Common New

▼ public class JsonUtil

1. Class Information

- **Class Name:** `JsonUtil`
- **Purpose:**
  - Provides a fixed version of `JsonObject.append()`, preventing nested arrays.
  - Ensures that a given key in a `JSONObject` always contains a valid `JSONArray`.
  - Allows appending `JSONObject` values to a `JSONArray` within a `JSONObject`.

---

2. Attributes and Methods

Attributes

(None—this class only contains static methods.)

Methods

Method Signature	Access Modifier	Description
`public static JSONObject append(JSONObject o, String key, JSONObject value) throws JSONException`	`public static`	Appends a `JSONObject` value to a `JSONArray` within a `JSONObject`. Ensures correct structure and avoids nested arrays.

---

3. Relationships

Inheritance

- **Parent Class:** None
- **Child Classes:** None

Associations

- None

▼ public class DialectUtil

---

1. Class Information

- **Class Name:** `DialectUtil`
- **Purpose:**
  - Provides methods for handling database dialect differences.

- Converts SQL statements between different database dialects (HSQLDB and PostgreSQL).
- Offers utility functions for date calculations, timestamps, and null handling.

2. Attributes and Methods

Attributes

(None—this class contains only static methods.)

Methods

Method Signature	Access Modifier	Description
`public static boolean isObjectNotFound(String message)`	`public static`	Checks if the given error message corresponds to an object-not-found error for HSQLDB or PostgreSQL.
`public static String transform(String sql)`	`public static`	Transforms the given SQL query to match the current database dialect.
`public static String transformToPostgresql(String sql)`	`public static`	Converts HSQLDB-specific SQL syntax to PostgreSQL-compatible syntax.
`public static String getDateDiff(String field, String diff, String unit)`	`public static`	Returns the SQL expression to calculate the difference between two dates in the appropriate database dialect.
`public static String getTimeStamp(String value)`	`public static`	Returns a timestamp in milliseconds, formatted for the current database dialect.
`public static String getNullParameter(String parameterName, Object value)`	`public static`	Returns the SQL representation of a bound parameter, handling null values properly for JPA queries.

3. Relationships

Inheritance

- **Parent Class:** None
- **Child Classes:** None

Associations

- **Dependency:**
  - `EMF`: Utility class for determining the current database driver.
  - `RuntimeException`: Thrown when an unknown database type is encountered.

▼ public class NetworkUtil

1. Class Information
   - Class Name: NetworkUtil

- Purpose: The `NetworkUtil` class provides utility methods to facilitate network-related operations. Its primary function is to attempt port mapping on a network gateway using UPnP, allowing external access to services running on a local machine.

2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
`log`	`private static final`	`Logger`	Logger instance for logging network-related messages.

   - Methods:

Method Signature	Access Modifier	Description
`boolean mapTcpPort(int port)`	`public static`	Attempts to map a TCP port on the gateway using UPnP. Returns `true` if successful, `false` otherwise.

3. Relationships:
   - Inheritance: NA
   - Associations :
     - Aggregation :
       - Logger
     - Dependency :
       - GatewayDiscover
       - GatewayDevice

▼ public class JsonValidationUtil

1. Class Information
   - Class Name: JsonValidationUtil
   - Purpose: This class provides static methods to validate JSON properties by checking if they exist and ensuring they are of the expected data type. It throws exceptions when the validation fails, helping maintain data integrity in JSON processing.
2. Attributes and Methods
   - Attributes: NA
   - Methods:

Method Signature	Access Modifier	Description

`void validateJsonRequired(JsonNode n, String name)`	`public static`	Checks if a JSON node contains a required property. Throws an exception if the property is missing.
`void validateJsonObject(JsonNode n, String name, boolean required)`	`public static`	Ensures a specified property exists and is a JSON object. Throws an exception if validation fails.
`validateJsonNumber(JsonNode n, String name, boolean required)`	`public static`	Validates that a property exists and is a number. Throws an exception if validation fails.
`void validateJsonLong(JsonNode n, String name, boolean required)`	`public static`	Ensures a property exists and is a long integer. Throws an exception if validation fails.
`void validateJsonString(JsonNode n, String name, boolean required)`	`public static`	Ensures a property exists and is a string. Throws an exception if validation fails.
`void validateJsonArray(JsonNode n, String name, boolean required)`	`public static`	Checks if a property exists and is an array. Throws an exception if validation fails.

3. Relationships:
    - Inheritance: NA
    - Associations : NA

▼

▼ public class AppContext

1. Class Information
    - Class Name: AppContext
    - Purpose:
        - Provides a centralized application context to manage services and event buses.
        - Ensures only one instance of core services (FeedService, IndexingService) exists.
        - Manages asynchronous event handling using event buses and thread executors.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
instance	private static	AppContext	Singleton instance of AppContext.
eventBus	private	EventBus	Synchronous event bus for handling application events.
asyncEventBus	private	EventBus	Asynchronous event bus for handling background events.

mailEventBus	private	EventBus	Event bus for email processing.
importEventBus	private	EventBus	Event bus for handling import jobs.
feedService	private	FeedService	Manages RSS feed synchronization.
indexingService	private	IndexingService	Manages Lucene search indexing.
asyncExecutorList	private	List<ExecutorService>	List of asynchronous executors for handling background tasks.

- Methods:

Method Signature	Access Modifier	Description
AppContext()	private	Private constructor to enforce singleton pattern.
AppContext getInstance()	public static	Returns the singleton instance of AppContext.
void resetEventBus()	private	Initializes and resets all event buses.
EventBus newAsyncEventBus()	private	Creates a new asynchronous event bus with a dedicated executor.
void waitForAsync()	public	Waits for all asynchronous tasks to complete.
EventBus getEventBus()	public	Returns the synchronous event bus.
EventBus getAsyncEventBus()	public	Returns the asynchronous event bus.
EventBus getMailEventBus()	public	Returns the event bus for email events.
EventBus getImportEventBus()	public	Returns the event bus for import operations.
FeedService getFeedService()	public	Returns the feed synchronization service.
IndexingService getIndexingService()	public	Returns the Lucene indexing service.

3. Relationships:
   - Inheritance: NA
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - AppContext
       - ConfigDao
       - Config
       - EnvironmentUtil

- Aggregation:
  - EventBus
  - ExecutorService
- Composition:
  - FeedService
  - IndexingService

▼ public class ReaderAgent

1. Class Information
   - Class Name: `ReaderAgent`
   - Purpose: This class functions as the Windows agent for managing the lifecycle of the Reader application. It handles deployment, UI configuration, process elevation, and application startup operations.
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
listeners	private	`List<DeploymentStatusListener>`	Stores a list of deployment status listeners.
trayController	private	`TrayController`	Manages system tray operations.
frame	private	`AgentFrame`	Represents the UI frame for the application.
elevated	private	`boolean`	Indicates whether the process is running with elevated privileges.
readerDeployer	private	`ReaderDeployer`	Handles the deployment status and control operations.
setting	private	`Setting`	Stores application settings.

   - Methods:

Method Signature	Access Modifier	Description
`ReaderAgent()`	public	Constructor that initializes settings, deployment, and UI configurations.
`void notifyDeploymentInfo()`	public	Notifies all listeners about the current deployment status.
`private void setLookAndFeel()`	private	Configures the UI look and feel for Windows.

`void checkElevation(String... args)`	public	Checks and elevates process privileges if required.
`private boolean isElevationNeeded()`	private	Determines if elevation is required based on OS version.
`void addListener(DeploymentStatusListener listener)`	public	Registers a new deployment status listener.
`void showStatusPanel()`	public	Displays the application's status panel.
`void showTrayIconMessage()`	public	Shows a notification message in the system tray.
`void exit()`	public	Uninstalls tray components and exits the application.
`void openBrowser()`	public	Opens the Reader application in a web browser.
`private void start(List<String> args)`	private	Handles application startup logic based on provided arguments.
`static void main(String[] args)`	public	Main entry point for launching the agent.
`ReaderDeployer getReaderDeployer()`	public	Retrieves the `ReaderDeployer` instance.
`private boolean isElevated()`	private	Returns whether the process is running with elevated privileges.
`Setting getSetting()`	public	Returns the `Setting` instance.

3. Relationships:
    - Inheritance: NA
    - Associations
        - Dependency:
            - DeploymentStatus
            - EnvironmentUtil
            - UIManager
            - JOptionPane
            - ProcessBuilder
            - Desktop
        - Aggregation:
            - DeploymentStatusListener
        - Composition:
            - AgentFrame
            - ReaderDeployer

▼ public class ReaderDeployer

1. Class Information
    - Class Name: ReaderDeployer
    - Purpose: This class deploys the Reader application as a standalone service using an embedded Jetty server. It manages server startup, configuration, error handling, and provides deployment status updates.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
READER_WAR	private static final	String	Constant representing the WAR file name.
startTime	private	Date	The startup time of the server.
server	private	Server	The embedded Jetty server instance.
readerAgent	private	ReaderAgent	Reference to the `ReaderAgent` managing deployment.
exception	private	Exception	Stores any exception occurring during server startup.
serverState	private	ServerState	Tracks the current state of the server.

    - Methods:

Method Signature	Access Modifier	Description
`ReaderDeployer(ReaderAgent readerAgent)`	public	Constructor initializing the deployer with the reader agent.
`void start()`	public	Starts the Jetty server, sets up security, and deploys the application.
`void stop()`	public	Stops the Jetty server if running.
`String getErrorMessage()`	public	Returns a user-friendly error message if server startup fails.
`int getMemoryUsed()`	public	Returns the amount of memory used by the application in MB.
`String getUrl()`	public	Constructs and returns the deployment URL based on settings.
`DeploymentStatus getDeploymentStatus()`	public	Retrieves the current deployment status.
`void lifeCycleStarting(LifeCycle event)`	public	Updates server state to STARTING and notifies agent.
`void lifeCycleStarted(LifeCycle event)`	public	Updates server state to STARTED and notifies agent.
`void lifeCycleStopping(LifeCycle event)`	public	Updates server state to STOPPING and notifies agent.

`void lifeCycleStopped(LifeCycle event)`	public	Updates server state to STOPPED and notifies agent.
`void lifeCycleFailure(LifeCycle event, Throwable cause)`	public	Handles lifecycle failure and notifies agent.

3. Relationships:
    - Inheritance: Implements `LifeCycle.Listener` interface for handling server lifecycle events.
    - Associations
        - Dependency:
            - Setting
            - DeploymentStatus
            - SelectChannelConnector
            - SslSocketConnector
            - WebAppContext
        - Aggregation:
            - ReaderAgent

▼ public class AgentFrame

1. Class Information
    - Class Name: AgentFrame
    - Purpose: The `AgentFrame` class extends `JFrame` and implements `DeploymentStatusListener` to provide a user interface for managing and monitoring the reader agent. It contains status and settings panels and updates the UI based on deployment status.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Serial version identifier for serialization.
`statusPanel`	`private final`	`StatusPanel`	Panel displaying the server status.
`settingPanel`	`private final`	`SettingPanel`	Panel for configuring settings.
`tabbedPane`	`private`	`JTabbedPane`	Tabbed pane for switching between status and settings panels.
`closeButton`	`private`	`JButton`	Button to close the window.
`startedImage`	`private`	`Image`	Icon displayed when the agent is running.
`stoppedImage`	`private`	`Image`	Icon displayed when the agent is stopped.

    - Methods:

Method Signature	Access Modifier	Description
`AgentFrame(ReaderAgent readerAgent)`	`public`	Constructor initializing the UI, loading icons, and registering as a listener.
`centerComponent()`	`public`	Centers the frame in the middle of the screen.
`initComponent()`	`private`	Initializes UI components and layout.
`showStatusPanel()`	`public`	Displays the status panel, refreshes settings, and brings the frame to the front.
`showSettingPanel()`	`public`	Displays the settings panel, refreshes settings, and brings the frame to the front.
`setIcon(Image image)`	`private`	Updates the frame icon image based on deployment status.
`notifyDeploymentStatus(DeploymentStatus deploymentStatus)`	`public`	Updates the UI icon based on deployment status changes.

3. Relationships:
    - Inheritance:
        - Parent Class: JFrame
        - **Implemented Interface:** `DeploymentStatusListener`
    - Associations
        - Dependency:
            - ReaderAgent
            - DeploymentStatus
            - Toolkit
            - MessageUtil
        - Aggregation:
            - Image
            - JButton
            - JTabbedPane
        - Composition:
            - StatusPanel
            - SettingPanel

▼ public class TrayController

1. Class Information
    - Class Name: TrayController
    - Purpose: The `TrayController` class manages the system tray icon for the Reader application. It provides menu options for opening the browser, showing the control panel, and quitting the application. It also updates the tray icon based on the deployment status.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`readerAgent`	private	`ReaderAgent`	Reference to the `ReaderAgent` for controlling application behavior.
`trayIcon`	private	`TrayIcon`	Represents the system tray icon.
`openAction`	private	`Action`	Action for opening the browser.
`controlPanelAction`	private	`Action`	Action for opening the control panel.
`hideAction`	private	`Action`	Action for quitting the application.
`startedImage`	private	`Image`	Image to be displayed when the reader is running.
`stoppedImage`	private	`Image`	Image to be displayed when the reader is stopped.

    - Methods:

Method Signature	Access Modifier	Description
`TrayController(ReaderAgent readerAgent)`	public	Constructor that initializes the tray icon and menu actions.
`void showMessage()`	public	Displays a notification message in the system tray.
`private Image createImage(String resourceName)`	private	Loads an image resource for the tray icon.
`private MenuItem createMenuItem(Action action)`	private	Creates a menu item with an associated action.
`void uninstallComponents()`	public	Removes the tray icon from the system tray.
`private void setTrayImage(Image image)`	private	Updates the tray icon image based on server state.
`void notifyDeploymentStatus(DeploymentStatus deploymentStatus)`	public	Updates the tray icon based on the server's deployment status.

3. Relationships:
    - Inheritance:

    - Parent Class: `DeploymentStatusListener` (Implements interface to receive deployment status updates)
    - Child Classes:
- Associations
    - Dependency:
        - SystemTray
        - MessageUtil
        - DeploymentStatus
        - ServerState
    - Aggregation:
        - Aggregation
        - Action
    - Composition:
        - TrayIcon
        - Image
        - PopupMenu
        - MenuItem

▼

▼ public class IndexingService

1. Class Information
    - Class Name: IndexingService
    - Purpose:
        - Maintains a searchable Lucene index of RSS articles.
        - Supports scheduled updates and manual index rebuilding.
        - Ensures efficient search queries for users based on article content.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
log	private static	Logger	Logger for indexing errors.
directory	private	Directory	Represents Lucene's index storage.
directoryReader	private	DirectoryReader	Allows reading the Lucene index.

luceneStorageConfig	private	String	Determines whether RAM or File storage is used.

- Methods:

Method Signature	Access Modifier	Description
void startUp()	protected	Initializes Lucene storage, setting it to RAM or File storage.
void shutDown()	protected	Closes the Lucene index and reader on shutdown.
void runOneIteration()	protected	Runs one scheduled update (currently does nothing).
Scheduler scheduler()	protected	Schedules indexing every hour.
PaginatedList<UserArticleDto> searchArticles(String userId, String searchQuery, Integer offset, Integer limit)	public	Executes a search query on articles, returning paginated results.
void rebuildIndex()	public	Destroys and rebuilds the Lucene index asynchronously.
Directory getDirectory()	public	Returns the Lucene directory instance.
DirectoryReader getDirectoryReader()	public	Returns a valid Lucene reader, reopening it if needed.

3. Relationships:
   - Inheritance:
     - Parent Class: AbstractScheduledService
     - Child Classes:
   - Associations
     - Dependency:
       - TransactionUtil: Ensures safe indexing transactions.
       - ArticleDao: Searches articles using Lucene queries.
       - UserArticleDao: Retrieves user-specific article metadata.
       - PaginatedLists: Handles paginated search results.
       - RebuildIndexAsyncEvent: Triggers index rebuilding when needed.
       - UserArticleCriteria
       - LoggerFactory
       - PaginatedList

- Aggregation:
  - Directory: Manages Lucene's index storage.
  - DirectoryReader: Reads existing indexes.
- Composition:

▼ public class RebuildIndexAsyncEvent

1. Class Information
   - Class Name: RebuildIndexAsyncEvent
   - Purpose: This class serves as an event object indicating that the system should initiate an asynchronous rebuild of the index of stored RSS feed articles. It is useful for ensuring that the indexing system is kept up-to-date, allowing for fast retrieval and searching of articles.
2. Attributes and Methods
   - Attributes: NA
   - Methods:

Method Signature	Access Modifier	Description
String toString()	public	Overrides the default toString() method to provide a string representation of the object using Objects.toStringHelper() from Guava.

3. Relationships:
   - Inheritance:
     - Parent Class:
   - Associations
     - Dependency: NA

▼ public class RebuildIndexAsyncListener

1. Class Information
   - Class Name: RebuildIndexAsyncListener
   - Purpose: This class listens for `RebuildIndexAsyncEvent` and triggers the process of rebuilding the article index. It retrieves all articles from the database and re-indexes them to improve search efficiency.
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
log	private static final	Logger	Logger instance for logging rebuild index events.

- Methods:

Method Signature	Access Modifier	Description
void onArticleCreated(RebuildIndexAsyncEvent rebuildIndexAsyncEvent)	public	Listens for RebuildIndexAsyncEvent, fetches all articles, and triggers re-indexing.

3. Relationships:
   - Inheritance: NA
   - Associations
     - Aggregation :
       - ArticleDao
     - Dependency:
       1. RebuildIndexAsyncEvent - Listens for the event and processes it when triggered.
       2. TransactionUtil - Manages transactions while fetching articles and rebuilding indexes.

▼ public class SortCriteria

1. Class Information
   - Class Name: SortCriteria
   - Purpose: Defines sorting criteria for database queries
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
column	private	int	Index of the column to be sorted
asc	private	boolean	Defines whether sorting is ascending (true) or descending (false).
sortQuery	private	String	Optional custom query string for sorting.

   - Methods:

Method Signature	Access Modifier	Description
SortCriteria(String sortQuery)	public	Constructor that initializes sorting using custom query string
SortCriteria(Integer column, Boolean asc)	public	Constructor that initializes sorting based on column index and order.
String getSortQuery()	public	Returns the custom sorting query string.

int getColumn()	public	Returns the column index to sort by.
boolean isAsc()	public	Returns true if sorting is ascending, otherwise false

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations : NA
     - Dependency:
     - Aggregation:
     - Composition:

▼ public class QueryParam

1. Class Information
   - Class Name: QueryParam
   - Purpose:
     - Encapsulates parameters required for executing database queries.
     - Supports sorting, filtering, and grouping criteria for constructing complex queries.
     - Allows mapping query results using a ResultMapper.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
queryString	private	String	SQL or JPQL query string.
criteriaList	private	List<String>	Stores additional **query conditions**.
parameterMap	private	Map<String, Object>	Maps **query parameters** to their values.
sortCriteria	private	SortCriteria	Defines sorting rules for the query.
filterCriteria	private	FilterCriteria	Defines filtering rules for the query.
groupByList	private	List<String>	Stores **grouping criteria** for the query.
resultMapper	private	ResultMapper	Maps **query results** for native queries.

   - Methods:

Method Signature	Access Modifier	Description

`QueryParam(String queryString, List<String> criteriaList, Map<String, Object> parameterMap, SortCriteria sortCriteria, FilterCriteria filterCriteria, List<String> groupByList, ResultMapper resultMapper)`	public	Constructor to initialize a query with full parameter options.
`QueryParam(String queryString, List<String> criteriaList, Map<String, Object> parameterMap, SortCriteria sortCriteria, FilterCriteria filterCriteria, ResultMapper resultMapper)`	public	Constructor for queries without groupBy options.
`String getQueryString()`	public	Returns the query string.
`void setQueryString(String queryString)`	public	Sets the query string.
`SortCriteria getSortCriteria()`	public	Retrieves sorting criteria.
`void setSortCriteria(SortCriteria sortCriteria)`	public	Sets sorting criteria.
`List<String> getCriteriaList()`	public	Retrieves filtering criteria.
`Map<String, Object> getParameterMap()`	public	Retrieves query parameters.
`FilterCriteria getFilterCriteria()`	public	Retrieves filter criteria.
`List<String> getGroupByList()`	public	Retrieves group-by columns.
`ResultMapper getResultMapper()`	public	Retrieves the result mapper for native queries.

3. Relationships:
    - Inheritance: NA
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - SortCriteria
            - FilterCriteria
            - Mapper
        - Aggregation:
        - Composition:

▼ public class FilterCriteria

1. Class Information
    - Class Name: FilterCriteria
    - Purpose:

- Encapsulates a list of FilterColumn instances, representing filtering conditions for a database query.
- Provides a structured approach to apply multiple filtering conditions dynamically.

2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
filterColumnList	private	List<FilterColumn>	A list of filtering conditions to be applied to a query.

   - Methods:

Method Signature	Access Modifier	Description
FilterCriteria(List<FilterColumn> filterColumnList)	public	Initializes the filtering criteria with a given list of filters.
List<FilterColumn> getFilterColumnList()	public	Returns the list of filter conditions.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
     - Aggregation:
       - FilterColumn: Contains FilterColumn objects representing filtering conditions
     - Composition:

▼ public class FilterColumn

1. Class Information
   - Class Name: FilterColumn
   - Purpose:
     - Represents a filter condition applied to a specific database column.
     - Supports dynamic query filtering by generating predicates and parameters.
     - Designed as an **abstract class**, requiring subclasses to define how filtering is applied.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
column	protected	String	The name of the column to be filtered.
filter	protected	String	The filter expression or value to apply.

- Methods:

Method Signature	Access Modifier	Description
FilterColumn(String column, String filter)	public	Initializes the column name and filter expression.
String getColumn()	public	Returns the column being filtered.
String getFilter()	public	Returns the filter expression.
abstract String getPredicate()	public abstract	Subclasses must implement this to generate the SQL predicate for the filter.
abstract Object getParamValue()	public abstract	Subclasses must implement this to return the value associated with the filter parameter.
String getParamName()	public	Generates a unique parameter name based on the column name.
boolean hasParam()	public	Returns true if the column name is not null, indicating a valid filter.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
     - Aggregation:
     - Composition:

▼ public class SearchResource

1. Class Information

- **Class Name**: SearchResource
- **Purpose**: Handles search requests for articles in the reader application

2. Attributes and Methods

Attributes

None explicitly defined

Methods

Method	Access Modifier	Description
`get(String query, Integer limit, Integer offset)`	Public	- Handles GET requests to search articles - Takes query parameters for pagination and search terms - Returns JSON response with search results

3. Relationships

Inheritance

- **Parent Class**: BaseResource
- **Child Classes**: None

Associations

Dependencies

- IndexingService
- ArticleAssembler
- ValidationUtil

Aggregation :

IndexingService

PaginatedList<UserArticleDto>

▼ public class PaginatedLists

1. Class Information
   - Class Name: PaginatedLists
   - Purpose: A utility class for handling paginated database queries using JPA. It constructs paginated lists, executes queries efficiently, and manages sorting, filtering, and counting of results.
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
DEFAULT_PAGE_SIZE	private	int	Default size of a page
MAX_PAGE_SIZE	private	int	Maximum size of a page

   - Methods:

Method Signature	Access Modifier	Description
public static <E> PaginatedList<E> create(Integer pageSize, Integer offset)	public	Creates a paginated list with the specified page size and offset.

public static <E> PaginatedList<E> create()	public	Creates a paginated list with default page size and offset.
public static <E> List<E> executeQuery(QueryParam queryParam)	public	Executes a non-paginated query and returns the result list.
public static <E> void executePaginatedQuery(PaginatedList<E> paginatedList, QueryParam queryParam, SortCriteria sortCriteria)	public	Executes a paginated query by counting and retrieving results.
private static <E> void executeCountQuery(PaginatedList<E> paginatedList, QueryParam queryParam)	private	Counts the number of records matching the query
private static <E> void executeResultQuery(PaginatedList<E> paginatedList, QueryParam queryParam)	private	Executes the paginated query and retrieves results
private static String getQueryString(QueryParam queryParam)	private	Constructs the SQL query string with filtering(WHERE) and grouping(GROUP BY)
private static String getNativeCountQuery(QueryParam queryParam)	private	Creates a native SQL query to count the number of records.
private static String getOrderByClause(SortCriteria sortCriteria)	private	Generates an ORDER BY clause for sorting results based on SortCriteria
private static void mapQueryParam(Query query, QueryParam queryParam)	private	Maps query parameters for filtering
private static void mapFilterColumn(Query query, QueryParam queryParam)	private	Maps filter criteria to query parameters.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - PaginatedList<E>: Uses it to store paginated results
            - QueryParam: Holds query details such as filtering, sorting, and parameters.
            - EntityManager: Executes database queries.
            - SortCriteria: Defines sorting order of query results.
            - ThreadLocalContext: Provides a thread-local EntityManager instance.

        - FilterColumn: Handles query filtering conditions.
    - Aggregation:
    - Composition:

▼ public class PaginatedList<T>

1. Class Information
    - Class Name: PaginatedList
    - Purpose: The class provides a structured way to handle pagination for lists of feed articles or other records within the system.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
limit	private	int	Maximum number of records per page.
offset	private	int	Starting point of the current page in the dataset.
resultCount	private	int	Total number of records available.
resultList	private	List<T>	The list of records currently being displayed.

    - Methods:

Method Signature	Access Modifier	Description
PaginatedList(int pageSize, int offset)	public	Constructor that initializes the pagination settings.
int getResultCount()	public	Returns the total number of records.
void setResultCount(int resultCount)	public	Sets the total number of records.
List<T> getResultList()	public	Returns the list of records on the current page.
void setResultList(List<T> resultList)	public	Sets the list of records for the current page.
int getLimit()	public	Returns the maximum number of records per page.
int getOffset()	public	Returns the starting offset of the current page.

3. Relationships:
    - Inheritance: NA
        - Parent Class:

    - Child Classes:
- Associations : NA
    - Dependency:
    - Aggregation:
    - Composition:

▼ public class LuceneUtil

1. Class Information
    - Class Name: LuceneUtil
    - Purpose:
        - Provides a utility method for safely executing Lucene indexing operations.
        - Ensures that index writers are properly created, used, and closed, reducing the risk of locked indexes or incomplete transactions.
        - Supports custom indexing logic via the LuceneRunnable interface.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
log	private static	Logger	Logger for logging errors related to Lucene indexing.

    - Methods:

Method Signature	Access Modifier	Description
void handle(LuceneRunnable runnable)	public static	Encapsulates a Lucene indexing operation, ensuring proper error handling and transaction rollback.

    - Nested Interface:

Interface	Description
LuceneRunnable	Defines a functional interface for executing custom indexing logic within the Lucene context.

    - Interface Method:

Method Signature	Access Modifier	Description
void run(IndexWriter indexWriter)	public abstract	Defines custom indexing logic that runs within a Lucene transaction.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - LuceneRunnable: Allows passing custom indexing logic to the handle method.
            - ReaderStandardAnalyzer
            - IndexWriterConfig
            - SerialMergeScheduler
            - Directory
            - IndexWriter
        - Aggregation:
        - Composition:
            - Logger

▼

▼ public class LogCriteria

1. Class Information
    - Class Name: LogCriteria
    - Purpose: The `LogCriteria` class provides a structured way to filter logs based on level, tag, and message. It allows for setting and retrieving these parameters in a case-insensitive manner.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`level`	private	String	Represents the logging level (e.g., DEBUG, WARN, ERROR).
`tag`	private	String	Represents the logger name or tag used for logging.
`message`	private	String	Represents the message content of the log entry.

    - Methods:

Method Signature	Access Modifier	Description
`public String getLevel()`	public	Retrieves the logging level.

`public LogCriteria setLevel(String level)`	public	Sets the logging level (converted to lowercase).
`public String getTag()`	public	Retrieves the logger tag.
`public LogCriteria setTag(String tag)`	public	Sets the logger tag (converted to lowercase).
`public String getMessage()`	public	Retrieves the log message content.
`public LogCriteria setMessage(String message)`	public	Sets the log message content (converted to lowercase).

3. Relationships:
    - Inheritance: NA
    - Associations : NA

▼ public class LogEntry

1. Class Information
    - Class Name: LogEntry
    - Purpose: The `LogEntry` class is a simple data structure designed to store details of individual log entries, including the timestamp, logging level, tag, and log message. This class is useful for managing and processing log records in a structured way.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`timestamp`	private	`long`	The time at which the log entry was created.
`level`	private	`String`	The severity level of the log (e.g., DEBUG, WARN).
`tag`	private	`String`	The identifier or tag of the logger.
`message`	private	`String`	The actual log message content.

    - Methods:

Method Signature	Access Modifier	Description
`LogEntry(long timestamp, String level, String tag, String message)`	public	Constructor that initializes the log entry with a timestamp, level, tag, and message.
`long getTimestamp()`	public	Returns the timestamp of the log entry.
`String getLevel()`	public	Retrieves the logging level of the entry.
`String getTag()`	public	Gets the logger name or tag.
`String getMessage()`	public	Retrieves the log message content.

3. Relationships:

- Inheritance: NA
  - Parent Class:
  - Child Classes:
- Associations : NA

  This class may be used by a logging system that collects multiple `LogEntry` objects for logging purposes.

  If part of a logging framework, multiple `LogEntry` instances might be aggregated in a collection such as a `List<LogEntry>` in a logger class.

▼ public class MemoryAppender

1. Class Information
   - Class Name:
   - Purpose: The `MemoryAppender` class extends `AppenderSkeleton` from Log4J and serves as an in-memory log appender. It maintains a queue of log entries up to a specified size limit and provides methods for retrieving and filtering logs based on criteria.
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
`size`	private	`int`	Maximum size of the queue storing log events.
`logQueue`	private final	`Queue<LogEntry>`	A thread-safe queue (`ConcurrentLinkedQueue`) holding log entries.

   - Methods:

Method Signature	Access Modifier	Description
`boolean requiresLayout()`	public	Returns `false` since this appender does not require a layout.
`void close()`	public synchronized	Closes the appender, preventing further log entries from being added.
`void append(LoggingEvent event)`	public synchronized	Appends a new log entry to the queue, removing the oldest entry if the size exceeds the limit. Logs a warning if the appender is closed.
`private String getLoggerName(LoggingEvent event)`	private	Extracts and returns the class name of the logger, excluding the package name.
`Queue<LogEntry> getLogList()`	public	Returns the queue containing log entries.
`void setSize(int size)`	public	Sets the maximum allowed size of the log queue.

`void find(LogCriteria criteria, PaginatedList<LogEntry> list)`	public	Filters logs based on search criteria (`level` , `tag` , and `message`) and stores the results in a `PaginatedList` .

3. Relationships:
    - Inheritance:
        - Parent Class: AppenderSkeleton (from Log4j)
    - Associations
        - Dependency:
            1. LoggingEvent - Used to extract log details (level, message, logger name) when appending logs.
            2. LogCriteria - Used to filter logs in the `find()` method.
            3. PaginatedList<LogEntry> - Stores the filtered list of log entries.
            4. LogLog
        - Composition: LogEntry - Represents individual log entries stored within `logQueue` .

▼ public class DeadEventListener

1. Class Information
    - Class Name: DeadEventListener
    - Purpose: The purpose of this class is to listen for and log any events that have no registered subscribers within the system. It acts as a catchall mechanism to track unexpected or unhandled events.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
log	private static final	Logger	Logger instance for logging unprocessed events.

    - Methods:

Method Signature	Access Modifier	Description
void onDeadEvent(DeadEvent deadEvent)	public	Listens for dead events and logs them as errors.

3. Relationships:
    - Inheritance: NA
    - Associations
        - Composition:
            - Logger

    - Dependency:
        - DeadEvent
        - Subscribe
        - LoggerFactory

▼

▼ public class HttpUtil

1. Class Information
    - Class Name: HttpUtil
    - Purpose: The `HttpUtil` class is a helper class that facilitates HTTP communication by providing methods to read data from a URL and send data to a URL using POST requests.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`log`	`private static final`	`Logger`	Logger instance for logging errors and messages.

    - Methods:

Method Signature	Access Modifier	Description
`String readUrlIntoString(URL url)`	`public static`	Reads content from the given URL and returns it as a `String`. Returns `null` in case of failure.
`String postUrl(URL url, String data)`	`public static`	Sends a POST request to the given URL with the provided data and returns the response as a `String`. Throws `IOException` in case of failure.

3. Relationships:
    - Inheritance: NA
    - Associations
        - Dependency :
            - `URL`, `URLConnection`
        - Aggregation :
            - Logger

▼ public class UrlUtil

1. Class Information

- Class Name: UrlUtil
- Purpose: The `UrlUtil` class offers static utility methods for handling URLs in the context of articles and feeds. It ensures URLs are correctly formatted, extracts base URLs, and constructs complete URLs from relative paths.

2. Attributes and Methods
   - Attributes: NA
   - Methods:

Method Signature	Access Modifier	Description
`String getBaseUri(Feed feed, Article article)`	public static	Retrieves the base URI for an article based on its associated feed or a derived URL.
`String getBaseUri(String urlString)`	public static	Extracts the base URL from a given full URL string, removing paths beyond the root domain.
`String completeUrl(String baseUrl, String relativeUrl)`	public static	Completes and validates relative URLs by merging them with a base URL.

3. Relationships:
   - Inheritance: NA
   - Associations
     - Aggregation :
       - Feed
       - Article

▼

▼ public class Helper

1. Class Information
   - Class Name: `Helper`
   - Purpose: The `Helper` class facilitates JavaScript execution, logging, file existence checks, and scheduled task execution within the ad-blocking system.
2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
`log`	private static	`Logger`	Logger instance for debugging logs.
`js`	private	`JSEngine`	JavaScript execution engi

   - Methods:

Method Signature	Access Modifier	Description
`Helper(JSEngine js)`	public	Constructor initializing the JavaScript engine.
`void log(Object msg)`	public	Logs the given message at the debug level.
`void load(String path) throws Exception`	public	Loads and evaluates a JavaScript file from the resources.
`boolean fileExists(String stringPath)`	public	Checks whether a file exists at the given path.
`boolean canAutoupdate()`	public	Always returns `true`, indicating auto-update capability.
`TimerTask timerTask(Runnable runnable)`	public	Creates and returns a `TimerTask` for scheduling tasks.

3. Relationships:
    - Inheritance: NA
    - Associations
        - Composition :
            - JSEngine
        - Dependency:
            - `Logger`: Utilized for debugging and logging.
            - `Resources` : Fetches JavaScript resources from the application.
            - `Files,Path and` `Paths` : Used to check file existence.
            - `TimerTask` : Creates tasks for scheduled execution.
            - LoggerFactory

▼ public class JSEngine

1. Class Information
    - Class Name: JSEngine
    - Purpose: The `JSEngine` class is a utility class that initializes and manages a JavaScript execution engine (`ScriptEngine`). It allows for evaluating JavaScript code dynamically and passing key-value pairs to the script engine.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`engine`	private	`ScriptEngine`	Instance of Java's `ScriptEngine` to evaluate JavaScript code.

    - Methods:

Method Signature	Access Modifier	Description
`JSEngine()`	public	Constructor that initializes the JavaScript engine using `ScriptEngineManager`.
`Object evaluate(String script) throws ScriptException`	public	Evaluates the given JavaScript code and returns the result.
`void put(String key, Object value)`	public	Stores a key-value pair in the script engine's global context.

3. Relationships: NA
    - Associations
        - Dependency: The class depends on `ScriptEngine` and `ScriptEngineManager` for JavaScript execution.
        - Composition: `JSEngine` contains an instance of `ScriptEngine`, meaning it cannot function without this component.

▼ public abstract class DbOpenHelper

1. Class Information
    - Class Name: DbOpenHelper
    - Purpose:
        - Manages database versioning and schema updates.
        - Executes SQL scripts to migrate the database from an older version to a new one.
        - Handles database connection management through Hibernate.
        - Logs database operations and errors for debugging.
2. Attributes and Methods
    - Attributes:

Attribute Name	Access Modifier	Type	Description
log	private static	Logger	Logger for database update operations.
connectionHelper	private final	ConnectionHelper	Manages database connections for executing updates.
sqlStatementLogger	private final	SqlStatementLogger	Logs SQL statements being executed.
exceptions	private final	List<Exception>	Stores exceptions encountered during database updates.
formatter	private	Formatter	Formats SQL queries for logging.
haltOnError	private	boolean	Determines if execution should stop on SQL errors.

stmt	private	Statement	SQL statement executor.

- Methods:

Method Signature	Access Modifier	Return Type	Description
DbOpenHelper(ServiceRegistry serviceRegistry)	public	DbOpenHelper	Constructor that initializes database connection and logging services.
void open()	public	void	Opens a database connection and applies incremental updates.
void executeAllScript(int version)	protected	void	Executes all SQL scripts for a given version.
void executeScript(InputStream inputScript)	private	void	Reads and executes an SQL script.
void onCreate()	public abstract	void	Abstract method for schema creation (must be implemented by subclasses).
void onUpgrade(int oldVersion, int newVersion)	public abstract	void	Abstract method for database version upgrades (must be implemented by subclasses).
List<?> getExceptions()	public	List<?>	Returns a list of exceptions encountered during execution.
void setHaltOnError(boolean haltOnError)	public	void	Sets whether execution should stop when an error occurs.
void setFormat(boolean format)	public	void	Sets whether SQL queries should be formatted before execution.

3. Relationships:
   - Inheritance: Abstract Class
   - Associations
     - Dependency:
       - DialectUtil: Handles dialect-specific SQL transformations.
       - ConfigUtil: Fetches database version from configuration files.
       - ResourceUtil: Loads SQL scripts from resources.

        - EnvironmentUtil: Determines the execution environment (web, test, etc.).
        - ServiceRegistry
        - JdbcServices
    - Aggregation:’
        - Formatter
        - SqlStatementLogger
    - Composition:
        - Logger

▼ public abstract class ReaderHttpClient<T>

1. Class Information
    - Class Name: ReaderHttpClient
    - Purpose: Abstract class for establishing an http connection and processing data streams from urls.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
USER_AGENT	private	String	Default agent for http requests
sslSocketFactory	private	SSLSocketFactory	SSL factory for secure connections
timeout	private	int	Timeout for connections

    - Methods:

Method Signature	Access Modifier	Description
T open(URL url)	public	Opens an HTTP(S) connection to a given URL and processes data.
HttpURLConnection buildHttpConnection(URL url)	private	Builds and configures HTTP connection
abstract T process(InputStream is)	public	Abstract method for processing response stream
void setTimeout(int timeout)	public	Sets connection timeout duration

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency

        - `EnvironmentUtil`
        - `CertUtil`
    - **Aggregation**
        - `HttpURLConnection`
        - `HttpsURLConnection`
        - `URL`

▼ public final class ReaderStandardAnalyzer

1. Class Information
    - Class Name: ReaderStandardAnalyzer
    - Purpose: The `ReaderStandardAnalyzer` class extends `StopwordAnalyzerBase` and serves as a Lucene-based text analyzer for processing RSS feed content. It performs tokenization, applies filters such as stop-word removal, case normalization, and HTML stripping, and ensures optimal text processing for indexing and searching feeds.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
`DEFAULT_MAX_TOKEN_LENGTH`	`public static final`	`int`	Defines the default maximum token length (255).
`maxTokenLength`	`private`	`int`	Stores the maximum allowed token length for analysis.
`STOP_WORDS_SET`	`public static final`	`CharArraySet`	A predefined set of English stop words used for filtering.

    - Methods:

Method Signature	Access Modifier	Description
`ReaderStandardAnalyzer(Version matchVersion, CharArraySet stopWords)`	`public`	Constructor that initializes the analyzer with specified stop words.
`ReaderStandardAnalyzer(Version matchVersion)`	`public`	Constructor that initializes the analyzer with default stop words.
`ReaderStandardAnalyzer(Version matchVersion, Reader stopwords)`	`public`	Constructor that loads stop words from a `Reader` input.
`setMaxTokenLength(int length)`	`public`	Sets the maximum allowed token length for text processing.

`getMaxTokenLength()`	`public`	Returns the current maximum token length.
`createComponents(String fieldName, Reader reader)`	`protected`	Creates token stream components, applying filters such as tokenization, stop-word removal, and case normalization.
`initReader(String fieldName, Reader reader)`	`protected`	Initializes the reader with an additional HTML stripping filter for `title` and `description` fields.

3. Relationships:
    - Inheritance:
        - Parent Class: `StopwordAnalyzerBase` (from Apache Lucene)
    - Associations
        - Dependency :
            - Reader
        - Aggregation:
            1. TokenStreamComponents - Aggregates different filters and tokenizers to process text effectively.
            2. StandardTokenizer
            3. StandardFilter
            4. LowerCaseFilter
            5. StopFilter
            6. HTMLStripCharFilter
        - Composition:
            1. CharArraySet

▼ public class AdblockUtil

▼

1. Class Information
    - Class Name: AdblockUtil
    - Purpose: This class provides utilities for handling Adblock subscriptions, evaluating JavaScript-based filtering rules, and managing subscription updates and filtering operations.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description

`log`	`private static final`	`Logger`	Logger instance for logging messages and errors.
`subscriptions`	`private`	`List<Subscription>`	List of known adblock filter subscriptions.
`js`	`private`	`JSEngine`	JavaScript engine to execute Adblock-related scripts.
`interactive`	`private`	`boolean`	Flag indicating if the system is in interactive mode.

- Methods:

Method Signature	Access Modifier	Description
`void start() throws Exception`	`public`	Initializes the JavaScript engine and sets up the environment.
`List<Subscription> getSubscriptions()`	`public`	Retrieves the list of available Adblock subscriptions.
`Subscription getSubscription(String url)`	`public`	Returns a subscription based on its URL.
`void setSubscription(Subscription subscription) throws Exception`	`public`	Adds a new subscription and removes previous ones.
`void refreshSubscription() throws ScriptException`	`public`	Refreshes active subscriptions.
`Subscription offerSubscription()`	`public`	Determines the best subscription to offer based on locale.
`boolean verifySubscriptions() throws ScriptException`	`public`	Checks if at least one subscription is active and valid.
`Boolean matches(String url, String query, String reqHost, String refHost, String accept) throws Exception`	`public`	Evaluates whether a request matches any filter rule.
`void startInteractive() throws ScriptException`	`public`	Notifies the JS engine to enter interactive mode.
`void stopInteractive() throws ScriptException`	`public`	Notifies the JS engine to exit interactive mode.
`String checkLocalePrefixMatch(String[] prefixes)`	`public`	Checks if a subscription prefix matches the user's locale.

3. Relationships:
   - Inheritance: NA
   - Associations
     1. Logger - Utilized for logging errors and events
        - Aggregation:

1. Subscription - The subscriptions list stores multiple Subscription objects.
2. SubscriptionParser

- Composition:
  1. JSEngine - The AdblockUtil class directly instantiates and manages a JSEngine object.
  2. Helper

▼ public class TextPlainMessageBodyWriter

1. Class Information

- **Class Name**: TextPlainMessageBodyWriter
- **Purpose**: Handles conversion of JSONObject to text/plain response format

2. Attributes and Methods

Attributes

None explicitly defined

Methods

Method	Access Modifier	Description
`isWriteable(Class<?> type, Type genericType, Annotation[] annotations, MediaType mediaType)`	Public	Checks if the writer can handle the type
`getSize(JSONObject array, Class<?> type, Type genericType, Annotation[] annotations, MediaType mediaType)`	Public	Returns the size of the content
`writeTo(JSONObject jsonObject, ...)`	Public	Writes the JSON object to output stream

3. Relationships

Inheritance

- **Parent Class**: None
- **Child Classes**: None
- **Implements**: MessageBodyWriter<JSONObject>

Associations

Dependencies

- JSONObject
- JSONException
- Various Jersey utilities

Aggregation :

ReaderWriter

▼ public class ArticleAssembler

1. Class Information
   - Class Name: ArticleAssembler
   - Purpose: The primary purpose of this class is to assemble JSON representations of `UserArticleDto` objects. It transforms article data into a structured JSON format, making it easier for the frontend or API consumers to process and display article information.
2. Attributes and Methods
   - Attributes: NA
   - Methods:

Method Signature	Access Modifier	Description
`asJson(UserArticleDto userArticle) throws JSONException`	`public static`	Converts a `UserArticleDto` object into a `JSONObject`, preserving all relevant article metadata.

3. Relationships:
   - Inheritance: NA
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - UserArticleDto
       - JSONObject

▼ public abstract class ResutMapper<T>

1. Class Information
   - Class Name: ResultMapper<T>
   - Purpose: This abstract class serves as a base class for mapping a result set to a DTO. It provides generic mapping capabilities, converting SQL query results into Java objects.
2. Attributes and Methods
   - Attributes: NA
   - Methods:

Method Signature	Access Modifier	Description

`abstract T map(Object[] cols)`	`public abstract`	Abstract method to be implemented by subclasses for mapping a single row to an object of type `T`.
`List<T> map(List<Object[]> rows)`	`public`	Iterates over rows and maps each one using the `map` method.
`Character characterValue(Object o)`	`protected`	Returns the `Character` value of an object if applicable.
`String characterValueAsString(Object o)`	`protected`	Converts a `Character` value to a `String`.
`String stringValue(Object o)`	`protected`	Converts an object to a `String`, handling special cases like `Clob`.
`Boolean booleanValue(Object o)`	`protected`	Converts various object types (e.g., `String`, `Boolean`, `Integer`) to a `Boolean` value.
`Integer intValue(Object o)`	`protected`	Converts an object to an `Integer`, handling cases like `BigInteger` and `String`.
`Long longValue(Object o)`	`protected`	Converts an object to a `Long`, handling cases like `BigInteger` and `BigDecimal`.
`Float floatValue(Object o)`	`protected`	Converts an object to a `Float`, handling `BigDecimal` values.
`Double doubleValue(Object o)`	`protected`	Converts an object to a `Double`, handling `BigDecimal` values.
`BigDecimal bigDecimalValue(Object o)`	`protected`	Converts an object to a `BigDecimal`.
`Date dateValue(Object o)`	`protected`	Returns the `Date` value of an object.
`String getString(Clob clob)`	`protected`	Extracts a `String` from a `Clob` object, handling exceptions.
`String arrayValue(Object o)`	`protected`	Returns a `String` representation of an object, defaulting to an empty string if null.

3. Relationships:
    - Inheritance: NA
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency: Proxy, SerializableClobProxy, IOUtils
        - Aggregation:
        - Composition:

▼ public abstract class BaseDao

1. Class Information
    - Class Name: BaseDao
    - Purpose:

- Provides generic data access operations for entities.
- Supports filtering, sorting, and pagination.
- Encapsulates query execution logic.

2. Attributes and Methods
   - Attributes:
   - Methods:

Method Signature	Access Modifier	Description
void findByCriteria(PaginatedList<T> list, C criteria, SortCriteria sortCriteria, FilterCriteria filterCriteria)	public	Performs a paginated search based on given criteria, sorting, and filtering. Updates the provided paginated list.
List<T> findByCriteria(C criteria, SortCriteria sortCriteria, FilterCriteria filterCriteria)	public	Returns a list of items matching the criteria, with optional sorting and filtering.
List<T> findByCriteria(C criteria)	public	Returns a list of items matching the criteria without sorting or filtering.
T findFirstByCriteria(C criteria)	public	Returns the first matching result from the list of items matching the criteria.
QueryParam getQueryParam(C criteria, FilterCriteria filterCriteria)	protected abstract	Abstract method to be implemented by subclasses for constructing query parameters.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - PaginatedList: Used as a parameter to store paginated query results.
       - PaginatedLists: Used to execute paginated queries.
       - SortCriteria: Defines sorting rules for queries.
       - FilterCriteria: Defines filtering rules for queries.
       - QueryParam: Represents query parameters used in data retrieval.

▼

▼ public class ValidationUtil

1. Class Information
    - Class Name: ValidationUtil
    - Purpose: Provides validation methods for various types of input parameters like strings, emails, URLs, etc.
2. Attributes and Methods
    - Attributes:
        - EMAIL_PATTERN (private static) Pattern: Regular expression for email validation
        - HTTP_URL_PATTERN (private static) Pattern: Regular expression for HTTP URL validation
        - ALPHANUMERIC_PATTERN (private static) Pattern: Regular expression for alphanumeric validation
    - Methods:
        - validateRequired(Object s, String name): public static void - Validates that an object is not null
        - validateLength(String s, String name, Integer lengthMin, Integer lengthMax, boolean nullable): public static String - Validates string length with nullable option
        - validateLength(String s, String name, Integer lengthMin, Integer lengthMax): public static String - Validates string length
        - validateStringNotBlank(String s, String name): public static String - Validates string is not blank
        - validateEmail(String s, String name): public static void - Validates email format
        - validateHttpUrl(String s, String name): public static String - Validates HTTP URL format
        - validateAlphanumeric(String s, String name): public static void - Validates alphanumeric string
        - validateDate(String s, String name, boolean nullable): public static Date - Validates and parses date
3. Relationships
    - Inheritance: None
    - Associations
        - Dependencies:
            - ClientException
            - JSONException
            - DateTime (Joda Time)
            - Pattern
            - MessageFormat

▼

▼ public class CategoryResource

1. Class Information

- Class Name: CategoryResource
- Purpose: Handles CRUD operations for article categories

2. Attributes and Methods

Methods

Method Signature	Access Modifier	Description
list()	public	Returns all categories
get(String id, boolean unread, Integer limit, String afterArticle)	public	Gets articles in a category
add(String name)	public	Creates new category
delete(String id)	public	Deletes a category
read(String id)	public	Marks all articles in category as read
update(String id, String name, Integer order, Boolean folded)	public	Updates category properties

3. Relationships

Inheritance

- Parent Class: BaseResource
- Child Classes: None

Associations

Dependencies:

▼ public class CategoryDao

1. Class Information
   - Class Name: CategoryDao
   - Purpose: This class handles operations related to category management in the Rudra article reader application. It provides functionality to create, update, delete, and reorder categories while managing user-specific categories for feed organization.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description

em	private	EntityManager	Manages persistence for categories

- Methods:

Method Signature	Access Modifier	Description
String create(Category category)	public	Creates a new category and assigns it a UUID
Category update(Category category)	public	Updates attributes of an existing category
void reorder(Category category, int order)	public	Reorders categories within the same parent category
void delete(String id)	public	Soft deletes a category by setting the delete date
Category getRootCategory(String userId)	public	Retrieves the root category for a specific user
Category getCategory(String id, String userId)	public	Fetches a specific active category by its ID and user
int getCategoryCount(String parentId, String userId)	public	Counts the subcategories under a parent category
List<Category> findAllCategory(String userId)	public	Retrieves all active categories for a specific user
List<Category> findSubCategory(String parentId, String userId)	public	Retrieves subcategories under a specific parent category

3. Relationships

Inheritance

- **Parent Class:** None
- **Child Classes:** None

Associations

- **Dependency:**
  - `EntityManager` (for database interaction)
  - `ThreadLocalContext` (for retrieving `EntityManager`)
  - `Query` (for executing database queries)
  - `Category` (as the data model)
- **Aggregation:** None
- **Composition:** None

▼ public class Category

1. Class Information
   - Class Name: Category

- Purpose:
  - Allows users to organize feeds into hierarchical categories.
  - Provides an ordering mechanism (order field) for UI presentation.
  - Supports soft deletion using the deleteDate field.

2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique category identifier (Primary Key).
userId	private	String	User ID (Each category belongs to a user).
parentId	private	String	Parent category ID (for hierarchical categories).
name	private	String	Category name.
order	private	Integer	Display order of this category.
folded	private	boolean	Indicates whether the category is collapsed in the UI.
createDate	private	Date	Date the category was created.
deleteDate	private	Date	Date the category was deleted (soft delete).

   - Methods:

Method Signature	Access Modifier	Description
public String getId()	public	Returns the category ID.
public void setId(String id)	public	Sets the category ID.
public String getUserId()	public	Returns the associated user ID.
public void setUserId(String userId)	public	Sets the user ID.
public String getParentId()	public	Returns the parent category ID.
public void setParentId(String parentId)	public	Sets the parent category ID.
public String getName()	public	Returns the category name.
public void setName(String name)	public	Sets the category name.
public Integer getOrder()	public	Returns the display order.
public void setOrder(Integer order)	public	Sets the display order.
public boolean isFolded()	public	Returns whether the category is collapsed in UI.
public void setFolded(boolean folded)	public	Sets whether the category is folded.
public Date getCreateDate()	public	Returns the creation date.

public void setCreateDate(Date createDate)	public	Sets the creation date.
public Date getDeleteDate()	public	Returns the deletion date.
public void setDeleteDate(Date deleteDate)	public	Sets the deletion date.
public String toString()	public	Returns a string representation of the category.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
        - Aggregation:
            - Feed
        - Composition:
            - Category: Parent category is in composition with subcategories

▼

▼ public class Config

1. Class Information
    - Class Name: Config
    - Purpose:
        - Allows storing and retrieving configuration settings dynamically.
        - Provides a flexible way to manage system parameters without hardcoding them.
        - Supports typed configuration keys using the ConfigType enum.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
id	private	ConfigType	Unique configuration key (Primary Key, Enum).
value	private	String	Configuration value (string format).

    - Methods:

Method Signature	Access Modifier	Description
ConfigType getId()	public	Returns the configuration key.

void setId(ConfigType id)	public	Sets the configuration key.
String getValue()	public	Returns the configuration value.
void setValue(String value)	public	Sets the configuration value.
String toString()	public	Returns a string representation of the configuration parameter.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - ConfigUtil
            - ConfigDao
        - Aggregation:
            - ConfigType
        - Composition:

▼ public class ConfigDao

1. Class Information
    - Class Name: ConfigDao
    - Purpose: The `ConfigDao` class provides data access functionality to retrieve configuration parameters from the persistent storage by their IDs.
2. Attributes and Methods
    - Attributes:

Attribute Name	Access Modifier	Type	Description
em	private	EntityManager	Manages persistence for configuration entities

    - Methods:

Method Signature	Access Modifier	Description
Config getById(ConfigType id)	public	Retrieves a configuration parameter by its unique identifier

3. Relationships:

- Inheritance:
  - Parent Class: None
  - Child Classes: None
- Associations:
  - Dependency: Uses `EntityManager` to access persistent storage and manage configurations
  - Aggregation: None
  - Composition: None

▼ public class ConfigUtil

1. Class Information

- **Class Name:** `ConfigUtil`
- **Purpose:** Provides methods to retrieve configuration values in different formats (String, Integer, Boolean) from a database and a resource bundle.

2. Attributes and Methods

Attributes

This class does not define any instance attributes.

Methods

Method Signature	Access Modifier	Description
`public static String getConfigStringValue(ConfigType configType)`	`public static`	Retrieves the textual value of a configuration parameter from the database. Throws `IllegalStateException` if the parameter is undefined.
`public static ResourceBundle getConfigBundle()`	`public static`	Returns the configuration resource bundle (`config.properties`).
`public static int getConfigIntegerValue(ConfigType configType)`	`public static`	Retrieves the integer value of a configuration parameter. Throws `IllegalStateException` if the parameter is undefined.
`public static boolean getConfigBooleanValue(ConfigType configType)`	`public static`	Retrieves the boolean value of a configuration parameter. Throws `IllegalStateException` if the parameter is undefined.

3. Relationships

Inheritance

- **Parent Class:** None

- **Child Classes:** None

Associations

- **Dependency:**
  - `ConfigDao` (to retrieve configuration data from the database)
  - `Config` (for configuration parameter values)
  - `ConfigType` (for identifying configuration parameters)
- **Aggregation:** None
- **Composition:** None

▼ public class MessageUtil

1. Class Overview

- **Class Name:** `MessageUtil`
- **Purpose:** Provides utility methods for retrieving localized messages from a resource bundle (`messages.properties`).

---

2. Class Attributes

Attribute	Access Modifier	Type	Description
`locale`	`private static final`	`Locale`	Stores the default system locale for message retrieval.

---

3. Methods

Method Signature	Access Modifier	Return Type	Description
`public static String getMessage(String key, Object... args)`	`public static`	`String`	Retrieves and formats a localized message based on a given key. If the key is missing, it returns `"**key**"`.

---

4. Relationships & Dependencies

- **Depends on:**
  - `ResourceBundle` (for loading messages)
  - `Locale` (for language localization)
  - `MissingResourceException` (for handling missing keys)
  - `MessageFormat` (for message formatting)

▼ public class EnvironmentUtil

1. Class Overview

- **Class Name:** `EnvironmentUtil`
- **Purpose:** Provides utility methods for retrieving system environment properties such as OS type, user directories, application settings, and configuration flags.

2. Class Attributes

Attribute	Access Modifier	Type	Description
`OS`	`private static`	`String`	Stores the lowercase system OS name.
`TEST_ENV`	`private static`	`String`	Retrieves the `test` system property to detect if the app is in a test environment.
`WINDOWS_APPDATA`	`private static`	`String`	Retrieves the Windows `APPDATA` environment variable.
`MAC_OS_USER_HOME`	`private static`	`String`	Retrieves the Mac OS home directory from the `user.home` system property.
`READER_HOME`	`private static`	`String`	Retrieves the home directory for a reader application.
`APPLICATION_LOG_ENABLED`	`private static`	`String`	Retrieves the `application.log.enabled` system property to check if additional logging is enabled.
`SSL_TRUST_ALL`	`private static`	`String`	Retrieves the `ssl.trust.all` system property to check if all SSL certificates should be trusted.
`HIBERNATE_PROPERTIES`	`private static`	`String`	Retrieves the path to external `hibernate.properties`.
`webappContext`	`private static`	`boolean`	Indicates if the application is running in a web context.

3. Methods

Method Signature	Access Modifier	Return Type	Description
`public static boolean isWindows()`	`public static`	`boolean`	Checks if the OS is Windows.
`public static boolean isMacOs()`	`public static`	`boolean`	Checks if the OS is Mac OS.
`public static boolean isUnix()`	`public static`	`boolean`	Checks if the OS is Unix/Linux.
`public static boolean isUnitTest()`	`public static`	`boolean`	Checks if the application is running in a test environment.
`public static String getWindowsAppData()`	`public static`	`String`	Returns the `AppData` directory on Windows.
`public static String getMacOsUserHome()`	`public static`	`String`	Returns the Mac OS home directory.
`public static String getReaderHome()`	`public static`	`String`	Returns the reader application's home directory.

`public static String getHibernateProperties()`	`public static`	`String`	Returns the path of external `hibernate.properties`.
`public static boolean isApplicationLogEnabled()`	`public static`	`boolean`	Checks if additional logging is enabled.
`public static boolean isSslTrustAll()`	`public static`	`boolean`	Checks if all SSL certificates are trusted.
`public static boolean isWebappContext()`	`public static`	`boolean`	Checks if the application is running in a web context.
`public static void setWebappContext(boolean webappContext)`	`public static`	`void`	Sets the `webappContext` flag.

4. Relationships & Dependencies

- None

▼ public class ResourceUtil

1. Class Information

- **Class Name:** `ResourceUtil`
- **Purpose:**
  - Provides methods to list files from both file system directories and JAR archives.
  - Loads properties files from a given URL.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
None (Class contains only static methods)	-	-	-

Methods

Method Signature	Access Modifier	Description
`public static List<String> list(Class<?> clazz, String path, FilenameFilter filter) throws URISyntaxException, IOException`	`public static`	Lists files inside a directory (filesystem or JAR) based on a given filter.
`public static List<String> list(Class<?> clazz, String path) throws URISyntaxException, IOException`	`public static`	Lists files inside a directory (filesystem or JAR) without a filter.
`public static Map<Object, Object> loadPropertiesFromUrl(URL url) throws RuntimeException`	`public static`	Loads a properties file from a specified URL and returns a map of key-value pairs.

3. Relationships

Inheritance

- **Parent Class:** `None` (Directly extends `java.lang.Object`).
- **Child Classes:** `None`.

Associations

Type	Related Class	Description
Dependency	`java.io.File`	Used to represent files and directories on the filesystem.
Dependency	`java.net.URL`	Used to retrieve resources from URLs.
Dependency	`java.util.List`	Used to store the list of file names.
Dependency	`java.util.Properties`	Used to store and retrieve properties from the properties file.
Dependency	`java.util.Set`	Used to store unique file names extracted from JAR files.
Aggregation	`java.util.Enumeration`	Used to iterate over JAR file entries.
Composition	`java.util.JarFile`	Represents a JAR archive from which files are retrieved.

▼ public class DirectoryUtil

1. Class Information

- **Class Name:** `DirectoryUtil`
- **Purpose:**
  - Provides methods to determine and create necessary directories for storing application data.
  - Determines base directory based on environment conditions.
  - Returns specific subdirectories for various application needs (database, logs, favicons, etc.).

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
None (Class contains only static methods)	-	-	-

Methods

Method Signature	Access Modifier	Description
`public static File getBaseDataDirectory()`	`public static`	Returns the base data directory, creating it if necessary.

`public static File getDbDirectory()`	`public static`	Returns the database directory inside the base data directory.
`public static File getFaviconDirectory()`	`public static`	Returns the favicons directory inside the base data directory.
`public static File getLuceneDirectory()`	`public static`	Returns the Lucene indexes directory inside the base data directory.
`public static File getLogDirectory()`	`public static`	Returns the log directory inside the base data directory.
`private static File getDataSubDirectory(String subdirectory)`	`private static`	Returns a specific subdirectory inside the base data directory, creating it if necessary.

3. Relationships

Inheritance

- **Parent Class:** `None` (Directly extends `java.lang.Object`).
- **Child Classes:** `None`.

Associations

Type	Related Class	Description
Dependency	`java.io.File`	Used to represent and create directories.
Dependency	`EnvironmentUtil`	Used to determine the appropriate base directory based on the environment.

▼ public class SessionUtil

1. Class Information

- **Class Name:** `SessionUtil`
- **Purpose:**
  - Provides a method to retrieve the current Hibernate session.
  - Uses `SessionFactory` to manage database interactions.
  - Ensures a single session instance per request.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
None (Class contains only static methods)	-	-	-

Methods

Method Signature	Access Modifier	Description
`private SessionUtil()`	`private`	Private constructor to prevent instantiation.
`public static Session getCurrentSession()`	`public static`	Returns the current Hibernate session from `SessionFactory`.

3. Relationships

Inheritance

- **Parent Class:** `None` .
- **Child Classes:** `None`.

Associations

Type	Related Class	Description
Dependency	`EMF`	Utility class for managing entity manager factory instances.

▼ public class ThreadLocalContext

1. Class Information
   - Class Name: ThreadLocalContext
   - Purpose:
     - Manages transactional contexts on a per-thread basis.
     - Ensures each request has its own database EntityManager.
     - Implements cleanup operations to prevent memory leaks.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
threadLocalContext	public static	ThreadLocal<ThreadLocalContext>	Stores the context per-thread.
entityManager	private	EntityManager	Manages the database transactions for a given thread.

   - Methods:

Method Signature	Access Modifier	Description
ThreadLocalContext get()	public static	Returns the current thread's context. Initializes if not present.
void cleanup()	public static	Removes the context from ThreadLocal storage to prevent memory leaks.
boolean isInTransactionalContext()	public	Checks if the thread is within a database transaction.
EntityManager getEntityManager()	public	Returns the EntityManager instance.
void setEntityManager(EntityManager entityManager)	public	Assigns an EntityManager to the thread context.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - EntityManager: Used to manage database transactions for the request
        - Aggregation:
        - Composition:

▼

▼ public class AppResource

1. Class Information

- Class Name: AppResource
- Purpose: Handles application-level operations and monitoring

2. Attributes and Methods

Attributes

None specific (inherits from BaseResource)

Methods

Method Signature	Access Modifier	Description
version()	public	Returns application version information

log()	public	Retrieves application logs
batchReindex()	public	Rebuilds the search index
mapPort()	public	Maps application port using UPnP

3. Relationships

Inheritance

- Parent Class: BaseResource
- Child Classes: None

Associations

Dependencies:

- ConfigUtil
- MemoryAppender
- NetworkUtil

▼ public class AllResource

1. Class Information

- Class Name: AllResource
- Purpose: Manages REST endpoints for retrieving and managing all articles across the system

2. Attributes and Methods

Attributes

None specific to this class (inherits from BaseResource)

Methods

Method Signature	Access Modifier	Description
get(boolean unread, Integer limit, String afterArticle)	public	Returns all articles with pagination and filtering options
read()	public	Marks all articles as read

3. Relationships

Inheritance

- Parent Class: BaseResource
- Child Classes: None

Associations

Dependencies:

- UserArticleDao
- FeedSubscriptionDao
- ArticleAssembler
- Various DTO and criteria classes

▼

▼ public class ForbiddenClientException

1. Class Information

- **Class Name:** `ForbiddenClientException`
- **Purpose:**
  - This class represents a custom exception thrown when a client tries to access a restricted resource without proper authorization.
  - It extends `WebApplicationException` and automatically returns an HTTP 403 Forbidden response in JSON format.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Used for serialization compatibility.

Methods

Method Signature	Access Modifier	Description
`ForbiddenClientException()`	`public`	Constructor that builds a JSON response with HTTP status 403 (Forbidden).

---

3. Relationships

Inheritance

- **Parent Class:** `WebApplicationException` (from `javax.ws.rs`)
- **Child Classes:** None (Final implementation).

▼ public class ClientException

1. Class Information

- **Class Name:** `ClientException`
- **Purpose:** This class represents a client-side exception, typically used for validation errors or user input errors. It extends `WebApplicationException` and logs the error while also returning a structured JSON response.

---

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Serialization ID for version control
`log`	`private static final`	`Logger`	Logger instance for logging errors

Methods:

Method Signature	Access Modifier	Description
`ClientException(String type, String message, Exception e) throws JSONException`	`public`	Constructs a `ClientException` with an error type, message, and logs the exception.
`ClientException(String type, String message) throws JSONException`	`public`	Constructs a `ClientException` with an error type and message, and returns a JSON response.

---

3. Relationships

Inheritance:

- **Parent Class:** `WebApplicationException`
- **Child Classes:** None

Associations:

- **Dependency:** Uses `Logger` from `org.slf4j.LoggerFactory` for logging.
- **Aggregation:** None
- **Composition:** None

▼ public class ServerException

1. Class Information

- **Class Name:** `ServerException`
- **Purpose:** This class represents a server-side exception, typically used for internal errors such as database failures. It extends `WebApplicationException`, logs the error, and returns a structured JSON response.

---

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Serialization ID for version control
`log`	`private static final`	`Logger`	Logger instance for logging server-side errors

Methods:

Method Signature	Access Modifier	Description
`ServerException(String type, String message, Exception e) throws JSONException`	`public`	Constructs a `ServerException` with an error type, message, logs the exception, and returns a JSON response.
`ServerException(String type, String message) throws JSONException`	`public`	Constructs a `ServerException` with an error type and message and returns a JSON response.

3. Relationships

Inheritance:

- **Parent Class:** `WebApplicationException`
- **Child Classes:** None

Associations:

- **Dependency:** Uses `Logger` from `org.slf4j.LoggerFactory` for logging.
- **Aggregation:** None
- **Composition:** None

▼ public class GenericExceptionMapper

1. Class Information

- **Class Name:** `GenericExceptionMapper`
- **Purpose:** This class serves as a global exception handler that maps all unhandled exceptions to a standard JSON response. It ensures that both `WebApplicationException` and unknown exceptions are appropriately processed and logged.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description

`log`	`private static final`	`Logger`	Logger instance for logging errors

Methods:

Method Signature	Access Modifier	Description
`Response toResponse(Exception e)`	`public`	Overrides the `toResponse` method from `ExceptionMapper`. It handles `WebApplicationException` separately and logs unknown errors, returning a JSON response with an error message.

3. Relationships

Inheritance:

- **Parent Interface:** `ExceptionMapper<Exception>`
- **Implemented Methods:** `toResponse(Exception e)`

Associations:

- **Dependency:**
  - Uses `Logger` from `org.slf4j.LoggerFactory` for logging.
  - Uses `WebApplicationException` from `javax.ws.rs.core` for handling known exceptions.
- **Aggregation:** None
- **Composition:** None

▼

▼ public class ManagedProviderConnectionHelper

1. Class Information

- **Class Name:** `ManagedProviderConnectionHelper`
- **Purpose:** This class implements the `ConnectionHelper` interface to manage database connections using a `StandardServiceRegistryImpl`. It provides functionality to prepare, retrieve, and release database connections in a managed environment.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`cfgProperties`	`private`	`Properties`	Stores configuration properties for the database connection.
`serviceRegistry`	`private`	`StandardServiceRegistryImpl`	Holds the Hibernate service registry for managing database services.

`connection`	`private`	`Connection`	Represents the database connection used by this helper.

Methods:

Method Signature	Access Modifier	Description
`ManagedProviderConnectionHelper(Properties cfgProperties)`	`public`	Constructor that initializes the helper with configuration properties.
`void prepare(boolean needsAutoCommit) throws SQLException`	`public`	Prepares the connection, enabling auto-commit if required.
`static StandardServiceRegistryImpl createServiceRegistry(Properties properties)`	`private static`	Creates and returns a `StandardServiceRegistryImpl` instance using the provided properties.
`Connection getConnection() throws SQLException`	`public`	Returns the current database connection.
`void release() throws SQLException`	`public`	Releases both the connection and the service registry.
`void releaseConnection() throws SQLException`	`private`	Closes the database connection and clears SQL warnings.
`void releaseServiceRegistry()`	`private`	Destroys the service registry.

3. Relationships

Inheritance:

- **Parent Interface:** `ConnectionHelper`

Associations:

- **Dependency:**
  -
- **Aggregation:**
- **Composition:**
  - Composes a `Connection` object, which is managed and released as part of the class lifecycle.

▼ public class SuppliedConnectionProviderConnectionHelper

1. Class Information

- **Class Name:** `SuppliedConnectionProviderConnectionHelper`
- **Purpose:**
  - Implements the `ConnectionHelper` interface.
  - Manages database connections using a supplied `ConnectionProvider`.

- Handles connection lifecycle, including auto-commit toggling and connection release.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`provider`	`private`	`ConnectionProvider`	Provides database connections.
`connection`	`private`	`Connection`	Stores the current database connection.
`toggleAutoCommit`	`private`	`boolean`	Indicates whether auto-commit was toggled for the connection.

Methods:

Method Signature	Access Modifier	Description
`SuppliedConnectionProviderConnectionHelper(ConnectionProvider provider)`	`public`	Constructor that initializes the helper with a connection provider.
`void prepare(boolean needsAutoCommit) throws SQLException`	`public`	Prepares the connection, setting auto-commit if required.
`Connection getConnection() throws SQLException`	`public`	Returns the current database connection.
`void release() throws SQLException`	`public`	Releases the database connection, restoring auto-commit if toggled.

3. Relationships

Inheritance:

- **Parent Interface:** `ConnectionHelper`
- **Implemented Methods:** `prepare()`, `getConnection()`, `release()`

Associations:

- **Dependency:**
  - Uses `SqlExceptionHelper` (user-defined) to handle SQL warnings.

▼ public class EntityManagerUtil

1. Class Information
   - Class Name: EntityManagerUtil

- Purpose: Provides a utility method for flushing the JPA EntityManager, ensuring that all pending database operations are committed immediately.

2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description

   - Methods:

Method Signature	Access Modifier	Description
void flush()	public static	Calls flush() on the current EntityManager to synchronize pending database changes.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - ThreadLocalContext: Uses it to retrieve the current EntityManager instance for the active database transaction.
     - Aggregation:
     - Composition:

▼ public class EMF

1. Class Information
   - Class Name: EMF
   - Purpose:
     - Initializes and manages the Hibernate EntityManagerFactory.
     - Loads Hibernate configuration properties dynamically.
     - Creates or upgrades the database schema using DbOpenHelper.
     - Provides methods to identify the current database driver.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
log	private static final	Logger	Logs messages and errors related to EntityManagerFactory.

properties	private static	Map<Object, Object>	Stores Hibernate configuration properties.
emfInstance	private static	EntityManagerFactory	Singleton instance of EntityManagerFactory.

- Methods:

Method Signature	Access Modifier	Description
Map<Object, Object> getEntityManagerProperties()	private static	Loads Hibernate properties from an external file, packaged resources, or environment variables.
Map<Object, Object> getEntityManagerPropertiesFromEnvironment()	private static	Generates default Hibernate properties for an HSQLDB database.
EMF()	private	Prevents instantiation of EMF (Singleton pattern).
EntityManagerFactory get()	public static	Returns the singleton EntityManagerFactory instance.
boolean isDriverHsql()	public static	Checks if the application is using HSQLDB.
boolean isDriverPostgresql()	public static	Checks if the application is using PostgreSQL.
String getDriver()	public static	Retrieves the current database driver class name from the configuration properties.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - DbOpenHelper: Used to **initialize and upgrade** the database schema.
       - ResourceUtil: Loads Hibernate configuration properties from external sources.
       - EnvironmentUtil: Retrieves environment-specific Hibernate configurations.
       - DirectoryUtil: Provides the directory path for the database files.
       - ServiceRegistry: Used to configure Hibernate services.
     - Aggregation:

- Composition:

▼

▼ public class DeploymentStatus

1. Class Information

- **Class Name:** `DeploymentStatus`
- **Purpose:**
  - Represents the deployment status of a server.
  - Stores information about server state, startup time, URL, memory usage, and deployment errors.

---

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`serverState`	`private final`	`ServerState`	Represents the lifecycle state of the server.
`startTime`	`private final`	`Date`	Stores the startup time of the server.
`url`	`private final`	`String`	The URL where the server is running.
`memoryUsed`	`private final`	`int`	Memory consumption in MB.
`errorMessage`	`private final`	`String`	Error message in case of a deployment failure.

Methods:

Method Signature	Access Modifier	Description
`DeploymentStatus(ServerState serverState, Date startTime, String url, int memoryUsed, String errorMessage)`	`public`	Constructor to initialize the deployment status.
`ServerState getServerState()`	`public`	Returns the current state of the server.
`Date getStartTime()`	`public`	Returns the server startup time.
`String getUrl()`	`public`	Returns the server URL.
`int getMemoryUsed()`	`public`	Returns the memory used by the server.
`String getErrorMessage()`	`public`	Returns the error message if deployment failed.

---

3. Relationships

Inheritance:

- **Parent Class:** None
- **Child Classes:** None

Associations:

- **Composition:**
  - `DeploymentStatus` contains `ServerState`, an internal enum.

▼ public class Constants

Here is the documentation for the `Constants` class in the requested format:

---

1. Class Information

- **Class Name:** `Constants`
- **Purpose:** Provides a central location for storing and managing static constant values used throughout the application.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`REQUEST_CODE_ARTICLES`	`public static final`	`int`	Request code for fetching the articles list.
`REQUEST_CODE_MANAGE_CATEGORIES`	`public static final`	`int`	Request code for managing categories.

Methods

This class does not contain any methods.

---

3. Relationships

- **Inheritance:**
  - **Parent Class:** None
  - **Child Classes:** None
- **Associations:**
  - **Dependency:** Used by various parts of the application where request codes are needed.
  - **Aggregation:** None
  - **Composition:** None

▼ public class DateUtil

1. Class Overview

- **Class Name:** `DateUtil`
- **Purpose:** Provides methods for handling time zones, parsing dates, and converting time zone codes to offsets.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`log`	`private static final`	`Logger`	Logger for logging warnings and errors.
`TIMEZONE_CODE_MAP`	`private static final`	`ImmutableMap<String, String>`	A map of time zone codes and their corresponding offsets.

Methods

Method Name	Access Modifier	Return Type	Parameters	Description
`guessTimezoneOffset`	`public static`	`String`	`String date`	Replaces a time zone code in the given date string with its corresponding offset.
`parseDate`	`public static`	`Date`	`String date, DateTimeFormatter df`	Parses a date string using the provided formatter, attempting to replace time zone codes if necessary.

---

3. Relationships

- **Inheritance:**
  - **Parent Class:** None
  - **Child Classes:** None
- **Associations:**
  - **Dependency:** Uses `Logger`, `ImmutableMap`, `StringUtils`, `MessageFormat`, and `DateTimeFormatter` for various functionalities.

▼

▼ public class Setting

1. Class Information

- **Class Name:** `Setting`
- Purpose :

- Loads configuration settings from a properties file.
- Provides default values for various settings.
- Allows saving updated settings back to the file.
- Handles security-related settings like SSL configuration.

Attributes

2. Attributes and Methods

Attributes :

Attribute Name	Access Modifier	Type	Description	
`READER_AGENT_PROPERTIES_FILE`	`private static final`	`String`	Name of the properties file storing reader configurations.	
`DEFAULT_HOST`	`public static final`	`String`	Default hostname for the server.	
`DEFAULT_PORT`	`public static final`	`int`	Default port for the server.	
`DEFAULT_CONTEXT_PATH`	`public static final`	`String`	Default context path for the server.	
`DEFAULT_AUTO_START`	`public static final`	`boolean`	Determines if the server starts automatically.	
`DEFAULT_SECURE`	`public static final`	`boolean`	Defines whether SSL is enabled by default.	
`host`	`private`	`String`	Stores the hostname.	
`port`	`private`	`int`	Stores the port number.	
`contextPath`	`private`	`String`	Stores the context path for the application.	
`readerHome`	`private`	`String`	Stores the home directory of the reader.	
`autoStart`	`private`	`boolean`	Indicates whether the	

			server starts automatically.	
`secure`	`private`	`boolean`	Indicates whether SSL is enabled.	
`keyStorePath`	`private`	`String`	Path to the SSL key store.	
`keyStorePassword`	`private`	`String`	Password for the SSL key store.	
`keyManagerPassword`	`private`	`String`	Password for the key manager.	

Methods ：

Method Signature	Access Modifier	Description
`public Setting()`	`public`	Constructor that initializes default values and sets `readerHome` based on the operating system.
`public void read()`	`public`	Reads settings from the properties file.
`public void save() throws Exception`	`public`	Saves settings to the properties file.
`public String getHost()`	`public`	Returns the hostname.
`public void setHost(String host)`	`public`	Sets the hostname.
`public int getPort()`	`public`	Returns the port number.
`public void setPort(int port)`	`public`	Sets the port number.
`public String getContextPath()`	`public`	Returns the context path.
`public void setContextPath(String contextPath)`	`public`	Sets the context path.
`public String getReaderHome()`	`public`	Returns the reader home directory.
`public void setReaderHome(String readerHome)`	`public`	Sets the reader home directory.
`public boolean isAutoStart()`	`public`	Returns whether the server auto-starts.
`public void setAutoStart(boolean autoStart)`	`public`	Sets the auto-start flag.
`public boolean isSecure()`	`public`	Returns whether SSL is enabled.
`public void setSecure(boolean secure)`	`public`	Sets SSL mode.
`public String getKeyStorePath()`	`public`	Returns the key store path.
`public void setKeyStorePath(String keyStorePath)`	`public`	Sets the key store path.
`public String getKeyStorePassword()`	`public`	Returns the key store password.
`public void setKeyStorePassword(String keyStorePassword)`	`public`	Sets the key store password.
`public String getKeyManagerPassword()`	`public`	Returns the key manager password.
`public void setKeyManagerPassword(String keyManagerPassword)`	`public`	Sets the key manager password.

Associations

Inheritance

- **Child Classes:** None
- **Parent Class:** None
- **Composition:** None
- **Aggregation:** None
- **Dependency:**
  - `EnvironmentUtil`: Used for determining the OS-specific reader home directory.
  - `Properties`: Used for reading and writing settings to a file.
  - `Closer`: Used for safely handling file input/output operations.
  - `File`, `FileInputStream`, `FileOutputStream`, `InputStream`, `OutputStream`: Used for file handling.

▼ public class SettingPanel

---

1. Class Information

- **Class Name:** `SettingPanel`
- **Purpose:**
  - Provides an interface to modify server-related settings.
  - Reads configuration values and updates UI fields accordingly.
  - Validates and saves user input to a properties file.
  - Handles security-related settings such as keystore management.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Used for serialization.
`INTEGER_FORMAT`	`private static final`	`Format`	Defines numeric formatting for input fields.
`portTextField`	`private`	`JFormattedTextField`	Field for entering the server port number.
`contextPathComboBox`	`private`	`JComboBox<String>`	Dropdown menu for selecting the server context path.
`autoStartCheckBox`	`private`	`JCheckBox`	Checkbox to enable/disable auto-start functionality.

`secureCheckBox`	`private`	`JCheckBox`	Checkbox for enabling SSL security.
`keyStorePathTextField`	`private`	`JFormattedTextField`	Input field for keystore path.
`keyStorePasswordTextField`	`private`	`JFormattedTextField`	Input field for keystore password.
`keyManagerPasswordTextField`	`private`	`JFormattedTextField`	Input field for key manager password.
`defaultButton`	`private`	`JButton`	Button to reset settings to default values.
`saveButton`	`private`	`JButton`	Button to save settings to the properties file.
`readerAgent`	`private`	`ReaderAgent`	Reference to the reader agent that provides settings data.

Methods

Method Signature	Access Modifier	Description
`public SettingPanel(ReaderAgent readerAgent)`	`public`	Initializes the settings panel with `readerAgent` data.
`public void readSetting()`	`public`	Loads configuration values into UI components.
`public void saveSetting() throws Exception`	`public`	Validates user input and saves changes to the settings file.
`private void initComponent()`	`private`	Creates and configures UI elements.
`private String getContextPath() throws Exception`	`private`	Retrieves and validates the context path from input.
`private int getPort() throws Exception`	`private`	Retrieves and validates the port number.
`private String getKeyStorePath() throws Exception`	`private`	Retrieves and validates the keystore path.
`private String getKeyStorePassword() throws Exception`	`private`	Retrieves and validates the keystore password.
`private String getKeyManagerPassword() throws Exception`	`private`	Retrieves and validates the key manager password.

3. Relationships

Inheritance

- **Parent Class:** `JPanel`
- **Child Classes:** None

Associations

- **Dependency:**
  - `ReaderAgent`: Used to retrieve and update settings.

- `Setting`: Stores configuration values and handles persistence.
- `MessageUtil`: Provides localized UI messages.

- **Aggregation:**
  - `SettingPanel` consists of multiple UI components (`JTextField`, `JCheckBox`, `JButton`).
- **Composition:**
  - `SettingPanel` has a strong association with `readerAgent`, which provides access to `Setting`.

▼ public class StatusPanel

1. Class Information

- **Class Name:** `StatusPanel`
- **Purpose:**
  - Provides real-time status updates on server deployment.
  - Allows users to start or stop the server through UI controls.
  - Displays memory usage, error messages, and the server URL.
  - Implements `DeploymentStatusListener` to react to server state changes.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`serialVersionUID`	`private static final`	`long`	Used for serialization.
`DATE_FORMAT`	`private static final`	`DateFormat`	Formats timestamps for display.
`readerAgent`	`private final`	`ReaderAgent`	Reference to the agent managing deployments.
`statusTextField`	`private`	`JTextField`	Displays the current deployment status.
`startedTextField`	`private`	`JTextField`	Displays the start time of the server.
`memoryTextField`	`private`	`JTextField`	Displays memory usage information.
`errorTextField`	`private`	`JTextArea`	Displays any error messages from the server.
`startButton`	`private`	`JButton`	Button to start the server.
`stopButton`	`private`	`JButton`	Button to stop the server.
`urlButton`	`private`	`JButton`	Clickable button to open the server URL.

Methods

Method Signature	Access Modifier	Description
`public StatusPanel(ReaderAgent readerAgent)`	`public`	Initializes the panel with `readerAgent` and sets up event listeners.
`private void initComponent()`	`private`	Creates and configures UI elements for displaying status information.

`public void notifyDeploymentStatus(DeploymentStatus status)`	`public`	Updates the UI based on the latest deployment status.

3. Relationships

Inheritance

- **Parent Class:** `JPanel`
- **Child Classes:** None

Associations

- **Implements:**
  - `DeploymentStatusListener`: Allows the panel to react to status changes.
- **Dependency:**
  - `ReaderAgent`: Manages the server and handles actions like start/stop.
  - `DeploymentStatus`: Provides information about the current server state.
  - `MessageUtil`: Retrieves localized messages for UI labels.
  - `ServerState`: Enum defining server states (STOPPED, STARTING, etc.).
- **Aggregation:**
  - `StatusPanel` contains multiple UI elements (`JTextField`, `JTextArea`, `JButton`).

▼

▼ public class Job

1. Class Information
   - Class Name: Job
   - Purpose:
     - Stores details about background jobs running in the system.
     - Tracks job execution status, timing, and ownership.
     - Facilitates job management and logging for system administrators.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the job.
userId	private	String	User who initiated the job (can be null for system jobs).
name	private	String	Name of the job (e.g., "Feed Sync", "Import Feeds").
createDate	private	Date	Timestamp when the job was created.

startDate	private	Date	Timestamp when the job started.
endDate	private	Date	Timestamp when the job ended (null if still running).
deleteDate	private	Date	Timestamp when the job was marked for deletion.

- Methods:

Method Signature	**Access Modifier**	**Description**
String getId()	public	Returns the job ID.
void setId(String id)	public	Sets the job ID.
String getUserId()	public	Returns the user ID associated with the job.
void setUserId(String userId)	public	Sets the user ID for the job.
String getName()	public	Returns the job name.
void setName(String name)	public	Sets the job name.
Date getCreateDate()	public	Returns the creation timestamp.
void setCreateDate(Date createDate)	public	Sets the creation timestamp.
Date getStartDate()	public	Returns the job start time.
void setStartDate(Date startDate)	public	Sets the job start time.
Date getEndDate()	public	Returns the job end time.
void setEndDate(Date endDate)	public	Sets the job end time.
Date getDeleteDate()	public	Returns the job deletion timestamp.
void setDeleteDate(Date deleteDate)	public	Sets the job deletion timestamp.
String toString()	public	Returns a string representation of the entity.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:
       - User
     - Aggregation:
     - Composition:

▼ public class JobDao

1. Class Information

- **Class Name:** JobDao
- **Purpose:** Handles database operations related to job entities, including CRUD operations and query-based retrievals.

2. Methods and Their Details

Method Signature	Access Modifier	Description
`protected QueryParam getQueryParam(JobCriteria criteria, FilterCriteria filterCriteria)`	protected	Generates query parameters for filtering and sorting job records.
`String create(Job job)`	public	Creates a new job record and returns its UUID.
`Job getActiveJob(String id)`	public	Retrieves an active job by its ID. Returns `null` if no result is found.
`void delete(String id)`	public	Marks a job as deleted by setting its deletion date.
`Job update(Job job)`	public	Updates the start and end dates of an existing job.

3. Relationships

- **Associations:**
  - Extends `BaseDao<JobDto, JobCriteria>` for standard data access functions.
  - Utilizes `EntityManager` for persistence operations.
  - Works with `Job`, `JobDto`, and `JobCriteria` classes.
- **Dependency:**
  - Uses `ThreadLocalContext` for managing the current `EntityManager`.

▼ public class JobDto

1. Class Information

- **Class Name:** `JobDto`
- **Purpose:** Represents the job metadata including its timestamps, user association, and lifecycle information.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`id`	`private`	`String`	Unique job ID.
`name`	`private`	`String`	Name of the job.
`userId`	`private`	`String`	Unique ID of the user associated with the job.

`createTimestamp`	`private`	`Long`	Timestamp representing job creation date.
`startTimestamp`	`private`	`Long`	Timestamp for when the job started.
`endTimestamp`	`private`	`Long`	Timestamp for when the job ended.

Methods

Method Signature	Access Modifier	Description
`String getId()`	`public`	Gets the job ID.
`void setId(String id)`	`public`	Sets the job ID.
`String getName()`	`public`	Gets the job name.
`void setName(String name)`	`public`	Sets the job name.
`String getUserId()`	`public`	Gets the associated user ID.
`void setUserId(String userId)`	`public`	Sets the associated user ID.
`Long getCreateTimestamp()`	`public`	Gets the creation timestamp.
`void setCreateTimestamp(Long ts)`	`public`	Sets the creation timestamp.
`Long getStartTimestamp()`	`public`	Gets the job start timestamp.
`void setStartTimestamp(Long ts)`	`public`	Sets the job start timestamp.
`Long getEndTimestamp()`	`public`	Gets the job end timestamp.
`void setEndTimestamp(Long ts)`	`public`	Sets the job end timestamp.

---

3. Relationships

Inheritance

- **Parent Class:** None
- **Child Classes:** None

Associations

- **Dependency:** Potential association with user data or job scheduler subsystems
- **Aggregation:** None
- **Composition:** None

▼ public class JobEvent

1. Class Information
    - Class Name: JobEvent
    - Purpose:
        - Tracks specific events occurring during job execution.
        - Provides historical records of a job's progress, errors, and key actions.
        - Facilitates debugging and analysis of job execution patterns.
2. Attributes and Methods

- Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the job event.
jobId	private	String	Reference to the job this event belongs to.
name	private	String	Descriptive name of the event (e.g., "Job Started", "Step Completed").
value	private	String	Additional event data (e.g., status details, error messages).
createDate	private	Date	Timestamp when the event was recorded.
deleteDate	private	Date	Timestamp when the event was deleted (if applicable).

- Methods:

Method Signature	Access Modifier	Description
String getId()	public	Returns the job event ID.
void setId(String id)	public	Sets the job event ID.
String getJobId()	public	Returns the job ID associated with this event.
void setJobId(String jobId)	public	Sets the job ID for the event.
String getName()	public	Returns the event name.
void setName(String name)	public	Sets the event name.
String getValue()	public	Returns the event value.
void setValue(String value)	public	Sets the event value.
Date getCreateDate()	public	Returns the event creation timestamp.
void setCreateDate(Date createDate)	public	Sets the event creation timestamp.
Date getDeleteDate()	public	Returns the event deletion timestamp.
void setDeleteDate(Date deleteDate)	public	Sets the event deletion timestamp.
String toString()	public	Returns a string representation of the entity.

3. Relationships:
   - Inheritance:
     - Parent Class:
     - Child Classes:
   - Associations
     - Dependency:

        - Job: Each job event is associated with a job
    - Aggregation:
    - Composition:

▼ public class JobEventCriteria

1. Class Information

- **Class Name:** `JobEventCriteria`
- **Purpose:**
  This class serves as a criteria object for filtering job events based on specific parameters such as the job ID.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
jobId	private	String	Identifier for the job associated with an event.

Methods

Method Signature	Access Modifier	Description
String getJobId()	public	Retrieves the job ID.
JobEventCriteria setJobId(String jobId)	public	Sets the job ID and returns the updated instance of `JobEventCriteria`.

3. Relationships

- **Inheritance:**
  - **Parent Class:** None
  - **Child Classes:** None
- **Associations:**
  - **Dependency:** None explicitly defined
  - **Aggregation:** None
  - **Composition:** None

▼ public class JobEventDao

1. Class Information

- **Class Name:** `JobEventDao`

- **Purpose:** Provides data access methods for handling job event records, including creation, deletion, and query-based retrievals.

---

2. Methods and Their Details

Method Signature	Access Modifier	Description
`protected QueryParam getQueryParam(JobEventCriteria criteria, FilterCriteria filterCriteria)`	protected	Generates query parameters for filtering and sorting job event records.
`String create(JobEvent jobEvent)`	public	Creates a new job event and returns its UUID.
`void delete(String id)`	public	Marks a job event as deleted by setting the deletion date.

---

3. Relationships

- **Associations:**
  - Extends `BaseDao<JobEventDto, JobEventCriteria>` for standard data access methods.
  - Utilizes `EntityManager` for persistence operations.
  - Works with `JobEvent`, `JobEventDto`, and `JobEventCriteria` classes.
- **Dependency:**
  - Relies on `ThreadLocalContext` to access the current `EntityManager`.

▼ public class JobEventMapper

1. Class Information

- **Class Name:** `JobEventMapper`
- **Purpose:** Responsible for mapping raw database result sets to `JobEventDto` objects, encapsulating information about job events such as event IDs, names, and values.

2. Attributes and Methods

Attributes:

This class does not have instance attributes besides those inherited from its superclass `ResultMapper`.

Methods:

Method Signature	Access Modifier	Description
`public JobEventDto map(Object[] o)`	`public`	Maps an array of raw database values to a `JobEventDto`.

3. Relationships

Inheritance:

- **Parent Class:** `ResultMapper<JobEventDto>`
- **Child Classes:** None

Associations:

- **Dependency:** Depends on `JobEventDto` for the target mapping type.
- **Aggregation:** Not applicable
- **Composition:** Not applicable

▼ public class JobMapper

1. Class Information

- **Class Name:** `JobMapper`
- **Purpose:** Responsible for mapping raw database result sets to `JobDto` objects, which encapsulate information about jobs such as job IDs, names, timestamps, and associated user information.

2. Attributes and Methods

Attributes:

This class does not have instance attributes besides those inherited from its superclass `ResultMapper`.

Methods:

Method Signature	Access Modifier	Description
`public JobDto map(Object[] o)`	`public`	Maps an array of raw database values to a `JobDto`.

3. Relationships

Inheritance:

- **Parent Class:** `ResultMapper<JobDto>`
- **Child Classes:** None

Associations:

- **Dependency:** Depends on `JobDto` for the target mapping type.
- **Aggregation:** Not applicable
- **Composition:** Not applicable