isec
Politécnico de Coimbra

Integração de Dados

Integração de Dados com XML – ESCRITORES

Docente

Anabela Borges Simões

Alunos

Paulo Henrique Figueira Pestana de Gouveia - a2020121705

Daniel Sacarrão - a2021113430

11 de junho de 2023

Índice

1 Introdução

Este trabalho apresenta o projeto desenvolvido no âmbito da disciplina de Integração de Dados, no 2º ano - 2º semestre do curso de Engenharia Informática, no ano letivo 2022/2023. O objetivo deste projeto foi criar uma aplicação em Java que permita a integração de dados provenientes de fontes heterogéneas, distribuídas e autónomas, apresentando-os de forma unificada ao utilizador.

O programa desenvolvido consiste em vários Wrappers, que foram implementados utilizando expressões regulares para extrair a informação desejada. Foram utilizadas duas fontes de dados principais: a página da Wikipédia e o site da Bertrand. A partir dessas fontes, foi extraída informação sobre escritores famosos e suas obras, que foram organizadas e integradas em dois arquivos XML: escritores.xml e obras.xml.

No decorrer deste relatório, serão abordadas as principais etapas do projeto, desde a análise das fontes de dados, definição do esquema global, implementação dos Wrappers, manipulação dos arquivos XML, validação do modelo adotado, realização de pesquisas utilizando XPath, geração de arquivos de output com XSLT e XQuery, até a implementação da interface gráfica. Serão apresentados os desafios encontrados, as decisões tomadas e as justificativas para as escolhas feitas, além da avaliação dos resultados alcançados.

A expectativa é que o projeto demonstre as competências adquiridas no que diz respeito à análise de situações de integração de dados, manipulação de XML, utilização de expressões regulares e desenvolvimento de soluções funcionais e eficazes para o problema proposto.

2 Estrutura do Trabalho

O trabalho está dividido em diferentes componentes. A seguir, está a lista dos principais componentes presentes no trabalho:

- Escritores.java(Classe Escritores)
- Obra.java(Classe Obras)
- main.java
- Frame.java(Interface Gráfica)
- Wrappers.java(Extração de Informações e Criação das Classes de Escritores e Obras)
- HttpRequestFunctions.java(Funções para Requisições HTTP)
- Functions.java(Função para Leitura de Arquivos)
- EscritoresXML.java(Manipula dados de escritores em formato XML)
- ObrasXML.java(Manipula dados de obras em formato XML)
- ValidarXML.java(Responsável por validar documentos XML com DTD ou XSD)
- JDOMFunctions_Validar.java(Responsável por validar documentos XML utilizando DTD ou XSD)
- XMLJDomFunctions.java(Classe que contém funções para ler e escrever documentos XML utilizando a biblioteca JDOM)
- XPathFunctions.java(Classe que contém funções para executar consultas XPath em documentos XML utilizando a biblioteca Saxon)
- JDOMFunctions_XSLT.java(Classe que contém funções para realizar transformações XSLT em documentos XML utilizando a biblioteca JDOM2)
- SaxonFunctions_XQuery.java(Classe que contém funções para executar consultas XQuery em documentos XML utilizando a biblioteca Saxon)

3 Implementação dos ficheiros

3.1 Wrappers.java

3.1.1 criaObra(String nome,int codigo)

Esta função recebe um nome e o código do escritor como parâmetros e retorna uma lista de objetos "Obra". Essa função realiza uma série de ações, como obter o ISBN de uma obra, obter o link da obra, obter o autor, o título, o preço, a capa e a editora da obra. Em seguida, ela cria um objeto "Obra"com essas informações e adiciona à lista de obras.

3.1.2 criaEscritor(String autor)

Esta função recebe um nome de autor como parâmetro e retorna um objeto "Escritores". Ela realiza várias ações, como obter o link do escritor, o nome, a nacionalidade, a foto, o género, a data de nascimento, a data de falecimento, os prémios, as ocupações e outras informações do escritor. Em seguida, ela cria um objeto "Escritores"com essas informações e retorna.

3.1.3 obtem_link(String isbn)

Esta função recebe um ISBN como parâmetro e retorna o link de uma obra. Ela realiza uma requisição HTTP para um site de pesquisa de livros (no exemplo, a Bertrand) com o ISBN como parâmetro de busca. Em seguida, ela analisa o conteúdo da página HTML usando regex em busca do link da obra.

3.1.4 obtem_link_escritor(String autor)

Esta função recebe o nome de um escritor como parâmetro e retorna o link do escritor na Wikipedia. Ela realiza uma requisição HTTP para a Wikipédia com o nome do autor como parâmetro de busca. Em seguida, retorna o link encontrado.

3.1.5 Restantes funções

As restantes funções seguem o mesmo padrão, recebendo os parâmetros necessários para realizar uma requisição HTTP e, em seguida, analisar o conteúdo da página HTML a fim de extrair as informações desejadas.

Funções restantes são:

- `obtem_isbn(String nome)`
- `obtem_titulo(String link, String autor)`
- `obtem_autor(String link)`
- `obtem_editora(String link)`
- `obtem_capa(String link)`
- `obtem_preco(String link)`
- `obtem_nome(String link)`
- `obtem_data_nascimento(String link)`
- `obtem_data_falecimento(String link)`
- `obtem_nacionalidade(String link)`
- `obtem_foto(String link)`
- `obtem_genero(String link)`
- `obtem_premios(String link)`
- `obtem_ocupacoes(String link)`

3.2 EscritoresXML.java

O ficheiro EscritoresXML.java contém uma classe chamada EscritoresXML, que fornece métodos para manipulação de escritores em um documento XML. O objetivo dessa classe é adicionar, remover, editar e verificar informações de escritores no documento XML.

3.2.1 Método adicionaEscritor(Escritores esc, Document doc)

Adiciona um novo escritor ao documento XML, incluindo suas informações como nome, data de nascimento, data de falecimento, nacionalidade, género literário, ocupações e prémios.

3.2.2 Método removeEscritor(String autor, Document doc)

Remove um escritor do documento XML com base em seu nome.

3.2.3 Método verificaEscritor(String autor, Document doc)

Verifica as informações de um escritor no documento XML com base em seu nome e retorna um mapa com os detalhes encontrados, como nome, data de nascimento, data de falecimento, nacionalidade, género literário, ocupações e prémios.

3.2.4 Método editaInformacaoEscritor(String escritor, String key, String novaInfo, Document doc)

Edita uma informação específica de um escritor no documento XML, com base em seu nome, chave da informação e novo valor fornecido.

3.2.5 Método getIdEscritor(String escritor, Document doc)

Obtém o ID de um escritor no documento XML com base em seu nome.

3.2.6 Método removeInformacao(String autor, String key, String atributo, Document doc)

Remove uma informação específica de um escritor no documento XML, com base em seu nome, chave da informação e valor fornecido.

3.2.7 Método adicionaInformacao(String autor, String key, String atributo, Document doc)

Adiciona uma nova informação a um escritor no documento XML, com base em seu nome, chave da informação e valor fornecido.

3.3 ObrasXML.java

3.3.1 Método adicionaObras(Obra obr, Document doc)

Adiciona uma nova obra ao documento XML, incluindo informações como código do autor, ISBN, título, autor, editora, capa e preço. A obra é representada por um objeto da classe Obra.

3.3.2 Método removeObras(String idAutor, Document doc)

Remove uma obra do documento XML com base no código do autor. Percorre todas as obras no documento e remove aquela que possui o código do autor correspondente.

3.4 Escritores.java e Obras.java

A classe Escritores representa escritores e contém atributos como id, nome, nacionalidade, fotografia, género, entre outros. Ela possui métodos para acessar e modificar esses atributos, como getters e setters. Além disso, a classe possui métodos para definir o contador de identificadores, obter o identificador do escritor e lidar com ocupações, prémios, datas de nascimento e morte, link e nome de pesquisa do escritor. Em resumo, a classe Escritores é usada para gerências de informações relacionadas aos escritores.

A classe Obras representa obras literárias e possui atributos como código do escritor, isbn, titulo, autor, editora, capa e preço. Ela fornece métodos para acessar e modificar esses atributos, como getters e setters. A classe também inclui um construtor para inicializar os atributos da obra. Em resumo, a classe Obras é utilizada para gerências das informações sobre obras literárias, permitindo o acesso e manipulação dos seus atributos.

3.5 JDOMFunctions_XSLT.java

A classe JDOMFunctions_XSLT contém várias funções relacionadas à manipulação e transformação de documentos XML usando a biblioteca JDOM e a linguagem XSLT.

As funções transformaDocumento e transformaDocumento2 foram as mesmas dadas nas aulas práticas.

3.5.1 Métodos escritoresFotosTabela, listarEscritores e mostarObras

As funções escritoresFotosTabela, listarEscritores e mostarObras são exemplos específicos de transformações XSLT aplicadas a documentos XML. Elas carregam o documento XML correspondente, aplicam a transformação XSLT especificada e geram um arquivo de saída, como um arquivo HTML ou um arquivo de texto. Esses arquivos podem ser visualizados ou utilizados posteriormente.

- escritoresFotosTabela aplica uma transformação XSLT em um documento XML de escritores e gera um arquivo HTML que exibe uma tabela com as fotos dos escritores.
- listarEscritores aplica uma transformação XSLT em um documento XML de escritores e gera um arquivo de texto que lista os escritores.
- (Extra)mostrarObras aplica uma transformação XSLT em um documento XML de obras e gera um arquivo HTML que exibe as informações das obras agrupadas por escritor ordenadas por preço.

3.6 SaxonFunctions_XQuery.java

A classe SaxonFunctions_XQuery contém várias funções relacionadas à execução de consultas XQuery usando a biblioteca Saxon.

As funções xQueryToText, xQueryToHtml e xQueryToXml foram as mesmas dadas nas aulas práticas.

3.6.1 Método top5ObrasCaras

A função top5ObrasCaras executa uma consulta XQuery que retorna as cinco obras mais caras e as armazena em um arquivo XML chamado "top5ObrasCaras.xml". Em seguida, lê o arquivo XML e retorna o documento correspondente.

3.6.2 Método escritoresObras

A função escritoresObras executa uma consulta XQuery que recebe o ID de um escritor como parâmetro e gera um arquivo HTML chamado "obrasEscritor.html"que exibe as obras desse escritor. O arquivo HTML é aberto em um navegador.

3.6.3 Método juntarEscritoresObras

A função juntarEscritoresObras executa uma consulta XQuery que combina dois documentos XML (escritores e obras) e gera um novo arquivo XML chamado "xmlJuntos.xml". Em seguida, lê o arquivo XML e retorna o documento correspondente.

3.6.4 Método avgPrecoEscritores

A função avgPrecoEscritores executa uma consulta XQuery que calcula a média de preços das obras de cada escritor e gera um arquivo XML chamado "avgPrecoEscritores.xml". Em seguida, lê o arquivo XML e retorna o documento correspondente.

3.7 XPathFunctions.java

A classe XPathFunctions contém várias funções relacionadas à execução de consultas XPath usando a biblioteca Saxon.

As funções executaXpath e listaResultado foram as mesmas dadas nas aulas práticas.

3.7.1 Método getLastIdFromXML

A função getLastIdFromXML recebe o nome de um arquivo XML como parâmetro e retorna o valor do último ID de escritor no arquivo. Ela usa uma consulta XPath para extrair o último ID do atributo "id"dos elementos "escritor".

3.7.2 Método pesquisa_nome_autor

A função pesquisa_nome_autor recebe o nome de um autor como parâmetro e realiza uma pesquisa no arquivo "escritores.xml"usando uma consulta XPath. Ela retorna uma lista de informações relevantes sobre o autor, como ID, nome, nacionalidade, data de nascimento, data de falecimento, género literário, ocupações e prémios.

3.7.3 Método pesquisa_nacionalidade_autor

A função pesquisa_nacionalidade_autor recebe uma nacionalidade como parâmetro e realiza uma pesquisa no arquivo "escritores.xml"usando uma consulta XPath. Ela retorna uma lista de nomes de autores que possuem a nacionalidade especificada.

3.7.4 Método pesquisa_obras_autor

A função pesquisa_obras_autor recebe o ID de um autor como parâmetro e realiza uma pesquisa no arquivo "obras.xml"usando uma consulta XPath. Ela retorna uma lista de obras associadas ao autor.

3.7.5 Método escritor_mais_premiado

A função escritor_mais_premiado realiza uma consulta XPath no arquivo "escritores.xml"para determinar o escritor mais premiado. Ela retorna o nome do escritor mais premiado.

3.7.6 Método livros_editora_preco

A função livros_editora_preco recebe o nome de uma editora e um valor mínimo de preço como parâmetros e realiza uma pesquisa no arquivo "obras.xml"usando uma consulta XPath. Ela retorna uma lista de livros publicados pela editora especificada com preço acima do valor mínimo.

3.7.7 Método (Extra)averagePriceOfAwardWinningAuthors

A função (Extra)averagePriceOfAwardWinningAuthors calcula a média dos preços das obras dos autores que possuem pelo menos um prémio. Ela realiza uma consulta XPath no arquivo "escritores.xml"para obter os IDs dos autores premiados e, em seguida, realiza consultas XPath no arquivo "obras.xml"para obter os preços das obras desses autores. A função retorna a média dos preços.

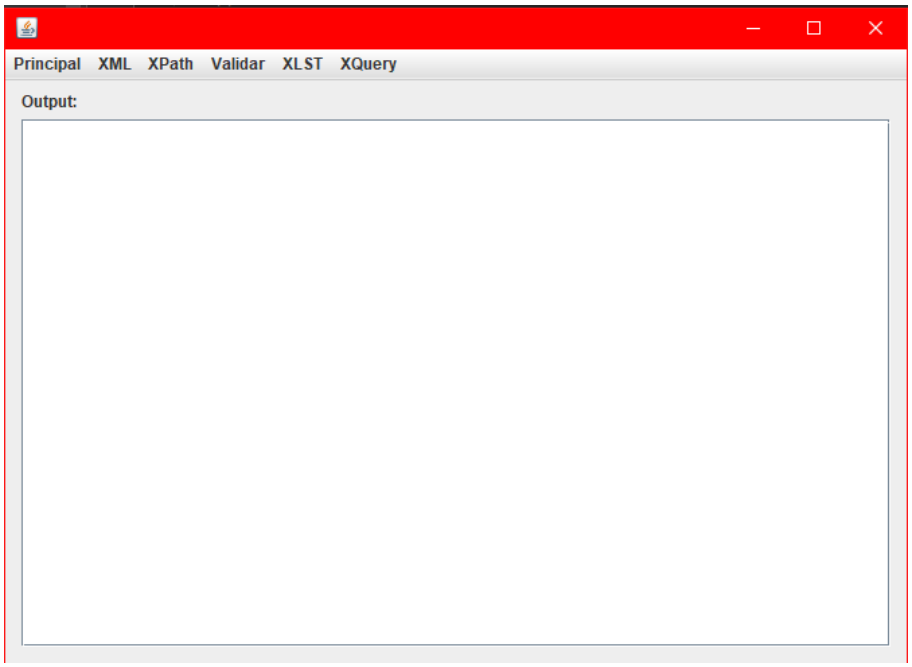
3.8 Restantes ficheiros em falta

Os restantes ficheiros:

- `HttpRequestFunctions.java`
- `Functions.java`
- `ValidarXML.java`
- `JDOMFunctions_Validar`
- `XMLJDomFunctions.java`

São ficheiros que foram fornecidos em aula e que não foram alterados em termos de código, apenas usados no contexto adequado.

4 Interface Gráfica



A interface deste trabalho permite visualizar o conteúdo do ficheiro XML, realizar operações como adicionar ou remover um escritor e também alterar os atributos de uma escritor existente.

Além disso, a interface suporta a validação do XML em relação a DTD (Documento de Definição de Tipo) e XSD (Esquema XML) para garantir a conformidade com as regras definidas.

Através da interface, é possível realizar pesquisas XPATH para buscar informações específicas com base nos atributos disponíveis. Também é possível realizar pesquisas XSLT para transformar o XML de acordo com regras definidas em um arquivo XSLT. É possível executar consultas XQuery para obter informações mais complexas do XML.

Por fim, a interface permite gerar saídas com os resultados obtidos em cada uma dessas operações.

4.1 Principal

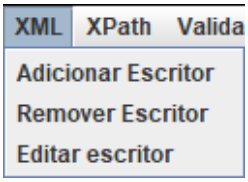
4.1.1 Abrir

Responsável por abrir um arquivo XML selecionado pelo usuário. Ele exibe uma caixa de diálogo para selecionar o arquivo, lê o conteúdo do arquivo e o exibe na interface do usuário. Se o arquivo não for um XML válido, exibe uma mensagem de aviso.

4.1.2 Sair

Responsável por encerrar a aplicação quando o item de menu "Sair"é selecionado.

4.2 XML



4.2.1 Adicionar Escritor

Ele é responsável por exibir uma caixa de diálogo chamada "adicionarEscritorDialog". É onde o utilizador introduz o nome do escritor a ser adicionado e as suas respetivas obras. Se suceder, são criados ou adicionados aos seus respetivos ficheiros "escritores.xml"e "obras.xml".

4.2.2 Remover Escritor

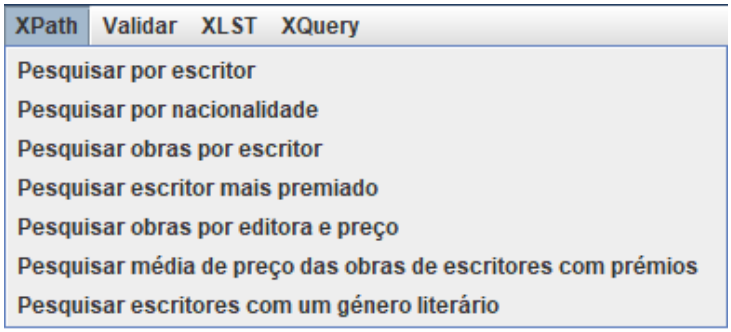
Ele é responsável por exibir uma caixa de diálogo chamada "removerEscritorDialog". É onde o utilizador introduz o nome do escritor a ser removido e as suas respetivas obras. Se suceder, são removidas dos respetivos ficheiros "escritores.xml"e "obras.xml".

4.2.3 Editar Escritor

Ele é responsável por exibir uma caixa de diálogo chamada "editarEscritorDialog". É onde o utilizador introduz o nome do escritor a ser editado, e são ativadas os atributos possíveis a alterar. Ainda podem ser adicionados ou removidos prémios, ocupações

e género literário do escritor se o desejar. Se sucedido, é guardado as alterações no ficheiro "escritores.xml".

4.3 XPath



4.3.1 Pesquisar por escritor

Realiza uma pesquisa pelo nome do escritor utilizando XPath adequado para extrair a informação associada ao mesmo.

4.3.2 Pesquisar por nacionalidade

Realiza uma pesquisa pelo nome da nacionalidade utilizando XPath adequado para extrair os escritores associados ao mesmo.

4.3.3 Pesquisar obras por escritor

Realiza uma pesquisa pelo nome do escritor utilizando XPath adequado para extrair o id associada ao mesmo, para depois fazer a ligação com as obras associadas a esse escritor.

4.3.4 Pesquisar escritor mais premiado

Realiza uma pesquisa no xml com XPath para extrair o escritor com mais prémios.

4.3.5 Pesquisar obras por editora e preço

Realiza uma pesquisa pelo nome da editora e o preço mínimo da obra utilizando XPath adequado para extrair as obras associadas a essas condições.

4.3.6 (Extra)Pesquisar média de preço das obras de escritores com prémios

Realiza primeiro uma pesquisa XPath no "escritores.xml"para extrair os ids dos escritores com pelo menos um prémio, em seguida, com outra pesquisa XPath percorre "obras.xml"para extrair os preços daquelas com id idêntico ao escritor, e no final, faz a média desse valor.

4.3.7 (Extra)Pesquisar escritores com um género literário

Realiza uma pesquisa pelo nome do género literário e utilizando XPath adequado para extrair os escritores associadas ao mesmo.

4.4 Validar

4.4.1 Escritores DTD

Valida o ficheiro "escritores.xml"usando o DTD.

4.4.2 Escritores XSD

Valida o ficheiro "escritores.xml"usando o XSD.

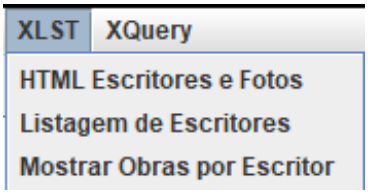
4.4.3 Obras DTD

Valida o ficheiro "obras.xml"usando o DTD

4.4.4 Obras XSD

Valida o ficheiro "obras.xml"usando o XSD.

4.5 XLST



4.5.1 HTML Escritores e Fotos

Transforma "escritores.xml"usando um ficheiro XSL chamado "HtmlEscritoresFotos.xsl". O resultado da transformação é armazenado em um novo documento.

Após a criação do ficheiro HTML, ele é aberto em um browser padrão usando a classe Desktop.

4.5.2 Listagem de Escritores

Lê o ficheiro "escritores.xml"e realiza uma transformação usando o ficheiro XSL "escritores.xsl".

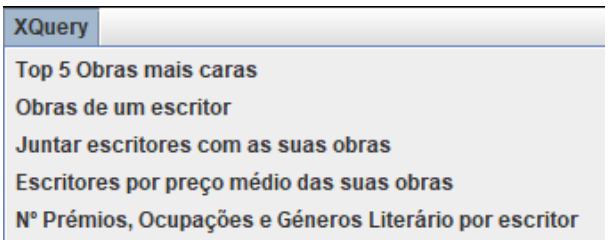
É criado um ficheiro ".txt"com os escritores presentes no xml.

4.5.3 Mostrar Obras por Escritor

Lê o ficheiro "escritores.xml"e realiza uma transformação usando o ficheiro XSL "escritores.xsl".

É criado um ficheiro ".txt"com os escritores presentes no xml.

4.6 XQuery



4.6.1 Top 5 Obras mais caras

Executa uma consulta XQuery para obter as cinco obras mais caras e retorna um documento XML com essas informações.

4.6.2 Obras de um escritor

Executa uma consulta XQuery para obter as obras de um determinado escritor identificado pelo seu Id. Em seguida, ele gera um arquivo HTML com o resultado da consulta e o abre no browser padrão.

4.6.3 Juntar escritores com as suas obras

Realiza uma junção de dois ficheiros XML, "escritores.xml"e "obras.xml", utilizando uma consulta XQuery chamada "juntarXML.xql". O resultado da junção é armazenado em um novo arquivo XML chamado "xmlJuntos.xml".

4.6.4 Escritores por preço médio das suas obras

Calcula a média dos preços das obras para cada escritor no arquivo XML "obras.xml"utilizando uma consulta XQuery chamada "avgPrecoEscritores.xql". O resultado é armazenado em um novo arquivo XML chamado "avgPrecoEscritores.xml".

4.6.5 Nº Prémios, Ocupações e Géneros Literário por escritor

Gera um relatório dos escritores utilizando uma consulta XQuery chamada "relatorioEscritores.xql"que tem o número de prémios, ocupações e géneros literários. O resultado da consulta é armazenado em um arquivo XML chamado "relatorioEscritores.xml".

5 Conclusão

Em conclusão, este trabalho nos permitiu aprimorar nossos conhecimentos em Integração de Dados, com foco especial na utilização de wrappers, geração de ficheiros XML e validações com DTD e XSD. Ao longo do projeto, fomos capazes de explorar diferentes técnicas e ferramentas para manipular e integrar dados em formato XML.

Utilizando wrappers, conseguimos encapsular a lógica de criação e manipulação de escritores e suas respetivas obras. Isso nos permitiu adicionar escritores e obras aos ficheiros "escritores.xml"e "obras.xml"de forma estruturada e consistente na geração de ficheiros XML.

Além disso, implementamos validações com DTD e XSD para garantir a conformidade dos ficheiros XML gerados. Isso nos ajudou a evitar erros e inconsistências nos dados, melhorando a qualidade e a integridade das informações armazenadas.

Ao longo do processo, adquirimos conhecimentos práticos sobre a utilização de consultas XPath,XSD e XQuery para buscar e transformar dados XML. Essas consultas foram essenciais para a geração de ficheiros conforme queríamos.

No geral, este projeto nos proporcionou uma experiência valiosa no campo da Integração de Dados e nos permitiu aplicar conceitos e técnicas aprendidas durante o curso. Estamos satisfeitos com os resultados obtidos e acreditamos que adquirimos habilidades que serão úteis em projetos futuros envolvendo a manipulação e integração de dados em formato XML.

Referências bibliográficas

[1] "Modelação e Design," Moodle/Documentos Teóricos.