PostgreSQL demo

This demo shows how to run midPoint container with PstgreSQL repository implementation.

The image can be found in the Evolveum/midpoint-docker GitHub project.

Starting

```
$ cd demo/postgresql
$ docker-compose up --build
```

After `docker-compose up` command successfully finishes you should see something like this on the console:

```
postgresql_midpoint_server_1  | 2019-02-22 15:07:50,222 [] [main] INFO (org.springframework.boot.web.embedded.
tomcat.TomcatWebServer): Tomcat started on port(s): 8080 (http) with context path '/midpoint'
postgresql_midpoint_server_1  | 2019-02-22 15:07:50,230 [] [main] INFO (com.evolveum.midpoint.web.boot.
MidPointSpringApplication): Started MidPointSpringApplication in 74.425 seconds (JVM running for 77.109)
```

Now you can log into midPoint using http://localhost:8080/midpoint URL, with an user of `administrator` and a password of `5ecr3t`.

You can safely ignore console messages like this:

```
postgresql_midpoint_data_1  | ERROR:  could not serialize access due to read/write dependencies among
transactions
postgresql_midpoint_data_1  | DETAIL:  Reason code: Canceled on identification as a pivot, during write.
postgresql_midpoint_data_1  | HINT:  The transaction might succeed if retried.
```

This is a part of standard midPoint conflict resolution process. The mentioned transactions are really retried and they succeed eventually.

Containers

The `demo/postgresql` composition contains the following containers:

Container name	Description
`postgresql_midpoint_server_1`	This is the standard container providing midPoint functionality. It contains standalone Tomcat running midPoint application.
`postgresql_midpoint_data_1`	This container hosts midPoint repository; this time it is implemented on PostgreSQL 9.5 database.

Communication

The containers publish the following TCP ports. (*Port mapped to localhost* denotes the mapping of container port to the host port where it can be reached from the outside.)

Container	Port number	Port mapped to localhost	Description
`postgresql_midpoint_server_1`	8080	8080	HTTP port to be used to connect to midPoint application
`postgresql_midpoint_data_1`	5432	5432	Port used to connect to the PostgreSQL database

Docker volumes

The following volumes are created to persist data and other relevant files.

Volume name	Description	Used by container
`postgresql_midpoint_home`	The midPoint home directory. Contains schema extensions, logs, custom libraries, custom ConnId connectors, and so on.	`postgresql_midpoint_server_1`
`postgresql_midpoint_data`	Volume hosting PostgreSQL database used by midPoint.	`postgresql_midpoint_data_1`

Configuring the composition

The following configuration properties are supported. Please refer to the main documentation page for their explanation.

Property	Default value
`REPO_DATABASE_TYPE`	`postgresql`
`REPO_JDBC_URL`	`default`
`REPO_HOST`	`midpoint_data`
`REPO_PORT`	`default`
`REPO_DATABASE`	`midpoint`
`REPO_USER`	`midpoint`
`REPO_PASSWORD_FILE`	`/run/secrets/mp_database_password.txt`
`REPO_MISSING_SCHEMA_ACTION`	`create`
`REPO_UPGRADEABLE_SCHEMA_ACTION`	`stop`
`REPO_SCHEMA_VERSION_IF_MISSING`	
`REPO_SCHEMA_VARIANT`	
`MP_MEM_MAX`	`2048m`
`MP_MEM_INIT`	`1024m`
`MP_JAVA_OPTS`	
`MP_KEYSTORE_PASSWORD_FILE`	`/run/secrets/mp_keystore_password.txt`
`TIMEZONE`	`UTC`

You can tailor these to your needs.

The following Docker secrets are used:

Secret	Location
`mp_database_password.txt`	`configs-and-secrets/midpoint/application/database_password.txt`
`mp_keystore_password.txt`	`configs-and-secrets/midpoint/application/keystore_password.txt`

The following configuration files are used:

Target file	Source location	Description

`/opt/midpoint/var/`	midpoint_server/container_files/mp-home/	When `postgresql_midpoint_server_1` is created, the files from this directory are copied to the Midpoint home directory in the container.

You can modify or replace these files as needed.