Annotations use in the API

Introduction

This document describes annotations used in the API commands and commands' responses.

API command annotations

Each API command should include following annotations:

@APICommand - command level @. Used by the API doc builder only

Fields:

- name (String, should be unique and is required) - api command name to use for over the wire requests. Should be camelcased.
- description (String, empty if not specified) - provides brief description for the API command.
- usage (String, empty if not specified) - gives the example of the command usage (supported parameters combinations, any other details that are too big to be included to the description)
- includeInApiDoc (boolean, true if not specified) - defines if the command should be included to the API doc
- since (String, empty if not specified) - mentions the product version the command was introduced in.
- responseObject (class<? extends BaseResponse>) - type of the response object for the command.
- requestHasSensitiveInfo (boolean, defaulted to true)
- responseHasSensitiveInfo (boolean, defaulted to true)
- authorized(RoleType[], empty by default) - define the user account roles eligible to execute this command. This can be used to enable default ACL for an API.
- entityType(IAMEntityType[], empty by default) -

@Parameter - api request parameter level @. Used by Api doc builder as well as by ApiDispatcher (class dispatching the api request and doing basic verification for the parameters)

Fields:

- name (String, empty if not specified) - parameter name. The name should be defined in ApiConstants class as its value is going to be used by the corresponding API response object. Lower case only
- description (String, empty if not specified) - brief description for the parameter
- required (boolean, false if not specified) - if parameter is required by the command. When required=true and the corresponding parameter is passed in, the command will fail at the dispatch level.
- type (CommandType enum, defaulted to Object if not specified) - parameter type (see CommandType enum below). When passed value's type is different from whatever is defined in the annotation, the command will fail at the dispatch level.
- entityType (a response class) - if field will have a uuid, developer need to put type to CommandType.UUID and put entityType to a response class. The response class via @EntityReference should point to an interface that is implemented by a VO class that has a @Table from which the uuid can be used to query and get a VO object
- collectionType (CommandType enum, defaulted to Object if not specified) - if the type of the parameter is instance of List, the collection type has to be defined. The type of all collection elements should be checked against this parameter. Happens at the dispatch level as well.
- expose (boolean, defaulted to true if not specified) - if set to false, 1) the parameter will be ignored even if it's passed to the request 2) Mostly used for the cases when we need an instance variable for the command, but don't want to expose it to the caller.
- includeInApiDoc (boolean, defaulted to true if not specified) - if set to false, won't be included to the API doc.
- length (integer, defaulted to 255 if not specified) - max length for the String parameters.
- since (String, empty if not specified) - mentions the product version the parameter was introduced in.
- authorized(RoleType[], empty by default) - define the Account roles eligible to pass this parameter to the request. By default, its everyone
- validations(ApiArgValidator[], empty by default) - define parameter validations to do string based null or empty checks (!Strings.isNullOrEmpty using ApiArgValidator.NotNullOrEmpty) or number checks (> 0, using ApiArgValidator.PositiveNumber)
- String retrieveMethod() default "getById" -

```
public enum CommandType { BOOLEAN, DATE, FLOAT, INTEGER, SHORT, LIST, LONG, OBJECT, MAP, STRING, TZDATE, UUID }
```

@ACL- api request parameter level @.

Used to invoke IAM to do access control check on the entity identified by the annotated parameter.

Fields:

- accessType - type of access needed on the entity. AccessType Enum, default AccessType.UseEntry;
- checkKeyAccess - boolean default false; For CommandType .MAP, this indicates if the 'key' entities of the Map need to be checked access for
- checkValueAccess - boolean default false; For CommandType .MAP, this indicates if the 'value' entities of the Map need to be checked access for

API response annotations

@EntityReference - annotation on a response class, this points to an interface to a VO and is used to get the dao to translate over the wire uuid parameter to get a VO object

@SerializedName - used by the code for api response serialization and by API doc for defining the API response format.

Fields:

name - the response parameter name. Should be lower case.

@Param - used by API doc writer only

Fields:

- description (String, empty if not specified) - brief description for the parameter
- includeInApiDoc (boolean, defaulted to true if not specified) - if set to false, won't be included to the API doc.
- since (String, empty if not specified) - mentions the product version the parameter was introduced in.
- responseObject (defaulted to Object if not specified) - if the response parameter is the response object on its own (like Nic object in the userVmResponse), then the corresponding responseObject should be defined here.