OGC API Coverages

Table of Contents

Open Geospatial Consortium

Submission Date: <yyyy-mm-dd>

Approval Date: <yyyy-mm-dd>

Publication Date: 2019-03-06

External identifier of this OGC® document: http://www.opengis.net/doc/{doc-type}/{standard}/{m.n}

Internal reference number of this OGC® document: YY-nnnrx

Version: 0.0.1

Category: OGC® Implementation Specification

Editor: Charles Heazel

OGC API Coverages

Warning

This document is not an OGC Standard. This document is distributed for review and comment. This document is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: OGC®ImplementationSpecification
Document subtype: if applicable
Document stage: Draft
Document language: English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize

you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

i. Abstract

<Insert Abstract Text here>

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, <tags separated by commas>

iii. Preface

NOTE	Insert Preface Text here. Give OGC specific commentary: describe the technical content, reason for document, history of the document and precursors, and plans for future work. > Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights. Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

iv. Submitting organizations

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

Organization name(s)

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name Affiliation

Chapter 1. Scope

This [OGC API Coverages](https://github.com/opengeospatial/ogc_api_coverages) specification establishes how to access coverages as defined by the [Coverage Implementation Schema (CIS) 1.1](http://docs.opengeospatial.org/is/09-146r6/09-146r6.html) through [OpenAPI](https://www.openapis.org/).

1.1. Current scope:

- Only gridded coverages are addressed, not MultiPoint/Curve/Surface/SolidCoverages. Reason is that gridded coverages receive most attention today.
- Only GeneralGridCoverage is addressed, other coverage types will follow later. Reason is to have a first version early which shows and allows to evaluate the principles.
- Only coverage extraction functionality is considered, not general processing (as is provided with Web Coverage Service (WCS) extensions such as the Processing Extension). Exceptions from this rule are subsetting including band subsetting, scaling, and CRS conversion and data format encoding, given their practical relevance.
- Subsetting is considered in the query component only for now. As typically all dimensions in a coverage are of same importance subsetting might not fit perfectly in the hierarchical nature of the path component. Further, subsetting may reference any axis and leave out any other, which makes positional parameters unsuitable. Nevertheless subsetting in the path component particularly limited to fixed subsets might be considered in a future version.

As such, the functionality provided below resembles OGC Web Coverage Service (WCS) 2.1 Interface Standard - Core http://docs.opengeospatial.org/is/17-089r1/17-089r1.html.

Chapter 2. Conformance

This standard defines XXXX.

Requirements for N standardization target types are considered: * AAAA * BBBB

Conformance with this standard shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

In order to conform to this OGC® interface standard, a software implementation shall choose to implement: * Any one of the conformance levels specified in Annex A (normative). * Any one of the Distributed Computing Platform profiles specified in Annexes TBD through TBD (normative).

All requirements-classes and conformance-classes described in this document are owned by the standard(s) identified.

Chapter 3. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

Insert References here. If there are no references, state “There are no normative references”.

References are to follow the Springer LNCS style, with the exception that optional information may be appended to references: DOIs are added after the date and web resource references may include an access date at the end of the reference in parentheses. See examples from Springer and OGC below.

Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)

May, P., Ehrlich, H.C., Steinke, T.: ZIB Structure Prediction Pipeline: Composing a Complex Biological Workflow through Web Services. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1148–1158. Springer, Heidelberg (2006)

Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)

Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: 10th IEEE International Symposium on High Performance Distributed Computing, pp. 181–184. IEEE Press, New York (2001)

NOTE

Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. Technical report, Global Grid Forum (2002)

National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov

ISO / TC 211: ISO 19115-1:2014 Geographic information — Metadata — Part 1: Fundamentals (2014)

ISO / TC 211: ISO 19157:2013 Geographic information — Data quality (2013)

ISO / TC 211: ISO 19139:2007 Geographic information — Metadata — XML schema implementation (2007)

ISO / TC 211: ISO 19115-3: Geographic information — Metadata — Part 3: XML schemas (2016)

OGC: OGC 15-097 OGC Geospatial User Feedback Standard. Conceptual Model (2016)

OGC: OGC 12-019, OGC City Geography Markup Language (CityGML) Encoding Standard (2012)

OGC: OGC 14-005r3, OGC IndoorGML (2014)

Chapter 4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r8], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

For the purposes of this document, the following additional terms and definitions apply.

4.1. term name

text of the definition

Chapter 5. Conventions

This sections provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

5.1. Identifiers

The normative provisions in this standard are denoted by the URI

http://www.opengis.net/spec/{standard}/{m.n}

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base.

Chapter 6. Overview

6.1. Evolution from OGC Web Services

OGC Web Service (OWS) standards implemented a Remote-Procedure-Call-over-HTTP architectural style using XML for payloads. This was the state-of-the-art when OGC Web Services were originally designed in the late 1990s and early 2000s. The OGC API (OAPI) Common specifies the common kernel of an API approach to services that follows the current Web architecture. In particular, the W3C/OGC best practices for sharing Spatial Data on the Web as well as the W3C best practices for sharing Data on the Web.

Beside the general alignment with the architecture of the Web (e.g., consistency with HTTP/HTTPS, hypermedia controls), another goal for OGC API standards is modularization. This goal has several facets:

- Clear separation between core requirements and more advanced capabilities. This document specifies the requirements that are relevant for almost everyone who wants to share or use coverage data on a fine-grained level. Additional capabilities that several communities are using today will be specified as extensions to the Core API.
- Technologies that change more frequently are decoupled and specified in separate modules ("requirements classes" in OGC terminology). This enables, for example, the use/re-use of new encodings for spatial data or API descriptions.
- Modularization is not just about a single "service". OGC APIs will provide building blocks that can be reused in APIs in general. In other words, a server supporting the WCS 3.0 API should not be seen as a standalone WCS service. Rather it should be viewed as a collection of API building blocks which together implement WCS 3.0 capabilities. A corollary of this is that it should be possible to implement an API that simultaneously conforms to conformance classes from WFS 3.0, WCS 3.0 and other OGC Web API standards.

This approach intends to support two types of client developers:

- Those that have never heard about OGC. Developers should be able to create a client using the API definition without the need to read an OGC standard (they may need to learn a little bit about geometry, etc.);
- Those that want to write a "generic" client that can access OGC APIs, i.e. are not specific for a particular API.

As a result of this modernization, OGC API implementations are not necessarily always backwards compatible with OWS implementations per se, although - for the protection of implementation and service assets - compatibility will be considered as much as possible. However, a design goal is to define OGC APIs in a way so that an OAPI interface can be mapped to an OWS implementation. OGC APIs are intended to be simpler and more modern, but still an evolution from the previous versions and their implementations.

The OGC coverage data and service model is well prepared for such a step: * Both the coverage data model (CIS) and the service model (WCS) are modular suites of standards, with a focus on simplicity and ease of use in the CIS and WCS Core and a rich ecosystem of coverage-specific structuring and

functionality in the extensions. Application Profiles provide specialized services, e.g., for remote sensing imagery. * The WCS Core provides the very basic functionality, coverage access, subsetting, and format encoding. Hence, WCS Core is an ideal starting point for an OAPI Coverages service. * The WCS suite has a clear separation in functionality and protocol binding (i.e., request language). For example, all WCS Core and Extension functionality can be expressed equivalently in GET/KVP, POST/XML, and SOAP. Adding an OpenAPI protocol binding, therefore, can easily fit into this ecosystem, thereby avoiding a backwards incompatible change (like "WCS 3.0") and rather establishing incrementally a WCS 2.2.

The modernization is discussed in more detail and from a coverage independent view here.

6.2. Encodings

NOTE	Metadata will be returned as well as complete coverages. For the former, HTML should be included for this purpose, for the latter some format like should be included. Additional encodings will be needed to support coverage responses, such as GML and RDF already supported by CIS in addition to JSON.

This standard does not mandate any particular encoding or format. However, it does provide extensions for encodings which are commonly used in OGC APIs. In addition to HTML as the standard encoding for human-centric Web content, rules for commonly used encodings for spatial data on the web are provided (GeoJSON, GML).

None of these encodings is mandatory and an implementation of the *Core* requirements class may implement none of them. Alternatively, an implementation of the *Core* requirements class may choose to implement completely different encodings instead.

Support for HTML is recommended. HTML is the core language of the World Wide Web. An API that supports HTML will support browsing the spatial resources with a web browser and will also enable search engines to crawl and index those resources.

CAUTION	The search engine point is not valid. HTML describes syntax, not semantics (remember why we introduced XML and RDF?). See Issue 13.

GeoJSON is a commonly used format that is simple to understand and well supported by tools and software libraries. Since most Web developers are comfortable with using a JSON-based format, it is recommended for APIs which expose feature data as long as the data can be represented in GeoJSON for the intended use.

Some examples of cases that are out-of-scope for GeoJSON are:

- When solids are used for geometries (e.g. in a 3D city model),
- Geometries that include non-linear curve interpolations that cannot be simplified (e.g., use of arcs in authoritative geometries),
- Geometries that have to be represented in a coordinate reference system that is not based on WGS 84 longitude/latitude (e.g. an authoritative national reference system),
- Features that have more than one geometric property.

Among the shortcomings of GeoJSON, which have been debated widely, is the substantially restricted support for multi-dimensional coordinates and coordinate reference systems. For exampele, European ETRS89 is not supported.

The recommendations for using HTML and GeoJSON reflect the importance of HTML and the current popularity of JSON-based data formats. As the practices in the Web community evolve, the recommendations will likely be updated in future versions of this standard to provide guidance on using other encodings.

This part of the OAPI standard does not provide any guidance on other encodings. The supported encodings, or more precisely the media types of the supported encodings, can be determined from the API definition. The desired encoding is selected using HTTP content negotiation.

For example, if the server supports GeoJSON Text Sequences an encoding that is based on JSON text sequences, and GeoJSON, and supports streaming by making the data incrementally parseable, could use the media type `application/geo+json-seq` in the header of a corresponding request to the server.

In addition, HTTP supports compression. Therefore the standard HTTP compression mechanisms can be used to reduce the size of messages between the server and the client.

6.3. Examples

NOTE	Update the following content once a coverage example has been agreed on.

This document uses a simple example throughout the document: The dataset contains buildings and the API provides access to them through a single feature collection ("buildings") and two encodings, GeoJSON and HTML.

The buildings have a few (optional) properties: the polygon geometry of the building footprint, a name, the function of the building (residential, commercial or public use), the floor count and the timestamp of the last update of the building feature in the dataset.

This example serves to illustrate the concepts underlying OGC APIs. It does not indicate that OGC APIs are always feature based. Other resource types can and will be implemented as well. But the basic capapablities described in this specification will apply to all.

Chapter 7. Clause containing normative material

Paragraph

7.1. Requirement Class Common

Paragraph – intro text for the requirement class.

Use the following table for Requirements Classes.

Unresolved directive in clause_7_core.adoc - include::requirements/requirements_class_common.adoc[]

7.1.1. Examples of Common applied to Coverages

- http://acme.com/oapi/collections/{collectionid}/coverages?bbox=160.6,-55.95,-170,-25.89 — returns a list of all coverages intersecting in a specific collection that is in the New Zealand economic zone

7.2. Requirement Class Path

An API exposes resources though a resource path.

General Template:

```
{root}/collections/{collectionid}/coverages/{coverageid/{a}
```

```
protocol := "http" | "https"
server   := server DNS name
root     := {protocol}:{server}
a := {b} | "rangetype" | "metadata" | "rangeset"
b := /domainset | /domainset/{c}
c := generalgrid/{d}
d := "srsname" | "axislabels"
```

Use the following table for Requirements Classes.

Unresolved directive in clause_7_core.adoc - include::requirements/requirements_class_path.adoc[]

7.3. Requirement Class Subset

Without any subsetting parameter the whole coverage extent is the target resource addressed. If a subsetting operation is provided then the coverage subset indicated is the target resource addressed.

Coverage subsetting is indicated through the SUBSET parameter name. The value following the "=" symbol is built as follows:

```
SubsetSpec:            SUBSET = axisName (intervalOrPoint)
axisName:              {NCName}
intervalOrPoint:       interval | point
interval:              low : high
low:                   point | *
high:                  point | *
point:                 {number} | "{text}"
```

```
Where:
" = double quote = ASCII code 0x42,
{NCName} is an XML-style identifier not containing ":" (colon) characters,
{number} is an integer or floating-point number, and
{text} is some general ASCII text (such as a time and date notation in ISO 8601).
```

Unresolved directive in clause_7_core.adoc - include::requirements/requirements_class_subset.adoc[]

7.3.1. Subsetting Examples

- http://acme.com/oapi/collections/{collectionid}/coverages/{coverageid}?SUBSET=Lat(40) — returns a coverage cutout between (40,10) and (50,20), as multipartcoverage
- http://acme.com/oapi/collections/{collectionid}/coverages/{coverageid}/rangeset?SUBSET=Lat(40) — returns a coverage cutout between (40,10) and (50,20), in the coverage's Native Format
- http://acme.com/oapi/collections/{collectionid}/coverages/{coverageid}?SUBSET=time("2019-03-27") — returns a coverage slice at the timestamp given (in case the coverage is Lat/Long/time the result will be a 2D image)

7.4. Requirements Class Encodings

Unresolved directive in clause_7_core.adoc - include::requirements/requirements_class_encoding.adoc[]

Chapter 8. Requirements classes for encodings

NOTE	The following content is from the OAPI Common specification and should be modified for Coverage APIs. Keep in mind that both coverages and metadata are supplied by the API. So HTML and GeoJSON encoding may be retained for the metadata responses.

8.1. Overview

This clause specifies four pre-defined requirements classes for encodings to be used by an OGC API implementation. These encodings are commonly used encodings for spatial data on the web:

- HTML
- GeoJSON
- Geography Markup Language (GML), Simple Features Profile, Level 0
- Geography Markup Language (GML), Simple Features Profile, Level 2

None of these encodings are mandatory and an implementation of the Core requirements class may also implement none of them but implement another encoding instead.

The Core requirements class includes recommendations to support HTML and GeoJSON as encodings, where practical. Clause 6 (Overview) includes a discussion about recommended encodings.

8.2. Requirement Class "HTML"

Geographic information that is only accessible in formats like GeoJSON or GML has two issues:

- The data is not discoverable using the most common mechanism for discovering information, that is the search engines of the Web;
- The data can not be viewed directly in a browser - additional tools are required to view the data.

Therefore, sharing data on the Web should include publication in HTML. To be consistent with the Web, it should be done in a way that enables users and search engines to access all data.

This is discussed in detail in Best Practice 2: Make your spatial data indexable by search engines [SDWBP]. This standard therefore recommends supporting HTML as an encoding.

Unresolved directive in clause_8_encodings.adoc - include::requirements/requirements_class_html.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/html/REQ_definition.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/html/REQ_content.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/html/REC_schema-org.adoc[]

8.3. Requirement Class "GeoJSON"

GeoJSON is a commonly used format that is simple to understand and well supported by tools and software libraries. Since most Web developers are comfortable with using a JSON-based format supporting GeoJSON is recommended, if the feature data can be represented in GeoJSON for the intended use.

Unresolved directive in clause_8_encodings.adoc - include::requirements/requirements_class_geojson.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/geojson/REQ_definition.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/geojson/REQ_content.adoc[]

Templates for the definition of the schemas for the GeoJSON responses in OpenAPI definitions are available at featureCollectionGeoJSON.yaml and featureGeoJSON.yaml. These are generic schemas that do not include any application schema information about specific feature types or their properties.

Example 1. A GeoJSON FeatureCollection Object response

In the example below, only the first and tenth feature is shown. Coordinates are not shown.

```
{
  "type" : "FeatureCollection",
  "links" : [ {
    "href" : "http://data.example.com/collections/buildings/items/?f=json",
    "rel" : "self",
    "type" : "application/geo+json",
    "title" : "this document"
  }, {
    "href" : "http://data.example.com/collections/buildings/items/?f=html",
    "rel" : "alternate",
    "type" : "text/html",
    "title" : "this document as HTML"
  }, {
    "href" :
"http://data.example.com/collections/buildings/items/?f=json&startIndex=10&limit=1
0",
    "rel" : "next",
    "type" : "application/geo+json",
    "title" : "next page"
  } ],
  "timeStamp" : "2018-04-03T14:52:23Z",
  "numberMatched" : 123,
  "numberReturned" : 10,
  "features" : [ {
    "type" : "Feature",
    "id" : "123",
    "geometry" : {
      "type" : "Polygon",
      "coordinates" : [ ... ]
    },
    "properties" : {
      "function" : "residential",
      "floors" : "2",
      "lastUpdate" : "2015-08-01T12:34:56Z"
    }
  }, { ...
  }, {
    "type" : "Feature",
    "id" : "132",
    "geometry" : {
      "type" : "Polygon",
      "coordinates" : [ ... ]
    },
    "properties" : {
      "function" : "public use",
      "floors" : "10",
      "lastUpdate" : "2013-12-03T10:15:37Z"
    }
  } ]
}
```

Example 2. A GeoJSON Feature Object response

In the example below, coordinates are not shown.


```
{
  "type" : "Feature",
  "links" : [ {
    "href" : "http://data.example.com/collections/buildings/items/123/?f=json",
    "rel" : "self",
    "type" : "application/geo+json",
    "title" : "this document"
  }, {
    "href" : "http://data.example.com/collections/buildings/items/123/?f=html",
    "rel" : "alternate",
    "type" : "text/html",
    "title" : "this document as HTML"
  }, {
    "href" : "http://data.example.com/collections/buildings/items",
    "rel" : "collection",
    "type" : "application/geo+json",
    "title" : "the collection document"
  } ],
  "id" : "123",
  "geometry" : {
    "type" : "Polygon",
    "coordinates" : [ ... ]
  },
  "properties" : {
    "function" : "residential",
    "floors" : "2",
    "lastUpdate" : "2015-08-01T12:34:56Z"
  }
}
```


8.4. Requirement Class "Geography Markup Language (GML), Simple Features Profile, Level 0"

In addition to HTML and GeoJSON, a significant volume of feature data is available XML-based formats, notably GML. Therefore, this standard specifies requirement classes for GML. The Simple Features Profile, Level 0, is the simplest profile of GML and is typically supported by tools. The GML Simple Features Profile is restricted to data with 2D geometries supported by most tools. In addition, the Level 0 profile is limited to features that can be stored in a tabular data structure.

Unresolved directive in clause_8_encodings.adoc - include::requirements/requirements_class_gmlsf0.adoc[]

Unresolved directive in clause_8_encodings.adoc -

include::requirements/gmlsf0/REQ_definition.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/gmlsf0/REQ_content.adoc[]

Table 1. Media types and XML elements for each resource

Resource	Path	XML root element
Landing page	`/`	`wfs:LandingPage`
Conformance classes	`/conformance`	`wfs:ConformsTo`
Feature collections metadata	`/collections`	`wfs:Collections`
Feature collection metadata	`/collections/{collectionId}`	`wfs:Collections`, with just one entry for the collection `collectionId`
Feature collection	`/collections/{collectionId}/items`	`wfs:FeatureCollection`
Feature	`/collections/{collectionId}/items/{featureId}`	substitutable for `gml:AbstractFeature`

The namespace prefixes used above are:

- `wfs`: `http://www.opengis.net/wfs/3.0`
- `gml`: `http://www.opengis.net/gml/3.2`
- `atom`: `http://www.w3.org/2005/Atom`
- `xlink`: `http://www.w3.org/1999/xlink`

NOTE The above examples should be adapted to coverages, ie: referencing WCS and CIS instead of WSF and GML 3.2.

The API definition resource at path `/api` is not included in Table 1. This requirements class does not prescribe any API definition approach.

The mapping of the content from the responses specified in the Core requirements class to the XML is straightforward. All links are encoded using `atom:link` elements except in GML features where simple Xlinks are used.

Annex C has example responses in XML.

NOTE The `wfs:FeatureCollection` element deliberately goes beyond the permitted content specified in the GML Simple Features Profile, section 8.4.2. This is necessary to support the hypermedia controls and other relevant content for a Web Feature Service API.

8.5. Requirement Class "Geography Markup Language (GML), Simple Features Profile, Level 2"

The difference between this requirement class and the Level 0 requirements class is that non-

spatial feature properties are not restricted to atomic values (strings, numbers, etc.).

Unresolved directive in clause_8_encodings.adoc - include::requirements/requirements_class_gmlsf2.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/gmlsf2/REQ_definition.adoc[]

Unresolved directive in clause_8_encodings.adoc - include::requirements/gmlsf2/REQ_content.adoc[]

NOTE	The coverage GML encoding corresponding to GML Simple Features is defined in OGC CIS.

Chapter 9. Requirements class "OpenAPI 3.0"

NOTE

The OAS requirements class should be limited to OpenAPI 3.0 requirements that are specific to coverage resources. These requirements are an extension to the OpenAPI requirements in OAPI Common.

What follows is from OAPI common. It should be replaced with approapriate coverage centered requirements.

9.1. Basic requirements

APIs conforming to this requirements class document themselves by an OpenAPI Document.

Unresolved directive in clause_9_oas.adoc - include::requirements/requirements_class_oas30.adoc[]

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_oas-definition-1.adoc[]

CAUTION

ISSUE 117
The OpenAPI media type has not been registered yet with IANA and will likely change. We need to update the media type after registration.

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_oas-definition-2.adoc[]

CAUTION

Related to ISSUE 90
If we have a rigid path pattern there seems to be no need to add requirements for fixed operationId values. However, if the path pattern would be flexible, maybe we should require specific operationIds for selected resources?

Two example OpenAPI documents are included in Annex B.

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_oas-impl.adoc[]

9.2. Complete definition

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_completeness.adoc[]

Note that APIs that, for example, are access-controlled (see Security), support web cache validation, CORS or that use HTTP redirection will make use of additional HTTP status codes beyond regular codes such as `200` for successful GET requests and `400`, `404` or `500` for error situations. See [http_status_codes].

Clients have to be prepared to receive responses not documented in the OpenAPI definition. For example, additional errors may occur in the transport layer outside of the server.

9.3. Exceptions

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_exception-codes.adoc[]

Example 3. An exception response object definition

```
description: An error occurred.
content:
  application/json:
    schema:
      $ref:
https://raw.githubusercontent.com/opengeospatial/OAPI/openapi/schemas/exception.ya
ml
  text/html:
    schema:
      type: string
```

9.4. Security

Unresolved directive in clause_9_oas.adoc - include::requirements/oas30/REQ_security.adoc[]

The OpenAPI specification currently supports the following security schemes:

- HTTP authentication,
- an API key (either as a header or as a query parameter),
- OAuth2's common flows (implicit, password, application and access code) as defined in RFC6749, and
- OpenID Connect Discovery.

CAUTION	ISSUE 41 How does a client determine which security protocols/standards/etc. a server supports?

Chapter 10. Media Types for any data encoding(s)

A section describing the MIME-types to be used is mandatory for any standard involving data encodings. If no suitable MIME type exists in http://www.iana.org/assignments/media-types/index.html then this section may be used to define a new MIME type for registration with IANA.

Annex A: Conformance Class Abstract Test Suite (Normative)

NOTE	Ensure that there is a conformance class for each requirements class and a test for each requirement (identified by requirement name and number)

A.1. Conformance Class A

A.1.1. Requirement 1

Test id:	/conf/conf-class-a/req-name-1
Requirement:	/req/req-class-a/req-name-1
Test purpose:	Verify that…
Test method:	Inspect…

A.1.2. Requirement 2

Unresolved directive in OAPI_Coverages.adoc - include::annex-history.adoc[]

Unresolved directive in OAPI_Coverages.adoc - include::annex-bibliography.adoc[]