do XSD:

Tipagem de Dados: Você pode definir se um elemento é um texto, número, data, booleano, etc. (xs:string, xs:integer, xs:date). Podem ser criados tipos personalizados.

Restrições (Facets): Pode definir regras mais complexas, como um número mínimo e máximo de ocorrências, ou padrões de texto (ex: um CEP deve ter 8 dígitos).

Sintaxe XML: O próprio arquivo de schema (.xsd) é um documento XML, facilitando seu processamento.

Exemplo de XSD

schema_livro.xsd

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="livro">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="titulo" type="xs:string"/>
        <xs:element name="autor" type="xs:string"/>
        <xs:element name="ano_publicacao" type="xs:integer"/>
      </xs:sequence>
      <xs:attribute name="isbn" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Associando o XSD ao XML

Se você tentasse colocar um texto no campo <ano_publicacao>, o validador XSD apontaria um erro, pois ele espera um integer (número inteiro).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<livro xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:noNamespaceSchemaLocation="schema_livro.xsd"
       isbn="978-85-359-0277-3">
  <titulo>Dom Casmurro</titulo>
  <autor>Machado de Assis</autor>
  <ano_publicacao>1899</ano_publicacao>
</livro>
```

Módulo 4: Tecnologias Associadas e Encerramento

XPath (XML Path Language)

XPath é uma linguagem para navegar e selecionar partes de um documento XML.

Pense nela como o "endereço" de um elemento dentro do arquivo, sendo fundamental para encontrar informações específicas.

Onde e Como Usar o XPath na Prática?

Você não escreve XPath "no vácuo". Você o utiliza dentro de outras ferramentas ou linguagens, como em ferramentas de desenvolvimento de navegadores (Chrome, Firefox, etc.)

Abra as **"Ferramentas de Desenvolvedor"** (clique com o botão direito na página e "Inspecionar" ou pressione F12).

Vá para a aba "Elements".

Pressione Ctrl+F ou Cmd+F. Uma barra de busca aparecerá.

Cole sua expressão XPath ali. Ele destaca os elementos correspondentes no HTML (que é um tipo de XML). Esta é a forma mais fácil e rápida de testar expressões!

XPath (XML Path Language)

Expressões Básicas:

/ : Inicia a partir do nó raiz.

//: Seleciona nós em qualquer lugar do documento.

. : Seleciona o nó atual.

..: Seleciona o pai do nó atual.

@: Seleciona atributos.

XPath (XML Path Language)

Exemplo de XPath (usando nosso XML do livro):

//livro/titulo: Seleciona o elemento <titulo> que é filho de <livro>.

//autor: Seleciona todos os elementos <autor> no documento.

//livro/@isbn: Seleciona o atributo isbn do elemento <livro>.

Exemplo:

Aplicações Práticas do XML

🧾 **Nota Fiscal Eletrônica (NF-e):** No Brasil, toda NF-e é, na verdade, um arquivo .xml padronizado pelo governo. Ele contém todos os dados da transação (emitente, destinatário, produtos, impostos).

🌐 **Web Services (SOAP):** Protocolo para troca de mensagens estruturadas entre sistemas, usando XML como formato padrão.

⚙️ **Arquivos de Configuração:** Muitas aplicações (especialmente em Java e .NET) usam arquivos XML para guardar suas configurações.

Conclusão

Recapitulando: Vimos que o XML é uma tecnologia poderosa para **estruturar, armazenar e transportar dados**. Aprendemos sua sintaxe básica, como validá-lo com XSD para garantir a integridade e como usar XPath para consultar informações.

Mensagem Final: O XML é uma tecnologia madura e fundamental que serve de base para muitas outras, como a Nota Fiscal Eletrônica, que impacta todas as empresas do Brasil.

Obrigado!

Perguntas?

GitHub: https://github.com/GlerystonMatos
Medium: https://medium.com/@glerystonmatos
Linkedin: https://www.linkedin.com/in/glerystonmatos
Youtube: https://www.youtube.com/user/Glerystonmatos

DUG-CE:
Grupo: https://t.me/DUGCE
Canal: https://www.youtube.com/c/DUGCE-CANAL

Download apresentação: