## Contents

1 Introduction ..................................................  1

2 Numeric forecast models .....................................  3
   2.1 WRF (Weather Research Forecast) ..........................  4
   2.2 WW3 (Wave Watch III) ......................................  5
   2.3 SWAN (Simulating Waves Nearshore) .......................  5
   2.4 ROMS (Regional Ocean Modeling System) ...................  5
   2.5 MOHID .....................................................  6

3 Common questions .............................................  7
   3.1 Getting an API key .........................................  7
   3.2 Coordinate systems .........................................  7
   3.3 Use of uppercase and lowercase .............................  7
   3.4 Languages ..................................................  8
   3.5 Time zones ..................................................  8
   3.6 Temporal formats ..........................................  8
   3.7 Numeric formats ..........................................  9
   3.8 Images ......................................................  9
   3.9 Response formats .......................................  9
   3.10 Exceptions ...............................................  9

4 Operations .................................................... 11
   4.1 Introduction .............................................. 11
   4.2 Common parameters to the /getNumericForecastInfo, /getTidesInfo, /getSolarInfo operations ............ 11
   4.3 Temporal interval ......................................... 13
   4.4 Output formats supported by operations /getNumericForecastInfo, /getTidesInfo and /getSolarInfo .... 13
   4.5 Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations .... 13
   4.6 Behavior on error ......................................... 18
   4.7 Behavior in the absence of data ........................... 26

5 /findPlaces operation ........................................ 31
   5.1 Introduction .............................................. 31
   5.2 Parameters ............................................... 31
   5.3 Results .................................................... 32
   5.4 Examples ................................................... 35

6 /getNumericForecastInfo operation .......................... 37
   6.1 Introduction .............................................. 37
   6.2 Temporal frame ............................................ 39
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3</td>
<td>Parameters</td>
<td>39</td>
</tr>
<tr>
<td>6.4</td>
<td>Results</td>
<td>42</td>
</tr>
<tr>
<td>6.5</td>
<td>Examples</td>
<td>49</td>
</tr>
<tr>
<td>7</td>
<td>/getTidesInfo operation</td>
<td>51</td>
</tr>
<tr>
<td>7.1</td>
<td>Introduction</td>
<td>51</td>
</tr>
<tr>
<td>7.2</td>
<td>Temporal frame</td>
<td>53</td>
</tr>
<tr>
<td>7.3</td>
<td>Parameters</td>
<td>53</td>
</tr>
<tr>
<td>7.4</td>
<td>Results</td>
<td>54</td>
</tr>
<tr>
<td>7.5</td>
<td>Examples</td>
<td>58</td>
</tr>
<tr>
<td>8</td>
<td>/getSolarInfo operation</td>
<td>59</td>
</tr>
<tr>
<td>8.1</td>
<td>Introduction</td>
<td>59</td>
</tr>
<tr>
<td>8.2</td>
<td>Temporal frame</td>
<td>59</td>
</tr>
<tr>
<td>8.3</td>
<td>Parameters</td>
<td>59</td>
</tr>
<tr>
<td>8.4</td>
<td>Results</td>
<td>59</td>
</tr>
<tr>
<td>8.5</td>
<td>Observations</td>
<td>62</td>
</tr>
<tr>
<td>8.6</td>
<td>Examples</td>
<td>62</td>
</tr>
<tr>
<td>9</td>
<td>Exceptions</td>
<td>65</td>
</tr>
<tr>
<td>10</td>
<td>A1. What’s new in version v3</td>
<td>67</td>
</tr>
<tr>
<td>11</td>
<td>A2. About this document</td>
<td>69</td>
</tr>
<tr>
<td>12</td>
<td>A3. Time zones</td>
<td>71</td>
</tr>
</tbody>
</table>
Introduction

The MeteoSIX API is a free web service for accessing meteorological and oceanographic information. It offers:

- The results of the different numeric forecast models, meteorological and oceanographic, that come directly from the outputs of the models, with any supervision by experts.
- Sunrise and sunset hours.
- Tides forecast.

In addition, in order to facilitate queries, it provides methods for obtaining some types of geographical entities.

Meteorological and oceanographic forecast information comes from the models executed daily by MeteoGalicia. These are executed on different grids, of different resolutions and spatial-temporal coverages (see Numeric forecast models). The information about sunrise and sunset hours is available for any point (see /getSolarInfo operation). Tides information is available for the Galician coast (see /getTidesInfo operation).

The current version of the API, which this manual refers to, is version v3 (3.0.0). This version is not compatible with previous versions, given that some parameters and behaviors have changed.
The numeric forecast information served by the API comes directly from the outputs of the forecast models executed daily by MeteoGalicia, without any expert supervision.

There are different models, and each of them runs on different grids (different coverage areas and resolutions). These executions usually last several hours, and their ending hour varies from one model/grid to another. For a grid, the execution ending hour can change from one day to another, and thus, the ending hours shown in the following tables are approximated. Once each execution ends, it can take several minutes until it is available through the API.

The temporal frequency for which both the models and the API have numeric forecast data is 1 hour.

Models and grids currently offered by the MeteoSIX API are the following (hours are expressed in UTC - Coordinated Universal Time):
### 2.1 WRF (Weather Research Forecast)

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Grid</th>
<th>Resolution</th>
<th>Execution start</th>
<th>Approximated execution end</th>
<th>First forecast hour</th>
<th>Forecast horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artabro1Km</td>
<td></td>
<td>1 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>07:30 UTC 19:30 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>72 h 84 h</td>
</tr>
<tr>
<td>RiasBaixas1Km</td>
<td></td>
<td>1 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>07:30 UTC 19:30 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>72 h 84 h</td>
</tr>
<tr>
<td>NortePortugal1Km</td>
<td></td>
<td>1 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>07:30 UTC 19:30 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>72 h 84 h</td>
</tr>
<tr>
<td>04km</td>
<td></td>
<td>4 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>05:00 UTC 17:00 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>96 h 84 h</td>
</tr>
<tr>
<td>12km</td>
<td></td>
<td>12 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>05:00 UTC 17:00 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>96 h 84 h</td>
</tr>
<tr>
<td>36km</td>
<td></td>
<td>36 km</td>
<td>00:00 UTC 12:00 UTC</td>
<td>05:00 UTC 17:00 UTC</td>
<td>01:00 UTC 13:00 UTC</td>
<td>96 h 84 h</td>
</tr>
</tbody>
</table>
## 2.2 WW3 (Wave Watch III)

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Grid</th>
<th>Resolution</th>
<th>Execution start</th>
<th>Approximated execution end</th>
<th>First forecast hour</th>
<th>Forecast horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galicia</td>
<td>0,05º</td>
<td>00:00 UTC</td>
<td>05:00 UTC 00:00 UTC 12:00 UTC 17:00 UTC</td>
<td>12:00 UTC 00:00 UTC</td>
<td>109 h</td>
<td>97 h</td>
</tr>
<tr>
<td>Iberica</td>
<td>0,25º</td>
<td>00:00 UTC</td>
<td>05:00 UTC 00:00 UTC 12:00 UTC 17:00 UTC</td>
<td>12:00 UTC 00:00 UTC</td>
<td>109 h</td>
<td>97 h</td>
</tr>
<tr>
<td>AtlanticoNorte</td>
<td>0,5º</td>
<td>00:00 UTC</td>
<td>05:00 UTC 00:00 UTC 12:00 UTC 17:00 UTC</td>
<td>12:00 UTC 00:00 UTC</td>
<td>109 h</td>
<td>97 h</td>
</tr>
</tbody>
</table>

## 2.3 SWAN (Simulating Waves Nearshore)

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Grid</th>
<th>Resolution</th>
<th>Execution start</th>
<th>Approximated execution end</th>
<th>First forecast hour</th>
<th>Forecast horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artabro</td>
<td>500 m</td>
<td>12:00 UTC</td>
<td>22:00 UTC</td>
<td>12:00 UTC</td>
<td>85 h</td>
<td></td>
</tr>
<tr>
<td>RiasBaixas</td>
<td>250 m</td>
<td>12:00 UTC</td>
<td>22:00 UTC</td>
<td>12:00 UTC</td>
<td>85 h</td>
<td></td>
</tr>
</tbody>
</table>

## 2.4 ROMS (Regional Ocean Modeling System)

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Grid</th>
<th>Resolution</th>
<th>Execution start</th>
<th>Approximated execution end</th>
<th>First forecast hour</th>
<th>Forecast horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galicia</td>
<td>0,02º</td>
<td>00:00 UTC</td>
<td>09:30 UTC</td>
<td>00:00 UTC</td>
<td>97 h</td>
<td></td>
</tr>
</tbody>
</table>
## 2.5 MOHID

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Grid</th>
<th>Resolution</th>
<th>Execution start</th>
<th>Approximated execution end</th>
<th>First forecast hour</th>
<th>Forecast horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Artabro</td>
<td>0,003º</td>
<td>00:00 UTC</td>
<td>12:30 UTC</td>
<td>00:00 UTC</td>
<td>49 h</td>
</tr>
<tr>
<td></td>
<td>Arousa</td>
<td>0,003º</td>
<td>00:00 UTC</td>
<td>12:30 UTC</td>
<td>00:00 UTC</td>
<td>49 h</td>
</tr>
<tr>
<td></td>
<td>Vigo</td>
<td>0,003º</td>
<td>00:00 UTC</td>
<td>12:30 UTC</td>
<td>00:00 UTC</td>
<td>49 h</td>
</tr>
</tbody>
</table>

For further details about the different forecast models see [http://www.meteogalicia.es/modelos/index.action](http://www.meteogalicia.es/modelos/index.action).
CHAPTER 3

Common questions

The API provides a set of operations that are executed through HTTP GET and HTTP POST requests to the service URL (API domain):

http://servizos.meteogalicia.es/apiv3/

The common structure of all requests is the following:

http://servizos.meteogalicia.es/apiv3/ruta_operación?[parámetros_operación]

For example:


Each type of request supports different types of parameters, which are described in the next sections.

3.1 Getting an API key

The use of the API is free but a user key is needed and it must be included in every request. A key can be requested through this form. (The same form can be used for recover a forgotten key).

This key is for private use, specific for each API user, and it is associated to the e-mail of the user. Requests that do not included a valid key are rejected with an exception message.

Note: the users of previous versions of the API maintain their keys active. It is not necessary that they request a new one.

3.2 Coordinate systems

Some API operations return data containing geometries. Also, some operations can take geometric values as parameters (coordinates). Currently, the only supported coordinate system for geometric data is WGS 84 (EPSG:4326).

3.3 Use of uppercase and lowercase

All the parameters except the API_KEY parameter do not distinguish between upper or lowercase, but using the values as they appear in this document is recommended. The same applies to the names of those parameters. Thus, the parameter:
models=WRF,WW3

is equivalent to:

MODELS=wrf,ww3

and so it is to:

MoDeLS=wRF,Ww3

### 3.4 Languages

The API supports the following languages:

- Galician
- Spanish
- English

The default language is English.

### 3.5 Time zones

The API allows to specify a time zone for dates and times. If no time zone is specified, the data of the responses are in the Galician time zone, that is, UTC+1 in winter and UTC+2 in summer.

The supported time zones are listed in the appendix A3. Time zones.

### 3.6 Temporal formats

The format used in the dates and times returned by the API is the following:

`yyyy-MM-ddTHH:mm:ssZZ`

where:

- `yyyy`: year.
- `MM`: month.
- `dd`: day of month.
- `HH`: hour (in 24 hour format).
- `mm`: minutes.
- `ss`: seconds.
- `ZZ`: deviation with respect to the UTC time. The ZZ characters are replaced by a code representing the deviation of the hour with respect to the UTC hour. For example, for a time in Galician local time, in summer, the ZZ characters are replaced by the code +02; for a time in Galician local time, in winter, they are replaced by the code +01. For times in UTC, the ZZ characters are replaced by the code Z.

Examples:

---

1 UTC is the [Coordinated Universal Time](https://en.wikipedia.org/wiki/Coordinated_Universal_Time).
In the requests parameters with date and time values, the format must be:

```
yyyy-MM-ddTHH:mm:ss
```

It is the same format, but without the difference with respect to the UTC hour.

### 3.7 Numeric formats

The decimal separator is always the dot, ".", not depending on the language.

### 3.8 Images

For some variables, like wind or sky state, besides the values themselves, the API returns references (URLs) to icons of meteograms representing the values. All those images are in PNG format and are 24x24 pixels sized.

### 3.9 Response formats

Depending on the kind of operation and on the result, the supported formats are the following:

- GeoJSON
- GML 3.2.1
- HTML
- KML 2.0

The desired output format can be specify as a request parameter. If the request does not specify any format, the response will be returned using the default format.

### 3.10 Exceptions

In case of error, the API will return information about it, in the error format specified in the request or in the default error format. Supported formats for exceptions are:

- JSON
- XML

---

2 Even when the declared MIME-TYPE is JSON (GeoJSON lacks of specific MIME-TYPE), the format used is GeoJSON, which adds to JSON support for geographic information. GeoJSON will be used along this document, though sometimes JSON will be the chosen word.
4.1 Introduction

API v3 supports four operations:

• /findPlaces operation:
  Allows to search places by their name.
• /getNumericForecastInfo operation:
  Returns numeric forecast information (atmospheric and oceanographic).
• /getTidesInfo operation:
  Returns tides information.
• /getSolarInfo operation:
  Returns information about sunrise and sunset hours.

The /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations share some common characteristics such as parameters and structure of the response. They are described below.

4.2 Common parameters to the /getNumericForecastInfo, /getTidesInfo, /getSolarInfo operations

The parameters common to these operations are:
<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory</th>
<th>Possible values</th>
<th>Default value</th>
<th>Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>API_KEY</td>
<td>YES</td>
<td>API key</td>
<td>-</td>
<td>See Getting an API key section.</td>
</tr>
<tr>
<td>locationIds †</td>
<td>NO</td>
<td>Place identifiers</td>
<td>-</td>
<td>Place identifiers list on which the data will be obtained, comma separated (see /findPlaces operation). If there are no places with those identifier, an exception is returned. Currently, there is a limit of 20 locations, if more localizations are indicated an exception will be returned.</td>
</tr>
<tr>
<td>coords ‡</td>
<td>NO</td>
<td>xa,ya;xb,yb;...</td>
<td>-</td>
<td>Places coordinate list over which the data is requested. Each coordinate pair is separated of next by semicolon. Each pair, longitude and latitude are separated by comma. Order is longitude,latitude. Currently, there is a limit of 20 coordinate pairs, if any more pair is indicated, an exception will be returned.</td>
</tr>
<tr>
<td>startTime</td>
<td>NO</td>
<td>Time instant, in the format yyyy-MM-ddTHH:mm:ss</td>
<td>Actual instant</td>
<td>First temporal instant for which information is requested. See Temporal interval.</td>
</tr>
<tr>
<td>endTime</td>
<td>NO</td>
<td>Time instant, in the format yyyy-MM-ddTHH:mm:ss</td>
<td>-</td>
<td>Last time instant for which data will be returned. If omitted, data until the last available time instant will be returned, within the established limits for each request. See Temporal interval.</td>
</tr>
<tr>
<td>lang</td>
<td>NO</td>
<td>'gl' 'es' 'en'</td>
<td>'en'</td>
<td>Language in which texts will be returned, including the texts in exceptions.</td>
</tr>
<tr>
<td>tz</td>
<td>NO</td>
<td>Time zone identifer (see Time zones)</td>
<td>“Europe/Madrid”</td>
<td>Time zone in which hours will be displayed.</td>
</tr>
<tr>
<td>CRS</td>
<td>NO</td>
<td>Coordinate system identifier</td>
<td>‘EPSG:4326’</td>
<td>Currently the only possible value is EPSG:4326.</td>
</tr>
<tr>
<td>format</td>
<td>NO</td>
<td>'gml3' 'kml' 'application/json' 'text/html'</td>
<td>'application/json'</td>
<td>Format in which the results will be returned.</td>
</tr>
<tr>
<td>style</td>
<td>NO</td>
<td>Name of the style to apply to responses with HTML format</td>
<td>‘default’</td>
<td>It can only be specified when the value of the parameter format is text/html. Currently, the only accepted value is default.</td>
</tr>
<tr>
<td>exceptionsFormat</td>
<td>NO</td>
<td>'application/json' 'application/xml'</td>
<td>'application/json'</td>
<td>Format in which the exceptions will be returned.</td>
</tr>
</tbody>
</table>

There are, then, two ways of indicate points over which a query will be executed:

- From a set of coordinate pairs x,y.

---

† In version v2 of the API this parameter was named locationId.
‡ One and only one of the two parameters, locationIds or coords, must be present.
§ In version v2 of the API this parameter was named coord.
• From a set of place id previously obtained by /findPlaces operation.

4.3 Temporal interval

startTime and endTime parameters indicate temporal interval for which data would be returned. Both parameters must follow yyyy-MM-ddTHH:mm:ss format, but in some operations only day is taken in account, not hour. In in-depth operation sections, it is indicated if hour is taken in account or not.

In cases when hour is taken in account, only data between startTime and endTime is returned. For example, if startTime value is 2014-05-06T07:01:00, values for 07:00:00 of 6th day would not be returned, in the case of numeric forecast, first value that would be returned is 08:00:00 of that day (it would be the first time instant with data after or equals 07:01:00).

The lower bound for temporal interval of the requests is the first time instant of the actual day (day when the request is made). There is no upper bound, but for each operation, a limit in the total number of days that can be requested exists (counting from the first day that is requested, not the actual day; see sections dedicated for each operation). If startTime or endTime are previous to first time instant of the actual day, an exception is returned. If endTime is previous to startTime, an exception is also returned. If endTime is previous to the actual time instant, it is mandatory to specify startTime in the request (except when the operation does not take in account the hour).

If no parameter startTime is indicated, data available from the actual time instant is returned, up to the endTime (if it was indicated in the request), or up to a default number of days for each operation.

4.4 Output formats supported by operations /getNumericForecastInfo, /getTidesInfo and /getSolarInfo

• GML: returns data in GML format, 3.2.1 version.
• HTML: returns a HTML document with requested data. It includes references to CSS files and PNG icons.
• JSON: it returns data in GeoJSON format.
• KML: returns data with KML 2.0 format. Information comes embedded within the description element in the form of HTML (with the same contents specified by the HTML format).

4.5 Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations

Data returned by these operations are structured in the following way:

• Locations: data is returned for one or more localizations.
  – Days: each location contains information for one or more days. Information for full day may not be returned for each day, but for a time subset contained within the day.
    • Variables: Within each day, information is organized by variables.
      • Values: in each variable, values themselves are included, ordered by time instants they correspond to.

To represent the location set, following data structure is used:
For the geometry (coordinates) of each location, following representations are used:

<table>
<thead>
<tr>
<th>Format</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GML</td>
<td>It is represented with a <code>gml:point</code> element contained in a <code>geometry</code> element.</td>
</tr>
<tr>
<td>HTML</td>
<td>They are represented as text inside HTML.</td>
</tr>
<tr>
<td>JSON</td>
<td>It is represented by a point geometry object (&quot;type&quot;: &quot;Point&quot;).</td>
</tr>
<tr>
<td>KML</td>
<td>Represented by a <code>Point</code> element.</td>
</tr>
</tbody>
</table>

When request contains `locationIds` parameter, following attributes for each valid requested location are included: id, name, municipality, province and place type:

<table>
<thead>
<tr>
<th>Format</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GML</td>
<td>They are represented as attributes inside the <code>location</code> element of each <code>gml:featureMember</code> (id, name, municipality, province and type).</td>
</tr>
<tr>
<td>HTML</td>
<td>They are represented as text inside HTML.</td>
</tr>
<tr>
<td>JSON</td>
<td>The are represented as attributes of the <code>properties</code> object of each <code>feature</code> (id, name, municipality, province and type).</td>
</tr>
<tr>
<td>KML</td>
<td>Each attribute is represented as a <code>Data</code> element within the <code>ExtendedData</code> element of the <code>Placemark</code> with the following names (name): <code>location_id</code>, <code>location_name</code>, <code>location_municipality</code>, <code>location_province</code>, <code>location_type</code>.</td>
</tr>
</tbody>
</table>

Each location contains a set of one or more days (days) \(^1\). In HTML and KML, information for each day is shown in HTML tables. In JSON and GML, the temporal interval for which data is being returned within each day is indicated. This means that available data has been returned for that temporal interval, but:

- For part of the temporal interval may not be available data, in this case, value for the instant will be null or blank.
- For the part of the day that is left outside the temporal interval may be data that there is not being returned. This happens, for example, when they fall outside the temporal interval indicated in the request or, if no temporal interval was indicated in the request, they correspond to instants previous to the actual temporal instant.

Temporal range that covers each day is specified as following:

<table>
<thead>
<tr>
<th>Format</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GML</td>
<td><code>gml:TimePeriod</code> element is used, containing <code>gml:begin</code> and <code>gml:end</code> elements with initial and final period time instants respectively.</td>
</tr>
<tr>
<td>JSON</td>
<td>“timePeriod” element is used, containing <code>begin</code> and <code>end</code> with the initial and final instants of the time period, respectively.</td>
</tr>
</tbody>
</table>

In both cases, temporal instants will have the yyyy-MM-ddTHH:mm:ss+XX format, as indicated in the section Temporal formats. Information for each day is joined in one or more contained variables in a `variables` element or object.

More in detail for each response format:

- **JSON:**
  GeoJSON object with the following structure \(^4\) is retrieved:

\(^1\) In case of `/getTidesInfo` operation, there are more elements inside `properties` object (JSON) and `location` element (GML).

\(^4\)
[ "type": "FeatureCollection", 
  "crs": { 
    "type": "name", 
    "properties": { 
      "name": CRS 
    } 
  }, 
  "features": [ 
    { 
      "type": "Feature", 
      "geometry": { 
        "type": "Point", 
        "coordinates": [ 
          X, 
          Y 
        ] 
      }, 
      "properties": { 
        "id": ID, 
        "name": NAME, 
        "municipality": MUNICIPALITY, 
        "province": PROVINCE, 
        "type": TYPE, 
        "days": DAYS_ARRAY 
      } 
    } 
  ] 
}

Where:

- **CRS** indicates used coordinate system (currently, it is always urn:ogc:def:crs:OGC:1.3:CRS84).
- **X** and **Y** are the location coordinates (longitude and latitude respectively).
- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only returned when the operation is called with **locationIds** parameter.
- **DAYS_ARRAY** is a JSON array containing as many elements as days for which information is returned. Each day follows this structure:

  
  ```
  { 
    "timePeriod": { 
      "begin": { 
        "timeInstant": START_DATE 
      }, 
      "end": { 
        "timeInstant": END_DATE 
      } 
    }, 
    "variables": VARIABLES_ARRAY 
  }
  
  Where:
  
  * **START_DATE** and **END_DATE** respectively represent, initial and last period time instants for which values are being returned within the day.
* **VARIABLES\_ARRAY** is a JSON array of **variable** objects. The **variable** object structure varies for each specific operation and it is detailed in the section dedicated for each operation.

- **GML:**

GeoJSON object with the following structure is retrieved:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns="http://www.meteogalicia.es/meteosix"
xmlns:gml="http://www.opengis.net/gml">
  <gml:boundedBy>
    <gml:Envelope srsName=CRS>
      <gml:pos>MIN_Y MIN_X</gml:pos>
      <gml:pos>MAX_Y MAX_X</gml:pos>
    </gml:Envelope>
  </gml:boundedBy>
  ...
  <gml:featureMember>
    <location id=ID name=NAME municipality=MUNICIPALITY province=PROVINCE type=TYPE>
      <geometry>
        <gml:Point srsName=CRS>
          <gml:pos>
            Y X
          </gml:pos>
        </gml:Point>
      </geometry>
      <days>
        DAY_1
        DAY_2
        ...
      </days>
    </location>
  </gml:featureMember>
  ...
</gml:FeatureCollection>
```

Where:

- **CRS** indicates used coordinate system (currently, it is always **EPSG:4326**).
- **MIN\_Y, MIN\_X, MAX\_Y, MAX\_X**: minimum and maximum coordinates of the **Envelope** of the set of all locations. Order is latitude, longitude.
- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. The are only returned when the operation is called including the **locationIds** parameter.
- **Y** and **X** are the location coordinates (latitude and longitude respectively).
- **DAY\_1, DAY\_2**... are type **day** elements. Each day has the following structure:

```xml
<day>
  <gml:TimePeriod>
    <gml:begin>
      <gml:TimeInstant>
        <gml:timePosition>
          START_DATE
        </gml:timePosition>
      </gml:TimeInstant>
    </gml:begin>
    <gml:end>
      <gml:TimeInstant>
```

Chapter 4. Operations
Where:

* **START_DATE** and **END_DATE** respectively represent, initial and last period time instants for which values are being returned within the day.

* **VARIABLE 1, VARIABLE 2...** are type **variable** elements. Structure of a **variable** element, varies, again, for each specific operation and it is detailed in the section corresponding to each operation.

**• HTML:**

Response is a HTML document with its corresponding CSS. For example, for numeric forecast it will be like this:

**Forecast for latitude=44, longitude=-8**

<table>
<thead>
<tr>
<th>Summary</th>
<th>0h</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>7h</th>
<th>8h</th>
<th>9h</th>
<th>10h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sky state</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wind</td>
<td>Min</td>
<td>Max</td>
<td>15</td>
<td>53</td>
<td>19</td>
<td>19</td>
<td>18</td>
<td>17</td>
<td>15</td>
<td>15</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

**• KML:**

Response is a KML document with the following structure:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
<Document>
    <Placemark id="point_1">
        <ExtendedData>
            ...
        </ExtendedData>
    </Placemark>
</Document>
</kml>
```
Where:

- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only returned when the operation is called with `locationIds` parameter.

- **DESCRIPTION** is a type `description` element which contains a HTML document, the same as the one obtained if the response format was HTML. HTML code is inside a `CDATA` block.

- **X** and **Y** are the location coordinates (longitude and latitude respectively).

Each `Placemark` has an `id` attribute which values are `point_1`, `point_2`,... and so on for each `Placemark` contained into the response.

### 4.6 Behavior on error

In case of an error or exception, if it is not possible to retrieve full data, two cases are differentiated:

- **Errors that affect all requested locations**: in this case, response is an exception message in the default exception format or in the requested exception format. See *Behavior on global error*.

- **Errors that affect specific locations**: in this case, data is retrieved for locations that are not affected by errors, and for those locations that are affected by error, an exception message is returned. Response will be in the default data format or in the data format specified in the request (not exception format). In those blocks corresponding to error affected locations, an exception message will be included. See *Behavior on error for some locations*.

If all locations are affected by different errors (not a common error for all of them), indicated in the second case will be followed.
4.6.1 Behavior on global error

When an error is affecting all localizations for which data was requested, an exception message is retrieved. Format used will be default exception format or exception format indicated in the request parameter. In all formats, message contains:

- Exception code.
- Descriptive message.

In Appendix Exceptions all exception types are listed. Exception messages structure is:

- **JSON**:
  ```json
  { 
      "exception": { 
          "code": "CODE", 
          "message": "MESSAGE" 
      } 
  }
  ```

- **XML**:
  ```xml
  <?xml version="1.0" encoding="UTF-8"?>
  <Exception code="CODE">
      <message>MESSAGE</message>
  </Exception>
  ```

4.6.2 Behavior on error for some locations

When errors affect one or more locations, but not the same error for all of them, data for locations that are not affected by errors is returned, and for those locations affected by errors, an exception message will be returned. Response will follow default data format or data format indicated in the request (not exception format). In blocks corresponding of the affected locations by errors there will be an exception message instead of data. Exception messages structure for a specific location will be the following:

- **GML**:
  ```xml
  <?xml version="1.0" encoding="UTF-8"?>
    ...
    <gml:featureMember>
      <location>
        ...
        <exception code="CODE" value="MESSAGE"/>
        ...
      </location>
    </gml:featureMember>
    ...
  </gml:FeatureCollection>
  ```

  The geometry element is always included. If the location does not correspond with valid coordinates, its value is null:
  ```xml
  <geometry>
    <gml:null>missing</gml:null>
  </geometry>
  ```
The *days* element is also included, in this case with no content:

```xml
<days/>
```

In requests that contain *locationIds* parameter, *location* element attributes that can be obtained (*id, name, ...*) will be included. If all locations are affected by errors (but not a common error for all of them, in which case, see *Behavior on global error*) and none has valid coordinates, *gml:boundedBy* will be null:

```xml
<gml:boundedBy>
  <gml:null>missing</gml:null>
</gml:boundedBy>
```

For example, response for request:

```http
http://servizos.meteogalicia.es/apiv3/getNumericForecastInfo?
locationIds=aa,33,71941,1002197&variables=sea_water_temperature,
  significative_wave_height&lang=en&format=gml3&API_KEY=***
```

is:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://meteogalicia.es/meteosix">
  <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4326">
      <gml:pos>42.52197 -8.85002</gml:pos>
      <gml:pos>42.6397 -7.511</gml:pos>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <location id="aa">
      <geometry>
        <gml:null>missing</gml:null>
      </geometry>
      <days />
      <exception code="207"
        message="The value ‘aa’ is not a valid location id." />
    </location>
  </gml:featureMember>
  <gml:featureMember>
    <location id="33">
      <geometry>
        <gml:null>missing</gml:null>
      </geometry>
      <days />
      <exception code="210"
        message="There is no location with a location id equals to 33" />
    </location>
  </gml:featureMember>
  <gml:featureMember>
    <location id="71941" name="Monforte de Lemos"
      municipality="MONFORTE DE LEMOS" province="Lugo"
      type="locality">
      <geometry>
        <gml:Point srsName="EPSG:4326">
          <gml:pos>42.52197 -7.511</gml:pos>
        </gml:Point>
      </geometry>
    </location>
  </gml:featureMember>
</gml:FeatureCollection>
```
For HTML, in locations affected with errors, information about the exception will be included instead the HTML corresponding of those locations. For example, response for request:


is:

• HTML:
Forecast for latitude=-9, longitude=44:
Specified point falls outside the geographic limits for which there are data.

The value ‘aa’ are not valid coordinates.

Forecast for latitude=43.437222, longitude=-8.393145

<table>
<thead>
<tr>
<th></th>
<th>Summary</th>
<th>0h</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>7h</th>
<th>8h</th>
<th>9h</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Temperature</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td>13</td>
<td>13</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Significant wave height</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Min</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is about a request about three locations. For two of them – 44,-9 and aa – exception message is returned and for the other – -8.393145,43.43722239 – data is returned.

**JSON:**

For JSON, in locations affected with errors, information about the error is described in the exception JSON object inside the corresponding Feature object. As previously noted, exception JSON object is formed by two elements, code and message. The geometry element is always included. If location does not correspond to valid coordinates, its value is null:

```
geometry:null
```

days element is also included, in this case with null value:

```
days:null
```

In requests that contain locationIds parameter, response includes id,*name*,*municipality*... attributes (those that cannot be obtained will have null value). For example, response to request:

```
&variables=sea_water_temperature,
significant_wave_height&lang=en&
format=application/json&API_KEY=***
```

is:

```
{
    type:"FeatureCollection",
    crs:{
```
4.6. Behavior on error
• **KML:**

For KML, in locations affected by errors, error message will be included inside the *ExtendedData* block of the corresponding *Placemark*, in *Data* elements with *name* as *exceptionCode* and *exceptionMessage*:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Document>
    <Placemark id="point_1">
      <ExtendedData>
        ...
        <Data name="exceptionCode">
          <value>CODE</value>
        </Data>
        <Data name="exceptionMessage">
          <value>MESSAGE</value>
        </Data>
        ...
      </ExtendedData>
      ...
    </Placemark>
    ...
    <Placemark id="point_2">
      ...
    </Placemark>
    ...
  </Document>
</kml>
```

If error occurs because point coordinates is not valid, *Point* element is included as well. In requests that contain `locationIds` parameter, *Data* elements for location_id, location_name... that can be obtained are also being included. For example, response to request:

```
&variables=sea_water_temperature,significative_wave_height&lang=en&format=kml&API_KEY=***
```

is:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Document>
    <Placemark id="point_1">
      <ExtendedData>
        ...
        <Data name="location_id">
          <value>33</value>
        </Data>
        <Data name="exceptionCode">
          <value>210</value>
        </Data>
        ...
      </ExtendedData>
      ...
    </Placemark>
    ...
  </Document>
</kml>
```
4.6. Behavior on error
4.7 Behavior in the absence of data

In this section, behavior when no full data can be retrieved is described. Absence of data can be achieved in three levels:

- Days for which no data can be retrieved.
- Variables for which no data can be obtained for a full day.
- Time instant, contained in the temporal interval declared for the day, for which there are no data.

The following shows how these cases are handled in each format:

**JSON:**

If no data exists for any of the requested days, an exception is returned, except the case that data for more than one location has been requested. In this case, an exception message is returned inside each location with no data. For example:

```json
{
   "type":"FeatureCollection",
   "crs":{
      "type":"name",
      "properties":{
         "name":"urn:ogc:def:crs:OGC:1.3:CRS84"
      }
   },
   "features":[
      {
         "type":"Feature",
         "geometry":{
            "type":"Point",
            "coordinates":[-8.29504, 43.47569]
         },
         "properties":null,
         "exception":{
            "code":"216",
            "message":"Specified point falls outside the geographic limits for which there are data"
         }
      },
      {
         "type":"Feature",
         "geometry":{
            "type":"Point",
            "coordinates":[-8.19505, 43.47569]
         }
      }
   ]
}
```
If no data exists for some of the requested days but it exists for some other days, behavior is the following:

- for each day with no data between the actual instant (or startDate if present) and the last day with data, an object with the following structure is returned:

  ```json
  {
    "timePeriod": {
      "begin": {
        "timeInstant": "yyyy-MM-ddTHH:mm:ss"
      },
      "end": {
        "timeInstant": "yyyy-MM-ddTHH:mm:ss"
      }
    },
    "variables": null
  }
  ```

- If the operation was invoked specifying endTime parameter, for all days without data between the last day with data and endTime, an object with the previous structure is returned.

- If the operation was invoked without specifying endTime parameter, days without data after last day with data do not show in the response.

When a variable has no data for any time instant in the temporal interval of the day object, values object will be null. The day object structure will be the following:

```json
{
  "timePeriod": {
    "begin": {
      "timeInstant": "yyyy-MM-ddTHH:mm:ss"
    },
    "end": {
      "timeInstant": "yyyy-MM-ddTHH:mm:ss"
    }
  },
  "variables": [
    ...,
    {
      "name": VARIABLE_NAME,
      ...,
      "values": null,
      ...
    }
  ]
}
```

Case of data absence for a specific temporal interval is described in the dedicated sections for each operation.

- **GML:**

  4.7. Behavior in the absence of data
If no data exists for any of the requested days, an exception is returned, except the case that data for more than one location has been requested. In this case, an exception message is returned inside each location with no data. For example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns="http://meteogalicia.es/meteosix"
xmlns:gml="http://www.opengis.net/gml">
  <gml:boundedBy>
    <gml:Envelope srsName="EPSG:4326">
      <gml:pos>43.47569 -8.29504</gml:pos>
      <gml:pos>43.47569 -8.19505</gml:pos>
    </gml:Envelope>
  </gml:boundedBy>
  <gml:featureMember>
    <location>
      <geometry>
        <gml:Point srsName="EPSG:4326">
          <gml:pos>43.47569 -8.29504</gml:pos>
        </gml:Point>
      </geometry>
      <days/>
      <exception code="216"
        message="Specified point falls outside the geographic limits for which there are data."/>
    </location>
  </gml:featureMember>
  <gml:featureMember>
    <location>
      <geometry>
        <gml:Point srsName="EPSG:4326">
          <gml:pos>43.47569 -8.19505</gml:pos>
        </gml:Point>
      </geometry>
      <days>
        <day>
          <gml:TimePeriod>
            <gml:begin>
              <gml:TimeInstant>
                <gml:timePosition>2014-05-07T19:41:36+02</gml:timePosition>
              </gml:TimeInstant>
            </gml:begin>
            <gml:end>
              <gml:TimeInstant>
                <gml:timePosition>2014-05-07T20:41:36+02</gml:timePosition>
              </gml:TimeInstant>
            </gml:end>
          </gml:TimePeriod>
        </day>
        ...
      </days>
    </location>
  </gml:featureMember>
</gml:FeatureCollection>
```

If no data exists for some of the requested days but it exists for some other days, behavior is the following:

- For each day without data between the actual time instant (or startDate if present) and the last day with data, a day element with the following structure is returned:

```xml
<day>
  <gml:TimePeriod>
    <gml:begin>
      <gml:TimeInstant>
        <gml:timePosition>yyyy-MM-ddTHH:mm:ss</gml:timePosition>
      </gml:TimeInstant>
    </gml:begin>
    <gml:end>
      <gml:TimeInstant>
        <gml:timePosition>yyyy-MM-ddTHH:mm:ss</gml:timePosition>
      </gml:TimeInstant>
    </gml:end>
  </gml:TimePeriod>
</day>
```
If the operation was invoked specifying `endTime` parameter, for days without data between last day with data and `endTime`, a `day` element with the following structure is returned.

- If the operation was invoked without specifying `endTime` parameter, days without data after last day with data do not show in the response.

When a variable has no data for any time instant inside `day` element time period, `variable` element will be empty. The `day` element structure will be the following:

```
<day>
  <gml:TimePeriod>
    <gml:begin>
      <gml:TimeInstant>
        <gml:timePosition>yyyy-MM-ddTHH:mm:ss</gml:timePosition>
      </gml:TimeInstant>
    </gml:begin>
    <gml:end>
      <gml:TimeInstant>
        <gml:timePosition>yyyy-MM-ddTHH:mm:ss</gml:timePosition>
      </gml:TimeInstant>
    </gml:end>
  </gml:TimePeriod>
  <variables>
    ...
    <variable name="VARIABLE_NAME" ... />
    ...
  </variables>
</day>
```

Case of data absence for a specific temporal interval is described in the dedicated sections for each operation.

- **HTML:**

In the response, only days with data are included. Inside each day, only variables with data for at least one temporal instant are included. In day instants with no data (or not requested), dash is included “-”.

If no data exists for any of the requested days, an exception is returned, except when data was requested for more than one location. In this case, a HTML with specific content for each location is returned, as you may see in the following image:

```
Forecast for latitude=43.47569, longitude=-8.19504:
Currently, there is no available data for the specified time period.
```

```
Forecast for latitude=43.47569, longitude=-8.19505:
Currently, there is no available data for the specified time period.
```

*All hours in local time of Galicia*

- **KML:**

For the KML format, mentioned for HTML format is applied to the `description` element of the KML document. In the case that data was requested for more than one location and there is no data for any (or all) localization,
an exception message is returned for each locality with no data. For example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
  <Document>
    <Placemark id="point_1">
      <ExtendedData>
        <Data name="exceptionCode">
          <value>216</value>
        </Data>
        <Data name="exceptionMessage">
          <value>Specified point falls outside the geographic limits for which there are data.</value>
        </Data>
      </ExtendedData>
      <Point>
        <coordinates>-8.29504,43.47569</coordinates>
      </Point>
    </Placemark>
    <Placemark id="point_2">
      <description>&lt;!DOCTYPE html PUBLIC ...</description>
    </Placemark>
  </Document>
</kml>
```
5.1 Introduction

This operation is used to locate places from a character string. The returned information contains, for each place, some alphanumeric attributes (name, municipality name to which it belongs to, etc.), an identifier that will allow to reference the place in the other operations, and its coordinates.

Type or types of places among which to search can be indicated. If this parameter is not established, the search takes place in all the types of places. Place types currently supported are:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>locality</td>
<td>Population entities</td>
<td>Galicia</td>
</tr>
<tr>
<td>beach</td>
<td>Beaches</td>
<td>Galicia</td>
</tr>
</tbody>
</table>

The matching criteria is as follows: it will return all places whose name starts with, contains or ends with the specified character string, be it uppercase or lowercase. Maximum number of results that can be returned is 1000.

The information about population entities was provided by the SITGA (Galician Territorial Information System, http://sitga.xunta.es) and the localization of the beaches was provided by INTECMAR (Technological Institute for the Control of the Marine Environment of Galicia)

5.2 Parameters

Supported parameters in this operation are:

<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory</th>
<th>Possible values</th>
<th>Default value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>API_KEY</td>
<td>YES</td>
<td>API key</td>
<td>-</td>
<td>See Getting an API key.</td>
</tr>
<tr>
<td>location</td>
<td>YES</td>
<td>Any character string</td>
<td>-</td>
<td>For example “ouren” or “coru”.</td>
</tr>
<tr>
<td>types</td>
<td>NO</td>
<td>Comma separated sequence of the types of places to</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>search (see the table types of places)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lang</td>
<td>NO</td>
<td>‘gl’ ‘es’ ‘en’</td>
<td>‘en’</td>
<td>Language in which texts are returned,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>including texts in the exceptions.</td>
</tr>
<tr>
<td>format</td>
<td>NO</td>
<td>‘gml3’ ‘kml’ ‘application/json’</td>
<td>‘application/json’</td>
<td>Format in which the results are</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>returned.</td>
</tr>
<tr>
<td>exceptionsFormat</td>
<td>NO</td>
<td>‘application/xml’ ‘application/json’</td>
<td>‘application/json’</td>
<td>Format in which the exceptions are returned.</td>
</tr>
</tbody>
</table>
5.3 Results

This operation returns its results following the format specified by the `format` parameter. It’s a set of one or more places. Depending on the specified format, the returned data structure will be:

- A `FeatureCollection`, in the case of GeoJSON, with as many `Feature` objects as coincidences were found.
- A `gml:FeatureCollection` in the case of GML, with as many `gml:FeatureMember` elements as coincidences were found.
- A KML document in the case of KML with as many `<Placemark>` elements as coincidences were found.

Anyway, returned geometries will always be points. The coordinates of these points may be slightly altered once in a while without notice in order to improve their collation or adjust them to the geometries of the grids of the models and optimize the returned forecast, so it’s not guaranteed that they are always the same.

Each place contains the following attributes, presented differently depending on the chosen output format:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Identifier (to use in future searches by id)</td>
</tr>
<tr>
<td>name</td>
<td>Name of the place</td>
</tr>
<tr>
<td>municipality</td>
<td>Name of the municipality which it belongs to</td>
</tr>
<tr>
<td>province</td>
<td>Name of the province which it belongs to</td>
</tr>
<tr>
<td>type</td>
<td>Type of place</td>
</tr>
<tr>
<td>geometry</td>
<td>Place coordinates</td>
</tr>
</tbody>
</table>

For geometries (coordinates) of each location, following representations are used:

<table>
<thead>
<tr>
<th>Format</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GML</td>
<td>It is represented with a <code>gml:point</code> element contained in a <code>geometry</code> element.</td>
</tr>
<tr>
<td>HTML</td>
<td>It is represented as text inside HTML.</td>
</tr>
<tr>
<td>JSON</td>
<td>It is represented with a type point (&quot;type&quot;: &quot;Point&quot;) <code>geometry</code> object.</td>
</tr>
<tr>
<td>KML</td>
<td>It is represented with a <code>Point</code> element.</td>
</tr>
</tbody>
</table>

More in detail for each response format:

- **JSON:**

  It returns a GeoJSON object with the following structure:

  ```json
  {
    "type" : "FeatureCollection",
    "crs" : {
      "type" : "name",
      "properties" : {
        "name" : CRS
      }
    },
    "features" : FEATURES_ARRAY
  }
  ```

  Where:
  - `CRS` indicated the used coordinate system (currently, it is always `urn:ogc:def:crs:OGC:1.3:CRS84`).
  - `FEATURES_ARRAY` is an JSON array (`[element1, element2...]`), where each element is a `feature` object with the following form:

    ```json
    {
      "type" : "Feature",
    }
    ```
"geometry" : {
    "type" : "Point",
    "coordinates" : [X, Y]
},
"properties" : {
    "id" : ID,
    "name" : NAME,
    "municipality" : MUNICIPALITY,
    "province" : PROVINCE,
    "type" : TYPE
}
}

Where:
* X and Y are the location coordinates (longitude and latitude respectively).
* ID, NAME, MUNICIPALITY, PROVINCE and TYPE are the attributes, already mentioned, of the location.

• GML:

It returns a GML document whose root is a gml:featureCollection element with the following structure:

```xml
<gml:FeatureCollection xmlns="http://www.meteogalicia.es/meteosix"
 xmlns:gml="http://www.opengis.net/gml">
 <gml:boundedBy>
   <gml:Envelope srsName=CSR>
     <gml:pos>MIN_Y MIN_X</gml:pos>
     <gml:pos>MAX_Y MAX_X</gml:pos>
   </gml:Envelope>
 </gml:boundedBy>
  FEATURE_MEMBER_1
  FEATURE_MEMBER_2
  ...
</gml:FeatureCollection>
```

Where:
* CRS indicates the used coordinate system (currently, it is always EPSG:4326).
* MIN_Y, MIN_X, MAX_Y and MAX_X are the minimum and maximum coordinates of the Envelope with the set of all locations. Order is latitude, longitude.
* FEATURE_MEMBER_1, FEATURE_MEMBER_2... are elements of gml:featureMember type, which in turn present the following structure:

```xml
<gml:featureMember>
  <location>
    <geometry>
      <gml:Point srsName=CSR>
        <gml:pos>Y X</gml:pos>
      </gml:Point>
    </geometry>
    <id ID</id>
    <name>NAME</name>
    <municipality>MUNICIPALITY</municipality>
    <province>PROVINCE</province>
  </location>
```
Where:

- * Y and X are the location coordinates (latitude and longitude respectively).
- * ID, NAME, MUNICIPALITY, PROVINCE and TYPE are the attributes, already mentioned, of the location.

- **KML:**

It returns a KML document with the following structure:

```xml
<kml xmlns="http://earth.google.com/kml/2.0">
  <Document>
    <Placemark id="point_1"
      <description>NAME - MUNICIPALITY (PROVINCE)</description>
      <Point>
        <coordinates>X,Y</coordinates>
      </Point>
      <ExtendedData>
        <Data name="location_id">
          <value>ID</value>
        </Data>
        <Data name="location_name">
          <value>NAME</value>
        </Data>
        <Data name="location_municipality">
          <value>MUNICIPALITY</value>
        </Data>
        <Data name="location_province">
          <value>PROVINCE</value>
        </Data>
        <Data name="location_type">
          <value>TYPE</value>
        </Data>
      </ExtendedData>
    </Placemark>
    <Placemark id="point_2">
      ...
    </Placemark>
    ...
  </Document>
</kml>
```

Where:

- X and Y are the location coordinates (longitude and latitude respectively).
- ID, NAME, MUNICIPALITY, PROVINCE and TYPE are the attributes, already mentioned, of the location.

Each Placemark has an id attribute whose values are point_1, point_2... and so on, for each Placemark contained in the response.

If no result is found, the following is returned:

- **JSON:**

```json
{
  "crs": {
```
"properties": {
    "name": "urn:ogc:def:crs:OGC:1.3:CRS84"
},
"type": "name"
},
"features": null,
"type": "FeatureCollection"
}

- GML:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<gml:FeatureCollection xmlns="http://meteogalicia.es/meteosix"
xmlns:gml="http://www.opengis.net/gml"/>
```

- KML:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
<Document/>
</kml>
```

### 5.4 Examples

Search for places that contain the text ‘oure’:


Search for places that contain the text ‘lanza’:


Search for beaches that contain the text ‘lanza’:

### 6.1 Introduction

This operation provides information about the numeric forecast for different meteorological and oceanographic variables. Some variables are available in various models of prediction and grids (see section *Numeric forecast models*). Following table show available variables. For each one, models in which it is found, units in which data can be returned and whether or not the data have an associated symbol is indicated.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Models</th>
<th>Values</th>
<th>Units</th>
<th>Default units</th>
<th>Associated symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>sky_state</td>
<td>Sky state</td>
<td>WRF</td>
<td>SUNNY, HIGH_CLOUDS, PARTLY_CLOUDY, OVERCAST, CLOUDY, FOG, SHOWERS, OVERCAST_AND_SHOWERS, INTERMITENT_SNOW, DRIZZLE, RAIN, SNOW, STORMS, MIST, FOG_BANK, MID_CLOUDS, WEAK_RAIN, WEAK_SHOWERS, STORM_THEN_CLOUDY, MELTED_SNOW, MELTED_SNOW, MELTED_SNOW, MELTED_SNOW</td>
<td>- ^1</td>
<td>-</td>
<td>YES</td>
</tr>
<tr>
<td>temperature</td>
<td>Temperature</td>
<td>WRF</td>
<td>Integer</td>
<td>degC degK degF</td>
<td>degC</td>
<td>NO</td>
</tr>
<tr>
<td>precipitation_amount</td>
<td>Accumulated precipitation during the previous hour</td>
<td>WRF</td>
<td>Real number (with 2 decimals)</td>
<td>lm2</td>
<td>lm2</td>
<td>NO</td>
</tr>
<tr>
<td>wind</td>
<td>Wind</td>
<td>WRF</td>
<td>It returns the wind module and direction values, both real numbers with 2 decimals</td>
<td>kmh_deg ms_deg mph_deg kt_deg</td>
<td>kmh_deg</td>
<td>YES</td>
</tr>
</tbody>
</table>

^1 If the units is specified, the unit for this variable must be the empty string (without white spaces)
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Type</th>
<th>Unit(s)</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>relative_humidity</td>
<td>Relative humidity</td>
<td>WRF</td>
<td>perc</td>
<td>perc</td>
</tr>
<tr>
<td>cloud_area_fraction</td>
<td>Cloud coverage</td>
<td>WRF</td>
<td>perc</td>
<td>perc</td>
</tr>
<tr>
<td>air_pressure_at_sea_level</td>
<td>Air pressure at sea level</td>
<td>WRF</td>
<td>Integer</td>
<td>hpa</td>
</tr>
<tr>
<td>snow_level</td>
<td>Snow level</td>
<td>WRF</td>
<td>Integer</td>
<td>m</td>
</tr>
<tr>
<td>sea_water_temperature</td>
<td>Sea water temperature</td>
<td>ROMS, MOHID</td>
<td>Integer</td>
<td>degC, degK, degF</td>
</tr>
<tr>
<td>significative_wave_height</td>
<td>Wave height</td>
<td>WW3, SWAN</td>
<td>Integer</td>
<td>m, ft</td>
</tr>
<tr>
<td>mean_wave_direction</td>
<td>Wave direction</td>
<td>WW3, SWAN</td>
<td>Integer</td>
<td>deg</td>
</tr>
<tr>
<td>relative_peak_period</td>
<td>Wave period</td>
<td>WW3, SWAN</td>
<td>Integer</td>
<td>s</td>
</tr>
<tr>
<td>sea_water_salinity</td>
<td>Sea water salinity</td>
<td>ROMS, MOHID</td>
<td>Integer</td>
<td>psu</td>
</tr>
</tbody>
</table>

Where:

- `degC`: Celsius degrees (°C)
- `degK`: Kelvin degrees (°K)
- `degF`: Fahrenheit degrees (°F)
- `kmh-deg`: kilometers per hour (km/h) – degrees (°)
- `ms-deg`: meters per second (m/s) – degrees (°)
- `mph-deg`: miles per hour (mph) – degrees (°)
- `kt-deg`: knots (kt) – degrees (°)
- `m`: meters (m)
- `ft`: feet (ft)
- `lm2`: liters per squared meter (l/m²)
- `perc`: percentage (%)
- `hpa`: hectopascals (hPa)
- `pa`: pascals (Pa)
- `atm`: atmospheres (atm)
- `s`: seconds (s)
- `deg`: degrees (°)
- `psu`: practical salinity units (psu)
6.2 Temporal frame

This operation has in account the value of hours, minutes and seconds of the *startTime* and *endTime* parameters. Maximum number of days that can be requested in a request is 7. If not *endTime* parameter is indicated, all available data is returned by default (from *actual* instant or from indicated instant in *startTime* parameter) to the maximum of 7 days. In practical way, available data depends on each model and grid (see section *Numeric forecast models*).

6.3 Parameters

Besides those described in the *common parameters table*, the available parameters for this operation are:
<table>
<thead>
<tr>
<th>Name</th>
<th>Mandatory</th>
<th>Possible values</th>
<th>Default value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoAdjustPosition</td>
<td>NO</td>
<td>true or false</td>
<td>true</td>
<td>If the value is true, in the near coastline points and in order to give a more reliable prediction for some variables (air temperature, wind and oceanographic variables) is performed automatically adjusting the position where the data is taken. This way, point where data is taken can be slightly different from the point indicated in the request.</td>
</tr>
<tr>
<td>variables</td>
<td>NO</td>
<td>Comma separated sequence of variable names, among the names specified in the forecast variables table</td>
<td>sky_state, temperature, wind, precipitation_amount</td>
<td></td>
</tr>
<tr>
<td>models</td>
<td>NO</td>
<td>Comma separated sequence of model names. Possible names are: WRF, WW3, SWAN, ROMS and MOHID.</td>
<td>-</td>
<td>If established, it must contain as many elements as variables are indicated by the variables parameter. Each model refers to the variable occupying the same position in that parameter. An element can be an empty string and in that case the highest accuracy model will apply.</td>
</tr>
<tr>
<td>grids</td>
<td>NO</td>
<td>Comma separated sequence of grid names, among the names indicated in the Grid column in the tables: WRF, WW3, SWAN, ROMS and MOHID</td>
<td>-</td>
<td>If established, it must contain as many elements as variables are indicated by the variables parameter. Each grid refers to the variable and model occupying the same position in their respective parameters. An element can be an empty string and in that case the best grid will apply.</td>
</tr>
<tr>
<td>units</td>
<td>NO</td>
<td>Comma separated by names of units among the names indicated in the cells corresponding to the column Units of the forecast variables table</td>
<td>See forecast variables table</td>
<td>If established, it must contain as many elements as variables are indicated in the variables parameter. Each unit refers to the variable occupying the same position in the variables parameter. An element can be an empty string and in that case default unit will apply.</td>
</tr>
</tbody>
</table>
Depending on what is specified in the request, behavior is as follows:

- If variables are not specified in the request, variables that are taken by default are: `sky_state, temperature, wind, precipitation_amount`.

- If the request does not specify models or grids, for each variable will be returned the values correspond to the executions of the best models and grids that are available. It is possible that the best model-grid combination has not data from a given instant, but some other model-grid combinations has. API checks, for each day and variable, how many time instants each model-grid pair has available, and the pair model-grid with more data is used. When multiple model-grid pairs have the same number of data, the best grid is used. For example, the request:

  ```
  ```

  will return:
  - Values of variable `significant_wave_height` of the `Artabro` grid of the `SWAN` model for the first days and the `AtlanticoNorte` grid of the `WW3` model for the last days.
  - Values of variable `sea_water_temperature` of the `Artabro` grid of the `MOHID` model for the first days and the `Galicia` grid of the `ROMS` model for the last days.

- If in the request, variables and grids are specified, but not the grids, previous comment is applied, but considering only the indicated models of the request (not other models). For example, the request:

  ```
  ```

  will return:
  - Values of variable `temperature` of the `Artabro1Km` grid of the `WRF` model for the first days and the `04km` grid of the `WRF` model for the last days.
  - Values of variable `significant_wave_height` of the `AtlanticoNorte` grid of the `WW3` model.
  - Values of variable `sea_water_temperature` of the `Galicia` grid of the `ROMS` model.

- If in the request variables, models and grids are specified, for each variable, values corresponding of the specified models over the specified grids will be returned.

- It is possible to request information for the same variable in multiple grids and models. For example, the request:

  ```
  grids=04km,12km,36km,Galicia,AtlanticoNorte,Galicia&API_KEY=***
  ```

  will return:
  - Values of variable `sky_state` of the `04km` grid of the `WRF` model.

---

2To make this adjustment, information derived from data from the following sources is used:

- Coastline of the Iberian Peninsula: Instituto Hidrográfico de la Armada (Spain): http://www.armada.mde.es/ihm/Aplicaciones/pruebas/costa/LINEA_DECOSTA.zip
- Boundaries of the regions of Spain: Centro Nacional de Información Geográfica (Spain) http://centrodedescargas.cnig.es/CentroDescargas/equipamiento/lineas_limite.zip
- NUT1 Boundaries of Portugal: Direção-Geral do Território (Portugal): http://mapas.igeo.pt/ows/caop/continente/?SERVICE=WFS&VERSION=1.0.0&REQUEST=

6.3. Parameters
- Values of variable \textit{sky\_state} of the 12km grid of the \textit{WRF} model.
- Values of variable \textit{temperature} of the 36km grid of the \textit{WRF} model.
- Values of variable \textit{significative\_wave\_height} of the \textit{Galicia} grid of the \textit{WW3} model.
- Values of variable \textit{significative\_wave\_height} of the \textit{AtlanticoNorte} grid of the \textit{WW3} model.
- Values of variable \textit{sea\_water\_temperature} of the \textit{Galicia} grid of the \textit{ROMS} model.

- It is possible to specify model and grid for some variables and not do for others (that would use the best available, as explained above). To do this, position where model or grid name is retained without indicating any text (empty string). For example, the request:

\begin{verbatim}
\end{verbatim}

will return:

- Values of variable \textit{significative\_wave\_height} of the \textit{AtlanticoNorte} grid of the \textit{WW3} model.
- Values of variable \textit{sea\_water\_temperature} of the \textit{Artabro} grid of the \textit{MOHID} model for the first days and the \textit{Galicia} grid of the \textit{ROMS} model for the last days.

\textbf{Observations:}

- If the request specifies some model, it is mandatory to specify variables.
- If the request specifies some grid, it is mandatory to specify the model that corresponds the grid.
- If the number of variables indicated does not match with the number of model or number of grids (counting empty strings), an exception will be returned.

\subsection*{6.4 Results}

Structure of the returned data for this operation is the indicated in the section \textit{Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations}. If a variable is requested for more than a model-grid combination, each day will contain a \textit{variable} object for each combination. Each \textit{variable} contains the following information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the variable.</td>
</tr>
<tr>
<td>model</td>
<td>The name of the model from which the data come from.</td>
</tr>
<tr>
<td>grid</td>
<td>The name of the grid from which the data come from.</td>
</tr>
<tr>
<td>units</td>
<td>The name of the unit of measure in which the data is returned. Not present for \textit{sky_state} and \textit{wind} variables.</td>
</tr>
<tr>
<td>moduleUnits</td>
<td>The name of the unit of measure in which the wind module is returned. Only present for \textit{wind} variable.</td>
</tr>
<tr>
<td>directionUnits</td>
<td>The name of the unit of measure in which the wind direction is returned. Only present for \textit{wind} variable.</td>
</tr>
<tr>
<td>geometry</td>
<td>The coordinates of the final point from which the values come from. This does not have to match exactly with the point for which the request was made.</td>
</tr>
<tr>
<td>values</td>
<td>It is the set of variable values (variable-model-grid combination) contained within the temporal interval of the \textit{day} object. Each value contains the data indicated in the following table.</td>
</tr>
</tbody>
</table>

Content of the \textit{values} of the variables:
<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeInstant</td>
<td>Hour that corresponds to the value, in format yyyy-MM-ddTHH:mm:ss±XX.</td>
</tr>
<tr>
<td>modelRun</td>
<td>The model execution date from which the data came from, in yyyy-MM-ddTHH:mm:ss±XX format.</td>
</tr>
<tr>
<td>value</td>
<td>The value of the variable. It can be a numeric value or a character string, depending on the specific variable. It is not shown for wind variable.</td>
</tr>
<tr>
<td>moduleValue</td>
<td>The wind module value. It is only shown for wind variable.</td>
</tr>
<tr>
<td>directionValue</td>
<td>The wind direction value. It is only shown for wind variable.</td>
</tr>
<tr>
<td>iconURL</td>
<td>In variables with associated symbol (see variables table) it is included this attribute with a URL from which to obtain the icon that represents the value. Icons are part of MeteoSIX</td>
</tr>
</tbody>
</table>

More in detail for each response format:

- **JSON:**

  The VARIABLES_ARRAY array will contain as many variable objects as variable-model-grid combinations have been requested, (unless there is no data for any of the requested variables, in which case, the noted in Behavior in the absence of data will be followed). Structure of each variable is as follows:

  - **sky_state** variable:
    
    ```json
    {   
      "name": VARIABLE_NAME,  
      "model": MODEL_NAME,  
      "grid": GRID_NAME,  
      "geometry": {  
        "type": "Point",  
        "coordinates": [  
          X,  
          Y  
        ]  
      }  
    "values": VALUES_ARRAY
    }
    ```

  - **wind** variable:
    
    ```json
    {   
      "name": VARIABLE_NAME,  
      "model": MODEL_NAME,  
      "grid": GRID_NAME,  
      "moduleUnits": MODULE_UNIT_NAME,  
      "directionUnits": DIRECTION_UNIT_NAME,  
      "geometry": {  
        "type": "Point",  
        "coordinates": [  
          X,  
          Y  
        ]  
      }  
    "values": VALUES_ARRAY
    }
    ```

  - Remaining variables:
    
    ```json
    {   
      "name": VARIABLE_NAME,  
      "model": MODEL_NAME,  
      "grid": GRID_NAME,  
      "units": UNIT_NAME,  
      "geometry": {  
        "type": "Point",  
        
    ```
"coordinates": [
X,
Y
]
"values": VALUES_ARRAY
}

Where:
- `VARIABLE_NAME` is the name of the variable.
- `MODEL_NAME` is the name of the model from which the values come from.
- `GRID_NAME` is the grid used in the execution from which the values come from.
- `MODULE_UNIT_NAME` is the unit of measure in which the wind data module is returned.
- `DIRECTION_UNIT_NAME` is the unit of measure in which the wind data direction is returned.
- `UNIT_NAME` is the name of the measurement units in which the data is returned.
- X and Y are the point coordinates from which the data come from (longitude and latitude respectively). This point does not have to match exactly with the point for which the request was made.
- `VALUES_ARRAY` is a JSON array that contains the data for that variable-model-grid combination within the temporal interval of the `day` object.

Elements of `VALUES_ARRAY` have the following structure:

- In variables without associated symbol:

  
  {
    "timeInstant": TIME_INSTANT,
    "modelRun": MODEL_RUN,
    "value": VALUE
  }

- In variables with associated symbol, except the `wind` variable:

  
  {
    "timeInstant": TIME_INSTANT,
    "modelRun": MODEL_RUN,
    "value": VALUE,
    "iconURL": ICON_URL
  }

- In the `wind` variable:

  
  {
    "timeInstant": TIME_INSTANT,
    "modelRun": MODEL_RUN,
    "moduleValue": MOD_VALUE,
    "directionValue": DIR_VALUE,
    "iconURL": ICON_URL
  }

Where:
- `TIME_INSTANT` is the hour that corresponds to the data, with format `yyyy-MM ddTHH:mm:ss+XX`.
- `MODEL_RUN` is the start time, in `yyyy-MM-ddTHH:mm:ss+XX` format, of the execution of the model from which the values come from.
- `VALUE` is the value, whose data type depends on the concerned variable (see `forecast variables table`).
– **ICON_URL** is a URL, of public access, that points to the icon that represents the value (provided by MeteoSIX).

– **MOD_VALUE** is the wind module value.

– **DIR_VALUE** is the wind direction value.

In the case of no data available for a specific instant of all that may have data (currently, o’clock hours), format for that instant will be:

– In variables without associated symbol:

```json
{
  "timeInstant": TIME_INSTANT,
  "modelRun": null,
  "value": null
}
```

– In variables with associated symbol, except wind variable:

```json
{
  "timeInstant": TIME_INSTANT,
  "modelRun": null,
  "value": null,
  "iconURL": null
}
```

– In the wind variable:

```json
{
  "timeInstant": TIME_INSTANT,
  "modelRun": null,
  "moduleValue": null,
  "directionValue": null,
  "iconURL": null
}
```

Where:

– **TIME_INSTANT** is the temporal instant.

If a variable object has no data inside the temporal period of the day object, the variable object content will be as follows:

- **name** will contain the variable name.
- **model** will contain the model name if the request specified a model, otherwise it will be null (null).
- **grid** will contain the grid name if the request specified a grid, otherwise it will be null (null).
- **units** will be null (null) in wind case and for the rest of variables will not be present.
- **moduleUnits** will be null (null) in wind case and for the rest of variables will not be present.
- **directionUnits** will be null (null) in wind case and for the rest of variables will not be present.
- **geometry** will always be null (null).
- **values** will always be null (null).

- **GML:**

Each variable element has the following structure:

– **sky_state** variable:
<variable name=VARIABLE_NAME model=MODEL_NAME grid=GRID_NAME>
   <geometry>
      <gml:Point srsName=CRS>
         <gml:pos>Y X</gml:pos>
      </gml:Point>
   </geometry>
   <values>
      VALUE_1
      VALUE_2
      ...
   </values>
</variable>

– wind variable:

<variable name=VARIABLE_NAME model=MODEL_NAME grid=GRID_NAME moduleUnits=MODULE_UNIT_NAME directionUnits=DIRECTION_UNIT_NAME>
   <geometry>
      <gml:Point srsName=CRS>
         <gml:pos>Y X</gml:pos>
      </gml:Point>
   </geometry>
   <values>
      VALUE_1
      VALUE_2
      ...
   </values>
</variable>

– Remaining variables:

<variable name=VARIABLE_NAME model=MODEL_NAME grid=GRID_NAME units=UNIT_NAME>
   <geometry>
      <gml:Point srsName=CRS>
         <gml:pos>Y X</gml:pos>
      </gml:Point>
   </geometry>
   <values>
      VALUE_1
      VALUE_2
      ...
   </values>
</variable>

Where:

– VARIABLE_NAME is the name of the variable.
– MODEL_NAME is the model from which the values come from.
– GRID_NAME is the grid used in the execution from which the values come from.
– CRS indicates the coordinate system used (currently, it is always EPSG:4326).
– MODULE_UNIT_NAME is the unit of measure in which the wind data module is returned.
– DIRECTION_UNIT_NAME is the unit of measure in which the wind data direction is returned.
– UNIT_NAME is the name of the measurement units in which the data is returned.
– Y and X are the coordinates of the point from which the values come from (latitude and longitude respectively). This point does not have to match exactly with the point for which the request was made.

– \textit{VALUE}_1, \textit{VALUE}_2... are values for that variable-model-grid combination inside the temporal interval of the \textit{day} object.

In turn, each of these \textit{VALUE}_1, \textit{VALUE}_2... elements have the following structure:

– In variables without associated symbol:

\begin{verbatim}
<\textit{hourValue}>
 <\textit{gml:TimeInstant}>
  <\textit{gml:timePosition}>TIME_INSTANT</gml:timePosition>
 </gml:TimeInstant>
 <\textit{modelRun}>
  <\textit{gml:TimeInstant}>
   <\textit{gml:timePosition}>MODEL_RUN</gml:timePosition>
  </gml:TimeInstant>
 </modelRun>
 <value>VALUE</value>
</\textit{hourValue}>
\end{verbatim}

– In variables with associated symbol, except \textit{wind} variable:

\begin{verbatim}
<\textit{hourValue}>
 <\textit{gml:TimeInstant}>
  <\textit{gml:timePosition}>TIME_INSTANT</gml:timePosition>
 </gml:TimeInstant>
 <\textit{modelRun}>
  <\textit{gml:TimeInstant}>
   <\textit{gml:timePosition}>MODEL_RUN</gml:timePosition>
  </gml:TimeInstant>
 </modelRun>
 <value>VALUE</value>
 <iconURL>ICON_URL</iconURL>
</\textit{hourValue}>
\end{verbatim}

– In the \textit{wind} variable:

\begin{verbatim}
<\textit{hourValue}>
 <\textit{gml:TimeInstant}>
  <\textit{gml:timePosition}>TIME_INSTANT</gml:timePosition>
 </gml:TimeInstant>
 <\textit{modelRun}>
  <\textit{gml:TimeInstant}>
   <\textit{gml:timePosition}>MODEL_RUN</gml:timePosition>
  </gml:TimeInstant>
 </modelRun>
 <moduleValue>MOD_VALUE</moduleValue>
 <directionValue>DIR_VALUE</directionValue>
 <iconURL>ICON_URL</iconURL>
</\textit{hourValue}>
\end{verbatim}

Where:

– \textit{TIME_INSTANT} is the hour that corresponds to the data, with format \texttt{yyyyMMdd\thh:mm:ss+XX}.

– \textit{MODEL_RUN} is the start hour of the execution from which the data come from, with format \texttt{yyyyMMdd\thh:mm:ss+XX}.

– \textit{VALUE} is the value, whose data type depends on the concerned variable (see \textit{forecast variables table}).
– **ICON_URL** is a URL, of public access, that points to the icon that represents the value (provided by MeteoSIX).

– **MOD_VALUE** is the wind module value.

– **DIR_VALUE** is the wind direction value.

In the case of no data available for a specific instant of all that may have data (currently, o’clock hours), format for that instant will be:

– In variables without associated symbol:

```xml
<hourValue>
  <gml:TimeInstant>
    <gml:timePosition>TIME_INSTANT</gml:timePosition>
  </gml:TimeInstant>
  <modelRun/>
  <value/>
</hourValue>
```

– In variables with associated symbol, except *wind* variable:

```xml
<hourValue>
  <gml:TimeInstant>
    <gml:timePosition>TIME_INSTANT</gml:timePosition>
  </gml:TimeInstant>
  <modelRun/>
  <value/>
  <iconURL/>
</hourValue>
```

– In the *wind* variable:

```xml
<hourValue>
  <gml:TimeInstant>
    <gml:timePosition>TIME_INSTANT</gml:timePosition>
  </gml:TimeInstant>
  <modelRun/>
  <moduleValue/>
  <directionValue/>
  <iconURL/>
</hourValue>
```

Where:

– **TIME_INSTANT** is the temporal instant for which there is no data.

If a variable has no data inside the temporal interval of the *day* element in which it is contained, variable will contain neither *geometry* or *values* elements, and the attributes:

- **name** will contain the variable name.
- **model** will contain the model name if the request specified a model, otherwise it will not be shown.
- **grid** will contain the grid name if the request specified a grid, otherwise it will not be shown.
- **units** will not show up.
- **moduleUnits** will not show up.
- **directionUnits** will not show up.

**HTML:**
Besides the information returned in the other formats, HTML includes daily summaries for each variable. If a variable does not have values for a time instant (either because there are no data or because it is a past value) a dash - is shown. Thus, for example, the response for a request asking for the wind and sky_state variables could be:

Forecast for latitude=44, longitude=-8

<table>
<thead>
<tr>
<th>Summary</th>
<th>0h</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>7h</th>
<th>8h</th>
<th>9h</th>
<th>10h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sky state</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wind</td>
<td>Min 15 Max 53</td>
<td>-</td>
<td>-</td>
<td>19</td>
<td>19</td>
<td>18</td>
<td>17</td>
<td>15</td>
<td>17</td>
<td>17</td>
<td>16</td>
</tr>
</tbody>
</table>

Tuesday April 30th, 2013

<table>
<thead>
<tr>
<th>Summary</th>
<th>0h</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>7h</th>
<th>8h</th>
<th>9h</th>
<th>10h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sky state</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wind</td>
<td>Min 46 Max 51</td>
<td>50</td>
<td>49</td>
<td>51</td>
<td>49</td>
<td>50</td>
<td>50</td>
<td>49</td>
<td>48</td>
<td>48</td>
<td>49</td>
</tr>
</tbody>
</table>

Wednesday May 1st, 2013

<table>
<thead>
<tr>
<th>Summary</th>
<th>0h</th>
<th>1h</th>
<th>2h</th>
<th>3h</th>
<th>4h</th>
<th>5h</th>
<th>6h</th>
<th>7h</th>
<th>8h</th>
<th>9h</th>
<th>10h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sky state</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wind</td>
<td>Min 35 Max 51</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

- KML

The response is a KML document with the structure described in the Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations section.

6.5 Examples

Obtain default data for a specific point:


Get the same standard forecast data for a specific point in HTML format:


Get, for Loureda (Arteixo), place with id 42917, forecast for the temperature variable in the 4 and 12 km grids, and for the sky state in the 36 km grid, all of them of the WRF model and in GML format:

http://servizos.meteogalicia.es/apiv3/getNumericForecastInfo?locationIds=42917&variables=temperature,temperature,sky_state&models=WRF,WRF,WRF&grids=04km,12km,36km&format=gml3&API_KEY=***

Obtain previous data in KML format and with the temperature values in Kelvin degrees:
Obtain previous data in HTML format and for a specific time interval:

&variables=temperature,temperature,sky_state&
models=WRF,WRF,WRF&grids=04km,12km,36km&format=text/html&
units=degk,degk,&startTime=2013-03-07T15:00:00&
endTime=2013-03-08T08:00:00&API_KEY=***

**Note**: for executing this last request it is necessary to adjust *startTime* and *endTime* parameters to current values.
/getTidesInfo operation

7.1 Introduction

This operation offers information about tides for points in the Galician coast, or close to it. For giving information, these 15 ports are taking as base:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A Coruña</td>
</tr>
<tr>
<td>3</td>
<td>Vigo</td>
</tr>
<tr>
<td>4</td>
<td>Vilagarcía</td>
</tr>
<tr>
<td>6</td>
<td>Ría de Foz</td>
</tr>
<tr>
<td>7</td>
<td>Corcubión</td>
</tr>
<tr>
<td>8</td>
<td>Ría de Camariñas</td>
</tr>
<tr>
<td>9</td>
<td>Ría de Corme</td>
</tr>
<tr>
<td>10</td>
<td>A Guarda</td>
</tr>
<tr>
<td>11</td>
<td>Ribeira</td>
</tr>
<tr>
<td>12</td>
<td>Muros</td>
</tr>
<tr>
<td>13</td>
<td>Pontevedra</td>
</tr>
<tr>
<td>14</td>
<td>Ferrol puerto exterior</td>
</tr>
<tr>
<td>15</td>
<td>Marín</td>
</tr>
<tr>
<td>16</td>
<td>Ferrol</td>
</tr>
</tbody>
</table>

The available information for those ports is the tides daily summary: hour and tides height for each high tides and low tides of the day. Each of these 15 ports is associated, by proximity, to a reference port among these eight ports:

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A Coruña</td>
</tr>
<tr>
<td>2</td>
<td>Xixón</td>
</tr>
<tr>
<td>3</td>
<td>Vigo</td>
</tr>
<tr>
<td>4</td>
<td>Vilagarcía</td>
</tr>
<tr>
<td>10</td>
<td>A Guarda</td>
</tr>
<tr>
<td>14</td>
<td>Ferrol puerto exterior</td>
</tr>
<tr>
<td>15</td>
<td>Marín</td>
</tr>
<tr>
<td>16</td>
<td>Ferrol</td>
</tr>
</tbody>
</table>

For these reference ports there is the tides height estimation for every 30 minutes. In the following figure, you can see the location of the ports. Reference ports are marked in red.
When this operation is called for a point \( p \) (see next figure) close to the Galician coast (in the figure, within the yellow area), it will be returned, on the one hand, the daily summary of the nearest port to \( p \) among the considered 15, port 1, Muros in this case and also tides height values every 30 minutes, corresponding to the reference port closest to port 1, port 2, Vilagarcía in this case.
7.2 Temporal frame

The maximum number of days that can be requested in a petition is 30. The farthest instant for which data is returned is 60 days after the actual day and you can not request data prior to current day. If the temporal range is not specified in the request, data is returned for 5 days from today. If only the start time is specified, data is returned for 5 days from the date of the initial instant (this one included). If only the end time is specified, data is returned from today until the end time (both included), provided that the duration does not exceed the maximum allowed. See further observations on the parameters startTime and endTime.

7.3 Parameters

The parameters supported by this operation are indicated in the: ref: table <ref-common_parameters> common parameters. In the case of startTime and endTime, they will have to be indicated with their full format (yyyy-MM-ddTHH:mm:ss) but only the year, month and day specified will be taken into account (ignoring the part THH:mm:ss).
7.4 Results

Data structure returned by this operation is the indicated in the section *Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations*. Moreover, information about the port, *port* and reference port, *referencePort*, is given for each location. For each port, the following information is returned:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Identifier of the port.</td>
</tr>
<tr>
<td>units</td>
<td>Name of the place where the port is located.</td>
</tr>
<tr>
<td>units</td>
<td>Coordinates of the port.</td>
</tr>
</tbody>
</table>

Within each day, variable content is the following:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Name of the variable, in this case <em>tides</em> 1.</td>
</tr>
<tr>
<td>units</td>
<td>Units used when showing data.</td>
</tr>
<tr>
<td>summary</td>
<td>Information about high tides and low tides (see <em>summary</em>).</td>
</tr>
<tr>
<td>values</td>
<td>The tides value each half an hour (see <em>tides values table</em>).</td>
</tr>
</tbody>
</table>

*summary* is the set of high tides and low tides of the day. Each element has the following content:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Identifier of the data.</td>
</tr>
<tr>
<td>state</td>
<td>State of the tides. It can take the values of <em>High tides</em> for high tides and <em>Low tides</em> for low tides.</td>
</tr>
<tr>
<td>TimeInstant</td>
<td>Temporal instant when high tides or low tides takes place, as appropriate.</td>
</tr>
<tr>
<td>height</td>
<td>Tides height, in meters.</td>
</tr>
</tbody>
</table>

*values* is the data set of tides height every half an hour. Each element has the following content:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeInstant</td>
<td>Temporal instant when data is given.</td>
</tr>
<tr>
<td>height</td>
<td>Tides height, in meters.</td>
</tr>
</tbody>
</table>

More in detail for each return format:

- **JSON**:

  It returns a GeoJSON object, where *properties* object of each *feature* has the following structure:

  
  "properties": {
    "id": ID,
    "name": NAME,
    "municipality": MUNICIPALITY,
    "province": PROVINCE,
    "type": TYPE,
    "port": {
      "id": PORT_IDENTIFIER,
      "name": PORT_NAME,
      "geometry": {
        "type": "Point",
        "coordinates": [
          PORT_X,
          PORT_Y
        ]
      }
    },
    "referencePort": {
      "id": REFERENCE_PORT_IDENTIFIER,
      "name": REFERENCE_PORT_NAME,
      "geometry": {
        "type": "Point",
        "coordinates": [
          REFERENCE_PORT_X,
          REFERENCE_PORT_Y
        ]
      }
    }
  },

  1In v2 version of the API, this variable was called *tides_info*
"name": REFERENCE_PORT_NAME,
"geometry": {
  "type": "Point",
  "coordinates": [
    REFERENCE_PORT_X,
    REFERENCE_PORT_Y
  ]
},
"days": [
  ...
  {
    "timePeriod": {
      "begin": {
        "timeInstant": BEGIN_TIME_PERIOD
      },
      "end": {
        "timeInstant": END_TIME_PERIOD
      }
    },
    "variables": [  
      {
        "name": "tides",
        "units": "m",
        "summary": [  
          ...
          {
            "id": VALUE_IDENTIFIER,
            "state": TIDES_STATE,
            "timeInstant": LOW_OR_HIGH_TIDES_TIME_INSTANT,
            "height": LOW_OR_HIGH_TIDES_TIDES_HEIGHT,
          },
          ...
        ],
        "values": [  
          ...
          {
            "timeInstant": TIME_INSTANCE,
            "height": TIDES_HEIGHT
          },
          ...
        ]
      }
    ]
  }
...
]

Where:

- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only included when operation is invoked with **locationIds** parameter.

- **PORT_IDENTIFIER** is the port identifier where high and low tides data is taken.

- **PORT_NAME** is the port name where high and low tides data is taken.

- **PORT_X** and **PORT_Y** are the port coordinates where high and low tides data is taken (longitude and latitude respectively).
– *REFERENCE_PORT_IDENTIFIER* is the reference port identifier where height tides values data is taken every 30 minutes.

– *REFERENCE_PORT_NAME* is the reference port name where height tides values data is taken every 30 minutes.

– *REFERENCE_PORT_X* and *REFERENCE_PORT_Y* are the reference port coordinates where height tides values data is taken every 30 minutes (latitude and longitude respectively).

– *BEGIN_TIME_PERIOD* is, in this case, the first time instant of the day.

– *END_TIME_PERIOD* is, in this case, the last time instant of the day.

– *VALUE_IDENTIFIER* is a data identifier. It takes consecutive values from 1 (1,2,3...).

– *TIDES_STATE* indicates if it is high tides or low tides, *High tides* and *Low tides* respectively.

– *LOW_OR_HIGH_TIDES_TIME_INSTANT* indicates the instant when a high tides or a low tides takes place.

– *LOW_OR_HIGH_TIDES_TIDES_HEIGHT* indicates the height a high tides or a low tides.

– *TIME_INSTANT* indicates the instant when a tides height value is given.

– *TIDES_HEIGHT* indicates the tides height for a given instant.

If no data available for a given day, *units* attribute will be omitted.

• **GML:**

It returns a GML document where each *location* element has the following structure:

```xml
<location id=ID name=NAME municipality=MUNICIPALITY province=PROVINCE type=TYPE>
  <geometry>
    <gml:Point srsName=CRS>
      <gml:pos>Y X</gml:pos>
    </gml:Point>
  </geometry>

  <port>
    <id>PORT_IDENTIFIER</id>
    <name>PORT_NAME</name>
    <gml:Point srsName=CRS>
      <gml:pos>PORT_Y PORT_X</gml:pos>
    </gml:Point>
  </port>

  <referencePort>
    <id>REFERENCE_PORT_IDENTIFIER</id>
    <name>REFERENCE_PORT_NAME</name>
    <gml:Point srsName=CRS>
      <gml:pos>REFERENCE_PORT_Y REFERENCE_PORT_X</gml:pos>
    </gml:Point>
  </referencePort>

  <days>...
   <day>
    <gml:TimePeriod>
      <gml:begin>
        <gml:TimeInstant>
          <gml:timePosition>BEGIN_TIME_PERIOD</gml:timePosition>
        </gml:TimeInstant>
      </gml:begin>

      <gml:end>
        <gml:TimeInstant>
        </gml:end>
    </gml:TimePeriod>
   </day>
  </days>
</location>
```
Where:

- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only included when operation is invoked with `locationIds` parameter.

- **CRS** is the used coordinate system (currently, it is always EPSG:4326).

- **Y** and **X** are the location coordinates (latitude and longitude respectively).

- **PORT_IDENTIFIER** is the port identifier where high and low tides data is taken.

- **PORT_NAME** is the port name where high and low tides data is taken.

- **PORT_Y** and **PORT_X** are the port coordinates for where high tides and low tides data is taken (latitude and longitude respectively).

- **REFERENCE_PORT_IDENTIFIER** is the reference port identifier where height tides values data is taken every 30 minutes.

- **REFERENCE_PORT_NAME** is the reference port name where height tides values data is taken every 30 minutes.

- **REFERENCE_PORT_Y** and **REFERENCE_PORT_X** are the reference port coordinates for where tides height data is taken every 30 minutes (latitude and longitude respectively).

- **BEGIN_TIME_PERIOD** is, in this case, the first time instant of the day.
– *END_TIME_PERIOD* is, in this case, the last time instant of the day.
– *VALUE_IDENTIFIER* is a data identifier. It takes consecutive values from 1 (1,2,3...).
– *TIDES_STATE* indicates if it is high tides or low tides, *High tides* and *Low tides* respectively.
– *LOW_OR_HIGH_TIDES_TIME_INSTANT* indicates the instant when a high tides or a low tides takes place.
– *LOW_OR_HIGH_TIDES_TIDES_HEIGHT* indicates the height a high tides or a low tides.
– *TIME_INSTANT* indicates the instant when a tides height value is given.
– *TIDES_HEIGHT* indicates the tides height for a given instant.

If no data available for a given day, *units* attribute will be omitted.

• **HTML:**

The response is a HTML document, with the CSS associated, it looks this way:

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>Latitude</th>
<th>Longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Ferrol</td>
<td>43.4667</td>
<td>-8.2333</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>State</th>
<th>Height</th>
<th>Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low tides</td>
<td>0.9 m</td>
<td>01:13</td>
</tr>
<tr>
<td>High tides</td>
<td>4.3 m</td>
<td>07:29</td>
</tr>
<tr>
<td>Low tides</td>
<td>1.1 m</td>
<td>13:31</td>
</tr>
<tr>
<td>High tides</td>
<td>4.4 m</td>
<td>19:51</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>Latitude</th>
<th>Longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A Coruña</td>
<td>43.35</td>
<td>-8.3833</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Time</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00</td>
<td>1.266 m</td>
</tr>
<tr>
<td>00:30</td>
<td>1.049 m</td>
</tr>
<tr>
<td>01:00</td>
<td>0.944 m</td>
</tr>
<tr>
<td>01:30</td>
<td>0.953 m</td>
</tr>
<tr>
<td>02:00</td>
<td>1.07 m</td>
</tr>
<tr>
<td>02:30</td>
<td>1.284 m</td>
</tr>
<tr>
<td>03:00</td>
<td>1.579 m</td>
</tr>
<tr>
<td>04:30</td>
<td>1.935 m</td>
</tr>
<tr>
<td>04:00</td>
<td>2.331 m</td>
</tr>
<tr>
<td>04</td>
<td>2.74</td>
</tr>
</tbody>
</table>

**Tuesday April 30th, 2013**

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>Latitude</th>
<th>Longitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Ferrol</td>
<td>43.4667</td>
<td>-8.2333</td>
</tr>
</tbody>
</table>

• **KML:**

The response is a KML document with the indicated structure in the *Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations* section.

### 7.5 Examples

Get tides data for the nearest port to a given point:

8.1 Introduction

This operation provides information for a set of days about the sunrise and sunset hours and noon (highest point) and total light hours. All this information is available for anywhere in the world.

8.2 Temporal frame

The maximum number of days that can be requested is 365. Farthest instant for which data are returned is 365 days after the current day and data can not be request prior to the current. If the time range is specified in the request, data returned for five days from today. If only the start time is specified, data is returned 5 days from the date of the initial time (this included). If only the end time is specified, data is returned from today until the end time (both included), provided that the duration does not exceed the maximum allowed. See further observations on the parameters `startTime` and `endTime`.

8.3 Parameters

Parameters supported by this operation are the same as pointed in `common parameters table`. In the case of `startTime` and `endTime` parameters, they must be indicated with their full format (yyyy-MM-ddTHH:mm:ss) but only year, month and day will be taken in account (THH:mm:ss will be ignored).

8.4 Results

The structure of the returned results is the one pointed in `Structure of the response for /getNumericForecastInfo, /getTidesInfo and /getSolarInfo operations`. Each day, variable content will be as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the variable, <code>solar</code> in this case.</td>
</tr>
<tr>
<td>sunrise</td>
<td>Time instant for the sunrise.</td>
</tr>
<tr>
<td>midday</td>
<td>Time instant for the midday.</td>
</tr>
<tr>
<td>sunset</td>
<td>Time instant for the sunset.</td>
</tr>
<tr>
<td>duration</td>
<td>Duration of solar light</td>
</tr>
</tbody>
</table>

More in detail for each return format:
• **JSON:**

It returns a GeoJSON object where properties object of each feature has the following structure:

```
"properties": {
  "id": ID,
  "name": NAME,
  "municipality": MUNICIPALITY,
  "province": PROVINCE,
  "type": TYPE,
  "days": [
    ...
    { "timePeriod": {
      "begin": {
        "timeInstant": BEGIN_TIME_PERIOD
      },
      "end": {
        "timeInstant": END_TIME_PERIOD
      }
    },
    "variables": [
      {
        "name": "solar",
        "sunrise": SUNRISE_TIME_INSTANT,
        "midday": MIDDAY_TIME_INSTANT,
        "sunset": SUNSET_TIME_INSTANT,
        "duration": DURATION
      }
    ]
  ]
}
```

Where:

- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only included when the operation is invoked with locationIds parameter.

- **BEGIN_TIME_PERIOD** is, in this case, the first time instant of the day.

- **END_TIME_PERIOD** is, in this case, the last time instant of the day.

- **SUNRISE_TIME_INSTANT** is time instant for the sunrise for that day.

- **MIDDAY_TIME_INSTANT** is time instant for the midday for that day.

- **SUNSET_TIME_INSTANT** is time instant for the sunset for that day.

- **DURATION** is the duration of the day (sun hours), with format Xh Xm, for example 9h 12m.

• **GML:**

It returns a GML document where each location element has the following structure:

```
<Location id=ID name=NAME municipality=MUNICIPALITY province=PROVINCE type=TYPE>
  <geometry>
    <gml:Point srsName=CRS>
      <gml:pos>Y X</gml:pos>
    </gml:Point>
  </geometry>
</Location>
```
...<day>
    <gml:TimePeriod>
        <gml:begin>
            <gml:TimeInstant>
                <gml:timePosition>BEGIN_TIME_PERIOD</gml:timePosition>
            </gml:TimeInstant>
        </gml:begin>
        <gml:end>
            <gml:TimeInstant>
                <gml:timePosition>END_TIME_PERIOD</gml:timePosition>
            </gml:TimeInstant>
        </gml:end>
    </gml:TimePeriod>
    <variables>
        <variable name="solar">
            <solar>
                <gml:TimeInstant>
                    <gml:timePosition>SUNRISE_TIME_INSTANT</gml:timePosition>
                </gml:TimeInstant>
            </solar>
            <midday>
                <gml:TimeInstant>
                    <gml:timePosition>MIDDAY_TIME_INSTANT</gml:timePosition>
                </gml:TimeInstant>
            </midday>
            <sunset>
                <gml:TimeInstant>
                    <gml:timePosition>SUNSET_TIME_INSTANT</gml:timePosition>
                </gml:TimeInstant>
            </sunset>
        </variable>
    </variables>
</day>
</days>
</location>

Where:

- **ID, NAME, MUNICIPALITY, PROVINCE** and **TYPE** are location attributes. They are only included when the operation is invoked with `locationIds` parameter.
- **CRS** is the used coordinate system (currently, it is always EPSG:4326).
- **Y** and **X** are the location coordinates (latitude and longitude respectively).
- **BEGIN_TIME_PERIOD** is, in this case, the first time instant of the day.
- **END_TIME_PERIOD** is, in this case, the last time instant of the day.
- **SUNRISE_TIME_INSTANT** is time instant for the sunrise for that day.
- **MIDDAY_TIME_INSTANT** is time instant for the midday for that day.
- **SUNSET_TIME_INSTANT** is time instant for the sunset for that day.
- **DURATION** is the duration of the day (sun hours), with format Xh Xm, for example 9h 12m.

• **HTML:**

The response is a HTML document, with the CSS styles sheet, and it looks as follows:
• **KML:**

The response is a KML document with the structure specified in the *Structure of the response for /getNumeric-ForecastInfo, /getTidesInfo and /getSolarInfo operations* section.

### 8.5 Observations

This operation returns values for the exact point included in the request. For that reason, the *variable* elements do not include the *geometry* sub-element in any of the output formats.

### 8.6 Examples

Obtain default data for a specific point:

Obtain data for three specific days, with HTML format:

http://servizos.meteogalicia.es/apiv3/getSolarInfo?
coords=-8.350573861318628,43.3697102138535&format=text/html&
startTime=2014-03-07T00:00:00&
endTime=2014-03-09T00:00:00&API_KEY=***

Note: to run this last request is necessary to adjust the parameters startTime and endTime to current values.
Chapter 9

Exceptions

API exception types that can arise according to the operations are listed next.

**Common exceptions to all operations**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>Internal error in the application or in some data server</td>
</tr>
<tr>
<td>001</td>
<td>Response into the maximum response time cannot be obtained</td>
</tr>
<tr>
<td>002</td>
<td>Some of the indicated parameters don’t exist or are mispelled</td>
</tr>
<tr>
<td>003</td>
<td>Some of the indicated parameters are duplicated</td>
</tr>
<tr>
<td>004</td>
<td>Some of the indicated models are empty</td>
</tr>
<tr>
<td>005</td>
<td>The API_KEY parameter is not found</td>
</tr>
<tr>
<td>006</td>
<td>The API_KEY is invalid</td>
</tr>
<tr>
<td>007</td>
<td>The indicated language (lang parameter) does not exist, it’s misspelled or it’s not supported</td>
</tr>
<tr>
<td>008</td>
<td>The format parameter indicates an unsupported format</td>
</tr>
<tr>
<td>009</td>
<td>The exceptionsFormat parameter indicates an unsupported format</td>
</tr>
<tr>
<td>010</td>
<td>The indicated coordinate system (CRS parameter) is not supported or it’s invalid</td>
</tr>
<tr>
<td>011</td>
<td>The indicated style (style parameter) is not supported or it’s invalid</td>
</tr>
</tbody>
</table>

**Exceptions that can arise in the operation /findPlaces**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>The location parameter value was not specified</td>
</tr>
<tr>
<td>101</td>
<td>The types parameter is invalid</td>
</tr>
<tr>
<td>102</td>
<td>The types parameter contains some invalid values</td>
</tr>
</tbody>
</table>

**Exceptions that can arise in the operations /getNumericForecastInfo, /getTidesInfo and /getSolarInfo**
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Neither <code>locationIds</code> nor <code>coords</code> parameters were specified</td>
</tr>
<tr>
<td>201</td>
<td>The <code>locationIds</code> and <code>coords</code> parameters have been included at the same time</td>
</tr>
<tr>
<td>202</td>
<td>More points than those allowed were indicated (there is a limit of 20 points currently)</td>
</tr>
<tr>
<td>203</td>
<td>Some of the values for <code>coords</code> parameter are empty</td>
</tr>
<tr>
<td>204</td>
<td>Some of the values for <code>locationIds</code> parameter are empty</td>
</tr>
<tr>
<td>205</td>
<td>The <code>types</code> parameter format is invalid</td>
</tr>
<tr>
<td>206</td>
<td>Some of the values for <code>coords</code> parameter are invalid or misspelled</td>
</tr>
<tr>
<td>207</td>
<td>The <code>types</code> parameter format is invalid</td>
</tr>
<tr>
<td>208</td>
<td>Some of the values for <code>locationIds</code> parameter are invalid or misspelled</td>
</tr>
<tr>
<td>209</td>
<td>The format of some of the indicated dates is invalid (it must be <code>yyyy-MM-ddTHH:mm:ss</code>)</td>
</tr>
<tr>
<td>210</td>
<td>The <code>tz</code> parameter value is invalid</td>
</tr>
<tr>
<td>211</td>
<td>No place with the indicated location id could be found</td>
</tr>
<tr>
<td>212</td>
<td>The indicated start time is later than the end time</td>
</tr>
<tr>
<td>213</td>
<td>The indicated start time is previous than the current day</td>
</tr>
<tr>
<td>214</td>
<td>The indicated end time is previous than the current day</td>
</tr>
<tr>
<td>215</td>
<td>The indicated temporal interval is too big</td>
</tr>
<tr>
<td>216</td>
<td>There are no data for the indicated temporal interval</td>
</tr>
<tr>
<td>217</td>
<td>Indicated point or points fall outside the geographic limits for which there are data</td>
</tr>
</tbody>
</table>

Exceptions that can arise in the operation `/getNumericForecastInfo`

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>The <code>variables</code> parameter value is invalid</td>
</tr>
<tr>
<td>301</td>
<td>The <code>models</code> parameter value is invalid</td>
</tr>
<tr>
<td>302</td>
<td>The <code>grids</code> parameter value is invalid</td>
</tr>
<tr>
<td>303</td>
<td>The <code>units</code> parameter value is invalid</td>
</tr>
<tr>
<td>304</td>
<td>The <code>autoAdjustPosition</code> parameter value is invalid</td>
</tr>
<tr>
<td>305</td>
<td>The value of the indicated units for the <code>wind</code> variable is invalid</td>
</tr>
<tr>
<td>306</td>
<td>The number of variables is not equal to the number of models</td>
</tr>
<tr>
<td>307</td>
<td>The number of variables is not equal to the number of grids</td>
</tr>
<tr>
<td>308</td>
<td>The number of variables is not equal to the number of units</td>
</tr>
<tr>
<td>309</td>
<td>Some of the indicated variables don’t exist</td>
</tr>
<tr>
<td>310</td>
<td>Some of the indicated models don’t exist</td>
</tr>
<tr>
<td>311</td>
<td>Some of the indicated grids don’t exist</td>
</tr>
<tr>
<td>312</td>
<td>Some of the indicated units don’t exist</td>
</tr>
<tr>
<td>313</td>
<td>There are repeated variables. In this case, different models for each one must be indicated. If a pair variable-model is repeated, each one must indicate a different grid.</td>
</tr>
<tr>
<td>314</td>
<td>Values for <code>grids</code> parameter were indicated without model</td>
</tr>
<tr>
<td>315</td>
<td>One of the indicated models is not applicable to that variable</td>
</tr>
<tr>
<td>316</td>
<td>One of the indicated grids is not applicable to that variable</td>
</tr>
<tr>
<td>317</td>
<td>One of the indicated units is not applicable to that variable</td>
</tr>
<tr>
<td>318</td>
<td><code>endTime</code> parameter indicates a previous instant of current time. In that case, <code>startTime</code> parameter must be included</td>
</tr>
<tr>
<td>319</td>
<td>The indicated temporal interval is too small</td>
</tr>
</tbody>
</table>

Exceptions that can arise in the operation `/getTidesInfo`

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>The port information to retrieve tides data is not available</td>
</tr>
</tbody>
</table>

Apendixes:
A1. What’s new in version v3

Note: the version v3 of the API is not compatible with prior versions.

These are the main news of the v3 of the API:

- **Support for multiple points requests**
  In some operations, users can request information for more than one point.

- **New ports in tides information**
  New ports and reference ports have been added since version v2. See /getTidesInfo operation.

- **Improvements in coastline forecast**
  Added autoAdjustPosition option, enabled by default, it makes that sometimes for points near coastline and certain variables, instead of retrieving data for point indicated in the request, it is retrieve for a certain near point where it is supposed that the forecast is more reliable. See operation /getNumericForecastInfo parameters.

- **Changes in parameters**
  The coord and locationId parameters are now called coords and locationIds respectively. See common parameters table. Also, request with not valid, repeated or empty parameters are not allowed now.

- **Error correction**
  Some errors have been fixed, for example, coordinate order for GML format (right order is latitude, longitude), scenarios where data was not retrieved even when it was available...

- **Changes in the behavior on data absence**
  When no all data is available, some cases where v2 returned an exception, now all available data is returned. For example, if there is no data for a certain day or variable, data is returned for those variables or days that have data. See Behavior in the absence of data.

- **Changes in exceptions**
  New exception types have been added, and some of the prior versions exceptions were removed. In addition, exception codes have changed. See Exceptions.

- **Changes in data structure**
  Adjustments have been made in all formats. Among others, now information about ports and reference ports are included only once for each location, instead of including them inside each day. In addition, values for start and end instants of each day now do not correspond to the first and last time for which data is available but indicate the interval over which data is being returned (outside it could also be data) and no need to match exactly o’clock hours, information about coordinate system was repositioned, model execution hour is indicated for each value... In GML, changes were made to conform version 3.2.1.
• Adjustments in temporal range management

Among others, when no start time is indicated, data is retrieved from the current instant (in prior versions of the API, for HTML format, data was retrieved from the start of the current day). See Temporal interval.
A2. About this document

Change list

- 20/05/2014
  
  Initial version of the v3 API guide.
The accepted time zones (parameter tz) are the following:

- Europe/Madrid
- UTC
- Europe/Lisbon
- Europe/London
- Europe/Paris
- Etc/GMT+12
- Etc/GMT+11
- Pacific/Midway
- Pacific/Niue
- Pacific/Pago_Pago
- Pacific/Samoa
- US/Samoa
- America/Adak
- America/Atka
- Etc/GMT+10
- HST
- Pacific/Fakaofo
- Pacific/Honolulu
- Pacific/Johnston
- Pacific/Rarotonga
- Pacific/Tahiti
- SystemV/HST10
- US/Aleutian
- US/Hawaii
- Pacific/Marquesas
• AST
• America/Anchorage
• America/Juneau
• America/Nome
• America/Sitka
• America/Yakutat
• Etc/GMT+9
• Pacific/Gambier
• SystemV/YST9
• SystemV/YST9YDT
• US/Alaska
• America/Dawson
• America/Ensenada
• America/Los_Angeles
• America/Metlakatla
• America/Santa_Isabel
• America/Tijuana
• America/Vancouver
• America/Whitehorse
• Canada/Pacific
• Canada/Yukon
• Etc/GMT+8
• Mexico/BajaNorte
• PST
• PST8PDT
• Pacific/Pitcairn
• SystemV/PST8
• SystemV/PST8PDT
• US/Pacific
• US/Pacific-New
• America/Boise
• America/Cambridge_Bay
• America/Chihuahua
• America/Dawson_Creek
• America/Denver
• America/Edmonton
• America/Hermosillo
• America/Inuvik
• America/Mazatlan
• America/Ojinaga
• America/Phoenix
• America/Shiprock
• America/Yellowknife
• Canada/Mountain
• Etc/GMT+7
• MST
• MST7MDT
• Mexico/BajaSur
• Navajo
• PNT
• SystemV/MST7
• SystemV/MST7MDT
• US/Arizona
• US/Mountain
• America/Bahia_Banderas
• America/Belize
• America/Cancun
• America/Chicago
• America/Costa_Rica
• America/El_Salvador
• America/Guatemala
• America/Indiana/Knox
• America/Indiana/Tell_City
• America/Knox_IN
• America/Managua
• America/Matamoros
• America/Menominee
• America/Merida
• America/Mexico_City
• America/Monterrey
• America/North_Dakota/Beulah
• America/North_Dakota/Center
• America/North_Dakota/New_Salem
• America/Rainy_River
• America/Regina
• America/Resolute
• America/Regina
• America/Swift_Current
• America/Tegucigalpa
• America/Winnipeg
• CST
• CST6CDT
• Canada/Central
• Canada/East-Saskatchewan
• Canada/Saskatchewan
• Chile/EasterIsland
• Etc/GMT+6
• Mexico/General
• Pacific/Easter
• Pacific/Galapagos
• SystemV/CST6
• SystemV/CST6CDT
• US/Central
• US/Indiana-Starke
• America/Atikokan
• America/Bogota
• America/Cayman
• America/Coral_Harbour
• America/Detroit
• America/Fort_Wayne
• America/Grand_Turk
• America/Guayaquil
• America/Havana
• America/Indiana/Indianapolis
• America/Indiana/Marengo
• America/Indiana/Petersburg
• America/Indiana/Vevay
• America/Indiana/Vincennes
• America/Indiana/Winamac
• America/Indianapolis
• America/Iqaluit
• America/Jamaica
• America/Kentucky/Louisville
• America/Kentucky/Monticello
• America/Lima
• America/Louisville
• America/Montreal
• America/Nassau
• America/New_York
• America/Nipigon
• America/Panama
• America/Pangnirtung
• America/Port-au-Prince
• America/Thunder_Bay
• America/Toronto
• Canada/Eastern
• Cuba
• EST
• EST5EDT
• Etc/GMT+5
• IET
• Jamaica
• SystemV/EST5
• SystemV/EST5EDT
• US/East-Indiana
• US/Eastern
• US/Michigan
• America/Caracas
• America/Anguilla
• America/Antigua
• America/Argentina/San_Luis
• America/Aruba
• America/Asuncion
• America/Barbados
• America/Blanc-Sablon
• America/Boa_Vista
• America/Campo_Grande
• America/Cuiaba
• America/Curacao
• America/Dominica
• America/Eirunepe
• America/Glace_Bay
• America/Goose_Bay
• America/Grenada
• America/Guadeloupe
• America/Guaya
• America/Halifax
• America/Kralendijk
• America/La_Paz
• America/Lower_Princes
• America/Manaus
• America/Marigot
• America/Martinique
• America/Moncton
• America/Montserrat
• America/Port_of_Spain
• America/Porto_Acre
• America/Porto_Velho
• America/Puerto_Rico
• America/Rio_Branco
• America/Santiago
• America/Santo_Domingo
• America/St_Barthelemy
• America/St_Kitts
• America/St_Lucia
• America/St_Thomas
• America/St_Vincent
• America/Thule
• America/Tortola
• America/Virgin
• Antarctica/Palmer
• Atlantic/Bermuda
• Atlantic/Stanley
• Brazil/Acre
• Brazil/West
• Canada/Atlantic
• Chile/Continental
• Etc/GMT+4
• PRT
• SystemV/AST4
• SystemV/AST4ADT
• America/St_Johns
• CNT
• Canada/Newfoundland
• AGT
• America/Araguaina
• America/Argentina/Buenos_Aires
• America/Argentina/Catamarca
• America/Argentina/ComodRivadavia
• America/Argentina/Cordoba
• America/Argentina/Jujuy
• America/Argentina/La_Rioja
• America/Argentina/Mendoza
• America/Argentina/Rio_Gallegos
• America/Argentina/Salta
• America/Argentina/San_Juan
• America/Argentina/Tucuman
• America/Argentina/Ushuaia
• America/Bahia
• America/Belem
• America/Buenos_Aires
• America/Catamarca
• America/Cayenne
• America/Cordoba
• America/Fortaleza
• America/Godthab
• America/Jujuy
• America/Maceio
• America/Mendoza
• America/Miquelon
• America/Montevideo
• America/Paramaribo
• America/Recife
• America/Rosario
• America/Santarem
• America/Sao_Paulo
• Antarctica/Rothera
• BET
• Brazil/East
• Etc/GMT+3
• America/Noronha
• Atlantic/South_Georgia
• Brazil/DeNoronha
• Etc/GMT+2
• America/Scoresbysund
• Atlantic/Azores
• Atlantic/Cape_Verde
• Etc/GMT+1
• Africa/Abidjan
• Africa/Accra
• Africa/Bamako
• Africa/Banjul
• Africa/Bissau
• Africa/Casablanca
• Africa/Conakry
• Africa/Dakar
• Africa/El_Aaiun
• Africa/Freetown
• Africa/Lome
• Africa/Monrovia
• Africa/Nouakchott
• Africa/Ouagadougou
• Africa/Sao_Tome
• Africa/Timbuktu
• America/Danmarkshavn
• Atlantic/Canary
• Atlantic/Faeroe
• Atlantic/Faroe
• Atlantic/Madeira
• Atlantic/Reykjavik
• Atlantic/St_Helena
• Eire
• Etc/GMT
• Etc/GMT+0
• Etc/GMT-0
• Etc/GMT0
• Etc/Greenwich
• Etc/UCT
• Etc/UTC
• Etc/Universal
• Etc/Zulu
• Europe/Belfast
• Europe/Dublin
• Europe/Guernsey
• Europe/Isle_of_Man
• Europe/Jersey
• GB
• GB-Eire
• GMT
• GMT0
• Greenwich
• Iceland
• Portugal
• UCT
• Universal
• WET
• Zulu
• Africa/Algiers
• Africa/Bangui
• Africa/Brazzaville
• Africa/Ceuta
• Africa/Douala
• Africa/Kinshasa
• Africa/Lagos
• Africa/Libreville
• Africa/Luanda
• Africa/Malabo
• Africa/Ndjamena
• Africa/Niamey
• Africa/Porto-Novo
• Africa/Tunis
• Africa/Windhoek
• Arctic/Longyearbyen
• Atlantic/Jan_Mayen
• CET
• ECT
• Etc/GMT-1
• Europe/Amsterdam
• Europe/Andorra
• Europe/Belgrade
• Europe/Berlin
• Europe/Bratislava
• Europe/Brussels
• Europe/Budapest
• Europe/Copenhagen
• Europe/Gibraltar
• Europe/Ljubljana
• Europe/Luxembourg
• Europe/Malta
• Europe/Monaco
• Europe/Oslo
• Europe/Podgorica
• Europe/Prague
• Europe/Rome
- Europe/San_Marino
- Europe/Sarajevo
- Europe/Skopje
- Europe/Stockholm
- Europe/Tirane
- Europe/Vaduz
- Europe/Vatican
- Europe/Vienna
- Europe/Warsaw
- Europe/Zagreb
- Europe/Zurich
- MET
- Poland
- ART
- Africa/Blantyre
- Africa/Bujumbura
- Africa/Cairo
- Africa/Gaborone
- Africa/Harare
- Africa/Johannesburg
- Africa/Kigali
- Africa/Labumbashi
- Africa/Lusaka
- Africa/Maputo
- Africa/Maseru
- Africa/Mbabane
- Africa/Tripoli
- Asia/Amman
- Asia/Beirut
- Asia/Damascus
- Asia/Gaza
- Asia/Hebron
- Asia/Istanbul
- Asia/Jerusalem
- Asia/Nicosia
- Asia/Tel_Aviv
• CAT
• EET
• Egypt
• Etc/GMT-2
• Europe/Athens
• Europe/Bucharest
• Europe/Chisinau
• Europe/Helsinki
• Europe/Istanbul
• Europe/Kiev
• Europe/Mariehamn
• Europe/Nicosia
• Europe/Riga
• Europe/Simferopol
• Europe/Sofia
• Europe/Tallinn
• Europe/Tiraspol
• Europe/Uzhgorod
• Europe/Vilnius
• Europe/Zaporozhye
• Israel
• Libya
• Turkey
• Africa/Addis_Ababa
• Africa/Asmara
• Africa/Asmera
• Africa/Dar_es_Salaam
• Africa/Djibouti
• Africa/Juba
• Africa/Kampala
• Africa/Khartoum
• Africa/Mogadishu
• Africa/Nairobi
• Antarctica/Syowa
• Asia/Aden
• Asia/Baghdad
• Asia/Bahrain
• Asia/Kuwait
• Asia/Qatar
• Asia/Riyadh
• EAT
• Etc/GMT-3
• Europe/Kaliningrad
• Europe/Minsk
• Indian/Antananarivo
• Indian/Comoro
• Indian/Mayotte
• Asia/Riyadh87
• Asia/Riyadh88
• Asia/Riyadh89
• Mideast/Riyadh87
• Mideast/Riyadh88
• Mideast/Riyadh89
• Asia/Tehran
• Iran
• Asia/Baku
• Asia/Dubai
• Asia/Muscat
• Asia/Tbilisi
• Asia/Yerevan
• Etc/GMT-4
• Europe/Moscow
• Europe/Samara
• Europe/Volgograd
• Indian/Mahe
• Indian/Mauritius
• Indian/Reunion
• NET
• W-SU
• Asia/Kabul
• Antarctica/Mawson
• Asia/Aqtau
• Asia/Aqtobe
• Asia/Ashgabat
• Asia/Ashkhabad
• Asia/Dushanbe
• Asia/Karachi
• Asia/Oral
• Asia/Samarkand
• Asia/Tashkent
• Etc/GMT-5
• Indian/Kerguelen
• Indian/Maldives
• PLT
• Asia/Calcutta
• Asia/Colombo
• Asia/Kolkata
• IST
• Asia/Kathmandu
• Asia/Katmandu
• Antarctica/Vostok
• Asia/Almaty
• Asia/Bishkek
• Asia/Dacca
• Asia/Dhaka
• Asia/Qyzylorda
• Asia/Thimbu
• Asia/Thimphu
• Asia/Yekaterinburg
• BST
• Etc/GMT-6
• Indian/Chagos
• Asia/Rangoon
• Indian/Cocos
• Antarctica/Davis
• Asia/Bangkok
• Asia/Ho_Chi_Minh
• Asia/Hovd
• Asia/Jakarta
• Asia/Novokuznetsk
• Asia/Novosibirsk
• Asia/Omsk
• Asia/Phnom_Penh
• Asia/Pontianak
• Asia/Saigon
• Asia/Vientiane
• Etc/GMT-7
• Indian/Christmas
• VST
• Antarctica/Casey
• Asia/Brunei
• Asia/Choibalsan
• Asia/Chongqing
• Asia/Chungking
• Asia/Harbin
• Asia/Hong_Kong
• Asia/Kashgar
• Asia/Krasnoyarsk
• Asia/Kuala_Lumpur
• Asia/Kuching
• Asia/Macao
• Asia/Macau
• Asia/Makassar
• Asia/Manila
• Asia/Shanghai
• Asia/Singapore
• Asia/Taipei
• Asia/Ujung_Pandang
• Asia/Ulaanbaatar
• Asia/Ulan_Bator
• Asia/Urumqi
• Australia/Perth
• Australia/West
• CTT
• Etc/GMT-8
• Hongkong
• PRC
• Singapore
• Australia/Eucla
• Asia/Dili
• Asia/Irkutsk
• Asia/Jayapura
• Asia/Pyongyang
• Asia/Seoul
• Asia/Tokyo
• Etc/GMT-9
• JST
• Japan
• Pacific/Palau
• ROK
• ACT
• Australia/Adelaide
• Australia/Broken_Hill
• Australia/Darwin
• Australia/North
• Australia/South
• Australia/Yancowinna
• AET
• Antarctica/DumontDUrville
• Asia/Yakutsk
• Australia/ACT
• Australia/Brisbane
• Australia/Canberra
• Australia/Currie
• Australia/Hobart
• Australia/Lindeman
• Australia/Melbourne
• Australia/NSW
• Australia/Queensland
• Australia/Sydney
• Australia/Tasmania
• Australia/Victoria
• Etc/GMT-10
• Pacific/Chuuk
• Pacific/Guam
• Pacific/Port_Moresby
• Pacific/Saipan
• Pacific/Truk
• Pacific/Yap
• Australia/LHI
• Australia/Lord_Howe
• Antarctica/Macquarie
• Asia/Sakhalin
• Asia/Vladivostok
• Etc/GMT-11
• Pacific/Efate
• Pacific/Guadalcanal
• Pacific/Kosrae
• Pacific/Noumea
• Pacific/Pohnpei
• Pacific/Ponape
• SST
• Pacific/Norfolk
• Antarctica/McMurdo
• Antarctica/South_Pole
• Asia/Anadyr
• Asia/Kamchatka
• Asia/Magadan
• Etc/GMT-12
• Kwajalein
• NST
• NZ
• Pacific/Auckland
• Pacific/Fiji
• Pacific/Funafuti
• Pacific/Kwajalein
• Pacific/Majuro
• Pacific/Nauru
• Pacific/Tarawa
• Pacific/Wake
• Pacific/Wallis
• NZ-CHAT
• Pacific/Chatham
• Etc/GMT-13
• MIT
• Pacific/Apiia
• Pacific/Enderbury
• Pacific/Tongatapu
• Etc/GMT-14
• Pacific/Kiritimati