

---

# **callsignlookuptools**

***Release 1.0.1***

**classabbyamp, 0x5c**

**Sep 28, 2021**



# CONTENTS

<b>1</b>	<b>CLI Usage</b>	<b>3</b>
<b>2</b>	<b>API Reference</b>	<b>5</b>
2.1	QRZ . . . . .	5
2.2	Callook . . . . .	7
2.3	HamQTH . . . . .	8
2.4	QRZCQ . . . . .	11
<b>3</b>	<b>Data Types</b>	<b>15</b>
3.1	Callsign Data . . . . .	15
3.2	Exceptions . . . . .	18
3.3	Helper Data Types . . . . .	18
3.4	Enums . . . . .	21
<b>4</b>	<b>Installation</b>	<b>23</b>
<b>5</b>	<b>API Support</b>	<b>25</b>
<b>6</b>	<b>Getting Started</b>	<b>27</b>
6.1	Sync . . . . .	27
6.2	Async . . . . .	27
<b>7</b>	<b>License</b>	<b>29</b>
	<b>Index</b>	<b>31</b>



A [QRZ](#), [Callook](#), [HamQTH](#), and [QRZCQ](#) API interface in Python with sync and async support.



## CLI USAGE

**Note:** To use the CLI, install with the extra cli (e.g. `pip install callsignlookuptools[cli]`) or otherwise install the library `typer[all]` and `click-help-colors`.

`callsignlookuptools` has a basic CLI interface, which can be run using:

```
$ python3 -m callsignlookuptools
```

It can be used with the following arguments:

```
$ python3 -m callsignlookuptools --help
Usage: python -m callsignlookuptools [OPTIONS] COMMAND [ARGS]...

  A QRZ, Callook, HamQTH, and QRZCQ API interface in Python with sync and async support.

Options:
  -v, --version  Show the version of this program and exit.
  -h, --help     Show this message and exit.

Commands:
  callook  Use Callook to look up a callsign
  hamqth   Use HamQTH to look up a callsign
  qrz      Use QRZ to look up a callsign
  qrzcq    Use QRZCQ to look up a callsign
```

Each lookup source is a subcommand, and any source-specific options can be viewed by using the `-h` or `--help` argument on that subcommand.

For example, the QRZ lookup source has the following options:

```
$ python3 -m callsignlookuptools qrz --help
Usage: python -m callsignlookuptools qrz [OPTIONS] CALL

  Use QRZ to look up a callsign

  Requires a QRZ account and an XML Logbook Data or QRZ Premium subscription

Options:
  -u, --user, --username TEXT  QRZ username (will be prompted if not provided)  ↵
  ↵[required]
  -p, --pass, --password TEXT  QRZ password (will be prompted if not provided)  ↵
  ↵[required]
```

(continues on next page)

(continued from previous page)

CALL	The callsign to look up [required]
-h, --help	Show this message and exit.

An example invocation that looks up W1XYZ on QRZ, where the username is provided, and the password will be prompted for:

<pre>\$ python3 -m callsignlookuptools qrz --username w1aw W1XYZ</pre>
--



## API REFERENCE

### 2.1 QRZ

```
class callsignlookuptools.QrzSyncClient(username, password, session_key="",  
                                         useragent='python-callsignlookuptools-v1.0.1', session=None)
```

Synchronous QRZ API client

**Parameters**

- **username** (str) – QRZ username
- **password** (str) – QRZ password
- **session\_key** (str) – QRZ login session key
- **useragent** (str) – Useragent for QRZ
- **session** (Optional[Session]) – A requests session to use for requests

**search**(callsign)

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**property password:** str

**Getter** gets password

**Setter** sets password

**Return type** str

**property session:** requests.Session

**Getter** gets the requests session

**Setter** sets the requests session

**Return type** Session

**property session\_key:** str

**Getter** gets API session key

**Setter** sets API session key

**Return type** str

**property useragent:** **str**

**Getter** gets useragent

**Setter** sets useragent

**Return type** **str**

**property username:** **str**

**Getter** gets username

**Setter** sets username

**Return type** **str**

**class** callsignlookuptools.**QrzAsyncClient**(*username, password, session\_key="",  
useragent='python-callsignlookuptools-v1.0.1', session=None*)

Asynchronous QRZ API client

**Parameters**

- **username** (**str**) – QRZ username
- **password** (**str**) – QRZ password
- **session\_key** (**str**) – QRZ login session key
- **useragent** (**str**) – Useragent for QRZ
- **session** (**Optional[ClientSession]**) – An aiohttp session to use for requests

**async classmethod new**(*username, password, session\_key="",  
useragent='python-callsignlookuptools-v1.0.1', session=None*)

Creates a QrzAsyncClient object and automatically starts a session if not provided.

**Parameters**

- **username** (**str**) – QRZ username
- **password** (**str**) – QRZ password
- **session\_key** (**str**) – QRZ login session key
- **useragent** (**str**) – Useragent for QRZ
- **session** (**Optional[ClientSession]**) – An aiohttp session to use for requests

**Return type** *QrzAsyncClient*

**async search**(*callsign*)

Search for a callsign

**Parameters** **callsign** (**str**) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**async close\_session**()

Closes the aiohttp.ClientSession session

**property password:** **str**

**Getter** gets password

**Setter** sets password

**Return type** str

**property session:** Optional[aiohttp.client.ClientSession]

**Getter** gets the aiohttp session

**Setter** sets the aiohttp session

**Return type** Optional[ClientSession]

**property session\_key:** str

**Getter** gets API session key

**Setter** sets API session key

**Return type** str

**async start\_session()**  
Creates a new aiohttp.ClientSession

**property useragent:** str

**Getter** gets useragent

**Setter** sets useragent

**Return type** str

**property username:** str

**Getter** gets username

**Setter** sets username

**Return type** str

## 2.2 Callook

**class** callsignlookuptools.CallookSyncClient(*session=None*)  
Synchronous Callook API client

**Parameters** *session* (Optional[Session]) – A requests session to use for requests

**search**(*callsign*)  
Search for a callsign

**Parameters** *callsign* (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**property session:** requests.Session

**Getter** gets the requests session

**Setter** sets the requests session

**Return type** Session

**class** callsignlookuptools.CallookAsyncClient(*session=None*)  
Asynchronous Callook API client

**Parameters** *session* (Optional[ClientSession]) – An aiohttp session to use for requests

**async classmethod new**(*session=None*)

Creates a CallookAsyncClient object and automatically starts a session if not provided.

**Parameters** **session** (Optional[ClientSession]) – An aiohttp session to use for requests

**Return type** *CallookAsyncClient*

**async search**(*callsign*)

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**async close\_session**()

Closes the aiohttp.ClientSession session

**property session:** Optional[aiohttp.client.ClientSession]

**Getter** gets the aiohttp session

**Setter** sets the aiohttp session

**Return type** Optional[ClientSession]

**async start\_session**()

Creates a new aiohttp.ClientSession

## 2.3 HamQTH

```
class callsignlookuptools.HamQthSyncClient(username, password, session_key="",  
                                           useragent='python-callsignlookuptools-v1.0.1',  
                                           session=None)
```

Synchronous HamQTH API client

**Parameters**

- **username** (str) – HamQTH username
- **password** (str) – HamQTH password
- **session\_key** (str) – HamQTH login session key
- **useragent** (str) – Useragent for HamQTH
- **session** (Optional[Session]) – A requests session to use for requests

**search**(*callsign*)

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**property password:** str

**Getter** gets password

```

    Setter sets password
    Return type str

property session: requests.Session
    Getter gets the requests session
    Setter sets the requests session
    Return type Session

property session_key: str
    Getter gets API session key
    Setter sets API session key
    Return type str

property useragent: str
    Getter gets useragent
    Setter sets useragent
    Return type str

property username: str
    Getter gets username
    Setter sets username
    Return type str

class callsignlookuptools.HamQthAsyncClient(username, password, session_key="",
                                           useragent='python-callsignlookuptools-v1.0.1',
                                           session=None)

Asynchronous HamQTH API client

Parameters
    • username (str) – HamQTH username
    • password (str) – HamQTH password
    • session_key (str) – HamQTH login session key
    • useragent (str) – Useragent for HamQTH
    • session (Optional[ClientSession]) – An aiohttp session to use for requests

async classmethod new(username, password, session_key="",
                       useragent='python-callsignlookuptools-v1.0.1', session=None)
    Creates a HamQthAsyncClient object and automatically starts a session if not provided.

Parameters
    • username (str) – HamQTH username
    • password (str) – HamQTH password
    • session_key (str) – HamQTH login session key
    • useragent (str) – Useragent for HamQTH
    • session (Optional[ClientSession]) – An aiohttp session to use for requests

```

**async search**(*callsign*)

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**async close\_session**()

Closes the aiohttp.ClientSession session

**property password:** str

**Getter** gets password

**Setter** sets password

**Return type** str

**property session:** Optional[aiohttp.client.ClientSession]

**Getter** gets the aiohttp session

**Setter** sets the aiohttp session

**Return type** Optional[ClientSession]

**property session\_key:** str

**Getter** gets API session key

**Setter** sets API session key

**Return type** str

**async start\_session**()

Creates a new aiohttp.ClientSession

**property useragent:** str

**Getter** gets useragent

**Setter** sets useragent

**Return type** str

**property username:** str

**Getter** gets username

**Setter** sets username

**Return type** str

## 2.4 QRZCQ

```
class callsignlookuptools.QrzcqSyncClient(username, password, session_key="",
                                         useragent='python-callsignlookuptools-v1.0.1',
                                         session=None)
```

Synchronous QRZCQ API client

### Parameters

- **username** (str) – QRZCQ username
- **password** (str) – QRZCQ password
- **session\_key** (str) – QRZCQ login session key
- **useragent** (str) – Useragent for QRZCQ
- **session** (Optional[Session]) – A requests session to use for requests

**search**(callsign)

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

**property password:** str

**Getter** gets password

**Setter** sets password

**Return type** str

**property session:** requests.Session

**Getter** gets the requests session

**Setter** sets the requests session

**Return type** Session

**property session\_key:** str

**Getter** gets API session key

**Setter** sets API session key

**Return type** str

**property useragent:** str

**Getter** gets useragent

**Setter** sets useragent

**Return type** str

**property username:** str

**Getter** gets username

**Setter** sets username

**Return type** str

```
class callsignlookuptools.QrzCqAsyncClient(username, password, session_key="",
                                           useragent='python-callsignlookuptools-v1.0.1',
                                           session=None)
```

Asynchronous QRZCQ API client

**Parameters**

- **username** (str) – QRZCQ username
- **password** (str) – QRZCQ password
- **session\_key** (str) – QRZCQ login session key
- **useragent** (str) – Useragent for QRZCQ
- **session** (Optional[ClientSession]) – An aiohttp session to use for requests

```
async classmethod new(username, password, session_key="",
                      useragent='python-callsignlookuptools-v1.0.1', session=None)
```

Creates a QrzCqAsyncClient object and automatically starts a session if not provided.

**Parameters**

- **username** (str) – QRZCQ username
- **password** (str) – QRZCQ password
- **session\_key** (str) – QRZCQ login session key
- **useragent** (str) – Useragent for QRZCQ
- **session** (Optional[ClientSession]) – An aiohttp session to use for requests

```
async search(callsign)
```

Search for a callsign

**Parameters** **callsign** (str) – the callsign to look up

**Return type** *CallsignData*

**Returns** the callsign data from the lookup service

**Raises** *common.exceptions.CallsignLookupError* on network or parsing error

```
async close_session()
```

Closes the aiohttp.ClientSession session

**property password:** str

**Getter** gets password

**Setter** sets password

**Return type** str

**property session:** Optional[aiohttp.client.ClientSession]

**Getter** gets the aiohttp session

**Setter** sets the aiohttp session

**Return type** Optional[ClientSession]

**property session\_key:** str

**Getter** gets API session key

**Setter** sets API session key



**Return type** str

**async start\_session()**

Creates a new aiohttp.ClientSession

**property useragent: str**

**Getter** gets useragent

**Setter** sets useragent

**Return type** str

**property username: str**

**Getter** gets username

**Setter** sets username

**Return type** str



## DATA TYPES

---

**Note:** If an attribute of any class is unknown or not included from that data source, it will be `None` or a similar enum value.

---

### 3.1 Callsign Data

```
class callsignlookuptools.CallsignData(query, raw_data, data_source, type=None, callsign=None,
                                       aliases=None, trustee=None, lic_class=None, lic_codes=None,
                                       effective_date=None, expire_date=None, last_action_date=None,
                                       prev_call=None, prev_lic_class=None, modified_date=None,
                                       name=None, address=None, dxcc=None, dxcc_prefix=None,
                                       qth=None, continent=None, latlong=None, grid=None,
                                       county=None, district=None, oblast=None, dok=None,
                                       sondok=None, plot=None, fips=None, msa=None,
                                       area_code=None, cq_zone=None, itu_zone=None, iota=None,
                                       geoloc_src=None, timezone=None, qsl=None, born=None,
                                       licensed=None, email=None, username=None, url=None,
                                       page_views=None, db_serial=None, bio=None, image=None,
                                       social_media=None, uls_url=None, frn=None)
```

Represents the data for a callsign retrieved from a lookup service

**query:** `str`

the callsign searched for

**raw\_data:** `pydantic.main.BaseModel`

the raw data, as parsed by pydantic from the API response. Probably not needed for most use cases.

**data\_source:** `callsignlookuptools.common.enums.DataSource`

the lookup service the data comes from

**type:** `Optional[callsignlookuptools.common.enums.CallsignType] = None`

the type of license the callsign is associated with

**callsign:** `Optional[str] = None`

the callsign, as received from the lookup service. Not always the same as the query.

**aliases:** `Optional[list] = None`

alias callsigns

**trustee:** `Optional[callsignlookuptools.common.dataclasses.Trustee] = None`

trustee info

**lic\_class:** `Optional[Union[str, callsignlookuptools.common.enums.LicenseClass]] = None`  
license class. Some lookup services have a defined set of classes.

**lic\_codes:** `Optional[str] = None`  
license codes

**effective\_date:** `Optional[datetime.datetime] = None`  
license effective/grant date

**expire\_date:** `Optional[datetime.datetime] = None`  
license expiration date

**last\_action\_date:** `Optional[datetime.datetime] = None`  
license last updated date

**prev\_call:** `Optional[str] = None`  
previous callsign

**prev\_lic\_class:** `Optional[callsignlookuptools.common.enums.LicenseClass] = None`  
previous license class

**modified\_date:** `Optional[datetime.datetime] = None`  
lookup service record modification date

**name:** `Optional[callsignlookuptools.common.dataclasses.Name] = None`  
licensee name

**address:** `Optional[callsignlookuptools.common.dataclasses.Address] = None`  
licensee address

**dxcc:** `Optional[callsignlookuptools.common.dataclasses.Dxcc] = None`  
licensee DXCC entity

**dxcc\_prefix:** `Optional[str] = None`  
licensee DXCC's primary callsign prefix

**qth:** `Optional[str] = None`  
licensee location

**continent:** `Optional[callsignlookuptools.common.enums.Continent] = None`  
licensee continent

**latlong:** `Optional[gridtools.gridtools.LatLong] = None`  
latitude and longitude of address or QTH

**grid:** `Optional[gridtools.gridtools.Grid] = None`  
grid square locator of address or QTH

**county:** `Optional[str] = None`  
county of address or QTH

**district:** `Optional[str] = None`  
district of address or QTH

**oblast:** `Optional[str] = None`  
oblast of address or QTH (Russia only)

**dok:** `Optional[str] = None`  
DOK name (Germany only)

**sondok:** `Optional[bool] = None`  
whether the DOK is a Sonder-DOK (Germany only)

**plot:** `Optional[str] = None`  
 Polish OT number (Poland only)

**fips:** `Optional[str] = None`  
 Federal Information Processing Standards number (USA only)

**msa:** `Optional[str] = None`  
 Metro Service Area (USA only)

**area\_code:** `Optional[str] = None`  
 telephone area code (USA only)

**cq\_zone:** `Optional[int] = None`  
 CQ zone

**itu\_zone:** `Optional[int] = None`  
 ITU zone

**iota:** `Optional[str] = None`  
 Islands on the Air designator

**geoloc\_src:** `Optional[callsignlookuptools.common.enums.GeoLocSource] = None`  
 geolocation information source

**timezone:** `Optional[callsignlookuptools.common.dataclasses.Timezone] = None`  
 licensee time zone

**qsl:** `Optional[callsignlookuptools.common.dataclasses.Qsl] = None`  
 QSL info

**born:** `Optional[int] = None`  
 year born

**licensed:** `Optional[int] = None`  
 year licensed

**email:** `Optional[str] = None`  
 licensee email address

**username:** `Optional[str] = None`  
 username of license page manager

**url:** `Optional[str] = None`  
 url of the webpage for the callsign

**page\_views:** `Optional[int] = None`  
 callsign page views

**db\_serial:** `Optional[str] = None`  
 QRZ database serial number

**bio:** `Optional[callsignlookuptools.common.dataclasses.Bio] = None`  
 biography info

**image:** `Optional[callsignlookuptools.common.dataclasses.Image] = None`  
 profile image info

**social\_media:** `Optional[callsignlookuptools.common.dataclasses.SocialMedia] = None`  
 social media info

**uls\_url:** `Optional[str] = None`  
 ULS record url (USA only)

**frn:** `Optional[str] = None`  
FRN (USA only)

## 3.2 Exceptions

**class** `callsignlookuptools.CallsignLookupError(*args)`  
The exception raised when something goes wrong in callsignlookuptools

## 3.3 Helper Data Types

**class** `callsignlookuptools.common.dataclasses.Dxcc(id=None, name=None)`  
Represents a DXCC entity

**id:** `Optional[int] = None`  
entity ID

**name:** `Optional[str] = None`  
entity name

**class** `callsignlookuptools.common.dataclasses.Address(attn=None, line1=None, line2=None, line3=None, city=None, state=None, zip=None, country=None, country_code=None)`  
Represents a mailing address

**attn:** `Optional[str] = None`  
Attention address line, this line should be prepended to the address

**line1:** `Optional[str] = None`  
address line 1

**line2:** `Optional[str] = None`  
address line 2

**line3:** `Optional[str] = None`  
address line 3

**city:** `Optional[str] = None`  
city

**state(USA Only):** `Optional[str] = None`  
state (USA Only)

**zip:** `Optional[str] = None`  
Zip/postal code

**country:** `Optional[str] = None`  
country name for the QSL mailing address

**country\_code:** `Optional[int] = None`  
dxcc entity code for the mailing address country

**class** `callsignlookuptools.common.dataclasses.Name(first=None, name=None, nickname=None, formatted_name=None)`  
Represents a name

```

first: Optional[str] = None
    first name(s)

name: Optional[str] = None
    last name or full name

nickname: Optional[str] = None
    A different or shortened name used on the air

formatted_name: Optional[str] = None
    Combined full name and nickname in the format used by QRZ. This format is subject to change.

class callsignlookuptools.common.dataclasses.Trustee(callsign=None, name=None)
    Represents a club callsign trustee (USA only)

    callsign: Optional[str] = None
        trustee callsign

    name: Optional[str] = None
        trustee name

class callsignlookuptools.common.dataclasses.Qsl(info=None, bureau_info=None,
    eqsl=<QslStatus.UNKNOWN: None>,
    lotw=<QslStatus.UNKNOWN: None>,
    mail=<QslStatus.UNKNOWN: None>,
    bureau=<QslStatus.UNKNOWN: None>)

    Represents information about QSL methods

    info: Optional[str] = None
        info about QSLing, e.g. QSL manager info

    bureau_info: Optional[str] = None
        info about QSLing via bureau

    eqsl: callsignlookuptools.common.enums.QslStatus = None
        whether eQSL is accepted

    lotw: callsignlookuptools.common.enums.QslStatus = None
        whether Logbook of the World QSL is accepted

    mail: callsignlookuptools.common.enums.QslStatus = None
        whether direct mail QSL is accepted

    bureau: callsignlookuptools.common.enums.QslStatus = None
        whether bureau QSL is accepted

class callsignlookuptools.common.dataclasses.Bio(size=None, updated=None)
    Represents metadata for a QRZ bio

    size: Optional[int] = None
        approximate size in bytes

    updated: Optional[datetime.datetime] = None
        when the bio was last updated

class callsignlookuptools.common.dataclasses.Image(url=None, size=None, height=None,
    width=None)

    Represents an image

    url: Optional[str] = None
        image url

```

**size: Optional[int] = None**  
image size in bytes

**height: Optional[int] = None**  
image height in pixels

**width: Optional[int] = None**  
image width in pixels

```
class callsignlookuptools.common.dataclasses.SocialMedia(website=None, jabber=None, icq=None,
                                                         msn=None, skype=None, facebook=None,
                                                         twitter=None, google_plus=None,
                                                         youtube=None, linkedin=None,
                                                         flickr=None, vimeo=None)
```

represents social media info

**website: Optional[str] = None**  
website url

**jabber: Optional[str] = None**  
Jabber username

**icq: Optional[str] = None**  
ICQ number

**msn: Optional[str] = None**  
MSN username

**skype: Optional[str] = None**  
Skype username

**facebook: Optional[str] = None**  
Facebook profile url

**twitter: Optional[str] = None**  
Twitter profile url

**google\_plus: Optional[str] = None**  
Google+ profile url

**youtube: Optional[str] = None**  
YouTube channel url

**linkedin: Optional[str] = None**  
LinkedIn profile url

**flickr: Optional[str] = None**  
Flickr profile url

**vimeo: Optional[str] = None**  
Vimeo profile url

```
class callsignlookuptools.common.dataclasses.Timezone(utc_offset=None, us_timezone=None,
                                                         observes_dst=None)

    Timezone(utc_offset: Optional[str] = None, us_timezone: Optional[str] = None, observes_dst: Optional[bool]
    = None)
```



## 3.4 Enums

**enum** callsignlookuptools.common.enums.DataSource(*value*)

Describes the callsign data lookup source

Valid values are as follows:

CALLOOK

HAMQTH

QRZ

QRZCQ

**enum** callsignlookuptools.common.enums.Continent(*value*)

Represents a continent

Valid values are as follows:

AF

AN

AS

EU

NA

OC

SA

NONE

**enum** callsignlookuptools.common.enums.CallsignType(*value*)

Describes what kind of license the license holder has

Valid values are as follows:

CLUB

MILITARY

RACES

RECREATION

PERSON

NONE

**enum** callsignlookuptools.common.enums.LicenseClass(*value*)

Describes the class of a license

Valid values are as follows:

NOVICE

TECHNICIAN

TECHNICIAN\_PLUS

GENERAL

ADVANCED

EXTRA

**NONE**

**enum** callsignlookuptools.common.enums.**GeoLocSource**(*value*)

Describes where the lat/long data in a QrzCallsignData object comes from

Valid values are as follows:

**USER**

**GEOCODE**

**GRID**

**ZIP**

**STATE**

**DXCC**

**NONE**

**enum** callsignlookuptools.common.enums.**QslStatus**(*value*)

Describes whether a type of QSL is accepted

Valid values are as follows:

**YES**

**NO**

**UNKNOWN**

## INSTALLATION

callsignlookuptools requires Python 3.9 at minimum. Install by running:

```
# synchronous requests only
$ pip install callsignlookuptools

# asynchronous aiohttp only
$ pip install callsignlookuptools[async]

# both sync and async
$ pip install callsignlookuptools[all]

# enable the CLI
$ pip install callsignlookuptools[cli]
```

---

**Note:** If requests, aiohttp, or typer[all] and click-help-colors are installed another way, you will also have access to the sync, async, or command-line interface, respectively.

---



## **API SUPPORT**

Some of the supported callsign lookup APIs require accounts and/or paid subscriptions to be used.

Site	Requirements
<a href="#">QRZ</a>	QRZ account and <a href="#">XML Logbook Data</a> or QRZ Premium subscription
<a href="#">Callook</a>	None
<a href="#">HamQTH</a>	HamQTH account
<a href="#">QRZCQ</a>	QRZCQ account and <a href="#">QRZCQ Premium subscription</a>



## GETTING STARTED

Using CallsignLookupTools is designed to be very simple. The following examples show basic use of the library.

### 6.1 Sync

```
# import the sync client for the service you want to use
from callsignlookuptools import QrzSyncClient, CallsignLookupError

# instantiate the lookup client
# some clients require a username, password, or other arguments
lookup_client = QrzSyncClient(username="...", password="...")

# perform a search query
try:
    # this will be a CallsignData object
    lookup_result = lookup_client.search("W1AW")
    # if an error occurs while performing the query,
    # it will raise a CallsignLookupError
except CallsignLookupError as e:
    print(e)
else:
    print(lookup_result)
```

### 6.2 Async

```
import asyncio

# import the async client for the service you want to use
from callsignlookuptools import QrzAsyncClient, CallsignLookupError

# instantiate the lookup client
# some clients require a username, password, or other arguments
lookup_client = QrzAsyncClient(username="...", password="...")

# for the async client, queries must be run inside coroutines
async def run_query():
    # perform a search query
```

(continues on next page)

(continued from previous page)

```
try:
    # this will be a CallsignData object
    lookup_result = await lookup_client.search("W1AW")
    # if an error occurs while performing the query,
    # it will raise a CallsignLookupError
except CallsignLookupError as e:
    print(e)
else:
    print(lookup_result)
    # if you're using the internally-generated session,
    # make sure to clean up
    await lookup_client.close_session()

# run the task
loop = asyncio.get_event_loop()
loop.run_until_complete(run_query())
```



## **LICENSE**

Copyright 2021 classabbyamp, 0x5c

Released under the BSD 3-Clause License. See LICENSE for the full license text.



## A

`address` (*callsignlookuptools.CallsignData* attribute), 16  
**Address** (class in *callsignlookuptools.common.dataclasses*), 18  
**ADVANCED** (*callsignlookuptools.common.enums.LicenseClass* attribute), 21  
**AF** (*callsignlookuptools.common.enums.Continent* attribute), 21  
**aliases** (*callsignlookuptools.CallsignData* attribute), 15  
**AN** (*callsignlookuptools.common.enums.Continent* attribute), 21  
**area\_code** (*callsignlookuptools.CallsignData* attribute), 17  
**AS** (*callsignlookuptools.common.enums.Continent* attribute), 21  
**attn** (*callsignlookuptools.common.dataclasses.Address* attribute), 18

## B

**bio** (*callsignlookuptools.CallsignData* attribute), 17  
**Bio** (class in *callsignlookuptools.common.dataclasses*), 19  
**born** (*callsignlookuptools.CallsignData* attribute), 17  
**bureau** (*callsignlookuptools.common.dataclasses.Qsl* attribute), 19  
**bureau\_info** (*callsignlookuptools.common.dataclasses.Qsl* attribute), 19

## C

**CALLOOK** (*callsignlookuptools.common.enums.DataSource* attribute), 21  
**CallookAsyncClient** (class in *callsignlookuptools*), 7  
**CallookSyncClient** (class in *callsignlookuptools*), 7  
**callsign** (*callsignlookuptools.CallsignData* attribute), 15  
**callsign** (*callsignlookuptools.common.dataclasses.Trustee* attribute), 19  
**CallsignData** (class in *callsignlookuptools*), 15

**CallsignLookupError** (class in *callsignlookuptools*), 18  
**city** (*callsignlookuptools.common.dataclasses.Address* attribute), 18  
**close\_session()** (*callsignlookuptools.CallookAsyncClient* method), 8  
**close\_session()** (*callsignlookuptools.HamQthAsyncClient* method), 10  
**close\_session()** (*callsignlookuptools.QrzAsyncClient* method), 6  
**close\_session()** (*callsignlookuptools.QrzCqAsyncClient* method), 12  
**CLUB** (*callsignlookuptools.common.enums.CallsignType* attribute), 21  
**continent** (*callsignlookuptools.CallsignData* attribute), 16  
**country** (*callsignlookuptools.common.dataclasses.Address* attribute), 18  
**country\_code** (*callsignlookuptools.common.dataclasses.Address* attribute), 18  
**county** (*callsignlookuptools.CallsignData* attribute), 16  
**cq\_zone** (*callsignlookuptools.CallsignData* attribute), 17

## D

**data\_source** (*callsignlookuptools.CallsignData* attribute), 15  
**db\_serial** (*callsignlookuptools.CallsignData* attribute), 17  
**district** (*callsignlookuptools.CallsignData* attribute), 16  
**dok** (*callsignlookuptools.CallsignData* attribute), 16  
**dxcc** (*callsignlookuptools.CallsignData* attribute), 16  
**DXCC** (*callsignlookuptools.common.enums.GeoLocSource* attribute), 22  
**Dxcc** (class in *callsignlookuptools.common.dataclasses*), 18  
**dxcc\_prefix** (*callsignlookuptools.CallsignData* attribute), 16

## E

`effective_date` (`callsignlookuptools.CallsignData` attribute), 16  
`email` (`callsignlookuptools.CallsignData` attribute), 17  
`eqsl` (`callsignlookuptools.common.dataclasses.Qsl` attribute), 19  
`EU` (`callsignlookuptools.common.enums.Continent` attribute), 21  
`expire_date` (`callsignlookuptools.CallsignData` attribute), 16  
`EXTRA` (`callsignlookuptools.common.enums.LicenseClass` attribute), 21

## F

`facebook` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20  
`fips` (`callsignlookuptools.CallsignData` attribute), 17  
`first` (`callsignlookuptools.common.dataclasses.Name` attribute), 18  
`flickr` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20  
`formatted_name` (`callsignlookuptools.common.dataclasses.Name` attribute), 19  
`frn` (`callsignlookuptools.CallsignData` attribute), 17

## G

`GENERAL` (`callsignlookuptools.common.enums.LicenseClass` attribute), 21  
`GEOCODE` (`callsignlookuptools.common.enums.GeoLocSource` attribute), 22  
`geoloc_src` (`callsignlookuptools.CallsignData` attribute), 17  
`google_plus` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20  
`grid` (`callsignlookuptools.CallsignData` attribute), 16  
`GRID` (`callsignlookuptools.common.enums.GeoLocSource` attribute), 22

## H

`HAMQTH` (`callsignlookuptools.common.enums.DataSource` attribute), 21  
`HamQthAsyncClient` (class in `callsignlookuptools`), 9  
`HamQthSyncClient` (class in `callsignlookuptools`), 8  
`height` (`callsignlookuptools.common.dataclasses.Image` attribute), 20

## I

`icq` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20  
`id` (`callsignlookuptools.common.dataclasses.Dxcc` attribute), 18  
`image` (`callsignlookuptools.CallsignData` attribute), 17  
`Image` (class in `callsignlookuptools.common.dataclasses`), 19  
`info` (`callsignlookuptools.common.dataclasses.Qsl` attribute), 19  
`iota` (`callsignlookuptools.CallsignData` attribute), 17  
`itu_zone` (`callsignlookuptools.CallsignData` attribute), 17

## J

`jabber` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20

## L

`last_action_date` (`callsignlookuptools.CallsignData` attribute), 16  
`latlong` (`callsignlookuptools.CallsignData` attribute), 16  
`lic_class` (`callsignlookuptools.CallsignData` attribute), 15  
`lic_codes` (`callsignlookuptools.CallsignData` attribute), 16  
`licensed` (`callsignlookuptools.CallsignData` attribute), 17  
`line1` (`callsignlookuptools.common.dataclasses.Address` attribute), 18  
`line2` (`callsignlookuptools.common.dataclasses.Address` attribute), 18  
`line3` (`callsignlookuptools.common.dataclasses.Address` attribute), 18  
`linkedin` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20  
`lotw` (`callsignlookuptools.common.dataclasses.Qsl` attribute), 19

## M

`mail` (`callsignlookuptools.common.dataclasses.Qsl` attribute), 19  
`MILITARY` (`callsignlookuptools.common.enums.CallsignType` attribute), 21  
`modified_date` (`callsignlookuptools.CallsignData` attribute), 16  
`msa` (`callsignlookuptools.CallsignData` attribute), 17  
`msn` (`callsignlookuptools.common.dataclasses.SocialMedia` attribute), 20

## N

NA (*callsignlookuptools.common.enums.Continent attribute*), 21

name (*callsignlookuptools.CallsignData attribute*), 16

name (*callsignlookuptools.common.dataclasses.Dxcc attribute*), 18

name (*callsignlookuptools.common.dataclasses.Name attribute*), 19

name (*callsignlookuptools.common.dataclasses.Trustee attribute*), 19

Name (*class in callsignlookuptools.common.dataclasses*), 18

new() (*callsignlookuptools.CallookAsyncClient class method*), 8

new() (*callsignlookuptools.HamQthAsyncClient class method*), 9

new() (*callsignlookuptools.QrzAsyncClient class method*), 6

new() (*callsignlookuptools.QrzCqAsyncClient class method*), 12

nickname (*callsignlookuptools.common.dataclasses.Name attribute*), 19

NO (*callsignlookuptools.common.enums.QslStatus attribute*), 22

NONE (*callsignlookuptools.common.enums.CallsignType attribute*), 21

NONE (*callsignlookuptools.common.enums.Continent attribute*), 21

NONE (*callsignlookuptools.common.enums.GeoLocSource attribute*), 22

NONE (*callsignlookuptools.common.enums.LicenseClass attribute*), 22

NOVICE (*callsignlookuptools.common.enums.LicenseClass attribute*), 21

## O

oblast (*callsignlookuptools.CallsignData attribute*), 16

OC (*callsignlookuptools.common.enums.Continent attribute*), 21

## P

page\_views (*callsignlookuptools.CallsignData attribute*), 17

password (*callsignlookuptools.HamQthAsyncClient property*), 10

password (*callsignlookuptools.HamQthSyncClient property*), 8

password (*callsignlookuptools.QrzAsyncClient property*), 6

password (*callsignlookuptools.QrzCqAsyncClient property*), 12

password (*callsignlookuptools.QrzCqSyncClient property*), 11

password (*callsignlookuptools.QrzSyncClient property*), 5

PERSON (*callsignlookuptools.common.enums.CallsignType attribute*), 21

plot (*callsignlookuptools.CallsignData attribute*), 16

prev\_call (*callsignlookuptools.CallsignData attribute*), 16

prev\_lic\_class (*callsignlookuptools.CallsignData attribute*), 16

## Q

QRZ (*callsignlookuptools.common.enums.DataSource attribute*), 21

QrzAsyncClient (*class in callsignlookuptools*), 6

QRZCQ (*callsignlookuptools.common.enums.DataSource attribute*), 21

QrzCqAsyncClient (*class in callsignlookuptools*), 11

QrzCqSyncClient (*class in callsignlookuptools*), 11

QrzSyncClient (*class in callsignlookuptools*), 5

qsl (*callsignlookuptools.CallsignData attribute*), 17

Qsl (*class in callsignlookuptools.common.dataclasses*), 19

qth (*callsignlookuptools.CallsignData attribute*), 16

query (*callsignlookuptools.CallsignData attribute*), 15

## R

RACES (*callsignlookuptools.common.enums.CallsignType attribute*), 21

raw\_data (*callsignlookuptools.CallsignData attribute*), 15

RECREATION (*callsignlookuptools.common.enums.CallsignType attribute*), 21

## S

SA (*callsignlookuptools.common.enums.Continent attribute*), 21

search() (*callsignlookuptools.CallookAsyncClient method*), 8

search() (*callsignlookuptools.CallookSyncClient method*), 7

search() (*callsignlookuptools.HamQthAsyncClient method*), 9

search() (*callsignlookuptools.HamQthSyncClient method*), 8

search() (*callsignlookuptools.QrzAsyncClient method*), 6

search() (*callsignlookuptools.QrzCqAsyncClient method*), 12

search() (*callsignlookuptools.QrzCqSyncClient method*), 11

search() (*callsignlookuptools.QrzSyncClient* method), 5  
 session (*callsignlookuptools.CallookAsyncClient* property), 8  
 session (*callsignlookuptools.CallookSyncClient* property), 7  
 session (*callsignlookuptools.HamQthAsyncClient* property), 10  
 session (*callsignlookuptools.HamQthSyncClient* property), 9  
 session (*callsignlookuptools.QrzAsyncClient* property), 7  
 session (*callsignlookuptools.QrzCqAsyncClient* property), 12  
 session (*callsignlookuptools.QrzCqSyncClient* property), 11  
 session (*callsignlookuptools.QrzSyncClient* property), 5  
 session\_key (*callsignlookuptools.HamQthAsyncClient* property), 10  
 session\_key (*callsignlookuptools.HamQthSyncClient* property), 9  
 session\_key (*callsignlookuptools.QrzAsyncClient* property), 7  
 session\_key (*callsignlookuptools.QrzCqAsyncClient* property), 12  
 session\_key (*callsignlookuptools.QrzCqSyncClient* property), 11  
 session\_key (*callsignlookuptools.QrzSyncClient* property), 5  
 size (*callsignlookuptools.common.dataclasses.Bio* attribute), 19  
 size (*callsignlookuptools.common.dataclasses.Image* attribute), 19  
 skype (*callsignlookuptools.common.dataclasses.SocialMedia* attribute), 20  
 social\_media (*callsignlookuptools.CallsignData* attribute), 17  
 SocialMedia (class in *callsignlookuptools.common.dataclasses*), 20  
 sondok (*callsignlookuptools.CallsignData* attribute), 16  
 start\_session() (*callsignlookuptools.CallookAsyncClient* method), 8  
 start\_session() (*callsignlookuptools.HamQthAsyncClient* method), 10  
 start\_session() (*callsignlookuptools.QrzAsyncClient* method), 7  
 start\_session() (*callsignlookuptools.QrzCqAsyncClient* method), 13  
 state (*callsignlookuptools.common.dataclasses.Address* attribute), 18  
 STATE (*callsignlookuptools.common.enums.GeoLocSource* attribute), 22

## T

TECHNICIAN (*callsignlookuptools.common.enums.LicenseClass* attribute), 21  
 TECHNICIAN\_PLUS (*callsignlookuptools.common.enums.LicenseClass* attribute), 21  
 timezone (*callsignlookuptools.CallsignData* attribute), 17  
 Timezone (class in *callsignlookuptools.common.dataclasses*), 20  
 trustee (*callsignlookuptools.CallsignData* attribute), 15  
 Trustee (class in *callsignlookuptools.common.dataclasses*), 19  
 twitter (*callsignlookuptools.common.dataclasses.SocialMedia* attribute), 20  
 type (*callsignlookuptools.CallsignData* attribute), 15

## U

uls\_url (*callsignlookuptools.CallsignData* attribute), 17  
 UNKNOWN (*callsignlookuptools.common.enums.QslStatus* attribute), 22  
 updated (*callsignlookuptools.common.dataclasses.Bio* attribute), 19  
 url (*callsignlookuptools.CallsignData* attribute), 17  
 url (*callsignlookuptools.common.dataclasses.Image* attribute), 19  
 USER (*callsignlookuptools.common.enums.GeoLocSource* attribute), 22  
 useragent (*callsignlookuptools.HamQthAsyncClient* property), 10  
 useragent (*callsignlookuptools.HamQthSyncClient* property), 9  
 useragent (*callsignlookuptools.QrzAsyncClient* property), 7  
 useragent (*callsignlookuptools.QrzCqAsyncClient* property), 13  
 useragent (*callsignlookuptools.QrzCqSyncClient* property), 11  
 useragent (*callsignlookuptools.QrzSyncClient* property), 6  
 username (*callsignlookuptools.CallsignData* attribute), 17  
 username (*callsignlookuptools.HamQthAsyncClient* property), 10  
 username (*callsignlookuptools.HamQthSyncClient* property), 9  
 username (*callsignlookuptools.QrzAsyncClient* property), 7  
 username (*callsignlookuptools.QrzCqAsyncClient* property), 13  
 username (*callsignlookuptools.QrzCqSyncClient* property), 11

username (*callsignlookuptools.QrzSyncClient* property),  
[6](#)

## V

vimeo (*callsignlookup-  
tools.common.dataclasses.SocialMedia* at-  
tribute), [20](#)

## W

website (*callsignlookup-  
tools.common.dataclasses.SocialMedia* at-  
tribute), [20](#)

width (*callsignlookuptools.common.dataclasses.Image*  
attribute), [20](#)

## Y

YES (*callsignlookuptools.common.enums.QslStatus*  
attribute), [22](#)

youtube (*callsignlookup-  
tools.common.dataclasses.SocialMedia* at-  
tribute), [20](#)

## Z

zip (*callsignlookuptools.common.dataclasses.Address*  
attribute), [18](#)

ZIP (*callsignlookuptools.common.enums.GeoLocSource*  
attribute), [22](#)