TRS-cli

Release 0.7.0

Nov 11, 2021

Modules

CHAPTER 1

trs_cli

1.1 trs_cli package

1.1.1 Submodules

1.1.2 trs_cli.client module

Class implementing TRS client.

class trs_cli.client.**TRSClient**(*uri: str*, *port: int = None*, *base_path: str = 'ga4gh/trs/v2'*, *use_http: bool = False*, *token: Optional[str] = None*)

Bases: object

Client to communicate with a GA4GH TRS instance. Supports additional endpoints defined in TRS-Filer (https://github.com/elixir-cloud-aai/trs-filer).

Parameters

- **uri** – Either the base URI of the TRS instance to connect to in either 'https' or 'http' schema (note that fully compliant TRS instances will use 'https' exclusively), e.g., *https://my-trs.app*, OR a hostname-based TRS URI, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **port** – Override default port at which the TRS instance can be accessed. Only required for TRS instances that are not fully spec-compliant, as the default port is defined in the TRS documentation, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **base-path** – Override default path at which the TRS API is accessible at the given TRS instance. Only required for TRS instances that are not fully spec-compliant, as the default port is defined in the TRS documentation, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **use_http** – Set the URI schema of the TRS instance to *http* instead of *https'when a TRS URI was provided to 'uri*.

- **token** – Bearer token to send along with TRS API requests. Set if required by TRS implementation. Alternatively, specify in API endpoint access methods.

uri
URI to TRS endpoints, built from *uri*, *port* and *base_path*, e.g.,"https://my-trs.app:443/ga4gh/trs/v2".

token
Bearer token for gaining access to TRS endpoints.

headers
Dictionary of request headers.

classmethod config(*debug: bool = False*, *no_validate: bool = False*) → None
Class configuration.

Parameters

- **debug** – Set to print error tracebacks.
- **no_validate** – Set to skip validation of error responses.

delete_tool(*id: str*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → str
Delete a tool.

id: A unique identifier of the tool to be deleted, scoped to this registry OR a TRS URI. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris

accept: Requested content type. token: Bearer token for authentication. Set if required by TRS

implementation and if not provided when instatiating client or if expired.

Returns ID of deleted TRS tool in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

delete_tool_class(*id: str*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → str
Delete a tool class.

Parameters

- **id** – Identifier of tool class to be deleted.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of deleted TRS toolClass in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

delete_version(*id: str*, *version_id: Optional[str] = None*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → str

Delete a tool version.

Parameters

- **id** – A unique identifier of the tool whose version is to be deleted, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version to be deleted, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of deleted TRS tool version in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_containerfiles(*id: str*, *version_id: Optional[str] = None*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[List[trs_cli.models.FileWrapper], trs_cli.models.Error]

Retrieve the file wrappers for all containerfiles associated with a specified tool version.

Parameters

- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either a list of *FileWrapper* instances in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_descriptor(*type: str*, *id: str*, *version_id: Optional[str] = None*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[trs_cli.models.Error, trs_cli.models.FileWrapper, str]

Retrieve the file wrapper for the primary descriptor of a specified tool version and descriptor type.

Parameters

- **type** – The output type of the descriptor. Plain types return the bare descriptor while the "non-plain" types return a descriptor wrapped with metadata. Allowed values include "CWL", "WDL", "NFL", "GALAXY", "SMK", "PLAIN_CWL", "PLAIN_WDL", "PLAIN_NFL", "PLAIN_GALAXY", "PLAIN_SMK".
- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either an instance of *FileWrapper* in case of a *200* or *201* response, an instance of *Error* for all other JSON reponses, or a string with file contents for *text/plain* responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

get_descriptor_by_path(*type: str*, *path: str*, *id: str*, *version_id: Optional[str] = None*, *encode_path: bool = False*, *accept: Optional[str] = None*, *token: Optional[str] = None*) → Union[trs_cli.models.Error, trs_cli.models.FileWrapper, str]

Retrieve the file wrapper for an indicated file for the specified tool version and descriptor type.

Parameters

- **type** – The output type of the descriptor. Plain types return the bare descriptor while the "non-plain" types return a descriptor wrapped with metadata. Allowed values include "CWL", "WDL", "NFL", "GALAXY", "SMK", "PLAIN_CWL", "PLAIN_WDL", "PLAIN_NFL", "PLAIN_GALAXY", "PLAIN_SMK". Setting one of the "**PLAIN_**" types will set the default accepted content type to "text/plain" (usually "application/json").
- **path** – Path, including filename, of descriptor or associated file relative to the primary descriptor file.
- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the

version_id retrieved from the TRS URI is overridden.

- **encode_path** – Percent/URL-encode *path* (may be required by some TRS implementations).
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either an instance of *FileWrapper* in case of a *200* or *201* response, an instance of *Error* for all other JSON reponses, and a string with file contents for 'text/plain' responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_files(*type: str*, *id: str*, *version_id: Optional[str] = None*, *format: Optional[str] = None*, *outfile: Optional[pathlib.Path] = None*, *token: Optional[str] = None*) → Union[List[trs_cli.models.ToolFile], trs_cli.models.Error, pathlib.Path, requests.models.Response]

Retrieve file information or ZIP archive of all files.

Parameters

- **type** – The output type of the descriptor. Plain types return the bare descriptor while the "non-plain" types return a descriptor wrapped with metadata. Allowed values include "CWL", "WDL", "NFL", "GALAXY", "SMK", "PLAIN_CWL", "PLAIN_WDL", "PLAIN_NFL", "PLAIN_GALAXY", "PLAIN_SMK".
- **id** – A unique identifier of the tool, scoped to this registry OR a hostname-based TRS URI. If TRS URIs include the version information, passing a *version_id* is optional.
- **version_id** – An optional identifier of the tool version, scoped to this registry. It is optional if version info is included in the TRS URI. If passed, then the existing *version_id* retreived from the TRS URI is overridden.
- **format** – Returns a zip file of all files when format=zip is specified.
- **outfile** – Name of zip archive when *format* is set to 'zip'. Ignored otherwise. If not specified, the filename is set based on the URL by taking the part after the last '/' and stored in the current working directory.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either a list of instances of *ToolFile* in case of a *200* response, an instance of *Error* for all other JSON reponses, and the absolute path of the ZIP archive if *format* is set to 'zip'.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.
- `IOError` – The ZIP archive could not be written.

get_service_info(*accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[trs_cli.models.Service, trs_cli.models.Error]

Retrieve service info.

Parameters

- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either an instance of *Service* in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_tests(*type: str*, *id: str*, *version_id: Optional[str] = None*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[trs_cli.models.Error, List[trs_cli.models.FileWrapper], str]

Retrieve the file wrappers for all tests associated with a specified tool version and descriptor type.

Parameters

- **type** – The output type of the descriptor. Plain types return the bare descriptor while the "non-plain" types return a descriptor wrapped with metadata. Allowed values include "CWL", "WDL", "NFL", "GALAXY", "SMK", "PLAIN_CWL", "PLAIN_WDL", "PLAIN_NFL", "PLAIN_GALAXY", "PLAIN_SMK".
- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either a list of *FileWrapper* instances in case of a *200* or *201* response, an instance of *Error* for all other JSON reponses, and a string for 'text/plain' responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_tool(*id: str*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[trs_cli.models.Error, str, trs_cli.models.Tool]

Retrieve tool with the specified identifier.

Parameters

- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either an instance of *Tool* in case of a *200* or *201* response, an instance of *Error* for all other JSON reponses, and a string for 'text/plain' responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_tool_classes(*accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[List[trs_cli.models.ToolClass], trs_cli.models.Error]

Retrieve tool classes.

Parameters

- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either a list of instances of *ToolClass* in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_tools(*accept: str = 'application/json'*, *token: Optional[str] = None*, *id: Optional[str] = None*, *alias: Optional[str] = None*, *toolClass: Optional[str] = None*, *descriptorType: Optional[str] = None*, *registry: Optional[str] = None*, *organization: Optional[str] = None*, *name: Optional[str] = None*, *toolname: Optional[str] = None*, *description: Optional[str] = None*, *author: Optional[str] = None*, *checker: Optional[bool] = None*, *limit: Optional[int] = None*, *offset: Optional[int] = None*) → Union[List[trs_cli.models.Tool], trs_cli.models.Error]

List all tools.

Filter parameters to subset the tools list can be specified. Filter parameters are additive.

Parameters

- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.
- **id** – Return only entries with the given identifier.
- **alias** – Return only entries with the given alias.

- **toolClass** – Return only entries with the given subclass name.
- **descriptorType** – Return only entries with the given descriptor type.
- **registry** – Return only entries from the given registry.
- **organization** – Return only entries from the given organization.
- **name** – Return only entries with the given image name.
- **toolname** – Return only entries with the given tool name.
- **description** – Return only entries with the given description.
- **author** – Return only entries from the given author.
- **checker** – Return only checker workflows.
- **limit** – Number of records when paginating results.
- **offset** – Start index when paginating results.

Returns Unmarshalled TRS response as either a list of instances of *Tool* in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_version(*id: str*, *version_id: Optional[str] = None*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[trs_cli.models.ToolVersion, trs_cli.models.Error]

Retrieve tool version with the specified identifiers.

Parameters

- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. If a TRS URI is passed and includes the version identifier, passing a *version_id* is optional. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either an instance of *ToolVersion* in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *trs_cli.errors.InvalidResponseError* – The response could not be validated against the API schema.

get_versions(*id: str*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → Union[List[trs_cli.models.ToolVersion], trs_cli.models.Error]

Returns all versions of the specified tool..

Parameters

- **id** – A unique identifier of the tool, scoped to this registry OR a TRS URI. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns Unmarshalled TRS response as either a list of instances of *ToolVersion* in case of a *200* response, or an instance of *Error* for all other JSON reponses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

no_validate = False

post_service_info(*payload: Dict[KT, VT]*, *token: Optional[str] = None*) → None

Register service info.

Parameters

- **payload** – Service info data.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

post_tool(*payload: Dict[KT, VT]*, *accept: str = 'application/json'*, *token: Optional[str] = None*) → str

Register a tool.

Parameters

- **payload** – Tool data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS tool in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.

- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

post_tool_class(*payload: Dict[KT, VT], accept: str = 'application/json', token: Optional[str] = None*) → str

Register a tool class.

Parameters

- **payload** – Tool class data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS toolClass in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

post_version(*id: str, payload: Dict[KT, VT], accept: str = 'application/json', token: Optional[str] = None*) → str

Register a tool version.

Parameters

- **id** – A unique identifier of the tool to be registered, scoped to this registry OR a TRS URI. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **payload** – Tool version data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS tool version in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

put_tool(*id: str, payload: Dict[KT, VT], accept: str = 'application/json', token: Optional[str] = None*) → str

Create a tool object with a predefined ID. Overwrites any existing tool object with the same ID.

Parameters

- **id** – Identifier of tool to be created or overwritten.
- **payload** – Tool data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS tool in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

put_tool_class(*id: str, payload: Dict[KT, VT], accept: str = 'application/json', token: Optional[str] = None*) → str

Create a tool class with a predefined unique ID. Overwrites any existing tool object with the same ID.

Parameters

- **id** – Identifier of tool class to be created/overwritten.
- **payload** – Tool class data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS toolClass in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

put_version(*id: str, version_id: str, payload: Dict[KT, VT], accept: str = 'application/json', token: Optional[str] = None*) → str

Create a tool version object with a predefined ID. Overwrites any existing tool version object with the same ID.

Parameters

- **id** – A unique identifier of the tool to be registered, scoped to this registry OR a TRS URI. For more information on TRS URIs, cf. https://ga4gh.github.io/tool-registry-service-schemas/DataModel/#trs_uris
- **version_id** – Identifier of the tool version to be registered, scoped to this registry. It is optional if a TRS URI is passed and includes version information. If provided nevertheless, then the *version_id* retrieved from the TRS URI is overridden.
- **payload** – Tool version data.
- **accept** – Requested content type.
- **token** – Bearer token for authentication. Set if required by TRS implementation and if not provided when instatiating client or if expired.

Returns ID of registered TRS tool version in case of a *200* response, or an instance of *Error* for all other responses.

Raises

- `requests.exceptions.ConnectionError` – A connection to the provided TRS instance could not be established.
- `pydantic.ValidationError` – The object data payload could not be validated against the API schema.
- *`trs_cli.errors.InvalidResponseError`* – The response could not be validated against the API schema.

retrieve_files(*out_dir: Union[str, pathlib.Path], type: str, id: str, version_id: Optional[str] = None, encode_path: bool = False, token: Optional[str] = None*) → Dict[str, List[str]]

Write tool version file contents for a given descriptor type to files.

DEPRECATED: Use *.get_files* with *format=zip* instead.

Parameters

- **out_dir** – Directory to write requested files to. Will be attempted to create if it does not exist.
- **type** – The output type of the descriptor. Plain types return the bare descriptor while the "non-plain" types return a descriptor wrapped with metadata. Allowed values include "CWL", "WDL", "NFL", "GALAXY", "SMK", "PLAIN_CWL", "PLAIN_WDL", "PLAIN_NFL", "PLAIN_GALAXY", "PLAIN_SMK".
- **id** – A unique identifier of the tool, scoped to this registry OR a hostname-based TRS URI. If TRS URIs include the version information, passing a *version_id* is optional.
- **version_id** – An optional identifier of the tool version, scoped to this registry. It is optional if version info is included in the TRS URI. If passed, then the existing *version_id* retreived from the TRS URI is overridden.
- **encode_path** – Percent/URL-encode relative paths of files (may be required by some TRS implementations).

Returns Dictionary of *FileType* enumerator values (e.g., *TEST_FILE*, *PRIMARY_DESCRIPTOR*) as keys and a list of paths, relative to *out_dir* as values.

1.1.3 trs_cli.errors module

exception trs_cli.errors.**ContentTypeUnavailable**
Bases: Exception

Exception raised when an unavailable content type is requested.

exception trs_cli.errors.**FileInformationUnavailable**
Bases: Exception

Exception raised when information for a file associated with a descriptor is unavailable.

exception trs_cli.errors.**InvalidResourceIdentifier**
Bases: Exception

Exception raised for invalid tool/version identifiers.

exception trs_cli.errors.**InvalidResponseError**
Bases: Exception

Exception raised when an invalid API response is encountered.

exception trs_cli.errors.**InvalidURI**
Bases: Exception

Exception raised for invalid URIs.

trs_cli.errors.**exception_handler**(*_type: type*, *value: BaseException*, *traceback: traceback*, *print_traceback: bool = False*) → None
Error handler for all exceptions.

1.1.4 trs_cli.models module

class trs_cli.models.**Checksum**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ChecksumRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**CustomBaseModel**
Bases: pydantic.main.BaseModel

Settings subclass.

class Config
Bases: object

Configuration for *pydantic* model class.

arbitrary_types_allowed = False

extra = 'forbid'

class trs_cli.models.**DescriptorType**
Bases: str, enum.Enum

An enumeration.

CWL = 'CWL'

GALAXY = 'GALAXY'

NFL = 'NFL'

SMK = 'SMK'

WDL = 'WDL'

class trs_cli.models.**Error**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**FileType**
Bases: enum.Enum

An enumeration.

CONTAINERFILE = 'CONTAINERFILE'

OTHER = 'OTHER'

PRIMARY_DESCRIPTOR = 'PRIMARY_DESCRIPTOR'

SECONDARY_DESCRIPTOR = 'SECONDARY_DESCRIPTOR'

TEST_FILE = 'TEST_FILE'

class trs_cli.models.**FileType1**
Bases: enum.Enum

An enumeration.

CONTAINERFILE = 'CONTAINERFILE'

OTHER = 'OTHER'

PRIMARY_DESCRIPTOR = 'PRIMARY_DESCRIPTOR'

SECONDARY_DESCRIPTOR = 'SECONDARY_DESCRIPTOR'

TEST_FILE = 'TEST_FILE'

class trs_cli.models.**FileWrapper**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**FileWrapperRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**FilesRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ImageData**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ImageDataRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ImageType**
Bases: str, enum.Enum

An enumeration.

Conda = 'Conda'

Docker = 'Docker'

Singularity = 'Singularity'

class trs_cli.models.**Organization**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**Organization1**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**OtherType**
Bases: enum.Enum

An enumeration.

JSON = 'JSON'

OTHER = 'OTHER'

class trs_cli.models.**Service**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ServiceRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ServiceType**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ServiceTypeRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**Tool**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolClass**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolClassRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolClassRegisterId**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolFile**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolFileRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolVersion**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolVersionRegister**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**ToolVersionRegisterId**
Bases: *trs_cli.models.CustomBaseModel*

class trs_cli.models.**TypeRegister**
Bases: *trs_cli.models.CustomBaseModel*

CHAPTER 2

Indices and tables

- genindex
- modindex

Python Module Index

t

Index