

Liquibase

Tomasz Słowik

Spis treści:

1. Co to jest liquibase?
2. Zasada działania
3. Jak powinno się z tego korzystać?
4. Przydatne informacje
5. Plusy i minusy

Co to jest liquibase?

Przeważająca większość aplikacji webowych wykorzystuje w większym lub mniejszym stopniu gdzieś „pod spodem” bazy danych. Przy rozwoju takich aplikacji, często spotykanym problemem jest zarządzanie zmianami w strukturze bazy danych (ciągle dochodzą nowe kolumny, zmieniane są typu kolumn itd.). Bardzo przydatne może okazać się wtedy narzędzie Liquibase. W bardzo prosty sposób można za jego pomocą zapanować nad wszelkimi zmianami, które zostaną dokonane po wdrożeniu już u klienta aplikacji. Liquibase w swoisty sposób zarządza wersjami struktury bazy danych. Zmiany są dokonywane poprzez Changesety które są wykonywane przez skrypt liquibase.

Wspierane języki:



Wspierane bazy danych:

Database	Type Name	Notes
MySQL	mysql	No Issues
PostgreSQL	postgresql	8.2+ is required to use the "drop all database objects" functionality.
Oracle	oracle	11g driver is required when using the diff tool on databases running with AL32UTF8 or AL16UTF16
Sql Server	mssql	No Issues
Sybase_Enterprise	sybase	ASE 12.0+ required. "select into" database option needs to be set. Best driver is JTDS. Sybase does not support transactions for DDL so rollbacks will not work on failures. Foreign keys can not be dropped which can break the rollback or dropAll functionality.
Sybase_Anywhere	asany	Since 1.9
DB2	db2	No Issues. Will auto-call REORG when necessary.
Apache_Derby	derby	No Issues
HSQL	hsqldb	No Issues
H2	h2	No Issues
Informix	informix	No Issues
Firebird	firebird	No Issues
SQLite	sqlite	No Issues

Zasada działania

Krok 1: Stwórz plik ChangeLog

```
<?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">

</databaseChangeLog>
```

Krok 2: Dodaj Changeset

```
<?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">

  <changeSet id="1" author="bob">
    <createTable tableName="department">
      <column name="id" type="int">
        <constraints primaryKey="true" nullable="false"/>
      </column>
      <column name="name" type="varchar(50)">
        <constraints nullable="false"/>
      </column>
      <column name="active" type="boolean" defaultValueBoolean="true"/>
    </createTable>
  </changeSet>

</databaseChangeLog>
```

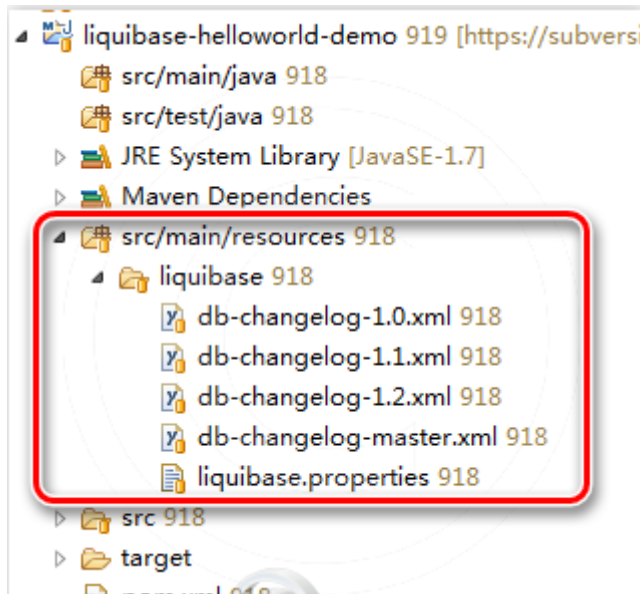
Krok 3: Uruchom skrypt

```
liquibase --driver=com.mysql.jdbc.Driver \  
  --classpath=/path/to/classes \  
  --changeLogFile=com/example/db.changelog.xml \  
  --url="jdbc:mysql://localhost/example" \  
  --username=user \  
  --password=asdf \  
migrate
```

Jak się powinno z tego korzystać?

Pliki changelog trzymać wewnątrz plików źródłowych projektu w pobliżu klas dostępu do bazy danych.

Przykładowa struktura folderów i plików

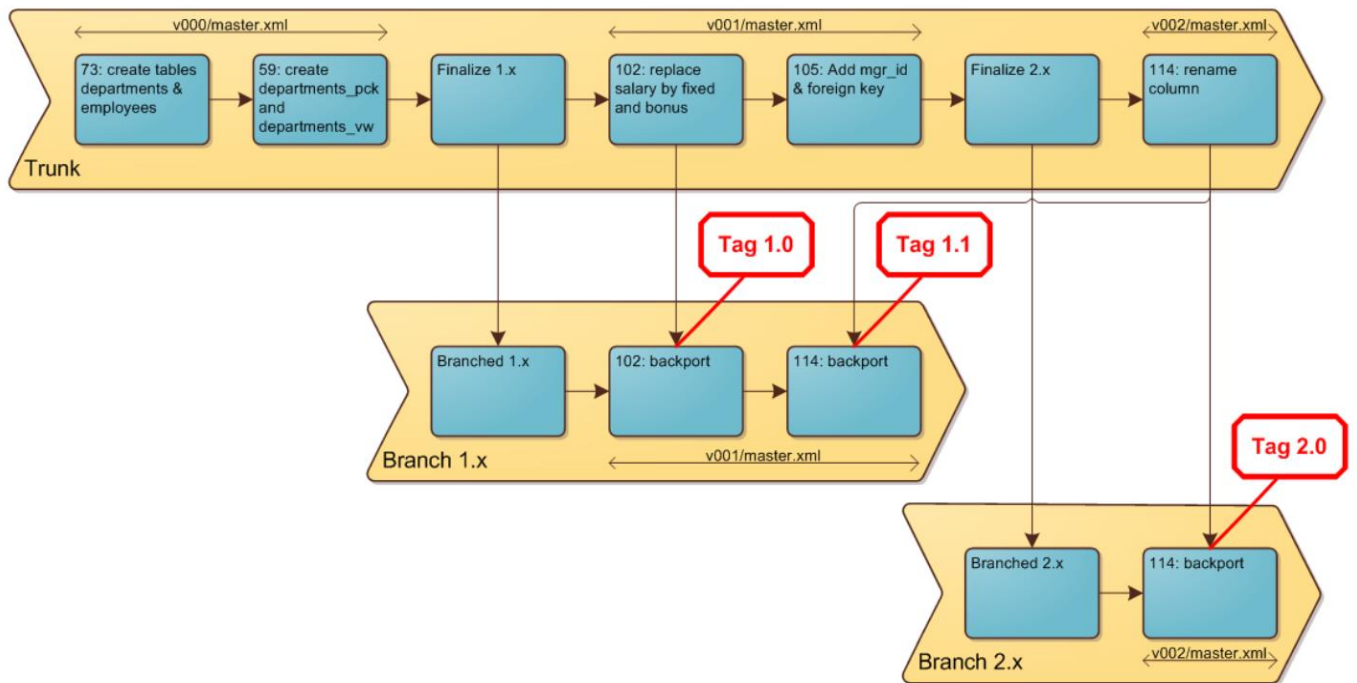


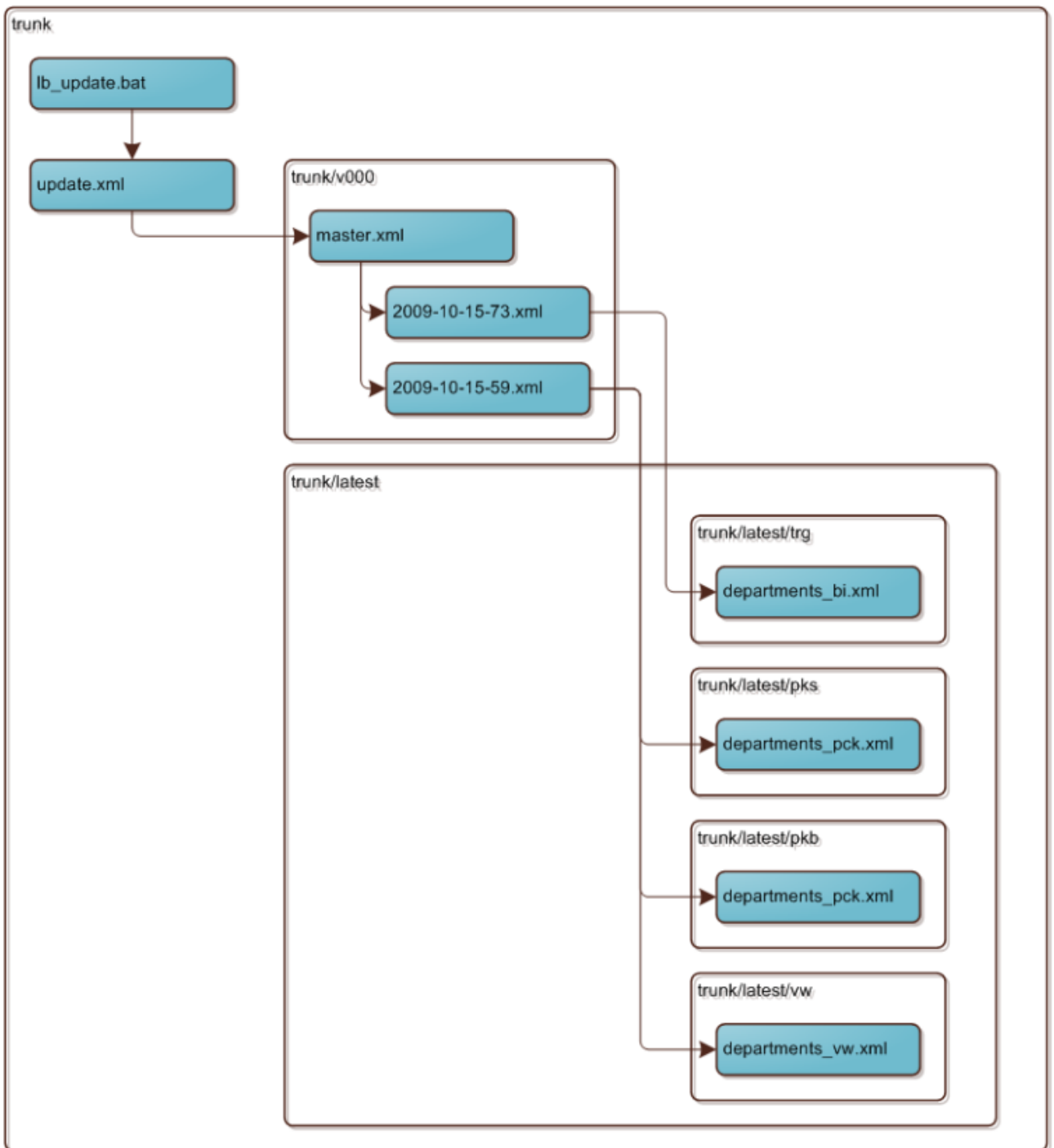
Zawartość pliku changelog-master.xml:

```
<?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>
```

Wersjonowanie zmian na bazie danych





Przydatne informacje

Skrypt liquibase tworzy następujące dwie tabele w bazie na której został uruchomiony:

DATABASECHANGELOG

Column	Standard Data Type	Description
ID	VARCHAR(255)	Value from the changeSet "id" attribute
AUTHOR	VARCHAR(255)	Value from the changeSet "author" attribute
FILENAME	VARCHAR(255)	Path to the changelog. This may be an absolute path or a relative path depending on how the changelog was passed to Liquibase. For best results, it should be a relative path
DATEEXECUTED	DATETIME	Date/time of when the changeSet was executed. Used with ORDEREXECUTED to determine rollback order
ORDEREXECUTED	INT	Order that the changeSets were executed. Used in addition to DATEEXECUTED to ensure order is correct even when the databases datetime supports poor resolution. NOTE: The values are only guaranteed to be increasing within an individual update run. There are times where they will restart at zero.
EXECTYPE	VARCHAR(10)	Description of how the changeSet was executed. Possible values include "EXECUTED", "FAILED", "SKIPPED", "RERAN", and "MARK_RAN",
MD5SUM	VARCHAR(35)	Checksum of the changeSet when it was executed. Used on each run to ensure there have been no unexpected changes to changSet in the changelog file
DESCRIPTION	VARCHAR(255)	Short auto-generated human readable description of changeSet
COMMENTS	VARCHAR(255)	Value from the changeSet "comment" attribute
TAG	VARCHAR(255)	Tracks which changeSets correspond to tag operations.
LIQUIBASE	VARCHAR(20)	Version of Liquibase used to execute the changeSet


DATABASECHANGELOGLOCK

Column	Standard Data Type	Description
ID	INT	Id of the lock. Currently there is only one lock, but is available for future use
LOCKED	INT	Set to "1" if the Liquibase is running against this database. Otherwise set to "0"
LOCKGRANTED	DATETIME	Date and time that the lock was granted
LOCKEDBY	VARCHAR(255)	Human-readable description of who the lock was granted to.

Liquibase umożliwia wycofywanie wprowadzonych zmian

Command	Description
rollback <tag>	Rolls back the database to the state it was in when the tag was applied.
rollbackToDate <date/time>	Rolls back the database to the state it was in at the given date/time.
rollbackCount <value>	Rolls back the last <value> change sets.
rollbackSQL <tag>	Writes SQL to roll back the database to the state it was in when the tag was applied to STDOUT.
rollbackToDateSQL <date/time>	Writes SQL to roll back the database to the state it was in at the given date/time version to STDOUT.
rollbackCountSQL <value>	Writes SQL to roll back the last <value> change sets to STDOUT.
futureRollbackSQL	Writes SQL to roll back the database to the current state after the changes in the changeslog have been applied.
updateTestingRollback	Updates the database, then rolls back changes before updating again.
generateChangeLog	generateChangeLog of the database to standard out. v1.8 requires the dataDir parameter currently.

Plusy i minusy:

	
<ul style="list-style-type: none">• Automatyczne tworzenie dokumentacji• Łatwe wersjonowanie• Czytelna struktura zmian na bazie danych• Możliwość wycofywania wprowadzonych zmian	<ul style="list-style-type: none">• Wymaga poświęcenia więcej czasu• Nie jest przydatny przy małych projektach