# Table of Contents

1. Scope ........................................................................................................... 7
2. Conformance .................................................................................................. 8
3. References ..................................................................................................... 9
4. Terms and definitions .................................................................................. 10
  4.1. Abbreviated terms .................................................................................. 11
5. Conventions .................................................................................................. 12
  5.1. Identifiers ............................................................................................... 12
  5.2. Link relations ......................................................................................... 12
  5.3. Use of HTTPS ......................................................................................... 13
  5.4. HTTP URIs ............................................................................................ 13
  5.5. API definition ........................................................................................ 13
     5.5.1. General remarks ........................................................................... 13
     5.5.2. Role of OpenAPI ........................................................................... 14
     5.5.3. References to OpenAPI components in normative statements ........ 14
     5.5.4. Paths in OpenAPI definitions ......................................................... 14
     5.5.5. Reusable OpenAPI components ..................................................... 15
6. Overview ........................................................................................................ 16
  6.1. Evolution from OGC Web Services ......................................................... 16
  6.2. Tiles and maps ....................................................................................... 17
  6.3. How to approach an OGC API ............................................................... 18
7. Requirement Class "Tiles Core" .................................................................... 21
  7.1. Overview ............................................................................................... 21
  7.2. General ................................................................................................... 22
  7.3. API landing page .................................................................................... 22
     7.3.1. Response ....................................................................................... 23
  7.4. Declaration of conformance classes ....................................................... 23
     7.4.1. Response ....................................................................................... 23
  7.5. Collections ............................................................................................. 23
  7.6. Collection ................................................................................................ 24
     7.6.1. Collection Links ............................................................................ 24
  7.7. Tiles description ...................................................................................... 25
     7.7.1. Operation ....................................................................................... 25
     7.7.2. Response ....................................................................................... 25
  7.8. Tiled data from one collection ................................................................. 29
     7.8.1. Operation ....................................................................................... 29
     7.8.2. Parameter tileMatrixSetId .............................................................. 29
     7.8.3. Parameter tileMatrix ...................................................................... 29
     7.8.4. Parameter tileRow ......................................................................... 30
<table>
<thead>
<tr>
<th>Requirement Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Tiles Multi-tiles&quot;</td>
<td>53</td>
</tr>
<tr>
<td>Overview</td>
<td>53</td>
</tr>
<tr>
<td>Declaration of conformance classes</td>
<td>53</td>
</tr>
<tr>
<td>Response</td>
<td>53</td>
</tr>
<tr>
<td>Tiles description</td>
<td>54</td>
</tr>
<tr>
<td>Response</td>
<td>54</td>
</tr>
<tr>
<td>Multiple tiles from one collection</td>
<td>55</td>
</tr>
<tr>
<td>Operation</td>
<td>55</td>
</tr>
<tr>
<td>Parameter tileMatrixSetId</td>
<td>56</td>
</tr>
<tr>
<td>Parameter bbox</td>
<td>56</td>
</tr>
<tr>
<td>Parameter scaleDenominator</td>
<td>58</td>
</tr>
<tr>
<td>Parameter multiTileType</td>
<td>58</td>
</tr>
<tr>
<td>Formats</td>
<td>60</td>
</tr>
<tr>
<td>Response</td>
<td>60</td>
</tr>
<tr>
<td>Error conditions</td>
<td>64</td>
</tr>
<tr>
<td>&quot;Tiles Collections Multi-tiles&quot;</td>
<td>65</td>
</tr>
<tr>
<td>Overview</td>
<td>65</td>
</tr>
<tr>
<td>API landing page</td>
<td>65</td>
</tr>
<tr>
<td>Declaration of conformance classes</td>
<td>65</td>
</tr>
<tr>
<td>Response</td>
<td>65</td>
</tr>
<tr>
<td>Tiles description</td>
<td>66</td>
</tr>
<tr>
<td>Response</td>
<td>66</td>
</tr>
<tr>
<td>Multiple tiles from more than one collection</td>
<td>67</td>
</tr>
<tr>
<td>Operation</td>
<td>67</td>
</tr>
<tr>
<td>Parameter tileMatrixSetId</td>
<td>67</td>
</tr>
<tr>
<td>Parameter bbox</td>
<td>68</td>
</tr>
<tr>
<td>Parameter scaleDenominator</td>
<td>70</td>
</tr>
<tr>
<td>Parameter multiTileType</td>
<td>70</td>
</tr>
<tr>
<td>Parameter Collections</td>
<td>72</td>
</tr>
<tr>
<td>Formats</td>
<td>72</td>
</tr>
<tr>
<td>Response</td>
<td>73</td>
</tr>
<tr>
<td>Error conditions</td>
<td>75</td>
</tr>
<tr>
<td>&quot;Maps Core&quot;</td>
<td>76</td>
</tr>
<tr>
<td>Overview</td>
<td>76</td>
</tr>
<tr>
<td>General</td>
<td>76</td>
</tr>
<tr>
<td>API landing page</td>
<td>77</td>
</tr>
<tr>
<td>Response</td>
<td>77</td>
</tr>
<tr>
<td>Declaration of conformance classes</td>
<td>77</td>
</tr>
</tbody>
</table>
OGC Testbed-15: Maps and Tiles API Engineering Report

COPYRIGHT

Copyright © 2020 Open Geospatial Consortium. To obtain additional rights of use, visit http://www.opengeospatial.org/

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Public Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.
LICENSE AGREEMENT

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

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

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

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

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to
indicate compliance with any LICENSOR standards or specifications.

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

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.
i. Abstract

In 2017 the OGC began a focused effort to develop Application Programming Interface (API) standards that support the Resource Oriented Architecture and make use of the OpenAPI specification. As part of this effort, this OGC Testbed 15 Engineering Report (ER) defines a proof-of-concept of an API specification for maps and tiles.

The OGC API Maps and Tiles draft specification described in this ER builds on the precedent of the OGC API - Features - Part 1: Core standard. The OGC API - Tiles draft specification describes a service that retrieves data representations as tiles, which are generally small compared with the geographic extent of the data. In the draft specification, the assumption is that tiles are organized into Tile Matrix Sets consisting of regular tile matrices available at different scales or resolutions. The OGC API – Tiles draft specification is described as a building block that can be plugged into an OGC API - Features service to retrieve tiled feature data (sometimes called vector tiles) or to an OGC API – Maps implementation to retrieve rendered tiles (sometimes called map tiles). In the future, the OGC API - Tiles draft specification could extend other specifications, one possible candidate being the emerging OGC API – Coverages draft specification.

The OGC API - Maps draft specification describes an API that presents data as maps by applying a style. These maps can be retrieved in a tiled structure (if OGC API - Tiles is approved as an OGC Implementation Standard) or as maps of any size generated on-the-fly. The OGC API - Maps draft specification implements some functionality, specified in the Web Map Tile Service (WMTS) 1.0 standard, related to the use of styles by using the Styles API draft specification that was developed in the Testbed-15 Open Portrayal Framework thread.

The draft Maps and Tiles API specifications are designed in a modular way. With the exception of the core requirements, the other conformance classes describe functionality that can be considered optional characteristics that can be combined by server implementations at will.

At the beginning of Testbed-15, the original proposed title for this ER was “OGC Testbed-15: Web Map Tiling Service Draft Specification Engineering Report” but in the course of the Testbed-15 that title was changed to better represent the content.

i.i Executive Summary

This engineering report presents the OGC API – Maps draft specification, as well as the OGC API – Tiles draft specifications. The draft specifications described in this ER have been submitted to the OGC Web Map Service Standards Working Group (WMS SWG) as an important step towards the creation of two new OGC API standards for maps and tiles. The use of these new OGC API specifications will increase the chances that maps and tiles will be used for geospatially focused API work as well as for non-geospatial APIs that might need map visualization capabilities.

These draft specifications cover the following conformance classes:

- The tiles core conformance class describes how to retrieve tiles representing one collection in one of the eight tile matrix sets described in the OGC 17-083r2 Two Dimensional Tile Matrix Set.
- The tiles collections conformance class describes how to retrieve tiles representing more than one collection.
- The tiles tilermatrixset conformance class describes how to include a personalized tile matrix set
definition

• The tiles info conformance class proposes a mechanism to provide extra information about one point in a tile. This mechanism requires further elaboration and discussion in the WMS SWG.

• The tiles multitile conformance class describes how to get more than one tile in a single operation.

• The maps core conformance class describes how to transform a data resource into a map resource in the default style.

• The maps styles conformance class describes how to transform a data resource into a map resource applying a style.

• The map maps conformance class proposes a mechanism to deliver maps. This mechanism requires further elaboration and discussion in the WMS SWG.

• The maps collections conformance class describes how to transform a list of data resources (collections) into a map resource.

The WMS SWG is expected to consider these draft specifications as two separate documents. These documents are not OGC Standards. The documents were created as deliverables in an OGC testbed and are not an official position of the OGC membership. It is envisioned that the SWG will modify the requirements classes with the objective of achieving consensus in the community and be able to deliver an OGC API - Tiles Standard and an OGC API - Maps Standard.

ii. Keywords

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

ogcdoc, OGC document, WMS, WMTS, API, OpenAPI, OGC API, tile, map, portrayal, JSON, tileMatrixSetId, services, coverages

iii. Preface

The content of this Engineering Report (ER) is a response to the need to evolve the OGC Web Services architecture into Web APIs that can be integrated into web-based systems combined with non-geospatial functionality or to add geospatial functionality to existing APIs. The goal is to develop a single OGC API that is designed to allow services that can simultaneously serve maps and tiles. Another objective is to allow for both tiled feature data (sometimes called vector tiles) or rendered tiles (sometimes called map tiles) (or even other resource types, e.g. coverages) in the same specification.

Maps and tiles are described as a single document in this Engineering Report to demonstrate that it is possible to create a single service that serves both. Nevertheless, the document has been structured in a way that it will be easy to separate the OGC API – Maps specification from the OGC API - Tiles specification as that is considered appropriate in future OGC WMS SWG work.

Suggested additions, changes and comments on this standard are welcome and encouraged. Such suggestions may be submitted using the online change request form at the OGC Standards Tracker: http://ogc.standardstracker.org

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

UAB-CREAF

v. Submitters

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

Joan Masó, CREAF

vi. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

<table>
<thead>
<tr>
<th>Name</th>
<th>Organization</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joan Maso Pau</td>
<td>Universitat Autònoma de Barcelona (CREAF)</td>
<td>Editor</td>
</tr>
</tbody>
</table>

vii. Foreword

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.
Chapter 1. Scope

This OGC Testbed-15 ER provides an approach to a Web Map Tiling Service (WMTS) by combining a service that is able to serve tiled feature data as well as map tiles. To do this, the ER lays the foundations for two future OGC standards: OGC API - Maps and OGC API - Tiles. The OGC API - Tiles draft specification describes an API building block that can be applied to an implementation of the OGC API – Features standard to serve tiled feature data. The OGC API – Maps draft specification, which is partially defined in this ER, has been limited in scope to the parts that are necessary for enabling the OGC API - Tiles building block to generate map tiles.

These draft specifications are written in the context of the OGC API effort and abandon the OGC web services style (KVP, SOAP, etc.). The draft specifications adopt the new OGC Web API style where services use a resource-oriented architecture along with the standard web verbs and an OpenAPI description document. The draft specifications are documented for consideration by the OGC Standards Program.
Chapter 2. Conformance

**NOTE** This section will be populated by the SWG
Chapter 3. References

The following normative documents are referenced in this document.

- **OGC: OGC 10-100r3, Geography Markup Language (GML) simple features profile (with Corrigendum) - 2.0 (2011)** [http://portal.opengeospatial.org/files/?artifact_id=42729]

**NOTE**

This draft specification produced by the Testbed-15 project assumes that an OGC API - Common standard exists. This was a necessary assumption in the Testbed-15 project. However, by the time of finalizing this engineering report an initial OGC API - Common draft specification had been produced in the OWS Common SWG [1]. The authors of this document are assuming that some parts of the OGC API - Features standard will form part of a new OGC API - Commons in a near future with almost no variation, but with the text generalized to cover resources types other than features.
Chapter 4. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply. In addition, the following terms and definitions apply.

- **coordinate reference system**
  coordinate system that is related to the real world by a datum term name (source: ISO 19111)

- **dataset**
  collection of data (source: ISO 19115-1)

  NOTE [DCAT] defines a dataset as a collection of data, published or curated by a single agent, and available for access or download in one or more formats. The use of ‘collection’ in the definition from [DCAT] is broader than the use of the term collection in this specification.

- **distribution**
  represents an accessible form of a dataset [DCAT]

  EXAMPLE: a downloadable file, an RSS feed or an API.

- **collection**
  a set of elements from a dataset

- **element**
  entities that are part of a collection

- **layer**
  basic unit of geographic information that may be requested as a map from a server.

  NOTE In this draft specification, a layer identifier is assimilated to a collection identifier

- **map**
  portrayal of geographic information in a particular style as a digital representation file suitable for a display on a visualization device.

- **multi-tile**
  a mechanism to wrap or reference more than one tile, sharing the same tiling scheme, together. An example of a wrapping strategy is the encapsulation of several tiles in a ZIP file.

- **portrayal**
  presentation of information to humans (source: ISO 19117)

- **tile**
  a small rectangular representation of geographic data, often part of a set of such elements, covering a tiling scheme and sharing similar information content and graphical styling. A tile
can be uniquely defined in a tile matrix by one integer index in each dimension. Tiles are mainly used for fast transfer (particularly in the web) and easy display at the resolution of a rendering device. Tiles can be grid based pictorial representations, coverage subsets, or feature based representations (e.g., vector tiles). (source: OGC 17-083r2)

• **tile matrix**

  a grid tiling scheme that defines how space is partitioned into a set of conterminous tiles at a fixed scale (source: OGC 17-083r2).

  NOTE A tile matrix constitutes a tessellation of the space that resembles a matrix in a 2D space characterized by a matrix width (columns) and a matrix height (rows).

• **tile matrix set**

  a tiling scheme composed of collections of tile matrices defined at different scales covering approximately the same area and has a common coordinate reference system (source: OGC 17-083r2).

• **Web API**

  API using an architectural style that is founded on the technologies of the Web [DWBP]

  NOTE Best Practice 24: Use Web Standards as the foundation of APIs [https://www.w3.org/TR/dwbp/#APIHttpVerbs] in the W3C Data on the Web Best Practices provides more detail.

### 4.1. Abbreviated terms

- API Application Programming Interface
- CRS Coordinate Reference System
- JSON JavaScript Object Notation
- OWS OGC Web Services
- WCS Web Coverage Service
- WMS Web Map Service
- WMTS Web Map Tile Service
- YAML YAML Ain't Markup Language
Chapter 5. Conventions

5.1. Identifiers

The normative provisions in this draft specification are denoted by the URI http://www.opengis.net/spec/ogcapi-tiles-1/1.0 and http://www.opengis.net/spec/ogcapi-maps-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 registered link relation types are used in this document.

- **alternate**: Refers to a substitute for this context.
- **collection**: The target IRI points to a resource which represents the collection resource for the context IRI.
- **describedBy**: Refers to a resource providing information about the link’s context.
- **item**: The target IRI points to a resource that is a member of the collection represented by the context IRI
- **next**: Indicates that the link’s context is a part of a series, and that the next in the series is the link target.
- **license**: Refers to a license associated with this context.
- **prev**: Indicates that the link’s context is a part of a series, and that the previous in the series is the link target.
  - This relation is only used in examples.
- **self**: Conveys an identifier for the link’s context.
- **service-desc**: Identifies service description for the context that is primarily intended for consumption by machines.
  - API definitions are considered service descriptions.
- **service-doc**: Identifies service documentation for the context that is primarily intended for human consumption.

In addition, the following link relation types were identified in OGC API Common and are used for which no applicable registered link relation type could be identified.

- **items**: Refers to a resource that is comprised of members of the collection represented by the link’s context.
- **conformance**: Refers to a resource that identifies the specifications that the link’s context conforms to.
• **data**: Indicates that the link’s context is a distribution of a dataset that is an API and refers to the root resource of the dataset in the API.

The following relation types are needed in the OGC API Maps and Tiles draft specifications.

• **tiling-schemes**: Refers to tiling schemes used by this service.

• **tiles**: Refers to a resource that is represented as a collection of tiles of the resource represented by the context IRI

• **map**: Refers to a resource that is a rendered map representation of the resource represented by the context IRI

• **attributes**: Refers to a resource that provides values for some attributes of the resource represented by the context IRI

Each resource representation includes an array of links. Implementations are free to add additional links for all resources provided by the API. For example, an **enclosure** link could reference a bulk download of a collection. Or a **related** link on a feature could reference a related feature.

### 5.3. Use of HTTPS

For simplicity, this document 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 RFC2818 (HTTPS), not RFC2616 (HTTP).

### 5.4. HTTP URIs

This document does not restrict the lexical space of URIs used in the API beyond the requirements of the RFC2616 (HTTP) and RFC3986 (URI Syntax) IETF RFCs. If URIs include reserved characters that are delimiters in the URI subcomponent, these have to be percent-encoded. See Clause 2 of RFC3986 for details.

### 5.5. API definition

#### 5.5.1. General remarks

Good documentation is essential for every API so that developers can more easily learn how to use the API. In the best case, documentation will be available in HTML and in a format that can be processed by software to connect to the API.

This draft specification specifies requirements and recommendations for APIs that share maps and tiles and that want to follow a standard way of doing so. In general, APIs will go beyond the requirements and recommendations stated in this draft specification - or other parts of the OGC API family of standards - and will support additional operations, parameters, etc. that are specific to the API or the software tool used to implement the API.
5.5.2. Role of OpenAPI

This document uses OpenAPI 3.0 fragments as examples and to formally state requirements. However, using OpenAPI 3.0 is not required for implementing a server.

Therefore, the Core requirements class only requires that an API definition is provided and linked from the landing page.

In this document, fragments of OpenAPI definitions are shown in YAML (YAML Ain't Markup Language) since YAML was designed to be easier to read than JSON and is typically used in OpenAPI editors. YAML is described by its authors as a human friendly data serialization standard for all programming languages.

5.5.3. References to OpenAPI components in normative statements

Some normative statements (requirements, recommendations and permissions) use a phrase that a component in the API definition of the server must be "based upon" a schema or parameter component in the OGC schema repository.

In the case above, the following changes to the pre-defined OpenAPI component are permitted.

• If the server supports an XML encoding, xml properties may be added to the relevant OpenAPI schema components.

• The range of values of a parameter or property may be extended (additional values) or constrained (if a subset of all possible values is applicable to the server). An example for a constrained range of values is to explicitly specify the supported values of a string parameter or property using an 'enum' enumeration.

• The default value of a parameter may be changed or added unless a requirement explicitly prohibits this.

• Additional properties may be added to the schema definition of a Response Object.

• Informative text may be changed or added, like comments or description properties.

For API definitions that do not conform to the OpenAPI Specification 3.0, the normative statement should be interpreted in the context of the API definition language used.

5.5.4. Paths in OpenAPI definitions

All paths in an OpenAPI definition are relative to a base URL of the server.
Example 1. URL of the OpenAPI definition

If the OpenAPI Server Object looks like this:

```json
servers:
- url: https://dev.example.org/
  description: Development server
- url: https://data.example.org/
  description: Production server
```

The path "/mypath" in the OpenAPI definition of a Web API would be the URL `https://data.example.org/mypath` for the production server.

5.5.5. Reusable OpenAPI components

Reusable components for OpenAPI definitions for implementations of OGC APIs are referenced from this document.
Chapter 6. Overview

6.1. 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 when some of the initial versions of OGC Web Services were originally designed in the late 1990s and early 2000s. There is now another architectural style, one based on Representational State Transfer (REST). The RESTful API style is proposed as an alternative to the RPC pattern. A RESTful API style is resource-oriented instead of service-oriented. The OGC API - Maps and Tiles draft specification specifies an API that follows this Web architecture and in particular the W3C/OGC best practices for sharing Spatial Data on the Web [https://www.w3.org/TR/sdw-bp/] as well as the W3C best practices for sharing Data on the Web.

The OGC API – Common draft specification specifies the common kernel for an API approach to services that follows current resource-oriented architecture best practices. The draft OGC API - Common specification is the foundation upon which OGC APIs will be built. This Common API is to be extended by a number of resource-specific OGC API standards. The draft specifications documented in this ER extend OGC API - Common to support Map and Tile resources.

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

- Clear separation between core requirements and more advanced capabilities. The OGC API – Maps and Tiles draft specifications present the requirements that are relevant for almost everyone who wants to share or use tiled maps at a fine-grained level. Additional capabilities that several communities are using today will be specified as extensions to the Core API.

- Technologies that change more frequently are decoupled and specified in separate modules ("requirements classes" in OGC terminology). This enables, for example, the use/re-use of new encodings for spatial data or API descriptions.

- Modularization is not just about a single "service". OGC APIs will provide building blocks that can be reused in APIs in general. In other words, a server supporting the OGC API - Tiles should not be seen as a standalone service. Rather it should be viewed as a collection of API building blocks which together implement Map and Tile capabilities. A corollary for this is that it should be possible to implement an API that simultaneously conforms to conformance classes from the Feature, Coverage, Map, Tiles, and other future OGC Web API standards.

This approach intends 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 API definition without the need to adopt a specific OGC approach. For example, they no longer need to read how to implement a GetCapabilities, allowing focus on the geospatial aspects.

- Those that want to write a "generic" client that can access OGC APIs. In other words, they are not specific to a particular API.
As a result of following a RESTful approach, OGC API implementations are not backwards compatible with OGC Web Service (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 an 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 (e.g. by initially implementing facades in front of the current OWS services).

This ER provides simple examples throughout. The examples are based on a dataset that contains buildings and the API provides access to the datasets via a single feature collection (“buildings”) and two encodings: JSON and Hypertext Markup Language (HTML).

6.2. Tiles and maps

WMS and WMTS 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 can be selected by the user within limits and in WMTS the number of rows and columns of the response is predefined in the tile matrix set.

With time, the concept of a tile has been generalized to other data models such as feature data (some vendors use the expression vector tiles) and even to coverage data. The draft specifications present an approach to tiles that can be applied to almost every resource type that returns data representations. If applied in conjunction with the OGC API - Features standard and on top of a feature collection, the expected result is tiled feature data. If applied in conjunction with the OGC API - Maps draft specification and on top of a collection that is transformed into a map by applying a style, the result should be map tiles (usually in an image format).

In the draft specifications the OGC API - Tiles is almost fully described. The specification includes the core and extensions for defining tile matrix sets, tiles from more than one collection, multi-tiles (alias multitiles) from a single collection and multitiles from more than one collection. An info extension is foreseen but not fully developed. In contrast, OGC API - Maps is only partially described based on Testbed-15 requirements. The Maps API is described only to the extent to support map tiles created on top of a map created by selecting a collection with style or multiple collections with styles. This draft specification contains a section for retrieving a map of an arbitrary number of rows and columns but is not fully formulated. Other extensions for maps are also foreseen. In the future, the WMS SWG could take this document and complete the missing capabilities.
6.3. How to approach an OGC API

There are two ways to approach an OGC API.

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

For the first approach, many resources in the API include links with rel properties explaining the reason for this relation. The following figure illustrates the links.

*Figure 1. Modular approach in the Maps and Tiles draft specification*
For the second approach, the section OpenAPI Examples will provide some examples of OpenAPI definition documents that enumerate the paths to get to the necessary resources directly.

Table 1. Overview of resources and common direct links defined in the API

<table>
<thead>
<tr>
<th>Resource name</th>
<th>Common path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Landing page</td>
<td>/</td>
</tr>
<tr>
<td>Conformance declaration</td>
<td>/conformance</td>
</tr>
<tr>
<td>Collections</td>
<td>/collections</td>
</tr>
<tr>
<td>Collection</td>
<td>/collections/{collectionId}</td>
</tr>
<tr>
<td>Tiling Schemas</td>
<td>/tileMatrixSets</td>
</tr>
<tr>
<td>Tiling Schema</td>
<td>/tileMatrixSets/{tileMatrixSetId}</td>
</tr>
<tr>
<td>Tiles</td>
<td></td>
</tr>
<tr>
<td>Vector Tiles description</td>
<td>/collections/{collectionId}/tiles</td>
</tr>
<tr>
<td>Resource name</td>
<td>Common path</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Vector Tiles description from collections</td>
<td>/tiles</td>
</tr>
<tr>
<td>Vector Tile</td>
<td>/collections/{collectionId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}</td>
</tr>
<tr>
<td>Vector tile collections¹</td>
<td>/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}</td>
</tr>
<tr>
<td>Vector Multi-tiles</td>
<td>/collections/{collectionId}/tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>Vector Multi-tiles collections¹</td>
<td>/tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>Map tiles</td>
<td></td>
</tr>
<tr>
<td>Map tiles description</td>
<td>/collections/{collectionId}/map/</td>
</tr>
<tr>
<td>Map tiles description collections¹</td>
<td>/map/tiles</td>
</tr>
<tr>
<td>Map tile</td>
<td>/collections/{collectionId}/map/{styleId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}</td>
</tr>
<tr>
<td>Map tile collections¹</td>
<td>/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}</td>
</tr>
<tr>
<td>Map tile multi-tiles</td>
<td>/collections/{collectionId}/map/{styleId}/tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>Map tile multi-tiles collections¹</td>
<td>/map/tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>Maps</td>
<td></td>
</tr>
<tr>
<td>Maps description</td>
<td>/collections/{collectionId}/map</td>
</tr>
<tr>
<td>Maps description collections¹</td>
<td>/map</td>
</tr>
</tbody>
</table>

¹: In the first column of the table, the word “collections” means “from more than one collection”
Chapter 7. Requirement Class "Tiles Core"

7.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

Target type: Web API

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2616 (HTTP/1.1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 2818 (HTTP over TLS)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 3339 (Date and Time on the Internet: Timestamps)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC 8288 (Web Linking)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/tilematrixset/1.0/req/tilematrixset2d">http://www.opengis.net/spec/tilematrixset/1.0/req/tilematrixset2d</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections</a></td>
</tr>
</tbody>
</table>

An API that implements this conformance class provides access to tiled resources of a dataset [https://www.w3.org/TR/vocab-dcat/#class-dataset]. In other words, the API enables the distribution [https://www.w3.org/TR/vocab-dcat/#class-distribution] of that dataset. An implementation of the OGC API - Features standard, for example, could be another distribution.

The entry point is a Landing page (path /).

The Landing page provides links to:

- the API definition (path /api, link relation service-desc),
- the Conformance declaration (path /conformance, link relation conformance), and
- the Collections (path /collections, link relation data).

The API definition describes the capabilities of the API instance that can be used by clients to retrieve resources from the API or by development tools to support the implementation of API servers and clients. Accessing the API definition using HTTP GET returns a description of the API.

The Conformance declaration states the requirements classes from standards or community specifications, identified by a Uniform Resource Identifier (URI), that the API conforms to. Clients can, but are not required to, use this information. Accessing the Conformance declaration using HTTP GET returns the list of URIs of requirements classes implemented by the API.

The core of the OGC API - Tiles draft specification (as defined in this chapter) does not mandate the inclusion of an explicit definition of any TileMatrixSet. This draft specification assumes that clients and services know about the eight TileMatrixSets defined in OGC 17-083r2 annex D and there is no need to communicate these definitions. An extension to the core provides the capability to include definitions of flexible TileMatrixSets that are explicitly defined.

This draft specification assumes that data is organized into one or more collections. Collections provides information about the collections and enumerate the collection identifier.
This document does not specify requirements for collections, and they can consist of features, coverages, a resource that does not represent data per-se (e.g. an annotation) any other resource that can be represented in a tile. collectionId replaces the concept of layer in WMS and WMTS. Maps or tiles can be generated from one collection (or a combination of collections as an extension).

Accessing Collections using HTTP GET returns a response that contains at least the list of collections. Accessing Collections/{collectionId} using HTTP GET returns a description of a collection with an indication of whether the collection can be retrieved as a map or a tile or both. Accessing the items of a collection is out of the scope of this draft specification but is described in other draft OGC API specifications for features or coverages, for instance. For each Collection, a link to metadata about the collection is available (path /collections/{collectionId}) with key information about the collection. This information includes:

- A local identifier for the collection that is unique for the dataset;
- An optional title and description for the collection;
- An optional extent that can be used to provide an indication of the spatial and temporal extent of the collection - typically derived from the data;
- A list of TileMatrixSetLink objects relating to the available tiling schemas supported by the collection (from the linked TileMatrixSet member, the client can determine the coordinate reference systems (CRS) in which tiles may be returned by the API)

The Collection resource is available at path /collections/{collectionId}, often with more details than included in the Collections response. In particular, there is a list of links. If there is a link to more metadata about tiles, the collection is available directly as tiles. In the metadata about tiles there are also links and at least one of these links will provide the template to get individual tiles.

### 7.2. General

<table>
<thead>
<tr>
<th>Requirement 1</th>
<th>/req/tiles/core/api-common</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>An OGC API – Tiles implementation SHALL comply with the requirements specified in the <a href="http://www.opengis.net/spec/OAPI_Common/1.0/req/core">http://www.opengis.net/spec/OAPI_Common/1.0/req/core</a> and <a href="http://www.opengis.net/spec/OAPI_Common/1.0/req/collections">http://www.opengis.net/spec/OAPI_Common/1.0/req/collections</a> Requirements Classes of the OGC API-Common version 1.0 Standard.</td>
</tr>
</tbody>
</table>

In practice, this means that the landing page and the conformance page follow OGC API - Common core and collections requirement classes. This draft specification provides extra additions to the OGC API - Common requirements that are particular to tiles.

### 7.3. API landing page

The landing page provides links to start exploring the resources offered by the API. The landing page mainly consists of a list of links. OGC API - Common already requires some common links that
are enough for this draft specification core.

7.3.1. Response

There are no required variations to the landing page.

7.4. Declaration of conformance classes

To support "generic" clients that want to access multiple OGC API standards and extensions - and not "just" a specific API / server, the API has to declare the requirements classes it implements and conforms to.

7.4.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 2</th>
<th>/req/tiles/core/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the tiles core conformance class as links to <a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. Below is an example fragment of the response to an OGC API - Tiles conformance information page.

Example 2. Conformance Information Page fragment

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

7.5. Collections

This draft specification includes dependencies on OGC API - Common collections. Collections are mandatory in the core of this draft specification because collections are the object that will be included in a tile.

Collections will enumerate the collectionId identifiers available in this implementation of the OGC API draft specification as well as basic information about each collectionId: id, title, description,
extent, CRS and links. This common response is considered enough for a general description of the collection.

<table>
<thead>
<tr>
<th>Requirement 3</th>
<th>/req/tiles/core/tc-md-collection-links</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>For each collection included in the response, a links property of the collection SHALL include a link to the description of the collection (rel: item) (in addition to other links specified in OGC API Commons).</td>
</tr>
</tbody>
</table>

More specific details about the collection can be found following the link to the individual collections that follow the pattern /collections/{collectionId}

**NOTE** The collectionId substitutes the concept of "layer" in WMTS 1.0.

### 7.6. Collection

This draft specification includes dependencies on the OGC API - Common collection requirement. The response to the operation is extended with a new link for the tiles description.

#### 7.6.1. Collection Links

<table>
<thead>
<tr>
<th>Requirement 4</th>
<th>/req/tiles/core/tc-tile-desc-links</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A links property of the collection SHALL include a link to the description of the tiles (rel: tiles) (in addition to other links specified in OGC API Commons).</td>
</tr>
</tbody>
</table>
7.7. Tiles description

The response to this operation contains the necessary metadata to enable a client application to formulate a tile request from a single collection.

7.7.1. Operation

<table>
<thead>
<tr>
<th>Requirement 5</th>
<th>/req/tiles/core/sct-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Every resource available as tiles SHALL support an operation to retrieve the description of the tiles the API implementation can provide, available as a HTTP GET request to a URI that will be composed by two parts: the initial part is the URI of a resource that can be represented as tiles and the final part follows the pattern /tiles. Only the resources or collection that supports this operation can be retrieved as tiles.</td>
</tr>
</tbody>
</table>

The request of this operation has no parameters.

7.7.2. Response

A successful response to a tiles request for a collection that can be retrieved as tiles will respond with a data structure with specific information necessary to get tiles representing the resource collection. In this core draft specification, the response is only required to inform about from which tile matrix sets tiles can be retrieved and the URL template to a tile.
A

The content of the response to a successful execution SHALL contain a property called `tileMatrixSetLinks` with a list of `tileMatrixSetLink` objects following a data model defined in the clause 7.3 of OGC 17-083r2. In the core specification `tileMatrixSetLink` is only used for referencing the supported TileMatrixSets for the tiles limiting it to the following schema (expressed as an OpenAPI Specification 3.0 fragment):

```yaml
tileMatrixSetLink-set:
  description: This list of tileMatrixSetLink objects, as defined in OGC 17-083r2 supported by this collectionId.
  type: array
  items:
    $ref: '#/components/schemas/tileMatrixSetLink-entry'
tileMatrixSetLink-entry:
  type: object
  required:
    - tileMatrixSet
  properties:
    tileMatrixSet:
      type: string
      example: 'WebMercatorQuad'
tileMatrixSetURI:
      type: string
      format: uri
      example: 'http://www.opengis.net/def/tilematrixset/OGC/1.0/WebMercatorQuad'
```

Example 4. Example of a tiles metadata response.

```json
{
    "tileMatrixSetLinks": [
        {
            "tileMatrixSet": "WorldMercatorWGS84Quad",
            "tileMatrixSetURI": "http://schemas.opengis.net/tms/1.0/json/examples/WorldMercatorWGS84Quad.json"
        }
    ],
    "links": [
        {
            "href": "http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}/{tileMatrixSet}/{tileRow}/{tileCol}.png",
            "rel": "item",
            "type": "image/png"
        }
    ]
}
```

**Recommendation 7**

A  This core requirements class does not provide any mechanism to defined TileMatrixSets so if this mechanism is not provided in an extension, the tileMatrixSetURI SHOULD point to one of the 8 URIs defined in the OGC 17-083r2 Annex D.

B  The server SHOULD provide the client a way to get full description of the TileMatrixSet. Even if the TileMatrixSet is not directly defined by the API, when a full definition of the TileMatrixSet is available as a resolvable URL, a resolvable URL SHOULD be used as the value of the tileMatrixSetURI.

Resolvable URLs for the 8 URIs defined in the OGC 17-083r2 Annex D are available in the OGC schemas repository in XML, JSON and RDF formats. For example, JSON descriptions can be found here: [http://schemas.opengis.net/tms/1.0/json/examples/](http://schemas.opengis.net/tms/1.0/json/examples/)

**Requirement 8**

A  The content of the response to a successful execution SHALL include at least a link to a tile URI template (rel: `item`).
These links SHALL provide a URL template with the fragment `/tiles` followed by the variables `{tileMatrixSetId}`, `{tileMatrix}`, `{tileRow}` and `{tileCol}`. Once the variables are substituted by their respective valid values, a URL to a tile is obtained.

There SHALL be a link to a tile URI template for each format that the server supports (the format is indicated in the ‘type’ attribute of the link).

One common order used in URL templates for tiles is `/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}`, but this draft specification allows for other URL template composition.

<table>
<thead>
<tr>
<th>URL template variable</th>
<th>Meaning</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>TileMatrixSetId</td>
<td>tile matrix set identifier</td>
<td>One of the identifiers included in Annex D of OGC 17-083r2 or an identifier defined by extensions of this core</td>
</tr>
<tr>
<td>TileMatrix</td>
<td>tile matrix identifier</td>
<td>Identifier of the tile matrix (representing a zoom level, a.k.a. a scale) listed in the TileMatrixSet definition</td>
</tr>
<tr>
<td>TileRow</td>
<td>row index of tile matrix</td>
<td>A non-negative integer between 0 and the MatrixHeight – 1. If there is a TileMatrixSetLimits the value is limited between MinTileRow and MaxTileRow</td>
</tr>
<tr>
<td>TileCol</td>
<td>column index of tile matrix</td>
<td>A non-negative integer between 0 and the MatrixWidth – 1. If there is a TileMatrixSetLimits the value is limited between MinTileCol and MaxTileCol</td>
</tr>
</tbody>
</table>

Example 5. link to get tiles as a URL template in the tiles metadata response fragment

```json
links:
[
  {
    "href": "http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}"
  }"tiles",
  "type": "image/png",
}
]```
7.8. Tiled data from one collection

The core of the OGC API -Tiles draft specification provides a mechanism to select and retrieve a tile in a TileMatrixSet. If the service does not advertise any other TileMatrixSet (this core does not describe any mechanism to do that, but an extension will do it) only the TileMatrixSet identifiers specified in the Annex D.1 of the OGC 17-083r2 standard can be used.

7.8.1. Operation

<table>
<thead>
<tr>
<th>Requirement 9</th>
<th>/req/tiles/core/tc-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Every tile SHALL be available as a HTTP GET request to a URI that will be composed by two parts: The first part is the URI of a resource that can be represented as tiles and the second part follows the pattern /tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}</td>
</tr>
</tbody>
</table>

Typical resources that can be retrieved as tiles are: features (/collections/{collectionId}), coverages (/collections/{collectionId}/coverage/{coverageId} or /coverage/{coverageId}) or maps (/collections/{collectionId}/map/styleId)).

NOTE The common path for coverages is still under discussion.

7.8.2. Parameter tileMatrixSetId

<table>
<thead>
<tr>
<th>Requirement 10</th>
<th>/req/tiles/core/tc-tilematrixsetid-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The operation SHALL support a parameter tileMatrixSetId with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</td>
</tr>
</tbody>
</table>

```yaml
name: tileMatrixSetId
in: path
description: Identifier of a specific tiling scheme. It can be one of those specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.
required: true
schema:
  type: string
example: WebMercatorQuad
```

7.8.3. Parameter tileMatrix
### 7.8.4. Parameter tileRow

<table>
<thead>
<tr>
<th>Requirement 12</th>
<th>/req/tiles/core/tc-tilerow-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The operation SHALL support a parameter <strong>tileRow</strong> with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</td>
</tr>
</tbody>
</table>

```yaml
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'
```

### 7.8.5. Parameter tileCol

<table>
<thead>
<tr>
<th>Requirement 13</th>
<th>/req/tiles/core/tc-tilecol-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The operation SHALL support a parameter <strong>tileCol</strong> with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</td>
</tr>
</tbody>
</table>

```yaml
name: tileCol
in: path
description:  |
required: true
schema:
  type: integer
  minimum: 0
  example: '30'
```
The operation SHALL support a parameter `tileCol` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```
    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
```

### 7.8.6. Response

A successful response to a tile request will be consistent with the media type of resource requested. This draft specification does not impose any media type. For example:

- For features the media type can be geojson, Geography Markup Language (GML), Mapbox vector tiles or other;
- For coverages the response may be a geotiff, GMLJP2, netCDF or other;
- For maps the response may be a JPEG, PNG, GMLJP2 or other.

<table>
<thead>
<tr>
<th>Requirement 14</th>
<th>/req/tiles/core/tc-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code <strong>200</strong>.</td>
</tr>
<tr>
<td>B</td>
<td>The content of that response SHALL be consistent with the format requested and represent elements inside or intersecting with the spatial extent of the geographical area of the tile identified by TileMatrixSet, TileMatrix, TileRow and TileCol.</td>
</tr>
</tbody>
</table>

Normally, the content partially outside the tile bounding box will be clipped and this is particularly true when tiles are in raster format. Nevertheless, tiles containing features in vector format may not clip features that are partially outside.

| Recommendation 1 | /rec/tiles/core/tc-success-scale |
A general summary of the HTTP status codes can be found in the OGC API - Common.

If the parameter value `tileMatrixSetId` is not available by the server for this resource or the parameters values `tileMatrix, tileRow, tileCol` are out-of-range, the status code of the response will be 404.
Chapter 8. Requirement Class "Tiles from more than one collection"

8.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
<th><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

In previous clauses tiles that are produced from one, and only one resource were discussed. This scenario is achieved by concatenating the tile path to a resource (e.g. a feature collection). This requirements class is an extension of the core requirements class that defines how to create tiles that combine more than one resource. This is achieved by having the tile path also available at the root of the service.

8.2. API landing page

The landing page provides links to start exploring the resources offered by the API. It mainly consists of a list of links. The core of this draft specification does not add anything to the links required by OGC API - Common. This requirements class for tiles from more than one collection requires a new link for getting the description of the tiles from more than one collection on top of the common ones.

8.2.1. Response

<table>
<thead>
<tr>
<th>Requirement 15</th>
<th>/req/tiles/collections/root-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API SHALL advertise a URI to retrieve tiles definitions defined by this service as links to the descriptions paths with rel: tiles.</td>
</tr>
</tbody>
</table>

In the landing page, in JSON format, the links follow the link schema defined in the OGC API - Common draft specification. Below is an example fragment of the response to an OGC API - Tiles landing page showing the new link.
Example 6. API Landing Page fragment that advertises the path to get tiles for more than one collection

```json
{
  links: [ 
    ..., 
    {
      "href": "http://data.example.org/tiles",
      "rel": "tiles",
      "type": "application/json",
      "title": "Link to information on map tiles combining more than one collection",
    }
  ]
}
```

8.3. Declaration of conformance classes

8.3.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 16</th>
<th>/req/tiles/collections/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the capability of generating tiles from multiple collections adding the conformance class for this capability as a link to <a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections</a>.</td>
</tr>
</tbody>
</table>

On the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API – Common draft specification. The following is an example fragment from the response to an OGC API - Tiles conformance information page showing the support for tiles from more than one collection

Example 7. Conformance Information Page fragment

```json
{
  "conformsTo": [ 
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
    "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections"
  ]
}
```
8.4. Tiles description

The response to the tiles description operation contains the necessary information to later formulate a tile request of tiles from more than one collection.

8.4.1. Operation

<table>
<thead>
<tr>
<th>Requirement 17</th>
<th>/req/tiles/collections/ts-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server SHALL support an operation to retrieve the description of the tiles from more than one collection, available as a HTTP GET request to a URI that is composed by two parts: the first part is the URI of a resource that can be represented as tiles (e.g. /map or simply /) and the second part follows the pattern /tiles.</td>
</tr>
</tbody>
</table>

The request of this operation has no parameters.

8.4.2. Response

A successful response to a tiles request for more than one collection will respond with a data structure with specific information necessary to get tiles representing the resource collection. In this requirements class, the response only provides the URL template to retrieve a tile.

<table>
<thead>
<tr>
<th>Requirement 18</th>
<th>/req/tiles/collections/ts-tile-examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The content of the response to a successful execution SHALL include at least one link to a tile URI template (rel: item).</td>
</tr>
<tr>
<td>B</td>
<td>These links SHALL provide a URL template with the fragment /tiles followed by the variables {tileMatrixSetId}, {tileMatrix}, {tileRow} and {tileCol}. Once the variables are substituted by their respective valid values, a URL to a tile is obtained.</td>
</tr>
<tr>
<td>C</td>
<td>There SHALL be a link to a tile URI template for each format that the server supports (the format is indicated in the 'type' attribute of the link)</td>
</tr>
</tbody>
</table>

One common order used in URL templates for tiles is: /tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}. However, this draft specification allows for other URL template composition.

*Table 3. URI template variables for tiles and possible values*
<table>
<thead>
<tr>
<th>URL template variable</th>
<th>Meaning</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>TileMatrixSetId</td>
<td>tile matrix set identifier</td>
<td>The identifiers included in Annex D of OGC 17-083r2 or defined by extensions of the core requirements class.</td>
</tr>
<tr>
<td>TileMatrix</td>
<td>tile matrix identifier</td>
<td>Identifier of the tile matrix (representing a zoom level, a.k.a. a scale) listed in the TileMatrixSet definition.</td>
</tr>
<tr>
<td>TileRow</td>
<td>row index of tile matrix</td>
<td>A non-negative integer between 0 and the MatrixHeight – 1. If there is a TileMatrixSetLimits the value is limited between MinTileRow and MaxTileRow.</td>
</tr>
<tr>
<td>TileCol</td>
<td>column index of tile matrix</td>
<td>A non-negative integer between 0 and the MatrixWidth – 1. If there is a TileMatrixSetLimits the value is limited between MinTileCol and MaxTileCol.</td>
</tr>
</tbody>
</table>

Example 8. API tiles response fragment with the link to retrieve tiles form more than one collection

```
links:
[
  {
    "href": "http://data.example.com/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}"
  ,
    "rel": "item",
    "type": "image/png"
  }
]
```

In general, the `tileMatrixSetLinks` and the `tileMatrixSetLimits` can be determined by examining this information in the individual collections. In some cases, the server could also include the `tileMatrixSetLinks` data structure as part of the response to this operation. Clients should be prepared to determine if a `tileMatrixSetLinks` data structure is not provided in certain combinations of collections by examining the tileMatrixSet values and limits from the information in the individual collections and calculating the limits as the most restrictive intersection of them.

8.5. Tiles from more than one collection

This operation allows retrieving a single tile that represents information coming from more than one collection.

8.5.1. Operation

| Requirement 19 | /req/tiles/collections/tcs-op |
The server SHALL support the HTTP GET operation at the path /tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}

### 8.5.2. Parameter tileMatrixSetId

*Requirement 20*

The operation SHALL support a parameter `tileMatrixSetId` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```yaml
name: tileMatrixSetId
in: path
description: Identifier of a specific tiling scheme. It can be one of the specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.
required: true
schema:
  type: string
example: WebMercatorQuad
```

### 8.5.3. Parameter tileMatrix

*Requirement 21*

The operation SHALL support a parameter `tileMatrix` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```yaml
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'
```

### 8.5.4. Parameter tileRow

*Requirement 22*

The operation SHALL support a parameter `tileRow` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):
The operation SHALL support a parameter `tileRow` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```yaml
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'
```

### 8.5.5. Parameter tileCol

<table>
<thead>
<tr>
<th>Requirement 23</th>
<th>/req/tiles/collections/tcs-tilecol-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The operation SHALL support a parameter <code>tileCol</code> with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</td>
</tr>
</tbody>
</table>

```yaml
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
```

### 8.5.6. Parameter Collections

| Requirement 24 | /req/tiles/collections/tcs-collections-definition |
The operation 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
```

The parameter `collections` SHALL contain a comma-separated list of collection identifiers.

Only the collections that advertise a link type=tiles in the `/collections/{collectionId}` SHALL be included.

Only the collections that support the same TileMatrixSetId parameter value SHALL be included.

---

### Recommendation 2 /rec/tiles/collections/tcs-collections-definition

**A**

If the parameter `collections` is missing, and when it is possible and sensible, all collections supporting the TileMatrixSetId parameter value SHOULD be represented in the tiles.

**B**

The collection ids that can be used for this operation SHOULD be listed in the description of the `collections` parameter in the API definition.

### Permission 1 /per/tiles/collections/tcs-collections-definition

**A**

If the parameter `collections` is missing and if it is not possible and sensible to represent all collections in tiles (e.g. it compromises performance or tiles are become packed with too many elements), the server is allowed to select only the most significant collections.
8.5.7. Response

A successful response to a tile request is consistent with the media type of the requested resource. This draft specification does not impose any media type. For example, for features the media type can be GeoJSON, GML or Mapbox vector tiles; for coverages it may be a GeoTIFF, GMLJP2, netCDF; and for maps it may be a JPEG, PNG, GMLJP2 or other.

<table>
<thead>
<tr>
<th>Requirement 25</th>
<th>/req/tiles/collections/tcs-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code <strong>200</strong>.</td>
</tr>
<tr>
<td>B</td>
<td>The content of that response SHALL be consistent with the format requested and represent elements inside or intersecting with the spatial extent of the geographical area of the tile identified by TileMatrixSet, TileMatrix, TileRow and TileCol.</td>
</tr>
<tr>
<td>C</td>
<td>The content of that response SHALL be simplified to comply with the scale denominator represented by the TileMatrix identified. Full resolution geographical elements will only be provided for the lower values of scale denominators.</td>
</tr>
</tbody>
</table>

8.5.8. Error conditions

If the value of the parameter `tileMatrixSetId` is not available by the server for this resource or the values of the parameters `tileMatrix`, `tileRow`, `tileCol` are out-of-range, the status code of the response is 404.

If the value of the parameter `collections` contains a collection id that does not exist on the server, the status code of the response is 404.

If the value of the parameter `collections` has a wrong format or combines collections and some of them are not compatible with the `tileMatrixSetId` value, the status code of the response is 500.
Chapter 9. Requirement Class "Tiles Tile Matrix Set"

9.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tmxs">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tmxs</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Target type</th>
<th>Web API</th>
</tr>
</thead>
</table>

| Dependency | http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core |
| Dependency | http://www.opengis.net/spec/tilematrixset/1.0/req/json-tilematrixset2d |
| Dependency | http://www.opengis.net/spec/tilematrixset/1.0/req/tilematrixsetlimits2d |
| Dependency | http://www.opengis.net/spec/tilematrixset/1.0/req/json-tilematrixsetlimits2d |
| Dependency | http://www.opengis.net/spec/tilematrixset/1.0/req/tilematrixsetlink2d |
| Dependency | http://www.opengis.net/spec/tilematrixset/1.0/req/json-tilematrixsetlink2d |

The *tiles core* requirements class states that the service can support the eight TileMatrixSets defined in the Annex D.1 of the OGC 17-083r2 standard by mentioning their identifiers without the need to describe them. This requirement class acts as an extension of the core requirements class that adds all the necessary elements to support other TileMatrixSets by adding a mechanism to fully describe TileMatrixSets that are specific to the API instance.

The entry point is a Landing page (path `/`).

The Landing page provides links to:

- the API definition (path `/api`, link relation `service-desc`),
- the Conformance declaration (path `/conformance`, link relation `conformance`), and
- the Collections (path `/collections`, link relation `data`).
- the TileMatrixSets (path `/tileMatrixSets`, link relation `tiling-schemes`).

9.2. API landing page

The landing page provides links to start exploring the resources offered by the API. It mainly consists of a list of links. The core of this draft specification does not add anything to the links required by OGC API - Common. This extension for TileMatrixSet requires new links for TileMatrixSets on top of the common ones.

9.2.1. Response

<table>
<thead>
<tr>
<th>Requirement 26</th>
<th>/req/tiles/tmxs/root-success</th>
</tr>
</thead>
</table>
The API SHALL advertise a URI to retrieve the list of TileMatrixSets defined by this service as links to the descriptions paths with rel=tiling-schemes.

In the landing page, in JSON format, the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response to an OGC API - Tiles landing page.

Example 9. API Landing Page fragment with links to TileMatrixSet descriptions

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

9.3. Declaration of conformance classes

9.3.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links for the core and collections requirements classes.

<table>
<thead>
<tr>
<th>Requirement 27</th>
<th>/req/tiles/tmxs/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the capability of generating tiles from multiple collections adding the conformance class for this capability as a link to <a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tmxs">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tmxs</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response to an OGC API tiles conformance information page.
9.4. TileMatrixSets

The TileMatrixSets operation retrieves links to the descriptions of the tile matrix sets supported by the API instance in addition to the eight TileMatrixSets defined in the Annex D.1 of the OGC 17-083r2 standard.

9.4.1. Operation

<table>
<thead>
<tr>
<th>Requirement 28</th>
<th>/req/tiles/tmxs/tmxs-tilematrixsets-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server SHALL support the HTTP GET operation at the path /tileMatrixSets.</td>
</tr>
</tbody>
</table>

9.4.2. Response

<table>
<thead>
<tr>
<th>Requirement 29</th>
<th>/req/tiles/tmxs/tmxs-tilematrixsets-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code 200.</td>
</tr>
<tr>
<td>B</td>
<td>The body of the response SHALL be a tileMatrixSets object listing the tilematrixsets supported by this server other than the eight ones defined in the Annex D of OGC 17-083r2 standard.</td>
</tr>
<tr>
<td>C</td>
<td>For each TileMatrixSet the response SHALL contain a TileMatrixSet id and a link to request the TileMatrixSet description.</td>
</tr>
</tbody>
</table>
Example 11. Schema for the TileMatrixSets resource

```json
  type: object
  required:
    - tileMatrixSets
  properties:
    tileMatrixSets:
      type: array
      items:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/id-link'
```

Example 12. Schema for id-link from OGC API - Common used in TileMatrixSets resource.

```json
  id-link:
    type: object
    description: |- 
      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` will respond an array of id-link listing the `resourceId` and the links to get it. /collections and /collections/{collectionId} 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
      title:
        type: string
      links:
        type: array
        minItems: 1
        items:
          $ref: '#/components/schemas/link'
```
Example 13. Example for the TileMatrixSets resource

```json
{
    "tileMatrixSets": [
        {
            "id": "MyWebMercatorQuad",
            "title": "My Google Maps Compatible for the World",
            "links": [
                {
                    "href": "https://data.example.org/tileMatrixSet/MyWebMercatorQuad",
                    "rel": "item",
                    "type": "application/json"
                }
            ]
        }
    ]
}
```

9.5. TileMatrixSet

The TileMatrixSet operation retrieves the full description of a tile matrix set supported by the API instance following the schema described in the OGC 17-083r2 standard.

9.5.1. Operation

<table>
<thead>
<tr>
<th>Requirement 30</th>
<th>/req/tiles/tmxs/tmxs-tilematrixset-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server SHALL support the HTTP GET operation at the path /tileMatrixSet/{tileMatrixSetId}.</td>
</tr>
<tr>
<td>A</td>
<td>The parameter tileMatrixSetId is each id property in the tileMatrixSets response.</td>
</tr>
</tbody>
</table>

9.5.2. Response

<table>
<thead>
<tr>
<th>Requirement 31</th>
<th>/req/tiles/tmxs/tmxs-tilematrixset-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code 200.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>B</td>
<td>The body of the response SHALL follow the TileMatrixSet data model defined in the <a href="http://www.opengeospatial.org/standard/tilematrixset/1.0/req/tilematrixset2d">http://www.opengeospatial.org/standard/tilematrixset/1.0/req/tilematrixset2d</a> requirements class of the Clause 7 in the OGC 17-083r2 standard.</td>
</tr>
<tr>
<td>C</td>
<td>The body of the response SHALL be encoded in JSON following the requirements class <a href="http://www.opengeospatial.org/standard/tilematrixset/1.0/req/json-tilematrixset2d">http://www.opengeospatial.org/standard/tilematrixset/1.0/req/json-tilematrixset2d</a> of the Clause 9 in the OGC 17-083r2 standard.</td>
</tr>
</tbody>
</table>

**Recommendation 3** /rec/tiles/tmxs/tilematrixset-response

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server may support a tileMatrixSetId that is one of the eight TileMatrixSets defined in the Annex D of OGC 17-083r2 and return a successful response with a description identical to the one in the Annex D of OGC 17-083r2.</td>
</tr>
</tbody>
</table>
Example 14. Fragment of a TileMatrixSet resource

```json
{
    "title": "Google Maps Compatible for the World",
    "abstract": "The most common TileMatrixSet, used in most of the main IT map browsers. It was initially popularized by Google Maps",
    "identifier": "WebMercatorQuad",
    "supportedCRS": "http://www.opengis.net/def/crs/EPSG/0/3857",
    "wellKnownScaleSet": "http://www.opengis.net/def/wkss/OGC/1.0/GoogleMapsCompatible",
    "tileMatrix": [
        ...,
        {
            "title": "Google Maps Compatible for the World zoom level 3",
            "abstract": "Google Maps Compatible zoom level 3 that is equivalent to a scale of 1:69885283.00358972 and has 19567.87924100512 meters of pixel size in the equator",
            "identifier": "3",
            "scaleDenominator": 69885283.00358972,
            "topLeftCorner": [
                -20037508.3427892,
                20037508.3427892
            ],
            "tileWidth": 256,
            "tileHeight": 256,
            "matrixHeight": 8,
            "matrixWidth": 8
        }
        ...
    ]
}
```

9.6. Tiles

The requirements class described in this section also defines an extra element limits in the tiles metadata returned by a successful /collection/{collectionId}/tiles request that can be used for the API instance to document limitations in the scales and extents supported in the context of the tile matrix set that is defined in a more unrestricted way.

9.6.1. Collection extra properties

| Requirement 32 | /req/tiles/tmxs/stc-limits |
|   | If the extent of the available tiles in the server is smaller than the extent of the TileMatrixSet, the object `tileMatrixSetLinks` in the response to a successful execution of the `tiles` request SHALL contain a property called `tileMatrixSetLimits` that is an array that specifies the limitations in the area available for this collection for each TileMatrix. `tileMatrixSetLink` object follows a data model defined in the clause 7.3 of OGC 17-083r2 that can be encoded in the following schema (shown as an OpenAPI Specification 3.0 fragment): |
The server SHALL only successfully respond with tiles for the mentioned scales and in the range of tilecol and tilerow defined. If the range of tilecol and tilerow is missing for a scale, all tilecol and tilerow values SHALL be made available by the server for this scale.

Example 15. Fragment of a Tiles resource with limits

```json
{
    "tileMatrixSetLinks": [
        {
            "type": "tileMatrixSetLink",
            "tileMatrixSet": 
            "http://www.opengis.net/def/tilematrixset/OGC/1.0/WebMercatorQuad",
            "tileMatrixSetLimits": [
                {
                    "type": "tileMatrixSetLimits",
                    "tileMatrix": "5",
                    "minTileRow": 0,
                    "maxTileRow": 1,
                    "minTileCol": 3,
                    "maxTileCol": 4
                }
            ]
        }
    ],
    ...
    "links": [
        ...
        {
            "href": 
            "http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}/tileMatrix/{tileRow}/{tileCol}.png",
            "rel": "item",
            "type": "image/png",
            "$ref": "https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#components/examples/link-tiles-tile"
        }
    ]
}
```
Chapter 10. Requirement Class "Tiles Info"

10.1. Overview

NOTE

This section should be elaborated by a SWG and only some hints are provided in this Engineering Report

WARNING

Some subsections are intentionally left blank.

Requirements Class

<table>
<thead>
<tr>
<th><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/info">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/info</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
</tr>
<tr>
<td>Dependency</td>
</tr>
</tbody>
</table>

This requirements class makes data contained in tiles a little more informative than just "nice pictures" by allowing clients to implement a click user event. By clicking on a pixel in the screen that shows a tile, the user will receive some textual information describing what is shown in that pixel. For example, by clicking on a tile containing elevation data the user will get the elevation value.

NOTE

The use of pixel in the screen can create the wrong impression that this operation is restricted to "raster based tiles". This is not necessarily true. The Two Dimensional Tile Matrix Set standard (OGC 17-083r2) discusses how tile matrices are created for an optimum resolution in the screen, even if they might be entirely feature based.

When fully completed, the new OGC API architecture should be able to integrate several representations of the same resource. This way a digital elevation model could be accessible as a tile and also as a coverage. The coverage part should be able to provide elevation values to the client. When that day arrives, this info requirements class will no longer be needed as the coverage functionality will provide the client with enough data to emulate this extension and some other extra interactions such as the capability to create vertical profiles.

10.2. Overview

TBD

10.3. Declaration of conformance classes

10.3.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

In the conformance page (typically in JSON format) the links follow the link schema defined in the
OGC API - Common draft specification. The following is an example fragment of the response to an OGC API - Tiles conformance information page.

**Example 16. Conformance Information Page fragment**

```json
{
   "conformsTo": [
      "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
      "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
      "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core",
      "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/info"
   ]
}
```

### 10.4. Collection

This draft specification includes dependencies on OGC API - Common collection. The response to the operation is extended with the necessary information to formulate a tile response for this collection.

**10.4.1. Collection Links**

**Example 17. API collection response fragment**

```json
links:
[ {
   "href": "http://data.example.com/collections/buildings/tiles/WorldMercatorWGS84Quad/0/0/0",
   "rel": "tiles",
   "type": "image/png",
 },
 { 
   "href": "http://data.example.com/collections/buildings/tiles/WorldMercatorWGS84Quad/0/0/0/info",
   "rel": "attributes",
   "type": "text/html",
 } ]
```

### 10.5. FeatureInfo

Implementations of the OGC API – Maps and OGC API - Tiles draft specifications may support
requests for information about the features present at a particular pixel location in the screen on a map tile. Requests for feature information will specify the tile along with a pixel location on that tile. The server will provide information on the features present at or near the location specified by the client request. The server may choose what information to provide about the nearby features.

10.5.1. FeatureInfo document

A FeatureInfo document is the resource representation of a FeatureInfo resource in resource oriented architectural style. The FeatureInfo document SHALL be in the format specified in the request when that format has been advertised in the ServiceMetadata document as available for that FeatureInfo resource.

For better interoperability between servers and clients, the Geography Markup Language (GML) simple features profile (with Corrigendum) (2.0) [10-100r3] as a supported document format for FeatureInfo resources is recommended. The Simple Features Profile of GML defines three levels of content in three profiles with different degrees of constraints to the GML flexibility. Support for the most constrained one (level 0) that results in a simpler GML document is strongly recommended. In the context of that profile only simple XML types can be used as thematic properties and cardinality greater than one is not allowed. Servers and clients SHALL specify the MIME type "application/gml+xml; version=3.1" as an InfoFormat value and the GML application schema of the response SHOULD conform to GML Simple Features profile level 0 when that GML profile is used. In most cases, only thematic attributes of the features are intended to be included in a FeatureInfo document but the Simple Features profiles were evidently intended to include the geometric information of the features in the GML objects. However, an application schema can be generated that does not include feature geometry and only describes non-geometric feature attribute types. This can be very useful to avoid unnecessarily requesting long sequences of position values in line or polygon layers.

Also, to allow easy presentation of the data, support for the HTML format (represented by an InfoFormat MIME type of "text/html") is also recommended.
Chapter 11. Requirement Class "Tiles Multi-tiles"

11.1. Overview

This requirement class opens the possibility to exchange multiple tiles covering a bounding box and belonging to one or more scales with a single client-server interaction.

<table>
<thead>
<tr>
<th>WARNING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Currently, this requirement class does not provide any way that clients or servers can limit the size of the multi-tile response. Even with a relatively small bounding box, the result of a multi-tile request (in particular to a collection that has tiles available at very small scale denominator values) could result in list of tiles too big for the server to generate or for the client to handle. The current specified default values for bounding box and scales range parameters will most probably incur in this problem. Before this requirement class is endorsed by the OGC, this issue should be addressed. One possible solution is to allow the server for specifying a maximum size limit (in kilobytes, or in number of tiles) and to force the server to start from the higher level of scale denominator and stop when the limit is reached. Adding a paging mechanism in the request could help by fragmenting big responses into smaller chunks that can be sequentially requested.</td>
</tr>
</tbody>
</table>

**Requirements Class**

<table>
<thead>
<tr>
<th>Requirements Class</th>
<th><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/multitiles">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/multitiles</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

In this requirements class, a mechanism is defined to request more than one tile from a single collection in a single request. This mechanism is called a ‘multi-tile’. The result can be a document listing the needed tiles to cover a bounding box or a package with all tiles inside.

11.2. Declaration of conformance classes

11.2.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

| Requirement 33 | /req/tiles/multitiles/conformance-success |
The API conformance path SHALL advertise the capability of generating tiles from multiple collections adding the conformance class for this capability as a link to `http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/multitiles`.

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response to an OGC API - Tiles conformance information page with support for multi-tiles.

Example 18. Conformance Information Page fragment

```json
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
    "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/multitiles"
  ]
}
```

11.3. Tiles description

The response to a tiles description request contains the necessary information to later formulate a tile or a multi-tile request for a collection.

11.3.1. Response

A successful response to a tiles request for a collection that can be retrieved as tiles will respond with a data structure with specific information necessary to get tiles representing the resource collection. This extension adds the URL template to a multi-tile.

<table>
<thead>
<tr>
<th>Requirement 34</th>
<th>/req/tiles/multitiles/mtc-multitiles-examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The content of the response to a successful execution SHALL include at least a link to a multi-tiles URI template (rel: <code>items</code>).</td>
</tr>
<tr>
<td>B</td>
<td>These links SHALL provide a URL template with the fragment <code>/tiles</code> followed by the variables <code>{tileMatrixSetId}</code>. Once the variables are substituted by their respective valid values, a URL to a multitiles is obtained.</td>
</tr>
<tr>
<td>C</td>
<td>There SHALL be a link to a multitle URI template for each format that the server supports (the format is indicated in the <code>type</code> attribute of the link).</td>
</tr>
</tbody>
</table>
One common order used in URL templates for tiles is .../tiles/{tileMatrixSetId} this draft specification allows for other URL template composition.

Table 4. URI template variables for tiles and possible values

<table>
<thead>
<tr>
<th>URL template variable</th>
<th>Meaning</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>TileMatrixSetId</td>
<td>tile matrix set identifier</td>
<td>The identifiers included in Annex D of OGC 17-083r2 or defined by extensions of the core specification.</td>
</tr>
</tbody>
</table>

Example 19. API tiles response fragment

```
links:
[
  {
    "href":  "http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}" ,
    "rel":  "items",
    "type":  "image/png",
  }
]
```

11.4. Multiple tiles from one collection

The following requirements provide a mechanism to select and retrieve a set of tiles at once following a TileMatrixSet.

11.4.1. Operation

<table>
<thead>
<tr>
<th>Requirement 35</th>
<th>/req/tiles/multitiles/mtc-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Tiles SHALL be available as HTTP GET requests to a URI that will be composed by two parts: a initial part is the URI of a resource that can be represented as tiles and the final part follows the pattern /tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>B</td>
<td>Only the resources or collections that advertise one of more links with type=tiles SHALL be requested as multiple tiles.</td>
</tr>
</tbody>
</table>

Typical resources that can be retrieved as tiles are: features (/collections/{collectionId}), coverages (/collections/{collectionId}/coverages/{coverageId} or /coverages/{coverageId}) or maps (/collections/{collectionId}/map/styleId)).
### 11.4.2. Parameter tileMatrixSetId

<table>
<thead>
<tr>
<th>Requirement 36</th>
<th>/req/tiles/multitiles/mtc-tilematrixsetid-definition</th>
</tr>
</thead>
</table>
| **A** | The operation SHALL support a parameter `tileMatrixSetId` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```json

name: tileMatrixSetId
in: path
description: Identifier of a specific tiling scheme. It can be one of the specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.
required: true
schema:
  type: string
  example: WebMercatorQuad
```

### 11.4.3. Parameter bbox

| Requirement 37 | /req/tiles/multitiles/mtc-bbox-definition |
### A

The operation SHALL support an optional parameter `bbox` to filter the area where tiles will be retrieved with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```yaml
name: bbox
in: query
description:
  'Only elements that have a geometry that intersects the bounding box are selected. The bounding box is provided as four or six numbers, depending on whether the coordinate reference system includes a vertical axis (elevation or depth):

  * Lower left corner, coordinate axis 1
  * Lower left corner, coordinate axis 2
  * Lower left corner, coordinate axis 3 (optional)
  * Upper right corner, coordinate axis 1
  * Upper right corner, coordinate axis 2
  * Upper right corner, coordinate axis 3 (optional)

  The coordinate reference system of the values is WGS 84 longitude/latitude (http://www.opengis.net/def/crs/OGC/1.3/CRS84) unless a different coordinate reference system is specified by another parameter in the API (e.g. `bbox-crs`).'

required: false
schema:
  type: array
  minItems: 4
  maxItems: 6
  items:
    type: number
    format: double
    style: form
    explode: false
```

### B

A TileMatrixSet definition points to a CRS. The coordinates of the `bbox` SHALL be in the CRS as specified in the definition of the TileMatrixSet identified by the `tileMatrixSetId`.

### C

If the `bbox` parameter is not specified, the server SHALL assume the whole extent of the tiles is requested.

This definition is inherited from OGC API - Common.
### 11.4.4. Parameter scaleDenominator

<table>
<thead>
<tr>
<th>Requirement 38</th>
<th>/req/tiles/multitiles/mtc-scaledenominator-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>The operation SHALL support an optional parameter <code>scaleDenominator</code> to filter the scales where tiles will be retrieved with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td></td>
<td>name: scaleDenominator</td>
</tr>
<tr>
<td></td>
<td>in: query</td>
</tr>
<tr>
<td></td>
<td>description:</td>
</tr>
<tr>
<td></td>
<td>A range of scale denominators (that can be used to generate a list of tileMatrix names).</td>
</tr>
<tr>
<td></td>
<td>required: false</td>
</tr>
<tr>
<td></td>
<td>style: form</td>
</tr>
<tr>
<td></td>
<td>explode: false</td>
</tr>
<tr>
<td></td>
<td>schema:</td>
</tr>
<tr>
<td></td>
<td>type: array</td>
</tr>
<tr>
<td></td>
<td>minItems: 2</td>
</tr>
<tr>
<td></td>
<td>maxItems: 2</td>
</tr>
<tr>
<td></td>
<td>items:</td>
</tr>
<tr>
<td></td>
<td>type: number</td>
</tr>
<tr>
<td></td>
<td>format: double</td>
</tr>
<tr>
<td></td>
<td>```</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>If the parameter is not specified, the server SHALL assume all TileMatrices (scales) SHALL be returned.</td>
</tr>
</tbody>
</table>

### Recommendation 4

<table>
<thead>
<tr>
<th>/rec/tiles/multitiles/mtc-scaledenominator-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong> To prevent mistakes identifying the scale denominator due to precision issues caused by lack of significant digits, the client should apply a tolerance to intervals. If the client wants to specify a single scale denominator, it will use a small interval with enough tolerance.</td>
</tr>
</tbody>
</table>

### 11.4.5. Parameter multiTileType

<table>
<thead>
<tr>
<th>Requirement 39</th>
<th>/req/tiles/multitiles/mtc-multitiletype-definition</th>
</tr>
</thead>
</table>
The operation SHALL support an optional parameter `multiTileType` that determines the type of the response and with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```
name: multiTileType
in: query
description: 'When successful, the service will respond to a query in one of two ways. It can provide a file with links to each tile or or it will provide the tiles in a package. The package can still contain the description of each tile

The allowed values for this parameter are 'url', 'tiles' and 'full'.

style: form
schema:
  type: string
default: tiles
enum:
  - url
  - tiles
  - full
example: full
```

If the value of the `multiTileType` parameter is set to `url`, the server SHALL return a list of the selected tiles in a format following the `tileSet` schema. Each tile description in the list will contain a URL to download the tile later.

If the value of the `multiTileType` parameter is set to `tiles` or if the parameter is not specified in the request, the server SHALL return a package (e.g. a ZIP file) that will include tiles as separated parts in the package.

If the value of the `multiTileType` parameter is set to `full`, the server SHALL return the tiles and a list of the selected tiles (in a format following the `tileSet` schema) as part of a package.

The server MAY only implement a subset of the enumerated values (url, tiles, full) for the parameter `multiTileType` and in this case it will only enumerate this subset in its schema.
11.4.6. Formats

In the cases of the multi-tile response, there are two formats involved. The multi-tile itself can be returned as a package (e.g. a ZIP file) that contains the tiles inside. The individual tiles also have their format. The format of the multi-tile is governed by the format procedure specified in the OGC API - Common. When the server supports multiple encodings for the individual tiles and the client has a preference for the tiles format, there is a need for communicating this preference to the server. This document does not mandate any particular approach how this is supported but provides the following recommendation.

<table>
<thead>
<tr>
<th><strong>Recommendation 5</strong></th>
<th><strong>/rec/tiles/multitiles/mtc-f-tile-definition</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>When the web interaction allows for HTTP format negotiation, the <code>Accept:</code> header is preferable to specify the required formats. In the case of multi-tile, a composed format is recommended following the pattern <code>application/vnd.ogc.multipart;container={multitile-media-type};tiles={tile-media-type}</code> (example: <code>application/vnd.ogc.multipart;container=application/x-zip-compressed;tiles=image/png</code>).</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>When the web interaction does not allow for controlling the HTTP format negotiation (e.g. URL in a HTML link), the operation MAY support an optional parameter <code>f-tile</code> to specify the tile media type that the client prefers and a parameter <code>f</code> for the media type of the multi-tile response.</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>The content of these parameters should be specified by the server instance as an enumeration of supported media types in the API description.</td>
</tr>
</tbody>
</table>

11.4.7. Response

A successful response to a set of tiles will be consistent with the media type of resource requested.

<table>
<thead>
<tr>
<th><strong>Requirement 40</strong></th>
<th><strong>/req/tiles/multitiles/mtc-success</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code <code>200</code>.</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>The content of that response SHALL be consistent with the format requested and be inside or intersect with the spatial extent of the geographical area represented by the 'bbox' and <code>scaleDenominator</code>.</td>
</tr>
<tr>
<td>C</td>
<td>If a list of the tiles has been requested, the content of that response SHALL contain a tileSet document be based upon the following OpenAPI 3.0 schema:</td>
</tr>
</tbody>
</table>
When a package is being returned and the package format supports expressing file paths of its parts (such as the ZIP file), each tile in the package SHALL have a path following the template: {tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}.{file-extension}. {file-extension} is the file extension that corresponds to the media type (e.g. "jpg" for image/jpeg).

Mainly this extension suggests 3 possible alternatives for a multi-tile response being the last one (full) the combination of the first two (url and package).

**List Response**

This format assumes that the client has a viewport to represent a geographic area defined by the bounding box and the scale (that defines the pixel size of the viewport) in the screen. This area should be populated with tiles. The server is expected to enumerate the tiles needed to populate the viewport and optionally to provide information on how to position the tiles in the viewport.

In the following example, we assume that the bounding box and scale provided implies a viewport of 336x446 pixels (height by width). The viewport is covered by 4 tiles. The client has requested a url type of multi-tile and negotiated a response a JSON format. The URL of each tile is provided, accompanied with information on the position of the top left corner of each one in the viewport.

**Example 20. Example of a tileSet document**
{
  "tileSet": [
    {
      "tileURL": "http://data.example.com/collections/buildings/tiles/WebMercatorQuad/2/0/0.png",
      "tileMatrix": 0,
      "tileRow": 0,
      "tileCol": 0,
      "width": 256,
      "height": 256,
      "top": -10,
      "left": -20
    },
    {
      "tileURL": "http://data.example.com/collections/buildings/tiles/WebMercatorQuad/2/0/1.png",
      "tileMatrix": 0,
      "tileRow": 0,
      "tileCol": 1,
      "width": 100,
      "height": 256,
      "top": -10,
      "left": 236
    },
    {
      "tileURL": "http://data.example.com/collections/buildings/tiles/WebMercatorQuad/2/1/0.png",
      "tileMatrix": 0,
      "tileRow": 1,
      "tileCol": 0,
      "width": 256,
      "height": 200,
      "top": 246,
      "left": -20
    },
    {
      "tileURL": "http://data.example.com/collections/buildings/tiles/WebMercatorQuad/2/1/1.png",
      "tileMatrix": 0,
      "tileRow": 1,
      "tileCol": 1,
      "width": 100,
      "height": 200,
      "top": 246,
      "left": 236
    }
  ]
}
Package Response

This format assumes that the client is interested in the tiles that cover a geographic area defined by the bounding box and the scale (or scales). The client knows what to do with the tiles and it is able to identify the tiles by their path using the URI template of the server as a pattern to extract the TileMatrix, TileRow and TileCol of each one.

Assuming that the client has requested a scale that fits with TileMatrix "2" and a bounding box that requires 2x2 tiles and that the client has requested a package type of multi-tile and negotiated a ZIP format, a ZIP file is produced and sent by the server with the following files and paths:

Table 5. Content of a package containing 4 tiles

<table>
<thead>
<tr>
<th>File</th>
<th>Path</th>
<th>TileMatrix</th>
<th>TileRow</th>
<th>TileCol</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.png</td>
<td>WebMercatorQuad/2/0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.png</td>
<td>WebMercatorQuad/2/0</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0.png</td>
<td>WebMercatorQuad/2/1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1.png</td>
<td>WebMercatorQuad/2/1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

11.4.8. Error conditions

A general summary of the HTTP status codes can be found in OGC API - Common.

If the parameter value tileMatrixSetId is not available by the server for this resource or the parameters values bbox or scaleDenominator are out-of-range, the status code of the response will be 404.
Chapter 12. Requirement Class "Tiles Collections Multi-tiles"

12.1. Overview

Requirements Class

<table>
<thead>
<tr>
<th>Requirement Class</th>
<th><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/cols-multitiles">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/cols-multitiles</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core</a></td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections</a></td>
</tr>
</tbody>
</table>

This requirements class defines a mechanism to request more than one tile from more than one collection in a single request. The result can be a document listing the needed tiles to cover a bounding box or a package with all tiles inside. This section shares most of the content with the previous one and intends to provide similar mechanisms. The main difference is the capability to request tiles that include elements of multiple collections provided by the parameter 'collections'.

12.2. API landing page

The landing page provides links to start exploring the resources offered by the API. It mainly consists of a list of links. The core requirement class of this draft specification does not add anything to the links required by OGC API - Common. The collections extension requires new links for the description of the tiles from more than one collection on top of the common ones that is inherited and needed by this extension.

12.3. Declaration of conformance classes

12.3.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 41</th>
<th>/req/tiles/cols-multitiles/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the capability of generating tiles from multiple collections adding the conformance class for this capability as a link to <a href="http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/cols-multitiles">http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/cols-multitiles</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response of an OGC API - Tiles conformance information page with links to the collections requirements class and this requirements class.
12.4. Tiles description

The response to this operation contains the necessary information to later formulate a tile request from more than one collection as described in the collections extension. This requirement class adds an extra link for the multi-tiles

12.4.1. Response

A successful response to a tiles request for more than one collection will respond with a data structure with specific information necessary to get tiles representing the resource collection. In this extension, the response informs about the URL template to retrieve multi-tiles.

<table>
<thead>
<tr>
<th>Requirement 42</th>
<th>/req/tiles/cols-multitiles/mtcs-multitiles-examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The content of the response to a successful execution SHALL include at least a link to a multi-tiles from multiple collections URI template (rel: items).</td>
</tr>
<tr>
<td>B</td>
<td>These links SHALL provide a URL template with the fragment /tiles followed by the variables {tileMatrixSetId}. Once the variables are substituted by their respective valid values, a URL to a multi-tiles endpoint is obtained.</td>
</tr>
<tr>
<td>C</td>
<td>There SHALL be a link to a multi-tile URI template for each format that the server supports (the format is indicated in the type attribute of the link)</td>
</tr>
</tbody>
</table>

One common order used in URL templates for tiles is .../tiles/{tileMatrixSetId}, but this draft specification allows for other URL template composition.

Table 6. URI template variables for tiles and possible values
### URL template variable

<table>
<thead>
<tr>
<th>URL template variable</th>
<th>Meaning</th>
<th>Possible values</th>
</tr>
</thead>
<tbody>
<tr>
<td>TileMatrixSetId</td>
<td>tile matrix set identifier</td>
<td>The identifiers included in Annex D of OGC 17-083r2 or defined by extensions of the core requirements class.</td>
</tr>
</tbody>
</table>

#### Example 22. API tiles response fragment

```json
links:
[
{
  "href": "http://data.example.com/tiles/{tileMatrixSetId}",
  "rel": "items",
  "type": "image/png",
}
]
```

### 12.5. Multiple tiles from more than one collection

This extension provides a mechanism to select and retrieve a set of tiles at once from a TileMatrixSet.

#### 12.5.1. Operation

<table>
<thead>
<tr>
<th>Requirement 43</th>
<th>/req/tiles/multitiles/mtcs-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Tiles SHALL be available as HTTP GET requests to a URI that will be composed by two parts: the first part is the URI of a resource that can be represented as tiles and the second part follows the pattern /tiles/{tileMatrixSetId}</td>
</tr>
<tr>
<td>B</td>
<td>Only the resources or collections that advertise one of more links with type=tiles SHALL be requested as multiple tiles.</td>
</tr>
</tbody>
</table>

#### 12.5.2. Parameter tileMatrixSetId

| Requirement 44 | /req/tiles/multitiles/mtcs-tilematrixsetid-definition |
The operation SHALL support a parameter `tileMatrixSetId` with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```json
  name: tileMatrixSetId
  in: path
  description: Identifier of a specific tiling scheme. It can be one of the specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.
  required: true
  schema:
    type: string
    example: WebMercatorQuad
```

### 12.5.3. Parameter `bbox`

<table>
<thead>
<tr>
<th>Requirement 45</th>
<th>/req/tiles/multitiles/mtcs-bbox-definition</th>
</tr>
</thead>
</table>
The operation SHALL support an optional parameter `bbox` to filter the area where tiles will be retrieved with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```json
name: bbox
in: query

description: 'Only elements that have a geometry that intersects the bounding box are selected. The bounding box is provided as four or six numbers, depending on whether the coordinate reference system includes a vertical axis (elevation or depth):

* Lower left corner, coordinate axis 1
* Lower left corner, coordinate axis 2
* Lower left corner, coordinate axis 3 (optional)
* Upper right corner, coordinate axis 1
* Upper right corner, coordinate axis 2
* Upper right corner, coordinate axis 3 (optional)

The coordinate reference system of the values is WGS 84 longitude/latitude ([http://www.opengis.net/def/crs/OGC/1.3/CRS84](http://www.opengis.net/def/crs/OGC/1.3/CRS84)) unless a different coordinate reference system is specified by another parameter in the API (e.g `bbox-crs`).'

required: false

schema:
  type: array
  minItems: 4
  maxItems: 6
  items:
    type: number
    format: double
    style: form
    explode: false
```

A TileMatrixSet definition points to a CRS. The coordinates of the `bbox` SHALL be in the CRS of specified in the definition of the TileMatrixSet identified by the `tileMatrixSetId`.

If the parameter is not specified, the server SHALL assume the whole extent of the tiles are requested.

This definition is inherited from OGC API - Common.
### 12.5.4. Parameter scaleDenominator

**Requirement 46**

<table>
<thead>
<tr>
<th>A</th>
<th>The operation SHALL support an optional parameter <code>scaleDenominator</code> to filter the scales where tiles will be retrieved with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):</th>
</tr>
</thead>
</table>
|  | ```
name: scaleDenominator
in: query
description: 'A range of scale denominators (that can be used to generate a list of tileMatrix names).'
required: false
style: form
explore: false
schema:
  type: array
  minItems: 2
  maxItems: 2
  items:
    type: number
    format: double
``` |
| B | If the parameter is not specified, the server SHALL assume all TileMatrices (scales) SHALL be returned. |

**Recommendation 6**

| A | To prevent mistakes identifying the scale denominator due to precision issues caused by lack of significant digits, the client should apply a tolerance to intervals. If the client wants to specify a single scale denominator, it will use a small interval with enough tolerance. |

### 12.5.5. Parameter multiTileType

**Requirement 47**

<table>
<thead>
<tr>
<th>A</th>
<th></th>
</tr>
</thead>
</table>
The operation SHALL support an optional parameter `multitileType` that determines the type of the response and with the following characteristics (shown as an OpenAPI Specification 3.0 fragment):

```yaml
name: multitileType
in: query
description: 'When successful, the service will respond to a query in one of two ways. It can provide a file with links to each tile or it will provide the tiles in a package. The package can still contain the description of each tile.

The allowed values for this parameter are 'url', 'tiles' and 'full'."
style: form
schema:
type: string
default: tiles
enum:
  - url
  - tiles
  - full
example: full
```

| A | The server MAY only implement a subset of the enumerated values (url, tiles, full) for the parameter `multitileType` and in this case the server will only enumerate this subset in its schema. |
| B | If the value of the `multitileType` parameter is set to `url` the server SHALL return a list of the selected tiles in a format following the `tileSet` schema. Each tile description in the list will contain a URL to download the tile later. |
| C | If the value of the `multitileType` parameter is set to `tiles` or if the parameter is not specified in the request, the server SHALL return a package (e.g. a ZIP file) that will include tiles as separated parts in the package. |
| D | If the value of the `multitileType` parameter is set to `full` the server SHALL return the tiles and a list of the selected tiles (in a format following the `tileSet` schema) as part of a package. |

**Permission 3**

/`per/tiles/multitiles/mtcs-multitiletype-definition`
### 12.5.6. Parameter Collections

<table>
<thead>
<tr>
<th>Requirement 48</th>
<th>/req/tiles/collections/mtcs-collections-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>The operation SHALL support an optional parameter <code>collections</code> with the following characteristics (shown as an OpenAPI Specification 3.0 fragment)</td>
</tr>
</tbody>
</table>

```yaml
name: collections
in: query
required: false
style: form
explode: false
schema:
  type: array
  items:
    type: string
```

| **B**          | `collections` SHALL contain a comma-separated list of collection identifiers. |
| **C**          | Only the collections that advertise a link type=tiles in the /collections/{collectionId} SHALL be included. |
| **D**          | Only the collections that support the same TileMatrixSetId parameter value SHALL be included |
| **C**          | If `collections` is missing, all collections supporting the TileMatrixSetId parameter value will be considered. |

### 12.5.7. Formats

In the cases of the multi-tile response, there are two formats involved. The multi-tile itself can be returned as a package (e.g. a ZIP file) that contains the tiles inside. The individual tile also has its own format. The format of the multi-tile is governed by the format procedure specified in the OGC API – Common draft specification. When the server supports multiple encodings for the individual tiles and the client has a preference for the tiles format, there is a need for communicating this preference to the server. This document does not mandate any particular approach for how this is supported but provides the following recommendation.

| Recommendation 7 | /rec/tiles/multitiles/mtcs-f-tile-definition |
The operation MAY support an optional parameter `f-tile` to specify the tile format that the client prefers as parts of the multi-tile response.

The content of this parameter should be specified by the server instance as an enumeration of supported media types.

### 12.5.8. Response

A successful response for a set of tiles will be consistent with the media type of the resource requested. This draft specification does not impose any media type but suggests the use of a package format.

<table>
<thead>
<tr>
<th>Requirement 49</th>
<th>/req/tiles/cols-multitiles/mtcs-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution of the operation SHALL be reported as a response with a HTTP status code <code>200</code>.</td>
</tr>
<tr>
<td>B</td>
<td>The content of that response SHALL be consistent with the format requested and be inside or intersect with the spatial extent of the geographical area represented by the 'bbox' and <code>scaleDenominator</code>.</td>
</tr>
</tbody>
</table>
If a list of the tiles has been requested, the content of that response SHALL contain a tileSet document be based upon the following OpenAPI 3.0 schema:
When a package is being returned and the package format supports expressing file paths of its parts (such as the ZIP file), each tile in the package SHALL have a path following the template:
{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}.{file-extension}. {file-extension} is the file extension that corresponds to the media type (e.g. "jpg" for image/jpeg).

12.5.9. Error conditions

A general summary of the HTTP status codes can be found in OGC API - Common.

If the parameter value of the parameter tileMatrixSetId is not available by the server for this resource or the values of the parameters bbox or scaleDenominator are out-of-range, the status code of the response will be 404.
Chapter 13. Requirement Class "Maps Core"

13.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core</a></td>
</tr>
<tr>
<td>Target type</td>
</tr>
<tr>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
</tr>
<tr>
<td>RFC 2616 (HTTP/1.1)</td>
</tr>
<tr>
<td>RFC 2818 (HTTP over TLS)</td>
</tr>
<tr>
<td>RFC 3339 (Date and Time on the Internet: Timestamps)</td>
</tr>
<tr>
<td>RFC 8288 (Web Linking)</td>
</tr>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core</a></td>
</tr>
<tr>
<td><a href="http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections</a></td>
</tr>
</tbody>
</table>

A map distribution of a dataset is a pictorial representation of the dataset or the collections it has been divided in. To create a pictorial representation a style is added to the data in the collections. Styles are defined internally and have a identifier. New styles can be added or modified if the OGC API - Maps draft specification works in combination with the OGC API - Styles draft specification (also developed in the Testbed-15). After associating collections to styles, a map can be retrieved by specifying a set of parameters that will determine its resolution (width, height, bounding box and CRS) or can be retrieved as tiles.

This section defines the core part of the OGC API - Maps draft specification that allows defining a map representation for a collection. To retrieve a fragment of the map, this section needs to be combined with an OGC API - Tiles draft specification or with the OGC API - Maps - Map extension draft specification.

To keep the core of the OGC API - Maps draft specification simple, only includes a mechanism to select the default style but it does not define any mechanism to define or select a style other than the default one. The core specification only assumes that the service is capable of knowing which is the default style while the client ignores all the details about it (including its name).

13.2. General

<table>
<thead>
<tr>
<th>Requirement 50</th>
<th>/req/maps/core/api-common</th>
</tr>
</thead>
<tbody>
<tr>
<td>The OGC API SHALL comply with the requirements specified in the <a href="http://www.opengis.net/spec/OAPI_Common/1.0/req/core">http://www.opengis.net/spec/OAPI_Common/1.0/req/core</a> and collections Requirements Classes of the OGC API-Common version 1.0 Standard.</td>
<td></td>
</tr>
</tbody>
</table>

In practice, this means that the landing page and the conformance page follow OGC API - Common core and collections requirements. This draft specification provides additions to the OGC API -
Common requirements that are particular to maps use case.

13.3. API landing page

The landing page provides links to start exploring the resources offered by the API instance. It mainly consists of a list of links. OGC API - Common already requires some common links that are enough for this core.

13.3.1. Response

There are no required variations to the landing page.

13.4. Declaration of conformance classes

To support "generic" clients that want to access multiple OGC API standards and extensions - and not "just" a specific API / server, the API has to declare the requirements classes it implements and conforms to.

13.4.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 51</th>
<th>/req/maps/core/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the maps core conformance class as links to <a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response of an OGC API maps conformance information page.

Example 23. Conformance Information Page fragment

```json
{
    "conformsTo": [
        "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
        "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
        "http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core"
    ]
}
```
13.5. Collections

This draft specification includes dependencies on OGC API - Common collections. Collections are mandatory in the core of this draft specification because collections are the object that can eventually be included in a map.

Collections enumerate the collectionId identifiers available in this API as well as basic information about each collectionId: id, title, description, extent, crs and links. This common response is considered enough for a general description of the collection.

<table>
<thead>
<tr>
<th>Requirement 52</th>
<th>/req/maps/core/mc-md-collection-links</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>For each collection included in the response, a <strong>links</strong> property of the collection SHALL include a link to the description of the collection (rel: <strong>item</strong>) (in addition to other links specified in OGC API Commons).</td>
</tr>
</tbody>
</table>

More specific details about the collection can be found following the link to the individual collections that follow the pattern /collections/{collectionId}

**NOTE**

The collectionId substitutes the concept of "layer" in WMS.

**NOTE**

In WMS, layers have a hierarchical dependency. The authors believe it is the responsibility of OGC API - Common to provide this functionally. At the time of writing this draft specification the OWS Common SWG has not yet considered this possibility.

13.6. Maps description

The maps core defines a **maps** resource that is associated with an operation that contains the necessary information to later formulate a map request for a collection. Nevertheless, the core does not require any mandatory information since the map core alone does not specify how to retrieve a map. This core does not mandate a map description operation. The map description cannot be described without considering other OGC API - Maps extensions.

13.6.1. Map description response

This core does not mandate a map description operation. Nevertheless, if it is defined by an OGC API - Maps extension, the core introduces recommendations for having two specific properties in the response of a map description that are inherited from WMS: cascade and opaque.

<table>
<thead>
<tr>
<th>Recommendation 8</th>
<th>/rec/maps/core/smc-opaque</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server may include a boolean property in the maps description response that contains a boolean property <strong>opaque</strong>.</td>
</tr>
</tbody>
</table>
‘false’ means that map data represents vector features that probably do not completely fill space. ‘true’ means map data are mostly or completely opaque.

If the property is not provided, it should be interpreted as false (the default value)

**Recommendation 9**

The server may include a numeric property in the map description response with the name `cascaded`.

0 means that the collection maps have not been retransmitted another map service or API. A positive number indicates how many times the collection map has been retransmitted.

If the property is not provided, it should be interpreted as 0 (the default value)

## 13.7. Maps

This OGC API - Maps core draft specification does not specify how to retrieve a map but it does specify that in order for a map to be retrievable a parameter `styleId` should be added to any operation that retrieves a map as maps or as tiles.

### 13.7.1. Operation

**Requirement 53**

Every map SHALL be available as a HTTP GET request to a URI that will be composed by three parts: the first part is the URI of a resource that can be represented as a map, the second part following the pattern `/map/{styleId}` and the third part completing the retrieval parameters

Only the resources (e.g. collection) that advertise one of more links following the pattern `.../map/{styleId}...` in the maps metadata can be retrieved as maps.

### 13.7.2. Parameter `styleId`

**Requirement 54**

`/req/maps/core/mc-styleId-definition`
<table>
<thead>
<tr>
<th>A</th>
<th>The operation SHALL support a parameter <code>styleId</code> with the characteristics defined (shown as OpenAPI Specification 3.0 fragment)</th>
</tr>
</thead>
</table>
|   | ```yaml
name: styleId
in: path
description:
  'The styleId that should be included in the map or tile. Each collectionId has a valid list of stylesId. To know the valid styleId values of each collectionId use /collections/{collectionId}.'
required: true
schema:
  type: string
``` |
| B | A map SHALL be available with `default` as `styleId` value. The server decides which is the default style. `default` is the only value defined by the core and other values might be defined as extensions. |
Chapter 14. Requirement Class "Map Styles"

14.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
<th><a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/styles">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/styles</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

The core of the OGC API - Maps draft specification introduces the possibility of creating a map by assigning a style to a resource (e.g. a collectionId) but does not specify how to declare the styles supported by each collection. Only with the core specification an API instance is only capable to request the default style and the client does not know anything about it. This requirement class extends the core requirements by specifying how to declare style names other than default that can be used to request maps. The OGC API - Maps draft specification implements some functionality, specified in the Web Map Tile Service (WMTS) 1.0 standard, related to the use of styles by using the Styles API draft specification that was developed in the Testbed-15 Open Portrayal Framework thread [2]. The OGC API - Styles draft specification will allow for the retrieval of the complete information about the style or to send new styles to the server.

14.2. Declaration of conformance classes

14.2.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 55</th>
<th>/req/maps/styles/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the capability of declaring styles by adding the conformance class for this capability as a link to <a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/styles">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/styles</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response of an OGC API - Maps conformance information page that declares support for the core and the styles extension.
**Example 24. Conformance Information Page fragment**

```json
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
    "http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core",
    "http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/styles"
  ]
}
```

### 14.3. Collection

This draft specification includes dependencies on OGC API - Common collection. The response to the operation is extended with the necessary information to formulate a map response for this collection.

#### 14.3.1. Collection Links to styles

This extension describes how to provide a list of styles in the collection description.

<table>
<thead>
<tr>
<th>Requirement 56</th>
<th>/req/maps/styles/smc-styles</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A successful execution SHALL contain a property called <code>styles</code> that enumerates a list of the styles available for the collection.</td>
</tr>
</tbody>
</table>

Each style in *styles* is an object that SHALL conform with the following data mode (shown as OpenAPI Specification 3.0 fragment):

```json

type: object
required:
  - id
properties:
  id:
    type: string
    nullable: true
  title:
    type: string
    nullable: true
  links:
    type: array
    nullable: true
    minItems: 1
    items:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
```
The mandatory element id can be used as a value for {styleId}.

The optional links element is useful for connecting to an OGC API – Styles implementation that allows for retrieving the styles description.

<table>
<thead>
<tr>
<th>Recommendation</th>
<th>/rec/maps/styles/smc-default-style</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 A</td>
<td>A successful execution may contain a property called defaultStyle points to the default style used when {styleId} is replaced by the word default</td>
</tr>
</tbody>
</table>
The value of the default style SHOULD be one of the ids listed in the property `styles`.

Each style in `styles` is an object that conforms with the following data mode (shown as OpenAPI Specification 3.0 fragment):

```json
{  "default-style": {    "type": "string",    "description": "the style id of a recommended default style to use for this collection. This is informative and optional.
example: 'topographic'"  }
}
```

Example 26. API collection response fragment

```
"defaultStyle": "topographic"
```

### 14.4. Maps description

The core of the OGC API - Maps draft specification defines maps resource that is associated with an operation that contains the necessary information to later formulate a map request for a collection. Nevertheless, the core does not require any mandatory information. This requirement class does not require any mandatory information, but the response of the operation is conditioned by the availability of more than one style per collection.

### 14.5. Maps

This OGC API - Maps style draft specification extension does not specify how to retrieve a map, but it does specify two parameters (`transparent` and `bgcolor`) in addition to the `styleId` defined in the core.

#### 14.5.1. Operation

#### 14.5.2. Parameter styleId

Apart from the `default` style value, this extension introduces the values for the `styleId` that were presented in the `collectionId` definition.

#### 14.5.3. Parameter transparent

<table>
<thead>
<tr>
<th>Requirement 57</th>
<th>/req/maps/styles/mc-transparent-definition</th>
</tr>
</thead>
</table>
The operation SHALL support an optional parameter `transparent` to force a transparent background with the characteristics defined (shown as OpenAPI Specification 3.0 fragment):

```yaml
name: transparent
in: query
description: 'Background transparency of map (default=true).'</ndescription:
required: false
style: form
explode: false
schema:
  type: boolean
default: true
```

If `transparent` is not specified, the server will use `true`.

14.5.4. Parameter `bgcolor`

**Requirement 58** `/req/maps/styles/mc-bgcolor-definition`

The operation SHALL support an optional parameter `bgcolor` to define a background color with the characteristics defined (shown as OpenAPI Specification 3.0 fragment):

```yaml
name: bgcolor
in: query
description: Hexadecimal red-green-blue[-alpha] color value for the background color. If alpha is not specified a binary opacity will be used depending on the transparent parameter.
required: false
style: form
explode: false
schema:
  type: string
default: #FFFFFF
```

If `bgcolor` is not specified, the server is free to choose a background color that's appropriate for the requested style, or `0xFFFFFFFF` (white) if no such information is available.

If the client wants to force an opaque color, apart from defining the appropriate background color...
it should ensure that the parameter `transparent` is set to `false`. For the formats that reserve a color to define transparency, it still makes sense to combine background color and `transparent=true` with the purpose of helping the server to select a color that does not interfere with the actual values and colors in the map.
Chapter 15. Requirement Class "Map Maps"

NOTE This section should be elaborated by a SWG and only some hints are provided in this Engineering Report. This section is out of the scope of the Testbed-15.

WARNING Some subsections are intentionally left blank.

15.1. Overview

This extension describes how a map can be retrieved by specifying a set of parameters that will determine its resolution (width, height, boundingbox and CRS).

The map can use the default style or it can select one of the styles available if the right extension is also added to the core.

15.2. General

In practice, this means that the landing page and the conformance page follow OGC API - Common core requirements. More is still TBD but mostly equivalent to the general parts of OGC API - Features requirements, though with the text generalized to other resource types. This draft specification provides extra additions to the OGC API - Common requirements that are particular of tiles.

15.3. API landing page

The landing page provides links to start exploring the resources offered by the API. It mainly consists of a list of links. OGC API - Common already requires some common links that are enough for this core.

15.3.1. Response

There are no required variations to the landing page.

With a /collections successful response it is possible to retrieve the list of collectionId and links to the /collections/{collectionId}. With a /collections/{collectionId} successful response, it is possible to discover the links to retrieve some maps. Note that other resources can also be retrieved as collections (e.g. coverages).

15.4. Declaration of conformance classes

To support "generic" clients that want to access multiple OGC API standards and extensions - and not "just" a specific API / server, the API has to declare the requirements classes it implements and conforms to.
15.4.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response of an implementation of the OGC API – Maps draft specification with the maps extension conformance information page.

*Example 27. Conformance Information Page fragment*

```json
{
   "conformsTo": [
      "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core",
      "http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections",
      "http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core",
      "http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/maps"
   ]
}
```

15.5. Collections

This draft specification includes dependencies on OGC API - Common collections. Collections are mandatory in the core of this draft specification because collections are the object that will be included in a tile.

Collections will enumerate the collectionId available in this API as well as basic information about each collectionId: id, title, description, extent, crs and links. This common response is considered enough for a general description of the collection. To retrieve more information, you should use /collections/{collectionId}

**NOTE** The collectionId substitutes the concept of "layer" in WMS.

15.6. Collection

This draft specification includes dependencies on OGC API - Common collection. The response to the operation is extended with a new link for the maps description.

15.6.1. Collection Links

TBD
15.6.2. Maps metadata

The core of the OGC API - Maps draft specification defines a maps resource that is associated to an operation contains the necessary information to later formulate a map request for a collection. Neither the core, nor the styles extension requires any mandatory information. This requirement class does require this operation to be able to retrieve a map as maps (this resource will not be present if the map is only available as tiles).

15.6.3. Operation

The request of this operation has no parameters.

15.6.4. Response

A successful response to a map request for a collection that can be retrieved as a map responds with a data structure with specific information necessary to get a fragment of the map representing the resource collection. In this extension, the response is only required to inform about the URL templates styles it supports.

In practice, since the map core alone does not specify how to retrieve a map, it is not possible to exemplify completely how the link looks like without considering other extensions. If the server also conforms to an extension to distribute the map as maps, the example will look like this.

*Example 28. API collection response fragment*

```json
links:
   [
   {
      "href": "http://data.example.com/collections/buildings/map/brown",
      "rel": "item",
      "type": "image/png",
   }
   ]
```

15.7. Maps from one collection

This draft specification specifies how to get maps from a single collection.

15.7.1. Operation

Typical resources that can be retrieved as maps are: features (/collections/{collectionId}), coverages (/collections/{collectionId}/coverage/{coverageId} or /coverage/{coverageId}).

15.7.2. Parameter crs

TBD
15.7.3. Parameter bbox
TBD

15.7.4. Parameter width
TBD

15.7.5. Parameter height
TBD

15.7.6. Response
A successful response of a tile request will be consistent with the media type of the resource requested. For features the media type can be geojson, GML or Mapbox vector tiles; for coverages it may be a GeoTIFF, GMLJP2, netCDF; and for maps it can be a JPEG, GMLJP2 or a PNG.

15.7.7. Error conditions
A general summary of the HTTP status codes can be found in OGC API - Common.

If the parameter crs is not available by the server for this resource or the parameters bbox, width, height are out-of-range, the status code of the response will be 404.
Chapter 16. Requirement Class "Map from more than one collection"

16.1. Overview

<table>
<thead>
<tr>
<th>Requirements Class</th>
<th><a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/collections</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Target type</td>
<td>Web API</td>
</tr>
<tr>
<td>Dependency</td>
<td><a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/core</a></td>
</tr>
</tbody>
</table>

In previous clauses maps that are produced form one and only one resource is discussed. This is achieved by concatenating the map path to a resource (e.g. a feature collection). This extension discusses the possibility of combining more than one resource to create a map. This is achieved by using by adding the map path to the root of the service.

16.2. Declaration of conformance classes

16.2.1. Response

The conformance page mainly consists of a list of links. OGC API - Common already requires some links.

<table>
<thead>
<tr>
<th>Requirement 59</th>
<th>/req/maps/collections/conformance-success</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The API conformance path SHALL advertise the capability of generating maps from multiple collections by adding the conformance class for this capability as a link to <a href="http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/collections">http://www.opengis.net/spec/ogcapi-maps-1/1.0/req/collections</a>.</td>
</tr>
</tbody>
</table>

In the conformance page (typically in JSON format) the links follow the link schema defined in the OGC API - Common draft specification. The following is an example fragment of the response of an OGC API - Maps conformance information page
16.3. Maps from more than one collection

16.3.1. Operation

<table>
<thead>
<tr>
<th>Requirement 60</th>
<th>/req/maps/collections/mcs-op</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The server SHALL support the HTTP GET operation at the path /maps</td>
</tr>
</tbody>
</table>

16.3.2. Parameter styles

<table>
<thead>
<tr>
<th>Requirement 61</th>
<th>/req/maps/collections/mcs-styles-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The operation SHALL support an optional parameter styles with the characteristics defined (shown as an OpenAPI Specification 3.0 fragment)</td>
</tr>
</tbody>
</table>

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

| B              | The parameter value SHALL be a list of comma-separated styles identifiers. If the parameter 'collections' exists, the list should be as long as 'collections' and each style identifier corresponds to one collection identifier. Default style can be represented as a blank name or with the default word |

If the parameter is missing, the default style is assumed for all collections enumerated.

### 16.3.3. Parameter Collections

<table>
<thead>
<tr>
<th>Requirement 62</th>
<th>/req/maps/collections/mcs-collections-definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>The operation SHALL support an optional parameter <code>collections</code> with the following characteristics (shown as an OpenAPI Specification 3.0 fragment)</td>
</tr>
<tr>
<td></td>
<td>```json</td>
</tr>
<tr>
<td></td>
<td>name: collections</td>
</tr>
<tr>
<td></td>
<td>in: query</td>
</tr>
<tr>
<td></td>
<td>required: false</td>
</tr>
<tr>
<td></td>
<td>style: form</td>
</tr>
<tr>
<td></td>
<td>explode: false</td>
</tr>
<tr>
<td></td>
<td>schema:</td>
</tr>
<tr>
<td></td>
<td>type: array</td>
</tr>
<tr>
<td></td>
<td>items:</td>
</tr>
<tr>
<td></td>
<td>type: string</td>
</tr>
<tr>
<td>B</td>
<td><code>collections</code> SHALL contain a comma-separated list of collection identifiers.</td>
</tr>
<tr>
<td>C</td>
<td>Only the collections that advertise a link following the <code>/map/{styleId}</code>... in the <code>/collections/{collectionId}</code> SHALL be included.</td>
</tr>
<tr>
<td>D</td>
<td>Only the collections that support the same CRS or the same <code>tileMatrixSetId</code> parameter value SHALL be included.</td>
</tr>
<tr>
<td>C</td>
<td>If <code>collections</code> is missing, all collections supporting the <code>crsId</code> or the <code>tileMatrixSetId</code> parameter value will be considered.</td>
</tr>
</tbody>
</table>

### 16.3.4. Response

To retrieve the map as a map or a tile another extension is needed. No requirements are provided here.
Appendix A: Abstract Test Suite


Examples of Abstract Test Suites are available in an annex of most ISO 191XX documents, one of the more useful is in ISO 19136. Note that this guidance may be more abstract than needed in an OGC® Implementation Standard.

NOTE The Abstract Test Suite will be defined and documented by the WMS SWG and is therefore not provided in this Engineering Report.
Appendix B: OpenAPI Domain

B.1. OpenAPI Domain common

This is the OGC API - Common domain OpenAPI developed and used in Testbed-15. This might be used and continued in the OWS Common SWG in the future. This is based on and extracted from OpenAPI examples for the OGC API - Features standard.

This corresponds to a URL .../ogc-api-map-common/1.0.0

```yaml
openapi: 3.0.2
info:
  title: OGC API Common
  description: |
    Common components used in the OGC API family of standards. Deeply inspired in Clemens Portele work.

    This document is also available in [GitHub](https://github.com/opengeospatial/oapi_common/tree/master/OAPI-Common/openapi).

    This document copies or is inspired in [GitHub](https://github.com/opengeospatial/WFS_FES/blob/master/core/openapi/bbox/ogcap_i-features-1.yaml) from Clemens Portele

  version: '1.0.0'
  contact:
    name: Joan Maso
    email: joan.maso@uab.cat
  license:
    name: OGC License

components:
  parameters:
    bbox:
      name: bbox
      in: query
      description: |
        Only elements that have a geometry that intersects the bounding box are selected.
        The bounding box is provided as four or six numbers, depending on whether the coordinate reference system includes a vertical axis (elevation or depth):

        * Lower left corner, coordinate axis 1
        * Lower left corner, coordinate axis 2
```
The coordinate reference system of the values is WGS 84 longitude/latitude (http://www.opengis.net/def/crs/OGC/1.3/CRS84) unless a different coordinate reference system is specified by another parameter in the API (e.g. 'bbox-crs').

For WGS 84 longitude/latitude the values are in most cases the sequence of minimum longitude, minimum latitude, maximum longitude and maximum latitude. However, in cases where the box spans the antimeridian the first value (west-most box edge) is larger than the third value (east-most box edge).

If an element has multiple spatial geometry properties, it is the decision of the server whether only a single spatial geometry property is used to determine the extent or all relevant geometries.

required: false

schema:
  type: array
  minItems: 4
  maxItems: 6
  items:
    type: number
    format: double
  style: form
  explode: false

collectionId:
  name: collectionId
  in: path
  description: local identifier of a collection
  required: true
  schema:
    type: string

datetime:
  name: datetime
  in: query
  description: |
    Either a date-time or an interval, open or closed. Date and time expressions adhere to RFC 3339. Open intervals are expressed using double-dots.

Examples:

* A date-time: "2018-02-12T23:20:50Z"
* A closed interval: "2018-02-12T00:00:00Z/2018-03-18T12:31:12Z"
* Open intervals: "2018-02-12T00:00:00Z/.." or "../2018-03-18T12:31:12Z"

Only elements that have a temporal property that intersects the value of 'datetime' are selected.
If an element 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.

```json
required: false
schema:
  type: string
style: form
explode: false
limit:
  name: limit
  in: query
  description: |
    The optional limit parameter limits the number of items that are presented in the response document.

    Only items are counted that are on the first level of the collection in the response document. Nested objects contained within the explicitly requested items shall not be counted.

required: false
schema:
  type: number
  format: integer
  minimum: 0
  maximum: 10000
  default: 10
style: form
explode: false
offset:
  name: offset
  in: query
  description: |
    The optional offset parameter indicates the index within the result set from which the server shall begin presenting results in the response document. The first element has an index of 0.

    If offset is greater or equal to the number of items in the collection, the server should return an empty list.

    Minimum = 0. Default = 0.
required: false
style: form
explode: false
schema:
  type: number
  format: integer
  minimum: 0
  default: 0
example: 0
```
resultType:

name: resultType
in: query
description: |

This service will respond to a query in one of two ways (excluding an exception response). It may either generate a complete response document containing resources that satisfy the operation or it may simply generate an empty response container that indicates the count of the total number of resources that the operation would return. Which of these two responses is generated is determined by the value of the optional resultType parameter.

The allowed values for this parameter are "results" and "hits".

If the value of the resultType parameter is set to "results", the server will generate a complete response document containing resources that satisfy the operation.

If the value of the resultType attribute is set to "hits", the server will generate an empty response document containing no resource instances.

The default value is "results".

required: false
style: form
explode: false
schema:
  type: string
default: results
enum:
  - hits
  - results
example: results

f-json-html:

name: f
in: query
description: |

The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.

Pre-defined values are "json" and "html". The response to other values is determined by the server.

required: false
style: form
explode: false
schema:
  type: string
enum:
  - application/json
  - text/html
example: application/json
f-json:
  name: f
  in: query
  description: |
    The format of the response. If no value is provided, the standard http
    rules apply, i.e., the accept header is used to determine the format.
    
    The only pre-defined value is "json". The response to other values is
    determined by the server.
  required: false
  style: form
  explode: false
  schema:
    type: string
    enum:
      - application/json
    example: application/json
  schemas:
    collection:
      # This object does not include the links element that should be added as and
      additional element using a personlized version of collection-link with allOf
      type: object
      required:
        - id
      properties:
        id:
          description: identifier of the collection used, for example, in URIs
          type: string
          example: buildings
        title:
          description: human readable title of the collection
          type: string
          example: Buildings in the city of Bonn
        description:
          description: a description of the collection
          type: string
          example: This collection contains buildings
        keywords:
          description: keywords about the elements in the collection
          type: array
          items:
            $ref: '#/components/schemas/keyword'
        attribution:
          description: |
            The provider of the source data for the collection. Map viewers normally
            show this information at the bottom of the map
          type: string
          example: OpenStreetMap
        extent:
          $ref: '#/components/schemas/extent'
#itemType:
  Deprecated. The links are indicating the availability of a collection as features, coverages etc
  
#  description: indicator about the type of the items in the collection (the default value is 'feature'; alternative values are 'coverage', 'image', etc).
#  type: string
#  default: feature
#  example: feature

crs:
  
  description: The list of coordinate reference systems supported by the service. The first item is the default coordinate reference system.
  
  type: array
  items:
    type: string
    default:
      - 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'
  example:
    - 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'
    - 'http://www.opengis.net/def/crs/EPSG/0/4326'

# crsLibrary:
# this is just an idea for preventing long lists of CRSs. We have to see if there is consensus adopting it.
#  type: string
#  description: |
#    Reference to a CRS library giving support to a comprehensive list of CRSs that are not advertised but supported anyway.
#  example: PROJ4G

crsSpatialExtents:
  type: array
  description: |
    Minimum spatial extent surrounding the spatial resource for each CRS available
  items:
    $ref: '#/components/schemas/spatialExtent'
  example:
    - bbox:
        - -180
        - -90
        - 180
        - 90
        crs: 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'
    - bbox:
        - -20037508.3427892
        - -20037508.3427892
        - 20037508.3427892
        - 20037508.3427892
        crs: 'http://www.opengis.net/def/crs/EPSG/0/3395'

collection-link:
  
  # This element is only used by '/collections' and is not directly by other APIs in '/collection/{collectionId}' because they probably will need to add other links to other resource types in the examples. Instead, it would be copied and
enriched with the right examples.

```json

`type: object`

required:
- `links`

properties:

  `links`:
  type: array
  items:
  $ref: '#/components/schemas/link'
  example:
  - $ref: '#/components/examples/link-collection-from-collections'
  - $ref: '#/components/examples/link-collection-describedBy'
  - $ref: '#/components/examples/link-collection-license-html'
  - $ref: '#/components/examples/link-collection-license-rdf'

`collections`:

  type: object
  required:
  - `links`
  - `collections`

  properties:

    `links`:
    type: array
    nullable: true
    items:
    $ref: '#/components/schemas/link'
    example:
    - `href: 'http://data.example.org/collections?f=json'`
      `rel: self`
      `type: application/json`
      `title: this document`
    - `href: 'http://data.example.org/collections?f=html'`
      `rel: alternate`
      `type: text/html`
      `title: this document as HTML`
    - `href: 'http://schemas.example.org/1.0/dataset.xsd'`
      `rel: describedBy`
      `type: application/xml`
      `title: GML application schema for Acme Corporation dataset data`
    - `href: 'http://download.example.org/dataset.gpkg'`
      `rel: enclosure`
      `type: application/geopackage+sqlite3`
      `title: Bulk download (GeoPackage)`
      `length: 472546`

`collections`:

  type: array
  items:
  allOf:
  - $ref: '#/components/schemas/collection'
  - $ref: '#/components/schemas/collection-link'

`confClasses`:

  type: object
```
required:
  - conformsTo
properties:
  conformsTo:
    type: array
    items:
      type: string
      format: uri
      example:
        - 'http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core'
exception:
  type: object
  required:
    - code
  properties:
    code:
      type: string
    description:
      type: string
      example:
        code: '500'
        description: 'An internal server error occured. Incident ID: 1234567. Please contact admin@example.org.'
extent:
  description: |-  
    The extent of the collection. In the Core only spatial and temporal extents are specified. Extensions may add additional members to represent other extents, for example, thermal or pressure ranges. It is recommended that the statial extent is expected in CRS84 except if this is not possible.
  type: object
  properties:
    spatial:
      $ref: '#/components/schemas/spatialExtent'
    temporal:
      $ref: '#/components/schemas/temporalExtent'
spatialExtent:
  description: |-  
    The spatial extent of the element in the collection.
  type: object
  required: bbox
  properties:
    bbox:
      $ref: '#/components/schemas/bbox'
crs:
    $ref: '#/components/schemas/crs'
bbox:
  description: |-  
    One or more bounding boxes that describe the spatial extent of the dataset. In the Core only a single bounding box is supported. Extensions may support additional areas. If multiple areas are provided, the union of the bounding
boxes describes the spatial extent.

type: array

minItems: 1

items:

description: |

West, south, east, north edges of the bounding box. The coordinates are in the coordinate reference system specified in 'crs'. By default this is WGS 84 longitude/latitude.

type: array

minItems: 4

maxItems: 6

items:

type: number

eample:

- 7.01
- 50.63
- 7.22
- 50.78

crs:

description: |

Coordinate reference system of the coordinates in the spatial extent (property 'bbox'). The default reference system is WGS 84 longitude/latitude.

In the Core this is the only supported coordinate reference system. Extensions may support additional coordinate reference systems and add additional enum values.

type: string

enum:

- 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'

default: 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'

temporalExtent:

description: |

The temporal extent of the element in the collection.

type: object

nullable: true

properties:

interval:

$ref: '#/components/schemas/temporalInterval'

trs:

$ref: '#/components/schemas/trs'

temporalInterval:

description: |

One or more time intervals that describe the temporal extent of the dataset. The value 'null' is supported and indicates an open time interval.

In the Core only a single time interval is supported. Extensions may support multiple intervals. If multiple intervals are provided, the union of the intervals describes the temporal extent.

type: array

nullable: true

minItems: 1

items:
Begin and end times of the time interval. The timestamps are in the coordinate reference system specified in `trs`. By default this is the Gregorian calendar.

type: array
minItems: 2
maxItems: 2
items:
  type: string
  format: date-time
  nullable: true
example:
  - '2010-02-15T12:34:56Z'
  - null

trs:
  description: |
  Coordinate reference system of the coordinates in the temporal extent (property `interval`). The default reference system is the Gregorian calendar.
  In the Core this is the only supported temporal reference system. Extensions may support additional temporal reference systems and add additional enum values.

type: string
enum:
  - 'http://www.opengis.net/def/uom/ISO-8601/0/Gregorian'
default: 'http://www.opengis.net/def/uom/ISO-8601/0/Gregorian'

landingPage:
  # This object does not include the links element that should be added as an additional element using a personalized version of landingPage-link with allOf
  type: object
  properties:
    title:
      type: string
      example: Buildings in Bonn
    description:
      type: string
      example: Access to data about buildings in the city of Bonn via a Web API that conforms to the OGC API Features specification.

landingPage-link:
  # This element is not directly by other APIs in '/landingPage' because they probably will need to add other links to other resource types in the examples. Instead, it would be copied and enriched with the right examples.
  type: object
  required:
    - links
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/link'
example:
id-link:
  type: object
  description: 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` will respond an array of id-link listing the `resourceId` and the links to get it. `/collections` and `/collections/{collectionId}` 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
  title:
    type: string
  links:
    type: array
    minItems: 1
  items:
    $ref: '#/components/schemas/link'

example:
  id: night
  title: Topographic night style
  links:
    - href: 'https://example.com/api/1.0/styles/night?f=mapbox'
      rel: stylesheet
      type: 'application/vnd.mapbox.style+json'
    - href: 'https://example.com/api/1.0/styles/night?f=sld10'
      rel: stylesheet
      type: 'application/vnd.ogc.sld+xml;version=1.0'

link:
  type: object
  required:
  - href

  properties:
  href:
    type: string
    example: 'http://data.example.com/buildings/123'
  rel:
    type: string
    example: alternate
type:
  type: string
  pattern: '^\(?[-a-z]{1,127}/[-\.a-z0-9]{1,127}$\)[a-z]+(-[a-z]+)*/[a-z0-9]+([-\.][a-z0-9]+)+$'
  example: application/geo+json

defined:
  type: string
  example: en

title:
  type: string
  example: 'Trierer Strasse 70, 53115 Bonn'

length:
  type: integer
  minimum: 0

keyword:
  required: true
  type: object
  nullable: true
  properties:
    keyword:
      type: string
      example: land cover
    code:
      type: string
      example: '4612'
    codeSpace:
      type: string
      example: https://www.eionet.europa.eu/gemet/en/concept/

numberMatched:
  description: The number of elements that match the selection parameters like `bbox`.
  type: integer
  minimum: 0
  example: 127

numberReturned:
  description: The number of elements in the collection.
  type: integer
  minimum: 0
  example: 10

timeStamp:
  description: This property indicates the time and date when the response was generated.
  type: string
The landing page provides links to the API definition (link relations `service-desc` and `service-doc`), the Conformance declaration (path `/conformance`, link relation `conformance`), and the Collections (path `/collections`, link relation `data`).

Access to data about buildings in the city of Bonn via a Web API that conforms to the OGC API specification.

This is just an example that To support "generic" clients that want to access multiple OGC API implementations - and not "just" a specific API / server, the server declares the conformance classes it implements and conforms to.

The collections shared by this API.
This response can be referenced directly for every service that wants only essential information at the collections level. /collections/collectionId might return more information.

The dataset is organized as one or more collections. This resource provides information about and access to the collections.

The response contains the list of collections. For each collection, a link to other resources is present (e.g. the items in the collection; path '/collections/{collectionId}/items', link relation 'items') as well as key information about the collection. This information includes:

* A local identifier for the collection that is unique for the dataset;
* A list of coordinate reference systems (CRS) in which geometries may be returned by the server. The first CRS is the default coordinate reference system (the default is always WGS 84 with axis order longitude/latitude);
* An optional title and description for the collection;
* An optional extent that can be used to provide an indication of the spatial and temporal extent of the collection - typically derived from the data;
* An optional indicator about the type of elements in the collection (the default value, if the indicator is not provided, is 'feature').

content:
application/json:
schema:
$ref: '#/components/schemas/collections'
text/html:
schema:
type: string
Collection:
description: |-  Information about the collection with id 'collectionId'. This is an example for commons only. A service should combine the schemas from common with others specific to its services.

The response might also contain a link to the elements in the collection (e.g. path '/collections/{collectionId}/items', link relation 'items') as well as key information about the collection. This information includes:

* A local identifier for the collection that is unique for the dataset;
* A list of coordinate reference systems (CRS) in which geometries may be returned by the server. The first CRS is the default coordinate reference system (the default is always WGS 84 with axis order longitude/latitude);
* An optional title and description for the collection;
* An optional extent that can be used to provide an indication of the spatial and temporal extent of the collection - typically derived from the data;
* An optional indicator about the type of the items in the collection (the default value, if the indicator is not provided, is 'feature').

content:
application/json:
  schema:
    allOf:
      - $ref: '#/components/schemas/collection'
      - $ref: '#/components/schemas/collection-link'

text/html:
  schema:
    type: string

NoContent:
  # Response associated to 204
  description: |
    No content (useful for OPTIONS)
  headers:
    Allow:
      $ref: '#/components/headers/Allow'

Created:
  # Response associated to 201
  description: |
    Resource created
  headers:
    Location:
      $ref: '#/components/headers/Location'

Updated:
  # Response associated to 204
  description: |
    The resource has been updated or created

Deleted:
  # Response associated to 204
  description: The resource has been deleted

NotModified:
  # Response associated to 304
  description: The resource has not been modified

Invalid:
  # Response associated to 400
  description: The resource is an invalid input

InvalidParam:
  # Response associated to 400
  description: Invalid or unknown query parameters

UnauthorizedAccess:
  # Response associated to 401
  description: Access not unauthorized

NotFoundException:
  # Response associated to 404
  description: The requested URI was not found

UnsupportedFormat:
  # Response associated to 406
  description: The media types accepted by the client are not supported for this resource

AlreadyExist:
  # Response associated to 409
  description: Resource with that id already exists
ServerError:
# Response associated to 500
description: A server error occurred
content:
  application/json:
    schema:
      $ref: '#/components/schemas/exception'
text/html:
    schema:
      type: string
headers:
  Location:
    schema:
      type: string
      format: uri
      description: |-
        URI of the new resource
  Allow:
    schema:
      type: string
      description: |-
        Comma separated list of verbs supported by the resource
Link:
  description: |-
    link header according to RFC 8288
  schema:
    type: string
    example: 'Link: <http://data.example.org/collections/buildings/items.json>; rel="self"; type="application/geo+json"'
examples:
  link-collection-from-collections:
    href: 'http://example.com/collections/buildings?f=json'
    rel: item
    type: 'application/json'
    title: CollectionId path description in the OpenAPI
  link-collection-this:
    href: 'http://example.com/collections/buildings?f=json'
    rel: self
    type: 'application/json'
    title: This document
  link-collection-describedBy:
    href: 'http://example.com/concepts/buildings.html'
    rel: describedBy
    type: text/html
  link-collection-license-html:
    href: 'https://creativecommons.org/publicdomain/zero/1.0/
    rel: license
    type: text/html
    title: CC0-1.0
  link-collection-license-rdf:
    href: 'https://creativecommons.org/publicdomain/zero/1.0/rdf'
B.2. OpenAPI Maps and tiles common

This is a OpenAPI domain file shared by the OGC API - Tiles and the OGC API - Maps developed and used in Testbed-15. This might be used and continued in the WMS SWG in the future.

This corresponds to a URL .../ogc-api-map-tiles/1.0.0

openapi: 3.0.2
info:
  title: OGC API Maps and tiles Building Blocks
Common components used in the OGC API maps and tiles standards.

This document is also available in [GitHub](https://github.com/opengeospatial/OGC-API-Map-Tiles/tree/master/standard/openapi).

version: "1.0.0"
contact:
  name: Joan Maso
  email: joan.maso@uab.cat
license:
  name: OGC License

components:
parameters:
  #############################
  # From OGC API Maps and Tiles
  #############################
  collections:
    name: collections
    in: query
    description: |-
      The collections that should be included in the response. The parameter value is a comma-separated list of collection identifiers. If the parameters is missing, some or all collections will be included.
      required: false
      style: form
      explode: false
      schema:
        type: array
        items:
          type: string
  infoCollections:
    name: infoCollections
    in: query
    description: |-
      The collections that are used in a response of a information request. The parameter value is a comma-separated list of collection identifiers. If the parameters is missing, all collections will be included.
      required: false
      style: form
      explode: false
      schema:
        type: array
        items:
          type: string
  coord_i:
    name: i
in: query
description: |-  
   Horizontal (x) coordinate within a map or tile.
required: true
schema:
   type: number
coord_j:
   name: j
   in: query
description: |-  
   Vertical (y) coordinate within a map or tile.
required: true
schema:
   type: number
infoTemplate:
   name: infoTemplate
   in: query
description: |-  
   Template used for the information response. This parameter can be used to 
   select among GML application schemas or JSON schemas and provide alternative 
   presentations for the same information. If the parameter is missing, the server will 
   select the first template available.
   explode: false
   schema:
      type: string
elevation:
   name: elevation
   in: query
description: |-  
   Elevation value
required: false
style: form
   explode: false
   schema:
      type: number
examples:
link-landingPage-map-tiles:
   href: 'http://data.example.org/map/tiles'
   rel: tiles
   type: application/json
   title: Link to information on map tiles combining more than one collection
link-collection-map-tiles:
   href: 'http://data.example.com/collections/buildings/map/tiles'
   rel: tiles
   type: 'application/json'
link-map-tiles-this:
   href: 'http://data.example.com/collections/buildings/map/tiles'
   rel: self
   type: 'application/json'
link-map-tiles-tile:
   href:
B.3. OpenAPI Tiles

This is the OGC API - Tiles OpenAPI domain file developed and used in Testbed-15. This file will be used by API implementation that are managing tiles. This might be used and continued in the WMS SWG in the future.

This corresponds to a URL .../ogc-api-tiles/1.0.0

```yaml
openapi: 3.0.2
info:
  title: OGC API Building Blocks for maps and tiles
```
Common components used in the OGC API maps and tiles standards.

This document is also available in [GitHub](https://github.com/opengeospatial/OGC-API-Map-Tiles/tree/master/standard/openapi).

```
version: "1.0.0"
contact:
    name: Joan Maso
    email: joan.maso@uab.cat
license:
    name: OGC License

components:
    responses:
        TileMatrixSets:
            description: |
                An array of tile matrix sets (tiling schemes).
            content:
                application/json:
                    schema:
                        $ref: '#/components/schemas/tileMatrixSets'
        tiles:
            description: |
                Description of the tiles.
            content:
                application/json:
                    schema:
                        $ref: '#/components/schemas/tiles'
    parameters:
        tileMatrixSetId:
            name: tileMatrixSetId
            in: path
            description: |
                Identifier of a specific tiling scheme. It can be one of those specified in Annex D.1 of the OGC 17-083r2 standard or one defined in this service.
            required: false
            schema:
                type: string
                example: WebMercatorQuad
        tileMatrix:
            name: tileMatrix
            in: path
            description: |
                Identifier selecting one of the scales defined in the TileMatrixSet and representing the scaleDenominator the tile. For example, Ireland is fully within the Tile at tileMatrix=5, tileRow=10 and tileCol=15.
            required: true
```
tileRow:
  name: tileRow
  in: path
  description: `- Row index of the tile on the selected TileMatrix. It cannot exceed the MatrixWidth-1 for the selected TileMatrix. For example, Ireland is fully within the Tile at tileMatrix=5, tileRow=10 and tileCol=15.
  required: true
schema:
  type: integer
  minimum: 0
  example: '827'

tileCol:
  name: tileCol
  in: path
  description: `Column index of the tile on the selected TileMatrix. It cannot exceed the MatrixHeight-1 for the selected TileMatrix. For example, Ireland is fully within the Tile at tileMatrix=5, tileRow=10 and tileCol=15.
  required: true
schema:
  type: integer
  minimum: 0
  example: 1231

scaleDenominator:
  name: scaleDenominator
  in: query
  description: `A range of scale denominators (that can be used to generate a list of tileMatrix names). Note that scale denominators have several significant digits. To prevent mistakes apply tolerances to intervals. If the client wants to specify a single scale denominator, it will use a small interval with enough tolerance.
  required: false
style: form
explode: false
schema:
  type: array
  minItems: 2
  maxItems: 2
  items:
    type: number
    format: double
multiTileType:
  name: multiTileType
  in: query
  description: `When successful, the service will respond to a query in one of two ways. It can provide a file with links to each tile or it will provide the tiles in a
The package can still contain the description of each tile

The allowed values for this parameter are 'url', 'tiles' and 'full'.
If the value of the 'multiTileType' parameter is set to 'url' causes the
server to return a list of the selected tiles in a format following the 'tileSet'
schema. tile description in the list will contain a URL to download the tile later.
If the value of the 'multiTileType' parameter is set to 'tiles' causes the
server to return a package (e.g. a ZIP file) that will include tiles as separated
parts in the package.
If the value of the 'multiTileType' parameter is set to 'full' causes the
server to return the tiles and a list of the selected tiles in a format following the
'tileSet' schema as part of a package.
The default value is 'tiles'.

```yaml
style: form
schema:
  type: string
  default: tiles
  enum:
    - url
    - tiles
    - full
example: full
```

Examples:
```
link-landingPage-tms-json:
  href: 'http://data.example.org/tileMatrixSets?f=json'
  rel: tiling-schema
  type: application/json
  title: List of tileMatrixSets implemented by this API in JSON

link-landingPage-tms-html:
  href: 'http://data.example.org/tileMatrixSets?f=html'
  rel: tiling-schema
  type: text/html
  title: List of tileMatrixSets implemented by this API in HTML

link-landingPage-tiles:
  href: 'http://data.example.org/tiles'
  rel: tiles
  type: application/json
  title: Link to information on tiles combining more than one collection

link-collection-tiles:
  href: 'http://data.example.com/collections/buildings/tiles'
  rel: tiles
  type: 'application/json'

link-tiles-this:
  href: 'http://data.example.com/collections/buildings/tiles'
  rel: self
  type: 'application/json'

link-tiles-tile:
  href: 'http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}.png'
  rel: item
```
type: 'image/png'

link-tiles-info:
  href: 'http://data.example.com/collections/buildings/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}/info'
  rel: attributes
  type: 'text/html'

link-tiles-col-this:
  href: 'http://data.example.com/tiles'
  rel: self
  type: 'application/json'

link-tiles-col-tile:
  href: 'http://data.example.com/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}.png'
  rel: item
  type: 'image/png'

link-tiles-col-info:
  href: 'http://data.example.com/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}/info'
  rel: attributes
  type: 'text/html'

schemas:
  collection-link:
    # This element is not directly linkable by other APIs because they probably will need to add other links to other resource types. Instead, it would be copied and enriched with more examples.
    type: object
    required:
      - links
    properties:
      links:
        type: array
        items:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'

example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-describedBy'
  - $ref: '#/components/examples/link-collection-tiles'

tiles:
  # This object does not include the links element that should be added as and additional element using tiles-link
  type: object
  required:
    - tileMatrixSetLink
  properties:
    tileMatrixSetLink:
      # A WMTS layer definition has id, title, description, keyword that are already defined in OWS Common
tileMatrixSetLinks:
  $ref: '#/components/schemas/tileMatrixSetLink-set'

infoTemplates:
  type: array
  description: |
    Supported valid templates for the info presentation
  items:
    type: string
    example:
      - table
      - record

tileMatrixSets:
  type: object
  required:
    - tileMatrixSets
  properties:
    tileMatrixSets:
      type: array
      items:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/id-link'
        example:
          - id: MyWebMercatorQuad
            title: My Google Maps Compatible for the World
            links:
              - href: 'https://data.example.org/tileMatrixSet/MyWebMercatorQuad'
              - rel: item
                type: 'application/json'

tileMatrixSet:
  required:
    - identifier
  type: object
  properties:
    identifier:
      type: object
      properties:
        title:
          description: Title of this tile matrix set, normally used for display to a human
          type: string
          example: Google Maps Compatible for the World
          abstract:
            description: Brief narrative description of this tile matrix set, normally available for display to a human
            type: string
            example: The most common TileMatrixSet, used in most of the main IT map browsers. It was initially popularized by Google Maps
          keywords:
            description: Unordered list of one or more commonly used or formalized word(s) or phrase(s) used to describe this dataset
            type: array
            items:
identifier:
  description: Tile matrix set identifier,
  type: string
  example: WebMercatorQuad
supportedCRS:
  description: Reference to one coordinate reference system (CRS)
  type: string
  example: http://www.opengis.net/def/crs/EPSG/0/3857
wellKnownScaleSet:
  description: Reference to a well-known scale set
  type: string
  example: http://www.opengis.net/def/wkss/OGC/1.0/GoogleMapsCompatible
tileMatrix:
  description: Describes a scale level and its tile matrix
  type: array
  items:
    $ref: '#/components/schemas/tileMatrix'tileMatrix:
  type: object
  required:
    - identifier
    - scaleDenominator
    - topLeftCorner
    - tileWidth
    - tileHeight
    - matrixWidth
    - matrixHeight
  properties:
    title:
      description: Title of this tile matrix, normally used for display to a human
      type: string
      example: Google Maps Compatible for the World zoom level 3
    abstract:
      description: Brief narrative description of this tile matrix, normally available for display to a human
      type: string
      example: 'Google Maps Compatible zoom level 3 that is equivalent to a scale of 1:69885283.00358972 and has 19567.87924100512 meters of pixel size in the equator'
keywords:
  description: keywords about the elements in the collection
  type: array
  items:
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/keyword'
identifier:
  description: Identifier selecting one of the scales defined in the collection
  type: string
  example: WebMercatorQuad
TileMatrixSet and representing the scaleDenominator the tile.

- **type**: string
  - **example**: '3'
- **scaleDenominator**:
  - **description**: Scale denominator level of this tile matrix
  - **type**: number
  - **example**: 69885283.00358972
- **topLeftCorner**:
  - **description**: Position in CRS coordinates of the top-left corner of this tile matrix
  - **type**: array
  - **items**:
    - **type**: number
    - **format**: double
    - **example**:
      - -20037508.3427892
      - 20037508.3427892
- **tileWidth**:
  - **description**: Width of each tile of this tile matrix in pixels
  - **type**: number
  - **format**: integer
  - **minimum**: 1
  - **example**: 256
- **tileHeight**:
  - **description**: Height of each tile of this tile matrix in pixels
  - **type**: number
  - **format**: integer
  - **minimum**: 1
  - **example**: 256
- **matrixHeight**:
  - **description**: Width of the matrix (number of tiles in width)
  - **type**: number
  - **format**: integer
  - **minimum**: 1
  - **example**: 8
- **matrixWidth**:
  - **description**: Height of the matrix (number of tiles in height)
  - **type**: number
  - **format**: integer
  - **minimum**: 1
  - **example**: 8
- **mapbox-vector-tile**:
  - **type**: string
  - **format**: binary
- **tileMatrixSetLink-set**:
  - **description**: |
  - **This list of tileMatrixSetLink objects, as defined in OGC 17-083r2, supported by this collectionId.**
  - **type**: array
  - **items**:
    - **$ref**: '#/components/schemas/tileMatrixSetLink-entry'
tileMatrixSetLink-entry:
  type: object
  required:
  - tileMatrixSet
  properties:
    tileMatrixSet:
      type: string
      example: 'WorldMercatorWGS84Quad'
    tileMatrixSetURI:
      type: string
      format: uri
      example: 'http://schemas.opengis.net/tms/1.0/json/examples/WorldMercatorWGS84Quad.json'

tileMatrixSetLimits:
  type: array
  minItems: 1
  items:
    $ref: '#/components/schemas/tileMatrixSetLimits-entry'

tileMatrixSetLimits-entry:
  type: object
  required:
  - tileMatrix
  - minTileRow
  - maxTileRow
  - minTileCol
  - maxTileCol
  properties:
    tileMatrix:
      type: string
      format: uri
      example: '5'
    minTileRow:
      type: number
      format: integer
      minimum: 0
      example: 0
    maxTileRow:
      type: number
      format: integer
      minimum: 0
      example: 1
    minTileCol:
      type: number
      format: integer
      minimum: 0
      example: 3
    maxTileCol:
      type: number
      format: integer
      minimum: 0
      example: 4
tileSet:
  description: |-
    This is the response for a multiple tiles request.
  type: object
  required: tileSet
  properties:
    tileSet:
      type: array
      items:
        $ref: '#/components/schemas/tileSetEntry'

tileSetEntry:
  description: |-
    This is an entry on a multiple tiles request.
  type: object
  required:
    - tileURL
    - tileMatrix
  properties:
    tileURL:
      type: string
      format: uri
      example: 'http://data.example.com/collections/buildings/tiles/WebMercatorQuad/0/0/0.png'
    tileMatrix:
      type: string
      example: 0
    tileRow:
      type: number
      example: 0
    tileCol:
      type: number
      example: 0
    width:
      type: number
      description: |-
        The width of the tile in rendering device pixels. If it exceeds the visual display area be should cut when displayed
      example: 256
    height:
      type: number
      description: |-
        The height of the tile in rendering device pixels. If it exceeds the visual display area be should cut when displayed
      example: 256
    top:
      type: number
      description: |-
        Vertical position from the top of the visual display area in pixels. Negative value means that the left side of the tile is outside the top-left corner of the display and should be cut when displayed
      example: -10
left:
  type: number
description: Horizontal position from the left of the visual display area in pixels.
  Negative value means that the left side of the tile is outside the top-left corner of
  the display and should be cut when displayed
example: -20

B.4. OpenAPI Maps

This is the OGC API - Maps OpenAPI domain file developed and used in Testbed-15. This file will be
used by API implementation that are managing maps. This might be used and continued in the
WMS SWG in the future.

This corresponds to a URL .../ogc-api-maps/1.0.0

```json
openapi: 3.0.2
# Revisions:
# 2019-07-14 Reparated from a more complex domains document
#
info:
title: OGC API - Maps Building Blocks
description: Common components used in the OGC API - Maps standards.

This document is also available in

version: "1.0.0"
contact:
  name: Joan Maso
  email: joan.maso@uab.cat
license:
  name: OGC License
  url: https://raw.githubusercontent.com/opengeospatial/OGC-API-Map-Tiles/master/LICENSE'

components:
schemas:
collection-link:
  #This element is not directly linkable by other APIs because they probably
  will need to add other links to other resource types. Instead, it would be copied and
  enriched with more examples.
type: object
required:
  - links
properties:
  links:
type: array
items:
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-describedBy'
  - $ref: '#/components/examples/link-collection-maps'

maps-link:
  # This element is not directly linkable by other APIs because they probably will need to add other links to other resource types. Instead, it would be copied and enriched with more examples.
  type: object
  required:
  - links
  properties:
    links:
      type: array
      items:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
      example:
        - $ref: '#/components/examples/link-maps-this'
        - $ref: '#/components/examples/link-maps-map'
        - $ref: '#/components/examples/link-maps-info'

collection:
  # This object does not include the links element that should be added as an additional element using collection-link
  type: object
  properties:
    styles:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/schemas/style-set'
    defaultStyle:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/schemas/default-style'
    maps:
      type: object
      properties:
        # A WMS layer definition has id, title, description, keyword that are already defined in OWS Common
        # wgs84BoundingBox is the 'extent' that is already defined in OWS Common
        # BoundingBox is the crsSpatialExtents that is already defined in OWS Common
        # For the moment we will assume that maps can only be produced in the crs's advertised by the collection in the common part.
        minScaleDenominator:
          type: number
          description: Minimum scale denominator (inclusive) for which it is appropriate to
generate a map of this collection. Requests outside this interval will return an HTTP
404. If it is not present we will assume there is no limit.

```yaml
example: 10
maxScaleDenominator:
  type: number
  description: |
    Maximum scale denominator (inclusive) for which it is appropriate to
generate a map of this collection. Requests outside this interval will return an HTTP
404 If it is not present we will assume there is no limit.

example: 10000000
```

#spatialResolution (resx) this should be part of the common metadata but
currently is not.

```yaml
recomendedFormat:
  type: string
  description: |
    Recommended output formats for a map request. Depending of the nature
of the data, a format might be better than another. Categorical data looks better in a
PNG but continuos data and pictures are smaller a JPEG. The map operation details all
available formats for the OGC API maps. In contrast, this is the better one for this
type of information. It would be one of the supported for the map operation

example: 'image/jpeg'
```

#queryable is not included here because it is indicated by a link with
'rel': 'info' but I'm not sure it is the right decision.

```yaml
cascaded:
  type: number
  description: |
    Indicates how main times the collection map has been retransmitted from
another map service or API (cascading). It it is not present the collection is not
cascaded.

example: 0
default: 0
```

opaque:
  type: boolean
  description: |
    Indicates whether the map data represents features that probably do not
completely fill space shows the background opaque (true) or transparent(false).

example: false
default: false

noSubsets:
  # this is a very old paramter in WMS and I recommend to deprecate it
type: boolean
  description: |
    Indicates whether the server can produce a map that is a subset of the
full bounding box.
example: false
```

fixedWidth:
  # this is a very old paramter in WMS and I recommend to deprecate it
type: number
  description: |
    Indicates that the server can only produce map of a fixed width instead
of an arbitrary width.
fixedHeight:
  # this is a very old parameter in WMS and I recommend to deprecate it
  type: number
  description: |
      Indicates that the server can only produce map of a fixed height instead of an arbitrary height

collection-patch:
  type: object
  properties:
    styles:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/schemas/style-set'
    defaultStyle:
      type: string
      nullable: true
      description: |
          the style id of a recommended default style to use for this collection
      example: 'topographic'

responses:
  tiles:
    description: |
        Description of the tiles.
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/maps'

parameters:
  mapId:
    name: mapId
    in: path
    description: Local identifier of a specific map created with a post operation
    required: true
    style: simple
    explode: false
    schema:
      type: string
  crsId:
    name: crsId
    in: path
    description: |
        Local identifier of a specific CRS. A list of all available CRSIds can be found under the /CRS path. The default CRS is WGS 84.
    required: true
    schema:
      type: string
    example: WGS84
  width:
    name: width
    in: query
    description: |
        Width in pixels of map picture.
required: false
style: form
explode: false
schema:
  type: number
  default: 256

height:
  name: height
  in: query
  description: |-
    Height in pixels of map picture.
  required: false
  style: form
  explode: false
  schema:
    type: number
    default: 256

transparent:
  name: transparent
  in: query
  description: |-
    Background transparency of map (default=true).
  required: false
  style: form
  explode: false
  schema:
    type: boolean
    default: true

bgcolor:
  name: bgcolor
  in: query
  description: |-
    Hexadecimal red-green-blue[-alpha] color value for the background color
    (default=0xFFFFFF). If alpha is not specified "opaque" opacity is assumed.
  required: false
  style: form
  explode: false
  schema:
    type: string
    default: 0xFFFFFF

map-crs:
  name: crs
  in: query
  description: |-
    CRSId used to render the map. It is also the CRS of the bbox parameter.
    You can only ask for CRSs that are valid for the collectionId.
  required: false
  style: form
  explode: false
  schema:
    type: string
example: 'http://www.opengis.net/def/crs/OGC/1.3/CRS84'  
default: 'http://www.opengis.net/def/crs/EPSG/0/3857'
examples:
  link-landingPage-maps:
    href: 'http://data.example.org/maps'
    rel: map
    type: application/json
    title: Link to information on maps combining more than one collection
  link-collection-maps:
    href: 'http://data.example.com/collections/buildings/maps'
    rel: map
    type: 'application/json'
  link-maps-this:
    href: 'http://data.example.com/collections/buildings/maps'
    rel: self
    type: 'application/json'
  link-maps-map:
    href: 'http://data.example.com/collections/buildings/maps/brown'
    rel: item
    type: 'image/png'
  link-maps-info:
    href: 'http://data.example.com/collections/buildings/maps/brown/info'
    rel: attributes
    type: 'text/html'
  link-maps-col-this:
    href: 'http://data.example.com/maps'
    rel: self
    type: 'application/json'
  link-maps-col-map:
    href: 'http://data.example.com/map'
    rel: map
    type: 'image/png'
  link-maps-col-info:
    href: 'http://data.example.com/map/info'
    rel: attributes
    type: 'text/html'
Appendix C: OpenAPI Examples

C.1. OpenAPI Example for vector tiles

In this example we present an imaginary API server that provides access to tiled vector data using the OGC API - Tiles draft specification.

```json
openapi: 3.0.0
# Revisions:
# 2019-06-14 Created from a more complex example
servers:
# Added by API Auto Mocking Plugin
  - description: SwaggerHub API Auto Mocking
    url: https://virtserver.swaggerhub.com/UAB-CREAF/ogc-api-tiles-opf-xmp-vt-more-1-collection/1.0.0
  - description: Server
    url: http://data.example.org
info:
title: |
  Tiled feature data service example. Part of the OGC API Maps and Tiles OpenAPI work.
description: |
  This is a draft of an example of a service following the OGC API maps and tiles draft specificaiton that produces tiled feature data.

This draft specification was developed in the Testbed-15 Open Portrayal Framework in collaboration with the OGC WMS SWG. The Map Tile API is a Web API for fetching and managing maps and tiles.

This example shows how to request tiled feature data (sometimes refered as tiled vector data or vector tiles) from one or more than one collections

For more background information see [the Vector Tiles Pilot Extension Engineering Report](https://github.com/opengeospatial/OGC-API-Map-Tiles/tree/master/standard/openapi).

version: "1.0.0"
contact:
  name: Joan Maso
  email: joan.maso@uab.cat
license:
  name: OGC License
tags:
  - name: OGC API Common
description: |
  Common characteristics of this API
```
name: Tiles metadata
description: |
Metadata about tiles and tileMatrixSets
- name: Tiled features from one collection
description: |
Data partitioned into a hierarchy of tiles of a collection
- name: Tiled data from more than one collection
description: |
Data representations, partitioned into a hierarchy of tiles of more that one collection.

paths:
'/':
get:
tags:
- OGC API Common
summary: landing page
description: |
The landing page provides links to the API definition, the conformance statements and to the feature collections in this dataset.
operationId: getLandingPage
parameters:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
responses:
'200':
description: |
Links to the API capabilities and the TileMatrixSets shared by this API.
content:
  application/json:
    schema:
      $ref: '#/components/schemas/landingPage'
text/html:
    schema:
      type: string
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/conformance':
get:
description: |
A list of all requirements classes specified in a standard that the server conforms to.
operationId: getConformanceDeclaration
parameters:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
tags:
- OGC API Common
responses:
'200':
description: |-  
the URIs of all requirements classes supported by this API

content:
application/json:
schema:
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/confClasses'
example:
  conformsTo:
    # OGC API Common core consists on the landingPage, conformance
    - 'http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core'
    # OGC API Common collections consists adds the capability to have
    - 'http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections'
    # We need to be sure which ones are still valid when adopting
    - 'http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections'
    # OGC API Tiles core consists on the capability to serve a tiles
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/core'
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/tms'
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections'
    # OGC API Tiles info consists on the capability to serve
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/info'
    # OGC API Tiles core consists on the capability to serve a tile
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections'
    # OGC API Tiles core consists on the capability to serve a info on
    - 'http://www.opengis.net/spec/ogcapi-tiles-1/1.0/req/collections-
    - info'
'400':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/InvalidParam'
'406':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/UnsupportedFormat'
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/ServerError'
'/collections':
get:
tag:
  - OGC API Common
summary: the collections in the dataset
operationId: getCollections
parameters:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
  responses:
  '200':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/Collections'
  '500':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/collections/{collectionId}':
  get:
    tags:
    - OGC API Common
    summary: describe a collection
    operationId: describeCollection
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
    responses:
      '200':
        description: |
          Metadata about the collection including style information.
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/collection'
          text/html:
            schema:
              type: string
      '404':
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
      '500':
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/tileMatrixSets':
  get:
    tags:
    - Tiles metadata
    summary: fetch all available tile matrix sets (tiling schemes)
    operationId: getTileMatrixSets
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
    responses:
      '200':
        description: |
          An array of tile matrix sets (tiling schemes).
        content:
application/json:
  schema:
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tileMatrixSets'
    '500':
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/tileMatrixSets/{tileMatrixSetId}':
  get:
    tags:
      - Tiles metadata
    summary: fetch a tile matrix sets (tiling scheme) by id
    operationId: getTileMatrixSetDescription
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
    responses:
      '200':
        description: tile matrix sets (a tiling scheme).
        content:
          application/json:
            schema:
              $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tileMatrixSet'
        '404':
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
        '500':
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/collections/{collectionId}/tiles':
  get:
    tags:
      - Tiles metadata
    summary: fetch a tiles description
    description: Retrieves the tiles description for this collection including the 'links' to get a 'tile', the 'TileMatrixSetLink' and the 'infoTemplate'
    operationId: describeTiles
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
    responses:
      '200':
        description: |
          Description of the tiles.
        content:
          application/json:
            schema:
$ref: '#/components/schemas/tiles'

'404':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'

'500':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/tiles':
get:
tags:
  - Tiles metadata
summary: fetch a tiles description
description: |
  Retrieves the tiles description for this collection including the 'links' to get a 'tile'
operationId: describeTilesCollections
responses:
  '200':
description: |
  Description of the tiles.
content:
  application/json:
schema:
    type: object
    required:
    - links
    properties:
      links:
        type: array
        items:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-tiles-col-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-tiles-col-tile'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-tiles-col-info'

'404':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'

'500':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/collections/{collectionId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}':
get:
tags:
  - Tiled features from one collection
**Summary:** fetch a tile from a collection

**Description:**
Retrieves the tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has a single collection (formerly referred as layer) with all selected features in the bounding box of the tile. The feature properties to include in the tile representation can be limited using a query parameter.

**Operation ID:** getTileOfCollectionId

**Parameters:**
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: '#/components/parameters/f-png-jpeg-vector'

**Responses:**
- '200':
  $ref: '#/components/responses/tile'
- '404':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
- '500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/collections/{collectionId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}'

**Tags:**
- Tiled features from one collection

**Summary:** fetch information about a point on a tile from a collection

**Description:**
Retrieves information on a point of a tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has a single collection (formerly referred as layer) with all selected features in the bounding box of the tile. The feature properties to include in the tile representation can be limited using a query parameter.

**Operation ID:** getFeatureInfoTileOfCollectionId

**Parameters:**
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: '#/components/parameters/f-png-jpeg-vector'
get:
tags:
- Tiled data from more than one collection
summary: fetch a tile from one or more collections
description: Retrieves a tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has multiple collections (formerly referred as layers) with all selected features in the bounding box of the tile.
operationId: getTileCollections
parameters:  
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/collections'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_i'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_j'
- $ref: '#/components/parameters/infoTemplate'
- $ref: '#/components/parameters/f-tile'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
responses:
  '200':
    $ref: '#/components/responses/info'
  '404':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
  '500':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}':
get:
- Tiled data from more than one collection

collection:
- Tiled data from more than one collection


description: Retrieves information about a point of a tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has a single collection (formerly referred as layer) with all selected features in the bounding box of the tile. The feature properties to include in the tile representation can be limited using a query parameter.

operationId: getFeatureInfoTileOfCollections

parameters:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/collections'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_i'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_j'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/infoTemplate'
- $ref: '#/components/parameters/f-tile'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/elevation'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'

responses:
components:
schemas:
  landingPage:
    allOf:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/landingPage'
      - $ref: '#/components/schemas/landingPage-link'
    
  landingPage-link:
    #This element is a duplicate of the one in OGC API common but it is enriched with more examples for maps and other resource types.
    type: object
    required:
      - links
    properties:
      links:
        type: array
        items:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
        example:
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-this'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-alternate'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-service-json'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-service-html'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-conformance'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-collections-json'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-collections-html'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-landingPage-tms-json'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-landingPage-tms-html'
          - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-landingPage-tiles'
collection-link:
  #This element is a duplicate of the one in OGC API common but it is enriched with more examples for maps and other resource types.
  type: object
  required:
    - links
  properties:
    links:
      type: array
      items:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-describedBy'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-license-html'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-license-rdf'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/examples/link-collection-items'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-collection-tiles'
tiles:
  allOf:
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tiles'
    - $ref: '#/components/schemas/tiles-link'
tiles-link:
  #This element is a duplicate of the one in OGC API tiles but it is enriched with more examples for tiles.
  type: object
  required:
    - links
  properties:
    links:
      type: array
      items:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-tiles-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-tiles-tile'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-tiles-info'
parameters:
  f-png-jpeg-vector:
    name: f
    in: query
    description: |
      The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.

      Pre-defined values are "mvt" for a Mapbox Vector Tile, "json" for a GeoJSON tiled feature collection and jpeg, png or gif for image based tiles
      The response to other values is determined by the server.
    required: false
    style: form
    explode: false
    schema:
      type: string
      enum:
        - image/png
        - image/jpeg
        - image/gif
        - mvt
        - application/json
    example: image/png

f-tile:
  name: f
  in: query
  description: |
    The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.

    Pre-defined values are "mvt" for a Mapbox Vector Tile, "json" for a GeoJSON tiled feature collection and jpeg, png or gif for image based tiles
    The response to other values is determined by the server.
  required: false
  style: form
  explode: false
  schema:
    type: string
    enum:
      - image/png
      - image/jpeg
      - image/gif
      - mvt
      - application/json
    example: image/png

responses:
tile:
  description: A tile of the collection.
  content:
    image/jpeg:
      schema:
C.2. OpenAPI Example for map tiles

In this example we present an imaginary API server that provides access to map tiles using the OGC API - Maps and OGC API - Tiles draft specifications.

```yaml
openapi: 3.0.0
servers:
  # Added by API Auto Mocking Plugin
  - description: SwaggerHub API Auto Mocking
    url: https://virtserver.swaggerhub.com/UAB-CREAF/ogc-api-map-tiles-opf-xmp-mt-more-1-collection/1.0.0
  - description: Server example
    url: http://data.example.org
info:
  title: |
    Tiled map data service example. Part of the OGC API Maps and Tiles OpenAPI work.
  description: |
    This is a draft of an example of a service following the OGC API maps and tiles that produces tiled map data.

It is elaborated in the Testbed-15 Open Portrayal Framework in collaboration with the WMS.SWG. The Map Tile API is a Web API for fetching and managing maps and tiles.
```
It illustrated how to request tiled maps from one or more than one collections.

version: "1.0.0"
contact:
  name: Joan Maso
  email: joan.maso@uab.cat
license:
  name: OGC License
tags:
- name: OGC API Common
description: |
  Common characteristics of this API
- name: Map tiles metadata
description: |
  Metadata about tiles and tileMatrixSets
- name: Tiled maps from one collection
description: |
  map (renderizations or vector tiles) partitioned into a hierarchy of tiles of a collection
- name: Tiled maps from more that one collection
description: |
  access to maps, partitioned into a hierarchy of tiles of more that one collection
paths:
  '/':
    get:
      tags:
        - OGC API Common
      summary: landing page
      description: |
        The landing page provides links to the API definition, the conformance statements and to the feature collections in this dataset.
      operationId: getLandingPage
      parameters:
        - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
      responses:
        '200':
          description: |
            Links to the API capabilities and the TileMatrixSets shared by this API.
          content:
            application/json:
              schema:
                $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/landingPage'
              example:
                title: Map tiles data service example.
                description: Map tiles data service example. Part of the OGC API Maps and Tiles OpenAPI work
links:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-this'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-alternate'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-service-json'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-service-html'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-conformance'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-collections-json'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-landingPage-collections-html'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/examples/link-landingPage-tms-json'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/examples/link-landingPage-map-tiles'


text/html:
schema:
type: string
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/conformance':
  get:
    description: "A list of all requirements classes specified in a standard that the server conforms to.
operationId: getConformanceDeclaration
parameters:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
tag:
  - OGC API Common
responses:
  '200':
    description: "the URIs of all requirements classes supported by this API
content:
  application/json:
    schema:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/confClasses'
example:
  conformsTo:
  # OGC API Common core consists on the landingPage, conformance
  - 'http://www.opengis.net/spec/ogcapi-common-1/1.0/req/core'
  # OGC API Common collections consists adds the capability to have
collections
# We need to be sure which ones are still valid when adopting OpenAPI.

```yaml
'/collections':
  get:
    tags:
      - OGC API Common
    summary: the collections in the dataset
    operationId: getCollections
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
    responses:
      '200':
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/Collections'
      '500':
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/collections/{collectionId}':
  get:
    tags:
      - OGC API Common
    summary: describe a collection
    operationId: describeCollection
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json-html'
```

responses: 

'200':

description: |-
   Metadata about the collection including style information.
content:
   application/json:
      schema:
         $ref: '#/components/schemas/collection'
   text/html:
      schema:
         type: string

'404':
   $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'

'500':
   $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/tileMatrixSets':
   get:
      tags:
         - Map tiles metadata
      summary: fetch all available tile matrix sets (tiling schemes)
      operationId: getTileMatrixSets
      parameters:
         - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
      responses:
         '200':
            description: |-
               An array of tile matrix sets (tiling schemes).
            content:
               application/json:
                  schema:
                     $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tileMatrixSets'
         '500':
            $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

'/tileMatrixSets/{tileMatrixSetId}':
   get:
      tags:
         - Map tiles metadata
      summary: fetch a tile matrix sets (tiling scheme) by id
      operationId: getTileMatrixSetDescription
      parameters:
         - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
         - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/f-json'
      responses:
         '200':
            description: |-
               An array of tile matrix sets (tiling schemes).
description: tile matrix sets (a tiling scheme).
content:
  application/json:
    schema:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tileMatrixSet'
'404':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/collections/{collectionId}/map/tiles' get:
  tags:
    - Map tiles metadata
  summary: fetch a tiles description
  description: |
    Retrieves the tiles description for this collection including the 'links' to get a 'tile', the 'TileMatrixSetLink' and the 'infoTemplate'
  operationId: describeMapTiles
  parameters:
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
  responses:
    '200':
      description: |
        Description of the map tiles.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/map-tiles'
'404':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
'/map/tiles' get:
  tags:
    - Map tiles metadata
  summary: fetch a map tiles description
  description: |
    Retrieves the map tiles description for this collection including the
    'links' to get a 'maps'
  operationId: describeMapTilesCollections
  responses:
    '200':
      description: |
        Links to the map tiles.
get:
  tags:
    - Tiled maps from one collection
  summary: fetch a tile from a collection
  description: Retrieves the tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has a single collection (formerly referred as layer) with all selected features in the bounding box of the tile. The feature properties to include in the tile representation can be limited using a query parameter.
  operationId: getMapTileOfCollectionId
  parameters:
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/collectionId'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/parameters/styleId'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/parameters/transparent'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
    - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
/collections/{collectionId}/map/{styleId}/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}/info:
  get:
    tags:
      - Tiled maps from one collection
    summary: fetch information about a point on a tile from a collection
    description: |
      Retrieves information on a point of a tile in the requested tileMatrixSet, on the requested tileMatrix in the TileMatrixSet, with the requested tile indices (tileRow, tileCol). The tile has a single collection (formerly referred as layer) with all selected features in the bounding box of the tile. The feature properties to include in the tile representation can be limited using a query parameter.
    operationId: getFeatureInfoMapTileOfCollectionId
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'

responses:
  '200':
    description: A tile of the collection.
    content:
      image/jpeg:
        schema:
          type: string
          format: binary
      image/png:
        schema:
          type: string
          format: binary
      image/gif:
        schema:
          type: string
          format: binary
      image/mvt:
        schema:
          type: string
          format: binary
      application/geo+json:
        schema:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'
  '404':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
  '500':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}:
get:
  tags:
  - Tiled maps from more that one collection
  summary: retrieves a map composed by one or more collections
  description: Retrieves a map in the requested crs, on the requested bbox designed to be shown in a device of a width and a height. Some formats require to apply a style in the server side (e.g. png, jpeg, gif) and some others might include a reference to a style to be applied in the client side. The feature properties to include in the tile representation can be limited using a query parameter.
  operationId: GetMapTileCollections
  parameters:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/collections'
  responses:
    '200':
      description: A getFeatureInfo of a tile of the collection.
      content:
        application/geo+json:
          schema:
            $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'
    '404':
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
    '500':
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/parameters/styles'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/parameters/transparent'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/parameters/bgcolor'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/elevation'
- $ref: '#/components/parameters/f-png-jpeg'

responses:
  '200':
    description: |-
      A multi-collection tile of the dataset.
    content:
      image/jpeg:
        schema:
          type: string
          format: binary
      image/png:
        schema:
          type: string
          format: binary
      image/gif:
        schema:
          type: string
          format: binary
      image/mvt:
        schema:
          type: string
          format: binary
      application/geo+json:
        schema:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'
  '404':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
  '500':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'

post:

tags:
Tiled maps from more than one collection

summary: fetch a map from one or more collections

description: |

Retrieves a map in the requested crs, on the requested bbox designed to be shown in a device of a width and a height. Some formats require to apply a style in the server side (e.g. png, jpeg, gif) and some others might include a reference to a style to be applied in the client side. The feature properties to include in the tile representation can be limited using a query parameter.

operationId: GetMapTileCollectionsBody

parameters:
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileCol'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/elevation'

responses:
  '200':
    description: |
    Retrieves a map that renders objects of the collectionId in the requested crs, on the requested bbox designed to be shown in a rendering device of a width and a height.

content:
  image/jpeg:
    schema:
      type: string
      format: binary
  image/png:
    schema:
      type: string
      format: binary
  image/gif:
    schema:
      type: string
      format: binary
  application/geo+json:
    schema:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'

  '400':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/Invalid'

  '404':
    $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
500:
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
requestBody:
  description: |
  - descriptions of the map an style in a SDL-SE encoding
  content:
    application/sdl-se+json:
      schema:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/schemas/simbologyEncoding'
'/map/tiles/{tileMatrixSetId}/{tileMatrix}/{tileRow}/{tileCol}/info':
  get:
    tags:
      - Tiled maps from more that one collection
    summary: retrieves a map composed by one or more collections
    description: |
      Retrieves a map in the requested crs, on the requested bbox designend to be
      shown in a device of a width and a height. Some formats require to apply a style in
      the server side (e.g. png, jpeg, gif) and some others might include a reference to a
      style to be applied in the client side. The feature properties to include in the tile
      representation can be limited using a query parameter.
    operationId: GetFeatureInfoTileCollections
    parameters:
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/collections'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-styles/1.0.0#/components/parameters/styles'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/parameters/transparent'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-maps/1.0.0#/components/parameters/bgcolor'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/tileMatrixSetId'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/tileMatrix'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/tileRow'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/tileCol'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/parameters/datetime'
      - $ref: '#/components/parameters/f-tile'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_i'
      - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/parameters/coord_j'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/infoTemplate'

```json

tiles/1.0.0#/components/parameters/infoTemplate'
- $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/f-json-html'

responses:
  '200':
    description: |
      A multi-collection tile of the dataset.
    content:
      image/jpeg:
        schema:
          type: string
          format: binary
      image/png:
        schema:
          type: string
          format: binary
      image/gif:
        schema:
          type: string
          format: binary
      image/mvt:
        schema:
          type: string
          format: binary
      application/geo+json:
        schema:
          $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/schemas/featureCollectionGeoJSON'
'404':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/NotFound'
'500':
  $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/responses/ServerError'
post:
  tags:
  - Tiled maps from more that one collection
  summary: fetch a map from one or more collections
  description: |
    Retrieves a map in the requested crs, on the requested bbox designed to be shown in a device of a width and a height. Some formats require to apply a style in the server side (e.g. png, jpeg, gif) and some others might include a reference to a style to be applied in the client side. The feature properties to include in the tile representation can be limited using a query parameter.
  operationId: GetFeatureInfoTileCollectionsBody
  parameters:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrixSetId'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileMatrix'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/parameters/tileRow'
```
responses:
'200':

description: |-
Retrieves a map that renders objects of the collectionId in the
requested crs, on the requested bbox desigend to be shown in a rendering device of a
width and a height.

content:
  image/jpeg:
    schema:
      type: string
      format: binary
  image/png:
    schema:
      type: string
      format: binary
  image/gif:
    schema:
      type: string
      format: binary
  application/geo+json:
    schema:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
features/1.0.0#/components/schemas/featureCollectionGeoJSON'

'400':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/Invalid'

'404':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/NotFound'

'500':
$ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
common/1.0.0#/components/responses/ServerError'

requestBody:
  description: |-
  descriptions of the map an style in a SDL-SE encoding

  content:
    application/se+json:
      schema:
        $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-
maps/1.0.0#/components/schemas/simbologyEncoding'

components:
  schemas:
    collection:
      allOf:
collection-link:

#This element is a duplicate of the one in OGC API common but it is enriched with more examples for maps and other resource types.

type: object
required:
- links
properties:
  links:
    type: array
    items:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-describedBy'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-license-html'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/examples/link-collection-license-rdf'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-features/1.0.0#/components/examples/link-collection-items'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/examples/link-collection-map-tiles'

map-tiles:
  allOf:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-tiles/1.0.0#/components/schemas/tiles'
  - $ref: '#/components/schemas/map-tiles-link'

map-tiles-link:

#This element is a duplicate of the one in OGC API common but it is enriched with more examples for maps and other resource types.

type: object
required:
- links
properties:
  links:
    type: array
    items:
      $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-common/1.0.0#/components/schemas/link'
example:
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/examples/link-map-tiles-this'
  - $ref: 'https://api.swaggerhub.com/domains/UAB-CREAF/ogc-api-map-tiles/1.0.0#/components/examples/link-map-tiles-tile'
tiles/1.0.0#/components/examples/link-map-tiles-info'

parameters:
  f-json-zip:
    name: f
    in: query
    description: |-  The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.
    required: false
    style: form
    explode: false
    schema:
      type: string
      enum:
        - image/json
        - application/zip
      example: image/json

  f-png-jpeg:
    name: f
    in: query
    description: |-  The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.
    required: false
    style: form
    explode: false
    schema:
      type: string
      enum:
        - image/png
        - image/jpeg
        - image/gif
      example: image/png

  f-tile:
    name: f
    in: query
    description: |-  The format of the response. If no value is provided, the standard http rules apply, i.e., the accept header is used to determine the format.
    required: false
    style: form
    explode: false

Pre-defined values are jpeg, png or gif for image based tiles  
The response to other values is determined by the server.
required: false
style: form
explode: false
schema:
  type: string
  enum:
    - image/png
    - image/jpeg
    - image/gif
  example: image/png
responses: { }
### Appendix D: Revision History

**Table 7. Revision History**

<table>
<thead>
<tr>
<th>Date</th>
<th>Editor</th>
<th>Release</th>
<th>Primary clauses modified</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 28, 2019</td>
<td>J. Masó</td>
<td>-</td>
<td>all</td>
<td>initial version as work for the OGC API Hackathon in London</td>
</tr>
<tr>
<td>September 9, 2019</td>
<td>J. Masó</td>
<td>-</td>
<td>all</td>
<td>Last version in the old document name with D014 and D16 together</td>
</tr>
<tr>
<td>October 1, 2019</td>
<td>J. Masó</td>
<td>0.1</td>
<td>all</td>
<td>first version with the correct ToC as D014</td>
</tr>
<tr>
<td>October 14, 2019</td>
<td>J. Masó</td>
<td>0.8</td>
<td>all</td>
<td>ready for OGC review</td>
</tr>
<tr>
<td>October 21, 2019</td>
<td>J. Masó</td>
<td>0.9</td>
<td>all</td>
<td>results of the OGC review</td>
</tr>
<tr>
<td>October 25, 2019</td>
<td>J. Masó</td>
<td>1.0</td>
<td>all</td>
<td>document ready for the 3 week rule in the Toulouse TC.</td>
</tr>
<tr>
<td>December 9, 2019</td>
<td>C. Reed</td>
<td>1.0</td>
<td>Various</td>
<td>Minor edits to Abstract, Exec Summary, and various other clauses prior to publication as a Public ER</td>
</tr>
<tr>
<td>January 2, 2020</td>
<td>G. Hobona</td>
<td>1.0</td>
<td>Various</td>
<td>Final staff review and edits of Testbed-15 engineering reports.</td>
</tr>
</tbody>
</table>
Appendix E: Bibliography
