ACLED CAST API

User Guide

Richard Holmes

https://ampersandstudio.uk/

March 2023

Contents

ACLED CAST API

Version 1.0

Introduction

The following document highlights the basic steps for interacting with the ACLED Conflict Alert System (CAST). The API is RESTful in nature and is accessed via Basic HTTP(S) authentication.

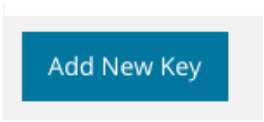
API Access Detail

The full URL for accessing the API is https://api.acleddata.com/cast/{command}, where "command" represents the action to be executed. See below for details regarding possible data types and usages.

API Developer Account

You must register for an API key in order to access the API. To register please visit https://developer.acleddata.com/**.**

Please note you should register from the device you intend to access the API from as your IP address will be linked to your account and may determine your access to the API. On completing the registration form you may be required to wait for approval or may have limited access to the API, until your account has been approved. The dashboard will display any notifications that relate to your account, that should be completed before proceeding to create an API Key. Once you are eligible to create a key you will see the following button is active, within the dashboard.



Clicking the button will generate a random API key for your account. **You will only be shown the key one time and must copy and save it in a secure location**. If you fail to copy the key you will need to revoke the active key and generate a new key. This does not affect your usage of the API or reset your account usage limit.

Sample API Calls and Responses

API calls may be made in any standard browser or using any programmatic language that is capable of making HTTP requests. The samples below demonstrate the URL to be called to make the request.

The following points should be noted:

- All requests will be denied without a key and email address.
- This API only uses the GET or POST request. A GET request is advised whereby all data sent will be contained within standard Query String parameter formats and URLencoded.
- All responses from the API shall be formatted as JSON unless specifically requested in either XML, CSV or TXT format.
- TXT format returns a plain text csv string.
- A limit of 5000 lines of data will be returned by default for CAST data types. Larger requests may be made, however.
- All fields, specific to the data type, will be returned by default. Reduced field lists can be requested for some data types.
- CAST data is returned in date order DESC (starting with the latest).

Response Format

By default the response is returned in JSON format but it's possible to return the response in XML, CSV and TXT format too. In order to return another format you simply add the relevant extension to the end of the command name so the query would look like the following:

Format	HTTP Request Format	MIME Type
JSON	https://api.acleddata.com/cast/{command}	application/json
XML	https://api.acleddata.com/cast/{command}.xml	text/xml
CSV	https://api.acleddata.com/cast/{command}.csv	text/csv
TXT	https://api.acleddata.com/cast/{command}.txt	text/plain

CAST

This data call returns the main dataset

Commands

Read

In order to retrieve the data you must make a GET or POST request to the following URL:

https://api.acleddata.com/cast/read?key={api_key}&email={email}

Returned Data (json only)

Attribute Name	Type	Description
status	int	A number representing the request status
success	boolean	A boolean representation on the success of the call
last_update	int	The number of hours since the last update to the data
count	int	The number of data rows returned
messages	array	An array of information messages that may require future action
data	array	The rows of data returned. For details of attributes returned in each row, see below.
filename	string	The filename that will be used for csv calls
error	array	The details of the error with a status as an integer and message as a string

Returned Data (json, xml, txt, csv)

Attribute Name	Type	Description
country	string	The name of the country
admin1	string	The name of the first-level administrative division
month	string	The month of events
year	int	The year of events
total_forecast	int	Total number of events forecasted
battles_forecast	int	Total number of *battles* events forecasted

Attribute Name	Type	Description
erv_forecast	int	Total number of *explosions/remote violence* events forecasted
vac_forecast	int	Total number of *violence against civilians* events forecasted
total_observed	int	Total number of events observed. This column will be populated once the given month has passed
battles_observed	int	Total number of *battles* events observed. This column will be populated once the given month has passed
erv_observed	int	Total number of *explosions/remote violence* events observed. This column will be populated once the given month has passed
vac_observed	int	Total number of *violence against civilians* events observed. This column will be populated once the given month has passed
timestamp	int or date	The unix timestamp (or date stamp) this data entry was last updated

Query Filters

Each field can be searched to filter the returned data. By default each field will search by = or LIKE based on the table below. This can be changed by sending a new query string name value pair, where the name has '_where' appended to it. The following table shows the default query type that will be used by each field. **The key and email address query must be included in all requests to indicate that you have registered your use of our API.**

Query Name	Type	Query String
key	**=**	**{api_key}**
email	**=**	**{email address associated with key}**
country	LIKE	?country={text}
admin1	LIKE	?admin1={text}
month	LIKE	?month={text}
year	=	?year={yyyy}
total_forecast	=	?total_forecast={text}
battles_forecast	=	?battles_forecast={text}
erv_forecast	=	?erv_forecast={text}
vac_forecast	=	?vac_forecast={text}
total_observed	=	?total_observed={text}
battles_observed	=	?battles_observed={text}
erv_observed	=	?erv_observed={text}
vac_observed	=	?vac_observed={text}
timestamp	>=	?timestamp={number or yyyy-mm-dd}

References

- & All LIKE queries will include a wildcard before and after the entered text.
- & Multiple queries can be searched with name/value pairs separated by &. Each field will be searched using AND so all arguments must match for data to be returned.
- & More complex queries can be searched to include the OR clause. See Complex Queries below.

To change the default query type you can add an additional name/value pair using the query name and appending '_where' to the query name. The query value could be LIKE or %3D for '='. Additional types of '<', '>' and 'BETWEEN may also be used, representing less than, greater than and between. The between requires the query name value to separate the 2 values with a |.

Example:
?key={api_key}&email={email address}&month={text}&month_where=%3D
?key={api_key}&email={email address}&timestamp={yyyy-mm-dd|yyyy-mm-dd}
&timestamp_where=BETWEEN

Limit Query & Pagination

By default there is a limit of 5000 rows of data returned. You can use the limit query name to change the default number. Setting limit as 0 will return all relevant data. It is likely returning all data will cause a timeout error and we therefore recommend using the page query instead. Each page will return 5000 rows of data.

Example:
?key={api_key}&email={email address}&limit=1000 will return 1000 rows of data.
?key={api_key}&email={email address}&page=1 will return the first 5000 rows of data.
?key={api_key}&email={email address}&page=2 will return the next 5000 rows of data.

Limit Fields Returned

By default all fields will be returned for each line of data. You can use the fields query name to change the field items returned. Multiple fields can be requested by separating each one with a pipe (|).

Example:
?key={api_key}&email={email address}&fields=month will return just the month field.
?key={api_key}&email={email address}&fields=month|year will return the month and years data for each row.

Complex Queries

By default all fields must match for the data to be returned. In some instances more complex queries may be required to use the OR clause. This is possible by separating the fields to join, by OR, with :OR: text. The main query item will be the first item in the join, followed by the other items split with :OR: . These can be used with other queries too.

Example:
?key={api_key}&email={email address}&{fieldname}={text}:OR:{fieldname2}={text2}:OR:{fieldname3}={text3} will return data where field = text OR field2 = text2 OR field3 = text3.

?key={api_key}&email={email address}&{fieldname}={text}:OR:{fieldname2}={text2}&country={country} will return data where field = text OR field2 = text2 AND country = country.

All items wrapped in {} must be replaced with the actual fields or text required. The curly brackets must also be removed from the query.