OX2OX Migration Framework Scheduler Technical Documentation for 2.0.0

2021-03-08

Copyright notice

Contents

1 General Information

1.1 Delivery Comment

This delivery was requested with following comment:

OMF Scheduler 2.0.0 Feature Delivery

1.2 Install Package Repository

This delivery is part of a restricted software repository:

https://software.open-xchange.com/components/omf-scheduler/stable/2.0.0/RHEL7
https://software.open-xchange.com/components/omf-scheduler/stable/2.0.0/DebianStretch
https://software.open-xchange.com/components/omf-scheduler/stable/2.0.0/DebianBuster

1.3 Build Dependencies

This delivery was build and tested with following dependencies:

```
RedHat:RHEL-7,Debian:Stretch,Debian:Buster
```

2 Shipped Packages and Version

2.1 Package open-xchange-omf-orchestrator

OMF Orchestrator CLI to interoperate with the OX2OX Migration Framework.
Version: 2.0.0-5
Type: Other

2.1.1 Installation

Install on nodes with package installer **apt-get** or **yum**:

```
<package installer> install open-xchange-omf-orchestrator
```

2.2 Package open-xchange-omf-scheduler

OMF Scheduler OX2OX Migration Framework Scheduler.
Version: 2.0.0-5
Type: Other

2.2.1 Installation

Install on nodes with package installer **apt-get** or **yum**:

```
<package installer> install open-xchange-omf-scheduler
```

A Configuration Files

File 1 /opt/open-xchange/omf/scheduler/etc/omf-scheduler.yml

```
# https://docs.micronaut.io/latest/guide/config.html#configurationProperties
---
```

```
micronaut:
  # SSL configuration
  # Required for production environments.
  # See https://docs.micronaut.io/latest/guide/index.html#https for details.
  ssl:
    enabled: true
    port: 8443
    key-store:
      path: file:/opt/open-xchange/omf/certs/keystore.p12
      type: PKCS12
      password: secret
  server:
    dual-protocol: true
    port: 8080

  http:
    services:
      # The omf-source service is use to collect health status and metrics from the Source
      # OMF nodes. The HTTP client can be configured (ex: ssl) here by referencing
      # https://docs.micronaut.io/latest/guide/configurationreference.html#io.micronaut.http.client.ServiceHttpClientConfiguration
      # and the subsequent sections related to micronaut.http.services.*
      omf-source:
        # Example SSL configuration in case a source uses a private certificate
        # ssl:
        #   trust-store:
        #     path: file:/opt/open-xchange/omf/certs/source.p12
        #     type: PKCS12
        #     password: secret
  application:
    name: omf-scheduler
  # Configure security including basic auth: https://micronaut-projects.github.io/micronaut-security/latest/guide/#basicAuth
  # Must be set to true or the Source Controller is not secure
  security:
    enabled: true
    # Change the security of the open api views to anonymous so that they can be viewed without credentials
    intercept-url-map:
      - pattern: /swagger/**
        access:
          - isAnonymous()
      - pattern: /swagger-ui/**
        access:
          - isAnonymous()
      - pattern: /rapidoc/**
        access:
          - isAnonymous()
      - pattern: /redoc/**
        access:
          - isAnonymous()
  # https://docs.micronaut.io/latest/guide/index.html#_configuring_caches
  #caches:
    #example:
      #charset: UTF-8
      #expire-after-access: 1h
  metrics:
    enabled: true
    export:
      # Creates an endpoint like http://host/prometheus - uses basic auth from
      # credentials under scheduler.http.admin
      prometheus:
        enabled: true
        step: PT1M
        descriptions: true
  router:
    # Adds api versioning: https://docs.micronaut.io/latest/guide/index.html#apiVersioning
    versioning:
      enabled: true
      parameter:
        enabled: true
        names: 'v'
```

```
      header:
        enabled: true
        names: 'X-API-VERSION'
    # Allows the openapi views to be seen
    static-resources:
      swagger:
        paths: classpath:META-INF/swagger
        mapping: /swagger/**
      redoc:
        paths: classpath:META-INF/swagger/views/redoc
        mapping: /redoc/**
      rapidoc:
        paths: classpath:META-INF/swagger/views/rapidoc
        mapping: /rapidoc/**
      swagger-ui:
        paths: classpath:META-INF/swagger/views/swagger-ui
        mapping: /swagger-ui/**
---
scheduler:
  id: 'scheduler-0'
  hostname: ''
  batch:
    presync:
      # Max number of contexts in a batch
      size: 10
      # Strategy to use when creating batches.
      # Current supported strategies:
      #    - fill-first: create batches up to the batch size then create the next batch
      #    - fill-equal: create batches of equal size
      strategy: fill-equal
    cutover:
      size: 10
      strategy: fill-equal
    preprovisioning:
      size: 10
      strategy: fill-equal
  kafka:
    queues:
      batch: "omf-batch"
      response: "omf-response"
    resize:
      batch: true
      response: true
  http:
    admin:
      # Basic auth creds
      username: admin
      password: secret
      controller:
        path: /omf/scheduler/admin
    migration:
      controller:
        path: /omf/scheduler/migration
  metrics:
    cache:
      windows.millis: 300000
      batches.millis: 300000
---
jackson:
  bean-introspection-module: true
  serialization:
    indent-output: true
    writeDatesAsTimestamps: false
---
datasources:
  # Used to persist scheduling data
  scheduler:
    # url should use createDatabaseIfNotExist=true if the database will not
    # already exist: https://dev.mysql.com/doc/connector-j/8.0/en/connector-j-reference-
        configuration-properties.html
    url: jdbc:mysql://localhost:3306/scheduler?createDatabaseIfNotExist=true
    username: root
```

```
    password: my-secret-pw
    driverClassName: org.mariadb.jdbc.Driver
    dialect: MYSQL
    maximumPoolSize: 10
    maxLifetime: 180000
  # Used to create/drop databases for sources. This is not really the "default" data source
  # but we need to use default because of bug https://github.com/micronaut-projects/micronaut-data/issues/598s
  default:
    url: jdbc:mysql://localhost:3306/
    username: root
    password: my-secret-pw
    driverClassName: org.mariadb.jdbc.Driver
    dialect: MYSQL
    maximumPoolSize: 5
    maxLifetime: 180000
---
endpoints:
  loggers:
    enabled: true
    sensitive: true
  health:
    discovery-client:
      enabled: false
  liquibase:
    # fails with missing transition, might be fixed in later Micronaut releases
    enabled: false
  info:
    enabled: true
    sensitive: true
    sourceCodeOrigin:
      enabled: true
      location: file:/opt/open-xchange/omf/scheduler/share/SourceCodeOrigin.txt
---
zookeeper:
  server: zookeeper:2181
---
kafka:
  bootstrap:
    servers: kafka-1:9092, kafka-2:9092, kafka-3:9092
  producers:
    batch-producer:
      enable.idempotence: true
      # This enables transactions for the Batch Producer
      # The value must be unique per application, but should
      # not change for the same app after a crash, etc.
      transactional.id: producer-1
  #consumers:
    #response-consumer:
---
liquibase:
  datasources:
    scheduler:
      change-log: 'classpath:liquibase/scheduler/liquibase-changelog.xml'
---
jooq:
  datasources:
    default:
      sql-dialect: 'MARIADB'
    scheduler:
      sql-dialect: 'MARIADB'
---
logger:
  levels:
    ROOT: INFO
    com.openxchange: INFO
    omf: INFO
    omf.scheduler.admin.AuthenticationProviderUserPassword: WARN
---
```

File 2 /opt/open-xchange/omf/orchestrator/etc/omf-orchestrator.yml

```
micronaut:
  application:
    name: omf
  http:
    services:
      # The OMF Source HTTP REST API service configuration
      # The url is dynamic and should not be specified here
      source:
        # If some sources don't have valid public certificates (e.g. use
        # self-signed certificates), then their certificates can be added to
        # a custom Java trust-store using the keytool command and configured
        # here.
        #
        # Example:
        # ssl:
        #   enabled: true
        #   trust-store:
        #     path: file:/opt/open-xchange/omf/certs/source.p12
        #     password: secret
        #     type: PKCS12

      # The OMF Scheduler Source HTTP REST API service configuration
      scheduler-admin-source:
        url: "${omf.scheduler.url}/omf/scheduler/admin/source/"

        # If the scheduler does not have a valid public certificate
        # (e.g. uses a self-signed certificate), then its certificate can be
        # configured here.
        ssl:
          enabled: true
          trust-store:
            path: file:/opt/open-xchange/omf/certs/scheduler.p12
            password: secret
            type: PKCS12

      scheduler-admin-target:
        url: "${omf.scheduler.url}/omf/scheduler/admin/target/"

        # If the scheduler does not have a valid public certificate
        # (e.g. uses a self-signed certificate), then its certificate can be
        # configured here.
        ssl:
          enabled: true
          trust-store:
            path: file:/opt/open-xchange/omf/certs/scheduler.p12
            password: secret
            type: PKCS12

      # The OMF Scheduler Migration HTTP REST API service configuration
      scheduler-migration:
        url: "${omf.scheduler.url}/omf/scheduler/migration/"

        # If scheduler-admin above has a custom SSL configuration,
        # then it needs to be repeated here.
        ssl:
          enabled: true
          trust-store:
            path: file:/opt/open-xchange/omf/certs/scheduler.p12
            password: secret
            type: PKCS12

omf:
  source:
    # List source api username and passwords by identifying them
    # with the name that will be used to create the source entry in OMF.
    #
    # This is not required, and the username and password can be entered
    # for each command when working with the source.
    #
    # Example:
```

```
    # mysource:
    #   username: admin
    #   password: secret
  scheduler:
    # Credentials for the scheduler
    # On multi-user systems, specifying the password in a configuration file
    # with proper file system permissions is preferred to specifying it on
    # the command line, since the command line is visible to all local users.
    #
    # Example:
    # username: admin
    # password: secret

    # Location of the scheduler. Only the protocol and host name need to be
    # specified.
    url: "https://localhost:8443"
  ui:
    color: true
    unicode: true
    expandIds: false
```