Terms of Service

This document describes the terms for using the MET Weather API. Changes to the terms will be announced on the mailing list.

The service has good capacity and can handle relatively large volumes of requests, but the capacity is not unlimited. We ask all users to respect the guidelines so that the service will be stable.

Most important rules:

- You must identify yourself
- You must not cause unnecessary traffic
- You must not overload our servers

General information about the service

All weather data on the api are continually updated. New data will be made available in the service continuously. In order to receive important notifications about changes to the service we strongly suggest you either subscribe to either our api-users mailing list or the RSS feed.

When new versions of products are introduced (usually after a beta period), the older versions will be deprecated. This can be monitored by checking for a 203 status code (instead of the usual 200); if so this should be logged and/or shown as a warning. Deprecated versions will be terminated after a reasonable time period (usually 1 month).

There are no guarantees of delivery regarding this service, or possibilities to obtain an SLA.

If you need to get a customized delivery of weather data and products, you can see what is offered in ECOMET catalogue on ecomet.eu.

Legal stuff

Identification

All requests must (if possible) include an identifying User Agent-string (UA) in the request with the application/domain name, optionally version number. You should also include a company email address or a link to the company website where we can find contact information. If we cannot contact you in case of problems, you risk being blocked without warning.

Examples of valid User-Agents:

```
"acmeweathersite.com support@acmeweathersite.com"
"AcmeWeatherApp/0.9 github.com/acmeweatherapp"
```

In cases where this is not possible (e.g. image links or simple cross-origin requests in Javascript), the website must instead identify itself using an `Origin` (Javascript) or `Referer` (for image links) header.

Attribution

All open data require attribution as specified in the CC BY 4.0 license:

You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

For more information, please see our Licensing and Data Policy.

Personal data

As a app or site developer, the confidentiality of the users' personal data are your own responsibility. If you call the API directly from an app or browser, the user's IP address will be stored in our logs, along with any possible geocoordinates used in requests. To guarantee anonymity we recommend using a proxy gateway so that the ip addresses of the users are not revealed to us.

All api.met.no access logs are stored in our own data center in Oslo, Norway. Company email addresses (used in User-Agent headers) are not considered personal information for GDPR and other privacy purposes. For more information, see our Privacy Policy Statement.

Technical issues

Traffic

- Do not ask too often, and don't repeat requests until the time indicated in the `Expires` response header.

- Cache data locally and and use the `If-Modified-Since` request header to avoid repeatedly downloading the same data.

- Don't generate unnecessary traffic, e.g. by repeated requests for data which never change (MetAlerts CAP files, images with timestamps and so on).

- Don't schedule many requests at the same time, e.g. every hour on the dot or when the forecast model runs are finished. Add a random number of minutes to the time of the requests as our data are continously updated. Spread your traffic evenly out over time so it makes a flat curve, not a sawtooth.

- When using requests with latitude/longitude, truncate all coordinates to max 4 decimals. There is no need to ask for weather forecasts with nanometer precision! For new products,

requests with 5+ decimals will return a 403 Forbidden.

- Avoid continuous updating of mobile devices. Applications on mobile devices must not retrieve new data as long as the application is not in use. If you need push notification (e.g. for MetAlerts), don't make more than one poll every 10 mins.

Bandwidth

Anything over 20 requests/second per application (total, not per client) requires special agreement. If you have a mobile app, this means the total traffic from all installations. For websites, this means the total for all servers and/or traffic coming directly from the browsers. Sites exceeding this limit are likely to be throttled, possibly also blocked if we cannot contact you for optimization.

Copy animations and other large objects to your own server if you expect heavy traffic. We have good bandwidth, but sites with heavy traffic which links directly to our animations can use more bandwidth than we have taken into account. It is desirable that you inform your users about the time you retrieved the data so that they can consider whether they look at outdated data.

Protocols

We only support encrypted HTTPS for security reasons. Unencrypted HTTP requests are redirected to HTTPS, but since this is both a security risk and generates unnecessary traffic it is not allowed over extended periods of time. If you don't fix it after a reasonable interval we reserve the right to block such traffic.

Compression

All clients must support redirects and gzip compression (Accept-Encoding: gzip, deflate) as described in RFC 2616.

Caching and proxy

Web servers and mobile apps should cache all API responses to avoid repeatedly asking for the same data.

Use the information found in any cache headers, see RFC 2616. For example, use If-Modified-Since requests if the Last-Modified header exists. Note that the If-Modified-Since header should be identical to the previous Last-Modified, not any random timestamp (and definitely not in the future).

Direct client-to-API connections

Browsers and mobile apps should not contact the API directly, but instead use a local proxy (backend for frontend) server where you can cache data and add identification/authentication details (this is the canonical way of doing authentication in React since you cannot store client secrets in Javascript).

Note: The conditions for using CORS are still under review and will be finalized later.

Low-volume websites and mobile apps may use *simple* cross-origin Javascript requests and direct image links, provided the sites identifies itself as described above. If you start generating a lot of traffic against the API, you must set up a caching proxy gateway and route all traffic through this.

Non-simple javascript requests (i.e. using authentication or other custom headers) and CORS preflights (whitelisting of your domain in the ``Access-Control-Allow-Origin` header) is explicitly not supported.

Access control

Throttling and traffic shaping

Clients which don't follow the terms are liable to being throttled. This means requests will start getting a 429 HTTP status code instead of any content. Always check response `Status` headers and limit traffic immediately if this should happen. If you don't suspect you have excessive amounts of traffic, the problem is likely to be a breach of the TOS, usually lack of identification.

Abuse

Deliberate breach of our TOS, as well as trying to circumvent traffic rate limiting measures and/or impersonating traffic from other clients will result in a permanent ban in the use of our services. This includes fake or random strings as identification in the User-Agent header.