



Implementación y gestión de bases de datos

Qué es el changelog

Qué es el changelog

Es un fichero que contiene todos los cambios que se requieren aplicar en la base de datos. Estos ficheros pueden ser XML, YAML, JSON o SQL. A continuación, algunos ejemplos:

Figura 1

Fichero XML

```
<databaseChangeLog
    xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ext="http://www.liquibase.org/xml/ns/dbchangelog-ext"
    xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog http://www.liquibase.org.
    http://www.liquibase.org/xml/ns/dbchangelog-ext http://www.liquibase.org/xml/ns/dbchangel
        <changeSet author="liquibase-docs" id="createTable-example">
            <createTable catalogName="cat"
                remarks="A String"
                schemaName="public"
                tableName="person"
                tablespace="A String">
                <column name="address" type="varchar(255)"/>
            </createTable>
        </changeSet>
    </databaseChangeLog>
```

Nota: adaptado de Arango. (2017).

<https://sdos.es/blog/gestionando-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Figura 2

Fichero YAML

```
databaseChangeLog:
    changeSet:
        id: createTable-example
        author: liquibase-docs
        changes:
            - createTable:
                catalogName: cat
                columns:
                    - column:
                        name: address
                        type: varchar(255)
                remarks: A String
                schemaName: public
                tableName: person
                tablespace: A String
```

Nota: adaptado de Arango. (2017).

<https://sdos.es/blog/gestionando-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Figura 3
Fichero JSON

```
{  
    "databaseChangeLog": [  
        "changeSet": {  
            "id": "createTable-example",  
            "author": "liquibase-docs",  
            "changes": [  
                {  
                    "createTable": {  
                        "catalogName": "cat",  
                        "columns": [  
                            {  
                                "column": {  
                                    "name": "address",  
                                    "type": "varchar(255)"  
                                }]  
                            },  
                            "remarks": "A String",  
                            "schemaName": "public",  
                            "tableName": "person",  
                            "tablespace": "A String"  
                        }]  
                }  
            ]  
        }  
    ]  
}
```

Nota: adaptado de Arango. (2017).

<https://sdos.es/blog/gestionando-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

1. Configuración en un proyecto en Maven.

Lo primero que se debe realizar es incluir el plugin en el fichero pom.xml, donde permitirá ejecutar Liquibase en el proyecto:

Figura 4

Configuración en un proyecto en Maven

```
<plugin>
    <groupId>org.liquibase</groupId>
    <artifactId>liquibase-maven-plugin</artifactId>
    <version>3.0.5</version>
    <configuration>
        <changeLogFile>src/main/resources/org/liquibase/db.changelog-master.xml</changeLogFile>
        <driver>oracle.jdbc.driver.OracleDriver</driver>
        <url>jdbc:oracle:thin:@tf-appserv-linux:1521:xe</url>
        <username>liquibaseTest</username>
        <password>pass</password>
    </configuration>
    <executions>
        <execution>
            <phase>process-resources</phase>
            <goals>
                <goal>update</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

Nota: adaptado de Arango. (2017).

<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Liquibase permite la conexión a la base de datos por medio de un datasource definido en el entorno de spring, pero se debe realizar la ejecución de forma automática en el arranque de la aplicación.

2. Estructura del directorio de puentes en Java.

Figura 5

Estructura del directorio en Java



Nota: adaptado de Arango. (2017)

<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Figura 6
Fichero DB.CHANGELOG-MASTER.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<databaseChangeLog
    xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
        http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd">

    <include file="com/example/db/changelog/db.changelog-1.0.xml"/>
    <include file="com/example/db/changelog/db.changelog-1.1.xml"/>
    <include file="com/example/db/changelog/db.changelog-2.0.xml"/>
</databaseChangeLog>
```

Nota: adaptado de Arango. (2017).
<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Figura 7
Fichero DB.CHANGELOG

```
<?xml version="1.0" encoding="UTF-8"?>
<databaseChangeLog
    xmlns="http://www.liquibase.org/xml/ns/dbchangelog/1.9"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog/1.9
        http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-1.9.xsd">
    <changeSet author="authorName" id="changelog-1.0">
        <createTable tableName="TablesAndTables">
            <column name="COLUMN1" type="TEXT">
                <constraints nullable="true" primaryKey="false" unique="false"/>
            </column>
        </createTable>
    </changeSet>
</databaseChangeLog>
```

Nota: adaptado de Arango. (2017).
<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

Una vez declarados cada uno de los ficheros, realizamos la ejecución mediante el correspondiente comando maven:

`mvn liquibase:update`

En caso de que ya se tenga el desarrollo de una base de datos, Liqui base permite la generación automática de todos los *changelogs* necesarios basándose en una base de datos ya existente. Para ello, debes descargar los ejecutables (línea de comandos) y lanzarlos mediante el siguiente comando:

Figura 8
Liquibase update

```
liquibase --driver=oracle.jdbc.OracleDriver \
--classpath=path\to\classes:jdbcdriver.jar \
--changeLogFile=com/example/db.changelog.xml \
--url="jdbc:oracle:thin:@localhost:1521:XE" \
--username=scott \
--password=tiger \
generateChangeLog
```

Nota: adaptado de Arango. (2017).
<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>

REFERENCIAS BIBLIOGRÁFICAS

Arango. (2017). Gestiona el versionado de los scripts de base de datos con Liquibase. SDOS.
<https://sdos.es/blog/gestionar-el-versionado-de-los-scripts-de-base-de-datos-con-liquibase>