Abrir un trabajador de trabajo

Recursos Relacionados

- Conceptos básicos de los trabajadores

Prerrequisitos

1. Ejecución de Zeebe Broker con punto final `localhost:26500` (predeterminado)
2. Ejecute el ejemplo Implementar un flujo de trabajo
3. Ejecute el ejemplo Crear una instancia de flujo de trabajo un par de veces

JobWorkerCreator.java

Fuente en github

```java
/*
 * Copyright Camunda Services GmbH and/or licensed to Camunda Services GmbH
under
 * one or more contributor license agreements. See the NOTICE file distributed
 * with this work for additional information regarding copyright ownership.
 * Licensed under the Zeebe Community License 1.0. You may not use this file
 * except in compliance with the Zeebe Community License 1.0.
 */
package io.zeebe.example.job;

import io.zeebe.client.ZeebeClient;
import io.zeebe.client.ZeebeClientBuilder;
import io.zeebe.client.api.response.ActivatedJob;
import io.zeebe.client.api.worker.JobClient;
import io.zeebe.client.api.worker.JobHandler;
import io.zeebe.client.api.worker.JobWorker;
import java.time.Duration;
import java.util.Scanner;

public class JobWorkerCreator {
  public static void main(final String[] args) {
    final String broker = "127.0.0.1:26500";

    final String jobType = "foo";

    final ZeebeClientBuilder builder =

ZeebeClient.newClientBuilder().brokerContactPoint(broker).usePlaintext();

    try (ZeebeClient client = builder.build()) {

      System.out.println("Opening job worker.");

      final JobWorker workerRegistration =
          client
              .newWorker()
              .jobType(jobType)
              .handler(new ExampleJobHandler())
              .timeout(Duration.ofSeconds(10))
              .open();

      System.out.println("Job worker opened and receiving jobs.");

      // call workerRegistration.close() to close it

      // run until System.in receives exit command
      waitUntilSystemInput("exit");
    }
  }

  private static void waitUntilSystemInput(final String exitCode) {
    try (Scanner scanner = new Scanner(System.in)) {
      while (scanner.hasNextLine()) {
        final String nextLine = scanner.nextLine();
        if (nextLine.contains(exitCode)) {
          return;
        }
```

```
      }
    }
  }

  private static class ExampleJobHandler implements JobHandler {
    @Override
    public void handle(final JobClient client, final ActivatedJob job) {
      // here: business logic that is executed with every job
      System.out.println(job);
      client.newCompleteCommand(job.getKey()).send().join();
    }
  }
}
```