Introduction

Table of Contents

Motivation

Motivation

During the development of Camunda process applications you have to choose if your applications is either *using* Camunda Engine or *is* Camunda Engine. Depending on this decision, you are accessing Camunda via REST or Java API.

While Camunda Engine Core API provides well-designed and easy-accessible programming interface for Java, the usage of REST interface requires additional development. In order to enable the usage of REST API from Java and allow for easy integration into Spring Boot applications, the Camunda REST Client Spring-Boot library has been developed.

Features

Features

The library supports the following features:

- Usage of Open Feign library to allow for high-customizable REST client
- Provides a SpringBoot starter for usage in standalone client mode
- Provides a SpringBoot starter for usage inside a process application
- Decode HTTP error response and simulate exceptions, as if they are thrown locally by the service.
- Implemented Services:
  - `RuntimeService`
    - Process start by key: `#startProcessInstanceByKey()`
    - Process start by id: `#startProcessInstanceById()`
    - Process instance query: `#createProcessInstanceQuery()`
    - Message correlation: `#correlateMessage()`, `#createMessageCorrelation()`
    - Signal event: `#signalEventReceived()`, `#createSignalEvent()`
    - Execution trigger: `#signal()`
    - Read variables: `#getVariable()`,`#getVariables()`, `#getVariableTyped()`, `#getVariablesTyped()`
    - Read local variables: `#getVariableLocal()`,`#getVariablesLocal()`, `#getVariableLocalTyped()`, `#getVariablesLocalTyped()`
    - Write variables: `#setVariable()`,`#setVariables()`, `#setVariableTyped()`, `#setVariablesTyped()`
    - Delete variables: `#removeVariable()`,`#removeVariables()`
    - Write local variables: `#setVariableLocal()`,`#setVariablesLocal()`, `#setVariableLocalTyped()`, `#setVariablesLocalTyped()`
    - Delete local variables: `#removeVariableLocal()`,`#removeVariablesLocal()`
  - `RepositoryService`
    - Query for process definitions: `#createProcessDefinitionQuery()`
  - `TaskService`
    - Query for tasks: `#createTaskQuery()`

    - Task assignment: `#claim()`, `#defer()`, `#resolve()`, `#addCandidateGroup()`, `#deleteCandidateGroup()`, `#addCadidateUser()`, `#deleteCandidateUser()`
    - Identity links: `#addUserIdentityLink()`, `#addGroupIdentityLink()`, `#deleteUserIdentityLink()`, `#deleteGroupIdentityLink()`,
    - Task completion: `#complete()`
    - Task deletion: `#deleteTasks()`
    - Task attributes: `#setPriority()`, `#setOwner()`, `#setAssignee()`, `#saveTask()`
    - Handling Errors and Escalation: `#handleBpmnError()`, `#handleEscalation()`
    - Read variables: `#getVariable()`,`#getVariables()`, `#getVariableTyped()`, `#getVariablesTyped()`
    - Read local variables: `#getVariableLocal()`,`#getVariablesLocal()`, `#getVariableLocalTyped()`, `#getVariablesLocalTyped()`
    - Write variables: `#setVariable()`,`#setVariables()`, `#setVariableTyped()`, `#setVariablesTyped()`
    - Delete variables: `#removeVariable()`,`#removeVariables()`
    - Write local variables: `#setVariableLocal()`,`#setVariablesLocal()`, `#setVariableLocalTyped()`, `#setVariablesLocalTyped()`
    - Delete local variables: `#removeVariableLocal()`,`#removeVariablesLocal()`
- `ExternalTaskService` We are not aiming to replace the existing External Task Client, but still provide an alternative implementation for some methods.
    - Complete a task by id: `#complete()`
    - Handle BPMN Errors: `#handleBpmnError()`
    - Handle failures: `#handleFailure()`

Solution Idea

Solution Idea

The library uses the popular Java REST client OpenFeign embedded into Spring-Cloud-Feign-Starter and provides implementations of Java Camunda Engine Core API, accessing the remote engine via REST API.

In doing so, the remote version of the Camunda Engine API can be easily integrated in existing application using a SpringBoot Starter.

Depending on your usage scenario, you can choose from two starters shipped by the library.

Further outlook

Further outlook

Support query methods of `RuntimeService` for:

- executions
- process instances
- incidents

Support methods of `ManagementService` for:

- jobs
- batches

Is the library missing a feature important for you? Please report it.