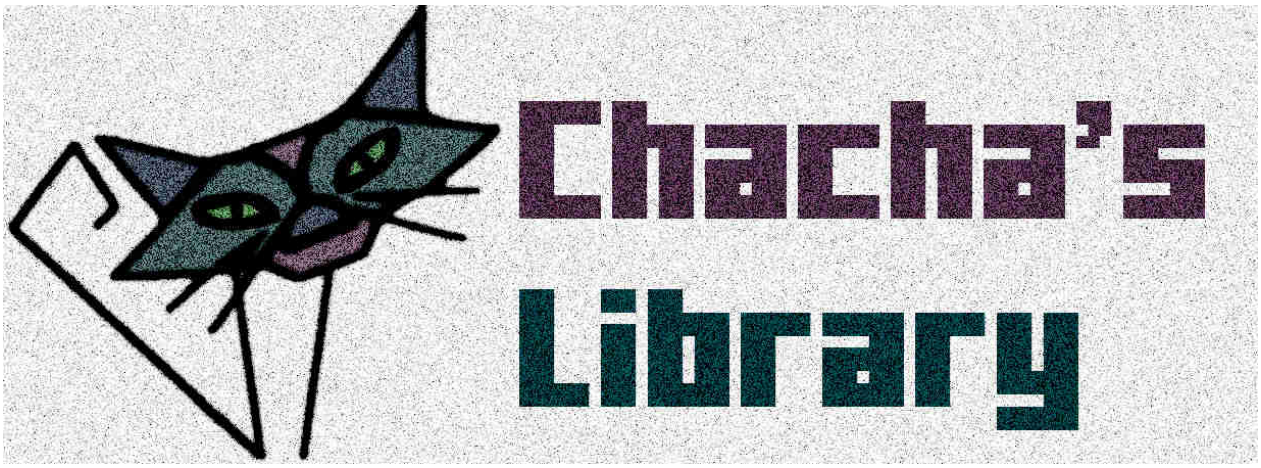
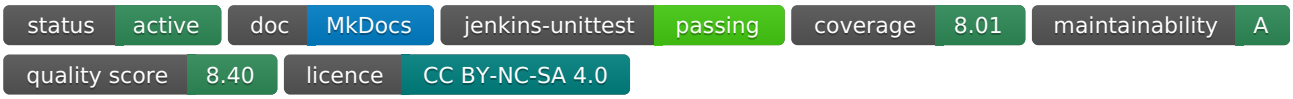


pyrestresource

User Manual

chacha

Table of contents



1. pyrestresource

A RESTful API library built on top of pydantic & uvicorn to make service API from a data model.

/!\ early in-progress project for internal use ATM.

Feel free to contribute.

Features (available): - type annotation used - support containers (dict) - support plugins (for hook and biding) - user auth - ACL - daemon mode

Features(planned): - group support - python internal model instance (with possible serialization/auto-save on-disk)

Limitations: - no nested reads / writes

Checkout Latest Documentation.

2. Usage

2.1 Pulvinar dolor

Donec dapibus est fermentum justo volutpat condimentum. Integer quis nunc neque. Donec dictum vehicula justo, in facilisis ex tincidunt in. Vivamus sollicitudin sem dui, id mollis orci facilisis ut. Proin sed pulvinar dolor. Donec volutpat commodo urna imperdiet pulvinar. Fusce eget aliquam risus. Vivamus viverra luctus ex, in finibus mi. Nullam elementum dapibus mollis. Ut suscipit volutpat ex, quis feugiat lacus consectetur eu.

2.2 Condimentum faucibus

Quisque auctor egestas sem, luctus suscipit ex maximus vitae. Duis facilisis augue et condimentum faucibus. Donec cursus, enim a sagittis egestas, lectus lorem eleifend libero, at tincidunt leo magna at libero. Nunc eros velit, suscipit luctus tempor vel, finibus et est. Curabitur efficitur pretium pulvinar. Donec urna lectus, vulputate quis turpis sed, placerat congue urna. Phasellus aliquet fermentum quam, non auctor elit porta nec. Morbi eu ligula at nisl ultricies condimentum vitae id ante.

2.3 Aliquam lacinia

In volutpat lorem ex, et fringilla nibh faucibus quis. Mauris et arcu elementum, auctor dui vitae, egestas arcu. Duis sit amet aliquam quam. Phasellus a odio turpis. Etiam tristique mi eu enim varius, eget facilisis est vestibulum. Aliquam lacinia nec purus sed luctus. Cras at laoreet erat.

3. Reference

3.1 Pyrestresource

3.1.1 metadata

Metadata module module

Attributes

__Name__ `module-attribute`

```
__Name__ = dist.metadata['Name']
```

__Summuary__ `module-attribute`

```
__Summuary__ = dist.metadata['Summary']
```

__version__ `module-attribute`

```
__version__ = version('pyrestresource')
```

dist `module-attribute`

```
dist = distribution('pyrestresource')
```

3.1.2 Helpers

Attributes

Functions

forward_exception

```
forward_exception(e: Exception, forward: bool) -> None
```

parse_dict_cookies

```
parse_dict_cookies(cookies: str) -> dict[str, str | None]
```

3.1.3 rest ACL

Classes

ACL_record

Bases: `BaseModel`

Attributes

rule `instance-attribute`

```
rule: ACL_rule
```

target `instance-attribute`

```
target: ACL_target
```

verbs `instance-attribute`

```
verbs: list[rsrc_verb]
```

ACL_rule

Bases: `Enum`

Attributes

ALLOW `class-attribute` `instance-attribute`

```
ALLOW = auto()
```

DENY `class-attribute` `instance-attribute`

```
DENY = auto()
```

ACL_target

Bases: `BaseModel`

ACL_target_group

Bases: `ACL_target`

Attributes

name `instance-attribute`

```
name: str
```

ACL_target_group_Any

Bases: `ACL_target_group`

Attributes

name `class-attribute` `instance-attribute`

```
name: str = '__ANY__'
```

ACL_target_user

Bases: `ACL_target`

Attributes

name `instance-attribute`

```
name: str
```

Functions

from_user_login `classmethod`

```
from_user_login(user_login: UserLogin) -> ACL_target_user
```

ACL_target_user_Annonymous

Bases: `ACL_target_user`

Attributes

name `class-attribute` `instance-attribute`

```
name: str = '__ANNONYMOUS__'
```

3.1.4 Rest exceptions

Classes

RestResourceConfigException

Bases: `RestResourceException`

RestResourceException

Bases: `Exception`

RestResourceHandlerException

Bases: `RestResourceException`

RestResourceHandlerException_BadRequest

Bases: `RestResourceHandlerException`

RestResourceHandlerException_Forbiden

Bases: `RestResourceHandlerException`

RestResourceHandlerException_MethodNotAllowed

Bases: `RestResourceHandlerException`

RestResourceHandlerException_ResourceNotFound

Bases: `RestResourceHandlerException`

RestResourceLoginException

Bases: `RestResourceException`

RestResourceLoginException_ClientChange

Bases: `RestResourceLoginException`

RestResourceLoginException_InvalidCredentials

Bases: `RestResourceLoginException`

RestResourceLoginException_InvalidSession

Bases: `RestResourceLoginException`

RestResourceLoginException_SessionTimeout

Bases: `RestResourceLoginException`

RestResourceModelException

Bases: `RestResourceException`

RestResourceModelException_ACL

Bases: `RestResourceModelException`

RestResourcePluginException

Bases: `RestResourceException`

RestResourcePluginException_InvalidPluginSignature

Bases: `RestResourcePluginException`

3.1.5 Rest login

CLI interface module

Classes

Login

Bases: `RestResourceBase`

Attributes

secret `class-attribute` `instance-attribute`

```
secret: Optional[str] = RestField(None, exclude=True, ACL=[ACL_record(verbs=[rsrc_verb.PUT], target=ACL_target_group_Any(), rule=ACL_rule.ALLOW),
ACL_record(verbs=[rsrc_verb.GET], target=ACL_target_group_Any(), rule=ACL_rule.DENY)])
```

username `class-attribute` `instance-attribute`

```
username: Optional[str] = RestField(None)
```

ResourcePlugin_Login

```
ResourcePlugin_Login(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin_RestResourceBase_default`

Attributes

ar_UserLogin `class-attribute` `instance-attribute`

```
ar_UserLogin: list[UserLogin] = []
```

Functions

handle_resource_get

```
handle_resource_get(resource: Login, params: RestRequestParams_RestResourceBase_GET) -> Login
```

handle_resource_put

```
handle_resource_put(resource: Login, params: RestRequestParams_RestResourceBase_PUT) -> Login
```

RestResourceBaseLogin

Bases: `RestResourceBase`

Attributes

login `class-attribute` `instance-attribute`

```
login: Login = RestField(default=Login(), plugin=ResourcePlugin_Login)
```

Functions

get_new_cookie_expiration_date

```
get_new_cookie_expiration_date() -> datetime
```

user_login

```
user_login(user_name: str, user_secret: str, request: RestRequest) -> str
```

UserLogin

Bases: `BaseModel`

Attributes

secret `instance-attribute`

```
secret: str
```

username `instance-attribute`

```
username: str
```

UserSession

Bases: `BaseModel`

Attributes

client `instance-attribute`

```
client: tuple[str, int] | tuple[] | None
```

last_update `instance-attribute`

```
last_update: datetime
```

user_login `instance-attribute`

```
user_login: UserLogin
```

Functions

3.1.6 Rest model

Classes

Functions

RestField

```
RestField(default: Any = PydanticUndefined, *, default_factory: Callable[[], Any] | None = _Unset, alias: str | None = _Unset, alias_priority: int | None = _Unset, validation_alias: str | 'AliasPath' | 'AliasChoices' | None = _Unset, serialization_alias: str | None = _Unset, title: str | None = _Unset, description: str | None = _Unset, examples: 'list[Any] | None' = _Unset, exclude: bool | None = _Unset, discriminator: str | None = _Unset, json_schema_extra: dict[str, Any] | Callable[[dict[str, Any]], None] | None = _Unset, frozen: bool | None = _Unset, validate_default: bool | None = _Unset, repr: bool = _Unset, init_var: bool | None = _Unset, kw_only: bool | None = _Unset, pattern: str | None = _Unset, strict: bool | None = _Unset, gt: float | None = _Unset, ge: float | None = _Unset, lt: float | None = _Unset, le: float | None = _Unset, multiple_of: float | None = _Unset, allow_inf_nan: bool | None = _Unset, max_digits: int | None = _Unset, decimal_places: int | None = _Unset, min_length: int | None = _Unset, max_length: int | None = _Unset, union_mode: Literal['smart', 'left_to_right'] = _Unset, ACL: list['ACL_record'] | None = _Unset, plugin: Type['ResourcePlugin'] | None = _Unset, **extra: Unpack[_EmptyKwargs]) -> Any
```

3.1.7 Rest request

A module to handle http requests context

Attributes

Classes

RequestFactory

Bases: `Generic[_T_RestRequestParams_POST, _T_RestRequestParams_DELETE, _T_RestRequestParams_GET, _T_RestRequestParams_PUT]`, `BaseModel`

RestRequets class factory

Attributes

cls_RestRequestParams_DELETE `class-attribute` `instance-attribute`

```
cls_RestRequestParams_DELETE: type[RestRequestParams_DELETE] = Field(default=RestRequestParams_DELETE)
```

cls_RestRequestParams_GET `class-attribute` `instance-attribute`

```
cls_RestRequestParams_GET: type[RestRequestParams_GET] = Field(default=RestRequestParams_GET)
```

cls_RestRequestParams_POST `class-attribute` `instance-attribute`

```
cls_RestRequestParams_POST: type[RestRequestParams_POST] = Field(default=RestRequestParams_POST)
```

cls_RestRequestParams_PUT `class-attribute` `instance-attribute`

```
cls_RestRequestParams_PUT: type[RestRequestParams_PUT] = Field(default=RestRequestParams_PUT)
```

Functions

get_RestRequest

```
get_RestRequest(root_resource: RestResourceBase, url: str, verb: rsrc_verb, data: dict, query_string: Optional[str] = None) -> RestRequest
```

get a RestRequets instance based on LUT_verb configuration

PARAMETER	DESCRIPTION
`url`	http url of the request **TYPE:** `str`
`verb`	http verb received **TYPE:** `rsrc_verb`
`data`	data associated with the request **TYPE:** `dict`

update_RestRequest

```
update_RestRequest(request: RestRequest) -> None
```

create an updated copy of a RestRequest object based on a different LUT_verb configuration Args: origin_request: the original request

RestRequest

```
RestRequest(type_request_params: type[_T_RestRequestParams], root_resource: RestResourceBase, url: str, verb: rsrc_verb, data: Optional[dict[str,
T_SupportedRESTFields]] = None, query_string: Optional[str] = None)
```

Bases: `Generic[_T_RestRequestParams]`

Main RestRequets class

class to handle a request context, that will be kept and updated while walking url parts

PARAMETER	DESCRIPTION
`type_request_params`	type of the request param **TYPE:** `type[_T_RestRequestParams]`
`url`	http url of the request **TYPE:** `str`
`verb`	http verb received **TYPE:** `rsrc_verb`
`data`	data associated with the request In this case, all other argument - but type_request_params - are ignored and inherited from the origin_request **TYPE:** `Optional[dict[str, T_SupportedRESTFields]]` **DEFAULT:** `None`
`query_string`	query arguments after url (eg: ?arg1=value1&arg2=value2 ...) **TYPE:** `Optional[str]` **DEFAULT:** `None`

Attributes

ReqParams `instance-attribute`

```
ReqParams: _T_RestRequestParams = type_request_params()
```

data `instance-attribute`

```
data: dict = data
```

groups `instance-attribute`

```
groups: list[ACL_target_group] = []
```

headers `instance-attribute`

```
headers: dict[str, None | str | dict[str, None | str]] = {'host': None, 'cookie': {}}
```

outgoing_cookie `instance-attribute`

```
outgoing_cookie: dict[str, str] = {}
```

result `instance-attribute`

```
result: Optional[str] = None
```

root_resource `instance-attribute`

```
root_resource: RestResourceBase = root_resource
```

url `instance-attribute`

```
url: str = url
```

url_stack `instance-attribute`

```
url_stack: list[str]
```

url_stack_index `instance-attribute`

```
url_stack_index: int = 0
```

user `instance-attribute`

```
user: ACL_target_user = ACL_target_user_Annonymous()
```

verb `instance-attribute`

```
verb: rsrc_verb = verb
```

Functions

add_group

```
add_group(group: ACL_target_group)
```

consume_url_stack

```
consume_url_stack(count: int) -> list[str]
```

consume some url stack elements

PARAMETER	DESCRIPTION
`count`	number of element to consume **TYPE:** `int`

get_client

```
get_client() -> tuple[str, int] | tuple[]
```

get_cookie

```
get_cookie(key: str) -> str | None
```

get_data

```
get_data() -> dict
```

get the request data

get_host

```
get_host() -> str | dict[str, str | None] | None
```

get_req_params

```
get_req_params() -> _T_RestRequestParams
```

get extracted req_params

get_resource_origin

```
get_resource_origin(deepness: int = 0) -> str
```

get current or previous (consumed) resource in the url

PARAMETER	DESCRIPTION
`deepness`	backward amount **TYPE:** `int` **DEFAULT:** `0`

get_result

```
get_result() -> Optional[str]
```

get_root_resource

```
get_root_resource() -> RestResourceBase
```

get_status

```
get_status() -> int
```

get_url

```
get_url() -> str
```

retrieve the raw url

get_url_stack

```
get_url_stack() -> list[str]
```

retrieve the current url stack

get_user

```
get_user()
```

get_verb

```
get_verb() -> rsrc_verb
```

get http request verb

reset_resp_cookie

```
reset_resp_cookie(key: str) -> None
```

reset_url_stack

```
reset_url_stack() -> None
```

set_client

```
set_client(client: tuple[str, int]) -> None
```

set_headers

```
set_headers(headers: list[Any]) -> None
```

set_resp_cookie_value

```
set_resp_cookie_value(key: str, value: str) -> None
```

set_resp_status

```
set_resp_status(status: int) -> None
```

set_result

```
set_result(result: str)
```

set_user

```
set_user(user: ACL_target_user)
```

update_ReqParams

```
update_ReqParams(type_request_params: type[_T_RestRequestParams])
```

Functions

3.1.8 Rest request opt

Classes

RestRequestParams

Bases: `BaseModel`

RestRequestParams_DELETE

Bases: `RestRequestParams`

RestRequestParams_Dict

Bases: `RestRequestParams`

RestRequestParams_Dict_DELETE

Bases: `RestRequestParams_Dict`, `RestRequestParams_DELETE`, `Generic[_T_DictKey]`

Attributes

API_key `class-attribute` `instance-attribute`

```
API_key: Optional[_T_DictKey] = None
```

RestRequestParams_Dict_GET

Bases: `RestRequestParams_Dict`, `RestRequestParams_GET`

RestRequestParams_Dict_POST

Bases: `RestRequestParams_Dict`, `RestRequestParams_POST`, `Generic[_T_DictKey]`

Attributes

API_key `class-attribute` `instance-attribute`

```
API_key: Optional[_T_DictKey] = None
```

RestRequestParams_Dict_elem_GET

Bases: `RestRequestParams_Dict`, `RestRequestParams_GET`

RestRequestParams_Dict_elem_PUT

Bases: `RestRequestParams_Dict`, `RestRequestParams_GET`

RestRequestParams_GET

Bases: `RestRequestParams`

RestRequestParams_POST

Bases: `RestRequestParams`

RestRequestParams_PUT

Bases: `RestRequestParams`

RestRequestParams_RestResourceBase

Bases: `RestRequestParams`

RestRequestParams_RestResourceBase_GET

Bases: `RestRequestParams_GET`, `RestRequestParams_RestResourceBase`

RestRequestParams_RestResourceBase_PUT

Bases: `RestRequestParams_PUT`, `RestRequestParams_RestResourceBase`

3.1.9 Rest resource

Attributes

Classes

RestResourceBase

Bases: `ABC`, `BaseModel`

Functions

__call__ `async`

```
__call__(scope, receive, send) -> None
```

check_acl_field

```
check_acl_field(request: RestRequest, req_index: int = 0) -> None
```

Check ACL on requested field access

check_acl_self

```
check_acl_self(request: RestRequest, new_data: Optional[dict[str, T_SupportedRESTFields]]) -> None
```

Check ACL on requested field operation (involving checking sub-fields)

process_request

```
process_request(url: str, verb: rsrc_verb = rsrc_verb.GET, data_json: Optional[str] = None, query_string: Optional[str] = None, client: Optional[tuple[str, int]] = None, headers: Optional[list[Any]] = None, http_mode: bool = False) -> RestRequest
```

read_body `async`

```
read_body(receive)
```

Read and return the entire body from an incoming ASGI message.

update

```
update(**new_data)
```

Functions

3.1.10 Rest resource handler

Attributes

Classes

ResourceHandler

```
ResourceHandler(resource: _T_Resource, url: Optional[str] = None, verb: Optional[rsrc_verb] = None, data: Optional[dict] = None, query_string: Optional[str]
= None, prev_handler: Optional[ResourceHandler] = None)
```

Bases: `ABC`, `Generic[_T_Resource, _T_RestRequestParams_POST, _T_RestRequestParams_DELETE, _T_RestRequestParams_GET, _T_RestRequestParams_PUT]`

Attributes

next_handler `instance-attribute`

```
next_handler: Optional[ResourceHandler] = None
```

prev_handler `instance-attribute`

```
prev_handler: Optional[ResourceHandler] = None
```

req `instance-attribute`

```
req: RestRequest
```

resource `instance-attribute`

```
resource: _T_Resource = resource
```

saved_url `instance-attribute`

```
saved_url: list[str] = []
```

Functions

access_resource

```
access_resource() -> _T_Resource
```

create_chained_handler `classmethod`

```
create_chained_handler(other: ResourceHandler, resource: _T_Resource) -> Self
```

get_request

```
get_request() -> RestRequest
```

process_verb

```
process_verb() -> Optional[_T_Resource | T_DictKey | list[T_DictKey]]
```

register_resource_handler `classmethod`

```
register_resource_handler(other_cls) -> None
```

ResourceHandler_RestResourceBase

```
ResourceHandler_RestResourceBase(resource: _T_Resource, url: Optional[str] = None, verb: Optional[rsrc_verb] = None, data: Optional[dict] = None,
query_string: Optional[str] = None, prev_handler: Optional[ResourceHandler] = None)
```

Bases: `ResourceHandler[RestResourceBase, _T_RestRequestParams_POST, _T_RestRequestParams_DELETE, RestRequestParams_RestResourceBase_GET, RestRequestParams_RestResourceBase_PUT]`

ResourceHandler_dict

```
ResourceHandler_dict(resource: _T_Resource, url: Optional[str] = None, verb: Optional[rsrc_verb] = None, data: Optional[dict] = None, query_string:
Optional[str] = None, prev_handler: Optional[ResourceHandler] = None)
```

Bases: `ResourceHandler[T_Dict, RestRequestParams_Dict_POST, RestRequestParams_Dict_DELETE, RestRequestParams_Dict_GET, _T_RestRequestParams_PUT]`

ResourceHandler_dict_elem

```
ResourceHandler_dict_elem(resource: _T_Resource, url: Optional[str] = None, verb: Optional[rsrc_verb] = None, data: Optional[dict] = None, query_string:
Optional[str] = None, prev_handler: Optional[ResourceHandler] = None)
```

Bases: `ResourceHandler[T_DictValues, _T_RestRequestParams_POST, _T_RestRequestParams_DELETE, RestRequestParams_RestResourceBase_GET, _T_RestRequestParams_PUT]`

ResourceHandler_simple

```
ResourceHandler_simple(resource: _T_Resource, url: Optional[str] = None, verb: Optional[rsrc_verb] = None, data: Optional[dict] = None, query_string:
Optional[str] = None, prev_handler: Optional[ResourceHandler] = None)
```

Bases: `ResourceHandler[T_SupportedRESTFields, _T_RestRequestParams_POST, _T_RestRequestParams_DELETE, _T_RestRequestParams_GET, _T_RestRequestParams_PUT]`

3.1.11 Rest resource plugin

Attributes

Classes

ResourcePlugin

```
ResourcePlugin(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ABC`

Attributes

__request `instance-attribute`

```
__request: RestRequest = request
```

__root_resource `instance-attribute`

```
__root_resource: RestResourceBase = root_resource
```

Functions

get_new_cookie_expiration_date

```
get_new_cookie_expiration_date() -> datetime
```

get_user_login

```
get_user_login() -> str
```

reset_resp_cookie

```
reset_resp_cookie(key: str) -> None
```

set_resp_cookie_value

```
set_resp_cookie_value(key: str, value: str) -> None
```

set_resp_status

```
set_resp_status(status: int) -> None
```

user_login

```
user_login(user_name: str, user_secret: str) -> str
```

ResourcePlugin_RestResourceBase

```
ResourcePlugin_RestResourceBase(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin`, `Generic[TV_RestResourceBase]`

Functions

handle_resource_get `abstractmethod`

```
handle_resource_get(resource: TV_RestResourceBase, params: RestRequestParams_RestResourceBase_GET) -> TV_RestResourceBase
```

handle_resource_put `abstractmethod`

```
handle_resource_put(resource: TV_RestResourceBase, params: RestRequestParams_RestResourceBase_PUT) -> TV_RestResourceBase
```

ResourcePlugin_RestResourceBase_default

```
ResourcePlugin_RestResourceBase_default(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin_RestResourceBase[TV_RestResourceBase]`

default implementation of RestResourcePlugin_RestResourceBase

Functions

handle_resource_get

```
handle_resource_get(resource: TV_RestResourceBase, params: RestRequestParams_RestResourceBase_GET) -> TV_RestResourceBase
```

handle_resource_put

```
handle_resource_put(resource: TV_RestResourceBase, params: RestRequestParams_RestResourceBase_PUT) -> TV_RestResourceBase
```

ResourcePlugin_dict

```
ResourcePlugin_dict(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin`, `Generic[_T_DictKey, _T_DictValues]`

Functions

handle_dict_delete `abstractmethod`

```
handle_dict_delete(resource_dict: dict[_T_DictKey, _T_DictValues], params: RestRequestParams_Dict_DELETE[_T_DictKey]) -> None
```

handle_dict_elem_get `abstractmethod`

```
handle_dict_elem_get(resource: TV_RestResourceBase, params: RestRequestParams_Dict_elem_GET) -> TV_RestResourceBase
```

handle_dict_elem_put `abstractmethod`

```
handle_dict_elem_put(resource: TV_RestResourceBase, params: RestRequestParams_Dict_elem_PUT) -> TV_RestResourceBase
```

handle_dict_get_keys `abstractmethod`

```
handle_dict_get_keys(resource_dict: dict[_T_DictKey, _T_DictValues], params: RestRequestParams_Dict_GET) -> list[_T_DictKey]
```

handle_dict_post `abstractmethod`

```
handle_dict_post(resource_dict: dict[_T_DictKey, _T_DictValues], resource: _T_DictValues, params: RestRequestParams_Dict_POST[_T_DictKey]) ->
Optional[_T_DictKey]
```

ResourcePlugin_dict_default

```
ResourcePlugin_dict_default(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin_dict[_T_DictKey, _T_DictValues]`

default implementation of RestResourcePlugin_dict

Functions

handle_dict_delete

```
handle_dict_delete(resource_dict: dict[_T_DictKey, _T_DictValues], params: RestRequestParams_Dict_DELETE[_T_DictKey]) -> None
```

handle_dict_elem_get

```
handle_dict_elem_get(resource: TV_RestResourceBase, params: RestRequestParams_Dict_elem_GET) -> TV_RestResourceBase
```

handle_dict_elem_put

```
handle_dict_elem_put(resource: TV_RestResourceBase, params: RestRequestParams_Dict_elem_PUT) -> TV_RestResourceBase
```

handle_dict_get_keys

```
handle_dict_get_keys(resource_dict: dict[_T_DictKey, _T_DictValues], params: RestRequestParams_Dict_GET) -> list[_T_DictKey]
```

handle_dict_post

```
handle_dict_post(resource_dict: dict[_T_DictKey, _T_DictValues], resource: _T_DictValues, params: RestRequestParams_Dict_POST[_T_DictKey]) ->
Optional[_T_DictKey]
```

ResourcePlugin_field

```
ResourcePlugin_field(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin`, `Generic[TV_SupportedRESTFields]`

Functions

handle_field_get `abstractmethod`

```
handle_field_get(resource: TV_SupportedRESTFields, params: RestRequestParams_GET) -> TV_SupportedRESTFields
```

handle_field_put `abstractmethod`

```
handle_field_put(resource: TV_SupportedRESTFields, params: RestRequestParams_PUT) -> TV_SupportedRESTFields
```

ResourcePlugin_field_default

```
ResourcePlugin_field_default(request: RestRequest, root_resource: RestResourceBase)
```

Bases: `ResourcePlugin_field[TV_SupportedRESTFields]`

default implementation of RestResourcePlugin_simple

Functions

handle_field_get

```
handle_field_get(resource: TV_SupportedRESTFields, params: RestRequestParams_GET) -> TV_SupportedRESTFields
```

handle_field_put

```
handle_field_put(resource: TV_SupportedRESTFields, params: RestRequestParams_PUT) -> TV_SupportedRESTFields
```

3.1.12 Rest resource rootpoint

Classes

RestResourceWalker_Root__tree_init

```
RestResourceWalker_Root__tree_init(resource: RestResourceBase | Type[RestResourceBase])
```

Bases: `RestResourceWalker_Root`

Attributes

cls_RestResourceWalker_Sub `class-attribute` `instance-attribute`

```
cls_RestResourceWalker_Sub = [RestResourceWalker_Sub_T_Dict__tree_init, RestResourceWalker_Sub_RestFields__tree_init,
RestResourceWalker_Sub_RestResourceBase__tree_init]
```

RestResourceWalker_Sub_RestFields__tree_init

```
RestResourceWalker_Sub_RestFields__tree_init(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] =
None, annotation: Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub_RestFields`

Functions

process

```
process() -> None
```

RestResourceWalker_Sub_RestResourceBase__tree_init

```
RestResourceWalker_Sub_RestResourceBase__tree_init(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub]
= None, annotation: Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub_RestResourceBase`

Functions

process

```
process() -> None
```

RestResourceWalker_Sub_T_Dict__tree_init

```
RestResourceWalker_Sub_T_Dict__tree_init(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] = None,
annotation: Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub_T_Dict`

Functions

process

```
process() -> None
```

Functions

register_rest_rootpoint

```
register_rest_rootpoint(klass: type[RestResourceBase])
```

3.1.13 Rest resource walker

Attributes

TV_RestResourceWalkerFutureResult `module-attribute`

```
TV_RestResourceWalkerFutureResult = TypeVar('TV_RestResourceWalkerFutureResult')
```

Classes

RestResourceWalkerFutureResult

```
RestResourceWalkerFutureResult(source: RestResourceWalker_Sub)
```

Bases: `ABC`, `Generic[TV_RestResourceWalkerFutureResult]`

Attributes

source `instance-attribute`

```
source: RestResourceWalker_Sub = source
```

Functions

chain_process_future

```
chain_process_future() -> Optional[TV_RestResourceWalkerFutureResult]
```

process_future `abstractmethod`

```
process_future(result: Optional[list[TV_RestResourceWalkerFutureResult]]) -> Optional[TV_RestResourceWalkerFutureResult]
```

RestResourceWalker_Root

```
RestResourceWalker_Root(resource: RestResourceBase | Type[RestResourceBase])
```

Attributes

cls_RestResourceWalker_Sub `class-attribute` `instance-attribute`

```
cls_RestResourceWalker_Sub: list[Type[RestResourceWalker_Sub]] = [RestResourceWalker_Sub_T_Dict, RestResourceWalker_Sub_RestFields,
RestResourceWalker_Sub_RestResourceBase]
```

resource `instance-attribute`

```
resource: Type[RestResourceBase]
```

subwalker_argument `instance-attribute`

```
subwalker_argument: Any = None
```

Functions

process

```
process(argument: Optional[Any] = None, deep_limit: Optional[int] = None) -> Optional[TV_RestResourceWalkerFutureResult]
```

RestResourceWalker_Sub

```
RestResourceWalker_Sub(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] = None, annotation:
Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `ABC`, `Generic[TV_RestResourceWalkerFutureResult]`

Attributes

annotation `instance-attribute`

```
annotation: type[RestResourceBase]
```

argument `instance-attribute`

```
argument: Any = argument
```

cls_RestResourceWalkerFutureResult `class-attribute` `instance-attribute`

```
cls_RestResourceWalkerFutureResult: Optional[type[RestResourceWalkerFutureResult[TV_RestResourceWalkerFutureResult]]] = None
```

future_result `instance-attribute`

```
future_result: Optional[RestResourceWalkerFutureResult[TV_RestResourceWalkerFutureResult]] = None
```

future_results_subs `instance-attribute`

```
future_results_subs: Optional[list[RestResourceWalkerFutureResult[TV_RestResourceWalkerFutureResult]]] = None
```

optional `instance-attribute`

```
optional: bool
```

parent `instance-attribute`

```
parent: Optional[RestResourceWalker_Sub] = parent
```

resource `instance-attribute`

```
resource: FieldInfo | Type[RestResourceBase] = resource
```

resource_name `instance-attribute`

```
resource_name: str = resource_name
```

subdatatype `instance-attribute`

```
subdatatype = get_args(self.annotation)
```

Functions

ProcessAnnotation `staticmethod`

```
ProcessAnnotation(resource: FieldInfo | Type[RestResourceBase]) -> tuple[type[Any], bool]
```

chain_process_future

```
chain_process_future() -> Optional[TV_RestResourceWalkerFutureResult]
```

check_type `abstractmethod` `classmethod`

```
check_type(resource: FieldInfo | Type[RestResourceBase]) -> tuple[bool, Type[Any], bool]
```

implementation interface to Factory. The factory will call this specialized method on each implementation to find a supported one.

collect_future_result

```
collect_future_result(process_future_result: Optional[RestResourceWalkerFutureResult[TV_RestResourceWalkerFutureResult]]) -> None
```

get `classmethod`

```
get(subs: list[type[RestResourceWalker_Sub]], resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] =
None, argument: Optional[Any] = None) -> Optional[RestResourceWalker_Sub]
```

get_future `abstractmethod`

```
get_future() -> Optional[RestResourceWalkerFutureResult]
```

get_sub_resources

```
get_sub_resources() -> list[tuple[str, FieldInfo]]
```

process

```
process()
```

RestResourceWalker_Sub_RestFields

```
RestResourceWalker_Sub_RestFields(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] = None,
annotation: Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub`

Functions

check_type `classmethod`

```
check_type(resource: FieldInfo | Type[RestResourceBase]) -> tuple[bool, Type[Any], bool]
```

get_future

```
get_future() -> Optional[RestResourceWalkerFutureResult]
```

RestResourceWalker_Sub_RestResourceBase

```
RestResourceWalker_Sub_RestResourceBase(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] = None,
annotation: Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub`

Functions

check_type `classmethod`

```
check_type(resource: FieldInfo | Type[RestResourceBase]) -> tuple[bool, Type[Any], bool]
```

get_future

```
get_future() -> Optional[RestResourceWalkerFutureResult]
```

get_sub_resources

```
get_sub_resources() -> list[tuple[str, FieldInfo]]
```

RestResourceWalker_Sub_T_Dict

```
RestResourceWalker_Sub_T_Dict(resource_name: str, resource: FieldInfo | Type[RestResourceBase], parent: Optional[RestResourceWalker_Sub] = None, annotation:
Optional[type[RestResourceBase]] = None, _optional: Optional[bool] = None, argument: Optional[Any] = None)
```

Bases: `RestResourceWalker_Sub`

Functions

check_type `classmethod`

```
check_type(resource: FieldInfo | Type[RestResourceBase]) -> tuple[bool, Type[Any], bool]
```

get_future

```
get_future() -> Optional[RestResourceWalkerFutureResult]
```

get_sub_resources

```
get_sub_resources() -> list[tuple[str, FieldInfo]]
```

3.1.14 Rest types

Attributes

NoneType `module-attribute`

```
NoneType = type(None)
```

TV_RestResourceBase `module-attribute`

```
TV_RestResourceBase = TypeVar('TV_RestResourceBase', bound='RestResourceBase')
```

TV_SupportedRESTFields `module-attribute`

```
TV_SupportedRESTFields = TypeVar('TV_SupportedRESTFields', UUID, str, int, float, bool, bytes, datetime, Path, IPv4Address, IPv4Network, NoneType)
```

T_AllSupportedContainers `module-attribute`

```
T_AllSupportedContainers = Union[T_Dict, 'RestResourceBase']
```

T_AllSupportedFields `module-attribute`

```
T_AllSupportedFields = T_Dict | T_FieldValue
```

T_Dict `module-attribute`

```
T_Dict = dict[T_DictKey, T_DictValues]
```

T_DictKey `module-attribute`

```
T_DictKey = Union[UUID, str, int, float, bool, bytes, Path, IPv4Address, IPv4Network]
```

T_DictValues `module-attribute`

```
T_DictValues = T_FieldValue
```

T_FieldValue `module-attribute`

```
T_FieldValue = Union[T_SupportedRESTFields, 'RestResourceBase']
```

T_Gen_DictKeys `module-attribute`

```
T_Gen_DictKeys: type = type({}.keys())
```

T_ListIndex `module-attribute`

```
T_ListIndex = NewType('T_ListIndex', int)
```

T_ListSize `module-attribute`

```
T_ListSize = NewType('T_ListSize', int)
```

T_SupportedRESTFields `module-attribute`

```
T_SupportedRESTFields = Union[UUID, str, int, float, bool, bytes, datetime, Path, IPv4Address, IPv4Network, NoneType]
```

T_T_DictKey `module-attribute`

```
T_T_DictKey = type[T_DictKey]
```

T_T_DictValues `module-attribute`

```
T_T_DictValues = type[T_DictValues]
```

T_T_FieldValue `module-attribute`

```
T_T_FieldValue = type(T_FieldValue)
```

Classes

rsrc_type

Bases: `Enum`

Attributes

dict `class-attribute` `instance-attribute`

```
dict = auto()
```

field `class-attribute` `instance-attribute`

```
field = auto()
```

list `class-attribute` `instance-attribute`

```
list = auto()
```

resource `class-attribute` `instance-attribute`

```
resource = auto()
```

rsrc_verb

Bases: `Enum`

Attributes

DELETE `class-attribute` `instance-attribute`

```
DELETE = auto()
```

GET `class-attribute` `instance-attribute`

```
GET = auto()
```

POST `class-attribute` `instance-attribute`

```
POST = auto()
```

PUT `class-attribute` `instance-attribute`

```
PUT = auto()
```