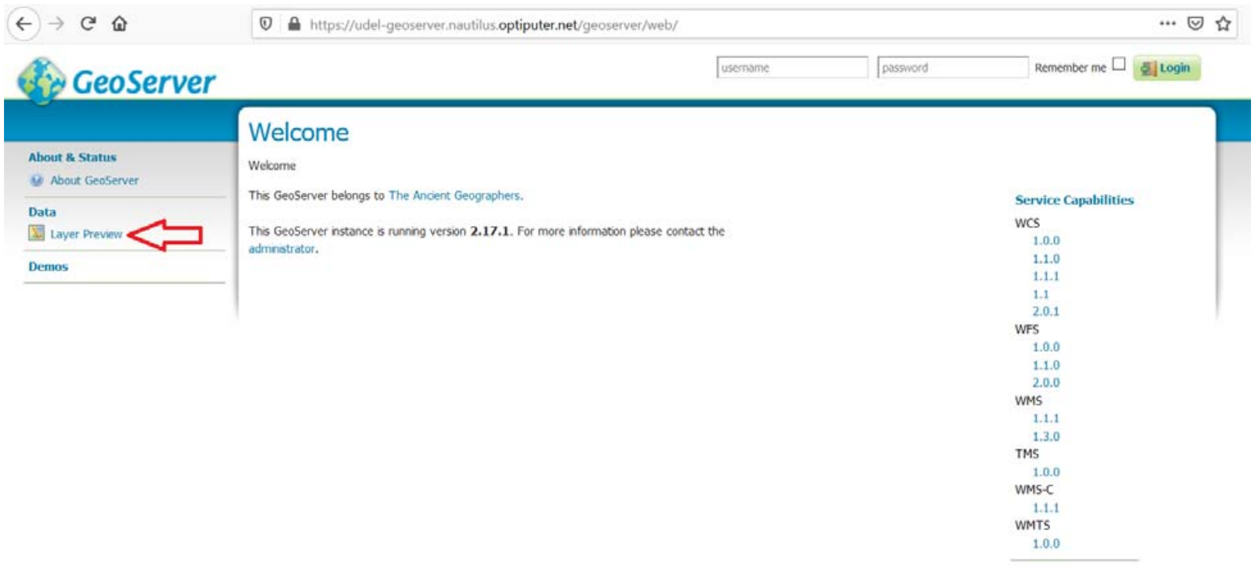
Project Wicced Data Access

Geoserver

A geoserver instance is available for visualizing data via your web browser. You can access it at:
https://udel-geoserver.nautilus.optiputer.net/geoserver/

1. Navigate to https://udel-geoserver.nautilus.optiputer.net/geoserver/ and click the 'Layer Preview' link in the left hand menu



https://udel-geoserver.nautilus.optiputer.net/geoserver/web/

GeoServer

username password Remember me Login

About & Status
About GeoServer
Data
Layer Preview
Demos

Welcome

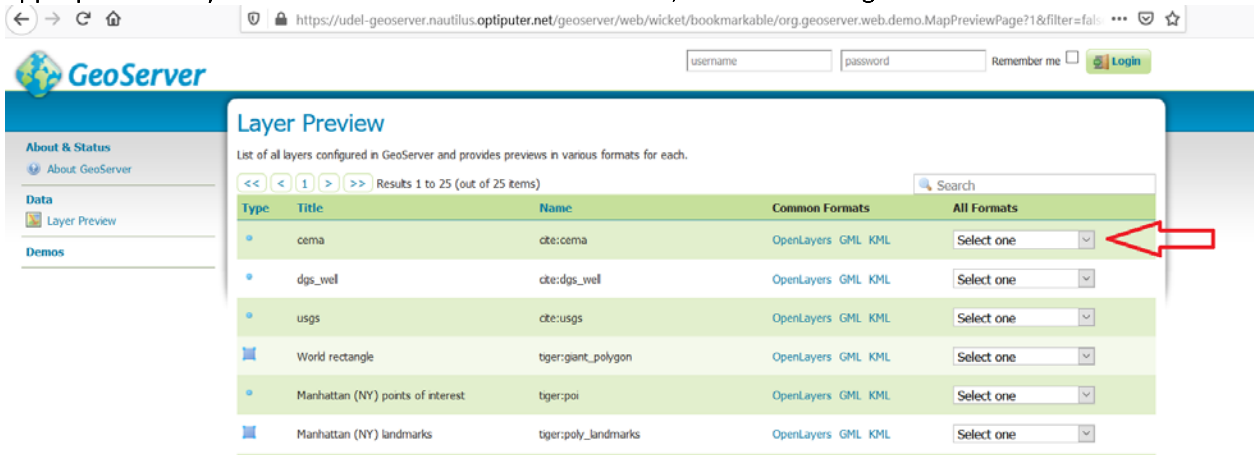
Welcome

This GeoServer belongs to The Ancient Geographers.

This GeoServer instance is running version **2.17.1**. For more information please contact the administrator.

Service Capabilities
WCS
1.0.0
1.1.0
1.1.1
1.1
2.0.1
WFS
1.0.0
1.1.0
2.0.0
WMS
1.1.1
1.3.0
TMS
1.0.0
WMS-C
1.1.1
WMTS
1.0.0

2. Once on the Layer Preview page, click on the drop down menu under the 'All Formats' table heading. A list of available data formats the server can handle are returned. Select the appropriate one you want to work with. For this tutorial, we will be using 'GeoJSON'



https://udel-geoserver.nautilus.optiputer.net/geoserver/web/wicket/bookmarkable/org.geoserver.web.demo.MapPreviewPage?1&filter=fals

GeoServer

username password Remember me Login

About & Status
About GeoServer
Data
Layer Preview
Demos

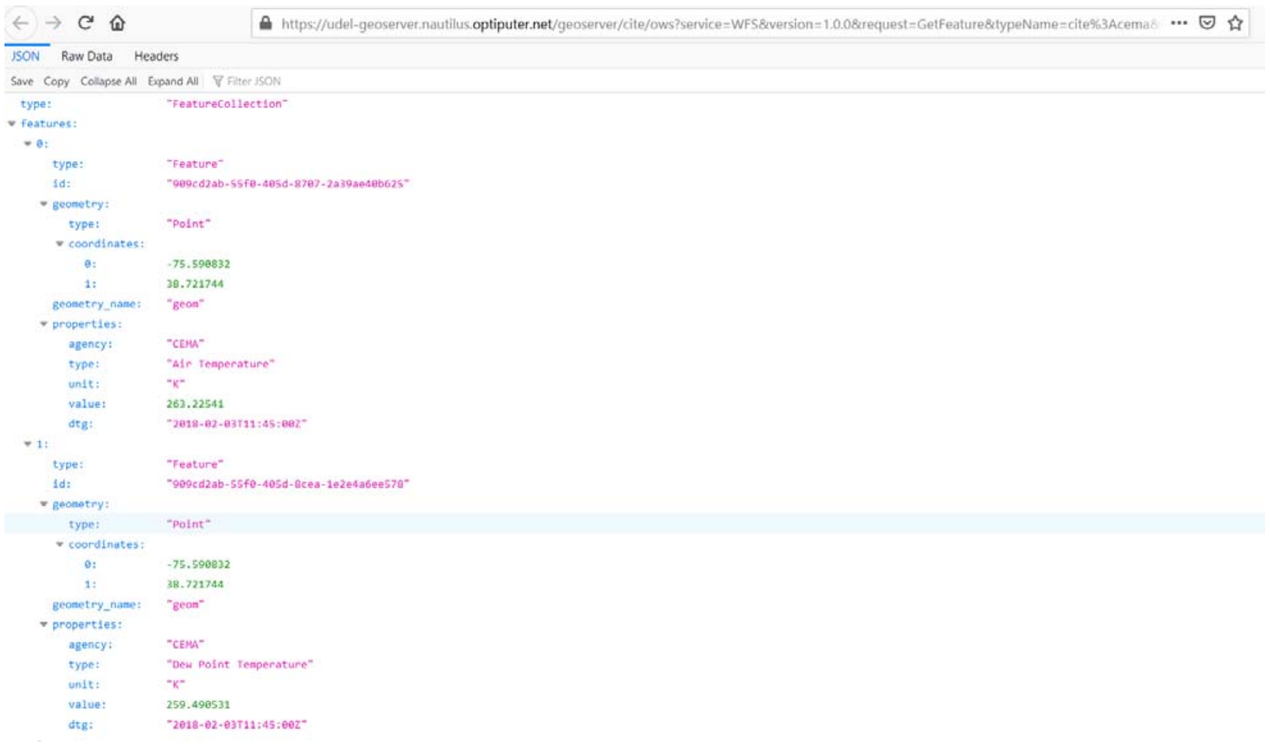
Layer Preview

List of all layers configured in GeoServer and provides previews in various formats for each.

<< < 1 > >> Results 1 to 25 (out of 25 items) Search

Type	Title	Name	Common Formats	All Formats
●	cema	cite:cema	OpenLayers GML KML	Select one
●	dgs_wel	cite:dgs_wel	OpenLayers GML KML	Select one
●	usgs	cite:usgs	OpenLayers GML KML	Select one
▨	World rectangle	tiger:giant_polygon	OpenLayers GML KML	Select one
●	Manhattan (NY) points of interest	tiger:poi	OpenLayers GML KML	Select one
▨	Manhattan (NY) landmarks	tiger:poly_landmarks	OpenLayers GML KML	Select one

3. A new window will open and display a default data result. You can edit the web address manually(recommended only for advanced users) or utilize the R or Python library for crafting the URL for you. (See section on R/Python library usage)


https://udel-geoserver.nautilus.optiputer.net/geoserver/cite/ows?service=WFS&version=1.0.0&request=GetFeature&typeName=cite%3Acema&
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON

```
type:                   "FeatureCollection"
▼ features:
  ▼ 0:
      type:             "Feature"
      id:               "909cd2ab-55f0-405d-8707-2a39ae40b625"
    ▼ geometry:
        type:           "Point"
      ▼ coordinates:
          0:            -75.590832
          1:            38.721744
      geometry_name:    "geom"
    ▼ properties:
        agency:         "CEMA"
        type:           "Air Temperature"
        unit:           "K"
        value:          263.22541
        dtg:            "2018-02-03T11:45:00Z"
  ▼ 1:
      type:             "Feature"
      id:               "909cd2ab-55f0-405d-8cea-1e2e4a6ee570"
    ▼ geometry:
        type:           "Point"
      ▼ coordinates:
          0:            -75.590832
          1:            38.721744
      geometry_name:    "geom"
    ▼ properties:
        agency:         "CEMA"
        type:           "Dew Point Temperature"
        unit:           "K"
        value:          259.490531
        dtg:            "2018-02-03T11:45:00Z"
```

4.

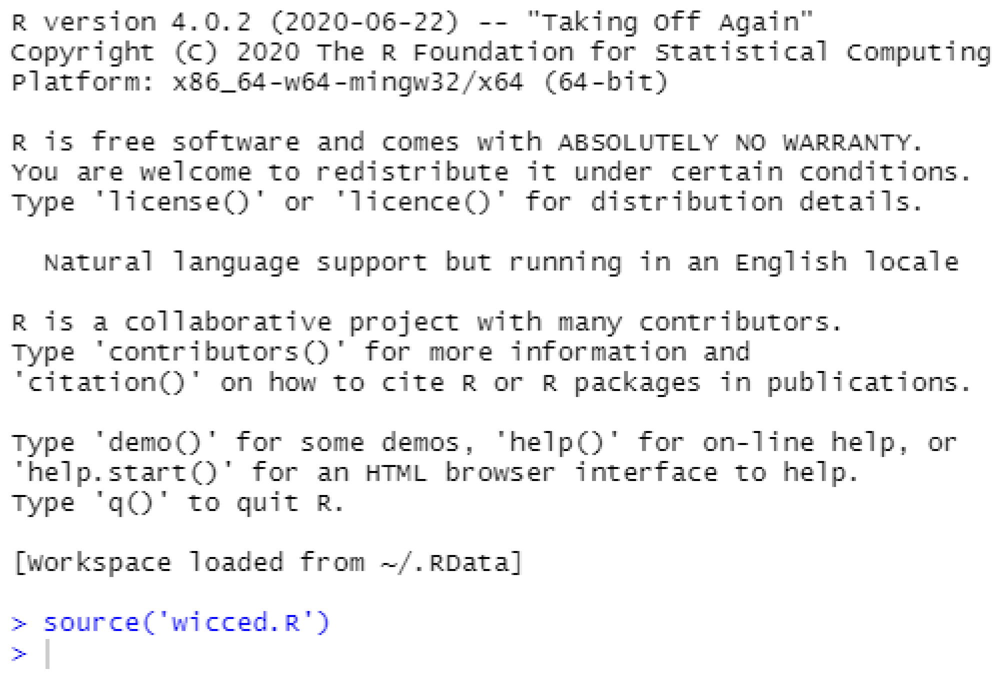
R/Python library usage

A python and R library are provided for aiding users in crafting urls for data retrieval. These libraries have been documented to make the code understandable so users can make modifications as they gain confidence working with the system.

R

1. Open R on your system.

2. Load the wicced.R library file into your R session
   source('/path/to/wicced.R')



```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> source('wicced.R')
> 
```

3. Start adding parameters to restrict data access(see library help at the end of this section)
4. After you have set your parameters, call someVariable = wicced.buildUrl() to retrieve the complete data url. You can copy this url into your web browser to view the data on a web page or download a data file.

R library Functions

Function wicced.addDate(date_string)
This function is used to add a date to the url. This will cause the server to return data points for this exact date time only. You should only use addDate or addDateRange, not both. If you do use both, preference will be given to addDate.
The date_string parameter must a character string of the format:
YYYY-MM-DD HH:mm
Where
YYYY – 4 digit year
MM – 2 digit month
DD – 2 digit day of month
HH – 2 digit hour on a 24-hour clock
mm – 2 digit minute after the hour

Examples:
To add January 1, 2010 at midnight:
wicced.addDate('2010-01-01 00:00')
To add July 1, 2015 at 11pm:
wicced.addDate('2015-07-01 23:00')

Function wicced.addDateRange(start_date, end_date)

This function is used to add a date range to the url. This will restrict returned data to be BETWEEN the start and end time given. You should only use addDate or addDateRange, not both. If you do use both, preference will be given to addDate.
The start_date and end_date must be a character string of the format:
YYYY-MM-DD HH:mm
Where
YYYY – 4 digit year
MM – 2 digit month
DD – 2 digit day of month
HH – 2 digit hour on a 24-hour clock
mm – 2 digit minute after the hour
NOTE: Please be considerate of other users when selecting date ranges. Using large ranges will negatively impact system performance.

Examples:
To select data between Jan. 1st and Jan 10th:
wicced.addDateRange ('2010-01-01 00:00', '2010-01-10 23:59')
To select data between July 1, 2015 at 11pm and July 2, 2015 at 4am:
wicced. addDateRange ('2015-07-01 23:00', '2015-07-02 04:00')

Function wicced.addDataType(data_type_name)
Select data for a specific data only. This will cause other data types to be removed from the returned data.
Note: This function is case sensitive. You can use the web server view to see data types that are loaded for a particular data source or reach out to us if you have questions.

Examples:
To select Air Temperature Data:
wicced.addDataType('Air Temperature')

Function wicced.addBox(ul_lon, ul_lat, lr_lon, lr_lat)
Restrict data to sites within the given bounding box. This will cause data outside the given lat/lon coordinates to be removed from the returned data.

Examples:
To select data at one specific station:
wicced.addBox(-75.594958, 38.542633, -75.594958, 38.542633)

Function wicked.changeFormat(format)
Changes the output format from the default(geojson) to something else. Valid inputs are 'application/json' or 'csv'.

Examples:
To change to CSV format:
wicced.changeFormat('csv')

Function wicced.buildUrl()
Use this function after adding the necessary dates, data type, and/or bounding box. This function will return a string of characters that can be used to retrieve data from the server. You can copy and paste the returned character string into your browser or other data retrieval function.

Examples:
someVariable <- wicced.buildUrl()

Function wicced.reset()
The reset function should be called after a call to buildUrl. It will reset the library code for a subsequent url to be generated or to clean up if you decide you made a mistake adding a parameter.

Examples:
wicced.reset()

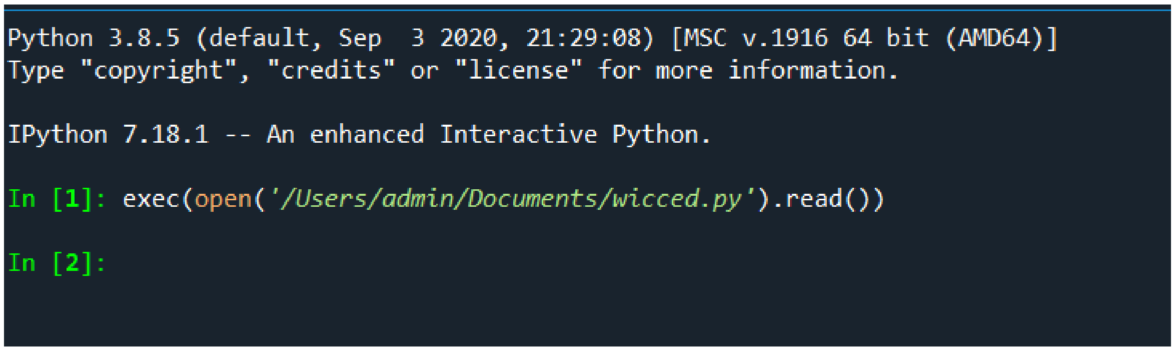
Sample R script:

```
source('wicced.R')
# add a specific data time to the url
wicced.addDate('2010-01-01 00:00')
# add a specific data type
wicced.addDataType('Air Temperature')
# subset our data returns to this point
wicced.addBox(-75.594958, 38.542633, -75.594958, 38.542633)
# generate the url string
url = wicced.buildUrl()
# load geojson library and retrieve data
library(geojsonR)
your_data = FROM_GeoJson(url)
```

Python

1. Open the python environment of your choosing.

2. Load the project wicced.py library. This will give you access to the wicced class for building a data access URL.



```
Python 3.8.5 (default, Sep  3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.18.1 -- An enhanced Interactive Python.

In [1]: exec(open('/Users/admin/Documents/wicced.py').read())

In [2]:
```

3. Start adding parameters to restrict data access(see library help at the end of this section)
4. After you have set your parameters, call the buildUrl() method to retrieve the complete data url. You can copy this url into your web browser to view the data on a web page or download a data file.

Python Library Functions

Method addDate(date_string)
This method is used to add a date to the url. This will cause the server to return data points for this exact date time only. You should only use addDate or addDateRange, not both. If you do use both, preference will be given to addDate.
The date_string parameter must a character string of the format:
YYYY-MM-DD HH:mm
Where
YYYY – 4 digit year
MM – 2 digit month
DD – 2 digit day of month
HH – 2 digit hour on a 24-hour clock
mm – 2 digit minute after the hour

<u>Examples</u>:
To add January 1, 2010 at midnight:
temp = wicced()
temp.addDate('2010-01-01 00:00')
To add July 1, 2015 at 11pm:
temp = wicced()
temp.addDate('2015-07-01 23:00')

Method addDateRange(start_date, end_date)

This method is used to add a date range to the url. This will restrict returned data to be BETWEEN the start and end time given. You should only use addDate or addDateRange, not both. If you do use both, preference will be given to addDate.
The start_date and end_date must be a character string of the format:
YYYY-MM-DD HH:mm

Where
YYYY – 4 digit year
MM – 2 digit month
DD – 2 digit day of month
HH – 2 digit hour on a 24-hour clock
mm – 2 digit minute after the hour
NOTE: Please be considerate of other users when selecting date ranges. Using large ranges will negatively impact system performance.

Examples:
To select data between Jan. 1st and Jan 10th:
```
temp = wicced()
temp.addDateRange (‘2010-01-01 00:00’, ‘2010-01-10 23:59’)
```
To select data between July 1, 2015 at 11pm and July 2, 2015 at 4am:
```
temp = wicced()
temp. addDateRange (‘2015-07-01 23:00’, ‘2015-07-02 04:00’)
```

Method addDataType(data_type_name)
Select data for a specific data only. This will cause other data types to be removed from the returned data.
Note: This method is case sensitive. You can use the web server view to see data types that are loaded for a particular data source or reach out to us if you have questions.

Examples:
To select Air Temperature Data:
```
temp = wicced()
temp.addDataType(‘Air Temperature’)
```

Method addBox(ul_lon, ul_lat, lr_lon, lr_lat)
Restrict data to sites within the given bounding box. This will cause data outside the given lat/lon coordinates to be removed from the returned data.

Examples:
To select data at one specific station:
```
temp = wicced()
temp.addBox(-75.594958, 38.542633, -75.594958, 38.542633)
```

Method changeFormat(format)
Changes the output format from the default(geojson) to something else. Valid inputs are ‘application/json’ or ‘csv’.

Examples**:**
To change to CSV format:

```
temp = wicced()
temp.changeFormat(‘csv’)
```

Method wicced.buildUrl()
Use this method after adding the necessary dates, data type, and/or bounding box. This method will return a string of characters that can be used to retrieve data from the server. You can copy and paste the returned character string into your browser or other data retrieval function.

<u>Examples</u>:

```
temp = wicced()
someVariable <- temp.buildUrl()
```

Sample Python script:

```
#load the library code
exec(open('/Users/admin/Documents/wicced.py').read())
#create the class
temp = wicced()
#add the desired date
temp.addDate('2010-01-01 00:00')
#restrict to Air Temperature Data only
temp.addDataType("Air Temperature")
#select a specific station only
temp.addBox(-75.594958, 38.542633, -75.594958, 38.542633)
#get the completed url
url = temp.buildUrl()
# load geojson library and retrieve data
import geopandas as gpd
gpd.read_file(url)
```