

# streaming-file-server (4.3.40)

Maksim Kostromin

Version 4.3.40, 2022-03-31 19:54:42 UTC

# Table of Contents

1. Introduction .....	2
2. Installation .....	3
2.1. download files .....	3
3. Run .....	4
3.1. postgres database .....	4
3.2. in-memory h2 database .....	4
3.3. take advantages of spring-boot executable jar .....	4
4. Usage scripts .....	5
4.1. simplicity bootstrap with automation shell-script .....	5
4.1.1. unix (bash) .....	5
4.1.2. windows (batch cmd) .....	5
5. Create new release .....	7

Documentation in PFD format is located:

# Chapter 1. Introduction

[Streaming file server](#) — java based project on top of spring-boot. This is a simple file-server which is allowed upload and download files with no memory limitation. It uses file multipart protocol

# Chapter 2. Installation

## 2.1. download files

*if you have docker installed and wanna use postgres, then [download docker-compose.yml file](#)*

```
wget https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/docker-compose.yml
```

*[file items service](#)*

```
wget https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/file-items-service-4.3.40.jar
```

*[file server](#)*

```
wget https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/file-server-4.3.40.jar
```

# Chapter 3. Run

## 3.1. postgres database

*install using postgres in docker*

```
# docker compose file for postgres database
docker-compose up -d

# file-items data service
java -jar file-items-service-4.3.40.jar --spring.profiles.active=db-pg

# file server
java -jar file-server-4.3.40.jar --app.upload.path=./path/to/file-storage

# cleanup
docker-compose down -v
```

## 3.2. in-memory h2 database

*if you do not have docker — feel free to use h2 in memory database for [file items service](#):*

```
java file-items-service-4.3.40.jar
# or
java file-items-service.jar --spring.profiles.active=db-h2
```

*and then run [file server](#):*

```
java file-items-service-4.3.40.jar --spring.profiles.active=db-h2
```

## 3.3. take advantages of spring-boot executable jar

*if you are using bash — run even simply*

```
wget https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/file-
items-service-4.3.40.jar
bash file-items-service-4.3.40.jar

wget https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/file-
server-4.3.40.jar
bash file-server-4.3.40.jar --app.upload.path=./path/to/file-storage
```

# Chapter 4. Usage scripts

## 4.1. simplicity bootstrap with automation shell-script

### 4.1.1. unix (bash)

*postgres in docker*

```
# get
wget https://github.com/daggerok/streaming-file-
server/releases/download/4.3.40/application.bash

# start
bash application.bash start ./path/to/file-storage

# stop
bash application.bash stop

# cleanup
bash application.bash clean ./path/to/file-storage
```

download: [application.bash](#)

*h2 in-memory database*

```
# fetch
wget https://github.com/daggerok/streaming-file-
server/releases/download/4.3.40/application-h2.bash

# start
bash application-h2.bash start ./path/to/file-storage

# stop
bash application-h2.bash stop

# cleanup
bash application-h2.bash clean ./path/to/file-storage
```

download: [application-h2.bash](#)

*note: binaries `wget`, `docker-compose` and of course `jre` (binaries: `java` and `jps`) are required*

### 4.1.2. windows (batch cmd)

### *postgres in docker*

```
@rem start
application.cmd start path\to\file-storage

@rem stop
application.cmd stop

@rem cleanup
application.cmd clean path\to\file-storage
```

download: [application.cmd](#)

### *h2 in-memory database*

```
@rem start
application-h2.cmd start path\to\file-storage

@rem stop
application-h2.cmd stop

@rem cleanup
application-h2.cmd clean path\to\file-storage
```

download: [application-h2.cmd](#)

*note: binaries `wget`, `docker-compose` and of course `jre` (binaries: `java` and `jps`) are required*



# Chapter 5. Create new release

to create new pre-release, simply do commit and push into master or main branch

to create new release do next

1. bump version in (better use IDE find and replace functionality):
  - a. build.gradle
  - b. README.md
  - c. scripts/application.cmd
  - d. scripts/application.bash
  - e. scripts/application-h2.cmd
  - f. scripts/application-h2.bash
2. commit amend and push
3. check CI if builds was successfully passed after triggering [github-pages](#) github action job
4. update created release page on github according last changes

download all files from github releases page: - [github.com/daggerok/streaming-file-server/releases/download/4.3.40/\\*](https://github.com/daggerok/streaming-file-server/releases/download/4.3.40/*)

---

links:

- [fix](#) [issue:](#) [SQLFeatureNotSupportedException: Method org.postgresql.jdbc.PgConnection.createClob\(\) is not yet implemented.](#)
  - [spring-mvc](#)
  - [spring](#)
  - [mustache template engine](#)
  - [apache fileUpload](#)
  - [lombok](#)
  - [vavr](#)
  - [bootstrap](#)
  - [bootstrap file-input](#)
  - [h2](#)
  - [postgres](#)
  - [docker](#)
  - [gradle](#)
-

Enjoy :)