



Класс/Интерфейс

Модуль

Назначение

Методы

PipelineBase (ABC)

src/bioetl/core/  
pipeline\_base.py

Абстрактный базовый класс табличного ETL-пайплайна. Оркеструет стадии extract→transform→validate→write, обеспечивая единый интерфейс, логирование, валидацию (через Pandera) и атомарную запись результатов 1 2 . Также централизованно управляет ресурсами (например, закрывает подключения) и поддерживает dry\_run режим без записи 3 4 .

```
run(output)
dry_run: bool
RunResult:
    пайплайна (
        transform
    необходимое
    времени и р
    результат за
    6 .<br> ex
pd.DataFrame
сырые данные
API, файлы и
DataFrame
pd.DataFrame
Абстрактны
трансформа
обогащение
валидацией
    8 .<br> va
pd.DataFrame
структуру да
Pandera-схема
свойство ли
DataFrame, в
отсутствии с
упорядочите
добавить хэ
строк 9 10
pd.DataFrame
WriteResult
заданным ко
записывает
Parquet по р
успешной за
запуска (нап
информации
количество :
    12 .<br> re
client: An
внешний ре
реестре для
    13 .<br> cl
Закрывает в
ресурсы, вы
dispose()
но не прерыв
```

Класс/Интерфейс	Модуль	Назначение	Методы
StageABC	<code>src/bioetl/core/pipeline/base.py</code>	Абстракция отдельной стадии пайплайна. Определяет интерфейс выполнения шага и освобождения ресурсов, обеспечивая изоляцию логики этапа и возможность композиции этапов внутри пайплайна <sup>2</sup> .	<code>run(context)</code> логику стадии данные и возвращаемые передачи словаря реализация этапа. ресурсами, используя закрывает сценарий завершения использование менеджера контекста
PipelineHookABC	<code>src/bioetl/core/pipeline/hook.py</code>	Интерфейс хуков конвейера для мониторинга. Получает события начала и окончания стадий, а также ошибки, не вмешиваясь в логику, и позволяет подключить логирование, метрики или другую обработку событий пайплайна <sup>19</sup> .	<code>on_stage_start(stage, context)</code> <code>None</code> : Вызывается в начале выполнения логирования этапа. StageABC() в успешном завершении фиксируется выполнением StageABC. Вызывается – позволяет реализовать
CLICommandABC	<code>src/bioetl/core/pipeline/cli_command.py</code>	Интерфейс для интеграции пайплайна с CLI. Определяет контракты для регистрации CLI-команды и запуска пайплайна с переданными параметрами конфигурации, обеспечивая единообразный способ запуска из командной строки <sup>19</sup> .	<code>register_command(command)</code> Регистрирует команду в CLI-приложении. <code>click / type</code> контейнер команд. PipelineConfig – Инициализирует данной конфигурации. PipelineBuilder – выполняет

Класс/Интерфейс	Модуль	Назначение	Методы
SourceClientABC	src/bioetl/core/pipeline/source/client.py	<p>Интерфейс клиента внешнего источника данных. Описывает unified API для извлечения данных из внешних систем (REST API, баз данных и т.п.), скрывая детали HTTP/DB запросов.</p> <p>Поддерживает одиночные и пагинированные выборки данных <sup>20</sup> <sub>21</sub>.</p>	<p>fetch_one Record   None возвращает или None, о 23 .&lt;br&gt; fe ClientRequ Возвраща полученным разбиение г 25 .&lt;br&gt; it ClientRequ Возвраща данных (кажд записей и и 26 27 .&lt;br&gt; me Any] : Пред (например, с маршруты) Закрывает с клиента, если сессии HTTP</p>

Класс/Интерфейс	Модуль	Назначение	Методы
-----------------	--------	------------	--------

BaseExternalDataClient	<code>src/bioetl/core/pipeline/source/client.py</code>	<p>Базовый класс для клиентов внешних API, реализующий <code>SourceClientABC</code>. Содержит общую логику пагинации и сбор данных из REST API (циклический запрос страниц, объединение параметров и т.п.) <sup>32</sup>. Конкретные клиенты (ChEMBL, PubChem и др.) наследуются от этого класса, переопределяя при необходимости специфичные части.</p>	<code>fetch_one</code>
			<code>Record   N</code>
RateLimiterABC	<code>src/bioetl/core/pipeline/source/client.py</code>	<p>Интерфейс ограничителя частоты запросов. Обеспечивает соблюдение лимитов нагрузки при обращении к внешнему API (например, не более N запросов в секунду), чтобы избежать превышения квот или блокировок <sup>20</sup>.</p>	<code>acquire()</code>
			<code>приостанавливается пока не наст следующего заданным ли время).&lt;br&gt; внутренние окна време</code>

Класс/Интерфейс	Модуль	Назначение	Методы
RetryPolicyABC	<code>src/bioetl/core/pipeline/source/client.py</code>	Интерфейс политики повторных попыток запроса. Определяет, в каких случаях и сколько раз повторять запрос при ошибках, а также с каким интервалом (включая стратегию экспоненциального backoff) <sup>20</sup> .	<code>should_reattempt: int</code> следует ли в при данном попытке (всегда повторить). <code>-&gt; float: float</code> секундах) исходя из нереализации экспоненциальной стратегии.
RequestBuilderABC	<code>src/bioetl/core/pipeline/source/request_builder.py</code>	Интерфейс построения запросов к внешнему источнику. Отвечает за формирование объекта запроса ( <code>ClientRequest</code> ) на основе входных параметров бизнес-логики (например, идентификаторов, фильтров), инкапсулируя логику формирования URL и параметров для клиента <sup>20</sup> .	<code>build_request: ClientRequest</code> объект запроса ( <code>ClientRequest</code> ). параметры на страницы. Ресурсы API (маршруты) формирует эти знания детально.
ResponseParserABC	<code>src/bioetl/core/pipeline/source/parser.py</code>	Интерфейс парсера отклика от внешнего источника. Определяет преобразование сырых данных ответа (обычно JSON) в стандартные записи (словарь <code>Record</code> ) для последующей обработки в пайплайне <sup>20</sup> . Отделяет логику разбора и нормализации ответа API от логики его получения.	<code>parse(raw_data: Any, page: int) Iterator[Record]</code> (например, JSON-страницы). Извлеченные записи ( <code>Record</code> ) позволяют организовать стримингом.
PaginatorABC	<code>src/bioetl/core/pipeline/source/parser.py</code>	Интерфейс логики пагинации для разбора ответов. Описывает способ извлечения списка элементов и маркера продолжения из полученного ответа API, абстрагируя различия в форматах пагинации у разных источников <sup>20</sup> .	<code>extract_items(response: Response, cursor: str) list[Record]</code> ответа списка страницы. Несложный JSON-ответе содержит элемент в формате <code>&lt;br&gt;</code> . <code>-&gt; str   None</code> маркер или страницу (на заданному курсором), дальнейших.

Класс/Интерфейс	Модуль	Назначение	Методы
TransformerABC	<code>src/bioetl/core/pipeline/processing/transform.py</code>	Интерфейс трансформации данных. Определяет детерминированные преобразования входного DataFrame без взаимодействия с внешними системами – например, очистку значений, переименование колонок, приведение типов – перед дальнейшим обогащением и валидацией <sup>43</sup> .	<code>transform</code> <code>pd.DataFrame</code> чисто функциональные изменения вне DataFrame, новый DataFrame с новыми данными. Гарантирует одинаковое идентичное обогащение (детерминированное)
LookupEnricherABC	<code>src/bioetl/core/pipeline/processing/enrich.py</code>	Интерфейс обогащения данных с помощью внешних справочных данных (lookup). Отвечает за добавление в таблицу недостающей информации из внешних источников/словарей (например, подтягивает данные по внешним ключам) <sup>43</sup> . Использует поставщика побочных данных ( <code>SideInputProviderABC</code> ) для доступа к справочникам.	<code>enrich(df)</code> <code>SideInputProviderABC</code> <code>pd.DataFrame</code> DataFrame, имена, данные из словаря, добавлять новые значения именем, данными из словаря, новый DataFrame с новой информацией
SideInputProviderABC	<code>src/bioetl/core/pipeline/processing/enrich.py</code>	Интерфейс поставщика сторонних (побочных) данных для обогащения. Абстрагирует доступ к справочникам, кэшам или внешним источникам, которые используются в процессах обогащения (lookup) для получения дополнительной информации <sup>43</sup> .	<code>get_data()</code> Предоставляет данные в заданному контексте. Например, словарь с данными в таблице, если это может осуществляться из файла, БД, API и т.д. подготовленные
DeduplicatorABC	<code>src/bioetl/core/pipeline/processing/dedup.py</code>	Интерфейс для выявления и удаления дубликатов в данных. Определяет контракт консолидации записей с одинаковыми бизнес-ключами и устранения повторов перед загрузкой данных, обеспечивая идемпотентность пайплайна <sup>43</sup> .	<code>deduplicate</code> <code>pd.DataFrame</code> записи в DataFrame определенные совпадения DataFrame, имена (при необходимости) информации стратегии слияния использует вычисления хэширования <code>MergeStrategy</code> для обработки дубликатов.

Класс/Интерфейс	Модуль	Назначение	Методы
BusinessKeyDeriverABC	<code>src/bioetl/core/pipeline/processing/dedup.py</code>	Интерфейс вычисления бизнес-ключа записи. Бизнес-ключ – человечески осмысленный уникальный идентификатор сущности (например, комбинация нескольких полей), используемый для определения дубликатов независимо от технических первичных ключей <sup>44</sup> <sup>45</sup> .	<code>derive_key</code> Вычисляет-business-key (словаря) на подмножестве. Например, для каждого ключом может быть название+value строку, однозначно определяющее бизнес-сущность.
HasherABC	<code>src/bioetl/core/pipeline/processing/dedup.py</code>	Интерфейс хеширования записи. Определяет способ вычисления детерминированного хэша (например, MD5, SHA) по содержимому записи для целей идентификации и сравнения записей <sup>46</sup> <sup>45</sup> . Хэши применяются для отслеживания изменений и проверки идемпотентности.	<code>hash(record)</code> Вычисляет контрольный хэш. Гарантирует содержание записи неизменным хэш. Реализует хеширование строки.
MergeStrategyABC	<code>src/bioetl/core/pipeline/processing/dedup.py</code>	Интерфейс стратегии слияния дубликатов. Определяет, как несколько записей, признанных дубликатами, объединяются в одну “консолидированную” запись перед загрузкой, чтобы информация не терялась <sup>47</sup> .	<code>merge(records)</code> <code>Record</code> : Объект, представляющий дубликатов. Реализация слияния записей браузера предпочитает максимальное объединять записи, содержащие одинаковые данные.
SchemaProviderABC	<code>src/bioetl/core/pipeline/validation/schema.py</code>	Интерфейс поставщика схемы данных. Предоставляет описание структуры и типов данных, ожидаемых на этапе валидации, абстрагируя конкретную реализацию схем (Pandera, Pydantic и т.п.) от остальной логики пайплайна <sup>45</sup> .	<code>get_schema</code> объект схемы <code>DataFrame</code> аналог из другого языка данных, обрабатывающих Конкретные схемы программы получать из

Класс/Интерфейс	Модуль	Назначение	Методы
ValidatorABC	<code>src/bioetl/core/pipeline/validation/validator.py</code>	<p>Интерфейс валидатора данных. Обобщает применение схемы и правил качества данных (DQ) к результирующему DataFrame, формируя отчет о найденных нарушениях. Позволяет отделить проверку данных от бизнес-логики пайплайна <sup>45</sup>.</p>	<code>validate()</code> <code>Validation</code> на соответствие правилам как <code>SchemaPro</code> и набор <code>DQ</code> проверок (не обязательно результата в сведения о г проверках (не ошибок)) <sup>45</sup>
DQRuleABC	<code>src/bioetl/core/pipeline/validation/dq_rules.py</code>	<p>Интерфейс правила качества данных (Data Quality Rule). Определяет отдельное проверочное правило, не выражаемое лишь схемой (например, межколоночные зависимости, допустимые диапазоны значений, уникальность и пр.). Выделение DQ-правил позволяет гибко расширять проверки качества без изменения схемы <sup>48</sup>.</p>	<code>check(df: list[DQIssue])</code> качества к DataFrame обнаруженных пустой списка проверку. Каждую проблему (недопустимого включает с Реализация проверять для результатов.
PathStrategyABC	<code>src/bioetl/core/pipeline/output/path_strategy.py</code>	<p>Интерфейс стратегии формирования путей вывода. Определяет, как строятся пути и имена файлов для результатов пайплайна, чтобы обеспечить детерминизм и организованность вывода (например, единообразное именование по сущности, источнику, дате) <sup>49</sup>.</p>	<code>make_path(str, base)</code> Генерирует для данного название папки запуска и базы, может добавлять подпапки по входных знаков одинаковый
WriterABC	<code>src/bioetl/core/pipeline/output/writer.py</code>	<p>Интерфейс компонента записи данных. Отвечает за непосредственную запись итогового DataFrame на диск (в файл) с обеспечением атомарности и соблюдением форматов. Инкапсулирует детали сортировки, выбора формата (CSV/Parquet) и предотвращения частично записанных результатов <sup>49</sup>.</p>	<code>write(df: DataFrame) &gt; None</code> : Загружает DataFrame в файл, должна: отсортировать по столбцам (за исключением пайплайна), расширение <code>.parquet</code> , имя и затем установить атомарности записи должны

Класс/Интерфейс	Модуль	Назначение	Методы
MetadataWriterABC	<code>src/bioetl/core/pipeline/output/writer.py</code>	Интерфейс записи метаданных запуска. Отвечает за сохранение вспомогательной информации о выполнении пайплайна – статистики и параметров запуска – в отдельный файл (например, <code>meta.yaml</code> рядом с данными) <sup>49</sup> .	<code>write_meta(meta_path: Path)</code> <code>meta_path: Path</code> метаданные файла. Реализует необходимый идентификатор времени начала этапов, количественный флаг <code>dry_run</code> файла, хэш-код человекачитаемого YAML или JSON.
ConfigResolverABC	<code>src/bioetl/core/pipeline/utils/config.py</code>	Интерфейс загрузки и разрешения конфигурации пайплайна. Инкапсулирует логику чтения конфигурационных файлов (например, YAML), слияния с дефолтами, применения переменных окружения и других преобразований, обеспечивая единый способ получения настроек для запуска пайплайна <sup>51</sup> .	<code>resolve(config_name: str)</code> <code>PipelineConfig</code> пайплайна генерации Реализация конфигурации <code>&lt;config_name&gt;</code> возвращающий объект Также может возвращать секретов из <code>SecretProvider</code> валидации с помощью проверки.
SecretProviderABC	<code>src/bioetl/core/pipeline/utils/config.py</code>	Интерфейс доступа к секретам (пароли, токены API и др.). Предоставляет абстракцию над хранилищем чувствительных данных (например, переменными окружения, Vault, AWS Secrets Manager), позволяя безопасно подключать секреты в конфигурацию пайплайна <sup>51</sup> .	<code>get_secret(secret_name: str)</code> Возвращает пароль по имени/ключу источника (настройки окружения с помощью обращения к <code>os.environ</code> ) и обеспечивает напрямую возврат пароля, который может генерироваться.
CacheABC	<code>src/bioetl/core/pipeline/utils/cache.py</code>	Интерфейс кэша для промежуточных данных. Предоставляет унифицированный доступ к механизму сохранения и повторного использования данных между запусками или между этапами (например, кэширование результатов запросов к API, чтобы не дублировать вызовы) <sup>51</sup> .	<code>get(key: Any)</code> из кэша по ключу такой записи <code>value: Any</code> заданный обработчиком (перезапись существующего или определение нового размера и т.д.)

Класс/Интерфейс	Модуль	Назначение	Методы
LoggerAdapterABC	src/bioetl/core/pipeline/utils/logging.py	Интерфейс адаптера логирования. Обеспечивает унифицированный способ структурированного логирования событий пайплайна, не зависящий от конкретной библиотеки логирования. Позволяет привязывать контекст (например, <code>run_id</code> , имя пайплайна) и логировать события пайплайна консистентно <sup>51</sup> .	bind(**kwargs) LoggerAdapter логгер/адаптер контекстным <code>run_id</code> для сообщений) **kwargs : информация названием/о полями данн значение в <code>str</code> , **kwa событие уро фиксирован ситуаций, сс данными.
TracerABC	src/bioetl/core/pipeline/utils/tracing.py	Интерфейс трассировки выполнения. Позволяет интегрировать пайплайн с системами трассировки (distributed tracing) или собственным механизмом отслеживания выполнения этапов. Предоставляет методы для обозначения начала/конца важных операций, что упрощает отладку и мониторинг производительности <sup>51</sup> .	start_span() Отмечает на или этапа с трассировоч например, с распределен трассировке None : Отме операции. И продолжите Реализация <code>start_spar</code> контексту.
ErrorPolicyABC	src/bioetl/core/pipeline/utils/error.py	Интерфейс политики обработки ошибок. Определяет стратегию реагирования на исключения, возникающие во время выполнения пайплайна: продолжать выполнение, пропускать текущий элемент, повторять попытку или прерывать пайплайн <sup>52</sup> . Позволяет отделить логику обработки сбоев от основного кода пайплайна.	on_error( Exception) оркестратор на стадии <code>s</code> ErrorAction дальше (нап текущую зап конвейер). Р исключения этапа и друг политики.

Класс/Интерфейс	Модуль	Назначение	Методы
ProgressReporterABC	src/bioetl/core/pipeline/utils/progress.py	<p>Интерфейс агрегированного репортинга прогресса пайплайна. Собирает и регистрирует статистику по ходу выполнения: количество успешно обработанных записей, количество отброшенных/невалидных, причины отбраковки и пр. – без влияния на сам процесс обработки <a href="#">45</a> <a href="#">53</a>. Разделяет сбор статистики от бизнес-логики, предоставляя единый источник правды о ходе выполнения.</p>	<p>on_stage_... Фиксирует н... (например, ... логирует ст... этапа).&lt;br&gt;... str) -&gt; No... обработанн... (вызывается обработки, л... валидатор... указанием причины).&lt;br&gt;... str) -&gt; No... стадии, окон... (время, коли... отброшенн... и сохранен... и</p>

- [1](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#) [16](#) [17](#) [18](#) [50](#) PipelineBase\_description.md  
[https://github.com/SatoryKono/bioactivity\\_data\\_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/\\_docs/PipelineBase\\_description.md](https://github.com/SatoryKono/bioactivity_data_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/_docs/PipelineBase_description.md)
- [2](#) [20](#) [43](#) [44](#) [45](#) [46](#) [49](#) [51](#) architecture\_plan.md  
[https://github.com/SatoryKono/bioactivity\\_data\\_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/\\_docs/architecture\\_plan.md](https://github.com/SatoryKono/bioactivity_data_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/_docs/architecture_plan.md)
- [19](#) [47](#) [48](#) [52](#) abstract\_objects\_implementation\_plan.md  
[https://github.com/SatoryKono/bioactivity\\_data\\_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/\\_docs/abstract\\_objects\\_implementation\\_plan.md](https://github.com/SatoryKono/bioactivity_data_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/_docs/abstract_objects_implementation_plan.md)
- [21](#) [22](#) [23](#) [24](#) [25](#) [26](#) [27](#) [28](#) [29](#) [30](#) [31](#) [32](#) [33](#) [34](#) [35](#) [36](#) [37](#) [38](#) [39](#) [40](#) [41](#) [42](#) client\_abc.py  
[https://github.com/SatoryKono/ChEMBL\\_data\\_acquisition/blob/14d2538df65b6d02dac4cb4703ee7ba34b501e32/src/bioetl/clients/base/client\\_abc.py](https://github.com/SatoryKono/ChEMBL_data_acquisition/blob/14d2538df65b6d02dac4cb4703ee7ba34b501e32/src/bioetl/clients/base/client_abc.py)
- [53](#) ProgressReporterABC.md  
[https://github.com/SatoryKono/bioactivity\\_data\\_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/\\_docs/ProgressReporterABC.md](https://github.com/SatoryKono/bioactivity_data_acquisition/blob/98f17f432532cddd56d0a07fd4796b37c54677ec/_docs/ProgressReporterABC.md)