Download data

Ground-based measurements

MeteoSwiss primarily uses the Federal Spatial Data Infrastructure FSDI which is operated by swisstopo

The documentation below relates to the data provided on the FSDI.

Numerical weather forecasting model data

'Numerical weather forecasting model ICON-CH1/2-EPS' data are made available for download by the CSCS - Swiss National Supercomputing Centre

> **INFO**
>
> The **different download options** are documented **here**.

How to download files automatically

The FSDI provides a REST API which adheres to the OGC STAC API standard.

Each dataset is in its own collection - calling the `/collections` endpoint will **show all collections available**:
`https://data.geo.admin.ch/collections`

Each collection has a description - calling the `/getCollections` endpoint, will **show all collection metadata of a particular collection**:
e.g. get the details of the collection "Automatic weather stations":
`https://data.geo.admin.ch/api/stac/v1/collections/ch.meteoschweiz.ogd-smn`

How to check for new data

When downloading data from the STAC API, you might want to make sure to always retrieve the most current data.

By default, asset objects are cached for 2 hours (Exception: Collections with 10 minute values are cached for 10 seconds).

We highly recommend to use preconditioning via the `If-Match` or `If-None-Match` headers (mostly the latter one) when making calls to the STAC API. This reduces unnecessary traffic.

- When the client sends an `If-None-Match` header containing the `ETag` of the current (local) version of the requested object, the server

compares it to the currently available resource's `ETag` on the server.

- Only in case the two values don't match, the requested object is sent.
- Otherwise the server responds with a `304 Not Modified` without a body, which tells the user (i.e. the client) that his version of the asset is still good to use.

For more information check swisstopo's STAC API documentation

Examples

Ground-based measurements

This Jupyter notebook shows a simplified workflow for downloading and processing ground-based measurements of station `Salen-Reutenen (HAI)` from the STAC API.

- The code used in the notebook is for demo purposes only. Code quality is not on production-grade level.
- The packages that are required in order to run the Jupyter notebook are specified in the Pipfile A simple `pipenv install` will install the dependencies in a virtual environment on your machine.

Numerical weather forecasting model data

The MeteoSwiss opendata-nwp-demos repository contains a collection of Jupyter notebooks that demonstrate how to access, download, and

visualise data from numerical weather prediction (NWP) ICON-CH1/2-EPS models.

How CSV files are structured

Data granularity

For all types of data MeteoSwiss uses standard granularities. Depending on the application not all granularities are available.

For Ground-based measurements the lowest granularity is usually called 'original data' (`Originalwert`). Higher granularities are called 'aggregations' or 'aggregated values'. The World Meteorological Organization (WMO) does issue guidelines on how national weather services have to aggregate values and MeteoSwiss does follow these guidelines.

> **TIP**
>
> If you need hourly, daily, monthly or yearly values, we strongly recommend that you download the according granularity. Downloading the raw data (10min) and calculating sums or means yourself, will not always lead to the same results! Furthermore for historic data it is possible that manual data corrections have only been applied on higher granularities (like hourly or daily data), which means that historic raw data can still contain errors.

This is the overview of the granularities for Ground-based as well as for Climate stations - Homogeneous data series and Climate precipitation stations - Homogeneous data series used by MeteoSwiss:

Granularity	Name	Description	Used for
`t`	10min value	At MeteoSwiss this is the standard granularity for realtime data of the automatic measurement network SwissMetNet (SMN) or the model output. Meteorological observations do also use this granularity but only offer values at fixed intervals like 6UTC, 12UTC and 18UTC (called `Terminwerte`)!	SMN OBS
`h`	Hourly value	Either aggregated from 10min values or provided by the instrument/network	Pollen

Granularity	Name	Description	Used for
`d`	Daily value	Used throughout the MeteoSwiss measurement network before automatisation in 1981 started. Today still used for manual precipitation and snow measurements. For automatic stations daily values are calculated using 10min values according to WMO guidelines.	NIME
`m`	Monthly value	Usually aggregated from daily values and widely used in climatology for homogenised data and norm values and for seasonal data. For some very old data series (before 1864) only monthly data exists!	Homogeneous climate data series Climate normals

Granularity	Name	Description	Used for
`y`	Yearly value	Usually aggregated from daily values and mostly used in climatology or climate change scenarios.	Climate change scenarios

Update frequency

For Ground-based and Atmosphere measurements as well as for Climate stations – Homogeneous data series and Climate precipitation stations - Homogeneous data series MeteoSwiss provides an optimised directory structure separating older historical data, which is not updated regularly, and more recent data, which is updated more often. For realtime data we provide a third "now" directory with a high update frequency.

This is the overview:

Type	Description	Update frequency	Used for
`historical`	From the start of the measurement until December 31st of last year	Once a year	Granularity `m`, `d`, `h`, `t`
`recent`	From January 1st of this year until yesterday	Daily at 12UTC	Granularity `m`, `d`, `h`, `t`
`now`	The most recent realtime data from yesterday 12UTC to now	Every 10min	Only Granularity `h`, `t`
`no type`	For certain data types this concept does not apply	varies	varies (e.g. Granularity `y`)

Column separators and decimal dividers

Generally, columns are separated with a semicolon (`;`).

If you are using Windows Excel in a language version other than German, change the separator to `;` when opening the file (respectively use the 'Text Conversion Assistant' in the 'Data' menu).

> **NOTE**
>
> Why did we choose the semicolon as a separator?
>
> - Microsoft uses CSV files with a semicolon as separator in the default settings of the German Windows Excel version.
> - We assume that a high proportion of users open our CSV files with this setting.
> - We are monitoring the development of the proportion of users with this setting and reserve the right to reconsider our decision after weighing up the pros and cons.

The decimal divider is a full stop (`.`).

Encoding

CSV files are encoded in `Windows-1252` to ensure that they are decoded correctly in Excel.

> **NOTE**
>
> `UTF-8 BOM` would actually be a more versatile and widely used encoding that would ensure the same thing. For technical reasons,

> however, we still use `Windows-1252`.

How date/time, time intervals and missing values are represented

Time stamps and time intervals

Date/Time is expressed as `dd.mm.yyyy HH:MM`.

All reference time stamps at MeteoSwiss are in UTC Depending on the granularity the time stamp does define different intervals:

- `t`: The sum, mean or max/min of the last 10 minutes (`ReferenceTS` 16:00 = `15:50:01 to 16:00:00`)
- `h`: The sum, mean or max/min of the last six 10min-values (`ReferenceTS` 16:00 = `15:10 to 16:00`). Please note: Hourly values before 2018 were calculated differently based on the `SYNOP` schedule (`ReferenceTS` 16:00 = `15:50 to 16:40`)!
- `d`: For most parameters the sum, mean or max/min from 00:00 to 23:50 of the according date. Exception for precipitation and snow (manual measurement times used for consistency) where the interval is 6:00 UTC until 5:50 UTC tomorrow (`ReferenceTS` 22.6.2023 = `22.6.2023 6:10 UTC to 23.6.2023 6:00 UTC`)

- `m`: The sum, mean or max/min of the whole month from 1st to last day of month (`ReferenceTS` 1.6.2023 = `1.6.2023 00:10 UTC to 30.6.2023 24:00 UTC`)
- `y`: The sum, mean or max/min of the whole year (`ReferenceTS` 1.1.2023 = `1.1.2023 00:10 UTC to 31.12.2023 24:00 UTC`)

Accordingly, it follows that:

- for granularity `t` and `h` the time stamp defines the end of the measurement interval and
- for higher granularities (`d`, `m` and `y`) the time stamp defines the beginning of the interval!

Missing values

Missing values (e.g. due to instrument failure) are empty fields. Empty columns are used when a parameter is not measured at all at a certain station.

How GRIB2 files are structured

Numerical weather forecasting model data provided by MeteoSwiss are distributed in the GRIB2 format a binary and highly efficient format standardised by the World Meteorological Organization (WMO) for the storage and exchange of meteorological gridded data.

GRIB2 files are designed to store multi-dimensional atmospheric fields, such as temperature, wind, or pressure across various heights, times, and geographical coordinates.

Overview of GRIB2 file content

Each GRIB2 file contains:

- A header with metadata such as model name, run time, parameter, and spatial resolution.
- Encoded binary data for a single variable, at a single reference time, for one forecast time step, model, and simulation type (deterministic or perturbed).

These files are designed for efficient storage and transfer of high-resolution model output in a compact format.

For a detailed breakdown of what is included in a single GRIB2 file, see the GRIB File Structure section

Reading GRIB2 files

Because GRIB2 files are binary and not human-readable, they require specialised tools or libraries for access.

With Python

GRIB2 files can be read and processed using Python libraries that support GRIB decoding and metadata extraction. These libraries integrate with common scientific Python tools for data analysis and visualisation.

Example workflows and tools are available in the Exploring GRIB files in Python

With command-line tools

The ecCodes command-line tools such as `grib_ls` and `grib_dump`, allow inspection and extraction of metadata and values directly from the terminal.

For examples and usage, see:

- Reading GRIB files using ecCodes

Edit this page