Описание функциональных характеристик
«Корпоративная шина данных «Kortex»

Правообладатель:
ООО "Бинари Брейнс"
ИНН: 5029292676
141052, Московская область, г.о. Мытищи, с. Марфино,
ул.Московская, д. 1Ж

О сервисе

«Kortex» — программная платформа (корпоративная шина данных) для интеграции и централизованного обмена данными между приложениями и сервисами.

Платформа предоставляет механизмы описания интеграций «Источник → Трансформации → Цель», приема и отправки данных по HTTP(S), параллельной/последовательной обработки, валидации, аутентификации и планирования по расписанию.

Решение написано на Kotlin/JVM 17 и поставляется в виде исполняемых артефактов/контейнеров.

Назначение

Платформа предназначена для автоматизации бизнес-процессов интеграции, требующих взаимодействия разнородных систем (веб-сервисы, базы данных, файловые хранилища, системы обмена сообщениями).

Возможности

- Источники данных:
  - HTTP Source (встроенный HTTP-сервер на Ktor Netty): регистрация эндпоинтов с заданным методом и путем; валидация Content-Type, контроль размера тела, проверка обязательных заголовков.
  - Schedule Source: таймерный запуск с заданным интервалом (секунды/минуты/часы).
- Аутентификация и ограничение доступа:
  - API Key (заголовок X-API-Key), Bearer Token, Basic Auth.
  - CORS-настройки; ограничение частоты запросов (rate limiting) по IP.
- Обработка и трансформация:

◦ Трансформации данных (синхронные и асинхронные), фильтрация, композиция, трансформация списков.

◦ Параллельная обработка веток (Parallel Integration) с агрегированием результатов и стратегиями обработки ошибок.

• Расширяемость протоколов:

◦ Поддержка расширений для REST, SOAP, JDBC, FTP, LDAP и систем обмена сообщениями (модули-плагины).

• Цели (targets):

◦ HTTP Target (ktor-client CIO): сборка запроса, заголовков и тела; поддержка JSON/XML/форм-данных; управление таймаутами; обработка ответов.

• Управление интеграциями:

◦ Декларативный конвейер: source → transform(s) → target(s), ветвление и объединение.

◦ Глобальная конфигурация HTTP-сервера на уровне приложения.

• Надежность и контроль:

◦ Настройка таймаутов/редиректов, обработка исключений, стандартные JSON-ответы об ошибках.

◦ Логирование в stdout и файл (logback.xml), уровни логов, идентификаторы запросов.

Требования к среде выполнения

• Операционные системы: Linux, Windows, macOS.

• JVM: OpenJDK/Oracle JDK/Axiom JDK 17+.

• Сетевые порты: TCP-порт HTTP-сервера (по умолчанию 8080) должен быть доступен потребителям API.

• Диск/логи: директория для логов (по умолчанию logs/kortex.log) или вывод в stdout при работе в контейнере.

Архитектура высокоуровневая

• Приложение: `KortexApplication` — управляет жизненным циклом интеграций, глобальной конфигурацией HTTP-сервера.

• Источники: `HttpSource`, `ScheduleSource` — обеспечивают триггеры запуска конвейеров.

• Трансформации: `Transform` и производные — композиция преобразований и фильтров.

• Параллельная обработка: `ParallelIntegrationStep` и билдер параллельных веток.

• Цели: `HttpTarget` — обращение к внешним HTTP-сервисам с объединением конфигураций и параметров.