# Dependências

# Atualizador

## Liquibase

Toda estrutura e carga da base de dados é mantida e gerenciada pelo framework chamado liquibase (<http://www.liquibase.org/>).

O liquibase é uma biblioteca opensource para rastrear e gerenciar alterações em uma base de dados. Foi criado em 2006 e é utilizado especialmente em ambiente de desenvolvimento ágil.

Todas as alterações são armazenadas em arquivos XML e identificadas por uma combinação das tags “id” e “autor” bem como o nome do arquivo. Suas principais funcionalidades são:

* Extensibilidade para criar alterações personalizadas;
* Atualização de banco de dados para a versão atual;
* Rollback de alterações do banco de dados para determinada data / hora;
* Banco de dados definidos por tag;
* SQL para atualizações de banco de dados e Reversões podem ser guardadas para revisão manual;
* Stand-alone e IDE .NET plug-in;
* Relatório de diferenças no banco de dados;
* Log de mudanças na geração de banco de dados;
* Suporte para vários banco de dados: MySQL, PostgreSQL, Oracle, MS-SQL, DB2, Informix, Firebird, SAPDB,SQLite entre outros;

No projeto existem duas tabelas que gerenciam toda vida do banco de dados, são elas:

* Databasechangelog;
* Databasechangeloglock;

## Suporte multi-banco

O liquibase gerencia dois bancos de dados no projeto, o banco: admin e o livePsicologo, cada banco tem o seu arquivo próprio com os dados de conexão, a configuração do banco admin está no arquivo liquibase-adm.properties e a configuração do banco livePsicologo está no arquivo liquibase- livePsicologo.properties, ambos dentro da pasta atualizador, dentro dos arquivos de configuração existem três linhas de dados que devem estar parametrizados conforme a configuração de seu banco de dados:

Na linha abaixo deve ser configurado o nome do Host a porta e a instancia respectivamente.

url=jdbc:oracle:thin:@localhost:1521:xe

Na linha abaixo deve ser configurado o nome do usuário.

username=admin

Na linha abaixo deve ser configurado a senha.

password=Oracle

Tendo estes arquivos parametrizados os dois bancos serão atualizados corretamente.

## Executando liquibase

Dentro da pasta do atualizador no projeto, existem dois arquivos Batch, que servem para facilitar a atualização múltipla dos bancos, o liquibase-update: quando esse arquivo é executado os bancos são atualizados para a versão mais recente e o liquibase-rollback-0.0.0: que serve para voltar o banco para a versão inicial, os mesmos arquivos existem no formato Shell para serem executados no Linux. Esses arquivos foram criados apenas para facilitar, já que o comando para atualizador o banco também pode ser digitado manualmente estando com o Prompt de comandos dentro da pasta atualizador e digitando a seguinte linha e substituindo as partes em vermelho conforme a baixo:

java –jar liquibase.jar --defaultsFile=./liquibase-adm.properties update

liquibase-adm.properties é o arquivo de configuração utilizado para fazer a conexão

update define a ação que será tomada, poderia ser por exemplo rollback 0.0.0.

## Estrutura de versionamento

Os arquivos de atualização e versionamento estão dentro de uma estrutura de pastas para garantir a organização e o bom entendimento das versões: dentro do projeto atualizador, no caminho “\src\main\resources\” (caminho que está definido dentro dos arquivos de configuração de cada banco de dados: liquibase-adm, liquibase- livePsicologo) existe a primeira subdivisão dos arquivos de atualização, com duas pastas:

adm-changes – Contem a estrutura de arquivos e pastas para gerenciamento do banco: Admin.

livePsicologo -changes – Contem a estrutura de arquivos e pastas para gerenciar o banco: livePsicologo.

A estrutura é composta por três níveis de versões: Conforme a imagem:

|  |  |  |
| --- | --- | --- |
| Nivel 1 | Nivel 2 | Nivel 3 |
| 1.x.x |
| Nivel 1 | 1.1.x |
| Nivel 1 | Nivel 2 | 1.1.1 |
| Nivel 1 | Nivel 2 | 1.1.2 |
| Nivel 1 | 1.2.x |
| Nivel 1 | Nivel 2 | 1.2.1 |
| 2.x.x |
| Nivel 1 | 2.1.x |

Dentro de cada uma dessas pastas existe um arquivo changelog.xml onde é mapeado o caminho para todas as pastas que definem versões nesse mesmo diretório. Nesse diretório é criada uma nova pasta cada vez que sai uma nova versão de nível 1.

Dentro de cada pasta nesse nível, existe mais um arquivo changelog.xml onde é mapeado o caminho para todas as pastas que definem versões nesse mesmo diretório. Nesses diretórios é criada uma nova pasta cada vez que sai uma nova versão de nível 2.

Dentro de cada pasta no nível 2 existe mais um arquivo changelog.xml