**Kitchen-time XML Parser**

Оглавление

[Архитектура системы Kitchen-time XML Parser 2](#_Toc85674254)

[Файл конфигурации 3](#_Toc85674255)

[Конфигурация cron 4](#_Toc85674256)

[Принцип работы CircuitBreaker 5](#_Toc85674257)

[Запуск и остановка работы парсера 6](#_Toc85674258)

Рекомендую ознакомиться с правой частью (блоки – это папки) картинки архитектуры (блок files). На ней изображены основные файлы, используемые в рамках работы системы.

Основные файлы:

Файл конфигурации парсеров (эксель с списком поставщиков, ссылок на их XML, список брендов) - /files/meta/xml-list.xlsx

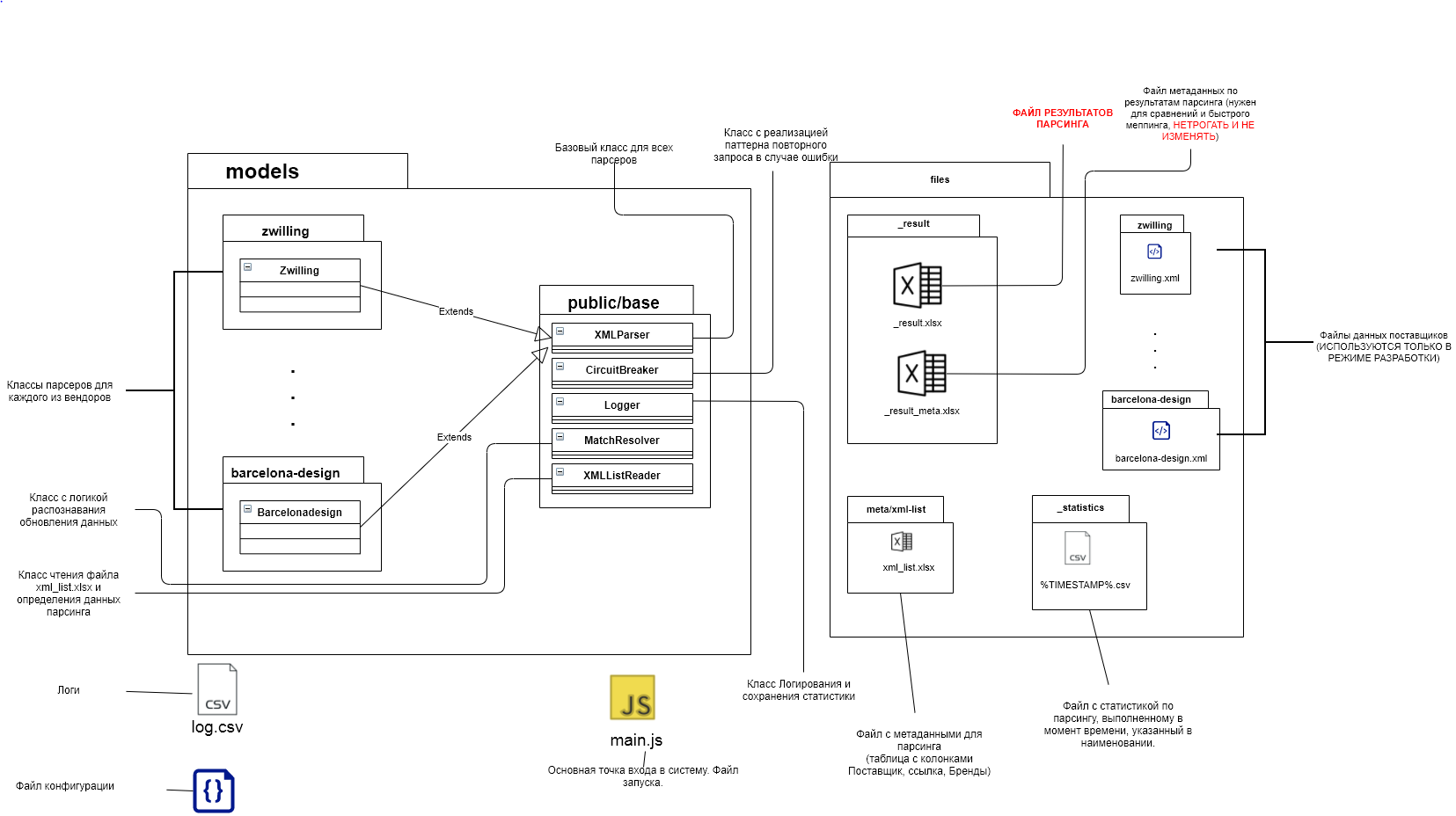
Файл результата отработки парсеров (все, что собралось за последний раз работы парсеров в одном файле) - /files/\_result/\_result.xlsx

Файлы статистики (содержит информацию о новых и старых записях в файле результатов; название файла - время, когда был совершен пирсинг) - /files/\_statistics/%TIMESTAMP%.csv

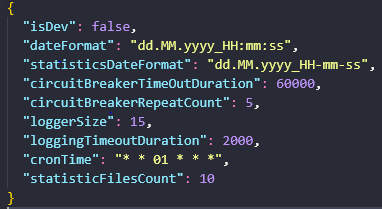
Файл логов (записи поведения работы парсера, какой парсер что делает) - /log.csv

Файл конфигурации работы парсера (скорректировать поведение парсера) - /config.json

# Архитектура системы Kitchen-time XML Parser



# Файл конфигурации



|  |  |  |
| --- | --- | --- |
| **Параметр** | **Значение** | **Пояснение** |
| isDev | Флаг режима разработки. | В случае, если стоит false, то парсер будет обрабатывать данные из папок в директорииfiles (нужно для отладки работы алгоритмов парсинга). В случае true – актуальные данные будут скачиваться с ссылок, указанных в таблице files/meta/xml\_list.xlsx |
| dateFormat | Формат даты для логов | Формат данных согласно <https://date-fns.org/v2.25.0/docs/format> |
| statisticsDateFormat | Формат даты для названия файлов статистик. Не используются символы “:”, так как они запрещены в названиях файлов в Windows | Формат данных согласно <https://date-fns.org/v2.25.0/docs/format> |
| circuitBreakerTimeOutDuration | Длительность таймаута для CircuitBreaker в миллисекундах | Задержка перед повторным запросом, в случае ошибки |
| circuitBreakerRepeatCount | Количество повторных запросов для CircuitBreaker | В случае ошибки при получении данных [CircuitBreaker](#_Принцип_работы_CircuitBreaker) выполнит повторный запрос с таймаутом указанное количество раз |
| loggerSize | Количество строк в файле логов | Используется для контролирования размера файла логов, так как они устаревают со временем |
| loggingTimeoutDuration | Задержка при логировании в миллисекундах | Так как логи пишутся в файловую систему, необходимо сделать задержку для уменьшения нагрузки на сервер |
| cronTime | Конфигурация повторного времени использования парсера | Более подробное использование [см. Конфигурация Cron](#_Конфигурация_cron) |
| statisticFilesCount | Количество файлов статистики в папке files/\_statistics | Так как данные статистики со временем устаревают, необходимо удалять старые для экономии дискового пространства |

Для применения изменений в файле конфигурации рекомендуется перезапустить систему.

# Конфигурация cron

Cron – библиотека, обеспечивающая отработку кода, в зависимости от текущего времени на часах, она имеет особую конфигурацию времени.



Таким образом настройка



Будет свидетельствовать о том, что алгоритмы обработки данных буду отрабатывать только когда на часах будет 01 (т.е. в час ночи). Другой пример:



Парсеры будут отрабатывать каждый раз, когда на часах будет 10 секунд, это значит, что каждую минуту, как только будет 10 секунд на часах - 00:00:10, 00:01:10, 00:02:10 … 00:59:10, 01:00:10, 01:01:10 – 23:59:10

# Принцип работы CircuitBreaker

Circuit Breaker – паттерн, примененный в рамках данного проекта, выступает в роли защиты от обвала работы системы, в случае неудачного запроса на сервер поставщика.

Идея паттерна следующая:

1. Сервер ничего не вернул на запрос системы (не пришел XML файл)
2. Система выполнит повторный запрос на сервер через интервал, указанный в файле конфигурации
3. Система будет выполнять повторные запросы столько раз, сколько указано в файле конфигурации.
4. Если же данные все-таки не пришли, система продолжит свою работу. В файл результатов парсинга перенесутся данные с прошлого результата парсинга.

# Запуск и остановка работы парсера

Постоянную поддержку работы парсера осуществляет утилита forever, принцип работы с ней:

sudo forever list - отобразить список активных сервисов (если парсер был запущен, то он отобразится в списке)

sudo forever start main.js - запуск парсера

sudo forever stop all - остановить работу всех сервисов

Во время работы парсера будут обновляться основные файлы логов и статистик.