Whitebox Workflows for Python (WbW) - Next-level geoprocessing

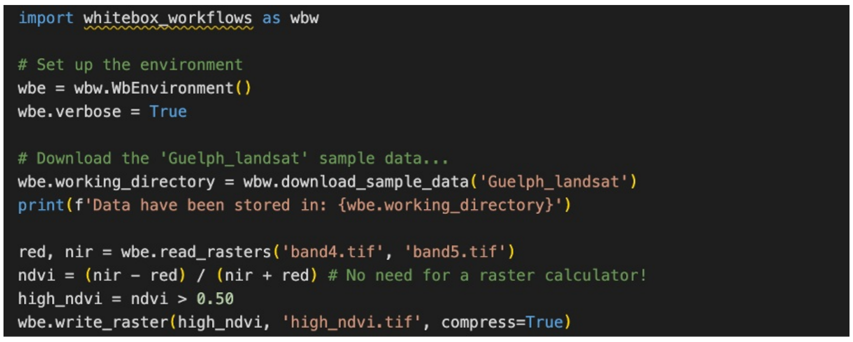
Hello {{ contact.FIRSTNAME }},

Over the past couple months, we've been working hard to bring you performant and accessible geospatial tools. Today, we're thrilled to announce the official release of [Whitebox Workflows for Python (WbW) v1.0](). Whitebox Workflows (WbW) is a Python library for advanced geoprocessing, including more than 400 functions for GIS and remote sensing analysis operations and for for manipulating common types of raster, vector and LiDAR geospatial data. WbW is an advanced geoprocessing library for manipulating and analyzing raster, vector and LiDAR geospatial data using Python scripting environments. WBW provides more powerful and natural Python-Whitebox interaction.

Whitebox Worfklows for Python v1.0

How WbW compares

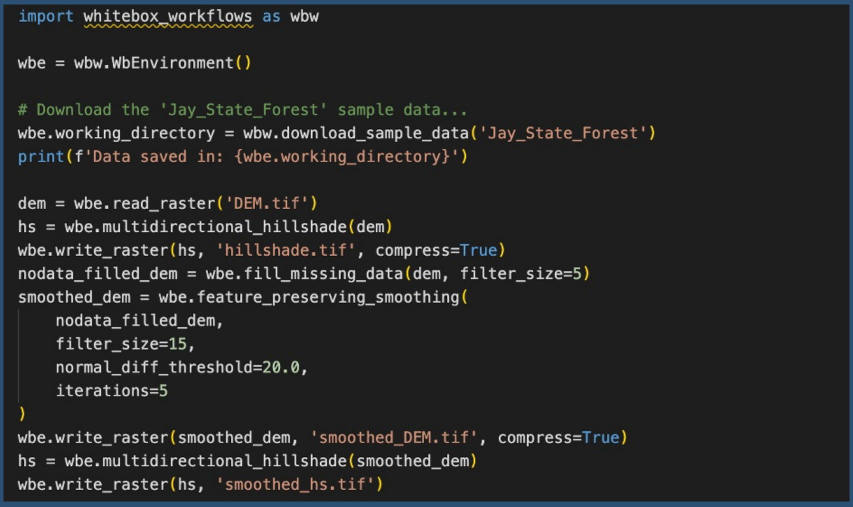
Based on the [WhiteboxTools Open Core (WbOC)](), WbW contains the 400+ geoprocessing tools that you have come to know and love about Whitebox. However, WbW has many advantages in comparison to the WbOC. The new WbW is specifically designed to **work with Python** to provide a natural geoprocessing environment. **WbW is a Python native extension module**, much like NumPy, SciPy and other scientific libraries, compared with WbOC, which is a command line application at its base. WbW is also developed in Rust, the **fast** and modern systems programming language.



```python
import whitebox_workflows as wbw

# Set up the environment
wbe = wbw.WbEnvironment()
wbe.verbose = True

# Download the 'Guelph_landsat' sample data...
wbe.working_directory = wbw.download_sample_data('Guelph_landsat')
print(f'Data have been stored in: {wbe.working_directory}')

red, nir = wbe.read_rasters('band4.tif', 'band5.tif')
ndvi = (nir - red) / (nir + red) # No need for a raster calculator!
high_ndvi = ndvi > 0.50
wbe.write_raster(high_ndvi, 'high_ndvi.tif', compress=True)
```

Learn more about WbW

WbW allows you to **directly interact with spatial objects** (rasters, vectors, and LiDAR point clouds) from Python, enabling you to write far more powerful geoprocessing scripts than previously possible. Interacting directly with data also significantly decreases the number of read/write operations during intermediate processing, **improving performance of complex workflows and reducing wear and tear on your expensive system hardware**. Oh, and did we mention that **WbW has an improved memory model for raster data**? With WbW, rasters are stored in memory using their native data type, rather than always using large 64-bit floats.



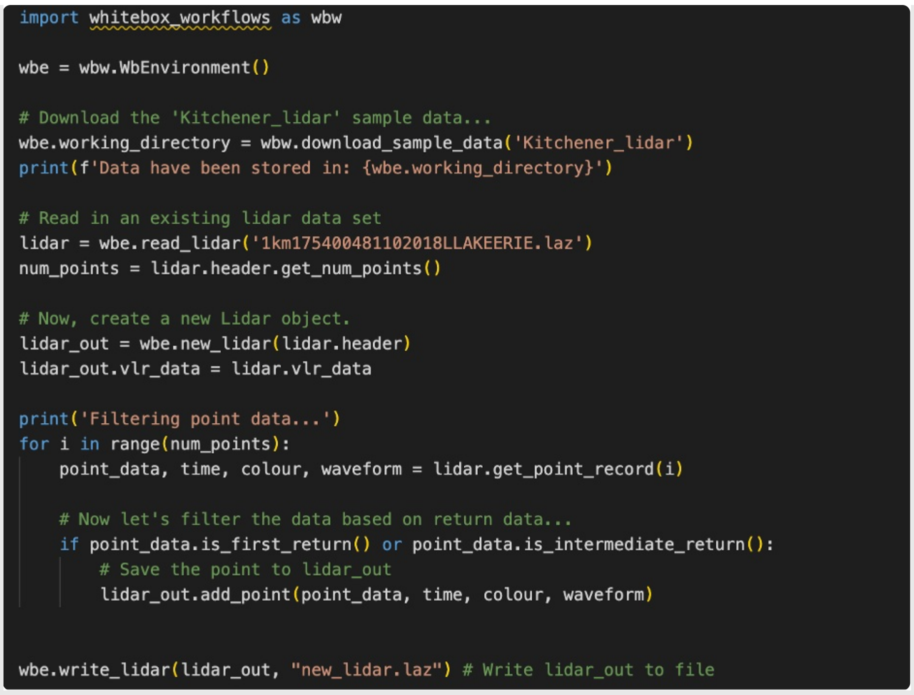
```python
import whitebox_workflows as wbw

wbe = wbw.WbEnvironment()

# Download the 'Jay_State_Forest' sample data...
wbe.working_directory = wbw.download_sample_data('Jay_State_Forest')
print(f'Data saved in: {wbe.working_directory}')

dem = wbe.read_raster('DEM.tif')
hs = wbe.multidirectional_hillshade(dem)
wbe.write_raster(hs, 'hillshade.tif', compress=True)
nodata_filled_dem = wbe.fill_missing_data(dem, filter_size=5)
smoothed_dem = wbe.feature_preserving_smoothing(
    nodata_filled_dem,
    filter_size=15,
    normal_diff_threshold=20.0,
    iterations=5
)
wbe.write_raster(smoothed_dem, 'smoothed_DEM.tif', compress=True)
hs = wbe.multidirectional_hillshade(smoothed_dem)
wbe.write_raster(hs, 'smoothed_hs.tif')
```

See how WbW compares

WbW licensing options

With all the effort that went into developing WbW, unfortunately we couldn't release it as an open-source project—just like you, we have to pay the bills to keep the lights on. However, we truly did want to make this product accessible to as many people as possible. And so, we priced an annual license at only **$10 USD per seat** (before applicable tax), making it ideal for large-team enterprise and research environments and for teaching laboratories. We are so confident that WbW will save you time and resources that we know the $10 price tag will quickly pay for itself!

Buy WbW today



```python
import whitebox_workflows as wbw

wbe = wbw.WbEnvironment()

# Download the 'Kitchener_lidar' sample data...
wbe.working_directory = wbw.download_sample_data('Kitchener_lidar')
print(f'Data have been stored in: {wbe.working_directory}')

# Read in an existing lidar data set
lidar = wbe.read_lidar('1km175400481102018LLAKEERIE.laz')
num_points = lidar.header.get_num_points()

# Now, create a new Lidar object.
lidar_out = wbe.new_lidar(lidar.header)
lidar_out.vlr_data = lidar.vlr_data

print('Filtering point data...')
for i in range(num_points):
    point_data, time, colour, waveform = lidar.get_point_record(i)

    # Now let's filter the data based on return data...
    if point_data.is_first_return() or point_data.is_intermediate_return():
        # Save the point to lidar_out
        lidar_out.add_point(point_data, time, colour, waveform)


wbe.write_lidar(lidar_out, "new_lidar.laz") # Write lidar_out to file
```

If you still aren't convinced, take a look at the Whitebox Workflows for Python User Manual and see for yourself. Or better yet, install it today (via the convenient 'pip' install) and start your free 3-day trial. Licenses can be purchased from the Whitebox Geospatial Inc website now.

WbW v1.0 is available for Windows, MacOS (Intel and Apple Silicon), and Linux (x86_64). In creating this product, our goal was to provide a more intimate way to interact with geospatial data while getting this product in the hands in as many geospatial professionals as possible. Although the WbOC still remains the foundational base of the Whitebox Platform, WbW is the clear winner for next-level geoprocessing! We hope that you have as much fun using it as we have.

See how WbW compares