XML: Extensible Markup Language

QXD0099 - Desenvolvimento de Software para Persistência

Universidade Federal do Ceará - *Campus* Quixadá

Prof. Francisco Victor da Silva Pinheiro
victorpinheiro@ufc.br

Slides baseados no material do Professor Régis Magalhães

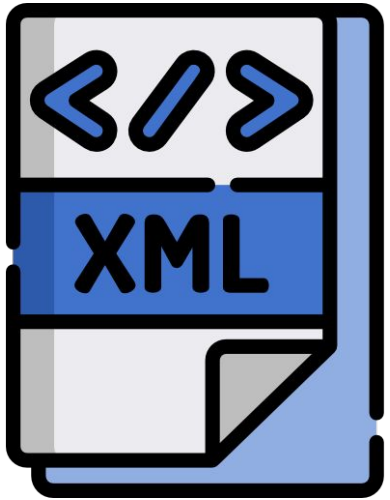
Agenda

- Motivação
- XML - Extensible Markup Language
- O que é XML?
- Modelo de dados hierárquico da XML
- Representação de documentos XML
- Documento XML - Atributos
- Estrutura de um documento XML
- Manipulações
- APIs para processamento de XML
- CRUD persistindo em XML

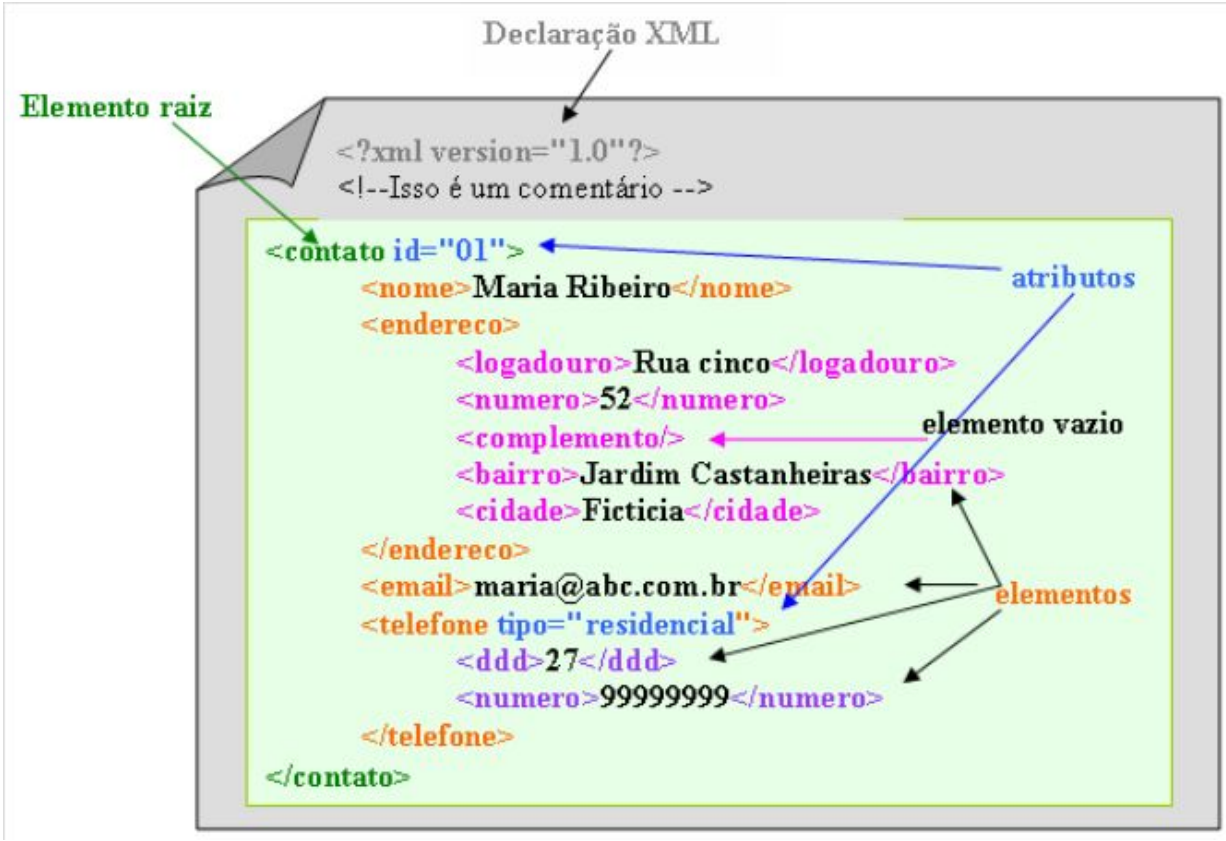
Motivação

- À medida que o comércio eletrônico e outras aplicações da Internet se tornam cada vez mais automatizadas, torna-se essencial a capacidade de trocar documentos Web entre diversos sites de computador e interpretar seu conteúdo de maneira automática.

- Essa necessidade foi um dos motivos que levaram ao desenvolvimento da XML.



Exemplo


Declaração XML
Elemento raiz
<?xml version="1.0"?>
<!--Isso é um comentário -->
<contato id="01">
<nome>Maria Ribeiro</nome>
<endereco>
<logadouro>Rua cinco</logadouro>
<numero>52</numero>
<complemento/>
<bairro>Jardim Castanheiras</bairro>
<cidade>Ficticia</cidade>
</endereco>
<email>maria@abc.com.br</email>
<telefone tipo="residencial">
<ddd>27</ddd>
<numero>99999999</numero>
</telefone>
</contato>
atributos
elemento vazio
elementos

O que é XML?

- XML (eXtensible Markup Language)
- Linguagem de marcação proposta pelo W3C
  - W3C (World Wide Web Consortium) – Órgão responsável pela recomendação de padrões e protocolos para a web.

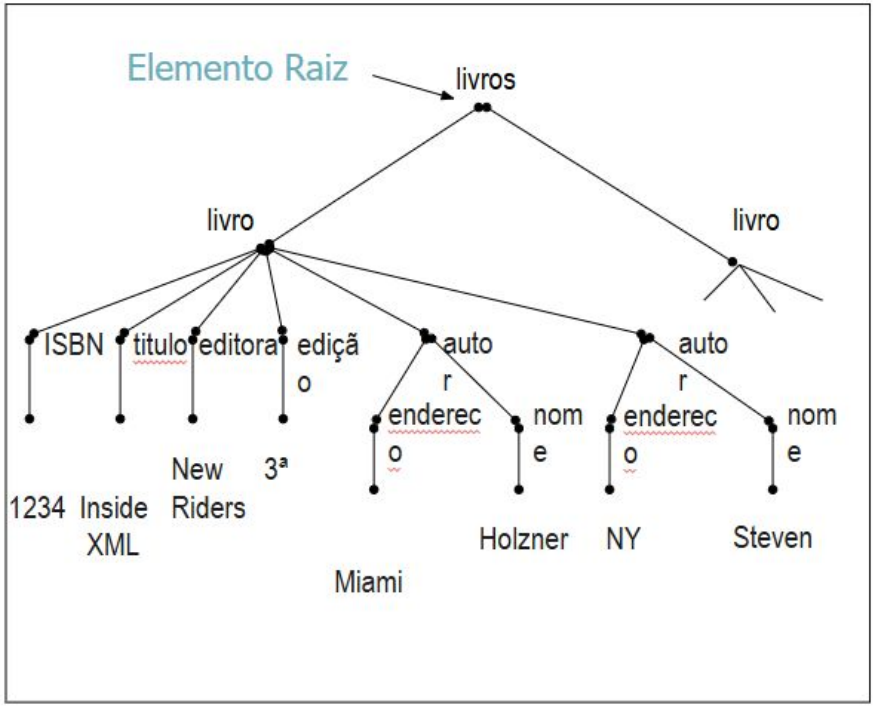
- Padrão para representação e troca de dados na Web.

- Descreve os dados, dando semântica a unidades de informação

- Solucionar as limitações de HTML

O que é XML?

```
<?xml version="1.0" >
<livros>
 <livro>
  <ISBN>1234</ISBN>
  <titulo>Inside XML</titulo>
  <editora>New Riders</editora>
  <edição>3ª</edição>
  <autor>
   <nome>Steven</nome>
   <endereco>NY</endereco>
  </autor>
  <autor>
   <nome>Holzner</nome>
   <endereco>Miami</endereco>
  </autor>
 </livro>
 <livro>  ...  </livro>
</livros>
```



Modelo de dados hierárquico da XML

- XML não é apenas um outra linguagem de marcadores
- A maioria das linguagens provê um conjunto fixo de marcadores.
- XML é extensível.

```
<livro>
    <titulo>Inside XML</titulo>
    <autor>Steven Holzner</autor>
    <preco>R$ 150,00 </preco>
</livro>
```

Modelo de dados hierárquico da XML

- O objeto básico em XML é o documento XML.
- Conceitos de estruturação principais:
- Elementos
  - Os elementos são identificados em um documento por sua tag de início e tag de fim.
  - Os nomes de tag são delimitados por sinais < ... >.
  - As tags de fim são identificadas ainda por uma barra, </ ... >.

- Atributos.
  - Os atributos em XML oferecem informações adicionais que descrevem elementos.
  - Existem conceitos adicionais na XML, como entidades, identificadores e referências.

Representação de documentos XML

- É possível representar um documento XML usando:
  - Representação textual
  - Estrutura de árvore.
    - Nós internos representam elementos complexos, enquanto os nós de folha representam elementos simples.
    - É por isso que o modelo XML é conhecido como um modelo de árvore ou um modelo hierárquico.
    - Em geral, não existe limite sobre os níveis de aninhamento dos elementos.

Documento XML - Atributos

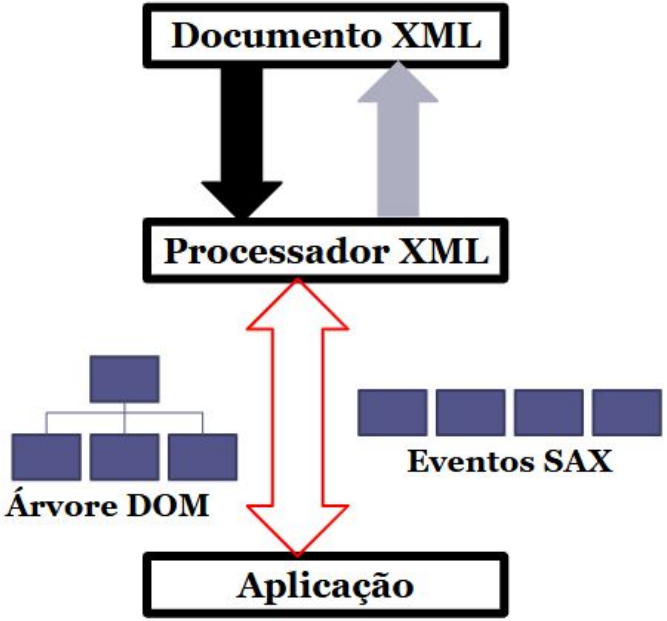
- Um elemento pode conter informação adicional sobre seu conteúdo armazenados em atributos.
  - Cada atributo é um par (nome, valor)
  - Valores dos atributos devem estar entre aspas

```
<livro isbn = "85.241.0591-9" >
    <titulo>Inside XML</titulo>
    <autor>Steven Holzner</autor>
    <preco>R$ 150,00 </preco>
</livro>
```

Estrutura de um documento XML

- Regras
  - Cada elemento possui um único pai (Exceção: O elemento raiz)
  - Cada elemento possui um número arbitrário de irmãos e filhos
    - Um elemento sem filho é denominado folha
  - Todo documento XML deve possuir uma raiz
  - Todas as tags devem ser fechadas
  - As tags devem estar bem aninhadas
  - As tags XML são “case sensitive”
  - Um elemento pode ter conteúdo vazio
    - <fone/>
    - <nada> </nada>

Manipulações



APIs para processamento de XML

- ElementTree (xml.etree.ElementTree): Biblioteca nativa e leve para processamento XML em Python.
  - Oferece métodos para analisar, modificar, e gerar XML.
  - É ideal para arquivos XML pequenos e médios, e possui uma interface simples.

```
import xml.etree.ElementTree as ET

tree = ET.parse('arquivo.xml')
root = tree.getroot()
for elem in root:
    print(elem.tag, elem.attrib)
```

APIs para processamento de XML

- Minidom (xml.dom.minidom): Também parte da biblioteca padrão de Python, fornece uma interface DOM para manipular XML.
  - Recomendado para casos em que é necessário ler e formatar XML, especialmente para visualização.
  - Gera XML com uma estrutura mais organizada e indentada.

```
from xml.dom import minidom

doc = minidom.parse("arquivo.xml")
root = doc.documentElement
print(root.tagName)
```

APIs para processamento de XML

- lxml: Uma biblioteca externa altamente eficiente e poderosa para processamento XML e HTML.
  - Baseada em libxml2 e libxslt, oferece recursos adicionais como XPath e XSLT, que são úteis para consultas complexas em XML.
  - É mais rápida e eficiente para lidar com grandes arquivos XML.

```
from lxml import etree

tree = etree.parse('arquivo.xml')
root = tree.getroot()
for element in root:
    print(element.tag, element.text)
```

APIs para processamento de XML

- BeautifulSoup (com parser lxml ou xml): Biblioteca útil para extrair dados de XML e HTML, facilitando o tratamento e a modificação de documentos XML com uma interface de busca intuitiva.
  - Exige a instalação adicional do BeautifulSoup4 e lxml ou html.parser.

```
from bs4 import BeautifulSoup

with open("arquivo.xml", "r") as file:
    soup = BeautifulSoup(file, "xml")
for tag in soup.find_all("tag_name"):
    print(tag.text)
```

APIs para processamento de XML

- xmltodict: Converte XML em dicionários Python de maneira simples, o que facilita a manipulação e o acesso aos dados.
  - Ideal para transformar XML em um formato JSON-like, permitindo integração com APIs que utilizam JSON.

```python
import xmltodict

with open("arquivo.xml") as file:
    data = xmltodict.parse(file.read())
print(data['root']['tag_name'])
```

CRUD persistindo em XML

```python
# Função para ler os dados do XML
def ler_dados_xml():
    produtos = []
    if os.path.exists(XML_FILE):
        tree = ET.parse(XML_FILE)
        root = tree.getroot()
        for elem in root.findall("produto"):
            produto = Produto(
                id=int(elem.find("id").text),
                nome=elem.find("nome").text,
                preco=float(elem.find("preco").text),
                quantidade=int(elem.find("quantidade").text)
            )
            produtos.append(produto)
    return produtos
```

A função `ler_dados_xml()` lê dados de um arquivo XML contendo informações de produtos. Ela verifica se o arquivo XML existe e, se existir:

1. Carrega o XML com `ElementTree` e obtém o elemento raiz.
2. Para cada elemento `<produto>` no XML, cria uma instância de `Produto`, preenchendo os atributos `id`, `nome`, `preco`, e `quantidade` com os dados encontrados.
3. Adiciona cada instância de `Produto` a uma lista `produtos`.
4. Retorna a lista `produtos` com todos os produtos extraídos.

CRUD persistindo em XML

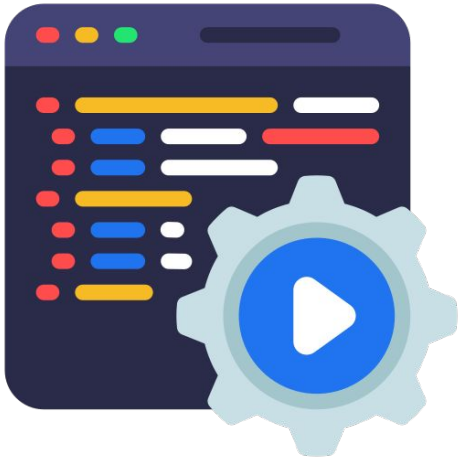
```
# Função para escrever os dados no XML
def escrever_dados_xml(produtos):
    root = ET.Element("produtos")
    for produto in produtos:
        produto_elem = ET.SubElement(root, "produto")
        ET.SubElement(produto_elem, "id").text = str(produto.id)
        ET.SubElement(produto_elem, "nome").text = produto.nome
        ET.SubElement(produto_elem, "preco").text =
str(produto.preco)
        ET.SubElement(produto_elem, "quantidade").text =
str(produto.quantidade)

    tree = ET.ElementTree(root)
    tree.write(XML_FILE)
```

A função `escrever_dados_xml(produtos)` grava uma lista de produtos em um arquivo XML. O processo é o seguinte:

1. Cria um elemento raiz `<produtos>`.
2. Para cada objeto `produto` na lista `produtos`:
   - Cria um subelemento `<produto>` dentro de `<produtos>`.
   - Adiciona subelementos `<id>`, `<nome>`, `<preco>`, e `<quantidade>` com os valores correspondentes convertidos para texto.
3. Cria uma árvore XML a partir do elemento raiz e escreve o conteúdo no arquivo XML (`XML_FILE`).
1.

Referências

- Elsmari, R., Navathe, Shamkant B. “Sistemas de Banco de Dados”. 6ª Edição, Pearson Brasil, 2011.
- Apostila FJ22 da Caelum – Cap. 4 - Trabalhando com XML
- https://fastapi.tiangolo.com/
- https://fastapi.tiangolo.com/tutorial/
- XML Tutorial – W3 Schools
  - http://www.w3schools.com/xml/
- XLST Tutorial – W3 Schools
  - http://www.w3schools.com/xsl/
  - www.xml.org
  - www.xml.xom
  - www.msdn.microsoft.com/xml
  - www.xmlsoftware.com
  - www.w3c.org



Obrigado!

Dúvidas?

Universidade Federal do Ceará - *Campus* Quixadá

Prof. Francisco Victor da Silva Pinheiro
victorpinheiro@ufc.br