Defining API Descriptors

Overview

This wiki page contains resources and tips & tricks related to defining API Descriptors.

CREST API Descriptor Specification

Examples

Class	Notes
org.forgerock.openidm.external.email.impl.EmailServiceImpl	• API Descriptor annotations • JSON Schema from flat files • Single "action"
org.forgerock.openidm.idp.impl.IdentityProviderService	• API Descriptor annotations • JSON Schema from annotated POJO classes • Multiple "actions" • Resource schema and "read" endpoint
org.forgerock.openidm.managed.ManagedObjectService	• API Descriptor from Describable interface • JSON Schema read from managed.json

Annotations

Two sets of annotations can often be used to describe portions of an API Descriptor. There are a set of CREST related annotations, that define the actual API endpoints, and another set of annotations that extend Jackson support for JSON Schema v4 generation.

CREST API Descriptor annotations

Java Annotation	Notes
SingletonProvider	
CollectionProvider	
RequestHandler	In OpenIDM, use, final Router router = new DescribableRouter(); for the parent class that adds RequestHandler instances onto the router.
Handler	
Path	
Operation	
ApiError	Defines error responses that the user of the API needs to be aware of.
Example	Points to a JSON example text-file. This is not JSON Schema, but an actual example of the resource / payload.
Parameter	Defines a parameter either as URL path-template (ParameterSource.PATH) or as a query-parameter (ParameterSource.ADDITIONAL).
Schema	Points to a JSON Schema v4 text-file or POJO class (see annotations below).
Create	
Read	
Update	
Delete	
Patch	
Query	

Action / Actions	Actions are unique, because custom request / response payload schemas can be defined, whereas the other CRUDPAQ operations are assumed to use one resource-schema for the endpoint. Use the "Actions" annotation when an endpoint supports multiple actions.

Annotations that generate JSON Schema,

The following annotations, when present on a Java POJO class, will be converted into JSON Schema v4 fields for the CREST resource, and/or "action" request /response.

Please also refer to the Jackson Annotations documentation (examples), which are often necessary when field names need to be renamed or hidden/ignored, for example.

JSON Schema Field	Java Annotation	Notes
type	provided by Jackson	Jackson determines the correct JSON type for the POJO class and nested fields.
properties	provided by Jackson	Fields on an object correspond to getter/setters on your POJO.
enum	provided by Jackson	A Java "enum" will be converted into a JSON Schema "enum".
options /enum_title	org.forgerock.api.annotations. EnumTitle	Provide titles for enum items.
description	org.forgerock.api.annotations. Description	Description of class or field.
title	org.forgerock.api.annotations. Title	Short title for class or field.
items	provided by Jackson	A List<String> or String[] field on the POJO will define itself as a JSON Schema "type":"array" and the "items" will have "type":"string". Warning: might be restricted to one array-item type (not multiple)
uniqueItems	org.forgerock.api.annotations. UniqueItems	"uniqueItems":true means that all array/collection items must be unique
readPolicy	org.forgerock.api.annotations. PropertyPolicies	
returnOnDemand	org.forgerock.api.annotations. PropertyPolicies	Whether or not a field is always returned or if it must be requested via "_fields".
writePolicy	org.forgerock.api.annotations. PropertyPolicies	
errorOnWritePolicyFailure	org.forgerock.api.annotations. PropertyPolicies	
propertyOrder	org.forgerock.api.annotations. PropertyOrder	Jackson normally orders properties (fields) in the order they are defined in the POJO class, so this annotation makes the ordering more specific if that is necessary.
minLength	javax.validation.constraints. Size	Minimum string length. Can be used to allow "empty string".
maxLength	javax.validation.constraints. Size	Maximum string length.
minItems	javax.validation.constraints. Size	Minimum items in an array/set.
maxItems	javax.validation.constraints. Size	Maximum items in an array/set.
minimum (for int /long)	javax.validation.constraints. Min	Minimum value (inclusive) for an int/long.
maximum (for int /long)	javax.validation.constraints. Max	Maximum value (inclusive) for an int/long.
minimum (for any? number)	javax.validation.constraints. DecimalMin	Minimum value (inclusive) for any Java number.
maximum (for any? number)	javax.validation.constraints. DecimalMax	Maximum value (inclusive) for any Java number.
exclusiveMinimum	javax.validation.constraints. DecimalMin	Minimum value (exclusive) for any Java number.
multipleOf	org.forgerock.api.annotations. MultipleOf	"multipleOf":2.0 means that the number is a multiple of 2 (e.g., 2, 4, 6...)

exclusiveMaximum	javax.validation.constraints.DecimalMin	Maximum value (exclusive) for any Java number.
required	javax.validation.constraints.NotNull	Specify field as "non-null", which translates to "required":true in JSON Schema.
readOnly	org.forgerock.api.annotations.ReadOnly	Field is read-only.
pattern	javax.validation.constraints.Pattern	Regex that a "string" field should match.
format	org.forgerock.api.annotations.Format	Supported "format" values by JSON type, • integer • int32 (default) • int64 • number • double (default) • float • int32 • int64 • string • byte (base64) • binary (base64) • full-date (RFC 3339) • date-time (RFC 3339) • password • uuid
default	org.forgerock.api.annotations.Default	The "default" value, represented as a string. NOTE: all JSON types are supported when defining a "default" field using a JSON Schema text-file (non-annotation)
additionalProperties	org.forgerock.api.annotations.AdditionalProperties	Useful when the JSON object is a "map" of "string" keys and values of the given type, but the keys might be only known at runtime.

JSON Schema Text Files

JSON Schema's for API Descriptor resources and request / response payloads can be defined in text files, using JSON Schema v4 syntax. For most common use-cases annotations on a POJO can generate the appropriate schema, but there are times when the annotations fall short. For example, the @Default annotation currently only supports string representations of simple default-values (e.g., "text", "1.0", "true"), so if you want to define a default JSON object or array value, you will need to use a raw JSON Schema text-file.