Universidade Federal de Juiz de Fora
Departamento de Ciência da Computação
DCC146 – Aspectos Teóricos da Computação

Trabalho Prático - Primeira Entrega

Beatriz Cunha Rodrigues - 201776038
João Pedro Sequeto Nascimento - 201776022
Marcus Vinícius V. A. Cunha - 201776013
Milles Joseph M e Silva - 201776026

Introdução

O objetivo dessa documentação é descrever as funcionalidades desenvolvidas, o projeto de estrutura de dados e decisões de projeto realizadas para a primeira entrega do trabalho prático, assim como auxiliar na compilação e execução.

Equipe de Desenvolvimento

A equipe de desenvolvimento é formada por quatro alunos:

- João Pedro Sequeto
- Beatriz Cunha Rodrigues
- Marcus Vinícius V. A. Cunha
- Milles Magalhães

Linguagem De Programação

A linguagem de programação escolhida para o desenvolvimento do projeto foi o Java, utilizando a versão 8, com o uso da ferramenta Maven para gerenciamento de dependências e automação da Build.

Estruturas de Dados

No desenvolvimento do projeto foram utilizadas as seguintes estruturas de dados para as seguintes funcionalidades:

- HashMap: o HashMap armazena as informações em pares, ou seja, é possível armazenar uma chave e um valor correspondente. Utilizamos essa estrutura para armazenar a tag pela praticidade e organização para identificar e trabalhar com as mesmas. Com isso, armazenamos como “key” o nome da tag (exemplo: “EQUALS”) e armazenamos o seu “value” como o valor correspondente a tag (exemplo: “=”). Sendo assim, conseguimos manter as informações de forma correspondente usando apenas uma estrutura.

Funcionamento da aplicação

A aplicação tem por objetivo, nesta primeira parte, ler as tags informadas pelo usuário e salvá-las ou lê-las em arquivos. Inicia-se a aplicação com o usuário digitando um input, que pode servir como comando ou como uma tag nova e sua definição. Caso seja um comando (iniciando com dois pontos a string), os que já estão implementados são os a seguir:

- :q - Termina a aplicação;
- :c - Carrega um arquivo .lex com as definições de tags;
- :l - Lista as definições de tag válidas já processadas pela aplicação;
- :s - Salva as tags em arquivo “tags.txt” dentro da pasta /files do projeto.

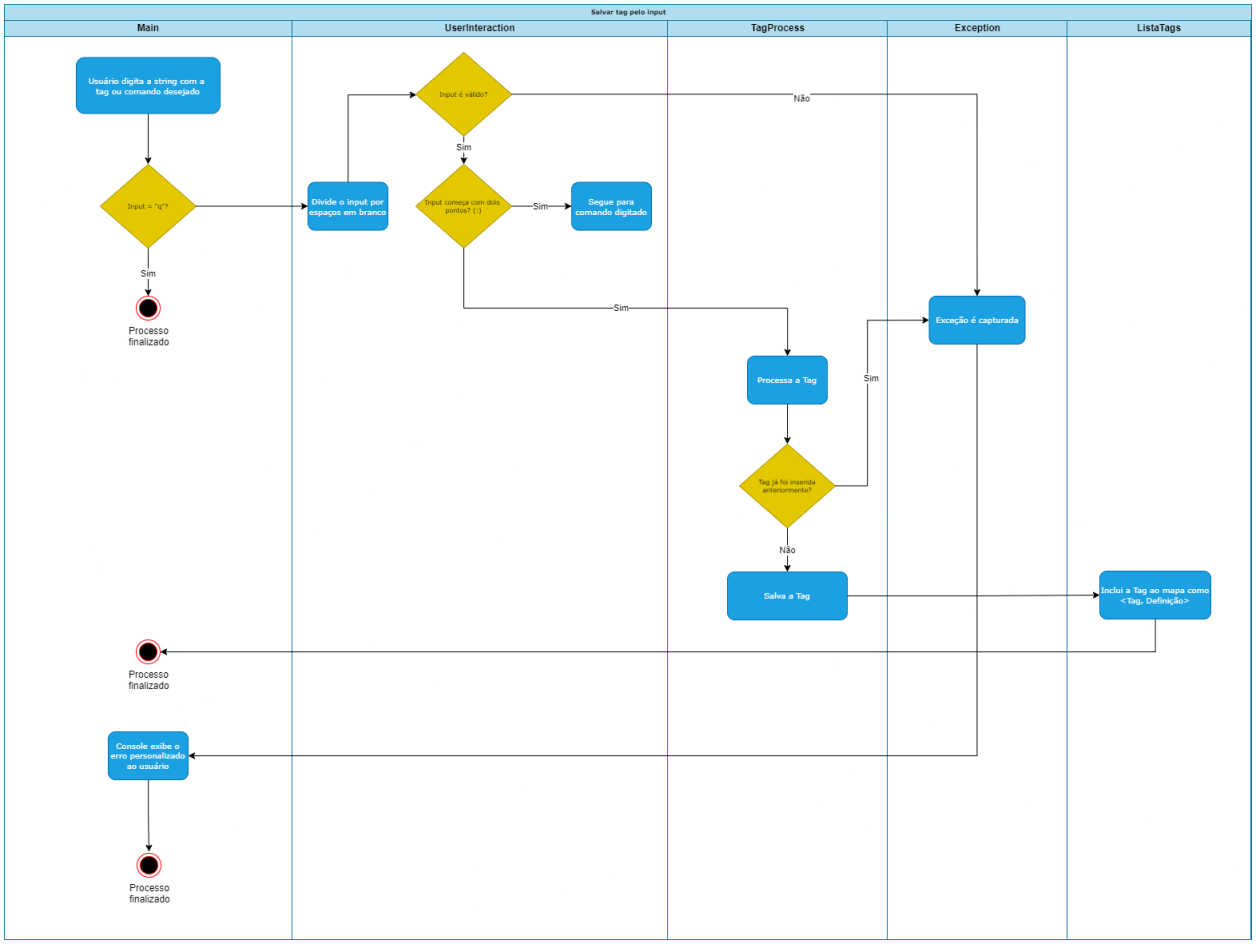
Todos os comandos que ainda não foram implementados apresentam a mensagem de aviso “Comando ainda não implementado” e, caso o comando seja inválido, haverá uma mensagem de erro “Comando não encontrado”.

Agora, na situação de não ser um comando, a aplicação irá entender que se trata de um nova tag e irá tentar processá-la. Após a validação do input para a nova tag, haverá uma verificação da existência da mesma pois uma tag não pode ser incluída duas vezes. Se não tiver sido processada ainda, esta será incluída no HashMap, sendo a chave o nome da tag e o seu valor a definição.

Em todas as situações de erro neste processamento (tag inválida ou já incluída), mensagens de erro personalizadas são mostradas ao usuário via console.

Diagrama de Sequência

O diagrama a seguir mostra o fluxo seguido para a funcionalidade mais crítica da primeira parte do trabalho, que é o processamento e inclusão da tag digitada pelo usuário e sua definição.



Módulos e Classes

O trabalho foi implementado em módulos para facilitar a implementação e manutenção do projeto. Os módulos e classes criados são os seguintes:

- Exceptions: módulo responsável pelas classes de exceção
    - InputErrorException: disparada caso seja digitado um input incorreto na tela de comandos (ex: string vazia ou fora do padrão).
- Interaction: módulo responsável pelas classes de interação de usuário;

    - UserInteraction: classe que filtra os comandos e definição de tag, caso aplicável.
- Model: módulo que contém as classes de domínio
    - ListaTags: classe que contém o mapa <Tag, Definição> e os métodos auxiliares.
- Resources: módulo que contém os processamentos dos elementos utilizados na aplicação
    - TagsProcess: classe que realiza o processamento da tag inserida no input para o mapa.
- Utils: módulo que contém classes auxiliares
    - FileUtils: classe que realiza gravação e escrita em arquivos;
    - IOUtils: classe para input e output em console, com mensagens de avisos, informações e erros em tela.

Página do projeto

- https://github.com/sequeto/Analisador-Lexico

Compilação e Execução

Para rodar o projeto, é necessário ter o JRE 8 (Java Runtime Enviroment) instalado. Sendo assim, basta abrir o prompt na pasta principal do projeto e digitar o seguinte comando:

- **java -jar analisador-0.0.1-SNAPSHOT-jar-with-dependencies.jar**