Documentation

Beanframework

Thank you so much for using beanframework project. 100% open source project under MIT license.

Version: 3.0.0

Created: 2 May, 2021

Github: Beanframework

Update: 2 May, 2021

If you have any questions that are beyond the scope of this help file, Please feel free to post via Github Discussion Page .

License

Installation

Create a new database:

1. By default, this project supported unicode character such as Chinese language, therefore the database must use CHARACTER SET utf8 COLLATE utf8_unicode_ci

```
CREATE SCHEMA `beanframework` DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci
```

2. Configure project by duplicating all .template files/folder in current location, without .template suffix:
   - `beanframework/bin/pom.xml.template` - Contains maven modules in project.
   - `beanframework/bin/server.bat.template` - Contains execute command to run application in windows environment.
   - `beanframework/bin/server.sh.template` - Contains execute command to run application in linux environment.
   - `beanframework/bin/install/app.xml.template` - Contains exe progrom to run application as service in windows environment.
   - `beanframework/bin/platform/pom.xml.template` - Contains platform maven configuration, such as configure jar packaging to include/exclude resources

- `beanframework/config/pom.xml.template` - Contains project modules/dependencies configuration.
- `beanframework/config/src/main/resources/*.template` - Contains application properties, ehcache and logback xml files for different environments.
- `beanframework/config/src/main/resources/import/dev.template` - Contains init data, init sql data and sample data

3. Optional: To startup application first time in command: Navigate to `/beanframework/bin` and run `mvnw clean install` command.
   - To download dependencies, compile project and package project into jar application.
4. Optional: To run application, execute command `/beanframework/bin/install/server.bat` or `/beanframework/bin/install/server.sh`
5. Optional: To run application as windows service, run app.exe at `/beanframework/bin/install/app.exe`
6. Access application endpoints:
   - Console: http://localhost:8080/console
   - Backoffice: http://localhost:8080/backoffice
   - Documentation: http://localhost:8080/documentation
7. Import sample data:
   - Access Console: http://localhost:8080/console
   - Login with default admin account: username: `admin` , password: `admin`
   - Goto Platform->Update menu, check all and update.
8. You are good to go for run your project now!

Project Structure

1. Below is the beanframework folder structure:
   - `bin` - Source files
     - `custom` - Create a custom modules and put in this folder, best practice for not mixing with original project structure and codes, and easily to upgrade software in future.
     - `install` - Install application as service in windows environments.
     - `modules` - Project default modules.
     - `platform` - Platform that startup this application with default properties. With IDE, it can run on server at `platform/src/main/java/com/beanframework/platform/PlatformApplication.java`
   - `config` - Mainly use for application properties and configurations for different environments. You can create more files prefix with different environments such as `application-[environment].properties` , and assign `spring.profiles.active=[environment]` in `application.properties`
     - `application.properties` - Highest priority properties file in Spring Boot.
     - Example of dev environment properties and configurations:
       - `application-dev.properties` - Override default properties for dev environments.
       - `ehcache-dev.xml` - Cache configuration for dev environments.
       - `logback-dev.xml` - Logback configuration for dev environments.
       - `import/dev/sampledata/*.csv` - Contains sample data. - This can be configured in application-dev.properties:
         - `module.imex.import.locations=classpath*:import/dev/sampledata/*.csv`
       - `import/dev/sql/*.sql` - Contains SQL query to be automatically executed every time when datasource connection is established or server startup.
         - This can be configured in application-dev.properties:
         - `platform.import.startup.enabled=true`
         - `platform.import.startup=classpath*:import/dev/sql/*.sql`
   - `data` - Data storage for this application
     - `media` - All the media stored in this directory.
     - `integration` - Spring integration directory to process files. Can be used for import/export files.
   - `log` - Application logging files, with archived rotated logging.
   - `temp` - Application temporary files.