1. Назначение компонента

Компонент предназначен для обработки запросов стандартного вида через определенных обработчиков. На основании переданных данных в JSON, используется определенный обработчик, выполняя запросы на получение данных, создание и редактирование данных.

Основными назначениями компонента является:

- авторизация;

- запросы к базе данных;

- работы с файловой системой (через базу данных или, непосредственно, файловую систему).

Работа с компонентом осуществляется с помощью стандартизированных JSON запросов.

1. Лицензия

Лицензия проекта - Eustrosoft BALES. Её описание можно найти в корне проекта.

1. Какие сторонние компоненты (библиотеки, фреймворки, компиляторы, препроцессоры) используются, для чего.

Компиляторы:

- Java 8 (OpenJDK 8 [1.8.301]).

Библотеки:

- javax.servlet-api 4.0.1;

- jsp-api 2.0 (для тестирования);

- org.postgresql (42.5.1);

- com.google.code.gson 2.10.1 (формирование ответов);

- org.projectlombok - Lombok 1.18.24;

- commons-io 2.11.0;

- org.eustrosoft.DBPool 0.0.1;

- org.eustrosoft.SessionCookie 0.0.1;

Сборщик проекта:

- apache maven 3.6.2

- org.apache.maven.plugins (maven-war-plugin, maven-compiler-plugin, projectlombok plugin);

Используемый сервер:

- apache tomcat 9.x.

1. Как собрать (из командной строки), как установить в эксплуатацию

Для начала необходимо скачать библиотеки, которые не находятся в хранилищах maven.

Библиотеки находятся на сайте:

- <http://ftp.eustrosoft.org/pub/eustrosoft.org/pkg/ConcepTIS/>

Необходимо скачать два JAR файла:

- qDBPool;

- qSessionCookie.

После этого их необходимо установить в папку m2 для пользователя, который будет собрать проект. Для этого описан скрипт, находящийся в корне проекта в папке «scripts». Скрипт называется “installMvn”, и он включает в себя подстановки, которые необходимо заполнить. Руководствоваться по именованию и версиям необходимо, учитывая данные в файле mvn основного проекта.

Также необходимо не забыть поменять параметры в файле cms.properties и logging.properties в соответствие с предпочтениями папок и файлов, которые хочется использовать, в случае логирования и использования файловую систему на основании дискового пространства, а не базы данных.

Версия, имя и пути должны соответствовать тем, что находятся в mvn файле проекта.

Далее необходимо собрать основной проект. Базовая сборка – в war файл, который необходимо поставить в папку приложений веб-сервера.

1. Структура дерева исходных текстов, где что лежит, сколько строк

Слева – количество строк, справа – относительный путь в проекте.

31 ./request-handler-service/src/main/java/com/eustrosoft/core/AliveServlet.java

61 ./request-handler-service/src/main/java/com/eustrosoft/core/Constants.java

40 ./request-handler-service/src/main/java/com/eustrosoft/core/context/DBPoolContext.java

15 ./request-handler-service/src/main/java/com/eustrosoft/core/context/EustrosoftUser.java

15 ./request-handler-service/src/main/java/com/eustrosoft/core/context/StorageContext.java

46 ./request-handler-service/src/main/java/com/eustrosoft/core/context/User.java

67 ./request-handler-service/src/main/java/com/eustrosoft/core/context/UsersContext.java

144 ./request-handler-service/src/main/java/com/eustrosoft/core/context/UserStorage.java

33 ./request-handler-service/src/main/java/com/eustrosoft/core/filter/CorsAllowFilter.java

92 ./request-handler-service/src/main/java/com/eustrosoft/core/filter/RequestLoggingFilter.java

26 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/BasicAuthHandler.java

272 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/CMSHandler.java

128 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/CMSRequestBlock.java

79 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/CMSResponseBlock.java

22 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/DownloadFileDetails.java

11 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/FileDownloadMap.java

19 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/cms/FileTicket.java

64 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/ExceptionBlock.java

131 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/BytesChunkFileHandler.java

28 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/BytesChunkFileRequestBlock.java

118 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/ChunkFileHandler.java

55 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/ChunkFileRequestBlock.java

53 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/FileHandler.java

150 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/FileRequestBlock.java

51 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/FileResponseBlock.java

133 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/HexFileHandler.java

45 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/file/HexFileRequestBlock.java

14 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/Handler.java

129 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/login/LoginHandler.java

75 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/login/LoginRequestBlock.java

64 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/login/LoginResponseBlock.java

53 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/ping/PingHandler.java

55 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/ping/PingRequestBlock.java

91 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/ping/PingResponseBlock.java

30 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/BasicRequest.java

16 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/MethodConstants.java

22 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/QTisRequestObject.java

15 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/Request.java

20 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/RequestBlock.java

52 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/requests/RequestObject.java

21 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/responses/BasicResponse.java

79 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/responses/QTisResponse.java

19 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/responses/Response.java

30 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/responses/ResponseBlock.java

11 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/responses/ResponseLang.java

76 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/DBConnector.java

78 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/DBWrapper.java

58 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/model/ResultSetAnswer.java

66 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/SQLHandler.java

72 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/SQLRequestBlock.java

128 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/SQLResponseBlock.java

45 ./request-handler-service/src/main/java/com/eustrosoft/core/handlers/sql/utils/ResultSetUtils.java

459 ./request-handler-service/src/main/java/com/eustrosoft/core/HttpRequestDispatcher.java

93 ./request-handler-service/src/main/java/com/eustrosoft/core/providers/DataSourceProvider.java

47 ./request-handler-service/src/main/java/com/eustrosoft/core/providers/SessionProvider.java

36 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/ColorTextUtil.java

70 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/FileDownloadService.java

122 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/FileUtils.java

77 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/Json.java

29 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/LogFormatter.java

19 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/PropertiesConstants.java

612 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/QJson.java

34 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/WebParams.java

48 ./request-handler-service/src/main/java/com/eustrosoft/core/tools/ZipService.java

13 ./request-handler-service/src/main/java/com/eustrosoft/datasource/exception/CMSException.java

65 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/CMSDataSource.java

11 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/Constants.java

23 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/FileDetails.java

31 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/HexFileParams.java

27 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/HexFileResult.java

34 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSDirectory.java

36 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSFile.java

36 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSGeneralObject.java

56 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSLink.java

35 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSObject.java

34 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/model/CMSType.java

22 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/parameters/CMSObjectUpdateParameters.ja

va

40 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/PropsContainer.java

34 ./request-handler-service/src/main/java/com/eustrosoft/datasource/sources/Source.java

44 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/constants/DBConstants.java

30 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/constants/SQLConstants.java

493 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/DBDataSource.java

490 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/DBFunctions.java

253 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/DBStatements.java

81 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/ExecStatus.java

21 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/DBObject.java

68 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/FDir.java

115 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/FFile.java

9 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/JsonFormat.java

9 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/ResultSetConverter.java

5 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/core/model/Updatable.java

161 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/queries/Query.java

38 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/ranges/FileType.java

73 ./request-handler-service/src/main/java/com/eustrosoft/dbdatasource/util/ResultSetUtils.java

30 ./request-handler-service/src/main/java/com/eustrosoft/filedatasource/constants/FileConstants.java

35 ./request-handler-service/src/main/java/com/eustrosoft/filedatasource/constants/Messages.java

417 ./request-handler-service/src/main/java/com/eustrosoft/filedatasource/FileCMSDataSource.java

42 ./request-handler-service/src/main/java/com/eustrosoft/filedatasource/util/FileUtils.java

316 ./request-handler-service/src/main/java/com/eustrosoft/tools/CMSDownloader.java

16 ./request-handler-service/src/main/java/com/eustrosoft/tools/Constants.java

43 ./request-handler-service/src/main/java/com/eustrosoft/tools/DateFormatter.java

67 ./request-handler-service/src/main/java/com/eustrosoft/tools/HexFileConverter.java

Всего – 8247 строк.

1. Архитектура проекта, что за что отвечает

Для начала, в проекте существуют константы и утильные классы, которые работают независимо от каких-либо компонентов. Этими классами являются все классы в папках «util», и файлы с названием «Constants» в имени файла.

В приложении присутствует две входных точки для взаимодействия. Этими точками являются два сервлета. Один из них показывает доступность веб-сервера – AliveServlet. Второй, непосредственно, обработчик запросов – HttpRequestDispatcher (находящийся в пакете com.eustrosoft.core).

Инициализация сервлетов осуществляется с помощью аннотаций.

Архитектура обработки запросов построена следующим образом:

1. Запрос «приходит» в сервлет с соответствующим маппингом и разбирается на составные части.
2. Определяется, какой обработчик будет обрабатывать тот или иной запрос. Потому что запросов внутренних для одного http запроса может быть несколько.
3. Запрос обрабатывается «Handler-ом», который отвечает в стандартном формате (вне зависимости от того прошёл ли запрос с ошибкой или без).

В проекте существует несколько интерфейсов: запрос, ответ, обработчик. Это базовые интерфейсы, которые обязан реализовывать каждый тип запроса, ответа или обработчик.

Иными словами – это контракт, который гарантирует, что обработка запроса пойдет в соответствие с планом. Интерфейсы запроса и ответа реализуют абстрактные классы «BasicRequest» и «BasicResponse» – они облегчают описание объектов запроса для определенных обработчиков.

Такой подход выбран в связи с разными требованиями к каждому определенному обработчику, которым необходимы разные параметры с отличающейся структурой. Таким образом, запросы и ответы – отдельные сущности, которыми можно манипулировать и фильтровать на наличие необходимых параметров, если это будет необходимо.

В данном варианте, обработчик получает интерфейс запроса, который явно преобразуется к типу запроса, который необходим обработчику. В дальнейшем, обработчик работает с запросом в своих интересах.

После обработки запроса обработчик должен вернуть стандартный ответ, который обрабатывается библиотекой GSON для последующей отправки ответа.

1. Протоколы взаимодействия между компонентами

Взаимодействие между Front-End и Back-End происходят с помощью стандартизированного json формата.

Стандарт описан в файле в проекте Documentation.

Перечисление API для работы с сервисом:

Запросы с авторизацией:

- POST /api/dispatch – необходим для обработки запросов в стандартном формате;

- GET /api/download – необходим для скачивания файлов из подсистемы CMS. В качестве параметра принимает id (ID – объекта для скачивания), path (полный путь до файла, который необходимо скачать), contentType (тип контента, который будет указан в ответе).

Запросы без авторизации:

- GET /alive – Получить состояние сервиса. 200 со значением “Alive!” или 500.

Перечисление возможных параметров ответа для каждого из типа запросов.

1. Подсистема PING.

Запрос стандартный – с подсистемой ping и запросом ping.

В качестве ответа приходят данные:

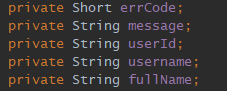


Рисунок - Ответ при запросе подсистемы Ping

1. Подсистема Login. S – login, R – login.

В запросе необходимы два параметра:



Рисунок - Параметры при запросе на Login

Ответ является стандартным блоком.



Рисунок - Ответ при запросе на Login

В качестве ответа может прийти код 0 или 501 при неудовлетворительной попытке логина. Выйти из системы можно с помощью такого же типа запроса, но с R = logout.

1. Подсистема CMS. S – cms. Является основной точкой использования системы.

При использовании CMS существует множество вариантов запросов. Они представлены ниже:

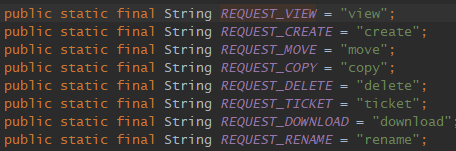


Рисунок - Варианты запросов при использовании CMS системы (R = cms)

При использовании данной подсистемы существуют следующие параметры для запроса в json. В зависимости от типа запроса используются различные параметры.

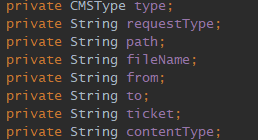


Рисунок - Параметры для запроса в системе CMS

Ответом является либо контент папки, либо стандартизированного формата ответ с ошибкой или кодом ответа 0.

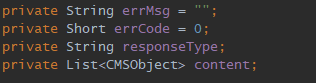


Рисунок - Параметры ответа для запроса в подсистему CMS

1. Направления доработки и развития

В идеальном варианте необходимо, чтобы процесс написания интерфейсов для запросов был минимален, а процесс перенаправления запросов был реализован не через написанную логику, а через определения аннотаций, экономя время для перенаправлений запросов в необходимый обработчик.

Также, необходимо обрабатывать моменты с ошибками при разборе запроса, ведь он может прийти некорректный. Создать собственный класс исключений. Провести рефактор классов и пакетов, удалить не использующийся или устаревший код.

Одна из доработок – апи для получения форматов запроса в виде JSON, что позволит легче ориентироваться при написании запросов.

Тестирование. Оно является неотъемлемой частью при создании больших высоконагруженных систем. Для наилучшего опыта, после рефактора необходимо написать интеграционные и нагрузочные тесты. Это является «программой-минимум». Юнит-тесты являются дорогостоящими и долгими в своей реализации. Интеграционные тесты идеально подходят для тестирования функционала и требуют меньше сил. Нагрузочное тестирование – второстепенное, относительно интеграционных тестов, но не менее важное. Оно позволит выявить слабые места в приложении и сосредоточить на них основные силы для увеличения производительности.

1. План рефакторинга

Первоначальному рефактору должен подвергнуться класс сервлета, принимающего запросы. Для облегчения чтения необходимо вынести создание определенных обработчиков, а также запросов с ответами, можно использовать фабрику. В метод фабрики будет передаваться запрос, а в качестве ответа, фабрика будет отправлять правильный тип обработчика. В рамках не использования «бинов», такой подход будет наиболее правильным.

Стоит пересмотреть интерфейсы для приложения файлов. Существующий подход не является плохим, но он может показаться избыточным. В интерфейсе находится множество методов, что противоречит принципам создания архитектуры приложения.

Необходимо очистить код от методов, в которых находится множество параметров, так как это осложняет использование этого кода в дальнейшем. При написании тестов, что юнит, что интеграционных, использование методов с большим количеством параметров может негативно сказаться на скорости и понимании при написании тестов. Чем меньше параметров – тем лучше. Больше всего это относится к методам работы с файловой системой. При использовании большого количества информации потребуется использовать ДТО классы, которые облегчат понимание кода.

В рамках избавления от второй JSON-библиотеки, необходимо определить точные интерфейсы для ответов и их формирование. Либо это будет переопределенный метод toString() для каждого класса, либо – полноценный класс, собирающий стандартный сформированный ответ и генерирующий json.

10. Выводы. Что было сделано правильно, что не правильно

Правильно:  
1. Контракт обработки запросов. Каждый запрос и ответ прозрачны – они имеют собственные классы и их легко отлаживать. Всегда понятно, на каком этапе произошла ошибка и где смотреть.

2. Классы, хранящие в себе состояние чего-либо. Например – соединение с бд. Данными классами можно пользоваться по всему проекту. Это облегчает их использование там, где это необходимо, без излишних действий по логину пользователя, проверки куки.

Неправильно:  
1. Лишняя библиотека json. Необходимо оставить одну.

2. Интерфейсы DataSource (CMS part) – очень перенасыщены методами. Их необходимо разбить на отдельные интерфейсы или пересмотреть.

3. Сервлеты также перенасыщены кодом. Они должны оставаться лаконичными и читаемыми, вся обработка запросов должна уйти в другие классы со своей архитектурой.

4. В пакете для обработчиков – оставить минимально необходимые пакеты и классы, все остальное – вынести в другой пакет или библиотеку.

5. Конфигурации присутствуют что в конфигурационных файлах, что в web.xml. Необходимо определиться с их назначением, локацией и выбрать что-то одно.

6. Есть типы запросов, которые могут не обработаться правильно. В применении с фронт-ендом этого может быть незаметно, но при дебаге запросами на сервер этого добиться просто, достаточно прислать запрос, не соответствующий протоколу. Для этого необходимо

1) почистить сервлеты;

2) создать классы «разбора» запросов;

3) определить логику при нарушении протокола.

7. Пакет обработки sql запросов нужно пересмотреть, использовать актуальные средства исполнения запросов и самые надежные.

8. Также, в пакете исполнения запросов могут появиться ошибки при использовании базы данных: если при создании файла клиент получит ошибку – папка может быть заблокирована и не обработана должным образом для снятия блокировки. Данные случаи необходимо обрабатывать в ручном режиме. Это ухудшит юзер-экспириенс. Вариантом для данного решения проблемы может стать создание единой структуры для обработки каждого запроса, где при каждой операции происходит правильная обработка того или иного объекта в базе данных.

9. Скачивание в 1 мегабайт может быть достаточно большим объемом, но следует и учитывать заголовки запроса, остальные данные, проходящие через сетевой канал и выбрать актуальный размер для чанков.