Приложение выполняет следующие функции:

1. Выгрузку данных из БД в xml-файл.
2. Синхронизацию таблицы БД по заданному xml-файлу.

В приложении используется файл настроек, система логирования, возможность указания наименования xml-файлов, доступ к СУБД посредством JDBC, запросы в виде SQL-выражений, в алгоритме синхронизации использоавн HashMap, чтение xml при помощи DOM, исходный код документирован в сответствии с JavaDoc.

Для работы с приложением необходимо сделать следующее:

1. Импортировать проект в среду разработки.
2. Перед запуском приложения необходимо отредактировать файл настроек расположенный в проекте **src/main/resources/config.properties**, в соотвествии с используемой СУБД.
3. Для настройки логирования необходимо отредактировать файл, расположенный по адресу **src/main/resources/log4j.properties**, а именно, указать путь для сохранения логов. Например: log4j.appender.file.File = C:\\TMP\\myproject.log
4. Желательно (но не обязательно) создать временный каталог, в который будут сохранены xml-файлы.

Далее необходимо создать и заполнить тестовую таблицу. Файл с бэкапом таблицы прилагается (bd.backup).

Точкой входа в программу является класс Main, расположенный в src/main/java/main

После запуска будет предложено указать существующий каталог для хранения выгруженных из БД таблиц. После чего у пользователя имеется пять команд, а именно:

1. сreate имя создаваемго xml-файла. Данной командой мы создаём «хранилища» для сохранения данных БД. Хранилищем является коллекция, в которой ключ – это имя файла, а значение – это другая коллекция, где ключ – натуральный ключ из БД (столбцы depcode, depjob), а значение –данные столбца description.
2. use имя\_любого созданного xml-файла. Данной командой мы можем выбрать любое созданное хранилище для дальнейших манипуляций с ним.
3. add . Данной командой мы выгружаем информацию из БД в созданное и выбранное хранилище.
4. loading . Данной командой мы синхронизируем таблицу БД в соответствии с выбранным хранилищем.
5. shutdown . Команда на завершение работы приложения.

Таким образом, для проверки работоспособности, после запуска в консоли IDE можно выполнить следующий ввод:

**C:\Java**

**create XXX**

**use XXX**

**add**

После этой команды можно вручную проверить созданный xml и убедиться, что выгрузка прошла успешно. Далее для проверки работоспособности синхронизации вносим изменения в нашу таблицу. (Например удаляем одну из строк и вносим изменения в одно из полей description)

**loading**

Ииии.. вуаля, всё работает, таблица БД синхронизирована в соответствии с заданным xml-файлом, удалённые строки вернулись, поля description приобрели свои первоначальные значения, идентичные строки остались нетронутыми. Для эксперимента можно создавать любое количество хранилищ и выбирать их в режиме реального времени для выгрузки/синхронизации.