



Brazilian Navy Hydrographic Center (CHM)
Operational Oceanography Division (CH-11)



National Buoy Program (PNBoia)

PNBoia API - Documentation

Version 1.2

May 2024

Summary

1. GENERAL DESCRIPTION.....	3
2. ACCESS.....	3
3. REQUEST METHOD.....	3
4. ENDPOINTS.....	4
4.1. Available buoys.....	4
4.1.1. <i>URL Structure</i>	4
4.1.2. <i>Response and constraints</i>	5
4.2. Metadata.....	5
4.2.1. <i>URL Structure</i>	5
4.2.2. <i>Response and constraints</i>	6
4.3. Qualified data.....	7
4.3.1. <i>URL Structure</i>	7
4.3.2. <i>Response and constraints</i>	7
5. BASIC USAGE GUIDE.....	8
6. EXAMPLE REQUEST.....	9
7. EXAMPLE CODE.....	10

1. GENERAL DESCRIPTION

The PNBoia API serves as a platform for accessing buoy data collected by the National Buoy Program (PNBoia) of the Brazilian Navy Hydrographic Center. This API allows users to retrieve and integrate real-time and historical information from the PNBoia buoys. Refer to Tables 1 and 2 from the Appendix for the available buoys.

2. ACCESS

Access to the API requires valid credentials. Users are encouraged to obtain their credentials by contacting the National Buoy Program via email at (chm.pnboia@marinha.mil.br) providing a desired username and a brief description of the intended usage of the data. Once the credentials are created, a user token will be sent to the user via email.

3. REQUEST METHOD

Users can retrieve data from the PNBoia API by making HTTP requests utilizing the GET method. The URL structure follows a defined format, adhering to standard URL conventions. Clients can construct requests by appending specific parameters (i.e. query strings) to the base API endpoint. Detailed guidelines for constructing valid URLs and examples are outlined in the documentation.

4. ENDPOINTS

Information of the PNBoia can be retrieved through a number of paths (i.e. endpoints) within the API. Three main types of endpoints are provided to the users: available buoys, metadata and qualified data endpoints (Table 1). Their description, use and constraints are described in the next subsections (Sections 4.1 to 4.3).

Table 1 - List of available endpoints.

Endpoint	Description
<i>info/available_buoys</i>	Returns a list of the available buoys
<i>info/metadata</i>	Returns metadata of a defined buoy
<i>qualified_data/pnboia</i>	Returns qualified data for PNBoia historical buoys
<i>qualified_data/bmobr</i>	Returns qualified data for BMOBr buoys
<i>qualified_data/spotter</i>	Returns qualified data for Spotter buoys

<code>qualified_data/triaxys</code>	Returns qualified data for Triaxys buoys
-------------------------------------	--

4.1. Available buoys

The `/info/available_buoys` endpoint can be used to retrieve a list of the available buoys for the user along with relevant information for each buoy such as: buoy ID, situation, datetime of the most recent data, position, local and metarea section. By using the query string `operative = True` (See section 4.1.1 - Table 2), the user can retrieve a list of the currently active buoys (i.e. generating real time data).

We encourage the user to consult this endpoint prior to the other endpoint types (i.e. metadata and qualified data) to make sure that the correct buoy ID and qualified data endpoint are being used for requesting data. This can be achieved by looking for the field “API Endpoint” (HTML response) or “api_endpoint” (JSON response). An example of the usage is described in Section 5.

4.1.1. URL Structure

The URL structure for the `/info/available_buoys` is composed as follows:

```
http://dados.pnboia.org/v1/info/available_buoys?token=[USER_TOKEN]&operative=[OPERATIVE]&response_type=[RESPONSE_TYPE]
```

Where:

Table 2 - Description of metadata request URL components.

Component	Description
<code>http://dados.pnboia.org</code>	Base URL
<code>/v1</code>	API Version
<code>/info/available_buoys</code>	Available buoys endpoint
<code>[USER_TOKEN]</code>	User token (provided on demand)
<code>[OPERATIVE]</code>	Whether to return only the currently operative boys [<i>True</i> or <i>False</i>] (default to <i>False</i>)
<code>[RESPONSE_TYPE]</code>	The response type of interest [<i>html</i> , <i>txt</i> or <i>json</i>] (default to <i>html</i>)

4.1.2. Response and constraints

The */info/available_buoys* endpoint allows users to retrieve buoy information in either HTML, JSON or TXT file format. The HTML response type can be used to quickly view the requested information using a web browser (Figure 1 - A). The JSON response type is useful when the user needs to integrate the information in a custom system (Figure 1 - B). Lastly, the TXT response response type responses a .txt file with the buoy metadata (Figure 1 - C). No relevant use constraints needs to be addressed for this endpoint.

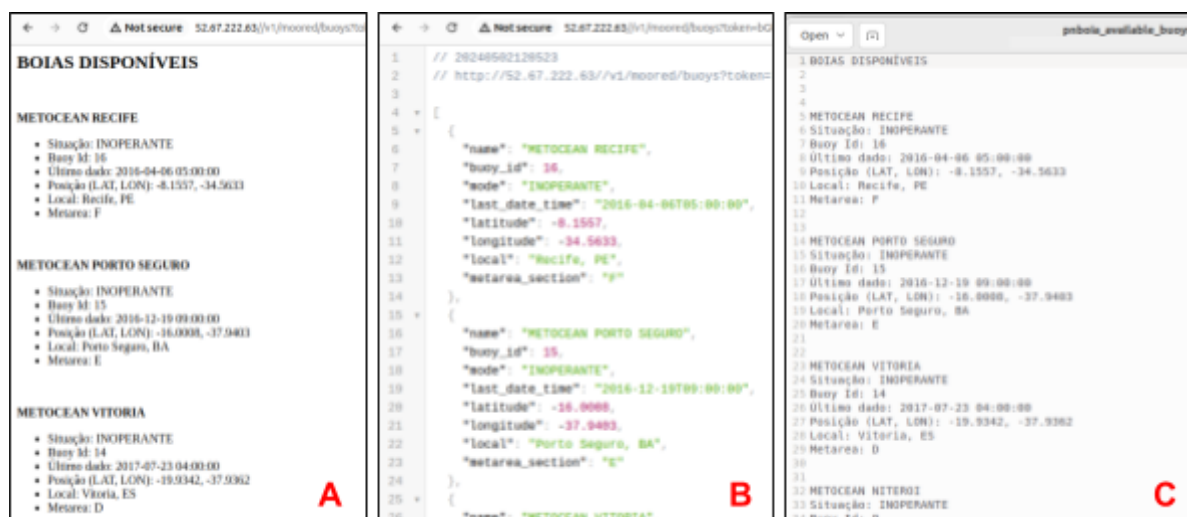


Figure 1 - Examples of HTML (A), JSON (B) and TXT file (C) responses of the available buoys endpoint.

4.2. Metadata

The */metadata* endpoint can be used to retrieve relevant information of a specific buoy, such as: current situation, history, mooring information, buoy (i.e. equipment) information, sensors configuration, parameters provided and quality control processes.

4.2.1. URL Structure

The URL structure for the */metadata* is composed as follows:

```
http://dados.pnboia.org/v1/info/metadata?token=[USER_TOKEN]&buoy_id=[BUOY_ID]&response_type=[RESPONSE_TYPE]
```

Where:

Table 3 - Description of metadata request URL components.

Component	Description
<i>http://dados.pnboia.org</i>	Base URL
<i>/v1</i>	API Version

<code>/info/metadata</code>	metadata endpoint
<code>[USER_TOKEN]</code>	User token (provided on demand)
<code>[BUOY_ID]</code>	Buoy id of the buoy of interest (Refer to Section 4.1)
<code>[RESPONSE_TYPE]</code>	The response type of interest [<i>html</i> , <i>txt</i> or <i>json</i>] (default to <i>html</i>)

4.2.2. Response and constraints

The `/metadata` endpoint allows users to retrieve buoy information in either HTML, JSON or TXT file format. The HTML response type can be used to quickly view the requested information using a web browser (Figure 2 - A). The JSON response type is useful when the user needs to integrate the information in a custom system (Figure 2 - B). Lastly, the TXT response type responses a .txt file with the buoy metadata (Figure 2 - C). No relevant use constraints needs to be addressed for this endpoint.



Figure 2 - Examples of HTML (A), JSON (B) and TXT file (C) responses of the metadata endpoint.

4.3. Qualified data

The `/qualified_data` endpoints can be used to retrieve actual environmental data generated by the buoys, such as meteorological and oceanographic parameters, with their respective flags generated by quality control tests.

4.3.1. URL Structure

The URL structure for the *qualified_data* endpoints is composed as follows:

```
http://dados.pnboia.org/v1/[ENDPOINT]?token=[USER_TOKEN]&buoy_id=[BUOY_ID]&last=[LAST]&start_date=[START_DATE]&end_date=[END_DATE]&response_type=[RESPONSE_TYPE]
```

Where:

Table 4 - Description of request URL components.

Component	Description
<code>http://dados.pnboia.org/</code>	Base URL
<code>/v1</code>	API Version
<code>[ENDPOINT]</code>	Qualified data endpoint to be requested (Refer to Table 1)
<code>[USER_TOKEN]</code>	User token (provided on demand)
<code>[BUOY_ID]</code>	Buoy id of the buoy of interest (Refer to Appendix - Table 1)
<code>[LAST]</code>	Whether to retrieve only the most recent data, <i>true</i> or <i>false</i> (default to <i>false</i>)
<code>[START_DATE]</code> (YYYY-mm-ddTHH:MM:SS)	Start datetime of the window of interest (inclusive) (ignored if <i>last = True</i>)
<code>[END_DATE]</code> (YYYY-mm-ddTHH:MM:SS)	End datetime of the window of interest (inclusive) (ignored if <i>last = True</i>)
<code>[RESPONSE_TYPE]</code>	The response type of interest, <i>json</i> or <i>csv</i> (default to <i>json</i>)

4.3.2. Response and constraints

The */qualified_data* endpoints allows users to retrieve data in either JSON or CSV file format. The JSON response is composed of a list of JSONs, each being structured by parameter name and value pairs (Figure 3). Refer to the Appendix - Table 2 for parameter names, data types, units, and descriptions.

For historical data (i.e. *last=True*), the API responses **are constrained to a maximum of 30 days of data**. Therefore, in case the requested timespan is greater than 30 days, the response will only contain 30 days of data, beginning in the start date provided. Users are encouraged to utilize pagination techniques or refine their queries to adhere to the specified temporal constraints. Further in this documentation, Code Snippet 2

(section Example Code) provides a Python example code for the API requests of timespans greater than 30 days.

```
[
  {
    "buoy_id": 10,
    "date_time": "2012-01-01T02:00:00",
    "latitude": -25.28112,
    "longitude": -44.92688,
    "battery": 12.37,
    "flag_battery": 0,
    "rh": 87.1,
    "flag_rh": 0,
    "wspd1": null,
    "flag_wspd1": 2,
    "wdir1": 92,
    "flag_wdir1": 0,
  }
]
```

Figure 1 - Example of a JSON response structure.

5. BASIC USAGE GUIDE

Bellow are described steps of the basic usage of the API:

1. Consult all available buoys with endpoint `/info/available_buoys` to define the desired buoy (see Section 4.1);
2. Once the buoy is chosen:
 - 2.1. Check the API Endpoint (to compose the qualified data URL structure)
 - 2.2. Get the buoy ID (to compose the qualified data URL structure as a query parameter);
3. Consult the the buoy's metadata with the chosen buoy ID (see Section 4.2);
4. Retrieve qualified data using the buoy ID and desired start and end dates.

6. EXAMPLE REQUEST

The following request URL retrieves data from buoy METOCEAN SANTOS (Buoy ID = 10) between 2012-01-01T00:00:00 to 2012-01-30T00:00:00 as a list of JSONs (Figure 1):

n

```

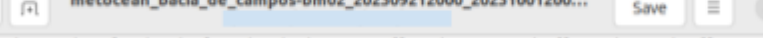
{"buoy id":10,"date time":"2012-01-11T02:00:00","latitude":-25.28112,"longitude":-44.92688,"battery":12.37,"flag battery":0,"rh":87.1,"flag rh":0,"wsdp1":null,"flag wdp1":0,"gust1":0.1,"flag gust1":0,"gust2":14.66,"flag gust2":0,"atmp":23.3,"flag atmp":0,"pres":1008.27,"flag pres":0,"srad":7.83,"flag srad":0,"cdir1":221,"flag cdir1":0,"cpd2":0.5935,"flag cpd2":0,"cdir2":219,"flag cdir2":0,"cpd3":0.5709,"flag cpd3":0,"cdir4":null,"flag cpd5":null,"flag cpd5":null,"cdir5":null,"flag cdir5":null,"cpd6":null,"flag cpd6":null,"cdir6":null,"flag cdir6":null,"cpd8":null,"flag cdir8":null,"flag cpd8":null,"cpd9":null,"flag cdir9":null,"cpd10":null,"flag cdir10":null,"flag cdir11":null,"cpd12":null,"flag cpd12":null,"cdir12":null,"flag cdir12":null,"cpd13":null,"flag cpd13":null,"cdir13":null,"flag cdir13":null,"cpd15":null,"flag cpd15":null,"cdir15":null,"flag cdir15":null,"cpd16":null,"flag cpd16":null,"cdir16":null,"flag cdir16":null,"flag cpd18":null,"cdir18":null,"flag cdir18":null,"cpd19":null,"flag cpd19":null,"cdir19":null,"flag cdir19":null,"cpd20":null,"flag cpd20":null,"tp1":6.5,"flag tp1":0,"mxxvht1":4.51,"flag mxxvht1":0,"wdvir1":127,"flag wdvir1":0,"wvspread1":33,"flag wvspread1":0,"swvht2":0,"flag tml":null,"flag tml":null,"pkdir1":null,"flag pkdir1":null,"pkspread1":null,"flag pkspread1":null,"sensors_data_flagged":null,"cont latitude":0.0}, {"buoy id":10,"date time":"2012-01-11T01:00:00","latitude":-25.28112,"longitude":-44.926945,"battery":12.42,"flag battery":0,"rh":81.9,"flag rh":0,"wsdp1":null,"flag wdp1":0,"cdir2":0,"gust1":0.1,"flag gust1":0,"gust2":15.44,"flag gust2":0,"atmp":23.5,"flag atmp":0,"pres":1008.6,"flag pres":0,"srad":7.5,"flag srad":0,"cpd1":0,"cdir1":215,"flag cdir1":0,"cpd2":0.6073,"flag cpd2":0,"cdir2":217,"flag cdir2":0,"cpd3":0.5821,"flag cpd3":0,"cdir4":null,"cpd5":null,"flag cpd5":null,"flag cpd5":null,"cdir5":null,"flag cdir5":null,"cpd6":null,"flag cpd6":null,"cdir6":null,"flag cdir6":null,"cpd8":null,"flag cdir8":null,"flag cpd8":null,"cpd9":null,"flag cdir9":null,"cpd10":null,"flag cdir10":null,"flag cdir11":null,"cpd12":null,"flag cpd12":null,"cdir12":null,"flag cdir12":null,"cpd13":null,"flag cpd13":null,"cdir13":null,"flag cdir13":null,"cpd15":null,"flag cpd15":null,"cdir15":null,"flag cdir15":null,"cpd16":null,"flag cpd16":null,"cdir16":null,"flag cdir16":null,"flag cpd18":null,"cdir18":null,"flag cdir18":null,"cpd19":null,"flag cpd19":null,"cdir19":null,"flag cdir19":null,"cpd20":null,"flag cpd20":null,"tp1":6.9,"flag tp1":0,"mxxvht1":4.05,"flag mxxvht1":0,"wdvir1":128,"flag wdvir1":0,"wvspread1":36,"flag wvspread1":0,"swvht2":0,"flag tml":null,"flag tml":null,"pkdir1":null,"flag pkdir1":null,"pkspread1":null,"flag pkspread1":null,"sensors_data_flagged":null,"cont latitude":0.0}, {"buoy id":10,"date time":"2012-01-11T00:00:00","latitude":-25.281035,"longitude":-44.926905,"battery":12.45,"flag battery":0,"rh":83.2,"flag rh":0,"wsdp1":null,"flag wdp1":0,"cdir2":0,"gust1":0.1,"flag gust1":0,"gust2":15.89,"flag gust2":0,"atmp":23.5,"flag atmp":0,"pres":1008.44,"flag pres":0,"srad":7.4,"flag srad":0,"cpd1":0,"cdir1":217,"flag cdir1":0,"cpd2":0.5881,"flag cpd2":0,"cdir2":218,"flag cdir2":0,"cpd3":0.5883,"flag cpd3":0,"cdir4":null,"cpd5":null,"flag cpd5":null,"flag cpd5":null,"cdir5":null,"flag cdir5":null,"cpd6":null,"flag cpd6":null,"cdir6":null,"flag cdir6":null,"cpd8":null,"flag cdir8":null,"flag cpd8":null,"cpd9":null,"flag cdir9":null,"cpd10":null,"flag cdir10":null,"flag cdir11":null,"cpd12":null,"flag cpd12":null,"cdir12":null,"flag cdir12":null,"cpd13":null,"flag cpd13":null,"cdir13":null,"flag cdir13":null,"cpd15":null,"flag cpd15":null,"cdir15":null,"flag cdir15":null,"cpd16":null,"flag cpd16":null,"cdir16":null,"flag cdir16":null,"flag cpd18":null,"cdir18":null,"flag cdir18":null,"cpd19":null,"flag cpd19":null,"cdir19":null,"flag cdir19":null,"cpd20":null,"flag cpd20":null,"tp1":6.3,"flag tp1":0,"mxxvht1":3.95,"flag mxxvht1":0,"wdvir1":128,"flag wdvir1":0,"wvspread1":34,"flag wvspread1":0,"swvht2":0,"flag tml":null,"flag tml":null,"pkdir1":null,"flag pkdir1":null,"pkspread1":null,"flag pkspread1":null,"sensors_data_flagged":null,"cont latitude":0.0}]






```

start and end dates.

values of the `[RESPONSE_TYPE]` string query to `csv`. If the request is performed with a regular web browser (e.g. google chrome, mozilla firefox), the file will either be automatically saved locally or through a download prompt, depending on your web browser preferences.

```
&start_date=2012-01-01T00:00:00&end_date=2012-01-30T00:00:00&response_type=csv
```



Open  metocean_bacia_de_campos-bm02_202309212000_202310012000... Save    

```
1 buoy_id,date,time,latitude,longitude,battery,flag_battery,rh,flag_rh,wspdl,flag_wspdl,wdirl,flag_wdirl
2 22,2023-10-01
3 18:00:00,-22.867100,-40.153000,13.70,0.94,10,9.8.61,0.317,0.8.89,0.290,0.10.82,0.11.28,0.23.60
4 22,2023-10-01
5 15:00:00,-22.869100,-40.154400,13.80,0.89.60,9.9.72,0.327,0.9.91,0.293,0.13.47,0.13.9,0.23.60
6 22,2023-10-01
7 12:00:00,-22.867800,-40.155600,13.50,0.89.10,9.11.48,0.333,0.11.85,0.299,0.14.8,0.14.92,0.22.70
8 22,2023-10-01
9 06:00:00,-22.862700,-40.151300,12.80,0.80.40,9.9.61,0.351,0.9.8,0.316,0.11.7,0.11.73,0.22.70
```

Figure 3 - Example of a CSV file response. Column names are provided in the first row of the file.

7. EXAMPLE CODE

Python code snippets are provided below as examples of requests for qualified data. The first example (Code Snippet 1) consists of a request for data from buoy METOCEAN SANTOS (Buoy ID = 10) between 2012-01-01T00:00:00 to 2012-01-30T00:00:00. In this case, since the requested timespan is not greater than 30 days, the simple request method satisfies what is needed. The second example (Code Snippet 2) shows a requested timespan of 3 months, where a pagination technique is applied to retrieve the desired dataset.

Code Snippet 1 - Simple request

```
import requests
from datetime import datetime

endpoint = 'qualified_data/pnboia'
buoy_id = 10
token = '[USER_TOKEN]' # pass your user token here as a string
start_date = datetime(2012,1,1,0,0,0).strftime("%Y-%m-%dT%H:%M:%S") # use
the datetime format YYYY-mm-ddTHH:MM:SS
end_date = datetime(2012,1,30,0,0,0).strftime("%Y-%m-%dT%H:%M:%S")
response_type = 'json'

url =
f"http://dados.pnboia.org/v1/{endpoint}?buoy_id={buoy_id}&token={token}&start_
t_date={start_date}&end_date={end_date}&response_type={response_type}"

data = requests.get(url).json()

print(data[-1]) # show the most recent retrieved data
```

Code Snippet 2 - Pagination

```
import requests
from datetime import datetime, timedelta

endpoint = 'qualified_data/pnboia'
buoy_id = 10
token = '[USER_TOKEN]'
start_date = datetime(2012,1,5,0,0,0)
end_date = datetime(2012,4,1,0,0,0)
response_type = 'json'

page = 1
current_date = start_date
data = []
```

```

print("Paginating PNBoia API...\n")
while True:
    print(f"Page {page}")
    page_end_date = current_date + timedelta(days=30)
    print(f"start date: {current_date}")
    print(f"end date: {page_end_date}")

    if page_end_date >= end_date:
        page_end_date = end_date

    url =
f'http://dados.pnboia.org/v1/{endpoint}?buoy_id={buoy_id}&token={token}&star
t_date={current_date.strftime("%Y-%m-%dT%H:%M:%S")}&end_date={page_end_date.
strftime("%Y-%m-%dT%H:%M:%S")}&response_type={response_type}'

    page_data = requests.get(url).json()
    data[0:] = page_data

    last_datetime =
datetime.strptime(page_data[-1]["date_time"], ("%Y-%m-%dT%H:%M:%S"))

    if last_datetime >= end_date:
        break

    current_date = last_datetime
    page += 1
    print("-"*10)

print("End of pagination.\n")

print(f"Number of observations: {len(data)}")
print(f"Start date: {data[-1]['date_time']}")
print(f"End date: {data[-1]['date_time']}\n")

print(data[-1]) # show the most recent retrieved data

```