OGC® DOCUMENT: 20-057

External identifier of this OGC® document: http://www.opengis.net/doc/IS/ogcapi-tiles-1/1.0

Open Geospatial Consortium

OGC API - TILES - PART 1: CORE

STANDARD
Implementation

APPROVED

Version: 1.0
Submission Date: 2022-06-15
Approval Date: 2022-08-26
Publication Date: 2022-11-10
Editor: Joan Masó, Jérôme Jacovella-St-Louis

Note

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.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF RECOMMENDATIONS

I ABSTRACT

The *OGC API — Tiles* Standard defines building blocks for implementing Web APIs that support the retrieval of tiled geospatial information. A Web API is an application programming interface for either a web server or a web browser [Wikipedia, 2022].

The OGC suite of Web API standards is an extensible framework for building HTTP based services that can be accessed in different applications on different platforms such as the Web, desktop, mobile, etc. The *OGC API — Tiles* Standard specifies how different forms/types of geospatial resources are supported, such as tiles of vector features ("vector tiles"), coverages, and maps (or imagery). Although OGC API — Tiles can be used independently, the building blocks can be combined with other OGC API Standards for additional capabilities or increased interoperability for specific types of data. The *OGC API — Tiles* Standard references the *OGC Two-Dimensional Tile Matrix Set (TMS) and Tile Set Metadata* Standard [OGC 17-083r4]. That Standard defines logical models and encodings for specifying tile matrix sets and describing tile sets. A tile matrix set is a tiling scheme that enables an application to partition and index space based on a set of regular grids defined for multiple scales in a Coordinate Reference System (CRS).

The *OGC API — Tiles* Standard is an alternative to the OGC's Web Map Tile Service (WMTS) Standard. Instead of a fixed Web interface, OGC API — Tiles focuses on simple reusable REST API building blocks which can be described using the OpenAPI specification. Whereas WMTS focused on map tiles, the *OGC API — Tiles* Standard is designed to support any form of tiled data.

II KEYWORDS

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

ogcdoc, OGC document, tiling, tiles, WMTS, map tiles, vector tiles, tiled feature data

III PREFACE

This document defines the *OGC API — Tiles — Part 1: Core* Standard. A Web API conforming to this Standard can serve tiles of spatially referenced data or maps with predefined content, extent, and resolution. Suggested additions, changes and comments on this standard are welcome and encouraged. Such suggestions may be submitted using the issues log on the GitHub repository: https://github.com/opengeospatial/ogcapi-tiles .

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 SECURITY CONSIDERATIONS

No security considerations have been made for this document.

V SUBMITTING ORGANIZATIONS

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

- Universitat Autònoma de Barcelona (CREAF)
- US Army Geospatial Center
- Ecere Corporation
- Esri

VI SUBMITTERS

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

Name	Affiliation
Joan Masó	Universitat Autònoma de Barcelona (CREAF)
Chuck Heazel	Heazel Tech
Jeff Harrison	US Army Geospatial Center (AGC)
Jérôme Jacovella-St-Louis	Ecere Corporation
Satish Sankaran	Esri

1

SCOPE

1 SCOPE

The *OGC API — Tiles* Standard specifies the behavior of Web APIs that provide access to tiles of one or more geospatial data resources (collections) that the Web API offers. This Standard defines how to discover which resources offered by the Web API can be retrieved as tiles, get metadata about the available tile sets (including according to which tile matrix set each tile set is partitioned and the limits of that tile set within a common potentially global tile matrix set) and how to request a tile. This Standard is sometimes referred to as the *Tiles API*.

The core conformance class is defined in a way that could be easily included in a web API, even if that API does not conform to the *OGC API — Common* Standard. A web API can combine some requirements classes of this OGC API Standard with those of other OGC API Standards (including *OGC API — Common*) to extend the scope of the Web API by adding functionality.

2

CONFORMANCE

2 CONFORMANCE

The *OGC API — Tiles* Standard defines multiple requirements classes and their associated conformance classes.

The standardization targets of all conformance classes are “Web APIs”.

The Tiles API requirements classes are summarized below and in more detail in subsequent clauses.

2.1. Requirements classes defining resources

Requirements Class “Core” (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core)

The Core Requirements Class specifies requirements that all implementation instances of the Tiles API must support if claiming conformance with this Standard. The Core class defines how to retrieve individual tiles by building a URI from three variables corresponding to the tile matrix, tile row and tile column for that tile.

Table 1 — Overview of resource and common direct links that correspond to tiles defined in the Core Requirements Class

RESOURCE NAME	COMMON PATH
Tile	`…/{tileMatrix}/{tileRow}/{tileCol}`

NOTE: The path template is recommended, but not prescribed. Ordering the parameters differently within the URI is allowed.

Requirements Class “TileSet” (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tileset)

The *TileSet* Requirements Class defines a mechanism for describing a tileset using a specific tile matrix set and a way of obtaining a templated link to the individual tiles.

Table 2 — Overview of resources and common direct links that correspond to the tileset

RESOURCE NAME	COMMON PATH
Tileset	`…/tiles/{tileMatrixSetId}`
Tile	`…/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`

Requirements Class "TileSets List" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tilesets-list)

The *TileSets List* Requirements Class defines a generic operation for retrieving a list of tilesets, without association to any particular type of resources.

Table 3 — Overview of resource and common direct link that corresponds to the tileset list

RESOURCE NAME	COMMON PATH
Tileset list	`…/tiles`

2.2. Requirements Classes defining data origins

Requirements Class "Dataset TileSets" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/dataset-tilesets)

The *Dataset Tilesets* Requirements Class defines how to retrieve all tiles for a dataset that could potentially consist of multiple geospatial data resources. All implementation instances of the Tiles API must implement this Requirements Class if they are claiming to support **dataset** tiles following this *OGC API — Tiles — Part 1: Core* Standard. Dataset tiles may combine content from multiple geospatial resources, regardless of whether those are available separately (as tiles or otherwise).

Table 4 — Overview of resource and common direct links that correspond to the dataset tileset

RESOURCE NAME	COMMON PATH
Vector tileset list	`/tiles`
Map tileset list	`/map/tiles`
Styled Map tileset list	`/styles/{styleId}/map/tiles`

Requirements Class "GeoData TileSets" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/geodata-tilesets)

The *GeoData TileSets* Requirements Class supports the retrieval of tiles from a specific geospatial data resource.

Table 5 — Overview of resource and common direct links that correspond to the geospatial data resources tilesets

RESOURCE NAME	EXAMPLE OF POSSIBLE PATHS
Vector tileset list	`/collections/{collectionId}/tiles`
Map tileset list	`/collections/{collectionId}/map/tiles`
Styled Map tileset list	`/collections/{collectionId}/styles/{styleId}/map/tiles`

2.3. Requirements classes defining query parameters

Requirements Class "Collections Selection" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections-selection)

The *Collections Selection* Requirements Class supports the listing of specific geospatial data resources from which to retrieve tiles such as for use with data set tiles.

Table 6 — Overview of resource and common direct links that correspond to geodata selection

RESOURCE NAME	EXAMPLE OF POSSIBLE PATHS
Vector Tileset	`/tiles/{tileMatrixSetId}?collections={collectionId},{collectionId},…`
Vector Tile	`/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}?collections={collectionId},{collectionId},…`
Map tileset	`/map/tiles/{tileMatrixSetId}?collections={collectionId},{collectionId},…`
Map tile	`/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}?collections={collectionId},{collectionId},…`

Requirements Class "DateTime" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/datetime)

The *DateTime* Requirements Class specifies how to provide tiles in a domain that has a generic time dimension.

2.4. Requirements classes for specific resource representations

Requirements Class "OpenAPI Specification 3.0 API definition" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/oas30)

The *OpenAPI Specification 3.0* Requirements Class specifies requirements for an OpenAPI 3.0 definition in addition to those defined in *OGC API — Common — Part 1: Core*.

Requirements Class "XML TileSet Metadata" (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/xml)

The *XML TileSet Metadata* Requirements Class specifies how to use XML as an alternative encoding for describing tilesets.

Requirements Classes for tile encodings

The *OGC API — Tiles* Standard does not mandate a specific encoding or format for representing tiles and remains flexible and extensible to other formats that users and providers might need. However, Requirements Classes are provided for the following common tile encodings:

- PNG (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/png)
- JPEG (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/jpeg)
- TIFF (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tiff)
- NetCDF (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/netcdf)
- GeoJSON (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/geojson)
- Mapbox Vector Tiles (http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/mvt)

All Tiles API Requirements Classes can be viewed as building blocks that can be implemented in combination with other more fundamental requirements classes that provide support for Web API discovery, conformity and Web API formal definition (e.g., OpenAPI). Possible alternatives for these fundamental Requirements Classes are *OGC API — Common — Part 1: Core* or *OGC API — Features — Part 1: Core*.

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

NOTE: Despite the fact that full paths and full path templates in the above resource tables are used in many implementations of the *OGC API — Tiles* Standard, these exact paths are ONLY examples and are NOT required by this Standard. Other paths are possible if correctly described by the Web API definition document and the links between resources.

2.5. Declaration of conformance

Conformance with this Standard shall be checked using all the relevant tests specified in Annex A (normative) of this document if the respective conformance class URIs listed in Table 7 are present in the Conformance Declaration response. 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 (OGC 08-134r11) and the OGC Compliance Testing website.

Table 7 — Conformance class URIs

CONFORMANCE CLASS	URI
Core	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/core
TileSet	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tileset
Tilesets list	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tilesets-list
Dataset tilesets	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/dataset-tilesets
Geodata tilesets	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/geodata-tilesets
Collections selection	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/collections-selection
DateTime	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/datetime
OpenAPI Specification 3.0	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/oas30
XML	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/xml
PNG	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/png
JPEG	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/jpeg
TIFF	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tiff
NetCDF	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/netcdf
GeoJSON	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/geojson
Mapbox Vector Tiles	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/mvt

3

NORMATIVE REFERENCES

3 NORMATIVE REFERENCES

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

Carl Reed: OGC 19-014r3, *Topic 22 — Core Tiling Conceptual and Logical Models for 2D Euclidean Space*. Open Geospatial Consortium (2020). https://docs.ogc.org/as/19-014r3/19-014r3.html.

Joan Masó , Jérôme Jacovella-St-Louis: OGC 17-083r4, *OGC Two Dimensional Tile Matrix Set and Tile Set Metadata*. Open Geospatial Consortium (2022). https://docs.ogc.org/is/17-083r4/17-083r4.html.

Charles Heazel: OGC API — Common — Part 1: Core (Draft). OGC 19-072, Open Geospatial Consortium, http://docs.ogc.org/DRAFTS/19-072.html

Charles Heazel: OGC API — Common — Part 2: Geospatial Data (Draft). OGC 20-024, Open Geospatial Consortium, http://docs.ogc.org/DRAFTS/20-024.html

Ben Domenico: OGC 10-090r3, *OGC Network Common Data Form (NetCDF) Core Encoding Standard version 1.0*. Open Geospatial Consortium (2011). https://portal.ogc.org/files/?artifact id=43732.

H. Butler, M. Daly, A. Doyle, S. Gillies, S. Hagen, T. Schaub: IETF RFC 7946, *The GeoJSON Format*. (2016). https://www.rfc-editor.org/info/rfc7946.

Adobe Developers Association: TIFF Specification Revision 6.0. (1992)

Emmanuel Devys, Ted Habermann, Chuck Heazel, Roger Lott, Even Rouault: OGC 19-008r4, *OGC GeoTIFF Standard*. Open Geospatial Consortium (2019). https://docs.ogc.org/is/19-008r4/19-008r4.html.

ISO/IEC: ISO/IEC 10918-1, *Information technology — Digital compression and coding of continuous-tone still images: Requirements and guidelines*. International Organization for Standardization, International Electrotechnical Commission, Geneva https://www.iso.org/standard/18902.html.

ISO/IEC: ISO/IEC 15948, *Information technology — Computer graphics and image processing — Portable Network Graphics (PNG): Functional specification*. International Organization for Standardization, International Electrotechnical Commission, Geneva https://www.iso.org/standard/29581.html.

NOTE: Certain conformance classes have a dependency on OGC API — Common — Part 1: Core or OGC API — Common — Part 2: Geospatial data. In some cases, it is possible to replace these dependencies with OGC 17-069r3 OGC API — Features — Part 1: Core.

4

TERMS AND DEFINITIONS

4 TERMS AND DEFINITIONS

This document uses the terms defined in OGC Policy Directive 49, 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 document and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the ‘ModSpec’. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

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

4.1. Terms and definitions

This document also uses terms defined in Sub-clause “Terms and Definitions” of OGC API — Common, Part 1: Core.

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

4.1.1. **coverage tile**

tile that contains information, often in a gridded form, where the values represent observations or measurements as a count, or quantity using some unit of measure.

Note 1 to entry: Coverage tiles are generated in combination with *OGC API — Coverages*, and can also be generated by combining a subset (trim) and resampling operation. Usually, visualizing a coverage tile on a rendering device implies mapping those values to colors.

4.1.2. **dataset**

a set of data, published or curated by a single agent, and available for access or download in one or more representations (modified from DCAT: https://www.w3.org/TR/vocab-dcat-2/#dcat-scope).

Note 1 to entry: A Web API implementing OGC API — Common often gives access to a single dataset which may be comprised of one or more *geospatial data resources*.

4.1.3. geospatial data resource

web accessible resource that consists of a set of geospatial data.

Note 1 to entry: In Web APIs implementing *OGC API — Common — Part 2: Geospatial Data*, geospatial data resources are referred to as collections and are defined in the *collections* conformance class.

Note 2 to entry: *geodata* is sometimes used in this document as an abbreviation of *geospatial data*

4.1.4. geospatial resource aspect

web accessible resource that represents a component of geospatial information (metadata, schemas…) or geospatial data accessed using a particular mechanism and data model (e.g., feature items, tiles, maps, coverages,…) of a more generic geospatial data resource (e.g., a collection).

Note 1 to entry: Not to be confused with a web accessible resource representation. While resource representations share the same path and are selected by format negotiation, geospatial aspects use different paths. Commonly a geospatial aspect is a subpath of a geospatial data resource.

4.1.5. landing page

any page whose primary purpose is to contain a description of something else. Landing pages often provide summaries or additional information about the thing that they describe. (W3C, URLs in Data Primer)

In the context of the OGC API Standards, a landing page serves as the root node of the API Resource tree and provides the information needed to navigate all the resources exposed through the API. The landing page provides access to the root of a dataset.

4.1.6. **map tile**

tile that contains information in a raster form where the values of cells are colors which can be readily displayed on rendering devices.

Note 1 to entry: Map tiles are generated in combination with *OGC API — Maps*.

4.1.7. **tile**

geometric shape with known properties that may or may not be the result of a tiling (tessellation) process. A tile consists of a single connected “piece” without “holes” or “lines” (topological disc).

In the context of a 2D *tile matrix*, a *tile* is one of the rectangular regions of space, which can be uniquely identified by row and column integer indices, making up the tile matrix.

In the context of a geospatial data *tile set*, a *tile* contains data for such a partition of space as part of an overall set of tiles for that tiled geospatial data.

Note 1 to entry: From OGC 19-014r1: Core Tiling Conceptual and Logical Models for 2D Euclidean Space

Note 2 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

Note 3 to entry: Tiles are useful to efficiently request, transfer, cache, display, store and process geospatial data for a specific resolution and area of interest, providing deterministic performance and scalability for arbitrarily large datasets.

Note 4 to entry: Tiles can contain a variety of data types, such as grid-based pictorial representations (map tiles), coverage subsets (coverage tiles), or feature-based representations (vector tiles).

4.1.8. **tile matrix**

tiling grid in a given 2D coordinate reference system, associated to a specific scale and partitioning space into regular conterminous *tiles*, each of which being assigned a unique identifier.

Note 1 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

Note 2 to entry: Each tile of a tile matrix is uniquely identifiable by integer indices for the row and the column. The number of rows is referred to as the *matrix height*, while the maximum number of columns is referred to as the *matrix width* (the number of columns can vary for different rows in *variable width tile matrices*).

4.1.9. tile matrix set

tiling scheme consisting of a set of *tile matrices* defined at different scales covering approximately the same area and having a common coordinate reference system.

Note 1 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

4.1.10. tile indexing scheme

scheme to uniquely reference a *tile* in a *tiling scheme* by the use of a unique identifier (or set of identifiers), and reversely, which unique identifier (or unique set of identifiers) corresponds to a space satisfying the geometric properties of a specific tile.

Note 1 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

4.1.11. tile set

a set of *tiles* resulting from tiling data according to a particular *tiling scheme*.

Note 1 to entry: From OGC 19-014r1: Core Tiling Conceptual and Logical Models for 2D Euclidean Space, but adapted to clarify that in the context of this document, a tile set refers specifically to a set of tiles containing data and following a common tiling scheme.

4.1.12. tiling scheme

scheme that defines how space is partitioned into individual *tiles*, potentially featuring multiple levels of detail (each tiling at a different granularity to reflect a different resolution or scale).

A tiling scheme defines the spatial reference system and the geometric properties of each tile defined by the scheme. Those properties include which space each tile occupies (the tile's spatial extent), as well as a tile coordinate origin if a particular corner of origin convention is established.

Note 1 to entry: A tiling scheme can be defined on top of a CRS as well as other spatial reference systems such as DGGS and other organizations including irregular ones. In this document, only tiling schemes based on CRSs are supported.

Note 2 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

4.1.13. **tile set metadata**

additional metadata beyond the common properties defining the *tile set*. Such metadata could be an abstract, the owner, the author, or other common metadata. [OGC 19-014r3]

metadata describing common properties defining a *tile set*, layers and styles used to produce the tile set, the limits of the tile matrix with actual data and common metadata such as abstract, owner, author, etc.

Note 1 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

4.1.14. **vector tile**

tiled vector feature data ADMITTED

tile that contains vector data that has been generalized (simplified) at the tile scale resolution and clipped by the tile boundaries.

Note 1 to entry: From OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard

4.1.15. **Web API**

API using an architectural style that is founded on the technologies of the Web. [source: OGC API — Features — Part 1: Core]

Note 1 to entry: See Best Practice 24: Use Web Standards as the foundation of APIs (W3C Data on the Web Best Practices) for more detail.

4.1.16. **Web API based implementation**

a server software that implements a Web API.

Note 1 to entry: The Web API based implementations mentioned in the context of this Standard declare conformity to at least one Conformance Class defined by this Standard.

5

CONVENTIONS

5 CONVENTIONS

This section provides details of conventions used in this document.

5.1. Identifiers

The normative provisions in this standard are denoted by the URI `http://www.opengis.net/spec/ogcapi-tiles-1/1.0`.

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

5.2. Link relations

To express relationships between resources, RFC 8288 (Web Linking) is used.

The following IANA link relation types are used in this Standard:

- **alternate**: Refers to a substitute for this context.
- **self**: Conveys an identifier for the link's context.
- **item**: The target IRI points to a resource that is a member of the collection represented by the context IRI.
- **service-desc**: Identifies a service description for the context that is primarily intended for consumption by machines. (Web API definitions are considered service descriptions)
- **service-doc**: Identifies service documentation for the context that is primarily intended for human consumption.

The following link relation types specified in the *Two Dimensional Tile Matrix Set and Tileset Metadata* Standard are used:

- **http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme**: The target IRI points to a resource that describes the TileMatrixSet according to the 2D-TMS standard.
- **http://www.opengis.net/def/rel/ogc/1.0/dataset**: The target IRI points to a resource representing the dataset (e.g., the root of an OGC Web API).
- **http://www.opengis.net/def/rel/ogc/1.0/geodata**: The target IRI points to a resource representing a collection of geospatial data.

In addition, the following link relation types are used for which no applicable IANA-registered link relation type could be identified:

- **http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector**: The target IRI points to a resource that describes how to provide tile sets of the context resource in vector format.
- **http://www.opengis.net/def/rel/ogc/1.0/tilesets-map**: The target IRI points to a resource that describes how to provide tile sets of the context resource in map format.
- **http://www.opengis.net/def/rel/ogc/1.0/tilesets-coverage**: The target IRI points to a resource that describes how to provide tile sets of the context resource in coverage format.
- **http://www.opengis.net/def/rel/ogc/1.0/tiling-schemes**: The target IRI points to a resource that lists one or more TileMatrixSets according to the 2D-TMS standard.

Used in combination with *OGC API — Features — Part 1: Core* or *OGC API — Common - Part 1: Core*, other link relation types will be used, including:

- **http://www.opengis.net/def/rel/ogc/1.0/conformance**: Refers to a resource that identifies the specifications that the link's context conforms to.

Used in combination with *OGC API — Features — Part 1: Core* or *OGC API — Common - Part 2: Geospatial Data*, other link relation types will be used, including:

- **http://www.opengis.net/def/rel/ogc/1.0/data**: Refers to the list of collections available for a dataset.

Each resource representation includes an array of links. Implementations are free to add additional links for all resources provided by the Web API.

5.3. Use of HTTPS

For simplicity, this Standard in general only refers to the HTTP protocol. This is not meant to exclude the use of HTTPS and simply is a shorthand notation for "HTTP or HTTPS." In fact, most servers are expected to use HTTPS, not HTTP.

6

OVERVIEW

6 OVERVIEW

6.1. Introduction

The *OGC API — Tiles* Standard defines building blocks that can be used in the implementation of Web API based servers and compatible clients to support the retrieval of tiled geospatial data that follow the structure defined in the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard (OGC 17-083r4) (see Clause 7), which is also known as the 2D TMS Standard. The *OGC API — Tiles* Standard assumes that the reader is familiar with the concepts in the OGC 17-083r4 Standard, such as a tile, tile set, tile set metadata, tile matrix and tile matrix set. If that is not the case, reading the OGC 17-083r4 overview subsection is recommended.

Services and clients are encouraged to support as many of the TileMatrixSets defined in Annex D and E of the OGC 17-083r4 Standard as possible. Other TileMatrixSets defined in the OGC TileMatrixSet register for all geospatial data resources to maximize interoperability should be considered. However, support for any specific TileMatrixSet is not required. Tiles that share the same TileMatrixSet definition can be easily visualized together in a data integration client.

The OGC API — Tiles standard is an alternative to the OGC Web Map Tile Service (WMTS) Standard [10]. The fundamental concept TileMatrixSet has not changed from the one used in WMTS. Therefore tiles provided via a WMTS instance and the ones generated by an *OGC API — Tiles* implementation instance that are based on the same TileMatrixSet can also be easily visualized and/or processed together. If the selected tile matrix set is the one called WebMercatorQuad (see OGC 17-083r4 Annex D.1), tiles are also compatible with the ones made available by some popular mass market approaches.

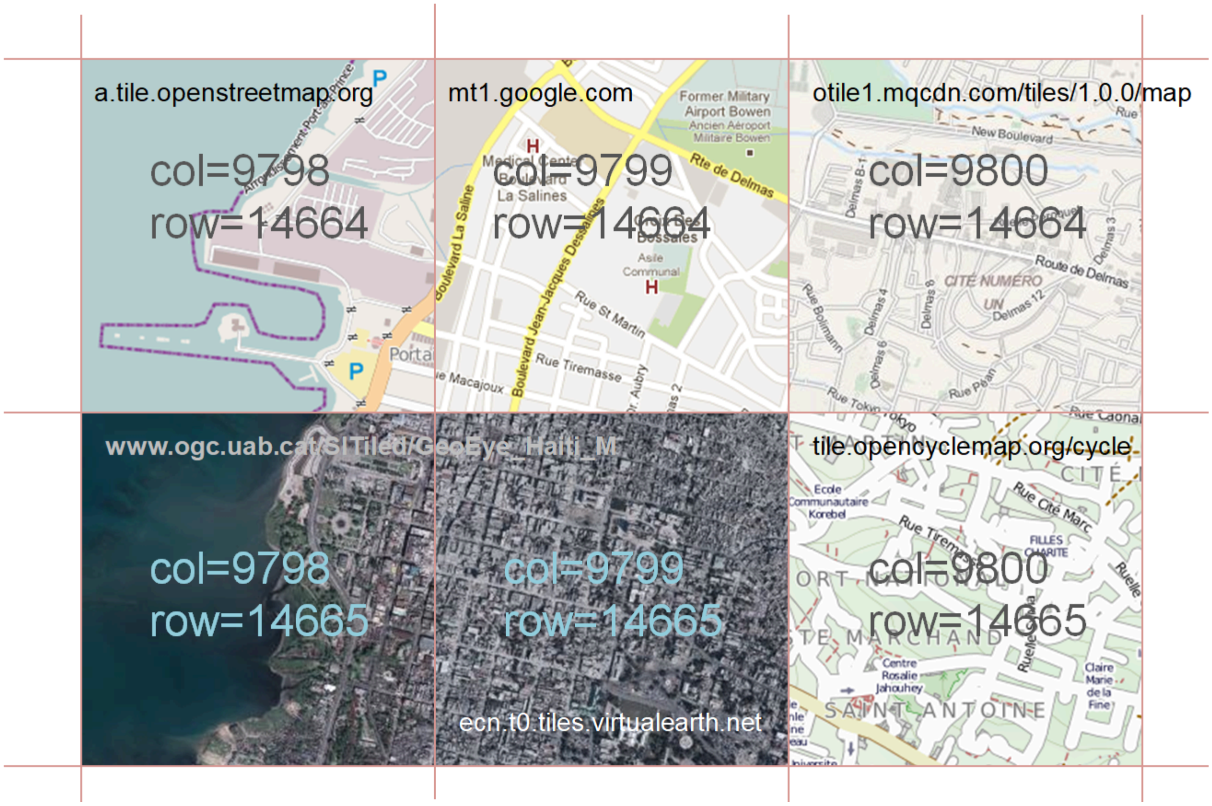

Figure 1 — Contiguous tiles from different services and APIs sharing same WebMercatorQuad tile matrix set and the tile matrix identified as 15 (sometimes called "zoom")

This Standard does not specify any requirement for the type of *geospatial data resource* that could be delivered as tiles. Provided that the geospatial data resources can be organized into tiles, any such resource can be supported regardless of whether they are maps, vector features, coverages, a resource that does not represent data per se (e.g., an annotation) and so forth.

NOTE: The geospatial data resources (e.g., collections) replace the concept of layer in the OGC Web Map Service (WMS) [11] and WMTS Standards. The concept of collections allows to provide multiple access mechanisms defined in complementary OGC API Standards, such as tiles and features, to the same collection of data (through different geospatial resource aspects). Geospatial data resources can advertise one or more lists of available tilesets (see Clause 8 and Clause 9). This Standard also defines how to link to tilesets originating from two specific data resources: OGC API datasets (see Clause 10) and collections (see Clause 11). However, other OGC APIs can provide other access and linking possibilities. Accessing the *geospatial data resource* content (other than as tiles) or its descriptions is possible but out of the scope of this Standard. If a description of the *geospatial data resource* is specified by another Standard, and this description has a mechanism to add links to other resources, this Standard indicates the need to add a link to the list of available tilesets.

The *OGC API — Tiles* standard does not specify how to get a Web API definition, the conformance class list, nor the collections lists. However, the Standard assumes that the first two are defined by an OGC API Standard (e.g., *OGC API — Common — Part 1: Core* or *OGC API —*

Features — Part 1: Core Standards) and the latter by an OGC API for collections (e.g., *OGC API — Common — Part 2: Geospatial data* or the *OGC API — Features — Part 1: Core* Standard).

6.2. Evolution from OGC Web Services

OGC Web Service (OWS) standards have historically implemented a Remote-Procedure-Call-over-HTTP architectural style using Extensible Markup Language (XML) for payloads. This was the state-of-the-art in the late 1990s and early 2000s when some of the initial versions of OGC Web Services Standards were designed and documented. Their service oriented architectural style now has a competing resource-oriented Web API style. The WMTS 1.0 standard defines a resource-oriented architectural style but lacks a Web API definition. The *OGC API — Tiles* Standard specifies a Web API that follows the architecture of the Web and in particular the W3C/OGC Best Practices for sharing Spatial Data on the Web [8] as well as the W3C Best Practices for sharing Data on the Web [9]. The bibliography lists several OGC Engineering Reports from initiatives that led to the development of OGC API — Tiles. Some of the Engineering Reports are from OGC Testbed initiatives [1] [16] , whereas others are from the OGC Vector Tiles Pilot [3] [4] [5] [6] [7].

This Standard defines the necessary elements to incorporate tile support into a Web API implementation. These elements can be incorporated in an API based on the *OGC API — Features — Part 1: Core* or can be incorporated in a Web API implementation based on the *OGC API — Common — Part 1: Core*. Both of those Standards specify a kernel of a Web API approach to services that follows current resource-oriented architecture practices in the OGC. OGC API — Common provides the foundation upon which implementations of the OGC API Standards can be built. OGC API — Common can be integrated with this Standard and other resource-specific OGC API Standards to build a Web API implementation. However, this Standard is also designed in a way that can extend OGC API — Common, but does not make OGC API — Common mandatory. As such this Standard can be reused as a building block in other APIs that do not follow the OGC API pattern.

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

- Clear separation between core requirements and more advanced capabilities. This *OGC API — Tiles — Part 1: Core* Standard defines the requirements that are relevant for anyone who wants to share or use Tiled Data at a fine-grained level. Additional capabilities that several communities are using today will be specified as extensions to this Standard.
- 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 Web API definition (such as a new version of the OpenAPI description document).
- Modularization is not just about a single "service". OGC APIs provide building blocks that can be reused in Web APIs in general. In other words, a server supporting OGC API — Tiles Requirements should not be seen as a standalone service. Rather, the web API implementation should be viewed as a collection of Web API building blocks which

together implement Tile capabilities. A corollary for this is that it should be possible to implement a Web API that concurrently conforms to conformance classes from the Features, Coverages, Maps, Tiles, and other future OGC API Standards.

The OGC APIs approach is intended to support two types of client developers:

- Those that have never heard about the OGC. Developers should be able to create a client using the Web API definition without the need to adopt a specific OGC approach (they no longer need to read how to implement a GetCapabilities response document, allowing them to focus on the geospatial resource aspects).
- Those that want to write a "generic" client that can access OGC APIs. In other words, they are not specific to a particular Web API.

As a result of following a RESTful approach, implementations of an OGC API are not backwards compatible with OWS implementations per se. However, a design goal is to define OGC APIs in a way that an OGC API interface can be mapped to or used as a façade to an existing OWS implementation (where appropriate). OGC APIs are intended to be simpler and more modern, but still an evolution from the previous versions and their implementations making the transition easy such as by initially implementing façades in front of the current OWS services.

6.3. Relationship to other OGC API standards

The OGC WMS and WMTS standards share the concept of a map and the capability to create and distribute maps at a limited resolution and size. In WMS, the number of rows and columns that a map should have can be selected by the user within limits. In WMTS the number of rows and columns of the tile is predefined in the tile matrix.

Over time, in the OGC, the concept of a tile, initially used for *map tiles* has been generalized to other data models such as feature data (some vendors use the expression *vector tiles*) and even to coverage data or processes that can be parallelized dividing space into tiles. The *OGC API — Tiles* Standard presents an approach to tiles that can be applied to almost any resource type that returns geospatial data. If implemented in conjunction with the *OGC API — Features* Standard and designed to access a feature collection, the expected result is tiled feature data. If implemented in conjunction with the *OGC API — Maps* candidate standard and designed to access a collection that is transformed into a map by applying a style, the result should be map tiles (usually in PNG or JPEG format).

The *OGC API — Tiles* Standard can be referenced by other OGC Standards that provide resources that can be offered as tiles. For example:

- The *OGC API — Maps* candidate standard specifies the link relation types to access map tilesets from a dataset or collection.
- The *OGC API — Styles* candidate standard defines paths to list available styles from which tilesets can also be accessed.

- The *OGC API — Coverages* candidate standard specifies the link relation types and specifics of retrieving coverage tiles.
- The *OGC API — Processes — Part 3: Workflows and Chaining* candidate standard provides a mechanism to trigger localized processing workflows as a result of retrieving tiles (for a specific area and resolution of interest).

This document is the first part of a series of *OGC API — Tiles* "parts" that use the core and extensions model. Future parts might specify other extensions, such as how to retrieve multiple tiles in a single request. Other standards or extensions of standards may also provide mechanisms (e.g., additional query parameters) to deal with additional dimensions such as elevation, or more advanced temporal capabilities than what is defined in this standard's *datetime* conformance class.

6.4. Using this standard independently

Although the *OGC API — Tiles* Standard is designed as a building block that can be leveraged by (or with) other OGC API Standards adding precisions about specific types of data available as tiles (e.g., *OGC API — Features* standard, and *OGC API — Maps* and *OGC API — Coverages* candidate standards), the conformance classes defined in this Standard are still concrete enough to make it possible to support distributing and requesting various types of tiled data, including coverages, vector features and maps, by relying strictly on the content herein and in the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 standard.

As informative guidance implementations should consider the following aspects.

6.4.1. Description of the domain

Three different mechanisms are defined by this Standard to describe the domain of the dimensions of the tiles, including spatiotemporal axes as well as additional dimensions.

In the Requirements Class "Tilesets List" (Clause 9), the collection description inherited from *OGC API — Common — Part 2* contains an `extent` property that can describe both the spatial and temporal domains of the data. In addition, the *Unified Additional Dimensions* common building block, used in the example OpenAPI definition, further specifies that additional dimensions shall be described in a similar way to the temporal dimension. An extra `grid` property in the example OpenAPI definition also allows specifying the resolution and the number of cells (for data organized as a regular grid) or a list of coordinates (for data organized as an irregular grid) along each dimension.

With the *TileSet* conformance class, the tileset metadata allows to specify a spatial bounding box for tiles as a whole, as well as for each individual collection of geospatial data represented or contained within the tiles (the *layers*). The resolution of these layers can also be specified by including the minimum and maximum cell size and equivalent scale denominators. The informative Annex J of the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 standard further extends this capability to describe the domain set by enhancing the schema to include bounds and resolution for additional dimensions as well being able to handle the specifics of unequal temporal units. Annex J also includes provisions to describe tile matrix sets featuring additional dimensions which not only extend in other dimensions but can also define divisions and down sampling of these additional dimensions for lower resolution tile matrices.

In addition to describing the bounds of the tileset dimensions, the Requirements Class "TileSet" (Clause 8) also supports specifying limits in terms of identifiers for the minimum and maximum tile matrices, tile rows, and tile columns for which data is available.

6.4.2. Description of the observed or measured properties

The Requirements Class "TileSet" (Clause 8), supports specifying the tileset metadata for the measured or observed properties for each collection of geospatial data represented or contained within the tiles (the *layers*). For each of these properties, a JSON schema and semantic information can be described. This schema can be used to describe properties for feature collections or the range type of coverages.

6.4.3. Available formats and tile response expectations

The Tiles API Standard, in Requirements classes for tile encodings (Clause 16), defines six requirements classes for specific encodings for different types of tiled data. Additional encodings can be supported using HTTP content negotiation, following conventions specific to those encodings. In this case requirements are expected to fall back to the closest encoding defined in Requirements classes for tile encodings (Clause 16) (e.g., using the GeoTIFF and netCDF conformance class as a model for other coverage data, the JPEG and PNG classes for other map tiles encodings, and the Mapbox Vector Tiles or GeoJSON for other vector tiles encodings). The informative Annex J of the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard also describes a mechanism that can be used to deliver and access 3D content using this standard, including 3D models either batched as a single mesh, or as points vector tiles referencing shared 3D models.

6.4.4. Limitations

Although implementations of the *OGC API — Tiles* Standard can be used "stand-alone", other OGC API Standards or draft specifications may provide additional capabilities and specify additional normative requirements describing how to retrieve specific types of tiled content. This includes describing in greater detail the domain, or the observed or measured properties within the tiled data. Conforming to these standards as well may enable greater interoperability. For example, for map tiles, this Standard does not define how a client requests a specific background color or whether tiles should be opaque or transparent expecting that the *OGC API — Maps* will do so.

6.5. How to approach an implementation of an OGC API Standard

There are at least two ways to approach an implementation of an OGC API Standard.

- Read the landing page, look for links, follow them and discover new links until the desired resource is found
- Read a Web API definition document that specifies a list of paths and path templates to resources.

For the first approach, many resources in the *OGC Web APIs* include links with *rel* properties to know the reason and purpose for this relation. The following figure illustrates the resources as ellipses and the links as arrows with the link *rel* as a label.

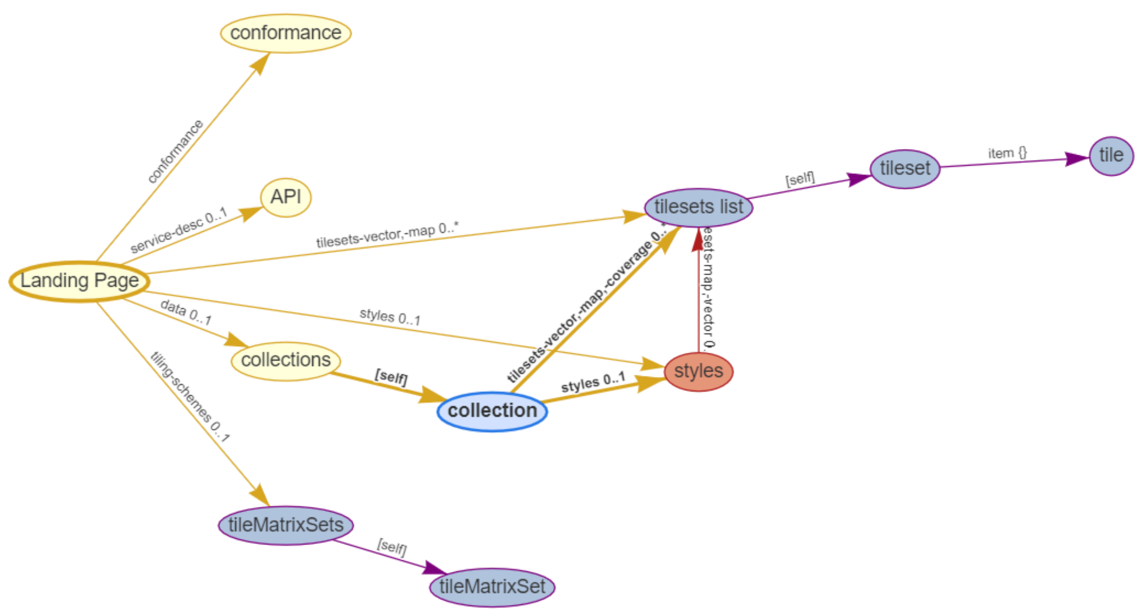

Figure 2 — Resources and relations to them via links

For the second approach, implementations should consider Requirements Class “OpenAPI Specification 3.0” (Clause 14) which specifies the use of *operationID* suffixes, providing a mechanism to associate API paths with the requirements class that they implement.

There is yet a third way to approach an implementation of an OGC API Standard that relies on assuming a set of predefined paths and path templates. These predefined paths are used in many examples in this Standard and are presented together in Table 8. Many implementations of this Standard will provide a Web API definition document (e.g. OpenAPI) using this set of predefined paths and path templates to get necessary resources directly. All this could mislead the reader into getting the false impression that the predefined paths are enforced. They are not. Therefore, building a client that is assuming a predefined set of paths is risky. Even so, many API implementations will actually follow the predefined set of paths and the client using this approach could be successful on many occasions. Again, be aware that these paths are not required by this Standard.

Table 8 — Overview of resources and common direct links that can be used to define an OGC API — Tiles implementation

RESOURCE NAME	COMMON PATH
Landing page[4]	`{datasetRoot}/`
Conformance declaration[4]	`{datasetRoot}/conformance`
Tiling Schemes[6]	`{datasetRoot}/tileMatrixSets`
Tiling Scheme[6] (tile matrix set[2])	`{datasetRoot}/tileMatrixSets/{tileMatrixSetId}`
Dataset Tiles	
Dataset Feature Tiles[3]	
Dataset tileset list[1,2]	`{datasetRoot}/tiles`
Dataset tileset metadata[1,2] (in one tile matrix set[2])	`{datasetRoot}/tiles/{tileMatrixSetId}`
Dataset feature tile[1,3]	`{datasetRoot}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`
Dataset Map tiles	
Map tileset list[2] (geospatial resources[1])	`{datasetRoot}/map/tiles`
Map tileset metadata[2] (geospatial resources[1])	`{datasetRoot}/map/tiles/{tileMatrixSetId}`
Map tile[1]	`{datasetRoot}/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`
Geospatial data collections[5]	
Collections[5]	`{datasetRoot}/collections`
Collection[5]	`{datasetRoot}/collections/{collectionId}`

RESOURCE NAME	COMMON PATH
Collection Feature Tiles[3]	
Feature tileset list[2]	`{datasetRoot}/collections/{collectionId}/tiles`
Feature tileset metadata[2]	`{datasetRoot}/collections/{collectionId}/tiles/{tileMatrixSetId}`
Feature tile[3]	`{datasetRoot}/collections/{collectionId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`
Collection Map tiles	
Map tileset list[2]	`{datasetRoot}/collections/{collectionId}/map/tiles`
Map tileset metadata[2]	`{datasetRoot}/collections/{collectionId}/map/tiles/{tileMatrixSetId}`
Map tile	`{datasetRoot}/collections/{collectionId}/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`
Coverage tiles	
Coverage tileset list[2]	`{datasetRoot}/collections/{collectionId}/coverage/tiles`
Coverage tileset metadata[2]	`{datasetRoot}/collections/{collectionId}/coverage/tiles/{tileMatrixSetId}`
Coverage tile	`{datasetRoot}/collections/{collectionId}/coverage/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`

[1] From the whole dataset or one or more geospatial resources or collections [2] Specified in the *OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0* standard [3] Some vendors use the expression *vector tiles* [4] Specified in the *OGC API — Common — Part 1: Core* standard [5] Specified in the *OGC API — Common — Part 2: Geospatial Data* candidate standard [6] Recommended but not required by the core of this standard

NOTE 1: Despite the fact that full path and full path templates in the previous table are used in many implementations of the *OGC API — Tiles* Standard, these exact paths are ONLY examples and are NOT required by this Standard. Other paths are possible if correctly described in by the Web API definition document and/or the links between resources. However, the *TileSets list* conformance class does require that paths listing tilesets end with …`/tiles`.

NOTE 2: The use of a `{tileMatrixSetId}` URI template variable is not required by this Standard. However, the *TileMatrixSet definition* permission proposes to make all tileset paths homogeneous by using it. A `{tileMatrixSetId}` template variable must NOT be used in templated links of the tileset metadata as defined in Requirements Class “TileSet” (Clause 8).

6.6. Why we call them “tiles”

The word *tile* is traditionally used to refer to a thin, flat or convex slab of hard material such as baked clay or plastic, laid in rows to cover walls, floors, and roofs. In this Standard, the same approach is used to cover the viewport of a computer screen with tiles representing parts of the world (geospatial features). Actually some examples of traditional tilesets representing geospatial features can also be found. They are tilesets with only one available tilematrix.

Figure 3 — Tiles in the floor of the monument of discovery in Lisbon, Portugal. (Lee Cannon April 2010, CC-BY-SA, https://www.flickr.com/photos/leecannon/5127274297)

Figure 4 — Tiles in the floor of the terminal 2 of the Prague Airport, Czech Republic. (Joan Masó, September 2022, CC0)

7

REQUIREMENTS CLASS "CORE"

7 REQUIREMENTS CLASS “CORE”

REQUIREMENTS CLASS 1: REQUIREMENTS CLASS CORE	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`
TARGET TYPE	Web API
PREREQUISITES	RFC 2616 (HTTP/1.1) RFC 2818 (HTTP over TLS) (optional) http://www.opengis.net/spec/tms/2.0/req/tilematrixset

The Core Requirements Class is generically designed to describe an HTTP GET operation, as well as its response, to retrieve tiles from a tileset which can be described by the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 standard. The Core introduces the idea of URI templates and variables associated with the TileMatrix, TileRow, and TileCol concepts defined in the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard but it does not prescribe a particular path or template form for the URL of those tiles.

The Core focuses on the individual tiles and does not describe the tileset as a whole. The implementer of a Web API should also implement support for tileset metadata in conjunction with the core requirements class as it provides important information to clients. This can be done by also implementing the Requirements Class “Tileset” (Clause 8) or integrating within another API providing this capability through another mechanism.

7.1. A tile

A tile resource is a geospatial resource presenting a fragment of a much bigger geospatial data resource that is spatially constrained at the boundaries of the selected tile in a tile matrix set.

7.1.1. Operation

An HTTP GET request allows for the retrieval of a single tile that represents information coming from geospatial data resources.

REQUIREMENT 1	
IDENTIFII	`/req/core/tc-op`

REQUIREMENT 1	
A	The tiles making up a tileset containing available data SHALL be available via an HTTP GET request to a URI that can be built from a template containing two or three variables. If the tileset is available for more than one tilematrix, the template SHALL contain three variables (such as `{tileMatrix}`, `{tileRow}` and `{tileCol}` as defined by the tileset conformance class, or `{z}`, `{y}` and `{x}`). If the tileset is available for a single tilematrix, the template SHALL contain two variables (such as `{tileRow}` and `{tileCol}` as defined by the tileset conformance class, or `{y}` and `{x}`). The URI is obtained by substituting the variables by their respective valid values.
B	These variables SHALL correspond to the tile matrix, tile row and tile column of a particular tile matrix set as defined by the 2D Tile Matrix Set standard.
C	The API SHALL provide a mechanism to obtain this template and associate the variables to their respective meaning, for example by implementing the tileset conformance class, or through an API definition.

NOTE 1: The *core* Requirements Class by itself does not prescribe specific names for these variables, nor does it define a specific mechanism to communicate the template.

Typical geospatial data resources that can be retrieved as tiles are: vector features (available at `/collections/{collectionId}/items` in OGC API — Features — Part 1: Core, for which it is recommended to provide tilesets at `/collections/{collectionId}/tiles`), maps (specified in OGC API — Maps) or coverages (specified in OGC API — Coverages).

RECOMMENDATION 1	
IDENTIFII	`/rec/core/tc-op`
A	A tiles implementation SHOULD consider using the tiles URI template variables in the following common order and form: `{tileMatrix}/{tileRow}/{tileCol}`
B	A tiles implementation SHOULD consider specifying the variables `{tileMatrix}` even if there is only one valid value for them.

NOTE 2: Clients should not assume the common order of URI template variable and should extract them from the examples in the “links” section or from the API description path templates.

The desired encoding is selected by the client using HTTP content negotiation. If necessary, the client can determine the supported encodings, or more precisely the media types of the supported encodings, from the Web API definition, or from the conformance declaration (see Clause 16).

The core of the OGC API — Tiles Standard provides a mechanism to select and retrieve a tile in a TileMatrixSet that is agreed between the client and server. This Core Requirements Class does not discuss the details of how the client and service agree on which TileMatrixSet they use.

7.1.2. Parameter tileMatrix

REQUIREMENT 2	
IDENTIFII	/req/core/tc-tilematrix-definition
A	If the API implements *OGC API — Common — Part 1: Core*, the definition of this operation SHALL support a parameter `tileMatrix` with the following characteristics (shown as OpenAPI Specification 3.0 fragment):

```
  name: tileMatrix
  in: path
  description: Identifier selecting one of the scales defined in the
TileMatrixSet and representing the scaleDenominator the tile.
  required: true
  schema:
    type: string
  example: '11'
```

7.1.3. Parameter tileRow

REQUIREMENT 3	
IDENTIFII	/req/core/tc-tilerow-definition
A	If the API implements *OGC API — Common — Part 1: Core*, the definition of this operation SHALL support a parameter `tileRow` with the following characteristics (shown as OpenAPI Specification 3.0 fragment):

```
  name: tileRow
  in: path
  description: Row index of the tile on the selected TileMatrix. It cannot
exceed the MatrixWidth-1 for the selected TileMatrix
  required: true
  schema:
    type: integer
    minimum: 0
  example: '827'
```

7.1.4. Parameter tileCol

REQUIREMENT 4	
IDENTIFII	/req/core/tc-tilecol-definition
A	If the API implements *OGC API — Common — Part 1: Core*, the definition of this operation SHALL support a parameter `tileCol` with the following characteristics (shown as OpenAPI Specification 3.0 fragment):

REQUIREMENT 4

```
  name: tileCol
  in: path
  description: Column index of the tile on the selected TileMatrix. It cannot
exceed the MatrixHeight-1 for the selected TileMatrix.
  required: true
  schema:
    type: integer
    minimum: 0
  example: 1231
```

7.1.5. Parameter tileMatrixSetId (optional)

PERMISSION 1

IDENTIFII	/per/core/tc-tilematrixset-definition
A	An extra {tileMatrixSetId} variable MAY be used by the API definition for simplification purposes and, this way, make all tileset paths homogeneous.
B	The {tileMatrixSetId} variable will be interpreted as one of TileMatrixSet identifiers supported by the resource. In other words, tileMatrixSetId represents all the TileMatrixSets supported by the resource.

C	The following OpenAPI Specification 3.0 fragment MAY be used in this case:
	`name: tileMatrixSetId`
	`in: path`
	`description: Identifier selecting one of the TileMatrixSetId supported by the resource.`
	`required: true`
	`schema:`
	`  type: string`
	`example: 'WebMercatorQuad'`

7.1.6. Response

A successful response to a tile GET operation will be consistent with the media type of the resource requested. This Standard does not impose any media type or file format. Examples of common media types are:

- For features, the media type of the response may be GeoJSON (application/geo+json) or Mapbox Vector Tiles (application/vnd.mapbox-vector-tile);
- For coverages, the media type of the response may be GeoTIFF (image/tiff) or netCDF (application/netcdf or application/x-netcdf);
- For maps, the media type of the response may be JPEG (image/jpeg) or a PNG (image/png).

REQUIREMENT 5	
IDENTIFII	`/req/core/tc-success`
A	A successful execution of the tile operation with content SHALL be reported as a response with an HTTP status code `200`.
B	The content of that response SHALL be consistent with the format requested via HTTP content negotiation and represent elements inside or intersecting with the spatial extent of the geographical area of the tile identified by the tile matrix, tile row, and tile column of the tileset's tile matrix set.
C	For image tiles, where this behavior is not explicitly overridden by an extension (e.g., a query parameter, or format-specific requirements), the width and height of the image (measured in raster pixels) SHALL be equal to the tileWidth and tileHeight of the corresponding tile matrix.
D	For gridded coverage tiles, where this behavior is not explicitly overridden by an extension (e.g., a query parameter, or format-specific requirements), for coverages whose cells span the whole area of the resolution, the width and height of the coverage tile (measured in cells) SHALL be the tileWidth and tileHeight of the corresponding tile matrix. In addition, for coverages whose cells are measurements or observations for a conceptually infinitely small point the width and height of the coverage tile SHALL be tileWidth + 1 and tileHeight + 1.

PERMISSION 2	
IDENTIFII	`/per/core/tc-core-tile-encoding`
A	This Standard does not impose any media type on the encoding of a response containing tiled feature data. For features, the media type MAY be GeoJSON, Mapbox Vector Tiles or some other format.
B	This Standard does not impose any media type on the encoding of a response containing tiled coverage data. For coverages, the media type MAY be a GeoTIFF, netCDF or some other format.
C	This Standard does not impose any media type on the encoding of a map tile response. For maps, the media type MAY be JPEG, PNG or some other format.

RECOMMENDATION 2	
IDENTIFII	`/rec/core/tc-multiple-media-types`
A	If both "image/png" and "image/jpeg" are supported at an endpoint and are present in the HTTP "Accept" request header with the same q value, and no other supported output format is present with a higher q value, the server SHOULD return the requested image in whichever of these two formats is considered optimal (typically PNG if there's transparency or for categorical maps where a limited number of colors is used, and JPEG otherwise).

Normally, the content partially outside the tile bounding box will be clipped at the extent of the bounding box. This can be done efficiently when tiles are in raster format (e.g., map tiles). However, tiles containing features in vector format may not clip features that are partially outside, or clip to a slightly enlarged bounding box (a buffer), to ensure continuity of features or for performance.

RECOMMENDATION 3	
IDENTIFII	`/rec/core/tc-success-scale`
A	The content in response of a tile request SHOULD be simplified to comply with the scale denominator represented by the TileMatrix identified. Full resolution geographical elements are only expected for the lower values of scale denominators.

RECOMMENDATION 4	
IDENTIFII	`/rec/core/tc-deepfullempty`
A	If a requested tile is empty (no data) and all tiles within its extent at all more detailed zoom levels (tile matrices) are guaranteed to also be empty, the response header SHOULD include `OATiles-hint: empty`.
B	If a requested tile is totally inside of a polygon or other situations that may result in a uniform content such as a solid color for a map, or a grid completely filled with the same value for a coverage and all tiles within its extent at all more detailed zoom levels (tile matrices) are guaranteed to also be in the same situation, the response header SHOULD include `OATiles-hint: full`.

NOTE: A client could use this information to avoid requesting the more detailed tiles of that area.

To enable search engines to easily discover the content offered by an implementation of OGC API — Tiles, as well as to enable web browsers to easily display the content offered by the Web APIs, this Standard allows for responses to operations to be encoded in HTML.

PERMISSION 3	
IDENTIFIEI	`/per/core/tc-core-html`
A	Every `200`-response of an operation of the server MAY support the media type `text/html`.

7.1.7. Error conditions

A general summary of the HTTP status codes can be found in OGC API — Features — Part 1: Core, version 1.0 as well as in OGC API — Common — Part 1: Core.

REQUIREMENT 6

IDENTIFII	`/req/core/tc-error`
A	If the path parameter values `tileMatrix`, `tileRow`, `tileCol` for a tile request are out-of-range (outside the tile matrix set or tile matrix set limits of the resource), the HTTP response SHALL use a status code `404` or a `400`. The response can also contain an exception report.
B	If the tile has no content due to lack of data in the area, but is within the data resource's tile matrix sets and tile matrix sets limits, the HTTP response will use the status code either 204 (indicating an empty tile with no content) or a 200 with the content of a blank response compatible with the requested media type (which may or may not be zero bytes long, depending on the output format).

7.2. Declaration of conformance classes

To support "generic" clients wishing to access multiple implementations of OGC API Standards and extensions — and not "just" a specific Web API / server, the Web API must declare the conformance classes it implements and conforms to.

7.2.1. Response

The conformance declaration mainly consists of a list of links.

REQUIREMENT 7

IDENTIFII	`/req/core/conformance-success`
A	If the API instance has a mechanism to advertise conformance classes, the list of conformance classes SHALL include the ones defined in this standard and listed in Table 7 that are supported by this API instance.

If the server also declares conformity to OGC API — Common — Part 1: Core or to OGC API — Features — Part 1: Core, version 1.0, then it has to consider the OGC API — Common requirements for declaring conformance, i.e. the use of a conformance declaration page. In the JSON format the conformance declaration page is an array of links following the link schema defined in the OGC API — Common or in OGC API — Features — Part 1: Core, version 1.0 standards. Below is an example fragment of a conformance declaration page conformant to OGC API — Common and OGC API — Tiles.

Example — Conformance Information Page fragment

```
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/core"
  ]
```

```
}
```

8

REQUIREMENTS CLASS "TILESET"

8 REQUIREMENTS CLASS “TILESET”

8.1. Overview

REQUIREMENTS CLASS 2: REQUIREMENTS CLASS TILESET	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tileset`
TARGET TYPE	Web API
PREREQUISITES	RFC 8288 (Web Linking) http://www.opengis.net/spec/tms/2.0/req/json-tilesetmetadata Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

The *tileset* Requirements Class provides a mechanism to retrieve metadata for a set of tiles of geospatial data tiled according to one specific TileMatrixSet. This Class also provides a mechanism to obtain a templated link to retrieve individual tiles as defined in the *core* conformance class.

This Class describes the HTTP GET operation for accessing a tileset resource and its response but does not prescribe a specific path. The response provides metadata as per the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard.

The essential elements of this metadata are:

- A link to the definition of the TileMatrixSet (either one registered in the OGC TileMatrixSet register that is controlled by the OGC Naming Authority (OGC-NA) SubCommittee (SC), or to a custom TileMatrixSet definition).
- A TileMatrixSet URI in the case of a TileMatrixSet registered with an authority (e.g. the OGC TileMatrixSet register).
- A Coordinate Reference System (usually provided as a URI).
- A templated link (URI) to individual tiles.
- A data type (indicating whether the tileset consists of vector, coverage or map tiles).

Metadata may optionally also provide additional information, such as:

- A title.

- A description.
- The limits of the tileset if it does not span the full extent of the TileMatrixSet.
- The geospatial data resources involved in the creation of the tiles (potentially including links to OGC API collections).
- A schema of the available properties contained within the tiles.
- Styles used to create the tiles.
- A central point on which a viewer may initially focus.
- Attribution.

A link to a definition of a TileMatrixSet is always required whether a custom TileMatrixSet or a registered TileMatrixSet is used. Having the Web API host a local definition of each supported TileMatrixSet to ensure availability is recommended.

8.2. Tileset resource

A tileset consists of a set of tiles obtained by partitioning geospatial data according to a particular TileMatrixSet. The tileset metadata contains all the information necessary for a client application to request tiles from the tileset.

8.2.1. Tileset path

This class does not specify a full path to a tileset. Generally, *tileset* resources are linked from a *tilesets list* resource (refer to the Requirements Class “Tilesets List” (Clause 9) for how to list available tilesets and link to individual tileset resources). Refer to the Requirements Class “Dataset Tilesets” (Clause 10) and Requirements Class “GeoData Tilesets” (Clause 11) for a description of two mechanisms that associate lists of tilesets to an OGC API landing page for a dataset and to geospatial data (collection) resources respectively. The expectation is that a tileset can be used as a building block in other Web APIs and can be used to provide tiles of different types of data such as maps, features, coverages, or other types of geospatial data that can benefit from tiling.

8.2.2. Response

A successful GET response to a tileset resource is metadata consisting of a data structure with the specific information necessary to build a complete GET request of the tiles representing the geospatial data resource.

REQUIREMENT 8

IDENTIFII	/req/tileset/description
A	The tileset endpoint SHALL support negotiation of an application/json response. In this case, a successful response of an HTTP GET for a specific tileset SHALL be encoded following the data model and JSON schema for tileset metadata, as defined by the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata Standard 2.0.
B	If the tileset endpoint also supports negotiation of an application/xml response, a successful response of an HTTP GET for a specific tileset SHALL be encoded following the data model and XML schema for tileset metadata, as defined by the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata Standard 2.0.
C	If the tileset uses a TileMatrixSet registered in a TileMatrixSet register (e.g. that of the OGC), the `tileMatrixSetURI` property SHALL link to the registered TileMatrixSet (e.g. `http://www.opengis.net/def/tilematrixset/{tileMatrixSet}`).
D	The links property SHALL include a link to the TileMatrixSet definition with relation type `http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme` following the tile matrix set schema, as defined by the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata Standard 2.0.
E	The tileset metadata SHALL include at least one templated link to individual tiles using the relation type `item`, and the template parameters `{tileMatrix}`, and `{tileRow}` and `{tileCol}`. Those variables are to be substituted by their respective valid values to obtain the URL to a tile.
F	If a tiles link template is specific to a particular format, the link SHALL contain the media type for that format in the "`type`" property. Otherwise, normal HTTP content type negotiation rules apply (`Accept:` header).
G	A property `templated` with a boolean `true` value SHALL be part of the link properties to indicate that the link needs to be processed to substitute the templated variables with valid values before being used as a URL to a tile.

In addition to the recommendation to include links in the HTTP response headers as specified in OGC API — Common — Part 1, this Standard recommends following the Link-Template HTTP Header Field draft specification, to include `Link-Template` for templated links, such as the URI for individual tiles.

RECOMMENDATION 5

IDENTIFII	/rec/tileset/header-linktemplates
A	The tileset metadata response SHOULD include a `Link-Template:` in the header following the Link-Template HTTP Header Field draft specification, for example, `Link-Template: </ogcapi/collections/blueMarble/map/tiles/GNOSISGlobalGrid/{tileMatrix}/{tileRow}/{tileCol}.png>; rel="item"; type="image/png"; var-base="/ogcapi/vars/"`.
B	The templated link in the tileset metadata SHOULD include a `varBase`, if the implementation additionally supports providing semantic information about the parameters.

See Clause 15 for providing an XML representation of the tileset metadata.

Support for alternative encodings for tileset metadata can be added, such as TileJSON.

Currently, use of the TileJSON specification usually implies a WebMercatorQuad TileMatrixSet and the reference to it is implicit. TileJSON version 3 provides an additional mechanism to cite data sources.

RECOMMENDATION 6	
IDENTIFII	`/rec/tileset/tmxslink`
A	To improve interoperability, this Standard recommends that the API provides tiles following the TileMatrix Sets defined in a TileMatrixSet register.
B	Using a TileMatrixSet not available in a register is also possible. In this case the API SHOULD provide a definition of the TileMatrixSet through a resolvable URI at an endpoint following the template: `/tileMatrixSets/{tileMatrixSetId}` (where `{tileMatrixSetId}` is a unique identifier of the TileMatrixSet in the service) following the TileMatrixSet schema in http://docs.ogc.org/is/17-083r4/17-083r4
C	When the API provides one or more definitions of TileMatrixSets through a resolvable URI at `/tileMatrixSets/{tileMatrixSetId}`, the `/tileMatrixSets` endpoint SHOULD respond with the list of TileMatrixSets defined by the API using an object with a `tileMatrixSets` property that contains an array of `TileMatrixSet` objects.
D	When the API provides a `/tileMatrixSets` endpoint, this endpoint SHOULD be mentioned in one or more links in the landing page of the API with the `rel` type `http://www.opengis.net/def/rel/ogc/1.0/tiling-schemes`.

NOTE 1: The common TileMatrixSets defined in Annex D of OGC 17-083r4 and the variable-width TileMatrixSets defined in Annex E of OGC 17-083r4 are registered in the OGC TileMatrixSets register available from the OGC Definitions Server http://www.opengis.net/def.

RECOMMENDATION 7	
IDENTIFIE	`/rec/tileset/bbox`
A	The tileset metadata SHOULD populate the bounding box element to describe the tileset extent.

RECOMMENDATION 8	
IDENTIFII	`/rec/tileset/conf-link`
A	Include a link with 'rel' type 'http://www.opengis.net/def/rel/ogc/1.0/conformance' to link to the conformance declaration page

Clients or servers are not required to support a specific default TileMatrixSet.

NOTE 2: The OGC TileMatrixSets register is based on the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata standard, version 2.0 (OGC 17-083r4). Refer to Annex D and E for commonly used TileMatrixSets.

NOTE 3: When this Requirements Class is used together with the GeoData Tilesets Requirements Class, the geospatial data resource URL is typically the first part of the URL template, but this standard does not mandate this (e.g., the tileset metadata and tiles could be hosted on more affordable object storage).

Example 1 — Example fragment of a tileset response for a common TileMatrixSet defined in the OGC TileMatrixSet register

```
{
  ...
  "tileMatrixSetURI": "http://www.opengis.net/def/tilematrixset/OGC/1.0/
WorldMercatorWGS84Quad",
  "dataType": "map",
  "crs": "http://www.opengis.net/def/crs/EPSG/0/3395",
  "links": [
    ...
    {
     "href": "http://data.example.com/collections/buildings/tiles/WorldMercator
WGS84Quad",
     "rel": "self",
     "type": "application/json",
     "title": "Buildings tileset tiled using World Mercator TileMatrixSet"
    },
    {
     "href": "http://schemas.opengis.net/tms/2.0/json/examples/WorldMercatorWGS
84Quad.json",
     "rel": "http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme",
     "type": "application/json",
     "title": "Definition of WorldMercatorQuad TileMatrixSet"
    },
    {
     "href": "http://data.example.com/collections/buildings/tiles/WorldMercator
WGS84Quad/{tileMatrix}/{tileRow}/{tileCol}.png",
     "templated": true,
     "rel": "item",
     "type": "image/png",
     "title": "Templated link for retrieving PNG tiles"
    }
   ...
  ],
  "tileMatrixSetLimits":
  [
    { "tileMatrix" : "0", "minTileRow" : 0, "maxTileRow" : 0, "minTileCol" : 0,
 "maxTileCol" : 0 },
    { "tileMatrix" : "1", "minTileRow" : 0, "maxTileRow" : 0, "minTileCol" : 1,
 "maxTileCol" : 1 },
    { "tileMatrix" : "2", "minTileRow" : 1, "maxTileRow" : 1, "minTileCol" : 2,
 "maxTileCol" : 2 },
    { "tileMatrix" : "3", "minTileRow" : 3, "maxTileRow" : 3, "minTileCol" : 4,
 "maxTileCol" : 4 },
    ...
  ]
}
```

NOTE 4: The use of "templated" is inspired by the JSON Hypertext Application Language (HAL), https://tools.ietf.org/html/draft-kelly-json-hal-08.

The following table explains the meaning of the URI template variables.

Table 9 — URI template variables for tiles and valid values

URL TEMPLATE VARIABLE	MEANING	POSSIBLE VALUES
TileMatrix	tile matrix identifier	Identifier of the tile matrix (representing a zoom level, a.k.a. a scale) listed in the TileMatrixSet definition
TileRow	row index of tile matrix	A non-negative integer between 0 and the MatrixHeight — 1. If there is a TileMatrixSetLimits the value is limited between MinTileRow and MaxTileRow
TileCol	column index of tile matrix	A non-negative integer between 0 and the MatrixWidth — 1. If there is a TileMatrixSetLimits the value is limited between MinTileCol and MaxTileCol

Example 2 — Example fragment of a tileset response for a Web API-defined TileMatrixSet

```
{
  ...
  "dataType": "map",
  "crs": "http://www.opengis.net/def/crs/EPSG/0/2001",
  "links": [
    ...
    {
     "href": "http://data.example.com/collections/buildings/tiles/CustomAntigua
TMS",
     "rel": "self",
     "type": "application/json",
     "title": "Buildings tileset tiled using custom Antigua TileMatrixSet"
    },
    {
     "href": "http://data.example.com/collections/buildings/tiles/CustomAntigua
TMS/{tileMatrix}/{tileRow}/{tileCol}.png",
     "templated": true,
     "rel": "item",
     "type": "image/png",
     "title": "Templated link for retrieving PNG tiles"
    },
    {
     "href": "http://data.example.com/tileMatrixSets/CustomAntiguaTMS",
     "rel": "http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme",
     "type": "application/json",
     "title": "Definition of custom Antigua TileMatrixSet"
    }
   ...
  ],
  "tileMatrixSetLimits":
  [
    ...
  ]
}
```

8.3. Web API-defined TileMatrixSets

This section provides more details on the recommendations for implementing TileMatrixSets in Web API based servers and client applications. In the future, an extension of this Standard may describe a mechanism to manage (create, update, ...) tiling schemes for a Web application implementing OGC API definitions including OGC API – Tiles Requirements Classes and make some of the recommendations mandatory.

8.3.1. Web API Landing Page

An OGC API landing page provides links to start exploring the resources offered by the Web API. A landing page mainly consists of a list of links. New links for TileMatrixSets on top of the ones defined in *OGC API — Common* are introduced.

Using the JSON format, the landing page links follow the link schema defined in *OGC API — Common*. The following is an example fragment of the response to an OGC API — Tiles landing page.

Example — Web API Landing Page fragment with links to TileMatrixSet descriptions

```
{
  "links": [
    ...
    {
      "href": "http://data.example.org/tileMatrixSets?f=json",
      "rel": "http://www.opengis.net/def/rel/ogc/1.0/tiling-schemes",
      "type": "application/json",
      "title": "List of tileMatrixSets implemented by this API in JSON"
    },
    {
      "href": "http://data.example.org/tileMatrixSets?f=html",
      "rel": "http://www.opengis.net/def/rel/ogc/1.0/tiling-schemes",
      "type": "text/html",
      "title": "List of tileMatrixSets implemented by this API in HTML"
    }
  ]
}
```

8.3.2. TileMatrixSets

An HTTP request to the TileMatrixSets endpoint retrieves a list of links to the descriptions of the tile matrix sets supported by the OGC Web API. They may be the TileMatrixSets defined in Annex D and Annex E of the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard, or in customized definitions. The response follows the schema below.

Example 1 — Schema for the TileMatrixSets resource

```
  type: object
  required:
    - tileMatrixSets
  properties:
```

```
    tileMatrixSets:
      type: array
      items:
        $ref: '#/components/schemas/id-link'
```

Example 2 — Schema for id-link used to validate TileMatrixSets resource

```
id-link:
  type: object
  description: |-
    Potentially reusable object that contains an id to a resource and links
where the object is described or a representation retrieved. Typically it is
useful for paths like `/resources` and `/resources/{resourceId}`. `/resources`
responds to an array of id-link listing the `resourceId` and the links to get
it. /collections and /collections/{collectionId} in OGC API - Common is an
exception to this pattern.
    The fact that `links` is an array can be used to advertise the same object
representation in different formats.
  required:
    - id
    - links
  properties:
    id:
      type: string
    uri:
      type: string
      format: uri
      description: If the Tile Matrix Set is registered in an authoritative
definitions server, this property should the reference to this definition
    title:
      type: string
    links:
      type: array
      minItems: 1
      items:
        $ref: '#/components/schemas/link'
```

Example 3 — Example for the TileMatrixSets resource

```
{
  "tileMatrixSets": [
    {
      "id": "MyWebMercatorQuad",
      "uri": "http://www.opengis.net/def/tilematrixset/OGC/1.0/WebMercatorQuad"
      "title": "My Google Maps Compatible for the World",
      "links": [
        {
          "rel": "self",
          "href": "https://data.example.org/tileMatrixSets/MyWebMercatorQuad",
          "type": "application/json",
          "title": "Local definition of WebMercatorQuad TileMatrixSet"
        }
      ]
    }
  ]
}
```

8.3.3. TileMatrixSet

An HTTP GET request to a TileMatrixSet endpoint retrieves the full description of a tile matrix set supported by Web API based servers and client applications following the schema described in the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 Standard. The response follows the TileMatrixSet schema.

Example — Fragment of a TileMatrixSet resource example

```
{
  "title": "My TileMatrixSet for the World",
  "id": "MyTMS",
  "uri": "http://www.opengis.net/def/tilematrixset/OGC/1.0/WebMercatorQuad",
  "crs": "http://www.opengis.net/def/crs/EPSG/0/3857",
  "wellKnownScaleSet": "http://www.opengis.net/def/wkss/OGC/1.0/
GoogleMapsCompatible",
  "tileMatrices": [
    ...
    {
      "title": "My zoom level 3",
      "id": "3",
      "scaleDenominator": 69885283.0035897,
      "cellSize": 19567.8792410051,
      "pointOfOrigin": [-20037508.3427892,20037508.3427892],
      "tileWidth": 256,
      "tileHeight": 256,
      "matrixWidth": 8,
      "matrixHeight": 8
    }
    ...
  ]
}
```

9

REQUIREMENTS CLASS "TILESETS LIST"

9 REQUIREMENTS CLASS “TILESETS LIST”

9.1. Overview

REQUIREMENTS CLASS 3: REQUIREMENTS CLASS TILESETS LIST	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tilesets-list`
TARGET TYPE	Web API
PREREQUISITE	Requirements class 2: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tileset`

This class defines a resource called *tilesets list* that provides a list of elements, each one including a link to an individual tileset resource and a small set of metadata. To obtain a complete definition of the Tileset you should follow the link to the Tileset (in application/json, you will retrieve a TileSetMetadata document; see Clause 8). Refer to the “Dataset Tilesets” (Clause 10) and “GeoData Tilesets” Clause 11 requirements classes that describe two complementary mechanisms for associating tilesets lists with datasets and geospatial data resources, respectively.

9.2. Tilesets list

A *tilesets list* resource links to a list of sets of tiles, each one belonging to a particular TileMatrixSet.

9.2.1. Tilesets path

This Class does not specify the full path to a tileset list but requires that it ends with `/tiles`.

REQUIREMENT 9	
IDENTIFIE	`/req/tilesets-list/tileset-path`
A	The API SHALL support a GET operation on a …`/tiles` path returning a list of available tilesets

9.2.2. Response

A successful GET response to a list of tilesets resource will respond with a data structure that lists the tileset URLs available (one for each Tile Matrix Set supported).

REQUIREMENT 10

IDENTIFII /req/tilesets-list/tileset-links

A A successful response of an HTTP GET SHALL consist of an object with a property called `tilesets` which is a list of available tilesets, each element containing: `dataType`, `crs`, a `link` to the tileset (with rel: `self`), a `link` to the tileMatrixSet definition (with rel: `http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme`) and `tileMatrixSetURI` (if the TMS is available in a register) (this is a subset of the tileset metadata; as defined by the 2D Tile Matrix Set and Tile Set Metadata standard).

B Each element of that list SHALL include a `link` to a resource providing the full version of the *tileset metadata*, using link relation `self`.

C The tileset-list endpoint SHALL support negotiation of an application/json response. In this case, a successful response of an HTTP GET SHALL be encoded following the JSON schema (that represents a subset of the TileSet Metadata in 2D Tile Matrix Set and Tile Set Metadata standard 2.0).

```
{
 "$schema": "http://json-schema.org/draft-07/schema#",
 "description": "List of available tilesets.",
 "type": "object",
 "required": [ "tilesets" ],
 "properties": {
  "links": {
   "type": "array",
   "items": {
    "$ref": "https://schemas.opengis.net/tms/2.0/json/link.json"
   }
  },
  "tilesets": {
   "type": "array",
   "items": {
    "type": "object",
    "required": [ "dataType", "links", "crs" ],
    "properties": {
     "title": {
      "description": "A title for this tileset",
      "type": "string"
     },
     "dataType": {
      "description": "Type of data represented in the tileset",
      "anyOf": [
       { "type": "string" },
                      {
                          "type": "string",
                          "enum": ["map", "vector", "coverage"]
                      }
      ]
     },
     "crs": {
      "allOf": [
       {
        "description": "Coordinate Reference System (CRS)"
       },
```

REQUIREMENT 10

```
              {
               "$ref": "https://schemas.opengis.net/tms/2.0/json/crs.json"
              }
             ]
            },
            "tileMatrixSetURI": {
             "description": "Reference to a Tile Matrix Set on an offical source for
Tile Matrix Sets such as the OGC NA definition server (http://www.opengis.net/
def/tms/). Required if the tile matrix set is registered on an open official
source.",
             "type": "string",
             "format": "uri"
            },
            "links": {
             "description": "Links to related resources. A 'self' link to the tileset
as well as a 'http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme' link to a
definition of the TileMatrixSet are required.",
             "type": "array",
             "items": {
              "$ref": "https://schemas.opengis.net/tms/2.0/json/link.json"
             }
            }
           }
          }
         }
        }
       }
```

D	If the tileset-list endpoint also supports negotiation of an application/xml response, each item in the tilesets list in a successful response of an HTTP GET SHALL be encoded following the data model and XML schema for tileset metadata, as defined by the 2D Tile Matrix Set and Tile Set Metadata standard 2.0.

Example — Example of a tilesets list response

```
{
  "tilesets": [
    ...
    {
      "title" : "Buildings (WebMercatorQuad)",
      "tileMatrixSetURI" : "http://www.opengis.net/def/tilematrixset/OGC/1.0/
WebMercatorQuad",
      "crs" : "http://www.opengis.net/def/crs/EPSG/0/3857",
      "dataType" : "vector",
      "links" : [
        {
          "rel": "self",
          "href": "http://data.example.com/collections/buildings/tiles/WebMerc
atorQuad",
          "type": "application/json"
        },
        {
          "rel": "http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme",
          "href": "http://data.example.com/tileMatrixSets/WebMercatorQuad",
          "type": "application/json"
        }
      ]
    }
    ...
  ]
}
```

RECOMMENDATION 9

IDENTIFII	`/rec/tilesets-list/tileset_title`
A	The tilesets array metadata (as defined by the 2D Tile Matrix Set and Tile Set Metadata standard) subset SHOULD include a short human readable title.

PERMISSION 4

IDENTIFII	`/per/tilesets-list/tilesets-api`
A	An API document can advertise a single resource path (expressed as a URI template) to get multiple tilesets.
B	This URI template will use the `{tileMatrixSetId}` variable. The `{tileMatrixSetId}` variable will be interpreted as one of TileMatrixSet identifiers supported by the resource. In other words, the URI template represents all the TileMatrixSets supported by the resource. Note that a `{tileMatrixSetId}` variable must NOT be used in the templated links of the tileset metadata.

10

REQUIREMENTS CLASS “DATASET TILESETS”

10 REQUIREMENTS CLASS “DATASET TILESETS”

10.1. Overview

REQUIREMENTS CLASS 4: REQUIREMENTS CLASS DATASET TILESETS	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/dataset-tilesets`
TARGET TYPE	Web API
PREREQUISITES	Requirements class 3: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tilesets-list` http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core

The Dataset Tilesets Requirements Class defines a mechanism to retrieve a list of tilesets for a dataset (a.k.a., a root resource) which may contain multiple geospatial data resources. This Class is used in conjunction with the Requirements Class “Tilesets List” (Clause 9) that specifies the response to this GET request.

Combined with the Requirements Class “Collections Selection” (Clause 12), selecting specific geospatial data resources (collections) to be combined and retrieved as tiles is possible. For retrieving the list of tilesets for only one collection, the Requirements Class “Geodata Tilesets” (Clause 11) can also be used if it is implemented.

If an OGC based Web API offers several geospatial data resources for a dataset, that API may limit the number of collections that can be retrieved together, and/or provide a limited subset of the collections by default.

10.2. General

This Requirements Class describes how to serve tilesets for a dataset provided as a Web API instance implementing *OGC API — Common — Part 1* requirements classes or OGC API — Features — Part 1: Core, version 1.0 as an alternative.

10.3. Web API Landing Page

The landing page provides links to start exploring the resources offered by the Web API. The landing page mainly consists of a list of links to root resources. This Requirements Class requires including in the landing page one or more link(s) to list(s) of tilesets available for the dataset.

10.3.1. Response

REQUIREMENT 11	
IDENTIFII	`/req/dataset-tilesets/landingpage`
A	If the API has a mechanism for exposing root resources (e.g., a landing page), the API SHALL advertise at least one URI to retrieve a tilesets list provided by this service with a link having a `rel` value: `http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector`, `http://www.opengis.net/def/rel/ogc/1.0/tilesets-map` or `http://www.opengis.net/def/rel/ogc/1.0/tilesets-coverage`.

In the landing page, in JSON format, the links follow the link schema defined in the *OGC API — Common — Part 1: Core* or in *OGC API — Features — Part 1: Core* 1.0. The example below shows a fragment of the response to an *OGC API — Tiles* landing page showing links to dataset tilesets.

Example — Web API Landing Page fragment that advertises dataset tilesets

```
{
  "links": [
    ...
    {
       "href": "http://data.example.org/tiles",
       "rel": "http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector",
       "type": "application/json",
       "title": "List of available vector features tilesets for the dataset"
    },
    {
       "href": "http://data.example.org/tiles",
       "rel": "http://www.opengis.net/def/rel/ogc/1.0/tilesets-map",
       "type": "application/json",
       "title": "List of available map tilesets for the dataset"
    }
  ]
}
```

RECOMMENDATION 10	
IDENTIFIER	`/rec/dataset-tilesets/landingpage`
A	The URI to the dataset tilesets will be `/tiles`.

10.4. Dataset tilesets

The dataset tilesets operation provides essential tileset metadata and links to tilesets resources that support requesting tiles from the dataset.

10.4.1. Operation

REQUIREMENT 12	
IDENTIFII	`/req/dataset-tilesets/operation`
A	The dataset resource (the root resource) SHALL have at least one tileset accessible at .../`tiles` supporting an HTTP GET operation.
B	The URI SHALL be composed of two parts: The initial part is the URI of the dataset resource (the root resource) that can be represented as tiles and the final part follows the pattern `/tiles`.

The request of this operation has no parameters.

10.4.2. Response

A successful response to a dataset tilesets list request is a list of tilesets as defined by the Requirements Class “Tileset List” (Clause 9) and the Requirements Class “Tileset” (Clause 8).

10.5. Tiles

The Requirements Class “Core” (Clause 7) defines how to retrieve a single tile from a tileset available for the dataset.

10.5.1. Response

The response is expected to represent the entire dataset. In a server providing access through a Web API to a complex dataset formed by several geospatial data resources, it can be useful to select specific sub-resources of interest when requesting data from this dataset. This can be achieved with the use of the Requirements Class “Collections Selection” (Clause 12) or as an automatic decision by the server.

RECOMMENDATION 11	
IDENTIFIE	`/rec/dataset-tilesets/geodata-selection`
A	When it is possible and sensible, all geospatial data resources (`/collections`) SHOULD be represented in the tiles.

PERMISSION 5	
IDENTIFII	`/per/dataset-tilesets/geodata-selection`
A	If it is not possible and sensible to represent all geospatial data resources (`/collections`) in tiles (e.g., it compromises performance or tiles are overcrowded with too many elements), the server MAY select only the most significant geospatial data resources.

11

REQUIREMENTS CLASS “GEODATA TILESETS”

11 REQUIREMENTS CLASS “GEODATA TILESETS”

11.1. Overview

REQUIREMENTS CLASS 5: REQUIREMENTS CLASS GEODATA TILESETS	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/geodata-tilesets`
TARGET TYPE	Web API
PREREQUISITES	Requirements class 3: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tilesets-list` http://www.opengis.net/spec/ogcapi-common-2/1.0/req/collections

The GeoData Tilesets Requirements Class assumes that data is organized into one or more geospatial data resources (e.g., the “collections” OGC API — Common — Part 2: Geospatial Data). Geospatial data resources are referenced using URIs.

The GeoData Tilesets Class defines how to specify link(s) to one or more list(s) of tilesets containing a representation of this geospatial data resource (path). This Class is used in conjunction with Clause 9 that specifies the response to this GET request.

11.2. General

RECOMMENDATION 12	
IDENTIFIE	`/rec/geodata-tilesets/api-common`
A	An implementation of this Standard SHOULD consider implementing the requirements specified in the http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core (OGC API — Common — Part 1 version 1.0).

This Requirements Class depends on *OGC API — Common — Part 2: Geospatial Data*, but stays flexible and does not require implementation of *OGC API — Common — Part 1*. This allows for other Web API architectures outside the OGC API framework to adopt this Class. However, a server implementing other OGC APIs is expected to implement *OGC API — Common — Part 1*. In practice, this means that the landing page and the conformance page follow *OGC API — Common* core requirements classes when used. Combining this building block with OGC API — Features — Part 1: Core, version 1.0 instead of *OGC API — Common — Part 2* is also possible.

11.3. Geospatial data resources

This Standard does not specify how geospatial data resources are exposed in a server that implements OGC APIs and whether they can be retrieved as geospatial data (e.g., feature items). For example, OGC API — Features — Part 1: Core, version 1.0 includes the definition of collections and each collection is available in the `/collections/{collectionId}` path. OGC API — Common will provide a similar mechanism. Other paths in the Web API could also give access to geospatial data resources.

NOTE: The concept of geospatial data resource path replaces the concept of "layer" found in WMTS 1.0 but is intended to result in a better integration between data visualization and data access.

REQUIREMENT 13	
IDENTIFII	`/req/geodata-tilesets/desc-links`
A	If the Web API based server has a mechanism for geospatial data resources to expose links to geospatial resource aspects (e.g., feature items, metadata...), the API implementation SHALL include at least one of three `link` with the `href` pointing to tilesets list for geospatial data resources and with rel: `http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector`, `http://www.opengis.net/def/rel/ogc/1.0/tilesets-map` and `http://www.opengis.net/def/rel/ogc/1.0/tilesets-coverage`.

For example, an implementation of the OGC API — Features — Part 1: Core Standard returns a list of links that include geospatial resource aspects for each geospatial data resource in the `/collections/{collectionId}` path. *OGC API — Common — Part 2: Geospatial Data* provides a similar mechanism. In the JSON response, the `links` array is where links to lists of tilesets must be added.

Example — Fragment of a collection with a links array including two items pointing to lists of tilesets (one for vector tiles and one for map tiles).

```
[
  ...
  {
    "id": "buildings",
    "title": "Buildings in the city of Bonn",
    "description": "This collection contains buildings",
    "attribution": "OpenStreetMap",
    "extent": {
      ...
    }
    "links": [
      ...
      {
```

```
        "href": "http://data.example.com/collections/buildings/tiles",
        "rel": "http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector",
        "type": "application/json",
        "title": "List of available vector tilesets for the collection of Bonn
buildings"
      },
      {
        "href": "http://data.example.com/collections/buildings/tiles",
        "rel": "http://www.opengis.net/def/rel/ogc/1.0/tilesets-map",
        "type": "application/json",
        "title": "List of available map tilesets for the collection of Bonn
buildings"
      }
    ]
  }
  ...
]
```

RECOMMENDATION 13	
IDENTIFIER	/rec/geodata-tilesets/desc-links
A	The URI SHOULD consist of the path of the geospatial data resources followed by `/tiles`.

11.4. Geospatial data resources tilesets list

The geospatial data resource tilesets operation provides essential tileset metadata and links to tilesets resources that support requesting tiles from the resource.

11.4.1. Operation

REQUIREMENT 14	
IDENTIFIER	/req/geodata-tilesets/operation
A	The geospatial data resource SHALL have an associated list of at least one tileset accessible at …`/tiles` supporting an HTTP GET operation.
B	The URI SHALL be composed of two parts: The initial part is the URI of the geospatial data resource that can be represented as tiles and the final part follows the pattern `/tiles`.

This standard does not specify the need for any additional query parameter in the GET request.

11.4.2. Response

A successful response to a geospatial data tilesets request is a list of tilesets as defined by the Clause 9 and Clause 8.

11.5. Tiles

A tile resource is a fragment of a larger single geospatial data resource that is spatially constrained at the boundaries of the selected tile in a tile matrix set. Details of the operation are described in the Clause 7.

12

REQUIREMENTS CLASS “COLLECTIONS SELECTION”

12 REQUIREMENTS CLASS “COLLECTIONS SELECTION”

12.1. Overview

REQUIREMENTS CLASS 6: REQUIREMENTS CLASS COLLECTIONS SELECTION	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections-selection`
TARGET TYPE	Web API
PREREQUISITE	http://www.opengis.net/spec/ogcapi-common-2/1.0/req/collections

In a server providing access through a Web API to a complex dataset formed by several geospatial data resources, selecting specific sub-resources of interest when requesting data from this dataset can be useful. This requirements class defines how to include a query parameter when requesting a resource (e.g., dataset tiles) to specify which geospatial data resources (a.k.a. collections) should be used to generate the response.

This Requirements Class can be implemented e.g. in conjunction with the Requirements Class “Dataset Tilesets” (Clause 10) or in conjunction with an equivalent requirements class from *OGC API — Maps*.

12.2. Operation

By default, the geospatial data resources to be included in the dataset tiles responses are unspecified but should represent the entire dataset. This Class adds a mechanism to select the geospatial data resources to be used to generate the derived resources (e.g., maps or tiles). In practice this enables the capability to generate resources involving more than one geospatial data sub-resource.

12.2.1. Parameter `collections`

REQUIREMENT 15	
IDENTIFIER	/req/collections-selection/query-collections

A	An operation that acts on a resource consisting of multiple geospatial data sub-resources (e.g., a resource derived from a root dataset) SHALL support an optional parameter `collections` with the following characteristics (shown as OpenAPI Specification 3.0 fragment)
	`name: collections`
	`in: query`
	`required: false`
	`style: form`
	`explode: false`
	`schema:`
	`  type: array`
	`  items:`
	`    type: string`

B	The parameter `collections` SHALL be supported by tileset resources and tiles resources for origins consisting of multiple geospatial data sub-resources (e.g., dataset tileset at `{datasetAPI}/tiles/WebMercatorQuad` and dataset tiles at `{datasetAPI}/tiles/WebMercatorQuad/{tileMatrix}/{tileRow}/{tileCol}`).
C	Implementations SHALL support a comma-separated list of either geospatial resource identifiers (e.g., collectionId's) and/or full URLs to geospatial resource identifiers.

When this parameter refers to more than one geospatial data resource, this parameter will use the comma (",") as the separator between the resource identifiers in the list. Additional white space will not be used to delimit list items. If a geospatial data resource identifier includes a space or comma, it shall be escaped using the URL encoding rules (IETF RFC 2396).

PERMISSION 6	
IDENTIFIER	/per/collections-selection/valid-collections
A	An implementation MAY return an error when the specified list of collections is not supported, for reasons such as an incompatible combination, or an unsupported encoding or TileMatrixSet for some of the selected collections.

12.2.2. Response

REQUIREMENT 16	
IDENTIFIER	/req/collections-selection/collections-response
A	Only the collections of geospatial data enumerated in the values of the `collections` parameter SHALL be used to generate the responses for the resources (tiles and tilesets) to which they apply.

REQUIREMENT 16	
B	If there is more than one collection name and the style applied does not specify otherwise, the comma-separated collections SHALL be included in the result starting by the first (leftmost) and ending by the last (rightmost).

NOTE: For tiles representing maps, sub-requirement B will result in the first collection being portrayed at the bottom of the display with the others rendered on top of the previous ones, one by one (the collection mentioned rightmost in the collections parameter will become topmost in the portrayal).

12.2.3. Error conditions

If the value of the parameter `collections` contains a resource id of URI that does not exist in the Web API implementation, the status code of the response is 400.

If the value of the parameter `collections` has a wrong format or combines one or more geospatial data resources that are not compatible (e.g., they do not have a compatible value specified by other parameters in the request), the status code of the response is 400.

13

REQUIREMENTS CLASS “DATETIME”

13 REQUIREMENTS CLASS “DATETIME”

13.1. Overview

REQUIREMENTS CLASS 7: REQUIREMENTS CLASS DATETIME	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/datetime`
TARGET TYPE	Web API
PREREQUISITE	Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

The DateTime Requirements Class defines the way date and time can be used as a parameter to filter the content in the tile resource. This Requirements Class imports most of the functionality from OGC API — Commons.

13.2. Describing the temporal extent

Depending on the type of resource, the way the temporal extent and the resolution of the datetime values available for the client to request are described may be different:

- For Geodata Tilesets, the collection description should specify the temporal extent of the resources. Tiles can be requested inside this extent. If the extent is specified in a way that instant values are provided (e.g. by listing them or by including a resolution), then it may be possible to request tiles for these instants.
- If tileset metadata is available, a future revision of the 2D Tile Matrix Set and Tile Set Metadata Standard could take precedence providing more details on the available values for datetime.

13.3. datetime query parameter request and response

This section is based on the OGC API — Common datetime module that is entirely reproduced here.

Requirements Module	
http://www.opengis.net/spec/ogcapi-common-2/1.0/rm/datetime	
Target type	Web API Query Parameter

The `datetime` parameter selects resources based on their temporal extent. The definition of temporal extent is specific to the resource type being filtered.

The `datetime` parameter is defined as follows:

REQUIREMENT 17

IDENTIFII `/req/collections/rc-datetime-definition`

A The `datetime` parameter SHALL have the following characteristics (using an OpenAPI Specification 3.0 fragment):

```
name: datetime
in: query
required: false
schema:
  type: string
style: form
explode: false
```

B Temporal geometries are either a date-time value or a time interval. The parameter value SHALL conform to the following syntax (using ABNF):

```
interval-closed     = date-time "/" date-time
interval-open-start = [".."] "/" date-time
interval-open-end   = date-time "/" [".."]
interval            = interval-closed / interval-open-start / interval-open-end
datetime            = date-time / interval
```

C The syntax of `date-time` is specified by RFC 3339, 5.6.

D Open ranges in time intervals at the start or end are supported using a double-dot (`..`) or an empty string for the start/end.

While the processing of the `datetime` parameter is specific to the resource and operation for which it is applied, there is a general set of requirements which all implementations must address.

REQUIREMENT 18	
IDENTIFII	`/req/collections/rc-datetime-response`
A	If the `datetime` parameter is provided by the client and supported by the server, then only resources that have a temporal geometry that intersects the temporal information in the `datetime` parameter SHALL be part of the result set. If a resource has multiple temporal properties, it is the decision of the server whether only a single temporal property is used to determine the extent or all relevant temporal properties.
B	The `datetime` parameter SHALL match all resources in the collection that are not associated with a temporal geometry.

"Intersects" means that the time (instant or period) specified in the parameter `datetime` includes a timestamp that is part of the temporal geometry of the resource (again, a time instant or period). For time periods this includes the start and end time.

Table 10 — A note about ISO 8601-2

Note	ISO 8601-2 distinguishes open start/end timestamps (double-dot) and unknown start/end timestamps (empty string). For queries, an unknown start/end has the same effect as an open start/end.

Example 1 — A date-time: February 12, 2018, 23:20:52 UTC:

```
datetime=2018-02-12T23%3A20%3A52Z
```

For resources with a temporal property that is a timestamp (like `lastUpdate`), a date-time value would match all resources where the temporal property is identical.

For resources with a temporal property that is a date or a time interval, a date-time value would match all resources where the timestamp is on that day or within the time interval.

Example 2 — Intervals: February 12, 2018, 00:00:00 UTC to March 18, 2018, 12:31:12 UTC:

```
datetime=2018-02-12T00%3A00%3A00Z%2F2018-03-18T12%3A31%3A12Z
```

February 12, 2018, 00:00:00 UTC or later:

```
datetime=2018-02-12T00%3A00%3A00Z%2F..
```

March 18, 2018, 12:31:12 UTC or earlier:

```
datetime=..%2F2018-03-18T12%3A31%3A12Z
```

A template for the definition of the parameter in YAML according to OpenAPI 3.0 is available at datetime.yaml.

13.4. subset=datetime query parameter request and response

The behavior of the subset parameter is imported from the OGC API — Common subset module. The entire module is reproduced here. However, some considerations may not apply for datetime and the generic behavior is limited by predetermining the axis name of the generic datetime in an additional requirement, a permission and a recommendation at the end of this subsection.

Requirements Module	
http://www.opengis.net/spec/ogcapi-common-2/1.0/rm/subset	
Target type	Web API Query Parameter

The `subset` parameter is used to select a subset of a geospatial resource.

The `subset` parameter is defined as follows:

REQUIREMENT 19	
IDENTIFII	`/req/collections/rc-subset-definition`

A	The `subset` parameter SHALL have the following characteristics (using an Augmented Backus Naur Form (ABNF) fragment):
	`SubsetSpec:       "subset"=axisName(intervalOrPoint)`
	`axisName:         {text}`
	`intervalOrPoint:  interval | point`
	`interval:         low : high`
	`low:              point | *`
	`high:             point | *`
	`point:            {number} | "{text}"`
	
	`Where:`
	`\" = double quote = ASCII code 0x42,`
	`{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).`

B	The axis name SHALL correspond to one of the axes of the Coordinate Reference System (CRS) of the target resource or else return a 400 status code.
C	If the intervalOrPoint values fall entirely outside the range of valid values defined for the identified axis, a 204 status code SHALL be returned.
D	For a CRS where an axis can wrap around, such as subsetting across the dateline (anti-meridian) in a geographic CRS, a `low` value greater than `high` SHALL be supported to indicate an extent crossing that wrapping point.

NOTE: When the intervalOrPoint values fall partially outside of the range of valid values defined by the CRS for the identified axis, the service is expected to return the non-empty portion of the resource resulting from the subset.

While the processing of the `subset` parameter is specific to the resource and operation for which it is applied, there is a general set of requirements which all implementations must address.

REQUIREMENT 20	
IDENTIFII	`/req/collections/rc-subset-response`
A	Only that part of the resource that falls within the bounds of the subset expression SHALL be returned.
B	If a lower limit of the subset expression is populated with an asterisk “*” THEN the minimum extent of the resource along that axis SHALL be selected.
C	If an upper limit of the subset expression is populated with an asterisk “*” THEN the maximum extent of the resource along that axis SHALL be selected.

REQUIREMENT 21	
IDENTIFII	`/req/datetime/axis`
A	To subset a generic time dimension, the server SHALL support “datetime” as axis name in the subset parameter

Example 1 — A date-time (subset): February 12, 2018, 23:20:52 UTC:
`subset=datetime(%222018-02-12T23%3A20%3A52Z%22)`

For resources with a temporal property that is a timestamp (like `lastUpdate`), a date-time value would match all resources where the temporal property is identical.

For resources with a temporal property that is a date or a time interval, a date-time value would match all resources where the timestamp is on that day or within the time interval.

Example 2 — Intervals (subset): February 12, 2018, 00:00:00 UTC to March 18, 2018, 12:31:12 UTC:
`subset=datetime(%222018-02-12T00%3A00%3A00Z%22%3A%222018-03-18T12%3A31%3A12Z%22)`

February 12, 2018, 00:00:00 UTC or later:

```
subset=datetime(%222018-02-12T00%3A00%3A00Z%22%3A*)
```

March 18, 2018, 12:31:12 UTC or earlier:

```
subset=datetime(*%3A%222018-03-18T12%3A31%3A12Z%22)
```

13.5. Actual date & time response header

RECOMMENDATION 14	
IDENTIFIE	`/rec/actual-datetime`

A	The server SHOULD add an HTTP header with `OGCAPI-datetime` as a name and a temporal geometry as a value, to indicate the instant or the temporal interval of the content of the resource. The temporal geometries value shall conform to the following syntax (using ABNF):
	`interval     = instant "/" instant`
	`datetime     = instant / interval`
	The syntax of `instant` is specified by RFC 3339, 5.6.

13.6. Closest date & time permission

PERMISSION 7	
IDENTIFII	`/per/datetime/closest`
A	In case the requested tile is not available in the exact requested datetime for the tile matrix, tile column and tile row, the closest or last previous time for which data is available MAY be returned by the server.

NOTE: An Earth Observation use case where this permission is useful is to allow retrieving a tile of the last datetime where imagery is available while taking into account that a certain geographic area may only be observed at an interval of "every few days" and availability may be irregular and conditioned by clouds.

14

REQUIREMENTS CLASS “OPENAPI SPECIFICATION 3.0”

14 REQUIREMENTS CLASS “OPENAPI SPECIFICATION 3.0”

14.1. Overview

REQUIREMENTS CLASS 8: REQUIREMENTS CLASS OPENAPI 3.0	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/oas30`
TARGET TYPE	Web API
PREREQUISITE	http://www.opengis.net/spec/ogcapi-common-1/1.0/req/oas30

The OpenAPI 3.0 Requirements Class details providing a definition of a Web API implementing the OGC API — Tiles Standard using the OpenAPI Specification version 3.0.

14.2. Web API OpenAPI description

The OpenAPI definition provides a description of the complete list of API resources. Reading this description, an application would have the full picture of the resources that the API provides, how to retrieve them, and what responses are expected for successful and unsuccessful requests. Without this API description an application would be forced to traverse all links, starting with the landing page, to get an equivalent full list of resources.

The (*oas30*) requirement class from *OGC API — Common — Part 1: Core* provides many details on general requirements that this requirements class adopts by dependency. In addition, extra requirements are provided. Also, the *OGC API — Common* requirements class provides details on how to request an API definition.

14.2.1. Response

14.2.1.1. Completeness

The OpenAPI definition resulting as a response of this request needs to take into consideration the relevant resources specified in this standard.

REQUIREMENT 22	
IDENTIFII	`/req/oas30/completeness`
A	The OpenAPI definition SHALL provide paths for all tileset, tilesets list and tile resources provided by the API instance
B	The `paths` defined in the OpenAPI definition SHALL be consistent with the links to the same resources provided by the landing page, collections, tileset and tilesets list resources.
C	The `paths` defined in the OpenAPI definition SHALL provide the description of the parameters that the tileset and tile resources need to operate that are specified in corresponding conformance classes

14.2.1.2. Reusable API components

Reusable components for creating OpenAPI definitions for implementations of this OGC API can be found in http://schemas.opengis.net/ogcapi/tiles/part1/1.0/openapi

A server implementation of the OGC API — Tiles Standard can use the content in the openapi folder in https://schemas.opengis.net/ogcapi/tiles/part1/1.0 to generate a response for the openapi description. The ogcapi-tiles-1.yaml file includes paths and components. An implementation should only include the paths that are implemented and remove the references to the rest. The components part includes parameters, responses and schemas that can be reused as-is. The `api` folder (https://schemas.opengis.net/ogcapi/tiles/part1/1.0/openapi/api/) contains JSON files that are templates with enumerated values for collections (all, coverages or vector), styles, tileMatrixSets. A particular implementation of this API should enumerate the actual resources exposed by the API in the same way. The server can select to dynamically implement responses to /api/* (where * is replaced by all-collections, styles, …) or hardcode the /api/* files with the actual list of resource identifiers in the enumerations.

To improve performance, the whole content of this folder can be bundled into a single document by executing a tool such as *swagger-cli*. This can be served for the OGC API — Common — Part 1 `service-desc` link from the landing page.

14.2.1.3. Path Operation Ids

The OpenAPI definition provides to a client application a set of paths that the client can use to interact with the API and get new resources. The OpenAPI description of each path provides a description of what parameters to use in the request and what to expect in the response. However, this standard is not proposing a fixed set of paths so there is an issue identifying the requirements classes pertaining to each path in an API instance. In other words, the OpenAPI description alone does not provide enough information by itself and there is a need to identify the requirements classes pertaining to a resource (a path) and complete the information necessary for the client to implement the necessary logic to generate the request and understand the response. This standard proposes a suffix mechanism to be applied to the `operationId` property of the path to list the requirement classes pertaining to each path. Each path should have a unique operationId suffix, so it is expected that the OpenAPI instance provides a prefix to the proposed suffixes that make each operationId unique.

REQUIREMENT 23	
IDENTIFII	`/req/oas30/operation-id`
A	The `paths` defined in the OpenAPI definition SHALL have an *operationId* value ending with the relevant dot-separated suffix corresponding to the resource as specified in Table 11.

Table 11 — OpenAPI operationId suffixes

ORIGIN	STYLED	DATA TYPE	RESOURCE	*OPERATIONID* SUFFIXES[1]
With the resource types and origins described in this document				
DataSet[6]		Vector	TileSetsList[5]	`.dataset.vector.getTileSetsList`
DataSet[6]		Vector	TileSet[4]	`.dataset.vector.getTileSet`
DataSet[6]		Vector	Tile[3]	`.dataset.vector.getTile`
DataSet[6]	Styled[8]	Vector	TileSetsList[5]	`.dataset.style.vector.getTileSetsList`
DataSet[6]	Styled[8]	Vector	TileSet[4]	`.dataset.style.vector.getTileSet`
DataSet[6]	Styled[8]	Vector	Tile[3]	`.dataset.style.vector.getTile`
DataSet[6]		Map[10]	TileSetsList[5]	`.dataset.map.getTileSetsList`
DataSet[6]		Map[10]	TileSet[4]	`.dataset.map.getTileSet`

ORIGIN	STYLED	DATA TYPE	RESOURCE	*OPERATIONID* SUFFIXES[1]
DataSet[6]		Map[10]	Tile[3]	`.dataset.map.getTile`
DataSet[6]	Styled[8]	Map[10]	TileSetsList[5]	`.dataset.style.map.getTileSetsList`
DataSet[6]	Styled[8]	Map[10]	TileSet[4]	`.dataset.style.map.getTileSet`
DataSet[6]	Styled[8]	Map[10]	Tile[3]	`.dataset.style.map.getTile`
Collection[7]		Vector	TileSetsList[5]	`.collection.vector.getTileSetsList`
Collection[7]		Vector	TileSet[4]	`.collection.vector.getTileSet`
Collection[7]		Vector	Tile[3]	`.collection.vector.getTile`
Collection[7]	Styled[8]	Vector	TileSetsList[5]	`.collection.style.vector.getTileSetsList`
Collection[7]	Styled[8]	Vector	TileSet[4]	`.collection.style.vector.getTileSet`
Collection[7]	Styled[8]	Vector	Tile[3]	`.collection.style.vector.getTile`
Collection[7]		Coverage[9]	TileSetsList[5]	`.collection.coverage.getTileSetsList`
Collection[7]		Coverage[9]	TileSet[4]	`.collection.coverage.getTileSet`
Collection[7]		Coverage[9]	Tile[3]	`.collection.coverage.getTile`
Collection[7]		Map[10]	TileSetsList[5]	`.collection.map.getTileSetsList`
Collection[7]		Map[10]	TileSet[4]	`.collection.map.getTileSet`
Collection[7]		Map[10]	Tile[3]	`.collection.map.getTile`
Collection[7]	Styled[8]	Map[10]	TileSetsList[5]	`.collection.style.map.getTileSetsList`
Collection[7]	Styled[8]	Map[10]	TileSet[4]	`.collection.style.map.getTileSet`
Collection[7]	Styled[8]	Map[10]	Tile[3]	`.collection.style.map.getTile`
With other potential resource types				
DataSet[6]		*other*	TileSetsList[5]	`.dataset.*.getTileSetsList`

ORIGIN	STYLED	DATA TYPE	RESOURCE	*OPERATIONID* SUFFIXES[1]
DataSet[6]		*other*	TileSet[4]	`.dataset.*.getTileSet`
DataSet[6]		*other*	Tile[3]	`.dataset.*.getTile`
DataSet[6]	Styled[8]	*other*	TileSetsList[5]	`.dataset.style.*.getTileSetsList`
DataSet[6]	Styled[8]	*other*	TileSet[4]	`.dataset.style.*.getTileSet`
DataSet[6]	Styled[8]	*other*	Tile[3]	`.dataset.style.*.getTile`
Collection[7]		*other*	TileSetsList[5]	`.collection.*.getTileSetsList`
Collection[7]		*other*	TileSet[4]	`.collection.*.getTileSet`
Collection[7]		*other*	Tile[3]	`.collection.*.getTile`
Collection[7]	Styled[8]	*other*	TileSetsList[5]	`.collection.style.*.getTileSetsList`
Collection[7]	Styled[8]	*other*	TileSet[4]	`.collection.style.*.getTileSet`
Collection[7]	Styled[8]	*other*	Tile[3]	`.collection.style.*.getTile`
With other potential origins				
other		Vector	TileSetsList[5]	`#.vector.getTileSetsList`
other		Vector	TileSet[4]	`#.vector.getTileSet`
other		Vector	Tile[3]	`#.vector.getTile`
other	Styled[8]	Vector	TileSetsList[5]	`#.style.vector.getTileSetsList`
other	Styled[8]	Vector	TileSet[4]	`#.style.vector.getTileSet`
other	Styled[8]	Vector	Tile[3]	`#.style.vector.getTile`
other		Coverage[9]	TileSetsList[5]	`#.coverage.getTileSetsList`
other		Coverage[9]	TileSet[4]	`#.coverage.getTileSet`
other		Coverage[9]	Tile[3]	`#.coverage.getTile`

ORIGIN	STYLED	DATA TYPE	RESOURCE	*OPERATIONID* SUFFIXES[1]
other		Map[10]	TileSetsList[5]	`#.map.getTileSetsList`
other		Map[10]	TileSet[4]	`#.map.getTileSet`
other		Map[10]	Tile[3]	`#.map.getTile`
other	Styled[8]	Map[10]	TileSetsList[5]	`#.style.map.getTileSetsList`
other	Styled[8]	Map[10]	TileSet[4]	`#.style.map.getTileSet`
other	Styled[8]	Map[10]	Tile[3]	`#.style.map.getTile`
other		*other*	TileSetsList[5]	`#.*.getTileSetsList`
other		*other*	TileSet[4]	`#.*.getTileSet`
other		*other*	Tile[3]	`#.*.getTile`
other	Styled[8]	*other*	TileSetsList[5]	`#.style.*.getTileSetsList`
other	Styled[8]	*other*	TileSet[4]	`#.style.*.getTileSet`
other	Styled[8]	*other*	Tile[3]	`#.style.*.getTile``

[1] '*' represents a *resource type* to be defined in another relevant standard. '#' represents an optional *origin* that could be defined in another relevant standard. [2] The suffixes derived from these resource types are not required by this standard and will be proposed by the relevant standard defining them. [3] The *Tile* resource is defined in Clause 7. [4] The *TileSet* resource is defined in Clause 8. [5] The *TileSetsList* resource is defined in Clause 9. [6] The *DataSet* origin is defined in Requirements Class "Dataset Tilesets" Clause 10 and depends on *OGC API — Common — Part 1: Core*. [7] The *Collection* origin is defined in Requirements Class "GeoData Tilesets" Clause 11 and depends on the *Collections* requirements class defined in *OGC API — Common — Part 2: Geospatial data* [8] Styled tilesets rely on the ability to list styles defined in *OGC API — Styles*[2]. [9] Coverage tilesets rely on the *Coverage tiles* conformance class defined in *OGC API — Coverages — Part 1: Core*[2]. [10] Map tilesets rely on the *Map tiles* conformance class defined in *OGC API — Maps — Part 1: Core*[2].

15

REQUIREMENTS CLASS "XML TILESET METADATA"

15 REQUIREMENTS CLASS “XML TILESET METADATA”

15.1. Overview

The *OGC API — Tiles* Standard provides several resources describing the service (landing page), the geospatial data resources, the tileset lists and the tilesets informing the client on how to retrieve tiled data. XML is a data-interchange format designed to facilitate structured data interchange between applications. The intention of this section is to define an XML encoding that could be implemented by Web APIs that implement OGC API — Tiles, but not to exclude the possibility of defining additional metadata encodings that may be offered by the Web APIs. Indeed, Web API implementations may adopt this XML requirements class and declare conformance to it in the list of conformance classes supported by the Web API. The declaration of XML in the conformance classes supported does not mean that all the resources provided by the Web API support an XML representation, but a general support of XML is expected.

This clause specifies a requirements class for an XML representation that may be implemented by a Web API, that implements OGC API — Tiles, for the tilesets list and a tileset resources in addition to the JSON representation required by Clause 8 and Clause 9.

If the service provides local TileMatrixSet definitions, it is recommended that those resources also support an XML representation conforming to the XML schema specified in the OGC Two Dimensional Tile Matrix Set and Tile Set Metadata 2.0 standard.

Please refer to *OGC API — Common* conformance classes defining the XML representation of common resources such as the landing page, conformance and collections.

15.2. TileSet and TileSets List XML representation

REQUIREMENTS CLASS 9: REQUIREMENTS CLASS XML TILESET METADATA	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/xml`
TARGET TYPE	Web API
PREREQUISITES	http://www.opengis.net/spec/tms/2.0/req/xml-tilesetmetadata Requirements class 2: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tileset`

REQUIREMENTS CLASS 9: REQUIREMENTS CLASS XML TILESET METADATA	
	Requirements class 3: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tilesets-list`

15.2.1. Operation

REQUIREMENT 24	
IDENTIFII	`/req/xml/definition`
A	`200`-responses of the server SHALL support the `application/xml` media type for the Tilesets list and TileSet resources.

15.2.2. Response

REQUIREMENT 25	
IDENTIFII	`/req/xml/content`
A	Every request to a TileSets list or TileSet resource which: 1. Receives a `200`-response 2. with the `Content-Type` header set to `application/xml` SHALL include, or link to, a payload encoded according to the Extensible Markup Language (XML) 1.0
B	The payload for these responses SHALL conform with the XML Schema specified for the resource in the OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata.

16

REQUIREMENTS CLASSES FOR TILE ENCODINGS

16 REQUIREMENTS CLASSES FOR TILE ENCODINGS

The *OGC API — Tiles* Standard is designed to support tiling data that can potentially be encoded and provided in many of the existing geospatial formats or new ones that could be invented in the future. Web API implementations may adopt these encodings and declare conformance to them in the list of conformance classes supported by the Web API. The intention of this section is NOT to limit the number of encodings offered by an implementation of OGC API — Tiles but to provide a minimum set of encodings that could be implemented by Web APIs that implement this Standard, as well as to provide a practical way to test conformance to this Standard. For each of these encodings, a Requirements Class is defined. Web API implementations are free to support other encodings and data formats that can be convenient in each case and use them even though they may not be listed as supported conformance classes by the Web API. In addition, the declaration of an encoding in the conformance classes supported does not mean that all the resources provided by the Web API should support all of them. Partial support could be determined by the differences in the nature of the data behind each collection.

16.1. Overview

This clause specifies six pre-defined requirements classes for encodings to be used by the tiles of the OGC API implementation:

- PNG
- JPEG
- TIFF
- NetCDF
- GeoJSON
- Mapbox Vector Tiles

NOTE: None of the encodings specified here is mandatory and an implementation of this standard may implement none of them but implement other encodings instead.

16.2. Requirements Class “PNG”

One fundamental use case for tiled geospatial content is visualization in a web browser. For this use case, selecting an encoding that can be natively interpreted by the web browser is fundamental. The PNG format is one of the most popular formats on the World Wide Web. The format is defined by the ISO Information technology — Computer graphics and image processing — Portable Network Graphics (PNG): Functional specification (ISO/IEC 15948:2004), which is also available as a W3C Recommendation at https://www.w3.org/TR/PNG. Note that the standard is listed as ISO/IEC 15948:2003 on the W3C Recommendation.

PNG supports lossless data compression. PNG supports palette-based images (with palettes of 24-bit RGB or 32-bit RGBA colors), grayscale images (with or without an alpha channel for transparency), and full-color non-palette-based RGB or RGBA images. The PNG working group designed the format for transferring images on the Internet, not for professional-quality print graphics. Therefore non-RGB color spaces such as CMYK are not supported.

REQUIREMENTS CLASS 10: REQUIREMENTS CLASS PNG	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/png`
TARGET TYPE	Web API
PREREQUISITES	ISO/IEC 15948 Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 26	
IDENTIFII	`/req/png/content`
A	Every 200-response of the server with the media type image/png SHALL be a PNG image representing only one tile
B	The colors of the PNG SHALL represent the geospatial features or coverage values in the tile.
C	The alpha channel of the PNG SHALL be used when partial transparency is required
D	All tiles representing parts of the same resource or resources and using the same style SHALL follow the same portrayal rules

NOTE: The way colors in a PNG image are mapped to geospatial features or coverage values is out of scope of this Standard. However, a common set of portrayal rules for all tiles representing part of the same resource is essential, as adjacent tiles are normally represented one next to the other and presented as a single image to the user.

16.3. Requirements Class “JPEG”

As above, selecting an encoding that can be natively interpreted by the web browser is fundamental. The JPEG format is one of the most popular formats on the World Wide Web and is defined by the ITU-T Recommendation T.81 and the ISO/IEC 10918-1 standard.

The JPEG compression algorithm operates at its best on photographs and paintings with smooth variations of tone and color. JPEG is best used for color and grayscale still images, but not for binary images. JPEG is also the most common format used by digital cameras to encode and store pictures. However, JPEG is not well suited for line drawings and other textual or iconic graphics, where the sharp contrasts between adjacent pixels can cause noticeable artifacts. Such images are better saved in a lossless graphics format such as PNG. Because JPEG is a lossy compression method, which reduces the image fidelity, it is inappropriate for exact reproduction of imaging data.

REQUIREMENTS CLASS 11: REQUIREMENTS CLASS JPEG	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/jpeg`
TARGET TYPE	Web API
PREREQUISITES	ISO/IEC 10918-1 Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 27	
IDENTIFII	`/req/jpeg/content`
A	Every 200-response of the server with the media type image/jpeg SHALL be a JPEG document representing only one tile
B	The colors of the JPEG SHALL represent coverage values in the tile.
C	All tiles representing parts of the same resource or resources and using the same style SHALL follow the same portrayal rules

NOTE 1: The way the colors in a JPEG image are mapped to geospatial features or coverage values is out of scope of this standard. However, a common set of portrayal rules for all tiles representing part of the same resource is essential, as adjacent tiles are normally represented one next to the other and presented as a single image to the user.

NOTE 2: The use of JPEG to represent linear features or color solid polygons is not recommended.

16.4. Requirements Class “TIFF”

One use case for tiles is to distribute fragmented coverage values as regular grids. In these circumstances, selecting an encoding that can store grid values in their original format and eventually compress them using lossless compression is the right solution. The TIFF format is one of the older formats but still is one of the most popular formats to preserve arrays of data values. The format is defined by the TIFF v6 specification, which is owned by Adobe Systems.

TIFF is a flexible, adaptable file format for handling images and data. The ability to store data in a lossless format makes a TIFF file a useful image archive. This is because, unlike standard JPEG files, a TIFF file can use lossless compression methods such as PackBits or LZW compression.

REQUIREMENTS CLASS 12: REQUIREMENTS CLASS TIFF	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tiff`
TARGET TYPE	Web API
PREREQUISITES	TIFF V6.0 Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 28	
IDENTIFII	`/req/tiff/content`
A	Every 200-response of the server with the media type image/tiff SHALL be a TIFF document representing only one tile
B	The TIFF file SHALL be organized in strips (avoiding organization in internal tiles).
C	All tiles representing parts of the same resource or resources and using the same style SHALL follow the same portrayal rules or represent data with the same reference and units of measure.

NOTE: TIFF is an ideal format for storing geospatial grid data with its original data values. However, TIFF can also be used for image palette or RGB imagery.

RECOMMENDATION 15	
IDENTIFIE	`/rec/tiff/geotiff`
A	A TIFF encoding SHOULD include georeference information in GeoTIFF format, and conforming to the OGC GeoTIFF 1.1 Standard (OGC 19-008r4)

REQUIREMENT 29	
IDENTIFII `/req/tiff/geotiff`	
A	If the TIFF encoding incorporates a GeoTIFF georeference, this information SHALL be consistent with the TileMatrixSet, TileMatrix, TileRow and TileCol

16.5. Requirements Class “NetCDF”

In the case of multidimensional regular grid tiles, as defined in Annex J of the 2D TMS Standard [OGC 17-083r4], there is a need for a format that can store multidimensional data in a natural way. The NetCDF format is one of the most popular formats for scientific data that is able to store multi-dimensional arrays of data values. NetCDF format is defined by the UCAR-Unidata community and standardized in the OGC Network Common Data Form (NetCDF) Core Encoding Standard version 1.0 (OGC 10-090r3).

REQUIREMENTS CLASS 13: REQUIREMENTS CLASS NETCDF	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/netcdf`
TARGET TYPE	Web API
PREREQUISITES	OGC Network Common Data Form (NetCDF) Core Encoding Standard version 1.0 Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 30	
IDENTIFII `/req/netcdf/content`	
A	Every 200-response of the server with the media type application/netcdf or application/x-netcdf SHALL be a NetCDF document representing only one tile
B	The NetCDF file SHALL contain only data in two or more dimensions

REQUIREMENT 31	
IDENTIFII `/req/netcdf/geo`	

REQUIREMENT 31	
A	If the NetCDF encoding incorporates a georeference, this information SHALL be consistent with the TileMatrixSet, TileMatrix, TileRow and TileCol

16.6. Requirements Class "GeoJSON"

GeoJSON is a commonly used format for representing features with simple geometries and related properties. GeoJSON is simple to understand and well supported by tools and software libraries. Since most Web developers are comfortable with using JSON-based formats, supporting GeoJSON is recommended for vector tiles. The caveat is whether the feature data can be represented in GeoJSON for the intended use.

REQUIREMENTS CLASS 14: REQUIREMENTS CLASS GEOJSON	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/geojson`
TARGET TYPE	Web API
PREREQUISITES	GeoJSON Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 32	
IDENTIFII	`/req/geojson/content`
A	Every 200-response of the server with the media type application/geo+json SHALL be a GeoJSON document representing only one tile.
B	The root of the GeoJSON document SHALL be: • A GeoJSON FeatureCollection Object for features. • A GeoJSON Feature Object for a single feature.
C	Regardless of the TileMatrixSet CRS, coordinates SHALL be in CRS84 as stated in the GeoJSON standard unless a *prior arrangement* applies to use an alternative coordinate reference system. The fact that a TileMatrixSet is used does not constitute a *prior arrangement*. For example, an extension could negotiate another CRS for coordinates with a query parameter.

PERMISSION 8	
IDENTIFIE	`/rec/geojson/overflow`
A	A GeoJSON content of a tile can contain features that are partially outside of the tile bounding box.

16.7. Requirements Class "Mapbox Vector Tiles"

Mapbox Vector Tiles is a well-known format for representing features with geometries in a tile [15]. The Mapbox Vector Tile format uses Google Protocol Buffers as an encoding format. Protocol Buffers are a language-neutral, platform-neutral extensible mechanism for serializing structured binary data. A Mapbox Vector Tile represents data based on a square extent within a tile matrix set. However, a Mapbox Vector Tile does not contain information about its bounds and TileMatrixSet. The file format assumes that the client knows the bounds and TileMatrixSet of the file before decoding it.

REQUIREMENTS CLASS 15: REQUIREMENTS CLASS MVT	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/mvt`
TARGET TYPE	Web API
PREREQUISITES	Mapbox Vector Tiles [15] Requirements class 1: `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core`

REQUIREMENT 33	
IDENTIFII	`/req/mvt/content`
A	Every 200-response of the server with the media type application/vnd.mapbox-vector-tile SHALL be a Mapbox Vector Tile document representing only one tile.
B	A feature SHALL contain a geometry field. A feature SHALL contain a type field as described in the Geometry Types section.
C	The grid space for the MVT coordinates SHALL map linearly to the coordinates in the tile extent expressed in the Tileset CRS unless a *prior arrangement* applies to use an alternative CRS; and in this case the linear mapping SHALL be done to the alternative CRS and not with the Tileset CRS. For example, an extension could negotiate another CRS for coordinates with a query parameter. In particular the 0,0 coordinate in the MVT maps to the top-left corner of the tile. The bottom-right corner of the tile corresponds the bottom-right corner of the MVT grid.

NOTE: The support of Mapbox Vector Tiles does not make the OGC API dependent on the Mapbox Vector Tile Specification. The support of Mapbox Vector Tiles is completely optional in a Web API. This Requirements Class is completely independent of the version of Mapbox Vector Tiles and future versions.

A

ANNEX A (NORMATIVE) ABSTRACT TEST SUITE

A

ANNEX A (NORMATIVE) ABSTRACT TEST SUITE

An implementation of this standard must satisfy the following system characteristics to be conformant with this specification.

A.1. Conformance Class “Core”

CONFORMANCE CLASS A.1	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/core`
SUBJECT	Requirements Class “Core”
TARGET TYPE	Web API

A.1.1. Declaration of conformance classes

A.1.1.1. Response

ABSTRACT TEST A.1	
IDENTIFIER	`/conf/core/conformance-success`
REQUIREMEI	Requirement 7: `/req/core/conformance-success`
TEST PURPOSE	Validate that the Conformance Declaration response complies with the required structure and contents.
TEST METHOD	1. If there is a Conformance Class declaration document, validate the response document against OpenAPI 3.0 schema confClasses.yaml

ABSTRACT TEST A.1	
	2. Validate that the document includes the conformance class “http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/core”

A.1.2. A tile

A.1.2.1. Operation

ABSTRACT TEST A.2	
IDENTIFIER	`/conf/core/tc-op`
REQUIREMEI	Requirement 1: `/req/core/tc-op`
TEST PURPOSE	With a provided template, validate that tiles are available for the right GET request.
TEST METHOD	1. Validate that tiles making up a tileset containing available data are available as an HTTP GET request to a URI that is built from a template containing three variables (such as `{tileMatrix}`, `{tileRow}` and `{tileCol}` as defined by the tileset conformance class, or `{z}`, `{y}` and `{x}`). The URI is obtained by substituting the variables by their respective valid values. 2. Validate that the variables in the URL correspond to the tile matrix, tile row and tile column of a particular tile matrix set as defined by the 2D Tile Matrix Set standard. 3. Validate that the API provides a mechanism for obtaining the template and associating the variables to their respective meaning, for example by implementing the tileset conformance class, or through an API definition.

NOTE: In practice, to test for conformance to the Core conformance class, the user of the test should provide a TileMatrixSet definition, a URL template (that contains the endpoint for tiles) with specific variable names, and a range of valid values for those variables and building the URLs. Once the information has been provided, the user can then execute the test.

A.1.2.2. Parameter tileMatrix

ABSTRACT TEST A.3	
IDENTIFIER	`/conf/core/tc-tilematrix-definition`
REQUIREMEI	Requirement 2: `/req/core/tc-tilematrix-definition`
TEST PURPOSE	Validate that there is a tileMatrix definition.

ABSTRACT TEST A.3	
TEST METHOD	1. If the API implements *OGC API — Common — Part 1: Core*, validate that the definition of a tile operation contains a mandatory string parameter `tileMatrix`.

A.1.2.3. Parameter tileRow

ABSTRACT TEST A.4	
IDENTIFIER	`/conf/core/tc-tilerow-definition`
REQUIREMEI	Requirement 3: `/req/core/tc-tilerow-definition`
TEST PURPOSE	Validate that there is a tileRow definition.
TEST METHOD	1. If the API implements *OGC API — Common — Part 1: Core*, validate that the definition of a tile operation contains a mandatory integer parameter `tileCol`.

A.1.2.4. Parameter tileCol

ABSTRACT TEST A.5	
IDENTIFIER	`/conf/core/tc-tilecol-definition`
REQUIREMEI	Requirement 4: `/req/core/tc-tilecol-definition`
TEST PURPOSE	Validate that there is a tileCol definition.
TEST METHOD	1. If the API implements *OGC API — Common — Part 1: Core*, validate that the definition of a tile operation contains a mandatory integer parameter `tileCol`.

A.1.2.5. Response

ABSTRACT TEST A.6	
IDENTIFIER	`/conf/core/tc-success`
REQUIREMEI	Requirement 5: `/req/core/tc-success`

ABSTRACT TEST A.6	
TEST PURPOSE	Validate that a successful execution with data responds with an HTTP status code `200`, the format is consistent with the requested and the elements represented in the tile are the ones present in the geographical area.
TEST METHOD	1. Validate that a successful execution of the operation with content responds with a HTTP status code `200`. 2. Validate that the content of that response is consistent with the format requested via HTTP content negotiation and represents elements inside or intersecting with the spatial extent of the geographical area of the tile identified by the tile matrix, tile row and tile column of the tileset's tile matrix set.

A.1.2.6. Error conditions

ABSTRACT TEST A.7	
IDENTIFIER	`/conf/core/tc-error`
REQUIREMENT	Requirement 6: `/req/core/tc-error`
TEST PURPOSE	Validate that the request for a tile that is out-of-range or empty, responds with the right content
TEST METHOD	1. If the path parameter values `tileMatrix`, `tileRow`, `tileCol` for a tile request are out-of-range (outside the tile matrix set or tile matrix set limits of the resource), validate that the HTTP response is a status code `404` or a `400`. 2. If the tile has no content due to lack of data in the area, but is within the data resource's tile matrix sets and tile matrix sets limits, validate that the HTTP response is a status code either 204 (indicating an empty tile with no content) or a 200 with the content of a blank response compatible with the requested media type (which may or may not be zero bytes long, depending on the output format; e.g., a 0 byte file if valid for the requested format, or a PNG with internal headers and no "pixels").

A.2. Conformance Class "Tileset"

CONFORMANCE CLASS A.2	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tileset`
SUBJECT	Requirements Class "Tileset"
TARGET TYPE	Web API

A.2.1. Tileset resource

A.2.1.1. Response

ABSTRACT TEST A.8	
IDENTIFIER	`/conf/tileset/description`
REQUIREMENT	Requirement 8: `/req/tileset/description`
TEST PURPOSE	Validate the content of a tileset description
TEST METHOD	1. Validate that the tileset endpoint SHALL support negotiation for an application/json response. In this case, a successful response of a HTTP GET for a specific tileset is encoded following the data model and JSON schema for tileset metadata, as defined by the 2D Tile Matrix Set and Tileset Metadata standard 2.0. 2. If the tileset endpoint also supports negotiation for an application/xml response, validate that a successful response of a HTTP GET for a specific tileset is encoded following the data model and XML schema for tileset metadata, as defined by the 2D Tile Matrix Set and Tileset Metadata standard 2.0. 3. If the tileset uses a TileMatrixSet registered in a TileMatrixSet register that is published through an accessible registry (e.g. the OGC TileMatrixSet register), validate that the `tileMatrixSetURI` property links to the registered TileMatrixSet (e.g. `http://www.opengis.net/def/tilematrixset/{tileMatrixSet}`). 4. Validate that the links property includes a link to the TileMatrixSet definition with relation type `http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme` following the tile matrix set schema, as defined by the 2D Tile Matrix Set and Tile Set Metadata standard 2.0. 5. Validate that the tileset metadata includes at least one templated link to individual tiles using the relation type `item`, and the template parameters `{tileMatrix}`, and `{tileRow}` and `{tileCol}`. Those variables are to be substituted by their respective valid values to obtain the URL to a tile. 6. If a tiles link template is specific to a particular format, validate that contains the media type for that format in the "`type`" property. Otherwise, normal HTTP content type negotiation rules apply (`Accept:` header). 7. Validate that a property `templated` with a boolean `true` value is part of the link properties to indicate that the link needs to be processed to substitute the templated variables with valid values before being used as a URL to a tile.

A.3. Conformance Class "Tilesets List"

CONFORMANCE CLASS A.3	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tilesets-list`
SUBJECT	Requirements Class “TileSets List”
TARGET TYPE	Web API

A.3.1. Tilesets list

A.3.1.1. Response

ABSTRACT TEST A.9	
IDENTIFIER	`/conf/tilesets-list/tileset-links`
REQUIREMEI	Requirement 10: `/req/tilesets-list/tileset-links`
TEST PURPOSE	Validate that the content of a tileset list
TEST METHOD	1. Validate that the API supports a GET operation on a …`/tiles` path returning a list of available tilesets 2. Validate that a successful response of a HTTP GET consists of a list of available tilesets each element containing a subset of the tileset metadata (as defined by the 2D Tile Matrix Set and Metadata standard), consisting of: `dataType, `` `crs``, a link to the tileset (with rel: self), a link to the TileMatrixSet defintion (with rel: `http://www.opengis.net/def/rel/ogc/1.0/tiling-scheme`) and `tileMatrixSetURI` (if the TMS is available in a registry). 3. Validate that each element of that list includes a link to a resource providing the full version of the tileset metadata, using link relation `self`. 4. Validate that the tileset-list endpoint supports negotiating an application/json response. In this case, validate that each tileset in the successful response of a HTTP GET is encoded following the data model and JSON schema for tileset metadata, as defined by the 2D Tile Matrix Set and Tileset Metadata standard 2.0. 5. If the tileset-list endpoint also supports negotiating an application/xml response, validate that each tileset in the successful response of a HTTP GET is encoded following the data model and XML schema for tileset metadata, as defined by the 2D Tile Matrix Set and Tileset Metadata standard 2.0. 6. Validate that the tileset is available as a URI that is composed by three parts: The first part is the URL of the geospatial resource (e.g., ‘…/map’) followed by the word ‘/tiles’, and finally followed by the id of the tile matrix set supported.

A.4. Conformance Class “Dataset Tilesets”

CONFORMANCE CLASS A.4	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/dataset-tilesets`
SUBJECT	Requirements Class “Dataset Tilesets”
TARGET TYPE	Web API

A.4.1. Web API landing page

A.4.1.1. Response

ABSTRACT TEST A.10	
IDENTIFIER	`/conf/dataset-tilesets/landingpage`
REQUIREMEI	Requirement 11: `/req/dataset-tilesets/landingpage`
TEST PURPOSE	Validate that it is possible to retrieve the tilesets list for the dataset
TEST METHOD	1. If the API has a mechanism to expose root resources (e.g., a landing page), validate that the API advertises at least one URI for retrieving a tilesets list that is provided by this service with a link having a `rel` value: `http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector`, `http://www.opengis.net/def/rel/ogc/1.0/tilesets-map` or `http://www.opengis.net/def/rel/ogc/1.0/tilesets-coverage`.

A.4.2. Dataset tilesets

A.4.2.1. Operation

ABSTRACT TEST A.11	
IDENTIFIER	`/conf/dataset-tilesets/operation`

ABSTRACT TEST A.11	
REQUIREMEI	Requirement 12: `/req/dataset-tilesets/operation`
TEST PURPOSE	Validate that there is an operation to get the tilesets list.
TEST METHOD	1. Validate that the dataset resource (the root resource) has at least one tileset accessible at …/`tiles` supporting an HTTP GET operation 2. Validate that the URI is composed of two parts: the initial part is the URI of the dataset resource (the root resource) that can be represented as tiles and the final part follows the pattern `/tiles`

A.5. Conformance Class “GeoData Tilesets”

CONFORMANCE CLASS A.5	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/geodata-tilesets`
SUBJECT	Requirements Class “GeoData Tilesets”
TARGET TYPE	Web API

A.5.1. Geospatial data resources

ABSTRACT TEST A.12	
IDENTIFIER	`/conf/geodata-tilesets/desc-links`
REQUIREMEI	Requirement 13: `/req/geodata-tilesets/desc-links`
TEST PURPOSE	Validate that it is possible to retrieve the tilesets list for the geospatial resource
TEST METHOD	1. If the API has a mechanism for their geospatial data resources to expose links to geospatial aspects (e.g., feature items, metadata…), validate that the API includes at least one of three `link` with the `href` pointing to tilesets list for geospatial data resources and with rel: `http://www.opengis.net/def/rel/ogc/1.0/tilesets-vector`, `http://www.opengis.net/def/rel/ogc/1.0/tilesets-map` and `http://www.opengis.net/def/rel/ogc/1.0/tilesets-coverage`.

A.5.2. Geospatial data resources tilesets list

A.5.2.1. Tilesets path

ABSTRACT TEST A.13	
IDENTIFIER	`/conf/geodata-tilesets/operation`
REQUIREMEI	Requirement 14: `/req/geodata-tilesets/operation`
TEST PURPOSE	Validate that there is an operation to get the tilesets list.
TEST METHOD	1. Validate that the geospatial data resource has an associated list of at least one tileset accessible at …`/tiles` supporting an HTTP GET operation 2. Validate that the URI is composed of two parts: the initial part is the URI of the geospatial data resource that can be represented as tiles and the final part follows the pattern `/tiles`

A.6. Conformance Class “Collections Selection”

CONFORMANCE CLASS A.6	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/collections-selection`
SUBJECT	Requirements Class “Collections Selection”
TARGET TYPE	Web API

A.6.1. Operation

A.6.1.1. Parameter `collections`

ABSTRACT TEST A.14	
IDENTIFIER	`/conf/collections-selection/query-collections`
REQUIREMENT	Requirement 15: `/req/collections-selection/query-collections`
TEST PURPOSE	Validate support of the optional parameter `collections`.
TEST METHOD	Given: A tileset for an origin consisting of multiple geospatial data sub-resources (e.g., described in the tileset metadata as being comprised of multiple layers). When: Requesting resources either for the tileset or for individual tiles (e.g., a dataset tileset at `{datasetAPI}/tiles/WebMercatorQuad` and dataset tiles at `{datasetAPI}/tiles/WebMercatorQuad/{tileMatrix}/{tileRow}/{tileCol}`), with a `collections` query parameter consisting of a comma-separated list of collectionIDs (consistent with the collection identifiers making up that tileset, as described for example, in the tileset metadata layers). Then: Assert that the parameter is accepted for valid lists of collectionIDs. Note that servers are free to restrict valid combinations of collectionIDs (permission /per/collections-selection/valid-collections). Tests should be restricted to few collections e.g., between one and five. If the sub-collections are also available individually, only collections advertising support for the same TileMatrixSet, CRS and encodings as used for the request should be selected.

A.6.1.2. Response

ABSTRACT TEST A.15	
IDENTIFIER	`/conf/collections-selection/collections-response`
REQUIREMENT	Requirement 16: `/req/collections-selection/collections-response`
TEST PURPOSE	Validate response when using `collections` parameter.
TEST METHOD	Given: Requests described in the previous abstract test (query-collections) When: The request is successful Then: Assert that the response to tilesets and tiles requests only include the selected collections. For tilesets, verify that only the selected collections are included in the layers. For tiles, verify that the content is limited to the data pertaining to the selected collections. If more than one collection is selected and no style applied specifies otherwise, validate that the selected collections are included in the response starting by the first (leftmost in the comma-separated list) and ending by the last (rightmost). In map tiles, this will result in the first collection

ABSTRACT TEST A.15	
	being portrayed at the bottom and the others are rendered on top of the previous ones, one by one (the rightmost collection will become topmost in the portrayal).

A.7. Conformance Class “DateTime”

CONFORMANCE CLASS A.7	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/datetime`
SUBJECT	Requirements Class “Datetime”
TARGET TYPE	Web API

A.7.1. `datetime` query parameter request and response

ABSTRACT TEST A.16	
IDENTIFIER	`/conf/collections/rc-op-datetime`
REQUIREMEN	Requirement 17: `/req/collections/rc-datetime-definition` Requirement 18: `/req/collections/rc-datetime-response`
TEST PURPOSE	Validate that resources can be identified and extracted from an API server using the datetime query parameter.
TEST METHOD	1. Select a valid datetime value which intersects a subset of the resource collections available through the API implementation. 2. Construct a datetime query parameter using the selected value. 3. Validate the datetime query parameter using /conf/collections/rc-datetime-definition 4. Issue an HTTP GET request to the URL `{root}/collections`. Include the validated datetime query parameter. 5. Validate that a document was returned with a status code 200 6. Validate the contents of the returned document using: a) http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections/rc-md-success and b) /conf/collections/rc-datetime-response and c) http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections/rc-paged-response.

ABSTRACT TEST A.17	
IDENTIFIER	/conf/collections/rc-datetime-definition
REQUIREMEI	Requirement 17: /req/collections/rc-datetime-definition
TEST PURPOSE	Validate that the dateTime query parameter is constructed correctly.

TEST METHOD	Verify that the datetime query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: datetime`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

ABSTRACT TEST A.18	
IDENTIFIER	/conf/collections/rc-datetime-response
REQUIREMENTS	Requirement 17: /req/collections/rc-datetime-definition Requirement 18: /req/collections/rc-datetime-response
TEST PURPOSE	Validate that the datetime query parameter is processed correctly.
TEST METHOD	DO FOR each Collection in the collections element of the response: 1. Extract the temporal geometry from the interval element of the extent property of the Collection resource. 2. IF there is a temporal geometry, verify that the temporal geometry intersects the temporal period defined by the datetime parameter. 3. IF there is a temporal geometry, validate that the processing of the datetime parameter complies with the syntax described in /req/collections/rc-datetime-definition (B, C, and D).

A.7.2. `subset=datetime` query parameter request and response

ABSTRACT TEST A.19	
IDENTIFIER	/conf/collections/rc-op-subset
REQUIREMEN	Requirement 19: /req/collections/rc-subset-definition Requirement 20: /req/collections/rc-subset-response

ABSTRACT TEST A.19

TEST PURPOSE	Validate that resources can be identified and extracted from an API server using the subset query parameter.
TEST METHOD	1. Select a valid subset value which intersects a subset of the resource collections available through the API implementation. 2. Construct a subset query parameter using the selected dimension name and value. 3. Validate the subset query parameter using /conf/collections/rc-subset-definition 4. Issue an HTTP GET request to the URL `{root}/collections`. Include the validated subset query parameter. 5. Validate that a document was returned with a status code 200 6. Validate the contents of the returned document using: a) http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections/rc-md-success and b) /conf/collections/rc-subset-response and c) http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections/rc-paged-response.

ABSTRACT TEST A.20

IDENTIFIER	`/conf/collections/rc-subset-definition`
REQUIREMEI	Requirement 19: `/req/collections/rc-subset-definition`
TEST PURPOSE	Validate that the subset query parameter is constructed correctly.
TEST METHOD	Verify that the `subset` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment): (see listing below)

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

  Where:
     \" = double quote = ASCII code 0x42,
     {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).
```

ABSTRACT TEST A.21

IDENTIFIER	`/conf/collections/rc-subset-response`
REQUIREMENTS	Requirement 19: `/req/collections/rc-subset-definition`

ABSTRACT TEST A.21	
	Requirement 20: `/req/collections/rc-subset-response` /req/collections/rc-subset-collection-response
TEST PURPOSE	Validate that the `subset` query parameter is processed correctly.
TEST METHOD	DO FOR each `Collection` in the `collections` element of the response: 1. Extract the dimension values from the `interval` element of the `extent` property of the `Collection` resource. 2. If the dimension is there, verify that the dimension interval intersects the dimension period defined by the `subset` parameter. 3. If the dimension is there, validate that the processing of the subset parameter complies with the syntax described in /req/collections/rc-subset-definition.

ABSTRACT TEST A.22	
IDENTIFIER	`/conf/datetime/axis`
REQUIREMENT	Requirement 21: `/req/datetime/axis`
TEST PURPOSE	Validate that a request of a subset of a datetime dimension is supported by the server
TEST METHOD	1. Validate the server support "datetime" as axisname in the subset parameter to subset the generic datetime dimension.

A.8. Conformance Class "OpenAPI Specification 3.0"

CONFORMANCE CLASS A.8	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/oas30`
SUBJECT	Requirements Class "OpenAPI Specification 3.0"
TARGET TYPE	Web API

A.8.1. Web API OpenAPI definition response

ABSTRACT TEST A.23	
IDENTIFIER	/conf/oas30/completeness
REQUIREMEI	Requirement 22: /req/oas30/completeness
TEST PURPOSE	Validate OpenAPI completeness and consistency
TEST METHOD	1. Validate that OpenAPI definition describes all tileset, tilesets list and tile resources provided by the API instance. 2. Validate that the OpenAPI paths are consistent with the links provided by the landing page, tileset and tilesets list resources. 3. Validate that the OpenAPI paths provide the description of the parameters that the tileset and tile resources need to operate as specified in the corresponding conformance classes.

ABSTRACT TEST A.24	
IDENTIFIER	/conf/oas30/operation-id
REQUIREMEI	Requirement 23: /req/oas30/operation-id
TEST PURPOSE	Validate use of operationID suffixes in OpenAPI definition
TEST METHOD	1. Validate that at least one instance of all path operations defined in the supported conformance classes can be discovered in the API definition by identifying them based on the operationIDs suffixes specified in Table 11.

A.9. Conformance Class "XML Tileset Metadata"

CONFORMANCE CLASS A.9	
IDENTIFIER	http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/xml
SUBJECT	Requirements Class "XML Tileset Metadata"
TARGET TYPE	Web API

A.9.1. XML encoding

ABSTRACT TEST A.25	
IDENTIFIER	`/conf/xml/definition`
REQUIREMENT	Requirement 24: `/req/xml/definition`
TEST PURPOSE	Validate the support of the required media type
TEST METHOD	1. Validate that `200`-responses of the server support the `application/xml` media type for the Tilesets list and TileSet resources.

ABSTRACT TEST A.26	
IDENTIFIER	`/conf/xml/content`
REQUIREMENT	Requirement 25: `/req/xml/content`
TEST PURPOSE	Validate the particularities of an XML response
TEST METHOD	1. Validate that every request to a TileSets list or TileSet resource which: Receives a `200`-response and with the `Content-Type` header set to `application/xml` includes, or links to, a payload encoded according to the Extensible Markup Language (XML) 1.0 2. Validate that the payload for these responses conforms with the XML Schema specified for the resource in the OGC 17-083r4: OGC Two Dimensional Tile Matrix Set and Tile Set Metadata

A.10. Conformance Classes for tile encodings

A.10.1. Conformance Class “PNG”

CONFORMANCE CLASS A.10	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/png`
SUBJECT	Requirements Class “PNG”

CONFORMANCE CLASS A.10	
TARGET TYPE	Web API

A.10.1.1. PNG encoding

ABSTRACT TEST A.27	
IDENTIFIER	`/conf/png/content`
REQUIREMEI	Requirement 26: `/req/png/content`
TEST PURPOSE	Validate the particularities of a PNG response
TEST METHOD	1. Validate that every 200-response of the server with the media type image/png contains a PNG document representing only one tile. 2. Validate that the colors of the PNG represent the geospatial features or coverage values in the tile. 3. Validate that the alpha channel of the PNG is used when partial transparency is required. 4. Validate that all tiles representing parts of the same resource or resources, and using the same style, are following the same portrayal rules.

A.10.2. Conformance Class “JPEG”

CONFORMANCE CLASS A.11	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/jpeg`
SUBJECT	Requirements Class “JPEG”
TARGET TYPE	Web API

A.10.2.1. JPEG Encoding

ABSTRACT TEST A.28	
IDENTIFIER	`/conf/jpeg/content`
REQUIREMEI	Requirement 27: `/req/jpeg/content`
TEST PURPOSE	Validate the particularities of a JPEG response
TEST METHOD	1. Validate that every 200-response of the server with the media type image/jpeg contains a JPEG document representing only one tile. 2. Validate that the colors of the JPEG represent coverage values or geospatial features in the tile. 3. Validate that all tiles representing parts of the same resource or resources using the same style are following the same portrayal rules.

A.10.3. Conformance Class “TIFF”

CONFORMANCE CLASS A.12	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/tiff`
SUBJECT	Requirements Class “TIFF”
TARGET TYPE	Web API

A.10.3.1. TIFF encoding

ABSTRACT TEST A.29	
IDENTIFIER	`/conf/tiff/content`
REQUIREMEI	Requirement 28: `/req/tiff/content`
TEST PURPOSE	Validate the particularities of a TIFF response
TEST METHOD	1. Validate that every 200-response of the server with the media type image/tiff contains a TIFF document representing only one tile 2. Validate that the TIFF file is organized in strips (avoiding organization in internal tiles). 3. Validate that all tiles representing parts of the same resource or resources and using the same style follows the same portrayal rules or represent data with the same reference and units of measure.

ABSTRACT TEST A.30	
IDENTIFIER	`/conf/tiff/geotiff`
REQUIREMEI	Requirement 29: `/req/tiff/geotiff`
TEST PURPOSE	Validate GeoTIFF consistency of the georeference
TEST METHOD	1. If the TIFF encoding incorporates a GeoTIFF georeference, validate that this information is consistent with the TileMatrixSet, TileMatrix, TileRow and TileCol

A.10.4. Conformance Class “NetCDF”

CONFORMANCE CLASS A.13	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/netcdf`
SUBJECT	Requirements Class “NetCDF”
TARGET TYPE	Web API

A.10.4.1. NetCDF encoding

ABSTRACT TEST A.31	
IDENTIFIER	`/conf/netcdf/content`
REQUIREMEI	Requirement 30: `/req/netcdf/content`
TEST PURPOSE	Validate the particularities of a NetCDF response
TEST METHOD	1. Validate that every 200-response of the server with the media type application/netcdf or application/x-netcdf contains a NetCDF document representing only one tile 2. Validate that the NetCDF file contains only data in two or more dimensions

ABSTRACT TEST A.32	
IDENTIFIER	`/conf/netcdf/geo`
REQUIREMENT	Requirement 31: `/req/netcdf/geo`
TEST PURPOSE	Validate NetCDF consistency of the georeference
TEST METHOD	1. Validate that the NetCDF encoding incorporates a georeference, this information is consistent with the TileMatrixSet, TileMatrix, TileRow and TileCol

A.10.5. Conformance Class “GeoJSON”

CONFORMANCE CLASS A.14	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/geojson`
SUBJECT	Requirements Class “GeoJSON”
TARGET TYPE	Web API

A.10.5.1. GeoJSON Encoding

ABSTRACT TEST A.33	
IDENTIFIER	`/conf/geojson/content`
REQUIREMENT	Requirement 32: `/req/geojson/content`
TEST PURPOSE	Validate the particularities of a GeoJSON response
TEST METHOD	1. Validate that every 200-response of the server with the media type application/geo+json contains a GeoJSON document representing only one tile. 2. Validate that root of the GeoJSON document is a GeoJSON FeatureCollection Object for features, or a GeoJSON Feature Object for a single feature. 3. Validate that regardless of the TileMatrixSet CRS, the geometry coordinates are in CRS84 as stated in the GeoJSON standard unless a *prior arrangement* applies to use an alternative coordinate reference system. The fact that a TileMatrixSet is used does not constitute a *prior arrangement*. For example, an extension could negotiate another CRS for coordinates with a query parameter.

A.10.6. Conformance Class “Mapbox Vector Tiles”

CONFORMANCE CLASS A.15	
IDENTIFIER	`http://www.opengis.net/spec/ogcapi-tiles-1/1.0/conf/mvt`
SUBJECT	Requirements Class “Mapbox Vector Tiles”
TARGET TYPE	Web API

A.10.6.1. Mapbox Vector Tiles Encoding

ABSTRACT TEST A.34	
IDENTIFIER	`/conf/mvt/content`
REQUIREMEI	Requirement 33: `/req/mvt/content`
TEST PURPOSE	Validate the particularities of a Mapbox Vector Tiles response
TEST METHOD	1. Validate that every 200-response of the server with the media type application/vnd.mapbox-vector-tile contains a Mapbox Vector Tile document representing only one tile. 2. Validate that a feature contains a geometry field. Validate that a feature contains a type field as described in the Geometry Types section. 3. Validate that the grid space for the MVT coordinates maps linearly to the coordinates in the tile extent expressed in the Tileset CRS unless a *prior arrangement* applies to use an alternative CRS; and in this case the linear mapping is done to the alternative CRS and not with the Tileset CRS. For example, an extension could negotiate another CRS for coordinates with a query parameter. In particular the 0,0 coordinate in the MVT maps to the top-left corner of the tile. The bottom-right corner of the tile corresponds the bottom-right corner of the MVT grid.

B

ANNEX B (INFORMATIVE) REVISION HISTORY

B

ANNEX B (INFORMATIVE) REVISION HISTORY

DATE	RELEASE	EDITOR	PRIMARY CLAUSES MODIFIED	DESCRIPTION
2019-03-21	Template	C. Heazel	all	initial template
2020-04-15	0.0.1	J. Maso	all	Several
2020-04-21	0.0.2	J. Maso	all	Several
2020-05-21	0.0.3	G. Hobona	Annex A	Fixed Conformance Class URI and added abstract tests
2022-03-30	0.8.0	J. Maso & J. St-Louis	all	Candidate standard version for public comments
2022-06-16	0.9.0	J. Maso & J. St-Louis	all	Initial version presented to the TC for approval
2022-07-14	1.0.0 draft 1	J. Maso & J. St-Louis	several	Editorial fixes; synchronized OpenAPI schemas with 2DTMS & TileSet metadata standard
2022-09-12	1.0.0 draft 2	G. Hobona	all	Conversion to metanorma
2022-09-15	1.0.0	J. Maso & J. St-Louis	several	Initial version 1.0
2022-10-24	1.0.0	G. Hobona	several	OGC Staff editorial review
2022-12-05	1.0.0	C. Reed, J. Maso, J. St-Louis & G. Hobona	multiple	This release of version 1.0 fixes minor non-normative issues that were in the 2022-11-10 release of version 1.0

BIBLIOGRAPHY

BIBLIOGRAPHY

This standard is deeply inspired by concepts defined in the following documents that preceded it. This standard offers an alternative interface to fulfill similar tasks described in these references:

1. Cavazzi S.: OGC Testbed-13: Vector Tiles Engineering Report (OGC 17-041), https://docs.ogc.org/per/17-041.html

2. Masó J.: Map Tile Service (WMTS) Simple Profile, version 1.0 (OGC 13-082r2), http://docs.ogc.org/is/13-082r2/13-082r2.html

3. Meek S.: OGC Vector Tiles Pilot: Summary Engineering Report (OGC 18-086r1), https://docs.ogc.org/per/18-086r1.html

4. Taleisnik S.: OGC Vector Tiles Pilot 2: Tile Set Metadata Engineering Report (OGC 19-082r1), http://docs.ogc.org/per/19-082r1.html#_metadatadate

5. Vretanos P.A.: OGC Vector Tiles Pilot: WFS 3.0 Vector Tiles Extension Engineering Report (OGC 18-078), https://docs.ogc.org/per/18-078.html

6. Vretanos P.A.: OGC Vector Tiles Pilot: WMTS Vector Tiles Extension Engineering Report (OGC 18-083), https://docs.ogc.org/per/18-083.html

7. Yutzler J.: Vector Tiles Pilot Extension Engineering Report (OGC 18-101), http://docs.ogc.org/per/18-101.html

8. W3C/OGC: Spatial Data on the Web Best Practices, W3C Working Group Note 28 September 2017, https://www.w3.org/TR/sdw-bp/

9. W3C: Data on the Web Best Practices, W3C Recommendation 31 January 2017, https://www.w3.org/TR/dwbp/

10. Masó J., Pomakis K. and Julià N.: Web Map Tile Service (WMTS), version 1.0. (OGC 07-057r7) http://portal.ogc.org/files/?artifact_id=35326

11. de la Beaujardiere J.: Web Map Service (WMS), version 1.3.0 (OGC 06-042), http://portal.ogc.org/files/?artifact_id=14416

12. Hobona G, Idol, T.: OGC Vector Tiles Pilot 2: Summary Engineering Report (OGC 19-088r2), http://docs.ogc.org/per/19-088r2.html

13. IANA: Link Relation Types, https://www.iana.org/assignments/link-relations/link-relations.xml

14. Ingensand J. and Maia K.: OGC Vector Tiles Pilot: Tiled Feature Data Conceptual Model Engineering Report (OGC 18-076), https://docs.ogc.org/per/18-076.html

15. Mapbox: Mapbox Vector Tile specification. https://github.com/mapbox/vector-tile-spec

16. Masó J.: OGC Testbed-15: Maps and Tiles API Engineering Report (OGC 19-069), https://docs.ogc.org/per/19-069.html