WMS visualisation from THREDDS (by KML and Python)

This pages describes how to make use of the WMS service by the THREDSS server (be sure your THREDDS server has enable WMS output, this has to be configurated).

This example describes how to create a time enabled dynamic KML visualising NetCDF's by means of WMS. A series of netCDF in time placed on a THREDDS server describe the growth of Plaice (a fish) in time in the North Sea.
First the dependencies:

```
import simplekml
import netCDF4
import urllib2
from bs4 import BeautifulSoup
from datetime import datetime, timedelta
```

There are two variables in this example:

- base html describing the contents of the folder on the THREDDS server

- output KML file

Using the urllib2 and BeautiffulSoup (4) libraries the html with the contents of the folder is used to create a list of available netCDF's. The daynum (in this case) is derived from the filename.

```
fhtml = 'http://opendap.deltares.nl/thredds/catalog/opendap/imares/plaice_large_1989/catalog.html'
file = urllib2.urlopen(fhtml)
data = file.read()
soup = BeautifulSoup(data)

# loop over the contents of the array
t = []
for link in soup.find_all('a'):
        aref = link.get('href')
        if aref.find('catalog.html?dataset=varopendap/') != -1:
                if aref.find('.nc') != -1:
                daynum = aref.split('_day_')[1].replace('.nc','')
                adate = datetime(1989,1,1)+(int(daynum)-1)*timedelta(hours=24)
                anc = aref.replace('catalog.html?dataset=varopendap/','http://opendap.deltares.nl/thredds/wms
/opendap/')
                t.append([adate,anc])
# sort the array
t.sort()
```

Use the first array to extract the geographic extent

```
# take the first netCDF and extract geografic extent from it

anc = t[0][1].replace('/wms/','/dodsC/')
ds = netCDF4.Dataset(anc)
latarr = ds.variables["lat"][:]
lonarr = ds.variables["lon"][:]
ds.close()

# create the kml
kml = simplekml.Kml()
doc = kml.newfolder(name = 'Growth of Plaice in 1989')
for i in range(len(t)):
        overlay = doc.newgroundoverlay(name="growth at{date}".format(date=t[i][0].isoformat()))
        alink = t[i][1]
        aref = '{anc}?VERSION=1.1.1&REQUEST=GetMap&bbox={lonmin},{latmin},{lonmax},{latmax}&SRS=EPSG%
3A4326&WIDTH=512&HEIGHT=512&LAYERS=Band1&STYLES=boxfill/sst_36&TRANSPARENT=TRUE&FORMAT=image
/gif&COLORSCALERANGE=-0.1,0.2'.format(anc=alink,lonmin=str(lonarr.min()),latmin=str(latarr.min()),lonmax=str
(lonarr.max()),latmax=str(latarr.max()))
        overlay.icon.href = aref
        overlay.latlonbox.west = lonarr.min()
        overlay.latlonbox.south = latarr.min()
        overlay.latlonbox.east = lonarr.max()
        overlay.latlonbox.north = latarr.max()
        overlay.icon.refreshmode ='onStop'
        overlay.icon.viewboundscale = 0.75
        overlay.timestamp = simplekml.TimeStamp(t[i][0].isoformat()+'Z')

#save the kml
kml.save(r'd:\test\GrowthOfPlaice.kml')
```

Playing a little bit with the STYLES and the COLORRANGE variables gives you the opportunity to style your kml.