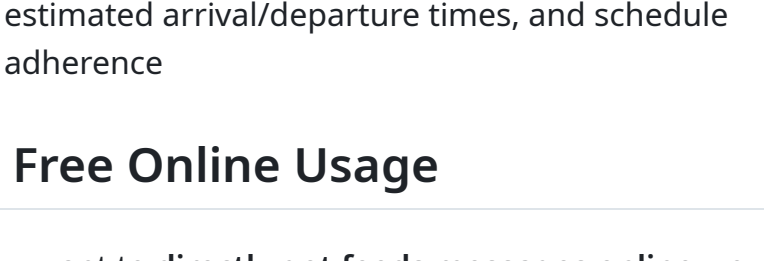


main		Go to file	Code
Jouca	[main]: Implementing cleari...	✓ a96fbbb · last week	
github/workflows	[main] Fix java versio...	last month	
mvnw/wrapper	Merge pull request #...	last month	
src	[main]: Implementin...	last week	
.env.example	[main]: Update env v...	last month	
.gitattributes	Initial commit (reset ...	6 months ago	
.gitignore	[main]: Test coverage	last month	
Dockerfile	Update eclipse-temu...	2 weeks ago	
LICENSE	[main]: Add license	last month	
README.md	[release]: Release ve...	last month	
docker-compose.yml	[main]: Added docke...	last month	
logo.png	[main]: Update logo	last month	
mvnw	Update dependency ...	last month	
mvnw.cmd	Update dependency ...	last month	
pom.xml	Merge pull request #...	2 weeks ago	
renovate.json	Add renovate.json	last month	

README

MIT license



Real-time transit data bridge
converting IDFM (Île-de-France
Mobilités) feeds to
standardized GTFS-Realtime
Protocol Buffers

LAST COMMITNOVEMBERCOMMIT ACTIVITY0/WEEK

GITHUBNO RELEASES OR REPO NOT FOUND

CREATED ATAPRIL

JAVA21SPRING BOOT3.5.6LICENSEMIT

IDFM GTFS-RT Bridge

A Spring Boot application that bridges IDFM (Île-de-France Mobilités) real-time transit data to GTFS-Realtime.

Overview

This service fetches real-time transit information from the IDFM network and converts it into standardized GTFS-Realtime Protocol Buffer files. It provides REST endpoints to access:

- GTFS-RT Alerts:** Service disruptions, delays, and transit alerts
- GTFS-RT Trip Updates:** Real-time vehicle positions, estimated arrival/departure times, and schedule adherence

Free Online Usage

If you want to **directly get feeds messages online**, you can use my **own HTTP links containing the GTFS-RT feeds!**

GTFS-RT Trip Updates :
<http://gtfsidfm.clarifygdps.com/gtfs-rt-trips-idfm>

GTFS-RT Alerts :
<http://gtfsidfm.clarifygdps.com/gtfs-rt-alerts-idfm>

Features

- Real-time Transit Data:** Automatic fetching and processing of IDFM transit data
- Multiple Format Support:** GTFS-Realtime (Protocol Buffers) and SIRI-Lite (JSON)
- Scheduled Updates:** Periodic data synchronization with configurable intervals
- Docker Support:** Easy deployment with Docker and Docker Compose
- Trip Matching:** Intelligent matching of real-time data with scheduled GTFS trips
- SQLite Database:** Local GTFS data storage for fast access
- High Performance:** Optimized for handling large transit networks

Technology Stack

- Java 21:** Modern Java runtime
- Spring Boot 3.5.6:** Application framework with scheduling support
- GTFS Realtime Bindings 0.0.8:** Protocol Buffer handling
- OneBusAway GTFS 10.2.0:** GTFS data processing
- SQLite 3.50.3.0:** Local database storage
- Jackson Databind:** JSON processing
- OpenCSV 5.12.0:** CSV file parsing
- Apache Commons DBCP2 2.13.0:** Database connection pooling
- Maven:** Build and dependency management
- Docker:** Containerization
- JaCoCo 0.8.12:** Code coverage reporting

Prerequisites

- Java 21 or higher
- Maven 3.6+
- Docker & Docker Compose (for containerized deployment)
- Node.js 22+ (for gtfs-import tool)
- IDFM API Key (required for accessing real-time data)

Quick Start

GTFS Database Setup

Before running the application, you need to set up the GTFS static data database:

- Download GTFS data** from IDFM or specify `GTFS_URL` in your `.env` file
- Import GTFS into SQLite** using the `gtfs-import` tool (Node.js required):

```
# Install node-gtfs globally
npm install -g gtfs

# Import GTFS data
gtfs-import --gtfsPath=path/to/gtfs.zip --
```

- Place `gtfs.db` in the project root directory

The application will use this database to match real-time updates with scheduled trips.

Using Pre-built Docker Image from GitHub Container Registry (Recommended)

- Create a `.env` file from the example**

You need to create a `.env` file based on the provided `.env.example` :

- Run the container**

```
docker run -d \
  --name gtfs_app \
  --env-file .env \
  -p 8507:8507 \
  ghcr.io/jouca/idfm_gtfs-rt:latest
```

The application will be available at
<http://localhost:8507>

Using Docker Compose (Build from Source)

- Clone the repository**

```
git clone https://github.com/Jouca/IDFM_GTI
cd IDFM_GTFS-RT
```

- Create environment file**

```
cp .env.example .env
# Edit .env with your configuration
```

- Build and run with Docker Compose**

```
docker compose up -d
```

The application will be available at
<http://localhost:8507>

Manual Installation

- Clone and build**

```
git clone https://github.com/Jouca/IDFM_GTI
cd IDFM_GTFS-RT
mvn clean package
```

- Run the application**

```
java -jar target/idfm_gtfs_rt-1.0.4.jar
```

Configuration

Configuration is managed through `application.properties` and environment variables:

```
# Application name (configurable via SPRING_API
spring.application.name=${SPRING_APPLICATION_N

# Server port (default: 8507, configurable via
server.port=${SERVER_PORT:8507}

# Minutes around now (+/-) to include theoretic
gtfsrt.cancellation.window.minutes=120

# Logging Configuration
logging.level.root=INFO
logging.level.org.jouca.idfm_gtfs_rt=INFO
logging.pattern.console=%d{yyyy-MM-dd HH:mm:ss}
```

Environment Variables

Create a `.env` file with the following variables:

```
# Spring Application Configuration
SPRING_APPLICATION_NAME=idfm_gtfs_rt

# Server Configuration
SERVER_PORT=8507

# API Configuration
# Your API key for accessing IDFM services
API_KEY=your_api_key_here

# (OPTIONAL) GTFS Data Source
GTFS_URL=https://example.com/path/to/gtfs.zip
```

See `.env.example` for a complete template with all available configuration options.

API Endpoints

GET /gtfs-rt-alerts-idfm

Download GTFS-RT alerts feed (Protocol Buffer format)

Response: Binary `.pb` file containing service alerts

Example:

```
curl -O http://localhost:8507/gtfs-rt-alerts-idfm.pb
```

GET /gtfs-rt-trips-idfm

Download GTFS-RT trip updates feed (Protocol Buffer format)

Response: Binary `.pb` file containing trip updates

Example:

```
curl -O http://localhost:8507/gtfs-rt-trips-idfm.pb
```

POST /getEntities

Retrieve specific trip entities by their IDs

Parameters:

- `tripIds` (required): Comma-separated list of trip IDs

Example:

```
curl -X POST "http://localhost:8507/getEntities?tripIds=1,2,3"
```

Response: JSON object mapping trip IDs to their entity data

Project Structure

```
idfm_gtfs-rt/
├── src/main/java/org/jouca/idfm_gtfs_rt/
│   ├── GTFSRTApplication.java      # Main ap
│   ├── controller/
│   │   ├── GTFSRTController.java   # REST AP
│   │   ├── fetchers/
│   │   │   ├── AlertFetcher.java    # Fetches
│   │   │   ├── GTFSFetcher.java     # Fetches
│   │   │   ├── SiriLiteFetcher.java # Fetches
│   │   ├── finders/
│   │   │   ├── TripFinder.java      # Matches
│   │   ├── generator/
│   │   │   ├── AlertGenerator.java  # Generati
│   │   │   ├── TripUpdateGenerator.java # Generati
│   │   ├── records/
│   │   │   ├── EstimatedCall.java   # Data mo
│   │   └── services/
│   │       ├── ScheduledTasks.java   # Backgrou
├── docker-compose.yml              # Docker (
├── Dockerfile                      # Docker :
└── pom.xml                         # Maven de
```

How It Works

- Data Fetching:** The application periodically fetches GTFS static data and real-time updates from IDFM
- Trip Matching:** Real-time data is matched with scheduled trips using the `TripFinder`
- Format Conversion:** Data is converted to GTFS-RT Protocol Buffers
- File Generation:** Updated feeds are written to `.pb` and `.json` files
- API Serving:** REST endpoints serve the latest feed data to clients

Docker Deployment

The application is containerized for easy deployment:

- Memory Limit:** 12GB
- CPU Limit:** 8 cores
- Port:** 8507
- Network:** Isolated bridge network (`gtfs_net`)
- Restart Policy:** unless-stopped (automatically restarts on failure)

Docker Commands

```
# Build and start
docker compose up -d

# Build without cache
docker compose build --no-cache

# View logs
docker compose logs -f

# Stop services
docker compose down

# Stop and remove volumes
docker compose down -v

# Restart services
docker compose restart
```

Resource Limits

The Docker container is configured with the following resource limits:

- Memory:** 12GB
- CPUs:** 8 cores
- Restart Policy:** unless-stopped

Testing

Run tests with Maven:

```
mvn test
```

Code Coverage

The project includes JaCoCo for code coverage analysis. After running tests, view the coverage report:

```
mvn test
open target/site/jacoco/index.html
```

Coverage reports are available in:

- HTML format: `target/site/jacoco/index.html`
- XML format: `target/site/jacoco/jacoco.xml`
- CSV format: `target/site/jacoco/jacoco.csv`

Development

Building from Source

```
# Clean and build
mvn clean package

# Skip tests
mvn clean package -DskipTests

# Run locally
mvn spring-boot:run
```

Adding Dependencies

Edit `pom.xml` and run:

```
mvn dependency:resolve
```

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

- Fork the repository
- Create your feature branch (`git checkout -b feature/AmazingFeature`)
- Commit your changes (`git commit -m 'Add some AmazingFeature'`)
- Push to the branch (`git push origin feature/AmazingFeature`)
- Open a Pull Request

License

This project is licensed under the MIT License - see the LICENSE file for details.

Author

@jouca

Acknowledgments

- [IDFM \(Île-de-France Mobilités\)](#) for providing transit data
- [MobilityData](#) for GTFS-Realtime specifications
- [OneBusAway](#) for GTFS processing libraries
- [NodeGTFS](#) for providing a library for decompressing a GTFS file into a SQLite database

Resources

- [GTFS-Realtime Specification](#)
- [SIRI-Lite Documentation](#)
- [Spring Boot Documentation](#)
- [API for GTFS-RT Trips: Next Departures \(Île-de-France Mobilités platform\) – Global Query](#)
- [API for GTFS-RT Alerts: Traffic Info Messages – Global Query](#)

Troubleshooting

Common Issues

Application won't start

About

Real-time transit data bridge
converting IDFM (Île-de-France
Mobilités) feeds to
standardized GTFS-Realtime
Protocol Buffers

gtfs transport gtfs-realtime

gtfs-rt siri-lite idfm

Readme

MIT license

Activity

9 stars

3 watching

0 forks

Report repository

Releases

5 tags

Packages 1

idfm_gtfs-rt

Contributors 2

Jouca

renovate[bot]

Languages

Java 99.7% Dockerfile 0.3%

- Verify java 21 is installed: `java -version`
- Check if port 8507 is available: `lsof -i :8507`
- Ensure `.env` file is properly configured
- Verify API_KEY is set in `.env` file

Docker container exits immediately

- Check container logs: `docker compose logs -f`
- Verify `.env` file exists and is properly formatted
- Ensure sufficient system resources (12GB RAM, 8 CPU cores)

No data in feeds

- Verify API_KEY is valid
- Check logs for API connection errors
- Ensure GTFS database (`gtfs.db`) is present and not corrupted
- Verify GTFS_URL (if set) points to a valid GTFS ZIP file

High memory usage

- This is expected for large transit networks
- Adjust `mem_limit` in `docker-compose.yml` if needed
- Monitor with: `docker stats gtfs_app`

Logging

To enable debug logging, update `application.properties` :

```
logging.level.org.jouca.idfm_gtfs_rt=DEBUG
```

Or set in `.env` :

```
LOGGING_LEVEL_ORG_JOUCA_IDFM_GTFS_RT=DEBUG
```

 Issues

If you encounter any issues, please file a bug report on the [GitHub Issues](#) page.

Note: This is an unofficial project and is not affiliated with IDFM or Île-de-France Mobilités.