apache-camel-spring-boot-1.x (0.0.1)

Maksim Kostromin

Version 0.0.1, 2018-06-15 01:29:31 UTC

Table of Contents

Travis CI status: [Build Status]

Chapter 1. Introduction

Reference documentation

See Travis CI integration tests for details

generated by generator-jvm yeoman generator (java-spring-boot_1.x)

Chapter 2. Implementation

2.1. camel spring configuration

src/main/resources/application-camel.properties

```
# spring boot keep running if you not using spring-boot-web for example...
camel.springboot.main-run-controller=true
# actuator support:
spring.profiles.active=camel
management.context-path=/actuator/
management.security.enabled=false
# we wanna also post to camel endpoints (curl -XPOST ...):
endpoints.camelroutes.read-only=false
```

2.2. move all files from directory

source directory: `/tmp/camel-in/` destination directory: `/tmp/camel-out/`

src/main/java/daggerok/app/MoveFilesCamelConfig.java

```
@Configuration
@RequiredArgsConstructor
public class MoveFilesCamelConfig {

  @Bean
  public RouteBuilder moveFilesRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {
        from("file:///tmp/camel-in")
            .routeId("dir-to-dir")
            .to("file:///tmp/camel-out");
      }
    };
  }
}
```

2.3. move files via jms

source directory: `/tmp/camel-jms-in/` process queue: `files` destination directory: `/tmp/camel-jms-out/`

move files via jms queue `jms:queue:files`

```
@Slf4j
@Configuration
@RequiredArgsConstructor
public class MoveFilesViaJmsCamelConfig {

  @Bean
  public RouteBuilder moveFilesViaJmsRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {

        from("file:///tmp/camel-jms-in")
            .routeId("in-dir-to-jms")
            .transform()
            .body(GenericFile.class, genericFile -> {
              final File file = File.class.cast(genericFile.getFile());
              try (final FileInputStream inputStream = new FileInputStream(file)) {
                final InputStreamReader streamReader = new InputStreamReader
(inputStream);
                final BufferedReader in = new BufferedReader(streamReader);
                return in.lines().collect(joining());
              } catch (IOException e) {
                log.error(e.getLocalizedMessage());
                throw new RuntimeException(e);
              }
            })
            .to("jms:queue:files")
        ;

        from("jms:queue:files")
            .routeId("jms-to-out-dir")
            .to("file:///tmp/camel-jms-out")
        ;
      }
    };
  }
}
```

jms component customizer

```
@Component
@RequiredArgsConstructor
class MoveFilesViaJmsComponentCustomizer implements ComponentCustomizer<JmsComponent>
{

  final ConnectionFactory connectionFactory;

  @Override
  public void customize(JmsComponent component) {
    component.setConnectionFactory(connectionFactory);
  }
}
```

2.4. change filename by using camel header

source directory: `/tmp/camel-filename-in/` destination directory: `/tmp/camel-filename-out/`

setup camel header to change output filename in pattern: `uuid-changed.txt`

```
@Configuration
@RequiredArgsConstructor
public class ChangeFilenameHeaderCamelConfig {

  @Bean
  public RouteBuilder filenameHeaderRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {
        from("file:///tmp/camel-filename-in")
            .routeId("change-filename-header")
            .setHeader("CamelFileName", () -> format("%s-changed.txt", UUID.
randomUUID().toString()))
            .to("file:///tmp/camel-filename-out");
      }
    };
  }
}
```

2.5. process exchange

source directory: `/tmp/camel-exchange-in/` destination directory: `/tmp/camel-exchange-out/`

man in a middle: do additional stuff with message during camel pipeline (for instance, log message body):

```
@Configuration
@RequiredArgsConstructor
public class ExchangeProcessorCamelConfig {

  @Bean
  public RouteBuilder processExchangeRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {
        from("file:///tmp/camel-exchange-in")
            .routeId("exchange-processor")
            .process().exchange(exchange -> {
              final Message inputMessage = exchange.getIn();
              final Object body = inputMessage.getBody(String.class);
              log.info("handling message body in exchange consumer: {}", body);
            })
            .to("file:///tmp/camel-exchange-out");
      }
    };
  }
}
```

2.6. easier exchange

source directory: `/tmp/camel-easier-exchange-in/` destination directory: `/tmp/camel-easier-exchange-out/`

same, but easier:

```
@Configuration
@RequiredArgsConstructor
public class EasierExchangeProcessorCamelConfig {

  @Bean
  public RouteBuilder easierExchangeRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {
        from("file:///tmp/camel-easier-exchange-in")
            .routeId("easier-exchange")
            .process()
            .message(message -> {
              final String body = message.getBody(String.class);
              final Map<String, Object> headers = message.getHeaders();
              log.info("easy body: {}", body);
              headers.entrySet().parallelStream()
                  .filter(e -> "CamelFileLength".contains(e.getKey()))
                  .forEach(e -> log.info("header({}): {}", e.getKey(), e.getValue()));
            })
            /* // absolutely same, even more easier!
            .body(String.class, (body, headers) -> {
              log.info("easiest: {}", body);
              headers.entrySet().parallelStream()
                  .filter(e -> "CamelFileLength".contains(e.getKey()))
                  .forEach(e -> log.info("header({}): {}", e.getKey(), e.getValue()));
            })
            */
            .to("file:///tmp/camel-easier-exchange-out");
      }
    };
  }
}
```

2.7. choice

source directory: `/tmp/camel-choice-in/` error directory: `/tmp/camel-choice-error/` destination directory: `/tmp/camel-choice-out/`

choice-when-<when-action>-otherwise-<otherwise-action>-endChoice

```
@Configuration
@RequiredArgsConstructor
public class ChoiceCamelConfig {

  @Bean
  public RouteBuilder choiceRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {
        from("file:///tmp/camel-choice-in")
            .routeId("choice")
            //@formatter:off
            .choice()
              .when(exchange -> !exchange.getMessage().getBody(String.class)
.toLowerCase().contains("err"))
                .to("file:///tmp/camel-choice-out")
              .otherwise()
                .to("file:///tmp/camel-choice-error")
            .endChoice()
            //@formatter:on
        ;
      }
    };
  }
}
```

Chapter 3. Actuator

get routes:

```
http get :8080/actuator/camel/routes
```

output:


```
[
    {
        "id": "change-filename-header",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142840
    },
    {
        "id": "choice",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142831
    },
    {
        "id": "easier-exchange",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142824
    },
    {
        "id": "exchange-processor",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142813
    },
    {
        "id": "dir-to-dir",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142798
    },
    {
        "id": "garbage-dir-to-dir",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142787
    },
    {
        "id": "in-dir-to-jms",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142774
```

```
    },
    {
        "id": "jms-to-out-dir",
        "status": "Started",
        "uptime": "2 minutes",
        "uptimeMillis": 142003
    }
]
```

now let's say some processing is working incorrectly, so we wanna stop `change-filename-header` *route:*

```
http post :8080/actuator/camel/routes/change-filename-header/stop

# output:
HTTP/1.1 200
Content-Length: 0
Date: Fri, 15 Jun 2018 00:10:07 GMT
X-Application-Context: application:spring-boot,camel
```

let's check if it was stopped:

```
http get :8080/actuator/camel/routes/change-filename-header/info
```

output:

```
{
    "id": "change-filename-header",
    "status": "Stopped",
    "uptimeMillis": 0
}
```

Chapter 4. Spring Integration

We also can integration awesome Apache Camel modules with Spring Integration

input directory: `/tmp/camel-spring-integration-in/`

1. again, do not forget about required jms coniguration

```
@Component
@RequiredArgsConstructor
class SpringIntegrationJmsComponentCustomizer implements ComponentCustomizer
<JmsComponent> {

  final ConnectionFactory connectionFactory;

  @Override
  public void customize(JmsComponent component) {
    component.setConnectionFactory(connectionFactory);
  }
}
```

2. if we wanna get file content form input directory `/tmp/camel-spring-integration-in` *and send it to* spring-integration for IntegrationFlow processing, then, camel configuration might looks like so:

```
@Configuration
@RequiredArgsConstructor
public class SpringIntegrationCamelConfig {

  @Bean
  public RouteBuilder springIntegrationRoute() {
    return new RouteBuilder() {
      @Override
      public void configure() throws Exception {

        from("file:///tmp/camel-spring-integration-in")
            .routeId("to-spring-integration")
            //@formatter:off
            .transform()
              .body(String.class, (s, headers) -> s)
            //@formatter:on
            .to("jms:queue:flow")
        ;

        from("jms:queue:flow")
            .to("spring-integration:inputMessageChannel")
        ;
      }
    };
  }
}
```

3. finally, spring IntegrationFlow config:

```
/**
 * Requires camel spring-integration-starter
 */
@Slf4j
@Configuration
class SpringIntegrationFlowConfig {

  @Bean
  public MessageChannel inputMessageChannel() {
    return MessageChannels.direct().get();
  }

  @Bean
  public IntegrationFlow inputMessageFlow() {
    return IntegrationFlows
        .from(inputMessageChannel())
        .handle(message -> {
          log.info("integration flow: {}", message.getPayload());
        })
        .get();
  }
}
```

Chapter 5. BOM

maven

```
  <dependencyManagement>
    <dependencies>
      <!-- now we can easily use compatible apache camel components -->
      <dependency>
        <groupId>org.apache.camel</groupId>
        <artifactId>camel-spring-boot-dependencies</artifactId>
        <version>${apache.camel.version}</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
```

gradle

```
dependencyManagement {
  imports {
    mavenBom "org.apache.camel:camel-spring-boot-dependencies:$apacheCamelVersion"
```

Chapter 6. Links

GitHub repo GitHub pages