

---

# **Aries Cloud Agent Python Documentation**

**See Contributors list on [GitHub](#)**

**Jun 30, 2022**



# ARIES CLOUD AGENT PYTHON - MODULES

<b>1</b>	<b>aries_cloudagent</b>	<b>3</b>
1.1	aries_cloudagent package . . . . .	3
1.1.1	Subpackages . . . . .	3
1.1.2	Submodules . . . . .	138
1.1.3	aries_cloudagent.version module . . . . .	138
<b>2</b>	<b>Indices and tables</b>	<b>139</b>
	<b>Python Module Index</b>	<b>141</b>
	<b>Index</b>	<b>145</b>



Hyperledger Aries Cloud Agent Python (ACA-Py) is a foundation for building decentralized identity applications and services running in non-mobile environments.

This is the Read The Docs site for the Hyperledger [Aries Cloud Agent Python](#). This site contains only the ACA-Py docstrings documentation extracted from the Python Code. For other documentation, please consult the links in the Readme for the [ACA-Py GitHub Repo](#).

If you are getting started with verifiable credentials or Aries, we recommend that you start with this [verifiable credentials and agents getting started guide](#).

Want to quick overview of the deployment model for ACA-Py? See [this document](#).

To investigate the code, use search or click the package links in the left menu to drill into the modules, subpackages and submodules that make up ACA-Py.

Developers that are interested in what DIDComm protocols are supported in ACA-Py should take a look at the [protocols](#) package. These should align with the corresponding [aries-rfcs protocols](#). Decorators defined in aries-rfcs and implemented in ACA-Py can be found [here](#). Some general purpose subpackages that might be of interest include [wallet](#) and [storage](#). For those agents playing different roles in a verifiable credential exchange, take a look at the [issuer](#), [holder](#) and [verifier](#) packages.

Please see the [ACA-Py Contribution guidelines](#) for how to contribute to ACA-Py, including for how to submit issues about ACA-Py.



## ARIES\_CLOUDAGENT

### 1.1 aries\_cloudagent package

Aries Cloud Agent.

#### 1.1.1 Subpackages

**aries\_cloudagent.admin package**

**Submodules**

**aries\_cloudagent.admin.base\_server module**

Abstract admin server interface.

```
class aries_cloudagent.admin.base_server.BaseAdminServer
```

Bases: `abc.ABC`

Admin HTTP server class.

```
abstract async start() → None
```

Start the webserver.

**Raises** `AdminSetupError` – If there was an error starting the webserver

```
abstract async stop() → None
```

Stop the webserver.

**aries\_cloudagent.admin.error module**

Admin error classes.

```
exception aries_cloudagent.admin.error.AdminError(*args, error_code: Optional[str] = None,  
                                                    **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base class for Admin-related errors.

```
exception aries_cloudagent.admin.error.AdminSetupError(*args, error_code: Optional[str] = None,  
                                                         **kwargs)
```

Bases: `aries_cloudagent.admin.error.AdminError`

Admin server setup or configuration error.

## aries\_cloudagent.admin.request\_context module

Admin request context class.

A request context provided by the admin server to admin route handlers.

```
class aries_cloudagent.admin.request_context.AdminRequestContext(profile:
    aries_cloudagent.core.profile.Profile,
    *, context: Optional[aries_cloudagent.config.injection_context.Injector]
    = None, settings: Optional[Mapping[str, object]]
    = None)
```

Bases: `object`

Context established by the Conductor and passed into message handlers.

```
inject(base_cls: Type[aries_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] =
    None) → aries_cloudagent.config.base.InjectType
```

Get the provided instance of a given class identifier.

### Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

```
inject_or(base_cls: Type[aries_cloudagent.config.base.InjectType], settings: Optional[Mapping[str,
    object]] = None, default: Optional[aries_cloudagent.config.base.InjectType] = None) →
    Optional[aries_cloudagent.config.base.InjectType]
```

Get the provided instance of a given class identifier or default if not found.

### Parameters

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**property injector:** `aries_cloudagent.config.injector.Injector`

Accessor for the associated *Injector* instance.

**property profile:** `aries_cloudagent.core.profile.Profile`

Accessor for the associated *Profile* instance.

**session()** → `aries_cloudagent.core.profile.ProfileSession`

Start a new interactive session with no transaction support requested.

**property settings:** `aries_cloudagent.config.settings.Settings`

Accessor for the context settings.

```
classmethod test_context(session_inject: Optional[dict] = None, profile:
    Optional[aries_cloudagent.core.profile.Profile] = None) →
    aries_cloudagent.admin.request_context.AdminRequestContext
```

Quickly set up a new admin request context for tests.

**transaction()** → `aries_cloudagent.core.profile.ProfileSession`

Start a new interactive session with commit and rollback support.



If the current backend does not support transactions, then commit and rollback operations of the session will not have any effect.

**update\_settings**(*settings: Mapping[str, object]*)  
Update the current scope with additional settings.

**aries\_cloudagent.admin.server module**

**aries\_cloudagent.askar package**

**Subpackages**

**aries\_cloudagent.askar.didcomm package**

**Submodules**

**aries\_cloudagent.askar.didcomm.v1 module**

**aries\_cloudagent.askar.didcomm.v2 module**

**Submodules**

**aries\_cloudagent.askar.profile module**

**aries\_cloudagent.askar.store module**

Aries-Askar backend store configuration.

```
class aries_cloudagent.askar.store.AskarOpenStore(config:  
                                                    aries_cloudagent.askar.store.AskarStoreConfig,  
                                                    created, store: aries_askar.Store)
```

Bases: `object`

Handle and metadata for an opened Askar store.

**async close()**  
Close previously-opened store, removing it if so configured.

**property name: str**  
Accessor for the store name.

```
class aries_cloudagent.askar.store.AskarStoreConfig(config: Optional[dict] = None)
```

Bases: `object`

A helper class for handling Askar store configuration.

**DEFAULT\_KEY = ''**

**DEFAULT\_KEY\_DERIVATION = 'kdf:argon2i:mod'**

**DEFAULT\_STORAGE\_TYPE = None**

**KEY\_DERIVATION\_ARGON2I\_INT = 'kdf:argon2i:int'**

**KEY\_DERIVATION\_ARGON2I\_MOD = 'kdf:argon2i:mod'**

**KEY\_DERIVATION\_RAW = 'RAW'**

**get\_uri**(*create: bool = False*) → *str*  
Accessor for the storage URI.

**async open\_store**(*provision: bool = False*) → *aries\_cloudagent.askar.store.AskarOpenStore*  
Open a store, removing and/or creating it if so configured.

**Raises**

- **ProfileNotFoundError** – If the store is not found
- **ProfileError** – If there is another aries\_askar error

**async remove\_store**()  
Remove an existing store.

**Raises**

- **ProfileNotFoundError** – If the wallet could not be found
- **ProfileError** – If there was another aries\_askar error

## aries\_cloudagent.cache package

### Submodules

#### aries\_cloudagent.cache.base module

Abstract base classes for cache.

**class** `aries_cloudagent.cache.base.BaseCache`  
Bases: `abc.ABC`

Abstract cache interface.

**acquire**(*key: str*)  
Acquire a lock on a given cache key.

**abstract async clear**(*key: str*)  
Remove an item from the cache, if present.

**Parameters** *key* – the key to remove

**abstract async flush**()  
Remove all items from the cache.

**abstract async get**(*key: str*)  
Get an item from the cache.

**Parameters** *key* – the key to retrieve an item for

**Returns** The record found or *None*

**release**(*key: str*)  
Release the lock on a given cache key.

**abstract async set**(*keys: Union[str, Sequence[str]], value: Any, ttl: Optional[int] = None*)  
Add an item to the cache with an optional ttl.

**Parameters**

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache

- **ttl** – number of second that the record should persist

**exception** `aries_cloudagent.cache.base.CacheError`(\*args, error\_code: *Optional[str]* = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base class for cache-related errors.

**class** `aries_cloudagent.cache.base.CacheKeyLock`(cache: `aries_cloudagent.cache.base.BaseCache`, key: *str*)

Bases: `object`

A lock on a particular cache key.

Used to prevent multiple async threads from generating or querying the same semi-expensive data. Not thread safe.

**property done:** `bool`

Accessor for the done state.

**property future:** `_asyncio.Future`

Fetch the result in the form of an awaitable future.

**property parent:** `aries_cloudagent.cache.base.CacheKeyLock`

Accessor for the parent key lock, if any.

**release()**

Release the cache lock.

**property result:** `Any`

Fetch the current result, if any.

**async set\_result**(value: *Any*, ttl: *Optional[int]* = None)

Set the result, updating the cache and any waiters.

## `aries_cloudagent.cache.in_memory` module

Basic in-memory cache implementation.

**class** `aries_cloudagent.cache.in_memory.InMemoryCache`

Bases: `aries_cloudagent.cache.base.BaseCache`

Basic in-memory cache class.

**async clear**(key: *str*)

Remove an item from the cache, if present.

**Parameters** **key** – the key to remove

**async flush()**

Remove all items from the cache.

**async get**(key: *str*)

Get an item from the cache.

**Parameters** **key** – the key to retrieve an item for

**Returns** The record found or *None*

**async set**(keys: *Union[str, Sequence[str]]*, value: *Any*, ttl: *Optional[int]* = None)

Add an item to the cache with an optional ttl.

Overwrites existing cache entries.

### Parameters

- **keys** – the key or keys for which to set an item
- **value** – the value to store in the cache
- **ttl** – number of seconds that the record should persist

### `aries_cloudagent.commands` package

Commands module common setup.

`aries_cloudagent.commands.available_commands()`  
Index available commands.

`aries_cloudagent.commands.load_command(command: str)`  
Load the module corresponding with a named command.

`aries_cloudagent.commands.run_command(command: str, argv: Optional[Sequence[str]] = None)`  
Execute a named command with command line arguments.

### Submodules

#### `aries_cloudagent.commands.help` module

Help command for indexing available commands.

`aries_cloudagent.commands.help.execute(argv: Optional[Sequence[str]] = None)`  
Execute the help command.

`aries_cloudagent.commands.help.main()`  
Execute the main line.

#### `aries_cloudagent.commands.provision` module

#### `aries_cloudagent.commands.start` module

#### `aries_cloudagent.commands.upgrade` module

### `aries_cloudagent.config` package

### Submodules

#### `aries_cloudagent.config argparse` module

#### `aries_cloudagent.config.banner` module

Module to contain logic to generate the banner for ACA-py.

**class** `aries_cloudagent.config.banner.Banner(border: str, length: int)`  
Bases: `object`

Management class to generate a banner for ACA-py.

**lr\_pad**(*content: str*)  
Pad string content with defined border character.

**Parameters** **content** – String content to pad

**print\_border**()  
Print a full line using the border character.

**print\_list**(*items*)  
Print a list of items, prepending a dash to each item.

**print\_spacer**()  
Print an empty line with the border character only.

**print\_subtitle**(*title*)  
Print a subtitle for a section.

**print\_title**(*title*)  
Print the main title element.

**print\_version**(*version*)  
Print the current version.

## aries\_cloudagent.config.base module

Configuration base classes.

**class** `aries_cloudagent.config.base.BaseInjector`  
Bases: `abc.ABC`

Base injector class.

**abstract** **copy**() → *aries\_cloudagent.config.base.BaseInjector*  
Produce a copy of the injector instance.

**abstract** **inject**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, Any]] = None*) → *aries\_cloudagent.config.base.InjectType*  
Get the provided instance of a given class identifier.

**Parameters**

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**abstract** **inject\_or**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, Any]] = None, default: Optional[aries\_cloudagent.config.base.InjectType] = None*) → *Optional[aries\_cloudagent.config.base.InjectType]*  
Get the provided instance of a given class identifier or default if not found.

**Parameters**

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**class** aries\_cloudagent.config.base.BaseProvider

Bases: `abc.ABC`

Base provider class.

**provide**(*settings*: `aries_cloudagent.config.base.BaseSettings`, *injector*: `aries_cloudagent.config.base.BaseInjector`)

Provide the object instance given a config and injector.

**class** aries\_cloudagent.config.base.BaseSettings(\*args, \*\*kwargs)

Bases: `Mapping[str, Any]`

Base settings class.

**abstract copy**() → `aries_cloudagent.config.base.BaseSettings`

Produce a copy of the settings instance.

**abstract extend**(*other*: `Mapping[str, Any]`) → `aries_cloudagent.config.base.BaseSettings`

Merge another mapping to produce a new settings instance.

**get\_bool**(\*var\_names, *default*: `Optional[bool] = None`) → `Optional[bool]`

Fetch a setting as a boolean value.

#### Parameters

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**get\_int**(\*var\_names, *default*: `Optional[int] = None`) → `Optional[int]`

Fetch a setting as an integer value.

#### Parameters

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**get\_str**(\*var\_names, *default*: `Optional[str] = None`) → `Optional[str]`

Fetch a setting as a string value.

#### Parameters

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**abstract get\_value**(\*var\_names, *default*: `Optional[Any] = None`) → `Any`

Fetch a setting.

#### Parameters

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**Returns** The setting value, if defined, otherwise the default value

**exception** aries\_cloudagent.config.base.ConfigError(\*args, *error\_code*: `Optional[str] = None`, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

A base exception for all configuration errors.

**exception** aries\_cloudagent.config.base.InjectionError(\*args, *error\_code*: `Optional[str] = None`, \*\*kwargs)

Bases: `aries_cloudagent.config.base.ConfigError`

The base exception raised by Injector and Provider implementations.

**exception** `aries_cloudagent.config.base.SettingsError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.config.base.ConfigError`

The base exception raised by *BaseSettings* implementations.

## `aries_cloudagent.config.base_context` module

Base injection context builder classes.

**class** `aries_cloudagent.config.base_context.ContextBuilder(settings: Optional[Mapping[str, Any]] = None)`

Bases: `abc.ABC`

Base injection context builder class.

**abstract async** `build_context()` → `aries_cloudagent.config.injection_context.InjectionContext`  
Build the base injection context.

**update\_settings**(`settings: Mapping[str, object]`)  
Update the context builder with additional settings.

## `aries_cloudagent.config.default_context` module

## `aries_cloudagent.config.error` module

Errors for config modules.

**exception** `aries_cloudagent.config.error.ArgsParseError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.config.base.ConfigError`

Error raised when there is a problem parsing the command-line arguments.

## `aries_cloudagent.config.injection_context` module

Injection context implementation.

**class** `aries_cloudagent.config.injection_context.InjectionContext(*, settings: Optional[Mapping[str, object]] = None, enforce_typing: bool = True)`

Bases: `aries_cloudagent.config.base.BaseInjector`

Manager for configuration settings and class providers.

**ROOT\_SCOPE** = 'application'

**copy**() → `aries_cloudagent.config.injection_context.InjectionContext`  
Produce a copy of the injector instance.

**inject**(`base_cls: Type[aries_cloudagent.config.base.InjectType]`, `settings: Optional[Mapping[str, object]] = None`) → `aries_cloudagent.config.base.InjectType`  
Get the provided instance of a given class identifier.

**Parameters**

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**inject\_or**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None, default: Optional[aries\_cloudagent.config.base.InjectType] = None*) → *Optional[aries\_cloudagent.config.base.InjectType]*

Get the provided instance of a given class identifier or default if not found.

**Parameters**

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**property injector:** *aries\_cloudagent.config.injector.Injector*

Accessor for scope-specific injector.

**injector\_for\_scope**(*scope\_name: str*) → *aries\_cloudagent.config.injector.Injector*

Fetch the injector for a specific scope.

**Parameters** **scope\_name** – The unique scope identifier

**property scope\_name:** *str*

Accessor for the current scope name.

**property settings:** *aries\_cloudagent.config.settings.Settings*

Accessor for scope-specific settings.

**start\_scope**(*scope\_name: str, settings: Optional[Mapping[str, object]] = None*) → *aries\_cloudagent.config.injection\_context.InjectionContext*

Begin a new named scope.

**Parameters**

- **scope\_name** – The unique name for the scope being entered
- **settings** – An optional mapping of additional settings to apply

**Returns** A new injection context representing the scope

**update\_settings**(*settings: Mapping[str, object]*)

Update the scope with additional settings.

**exception** *aries\_cloudagent.config.injection\_context.InjectionContextError*(\*args, error\_code: *Optional[str] = None, \*\*kwargs*)

Bases: *aries\_cloudagent.config.base.InjectionError*

Base class for issues in the injection context.

**class** *aries\_cloudagent.config.injection\_context.Scope*(*name, injector*)

Bases: *tuple*

**property injector**

Alias for field number 1

**property name**

Alias for field number 0



## aries\_cloudagent.config.injector module

Standard Injector implementation.

**class** `aries_cloudagent.config.injector.Injector`(*settings: Optional[Mapping[str, object]] = None, \*, enforce\_typing: bool = True*)

Bases: `aries_cloudagent.config.base.BaseInjector`

Injector implementation with static and dynamic bindings.

**bind\_instance**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], instance: aries\_cloudagent.config.base.InjectType*)

Add a static instance as a class binding.

**bind\_provider**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], provider: aries\_cloudagent.config.base.BaseProvider, \*, cache: bool = False*)

Add a dynamic instance resolver as a class binding.

**clear\_binding**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType]*)

Remove a previously-added binding.

**copy**() → `aries_cloudagent.config.base.BaseInjector`

Produce a copy of the injector instance.

**get\_provider**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType]*)

Find the provider associated with a class binding.

**inject**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None*) → `aries_cloudagent.config.base.InjectType`

Get the provided instance of a given class identifier.

### Parameters

- **cls** – The base class to retrieve an instance of
- **params** – An optional dict providing configuration to the provider

**Returns** An instance of the base class, or None

**inject\_or**(*base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None, default: Optional[aries\_cloudagent.config.base.InjectType] = None*) → `Optional[aries_cloudagent.config.base.InjectType]`

Get the provided instance of a given class identifier or default if not found.

### Parameters

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**property settings:** `aries_cloudagent.config.settings.Settings`

Accessor for scope-specific settings.

## aries\_cloudagent.config.ledger module

## aries\_cloudagent.config.logging module

Utilities related to logging.

**class** aries\_cloudagent.config.logging.LoggingConfigurator

Bases: `object`

Utility class used to configure logging and print an informative start banner.

**classmethod** `configure`(*logging\_config\_path: Optional[str] = None, log\_level: Optional[str] = None, log\_file: Optional[str] = None*)

Configure logger.

### Parameters

- **logging\_config\_path** – str: (Default value = None) Optional path to custom logging config
- **log\_level** – str: (Default value = None)

**classmethod** `print_banner`(*agent\_label, inbound\_transports, outbound\_transports, public\_did, admin\_server=None, banner\_length=40, border\_character=':'*)

Print a startup banner describing the configuration.

### Parameters

- **agent\_label** – Agent Label
- **inbound\_transports** – Configured inbound transports
- **outbound\_transports** – Configured outbound transports
- **admin\_server** – Admin server info
- **public\_did** – Public DID
- **banner\_length** – (Default value = 40) Length of the banner
- **border\_character** – (Default value = “:”) Character to use in banner
- **border** –

**aries\_cloudagent.config.logging.load\_resource**(*path: str, encoding: Optional[str] = None*) → `TextIO`

Open a resource file located in a python package or the local filesystem.

**Parameters** **path** – The resource path in the form of *dir/file* or *package:dir/file*

**Returns** A file-like object representing the resource

## aries\_cloudagent.config.plugin\_settings module

Settings implementation for plugins.

**class** aries\_cloudagent.config.plugin\_settings.PluginSettings(*values: Optional[Mapping[str, Any]] = None*)

Bases: `Mapping[str, Any]`

Retrieve immutable settings for plugins.

Plugin settings should be retrieved by calling:

`PluginSettings.for_plugin(settings, “my_plugin”, {“default”: “values”})`

This will extract the `PLUGIN_CONFIG_KEY` in “settings” and return a new `PluginSettings` instance.

**copy()** → *aries\_cloudagent.config.base.BaseSettings*

Produce a copy of the settings instance.

**extend**(*other: Mapping[str, Any]*) → *aries\_cloudagent.config.base.BaseSettings*

Merge another settings instance to produce a new instance.

**classmethod for\_plugin**(*settings: aries\_cloudagent.config.base.BaseSettings, plugin: str, default: Optional[Mapping[str, Any]] = None*) → *aries\_cloudagent.config.plugin\_settings.PluginSettings*

Construct a `PluginSettings` object from another settings object.

`PLUGIN_CONFIG_KEY` is read from settings.

**get\_value**(\**var\_names: str, default: Optional[Any] = None*)

Fetch a setting.

#### Parameters

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

## aries\_cloudagent.config.provider module

Service provider implementations.

**class** `aries_cloudagent.config.provider.CachedProvider`(*provider: aries\_cloudagent.config.base.BaseProvider, unique\_settings\_keys: tuple = ()*)

Bases: *aries\_cloudagent.config.base.BaseProvider*

Cache the result of another provider.

**provide**(*config: aries\_cloudagent.config.base.BaseSettings, injector: aries\_cloudagent.config.base.BaseInjector*)

Provide the object instance given a config and injector.

Instances are cached keyed on a SHA256 digest of the relevant subset of settings.

**class** `aries_cloudagent.config.provider.ClassProvider`(*instance\_cls: Union[str, type], \*ctor\_args, init\_method: Optional[str] = None, \*\*ctor\_kwargs*)

Bases: *aries\_cloudagent.config.base.BaseProvider*

Provider for a particular class.

**class** `Inject`(*base\_cls: type*)

Bases: `object`

A class for passing injected arguments to the constructor.

**provide**(*config: aries\_cloudagent.config.base.BaseSettings, injector: aries\_cloudagent.config.base.BaseInjector*)

Provide the object instance given a config and injector.

**class** `aries_cloudagent.config.provider.InstanceProvider`(*instance*)

Bases: *aries\_cloudagent.config.base.BaseProvider*

Provider for a previously-created instance.

**provide**(*config*: aries\_cloudagent.config.base.BaseSettings, *injector*: aries\_cloudagent.config.base.BaseInjector)  
Provide the object instance given a config and injector.

**class** aries\_cloudagent.config.provider.StatsProvider(*provider*: aries\_cloudagent.config.base.BaseProvider, *methods*: Sequence[str], \*, *ignore\_missing*: bool = True)

Bases: aries\_cloudagent.config.base.BaseProvider

Add statistics to the results of another provider.

**provide**(*config*: aries\_cloudagent.config.base.BaseSettings, *injector*: aries\_cloudagent.config.base.BaseInjector)  
Provide the object instance given a config and injector.

## aries\_cloudagent.config.settings module

Settings implementation.

**class** aries\_cloudagent.config.settings.Settings(*values*: Optional[Mapping[str, Any]] = None)  
Bases: aries\_cloudagent.config.base.BaseSettings, MutableMapping[str, Any]

Mutable settings implementation.

**clear\_value**(*var\_name*: str)  
Remove a setting.

**Parameters** **var\_name** – The name of the setting

**copy**() → aries\_cloudagent.config.base.BaseSettings  
Produce a copy of the settings instance.

**extend**(*other*: Mapping[str, Any]) → aries\_cloudagent.config.base.BaseSettings  
Merge another settings instance to produce a new instance.

**for\_plugin**(*plugin*: str, *default*: Optional[Mapping[str, Any]] = None)  
Retrieve settings for plugin.

**get\_value**(\**var\_names*, *default*=None)  
Fetch a setting.

**Parameters**

- **var\_names** – A list of variable name alternatives
- **default** – The default value to return if none are defined

**set\_default**(*var\_name*: str, *value*)  
Add a setting if not currently defined.

**Parameters**

- **var\_name** – The name of the setting
- **value** – The value to assign

**set\_value**(*var\_name*: str, *value*)  
Add a setting.

**Parameters**

- **var\_name** – The name of the setting

- **value** – The value to assign

**update**(*other: Mapping[str, Any]*)  
Update the settings in place.

### **aries\_cloudagent.config.util module**

Entrypoint.

**class** aries\_cloudagent.config.util.**BoundedInt**(*min: Optional[int] = None, max: Optional[int] = None*)  
Bases: **object**

Argument value parser for a bounded integer.

**class** aries\_cloudagent.config.util.**ByteSize**(*min: int = 0, max: Optional[int] = None*)  
Bases: **object**

Argument value parser for byte sizes.

aries\_cloudagent.config.util.**common\_config**(*settings: Mapping[str, Any]*)  
Perform common app configuration.

### **aries\_cloudagent.config.wallet module**

### **aries\_cloudagent.connections package**

#### **Subpackages**

### **aries\_cloudagent.connections.models package**

#### **Subpackages**

### **aries\_cloudagent.connections.models.diddoc package**

DID Document model support.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

**class** aries\_cloudagent.connections.models.diddoc.**DIDDoc**(*did: Optional[str] = None*)  
Bases: **object**

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**CONTEXT** = 'https://w3id.org/did/v1'

**add\_service\_pubkeys**(*service*: *dict*, *tags*: *Union[Sequence[str], str]*) →  
*List[aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]*

Add public keys specified in service. Return public keys so discovered.

**Parameters**

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

**Raises** **ValueError** – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

**property authnkey:** **dict**

Accessor for public keys marked as authentication keys, by identifier.

**classmethod deserialize**(*did\_doc*: *dict*) →  
*aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc*

Construct DIDDoc object from dict representation.

**Parameters** **did\_doc** – DIDDoc dict representation

**Raises** **ValueError** – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

**property did:** **str**

Accessor for DID.

**classmethod from\_json**(*did\_doc\_json*: *str*) →  
*aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc*

Construct DIDDoc object from json representation.

**Parameters** **did\_doc\_json** – DIDDoc json representation

Returns: DIDDoc from input json

**property pubkey:** **dict**

Accessor for public keys by identifier.

**serialize()** → **dict**

Dump current object to a JSON-compatible dictionary.

**Returns** dict representation of current DIDDoc

**property service:** **dict**

Accessor for services by identifier.

**set**(*item*: *Union[aries\_cloudagent.connections.models.diddoc.service.Service,*  
*aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]*) →  
*aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc*

Add or replace service or public key; return current DIDDoc.

**Raises** **ValueError** – if input item is neither service nor public key.

**Parameters** **item** – service or public key to set

Returns: the current DIDDoc

**to\_json()** → **str**

Dump current object as json (JSON-LD).

**Returns** json representation of current DIDDoc

```
class aries_cloudagent.connections.models.diddoc.LinkedDataKeySpec(ver_type, authn_type,
                                                                    specifier)
```

Bases: `tuple`

**property authn\_type**

Alias for field number 1

**property specifier**

Alias for field number 2

**property ver\_type**

Alias for field number 0

```
class aries_cloudagent.connections.models.diddoc.PublicKey(did: str, ident: str, value: str, pk_type:
```

*Op-*

*tional*[aries\_cloudagent.connections.models.diddoc.public

*= None, controller: Optional[str] =*

*None, authn: bool = False)*

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**property authn: bool**

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

**property controller: str**

Accessor for the controller DID.

**property did: str**

Accessor for the DID.

**property id: str**

Accessor for the public key identifier.

**to\_dict()** → `dict`

Return dict representation of public key to embed in DID document.

**property type: aries\_cloudagent.connections.models.diddoc.publickey.PublicKeyType**

Accessor for the public key type.

**property value: str**

Accessor for the public key value.

```
class aries_cloudagent.connections.models.diddoc.PublicKeyType(value)
```

Bases: `enum.Enum`

Class encapsulating public key types.

```
ED25519_SIG_2018 = LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018',
authn_type='Ed25519SignatureAuthentication2018', specifier='publicKeyBase58')
```

```
EDDSA_SA_SIG_SECP256K1 = LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018',
authn_type='Secp256k1SignatureAuthenticationKey2018', specifier='publicKeyHex')
```

```
RSA_SIG_2018 = LinkedDataKeySpec(ver_type='RsaVerificationKey2018',
authn_type='RsaSignatureAuthentication2018', specifier='publicKeyPem')
```

**property authn\_type: str**

Accessor for the authentication type identifier.

**static get**(val: *str*) → *aries\_cloudagent.connections.models.diddoc.publickey.PublicKeyType*

Find enum instance corresponding to input value (RsaVerificationKey2018 etc).

**Parameters** **val** – input value marking public key type

Returns: the public key type

**specification**(val: *str*) → *str*

Return specifier and input value for use in public key specification.

**Parameters** **val** – value of public key

Returns: dict mapping applicable specifier to input value

**property specifier:** *str*

Accessor for the value specifier.

**property ver\_type:** *str*

Accessor for the verification type identifier.

**class** *aries\_cloudagent.connections.models.diddoc.Service*(did: *str*, ident: *str*, typ: *str*, recip\_keys:

*Union[Sequence,*

*aries\_cloudagent.connections.models.diddoc.publickey.PublicKey,*

*routing\_keys: Union[Sequence,*

*aries\_cloudagent.connections.models.diddoc.publickey.PublicKey,*

*endpoint: str, priority: int = 0)*

Bases: *object*

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**property did:** *str*

Accessor for the DID value.

**property endpoint:** *str*

Accessor for the endpoint value.

**property id:** *str*

Accessor for the service identifier.

**property priority:** *int*

Accessor for the priority value.

**property recip\_keys:**

*List[aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]*

Accessor for the recipient keys.

**property routing\_keys:**

*List[aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]*

Accessor for the routing keys.

**to\_dict**() → *dict*

Return dict representation of service to embed in DID document.

**property type:** *str*

Accessor for the service type.



## Submodules

### aries\_cloudagent.connections.models.diddoc.diddoc module

DID Document classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

**class** aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc(*did: Optional[str] = None*)

Bases: `object`

DID document, grouping a DID with verification keys and services.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**CONTEXT** = 'https://w3id.org/did/v1'

**add\_service\_pubkeys**(*service: dict, tags: Union[Sequence[str], str]*) →

List[aries\_cloudagent.connections.models.diddoc.publickey.PublicKey]

Add public keys specified in service. Return public keys so discovered.

#### Parameters

- **service** – service from DID document
- **tags** – potential tags marking public keys of type of interest (the standard is still coalescing)

**Raises** `ValueError` – for public key reference not present in DID document.

Returns: list of public keys from the document service specification

**property authnkey:** `dict`

Accessor for public keys marked as authentication keys, by identifier.

**classmethod deserialize**(*did\_doc: dict*) →

aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc

Construct DIDDoc object from dict representation.

**Parameters** **did\_doc** – DIDDoc dict representation

**Raises** `ValueError` – for bad DID or missing mandatory item.

Returns: DIDDoc from input json

**property did:** `str`

Accessor for DID.

**classmethod from\_json**(*did\_doc\_json: str*) →

aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc

Construct DIDDoc object from json representation.

**Parameters** **did\_doc\_json** – DIDDoc json representation

Returns: DIDDoc from input json

**property pubkey:** `dict`

Accessor for public keys by identifier.

**serialize()** → `dict`

Dump current object to a JSON-compatible dictionary.

**Returns** dict representation of current DIDDoc

**property service:** `dict`

Accessor for services by identifier.

**set**(*item*: `Union[aries_cloudagent.connections.models.diddoc.service.Service, aries_cloudagent.connections.models.diddoc.publickey.PublicKey]`) → `aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc`

Add or replace service or public key; return current DIDDoc.

**Raises** `ValueError` – if input item is neither service nor public key.

**Parameters** *item* – service or public key to set

Returns: the current DIDDoc

**to\_json()** → `str`

Dump current object as json (JSON-LD).

**Returns** json representation of current DIDDoc

## aries\_cloudagent.connections.models.diddoc.publickey module

DID Document Public Key classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpec(ver_type,
                                                                           authn_type,
                                                                           specifier)
```

Bases: `tuple`

**property authn\_type**

Alias for field number 1

**property specifier**

Alias for field number 2

**property ver\_type**

Alias for field number 0

```
class aries_cloudagent.connections.models.diddoc.publickey.PublicKey(did: str, ident: str, value: str, pk_type: Optional[aries_cloudagent.connections.models.PublicKeyType] = None, controller: Optional[str] = None, authn: bool = False)
```

Bases: `object`

Public key specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**property authn:** `bool`

Accessor for the authentication marker.

Returns: whether public key is marked as having DID authentication privilege

**property controller:** `str`

Accessor for the controller DID.

**property did:** `str`

Accessor for the DID.

**property id:** `str`

Accessor for the public key identifier.

**to\_dict()** → `dict`

Return dict representation of public key to embed in DID document.

**property type:** `aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType`

Accessor for the public key type.

**property value:** `str`

Accessor for the public key value.

```
class aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType(value)
```

Bases: `enum.Enum`

Class encapsulating public key types.

```
ED25519_SIG_2018 = LinkedDataKeySpec(ver_type='Ed25519VerificationKey2018',
authn_type='Ed25519SignatureAuthentication2018', specifier='publicKeyBase58')
```

```
EDDSA_SA_SIG_SECP256K1 = LinkedDataKeySpec(ver_type='Secp256k1VerificationKey2018',
authn_type='Secp256k1SignatureAuthenticationKey2018', specifier='publicKeyHex')
```

```
RSA_SIG_2018 = LinkedDataKeySpec(ver_type='RsaVerificationKey2018',
authn_type='RsaSignatureAuthentication2018', specifier='publicKeyPem')
```

**property authn\_type:** `str`

Accessor for the authentication type identifier.

**static get**(*val: str*) → `aries_cloudagent.connections.models.diddoc.publickey.PublicKeyType`

Find enum instance corresponding to input value (RsaVerificationKey2018 etc).

**Parameters** *val* – input value marking public key type

Returns: the public key type

**specification**(*val: str*) → `str`

Return specifier and input value for use in public key specification.

**Parameters** *val* – value of public key

Returns: dict mapping applicable specifier to input value

**property specifier:** `str`

Accessor for the value specifier.

**property ver\_type:** `str`

Accessor for the verification type identifier.

### `aries_cloudagent.connections.models.diddoc.service` module

DID Document Service classes.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
class aries_cloudagent.connections.models.diddoc.service.Service(did: str, ident: str, typ: str,  
    recip_keys: Union[Sequence,  
    aries_cloudagent.connections.models.diddoc.publickey.PublicKey],  
    routing_keys: Union[Sequence,  
    aries_cloudagent.connections.models.diddoc.publickey.PublicKey],  
    endpoint: str, priority: int = 0)
```

Bases: `object`

Service specification to embed in DID document.

Retains DIDs as raw values (orientated toward indy-facing operations), everything else as URIs (oriented toward W3C-facing operations).

**property did:** `str`

Accessor for the DID value.

**property endpoint:** `str`

Accessor for the endpoint value.

**property id:** `str`

Accessor for the service identifier.

**property priority:** `int`

Accessor for the priority value.

**property recip\_keys:**

`List[aries_cloudagent.connections.models.diddoc.publickey.PublicKey]`

Accessor for the recipient keys.

**property routing\_keys:**

`List[aries_cloudagent.connections.models.diddoc.publickey.PublicKey]`

Accessor for the routing keys.

**to\_dict()** → `dict`

Return dict representation of service to embed in DID document.

**property type:** `str`

Accessor for the service type.

**aries\_cloudagent.connections.models.diddoc.util module**

DIDDoc utility methods.

Copyright 2017-2019 Government of Canada Public Services and Procurement Canada - buyandsell.gc.ca

Licensed under the Apache License, Version 2.0 (the “License”); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

`aries_cloudagent.connections.models.diddoc.util.canon_did(uri: str) → str`

Convert a URI into a DID if need be, left-stripping ‘did:sov:’ if present.

**Parameters** `uri` – input URI or DID

**Raises** `ValueError` – for invalid input.

`aries_cloudagent.connections.models.diddoc.util.canon_ref(did: str, ref: str, delimiter: Optional[str] = None)`

Given a reference in a DID document, return it in its canonical form of a URI.

**Parameters**

- **did** – DID acting as the identifier of the DID document
- **ref** – reference to canonicalize, either a DID or a fragment pointing to a location in the DID doc
- **delimiter** – delimiter character marking fragment (default ‘#’) or introducing identifier (‘;’) against DID resource

`aries_cloudagent.connections.models.diddoc.util.ok_did(token: str) → bool`

Whether input token looks like a valid decentralized identifier.

**Parameters** `token` – candidate string

Returns: whether input token looks like a valid schema identifier

`aries_cloudagent.connections.models.diddoc.util.resource(ref: str, delimiter: Optional[str] = None) → str`

Extract the resource for an identifier.

Given a (URI) reference, return up to its delimiter (exclusively), or all of it if there is none.

**Parameters**

- **ref** – reference
- **delimiter** – delimiter character (default None maps to ‘#’, or ‘;’ introduces identifiers)

### Submodules

`aries_cloudagent.connections.models.conn_record` module

`aries_cloudagent.connections.models.connection_target` module

### Submodules

`aries_cloudagent.connections.base_manager` module

`aries_cloudagent.connections.util` module

`aries_cloudagent.core` package

### Subpackages

`aries_cloudagent.core.in_memory` package

### Subpackages

`aries_cloudagent.core.in_memory.didcomm` package

### Submodules

`aries_cloudagent.core.in_memory.didcomm.derive_1pu` module

`aries_cloudagent.core.in_memory.didcomm.derive_ecdh` module

### Submodules

`aries_cloudagent.core.in_memory.profile` module

### Submodules

`aries_cloudagent.core.conductor` module

`aries_cloudagent.core.dispatcher` module

`aries_cloudagent.core.error` module

Common exception classes.

**exception** `aries_cloudagent.core.error.BaseError`(\*args, error\_code: *Optional[str]* = None, \*\*kwargs)  
Bases: `Exception`

Generic exception class which other exceptions should inherit from.

**property message:** `str`

Accessor for the error message.

**property roll\_up:** `str`

Accessor for nested error messages rolled into one line.

For display: aiohttp.web errors truncate after newline.

**exception** `aries_cloudagent.core.error.ProfileDuplicateError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.ProfileError`

Profile with the given name already exists.

**exception** `aries_cloudagent.core.error.ProfileError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base error for profile operations.

**exception** `aries_cloudagent.core.error.ProfileNotFoundError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.ProfileError`

Requested profile was not found.

**exception** `aries_cloudagent.core.error.ProfileSessionInactiveError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.ProfileError`

Error raised when a profile session is not currently active.

**exception** `aries_cloudagent.core.error.ProtocolDefinitionValidationError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when there is a problem validating a protocol definition.

**exception** `aries_cloudagent.core.error.ProtocolMinorVersionNotSupported(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Minimum minor version protocol error.

Error raised when protocol support exists but minimum minor version is higher than in @type parameter.

**exception** `aries_cloudagent.core.error.StartupError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Error raised when there is a problem starting the conductor.

## aries\_cloudagent.core.event\_bus module

A simple event bus.

**class** aries\_cloudagent.core.event\_bus.**Event**(*topic: str, payload: Optional[Any] = None*)

Bases: `object`

A simple event object.

**property** `payload`

Return this event's payload.

**property** `topic`

Return this event's topic.

**with\_metadata**(*metadata: aries\_cloudagent.core.event\_bus.EventMetadata*) →  
*aries\_cloudagent.core.event\_bus.EventWithMetadata*

Annotate event with metadata and return EventWithMetadata object.

**class** aries\_cloudagent.core.event\_bus.**EventBus**

Bases: `object`

A simple event bus implementation.

**async** **notify**(*profile: Profile, event: aries\_cloudagent.core.event\_bus.Event*)

Notify subscribers of event.

### Parameters

- **profile** (*Profile*) – context of the event
- **event** (*Event*) – event to emit

**subscribe**(*pattern: Pattern, processor: Callable*)

Subscribe to an event.

### Parameters

- **pattern** (*Pattern*) – compiled regular expression for matching topics
- **processor** (*Callable*) – async callable accepting profile and event

**unsubscribe**(*pattern: Pattern, processor: Callable*)

Unsubscribe from an event.

This method is idempotent. Repeated calls to unsubscribe will not result in errors.

### Parameters

- **pattern** (*Pattern*) – regular expression used to subscribe the processor
- **processor** (*Callable*) – processor to unsubscribe

**wait\_for\_event**(*waiting\_profile: Profile, pattern: Pattern, cond: Optional[Callable[[aries\_cloudagent.core.event\_bus.Event], bool]] = None*) →  
*Iterator[Awaitable[aries\_cloudagent.core.event\_bus.Event]]*

Capture an event and retrieve its value.

**class** aries\_cloudagent.core.event\_bus.**EventMetadata**(*pattern: Pattern, match: Match[str]*)

Bases: `tuple`

Metadata passed alongside events to add context.

**property** `match`

Alias for field number 1



**property pattern**

Alias for field number 0

**class** aries\_cloudagent.core.event\_bus.EventWithMetadata(*topic: str, payload: Any, metadata: aries\_cloudagent.core.event\_bus.EventMetadata*)

Bases: *aries\_cloudagent.core.event\_bus.Event*

Event with metadata passed alongside events to add context.

**property metadata:** *aries\_cloudagent.core.event\_bus.EventMetadata*

Return metadata.

**class** aries\_cloudagent.core.event\_bus.MockEventBusBases: *aries\_cloudagent.core.event\_bus.EventBus*

A mock EventBus for testing.

**async notify**(*profile: Profile, event: aries\_cloudagent.core.event\_bus.Event*)

Append the event to MockEventBus.events.

**aries\_cloudagent.core.goal\_code\_registry module**

Handle registration and publication of supported goal codes.

**class** aries\_cloudagent.core.goal\_code\_registry.GoalCodeRegistryBases: *object*

Goal code registry.

**goal\_codes\_matching\_query**(*query: str*) → Sequence[str]

Return a list of goal codes matching a query string.

**register\_controllers**(\**controller\_sets*)

Add new controllers.

**Parameters** *controller\_sets* – Mappings of controller to coroutines**aries\_cloudagent.core.oob\_processor module****aries\_cloudagent.core.plugin\_registry module**

Handle registration of plugin modules for extending functionality.

**class** aries\_cloudagent.core.plugin\_registry.PluginRegistry(*blocklist: Iterable[str] = []*)Bases: *object*

Plugin registry for indexing application plugins.

**async init\_context**(*context: aries\_cloudagent.config.injection\_context.InjectionContext*)

Call plugin setup methods on the current context.

**async load\_protocol\_version**(*context: aries\_cloudagent.config.injection\_context.InjectionContext, mod: module, version\_definition: Optional[dict] = None*)

Load a particular protocol version.

**async load\_protocols**(*context: aries\_cloudagent.config.injection\_context.InjectionContext, plugin: module*)

For modules that don't implement setup, register protocols manually.

**property plugin\_names:** Sequence[str]  
Accessor for a list of all plugin modules.

**property plugins:** Sequence[module]  
Accessor for a list of all plugin modules.

**post\_process\_routes**(app)  
Call route binary file response OpenAPI fixups if applicable.

**async register\_admin\_routes**(app)  
Call route registration methods on the current context.

**register\_package**(package\_name: str) → Sequence[module]  
Register all modules (sub-packages) under a given package name.

**register\_plugin**(module\_name: str) → module  
Register a plugin module.

**register\_protocol\_events**(context: aries\_cloudagent.config.injection\_context.InjectionContext)  
Call route register\_events methods on the current context.

**validate\_version**(version\_list, module\_name)  
Validate version dict format.

## aries\_cloudagent.core.profile module

Classes for managing profile information within a request context.

**class** aries\_cloudagent.core.profile.Profile(\*, context: Optional[aries\_cloudagent.config.injection\_context.InjectionContext] = None, name: Optional[str] = None, created: bool = False)

Bases: abc.ABC

Base abstraction for handling identity-related state.

**BACKEND\_NAME:** str = None

**DEFAULT\_NAME:** str = 'default'

**property backend:** str  
Accessor for the backend implementation name.

**async close**()  
Close the profile instance.

**property context:** aries\_cloudagent.config.injection\_context.InjectionContext  
Accessor for the injection context.

**property created:** bool  
Accessor for the created flag indicating a new profile.

**inject**(base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None) → aries\_cloudagent.config.base.InjectType  
Get the provided instance of a given class identifier.

### Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**inject\_or**(*base\_cls*: *Type*[*aries\_cloudagent.config.base.InjectType*], *settings*: *Optional*[*Mapping*[*str*, *object*]] = *None*, *default*: *Optional*[*aries\_cloudagent.config.base.InjectType*] = *None*) → *Optional*[*aries\_cloudagent.config.base.InjectType*]

Get the provided instance of a given class identifier or default if not found.

#### Parameters

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**property name:** *str*

Accessor for the profile name.

**async notify**(*topic*: *str*, *payload*: *Any*)

Signal an event.

**async remove**()

Remove the profile.

**abstract session**(*context*: *Optional*[*aries\_cloudagent.config.injection\_context.InjectionContext*] = *None*) → *aries\_cloudagent.core.profile.ProfileSession*

Start a new interactive session with no transaction support requested.

**property settings:** *aries\_cloudagent.config.base.BaseSettings*

Accessor for scope-specific settings.

**abstract transaction**(*context*: *Optional*[*aries\_cloudagent.config.injection\_context.InjectionContext*] = *None*) → *aries\_cloudagent.core.profile.ProfileSession*

Start a new interactive session with commit and rollback support.

If the current backend does not support transactions, then commit and rollback operations of the session will not have any effect.

**class** *aries\_cloudagent.core.profile.ProfileManager*

Bases: *abc.ABC*

Handle provision and open for profile instances.

**abstract async open**(*context*: *aries\_cloudagent.config.injection\_context.InjectionContext*, *config*: *Optional*[*Mapping*[*str*, *Any*]] = *None*) → *aries\_cloudagent.core.profile.Profile*

Open an instance of an existing profile.

**abstract async provision**(*context*: *aries\_cloudagent.config.injection\_context.InjectionContext*, *config*: *Optional*[*Mapping*[*str*, *Any*]] = *None*) → *aries\_cloudagent.core.profile.Profile*

Provision a new instance of a profile.

**class** *aries\_cloudagent.core.profile.ProfileManagerProvider*

Bases: *aries\_cloudagent.config.base.BaseProvider*

The standard profile manager provider which keys off the selected wallet type.

**MANAGER\_TYPES** = {'askar': 'aries\_cloudagent.askar.profile.AskarProfileManager',  
'in\_memory': 'aries\_cloudagent.core.in\_memory.InMemoryProfileManager', 'indy':  
'aries\_cloudagent.indy.sdk.profile.IndySdkProfileManager'}

**provide**(*settings*: *aries\_cloudagent.config.base.BaseSettings*, *injector*: *aries\_cloudagent.config.base.BaseInjector*)

Create the profile manager instance.

```
class aries_cloudagent.core.profile.ProfileSession(profile: aries_cloudagent.core.profile.Profile, *,
                                                  context: Optional[aries_cloudagent.config.injection_context.InjectionContext]
                                                  = None, settings: Optional[Mapping[str, Any]] =
                                                  None)
```

Bases: `abc.ABC`

An active connection to the profile management backend.

**property active:** `bool`

Accessor for the session active state.

**async commit()**

Commit any updates performed within the transaction.

If the current session is not a transaction, then nothing is performed.

**property context:** `aries_cloudagent.config.injection_context.InjectionContext`

Accessor for the associated injection context.

**inject**(base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None) → aries\_cloudagent.config.base.InjectType

Get the provided instance of a given class identifier.

#### Parameters

- **cls** – The base class to retrieve an instance of
- **settings** – An optional mapping providing configuration to the provider

**Returns** An instance of the base class, or None

**inject\_or**(base\_cls: Type[aries\_cloudagent.config.base.InjectType], settings: Optional[Mapping[str, object]] = None, default: Optional[aries\_cloudagent.config.base.InjectType] = None) → Optional[aries\_cloudagent.config.base.InjectType]

Get the provided instance of a given class identifier or default if not found.

#### Parameters

- **base\_cls** – The base class to retrieve an instance of
- **settings** – An optional dict providing configuration to the provider
- **default** – default return value if no instance is found

**Returns** An instance of the base class, or None

**property is\_transaction:** `bool`

Check if the session supports commit and rollback operations.

**property profile:** `aries_cloudagent.core.profile.Profile`

Accessor for the associated profile instance.

**async rollback()**

Roll back any updates performed within the transaction.

If the current session is not a transaction, then nothing is performed.

**property settings:** `aries_cloudagent.config.base.BaseSettings`

Accessor for scope-specific settings.

## aries\_cloudagent.core.protocol\_registry module

Handle registration and publication of supported protocols.

**class** aries\_cloudagent.core.protocol\_registry.ProtocolRegistry

Bases: `object`

Protocol registry for indexing message families.

**property controllers:** `Mapping[str, str]`

Accessor for a list of all protocol controller functions.

**property message\_types:** `Sequence[str]`

Accessor for a list of all message types.

**parse\_type\_string**(*message\_type*)

Parse message type string and return dict with info.

**async prepare\_disclosed**(*context*: `aries_cloudagent.config.injection_context.InjectionContext`, *protocols*: `Sequence[str]`)

Call controllers and return publicly supported message families and roles.

**property protocols:** `Sequence[str]`

Accessor for a list of all message protocols.

**protocols\_matching\_query**(*query*: `str`) → `Sequence[str]`

Return a list of message protocols matching a query string.

**register\_controllers**(\**controller\_sets*, *version\_definition*=None)

Add new controllers.

**Parameters** **controller\_sets** – Mappings of message families to coroutines

**register\_message\_types**(\**typesets*, *version\_definition*=None)

Add new supported message types.

**Parameters**

- **typesets** – Mappings of message types to register
- **version\_definition** – Optional version definition dict

**resolve\_message\_class**(*message\_type*: `str`) → `type`

Resolve a message\_type to a message class.

Given a message type identifier, this method returns the corresponding registered message class.

**Parameters** **message\_type** – Message type to resolve

**Returns** The resolved message class

## aries\_cloudagent.core.util module

Core utilities and constants.

`aries_cloudagent.did` package

Submodules

`aries_cloudagent.did.did_key` module

`aries_cloudagent.holder` package

Submodules

`aries_cloudagent.holder.routes` module

`aries_cloudagent.indy` package

Subpackages

`aries_cloudagent.indy.credx` package

Submodules

`aries_cloudagent.indy.credx.holder` module

`aries_cloudagent.indy.credx.issuer` module

`aries_cloudagent.indy.credx.verifier` module

`aries_cloudagent.indy.models` package

Submodules

`aries_cloudagent.indy.models.cred` module

`aries_cloudagent.indy.models.cred_abstract` module

`aries_cloudagent.indy.models.cred_def` module

`aries_cloudagent.indy.models.cred_precis` module

`aries_cloudagent.indy.models.cred_request` module

`aries_cloudagent.indy.models.non_rev_interval` module

`aries_cloudagent.indy.models.predicate` module

Utilities for dealing with predicates.

```

class aries_cloudagent.indy.models.predicate.Predicate(value)
    Bases: enum.Enum

    Enum for predicate types that indy-sdk supports.

    GE = Relation(fortran='GE', wql='$gte', math='>=', yes=<function
    Predicate.<lambda>>, no=<function Predicate.<lambda>>)

    GT = Relation(fortran='GT', wql='$gt', math='>', yes=<function Predicate.<lambda>>,
    no=<function Predicate.<lambda>>)

    LE = Relation(fortran='LE', wql='$lte', math='<=', yes=<function
    Predicate.<lambda>>, no=<function Predicate.<lambda>>)

    LT = Relation(fortran='LT', wql='$lt', math='<', yes=<function Predicate.<lambda>>,
    no=<function Predicate.<lambda>>)

    property fortran: str
        Fortran nomenclature.

    static get(relation: str) → aries_cloudagent.indy.models.predicate.Predicate
        Return enum instance corresponding to input relation string.

    property math: str
        Mathematical nomenclature.

    static to_int(value: Any) → int
        Cast a value as its equivalent int for indy predicate argument.

        Raise ValueError for any input but int, stringified int, or boolean.

        Parameters value – value to coerce

    property wql: str
        WQL nomenclature.

class aries_cloudagent.indy.models.predicate.Relation(fortran, wql, math, yes, no)
    Bases: tuple

    property fortran
        Alias for field number 0

    property math
        Alias for field number 2

    property no
        Alias for field number 4

    property wql
        Alias for field number 1

    property yes
        Alias for field number 3

```

`aries_cloudagent.indy.models.pres_preview` module

`aries_cloudagent.indy.models.proof` module

`aries_cloudagent.indy.models.proof_request` module

`aries_cloudagent.indy.models.requested_creds` module

`aries_cloudagent.indy.models.revocation` module

`aries_cloudagent.indy.models.schema` module

`aries_cloudagent.indy.models.xform` module

`aries_cloudagent.indy.sdk` package

### Submodules

`aries_cloudagent.indy.sdk.error` module

Indy error handling.

```
class aries_cloudagent.indy.sdk.error.IndyErrorHandler(message: Optional[str] = None, error_cls:
                                                    Type[aries_cloudagent.core.error.BaseError]
                                                    = <class
                                                    'aries_cloudagent.core.error.BaseError'>)
```

Bases: `object`

Trap `IndyError` and raise an appropriate `LedgerError` instead.

```
classmethod wrap_error(err_value: indy.error.IndyError, message: Optional[str] = None, error_cls:
                        Type[aries_cloudagent.core.error.BaseError] = <class
                        'aries_cloudagent.core.error.BaseError'>) →
                        aries_cloudagent.core.error.BaseError
```

Create an instance of `BaseError` from an `IndyError`.

`aries_cloudagent.indy.sdk.holder` module

Indy SDK holder implementation.

```
class aries_cloudagent.indy.sdk.holder.IndySdkHolder(wallet:
                                                    aries_cloudagent.indy.sdk.wallet_setup.IndyOpenWallet)
```

Bases: `aries_cloudagent.indy.holder.IndyHolder`

Indy-SDK holder implementation.

```
async create_credential_request(credential_offer: dict, credential_definition: dict, holder_id: str)
                                → Tuple[str, str]
```

Create a credential request for the given credential offer.

#### Parameters

- **credential\_offer** – The credential offer to create request for



- **credential\_definition** – The credential definition to create an offer for
- **holder\_did** – the DID of the agent making the request

**Returns** A tuple of the credential request and credential request metadata

**async create\_presentation**(*presentation\_request: dict, requested\_credentials: dict, schemas: dict, credential\_definitions: dict, rev\_states: Optional[dict] = None*) → str

Get credentials stored in the wallet.

**Parameters**

- **presentation\_request** – Valid indy format presentation request
- **requested\_credentials** – Indy format requested credentials
- **schemas** – Indy formatted schemas JSON
- **credential\_definitions** – Indy formatted credential definitions JSON
- **rev\_states** – Indy format revocation states JSON

**async create\_revocation\_state**(*cred\_rev\_id: str, rev\_reg\_def: dict, rev\_reg\_delta: dict, timestamp: int, tails\_file\_path: str*) → str

Create current revocation state for a received credential.

**Parameters**

- **cred\_rev\_id** – credential revocation id in revocation registry
- **rev\_reg\_def** – revocation registry definition
- **rev\_reg\_delta** – revocation delta
- **timestamp** – delta timestamp

**Returns** the revocation state

**async credential\_revoked**(*ledger: aries\_cloudagent.ledger.base.BaseLedger, credential\_id: str, fro: Optional[int] = None, to: Optional[int] = None*) → bool

Check ledger for revocation status of credential by cred id.

**Parameters** **credential\_id** – Credential id to check

**async delete\_credential**(*credential\_id: str*)

Remove a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to remove

**async get\_credential**(*credential\_id: str*) → str

Get a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to retrieve

**async get\_credentials**(*start: int, count: int, wql: dict*)

Get credentials stored in the wallet.

**Parameters**

- **start** – Starting index
- **count** – Number of records to return
- **wql** – wql query dict

**async get\_credentials\_for\_presentation\_request\_by\_referent**(*presentation\_request: dict, referents: Sequence[str], start: int, count: int, extra\_query: dict = {}*)

Get credentials stored in the wallet.

#### Parameters

- **presentation\_request** – Valid presentation request from issuer
- **referents** – Presentation request referents to use to search for creds
- **start** – Starting index
- **count** – Maximum number of records to return
- **extra\_query** – wql query dict

**async get\_mime\_type**(*credential\_id: str, attr: Optional[str] = None*) → Union[dict, str]

Get MIME type per attribute (or for all attributes).

#### Parameters

- **credential\_id** – credential id
- **attr** – attribute of interest or omit for all

**Returns:** Attribute MIME type or dict mapping attribute names to MIME types attr\_meta\_json = all\_meta.tags.get(attr)

**async store\_credential**(*credential\_definition: dict, credential\_data: dict, credential\_request\_metadata: dict, credential\_attr\_mime\_types=None, credential\_id: Optional[str] = None, rev\_reg\_def: Optional[dict] = None*) → str

Store a credential in the wallet.

#### Parameters

- **credential\_definition** – Credential definition for this credential
- **credential\_data** – Credential data generated by the issuer
- **credential\_request\_metadata** – credential request metadata generated by the issuer
- **credential\_attr\_mime\_types** – dict mapping attribute names to (optional) MIME types to store as non-secret record, if specified
- **credential\_id** – optionally override the stored credential id
- **rev\_reg\_def** – revocation registry definition in json

**Returns** the ID of the stored credential

**aries\_cloudagent.indy.sdk.issuer module**

**aries\_cloudagent.indy.sdk.profile module**

**aries\_cloudagent.indy.sdk.util module**

Indy utilities.

**async aries\_cloudagent.indy.sdk.util.create\_tails\_reader**(*tails\_file\_path: str*) → int

Get a handle for the blob\_storage file reader.

**async** `aries_cloudagent.indy.sdk.util.create_tails_writer(tails_base_dir: str) → int`  
 Get a handle for the blob\_storage file writer.

### `aries_cloudagent.indy.sdk.verifier` module

### `aries_cloudagent.indy.sdk.wallet_plugin` module

Utility for loading Postgres wallet plug-in.

`aries_cloudagent.indy.sdk.wallet_plugin.file_ext()`  
 Determine file extension based on platform.

`aries_cloudagent.indy.sdk.wallet_plugin.load_postgres_plugin(storage_config, storage_creds, raise_exc=False)`  
 Load postgres dll and configure postgres wallet.

### `aries_cloudagent.indy.sdk.wallet_setup` module

Indy-SDK wallet setup and configuration.

**class** `aries_cloudagent.indy.sdk.wallet_setup.IndyOpenWallet`(*config:*  
*aries\_cloudagent.indy.sdk.wallet\_setup.IndyWalletConfig*  
*created, handle, master\_secret\_id:*  
*str*)

Bases: `object`

Handle and metadata for an opened Indy wallet.

**async** `close()`  
 Close previously-opened wallet, removing it if so configured.

**property name:** `str`  
 Accessor for the opened wallet name.

**class** `aries_cloudagent.indy.sdk.wallet_setup.IndyWalletConfig`(*config: Optional[Mapping[str, Any]] = None*)

Bases: `object`

A helper class for handling Indy-SDK wallet configuration.

**DEFAULT\_FRESHNESS** = `False`

**DEFAULT\_KEY** = `''`

**DEFAULT\_KEY\_DERIVATION** = `'ARGON2I_MOD'`

**DEFAULT\_STORAGE\_TYPE** = `None`

**KEY\_DERIVATION\_ARGON2I\_INT** = `'ARGON2I_INT'`

**KEY\_DERIVATION\_ARGON2I\_MOD** = `'ARGON2I_MOD'`

**KEY\_DERIVATION\_RAW** = `'RAW'`

**async** `create_wallet()` → *aries\_cloudagent.indy.sdk.wallet\_setup.IndyOpenWallet*  
 Create a new wallet.

**Raises**

- **ProfileDuplicateError** – If there was an existing wallet with the same name

- **ProfileError** – If there was a problem removing the wallet
- **ProfileError** – If there was another libindy error

**async open\_wallet**(*created: bool = False*) → *aries\_cloudagent.indy.sdk.wallet\_setup.IndyOpenWallet*  
 Open wallet, removing and/or creating it if so configured.

**Raises**

- **ProfileError** – If wallet not found after creation
- **ProfileNotFoundError** – If the wallet is not found
- **ProfileError** – If the wallet is already open
- **ProfileError** – If there is another libindy error

**async remove\_wallet**()  
 Remove an existing wallet.

**Raises**

- **ProfileNotFoundError** – If the wallet could not be found
- **ProfileError** – If there was another libindy error

**property wallet\_access:** *dict*  
 Accessor the Indy wallet access info.

**property wallet\_config:** *dict*  
 Accessor for the Indy wallet config.

## Submodules

### **aries\_cloudagent.indy.holder module**

Base Indy Holder class.

**class** *aries\_cloudagent.indy.holder.IndyHolder*

Bases: *abc.ABC*

Base class for holder.

**CHUNK** = 256

**RECORD\_TYPE\_MIME\_TYPES** = 'attribute-mime-types'

**abstract async create\_credential\_request**(*credential\_offer: dict, credential\_definition: dict, holder\_did: str*) → *Tuple[str, str]*

Create a credential request for the given credential offer.

**Parameters**

- **credential\_offer** – The credential offer to create request for
- **credential\_definition** – The credential definition to create an offer for
- **holder\_did** – the DID of the agent making the request

**Returns** A tuple of the credential request and credential request metadata

**abstract async create\_presentation**(*presentation\_request: dict, requested\_credentials: dict, schemas: dict, credential\_definitions: dict, rev\_states: Optional[dict] = None*) → *str*

Get credentials stored in the wallet.

**Parameters**

- **presentation\_request** – Valid indy format presentation request
- **requested\_credentials** – Indy format requested credentials
- **schemas** – Indy formatted schemas JSON
- **credential\_definitions** – Indy formatted credential definitions JSON
- **rev\_states** – Indy format revocation states JSON

**abstract async create\_revocation\_state**(*cred\_rev\_id: str, rev\_reg\_def: dict, rev\_reg\_delta: dict, timestamp: int, tails\_file\_path: str*) → str

Create current revocation state for a received credential.

**Parameters**

- **cred\_rev\_id** – credential revocation id in revocation registry
- **rev\_reg\_def** – revocation registry definition
- **rev\_reg\_delta** – revocation delta
- **timestamp** – delta timestamp

**Returns** the revocation state

**abstract async credential\_revoked**(*ledger: aries\_cloudagent.ledger.base.BaseLedger, credential\_id: str, fro: Optional[int] = None, to: Optional[int] = None*) → bool

Check ledger for revocation status of credential by cred id.

**Parameters** **credential\_id** – Credential id to check

**abstract async delete\_credential**(*credential\_id: str*)

Remove a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to remove

**abstract async get\_credential**(*credential\_id: str*) → str

Get a credential stored in the wallet.

**Parameters** **credential\_id** – Credential id to retrieve

**abstract async get\_mime\_type**(*credential\_id: str, attr: Optional[str] = None*) → Union[dict, str]

Get MIME type per attribute (or for all attributes).

**Parameters**

- **credential\_id** – credential id
- **attr** – attribute of interest or omit for all

**Returns:** Attribute MIME type or dict mapping attribute names to MIME types attr\_meta\_json = all\_meta.tags.get(attr)

**abstract async store\_credential**(*credential\_definition: dict, credential\_data: dict, credential\_request\_metadata: dict, credential\_attr\_mime\_types=None, credential\_id: Optional[str] = None, rev\_reg\_def: Optional[dict] = None*)

Store a credential in the wallet.

**Parameters**

- **credential\_definition** – Credential definition for this credential

- **credential\_data** – Credential data generated by the issuer
- **credential\_request\_metadata** – credential request metadata generated by the issuer
- **credential\_attr\_mime\_types** – dict mapping attribute names to (optional) MIME types to store as non-secret record, if specified
- **credential\_id** – optionally override the stored credential id
- **rev\_reg\_def** – revocation registry definition in json

**Returns** the ID of the stored credential

**exception** `aries_cloudagent.indy.holder.IndyHolderError`(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base class for holder exceptions.

## aries\_cloudagent.indy.issuer module

Base Indy Issuer class.

**class** `aries_cloudagent.indy.issuer.IndyIssuer`

Bases: `abc.ABC`

Base class for Indy Issuer.

**abstract async create\_and\_store\_credential\_definition**(*origin\_did: str, schema: dict, signature\_type: Optional[str] = None, tag: Optional[str] = None, support\_revocation: bool = False*) → Tuple[str, str]

Create a new credential definition and store it in the wallet.

### Parameters

- **origin\_did** – the DID issuing the credential definition
- **schema\_json** – the schema used as a basis
- **signature\_type** – the credential definition signature type (default 'CL')
- **tag** – the credential definition tag
- **support\_revocation** – whether to enable revocation for this credential def

**Returns** A tuple of the credential definition ID and JSON

**abstract async create\_and\_store\_revocation\_registry**(*origin\_did: str, cred\_def\_id: str, revoc\_def\_type: str, tag: str, max\_cred\_num: int, tails\_base\_path: str*) → Tuple[str, str, str]

Create a new revocation registry and store it in the wallet.

### Parameters

- **origin\_did** – the DID issuing the revocation registry
- **cred\_def\_id** – the identifier of the related credential definition
- **revoc\_def\_type** – the revocation registry type (default CL\_ACCUM)
- **tag** – the unique revocation registry tag

- **max\_cred\_num** – the number of credentials supported in the registry
- **tails\_base\_path** – where to store the tails file

**Returns** A tuple of the revocation registry ID, JSON, and entry JSON

**abstract async create\_credential**(*schema: dict, credential\_offer: dict, credential\_request: dict, credential\_values: dict, cred\_ex\_id: str, revoc\_reg\_id: Optional[str] = None, tails\_file\_path: Optional[str] = None*) → Tuple[str, str]

Create a credential.

**Args** *schema*: Schema to create credential for *credential\_offer*: Credential Offer to create credential for *credential\_request*: Credential request to create credential for *credential\_values*: Values to go in credential *cred\_ex\_id*: credential exchange identifier to use in issuer cred rev rec *revoc\_reg\_id*: ID of the revocation registry *tails\_file\_path*: The location of the tails file

**Returns** A tuple of created credential and revocation id

**abstract async create\_credential\_offer**(*credential\_definition\_id*) → str

Create a credential offer for the given credential definition id.

**Parameters** *credential\_definition\_id* – The credential definition to create an offer for

**Returns** The created credential offer

**abstract async create\_schema**(*origin\_did: str, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str]*) → Tuple[str, str]

Create a new credential schema and store it in the wallet.

**Parameters**

- **origin\_did** – the DID issuing the credential definition
- **schema\_name** – the schema name
- **schema\_version** – the schema version
- **attribute\_names** – a sequence of schema attribute names

**Returns** A tuple of the schema ID and JSON

**abstract async credential\_definition\_in\_wallet**(*credential\_definition\_id: str*) → bool

Check whether a given credential definition ID is present in the wallet.

**Parameters** *credential\_definition\_id* – The credential definition ID to check

**make\_credential\_definition\_id**(*origin\_did: str, schema: dict, signature\_type: Optional[str] = None, tag: Optional[str] = None*) → str

Derive the ID for a credential definition.

**make\_schema\_id**(*origin\_did: str, schema\_name: str, schema\_version: str*) → str

Derive the ID for a schema.

**abstract async merge\_revocation\_registry\_deltas**(*fro\_delta: str, to\_delta: str*) → str

Merge revocation registry deltas.

**Parameters**

- **fro\_delta** – original delta in JSON format
- **to\_delta** – incoming delta in JSON format

**Returns** Merged delta in JSON format

**abstract async revoke\_credentials**(*revoc\_reg\_id: str, tails\_file\_path: str, cred\_rev\_ids: Sequence[str]*) → Tuple[str, Sequence[str]]

Revoke a set of credentials in a revocation registry.

**Parameters**

- **revoc\_reg\_id** – ID of the revocation registry
- **tails\_file\_path** – path to the local tails file
- **cred\_rev\_ids** – sequences of credential indexes in the revocation registry

**Returns** Tuple with the combined revocation delta, list of cred rev ids not revoked

**exception** `aries_cloudagent.indy.issuer.IndyIssuerError`(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Generic issuer error.

**exception** `aries_cloudagent.indy.issuer.IndyIssuerRevocationRegistryFullError`(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.indy.issuer.IndyIssuerError`

Revocation registry is full when issuing a new credential.

## aries\_cloudagent.indy.util module

Utilities for dealing with Indy conventions.

**async** `aries_cloudagent.indy.util.generate_pr_nonce()` → str

Generate a nonce for a proof request.

`aries_cloudagent.indy.util.indy_client_dir`(subpath: Optional[str] = None, create: bool = False) → str

Return '/'-terminated subdirectory of indy-client directory.

**Parameters**

- **subpath** – subpath within indy-client structure
- **create** – whether to create subdirectory if absent

`aries_cloudagent.indy.util.tails_path`(rev\_reg\_id: str) → str

Return path to indy tails file for input rev reg id.

## aries\_cloudagent.indy.verifier module

## aries\_cloudagent.ledger package

### Subpackages

### aries\_cloudagent.ledger.merkel\_validation package

### Submodules



## aries\_cloudagent.ledger.merkel\_validation.constants module

Constants for State Proof and LeafHash Inclusion Verification.

## aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler module

Utilities for Processing Replies to Domain Read Requests.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.decode_state_value(encoded_value)`  
Return val, lsn, lut from encoded state value.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.encode_state_value(value, seqNo, txnTime)`  
Return encoded state value.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.extract_params_write_request(data)`  
Return tree\_size, leaf\_index, audit\_path, expected\_root\_hash from reply.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.get_proof_nodes(reply)`  
Return proof\_nodes from reply.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.hash_of(text) → str`  
Return 256 bit hexadecimal digest of text.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_attr(did, attr_name, attr_is_hash=False)`  
→ bytes  
Return state\_path for ATTR.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_claim_def(authors_did, schema_seq_no, signature_type, tag)`  
Return state\_path for CLAIM DEF.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_nym(did)`  
→ bytes  
Return state\_path for NYM.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_revoc_def(authors_did, cred_def_id, revoc_def_type, revoc_def_tag)`  
→ bytes  
Return state\_path for REVOC\_DEF.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_revoc_reg_entry(revoc_reg_entry)`  
→  
bytes

Return state\_path for REVOC\_REG\_ENTRY.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_revoc_reg_entry_accum(revoc_reg_entry_accum)`

Return state\_path for REVOC\_REG\_ENTRY\_ACCUM.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.make_state_path_for_schema(authors_did, schema_name, schema_version)`  
→  
bytes

Return state\_path for SCHEMA.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.parse_attr_txn(txn_data)`  
Process txn\_data and parse attr\_txn based on attr\_type.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_attr_for_state(txn, path_only=False)`

Return key, value pair for state from ATTR.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_claim_def_for_state(txn, path_only=False)`

Return key-value pair for state from CLAIM\_DEF.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_for_state_read(reply)`  
Return state value from read requests reply.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_for_state_write(reply)`  
Return state key, value pair from write requests reply.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_attr_for_state(reply)`  
Return value for state from GET\_ATTR.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_claim_def_for_state(reply)`  
Return value for state from GET\_CLAIM\_DEF.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_nym_for_state(reply)`  
Return value for state from GET\_NYM.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_revoc_def_for_state(reply)`  
Return value for state from GET\_REVOC\_DEF.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_revoc_reg_delta_for_state(reply)`  
Return value for state from GET\_REVOC\_REG\_DELTA.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_revoc_reg_entry_accum_for_state(reply)`  
Return value for state from GET\_REVOC\_REG\_ENTRY\_ACCUM.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_revoc_reg_entry_for_state(reply)`  
Return value for state from GET\_REVOC\_REG\_ENTRY.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_get_schema_for_state(reply)`  
Return value for state from GET\_SCHEMA.

`aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_nym_for_state(txn)`  
Return encoded state path from NYM.

```
aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_revoc_def_for_state(txn,
                                                                                       path_only=False)
```

Return key-value pair for state from REVOC\_DEF.

```
aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_revoc_reg_entry_accum_for_state(txn,
                                                                                               path_only=False)
```

Return key-value pair for state from REVOC\_REG\_ENTRY\_ACCUM.

```
aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_revoc_reg_entry_for_state(txn,
                                                                                             path_only=False)
```

Return key-value pair for state from REVOC\_REG\_ENTRY.

```
aries_cloudagent.ledger.merkel_validation.domain_txn_handler.prepare_schema_for_state(txn,
                                                                                      path_only=False)
```

Return key-value pair for state from SCHEMA.

## aries\_cloudagent.ledger.merkel\_validation.hasher module

Merkle tree hasher for leaf and children nodes.

```
class aries_cloudagent.ledger.merkel_validation.hasher.HexTreeHasher(hashfunc=<built-in
                                                                    function
                                                                    openssl_sha256>)
```

Bases: `aries_cloudagent.ledger.merkel_validation.hasher.TreeHasher`

Merkle tree hasher for hex data.

```
hash_children(left, right)
```

Return parent node hash corresponding to 2 child nodes.

```
hash_leaf(data)
```

Return leaf node hash.

```
class aries_cloudagent.ledger.merkel_validation.hasher.TreeHasher(hashfunc=<built-in function
                                                                    openssl_sha256>)
```

Bases: `object`

Merkle tree hasher for bytes data.

```
hash_children(left, right)
```

Return parent node hash corresponding to 2 child nodes.

```
hash_leaf(data)
```

Return leaf node hash.

## aries\_cloudagent.ledger.merkel\_validation.merkel\_verifier module

Verify Leaf Inclusion.

```
class aries_cloudagent.ledger.merkel_validation.merkel_verifier.MerkleVerifier(hasher=<aries_cloudagent.ledger.merkel_validation.hasher.HexTreeHasher
                                                                    object>)
```

Bases: `object`

Utility class for verifying leaf inclusion.

```
async calculate_root_hash(leaf, leaf_index, audit_path, tree_size)
```

Calculate root hash, used to verify Merkel AuditPath.

Reference: section 2.1.1 of RFC6962.

### Parameters

- **leaf** – Leaf data.
- **leaf\_index** – Index of the leaf in the tree.
- **audit\_path** – A list of SHA-256 hashes representing the Merkle audit
- **path.** –
- **tree\_size** – tree size

**lsb(x)**

Return Least Significant Bits.

## **aries\_cloudagent.ledger.merkel\_validation.trie module**

Validates State Proof.

**class** aries\_cloudagent.ledger.merkel\_validation.trie.**SubTrie**(*root\_hash=None*)

Bases: `object`

Utility class for SubTrie and State Proof validation.

**async static** **get\_new\_trie\_with\_proof\_nodes**(*proof\_nodes*)

Return SubTrie created from proof\_nodes.

**property** **root\_hash**

Return 32 bytes string.

**set\_root\_hash**(*root\_hash=None*)

.

**async static** **verify\_spv\_proof**(*expected\_value, proof\_nodes, serialized=True*)

Verify State Proof.

## **aries\_cloudagent.ledger.merkel\_validation.utils module**

Merkel Validation Utils.

aries\_cloudagent.ledger.merkel\_validation.utils.**ascii\_chr**(*value*)

Return bytes object.

aries\_cloudagent.ledger.merkel\_validation.utils.**audit\_path\_length**(*index: int, tree\_size: int*)

Return AuditPath length.

### **Parameters**

- **index** – Leaf index
- **tree\_size** – Tree size

aries\_cloudagent.ledger.merkel\_validation.utils.**bin\_to\_nibbles**(*s*)

Convert string s to nibbles (half-bytes).

aries\_cloudagent.ledger.merkel\_validation.utils.**encode\_hex**(*b*)

Return bytes object for string or hexadecimal rep for bytes input.

**Parameters** **b** – string or bytes

aries\_cloudagent.ledger.merkel\_validation.utils.**sha3\_256**(*x*)

Return 256 bit digest.

`aries_cloudagent.ledger.merkel_validation.utils.unpack_to_nibbles(bindata)`  
Unpack packed binary data to nibbles.

**Parameters** `bindata` – binary packed from nibbles

`aries_cloudagent.ledger.multiple_ledger` package

### Submodules

`aries_cloudagent.ledger.multiple_ledger.base_manager` module

`aries_cloudagent.ledger.multiple_ledger.indy_manager` module

`aries_cloudagent.ledger.multiple_ledger.indy_vdr_manager` module

`aries_cloudagent.ledger.multiple_ledger.ledger_config_schema` module

`aries_cloudagent.ledger.multiple_ledger.ledger_requests_executor` module

`aries_cloudagent.ledger.multiple_ledger.manager_provider` module

### Submodules

`aries_cloudagent.ledger.base` module

Ledger base class.

**class** `aries_cloudagent.ledger.base.BaseLedger`

Bases: `abc.ABC`

Base class for ledger.

**BACKEND\_NAME:** `str = None`

**abstract async** `accept_txn_author_agreement(taa_record: dict, mechanism: str, accept_time: Optional[int] = None)`

Save a new record recording the acceptance of the TAA.

**property backend:** `str`

Accessor for the ledger backend name.

**abstract async** `create_and_send_credential_definition(issuer: aries_cloudagent.indy.issuer.IndyIssuer, schema_id: str, signature_type: Optional[str] = None, tag: Optional[str] = None, support_revocation: bool = False, write_ledger: bool = True, endorser_did: Optional[str] = None) → Tuple[str, dict, bool]`

Send credential definition to ledger and store relevant key matter in wallet.

#### Parameters

- **issuer** – The issuer instance to use for credential definition creation

- **schema\_id** – The schema id of the schema to create cred def for
- **signature\_type** – The signature type to use on the credential definition
- **tag** – Optional tag to distinguish multiple credential definitions
- **support\_revocation** – Optional flag to enable revocation for this cred def

**Returns** Tuple with cred def id, cred def structure, and whether it's novel

**abstract async create\_and\_send\_schema**(*issuer: aries\_cloudagent.indy.issuer.IndyIssuer, schema\_name: str, schema\_version: str, attribute\_names: Sequence[str], write\_ledger: bool = True, endorser\_did: Optional[str] = None*) → Tuple[str, dict]

Send schema to ledger.

**Parameters**

- **issuer** – The issuer instance to use for schema creation
- **schema\_name** – The schema name
- **schema\_version** – The schema version
- **attribute\_names** – A list of schema attributes

**did\_to\_nym**(*did: str*) → str

Remove the ledger's DID prefix to produce a nym.

**abstract async fetch\_txn\_author\_agreement**()

Fetch the current AML and TAA from the ledger.

**abstract async get\_all\_endpoints\_for\_did**(*did: str*) → dict

Fetch all endpoints for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**abstract async get\_credential\_definition**(*credential\_definition\_id: str*) → dict

Get a credential definition from the cache if available, otherwise the ledger.

**Parameters** **credential\_definition\_id** – The schema id of the schema to fetch cred def for

**abstract async get\_endpoint\_for\_did**(*did: str, endpoint\_type: aries\_cloudagent.ledger.endpoint\_type.EndpointType = EndpointType.ENDPOINT*) → str

Fetch the endpoint for a ledger DID.

**Parameters**

- **did** – The DID to look up on the ledger or in the cache
- **endpoint\_type** – The type of the endpoint (default 'endpoint')

**abstract async get\_key\_for\_did**(*did: str*) → str

Fetch the verkey for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**abstract async get\_latest\_txn\_author\_acceptance**()

Look up the latest TAA acceptance.

**abstract async get\_nym\_role**(*did: str*)

Return the role registered to input public DID on the ledger.

**Parameters** **did** – DID to register on the ledger.

**abstract async get\_revoc\_reg\_def**(*revoc\_reg\_id: str*) → dict

Look up a revocation registry definition by ID.

**abstract async get\_revoc\_reg\_delta**(*revoc\_reg\_id: str, timestamp\_from=0, timestamp\_to=None*) → Tuple[dict, int]

Look up a revocation registry delta by ID.

**abstract async get\_revoc\_reg\_entry**(*revoc\_reg\_id: str, timestamp: int*) → Tuple[dict, int]

Get revocation registry entry by revocation registry ID and timestamp.

**abstract async get\_schema**(*schema\_id: str*) → dict

Get a schema from the cache if available, otherwise fetch from the ledger.

**Parameters** *schema\_id* – The schema id (or stringified sequence number) to retrieve

**abstract async get\_txn\_author\_agreement**(*reload: bool = False*)

Get the current transaction author agreement, fetching it if necessary.

**abstract nym\_to\_did**(*nym: str*) → str

Format a nym with the ledger's DID prefix.

**abstract property read\_only: bool**

Accessor for the ledger read-only flag.

**abstract async register\_nym**(*did: str, verkey: str, alias: Optional[str] = None, role: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*) → Tuple[bool, dict]

Register a nym on the ledger.

**Parameters**

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

**abstract async rotate\_public\_did\_keypair**(*next\_seed: Optional[str] = None*) → None

Rotate keypair for public DID: create new key, submit to ledger, update wallet.

**Parameters** *next\_seed* – seed for incoming ed25519 keypair (default random)

**abstract async send\_revoc\_reg\_def**(*revoc\_reg\_def: dict, issuer\_did: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*)

Publish a revocation registry definition to the ledger.

**abstract async send\_revoc\_reg\_entry**(*revoc\_reg\_id: str, revoc\_def\_type: str, revoc\_reg\_entry: dict, issuer\_did: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*)

Publish a revocation registry entry to the ledger.

**taa\_digest**(*version: str, text: str*)

Generate the digest of a TAA record.

**abstract async txn\_endorse**(*request\_json: str, endorse\_did: Optional[aries\_cloudagent.wallet.did\_info.DIDInfo] = None*) → str

Endorse (sign) the provided transaction.

**abstract async txn\_submit**(*request\_json: str, sign: bool, taa\_accept: bool, sign\_did: aries\_cloudagent.wallet.did\_info.DIDInfo = <object object>*) → str

Write the provided (signed and possibly endorsed) transaction to the ledger.

```
abstract async update_endpoint_for_did(did: str, endpoint: str, endpoint_type:
    aries_cloudagent.ledger.endpoint_type.EndpointType =
    EndpointType.ENDPOINT, write_ledger: bool = True,
    endorser_did: Optional[str] = None) → bool
```

Check and update the endpoint on the ledger.

#### Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address
- **endpoint\_type** – The type of the endpoint (default 'endpoint')

```
class aries_cloudagent.ledger.base.Role(value)
```

Bases: `enum.Enum`

Enum for indy roles.

```
ENDORSER = (101,)
```

```
NETWORK_MONITOR = (201,)
```

```
ROLE_REMOVE = ('',)
```

```
STEWARD = (2,)
```

```
TRUSTEE = (0,)
```

```
USER = (None, '')
```

```
static get(token: Optional[Union[str, int]] = None) → aries_cloudagent.ledger.base.Role
```

Return enum instance corresponding to input token.

**Parameters token** – token identifying role to indy-sdk: “STEWARD”, “TRUSTEE”, “ENDORSER”, “” or None

```
to_indy_num_str() → str
```

Return (typically, numeric) string value that indy-sdk associates with role.

Recall that None signifies USER and “” signifies a role undergoing reset.

```
token() → str
```

Return token identifying role to indy-sdk.

## aries\_cloudagent.ledger.endpoint\_type module

Ledger utilities.

```
class aries_cloudagent.ledger.endpoint_type.EndpointType(value)
```

Bases: `enum.Enum`

Enum for endpoint/service types.

```
ENDPOINT = EndpointTypeName(w3c='Endpoint', indy='endpoint')
```

```
LINKED_DOMAINS = EndpointTypeName(w3c='LinkedDomains', indy='linked_domains')
```

```
PROFILE = EndpointTypeName(w3c='Profile', indy='profile')
```

```
static get(name: str) → aries_cloudagent.ledger.endpoint_type.EndpointType
```

Return enum instance corresponding to input string.

**property indy**

internally-facing, on ledger and in wallet.



**Type** Indy name of endpoint type

**property w3c**

externally-facing.

**Type** W3C name of endpoint type

**class** `aries_cloudagent.ledger.endpoint_type.EndpointTypeName(w3c, indy)`

Bases: `tuple`

**property indy**

Alias for field number 1

**property w3c**

Alias for field number 0

## `aries_cloudagent.ledger.error` module

Ledger related errors.

**exception** `aries_cloudagent.ledger.error.BadLedgerRequestError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.ledger.error.LedgerError`

The current request cannot proceed.

**exception** `aries_cloudagent.ledger.error.ClosedPoolError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.ledger.error.LedgerError`

Indy pool is closed.

**exception** `aries_cloudagent.ledger.error.LedgerConfigError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.ledger.error.LedgerError`

Base class for ledger configuration errors.

**exception** `aries_cloudagent.ledger.error.LedgerError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Base class for ledger errors.

**exception** `aries_cloudagent.ledger.error.LedgerTransactionError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.ledger.error.LedgerError`

The ledger rejected the transaction.

## aries\_cloudagent.ledger.indy module

Indy ledger implementation.

```
class aries_cloudagent.ledger.indy.IndySdkLedger(pool:
                                             aries_cloudagent.ledger.indy.IndySdkLedgerPool,
                                             profile: aries_cloudagent.core.profile.Profile)
```

Bases: *aries\_cloudagent.ledger.base.BaseLedger*

Indy ledger class.

**BACKEND\_NAME:** `str` = 'indy'

```
async accept_txn_author_agreement(taa_record: dict, mechanism: str, accept_time: Optional[int] =
                                   None)
```

Save a new record recording the acceptance of the TAA.

```
async build_and_return_get_nym_request(submitter_id: Optional[str], target_id: str) → str
Build GET_NYM request and return request_json.
```

```
async check_existing_schema(public_id: str, schema_name: str, schema_version: str, attribute_names:
                             Sequence[str]) → Tuple[str, dict]
```

Check if a schema has already been published.

```
async create_and_send_credential_definition(issuer: aries_cloudagent.indy.issuer.IndyIssuer,
                                             schema_id: str, signature_type: Optional[str] =
                                             None, tag: Optional[str] = None,
                                             support_revocation: bool = False, write_ledger:
                                             bool = True, endorser_id: Optional[str] = None)
                                             → Tuple[str, dict, bool]
```

Send credential definition to ledger and store relevant key matter in wallet.

### Parameters

- **issuer** – The issuer instance to use for credential definition creation
- **schema\_id** – The schema id of the schema to create cred def for
- **signature\_type** – The signature type to use on the credential definition
- **tag** – Optional tag to distinguish multiple credential definitions
- **support\_revocation** – Optional flag to enable revocation for this cred def

**Returns** Tuple with cred def id, cred def structure, and whether it's novel

```
async create_and_send_schema(issuer: aries_cloudagent.indy.issuer.IndyIssuer, schema_name: str,
                             schema_version: str, attribute_names: Sequence[str], write_ledger: bool
                             = True, endorser_id: Optional[str] = None) → Tuple[str, dict]
```

Send schema to ledger.

### Parameters

- **issuer** – The issuer instance creating the schema
- **schema\_name** – The schema name
- **schema\_version** – The schema version
- **attribute\_names** – A list of schema attributes

```
async credential_definition_id2schema_id(credential_definition_id)
```

From a credential definition, get the identifier for its schema.

**Parameters** `credential_definition_id` – The identifier of the credential definition from which to identify a schema

**async** `fetch_credential_definition(credential_definition_id: str) → dict`

Get a credential definition from the ledger by id.

**Parameters** `credential_definition_id` – The cred def id of the cred def to fetch

**async** `fetch_schema_by_id(schema_id: str) → dict`

Get schema from ledger.

**Parameters** `schema_id` – The schema id (or stringified sequence number) to retrieve

**Returns** Indy schema dict

**async** `fetch_schema_by_seq_no(seq_no: int)`

Fetch a schema by its sequence number.

**Parameters** `seq_no` – schema ledger sequence number

**Returns** Indy schema dict

**async** `fetch_txn_author_agreement() → dict`

Fetch the current AML and TAA from the ledger.

**async** `get_all_endpoints_for_did(did: str) → dict`

Fetch all endpoints for a ledger DID.

**Parameters** `did` – The DID to look up on the ledger or in the cache

**async** `get_credential_definition(credential_definition_id: str) → dict`

Get a credential definition from the cache if available, otherwise the ledger.

**Parameters** `credential_definition_id` – The schema id of the schema to fetch cred def for

**async** `get_endpoint_for_did(did: str, endpoint_type:`

*Optional[aries\_cloudagent.ledger.endpoint\_type.EndpointType] = None)*

`→ str`

Fetch the endpoint for a ledger DID.

**Parameters**

- `did` – The DID to look up on the ledger or in the cache
- `endpoint_type` – The type of the endpoint. If none given, returns all

**async** `get_indy_storage() → aries_cloudagent.storage.indy.IndySdkStorage`

Get an IndySdkStorage instance for the current wallet.

**async** `get_key_for_did(did: str) → str`

Fetch the verkey for a ledger DID.

**Parameters** `did` – The DID to look up on the ledger or in the cache

**async** `get_latest_txn_author_acceptance() → dict`

Look up the latest TAA acceptance.

**async** `get_nym_role(did: str) → aries_cloudagent.ledger.base.Role`

Return the role of the input public DID's NYM on the ledger.

**Parameters** `did` – DID to query for role on the ledger.

**async** `get_revoc_reg_def(revoc_reg_id: str) → dict`

Get revocation registry definition by ID; augment with ledger timestamp.

**async get\_revoc\_reg\_delta**(*revoc\_reg\_id: str, fro=0, to=None*) → Tuple[dict, int]

Look up a revocation registry delta by ID.

:param revoc\_reg\_id revocation registry id :param fro earliest EPOCH time of interest :param to latest EPOCH time of interest

:returns delta response, delta timestamp

**async get\_revoc\_reg\_entry**(*revoc\_reg\_id: str, timestamp: int*)

Get revocation registry entry by revocation registry ID and timestamp.

**async get\_schema**(*schema\_id: str*) → dict

Get a schema from the cache if available, otherwise fetch from the ledger.

**Parameters** *schema\_id* – The schema id (or stringified sequence number) to retrieve

**async get\_txn\_author\_agreement**(*reload: bool = False*) → dict

Get the current transaction author agreement, fetching it if necessary.

**async get\_wallet\_public\_did**() → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Fetch the public DID from the wallet.

**nym\_to\_did**(*nym: str*) → str

Format a nym with the ledger's DID prefix.

**property pool\_handle**

Accessor for the ledger pool handle.

**property pool\_name: str**

Accessor for the ledger pool name.

**property read\_only: bool**

Accessor for the ledger read-only flag.

**async register\_nym**(*did: str, verkey: str, alias: Optional[str] = None, role: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*) → Tuple[bool, dict]

Register a nym on the ledger.

**Parameters**

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

**async rotate\_public\_did\_keypair**(*next\_seed: Optional[str] = None*) → None

Rotate keypair for public DID: create new key, submit to ledger, update wallet.

**Parameters** *next\_seed* – seed for incoming ed25519 keypair (default random)

**async send\_revoc\_reg\_def**(*revoc\_reg\_def: dict, issuer\_did: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*)

Publish a revocation registry definition to the ledger.

**async send\_revoc\_reg\_entry**(*revoc\_reg\_id: str, revoc\_def\_type: str, revoc\_reg\_entry: dict, issuer\_did: Optional[str] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*)

Publish a revocation registry entry to the ledger.

**async submit\_get\_nym\_request**(*request\_json: str*) → str

Submit GET\_NYM request to ledger and return response\_json.

**taa\_rough\_timestamp()** → *int*

Get a timestamp accurate to the day.

Anything more accurate is a privacy concern.

**async txn\_endorse**(*request\_json: str, endorse\_did: Optional[aries\_cloudagent.wallet.did\_info.DIDInfo] = None*) → *str*

Endorse a (signed) ledger transaction.

**async txn\_submit**(*request\_json: str, sign: Optional[bool] = None, taa\_accept: Optional[bool] = None, sign\_did: aries\_cloudagent.wallet.did\_info.DIDInfo = <object object>*) → *str*

Submit a signed (and endorsed) transaction to the ledger.

**async update\_endpoint\_for\_did**(*did: str, endpoint: str, endpoint\_type: Optional[aries\_cloudagent.ledger.endpoint\_type.EndpointType] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*) → *bool*

Check and update the endpoint on the ledger.

#### Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address
- **endpoint\_type** – The type of the endpoint

**class aries\_cloudagent.ledger.indy.IndySdkLedgerPool**(*name: str, \*, checked: bool = False, keepalive: int = 0, cache: Optional[aries\_cloudagent.cache.base.BaseCache] = None, cache\_duration: int = 600, genesis\_transactions: Optional[str] = None, read\_only: bool = False, socks\_proxy: Optional[str] = None*)

Bases: *object*

Indy ledger manager class.

**async check\_pool\_config**() → *bool*

Check if a pool config has been created.

**async close**()

Close the pool ledger.

**async context\_close**()

Release the reference and schedule closing of the pool ledger.

**async context\_open**()

Open the ledger if necessary and increase the number of active references.

**async create\_pool\_config**(*genesis\_transactions: str, recreate: bool = False*)

Create the pool ledger configuration.

**property genesis\_txns: str**

Get the configured genesis transactions.

**async open**()

Open the pool ledger, creating it if necessary.

**class aries\_cloudagent.ledger.indy.IndySdkLedgerPoolProvider**

Bases: *aries\_cloudagent.config.base.BaseProvider*

Indy ledger pool provider which keys off the selected pool name.

```
provide(settings: aries_cloudagent.config.base.BaseSettings, injector:
    aries_cloudagent.config.base.BaseInjector)
    Create and open the pool instance.
```

## aries\_cloudagent.ledger.indy\_vdr module

Indy-VDR ledger implementation.

```
class aries_cloudagent.ledger.indy_vdr.IndyVdrLedger(pool:
    aries_cloudagent.ledger.indy_vdr.IndyVdrLedgerPool,
    profile: aries_cloudagent.core.profile.Profile)
```

Bases: *aries\_cloudagent.ledger.base.BaseLedger*

Indy-VDR ledger class.

```
BACKEND_NAME: str = 'indy-vdr'
```

```
async accept_txn_author_agreement(taa_record: dict, mechanism: str, accept_time: Optional[int] =
    None)
```

Save a new record recording the acceptance of the TAA.

```
async build_and_return_get_nym_request(submitter_did: Optional[str], target_did: str) → str
    Build GET_NYM request and return request_json.
```

```
async check_existing_schema(public_did: str, schema_name: str, schema_version: str, attribute_names:
    Sequence[str]) → Tuple[str, dict]
```

Check if a schema has already been published.

```
async create_and_send_credential_definition(issuer: aries_cloudagent.indy.issuer.IndyIssuer,
    schema_id: str, signature_type: Optional[str] =
    None, tag: Optional[str] = None,
    support_revocation: bool = False, write_ledger:
    bool = True, endorser_did: Optional[str] = None)
    → Tuple[str, dict, bool]
```

Send credential definition to ledger and store relevant key matter in wallet.

### Parameters

- **issuer** – The issuer instance to use for credential definition creation
- **schema\_id** – The schema id of the schema to create cred def for
- **signature\_type** – The signature type to use on the credential definition
- **tag** – Optional tag to distinguish multiple credential definitions
- **support\_revocation** – Optional flag to enable revocation for this cred def

**Returns** Tuple with cred def id, cred def structure, and whether it's novel

```
async create_and_send_schema(issuer: aries_cloudagent.indy.issuer.IndyIssuer, schema_name: str,
    schema_version: str, attribute_names: Sequence[str], write_ledger: bool
    = True, endorser_did: Optional[str] = None) → Tuple[str, dict]
```

Send schema to ledger.

### Parameters

- **issuer** – The issuer instance creating the schema
- **schema\_name** – The schema name
- **schema\_version** – The schema version

- **attribute\_names** – A list of schema attributes

**async credential\_definition\_id2schema\_id**(*credential\_definition\_id*)

From a credential definition, get the identifier for its schema.

**Parameters** **credential\_definition\_id** – The identifier of the credential definition from which to identify a schema

**async fetch\_credential\_definition**(*credential\_definition\_id: str*) → dict

Get a credential definition from the ledger by id.

**Parameters** **credential\_definition\_id** – The cred def id of the cred def to fetch

**async fetch\_schema\_by\_id**(*schema\_id: str*) → dict

Get schema from ledger.

**Parameters** **schema\_id** – The schema id (or stringified sequence number) to retrieve

**Returns** Indy schema dict

**async fetch\_schema\_by\_seq\_no**(*seq\_no: int*)

Fetch a schema by its sequence number.

**Parameters** **seq\_no** – schema ledger sequence number

**Returns** Indy schema dict

**async fetch\_txn\_author\_agreement**() → dict

Fetch the current AML and TAA from the ledger.

**async get\_all\_endpoints\_for\_did**(*did: str*) → dict

Fetch all endpoints for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**async get\_credential\_definition**(*credential\_definition\_id: str*) → dict

Get a credential definition from the cache if available, otherwise the ledger.

**Parameters** **credential\_definition\_id** – The schema id of the schema to fetch cred def for

**async get\_endpoint\_for\_did**(*did: str, endpoint\_type:*

*Optional[aries\_cloudagent.ledger.endpoint\_type.EndpointType] = None*) → str

Fetch the endpoint for a ledger DID.

**Parameters**

- **did** – The DID to look up on the ledger or in the cache
- **endpoint\_type** – The type of the endpoint. If none given, returns all

**async get\_key\_for\_did**(*did: str*) → str

Fetch the verkey for a ledger DID.

**Parameters** **did** – The DID to look up on the ledger or in the cache

**async get\_latest\_txn\_author\_acceptance**() → dict

Look up the latest TAA acceptance.

**async get\_nym\_role**(*did: str*) → *aries\_cloudagent.ledger.base.Role*

Return the role of the input public DID's NYM on the ledger.

**Parameters** **did** – DID to query for role on the ledger.

**async get\_revoc\_reg\_def**(*revoc\_reg\_id: str*) → dict

Get revocation registry definition by ID.

**async get\_revoc\_reg\_delta**(*revoc\_reg\_id*: *str*, *timestamp\_from*=0, *timestamp\_to*=None) → Tuple[dict, int]

Look up a revocation registry delta by ID.

:param *revoc\_reg\_id* revocation registry id :param *timestamp\_from* from time. a total number of seconds from Unix Epoch :param *timestamp\_to* to time. a total number of seconds from Unix Epoch

:returns delta response, delta timestamp

**async get\_revoc\_reg\_entry**(*revoc\_reg\_id*: *str*, *timestamp*: *int*) → Tuple[dict, int]

Get revocation registry entry by revocation registry ID and timestamp.

**async get\_schema**(*schema\_id*: *str*) → dict

Get a schema from the cache if available, otherwise fetch from the ledger.

**Parameters** *schema\_id* – The schema id (or stringified sequence number) to retrieve

**async get\_txn\_author\_agreement**(*reload*: *bool* = False) → dict

Get the current transaction author agreement, fetching it if necessary.

**async get\_wallet\_public\_did**() → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Fetch the public DID from the wallet.

**nym\_to\_did**(*nym*: *str*) → *str*

Format a nym with the ledger's DID prefix.

**property pool\_handle**

Accessor for the ledger pool handle.

**property pool\_name**: *str*

Accessor for the ledger pool name.

**property read\_only**: *bool*

Accessor for the ledger read-only flag.

**async register\_nym**(*did*: *str*, *verkey*: *str*, *alias*: Optional[*str*] = None, *role*: Optional[*str*] = None, *write\_ledger*: *bool* = True, *endorser\_did*: Optional[*str*] = None) → Tuple[bool, dict]

Register a nym on the ledger.

**Parameters**

- **did** – DID to register on the ledger.
- **verkey** – The verification key of the keypair.
- **alias** – Human-friendly alias to assign to the DID.
- **role** – For permissioned ledgers, what role should the new DID have.

**async rotate\_public\_did\_keypair**(*next\_seed*: Optional[*str*] = None) → None

Rotate keypair for public DID: create new key, submit to ledger, update wallet.

**Parameters** *next\_seed* – seed for incoming ed25519 keypair (default random)

**async send\_revoc\_reg\_def**(*revoc\_reg\_def*: dict, *issuer\_did*: Optional[*str*] = None, *write\_ledger*: *bool* = True, *endorser\_did*: Optional[*str*] = None)

Publish a revocation registry definition to the ledger.

**async send\_revoc\_reg\_entry**(*revoc\_reg\_id*: *str*, *revoc\_def\_type*: *str*, *revoc\_reg\_entry*: dict, *issuer\_did*: Optional[*str*] = None, *write\_ledger*: *bool* = True, *endorser\_did*: Optional[*str*] = None)

Publish a revocation registry entry to the ledger.

**async submit\_get\_nym\_request**(*request\_json*: *str*) → *str*

Submit GET\_NYM request to ledger and return response\_json.



**taa\_rough\_timestamp()** → int

Get a timestamp accurate to the day.

Anything more accurate is a privacy concern.

**async txn\_endorse**(request\_json: str, endorse\_did: Optional[aries\_cloudagent.wallet.did\_info.DIDInfo] = None) → str

Endorse (sign) the provided transaction.

**async txn\_submit**(request\_json: str, sign: bool, taa\_accept: bool, sign\_did: aries\_cloudagent.wallet.did\_info.DIDInfo = <object object>) → str

Write the provided (signed and possibly endorsed) transaction to the ledger.

**async update\_endpoint\_for\_did**(did: str, endpoint: str, endpoint\_type: Optional[aries\_cloudagent.ledger.endpoint\_type.EndpointType] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None) → bool

Check and update the endpoint on the ledger.

#### Parameters

- **did** – The ledger DID
- **endpoint** – The endpoint address
- **endpoint\_type** – The type of the endpoint

**class** aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedgerPool(name: str, \*, keepalive: int = 0, cache: Optional[aries\_cloudagent.cache.base.BaseCache] = None, cache\_duration: int = 600, genesis\_transactions: Optional[str] = None, read\_only: bool = False, socks\_proxy: Optional[str] = None)

Bases: object

Indy-VDR ledger pool manager.

**property** cfg\_path: pathlib.Path

Get the path to the configuration file, ensuring it's created.

**async** close()

Close the pool ledger.

**async** context\_close()

Release the reference and schedule closing of the pool ledger.

**async** context\_open()

Open the ledger if necessary and increase the number of active references.

**async** create\_pool\_config(genesis\_transactions: str, recreate: bool = False)

Create the pool ledger configuration.

**property** genesis\_hash: str

Get the hash of the configured genesis transactions.

**property** genesis\_txns: str

Get the configured genesis transactions.

**async** open()

Open the pool ledger, creating it if necessary.

### `aries_cloudagent.ledger.routes` module

### `aries_cloudagent.ledger.util` module

Ledger utilities.

**`async aries_cloudagent.ledger.util.notify_register_did_event`**(*profile*:  
`aries_cloudagent.core.profile.Profile`,  
*did*: *str*, *meta\_data*: *dict*)

Send notification for a DID post-process event.

### `aries_cloudagent.messaging` package

#### Subpackages

#### `aries_cloudagent.messaging.credential_definitions` package

##### Submodules

#### `aries_cloudagent.messaging.credential_definitions.routes` module

#### `aries_cloudagent.messaging.credential_definitions.util` module

#### `aries_cloudagent.messaging.decorators` package

##### Submodules

#### `aries_cloudagent.messaging.decorators.attach_decorator` module

#### `aries_cloudagent.messaging.decorators.base` module

#### `aries_cloudagent.messaging.decorators.default` module

#### `aries_cloudagent.messaging.decorators.localization_decorator` module

#### `aries_cloudagent.messaging.decorators.please_ack_decorator` module

#### `aries_cloudagent.messaging.decorators.service_decorator` module

#### `aries_cloudagent.messaging.decorators.signature_decorator` module

#### `aries_cloudagent.messaging.decorators.thread_decorator` module

#### `aries_cloudagent.messaging.decorators.timing_decorator` module

#### `aries_cloudagent.messaging.decorators.trace_decorator` module

**aries\_cloudagent.messaging.decorators.transport\_decorator module****aries\_cloudagent.messaging.jsonld package****Submodules****aries\_cloudagent.messaging.jsonld.create\_verify\_data module**

Contains the functions needed to produce and verify a json-ld signature.

This file was ported from <https://github.com/transmute-industries/Ed25519Signature2018/blob/master/src/createVerifyData/index.js>

```
aries_cloudagent.messaging.jsonld.create_verify_data.create_verify_data(data,
                                                                    signature_options,
                                                                    docu-
                                                                    ment_loader=None)
```

Encapsulate process of constructing string used during sign and verify.

**aries\_cloudagent.messaging.jsonld.credential module****aries\_cloudagent.messaging.jsonld.error module**

JSON-LD messaging Exceptions.

```
exception aries_cloudagent.messaging.jsonld.error.BadJWSHeaderError(*args, error_code:
                                                                    Optional[str] = None,
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError*

Exception indicating invalid JWS header.

```
exception aries_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError(*args, error_code:
                                                                    Optional[str] =
                                                                    None, **kwargs)
```

Bases: *aries\_cloudagent.core.error.BaseError*

Base exception class for JSON-LD messaging.

```
exception aries_cloudagent.messaging.jsonld.error.DroppedAttributeError(*args, error_code:
                                                                    Optional[str] = None,
                                                                    **kwargs)
```

Bases: *aries\_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError*

Exception used to track that an attribute was removed.

```
exception aries_cloudagent.messaging.jsonld.error.InvalidVerificationMethod(*args,
                                                                    error_code:
                                                                    Optional[str] =
                                                                    None, **kwargs)
```

Bases: *aries\_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError*

Exception indicating an invalid verification method in doc to verify.

**exception** `aries_cloudagent.messaging.jsonld.error.MissingVerificationMethodError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError`

Exception indicating missing verification method from signature options.

**exception** `aries_cloudagent.messaging.jsonld.error.SignatureTypeError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.messaging.jsonld.error.BaseJSONLDMessagingError`

Exception indicating Signature type error.

### `aries_cloudagent.messaging.jsonld.routes` module

### `aries_cloudagent.messaging.models` package

Common code for messaging models.

#### Submodules

#### `aries_cloudagent.messaging.models.base` module

#### `aries_cloudagent.messaging.models.base_record` module

#### `aries_cloudagent.messaging.models.openapi` module

#### `aries_cloudagent.messaging.schemas` package

#### Submodules

#### `aries_cloudagent.messaging.schemas.routes` module

#### `aries_cloudagent.messaging.schemas.util` module

#### Submodules

#### `aries_cloudagent.messaging.agent_message` module

#### `aries_cloudagent.messaging.base_handler` module

#### `aries_cloudagent.messaging.base_message` module

Base message.

**class** aries\_cloudagent.messaging.base\_message.BaseMessage

Bases: `abc.ABC`

Abstract base class for messages.

This formally defines a “minimum viable message” and provides an unopinionated class for plugins to extend in whatever way makes sense in the context of the plugin.

**abstract property** Handler: `Type[BaseHandler]`

Return reference to handler class.

**abstract classmethod** deserialize(*value: dict, msg\_format:*

`aries_cloudagent.messaging.base_message.DIDCommVersion =  
DIDCommVersion.v1`)

Return message object deserialized from value in format specified.

**abstract** serialize(*msg\_format: aries\_cloudagent.messaging.base\_message.DIDCommVersion =  
DIDCommVersion.v1*) → `dict`

Return serialized message in format specified.

**class** aries\_cloudagent.messaging.base\_message.DIDCommVersion(*value*)

Bases: `enum.Enum`

Serialized message formats.

**v1** = 1

**v2** = 2

## aries\_cloudagent.messaging.error module

Messaging-related error classes and codes.

**exception** aries\_cloudagent.messaging.error.MessageParseError(*\*args, error\_code: Optional[str] =  
None, \*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message parse error.

**error\_code** = 'message\_parse\_error'

**exception** aries\_cloudagent.messaging.error.MessagePrepareError(*\*args, error\_code: Optional[str] =  
None, \*\*kwargs*)

Bases: `aries_cloudagent.core.error.BaseError`

Message preparation error.

**error\_code** = 'message\_prepare\_error'

## aries\_cloudagent.messaging.request\_context module

## aries\_cloudagent.messaging.responder module

## aries\_cloudagent.messaging.util module

Utils for messages.

**aries\_cloudagent.messaging.util.canon**(*raw\_attr\_name: str*) → `str`

Canonicalize input attribute name for indy proofs and credential offers.

**Parameters** `raw_attr_name` – raw attribute name

**Returns** canonicalized attribute name

`aries_cloudagent.messaging.util.datetime_now()` → `datetime.datetime`  
Timestamp in UTC.

`aries_cloudagent.messaging.util.datetime_to_str(dt: Union[str, datetime.datetime])` → `str`  
Convert a datetime object to an indy-standard datetime string.

**Parameters** `dt` – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.encode(orig: Any)` → `str`  
Encode a credential value as an int.

Encode credential attribute value, purely stringifying any int32 and leaving numeric int32 strings alone, but mapping any other input to a stringified 256-bit (but not 32-bit) integer. Predicates in indy-sdk operate on int32 values properly only when their encoded values match their raw values.

**Parameters** `orig` – original value to encode

**Returns** encoded value

`aries_cloudagent.messaging.util.epoch_to_str(epoch: int)` → `str`  
Convert epoch seconds to indy-standard datetime string.

**Parameters** `epoch` – epoch seconds

`aries_cloudagent.messaging.util.str_to_datetime(dt: Union[str, datetime.datetime])` → `datetime.datetime`

Convert an indy-standard datetime string to a datetime.

Using a fairly lax regex pattern to match slightly different formats. In Python 3.7 `datetime.fromisoformat` might be used.

**Parameters** `dt` – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.str_to_epoch(dt: Union[str, datetime.datetime])` → `int`  
Convert an indy-standard datetime string to epoch seconds.

**Parameters** `dt` – May be a string or datetime to allow automatic conversion

`aries_cloudagent.messaging.util.time_now()` → `str`  
Timestamp in ISO format.

### `aries_cloudagent.messaging.valid` module

### `aries_cloudagent.multitenant` package

#### Subpackages

### `aries_cloudagent.multitenant.admin` package

#### Submodules

### `aries_cloudagent.multitenant.admin.routes` module

#### Submodules

## aries\_cloudagent.multitenant.askar\_profile\_manager module

## aries\_cloudagent.multitenant.base module

## aries\_cloudagent.multitenant.cache module

Cache for multitenancy profiles.

**class** aries\_cloudagent.multitenant.cache.ProfileCache(capacity: *int*)

Bases: `object`

Profile cache that caches based on LRU strategy.

**get**(key: *str*) → Optional[aries\_cloudagent.core.profile.Profile]

Get profile with associated key from cache.

If a profile is open but has been evicted from the cache, this will reinsert the profile back into the cache. This prevents attempting to open a profile that is already open. Triggers clean up.

**Parameters** **key** (*str*) – the key to get the profile for.

**Returns** Profile if found in cache.

**Return type** Optional[Profile]

**has**(key: *str*) → *bool*

Check whether there is a profile with associated key in the cache.

**Parameters** **key** (*str*) – the key to check for a profile

**Returns** Whether the key exists in the cache

**Return type** *bool*

**put**(key: *str*, value: aries\_cloudagent.core.profile.Profile) → *None*

Add profile with associated key to the cache.

If new profile exceeds the cache capacity least recently used profiles that are not used will be removed from the cache.

**Parameters**

- **key** (*str*) – the key to set
- **value** (*Profile*) – the profile to set

**remove**(key: *str*)

Remove profile with associated key from the cache.

**Parameters** **key** (*str*) – The key to remove from the cache.

## aries\_cloudagent.multitenant.error module

Multitenant error classes.

**exception** aries\_cloudagent.multitenant.error.WalletKeyMissingError(\*args, error\_code: Optional[*str*] = *None*, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Wallet key missing exception.

### `aries_cloudagent.multitenant.manager` module

### `aries_cloudagent.multitenant.manager_provider` module

Profile manager for multitenancy.

```
class aries_cloudagent.multitenant.manager_provider.MultitenantManagerProvider(root_profile)
    Bases: aries_cloudagent.config.base.BaseProvider
```

Multitenant manager provider.

Decides which manager to use based on the settings.

```
MANAGER_TYPES = {'askar-profile':
    'aries_cloudagent.multitenant.askar_profile_manager.AskarProfileMultitenantManager',
    'basic': 'aries_cloudagent.multitenant.manager.MultitenantManager'}
```

```
askar_profile_manager_path =
    'aries_cloudagent.multitenant.askar_profile_manager.AskarProfileMultitenantManager'
```

```
provide(settings: aries_cloudagent.config.base.BaseSettings, injector:
    aries_cloudagent.config.base.BaseInjector)
```

Create the multitenant manager instance.

### `aries_cloudagent.protocols` package

#### Subpackages

#### `aries_cloudagent.protocols.actionmenu` package

#### Subpackages

#### `aries_cloudagent.protocols.actionmenu.v1_0` package

#### Subpackages

#### `aries_cloudagent.protocols.actionmenu.v1_0.handlers` package

#### Submodules

#### `aries_cloudagent.protocols.actionmenu.v1_0.handlers.menu_handler` module

#### `aries_cloudagent.protocols.actionmenu.v1_0.handlers.menu_request_handler` module

#### `aries_cloudagent.protocols.actionmenu.v1_0.handlers.perform_handler` module

#### `aries_cloudagent.protocols.actionmenu.v1_0.messages` package

#### Submodules



`aries_cloudagent.protocols.actionmenu.v1_0.messages.menu` module

`aries_cloudagent.protocols.actionmenu.v1_0.messages.menu_request` module

`aries_cloudagent.protocols.actionmenu.v1_0.messages.perform` module

`aries_cloudagent.protocols.actionmenu.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.actionmenu.v1_0.models.menu_form` module

`aries_cloudagent.protocols.actionmenu.v1_0.models.menu_form_param` module

`aries_cloudagent.protocols.actionmenu.v1_0.models.menu_option` module

#### Submodules

`aries_cloudagent.protocols.actionmenu.v1_0.base_service` module

`aries_cloudagent.protocols.actionmenu.v1_0.controller` module

`aries_cloudagent.protocols.actionmenu.v1_0.driver_service` module

`aries_cloudagent.protocols.actionmenu.v1_0.message_types` module

Message type identifiers for Action Menus.

`aries_cloudagent.protocols.actionmenu.v1_0.routes` module

`aries_cloudagent.protocols.actionmenu.v1_0.util` module

#### Submodules

`aries_cloudagent.protocols.actionmenu.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.basicmessage` package

#### Subpackages

`aries_cloudagent.protocols.basicmessage.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.basicmessage.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.basicmessage.v1_0.handlers.basicmessage_handler` module

`aries_cloudagent.protocols.basicmessage.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.basicmessage.v1_0.messages.basicmessage` module

#### Submodules

`aries_cloudagent.protocols.basicmessage.v1_0.message_types` module

Message type identifiers for Connections.

`aries_cloudagent.protocols.basicmessage.v1_0.routes` module

#### Submodules

`aries_cloudagent.protocols.basicmessage.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.connections` package

#### Subpackages

`aries_cloudagent.protocols.connections.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.connections.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.connections.v1_0.handlers.connection_invitation_handler` module

`aries_cloudagent.protocols.connections.v1_0.handlers.connection_request_handler` module

`aries_cloudagent.protocols.connections.v1_0.handlers.connection_response_handler` module

`aries_cloudagent.protocols.connections.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.connections.v1_0.messages.connection_invitation` module

`aries_cloudagent.protocols.connections.v1_0.messages.connection_request` module

`aries_cloudagent.protocols.connections.v1_0.messages.connection_response` module

`aries_cloudagent.protocols.connections.v1_0.messages.problem_report` module

`aries_cloudagent.protocols.connections.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.connections.v1_0.models.connection_detail` module

#### Submodules

`aries_cloudagent.protocols.connections.v1_0.manager` module

`aries_cloudagent.protocols.connections.v1_0.message_types` module

Message type identifiers for Connections.

`aries_cloudagent.protocols.connections.v1_0.routes` module

#### Submodules

`aries_cloudagent.protocols.connections.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.coordinate_mediation` package

#### Subpackages

`aries_cloudagent.protocols.coordinate_mediation.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.keylist_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.keylist_query_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.keylist_update_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.keylist_update_response_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.mediation_deny_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.mediation_grant_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.mediation_request_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers.problem_report_handler` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages` package

#### Subpackages

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.inner` package

#### Submodules

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.inner.keylist_key` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.inner.keylist_query_paginate` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.inner.keylist_update_rule` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.inner.keylist_updated` module

#### Submodules

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.keylist` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.keylist_query` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.keylist_update` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.keylist_update_response` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.mediate_deny` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.mediate_grant` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.mediate_request` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.problem_report` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.coordinate_mediation.v1_0.models.mediation_record` module

#### Submodules

`aries_cloudagent.protocols.coordinate_mediation.v1_0.controller` module

Protocol controller for coordinate mediation.

**class** `aries_cloudagent.protocols.coordinate_mediation.v1_0.controller.Controller`(*protocol: str*)

Bases: `object`

Coordinate mediation protocol controller.

**determine\_goal\_codes**() → `Sequence[str]`  
Return defined `goal_codes`.

`aries_cloudagent.protocols.coordinate_mediation.v1_0.manager` module

`aries_cloudagent.protocols.coordinate_mediation.v1_0.message_types` module

Message type identifiers for Coordinate Mediation protocol.

`aries_cloudagent.protocols.coordinate_mediation.v1_0.routes` module

### Submodules

`aries_cloudagent.protocols.coordinate_mediation.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.coordinate_mediation.mediation_invite_store` module

Storage management for configuration-provided mediation invite.

Handle storage and retrieval of mediation invites provided through arguments. Enables having the mediation invite config be the same for *provision* and *starting* commands.

**class** `aries_cloudagent.protocols.coordinate_mediation.mediation_invite_store.MediationInviteRecord`(*invite*: *str*, *used*: *bool*)

Bases: `tuple`

A record to store mediation invites and their freshness.

**static from\_json**(*json\_invite\_record*: *str*) → *aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.MediationInviteRecord*

**Returns** a mediation invite record deserialized from a json string.

**property invite**  
Alias for field number 0

**to\_json**() → *str*

**Returns** The current record serialized into a json string.

**static unused**(*invite*: *str*) → *aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.MediationInviteRecord*

**Parameters invite** – invite string as provided by the mediator.

**Returns** An unused mediation invitation for the given invite string

**property used**  
Alias for field number 1

**class** `aries_cloudagent.protocols.coordinate_mediation.mediation_invite_store.MediationInviteStore`(*storage*: *aries\_c*)

Bases: `object`

Store and retrieve mediation invite configuration.

**INVITE\_RECORD\_CATEGORY** = 'config'

**MEDIATION\_INVITE\_ID** = 'mediation\_invite'

**async get\_mediation\_invite\_record**(*provided\_mediation\_invitation: Optional[str]*) → *Optional[aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.MediationInviteRecord]*

Provide the MediationInviteRecord to use/that was used for mediation.

Returned record may have been used already.

Stored record is updated if *provided\_mediation\_invitation* has changed. Updated record is marked as unused.

**Parameters** **provided\_mediation\_invitation** – mediation invite provided by user

**Returns** mediation invite to use/that was used to connect to the mediator. None if no invitation was provided/provisioned.

**async mark\_default\_invite\_as\_used**()

Mark the currently stored invitation as used if one exists.

**Raises** **NoDefaultMediationInviteException** – if trying to mark invite as used when there is no invite stored.

**async store**(*mediation\_invite:*

*aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.MediationInviteRecord*) →

*aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.MediationInviteRecord*

Store the mediator's invite for further use when starting the agent.

Update the currently stored invite if one already exists. This assumes a new invite and as such, marks it as unused.

**Parameters** **mediation\_invite** – mediation invite url

**Returns** stored mediation invite

**exception** **aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store.**

**NoDefaultMediationInviteException**

Bases: **Exception**

Raised if trying to mark a default invite as used when none exist.

## aries\_cloudagent.protocols.didexchange package

### Subpackages

#### aries\_cloudagent.protocols.didexchange.v1\_0 package

### Subpackages

#### aries\_cloudagent.protocols.didexchange.v1\_0.handlers package

### Submodules

#### aries\_cloudagent.protocols.didexchange.v1\_0.handlers.complete\_handler module

`aries_cloudagent.protocols.didexchange.v1_0.handlers.invitation_handler` module

`aries_cloudagent.protocols.didexchange.v1_0.handlers.request_handler` module

`aries_cloudagent.protocols.didexchange.v1_0.handlers.response_handler` module

`aries_cloudagent.protocols.didexchange.v1_0.messages` package

### Submodules

`aries_cloudagent.protocols.didexchange.v1_0.messages.complete` module

`aries_cloudagent.protocols.didexchange.v1_0.messages.problem_report_reason` module

DID Exchange problem report reasons.

```
class aries_cloudagent.protocols.didexchange.v1_0.messages.problem_report_reason.ProblemReportReason(val
    Bases: enum.Enum
```

Supported reason codes.

```
COMPLETE_NOT_ACCEPTED = 'complete_not_accepted'
```

```
INVITATION_NOT_ACCEPTED = 'invitation_not_accepted'
```

```
REQUEST_NOT_ACCEPTED = 'request_not_accepted'
```

```
REQUEST_PROCESSING_ERROR = 'request_processing_error'
```

```
RESPONSE_NOT_ACCEPTED = 'response_not_accepted'
```

```
RESPONSE_PROCESSING_ERROR = 'response_processing_error'
```

`aries_cloudagent.protocols.didexchange.v1_0.messages.request` module

`aries_cloudagent.protocols.didexchange.v1_0.messages.response` module

### Submodules

`aries_cloudagent.protocols.didexchange.v1_0.manager` module

`aries_cloudagent.protocols.didexchange.v1_0.message_types` module

Message type identifiers for Connections.



`aries_cloudagent.protocols.didexchange.v1_0.routes` module

#### Submodules

`aries_cloudagent.protocols.didexchange.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.discovery` package

#### Subpackages

`aries_cloudagent.protocols.discovery.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.discovery.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.discovery.v1_0.handlers.disclose_handler` module

`aries_cloudagent.protocols.discovery.v1_0.handlers.query_handler` module

`aries_cloudagent.protocols.discovery.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.discovery.v1_0.messages.disclose` module

`aries_cloudagent.protocols.discovery.v1_0.messages.query` module

`aries_cloudagent.protocols.discovery.v1_0.models` package

Package-wide code and data.

#### Submodules

`aries_cloudagent.protocols.discovery.v1_0.models.discovery_record` module

#### Submodules

`aries_cloudagent.protocols.discovery.v1_0.manager` module

### `aries_cloudagent.protocols.discovery.v1_0.message_types` module

Message type identifiers for Feature Discovery.

### `aries_cloudagent.protocols.discovery.v1_0.routes` module

### `aries_cloudagent.protocols.discovery.v2_0` package

#### Subpackages

### `aries_cloudagent.protocols.discovery.v2_0.handlers` package

#### Submodules

### `aries_cloudagent.protocols.discovery.v2_0.handlers.disclosures_handler` module

### `aries_cloudagent.protocols.discovery.v2_0.handlers.queries_handler` module

### `aries_cloudagent.protocols.discovery.v2_0.messages` package

#### Submodules

### `aries_cloudagent.protocols.discovery.v2_0.messages.disclosures` module

### `aries_cloudagent.protocols.discovery.v2_0.messages.queries` module

### `aries_cloudagent.protocols.discovery.v2_0.models` package

Package-wide code and data.

#### Submodules

### `aries_cloudagent.protocols.discovery.v2_0.models.discovery_record` module

#### Submodules

### `aries_cloudagent.protocols.discovery.v2_0.manager` module

### `aries_cloudagent.protocols.discovery.v2_0.message_types` module

Message type identifiers for Feature Discovery.

`aries_cloudagent.protocols.discovery.v2_0.routes` module

#### Submodules

`aries_cloudagent.protocols.discovery.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.endorse_transaction` package

#### Subpackages

`aries_cloudagent.protocols.endorse_transaction.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.endorsed_transaction_response_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.refused_transaction_response_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.transaction_acknowledgement_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.transaction_cancel_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.transaction_job_to_send_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.transaction_request_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.handlers.transaction_resend_handler` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.cancel_transaction` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.endorsed_transaction_response` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.messages_attach` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.refused_transaction_response` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.transaction_acknowledgement` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.transaction_job_to_send` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.transaction_request` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.messages.transaction_resend` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.models` package

### Submodules

`aries_cloudagent.protocols.endorse_transaction.v1_0.models.transaction_record` module

### Submodules

`aries_cloudagent.protocols.endorse_transaction.v1_0.controller` module

Protocol controller for endorse transaction.

**class** `aries_cloudagent.protocols.endorse_transaction.v1_0.controller.Controller`(*protocol: str*)

Bases: `object`

Endorse transaction protocol controller.

**determine\_goal\_codes**() → Sequence[str]

Return defined goal\_codes.

`aries_cloudagent.protocols.endorse_transaction.v1_0.manager` module

`aries_cloudagent.protocols.endorse_transaction.v1_0.message_types` module

Message type identifiers for Transactions.

**aries\_cloudagent.protocols.endorse\_transaction.v1\_0.routes module**

**aries\_cloudagent.protocols.endorse\_transaction.v1\_0.transaction\_jobs module**

Class to manage jobs in Connection Record.

```
class aries_cloudagent.protocols.endorse_transaction.v1_0.transaction_jobs.TransactionJob(value)  
    Bases: enum.Enum  
    Represents jobs in Connection Record.  
    TRANSACTION_AUTHOR = (1,)   
    TRANSACTION_ENDORSER = (2,)
```

**aries\_cloudagent.protocols.endorse\_transaction.v1\_0.util module**

**Submodules**

**aries\_cloudagent.protocols.endorse\_transaction.definition module**

Version definitions for this protocol.

**aries\_cloudagent.protocols.introduction package**

**Subpackages**

**aries\_cloudagent.protocols.introduction.v0\_1 package**

**Subpackages**

**aries\_cloudagent.protocols.introduction.v0\_1.handlers package**

**Submodules**

**aries\_cloudagent.protocols.introduction.v0\_1.handlers.forward\_invitation\_handler module**

**aries\_cloudagent.protocols.introduction.v0\_1.handlers.invitation\_handler module**

**aries\_cloudagent.protocols.introduction.v0\_1.handlers.invitation\_request\_handler module**

**aries\_cloudagent.protocols.introduction.v0\_1.messages package**

**Submodules**

**aries\_cloudagent.protocols.introduction.v0\_1.messages.forward\_invitation module**

**aries\_cloudagent.protocols.introduction.v0\_1.messages.invitation module**

`aries_cloudagent.protocols.introduction.v0_1.messages.invitation_request` module

### Submodules

`aries_cloudagent.protocols.introduction.v0_1.base_service` module

`aries_cloudagent.protocols.introduction.v0_1.demo_service` module

`aries_cloudagent.protocols.introduction.v0_1.message_types` module

Message type identifiers for Introductions.

`aries_cloudagent.protocols.introduction.v0_1.routes` module

### Submodules

`aries_cloudagent.protocols.introduction.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.issue_credential` package

### Subpackages

`aries_cloudagent.protocols.issue_credential.v1_0` package

### Subpackages

`aries_cloudagent.protocols.issue_credential.v1_0.handlers` package

### Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_ack_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_issue_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_offer_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_problem_report_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_proposal_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.handlers.credential_request_handler` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages` package

#### Subpackages

`aries_cloudagent.protocols.issue_credential.v1_0.messages.inner` package

#### Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.messages.inner.credential_preview` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_ack` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_issue` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_offer` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_problem_report` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_proposal` module

`aries_cloudagent.protocols.issue_credential.v1_0.messages.credential_request` module

`aries_cloudagent.protocols.issue_credential.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.models.credential_exchange` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.v1_0.controller` module

`aries_cloudagent.protocols.issue_credential.v1_0.manager` module

`aries_cloudagent.protocols.issue_credential.v1_0.message_types` module

`aries_cloudagent.protocols.issue_credential.v1_0.routes` module

`aries_cloudagent.protocols.issue_credential.v2_0` package

#### Subpackages

`aries_cloudagent.protocols.issue_credential.v2_0.formats` package

### Subpackages

`aries_cloudagent.protocols.issue_credential.v2_0.formats.indy` package

### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.formats.indy.handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.formats.ld_proof` package

### Subpackages

`aries_cloudagent.protocols.issue_credential.v2_0.formats.ld_proof.models` package

### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.formats.ld_proof.models.cred_detail` module

`aries_cloudagent.protocols.issue_credential.v2_0.formats.ld_proof.models.cred_detail_options` module

### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.formats.ld_proof.handler` module

### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.formats.handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers` package

### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_ack_handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_issue_handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_offer_handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_problem_report_handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_proposal_handler` module

`aries_cloudagent.protocols.issue_credential.v2_0.handlers.cred_request_handler` module



`aries_cloudagent.protocols.issue_credential.v2_0.messages` package

#### Subpackages

`aries_cloudagent.protocols.issue_credential.v2_0.messages.inner` package

#### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.messages.inner.cred_preview` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_ack` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_format` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_issue` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_offer` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_problem_report` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_proposal` module

`aries_cloudagent.protocols.issue_credential.v2_0.messages.cred_request` module

`aries_cloudagent.protocols.issue_credential.v2_0.models` package

#### Subpackages

`aries_cloudagent.protocols.issue_credential.v2_0.models.detail` package

#### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.models.detail.indy` module

`aries_cloudagent.protocols.issue_credential.v2_0.models.detail.id_proof` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.models.cred_ex_record` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.v2_0.controller` module

`aries_cloudagent.protocols.issue_credential.v2_0.manager` module

`aries_cloudagent.protocols.issue_credential.v2_0.message_types` module

`aries_cloudagent.protocols.issue_credential.v2_0.routes` module

#### Submodules

`aries_cloudagent.protocols.issue_credential.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.notification` package

#### Subpackages

`aries_cloudagent.protocols.notification.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.notification.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.notification.v1_0.handlers.ack_handler` module

`aries_cloudagent.protocols.notification.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.notification.v1_0.messages.ack` module

#### Submodules

`aries_cloudagent.protocols.notification.v1_0.message_types` module

Message and inner object type identifiers for present-proof protocol v2.0.

## Submodules

`aries_cloudagent.protocols.notification.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.out_of_band` package

## Subpackages

`aries_cloudagent.protocols.out_of_band.v1_0` package

## Subpackages

`aries_cloudagent.protocols.out_of_band.v1_0.handlers` package

## Submodules

`aries_cloudagent.protocols.out_of_band.v1_0.handlers.problem_report_handler` module

`aries_cloudagent.protocols.out_of_band.v1_0.handlers.reuse_accept_handler` module

`aries_cloudagent.protocols.out_of_band.v1_0.handlers.reuse_handler` module

`aries_cloudagent.protocols.out_of_band.v1_0.messages` package

## Submodules

`aries_cloudagent.protocols.out_of_band.v1_0.messages.invitation` module

`aries_cloudagent.protocols.out_of_band.v1_0.messages.problem_report` module

`aries_cloudagent.protocols.out_of_band.v1_0.messages.reuse` module

`aries_cloudagent.protocols.out_of_band.v1_0.messages.reuse_accept` module

`aries_cloudagent.protocols.out_of_band.v1_0.messages.service` module

`aries_cloudagent.protocols.out_of_band.v1_0.models` package

## Submodules

`aries_cloudagent.protocols.out_of_band.v1_0.models.invitation` module

`aries_cloudagent.protocols.out_of_band.v1_0.models.oob_record` module

### Submodules

#### `aries_cloudagent.protocols.out_of_band.v1_0.controller` module

Protocol controller for out-of-band.

```
class aries_cloudagent.protocols.out_of_band.v1_0.controller.Controller(protocol: str)
    Bases: object
    Out-of-band protocol controller.
    determine_goal_codes() → Sequence[str]
        Return defined goal_codes.
```

#### `aries_cloudagent.protocols.out_of_band.v1_0.manager` module

#### `aries_cloudagent.protocols.out_of_band.v1_0.message_types` module

Message and inner object type identifiers for Out of Band messages.

#### `aries_cloudagent.protocols.out_of_band.v1_0.routes` module

### Submodules

#### `aries_cloudagent.protocols.out_of_band.definition` module

Version definitions for this protocol.

#### `aries_cloudagent.protocols.present_proof` package

### Subpackages

#### `aries_cloudagent.protocols.present_proof.dif` package

### Submodules

#### `aries_cloudagent.protocols.present_proof.dif.pres_exch` module

#### `aries_cloudagent.protocols.present_proof.dif.pres_exch_handler` module

#### `aries_cloudagent.protocols.present_proof.dif.pres_proposal_schema` module

#### `aries_cloudagent.protocols.present_proof.dif.pres_request_schema` module

#### `aries_cloudagent.protocols.present_proof.dif.pres_schema` module

#### `aries_cloudagent.protocols.present_proof.indy` package

## Submodules

`aries_cloudagent.protocols.present_proof.indy.pres_exch_handler` module

`aries_cloudagent.protocols.present_proof.v1_0` package

## Subpackages

`aries_cloudagent.protocols.present_proof.v1_0.handlers` package

## Submodules

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_ack_handler` module

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_handler` module

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_problem_report_handler` module

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_proposal_handler` module

`aries_cloudagent.protocols.present_proof.v1_0.handlers.presentation_request_handler` module

`aries_cloudagent.protocols.present_proof.v1_0.messages` package

## Submodules

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation` module

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_ack` module

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_problem_report` module

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_proposal` module

`aries_cloudagent.protocols.present_proof.v1_0.messages.presentation_request` module

`aries_cloudagent.protocols.present_proof.v1_0.models` package

## Submodules

`aries_cloudagent.protocols.present_proof.v1_0.models.presentation_exchange` module

## Submodules

`aries_cloudagent.protocols.present_proof.v1_0.controller` module

`aries_cloudagent.protocols.present_proof.v1_0.manager` module

`aries_cloudagent.protocols.present_proof.v1_0.message_types` module

`aries_cloudagent.protocols.present_proof.v1_0.routes` module

`aries_cloudagent.protocols.present_proof.v2_0` package

#### Subpackages

`aries_cloudagent.protocols.present_proof.v2_0.formats` package

#### Subpackages

`aries_cloudagent.protocols.present_proof.v2_0.formats.dif` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.formats.dif.handler` module

`aries_cloudagent.protocols.present_proof.v2_0.formats.indy` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.formats.indy.handler` module

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.formats.handler` module

`aries_cloudagent.protocols.present_proof.v2_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.handlers.pres_ack_handler` module

`aries_cloudagent.protocols.present_proof.v2_0.handlers.pres_handler` module

`aries_cloudagent.protocols.present_proof.v2_0.handlers.pres_problem_report_handler` module

`aries_cloudagent.protocols.present_proof.v2_0.handlers.pres_proposal_handler` module

`aries_cloudagent.protocols.present_proof.v2_0.handlers.pres_request_handler` module

`aries_cloudagent.protocols.present_proof.v2_0.messages` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres` module

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres_ack` module

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres_format` module

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres_problem_report` module

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres_proposal` module

`aries_cloudagent.protocols.present_proof.v2_0.messages.pres_request` module

`aries_cloudagent.protocols.present_proof.v2_0.models` package

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.models.pres_exchange` module

#### Submodules

`aries_cloudagent.protocols.present_proof.v2_0.controller` module

`aries_cloudagent.protocols.present_proof.v2_0.manager` module

`aries_cloudagent.protocols.present_proof.v2_0.message_types` module

`aries_cloudagent.protocols.present_proof.v2_0.routes` module

#### Submodules

`aries_cloudagent.protocols.present_proof.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.problem_report` package

#### Subpackages

`aries_cloudagent.protocols.problem_report.v1_0` package

#### Submodules

`aries_cloudagent.protocols.problem_report.v1_0.handler` module

`aries_cloudagent.protocols.problem_report.v1_0.message` module

`aries_cloudagent.protocols.problem_report.v1_0.message_types` module

#### Submodules

`aries_cloudagent.protocols.problem_report.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.revocation_notification` package

#### Subpackages

`aries_cloudagent.protocols.revocation_notification.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.revocation_notification.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.revocation_notification.v1_0.handlers.revoke_handler` module

`aries_cloudagent.protocols.revocation_notification.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.revocation_notification.v1_0.messages.revoke` module

`aries_cloudagent.protocols.revocation_notification.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.revocation_notification.v1_0.models.rev_notification_record` module



## Submodules

**`aries_cloudagent.protocols.revocation_notification.v1_0.message_types` module**

Message type identifiers for Revocation Notification protocol.

**`aries_cloudagent.protocols.revocation_notification.v1_0.routes` module**

**`aries_cloudagent.protocols.revocation_notification.v2_0` package**

## Subpackages

**`aries_cloudagent.protocols.revocation_notification.v2_0.handlers` package**

## Submodules

**`aries_cloudagent.protocols.revocation_notification.v2_0.handlers.revoke_handler` module**

**`aries_cloudagent.protocols.revocation_notification.v2_0.messages` package**

## Submodules

**`aries_cloudagent.protocols.revocation_notification.v2_0.messages.revoke` module**

**`aries_cloudagent.protocols.revocation_notification.v2_0.models` package**

## Submodules

**`aries_cloudagent.protocols.revocation_notification.v2_0.models.rev_notification_record` module**

## Submodules

**`aries_cloudagent.protocols.revocation_notification.v2_0.message_types` module**

Message type identifiers for Revocation Notification protocol.

**`aries_cloudagent.protocols.revocation_notification.v2_0.routes` module**

## Submodules

**`aries_cloudagent.protocols.revocation_notification.definition` module**

Version definitions for this protocol.

`aries_cloudagent.protocols.routing` package

#### Subpackages

`aries_cloudagent.protocols.routing.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.routing.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.routing.v1_0.handlers.forward_handler` module

`aries_cloudagent.protocols.routing.v1_0.handlers.route_query_request_handler` module

`aries_cloudagent.protocols.routing.v1_0.handlers.route_query_response_handler` module

`aries_cloudagent.protocols.routing.v1_0.handlers.route_update_request_handler` module

`aries_cloudagent.protocols.routing.v1_0.handlers.route_update_response_handler` module

`aries_cloudagent.protocols.routing.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.routing.v1_0.messages.forward` module

`aries_cloudagent.protocols.routing.v1_0.messages.route_query_request` module

`aries_cloudagent.protocols.routing.v1_0.messages.route_query_response` module

`aries_cloudagent.protocols.routing.v1_0.messages.route_update_request` module

`aries_cloudagent.protocols.routing.v1_0.messages.route_update_response` module

`aries_cloudagent.protocols.routing.v1_0.models` package

#### Submodules

`aries_cloudagent.protocols.routing.v1_0.models.paginate` module

`aries_cloudagent.protocols.routing.v1_0.models.paginated` module

`aries_cloudagent.protocols.routing.v1_0.models.route_query_result` module

`aries_cloudagent.protocols.routing.v1_0.models.route_record` module

`aries_cloudagent.protocols.routing.v1_0.models.route_update` module

`aries_cloudagent.protocols.routing.v1_0.models.route_updated` module

#### Submodules

`aries_cloudagent.protocols.routing.v1_0.manager` module

`aries_cloudagent.protocols.routing.v1_0.message_types` module

Message type identifiers for Routing.

#### Submodules

`aries_cloudagent.protocols.routing.definition` module

Version definitions for this protocol.

`aries_cloudagent.protocols.trustping` package

#### Subpackages

`aries_cloudagent.protocols.trustping.v1_0` package

#### Subpackages

`aries_cloudagent.protocols.trustping.v1_0.handlers` package

#### Submodules

`aries_cloudagent.protocols.trustping.v1_0.handlers.ping_handler` module

`aries_cloudagent.protocols.trustping.v1_0.handlers.ping_response_handler` module

`aries_cloudagent.protocols.trustping.v1_0.messages` package

#### Submodules

`aries_cloudagent.protocols.trustping.v1_0.messages.ping` module

`aries_cloudagent.protocols.trustping.v1_0.messages.ping_response` module

#### Submodules

### `aries_cloudagent.protocols.trustping.v1_0.message_types` module

Message type identifiers for Trust Pings.

### `aries_cloudagent.protocols.trustping.v1_0.routes` module

#### Submodules

### `aries_cloudagent.protocols.trustping.definition` module

Version definitions for this protocol.

#### Submodules

### `aries_cloudagent.protocols.didcomm_prefix` module

DIDComm prefix management.

**class** `aries_cloudagent.protocols.didcomm_prefix.DIDCommPrefix(value)`

Bases: `enum.Enum`

Enum for DIDComm Prefix, old or new style, per Aries RFC 384.

**NEW** = `'https://didcomm.org'`

**OLD** = `'did:sov:BzCbsNYhMrjHiqZDTUASHg;spec'`

**qualify**(*msg\_type: Optional[str] = None*) → `str`

Qualify input message type with prefix and separator.

**classmethod** **qualify\_all**(*messages: dict*) → `dict`

Apply all known prefixes to a dictionary of message types.

**static** **qualify\_current**(*slug: Optional[str] = None*) → `str`

Qualify input slug with prefix currently in effect and separator.

**static** **set**(*settings: Mapping*)

Set current DIDComm prefix value in environment.

**static** **unqualify**(*qual: str*) → `str`

Strip prefix and separator from input, if present, and return result.

`aries_cloudagent.protocols.didcomm_prefix.qualify(msg_type: str, prefix: str)`

Qualify a message type with a prefix, if unqualified.

### `aries_cloudagent.resolver` package

Interfaces and base classes for DID Resolution.

**async** `aries_cloudagent.resolver.setup(context:`

`aries_cloudagent.config.injection_context.InjectionContext)`

Set up default resolvers.

## Subpackages

### `aries_cloudagent.resolver.default` package

Resolvers included in ACA-Py by Default.

## Submodules

### `aries_cloudagent.resolver.default.indy` module

### `aries_cloudagent.resolver.default.key` module

### `aries_cloudagent.resolver.default.web` module

## Submodules

### `aries_cloudagent.resolver.base` module

Base Class for DID Resolvers.

```
class aries_cloudagent.resolver.base.BaseDIDResolver(type_: Optional[aries_cloudagent.resolver.base.ResolverType]  
                                                    = None)
```

Bases: `abc.ABC`

Base Class for DID Resolvers.

#### **property native**

Return if this resolver is native.

**async resolve**(*profile*: `aries_cloudagent.core.profile.Profile`, *did*: `Union[str, pydid.DID]`) → `dict`  
Resolve a DID using this resolver.

**abstract async setup**(*context*: `aries_cloudagent.config.injection_context.InjectionContext`)  
Do asynchronous resolver setup.

#### **property supported\_did\_regex: Pattern**

Supported DID regex for matching this resolver to DIDs it can resolve.

Override this property with a class var or similar to use regex matching on DIDs to determine if this resolver supports a given DID.

#### **property supported\_methods: Sequence[str]**

Return supported methods.

DEPRECATED: Use `supported_did_regex` instead.

**async supports**(*profile*: `aries_cloudagent.core.profile.Profile`, *did*: `str`) → `bool`  
Return if this resolver supports the given DID.

Override this method to determine if this resolver supports a DID based on information other than just a regular expression; i.e. check a value in storage, query a resolver connection record, etc.

```
exception aries_cloudagent.resolver.base.DIDMethodNotSupported(*args, error_code: Optional[str]  
                                                             = None, **kwargs)
```

Bases: `aries_cloudagent.resolver.base.ResolverError`

Raised when no resolver is registered for a given did method.

```
exception aries_cloudagent.resolver.base.DIDNotFound(*args, error_code: Optional[str] = None,
**kwargs)
```

Bases: `aries_cloudagent.resolver.base.ResolverError`

Raised when DID is not found in verifiable data registry.

```
class aries_cloudagent.resolver.base.ResolutionMetadata(resolver_type:
    aries_cloudagent.resolver.base.ResolverType,
    resolver: str, retrieved_time: str, duration:
    int)
```

Bases: `tuple`

Resolution Metadata.

**property duration**

Alias for field number 3

**property resolver**

Alias for field number 1

**property resolver\_type**

Alias for field number 0

**property retrieved\_time**

Alias for field number 2

**serialize()** → `dict`

Return serialized resolution metadata.

```
class aries_cloudagent.resolver.base.ResolutionResult(did_document: dict, metadata:
    aries_cloudagent.resolver.base.ResolutionMetadata)
```

Bases: `object`

Resolution Class to pack the DID Doc and the resolution information.

**serialize()** → `dict`

Return serialized resolution result.

```
exception aries_cloudagent.resolver.base.ResolverError(*args, error_code: Optional[str] = None,
**kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base class for resolver exceptions.

```
class aries_cloudagent.resolver.base.ResolverType(value)
```

Bases: `enum.Enum`

Resolver Type declarations.

**NATIVE** = 'native'

**NON\_NATIVE** = 'non-native'

## aries\_cloudagent.resolver.did\_resolver module

the did resolver.

responsible for keeping track of all resolvers. more importantly retrieving did's from different sources provided by the method type.

**class** aries\_cloudagent.resolver.did\_resolver.DIDResolver(*registry:*  
aries\_cloudagent.resolver.did\_resolver\_registry.DIDResolverRegistry)

Bases: `object`

did resolver singleton.

**async** dereference(*profile:* aries\_cloudagent.core.profile.Profile, *did\_url:* str, \*, *cls:*  
Type[aries\_cloudagent.resolver.did\_resolver.ResourceType] = pydid.Resource) →  
aries\_cloudagent.resolver.did\_resolver.ResourceType

Dereference a DID URL to its corresponding DID Doc object.

**async** resolve(*profile:* aries\_cloudagent.core.profile.Profile, *did:* Union[str, pydid.DID]) → dict  
Resolve a DID.

**async** resolve\_with\_metadata(*profile:* aries\_cloudagent.core.profile.Profile, *did:* Union[str, pydid.DID])  
→ aries\_cloudagent.resolver.base.ResolutionResult  
Resolve a DID and return the ResolutionResult.

## aries\_cloudagent.resolver.did\_resolver\_registry module

In memmory storage for registering did resolvers.

**class** aries\_cloudagent.resolver.did\_resolver\_registry.DIDResolverRegistry  
Bases: `object`

Registry for did resolvers.

**register**(*resolver*) → None  
Register a resolver.

**property** resolvers: Sequence[aries\_cloudagent.resolver.base.BaseDIDResolver]  
Accessor for a list of all did resolvers.

## aries\_cloudagent.resolver.routes module

## aries\_cloudagent.revocation package

### Subpackages

## aries\_cloudagent.revocation.models package

### Submodules

## aries\_cloudagent.revocation.models.indy module

## aries\_cloudagent.revocation.models.issuer\_cred\_rev\_record module

## aries\_cloudagent.revocation.models.issuer\_rev\_reg\_record module

## aries\_cloudagent.revocation.models.revocation\_registry module

Classes for managing a revocation registry.

```
class aries_cloudagent.revocation.models.revocation_registry.RevocationRegistry(registry_id:
    Optional[str]
    = None, *,
    cred_def_id:
    Optional[str]
    = None,
    issuer_did:
    Optional[str]
    = None,
    max_creds:
    Optional[int]
    = None,
    reg_def_type:
    Optional[str]
    = None,
    tag: Optional[str]
    = None,
    tails_local_path:
    Optional[str]
    = None,
    tails_public_uri:
    Optional[str]
    = None,
    tails_hash:
    Optional[str]
    = None,
    reg_def:
    Optional[dict]
    = None)
```

Bases: `object`

Manage a revocation registry and tails file.

**MAX\_SIZE** = 32768

**MIN\_SIZE** = 4

**property cred\_def\_id:** `str`

Accessor for the credential definition ID.



```

classmethod from_definition(revoc_reg_def: dict, public_def: bool) →
    aries_cloudagent.revocation.models.revocation_registry.RevocationRegistry
    Initialize a revocation registry instance from a definition.

async get_or_fetch_local_tails_path()
    Get the local tails path, retrieving from the remote if necessary.

get_receiving_tails_local_path()
    Make the local path to the tails file we download from remote URI.

has_local_tails_file() → bool
    Test if the tails file exists locally.

property issuer_did: str
    Accessor for the issuer DID.

property max_creds: int
    Accessor for the maximum number of issued credentials.

property reg_def: dict
    Accessor for the revocation registry definition.

property reg_def_type: str
    Accessor for the revocation registry type.

property registry_id: str
    Accessor for the revocation registry ID.

async retrieve_tails()
    Fetch the tails file from the public URI.

property tag: str
    Accessor for the tag part of the revoc. reg. ID.

property tails_hash: str
    Accessor for the tails file hash.

property tails_local_path: str
    Accessor for the tails file local path.

property tails_public_uri: str
    Accessor for the tails file public URI.

```

## Submodules

## aries\_cloudagent.revocation.error module

Revocation error classes.

```
exception aries_cloudagent.revocation.error.RevocationError(*args, error_code: Optional[str] = None, **kwargs)
```

Bases: `aries_cloudagent.core.error.BaseError`

Base exception for revocation-related errors.

[illegible]

Bases: `aries_cloudagent.revocation.error.RevocationError`

Attempted to perform revocation-related operation where inapplicable.

**exception** `aries_cloudagent.revocation.error.RevocationRegistryBadSizeError`(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.revocation.error.RevocationError`

Attempted to create registry with maximum credentials too large or too small.

### `aries_cloudagent.revocation.indy` module

### `aries_cloudagent.revocation.manager` module

### `aries_cloudagent.revocation.recover` module

Recover a revocation registry.

`aries_cloudagent.revocation.recover.LOGGER` = <Logger `aries_cloudagent.revocation.recover` (WARNING)>

This module calculates a new ledger accumulator, based on the revocation status on the ledger vs revocations recorded in the wallet. The calculated transaction can be written to the ledger to get the ledger back in sync with the wallet. This function can be used if there were previous revocation errors (i.e. the credential revocation was successfully written to the wallet but the ledger write failed.)

**exception** `aries_cloudagent.revocation.recover.RevocRecoveryException`

Bases: `Exception`

Raise exception generating the recovery transaction.

**async** `aries_cloudagent.revocation.recover.fetch_txns`(genesis\_txns, registry\_id)

Fetch tails file and revocation registry information.

**async** `aries_cloudagent.revocation.recover.generate_ledger_rrrecovery_txn`(genesis\_txns, registry\_id, set\_revoked)

Generate a new ledger accum entry, based on wallet vs ledger revocation state.

### `aries_cloudagent.revocation.routes` module

### `aries_cloudagent.revocation.util` module

### `aries_cloudagent.storage` package

#### Subpackages

#### `aries_cloudagent.storage.vc_holder` package

#### Submodules

#### `aries_cloudagent.storage.vc_holder.askar` module

#### `aries_cloudagent.storage.vc_holder.base` module

[aries\\_cloudagent.storage.vc\\_holder.in\\_memory module](#)

[aries\\_cloudagent.storage.vc\\_holder.indy module](#)

[aries\\_cloudagent.storage.vc\\_holder.vc\\_record module](#)

[aries\\_cloudagent.storage.vc\\_holder.xform module](#)

## Submodules

[aries\\_cloudagent.storage.askar module](#)

[aries\\_cloudagent.storage.base module](#)

Abstract base classes for non-secrets storage.

**class** `aries_cloudagent.storage.base.BaseStorage`

Bases: `abc.ABC`

Abstract stored records interface.

**abstract async** `add_record(record: aries_cloudagent.storage.record.StorageRecord)`

Add a new record to the store.

**Parameters** `record` – *StorageRecord* to be stored

**abstract async** `delete_all_records(type_filter: str, tag_query: Optional[Mapping] = None)`

Remove all records matching a particular type filter and tag query.

**abstract async** `delete_record(record: aries_cloudagent.storage.record.StorageRecord)`

Delete an existing record.

**Parameters** `record` – *StorageRecord* to delete

**abstract async** `find_all_records(type_filter: str, tag_query: Optional[Mapping] = None, options: Optional[Mapping] = None)`

Retrieve all records matching a particular type filter and tag query.

**async** `find_record(type_filter: str, tag_query: Optional[Mapping] = None, options: Optional[Mapping] = None) → aries_cloudagent.storage.record.StorageRecord`

Find a record using a unique tag filter.

**Parameters**

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **options** – Dictionary of backend-specific options

**abstract async** `get_record(record_type: str, record_id: str, options: Optional[Mapping] = None) → aries_cloudagent.storage.record.StorageRecord`

Fetch a record from the store by type and ID.

**Parameters**

- **record\_type** – The record type
- **record\_id** – The record id
- **options** – A dictionary of backend-specific options

**Returns** A *StorageRecord* instance

**abstract async update\_record**(*record*: aries\_cloudagent.storage.record.StorageRecord, *value*: str, *tags*: Mapping)

Update an existing stored record's value and tags.

**Parameters**

- **record** – *StorageRecord* to update
- **value** – The new value
- **tags** – The new tags

**class** aries\_cloudagent.storage.base.BaseStorageSearch

Bases: abc.ABC

Abstract stored records search interface.

**abstract search\_records**(*type\_filter*: str, *tag\_query*: Optional[Mapping] = None, *page\_size*: Optional[int] = None, *options*: Optional[Mapping] = None) → aries\_cloudagent.storage.base.BaseStorageSearchSession

Create a new record query.

**Parameters**

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **page\_size** – Page size
- **options** – Dictionary of backend-specific options

**Returns** An instance of *BaseStorageSearchSession*

**class** aries\_cloudagent.storage.base.BaseStorageSearchSession

Bases: abc.ABC

Abstract stored records search session interface.

**async close**()

Dispose of the search query.

**abstract async fetch**(*max\_count*: Optional[int] = None) → Sequence[aries\_cloudagent.storage.record.StorageRecord]

Fetch the next list of results from the store.

**Parameters** **max\_count** – Max number of records to return. If not provided, defaults to the backend's preferred page size

**Returns** A list of *StorageRecord* instances

**class** aries\_cloudagent.storage.base.IterSearch(*search*: aries\_cloudagent.storage.base.BaseStorageSearchSession, *page\_size*: Optional[int] = None)

Bases: object

A generic record search async iterator.

aries\_cloudagent.storage.base.validate\_record(*record*: aries\_cloudagent.storage.record.StorageRecord, \*, *delete*=False)

Ensure that a record is ready to be saved/updated/deleted.

## aries\_cloudagent.storage.error module

Storage-related exceptions.

**exception** aries\_cloudagent.storage.error.StorageDuplicateError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.storage.error.StorageError*

Duplicate record found in storage.

**exception** aries\_cloudagent.storage.error.StorageError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.core.error.BaseError*

Base class for Storage errors.

**exception** aries\_cloudagent.storage.error.StorageNotFoundError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.storage.error.StorageError*

Record not found in storage.

**exception** aries\_cloudagent.storage.error.StorageSearchError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.storage.error.StorageError*

General exception during record search.

## aries\_cloudagent.storage.in\_memory module

## aries\_cloudagent.storage.indy module

Indy implementation of BaseStorage interface.

**class** aries\_cloudagent.storage.indy.IndySdkStorage(wallet: aries\_cloudagent.indy.sdk.wallet\_setup.IndyOpenWallet)

Bases: *aries\_cloudagent.storage.base.BaseStorage*, *aries\_cloudagent.storage.base.BaseStorageSearch*

Indy Non-Secrets interface.

**async** add\_record(record: aries\_cloudagent.storage.record.StorageRecord)

Add a new record to the store.

**Parameters** record – *StorageRecord* to be stored

**async** delete\_all\_records(type\_filter: str, tag\_query: Optional[Mapping] = None)

Remove all records matching a particular type filter and tag query.

**async** delete\_record(record: aries\_cloudagent.storage.record.StorageRecord)

Delete a record.

**Parameters** record – *StorageRecord* to delete

**Raises**

- **StorageNotFoundError** – If record not found
- **StorageError** – If a libindy error occurs

**async find\_all\_records**(*type\_filter*: *str*, *tag\_query*: *Optional[Mapping] = None*, *options*: *Optional[Mapping] = None*)

Retrieve all records matching a particular type filter and tag query.

**async get\_record**(*record\_type*: *str*, *record\_id*: *str*, *options*: *Optional[Mapping] = None*) → *aries\_cloudagent.storage.record.StorageRecord*

Fetch a record from the store by type and ID.

#### Parameters

- **record\_type** – The record type
- **record\_id** – The record id
- **options** – A dictionary of backend-specific options

**Returns** A *StorageRecord* instance

#### Raises

- **StorageError** – If the record is not provided
- **StorageError** – If the record ID not provided
- **StorageNotFoundError** – If the record is not found
- **StorageError** – If record not found

**search\_records**(*type\_filter*: *str*, *tag\_query*: *Optional[Mapping] = None*, *page\_size*: *Optional[int] = None*, *options*: *Optional[Mapping] = None*) → *aries\_cloudagent.storage.indy.IndySdkStorageSearch*

Search stored records.

#### Parameters

- **type\_filter** – Filter string
- **tag\_query** – Tags to query
- **page\_size** – Page size
- **options** – Dictionary of backend-specific options

**Returns** An instance of *IndySdkStorageSearch*

**async update\_record**(*record*: *aries\_cloudagent.storage.record.StorageRecord*, *value*: *str*, *tags*: *Mapping*)

Update an existing stored record's value and tags.

#### Parameters

- **record** – *StorageRecord* to update
- **value** – The new value
- **tags** – The new tags

#### Raises

- **StorageNotFoundError** – If record not found
- **StorageError** – If a libindy error occurs

**property wallet**: *aries\_cloudagent.indy.sdk.wallet\_setup.IndyOpenWallet*

Accessor for *IndyOpenWallet* instance.

```
class aries_cloudagent.storage.indy.IndySdkStorageSearch(store:
    aries_cloudagent.storage.indy.IndySdkStorage,
    type_filter: str, tag_query: Mapping,
    page_size: Optional[int] = None, options:
    Optional[Mapping] = None)

Bases: aries_cloudagent.storage.base.BaseStorageSearchSession

Represent an active stored records search.

async close()
    Dispose of the search query.

async fetch(max_count: Optional[int] = None) →
    Sequence[aries_cloudagent.storage.record.StorageRecord]
    Fetch the next list of results from the store.

Parameters max_count – Max number of records to return. If not provided, defaults to the
    backend's preferred page size

Returns A list of StorageRecord instances

Raises StorageSearchError – If the search query has not been opened
```

## aries\_cloudagent.storage.record module

Record instance stored and searchable by BaseStorage implementation.

```
class aries_cloudagent.storage.record.StorageRecord(type, value, tags: Optional[dict] = None, id:
    Optional[str] = None)

Bases: aries_cloudagent.storage.record.StorageRecord

Storage record class.
```

## aries\_cloudagent.tails package

### Submodules

### aries\_cloudagent.tails.base module

Tails server interface base class.

```
class aries_cloudagent.tails.base.BaseTailsServer
    Bases: abc.ABC

    Base class for tails server interface.

abstract async upload_tails_file(context:
    aries_cloudagent.config.injection_context.InjectionContext,
    rev_reg_id: str, tails_file_path: str, interval: float = 1.0, backoff:
    float = 0.25, max_attempts: int = 5) → Tuple[bool, str]

    Upload tails file to tails server.

Parameters
    • rev_reg_id – The revocation registry identifier
    • tails_file – The path to the tails file to upload
    • interval – initial interval between attempts
```

- **backoff** – exponential backoff in retry interval
- **max\_attempts** – maximum number of attempts to make

### **aries\_cloudagent.tails.error module**

Tails server related errors.

**exception** `aries_cloudagent.tails.error.TailsServerNotConfiguredError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Error indicating the tails server plugin hasn't been configured.

### **aries\_cloudagent.tails.indy\_tails\_server module**

### **aries\_cloudagent.transport package**

#### **Subpackages**

#### **aries\_cloudagent.transport.inbound package**

#### **Submodules**

#### **aries\_cloudagent.transport.inbound.base module**

#### **aries\_cloudagent.transport.inbound.delivery\_queue module**

#### **aries\_cloudagent.transport.inbound.http module**

#### **aries\_cloudagent.transport.inbound.manager module**

#### **aries\_cloudagent.transport.inbound.message module**

Classes representing inbound messages.

**class** `aries_cloudagent.transport.inbound.message.InboundMessage(payload: Union[str, bytes], receipt: aries_cloudagent.transport.inbound.receipt.Message *, connection_id: Optional[str] = None, session_id: Optional[str] = None, transport_type: Optional[str] = None)`

Bases: `object`

Container class linking a message payload with its receipt details.



**aries\_cloudagent.transport.inbound.receipt module**

Classes for representing message receipt details.

```
class aries_cloudagent.transport.inbound.receipt.MessageReceipt(*, connection_id: Optional[str]
                                                                = None, direct_response_mode:
                                                                Optional[str] = None, in_time:
                                                                Optional[datetime.datetime] =
                                                                None, raw_message:
                                                                Optional[str] = None,
                                                                recipient_verkey: Optional[str]
                                                                = None, recipient_did:
                                                                Optional[str] = None,
                                                                recipient_did_public:
                                                                Optional[bool] = None,
                                                                sender_did: Optional[str] =
                                                                None, sender_verkey:
                                                                Optional[str] = None, thread_id:
                                                                Optional[str] = None,
                                                                parent_thread_id: Optional[str]
                                                                = None)
```

Bases: `object`

Properties of an agent message's delivery.

**REPLY\_MODE\_ALL** = 'all'

**REPLY\_MODE\_NONE** = 'none'

**REPLY\_MODE\_THREAD** = 'thread'

**property connection\_id:** `str`  
 Accessor for the pairwise connection identifier.

**Returns** This context's connection identifier

**property direct\_response\_mode:** `str`  
 Accessor for the requested direct response mode.

**Returns** This context's requested direct response mode

**property direct\_response\_requested:** `str`  
 Accessor for the the state of the direct response mode.

**Returns** This context's requested direct response mode

**property in\_time:** `str`  
 Accessor for the datetime the message was received.

**Returns** This context's received time

**property parent\_thread\_id:** `Optional[str]`  
 Accessor for the identifier of the message parent thread.

**Returns** The delivery parent thread ID

**property raw\_message:** `str`  
 Accessor for the raw message text.

**Returns** The raw message text

**property recipient\_did:** `str`  
 Accessor for the recipient DID which corresponds with the verkey.

**Returns** The recipient DID

**property recipient\_did\_public:** `bool`

Check if the recipient did is public.

Indicates whether the message is associated with a public (ledger) recipient DID.

**Returns** True if the recipient's DID is public, else false

**property recipient\_verkey:** `str`

Accessor for the recipient verkey key used to pack the incoming request.

**Returns** The recipient verkey

**property sender\_did:** `str`

Accessor for the sender DID which corresponds with the verkey.

**Returns** The sender did

**property sender\_verkey:** `str`

Accessor for the sender public key used to pack the incoming request.

**Returns** This context's sender's verkey

**property thread\_id:** `str`

Accessor for the identifier of the message thread.

**Returns** The delivery thread ID

## `aries_cloudagent.transport.inbound.session` module

## `aries_cloudagent.transport.inbound.ws` module

## `aries_cloudagent.transport.outbound` package

### Submodules

## `aries_cloudagent.transport.outbound.base` module

Base outbound transport.

**class** `aries_cloudagent.transport.outbound.base.BaseOutboundTransport`(*wire\_format: Optional[aries\_cloudagent.transport.wire\_format.WireFormat] = None, root\_profile: Optional[aries\_cloudagent.core.profile.Profile] = None*)

Bases: `abc.ABC`

Base outbound transport class.

**property collector:** `aries_cloudagent.utils.stats.Collector`

Accessor for the stats collector instance.

**abstract async handle\_message**(*profile: aries\_cloudagent.core.profile.Profile, payload: Union[str, bytes], endpoint: str, metadata: Optional[dict] = None*)

Handle message.

**Parameters**

- **profile** – the profile that produced the message

- **payload** – message payload in string or byte format
- **endpoint** – URI endpoint for delivery
- **metadata** – Additional metadata associated with the payload

**abstract async start()**

Start the transport.

**abstract async stop()**

Shut down the transport.

**property wire\_format:** *aries\_cloudagent.transport.wire\_format.BaseWireFormat*

Accessor for a custom wire format for the transport.

**exception** *aries\_cloudagent.transport.outbound.base.OutboundDeliveryError*(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.transport.outbound.base.OutboundTransportError*

Base exception when a message cannot be delivered via an outbound transport.

**exception** *aries\_cloudagent.transport.outbound.base.OutboundTransportError*(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.transport.error.TransportError*

Generic outbound transport error.

**exception** *aries\_cloudagent.transport.outbound.base.OutboundTransportRegistrationError*(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: *aries\_cloudagent.transport.outbound.base.OutboundTransportError*

Outbound transport registration error.

## **aries\_cloudagent.transport.outbound.http module**

Http outbound transport.

**class** *aries\_cloudagent.transport.outbound.http.HttpTransport*(\*\*kwargs)  
Bases: *aries\_cloudagent.transport.outbound.base.BaseOutboundTransport*

Http outbound transport class.

**async handle\_message**(profile: *aries\_cloudagent.core.profile.Profile*, payload: Union[str, bytes], endpoint: str, metadata: Optional[dict] = None, api\_key: Optional[str] = None)

Handle message from queue.

### **Parameters**

- **profile** – the profile that produced the message
- **payload** – message payload in string or byte format
- **endpoint** – URI endpoint for delivery

- **metadata** – Additional metadata associated with the payload

```

is_external = False
schemes = ('http', 'https')
async start()
    Start the transport.
async stop()
    Stop the transport.

```

## aries\_cloudagent.transport.outbound.manager module

## aries\_cloudagent.transport.outbound.message module

## aries\_cloudagent.transport.outbound.status module

Enum representing captured send status of outbound messages.

```

class aries_cloudagent.transport.outbound.status.OutboundSendStatus(value)
    Bases: enum.Enum

    Send status of outbound messages.

    QUEUED_FOR_DELIVERY = 'queued_for_delivery'
    SENT_TO_EXTERNAL_QUEUE = 'sent_to_external_queue'
    SENT_TO_SESSION = 'sent_to_session'
    UNDELIVERABLE = 'undeliverable'
    WAITING_FOR_PICKUP = 'waiting_for_pickup'

    property topic
        Return an event topic associated with a given status.

```

## aries\_cloudagent.transport.outbound.ws module

Websockets outbound transport.

```

class aries_cloudagent.transport.outbound.ws.WsTransport(**kwargs)
    Bases: aries_cloudagent.transport.outbound.base.BaseOutboundTransport

    Websockets outbound transport class.

    async handle_message(profile: aries_cloudagent.core.profile.Profile, payload: Union[str, bytes], endpoint:
        str, metadata: Optional[dict] = None, api_key: Optional[str] = None)
        Handle message from queue.

        Parameters
        • profile – the profile that produced the message
        • payload – message payload in string or byte format
        • endpoint – URI endpoint for delivery
        • metadata – Additional metadata associated with the payload

    is_external = False

```

```
schemes = ('ws', 'wss')

async start()
    Start the outbound transport.

async stop()
    Stop the outbound transport.
```

## aries\_cloudagent.transport.queue package

### Submodules

#### aries\_cloudagent.transport.queue.base module

Abstract message queue.

```
class aries_cloudagent.transport.queue.base.BaseMessageQueue
    Bases: abc.ABC
```

Abstract message queue class.

```
abstract async dequeue(*, timeout: Optional[int] = None)
    Dequeue a message.
```

**Returns** The dequeued message, or None if a timeout occurs

**Raises**

- `asyncio.CancelledError` if the queue has been stopped –
- `asyncio.TimeoutError` if the timeout is reached –

```
abstract async enqueue(message)
    Enqueue a message.
```

**Parameters** `message` – The message to add to the end of the queue

**Raises** `asyncio.CancelledError` if the queue has been stopped –

```
abstract async join()
    Wait for the queue to empty.
```

```
abstract reset()
    Empty the queue and reset the stop event.
```

```
abstract stop()
    Cancel active iteration of the queue.
```

```
abstract task_done()
    Indicate that the current task is complete.
```

## aries\_cloudagent.transport.queue.basic module

Basic in memory queue.

**class** aries\_cloudagent.transport.queue.basic.**BasicMessageQueue**  
Bases: *aries\_cloudagent.transport.queue.base.BaseMessageQueue*

Basic in memory queue implementation class.

**async dequeue**(*\*, timeout: Optional[int] = None*)  
Dequeue a message.

**Returns** The dequeued message, or None if a timeout occurs

**Raises**

- **asyncio.CancelledError** if the queue has been stopped –
- **asyncio.TimeoutError** if the timeout is reached –

**async enqueue**(*message*)  
Enqueue a message.

**Parameters** *message* – The message to add to the end of the queue

**Raises** **asyncio.CancelledError** if the queue has been stopped –

**async join**()  
Wait for the queue to empty.

**make\_queue**()  
Create the queue instance.

**reset**()  
Empty the queue and reset the stop event.

**stop**()  
Cancel active iteration of the queue.

**task\_done**()  
Indicate that the current task is complete.

## Submodules

### aries\_cloudagent.transport.error module

Transport-related error classes and codes.

**exception** aries\_cloudagent.transport.error.**RecipientKeysError**(*\*args, error\_code: Optional[str] = None, \*\*kwargs*)

Bases: *aries\_cloudagent.transport.error.WireFormatError*

Extract recipient keys error.

**exception** aries\_cloudagent.transport.error.**TransportError**(*\*args, error\_code: Optional[str] = None, \*\*kwargs*)

Bases: *aries\_cloudagent.core.error.BaseError*

Base class for all transport errors.

**exception** `aries_cloudagent.transport.error.WireFormatEncodeError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.transport.error.WireFormatError`

Encoding error when packing the wire format.

**error\_code** = 'message\_encode\_error'

**exception** `aries_cloudagent.transport.error.WireFormatError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.transport.error.TransportError`

Base class for wire-format errors.

**exception** `aries_cloudagent.transport.error.WireFormatParseError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.transport.error.WireFormatError`

Parse error when unpacking the wire format.

**error\_code** = 'message\_parse\_error'

## `aries_cloudagent.transport.pack_format` module

## `aries_cloudagent.transport.stats` module

aiohhttp stats collector support.

**class** `aries_cloudagent.transport.stats.StatsTracer(*args: Any, **kwargs: Any)`

Bases: `aiohhttp`.

Attach hooks to client session events and report statistics.

**async** `connection_queued_end(session, context, params)`

Handle the end of a queued connection.

**async** `connection_queued_start(session, context, params)`

Handle the start of a queued connection.

**async** `connection_ready(session, context, params)`

Handle the end of connection acquisition.

**async** `dns_resolvehost_end(session, context, params)`

Handle the end of a DNS resolution.

**async** `dns_resolvehost_start(session, context, params)`

Handle the start of a DNS resolution.

**async** `request_end(session, context, params)`

Handle the end of request.

**async** `request_start(session, context, params)`

Handle the start of a request.

**async** `socket_connect_start(session, context, params)`

Handle the start of a socket connection.

## aries\_cloudagent.transport.wire\_format module

Abstract wire format classes.

**class** aries\_cloudagent.transport.wire\_format.**BaseWireFormat**

Bases: `object`

Abstract messaging wire format.

**abstract async encode\_message**(*session: aries\_cloudagent.core.profile.ProfileSession, message\_json: Union[str, bytes], recipient\_keys: Sequence[str], routing\_keys: Sequence[str], sender\_key: str*) → Union[str, bytes]

Encode an outgoing message for transport.

### Parameters

- **session** – The profile session for providing wallet access
- **message\_json** – The message body to serialize
- **recipient\_keys** – A sequence of recipient verkeys
- **routing\_keys** – A sequence of routing verkeys
- **sender\_key** – The verification key of the sending agent

**Returns** The encoded message

**Raises** **MessageEncodeError** – If the message could not be encoded

**abstract get\_recipient\_keys**(*message\_body: Union[str, bytes]*) → List[str]

Get all recipient keys from a wire message.

**Parameters** **message\_body** – The body of the message

**Returns** List of recipient keys from the message body

**Raises** **RecipientKeysError** – If the recipient keys could not be extracted

**abstract async parse\_message**(*session: aries\_cloudagent.core.profile.ProfileSession, message\_body: Union[str, bytes]*) → Tuple[dict, aries\_cloudagent.transport.inbound.receipt.MessageReceipt]

Deserialize an incoming message and further populate the request context.

### Parameters

- **session** – The profile session for providing wallet access
- **message\_body** – The body of the message

**Returns** A tuple of the parsed message and a message receipt instance

**Raises** **WireFormatParseError** – If the message can't be parsed

**class** aries\_cloudagent.transport.wire\_format.**JsonWireFormat**

Bases: `aries_cloudagent.transport.wire_format.BaseWireFormat`

Unencrypted wire format.

**abstract async encode\_message**(*session: aries\_cloudagent.core.profile.ProfileSession, message\_json: Union[str, bytes], recipient\_keys: Sequence[str], routing\_keys: Sequence[str], sender\_key: str*) → Union[str, bytes]

Encode an outgoing message for transport.

### Parameters

- **session** – The profile session for providing wallet access



- **message\_json** – The message body to serialize
- **recipient\_keys** – A sequence of recipient verkeys
- **routing\_keys** – A sequence of routing verkeys
- **sender\_key** – The verification key of the sending agent

**Returns** The encoded message

**Raises** **MessageEncodeError** – If the message could not be encoded

**get\_recipient\_keys**(*message\_body*: *Union[str, bytes]*) → List[str]

Get all recipient keys from a wire message.

**Parameters** **message\_body** – The body of the message

**Returns** List of recipient keys from the message body

**Raises** **RecipientKeysError** – If the recipient keys could not be extracted

**abstract async parse\_message**(*session*: *aries\_cloudagent.core.profile.ProfileSession*, *message\_body*: *Union[str, bytes]*) → Tuple[dict, *aries\_cloudagent.transport.inbound.receipt.MessageReceipt*]

Deserialize an incoming message and further populate the request context.

**Parameters**

- **session** – The profile session for providing wallet access
- **message\_body** – The body of the message

**Returns** A tuple of the parsed message and a message receipt instance

**Raises** **WireFormatParseError** – If the JSON parsing failed

## aries\_cloudagent.utils package

### Submodules

#### aries\_cloudagent.utils.classloader module

The classloader provides utilities to dynamically load classes and modules.

**class** *aries\_cloudagent.utils.classloader.ClassLoader*

Bases: *object*

Class used to load classes from modules dynamically.

**classmethod** **load\_class**(*class\_name*: *str*, *default\_module*: *Optional[str] = None*, *package*: *Optional[str] = None*)

Resolve a complete class path (ie. *typing.Dict*) to the class itself.

**Parameters**

- **class\_name** – the class name
- **default\_module** – the default module to load, if not part of in the class name
- **package** – the parent package to search for the module

**Returns** The resolved class

**Raises**

- **`ClassNotFoundError`** – If the class could not be resolved at path
- **`ModuleLoadError`** – If there was an error loading the module

**classmethod** `load_module(mod_path: str, package: Optional[str] = None) → module`  
Load a module by its absolute path.

**Parameters**

- **`mod_path`** – the absolute or relative module path
- **`package`** – the parent package to search for the module

**Returns** The resolved module or *None* if the module cannot be found

**Raises** **`ModuleLoadError`** – If there was an error loading the module

**classmethod** `load_subclass_of(base_class: Type, mod_path: str, package: Optional[str] = None)`  
Resolve an implementation of a base path within a module.

**Parameters**

- **`base_class`** – the base class being implemented
- **`mod_path`** – the absolute module path
- **`package`** – the parent package to search for the module

**Returns** The resolved class

**Raises**

- **`ClassNotFoundError`** – If the module or class implementation could not be found
- **`ModuleLoadError`** – If there was an error loading the module

**classmethod** `scan_subpackages(package: str) → Sequence[str]`  
Return a list of sub-packages defined under a named package.

**exception** `aries_cloudagent.utils.classloader.ClassNotFoundError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Class not found error.

**class** `aries_cloudagent.utils.classloader.DeferLoad(cls_path: str)`

Bases: `object`

Helper to defer loading of a class definition.

**property resolved**

Accessor for the resolved class instance.

**exception** `aries_cloudagent.utils.classloader.ModuleLoadError(*args, error_code: Optional[str] = None, **kwargs)`

Bases: `aries_cloudagent.core.error.BaseError`

Module load error.

## aries\_cloudagent.utils.dependencies module

Dependency related util methods.

`aries_cloudagent.utils.dependencies.assert_ursa_bbs_signatures_installed()`

Assert ursa\_bbs\_signatures module is installed.

`aries_cloudagent.utils.dependencies.is_indy_sdk_module_installed()`

Check whether indy (indy-sdk) module is installed.

**Returns** Whether indy (indy-sdk) is installed.

**Return type** `bool`

`aries_cloudagent.utils.dependencies.is_ursa_bbs_signatures_module_installed()`

Check whether ursa\_bbs\_signatures module is installed.

**Returns** Whether ursa\_bbs\_signatures is installed.

**Return type** `bool`

## aries\_cloudagent.utils.env module

Environment utility methods.

`aries_cloudagent.utils.env.storage_path(*subpaths, create: bool = False) → pathlib.Path`

Get the default aca-py home directory.

## aries\_cloudagent.utils.http module

## aries\_cloudagent.utils.jwe module

## aries\_cloudagent.utils.outofband module

## aries\_cloudagent.utils.repeat module

## aries\_cloudagent.utils.stats module

Classes for tracking performance and timing.

**class** `aries_cloudagent.utils.stats.Collector(*, enabled: bool = True, log_path: Optional[str] = None)`

Bases: `object`

Collector for a set of statistics.

**property enabled:** `bool`

Accessor for the collector's enabled property.

**extract** (*groups: Optional*[*Sequence*[*str*]] = None) → `dict`

Extract statistics for a specific set of groups.

**log** (*name: str, duration: float, start: Optional*[*float*] = None)

Log an entry in the statistics if the collector is enabled.

**mark** (\*names)

Make a custom decorator function for adding to the set of groups.

**reset()**

Reset the collector's statistics.

**property results:** `dict`

Accessor for the current set of collected statistics.

**timer(\*groups)**

Create a new timer attached to this collector.

**wrap(obj, prop\_name: Union[str, Sequence[str]], groups: Optional[Sequence[str]] = None, \*, ignore\_missing: bool = False)**

Wrap a method on a class or class instance.

**wrap\_coro(fn, groups: Sequence[str])**

Wrap a coroutine instance to collect timing statistics on execution.

**wrap\_fn(fn, groups: Sequence[str])**

Wrap a function instance to collect timing statistics on execution.

**class aries\_cloudagent.utils.stats.Stats**

Bases: `object`

A collection of statistics.

**extract(names: Optional[Sequence[str]] = None) → dict**

Summarize the stats in a dictionary.

**log(name: str, duration: float)**

Log an entry in the stats.

**class aries\_cloudagent.utils.stats.Timer(collector: aries\_cloudagent.utils.stats.Collector, groups: Sequence[str])**

Bases: `object`

Timer instance for a running task.

**classmethod now()**

Fetch a standard timer value.

**start() → aries\_cloudagent.utils.stats.Timer**

Start the timer.

**stop()**

Stop the timer.

## aries\_cloudagent.utils.task\_queue module

Classes for managing a set of asyncio tasks.

**class aries\_cloudagent.utils.task\_queue.CompletedTask(task: \_asyncio.Task, exc\_info: Tuple, ident: Optional[str] = None, timing: Optional[dict] = None)**

Bases: `object`

Represent the result of a queued task.

**class aries\_cloudagent.utils.task\_queue.PendingTask(coro: Coroutine, complete\_hook: Optional[Callable] = None, ident: Optional[str] = None, task\_future: Optional[\_asyncio.Future] = None, queued\_time: Optional[float] = None)**

Bases: `object`

Represent a task in the queue.

**cancel()**

Cancel the pending task.

**property cancelled**

Accessor for the cancelled property.

**property task: \_asyncio.Task**

Accessor for the task.

**class** aries\_cloudagent.utils.task\_queue.**TaskQueue**(*max\_active: int = 0, timed: bool = False, trace\_fn: Optional[Callable] = None*)

Bases: `object`

A class for managing a set of asyncio tasks.

**add\_active**(*task: \_asyncio.Task, task\_complete: Optional[Callable] = None, ident: Optional[str] = None, timing: Optional[dict] = None*) → `_asyncio.Task`

Register an active async task with an optional completion callback.

#### Parameters

- **task** – The asyncio task instance
- **task\_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task
- **timing** – An optional dictionary of timing information

**add\_pending**(*pending: aries\_cloudagent.utils.task\_queue.PendingTask*)

Add a task to the pending queue.

**Parameters pending** – The *PendingTask* to add to the task queue

**cancel()**

Cancel any pending or active tasks in the queue.

**cancel\_pending()**

Cancel any pending tasks in the queue.

**property cancelled: bool**

Accessor for the cancelled property of the queue.

**async complete**(*timeout: Optional[float] = None, cleanup: bool = True*)

Cancel any pending tasks and wait for, or cancel active tasks.

**completed\_task**(*task: \_asyncio.Task, task\_complete: Callable, ident: str, timing: Optional[dict] = None*)

Clean up after a task has completed and run callbacks.

**property current\_active: int**

Accessor for the current number of active tasks in the queue.

**property current\_pending: int**

Accessor for the current number of pending tasks in the queue.

**property current\_size: int**

Accessor for the total number of tasks in the queue.

**drain()** → `_asyncio.Task`

Start the process to run queued tasks.

**async flush()**

Wait for any active or pending tasks to be completed.

**property max\_active:** `int`

Accessor for the maximum number of active tasks in the queue.

**put**(*coro: Coroutine, task\_complete: Optional[Callable] = None, ident: Optional[str] = None*) →

*aries\_cloudagent.utils.task\_queue.PendingTask*

Add a new task to the queue, delaying execution if busy.

#### Parameters

- **coro** – The coroutine to run
- **task\_complete** – A callback to run on completion
- **ident** – A string identifier for the task

Returns: a future resolving to the asyncio task instance once queued

**property ready:** `bool`

Accessor for the ready property of the queue.

**run**(*coro: Coroutine, task\_complete: Optional[Callable] = None, ident: Optional[str] = None, timing: Optional[dict] = None*) → *\_asyncio.Task*

Start executing a coroutine as an async task, bypassing the pending queue.

#### Parameters

- **coro** – The coroutine to run
- **task\_complete** – An optional callback to run on completion
- **ident** – A string identifier for the task
- **timing** – An optional dictionary of timing information

Returns: the new asyncio task instance

**async wait\_for**(*timeout: float*)

Wait for all queued tasks to complete with a timeout.

**aries\_cloudagent.utils.task\_queue.coro\_ident**(*coro: Coroutine*)

Extract an identifier for a coroutine.

**async aries\_cloudagent.utils.task\_queue.coro\_timed**(*coro: Coroutine, timing: dict*)

Capture timing for a coroutine.

**aries\_cloudagent.utils.task\_queue.task\_exc\_info**(*task: \_asyncio.Task*)

Extract exception info from an asyncio task.

## aries\_cloudagent.utils.tracing module

## aries\_cloudagent.vc package

### Subpackages

## aries\_cloudagent.vc.ld\_proofs package

### Subpackages

## aries\_cloudagent.vc.ld\_proofs.crypto package

## Submodules

`aries_cloudagent.vc.ld_proofs.crypto.key_pair` module

`aries_cloudagent.vc.ld_proofs.crypto.wallet_key_pair` module

`aries_cloudagent.vc.ld_proofs.purposes` package

## Submodules

`aries_cloudagent.vc.ld_proofs.purposes.assertion_proof_purpose` module

Assertion proof purpose class.

```
class aries_cloudagent.vc.ld_proofs.purposes.assertion_proof_purpose.AssertionProofPurpose(*,
                                             date:
                                             Optional[datetime.datetime] =
                                             None,
                                             max_timestamp:
                                             Optional[datetime.datetime] =
                                             None)

Bases:
    aries_cloudagent.vc.ld_proofs.purposes.controller_proof_purpose.
    ControllerProofPurpose

Assertion proof purpose class.

term = 'assertionMethod'
```

`aries_cloudagent.vc.ld_proofs.purposes.authentication_proof_purpose` module

Authentication proof purpose class.

```
class aries_cloudagent.vc.ld_proofs.purposes.authentication_proof_purpose.AuthenticationProofPurpose(*,
                                                    challenge: str,
                                                    length: int,
                                                    domain: str,
                                                    max_time_delta: Optional[datetime] = None,
                                                    max_timestamp: Optional[datetime] = None)

Bases: aries_cloudagent.vc.ld_proofs.purposes.controller_proof_purpose.ControllerProofPurpose

Authentication proof purpose.

term = 'authentication'

update(proof: dict) → dict
    Update poof purpose, challenge and domain on proof.

validate(*, proof: dict, document: dict, suite: LinkedDataProof, verification_method: dict,
         document_loader: Callable[[str, dict], dict]) →
    aries_cloudagent.vc.ld_proofs.validation_result.PurposeResult
    Validate whether challenge and domain are valid.
```

## aries\_cloudagent.vc.ld\_proofs.purposes.controller\_proof\_purpose module

Controller proof purpose class.

```
class aries_cloudagent.vc.ld_proofs.purposes.controller_proof_purpose.ControllerProofPurpose(*,
                                                    term: str,
                                                    date: Optional[datetime] = None,
                                                    max_timestamp: Optional[datetime] = None)

Bases: aries_cloudagent.vc.ld_proofs.purposes.proof_purpose.ProofPurpose

Controller proof purpose class.
```



**validate**(\**proof: dict*, *document: dict*, *suite: LinkedDataProof*, *verification\_method: dict*,  
*document\_loader: Callable[[str, dict], dict]*) →  
*aries\_cloudagent.vc.ld\_proofs.validation\_result.PurposeResult*  
 Validate whether verification method of proof is authorized by controller.

### aries\_cloudagent.vc.ld\_proofs.purposes.credential\_issuance\_purpose module

Credential Issuance proof purpose class.

```
class aries_cloudagent.vc.ld_proofs.purposes.credential_issuance_purpose.CredentialIssuancePurpose(*,
date: Optional[datetime.datetime] = None,
max_timestamp_delta: Optional[datetime.timedelta] = None):
    """
    Bases: aries_cloudagent.vc.ld_proofs.purposes.assertion_proof_purpose.AssertionProofPurpose
    """
```

Credential Issuance proof purpose.

**validate**(\**proof: dict*, *document: dict*, *suite: LinkedDataProof*, *verification\_method: dict*,  
*document\_loader: Callable[[str, dict], dict]*) →  
*aries\_cloudagent.vc.ld\_proofs.validation\_result.PurposeResult*  
 Validate if the issuer matches the controller of the verification method.

### aries\_cloudagent.vc.ld\_proofs.purposes.proof\_purpose module

Base Proof Purpose class.

```
class aries_cloudagent.vc.ld_proofs.purposes.proof_purpose.ProofPurpose(*, term: str, date: Optional[datetime.datetime] = None,
max_timestamp_delta: Optional[datetime.timedelta] = None):
    """
    Bases: object
    """
```

Base proof purpose class.

**match**(*proof: dict*) → bool

Check whether the passed proof matches with the term of this proof purpose.

**update**(*proof: dict*) → dict

Update proof purpose on proof.

**validate**(\**proof: dict*, *document: dict*, *suite: LinkedDataProof*, *verification\_method: dict*,  
*document\_loader: Callable[[str, dict], dict]*) →  
*aries\_cloudagent.vc.ld\_proofs.validation\_result.PurposeResult*  
 Validate whether created date of proof is out of max\_timestamp\_delta range.

`aries_cloudagent.vc.ld_proofs.suites` package

### Submodules

`aries_cloudagent.vc.ld_proofs.suites.bbs_bls_signature_2020` module

`aries_cloudagent.vc.ld_proofs.suites.bbs_bls_signature_2020_base` module

`aries_cloudagent.vc.ld_proofs.suites.bbs_bls_signature_proof_2020` module

`aries_cloudagent.vc.ld_proofs.suites.ed25519_signature_2018` module

`aries_cloudagent.vc.ld_proofs.suites.jws_linked_data_signature` module

`aries_cloudagent.vc.ld_proofs.suites.linked_data_proof` module

`aries_cloudagent.vc.ld_proofs.suites.linked_data_signature` module

### Submodules

`aries_cloudagent.vc.ld_proofs.check` module

`aries_cloudagent.vc.ld_proofs.constants` module

JSON-LD, Linked Data Proof and Verifiable Credential constants.

`aries_cloudagent.vc.ld_proofs.document_loader` module

JSON-LD document loader methods.

**class** `aries_cloudagent.vc.ld_proofs.document_loader.DocumentLoader`(*profile:*  
*aries\_cloudagent.core.profile.Profile,*  
*cache\_ttl: int = 300*)

Bases: `object`

JSON-LD document loader.

**async** `load_document`(*url: str, options: dict*)

Load JSON-LD document.

Method signature conforms to PyLD document loader interface

Document loading is processed in separate thread to deal with async to sync transformation.

## aries\_cloudagent.vc.ld\_proofs.error module

Linked data proof exception classes.

**exception** aries\_cloudagent.vc.ld\_proofs.error.LinkedDataProofException

Bases: `Exception`

Base exception for linked data proof module.

## aries\_cloudagent.vc.ld\_proofs.ld\_proofs module

## aries\_cloudagent.vc.ld\_proofs.proof\_set module

## aries\_cloudagent.vc.ld\_proofs.validation\_result module

Proof verification and validation result classes.

```

class aries_cloudagent.vc.ld_proofs.validation_result.DocumentVerificationResult(*, verified:
    bool,
    document:
    Optional[dict]
    = None,
    results:
    Optional[List[aries_cloudagent.
    = None,
    errors:
    Optional[List[Exception]]
    = None)

```

Bases: `object`

Domain verification result class.

```

class aries_cloudagent.vc.ld_proofs.validation_result.ProofResult(*, verified: bool, proof:
    Optional[dict] = None, error:
    Optional[Exception] = None,
    purpose_result: Op-
    tional[aries_cloudagent.vc.ld_proofs.validation_
    = None)

```

Bases: `object`

Proof result class.

```

class aries_cloudagent.vc.ld_proofs.validation_result.PurposeResult(*, valid: bool, error:
    Optional[Exception] =
    None, controller:
    Optional[dict] = None)

```

Bases: `object`

Proof purpose result class.

`aries_cloudagent.vc.vc_ld` package

### Subpackages

`aries_cloudagent.vc.vc_ld.models` package

### Submodules

`aries_cloudagent.vc.vc_ld.models.credential` module

`aries_cloudagent.vc.vc_ld.models.linked_data_proof` module

### Submodules

`aries_cloudagent.vc.vc_ld.issue` module

`aries_cloudagent.vc.vc_ld.prove` module

`aries_cloudagent.vc.vc_ld.validation_result` module

`aries_cloudagent.vc.vc_ld.verify` module

`aries_cloudagent.wallet` package

Abstract and Indy wallet handling.

### Subpackages

`aries_cloudagent.wallet.models` package

### Submodules

`aries_cloudagent.wallet.models.wallet_record` module

### Submodules

`aries_cloudagent.wallet.askar` module

`aries_cloudagent.wallet.base` module

Wallet base class.

```
class aries_cloudagent.wallet.base.BaseWallet
```

Bases: `abc.ABC`

Abstract wallet interface.

**abstract async create\_local\_did**(*method: aries\_cloudagent.wallet.did\_method.DIDMethod, key\_type: aries\_cloudagent.wallet.key\_type.KeyType, seed: Optional[str] = None, did: Optional[str] = None, metadata: Optional[dict] = None*) → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Create and store a new local DID.

**Parameters**

- **method** – The method to use for the DID
- **key\_type** – The key type to use for the DID
- **seed** – Optional seed to use for DID
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** The created *DIDInfo*

**async create\_public\_did**(*method: aries\_cloudagent.wallet.did\_method.DIDMethod, key\_type: aries\_cloudagent.wallet.key\_type.KeyType, seed: Optional[str] = None, did: Optional[str] = None, metadata: dict = {}*) → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Create and store a new public DID.

**Parameters**

- **seed** – Optional seed to use for DID
- **did** – The DID to use
- **metadata** – Metadata to store with DID

**Returns** The created *DIDInfo*

**abstract async create\_signing\_key**(*key\_type: aries\_cloudagent.wallet.key\_type.KeyType, seed: Optional[str] = None, metadata: Optional[dict] = None*) → *aries\_cloudagent.wallet.did\_info.KeyInfo*

Create a new public/private signing keypair.

**Parameters**

- **key\_type** – Key type to create
- **seed** – Optional seed allowing deterministic key creation
- **metadata** – Optional metadata to store with the keypair

**Returns** A *KeyInfo* representing the new record

**abstract async get\_local\_did**(*did: str*) → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Find info for a local DID.

**Parameters** **did** – The DID for which to get info

**Returns** A *DIDInfo* instance for the DID

**abstract async get\_local\_did\_for\_verkey**(*verkey: str*) → *aries\_cloudagent.wallet.did\_info.DIDInfo*

Resolve a local DID from a verkey.

**Parameters** **verkey** – Verkey for which to get DID info

**Returns** A *DIDInfo* instance for the DID

**abstract async get\_local\_dids**() → Sequence[*aries\_cloudagent.wallet.did\_info.DIDInfo*]

Get list of defined local DIDs.

**Returns** A list of *DIDInfo* instances

**async get\_posted\_dids()** → Sequence[aries\_cloudagent.wallet.did\_info.DIDInfo]  
Get list of defined posted DIDs.

**Returns** A list of *DIDInfo* instances

**abstract async get\_public\_did()** → aries\_cloudagent.wallet.did\_info.DIDInfo  
Retrieve the public DID.

**Returns** The currently public *DIDInfo*, if any

**abstract async get\_signing\_key(verkey: str)** → aries\_cloudagent.wallet.did\_info.KeyInfo  
Fetch info for a signing keypair.

**Parameters** **verkey** – The verification key of the keypair

**Returns** A *KeyInfo* representing the keypair

**abstract async pack\_message(message: str, to\_verkeys: Sequence[str], from\_verkey: Optional[str] = None)** → bytes  
Pack a message for one or more recipients.

**Parameters**

- **message** – The message to pack
- **to\_verkeys** – The verkeys to pack the message for
- **from\_verkey** – The sender verkey

**Returns** The packed message

**abstract async replace\_local\_did\_metadata(did: str, metadata: dict)**  
Replace the metadata associated with a local DID.  
Prefer *set\_did\_endpoint()* to set endpoint in metadata.

**Parameters**

- **did** – DID for which to replace metadata
- **metadata** – The new metadata

**abstract async replace\_signing\_key\_metadata(verkey: str, metadata: dict)**  
Replace the metadata associated with a signing keypair.

**Parameters**

- **verkey** – The verification key of the keypair
- **metadata** – The new metadata to store

**abstract async rotate\_did\_keypair\_apply(did: str)** → None  
Apply temporary keypair as main for DID that wallet owns.

**Parameters** **did** – signing DID

**Raises**

- **WalletNotFoundError** – if wallet does not own DID
- **WalletError** – if wallet has not started key rotation

**abstract async rotate\_did\_keypair\_start(did: str, next\_seed: Optional[str] = None)** → str  
Begin key rotation for DID that wallet owns: generate new keypair.

**Parameters**

- **did** – signing DID
- **next\_seed** – seed for incoming ed25519 key pair (default random)

**Returns** The new verification key

**Raises** **WalletNotFoundError** – if wallet does not own DID

**async set\_did\_endpoint**(*did: str, endpoint: str, \_ledger: aries\_cloudagent.ledger.base.BaseLedger, endpoint\_type: Optional[aries\_cloudagent.ledger.endpoint\_type.EndpointType] = None, write\_ledger: bool = True, endorser\_did: Optional[str] = None*)

Update the endpoint for a DID in the wallet, send to ledger if public or posted.

**Parameters**

- **did** – DID for which to set endpoint
- **endpoint** – the endpoint to set, None to clear
- **ledger** – the ledger to which to send endpoint update if DID is public or posted
- **endpoint\_type** – the type of the endpoint/service. Only endpoint\_type 'endpoint' affects local wallet

**abstract async set\_public\_did**(*did: Union[str, aries\_cloudagent.wallet.did\_info.DIDInfo] → aries\_cloudagent.wallet.did\_info.DIDInfo*)

Assign the public DID.

**Returns** The updated *DIDInfo*

**abstract async sign\_message**(*message: Union[List[bytes], bytes], from\_verkey: str → bytes*)

Sign message(s) using the private key associated with a given verkey.

**Parameters**

- **message** – The message(s) to sign
- **from\_verkey** – Sign using the private key related to this verkey

**Returns** The signature

**abstract async unpack\_message**(*enc\_message: bytes → Tuple[str, str, str]*)

Unpack a message.

**Parameters** **enc\_message** – The encrypted message

**Returns** (message, from\_verkey, to\_verkey)

**Return type** A tuple

**abstract async verify\_message**(*message: Union[List[bytes], bytes], signature: bytes, from\_verkey: str, key\_type: aries\_cloudagent.wallet.key\_type.KeyType → bool*)

Verify a signature against the public key of the signer.

**Parameters**

- **message** – The message to verify
- **signature** – The signature to verify
- **from\_verkey** – Verkey to use in verification
- **key\_type** – The key type to derive the signature verification algorithm from

**Returns** True if verified, else False

## aries\_cloudagent.wallet.bbs module

BBS+ crypto.

**exception** aries\_cloudagent.wallet.bbs.**BbsException**(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

Base BBS exception.

aries\_cloudagent.wallet.bbs.**create\_bls12381g2\_keypair**(seed: Optional[bytes] = None) → Tuple[bytes, bytes]

Create a public and private bls12381g2 keypair from a seed value.

**Parameters** **seed** – Seed for keypair

**Returns** A tuple of (public key, secret key)

aries\_cloudagent.wallet.bbs.**sign\_messages\_bls12381g2**(messages: List[bytes], secret: bytes)  
Sign messages using a bls12381g2 private signing key.

**Parameters**

- **messages** (List[bytes]) – The messages to sign
- **secret** (bytes) – The private signing key

**Returns** The signature

**Return type** bytes

aries\_cloudagent.wallet.bbs.**verify\_signed\_messages\_bls12381g2**(messages: List[bytes], signature: bytes, public\_key: bytes) → bool

Verify an ed25519 signed message according to a public verification key.

**Parameters**

- **signed** – The signed messages
- **public\_key** – The public key to use in verification

**Returns** True if verified, else False

## aries\_cloudagent.wallet.crypto module

## aries\_cloudagent.wallet.did\_info module

KeyInfo, DIDInfo.

**class** aries\_cloudagent.wallet.did\_info.**DIDInfo**(did, verkey, metadata, method, key\_type)

Bases: `tuple`

**property** **did**

Alias for field number 0

**property** **key\_type**

Alias for field number 4

**property** **metadata**

Alias for field number 2

**property** **method**

Alias for field number 3



**property verkey**

Alias for field number 1

**class** aries\_cloudagent.wallet.did\_info.**KeyInfo**(*verkey, metadata, key\_type*)Bases: `tuple`**property key\_type**

Alias for field number 2

**property metadata**

Alias for field number 1

**property verkey**

Alias for field number 0

**aries\_cloudagent.wallet.did\_method module**

Did method enum.

**class** aries\_cloudagent.wallet.did\_method.**DIDMethod**(*value*)Bases: `enum.Enum`

DID Method class specifying DID methods with supported key types.

```
KEY = DIDMethodSpec(method_name='key', supported_key_types=[<KeyType.ED25519:
KeySpec(key_type='ed25519', multicodec_name='ed25519-pub',
multicodec_prefix=b'\xed\x01')>, <KeyType.BLS12381G2:
KeySpec(key_type='bls12381g2', multicodec_name='bls12_381-g2-pub',
multicodec_prefix=b'\xeb\x01')>], supports_rotation=False)
```

```
SOV = DIDMethodSpec(method_name='sov', supported_key_types=[<KeyType.ED25519:
KeySpec(key_type='ed25519', multicodec_name='ed25519-pub',
multicodec_prefix=b'\xed\x01')>], supports_rotation=True)
```

**from\_did()** → *aries\_cloudagent.wallet.did\_method.DIDMethod*

Get DID method instance from the method name.

**from\_metadata()** → *aries\_cloudagent.wallet.did\_method.DIDMethod*

Get DID method instance from metadata object.

Returns SOV if no metadata was found for backwards compatability.

**from\_method()** → Optional[*aries\_cloudagent.wallet.did\_method.DIDMethod*]

Get DID method instance from the method name.

**property method\_name:** `str`

Getter for did method name. e.g. sov or key.

**property supported\_key\_types:** List[*aries\_cloudagent.wallet.key\_type.KeyType*]

Getter for supported key types of method.

**supports\_key\_type**(*key\_type: aries\_cloudagent.wallet.key\_type.KeyType*) → `bool`

Check whether the current method supports the key type.

**property supports\_rotation:** `bool`

Check whether the current method supports key rotation.

```
class aries_cloudagent.wallet.did_method.DIDMethodSpec(method_name, supported_key_types,  
                                                    supports_rotation)
```

Bases: `tuple`

**property method\_name**

Alias for field number 0

**property supported\_key\_types**

Alias for field number 1

**property supports\_rotation**

Alias for field number 2

## **aries\_cloudagent.wallet.did\_posture module**

Ledger utilities.

**class** aries\_cloudagent.wallet.did\_posture.DIDPosture(*value*)

Bases: `enum.Enum`

Enum for DID postures: public, posted but not public, or in wallet only.

**POSTED** = DIDPostureSpec(moniker='posted', ordinal=1, public=False, posted=True)

**PUBLIC** = DIDPostureSpec(moniker='public', ordinal=0, public=True, posted=True)

**WALLET\_ONLY** = DIDPostureSpec(moniker='wallet\_only', ordinal=2, public=False, posted=False)

**static get**(*posture: Union[str, Mapping]*) → *aries\_cloudagent.wallet.did\_posture.DIDPosture*

Return enum instance corresponding to input string or DID metadata.

**property metadata:** `Mapping`

DID metadata for DID posture.

**property moniker:** `str`

Name for DID posture.

**property ordinal:** `Mapping`

public first, then posted and wallet-only.

**Type** Ordinal for presentation

**class** aries\_cloudagent.wallet.did\_posture.DIDPostureSpec(*moniker, ordinal, public, posted*)

Bases: `tuple`

**property moniker**

Alias for field number 0

**property ordinal**

Alias for field number 1

**property posted**

Alias for field number 3

**property public**

Alias for field number 2

## aries\_cloudagent.wallet.error module

Wallet-related exceptions.

**exception** aries\_cloudagent.wallet.error.WalletDuplicateError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.wallet.error.WalletError`

Duplicate record exception.

**exception** aries\_cloudagent.wallet.error.WalletError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.core.error.BaseError`

General wallet exception.

**exception** aries\_cloudagent.wallet.error.WalletNotFoundError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.wallet.error.WalletError`

Record not found exception.

**exception** aries\_cloudagent.wallet.error.WalletSettingsError(\*args, error\_code: Optional[str] = None, \*\*kwargs)

Bases: `aries_cloudagent.wallet.error.WalletError`

Invalid settings exception.

## aries\_cloudagent.wallet.in\_memory module

## aries\_cloudagent.wallet.indy module

## aries\_cloudagent.wallet.key\_pair module

Key pair storage manager.

**class** aries\_cloudagent.wallet.key\_pair.KeyPairStorageManager(store: `aries_cloudagent.storage.base.BaseStorage`)

Bases: `object`

Key pair storage manager.

**async** delete\_key\_pair(verkey: str)

Remove a previously-stored key pair record.

**Raises** `StorageNotFoundError` – If the record is not found

**async** find\_key\_pairs(tag\_query: Optional[Mapping] = None) → List[dict]

Find key pairs by tag query.

**async** get\_key\_pair(verkey: str) → dict

Retrieve signing key pair from storage by verkey.

### Parameters

- **storage** (`BaseStorage`) – The storage to use for querying
- **verkey** (str) – The verkey to query for

### Raises

- **StorageDuplicateError** – If more than one key pair is found for this verkey
- **StorageNotFoundError** – If no key pair is found for this verkey

**Returns** dict: The key pair data

```
async store_key_pair(public_key: bytes, secret_key: bytes, key_type:
                    aries_cloudagent.wallet.key_type.KeyType, metadata: dict = {}, tags: dict = {})
Store signing key pair in storage.
```

**Parameters**

- **public\_key** (*bytes*) – The public key
- **secret\_key** (*bytes*) – The secret key
- **key\_type** (*KeyType*) – The key type
- **metadata** (*dict*, *optional*) – The metadata
- **tags** (*dict*, *optional*) – The tags.

```
async update_key_pair_metadata(verkey: str, metadata: dict)
Update the metadata of a key pair record by verkey.
```

**Raises** **StorageNotFoundError** – If the record is not found.

## aries\_cloudagent.wallet.key\_type module

Key type enum.

```
class aries_cloudagent.wallet.key_type.KeySpec(key_type, multicodec_name, multicodec_prefix)
Bases: tuple
```

```
property key_type
    Alias for field number 0
```

```
property multicodec_name
    Alias for field number 1
```

```
property multicodec_prefix
    Alias for field number 2
```

```
class aries_cloudagent.wallet.key_type.KeyType(value)
Bases: enum.Enum
```

KeyType Enum specifying key types with multicodec name.

```
BLS12381G1 = KeySpec(key_type='bls12381g1', multicodec_name='bls12_381-g1-pub',
multicodec_prefix=b'\xea\x01')
```

```
BLS12381G1G2 = KeySpec(key_type='bls12381g1g2',
multicodec_name='bls12_381-g1g2-pub', multicodec_prefix=b'\xee\x01')
```

```
BLS12381G2 = KeySpec(key_type='bls12381g2', multicodec_name='bls12_381-g2-pub',
multicodec_prefix=b'\xeb\x01')
```

```
ED25519 = KeySpec(key_type='ed25519', multicodec_name='ed25519-pub',
multicodec_prefix=b'\xed\x01')
```

```
X25519 = KeySpec(key_type='x25519', multicodec_name='x25519-pub',
multicodec_prefix=b'\xec\x01')
```

**classmethod** `from_key_type(key_type: str) → Optional[aries_cloudagent.wallet.key_type.KeyType]`  
 Get KeyType instance from the key type identifier.

**classmethod** `from_multicodec_name(multicodec_name: str) → Optional[aries_cloudagent.wallet.key_type.KeyType]`  
 Get KeyType instance based on multicodec name. Returns None if not found.

**classmethod** `from_multicodec_prefix(multicodec_prefix: bytes) → Optional[aries_cloudagent.wallet.key_type.KeyType]`  
 Get KeyType instance based on multicodec prefix. Returns None if not found.

**classmethod** `from_prefixed_bytes(prefixed_bytes: bytes) → Optional[aries_cloudagent.wallet.key_type.KeyType]`  
 Get KeyType instance based on prefix in bytes. Returns None if not found.

**property** `key_type: str`  
 Getter for key type identifier.

**property** `multicodec_name: str`  
 Getter for multicodec name.

**property** `multicodec_prefix: bytes`  
 Getter for multicodec prefix.

**exception** `aries_cloudagent.wallet.key_type.KeyTypeException`  
 Bases: `BaseException`  
 Key type exception.

## aries\_cloudagent.wallet.routes module

## aries\_cloudagent.wallet.util module

Wallet utility functions.

`aries_cloudagent.wallet.util.abbr_verkey(full_verkey: str, did: Optional[str] = None) → str`  
 Given a full verkey and DID, return the abbreviated verkey.

`aries_cloudagent.wallet.util.b58_to_bytes(val: str) → bytes`  
 Convert a base 58 string to bytes.

`aries_cloudagent.wallet.util.b64_to_bytes(val: str, urlsafe=False) → bytes`  
 Convert a base 64 string to bytes.

`aries_cloudagent.wallet.util.b64_to_str(val: str, urlsafe=False, encoding=None) → str`  
 Convert a base 64 string to string on input encoding (default utf-8).

`aries_cloudagent.wallet.util.bytes_to_b58(val: bytes) → str`  
 Convert a byte string to base 58.

`aries_cloudagent.wallet.util.bytes_to_b64(val: bytes, urlsafe=False, pad=True, encoding: str = 'ascii') → str`  
 Convert a byte string to base 64.

`aries_cloudagent.wallet.util.default_did_from_verkey(verkey: str) → str`  
 Given a verkey, return the default indy did.

By default the did is the first 16 bytes of the verkey.

`aries_cloudagent.wallet.util.full_verkey(did: str, abbr_verkey: str) → str`  
 Given a DID and abbreviated verkey, return the full verkey.

**async** `aries_cloudagent.wallet.util.notify_endorse_did_event`(*profile:*  
*aries\_cloudagent.core.profile.Profile,*  
*did: str, meta\_data: dict*)

Send notification for a DID post-process event.

`aries_cloudagent.wallet.util.pad`(*val: str*) → *str*  
Pad base64 values if need be: JWT calls to omit trailing padding.

`aries_cloudagent.wallet.util.random_seed`() → *bytes*  
Generate a random seed value.

**Returns** A new random seed

`aries_cloudagent.wallet.util.set_urlsafe_b64`(*val: str, urlsafe: bool = True*) → *str*  
Set URL safety in base64 encoding.

`aries_cloudagent.wallet.util.str_to_b64`(*val: str, urlsafe=False, encoding=None, pad=True*) → *str*  
Convert a string to base64 string on input encoding (default utf-8).

`aries_cloudagent.wallet.util.unpad`(*val: str*) → *str*  
Remove padding from base64 values if need be.

### 1.1.2 Submodules

### 1.1.3 `aries_cloudagent.version` module

Library version information.

## INDICES AND TABLES

- `genindex`





## PYTHON MODULE INDEX

### a

aries\_cloudagent, 3  
aries\_cloudagent.admin, 3  
aries\_cloudagent.admin.base\_server, 3  
aries\_cloudagent.admin.error, 3  
aries\_cloudagent.admin.request\_context, 4  
aries\_cloudagent.askar, 5  
aries\_cloudagent.askar.didcomm, 5  
aries\_cloudagent.askar.store, 5  
aries\_cloudagent.cache, 6  
aries\_cloudagent.cache.base, 6  
aries\_cloudagent.cache.in\_memory, 7  
aries\_cloudagent.commands, 8  
aries\_cloudagent.commands.help, 8  
aries\_cloudagent.config, 8  
aries\_cloudagent.config.banner, 8  
aries\_cloudagent.config.base, 9  
aries\_cloudagent.config.base\_context, 11  
aries\_cloudagent.config.error, 11  
aries\_cloudagent.config.injection\_context, 11  
aries\_cloudagent.config.injector, 13  
aries\_cloudagent.config.logging, 14  
aries\_cloudagent.config.plugin\_settings, 14  
aries\_cloudagent.config.provider, 15  
aries\_cloudagent.config.settings, 16  
aries\_cloudagent.config.util, 17  
aries\_cloudagent.connections, 17  
aries\_cloudagent.connections.models, 17  
aries\_cloudagent.connections.models.diddoc, 17  
aries\_cloudagent.connections.models.diddoc.diddoc, 21  
aries\_cloudagent.connections.models.diddoc.publickey, 22  
aries\_cloudagent.connections.models.diddoc.service, 24  
aries\_cloudagent.connections.models.diddoc.util, 25  
aries\_cloudagent.core, 26  
aries\_cloudagent.core.error, 26  
aries\_cloudagent.core.event\_bus, 28  
aries\_cloudagent.core.goal\_code\_registry, 29  
aries\_cloudagent.core.plugin\_registry, 29  
aries\_cloudagent.core.profile, 30  
aries\_cloudagent.core.protocol\_registry, 33  
aries\_cloudagent.core.util, 33  
aries\_cloudagent.did, 34  
aries\_cloudagent.holder, 34  
aries\_cloudagent.indy, 34  
aries\_cloudagent.indy.credx, 34  
aries\_cloudagent.indy.holder, 40  
aries\_cloudagent.indy.issuer, 42  
aries\_cloudagent.indy.models, 34  
aries\_cloudagent.indy.models.predicate, 34  
aries\_cloudagent.indy.sdk, 36  
aries\_cloudagent.indy.sdk.error, 36  
aries\_cloudagent.indy.sdk.holder, 36  
aries\_cloudagent.indy.sdk.util, 38  
aries\_cloudagent.indy.sdk.wallet\_plugin, 39  
aries\_cloudagent.indy.sdk.wallet\_setup, 39  
aries\_cloudagent.indy.util, 44  
aries\_cloudagent.ledger, 44  
aries\_cloudagent.ledger.base, 49  
aries\_cloudagent.ledger.endpoint\_type, 52  
aries\_cloudagent.ledger.error, 53  
aries\_cloudagent.ledger.indy, 54  
aries\_cloudagent.ledger.indy\_vdr, 58  
aries\_cloudagent.ledger.merkel\_validation, 44  
aries\_cloudagent.ledger.merkel\_validation.constants, 45  
aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler, 45  
aries\_cloudagent.ledger.merkel\_validation.hasher, 47  
aries\_cloudagent.ledger.merkel\_validation.merkel\_verifier, 47  
aries\_cloudagent.ledger.merkel\_validation.trie, 48  
aries\_cloudagent.ledger.merkel\_validation.utils, 48  
aries\_cloudagent.ledger.multiple\_ledger, 49  
aries\_cloudagent.ledger.util, 62  
aries\_cloudagent.messaging, 62  
aries\_cloudagent.messaging.base\_message, 64

```

aries_cloudagent.messaging.credential_definitions, 71
62
aries_cloudagent.messaging.decorators, 62
aries_cloudagent.messaging.error, 65
aries_cloudagent.messaging.jsonld, 63
aries_cloudagent.messaging.jsonld.create_verifiable_data, 63
aries_cloudagent.messaging.jsonld.error, 63
aries_cloudagent.messaging.models, 64
aries_cloudagent.messaging.schemas, 64
aries_cloudagent.messaging.util, 65
aries_cloudagent.multitenant, 66
aries_cloudagent.multitenant.admin, 66
aries_cloudagent.multitenant.cache, 67
aries_cloudagent.multitenant.error, 67
aries_cloudagent.multitenant.manager_provider, 68
aries_cloudagent.protocols, 68
aries_cloudagent.protocols.actionmenu, 68
aries_cloudagent.protocols.actionmenu.definition, 69
aries_cloudagent.protocols.actionmenu.v1_0, 68
aries_cloudagent.protocols.actionmenu.v1_0.handlers, 68
aries_cloudagent.protocols.actionmenu.v1_0.messages, 69
aries_cloudagent.protocols.actionmenu.v1_0.messages.handlers, 68
aries_cloudagent.protocols.actionmenu.v1_0.models, 69
aries_cloudagent.protocols.basicmessage, 70
aries_cloudagent.protocols.basicmessage.definition, 70
aries_cloudagent.protocols.basicmessage.v1_0, 70
aries_cloudagent.protocols.basicmessage.v1_0.handlers, 70
aries_cloudagent.protocols.basicmessage.v1_0.messages, 70
aries_cloudagent.protocols.basicmessage.v1_0.messages.handlers, 70
aries_cloudagent.protocols.connections, 70
aries_cloudagent.protocols.connections.definition, 71
aries_cloudagent.protocols.connections.v1_0, 70
aries_cloudagent.protocols.connections.v1_0.handlers, 70
aries_cloudagent.protocols.connections.v1_0.message_types, 71
aries_cloudagent.protocols.connections.v1_0.messages, 71
aries_cloudagent.protocols.connections.v1_0.models, 71
aries_cloudagent.protocols.coordinate_mediation, 72
aries_cloudagent.protocols.coordinate_mediation.definition, 74
aries_cloudagent.protocols.coordinate_mediation.mediation, 74
aries_cloudagent.protocols.coordinate_mediation.v1_0, 72
aries_cloudagent.protocols.coordinate_mediation.v1_0.contracts, 73
aries_cloudagent.protocols.coordinate_mediation.v1_0.handlers, 72
aries_cloudagent.protocols.coordinate_mediation.v1_0.messages, 74
aries_cloudagent.protocols.coordinate_mediation.v1_0.messages.handlers, 72
aries_cloudagent.protocols.coordinate_mediation.v1_0.models, 73
aries_cloudagent.protocols.didcomm_prefix, 96
aries_cloudagent.protocols.didexchange, 75
aries_cloudagent.protocols.didexchange.definition, 77
aries_cloudagent.protocols.didexchange.v1_0, 75
aries_cloudagent.protocols.didexchange.v1_0.handlers, 75
aries_cloudagent.protocols.didexchange.v1_0.message_types, 76
aries_cloudagent.protocols.didexchange.v1_0.messages, 76
aries_cloudagent.protocols.didexchange.v1_0.messages.problem_reports, 76
aries_cloudagent.protocols.discovery, 77
aries_cloudagent.protocols.discovery.definition, 79
aries_cloudagent.protocols.discovery.v1_0, 77
aries_cloudagent.protocols.discovery.v1_0.handlers, 77
aries_cloudagent.protocols.discovery.v1_0.message_types, 78
aries_cloudagent.protocols.discovery.v1_0.messages, 77
aries_cloudagent.protocols.discovery.v1_0.models, 77
aries_cloudagent.protocols.discovery.v2_0, 78
aries_cloudagent.protocols.discovery.v2_0.handlers, 78
aries_cloudagent.protocols.discovery.v2_0.message_types, 78
aries_cloudagent.protocols.discovery.v2_0.messages, 78

```



- [aries\\_cloudagent.protocols.trustping.definition](#), 96
- [aries\\_cloudagent.protocols.trustping.v1\\_0](#), 95
- [aries\\_cloudagent.protocols.trustping.v1\\_0.handlers](#), 95
- [aries\\_cloudagent.protocols.trustping.v1\\_0.messages](#), 96
- [aries\\_cloudagent.protocols.trustping.v1\\_0.messages](#), 123
- [aries\\_cloudagent.resolver](#), 96
- [aries\\_cloudagent.resolver.base](#), 97
- [aries\\_cloudagent.resolver.default](#), 97
- [aries\\_cloudagent.resolver.did\\_resolver](#), 99
- [aries\\_cloudagent.resolver.did\\_resolver\\_registry](#), 99
- [aries\\_cloudagent.revocation](#), 99
- [aries\\_cloudagent.revocation.error](#), 101
- [aries\\_cloudagent.revocation.models](#), 99
- [aries\\_cloudagent.revocation.models.revocation\\_registry](#), 100
- [aries\\_cloudagent.revocation.recover](#), 102
- [aries\\_cloudagent.storage](#), 102
- [aries\\_cloudagent.storage.base](#), 103
- [aries\\_cloudagent.storage.error](#), 105
- [aries\\_cloudagent.storage.indy](#), 105
- [aries\\_cloudagent.storage.record](#), 107
- [aries\\_cloudagent.storage.vc\\_holder](#), 102
- [aries\\_cloudagent.tails](#), 107
- [aries\\_cloudagent.tails.base](#), 107
- [aries\\_cloudagent.tails.error](#), 108
- [aries\\_cloudagent.transport](#), 108
- [aries\\_cloudagent.transport.error](#), 114
- [aries\\_cloudagent.transport.inbound](#), 108
- [aries\\_cloudagent.transport.inbound.message](#), 108
- [aries\\_cloudagent.transport.inbound.receipt](#), 109
- [aries\\_cloudagent.transport.outbound](#), 110
- [aries\\_cloudagent.transport.outbound.base](#), 110
- [aries\\_cloudagent.transport.outbound.http](#), 111
- [aries\\_cloudagent.transport.outbound.status](#), 112
- [aries\\_cloudagent.transport.outbound.ws](#), 112
- [aries\\_cloudagent.transport.queue](#), 113
- [aries\\_cloudagent.transport.queue.base](#), 113
- [aries\\_cloudagent.transport.queue.basic](#), 114
- [aries\\_cloudagent.transport.stats](#), 115
- [aries\\_cloudagent.transport.wire\\_format](#), 116
- [aries\\_cloudagent.utils](#), 117
- [aries\\_cloudagent.utils.classloader](#), 117
- [aries\\_cloudagent.utils.dependencies](#), 119
- [aries\\_cloudagent.utils.env](#), 119
- [aries\\_cloudagent.utils.stats](#), 119
- [aries\\_cloudagent.utils.task\\_queue](#), 120
- [aries\\_cloudagent.vc](#), 122
- [aries\\_cloudagent.vc.ld\\_proofs.constants](#), 126
- [aries\\_cloudagent.vc.ld\\_proofs.document\\_loader](#), 126
- [aries\\_cloudagent.vc.ld\\_proofs.error](#), 127
- [aries\\_cloudagent.vc.ld\\_proofs.purposes](#), 123
- [aries\\_cloudagent.vc.ld\\_proofs.purposes.assertion\\_proof\\_purpose](#), 123
- [aries\\_cloudagent.vc.ld\\_proofs.purposes.authentication\\_proof\\_purpose](#), 123
- [aries\\_cloudagent.vc.ld\\_proofs.purposes.controller\\_proof\\_purpose](#), 124
- [aries\\_cloudagent.vc.ld\\_proofs.purposes.credential\\_issuance\\_purpose](#), 125
- [aries\\_cloudagent.vc.ld\\_proofs.purposes.proof\\_purpose](#), 125
- [aries\\_cloudagent.vc.ld\\_proofs.validation\\_result](#), 127
- [aries\\_cloudagent.version](#), 138
- [aries\\_cloudagent.wallet](#), 128
- [aries\\_cloudagent.wallet.base](#), 128
- [aries\\_cloudagent.wallet.bbs](#), 132
- [aries\\_cloudagent.wallet.did\\_info](#), 132
- [aries\\_cloudagent.wallet.did\\_method](#), 133
- [aries\\_cloudagent.wallet.did\\_posture](#), 134
- [aries\\_cloudagent.wallet.error](#), 135
- [aries\\_cloudagent.wallet.key\\_pair](#), 135
- [aries\\_cloudagent.wallet.key\\_type](#), 136
- [aries\\_cloudagent.wallet.models](#), 128
- [aries\\_cloudagent.wallet.util](#), 137

## A

`abbr_verkey()` (in module `aries_cloudagent.wallet.util`), 137  
`accept_txn_author_agreement()` (`aries_cloudagent.ledger.base.BaseLedger` method), 49  
`accept_txn_author_agreement()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 54  
`accept_txn_author_agreement()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 58  
`acquire()` (`aries_cloudagent.cache.base.BaseCache` method), 6  
`active` (`aries_cloudagent.core.profile.ProfileSession` property), 32  
`add_active()` (`aries_cloudagent.utils.task_queue.TaskQueue` method), 121  
`add_pending()` (`aries_cloudagent.utils.task_queue.TaskQueue` method), 121  
`add_record()` (`aries_cloudagent.storage.base.BaseStorage` method), 103  
`add_record()` (`aries_cloudagent.storage.indy.IndySdkStorage` method), 105  
`add_service_pubkeys()` (`aries_cloudagent.connections.models.diddoc.DIDDoc` method), 17  
`add_service_pubkeys()` (`aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc` method), 21  
`AdminError`, 3  
`AdminRequestContext` (class in `aries_cloudagent.admin.request_context`), 4  
`AdminSetupError`, 3  
`ArgsParseError`, 11  
`aries_cloudagent` module, 3  
`aries_cloudagent.admin` module, 3  
`aries_cloudagent.admin.base_server` module, 3  
`aries_cloudagent.admin.error` module, 3  
`aries_cloudagent.admin.request_context` module, 4  
`aries_cloudagent.askar` module, 5  
`aries_cloudagent.askar.didcomm` module, 5  
`aries_cloudagent.askar.store` module, 5  
`aries_cloudagent.cache` module, 6  
`aries_cloudagent.cache.base` module, 6  
`aries_cloudagent.cache.in_memory` module, 7  
`aries_cloudagent.commands` module, 8  
`aries_cloudagent.commands.help` module, 8  
`aries_cloudagent.config` module, 8  
`aries_cloudagent.config.banner` module, 8  
`aries_cloudagent.config.base` module, 9  
`aries_cloudagent.config.base_context` module, 11  
`aries_cloudagent.config.error` module, 11  
`aries_cloudagent.config.injection_context` module, 11  
`aries_cloudagent.config.injector` module, 13  
`aries_cloudagent.config.logging` module, 14  
`aries_cloudagent.config.plugin_settings` module, 14  
`aries_cloudagent.config.provider` module, 15  
`aries_cloudagent.config.settings` module, 16

<code>aries_cloudagent.config.util</code>	<code>aries_cloudagent.indy.sdk.util</code>
module, 17	module, 38
<code>aries_cloudagent.connections</code>	<code>aries_cloudagent.indy.sdk.wallet_plugin</code>
module, 17	module, 39
<code>aries_cloudagent.connections.models</code>	<code>aries_cloudagent.indy.sdk.wallet_setup</code>
module, 17	module, 39
<code>aries_cloudagent.connections.models.diddoc</code>	<code>aries_cloudagent.indy.util</code>
module, 17	module, 44
<code>aries_cloudagent.connections.models.diddoc.didinfo</code>	<code>aries_cloudagent.ledger</code>
module, 21	module, 44
<code>aries_cloudagent.connections.models.diddoc.public_key</code>	<code>aries_cloudagent.ledger.base</code>
module, 22	module, 49
<code>aries_cloudagent.connections.models.diddoc.services</code>	<code>aries_cloudagent.ledger.endpoint_type</code>
module, 24	module, 52
<code>aries_cloudagent.connections.models.diddoc.util</code>	<code>aries_cloudagent.ledger.error</code>
module, 25	module, 53
<code>aries_cloudagent.core</code>	<code>aries_cloudagent.ledger.indy</code>
module, 26	module, 54
<code>aries_cloudagent.core.error</code>	<code>aries_cloudagent.ledger.indy_vdr</code>
module, 26	module, 58
<code>aries_cloudagent.core.event_bus</code>	<code>aries_cloudagent.ledger.merkel_validation</code>
module, 28	module, 44
<code>aries_cloudagent.core.goal_code_registry</code>	<code>aries_cloudagent.ledger.merkel_validation.constants</code>
module, 29	module, 45
<code>aries_cloudagent.core.plugin_registry</code>	<code>aries_cloudagent.ledger.merkel_validation.domain_txn_handler</code>
module, 29	module, 45
<code>aries_cloudagent.core.profile</code>	<code>aries_cloudagent.ledger.merkel_validation.hasher</code>
module, 30	module, 47
<code>aries_cloudagent.core.protocol_registry</code>	<code>aries_cloudagent.ledger.merkel_validation.merkel_verifier</code>
module, 33	module, 47
<code>aries_cloudagent.core.util</code>	<code>aries_cloudagent.ledger.merkel_validation.trie</code>
module, 33	module, 48
<code>aries_cloudagent.did</code>	<code>aries_cloudagent.ledger.merkel_validation.utils</code>
module, 34	module, 48
<code>aries_cloudagent.holder</code>	<code>aries_cloudagent.ledger.multiple_ledger</code>
module, 34	module, 49
<code>aries_cloudagent.indy</code>	<code>aries_cloudagent.ledger.util</code>
module, 34	module, 62
<code>aries_cloudagent.indy.credx</code>	<code>aries_cloudagent.messaging</code>
module, 34	module, 62
<code>aries_cloudagent.indy.holder</code>	<code>aries_cloudagent.messaging.base_message</code>
module, 40	module, 64
<code>aries_cloudagent.indy.issuer</code>	<code>aries_cloudagent.messaging.credential_definitions</code>
module, 42	module, 62
<code>aries_cloudagent.indy.models</code>	<code>aries_cloudagent.messaging.decorators</code>
module, 34	module, 62
<code>aries_cloudagent.indy.models.predicate</code>	<code>aries_cloudagent.messaging.error</code>
module, 34	module, 65
<code>aries_cloudagent.indy.sdk</code>	<code>aries_cloudagent.messaging.jsonld</code>
module, 36	module, 63
<code>aries_cloudagent.indy.sdk.error</code>	<code>aries_cloudagent.messaging.jsonld.create_verify_data</code>
module, 36	module, 63
<code>aries_cloudagent.indy.sdk.holder</code>	<code>aries_cloudagent.messaging.jsonld.error</code>
module, 36	module, 63

<code>aries_cloudagent.messaging.models</code> module, 64	<code>aries_cloudagent.protocols.connections.v1_0.messages</code> module, 71
<code>aries_cloudagent.messaging.schemas</code> module, 64	<code>aries_cloudagent.protocols.connections.v1_0.models</code> module, 71
<code>aries_cloudagent.messaging.util</code> module, 65	<code>aries_cloudagent.protocols.coordinate_mediation</code> module, 72
<code>aries_cloudagent.multitenant</code> module, 66	<code>aries_cloudagent.protocols.coordinate_mediation.definition</code> module, 74
<code>aries_cloudagent.multitenant.admin</code> module, 66	<code>aries_cloudagent.protocols.coordinate_mediation.mediation</code> module, 74
<code>aries_cloudagent.multitenant.cache</code> module, 67	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0</code> module, 72
<code>aries_cloudagent.multitenant.error</code> module, 67	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.contr</code> module, 73
<code>aries_cloudagent.multitenant.manager_provider</code> module, 68	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.handl</code> module, 72
<code>aries_cloudagent.protocols</code> module, 68	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.messa</code> module, 74
<code>aries_cloudagent.protocols.actionmenu</code> module, 68	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.messa</code> module, 72
<code>aries_cloudagent.protocols.actionmenu.definition</code> module, 69	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.messa</code> module, 72
<code>aries_cloudagent.protocols.actionmenu.v1_0</code> module, 68	<code>aries_cloudagent.protocols.coordinate_mediation.v1_0.model</code> module, 73
<code>aries_cloudagent.protocols.actionmenu.v1_0.handlers</code> module, 68	<code>aries_cloudagent.protocols.didcomm_prefix</code> module, 96
<code>aries_cloudagent.protocols.actionmenu.v1_0.messages</code> module, 69	<code>aries_cloudagent.protocols.didexchange</code> module, 75
<code>aries_cloudagent.protocols.actionmenu.v1_0.messages</code> module, 68	<code>aries_cloudagent.protocols.didexchange.definition</code> module, 77
<code>aries_cloudagent.protocols.actionmenu.v1_0.models</code> module, 69	<code>aries_cloudagent.protocols.didexchange.v1_0</code> module, 75
<code>aries_cloudagent.protocols.basicmessage</code> module, 70	<code>aries_cloudagent.protocols.didexchange.v1_0.handlers</code> module, 75
<code>aries_cloudagent.protocols.basicmessage.definition</code> module, 70	<code>aries_cloudagent.protocols.didexchange.v1_0.message_types</code> module, 76
<code>aries_cloudagent.protocols.basicmessage.v1_0</code> module, 70	<code>aries_cloudagent.protocols.didexchange.v1_0.messages</code> module, 76
<code>aries_cloudagent.protocols.basicmessage.v1_0.handlers</code> module, 70	<code>aries_cloudagent.protocols.didexchange.v1_0.messages.probl</code> module, 76
<code>aries_cloudagent.protocols.basicmessage.v1_0.messages</code> module, 70	<code>aries_cloudagent.protocols.discovery</code> module, 77
<code>aries_cloudagent.protocols.basicmessage.v1_0.messages</code> module, 70	<code>aries_cloudagent.protocols.discovery.definition</code> module, 79
<code>aries_cloudagent.protocols.connections</code> module, 70	<code>aries_cloudagent.protocols.discovery.v1_0</code> module, 77
<code>aries_cloudagent.protocols.connections.definition</code> module, 71	<code>aries_cloudagent.protocols.discovery.v1_0.handlers</code> module, 77
<code>aries_cloudagent.protocols.connections.v1_0</code> module, 70	<code>aries_cloudagent.protocols.discovery.v1_0.message_types</code> module, 78
<code>aries_cloudagent.protocols.connections.v1_0.handlers</code> module, 70	<code>aries_cloudagent.protocols.discovery.v1_0.messages</code> module, 77
<code>aries_cloudagent.protocols.connections.v1_0.messages</code> module, 71	<code>aries_cloudagent.protocols.discovery.v1_0.models</code> module, 77



```

aries_cloudagent.protocols.discovery.v2_0      aries_cloudagent.protocols.notification.v1_0.messages
    module, 78                                module, 86
aries_cloudagent.protocols.discovery.v2_0.handlers aries_cloudagent.protocols.out_of_band
    module, 78                                module, 87
aries_cloudagent.protocols.discovery.v2_0.messages aries_cloudagent.protocols.out_of_band.definition
    module, 78                                module, 88
aries_cloudagent.protocols.discovery.v2_0.messages aries_cloudagent.protocols.out_of_band.v1_0
    module, 78                                module, 87
aries_cloudagent.protocols.discovery.v2_0.models aries_cloudagent.protocols.out_of_band.v1_0.controller
    module, 78                                module, 88
aries_cloudagent.protocols.endorse_transaction aries_cloudagent.protocols.out_of_band.v1_0.handlers
    module, 79                                module, 87
aries_cloudagent.protocols.endorse_transaction.definition aries_cloudagent.protocols.out_of_band.v1_0.message_types
    module, 81                                module, 88
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.out_of_band.v1_0.messages
    module, 79                                module, 87
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.out_of_band.v1_0.models
    module, 80                                module, 87
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.present_proof
    module, 79                                module, 88
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.present_proof.definition
    module, 80                                module, 91
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.present_proof.diff
    module, 79                                module, 88
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.protocols.present_proof.indy
    module, 80                                module, 88
aries_cloudagent.protocols.endorse_transaction.messages aries_cloudagent.projects
    module, 81                                module, 92
aries_cloudagent.protocols.introduction      aries_cloudagent.protocols.problem_report.definition
    module, 81                                module, 92
aries_cloudagent.protocols.introduction.definition aries_cloudagent.protocols.revocation_notification
    module, 82                                module, 92
aries_cloudagent.protocols.introduction.v0_1 aries_cloudagent.protocols.revocation_notification.definition
    module, 81                                module, 93
aries_cloudagent.protocols.introduction.v0_1.handlers aries_cloudagent.protocols.revocation_notification.v1_0
    module, 81                                module, 92
aries_cloudagent.protocols.introduction.v0_1.messages aries_cloudagent.protocols.revocation_notification.v1_0.handlers
    module, 82                                module, 92
aries_cloudagent.protocols.introduction.v0_1.messages aries_cloudagent.protocols.revocation_notification.v1_0.messages
    module, 81                                module, 93
aries_cloudagent.protocols.issue_credential aries_cloudagent.protocols.revocation_notification.v1_0.messages
    module, 82                                module, 92
aries_cloudagent.protocols.issue_credential.definition aries_cloudagent.protocols.revocation_notification.v1_0.messages
    module, 86                                module, 92
aries_cloudagent.protocols.notification      aries_cloudagent.protocols.revocation_notification.v2_0
    module, 86                                module, 93
aries_cloudagent.protocols.notification.definition aries_cloudagent.protocols.revocation_notification.v2_0.handlers
    module, 87                                module, 93
aries_cloudagent.protocols.notification.v1_0 aries_cloudagent.protocols.revocation_notification.v2_0.messages
    module, 86                                module, 93
aries_cloudagent.protocols.notification.v1_0.handlers aries_cloudagent.protocols.revocation_notification.v2_0.messages
    module, 86                                module, 93
aries_cloudagent.protocols.notification.v1_0.messages aries_cloudagent.protocols.revocation_notification.v2_0.messages
    module, 86                                module, 93

```



<code>aries_cloudagent.protocols.routing</code> module, 94	<code>aries_cloudagent.storage.record</code> module, 107
<code>aries_cloudagent.protocols.routing.definition</code> module, 95	<code>aries_cloudagent.storage.vc_holder</code> module, 102
<code>aries_cloudagent.protocols.routing.v1_0</code> module, 94	<code>aries_cloudagent.tails</code> module, 107
<code>aries_cloudagent.protocols.routing.v1_0.handlers</code> module, 94	<code>aries_cloudagent.tails.base</code> module, 107
<code>aries_cloudagent.protocols.routing.v1_0.messages</code> module, 95	<code>aries_cloudagent.tails.error</code> module, 108
<code>aries_cloudagent.protocols.routing.v1_0.messages</code> module, 94	<code>aries_cloudagent.transport</code> module, 108
<code>aries_cloudagent.protocols.routing.v1_0.models</code> module, 94	<code>aries_cloudagent.transport.error</code> module, 114
<code>aries_cloudagent.protocols.trustping</code> module, 95	<code>aries_cloudagent.transport.inbound</code> module, 108
<code>aries_cloudagent.protocols.trustping.definition</code> module, 96	<code>aries_cloudagent.transport.inbound.message</code> module, 108
<code>aries_cloudagent.protocols.trustping.v1_0</code> module, 95	<code>aries_cloudagent.transport.inbound.receipt</code> module, 109
<code>aries_cloudagent.protocols.trustping.v1_0.handlers</code> module, 95	<code>aries_cloudagent.transport.outbound</code> module, 110
<code>aries_cloudagent.protocols.trustping.v1_0.messages</code> module, 96	<code>aries_cloudagent.transport.outbound.base</code> module, 110
<code>aries_cloudagent.protocols.trustping.v1_0.messages</code> module, 95	<code>aries_cloudagent.transport.outbound.http</code> module, 111
<code>aries_cloudagent.resolver</code> module, 96	<code>aries_cloudagent.transport.outbound.status</code> module, 112
<code>aries_cloudagent.resolver.base</code> module, 97	<code>aries_cloudagent.transport.outbound.ws</code> module, 112
<code>aries_cloudagent.resolver.default</code> module, 97	<code>aries_cloudagent.transport.queue</code> module, 113
<code>aries_cloudagent.resolver.did_resolver</code> module, 99	<code>aries_cloudagent.transport.queue.base</code> module, 113
<code>aries_cloudagent.resolver.did_resolver_registry</code> module, 99	<code>aries_cloudagent.transport.queue.basic</code> module, 114
<code>aries_cloudagent.revocation</code> module, 99	<code>aries_cloudagent.transport.stats</code> module, 115
<code>aries_cloudagent.revocation.error</code> module, 101	<code>aries_cloudagent.transport.wire_format</code> module, 116
<code>aries_cloudagent.revocation.models</code> module, 99	<code>aries_cloudagent.utils</code> module, 117
<code>aries_cloudagent.revocation.models.revocation_registry</code> module, 100	<code>aries_cloudagent.utils.classloader</code> module, 117
<code>aries_cloudagent.revocation.recover</code> module, 102	<code>aries_cloudagent.utils.dependencies</code> module, 119
<code>aries_cloudagent.storage</code> module, 102	<code>aries_cloudagent.utils.env</code> module, 119
<code>aries_cloudagent.storage.base</code> module, 103	<code>aries_cloudagent.utils.stats</code> module, 119
<code>aries_cloudagent.storage.error</code> module, 105	<code>aries_cloudagent.utils.task_queue</code> module, 120
<code>aries_cloudagent.storage.indy</code> module, 105	<code>aries_cloudagent.vc</code> module, 122



BaseAdminServer	(class in aries_cloudagent.admin.base_server), 3	in build_and_return_get_nym_request()	(aries_cloudagent.ledger.indy_vdr.IndyVdrLedger method), 58
BaseCache	(class in aries_cloudagent.cache.base), 6	in build_context()	(aries_cloudagent.config.base_context.ContextBuilder method), 11
BaseDIDResolver	(class in aries_cloudagent.resolver.base), 97	bytes_to_b58()	(in aries_cloudagent.wallet.util), 137
BaseError	26	bytes_to_b64()	(in aries_cloudagent.wallet.util), 137
BaseInjector	(class in aries_cloudagent.config.base), 9	ByteSize	(class in aries_cloudagent.config.util), 17
BaseJSONLDMessagingError	63	<b>C</b>	
BaseLedger	(class in aries_cloudagent.ledger.base), 49	CachedProvider	(class in aries_cloudagent.config.provider), 15
BaseMessage	(class in aries_cloudagent.messaging.base_message), 64	CacheError	7
BaseMessageQueue	(class in aries_cloudagent.transport.queue.base), 113	CacheKeyLock	(class in aries_cloudagent.cache.base), 7
BaseOutboundTransport	(class in aries_cloudagent.transport.outbound.base), 110	calculate_root_hash()	(aries_cloudagent.ledger.merkel_validation.merkel_verifier.Merkel method), 47
BaseProvider	(class in aries_cloudagent.config.base), 9	cancel()	(aries_cloudagent.utils.task_queue.PendingTask method), 121
BaseSettings	(class in aries_cloudagent.config.base), 10	cancel()	(aries_cloudagent.utils.task_queue.TaskQueue method), 121
BaseStorage	(class in aries_cloudagent.storage.base), 103	cancel_pending()	(aries_cloudagent.utils.task_queue.TaskQueue method), 121
BaseStorageSearch	(class in aries_cloudagent.storage.base), 104	cancelled	(aries_cloudagent.utils.task_queue.PendingTask property), 121
BaseStorageSearchSession	(class in aries_cloudagent.storage.base), 104	cancelled	(aries_cloudagent.utils.task_queue.TaskQueue property), 121
BaseTailsServer	(class in aries_cloudagent.tails.base), 107	canon()	(in module aries_cloudagent.messaging.util), 65
BaseWallet	(class in aries_cloudagent.wallet.base), 128	canon_did()	(in aries_cloudagent.connections.models.diddoc.util), 25
BaseWireFormat	(class in aries_cloudagent.transport.wire_format), 116	canon_ref()	(in aries_cloudagent.connections.models.diddoc.util), 25
BasicMessageQueue	(class in aries_cloudagent.transport.queue.basic), 114	cfg_path	(aries_cloudagent.ledger.indy_vdr.IndyVdrLedgerPool property), 61
BbsException	132	check_existing_schema()	(aries_cloudagent.ledger.indy.IndySdkLedger method), 54
bin_to_nibbles()	(in aries_cloudagent.ledger.merkel_validation.utils), 48	check_existing_schema()	(aries_cloudagent.ledger.indy_vdr.IndyVdrLedger method), 58
bind_instance()	(aries_cloudagent.config.injector.Injector method), 13	check_pool_config()	(aries_cloudagent.ledger.indy.IndySdkLedgerPool method), 57
bind_provider()	(aries_cloudagent.config.injector.Injector method), 13	CHUNK	(aries_cloudagent.indy.holder.IndyHolder attribute), 40
BLS12381G1	(aries_cloudagent.wallet.key_type.KeyType attribute), 136	ClassLoader	(class in aries_cloudagent.utils.classloader), 117
BLS12381G1G2	(aries_cloudagent.wallet.key_type.KeyType attribute), 136	ClassNotFoundError	118
BLS12381G2	(aries_cloudagent.wallet.key_type.KeyType attribute), 136	ClassProvider	(class in aries_cloudagent.config.provider), 15
BoundedInt	(class in aries_cloudagent.config.util), 17		
build_and_return_get_nym_request()	(aries_cloudagent.ledger.indy.IndySdkLedger method), 54		

**ClassProvider.Inject** (class in `aries_cloudagent.config.provider`), 15  
**clear()** (`aries_cloudagent.cache.base.BaseCache` method), 6  
**clear()** (`aries_cloudagent.cache.in_memory.InMemoryCache` method), 7  
**clear\_binding()** (`aries_cloudagent.config.injector.Injector` method), 13  
**clear\_value()** (`aries_cloudagent.config.settings.Settings` method), 16  
**close()** (`aries_cloudagent.askar.store.AskarOpenStore` method), 5  
**close()** (`aries_cloudagent.core.profile.Profile` method), 30  
**close()** (`aries_cloudagent.indy.sdk.wallet_setup.IndyOpenStore` method), 39  
**close()** (`aries_cloudagent.ledger.indy.IndySdkLedgerPool` method), 57  
**close()** (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedgerPool` method), 61  
**close()** (`aries_cloudagent.storage.base.BaseStorageSearchController` method), 104  
**close()** (`aries_cloudagent.storage.indy.IndySdkStorageSearchController` method), 107  
**ClosedPoolError**, 53  
**collector** (`aries_cloudagent.transport.outbound.base.BaseOutboundTransport` property), 110  
**Collector** (class in `aries_cloudagent.utils.stats`), 119  
**commit()** (`aries_cloudagent.core.profile.ProfileSession` method), 32  
**common\_config()** (in module `aries_cloudagent.config.util`), 17  
**complete()** (`aries_cloudagent.utils.task_queue.TaskQueue` method), 121  
**COMPLETE\_NOT\_ACCEPTED** (`aries_cloudagent.protocols.didexchange.v1_0.messages` attribute), 76  
**completed\_task()** (`aries_cloudagent.utils.task_queue.TaskQueue` method), 121  
**CompletedTask** (class in `aries_cloudagent.utils.task_queue`), 120  
**ConfigError**, 10  
**configure()** (`aries_cloudagent.config.logging.LoggingConfigurator` class method), 14  
**connection\_id** (`aries_cloudagent.transport.inbound.receipt.MessageReceipt` property), 109  
**connection\_queued\_end()** (`aries_cloudagent.transport.stats.StatsTracer` method), 115  
**connection\_queued\_start()** (`aries_cloudagent.transport.stats.StatsTracer` method), 115  
**connection\_ready()** (`aries_cloudagent.transport.stats.StatsTracer` method), 115  
**CONTEXT** (`aries_cloudagent.connections.models.diddoc.DIDDoc` attribute), 17  
**CONTEXT** (`aries_cloudagent.connections.models.diddoc.diddoc.DIDDoc` attribute), 21  
**context** (`aries_cloudagent.core.profile.Profile` property), 30  
**context** (`aries_cloudagent.core.profile.ProfileSession` property), 32  
**context\_close()** (`aries_cloudagent.ledger.indy.IndySdkLedgerPool` method), 57  
**context\_close()** (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedgerPool` method), 61  
**context\_open()** (`aries_cloudagent.ledger.indy.IndySdkLedgerPool` method), 57  
**context\_open()** (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedgerPool` method), 61  
**ContextBuilder** (class in `aries_cloudagent.config.base_context`), 11  
**Controller** (`aries_cloudagent.connections.models.diddoc.PublicKey` property), 19  
**Controller** (`aries_cloudagent.connections.models.diddoc.publickey.PublicKey` property), 23  
**Controller** (class in `aries_cloudagent.protocols.coordinate_mediation.v1_0.messages`), 73  
**Controller** (class in `aries_cloudagent.protocols.endorse_transaction.v1_0.messages`), 87  
**Controller** (class in `aries_cloudagent.protocols.out_of_band.v1_0.messages`), 88  
**ControllerProofPurpose** (class in `aries_cloudagent.vc.ld_proofs.purposes.controller_proof_purposes`), 124  
**controllers** (`aries_cloudagent.core.protocol_registry.ProtocolRegistry` property), 33  
**copy()** (`aries_cloudagent.config.base.BaseInjector` method), 9  
**copy()** (`aries_cloudagent.config.injection_context.InjectionContext` method), 11  
**copy()** (`aries_cloudagent.config.injector.Injector` method), 13  
**copy()** (`aries_cloudagent.config.plugin_settings.PluginSettings` method), 15  
**copy()** (`aries_cloudagent.config.settings.Settings` method), 16  
**coro\_ident()** (in module `aries_cloudagent.utils.task_queue`), 122  
**coro\_timed()** (in module `aries_cloudagent.utils.task_queue`), 122  
**create\_and\_send\_credential\_definition()** (`aries_cloudagent.ledger.base.BaseLedger` method), 49  
**create\_and\_send\_credential\_definition()** (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 49

`method`), 54  
`create_and_send_credential_definition()`  
*(aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedgercreate\_schema()* *(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 58  
`create_and_send_schema()`  
*(aries\_cloudagent.ledger.base.BaseLedger*  
*method)*, 50  
`create_and_send_schema()`  
*(aries\_cloudagent.ledger.indy.IndySdkLedger*  
*method)*, 54  
`create_and_send_schema()`  
*(aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedgercreate\_verify\_data()* *(in* *module*  
*method)*, 58  
`create_and_store_credential_definition()`  
*(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 42  
`create_and_store_revocation_registry()`  
*(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 42  
`create_bls12381g2_keypair()` *(in* *module*  
*aries\_cloudagent.wallet.bbs)*, 132  
`create_credential()`  
*(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 43  
`create_credential_offer()`  
*(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 43  
`create_credential_request()`  
*(aries\_cloudagent.indy.holder.IndyHolder*  
*method)*, 40  
`create_credential_request()`  
*(aries\_cloudagent.indy.sdk.holder.IndySdkHolder*  
*method)*, 36  
`create_local_did()` *(aries\_cloudagent.wallet.base.BaseWallet*  
*method)*, 128  
`create_pool_config()`  
*(aries\_cloudagent.ledger.indy.IndySdkLedgerPool*  
*method)*, 57  
`create_pool_config()`  
*(aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedgerPool*  
*method)*, 61  
`create_presentation()`  
*(aries\_cloudagent.indy.holder.IndyHolder*  
*method)*, 40  
`create_presentation()`  
*(aries\_cloudagent.indy.sdk.holder.IndySdkHolder*  
*method)*, 37  
`create_public_did()`  
*(aries\_cloudagent.wallet.base.BaseWallet*  
*method)*, 129  
`create_revocation_state()`  
*(aries\_cloudagent.indy.holder.IndyHolder*  
*method)*, 41  
`create_revocation_state()`  
*(aries\_cloudagent.indy.sdk.holder.IndySdkHolder*  
*method)*, 37  
`create_signing_key()`  
*(aries\_cloudagent.wallet.base.BaseWallet*  
*method)*, 129  
`create_tails_reader()` *(in* *module*  
*aries\_cloudagent.indy.sdk.util)*, 38  
`create_tails_writer()` *(in* *module*  
*aries\_cloudagent.indy.sdk.util)*, 38  
`create_verify_data()` *(in* *module*  
*aries\_cloudagent.messaging.jsonld.create\_verify\_data)*,  
63  
`create_wallet()` *(aries\_cloudagent.indy.sdk.wallet\_setup.IndyWalletCon*  
*method)*, 39  
`created` *(aries\_cloudagent.core.profile.Profile* *prop-*  
*erty)*, 30  
`cred_def_id` *(aries\_cloudagent.revocation.models.revocation\_registry.Rev*  
*ocationRegistry)*, 100  
`credential_definition_id2schema_id()`  
*(aries\_cloudagent.ledger.indy.IndySdkLedger*  
*method)*, 54  
`credential_definition_id2schema_id()`  
*(aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger*  
*method)*, 59  
`credential_definition_in_wallet()`  
*(aries\_cloudagent.indy.issuer.IndyIssuer*  
*method)*, 43  
`credential_revoked()`  
*(aries\_cloudagent.indy.holder.IndyHolder*  
*method)*, 41  
`credential_revoked()`  
*(aries\_cloudagent.indy.sdk.holder.IndySdkHolder*  
*method)*, 37  
`CredentialIssuancePurpose` *(class* *in*  
*aries\_cloudagent.vc.ld\_proofs.purposes.credential\_issuance\_purp*  
125  
`current_active` *(aries\_cloudagent.utils.task\_queue.TaskQueue*  
*property)*, 121  
`current_pending` *(aries\_cloudagent.utils.task\_queue.TaskQueue*  
*property)*, 121  
`current_size` *(aries\_cloudagent.utils.task\_queue.TaskQueue*  
*property)*, 121  
**D**  
`datetime_now()` *(in* *module*  
*aries\_cloudagent.messaging.util)*, 66  
`datetime_to_str()` *(in* *module*  
*aries\_cloudagent.messaging.util)*, 66  
`decode_state_value()` *(in* *module*  
*aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler)*,  
45





property), 109  
 direct\_response\_requested  
 (aries\_cloudagent.transport.inbound.receipt.MessageReceipt property), 109  
 dns\_resolvehost\_end()  
 (aries\_cloudagent.transport.stats.StatsTracer method), 115  
 dns\_resolvehost\_start()  
 (aries\_cloudagent.transport.stats.StatsTracer method), 115  
 DocumentLoader (class in aries\_cloudagent.vc.ld\_proofs.document\_loader), 126  
 DocumentVerificationResult (class in aries\_cloudagent.vc.ld\_proofs.validation\_result), 127  
 done (aries\_cloudagent.cache.base.CacheKeyLock property), 7  
 drain() (aries\_cloudagent.utils.task\_queue.TaskQueue method), 121  
 DroppedAttributeError, 63  
 duration (aries\_cloudagent.resolver.base.ResolutionMetadata property), 98  
**E**  
 ED25519 (aries\_cloudagent.wallet.key\_type.KeyType attribute), 136  
 ED25519\_SIG\_2018 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 23  
 ED25519\_SIG\_2018 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 19  
 EDDSA\_SA\_SIG\_SECP256K1  
 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 23  
 EDDSA\_SA\_SIG\_SECP256K1  
 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 19  
 enabled (aries\_cloudagent.utils.stats.Collector property), 119  
 encode() (in module aries\_cloudagent.messaging.util), 66  
 encode\_hex() (in module aries\_cloudagent.ledger.merkel\_validation.utils), 48  
 encode\_message() (aries\_cloudagent.transport.wire\_format.BaseWireFormat method), 116  
 encode\_message() (aries\_cloudagent.transport.wire\_format.JsonWireFormat method), 116  
 encode\_state\_value() (in module aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler), 45  
 ENDORSER (aries\_cloudagent.ledger.base.Role attribute), 52  
 endpoint (aries\_cloudagent.connections.models.diddoc.Service property), 20  
 endpoint (aries\_cloudagent.connections.models.diddoc.service.Service property), 24  
 ENDPOINT (aries\_cloudagent.ledger.endpoint\_type.EndpointType attribute), 52  
 EndpointType (class in aries\_cloudagent.ledger.endpoint\_type), 52  
 EndpointTypeName (class in aries\_cloudagent.ledger.endpoint\_type), 53  
 enqueue() (aries\_cloudagent.transport.queue.base.BaseMessageQueue method), 113  
 enqueue() (aries\_cloudagent.transport.queue.basic.BasicMessageQueue method), 114  
 epoch\_to\_str() (in module aries\_cloudagent.messaging.util), 66  
 error\_code (aries\_cloudagent.messaging.error.MessageParseError attribute), 65  
 error\_code (aries\_cloudagent.messaging.error.MessagePrepareError attribute), 65  
 error\_code (aries\_cloudagent.transport.error.WireFormatEncodeError attribute), 115  
 error\_code (aries\_cloudagent.transport.error.WireFormatParseError attribute), 115  
 Event (class in aries\_cloudagent.core.event\_bus), 28  
 EventBus (class in aries\_cloudagent.core.event\_bus), 28  
 EventMetadata (class in aries\_cloudagent.core.event\_bus), 28  
 EventWithMetadata (class in aries\_cloudagent.core.event\_bus), 29  
 execute() (in module aries\_cloudagent.commands.help), 8  
 extend() (aries\_cloudagent.config.base.BaseSettings method), 10  
 extend() (aries\_cloudagent.config.plugin\_settings.PluginSettings method), 15  
 extend() (aries\_cloudagent.config.settings.Settings method), 16  
 extract() (aries\_cloudagent.utils.stats.Collector method), 119  
 extract() (aries\_cloudagent.utils.stats.Stats method), 120  
 extract\_params.write\_request() (in module aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler), 45  
**F**  
 fetch() (aries\_cloudagent.storage.base.BaseStorageSearchSession method), 104  
 fetch() (aries\_cloudagent.storage.indy.IndySdkStorageSearch method), 107  
 fetch\_credential\_definition()

[\(aries\\_cloudagent.ledger.indy.IndySdkLedger method\), 55](#)  
[fetch\\_credential\\_definition\(\)](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedger method](#)), 59  
[fetch\\_schema\\_by\\_id\(\)](#) ([aries\\_cloudagent.ledger.indy.IndySdkLedger method](#)), 55  
[fetch\\_schema\\_by\\_id\(\)](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedger method](#)), 59  
[fetch\\_schema\\_by\\_seq\\_no\(\)](#) ([aries\\_cloudagent.ledger.indy.IndySdkLedger method](#)), 55  
[fetch\\_schema\\_by\\_seq\\_no\(\)](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedger method](#)), 59  
[fetch\\_txn\\_author\\_agreement\(\)](#) ([aries\\_cloudagent.ledger.base.BaseLedger method](#)), 50  
[fetch\\_txn\\_author\\_agreement\(\)](#) ([aries\\_cloudagent.ledger.indy.IndySdkLedger method](#)), 55  
[fetch\\_txn\\_author\\_agreement\(\)](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedger method](#)), 59  
[fetch\\_txns\(\)](#) ([in module aries\\_cloudagent.revocation.recover](#)), 102  
[file\\_ext\(\)](#) ([in module aries\\_cloudagent.indy.sdk.wallet\\_plugin](#)), 39  
[find\\_all\\_records\(\)](#) ([aries\\_cloudagent.storage.base.BaseStorage method](#)), 103  
[find\\_all\\_records\(\)](#) ([aries\\_cloudagent.storage.indy.IndySdkStorage method](#)), 105  
[find\\_key\\_pairs\(\)](#) ([aries\\_cloudagent.wallet.key\\_pair.KeyPairStorageManager method](#)), 135  
[find\\_record\(\)](#) ([aries\\_cloudagent.storage.base.BaseStorage method](#)), 103  
[flush\(\)](#) ([aries\\_cloudagent.cache.base.BaseCache method](#)), 6  
[flush\(\)](#) ([aries\\_cloudagent.cache.in\\_memory.InMemoryCache method](#)), 7  
[flush\(\)](#) ([aries\\_cloudagent.utils.task\\_queue.TaskQueue method](#)), 121  
[for\\_plugin\(\)](#) ([aries\\_cloudagent.config.plugin\\_settings.PluginSettings class method](#)), 15  
[for\\_plugin\(\)](#) ([aries\\_cloudagent.config.settings.Settings method](#)), 16  
[fortran\(\)](#) ([aries\\_cloudagent.indy.models.predicate.Predicate property](#)), 35  
[fortran\(\)](#) ([aries\\_cloudagent.indy.models.predicate.Relation property](#)), 35  
[from\\_definition\(\)](#) ([aries\\_cloudagent.revocation.models.revocation\\_registry.RevocationRegistry class method](#)), 100  
[from\\_did\(\)](#) ([aries\\_cloudagent.wallet.did\\_method.DIDMethod method](#)), 133  
[from\\_json\(\)](#) ([aries\\_cloudagent.connections.models.diddoc.DIDDoc class method](#)), 18  
[from\\_json\(\)](#) ([aries\\_cloudagent.connections.models.diddoc.diddoc.DIDDoc class method](#)), 21  
[from\\_json\(\)](#) ([aries\\_cloudagent.protocols.coordinate\\_mediation.mediation static method](#)), 74  
[from\\_key\\_type\(\)](#) ([aries\\_cloudagent.wallet.key\\_type.KeyType class method](#)), 136  
[from\\_metadata\(\)](#) ([aries\\_cloudagent.wallet.did\\_method.DIDMethod method](#)), 133  
[from\\_method\(\)](#) ([aries\\_cloudagent.wallet.did\\_method.DIDMethod method](#)), 133  
[from\\_multicodec\\_name\(\)](#) ([aries\\_cloudagent.wallet.key\\_type.KeyType class method](#)), 137  
[from\\_multicodec\\_prefix\(\)](#) ([aries\\_cloudagent.wallet.key\\_type.KeyType class method](#)), 137  
[from\\_prefixed\\_bytes\(\)](#) ([aries\\_cloudagent.wallet.key\\_type.KeyType class method](#)), 137  
[full\\_verkey\(\)](#) ([in module aries\\_cloudagent.wallet.util](#)), 137  
[future](#) ([aries\\_cloudagent.cache.base.CacheKeyLock property](#)), 7

## G

[GE](#) ([aries\\_cloudagent.indy.models.predicate.Predicate attribute](#)), 35  
[generate\\_ledger\\_rrrecovery\\_txn\(\)](#) ([in module aries\\_cloudagent.revocation.recover](#)), 102  
[generate\\_pr\\_nonce\(\)](#) ([in module aries\\_cloudagent.indy.util](#)), 44  
[genesis\\_hash](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedgerPool property](#)), 61  
[genesis\\_txns](#) ([aries\\_cloudagent.ledger.indy.IndySdkLedgerPool property](#)), 57  
[genesis\\_txns](#) ([aries\\_cloudagent.ledger.indy\\_vdr.IndyVdrLedgerPool property](#)), 61  
[get\(\)](#) ([aries\\_cloudagent.cache.base.BaseCache method](#)), 6  
[get\(\)](#) ([aries\\_cloudagent.cache.in\\_memory.InMemoryCache method](#)), 7  
[get\(\)](#) ([aries\\_cloudagent.connections.models.diddoc.publickey.PublicKeyType static method](#)), 23  
[get\(\)](#) ([aries\\_cloudagent.connections.models.diddoc.PublicKeyType static method](#)), 19  
[get\(\)](#) ([aries\\_cloudagent.indy.models.predicate.Predicate static method](#)), 35  
[get\(\)](#) ([aries\\_cloudagent.ledger.base.Role static method](#)), 32



`get()` (`aries_cloudagent.ledger.endpoint_type.EndpointType` static method), 52  
`get()` (`aries_cloudagent.multitenant.cache.ProfileCache` method), 67  
`get()` (`aries_cloudagent.wallet.did_posture.DIDPosture` static method), 134  
`get_all_endpoints_for_did()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_all_endpoints_for_did()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_all_endpoints_for_did()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_bool()` (`aries_cloudagent.config.base.BaseSettings` method), 10  
`get_credential()` (`aries_cloudagent.indy.holder.IndyHolder` method), 41  
`get_credential()` (`aries_cloudagent.indy.sdk.holder.IndySdkHolder` method), 37  
`get_credential_definition()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_credential_definition()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_credential_definition()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_credentials()` (`aries_cloudagent.indy.sdk.holder.IndySdkHolder` method), 37  
`get_credentials_for_presentation_request_by_referent()` (`aries_cloudagent.indy.sdk.holder.IndySdkHolder` method), 37  
`get_endpoint_for_did()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_endpoint_for_did()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_endpoint_for_did()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_indy_storage()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_int()` (`aries_cloudagent.config.base.BaseSettings` method), 10  
`get_key_for_did()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_key_for_did()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_key_for_did()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_key_pair()` (`aries_cloudagent.wallet.key_pair.KeyPairStorageManager` method), 135  
`get_latest_txn_author_acceptance()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_latest_txn_author_acceptance()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_latest_txn_author_acceptance()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_local_did()` (`aries_cloudagent.wallet.base.BaseWallet` method), 129  
`get_local_did_for_verkey()` (`aries_cloudagent.wallet.base.BaseWallet` method), 129  
`get_local_dids()` (`aries_cloudagent.wallet.base.BaseWallet` method), 129  
`get_mediation_invite_record()` (`aries_cloudagent.protocols.coordinate_mediation.mediation_invite_record` method), 75  
`get_mime_type()` (`aries_cloudagent.indy.holder.IndyHolder` method), 41  
`get_mime_type()` (`aries_cloudagent.indy.sdk.holder.IndySdkHolder` method), 38  
`get_new_trie_with_proof_nodes()` (`aries_cloudagent.ledger.merkel_validation.trie.SubTrie` static method), 48  
`get_nym_role()` (`aries_cloudagent.ledger.base.BaseLedger` method), 50  
`get_nym_role()` (`aries_cloudagent.ledger.indy.IndySdkLedger` method), 55  
`get_nym_role()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger` method), 59  
`get_or_fetch_local_tails_path()` (`aries_cloudagent.revocation.models.revocation_registry.RevocationRegistry` method), 101  
`get_posted_dids()` (`aries_cloudagent.wallet.base.BaseWallet` method), 130  
`get_proof_nodes()` (in module `aries_cloudagent.ledger.merkel_validation.domain_txn_handler`), 45  
`get_provider()` (`aries_cloudagent.config.injector.Injector` method), 13  
`get_public_did()` (`aries_cloudagent.wallet.base.BaseWallet` method), 130  
`get_receiving_tails_local_path()` (`aries_cloudagent.revocation.models.revocation_registry.RevocationRegistry` method), 101  
`get_recipient_keys()` (`aries_cloudagent.transport.wire_format.BaseWireFormat` method), 116  
`get_recipient_keys()` (`aries_cloudagent.transport.wire_format.JsonWireFormat` method), 116

`method`), 117  
`get_record()` (`aries_cloudagent.storage.base.BaseStorage`  
`method`), 103  
`get_record()` (`aries_cloudagent.storage.indy.IndySdkStorage`  
`method`), 106  
`get_revoc_reg_def()`  
(`aries_cloudagent.ledger.base.BaseLedger`  
`method`), 50  
`get_revoc_reg_def()`  
(`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 55  
`get_revoc_reg_def()`  
(`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 59  
`get_revoc_reg_delta()`  
(`aries_cloudagent.ledger.base.BaseLedger`  
`method`), 51  
`get_revoc_reg_delta()`  
(`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 55  
`get_revoc_reg_delta()`  
(`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 59  
`get_revoc_reg_entry()`  
(`aries_cloudagent.ledger.base.BaseLedger`  
`method`), 51  
`get_revoc_reg_entry()`  
(`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 56  
`get_revoc_reg_entry()`  
(`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 60  
`get_schema()` (`aries_cloudagent.ledger.base.BaseLedger`  
`method`), 51  
`get_schema()` (`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 56  
`get_schema()` (`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 60  
`get_signing_key()` (`aries_cloudagent.wallet.base.BaseWallet`  
`method`), 130  
`get_str()` (`aries_cloudagent.config.base.BaseSettings`  
`method`), 10  
`get_txn_author_agreement()`  
(`aries_cloudagent.ledger.base.BaseLedger`  
`method`), 51  
`get_txn_author_agreement()`  
(`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 56  
`get_txn_author_agreement()`  
(`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 60  
`get_uri()` (`aries_cloudagent.askar.store.AskarStoreConfig`  
`method`), 6  
`get_value()` (`aries_cloudagent.config.base.BaseSettings`  
`method`), 10  
`get_value()` (`aries_cloudagent.config.plugin_settings.PluginSettings`  
`method`), 15  
`get_value()` (`aries_cloudagent.config.settings.Settings`  
`method`), 16  
`get_wallet_public_did()`  
(`aries_cloudagent.ledger.indy.IndySdkLedger`  
`method`), 56  
`get_wallet_public_did()`  
(`aries_cloudagent.ledger.indy_vdr.IndyVdrLedger`  
`method`), 60  
`goal_codes_matching_query()`  
(`aries_cloudagent.core.goal_code_registry.GoalCodeRegistry`  
`method`), 29  
`GoalCodeRegistry` (class in  
`aries_cloudagent.core.goal_code_registry`), 29  
`GT` (`aries_cloudagent.indy.models.predicate.Predicate` attribute), 35

## H

`handle_message()` (`aries_cloudagent.transport.outbound.base.BaseOutbound`  
`method`), 110  
`handle_message()` (`aries_cloudagent.transport.outbound.http.HttpTransport`  
`method`), 111  
`handle_message()` (`aries_cloudagent.transport.outbound.ws.WsTransport`  
`method`), 112  
`Handler` (`aries_cloudagent.messaging.base_message.BaseMessage`  
property), 65  
`has()` (`aries_cloudagent.multitenant.cache.ProfileCache`  
`method`), 67  
`has_local_tails_file()`  
(`aries_cloudagent.revocation.models.revocation_registry.RevocationRegistry`  
`method`), 101  
`hash_children()` (`aries_cloudagent.ledger.merkel_validation.hasher.HexTreeHasher`  
`method`), 47  
`hash_children()` (`aries_cloudagent.ledger.merkel_validation.hasher.TreeHasher`  
`method`), 47  
`hash_leaf()` (`aries_cloudagent.ledger.merkel_validation.hasher.HexTreeHasher`  
`method`), 47  
`hash_leaf()` (`aries_cloudagent.ledger.merkel_validation.hasher.TreeHasher`  
`method`), 47  
`hash_of()` (in module  
`aries_cloudagent.ledger.merkel_validation.domain_txn_handler`), 45  
`HexTreeHasher` (class in  
`aries_cloudagent.ledger.merkel_validation.hasher`), 47  
`HttpTransport` (class in  
`aries_cloudagent.transport.outbound.http`), 111  
`id` (`aries_cloudagent.connections.models.diddoc.PublicKey`  
property), 19

[id \(aries\\_cloudagent.connections.models.diddoc.publickey.PublicKey property\), 23](#)  
[id \(aries\\_cloudagent.connections.models.diddoc.ServiceProperty property\), 20](#)  
[id \(aries\\_cloudagent.connections.models.diddoc.service.ServiceProperty property\), 24](#)  
[in\\_time \(aries\\_cloudagent.transport.inbound.receipt.MessageReceipt property\), 109](#)  
[InboundMessage \(class in aries\\_cloudagent.transport.inbound.message\), 108](#)  
[indy \(aries\\_cloudagent.ledger.endpoint\\_type.EndpointType property\), 52](#)  
[indy \(aries\\_cloudagent.ledger.endpoint\\_type.EndpointTypeName property\), 53](#)  
[indy\\_client\\_dir\(\) \(in module aries\\_cloudagent.indy.util\), 44](#)  
[IndyErrorHandler \(class in aries\\_cloudagent.indy.sdk.error\), 36](#)  
[IndyHolder \(class in aries\\_cloudagent.indy.holder\), 40](#)  
[IndyHolderError, 42](#)  
[IndyIssuer \(class in aries\\_cloudagent.indy.issuer\), 42](#)  
[IndyIssuerError, 44](#)  
[IndyIssuerRevocationRegistryFullError, 44](#)  
[IndyOpenWallet \(class in aries\\_cloudagent.indy.sdk.wallet\\_setup\), 39](#)  
[IndySdkHolder \(class in aries\\_cloudagent.indy.sdk.holder\), 36](#)  
[IndySdkLedger \(class in aries\\_cloudagent.ledger.indy\), 54](#)  
[IndySdkLedgerPool \(class in aries\\_cloudagent.ledger.indy\), 57](#)  
[IndySdkLedgerPoolProvider \(class in aries\\_cloudagent.ledger.indy\), 57](#)  
[IndySdkStorage \(class in aries\\_cloudagent.storage.indy\), 105](#)  
[IndySdkStorageSearch \(class in aries\\_cloudagent.storage.indy\), 106](#)  
[IndyVdrLedger \(class in aries\\_cloudagent.ledger.indy\\_vdr\), 58](#)  
[IndyVdrLedgerPool \(class in aries\\_cloudagent.ledger.indy\\_vdr\), 61](#)  
[IndyWalletConfig \(class in aries\\_cloudagent.indy.sdk.wallet\\_setup\), 39](#)  
[init\\_context\(\) \(aries\\_cloudagent.core.plugin\\_registry.PluginRegistry method\), 29](#)  
[inject\(\) \(aries\\_cloudagent.admin.request\\_context.AdminRequestContext method\), 4](#)  
[inject\(\) \(aries\\_cloudagent.config.base.BaseInjector method\), 9](#)  
[inject\(\) \(aries\\_cloudagent.config.injection\\_context.InjectionContext method\), 11](#)  
[inject\(\) \(aries\\_cloudagent.core.profile.Profile method\), 13](#)  
[inject\(\) \(aries\\_cloudagent.core.profile.ProfileSession method\), 30](#)  
[inject\(\) \(aries\\_cloudagent.core.profile.ProfileSession method\), 32](#)  
[inject\\_for\(\) \(aries\\_cloudagent.admin.request\\_context.AdminRequestContext method\), 4](#)  
[inject\\_or\(\) \(aries\\_cloudagent.config.base.BaseInjector method\), 9](#)  
[inject\\_or\(\) \(aries\\_cloudagent.config.injection\\_context.InjectionContext method\), 12](#)  
[inject\\_or\(\) \(aries\\_cloudagent.config.injector.Injector method\), 13](#)  
[inject\\_or\(\) \(aries\\_cloudagent.core.profile.Profile method\), 30](#)  
[inject\\_or\(\) \(aries\\_cloudagent.core.profile.ProfileSession method\), 32](#)  
[InjectionContext \(class in aries\\_cloudagent.config.injection\\_context\), 11](#)  
[InjectionContextError, 12](#)  
[InjectionError, 10](#)  
[injector \(aries\\_cloudagent.admin.request\\_context.AdminRequestContext property\), 4](#)  
[injector \(aries\\_cloudagent.config.injection\\_context.InjectionContext property\), 12](#)  
[injector \(aries\\_cloudagent.config.injection\\_context.Scope property\), 12](#)  
[Injector \(class in aries\\_cloudagent.config.injector\), 13](#)  
[injector\\_for\\_scope\(\) \(aries\\_cloudagent.config.injection\\_context.InjectionContext method\), 12](#)  
[InMemoryCache \(class in aries\\_cloudagent.cache.in\\_memory\), 7](#)  
[InstanceProvider \(class in aries\\_cloudagent.config.provider\), 15](#)  
[InvalidVerificationMethod, 63](#)  
[INVITATION\\_NOT\\_ACCEPTED \(aries\\_cloudagent.protocols.didexchange.v1\\_0.messages.problem attribute\), 76](#)  
[invite \(aries\\_cloudagent.protocols.coordinate\\_mediation.mediation\\_invite property\), 74](#)  
[INVITE\\_RECORD\\_CATEGORY \(aries\\_cloudagent.protocols.coordinate\\_mediation.mediation\\_invite attribute\), 75](#)  
[is\\_external \(aries\\_cloudagent.transport.outbound.http.HttpTransport attribute\), 112](#)  
[is\\_external \(aries\\_cloudagent.transport.outbound.ws.WsTransport attribute\), 112](#)  
[is\\_indy\\_sdk\\_module\\_installed\(\) \(in module aries\\_cloudagent.utils.dependencies\), 119](#)  
[is\\_transaction \(aries\\_cloudagent.core.profile.ProfileSession property\), 32](#)

[is\\_ursa\\_bbs\\_signatures\\_module\\_installed\(\)](#) (in module `aries_cloudagent.utils.dependencies`), 119  
[issuer\\_did](#) (`aries_cloudagent.revocation.models.revocation_info.RevocationInfo` attribute), 101  
[IterSearch](#) (class in `aries_cloudagent.storage.base`), 104  
**J**  
[join\(\)](#) (`aries_cloudagent.transport.queue.base.BaseMessageQueue` method), 113  
[join\(\)](#) (`aries_cloudagent.transport.queue.basic.BasicMessageQueue` method), 114  
[JsonWireFormat](#) (class in `aries_cloudagent.transport.wire_format`), 116  
**K**  
[KEY](#) (`aries_cloudagent.wallet.did_method.DIDMethod` attribute), 133  
[KEY\\_DERIVATION\\_ARGON2I\\_INT](#) (`aries_cloudagent.askar.store.AskarStoreConfig` attribute), 5  
[KEY\\_DERIVATION\\_ARGON2I\\_INT](#) (`aries_cloudagent.indy.sdk.wallet_setup.IndyWalletConfig` attribute), 39  
[KEY\\_DERIVATION\\_ARGON2I\\_MOD](#) (`aries_cloudagent.askar.store.AskarStoreConfig` attribute), 5  
[KEY\\_DERIVATION\\_ARGON2I\\_MOD](#) (`aries_cloudagent.indy.sdk.wallet_setup.IndyWalletConfig` attribute), 39  
[KEY\\_DERIVATION\\_RAW](#) (`aries_cloudagent.askar.store.AskarStoreConfig` attribute), 5  
[KEY\\_DERIVATION\\_RAW](#) (`aries_cloudagent.indy.sdk.wallet_setup.IndyWalletConfig` attribute), 39  
[key\\_type](#) (`aries_cloudagent.wallet.did_info.DIDInfo` property), 132  
[key\\_type](#) (`aries_cloudagent.wallet.did_info.KeyInfo` property), 133  
[key\\_type](#) (`aries_cloudagent.wallet.key_type.KeySpec` property), 136  
[key\\_type](#) (`aries_cloudagent.wallet.key_type.KeyType` property), 137  
[KeyInfo](#) (class in `aries_cloudagent.wallet.did_info`), 133  
[KeyPairStorageManager](#) (class in `aries_cloudagent.wallet.key_pair`), 135  
[KeySpec](#) (class in `aries_cloudagent.wallet.key_type`), 136  
[KeyType](#) (class in `aries_cloudagent.wallet.key_type`), 136  
[KeyTypeException](#), 137  
**L**  
[LE](#) (`aries_cloudagent.indy.models.predicate.Predicate` attribute), 35  
[LedgerConfigError](#), 53  
[LedgerError](#), 53  
[LedgerTransactionError](#), 53  
[LINKED\\_DOMAINS](#) (`aries_cloudagent.ledger.endpoint_type.EndpointType` attribute), 52  
[LinkedDataKeySpec](#) (class in `aries_cloudagent.connections.models.diddoc`), 18  
[LinkedDataKeySpec](#) (class in `aries_cloudagent.connections.models.diddoc.publickey`), 22  
[LinkedDataProofException](#), 127  
[load\\_class\(\)](#) (`aries_cloudagent.utils.classloader.ClassLoader` class method), 117  
[load\\_command\(\)](#) (in module `aries_cloudagent.commands`), 8  
[load\\_document\(\)](#) (`aries_cloudagent.vc.ld_proofs.document_loader.DocumentLoader` method), 126  
[load\\_module\(\)](#) (`aries_cloudagent.utils.classloader.ClassLoader` class method), 118  
[load\\_postgres\\_plugin\(\)](#) (in module `aries_cloudagent.indy.sdk.wallet_plugin`), 39  
[load\\_protocol\\_version\(\)](#) (`aries_cloudagent.core.plugin_registry.PluginRegistry` method), 29  
[load\\_protocols\(\)](#) (`aries_cloudagent.core.plugin_registry.PluginRegistry` method), 29  
[load\\_resource\(\)](#) (in module `aries_cloudagent.config.logging`), 14  
[load\\_subclass\\_of\(\)](#) (`aries_cloudagent.utils.classloader.ClassLoader` class method), 118  
[log\(\)](#) (`aries_cloudagent.utils.stats.Collector` method), 119  
[log\(\)](#) (`aries_cloudagent.utils.stats.Stats` method), 120  
[LOGGER](#) (in module `aries_cloudagent.revocation.recover`), 102  
[LoggingConfigurator](#) (class in `aries_cloudagent.config.logging`), 14  
[lr\\_pad\(\)](#) (`aries_cloudagent.config.banner.Banner` method), 8  
[lsb\(\)](#) (`aries_cloudagent.ledger.merkel_validation.merkel_verifier.MerkleVerifier` method), 48  
[LT](#) (`aries_cloudagent.indy.models.predicate.Predicate` attribute), 35  
**M**  
[main\(\)](#) (in module `aries_cloudagent.commands.help`), 8  
[make\\_credential\\_definition\\_id\(\)](#) (`aries_cloudagent.indy.issuer.IndyIssuer` method), 43  
[make\\_queue\(\)](#) (`aries_cloudagent.transport.queue.basic.BasicMessageQueue` method), 114



`make_schema_id()` (*aries\_cloudagent.indy.issuer.IndyIssuer* method), 43  
`make_state_path_for_attr()` (in module *aries\_cloudagent.indy.issuer.IndyIssuer*), 43  
`make_state_path_for_claim_def()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 45  
`make_state_path_for_nym()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 45  
`make_state_path_for_revoc_def()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 45  
`make_state_path_for_revoc_reg_entry()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 45  
`make_state_path_for_revoc_reg_entry_accum()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 46  
`make_state_path_for_schema()` (in module *aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler*), 46  
`MANAGER_TYPES` (*aries\_cloudagent.core.profile.ProfileManagerProvider* attribute), 31  
`MANAGER_TYPES` (*aries\_cloudagent.multitenant.manager\_provider.MultiTenantManagerProvider* attribute), 68  
`mark()` (*aries\_cloudagent.utils.stats.Collector* method), 119  
`mark_default_invite_as_used()` (*aries\_cloudagent.protocols.coordinate\_mediation.mediator.Mediator* method), 75  
`match` (*aries\_cloudagent.core.event\_bus.EventMetadata* property), 28  
`match()` (*aries\_cloudagent.vc.ld\_proofs.purposes.proof\_purpose.ProofPurpose* method), 125  
`math` (*aries\_cloudagent.indy.models.predicate.Predicate* property), 35  
`math` (*aries\_cloudagent.indy.models.predicate.Relation* property), 35  
`max_active` (*aries\_cloudagent.utils.task\_queue.TaskQueue* property), 121  
`max_creds` (*aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry* property), 101  
`MAX_SIZE` (*aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry* attribute), 100  
`MEDIATION_INVITE_ID` (*aries\_cloudagent.protocols.coordinate\_mediation.mediator.Mediator* attribute), 75  
`MediationInviteRecord` (class in *aries\_cloudagent.protocols.coordinate\_mediation.mediator.Mediator*), 74  
`MediationInviteStore` (class in *aries\_cloudagent.protocols.coordinate\_mediation.mediator.Mediator*), 74  
`merge_revocation_registry_deltas()` (*aries\_cloudagent.indy.issuer.IndyIssuer* method), 43  
`MerkleVerifier` (class in *aries\_cloudagent.ledger.merkel\_validation.merkel\_verifier*), 45  
`message` (*aries\_cloudagent.core.error.BaseError* property), 26  
`message_types` (*aries\_cloudagent.core.protocol\_registry.ProtocolRegistry* property), 33  
`MessageParseError`, 65  
`MessagePrepareError`, 65  
`MessageReceipt` (class in *aries\_cloudagent.transport.inbound.receipt*), 45  
`metadata` (*aries\_cloudagent.core.event\_bus.EventWithMetadata* property), 29  
`metadata` (*aries\_cloudagent.wallet.did\_info.DIDInfo* property), 132  
`metadata` (*aries\_cloudagent.wallet.did\_info.KeyInfo* property), 133  
`metadata` (*aries\_cloudagent.wallet.did\_posture.DIDPosture* property), 134  
`method` (*aries\_cloudagent.wallet.did\_info.DIDInfo* property), 133  
`method_name` (*aries\_cloudagent.wallet.did\_method.DIDMethod* property), 133  
`method_name` (*aries\_cloudagent.wallet.did\_method.DIDMethodSpec* property), 133  
`MIN_SIZE` (*aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry* attribute), 100  
`MissingVerificationMethodError`, 63  
`MockEventBus` (class in *aries\_cloudagent.core.event\_bus*), 29  
`module`  
`aries_cloudagent`, 3  
`aries_cloudagent.admin`, 3  
`aries_cloudagent.admin.base_server`, 3  
`aries_cloudagent.admin.error`, 3  
`aries_cloudagent.admin.request_context`, 4  
`aries_cloudagent.askar`, 5  
`aries_cloudagent.askar.didcomm`, 5  
`aries_cloudagent.askar.store`, 5  
`aries_cloudagent.cache`, 6  
`aries_cloudagent.cache.base`, 6  
`aries_cloudagent.cache.in_memory`, 7  
`aries_cloudagent.commands`, 8  
`aries_cloudagent.commands.help`, 8  
`aries_cloudagent.config`, 8  
`aries_cloudagent.config.banner`, 8  
`aries_cloudagent.config.base`, 9  
`aries_cloudagent.config.base_context`, 11  
`aries_cloudagent.config.error`, 11

- [aries\\_cloudagent.config.injection\\_context, 11](#)
- [aries\\_cloudagent.config.injector, 13](#)
- [aries\\_cloudagent.config.logging, 14](#)
- [aries\\_cloudagent.config.plugin\\_settings, 14](#)
- [aries\\_cloudagent.config.provider, 15](#)
- [aries\\_cloudagent.config.settings, 16](#)
- [aries\\_cloudagent.config.util, 17](#)
- [aries\\_cloudagent.connections, 17](#)
- [aries\\_cloudagent.connections.models, 17](#)
- [aries\\_cloudagent.connections.models.diddoc, 17](#)
- [aries\\_cloudagent.connections.models.diddoc.didkey, 21](#)
- [aries\\_cloudagent.connections.models.diddoc.didkey, 22](#)
- [aries\\_cloudagent.connections.models.diddoc.services, 24](#)
- [aries\\_cloudagent.connections.models.diddoc.util, 25](#)
- [aries\\_cloudagent.core, 26](#)
- [aries\\_cloudagent.core.error, 26](#)
- [aries\\_cloudagent.core.event\\_bus, 28](#)
- [aries\\_cloudagent.core.goal\\_code\\_registry, 29](#)
- [aries\\_cloudagent.core.plugin\\_registry, 29](#)
- [aries\\_cloudagent.core.profile, 30](#)
- [aries\\_cloudagent.core.protocol\\_registry, 33](#)
- [aries\\_cloudagent.core.util, 33](#)
- [aries\\_cloudagent.did, 34](#)
- [aries\\_cloudagent.holder, 34](#)
- [aries\\_cloudagent.indy, 34](#)
- [aries\\_cloudagent.indy.credx, 34](#)
- [aries\\_cloudagent.indy.holder, 40](#)
- [aries\\_cloudagent.indy.issuer, 42](#)
- [aries\\_cloudagent.indy.models, 34](#)
- [aries\\_cloudagent.indy.models.predicate, 34](#)
- [aries\\_cloudagent.indy.sdk, 36](#)
- [aries\\_cloudagent.indy.sdk.error, 36](#)
- [aries\\_cloudagent.indy.sdk.holder, 36](#)
- [aries\\_cloudagent.indy.sdk.util, 38](#)
- [aries\\_cloudagent.indy.sdk.wallet\\_plugin, 39](#)
- [aries\\_cloudagent.indy.sdk.wallet\\_setup, 39](#)
- [aries\\_cloudagent.indy.util, 44](#)
- [aries\\_cloudagent.ledger, 44](#)
- [aries\\_cloudagent.ledger.base, 49](#)
- [aries\\_cloudagent.ledger.endpoint\\_type, 52](#)
- [aries\\_cloudagent.ledger.error, 53](#)
- [aries\\_cloudagent.ledger.indy, 54](#)
- [aries\\_cloudagent.ledger.indy\\_vdr, 58](#)
- [aries\\_cloudagent.ledger.merkel\\_validation, 44](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.constants, 45](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.domain\\_txn\\_hash, 45](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.hasher, 47](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.merkel\\_verification, 47](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.trie, 48](#)
- [aries\\_cloudagent.ledger.merkel\\_validation.utils, 48](#)
- [aries\\_cloudagent.ledger.multiple\\_ledger, 49](#)
- [aries\\_cloudagent.ledger.util, 62](#)
- [aries\\_cloudagent.messaging, 62](#)
- [aries\\_cloudagent.messaging.base\\_message, 64](#)
- [aries\\_cloudagent.messaging.credential\\_definitions, 62](#)
- [aries\\_cloudagent.messaging.decorators, 62](#)
- [aries\\_cloudagent.messaging.error, 65](#)
- [aries\\_cloudagent.messaging.jsonld, 63](#)
- [aries\\_cloudagent.messaging.jsonld.create\\_verify\\_data, 63](#)
- [aries\\_cloudagent.messaging.jsonld.error, 63](#)
- [aries\\_cloudagent.messaging.models, 64](#)
- [aries\\_cloudagent.messaging.schemas, 64](#)
- [aries\\_cloudagent.messaging.util, 65](#)
- [aries\\_cloudagent.multitenant, 66](#)
- [aries\\_cloudagent.multitenant.admin, 66](#)
- [aries\\_cloudagent.multitenant.cache, 67](#)
- [aries\\_cloudagent.multitenant.error, 67](#)
- [aries\\_cloudagent.multitenant.manager\\_provider, 68](#)
- [aries\\_cloudagent.protocols, 68](#)
- [aries\\_cloudagent.protocols.actionmenu, 68](#)
- [aries\\_cloudagent.protocols.actionmenu.definition, 69](#)
- [aries\\_cloudagent.protocols.actionmenu.v1\\_0, 68](#)
- [aries\\_cloudagent.protocols.actionmenu.v1\\_0.handlers, 68](#)
- [aries\\_cloudagent.protocols.actionmenu.v1\\_0.message\\_type, 69](#)
- [aries\\_cloudagent.protocols.actionmenu.v1\\_0.messages, 68](#)
- [aries\\_cloudagent.protocols.actionmenu.v1\\_0.models, 69](#)
- [aries\\_cloudagent.protocols.basicmessage, 69](#)

70  
aries\_cloudagent.protocols.basicmessage.definitions, 75  
70  
76  
aries\_cloudagent.protocols.basicmessage.v1\_0, aries\_cloudagent.protocols.didexchange.v1\_0.messages, 76  
70  
76  
aries\_cloudagent.protocols.basicmessage.v1\_0.handlers, aries\_cloudagent.protocols.didexchange.v1\_0.messages.p 76  
70  
76  
aries\_cloudagent.protocols.basicmessage.v1\_0.message\_types, aries\_cloudagent.protocols.discovery, 77  
70  
aries\_cloudagent.protocols.basicmessage.v1\_0.messages, aries\_cloudagent.protocols.discovery.definition, 77  
70  
aries\_cloudagent.protocols.connections, aries\_cloudagent.protocols.discovery.v1\_0, 77  
70  
aries\_cloudagent.protocols.connections.definition, aries\_cloudagent.protocols.discovery.v1\_0.handlers, 77  
71  
aries\_cloudagent.protocols.connections.v1\_0, aries\_cloudagent.protocols.discovery.v1\_0.message\_type 78  
70  
78  
aries\_cloudagent.protocols.connections.v1\_0.handlers, aries\_cloudagent.protocols.discovery.v1\_0.messages, 78  
70  
aries\_cloudagent.protocols.connections.v1\_0.message\_types, aries\_cloudagent.protocols.discovery.v1\_0.models, 77  
71  
aries\_cloudagent.protocols.connections.v1\_0.messages, aries\_cloudagent.protocols.discovery.v2\_0, 78  
71  
aries\_cloudagent.protocols.connections.v1\_0.models, aries\_cloudagent.protocols.discovery.v2\_0.handlers, 78  
71  
aries\_cloudagent.protocols.coordinate\_mediation, aries\_cloudagent.protocols.discovery.v2\_0.message\_type 78  
72  
aries\_cloudagent.protocols.coordinate\_mediation.definition, aries\_cloudagent.protocols.discovery.v2\_0.messages, 78  
74  
aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_store, aries\_cloudagent.protocols.discovery.v2\_0.models, 78  
74  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0, aries\_cloudagent.protocols.endorse\_transaction, 79  
72  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.controller, aries\_cloudagent.protocols.endorse\_transaction.definition, 79  
73  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.handlers, aries\_cloudagent.protocols.endorse\_transaction.v1\_0, 79  
72  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.message\_types, aries\_cloudagent.protocols.endorse\_transaction.v1\_0.co 79  
74  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.messages, aries\_cloudagent.protocols.endorse\_transaction.v1\_0.ha 79  
72  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.messages.inner, aries\_cloudagent.protocols.endorse\_transaction.v1\_0.me 79  
72  
aries\_cloudagent.protocols.coordinate\_mediation.v1\_0.models, aries\_cloudagent.protocols.endorse\_transaction.v1\_0.mo 79  
73  
aries\_cloudagent.protocols.didcomm\_prefix, 80  
96  
aries\_cloudagent.protocols.didexchange, aries\_cloudagent.protocols.endorse\_transaction.v1\_0.tr 81  
75  
81  
aries\_cloudagent.protocols.didexchange.definition, aries\_cloudagent.protocols.introduction, 81  
77  
aries\_cloudagent.protocols.didexchange.v1\_0, aries\_cloudagent.protocols.introduction.definition, 82  
75  
aries\_cloudagent.protocols.didexchange.v1\_0.handlers, aries\_cloudagent.protocols.introduction.v0\_1, 81





[aries\\_cloudagent.revocation.models.revocation\\_model, 100](#)  
[aries\\_cloudagent.revocation.recover, 102](#)  
[aries\\_cloudagent.storage, 102](#)  
[aries\\_cloudagent.storage.base, 103](#)  
[aries\\_cloudagent.storage.error, 105](#)  
[aries\\_cloudagent.storage.indy, 105](#)  
[aries\\_cloudagent.storage.record, 107](#)  
[aries\\_cloudagent.storage.vc\\_holder, 102](#)  
[aries\\_cloudagent.tails, 107](#)  
[aries\\_cloudagent.tails.base, 107](#)  
[aries\\_cloudagent.tails.error, 108](#)  
[aries\\_cloudagent.transport, 108](#)  
[aries\\_cloudagent.transport.error, 114](#)  
[aries\\_cloudagent.transport.inbound, 108](#)  
[aries\\_cloudagent.transport.inbound.message, 108](#)  
[aries\\_cloudagent.transport.inbound.receipt, 109](#)  
[aries\\_cloudagent.transport.outbound, 110](#)  
[aries\\_cloudagent.transport.outbound.base, 110](#)  
[aries\\_cloudagent.transport.outbound.http, 111](#)  
[aries\\_cloudagent.transport.outbound.status, 112](#)  
[aries\\_cloudagent.transport.outbound.ws, 112](#)  
[aries\\_cloudagent.transport.queue, 113](#)  
[aries\\_cloudagent.transport.queue.base, 113](#)  
[aries\\_cloudagent.transport.queue.basic, 114](#)  
[aries\\_cloudagent.transport.stats, 115](#)  
[aries\\_cloudagent.transport.wire\\_format, 116](#)  
[aries\\_cloudagent.utils, 117](#)  
[aries\\_cloudagent.utils.classloader, 117](#)  
[aries\\_cloudagent.utils.dependencies, 119](#)  
[aries\\_cloudagent.utils.env, 119](#)  
[aries\\_cloudagent.utils.stats, 119](#)  
[aries\\_cloudagent.utils.task\\_queue, 120](#)  
[aries\\_cloudagent.vc, 122](#)  
[aries\\_cloudagent.vc.ld\\_proofs.constants, 126](#)  
[aries\\_cloudagent.vc.ld\\_proofs.document\\_loader, 126](#)  
[aries\\_cloudagent.vc.ld\\_proofs.error, 127](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes, 123](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes.assertion\\_proof\\_purpose, 123](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes.authentication\\_proof\\_purpose, 123](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes.controller\\_proof\\_purpose, 124](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes.credential\\_issuance\\_proof\\_purpose, 125](#)  
[aries\\_cloudagent.vc.ld\\_proofs.purposes.proof\\_purpose, 125](#)  
[aries\\_cloudagent.vc.ld\\_proofs.validation\\_result, 127](#)  
[aries\\_cloudagent.version, 138](#)  
[aries\\_cloudagent.wallet, 128](#)  
[aries\\_cloudagent.wallet.base, 128](#)  
[aries\\_cloudagent.wallet.bbs, 132](#)  
[aries\\_cloudagent.wallet.did\\_info, 132](#)  
[aries\\_cloudagent.wallet.did\\_method, 133](#)  
[aries\\_cloudagent.wallet.did\\_posture, 134](#)  
[aries\\_cloudagent.wallet.error, 135](#)  
[aries\\_cloudagent.wallet.key\\_pair, 135](#)  
[aries\\_cloudagent.wallet.key\\_type, 136](#)  
[aries\\_cloudagent.wallet.models, 128](#)  
[aries\\_cloudagent.wallet.util, 137](#)  
[ModuleLoadError, 118](#)  
[moniker \(\*aries\\_cloudagent.wallet.did\\_posture.DIDPosture\* property\), 134](#)  
[moniker \(\*aries\\_cloudagent.wallet.did\\_posture.DIDPostureSpec\* property\), 134](#)  
[multicodec\\_name \(\*aries\\_cloudagent.wallet.key\\_type.KeySpec\* property\), 136](#)  
[multicodec\\_name \(\*aries\\_cloudagent.wallet.key\\_type.KeyType\* property\), 137](#)  
[multicodec\\_prefix \(\*aries\\_cloudagent.wallet.key\\_type.KeySpec\* property\), 136](#)  
[multicodec\\_prefix \(\*aries\\_cloudagent.wallet.key\\_type.KeyType\* property\), 137](#)  
[MultitenantManagerProvider \(class in \*aries\\_cloudagent.multitenant.manager\\_provider\*\), 68](#)

## N

[name \(\*aries\\_cloudagent.askar.store.AskarOpenStore\* property\), 5](#)  
[name \(\*aries\\_cloudagent.config.injection\\_context.Scope\* property\), 12](#)  
[name \(\*aries\\_cloudagent.core.profile.Profile\* property\), 31](#)  
[name \(\*aries\\_cloudagent.indy.sdk.wallet\\_setup.IndyOpenWallet\* property\), 39](#)  
[native \(\*aries\\_cloudagent.resolver.base.BaseDIDResolver\* property\), 97](#)  
[NATIVE \(\*aries\\_cloudagent.resolver.base.ResolverType\* attribute\), 98](#)  
[NETWORK\\_MONITOR \(\*aries\\_cloudagent.ledger.base.Role\* property\), 98](#)  
[NEW \(\*aries\\_cloudagent.protocols.didcomm\\_prefix.DIDCommPrefix\* property\), 96](#)

no (aries\_cloudagent.indy.models.predicate.Relation property), 35

NoDefaultMediationInviteException, 75

NON\_NATIVE (aries\_cloudagent.resolver.base.ResolverType attribute), 98

notify() (aries\_cloudagent.core.event\_bus.EventBus method), 28

notify() (aries\_cloudagent.core.event\_bus.MockEventBus method), 29

notify() (aries\_cloudagent.core.profile.Profile method), 31

notify\_endorse\_did\_event() (in module aries\_cloudagent.wallet.util), 137

notify\_register\_did\_event() (in module aries\_cloudagent.ledger.util), 62

now() (aries\_cloudagent.utils.stats.Timer class method), 120

nym\_to\_did() (aries\_cloudagent.ledger.base.BaseLedger method), 51

nym\_to\_did() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 56

nym\_to\_did() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60

## O

ok\_did() (in module aries\_cloudagent.connections.models.diddoc.util), 25

OLD (aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix attribute), 96

open() (aries\_cloudagent.core.profile.ProfileManager method), 31

open() (aries\_cloudagent.ledger.indy.IndySdkLedgerPool method), 57

open() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedgerPool method), 61

open\_store() (aries\_cloudagent.askar.store.AskarStore method), 6

open\_wallet() (aries\_cloudagent.indy.sdk.wallet\_setup.IndyWalletConfig method), 40

ordinal (aries\_cloudagent.wallet.did\_posture.DIDPosture property), 134

ordinal (aries\_cloudagent.wallet.did\_posture.DIDPostureSpec property), 134

OutboundDeliveryError, 111

OutboundSendStatus (class in aries\_cloudagent.transport.outbound.status), 112

OutboundTransportError, 111

OutboundTransportRegistrationError, 111

## P

pack\_message() (aries\_cloudagent.wallet.base.BaseWallet method), 130

pad() (in module aries\_cloudagent.wallet.util), 138

parent (aries\_cloudagent.cache.base.CacheKeyLock property), 7

parent\_thread\_id (aries\_cloudagent.transport.inbound.receipt.Message property), 109

parse\_attr\_txn() (in module aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler), 46

parse\_message() (aries\_cloudagent.transport.wire\_format.BaseWireFormat method), 116

parse\_message() (aries\_cloudagent.transport.wire\_format.JsonWireFormat method), 117

parse\_type\_string() (aries\_cloudagent.core.protocol\_registry.ProtocolRegistry method), 33

pattern (aries\_cloudagent.core.event\_bus.EventMetadata property), 28

payload (aries\_cloudagent.core.event\_bus.Event property), 28

PendingTask (class in aries\_cloudagent.utils.task\_queue), 120

plugin\_names (aries\_cloudagent.core.plugin\_registry.PluginRegistry property), 29

PluginRegistry (class in aries\_cloudagent.core.plugin\_registry), 29

plugins (aries\_cloudagent.core.plugin\_registry.PluginRegistry property), 30

PluginSettings (class in aries\_cloudagent.config.plugin\_settings), 14

pool\_handle (aries\_cloudagent.ledger.indy.IndySdkLedger property), 56

pool\_handle (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger property), 60

pool\_name (aries\_cloudagent.ledger.indy.IndySdkLedger property), 56

pool\_name (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger property), 60

POST (aries\_cloudagent.config.routes) (aries\_cloudagent.core.plugin\_registry.PluginRegistry method), 30

POSTED (aries\_cloudagent.wallet.did\_posture.DIDPosture attribute), 134

posted (aries\_cloudagent.wallet.did\_posture.DIDPostureSpec property), 134

Predicate (class in aries\_cloudagent.indy.models.predicate), 34

prepare\_attr\_for\_state() (in module aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler), 46

prepare\_claim\_def\_for\_state() (in module aries\_cloudagent.ledger.merkel\_validation.domain\_txn\_handler), 46

prepare\_disclosed() (aries\_cloudagent.core.protocol\_registry.ProtocolRegistry

```

method), 33
prepare_for_state_read() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 9
46
prepare_for_state_write() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 9
46
prepare_get_attr_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 20
46
prepare_get_claim_def_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 24
46
prepare_get_nym_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 4
46
prepare_get_revoc_def_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 32
46
prepare_get_revoc_reg_delta_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 30
46
prepare_get_revoc_reg_entry_accum_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 27
46
prepare_get_revoc_reg_entry_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 31
46
prepare_get_schema_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 27
46
prepare_nym_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 27
46
prepare_revoc_def_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 125
46
prepare_revoc_reg_entry_accum_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 127
47
prepare_revoc_reg_entry_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 27
47
prepare_schema_for_state() (in module aries_cloudagent.ledger.merkel_validation.domain_txn_handler, 33
47
print_banner() (aries_cloudagent.config.logging.LoggingConfiguration, 14
print_border() (aries_cloudagent.config.banner.Banner, 9
print_list() (aries_cloudagent.config.banner.Banner, 9
print_spacer() (aries_cloudagent.config.banner.Banner, 9

print_subtitle() (aries_cloudagent.config.banner.Banner, 9
print_title() (aries_cloudagent.config.banner.Banner, 9
print_version() (aries_cloudagent.config.banner.Banner, 9
priority (aries_cloudagent.connections.models.diddoc.Service, 20
priority (aries_cloudagent.connections.models.diddoc.service.Service, 24
ProblemReportReason (class in aries_cloudagent.protocols.didexchange.v1_0.messages.problem_report, 76
profile (aries_cloudagent.admin.request_context.AdminRequestContext, 4
profile (aries_cloudagent.core.profile.ProfileSession, 32
PROFILE (aries_cloudagent.ledger.endpoint_type.EndpointType, 52
Profile (class in aries_cloudagent.core.profile), 30
ProfileCache (class in aries_cloudagent.multitenant.cache), 67
ProfileDuplicateError, 27
ProfileError (class in aries_cloudagent.core.profile), 31
ProfileManager (class in aries_cloudagent.core.profile), 31
ProfileManagerProvider (class in aries_cloudagent.core.profile), 31
ProfileNotFoundError, 27
ProfileSession (class in aries_cloudagent.core.profile), 31
ProfileSessionInactiveError, 27
ProfilePurpose (class in aries_cloudagent.vc.ld_proofs.purposes.proof_purpose), 125
ProofResult (class in aries_cloudagent.vc.ld_proofs.validation_result), 127
ProtocolDehydrationValidationError, 27
ProtocolMinorVersionNotSupported, 27
ProtocolRegistry (class in aries_cloudagent.core.protocol_registry), 33
protocols (aries_cloudagent.core.protocol_registry.ProtocolRegistry, 33
protocols_matching_query()
provide() (aries_cloudagent.config.base.BaseProvider, 10
provide() (aries_cloudagent.config.provider.CachedProvider, 15
provide() (aries_cloudagent.config.provider.ClassProvider, 15

```

`provide()` (*aries\_cloudagent.config.provider.InstanceProvider* *aries\_cloudagent.wallet.util*), 138  
*method*), 15  
`provide()` (*aries\_cloudagent.config.provider.StatsProvider* *property*), 109  
*method*), 16  
`provide()` (*aries\_cloudagent.core.profile.ProfileManagerProvider* *property*), 51  
*method*), 31  
`provide()` (*aries\_cloudagent.ledger.indy.IndySdkLedgerPoolProvide**property*), 56  
*method*), 57  
`provide()` (*aries\_cloudagent.multitenant.manager\_provider.MultitenantManagerProvider*  
*method*), 68  
`provision()` (*aries\_cloudagent.core.profile.ProfileManager* *property*), 122  
*method*), 31  
`pubkey` (*aries\_cloudagent.connections.models.diddoc.DIDDoc* *property*), 20  
*property*), 18  
`pubkey` (*aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc**property*), 24  
*property*), 21  
`PUBLIC` (*aries\_cloudagent.wallet.did\_posture.DIDPosture* *property*), 109  
*attribute*), 134  
`public` (*aries\_cloudagent.wallet.did\_posture.DIDPostureSpec* *property*), 110  
*property*), 134  
`PublicKey` (class in *aries\_cloudagent.connections.models.diddoc*), 19  
`PublicKey` (class in *aries\_cloudagent.connections.models.diddoc*), 22  
`PublicKeyType` (class in *aries\_cloudagent.connections.models.diddoc*), 19  
*aries\_cloudagent.connections.models.diddoc*),  
`PublicKeyType` (class in *aries\_cloudagent.connections.models.diddoc*), 23  
*aries\_cloudagent.connections.models.diddoc*),  
`PurposeResult` (class in *aries\_cloudagent.vc.ld\_proofs.validation\_result*), 127  
*aries\_cloudagent.vc.ld\_proofs.validation\_result*),  
`put()` (*aries\_cloudagent.multitenant.cache.ProfileCache* *method*), 67  
*method*), 122  
`put()` (*aries\_cloudagent.utils.task\_queue.TaskQueue* *method*), 122

**Q**  
`qualify()` (*aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix* *method*), 96  
*method*), 96  
`qualify()` (in module *aries\_cloudagent.protocols.didcomm\_prefix*), 96  
`qualify_all()` (*aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix* *method*), 96  
*class method*), 96  
`qualify_current()` (*aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix* *method*), 96  
*static method*), 96  
`QUEUED_FOR_DELIVERY` (*aries\_cloudagent.transport.outbound.status.OutboundSendStatus* *attribute*), 112

**R**  
`random_seed()` (in module *aries\_cloudagent.wallet.util*), 138

`raw_message` (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* *property*), 109  
`read_only` (*aries\_cloudagent.ledger.base.BaseLedger* *property*), 51  
`read_only` (*aries\_cloudagent.ledger.indy.IndySdkLedger* *property*), 56  
`read_only` (*aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger* *property*), 60  
`ready` (*aries\_cloudagent.utils.task\_queue.TaskQueue* *property*), 122  
`recip_keys` (*aries\_cloudagent.connections.models.diddoc.Service* *property*), 20  
`recip_keys` (*aries\_cloudagent.connections.models.diddoc.service.Service* *property*), 24  
`recipient_did` (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* *property*), 110  
`recipient_did_public` (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* *property*), 110  
`recipient_verkey` (*aries\_cloudagent.transport.inbound.receipt.MessageReceipt* *property*), 110  
`RecipientKeysError`, 114  
`RECORD_TYPE_MIME_TYPES` (*aries\_cloudagent.indy.holder.IndyHolder* *attribute*), 40  
`reg_def` (*aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry* *property*), 101  
`reg_def_type` (*aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry* *property*), 101  
`register()` (*aries\_cloudagent.resolver.did\_resolver\_registry.DIDResolverRegistry* *method*), 99  
`register_admin_routes()` (*aries\_cloudagent.core.plugin\_registry.PluginRegistry* *method*), 30  
`register_controllers()` (*aries\_cloudagent.core.goal\_code\_registry.GoalCodeRegistry* *method*), 29  
`register_controllers()` (*aries\_cloudagent.core.protocol\_registry.ProtocolRegistry* *method*), 33  
`register_message_types()` (*aries\_cloudagent.core.protocol\_registry.ProtocolRegistry* *method*), 33  
`register_nym()` (*aries\_cloudagent.ledger.base.BaseLedger* *method*), 51  
`register_nym()` (*aries\_cloudagent.ledger.indy.IndySdkLedger* *method*), 56  
`register_nym()` (*aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger* *method*), 60  
`register_package()` (*aries\_cloudagent.core.plugin\_registry.PluginRegistry* *method*), 30  
`register_plugin()` (*aries\_cloudagent.core.plugin\_registry.PluginRegistry* *method*), 30



[register\\_protocol\\_events\(\)](#) (method), 99  
[\(aries\\_cloudagent.core.plugin\\_registry.PluginRegistry.resolve\\_message\\_class\(\)](#) (method), 30  
[registry\\_id](#) ([aries\\_cloudagent.revocation.models.revocation\\_registry.RevocationRegistry](#) property), 101  
[Relation](#) (class in [aries\\_cloudagent.indy.models.predicate](#)), 35  
[release\(\)](#) ([aries\\_cloudagent.cache.base.BaseCache](#) method), 6  
[release\(\)](#) ([aries\\_cloudagent.cache.base.CacheKeyLock](#) method), 7  
[remove\(\)](#) ([aries\\_cloudagent.core.profile.Profile](#) method), 31  
[remove\(\)](#) ([aries\\_cloudagent.multitenant.cache.ProfileCache](#) method), 67  
[remove\\_store\(\)](#) ([aries\\_cloudagent.askar.store.AskarStoreConfig](#) method), 6  
[remove\\_wallet\(\)](#) ([aries\\_cloudagent.indy.sdk.wallet\\_setup.IndyWalletConfig](#) method), 40  
[replace\\_local\\_did\\_metadata\(\)](#) ([aries\\_cloudagent.wallet.base.BaseWallet](#) method), 130  
[replace\\_signing\\_key\\_metadata\(\)](#) ([aries\\_cloudagent.wallet.base.BaseWallet](#) method), 130  
[REPLY\\_MODE\\_ALL](#) ([aries\\_cloudagent.transport.inbound.receipt.MessageReceipt](#) attribute), 109  
[REPLY\\_MODE\\_NONE](#) ([aries\\_cloudagent.transport.inbound.receipt.MessageReceipt](#) attribute), 109  
[REPLY\\_MODE\\_THREAD](#) ([aries\\_cloudagent.transport.inbound.receipt.MessageReceipt](#) attribute), 109  
[request\\_end\(\)](#) ([aries\\_cloudagent.transport.stats.StatsTracer](#) method), 115  
[REQUEST\\_NOT\\_ACCEPTED](#) ([aries\\_cloudagent.protocols.didexchange.v1\\_0.messages.problem\\_report.ProblemReportReason](#) attribute), 76  
[REQUEST\\_PROCESSING\\_ERROR](#) ([aries\\_cloudagent.protocols.didexchange.v1\\_0.messages.problem\\_report.ProblemReportReason](#) attribute), 76  
[request\\_start\(\)](#) ([aries\\_cloudagent.transport.stats.StatsTracer](#) method), 115  
[reset\(\)](#) ([aries\\_cloudagent.transport.queue.base.BaseMessageQueue](#) method), 113  
[reset\(\)](#) ([aries\\_cloudagent.transport.queue.basic.BasicMessageQueue](#) method), 114  
[reset\(\)](#) ([aries\\_cloudagent.utils.stats.Collector](#) method), 119  
[ResolutionMetadata](#) (class in [aries\\_cloudagent.resolver.base](#)), 98  
[ResolutionResult](#) (class in [aries\\_cloudagent.resolver.base](#)), 98  
[resolve\(\)](#) ([aries\\_cloudagent.resolver.base.BaseDIDResolver](#) method), 97  
[resolve\(\)](#) ([aries\\_cloudagent.resolver.did\\_resolver.DIDResolver](#) method), 99  
[resolve\\_message\\_class\(\)](#) ([aries\\_cloudagent.core.protocol\\_registry.ProtocolRegistry](#) method), 30  
[resolve\\_with\\_metadata\(\)](#) ([aries\\_cloudagent.resolver.did\\_resolver.DIDResolver](#) method), 99  
[resolved](#) ([aries\\_cloudagent.utils.classloader.DeferLoad](#) property), 118  
[resolver](#) ([aries\\_cloudagent.resolver.base.ResolutionMetadata](#) property), 98  
[resolver\\_type](#) ([aries\\_cloudagent.resolver.base.ResolutionMetadata](#) property), 98  
[ResolverError](#), 98  
[resolvers](#) ([aries\\_cloudagent.resolver.did\\_resolver\\_registry.DIDResolverRegistry](#) property), 99  
[ResolverType](#) (class in [aries\\_cloudagent.resolver.base](#)), 98  
[resource\(\)](#) (in [aries\\_cloudagent.connections.models.diddoc.util](#) module), 25  
[RESPONSE\\_NOT\\_ACCEPTED](#) ([aries\\_cloudagent.protocols.didexchange.v1\\_0.messages.problem\\_report.ProblemReportReason](#) attribute), 76  
[RESPONSE\\_PROCESSING\\_ERROR](#) ([aries\\_cloudagent.protocols.didexchange.v1\\_0.messages.problem\\_report.ProblemReportReason](#) attribute), 76  
[result](#) ([aries\\_cloudagent.cache.base.CacheKeyLock](#) property), 7  
[result\\_message\\_receipt](#) ([aries\\_cloudagent.utils.stats.Collector](#) property), 120  
[retrieve\\_tails\(\)](#) ([aries\\_cloudagent.revocation.models.revocation\\_registry.RevocationRegistry](#) method), 101  
[retrieved\\_time](#) ([aries\\_cloudagent.resolver.base.ResolutionMetadata](#) property), 98  
[RevocationError](#), 101  
[RevocationNotSupportedError](#), 101  
[RevocationRegistry](#) (class in [aries\\_cloudagent.revocation.models.revocation\\_registry](#)), 100  
[RevocationRegistryBadSizeError](#), 101  
[RevocationRecoveryException](#), 102  
[revoke\\_credentials\(\)](#) ([aries\\_cloudagent.indy.issuer.IndyIssuer](#) method), 43  
[Role](#) (class in [aries\\_cloudagent.ledger.base](#)), 52  
[ROLE\\_REMOVE](#) ([aries\\_cloudagent.ledger.base.Role](#) attribute), 52  
[roll\\_up](#) ([aries\\_cloudagent.core.error.BaseError](#) property), 26  
[rollback\(\)](#) ([aries\\_cloudagent.core.profile.ProfileSession](#) method), 32  
[root\\_hash](#) ([aries\\_cloudagent.ledger.merkel\\_validation.trie.SubTrie](#) property), 48

ROOT\_SCOPE (aries\_cloudagent.config.injection\_context.InjectionContext attribute), 11  
 rotate\_did\_keypair\_apply() (aries\_cloudagent.wallet.base.BaseWallet method), 130  
 rotate\_did\_keypair\_start() (aries\_cloudagent.wallet.base.BaseWallet method), 130  
 rotate\_public\_did\_keypair() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
 rotate\_public\_did\_keypair() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 56  
 rotate\_public\_did\_keypair() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60  
 routing\_keys (aries\_cloudagent.connections.models.diddoc.Service attribute), 20  
 routing\_keys (aries\_cloudagent.connections.models.diddoc.service.Service attribute), 24  
 RSA\_SIG\_2018 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 23  
 RSA\_SIG\_2018 (aries\_cloudagent.connections.models.diddoc.PublicKey attribute), 19  
 run() (aries\_cloudagent.utils.task\_queue.TaskQueue method), 122  
 run\_command() (in module aries\_cloudagent.commands), 8  
**S**  
 scan\_subpackages() (aries\_cloudagent.utils.classloader.ClassLoader class method), 118  
 schemes (aries\_cloudagent.transport.outbound.http.HttpTransport attribute), 112  
 schemes (aries\_cloudagent.transport.outbound.ws.WsTransport attribute), 113  
 Scope (class in aries\_cloudagent.config.injection\_context), 12  
 scope\_name (aries\_cloudagent.config.injection\_context.InjectionContext attribute), 12  
 search\_records() (aries\_cloudagent.storage.base.BaseStorageSearch method), 104  
 search\_records() (aries\_cloudagent.storage.indy.IndySdkStorage method), 106  
 send\_revoc\_reg\_def() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
 send\_revoc\_reg\_def() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 56  
 send\_revoc\_reg\_def() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60  
 send\_revoc\_reg\_entry() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
 send\_revoc\_reg\_entry() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 56  
 send\_revoc\_reg\_entry() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60  
 sender\_did (aries\_cloudagent.transport.inbound.receipt.MessageReceipt property), 110  
 sender\_verkey (aries\_cloudagent.transport.inbound.receipt.MessageReceipt property), 110  
 SENT\_TO\_EXTERNAL\_QUEUE (aries\_cloudagent.transport.outbound.status.OutboundSendStatus attribute), 112  
 SENT\_TO\_SESSION (aries\_cloudagent.transport.outbound.status.OutboundSendStatus attribute), 112  
 serialize() (aries\_cloudagent.connections.models.diddoc.DIDDoc attribute), 18  
 serialize() (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc attribute), 18  
 serialize() (aries\_cloudagent.messaging.base\_message.BaseMessage attribute), 65  
 serialize() (aries\_cloudagent.resolver.base.ResolutionMetadata method), 98  
 serialize() (aries\_cloudagent.resolver.base.ResolutionResult method), 98  
 service (aries\_cloudagent.connections.models.diddoc.DIDDoc attribute), 18  
 service (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc attribute), 22  
 Service (class in aries\_cloudagent.connections.models.diddoc), 20  
 Service (class in aries\_cloudagent.connections.models.diddoc.service), 24  
 session() (aries\_cloudagent.admin.request\_context.AdminRequestContext method), 4  
 session() (aries\_cloudagent.core.profile.Profile method), 31  
 set() (aries\_cloudagent.cache.base.BaseCache method), 6  
 set() (aries\_cloudagent.cache.in\_memory.InMemoryCache method), 7  
 set() (aries\_cloudagent.connections.models.diddoc.DIDDoc method), 18  
 set() (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc method), 22  
 set() (aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix static method), 96  
 set\_default() (aries\_cloudagent.config.settings.Settings method), 16  
 set\_did\_endpoint() (aries\_cloudagent.wallet.base.BaseWallet method), 131

[set\\_public\\_did\(\)](#) (*aries\_cloudagent.wallet.base.BaseWallet* method), 131  
[set\\_result\(\)](#) (*aries\_cloudagent.cache.base.CacheKeyLock* method), 7  
[set\\_root\\_hash\(\)](#) (*aries\_cloudagent.ledger.merkel\_validation\_utils* method), 48  
[set\\_urlsaf\\_b64\(\)](#) (in module *aries\_cloudagent.wallet.util*), 138  
[set\\_value\(\)](#) (*aries\_cloudagent.config.settings.Settings* method), 16  
[settings](#) (*aries\_cloudagent.admin.request\_context.AdminRequestContext* property), 4  
[settings](#) (*aries\_cloudagent.config.injection\_context.Injector* property), 12  
[settings](#) (*aries\_cloudagent.config.injector.Injector* property), 13  
[settings](#) (*aries\_cloudagent.core.profile.Profile* property), 31  
[settings](#) (*aries\_cloudagent.core.profile.ProfileSession* property), 32  
[Settings](#) (class in *aries\_cloudagent.config.settings*), 16  
[SettingsError](#), 11  
[setup\(\)](#) (*aries\_cloudagent.resolver.base.BaseDIDResolver* method), 97  
[setup\(\)](#) (in module *aries\_cloudagent.resolver*), 96  
[sha3\\_256\(\)](#) (in module *aries\_cloudagent.ledger.merkel\_validation\_utils*), 48  
[sign\\_message\(\)](#) (*aries\_cloudagent.wallet.base.BaseWallet* method), 131  
[sign\\_messages\\_bls12381g2\(\)](#) (in module *aries\_cloudagent.wallet.bbs*), 132  
[SignatureTypeError](#), 64  
[socket\\_connect\\_start\(\)](#) (*aries\_cloudagent.transport.stats.StatsTracer* method), 115  
[SOV](#) (*aries\_cloudagent.wallet.did\_method.DIDMethod* attribute), 133  
[specification\(\)](#) (*aries\_cloudagent.connections.models.diddoc.PublicKey* method), 23  
[specification\(\)](#) (*aries\_cloudagent.connections.models.diddoc.PublicKey* method), 20  
[specifier](#) (*aries\_cloudagent.connections.models.diddoc.LinkedDataKeySpec* property), 19  
[specifier](#) (*aries\_cloudagent.connections.models.diddoc.PublicKey* property), 22  
[specifier](#) (*aries\_cloudagent.connections.models.diddoc.PublicKey* property), 24  
[specifier](#) (*aries\_cloudagent.connections.models.diddoc.PublicKey* property), 20  
[start\(\)](#) (*aries\_cloudagent.admin.base\_server.BaseAdminServer* method), 3  
[start\(\)](#) (*aries\_cloudagent.transport.outbound.base.BaseOutboundTransport* method), 111  
[start\(\)](#) (*aries\_cloudagent.transport.outbound.http.HttpTransport* method), 112  
[start\(\)](#) (*aries\_cloudagent.transport.outbound.ws.WsTransport* method), 113  
[start\\_time\(\)](#) (*aries\_cloudagent.utils.stats.Timer* method), 120  
[start\\_scope\(\)](#) (*aries\_cloudagent.config.injection\_context.Injector* method), 12  
[StartupError](#), 27  
[Stats](#) (class in *aries\_cloudagent.utils.stats*), 120  
[StatsProvider](#) (class in *aries\_cloudagent.config.provider*), 16  
[StatsTracer](#) (class in *aries\_cloudagent.transport.stats*), 115  
[STEWARDSHIP](#) (*aries\_cloudagent.ledger.base.Role* attribute), 52  
[stop\(\)](#) (*aries\_cloudagent.admin.base\_server.BaseAdminServer* method), 3  
[stop\(\)](#) (*aries\_cloudagent.transport.outbound.base.BaseOutboundTransport* method), 111  
[stop\(\)](#) (*aries\_cloudagent.transport.outbound.http.HttpTransport* method), 112  
[stop\(\)](#) (*aries\_cloudagent.transport.outbound.ws.WsTransport* method), 113  
[stop\(\)](#) (*aries\_cloudagent.transport.queue.base.BaseMessageQueue* method), 113  
[stop\(\)](#) (*aries\_cloudagent.transport.queue.basic.BasicMessageQueue* method), 114  
[stop\(\)](#) (*aries\_cloudagent.utils.stats.Timer* method), 120  
[storage\\_path\(\)](#) (in module *aries\_cloudagent.utils.env*), 119  
[StorageDuplicateError](#), 105  
[StorageError](#), 105  
[StorageNotFoundError](#), 105  
[StorageRecord](#) (class in *aries\_cloudagent.storage.record*), 107  
[StorageSearchError](#), 105  
[store\(\)](#) (*aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invitation* method), 105  
[store\\_credential\(\)](#) (*aries\_cloudagent.indy.holder.IndyHolder* method), 105  
[store\\_credential\(\)](#) (*aries\_cloudagent.indy.sdk.holder.IndySdkHolder* method), 105  
[store\\_key\\_pair\(\)](#) (*aries\_cloudagent.wallet.key\_pair.KeyPairStorageManager* method), 105  
[str\\_to\\_b64\(\)](#) (in module *aries\_cloudagent.wallet.util*), 105  
[str\\_to\\_datetime\(\)](#) (in module *aries\_cloudagent.messaging.util*), 66  
[str\\_to\\_epoch\(\)](#) (in module *aries\_cloudagent.messaging.util*), 66  
[submit\\_get\\_nym\\_request\(\)](#) (*aries\_cloudagent.ledger.indy.IndySdkLedger* method), 56

submit\_get\_nym\_request() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60  
 subscribe() (aries\_cloudagent.core.event\_bus.EventBus method), 28  
 SubTrie (class in aries\_cloudagent.ledger.merkel\_validation.trie), 48  
 supported\_did\_regex (aries\_cloudagent.resolver.base.BaseDIDResolver property), 97  
 supported\_key\_types (aries\_cloudagent.wallet.did\_method.DIDMethod property), 133  
 supported\_key\_types (aries\_cloudagent.wallet.did\_method.DIDMethod property), 134  
 supported\_methods (aries\_cloudagent.resolver.base.BaseDIDResolver property), 97  
 supports() (aries\_cloudagent.resolver.base.BaseDIDResolver method), 97  
 supports\_key\_type() (aries\_cloudagent.wallet.did\_method.DIDMethod method), 133  
 supports\_rotation (aries\_cloudagent.wallet.did\_method.DIDMethod property), 133  
 supports\_rotation (aries\_cloudagent.wallet.did\_method.DIDMethod property), 134  
**T**  
 taa\_digest() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
 taa\_rough\_timestamp() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 56  
 taa\_rough\_timestamp() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 60  
 tag (aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry property), 101  
 tails\_hash (aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry property), 101  
 tails\_local\_path (aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry property), 101  
 tails\_path() (in module aries\_cloudagent.indy.util), 44  
 tails\_public\_uri (aries\_cloudagent.revocation.models.revocation\_registry.RevocationRegistry property), 101  
 TailsServerNotConfiguredError, 108  
 task (aries\_cloudagent.utils.task\_queue.PendingTask property), 121  
 task\_done() (aries\_cloudagent.transport.queue.base.BaseMessageQueue method), 113  
 task\_done() (aries\_cloudagent.transport.queue.basic.BasicMessageQueue method), 114  
 task\_exc\_info() (in module aries\_cloudagent.utils.task\_queue), 122  
 TaskQueue (class in aries\_cloudagent.utils.task\_queue), 121  
 term (aries\_cloudagent.vc.ld\_proofs.purposes.assertion\_proof\_purpose.AssertionProofPurpose attribute), 123  
 term (aries\_cloudagent.vc.ld\_proofs.purposes.authentication\_proof\_purpose.AuthenticationProofPurpose attribute), 124  
 test\_context() (aries\_cloudagent.admin.request\_context.AdminRequestContext class method), 4  
 thread\_id (aries\_cloudagent.transport.inbound.receipt.MessageReceipt property), 110  
 time\_now() (in module aries\_cloudagent.messaging.util), 66  
 Timer (class in aries\_cloudagent.utils.stats), 120  
 timer() (aries\_cloudagent.utils.stats.Collector method), 120  
 to\_dict() (aries\_cloudagent.connections.models.diddoc.PublicKey method), 19  
 to\_dict() (aries\_cloudagent.connections.models.diddoc.publickey.PublicKey method), 23  
 to\_dict() (aries\_cloudagent.connections.models.diddoc.Service method), 20  
 to\_dict() (aries\_cloudagent.connections.models.diddoc.service.Service method), 24  
 to\_indy\_json() (aries\_cloudagent.ledger.base.Role method), 52  
 to\_int() (aries\_cloudagent.indy.models.predicate.Predicate static method), 35  
 to\_json() (aries\_cloudagent.connections.models.diddoc.DIDDoc method), 18  
 to\_json() (aries\_cloudagent.connections.models.diddoc.diddoc.DIDDoc method), 22  
 to\_json() (aries\_cloudagent.protocols.coordinate\_mediation.mediation\_info.MediationInfo method), 74  
 token() (aries\_cloudagent.ledger.base.Role method), 52  
 topic (aries\_cloudagent.core.event\_bus.Event property), 28  
 topic (aries\_cloudagent.transport.outbound.status.OutboundSendStatus property), 112  
 transaction() (aries\_cloudagent.admin.request\_context.AdminRequestContext method), 4  
 transaction() (aries\_cloudagent.core.profile.Profile method), 31  
 TRANSACTION\_AUTHOR (aries\_cloudagent.protocols.endorse\_transaction.v1\_0.transaction\_v1\_0.transaction attribute), 81  
 TRANSACTION\_ENDORSER (aries\_cloudagent.protocols.endorse\_transaction.v1\_0.transaction\_v1\_0.transaction attribute), 81  
 TransactionJob (class in aries\_cloudagent.protocols.endorse\_transaction.v1\_0.transaction\_v1\_0.transaction), 81  
 TransportError, 114  
 TreeHasher (class in aries\_cloudagent.ledger.merkel\_validation.hasher), 114



47  
TRUSTEE (aries\_cloudagent.ledger.base.Role attribute),  
52  
txn\_endorse() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
txn\_endorse() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 57  
txn\_endorse() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 61  
txn\_submit() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
txn\_submit() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 57  
txn\_submit() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 61  
type (aries\_cloudagent.connections.models.diddoc.PublicKey property), 19  
type (aries\_cloudagent.connections.models.diddoc.publickey.PublicKeyType property), 23  
type (aries\_cloudagent.connections.models.diddoc.Service property), 20  
type (aries\_cloudagent.connections.models.diddoc.service.Service property), 24  
U  
UNDELIVERABLE (aries\_cloudagent.transport.outbound.status\_attribute), 112  
unpack\_message() (aries\_cloudagent.wallet.base.BaseWallet method), 131  
unpack\_to\_nibbles() (in module aries\_cloudagent.ledger.merkel\_validation.utils), 48  
unpad() (in module aries\_cloudagent.wallet.util), 138  
unqualify() (aries\_cloudagent.protocols.didcomm\_prefix.DIDCommPrefix static method), 96  
unsubscribe() (aries\_cloudagent.core.event\_bus.EventBus method), 28  
unused() (aries\_cloudagent.protocols.coordinate\_mediation.coordinate\_mediation static method), 74  
update() (aries\_cloudagent.config.settings.Settings method), 17  
update() (aries\_cloudagent.vc.ld\_proofs.purposes.authentication\_proof\_authentication\_proof method), 124  
update() (aries\_cloudagent.vc.ld\_proofs.purposes.proof\_purpose.proof\_purpose method), 125  
update\_endpoint\_for\_did() (aries\_cloudagent.ledger.base.BaseLedger method), 51  
update\_endpoint\_for\_did() (aries\_cloudagent.ledger.indy.IndySdkLedger method), 57  
update\_endpoint\_for\_did() (aries\_cloudagent.ledger.indy\_vdr.IndyVdrLedger method), 61  
update\_key\_pair\_metadata() (aries\_cloudagent.wallet.key\_pair.KeyPairStorageManager method), 136  
update\_record() (aries\_cloudagent.storage.base.BaseStorage method), 104  
update\_record() (aries\_cloudagent.storage.indy.IndySdkStorage method), 106  
update\_settings() (aries\_cloudagent.admin.request\_context.AdminRequestContext method), 5  
update\_settings() (aries\_cloudagent.config.base\_context.ContextBuilder method), 11  
update\_settings() (aries\_cloudagent.config.injection\_context.InjectionContext method), 12  
upload\_tails\_file() (aries\_cloudagent.tails.base.BaseTailsServer method), 107  
used (aries\_cloudagent.protocols.coordinate\_mediation.mediation\_invite\_mediation\_invite property), 74  
USER (aries\_cloudagent.ledger.base.Role attribute), 52  
V  
v1 (aries\_cloudagent.messaging.base\_message.DIDCommVersion attribute), 65  
v2 (aries\_cloudagent.messaging.base\_message.DIDCommVersion attribute), 65  
validate() (aries\_cloudagent.vc.ld\_proofs.purposes.authentication\_proof\_authentication\_proof method), 124  
validate() (aries\_cloudagent.vc.ld\_proofs.purposes.controller\_proof\_authentication\_proof\_authentication\_proof method), 124  
validate() (aries\_cloudagent.vc.ld\_proofs.purposes.credential\_issuance\_credential\_issuance method), 125  
validate() (aries\_cloudagent.vc.ld\_proofs.purposes.proof\_purpose.proof\_purpose method), 125  
validate\_before() (in module aries\_cloudagent.storage.base), 104  
validate\_version() (aries\_cloudagent.core.plugin\_registry.PluginRegistry method), 30  
value (aries\_cloudagent.connections.models.diddoc.PublicKey property), 19  
value (aries\_cloudagent.connections.models.diddoc.publickey.PublicKeyType property), 23  
value (aries\_cloudagent.connections.models.diddoc.LinkedDataKeySpoke property), 19  
value (aries\_cloudagent.connections.models.diddoc.publickey.LinkedDataKeySpoke property), 22  
ver\_type (aries\_cloudagent.connections.models.diddoc.publickey.PublicKeyType property), 24  
ver\_type (aries\_cloudagent.connections.models.diddoc.PublicKeyType property), 20  
verify\_message() (aries\_cloudagent.wallet.base.BaseWallet method), 131  
verify\_signed\_messages\_bls12381g2() (in module aries\_cloudagent.wallet.bbs), 132

`verify_spv_proof()` (*aries\_cloudagent.ledger.merkel\_validation.trie.SubTrie*  
*static method*), 48

`verkey` (*aries\_cloudagent.wallet.did\_info.DIDInfo* prop-  
*erty*), 132

`verkey` (*aries\_cloudagent.wallet.did\_info.KeyInfo* prop-  
*erty*), 133

**W**

`w3c` (*aries\_cloudagent.ledger.endpoint\_type.EndpointType*  
*property*), 53

`w3c` (*aries\_cloudagent.ledger.endpoint\_type.EndpointTypeName*  
*property*), 53

`wait_for()` (*aries\_cloudagent.utils.task\_queue.TaskQueue*  
*method*), 122

`wait_for_event()` (*aries\_cloudagent.core.event\_bus.EventBus*  
*method*), 28

`WAITING_FOR_PICKUP` (*aries\_cloudagent.transport.outbound.status.OutboundSendStatus*  
*attribute*), 112

`wallet` (*aries\_cloudagent.storage.indy.IndySdkStorage*  
*property*), 106

`wallet_access` (*aries\_cloudagent.indy.sdk.wallet\_setup.IndyWalletConfig*  
*property*), 40

`wallet_config` (*aries\_cloudagent.indy.sdk.wallet\_setup.IndyWalletConfig*  
*property*), 40

`WALLET_ONLY` (*aries\_cloudagent.wallet.did\_posture.DIDPosture*  
*attribute*), 134

`WalletDuplicateError`, 135

`WalletError`, 135

`WalletKeyMissingError`, 67

`WalletNotFoundError`, 135

`WalletSettingsError`, 135

`wire_format` (*aries\_cloudagent.transport.outbound.base.BaseOutboundTransport*  
*property*), 111

`WireFormatEncodeError`, 114

`WireFormatError`, 115

`WireFormatParseError`, 115

`with_metadata()` (*aries\_cloudagent.core.event\_bus.Event*  
*method*), 28

`wql` (*aries\_cloudagent.indy.models.predicate.Predicate*  
*property*), 35

`wql` (*aries\_cloudagent.indy.models.predicate.Relation*  
*property*), 35

`wrap()` (*aries\_cloudagent.utils.stats.Collector* *method*),  
120

`wrap_coro()` (*aries\_cloudagent.utils.stats.Collector*  
*method*), 120

`wrap_error()` (*aries\_cloudagent.indy.sdk.error.IndyErrorHandler*  
*class method*), 36

`wrap_fn()` (*aries\_cloudagent.utils.stats.Collector*  
*method*), 120

`WsTransport` (*class* *in*  
*aries\_cloudagent.transport.outbound.ws*),  
112