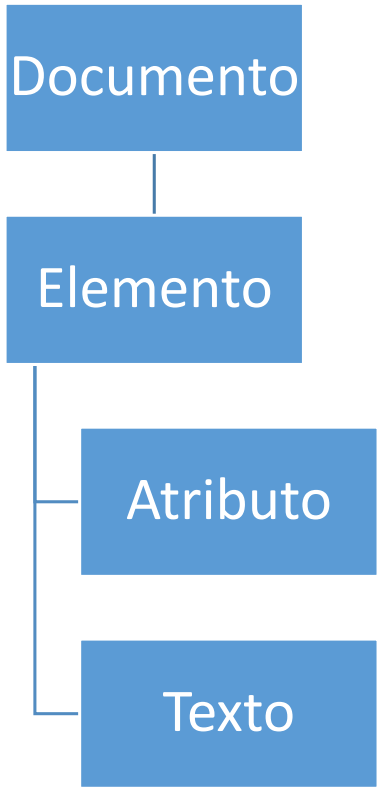
Manipulação de XML em Java
JaxP - *DOM e SAX*

Prof. Dr. Alexandre L'Erario

DOM – Document Object Model

- Padrão estruturado em árvore, independente de linguagem
- Lida com uma estrutura hierárquica
  - Utiliza amplamente os *parsers* de processamento
- Não utiliza notações
- Utiliza builder factory (padrão de projeto) como construtor principal – no caso de Java

DOM – Estrutura de documento



- Utiliza parsers para compor a estrutura de árvore
- No caso de Java, SAX

Sax – Simple API for XML

- JAXP - Java API for XML Processing
  - JaxP também tem DOM
- Orientado a Eventos
  - Você precisa desenvolver call-back's
- Acesso Serial
- Orientado para o processamento independente do estado
  - O manuseio de um elemento não depende dos elementos que vieram antes

https://docs.oracle.com/javase/tutorial/jaxp/sax/index.html

Sax - JaxP

- Com relação ao DOM: ➔ Recomenda o uso do DOM
  - Requer mais trabalho
  - Requer menos memória
  - Não tem representação interna de documento
- Sax não possui método próprio para geração de chaves hash
  - As classes in java.util and java.io são necessárias para o uso de Hash
- Não suporta StAx (Stream)

Manipular XML em Java

- É possível adotar mais de uma biblioteca
- Existem outras implementações além das mencionadas aqui
- Arquiteturas RESTFul tendem a não incorporar bibliotecas XML
  - Necessário incorporar manualmente
    - No pom.xml, por exemplo

Verificar se o XML é bem formatado

- Documento XML alocado em arquivo ou String

```
JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
        try {
            //cria um objeto da classe File referente ao arquivo XML
            jfc.showOpenDialog(jfc); File xmlfile = jfc.getSelectedFile();

            //Define a "fabrica" para objetos da classe DocumentBuilder
            DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

            //cria o objeto que vai processar o arquivo XML
            DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
           dBuilder.parse(xmlfile);
            System.out.println("O documento XML foi processado corretamente (bem-formado)");
```

Validar XML com XSD

```
try {
        JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
        jfc.setDialogTitle("Selecione o XML");  jfc.showOpenDialog(jfc);
        File xmlfile = jfc.getSelectedFile();
        DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
                jfc.setDialogTitle("Selecione agora o XSD"); jfc.showOpenDialog(jfc);
        File xsdfile =  jfc.getSelectedFile();
        SchemaFactory xsdFactory = SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);
        Schema schema = xsdFactory.newSchema(xsdfile);
        dbFactory.setSchema(schema);
        DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();
        dBuilder.parse(xmlfile);
        System.out.println("ok - documento válido");
    } catch (ParserConfigurationException | SAXException | IOException ex) {
        System.out.println("ERRO: " + ex.getMessage());
    }
```

Processar um XML

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Sessao>
  <Chat>
    <Participante id="1" nome="Alexandre"/>
    <Participante id="2" nome="Flavia"/>
    <Mensagem>
      <Data>2021-12-13Z</Data>
      <Hora>10:14:27.184Z</Hora>
      <Corpo>Oi como vai tudo bem?</Corpo>
    </Mensagem>
    <Mensagem>
      <Data>2021-12-13Z</Data>
      <Hora>10:15:27.184Z</Hora>
      <Corpo>Tudo!</Corpo>
    </Mensagem>
  </Chat>
</Sessao>
```

```
doc.getDocumentElement().normalize();
 NodeList nList = doc.getElementsByTagName("Sessao");

        Node nNode = nList.item(0);
        System.out.println("Elemento atual: " + nNode.getNodeName());

        NodeList nMensagens = nNode.getChildNodes();

        for (int i = 0; i < nMensagens.getLength(); i++) {
           Node nChat = nMensagens.item(i);
           if (nChat.getNodeName().equals("Chat")) {
              System.out.println("Estou no Chat");
              NodeList lChat = nChat.getChildNodes();
              for (int j = 0; j < lChat.getLength(); j++) {
                 System.out.println(lChat.item(j).getNodeName());
                 if (lChat.item(j).getNodeName().equals("Participante")) {
                    System.out.println("ID:" + lChat.item(j).getAttributes().getNamedItem("id").getNodeValue());
                    System.out.println("ID:" + lChat.item(j).getAttributes().getNamedItem("nome").getNodeValue());
                 }
                 if (lChat.item(j).getNodeName().equals("Mensagem")) {
                    NodeList lContent = lChat.item(j).getChildNodes();
                    for (int k = 0; k < lContent.getLength(); k++) {
                       if (!lContent.item(k).getNodeName().equals("#text")) {
                          System.out.println(lContent.item(k).getTextContent());
                       }}}}}}
```

Comparação geral

	JaxP		
	SAX	**DOM**	**JaxB**
Documento Objeto	N	S	S
Parser	S	S	N
LOC	Alto	Médio	Baixo
Nível	Baixo	Médio	Alto - Abstração
Dados manipulados em memória	S	S	S
Factory	S	S	S
Annotations	N	N	S
Acesso aleatório	N	S	S

Considerações finais

- Existem bibliotecas outras libs: jDom, Dom4j
- A retomada do padrão JaxB indica a migração do DOM/Sax para POJO