
WeatherAlerts Documentation

Release 0.5.x

Zeb Palmer

Mar 28, 2017

Contents

1	Documentation Contents	3
1.1	About WeatherAlerts	3
1.2	National Weather Service Alert Data	4
1.3	Using WeatherAlerts	5
1.4	Webservice	5
1.5	Code Documentation	7
1.6	Release Changes	10
	HTTP Routing Table	13
	Python Module Index	15

WeatherAlerts a python package that pulls in the National Weather Service (NWS) Common Alerting Protocol (CAP) Emergency / Severe Weather Alerts feed, parses it and provides interaction with currently active alerts.

Documentation Contents

About WeatherAlerts

This python module started as part of another project of mine. But since it is more useful as a standalone module, I've decided to move it to it's own project and open source it.

Since this project gets it's data from the National Weather Service XML/CAP feed, it's free and straight from the source. Other library's I've seen (or written) get data from 3rd parties that require an API key which in many cases requires a subscription or imposes use restrictions.

This code is provided under LGPLv3 as of version 0.5.x (see LICENSE.txt). If you do make improvements, please contribute back to this project. You can submit a git pull request or email me: zeb@zebpalmer.com

This project lives at github.com/zebpalmer/WeatherAlerts

Install

You can download and install the current stable version via PIP by runing: `pip install -U weatheralerts`

Alternativly you can download and install directly from the source code on github.

Bugs & Feature Requests

When you find one, please report it in the [issue tracker](#)

Goals

Use cases that I am considering in the development of WeatherAlerts.

- Simple command line tool for checking active, local alerts.
- Packaged module to call from other programs

- Daemon to run and notify alerts as they come in
- Nagios monitoring plugging
- A web service that given various paramaters will return json or raw text summaries of the requested data.
- Would love to see someone write a KDE plasmoid/widget that would pop up alerts

Author

This program is maintained by Zeb Palmer, a Linux Systems Engineer and Professional Photographer who writes a bit of python at work and play. Circle me on Google Plus [zebpalmer.com/](https://plus.google.com/zebpalmer.com/) and see my other work at ZebPalmer.com

Contact

There are several ways you can contact me or otherwise get help beyond the documentation.

Bug Reports & Feature Requests Please submit via the projects issue tracker on github <https://github.com/zebpalmer/WeatherAlerts/issues>

Random Chatter Circle me on Google+ [Zeb Palmer Google Plus](https://plus.google.com/+ZebPalmer)

Follow me on Twitter [@zebpalmer](https://twitter.com/zebpalmer)

Website For info on other projects, see my website, [Zeb Palmer](http://ZebPalmer.com)

National Weather Service Alert Data

Overview

This package pulls in ‘near realtime’ alert data from the National Weather Service (NWS) CAP index feed. It’s an ATOM/XML feed with additional CAP 1.1 defined fields.

Caveats

SAME Codes and Geo Location

Currently this project makes extensive use of SAME codes, it’s been noted that ‘not all NWS products are issued with a SAME Code’. However, ‘County/Zone codes are provided for all CAP 1.1 messages.’ I’ve noticed that a lot of Alaskan alerts do not ship with SAME codes, some don’t appear to ship with any geocodes... Most lower 48 alerts do, in fact, I haven’t seen one that didn’t. For this reason though, we’ll be incorporating county/zone codes and Storm based location information in the near future.

Near Realtime

The NWS states that the feed is updated in ‘near real time’, elsewhere it states that the feed is updated ‘roughly every two minutes’. It appears to me that it’s somewhere in the middle, certainly being updated often enough for non immediate life threatening alerts. If you live in an area prone to flash floods, tornados, or Tsunami’s, you should be relying on the National Weather Service Weather Radio for your primary alerting method.

Using WeatherAlerts

Simple example

These are three examples of the simplest implementation, import the WeatherAlerts class, create an instance requesting alerts for an area based on one or more SAMECODES or by requesting an entire State.

```
from weatheralerts import WeatherAlerts

# Alerts by a Samecode
nws = WeatherAlerts(samecodes='016027')
for alert in nws.alerts:
    print alert.title

# Alerts for a list of Samecodes
nws = WeatherAlerts(samecodes=['016027', '016001', '016073', '016075'])
for alert in nws.alerts:
    print alert.title

# Alerts for a State
nws = WeatherAlerts(state='ID')
for alert in nws.alerts:
    print "{0}: {1}".format(alert.areadesc, alert.title)
```

Webservice

Note: This feature is currently only in the 'dev' branch of WeatherAlerts and has not been released to PyPI yet. It will be added to the 0.5.0 release which should be available soon. This documentation is provided as a preview. You can try this out by downloading and installing the development branch from github.

The WeatherAlerts webservice allows you to setup a single webservice and request Severe Weather and other Emergency Alerts from multiple clients (data is provided as JSON). This can give you some flexibility in use as well as reducing the requests you make to the NWS servers if you are using multiple clients.

I will be offering a Live version of this webservice for experimentation in the near future, details of which will be documented here.

Start the Webservice

To setup a webservice that provides data for the entire US, run the code below.

```
from weatheralerts import WebApp
nws_ws = WebApp()
nws_ws.start()
```

If however, you know you will only be requesting data for one state, you can save a few electrons by specifying the state.

```
from weatheralerts import WebApp
nws_ws = WebApp(state='ID')
nws_ws.start()
```

Webservice API Documentation

GET /all

Will return json data for all alerts on the feed.

Example request:

```
GET /all HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: text/javascript

{
  "webservice": {
    "status": true,
    "disclaimer": "Don't rely on this for anything important, it's for_
↪ experimentation purposes only."
  },
  "alerts": []
}
```

GET /samecodes/ (*samecodes*)

Will return json data for alerts that match the specified samecode(s). When requesting data for multiple samecodes, separate them with commas, no spaces.

Example request:

```
GET /samecodes/012065,013281 HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

Example response:

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: text/javascript

{
  "webservice": {
    "status": true,
    "disclaimer": "Don't rely on this for anything important, it's for_
↪ experimentation purposes only."
  },
  "alerts": [
    {
      "zonecodes": [],
      "updated": "2013-03-28T23:29:00-04:00",
      "msgtype": "Alert",
      "link": "http://alerts.weather.gov/cap/wwacapget.php?x=FL124EF51C78C4.
↪ FloodWarning.124EF52B3A80FL.TAEFLSTAE.285023120b4e86a12ca32387f953554e",
      "event": "Flood Warning",
      "category": "Met",
      "severity": "Moderate",
      "effective": "2013-03-28T23:29:00-04:00",
    }
  ]
}
```

```
        "title": "Flood Warning issued March 28 at 11:29PM EDT until March 29_
↪at 8:00PM EDT by NWS",
        "summary": "...THE FLOOD WARNING CONTINUES FOR THE FOLLOWING RIVERS_
↪IN FLORIDA... AUCILLA RIVER AT LAMONT (US 27) AFFECTING JEFFERSON...MADISON AND_
↪TAYLOR COUNTIES..",
        "areadesc": "Jefferson; Madison; Taylor",
        "expiration": "2013-03-29T20:00:00-04:00",
        "published": "2013-03-28T23:29:00-04:00",
        "samecodes": [
            "012065",
            "012079",
            "012123"
        ],
        "urgency": "Expected"
    }
}
```

Code Documentation

Below you'll find the documentation for the various classes and methods in WeatherAlerts.

Note: This page is dynamically generated from the documentation written within the code

WeatherAlerts

File Information

Project Home: <http://github.com/zebpalmer/WeatherAlerts>

Original Author: Zeb Palmer <http://www.zebpalmer.com>

Documentation: <http://weatheralerts.readthedocs.org>

License: MIT - full text included in LICENSE.txt

Code Documentation

class WeatherAlerts (*state=None, samecodes=None, load=True, cachetime=3*)

WeatherAlerts object that controls interaction with the NWS CAP alerts feed as well as various geo data sources. Most interaction from users, scripts, etc will be through the api provided by this *WeatherAlerts* class. So, as we approach a more stable project, the API in this class will also become more stable.

- Defaults to National Feed, it can be quite large at times, you probably don't want to parse it very often.
- Set *state* to see all alerts on your state feed.
- For local alerts only, set *samecodes* to a single samecode string, or list of samecode strings.
- *cachetime* is set in minutes, default is 3.

alerts

returns the alerts list. If samecode(s) are specified when the WeatherAlerts object is created, this will only

return alerts for those samecodes. If no samecodes were given, it'll return all alerts for the state if one was specified otherwise for the entire U.S.

county_state_alerts (*county, state*)

Given a county and state, return alerts

event_state_counties ()

DEPRECATED: this will be moved elsewhere or dropped in the near future, stop using it. Return an event type and it's state(s) and counties (consolidated)

load_alerts ()

NOTE: use refresh() instead of this, if you are just needing to refresh the alerts list Gets raw xml (cap) from the Alerts feed, throws it into the parser and ends up with a list of alerts object, which it stores to self._alerts

refresh (*force=False*)

Refresh the alerts list. set *force* to True to force pulling a new list from the NWS, otherwise it'll only pull a new list if the cached copy is expired. (see cachetime)

samecode_alerts (*samecode*)

Returns alerts for a (single) SAME geocode. Only useful if you didn't specify samecodes when the WeatherAlerts object was created.

class GeoDB

Interact with samecodes data will be adding additional data (zip code lookup) in the future.

getfeedscope (*geocodes*)

Given multiple SAME codes, determine if they are all in one state. If so, it returns that state. Otherwise return 'US'. This is used to determine which NWS feed needs to be parsed to get all alerts for the requested SAME codes

getstate (*geosame*)

Given a SAME code, return the state that SAME code is in

location_lookup (*req_location*)

returns full location given samecode or county and state. Returns False if not valid.

currently locations are a dictionary, once other geo data is added, they will move to a location class/obj

lookup_county_state (*samecode*)

Given a samecode, return county, state

lookup_samecode (*local, state*)

Given County, State return the SAME code for specified location. Return False if not found

class SameCodes

Is used to download, parse and cache the SAME codes data from the web.

All interaction with the SAME codes data should be done with the GeoGB object

reload ()

force refresh of Same Codes

samecodes

public method to return the same codes list

class AlertsFeed (*state='US', maxage=3*)

Fetch the NWS CAP/XML Alerts feed for the US or a single state if requested if an instance of the GeoDB class has already been created, you can pass that as well to save some processing This will cache the feed (in local tempdir) for up to 'maxage' minutes

raw_cap (*refresh=False*)

Raw xml(cap) of the the feed. If a valid cache is available it is used, else a new copy of the feed is grabbed

Note: you can force refresh here, if you do, don't also manually call refresh

refresh ()

NOTE: You probably don't want to call this... This does not update the alerts loaded in the WeatherAlerts object, only the underlying feed. This is only used internally now and as such, will likely be deprecated soon. Please call *WeatherAlerts.refresh()* instead.

class CapParser (*raw_cap, geo=None*)

Parses the xml from the alert feed, creates and returns a list of alert objects.

FIXME: This is slow, messy, and painful to look at. I'll be totally rewriting it shortly.

get_alerts ()

Public method that parses

build_target_areas (*entry*)

Cleanup the raw target areas description string

class Alert (*cap_dict*)

Create an alert object with the cap dict created from cap xml parser.

This object won't be pretty... it's mostly a bunch of property methods to sanitize and muck around with the raw cap data. Using individual properties and methods instead of a special getattr so that we can more easily standardize the Alert API. This may be revisited in the future as the project becomes more stable.

areadesc

A more generic area description

category

Category of alert i.e. Met, Civil, etc

effective

Effective timestamp of the alert (datetime object)

event

alert event type

expiration

Expiration of the alert (datetime object)

link

msgtype

published

Published timestamp of the alert (datetime object)

samecodes

samecodes for the alert area

severity

Severity of alert i.e. minor, major, etc

summary

Alert summary

title

Alert title

updated

Last update to the alert (datetime object)

urgency

Alert urgency

zonecodes

UCG codes for the alert area (these are sometimes referred to as county codes, but that's not quite accurate)

Release Changes

Current Development is on 5.x

0.5.0rc * 100% Test Coverage * building python3 version at install, no longer maintaining separate code * rewrite * improved API organization * improved documentation * relicensed under MIT * reworked refresh logic (thanks to Michael W. for bug report) * Ignore erroneous data from NWS in the FIPS6 Fields * force lower case url paramaters to avoid 301 redirect by nws

Older Versions

Versions prior to 0.5.x are no longer supported. It is suggest you test and upgrade when possible.

v0.4.9

- Last of the 4.x branch
- rather large changes to classes
- still running v0.4.5 in python2 installs will update that in 0.4.8 which will begin a release canidate for v0.5.0

v0.4.5

- minor packaging changes
- added initial support for object reload based on age

v0.4.4

- Reorganized for easier packaging
- Supporting both Python2 and Python3 in the installer
- tox automated virtenv & tests for python 2.6, 2.7, 3.2
- Added command line monitoring script

v0.4.1

- Changing project name to better fit PyPi
- Packaging as an installable module

v0.4

- Added basic nose test script
- Refactored classes
- Added Alerts() class
- Optional json output (getting ready for web service/api)
- Various code cleanup/improvements

v0.3.1

- bugfix only

v0.3:

- refactored nagios plugin, it now uses (only) SAME code(s)
- Moved SAME code related methods to new class which can be used without parsing an alerts feed.

v0.2:

- moved master branch to python 3.x
- maintaining python2.x branch

v0.1:

- First tagged release
- genindex
- search

This module is maintained by Zeb Palmer, a Linux Systems Engineer and Professional Photographer who writes a bit of python at work and play. See my other work at [Zeb Palmer](#)

HTTP Routing Table

/all

GET /all,6

/samecodes

GET /samecodes/(samecodes),6

a

alert, 9

c

cap, 9

f

feed, 8

g

geo, 8

w

weather_alerts, 7

weatheralerts, 7

A

Alert (class in alert), 9
alert (module), 9
alerts (WeatherAlerts attribute), 7
AlertsFeed (class in feed), 8
areadesc (Alert attribute), 9

B

build_target_areas() (in module cap), 9

C

cap (module), 9
CapParser (class in cap), 9
category (Alert attribute), 9
county_state_alerts() (WeatherAlerts method), 8

E

effective (Alert attribute), 9
event (Alert attribute), 9
event_state_counties() (WeatherAlerts method), 8
expiration (Alert attribute), 9

F

feed (module), 8

G

geo (module), 8
GeoDB (class in geo), 8
get_alerts() (CapParser method), 9
getfeedscope() (GeoDB method), 8
getstate() (GeoDB method), 8

L

link (Alert attribute), 9
load_alerts() (WeatherAlerts method), 8
location_lookup() (GeoDB method), 8
lookup_county_state() (GeoDB method), 8
lookup_samecode() (GeoDB method), 8

M

msgtype (Alert attribute), 9

P

published (Alert attribute), 9

R

raw_cap() (AlertsFeed method), 8
refresh() (AlertsFeed method), 9
refresh() (WeatherAlerts method), 8
reload() (SameCodes method), 8

S

samecode_alerts() (WeatherAlerts method), 8
samecodes (Alert attribute), 9
SameCodes (class in geo), 8
samecodes (SameCodes attribute), 8
severity (Alert attribute), 9
summary (Alert attribute), 9

T

title (Alert attribute), 9

U

updated (Alert attribute), 9
urgency (Alert attribute), 9

W

weather_alerts (module), 7
WeatherAlerts (class in weather_alerts), 7
weatheralerts (module), 7

Z

zonecodes (Alert attribute), 10