Kanton Zürich
Baudirektion
Amt für Raumentwicklung
Geoinformation

Datenbereitstellung unter Open Government Data

REST API v1.0 – Interface documentation

Amt für Raumentwicklung
Geoinformation
Kontakt: Michael Boller, Leiter GIS-Koordination, Stampfenbachstrasse 12, 8090 Zürich
Telefon +41 43 259 26 95, www.giszh.zh.ch
5. Februar 2018

Autor	David Reksten (INSER) / DR
Klassifizierung	Nicht klassifiziert
Status	Definitive Fassung / Freigabe

Änderungsverzeichnis

Date	Version	Description	Author
2017-07-20	0.1	Initial version	D.Reksten
2017-08-03	0.2	Implemented feedback from M.Baumann	D.Reksten
2017-09-21	1.0	Corrections, for publication	D.Reksten
2017-10-26	1.0.1	Updated URL to https	M.Baumann
2017-11-10	1.1	Implemented feedback from client	D.Reksten
2017-12-12	1.1.1	Updated return status codes, product JSON	D.Reksten
2017-12-14	1.1.2	Minor corrections, p.10	D.Reksten
2018-02-05	1.2.0	New minor release	D.Reksten

Contents

1. Description

This document explains the REST API made available by the OGD ZH data extraction (“OGAPI”), version 1.0

2. What’s new

Version	Description
1.1.x → 1.2.0	Response JSON after order submission now contains the URL to query order status as well as the URL to download the results, when available. See page 11 for more information.

3. Authorization

This version of the API is open to the general public without any need for authorization or identification.

4. General

The API returns JSON objects, and it is recommended setting the HTTP "Accept" header as follows on each API call, even though this isn’t strictly enforced in this version of the API:

Header key	Header value
Accept	application/json; charset=UTF-8

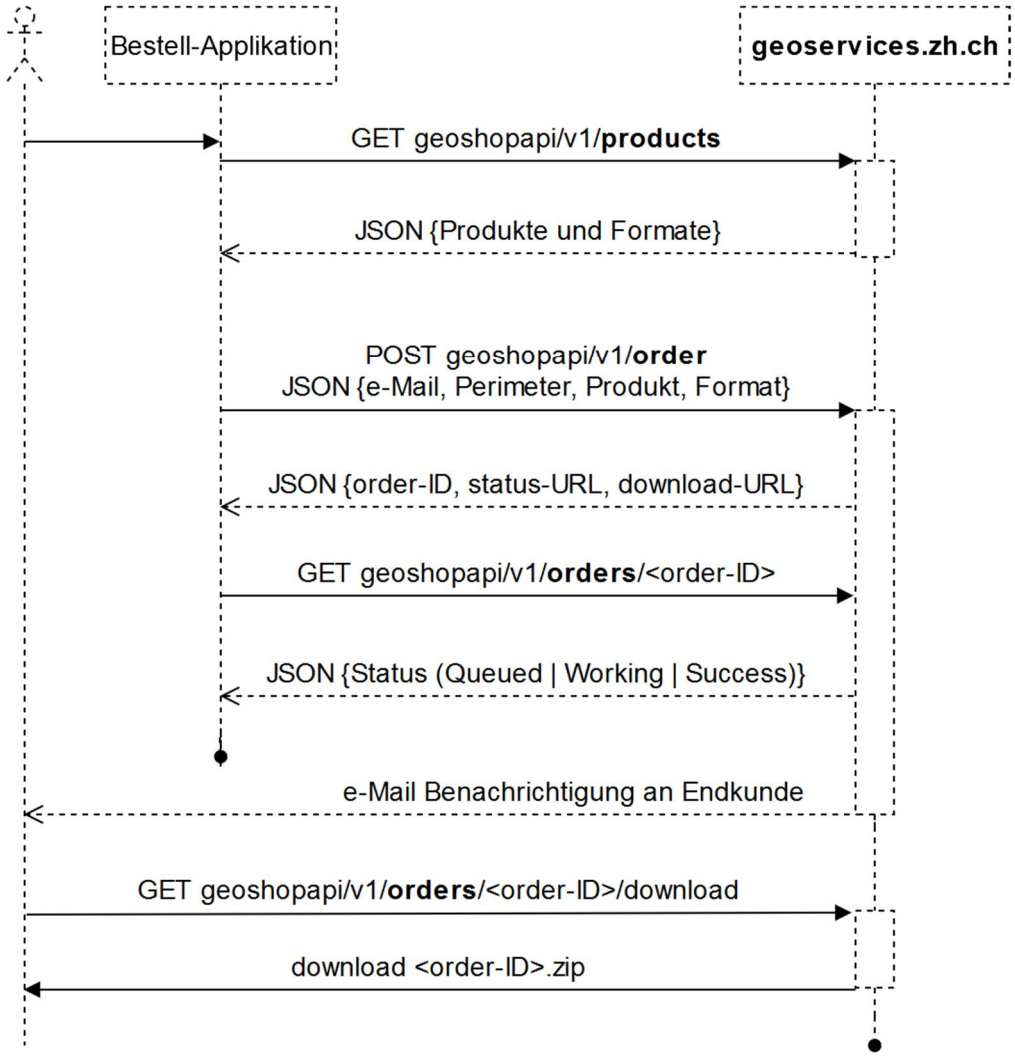
For all API operations that take a request entity (includes all operations that expect a JSON request body, e.g. POST requests), the content-type and character set should also be set as follows:

Header key	Header value
Content-Type	application/json; charset=UTF-8

Unless otherwise specified, all timestamps are to be assumed as according to ISO 8601, local time for Zürich.

5. Flow diagram

The following diagram demonstrates data flow for a complete use case of the REST API.



6. Resources

4.1 Products

The “products” resource supports the GET verb, which is used to retrieve a list of all the available products and their associated formats available for delivery. It also returns a list of all the currently valid communes, for use as an indirect order perimeter. The product list is returned as a timestamped JSON object, see appendix 2 for details.

GET

URL	https://geoservices.zh.ch/geoshopapi/v1/products
Request method	GET
Response body	JSON object, cf. appendix 2.

4.2 Orders

The “orders” resource currently supports the POST and GET verb to submit an order and to retrieve the order status respectively.

POST
Submits an order for processing. A successful request returns a JSON object containing technical order ID. This identifier can later be used to query the order status using the “GET method, see next chapter.

URL	https://geoservices.zh.ch/geoshopapi/v1/orders
Request method	POST

Request body example for a user defined selection polygon (DIRECT perimeter type)	{
	"email": "user@example.com",
	"perimeter_type": "DIRECT",
	"pdir_polygon": {
	"type": "Polygon",
	"coordinates": [
	[
	[2675300, 1251900],
	[2678100, 1251900],
	[2678100, 1253500],
	[2675300, 1253500],
	[2675300, 1251900]
	]
	]
	},

	<pre>"pdir_coordsys": "LV95",
	"products": [
	{"product_id": 121, "format_id": 128},
	{"product_id": 2, "format_id": 101},
	{"product_id": 343, "format_id": 61},
	{"product_id": 103, "format_id": 4}
	]
	}</pre>

Request body example for referenced perimeter selection (INDIRECT perimeter type) using the communes 0161 (Zollikon) and 0178 (Russikon).	<pre>{
	"email": "user@example.com",
	"perimeter_type": "INDIRECT",
	"pindir_layer_name": "COMMUNE",
	"pindir_ident": ["0161","0178"],
	"products": [
	{"product_id": 121, "format_id": 128},
	{"product_id": 2, "format_id": 101},
	{"product_id": 343, "format_id": 61},
	{"product_id": 103, "format_id": 4}
	]
	}</pre>

Example response	<pre>{
	"download_url":
	"http://geoservices.zh.ch:80/geoshopapi/v1/orders/e57ed/download",
	"order_id": "e57ed",
	"status_url":
	"http://geoservices.zh.ch:80/geoshopapi/v1/orders/e57ed",
	"timestamp": "2018-02-05T13:15:34"
	}</pre>

The supported parameters are:

Parameter	Description
email	Email address for the delivery notification when the data extraction has terminated. Example “user@example.com”
perimeter_type	Indicates the perimeter selection type for the order, possible values are “DIRECT” or “INDIRECT”: A DIRECT perimeter indicates that the selection polygon was drawn by the customer and that the polygon is transferred using the parameters pdir_polygon and pdir_coordsys. An INDIRECT perimeter references an existing polygon in a dataset managed by Kanton Zürich. The dataset type and the identifier of this

	polygon is transferred using the parameters pindir_layer_name and pindir_ident.
pindir_layer_name	Only used when perimeter_type = INDIRECT, this parameter specifies the dataset containing the selection polygon. Possible values are: “COMMUNE”: a selection based on commune (municipality) limits, or “PARCEL”: a selection based on one or several land parcel limits. Other selection datasets may be made available at a later date.
pindir_ident	Only used when perimeter_type = INDIRECT, this parameter expects a string containing an ID or a comma-separated list of IDs identifying selection polygons inside the dataset specified in pindir_layer_name. When pindir_layer_name = “COMMUNE”, the numbers are to be understood as the official numbers of the commune/municipality according to the FSO/OFT/BFS, e.g. Gemeinde Zollikon = “0161” or “161”. When pindir_layer_name = “PARCEL”, the numbers are to be understood as the unique identifier of a land lot according to its E-GRID (“Eidgenössische Grundstücksidentifikation”) without the prefix. Example E-GRID identifier: “CH557103779070”. For more information about E-GRID, please refer to https://www.cadastre.ch/de/manual-av/topic/realestate/egrid.html

pdir_polygon	Only used when PERIMETER_TYPE = DIRECT, this parameter defines the selection polygon as a GeoJSON geometry object. Example:
	
	`{`
	`  "type": "Polygon",`
	`  "coordinates": [`
	`  [`
	`  [2675300, 1251900],`
	`  [2678100, 1251900],`
	`  [2678100, 1253500],`
	`  [2675300, 1253500],`
	`  [2675300, 1251900]`
	`  ]`
	`  ]`
	`}`

	For detailed information about GeoJSON geometry objects, please refer to https://geojson.org/geojson-spec.html#geometry-objects Notes: • The only accepted geometry type is “Polygon”. • The polygon must be properly closed, meaning that the last coordinate must be equal to the first coordinate. • The polygon cannot contain any self-intersections. • Multi-part or donut geometries are not supported.
pdir_coordsys	Only used when **perimeter_type** = DIRECT, this parameter sets the coordinate system (Swiss terrestrial reference frame as defined by Swisstopo) used in **pdir_polygon**. Possible values are “LV95” or “LV03”.
products	Contains a list of ordered products with their associated format. Example: `"products": [ {"product_id": 121, "format_id": 128}, {"product_id": 2, "format_id": 101}, {"product_id": 343, "format_id": 61}, {"product_id": 103, "format_id": 4} ]` The **product_id** references the product.id attribute and the **format_id** references the **format.id** attribute as returned by the “**GET products**” service. Invalid references will result in an http error.

Response parameters are:

Parameter	Description
order_id	Unique order identifier as a string of 32 lower-case hexadecimal characters. The order identifier is generated by the server when the order is submitted and is used to query order status and to download the results.
timestamp	The server timestamp of the order submission.
status_url	A URL that can be used to request the order status as a JSON document.
download_url	A URL that can be used to download the ordered data. The URL will return “404 Not Found” until the order has been processed and the data is available.

Restrictions

Due to size and performance considerations, the following restrictions are currently in place:

- Raster and/or LAZ products can only be ordered using a DIRECT order perimeter with a surface smaller than 5 square kilometers. The exact surface area is subject to change without notice, the active limitation will be reported in the http error message.
- Raster and/or LAZ products may be ordered for an INDIRECT perimeter only when requesting one single PARCEL (« Grundstück ») at a time. The parcel geometry may be multi-part, however.
- Raster and/or LAZ products cannot be ordered for an INDIRECT perimeter referencing a COMMUNE (« Gemeinde »), regardless of size.
- None of the above restrictions are enforced for product 103: ”Raster-Übersichtsplan”

GET <order id>

Returns the status of a specified order ID as returned by “POST orders” call. The response contains the order status as well as an internal ID that is used by Kanton Zürich for support and debugging purposes.

Note that order IDs are only stored for about one week after order submission before they are deleted, after which the service will return 404 Not Found.

URL	https://geoservices.zh.ch/geoshopapi/v1/orders/<order id>
Request method	GET

Example response for a given order ID	{
	"finished": "2017-11-10T16:17:07",
	"internal_id": 532,
	"order": {
	"email": "dr@inser.ch",
	"perimeter_type": "INDIRECT",
	"pindir_ident": [
	211,
	"0261"
	],
	"pindir_layer_name": "COMMUNE",
	"products": [
	{
	"format_id": 128,
	"product_id": 51
	},
	{
	"format_id": 101,
	"product_id": 84
	}
	]
	},
	"order_id": "ee2efc3f61064a9db0d7c41b42df1c5a",

```
    "status": "SUCCESS",
    "submitted": "2017-11-10T16:16:11"
}
```


The above example indicates that order ID “ee2efc3f61064a9db0d7c41b42df1c5a” was successfully terminated on 10 November 2017 at 16:17:07.

Parameter	Description
finished	Timestamp indicating when the data extraction was terminated. The reception of the mail indicating the download link may take an additional, unspecified amount of time.
internal_id	Internal ID, should be used when reporting unexpected errors. Should not be used for anything else as no guarantees are given for long-term uniqueness.
order	Contains a copy of the submitted order parameters.
order_id	The order ID that was queried.
status	The current status of the order, see below for possible values.
submitted	Timestamp indicating when the job was first submitted.

The possible return values for “status” are:

Value	Description
SUBMITTED	The order has been received by the server but has not yet been added to the queue.
QUEUED	The order has been added to the queue and is waiting for an available server before it is processed.
WORKING: Processing	The order has been pulled from the queue and is currently being processed.
WORKING: Data transfer started	The order has been fulfilled and passed on to the WebTransfer module

SUCCESS: Data transfer succeeded	The order was fulfilled and an email containing the download link was sent to the specified address.
FAILURE: Output dataset exceeded maximum size, not delivered	The ordered dataset was too big.
FAILURE: Data transfer failed	WebTransfer reported an error transferring the ordered data and/or transmitting the mail containing the download link. Most probable cause is having specified an invalid mail address.

For programmatic interpretation of the status codes, use the following rule: if a colon exists, extract the substring before the colon, otherwise the whole string. The extracted string is to be considered as a value of an enumeration set.

Example: “WORKING: Processing” → “WORKING”

Please note that future versions of the API may split the second part of the status into a separate JSON attribute, thereby avoiding the need for the above-mentioned split.

GET <order id>/download
Requests the download of data from the specified order ID. If the order ID does not exist or the data isn’t available yet, the server will return 404 Not Found.

Note that order IDs are only stored for about one week after order submission before they are deleted, after which the service will return 404 Not Found.

URL	https://geoservices.zh.ch/geoshopapi/v1/orders/<order id>/download
Request method	GET

Appendix 1: HTTP response codes and errors

The API returns an HTTP status code for every request. Some errors may also return further information in a message encoded according to the Content-Type header that was returned (either application/json or text/plain).

200	OK	Success
202	Accepted	Success; the operation has been started.
400	Bad Request	The request was improperly formatted. Typically this means that the JSON is not well-formed or that the order did not conform to technical or logical constraints. Refer to the error message for further details.
404	Not Found	The URI requested is invalid or the requested resource does not exist. This can happen if the order status of an unknown order ID is requested.
504	Gateway timeout	A necessary back-end component is off-line or didn’t answer in a timely manner.

Appendix 2: Product list structure

The list of available products is returned as a JSON structure with the following schema:

```
{
  "definitions": {},
  "$schema": "http://json-schema.org/draft-06/schema#",
  "$id": "https://geoservices.zh.ch/geoshopapi/v1/orders.json",
  "type": "object",
  "properties": {
    "timestamp": {
      "type": "string"
    },
    "formats": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {
            "type": "integer"
          },
          "name": {
            "type": "string"
          }
        }
      }
    },
    "products": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {
            "type": "integer"
          },
          "name": {
            "type": "string"
          },
          "description": {
            "type": "string"
          },
          "type": {
            "type": "string"
          },
          "formats": {
            "type": "array",
            "items": {
              "type": "integer"
            }
          }
        }
      }
```

```
    },
    "communes": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "id": {
            "type": "string"
          },
          "name": {
            "type": "string"
          }
        }
      }
    }
  }
}
```

Note that for performance reasons the product list is cached internally and that the timestamp references the time the product list cache was last refreshed. The timestamp is therefore not necessarily current.

Sample JSON result (abridged for readability):

```
{
    "timestamp": "2017-12-07T15:36:26",
    "formats": [{
            "id": 2,
            "name": "ESRI Personal Geodatabase (.mdb)"
        },
        {
            "id": 3,
            "name": "ESRI Shapefile (.shp)"
        },
        {
            "id": 4,
            "name": "GeoTIFF (.tif)"
        }
    ],
    "products": [{
            "id": 20,
            "name": "Luftbilder 1981-2000",
            "description": "Historische Luftbilder Kanton ZH 1981 -
2000",
            "type": "Raster",
            "formats": [
                21
            ]
        },
        {
            "id": 53,
```

```
            "name": "Veloparkierungsanlagen",
            "description": "Veloparkierungsanlagen des Kantons Zürich
ohne Stadt Zürich.",
            "type": "Vektor",
            "formats": [
                2,
                3,
                25,
                101,
                128
            ]
        }
    ],
    "communes": [{
            "id": "0001",
            "name": "Aeugst am Albis"
        },
        {
            "id": "0002",
            "name": "Affoltern am Albis"
        }
    ]
}
```