Report Cloud Integration

Pierre-Louis Sergent, Meo Bienfait

- Report Cloud Integration
    - Introduction
    - Architecture database
    - Application structure
        - Data input
        - Classes
        - Integration patterns
        - Main class
    - Conclusion

Introduction

In this report, we will present a simple spring boot app that uses integration patterns to get data from json files and send them to a database.

Repo: https://github.com/PLsergent/gettingStartedIntegration

Architecture database

The script to create the database is in the file `script.sql`:

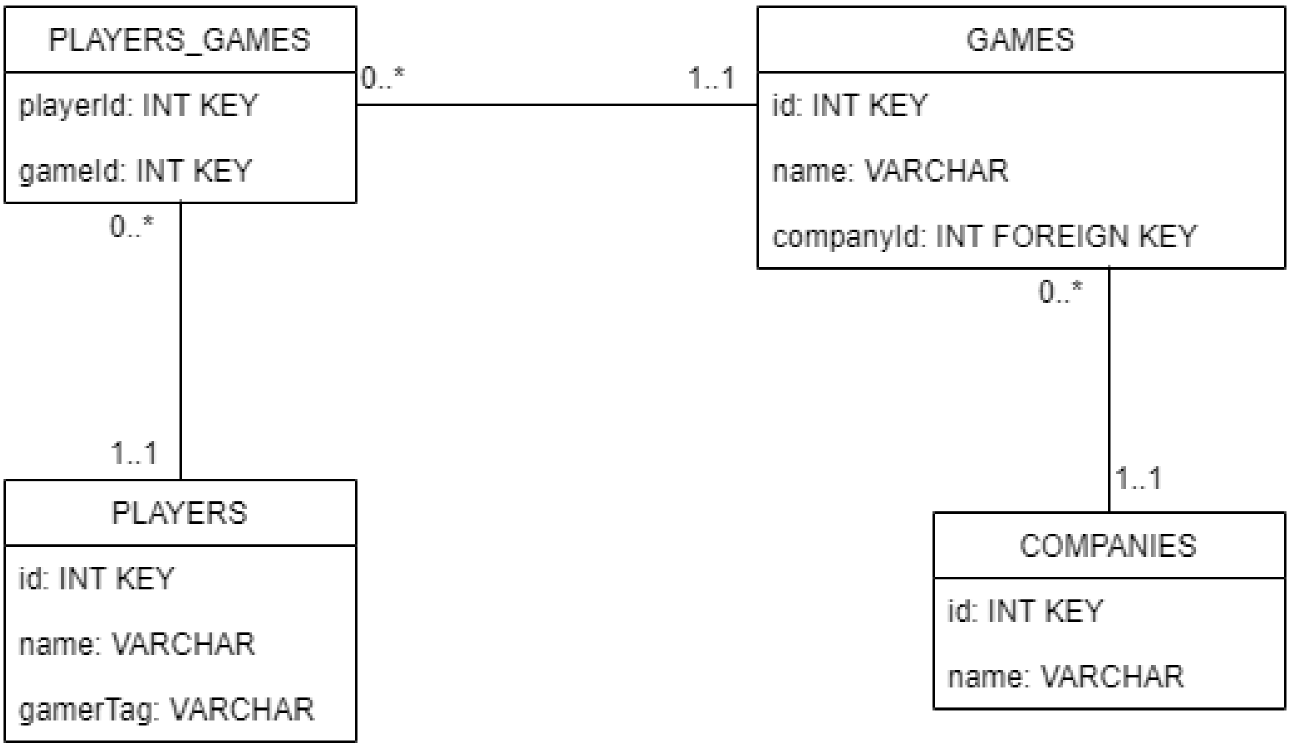
```
CREATE TABLE IF NOT EXISTS GAMES (
    id INT PRIMARY KEY,
    name VARCHAR(56),
    companyId INT
);

CREATE TABLE IF NOT EXISTS PLAYERS (
    id INT PRIMARY KEY,
    name VARCHAR(56),
    gamerTag VARCHAR(100)
);

CREATE TABLE IF NOT EXISTS COMPANIES (
    id INT PRIMARY KEY,
    name VARCHAR(56)
);

CREATE TABLE IF NOT EXISTS PLAYERS_GAMES (
    playerId INT PRIMARY KEY,
    gameId INT PRIMARY KEY
);
```

We can see that we have 4 tables:

- `GAMES` contains the games
- `PLAYERS` contains the players
- `COMPANIES` contains the companies
- `PLAYERS_GAMES` contains the relation between players and games

Essentially, we have a many-to-many relation between `PLAYERS` and `GAMES` and a many-to-one relation between `GAMES` and `COMPANIES`. Meaning that a game can have multiple players and players can play multiple games. A game can only have one company while a company can have multiple games.

Here is a schema of the database:



Application structure

Data input

Basically at the root of the project, we have the folder `dataIn`. This folder contains the json files that we will use to get the data. Each subfolder of `dataIn` is a different source of data. For example, the folder `dataIn/games`: games, `dataIn/companies`: companies, etc.

Then each files in the subfolders are the json files that we will use to get the data. For example, the file `dataIn/games/lol.json` contains the data of the game with the id `0`. The file `dataIn/companies/riot.json` contains the data of the company with the id `0`, etc. The name of the files are not important, only the content is.

Classes

The classes are in the package `java/test`. We have 3 classes:

- `Game`: represents a game
- `Player`: represents a player
- `Company`: represents a company

Game:

```
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Builder
@ToString
public class Game {
    private int id;
    public String name;
    public int companyId;
    public List<Integer> playersIds;
}
```

Player:

```
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Builder
@ToString
public class Player {
    private int id;
    public String name;
    public String gamerTag;
    public List<Integer> gamesIds;
}
```

Company:

```
@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Builder
@ToString
public class Company {
    private int id;
    public String name;
    public List<Integer> playersIds;
    public List<Integer> gamesIds;
}
```

Integration patterns

from_json_to_DB.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:int="http://www.springframework.org/schema/integration"
```

```xml
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:int-file="http://www.springframework.org/schema/integration/file"
       xmlns:jdbc="http://www.springframework.org/schema/jdbc"
       xmlns:task="http://www.springframework.org/schema/task"
       xmlns:int-jdbc="http://www.springframework.org/schema/integration/jdbc"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/integration
           http://www.springframework.org/schema/integration/spring-
integration-5.1.xsd
           http://www.springframework.org/schema/integration/file
           http://www.springframework.org/schema/integration/file/spring-
integration-file-5.1.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd
           http://www.springframework.org/schema/jdbc
           http://www.springframework.org/schema/jdbc/spring-jdbc.xsd
           http://www.springframework.org/schema/task
           http://www.springframework.org/schema/task/spring-task.xsd
           http://www.springframework.org/schema/integration/jdbc
           https://www.springframework.org/schema/integration/jdbc/spring-
integration-jdbc.xsd">

    <context:component-scan base-package="test"/>
    <int:poller id="poller" fixed-delay="2000"/>

    <jdbc:embedded-database id="gamers" type="H2">
        <jdbc:script location="classpath:script.sql"/>
    </jdbc:embedded-database>

    <int-file:inbound-channel-adapter directory="dataIn" id="jsonChannel"/>
    <int-file:inbound-channel-adapter directory="dataIn" id="gameChannel"/>
    <int-file:inbound-channel-adapter directory="dataIn" id="companyChannel"/>
    <int-file:inbound-channel-adapter directory="dataIn" id="playerChannel"/>

    <!-- GAMES -->
    <int-file:inbound-channel-adapter
            channel="jsonChannel"
            directory="dataIn/games"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>

    <int:json-to-object-transformer input-channel="jsonChannel" output-
channel="gameChannel" type="test.Game"/>

    <!-- PLAYERS -->
    <int-file:inbound-channel-adapter
            channel="jsonChannel"
            directory="dataIn/players"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>
    <int:json-to-object-transformer input-channel="jsonChannel" output-
channel="playerChannel" type="test.Player"/>

    <!-- COMPANIES -->
    <int-file:inbound-channel-adapter
            channel="jsonChannel"
            directory="dataIn/companies"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>

    <int:json-to-object-transformer input-channel="jsonChannel" output-
channel="companyChannel" type="test.Company"/>

    <!-- WRITE DATA TO GAMES TABLE -->
    <int-jdbc:outbound-channel-adapter
            query="INSERT INTO GAMES (id, name, companyId)
            values (:payload.id, :payload.name, :payload.company)"
            data-source="gamers"
            channel="gameChannel"/>

    <!-- WRITE DATA TO PLAYERS TABLE -->
        <int-jdbc:outbound-channel-adapter
            query="INSERT INTO ACTORS (id, name, gamerTag)
            values (:payload.id, :payload.name, :payload.gamer-tag)"
            data-source="gamers"
            channel="playerChannel"/>

    <!-- WRITE DATA TO COMPANIES TABLE -->
    <int-jdbc:outbound-channel-adapter
            query="INSERT INTO COMPANIES (id, name)
            values (:payload.id, :payload.name)"
            data-source="gamers"
            channel="companyChannel"/>

    <task:executor id="pollerExecutorDB"/>

</beans>
```

For each subfolders in the `dataIn` folder (games, players, companies), we have a channel adapter that will listen for new files. When a new file is detected, it will be sent to the `jsonChannel` channel. Then, a `json-to-object-transformer` will transform the JSON file into a Java object. Finally, the object will be sent to the corresponding channel (gameChannel, playerChannel, companyChannel) and will be written to the database.

What we failed to do:
We couldn't find the proper way to populate the join table PLAYERS_GAMES with the `gamesIds` and `playersIds` lists. We tried to use a `json-to-object-transformer` to transform the JSON file into a Java object, but we couldn't find a way to populate the join table with the lists.

from_json_to_objects

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
```

```
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:int="http://www.springframework.org/schema/integration"
       xmlns:int-file="http://www.springframework.org/schema/integration/file"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/integration
           http://www.springframework.org/schema/integration/spring-
integration-5.1.xsd
           http://www.springframework.org/schema/integration/file
           http://www.springframework.org/schema/integration/file/spring-
integration-file-5.1.xsd">

    <int-file:inbound-channel-adapter directory="dataIn" id="inputChannel"/>
    <int-file:inbound-channel-adapter directory="dataIn" id="outputChannel"/>
    <int:poller default="true" fixed-delay="2000"/>

    <!-- GAMES -->
    <int-file:inbound-channel-adapter
            channel="inputChannel"
            directory="dataIn/games"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>
    <int:json-to-object-transformer input-channel="inputChannel" output-
channel="outputChannel" type="test.Game"/>

    <!-- PLAYERS -->
    <int-file:inbound-channel-adapter
            channel="inputChannel"
            directory="dataIn/players"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>

    <int:json-to-object-transformer input-channel="inputChannel" output-
channel="outputChannel" type="test.Player"/>

    <!-- COMPANIES -->
    <int-file:inbound-channel-adapter
            channel="inputChannel"
            directory="dataIn/companies"
            filename-pattern="*.json">
    </int-file:inbound-channel-adapter>

    <int:json-to-object-transformer input-channel="inputChannel" output-
channel="outputChannel" type="test.Company"/>

    <!-- Next -->
    <int:recipient-list-router id="customRouter" input-channel="mainChannel">
        <int:recipient channel="serviceActivatorChannel"/>
        <int:recipient channel="aggregatorChannel"/>
    </int:recipient-list-router>

    <int:service-activator input-channel="serviceActivatorChannel" output-
```

```
channel="aggregatorChannel" ref="serviceID" method="gameService"/>
    <bean id="serviceID" class="test.GameService"/>

    <int:channel id="aggregatorChannel"/>

    <int:aggregator id="gamersAggregator"
                    input-channel="aggregatorChannel"
                    output-channel="outputChannel"
                    correlation-strategy-expression="payload.name"
                    release-strategy-expression="size()==2">
    </int:aggregator>

    <int:logging-channel-adapter channel="outputChannel" level="INFO"/>
</beans>
```

We tried to use a `recipient-list-router` to send the objects to a `service-activator` and then to an `aggregator`. The `service-activator` will call a `GameService` class. The `aggregator` will then send the result to the `outputChannel`. The `logging-channel-adapter` will then log the output.

Main class

```
@SpringBootApplication
@ImportResource("classpath:from_Json_to_DB.xml")
public class Main {

    public static void main(String[] args) {
        SpringApplication.run(Main.class, args);
        // ApplicationContext ctx = new
ClassPathXmlApplicationContext("from_json_to_objects.xml");
    }
}
```

This way we can run the application and the XML file will be loaded. We can then run the application and the files will be sent to the database (`@ImportResource("classpath:from_Json_to_DB.xml")`).

Conclusion

We successfully managed to create a Spring Boot application that will read JSON files and write them to a database. We also managed to create a Spring Boot application that will read JSON files and transform them into Java objects. Although we couldn't find a way to populate the join table `PLAYERS_GAMES` with the `gamesIds` and `playersIds` lists.