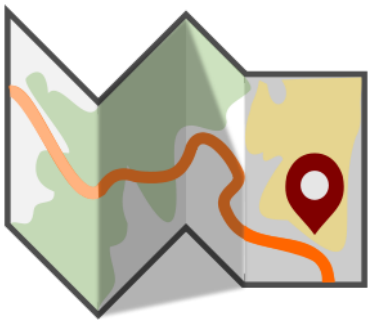
Getting Started with Python Programming

QGIS Tutorials and Tips



Author

Ujaval Gandhi

http://www.spatialthoughts.com

Getting Started With Python Programming

QGIS has a powerful programming interface that allows you to extend the core functionality of the software as well as write scripts to automate your tasks. QGIS supports the popular Python scripting language. Even if you are a beginner, learning a little bit of Python and QGIS programming interface will allow you to be much more productive in your work. This tutorial assumes no prior programming knowledge and is intended to give an introduction to python scripting in QGIS (PyQGIS).

Overview of the task

We will load a vector point layer representing all major airports and use python scripting to create a text file with the airport name, airport code, latitude and longitude for each of the airport in the layer.

Get the data

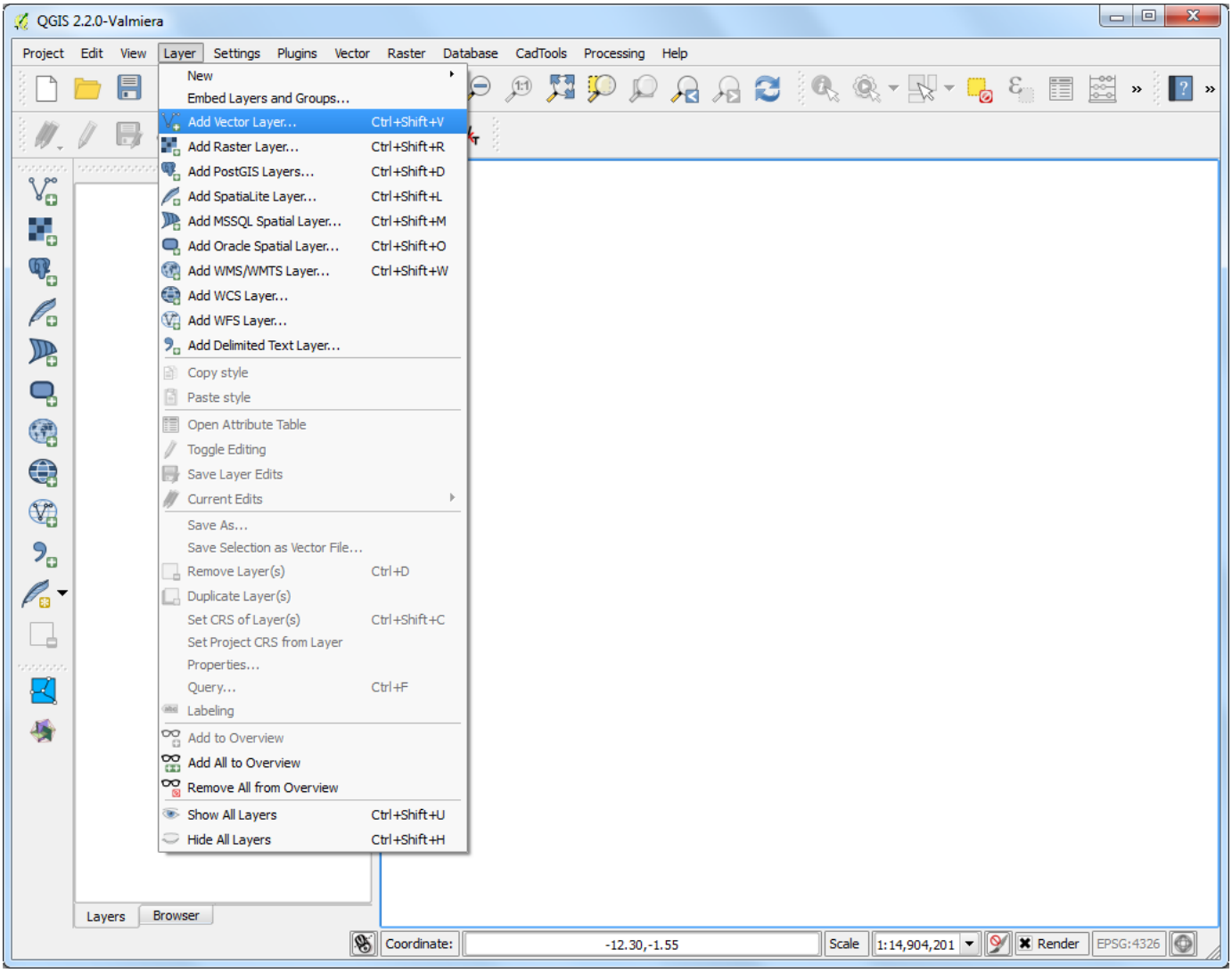
We will use the Airports dataset from Natural Earth.

Download the Airports shapefile.

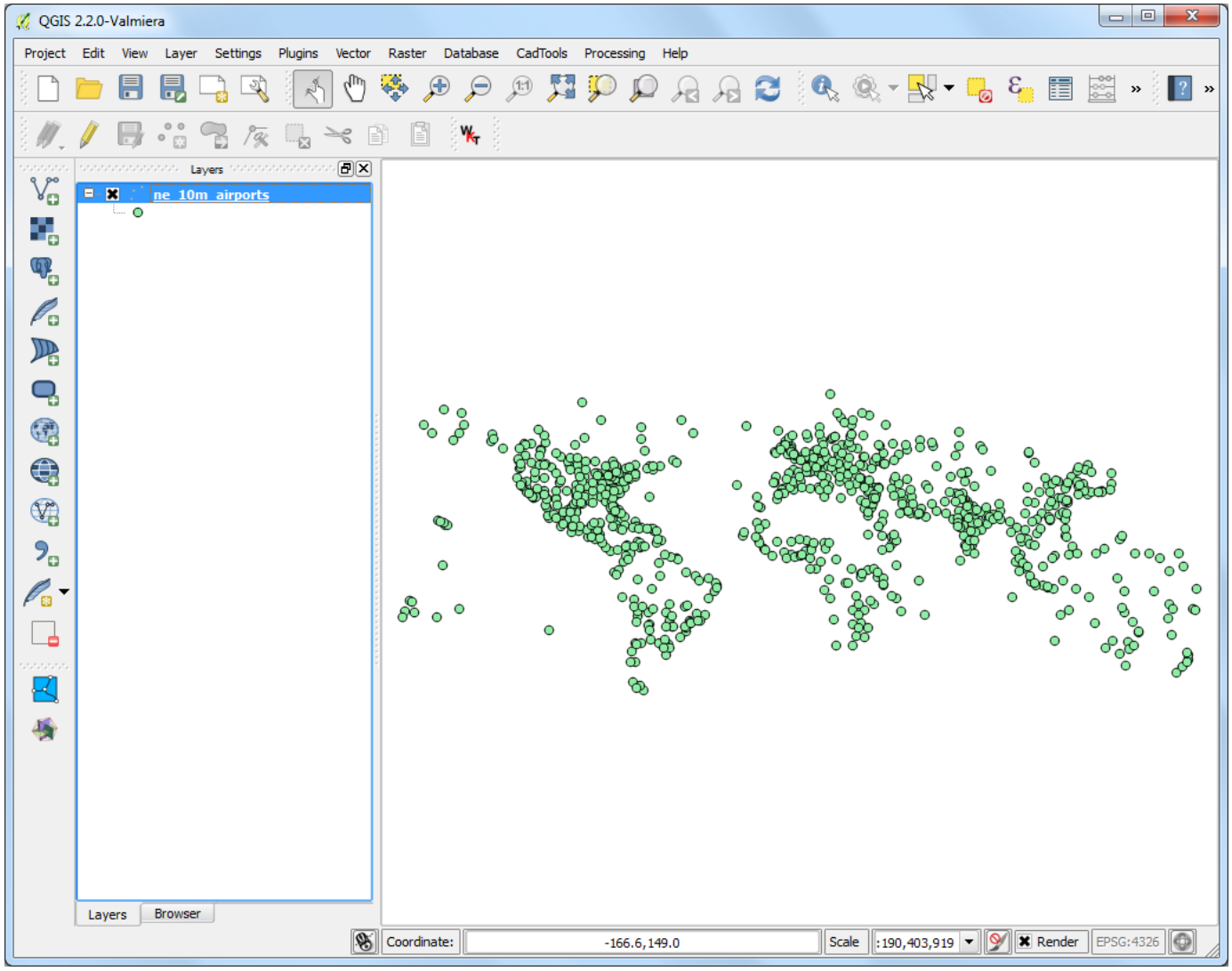
Data Source [NATURALEARTH]

Procedure

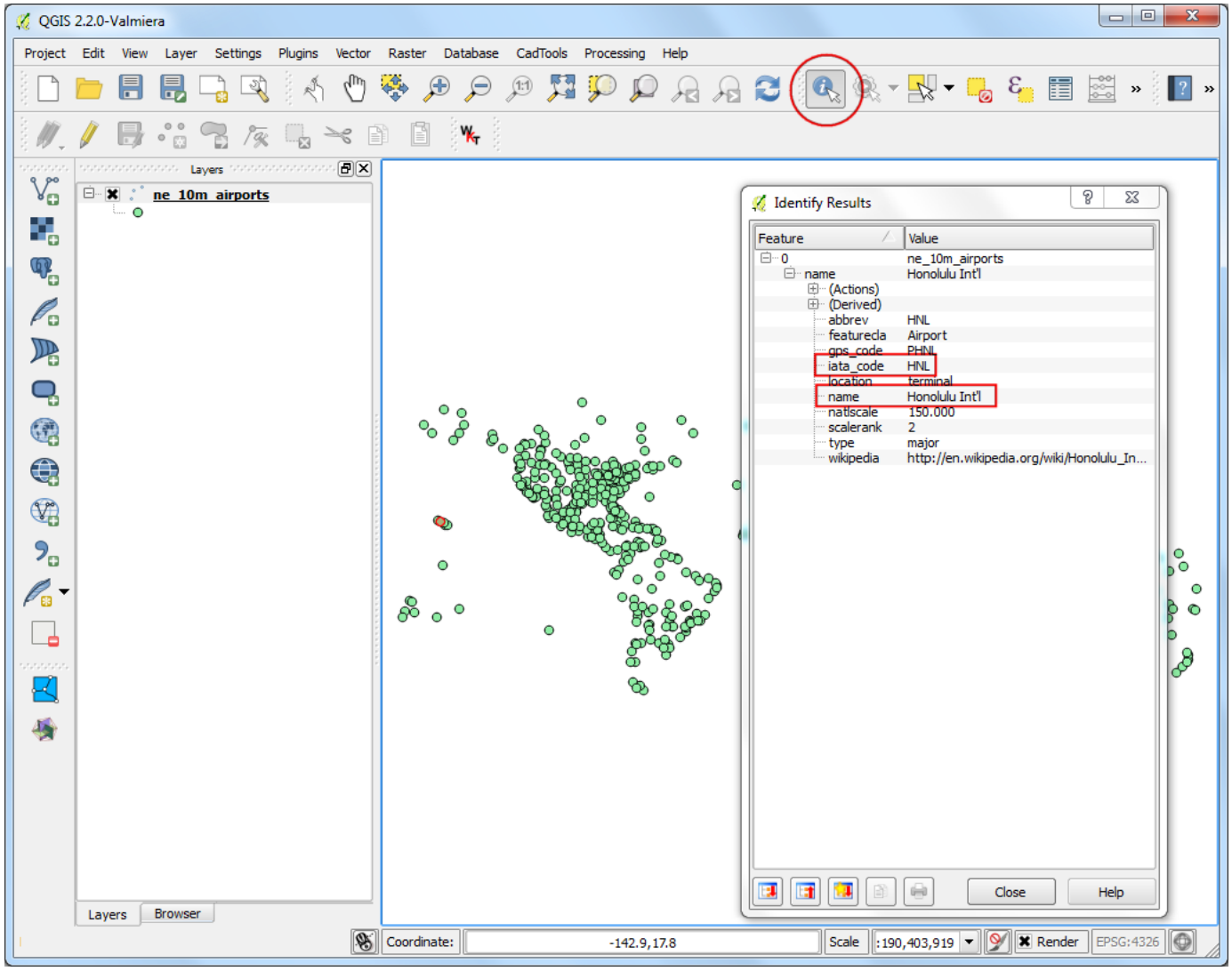
1. In QGIS, go to Layers ▸ Add Vector Layer. Browse to the downloaded `ne_10m_airports.zip` file and click Open. Select the `ne_10m_airports.shp` layer and click OK.



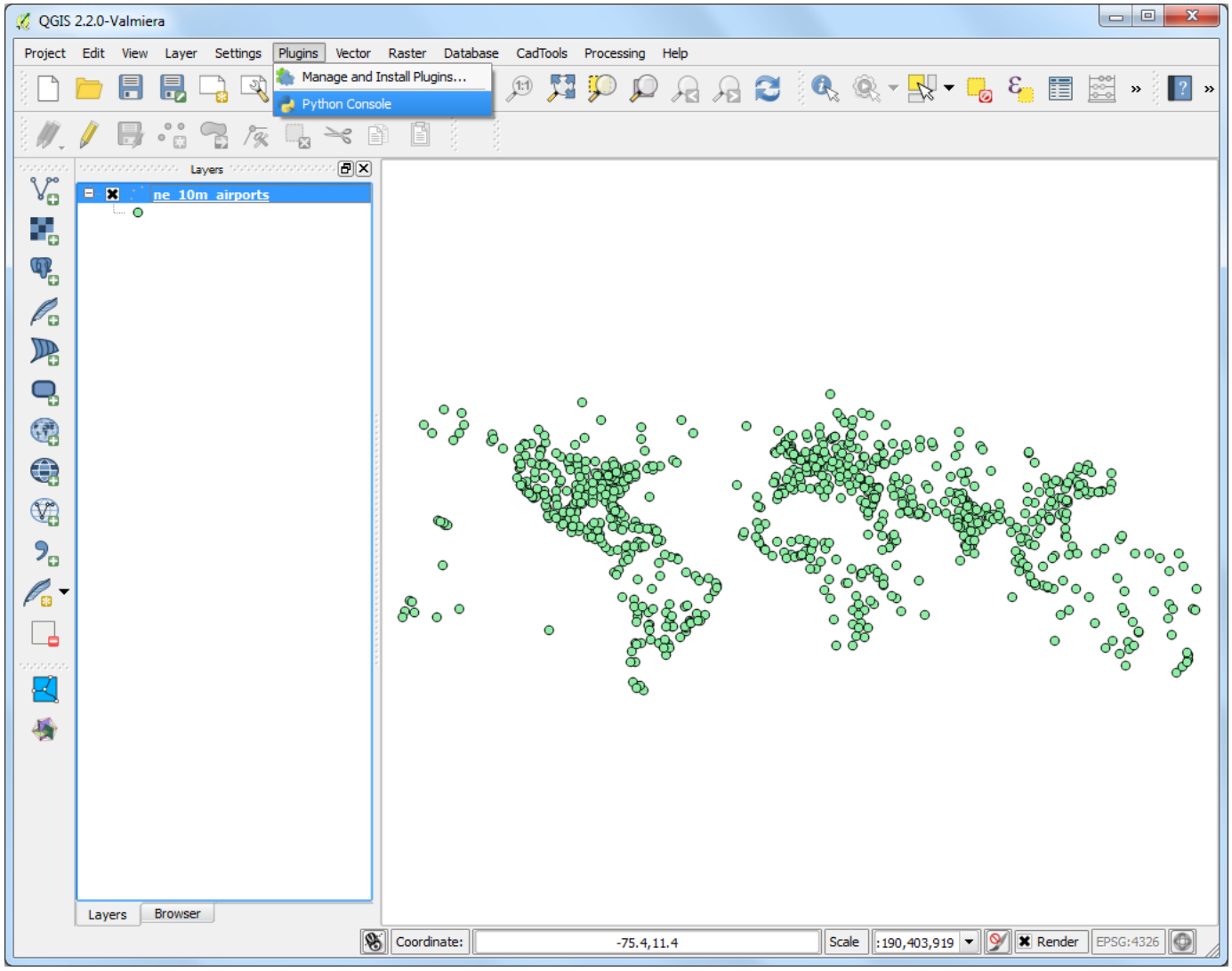
2. You will see the `ne_10m_airports` layer loaded in QGIS.



3. Select the Identify tool and click on any of the points to examine the available attributes. You will see that the name of the airport and it's 3 digit code are contained in the attributes `name` and `iata_code` respectively.

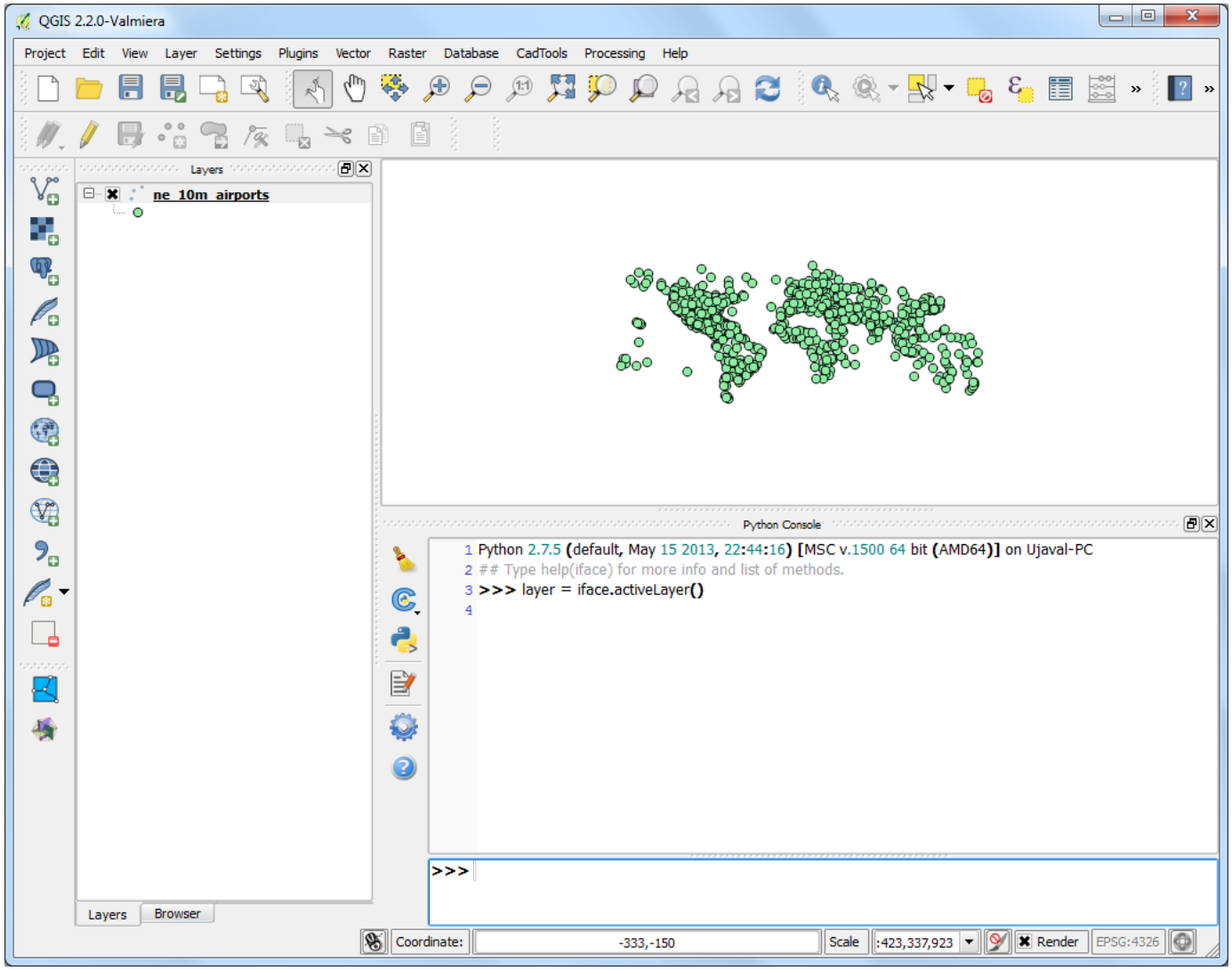


4. QGIS provides a built-in console where you can type python commands and get the result. This console is a great way to learn scripting and also to do quick data processing. Open the Python Console by going to Plugins ▸ Python Console.



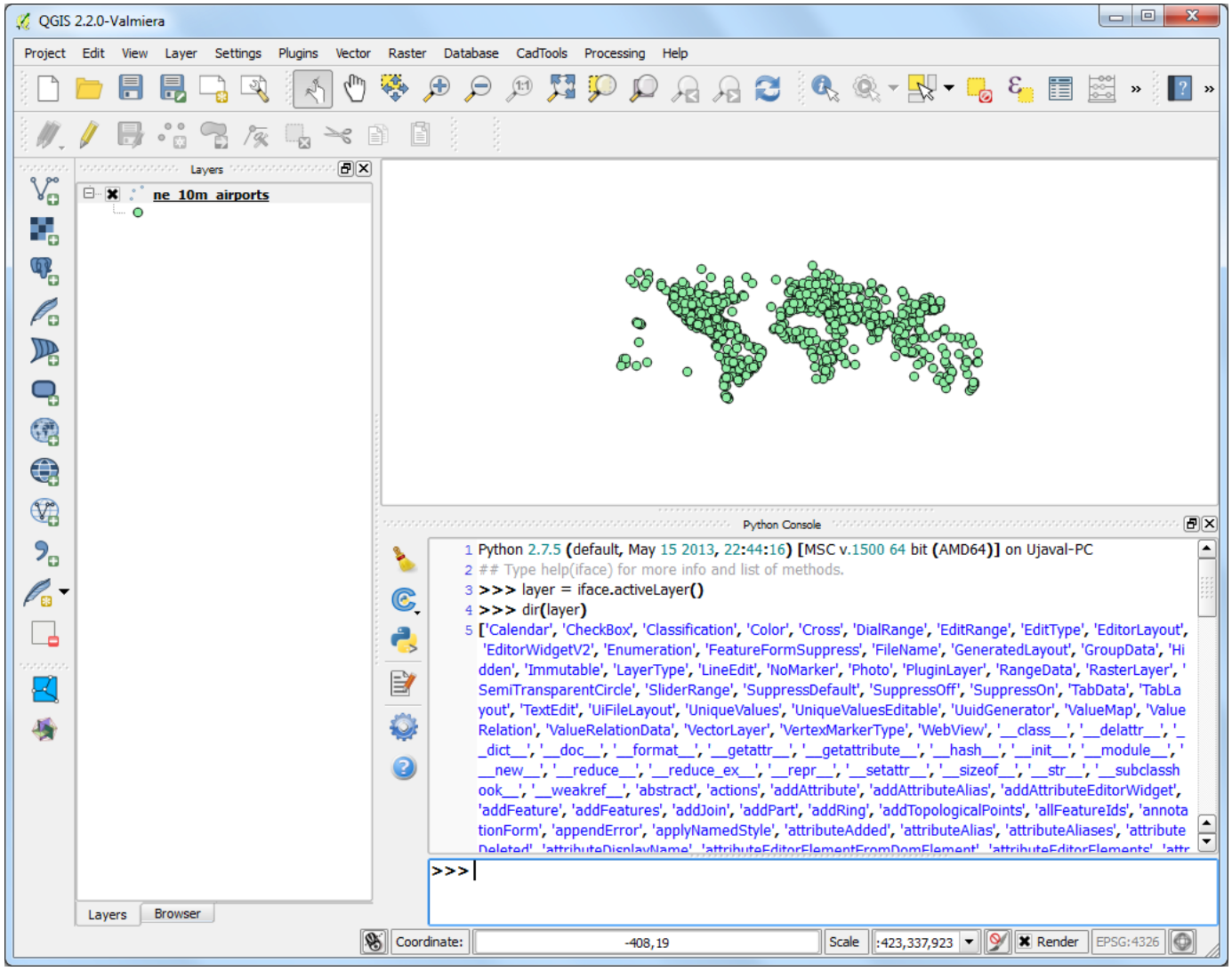
5. You will see a new panel open at the bottom of QGIS canvas. You will see a prompt like >>> at the bottom where you can type commands. For interacting with the QGIS environment, we must use the `iface` variable. To access the currently active layer in QGIS, you can type the following and press Enter. This command fetches the reference to the currently loaded layer and stores it in the `layer` variable.

```
layer = iface.activeLayer()
```



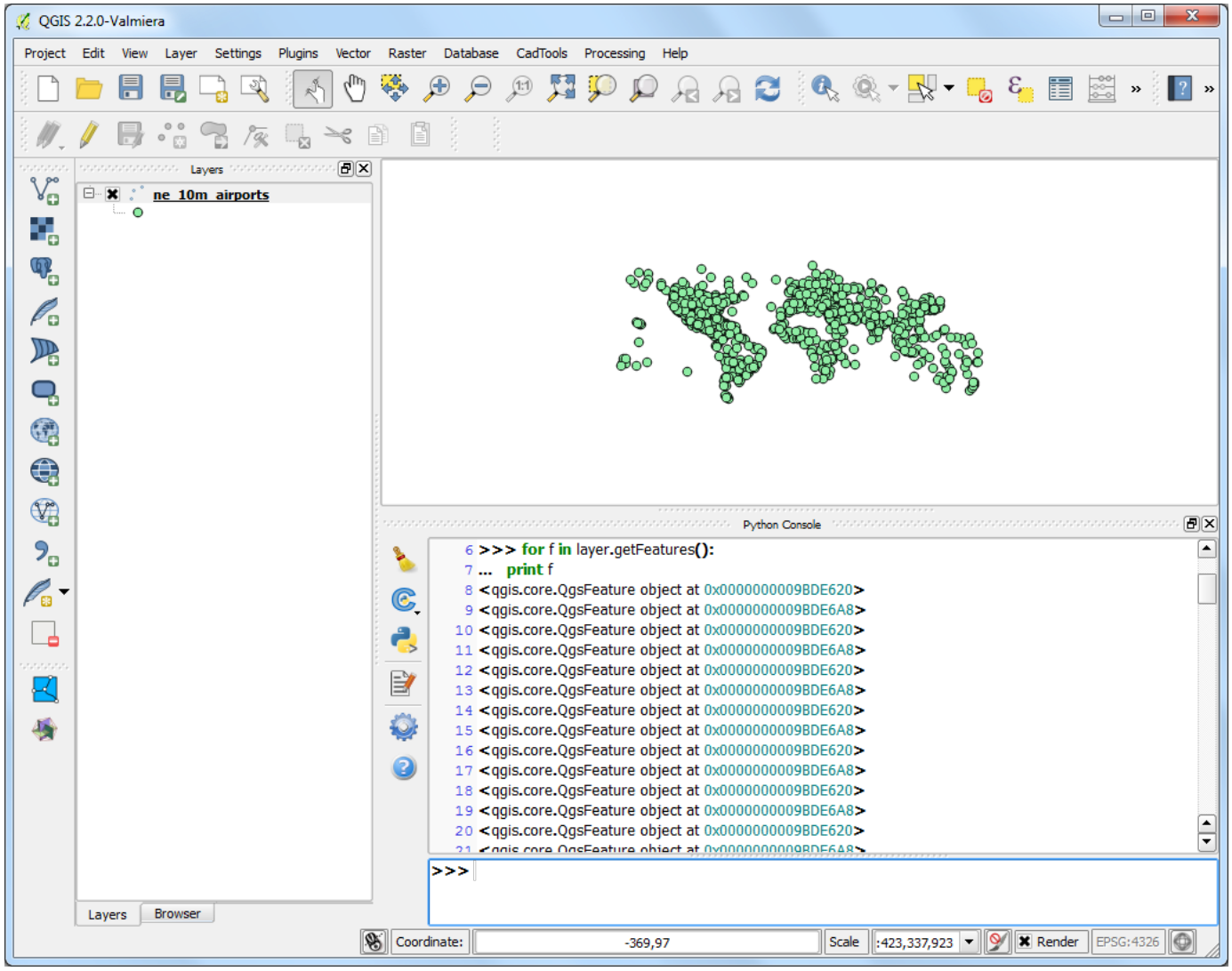
6. There is a handy function called `dir()` in python that shows you all available methods for any object. This is useful when you are not sure what functions are available for the object. Run the following command to see what operations we can do on the `layer` variable.

```
dir(layer)
```



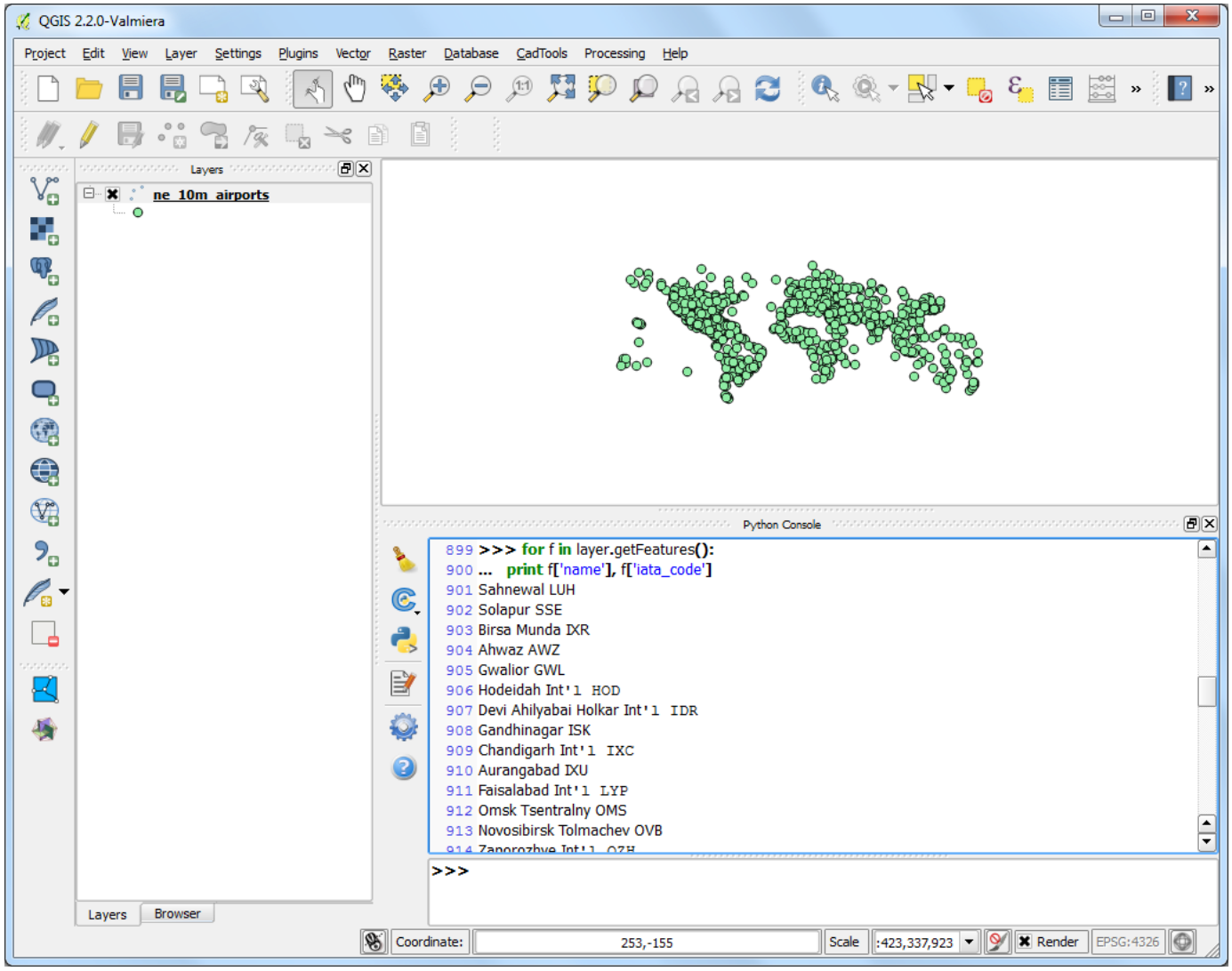
7. You will see a long list of available functions. For now, we will use a function called `getFeatures()` which will gets you the reference to all features of a layer. In our case, each feature will be a point representing an airport. You can type the following command to iterate through each of the features in the current layer. Make sure to add 2 spaces before typing the second line.

```
for f in layer.getFeatures():
  print f
```



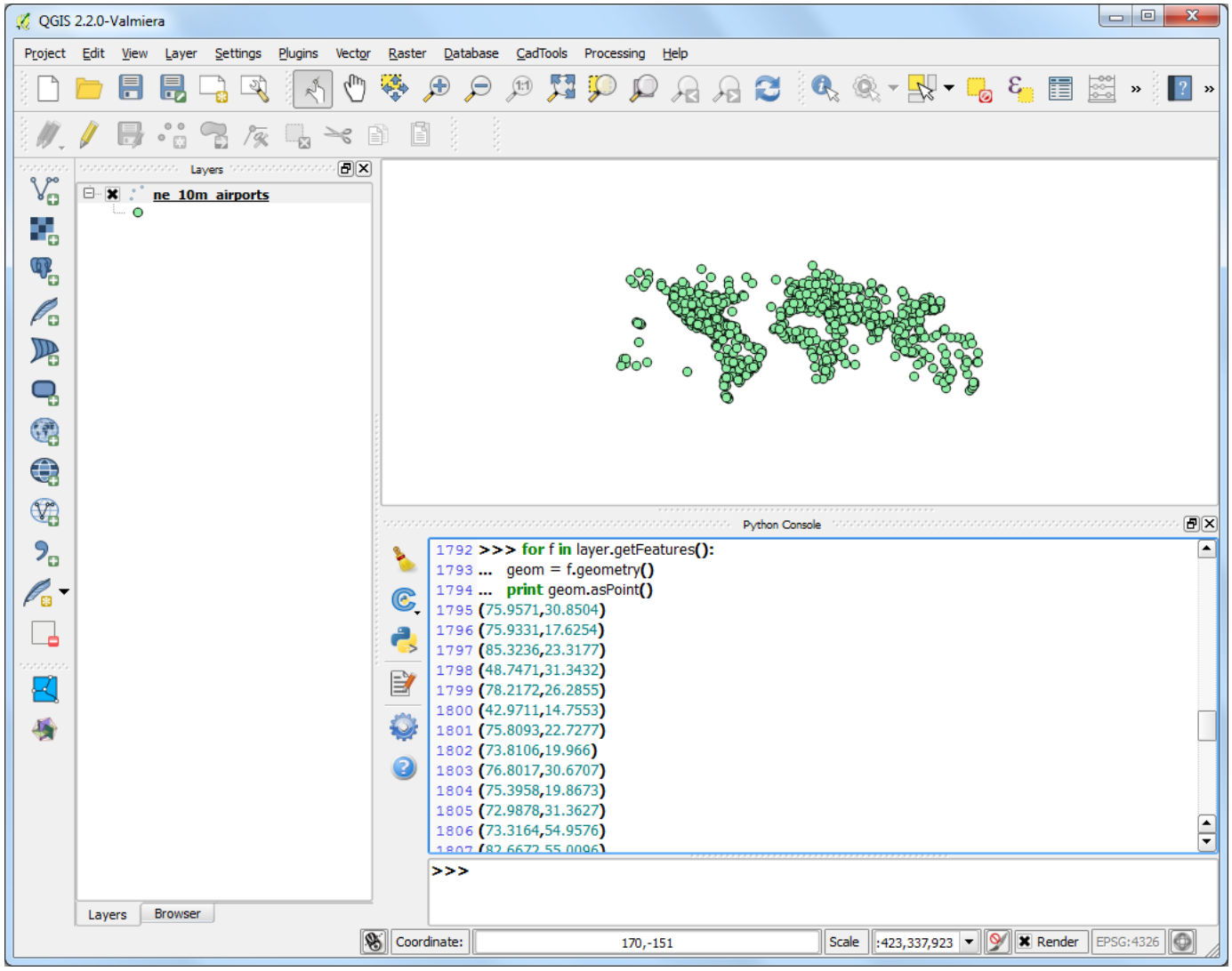
8. As you will see in the output, each line contains a reference to a feature within the layer. The reference to the feature is stored in the `f` variable. We can use the `f` variable to access the attributes of each feature. Type the following to print the `name` and `iata_code` for each airport feature.

```
for f in layer.getFeatures():
  print f['name'], f['iata_code']
```



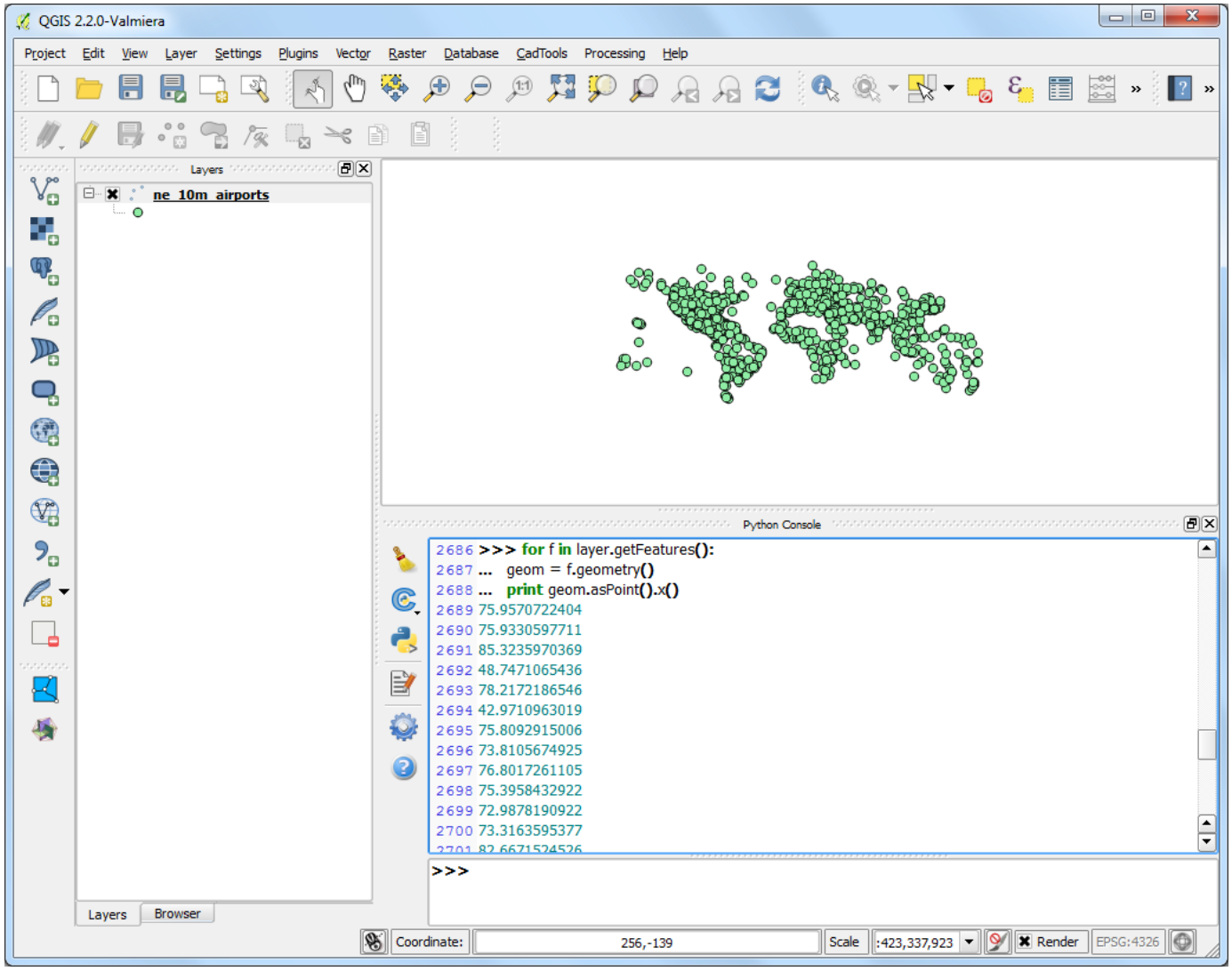
9. So now you know how to programatically access the attribute of each feature in a layer. Now, let's see how we can access the coordinates of the feature. The coordinates of a vector feature can be accessed by calling the `geometry()` function. This function returns a geometry object that we can store in the variable `geom`. You can run `asPoint()` function on the geometry object to get the x and y coordinates of the point. If your feature is a line or a polygon, you can use `asPolyline()` or `asPolygon()` functions. Type the following code at the prompt and press Enter to see the x and y coordinates of each feature.

```
for f in layer.getFeatures():
  geom = f.geometry()
  print geom.asPoint()
```



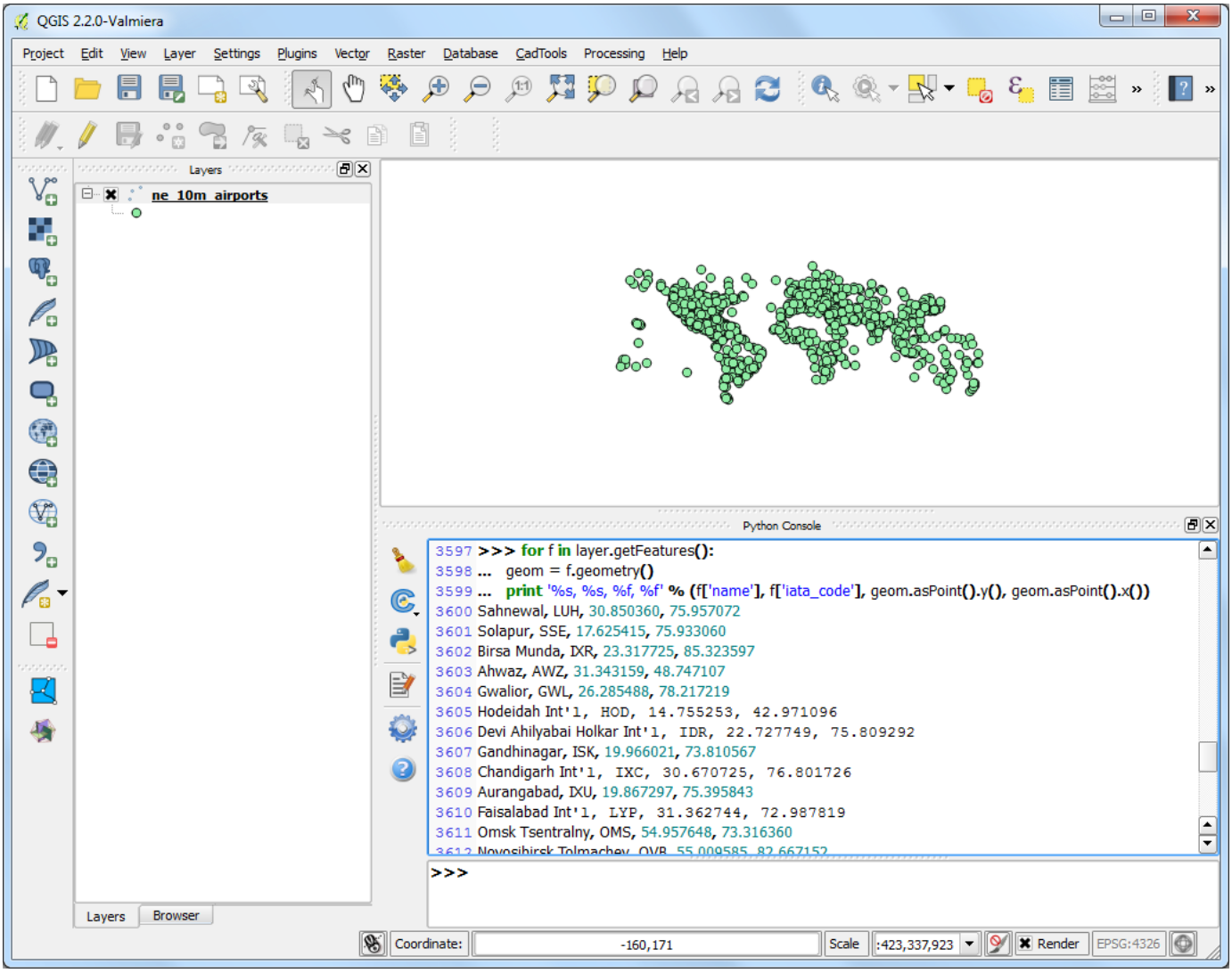
10. What if we wanted to get only the `x` cordinate of the feature? You can call the `x()` function on the point object and get its x coordinate.

```
for f in layer.getFeatures():
  geom = f.geometry()
  print geom.asPoint().x()
```



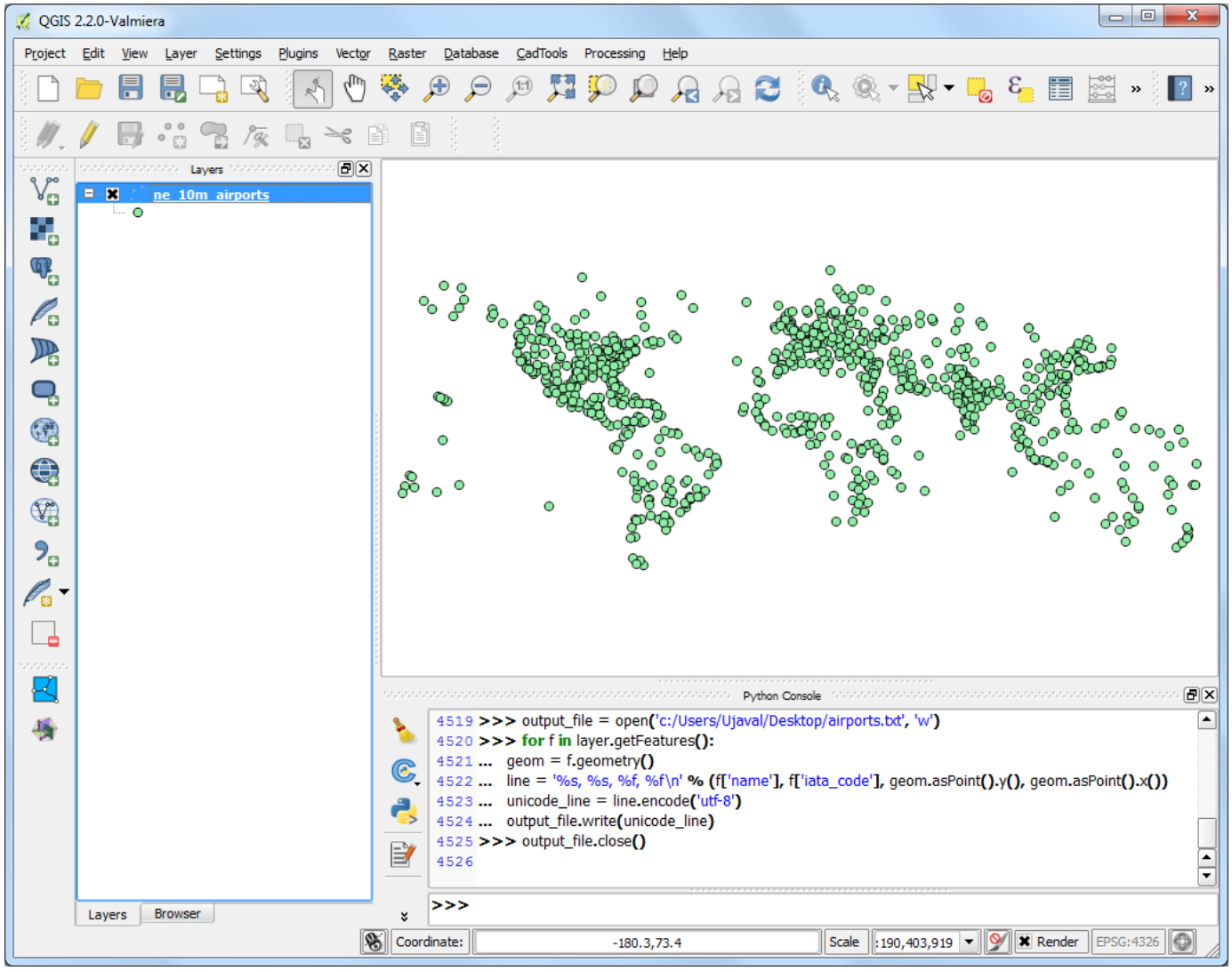
11. Now we have all the pieces that we can stitch together to generate our desired output. Type the following code to print the name, iata_code, latitude and longitude of each of the airport features. The `%s` and `%f` notations are ways to format a string and number variables.

```
for f in layer.getFeatures():
  geom = f.geometry()
  print '%s, %s, %f, %f' % (f['name'], f['iata_code'],
         geom.asPoint().y(), geom.asPoint().x())
```

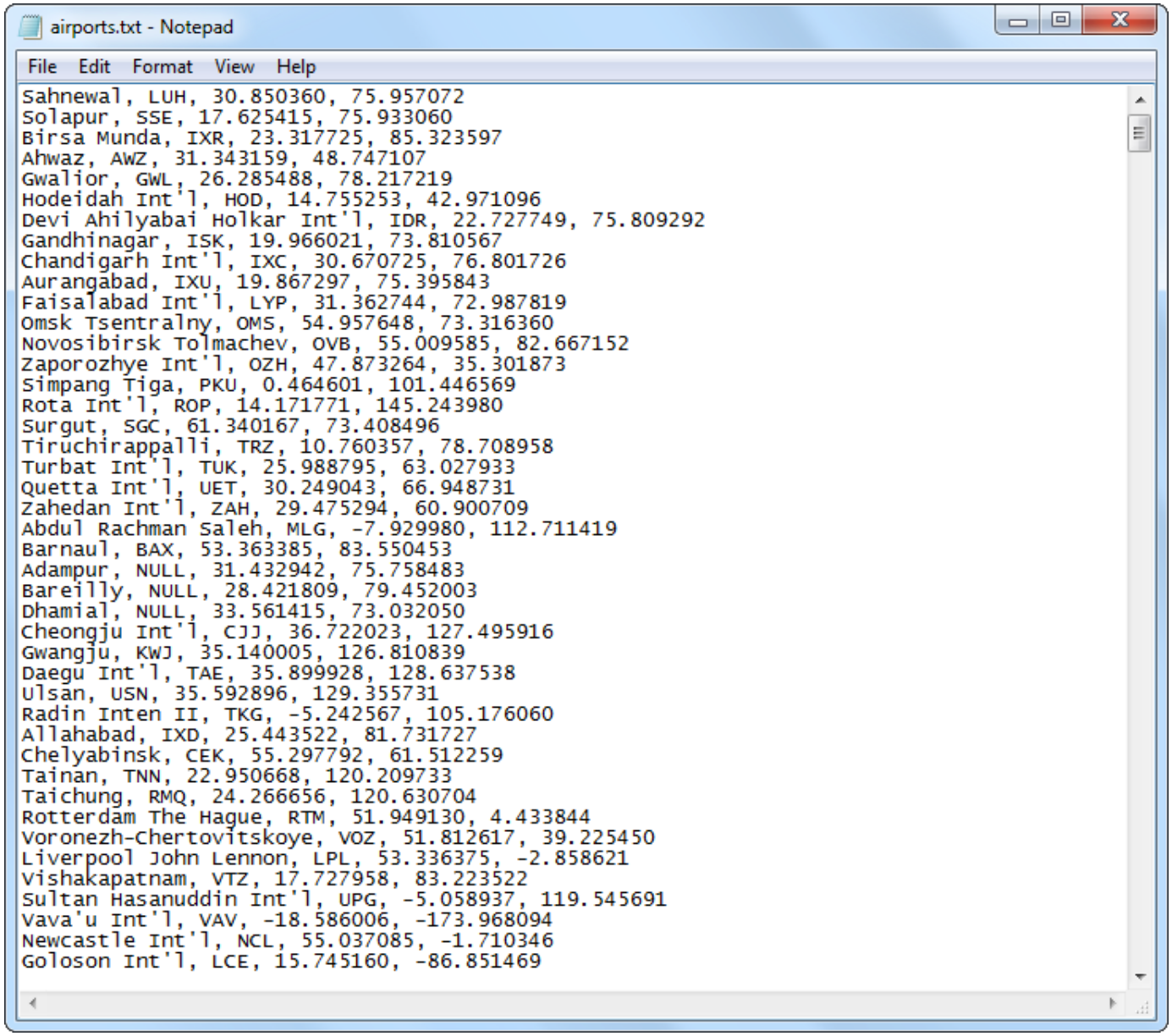


12. You can see the output printed on the console. A more useful way to store the output would be in a file. You can type the following code to create a file and write the output there. Replace the file path with a path on your own system. Note that we add `\n` at the end of our line formatting. This is to add a newline after we add the data for each feature. You should also note the `unicode_line = line.encode('utf-8')` line. Since our layer contains some features with unicode characters, we can't simply write it to a text file. We encode the text using the UTF-8 encoding and then write to the text file.

```
output_file = open('c:/Users/Ujaval/Desktop/airports.txt', 'w')
for f in layer.getFeatures():
  geom = f.geometry()
  line = '%s, %s, %f, %f\n' % (f['name'], f['iata_code'],
          geom.asPoint().y(), geom.asPoint().x())
  unicode_line = line.encode('utf-8')
  output_file.write(unicode_line)
output_file.close()
```



13. You can go to the output file location you specified and open the text file. You will see the data from the airports shapefile that we extracted using python scripting.


airports.txt - Notepad
File Edit Format View Help
Sahnewal, LUH, 30.850360, 75.957072
Solapur, SSE, 17.625415, 75.933060
Birsa Munda, IXR, 23.317725, 85.323597
Ahwaz, AWZ, 31.343159, 48.747107
Gwalior, GWL, 26.285488, 78.217219
Hodeidah Int'l, HOD, 14.755253, 42.971096
Devi Ahilyabai Holkar Int'l, IDR, 22.727749, 75.809292
Gandhinagar, ISK, 19.966021, 73.810567
Chandigarh Int'l, IXC, 30.670725, 76.801726
Aurangabad, IXU, 19.867297, 75.395843
Faisalabad Int'l, LYP, 31.362744, 72.987819
Omsk Tsentralny, OMS, 54.957648, 73.316360
Novosibirsk Tolmachev, OVB, 55.009585, 82.667152
Zaporozhye Int'l, OZH, 47.873264, 35.301873
Simpang Tiga, PKU, 0.464601, 101.446569
Rota Int'l, ROP, 14.171771, 145.243980
Surgut, SGC, 61.340167, 73.408496
Tiruchirappalli, TRZ, 10.760357, 78.708958
Turbat Int'l, TUK, 25.988795, 63.027933
Quetta Int'l, UET, 30.249043, 66.948731
Zahedan Int'l, ZAH, 29.475294, 60.900709
Abdul Rachman Saleh, MLG, -7.929980, 112.711419
Barnaul, BAX, 53.363385, 83.550453
Adampur, NULL, 31.432942, 75.758483
Bareilly, NULL, 28.421809, 79.452003
Dhamial, NULL, 33.561415, 73.032050
Cheongju Int'l, CJJ, 36.722023, 127.495916
Gwangju, KWJ, 35.140005, 126.810839
Daegu Int'l, TAE, 35.899928, 128.637538
Ulsan, USN, 35.592896, 129.355731
Radin Inten II, TKG, -5.242567, 105.176060
Allahabad, IXD, 25.443522, 81.731727
Chelyabinsk, CEK, 55.297792, 61.512259
Tainan, TNN, 22.950668, 120.209733
Taichung, RMQ, 24.266656, 120.630704
Rotterdam The Hague, RTM, 51.949130, 4.433844
Voronezh-Chertovitskoye, VOZ, 51.812617, 39.225450
Liverpool John Lennon, LPL, 53.336375, -2.858621
Vishakapatnam, VTZ, 17.727958, 83.223522
Sultan Hasanuddin Int'l, UPG, -5.058937, 119.545691
Vava'u Int'l, VAV, -18.586006, -173.968094
Newcastle Int'l, NCL, 55.037085, -1.710346
Goloson Int'l, LCE, 15.745160, -86.851469