H3: A Hexagonal Hierarchical Geospatial Indexing System

H3 is a geospatial indexing system using a hexagonal grid that can be (approximately) subdivided into finer and finer hexagonal grids, combining the benefits of a hexagonal grid with S2's hierarchical subdivisions.

Documentation is available at https://h3geo.org/. Developer documentation in Markdown format is available under the dev-docs directory.

- Post **bug reports or feature requests** to the GitHub Issues page
- Ask **questions** by posting to the H3 tag on StackOverflow
- There is also an H3 Slack workspace

Installing

We recommend using prebuilt bindings if they are available for your programming language. Bindings for Java, JavaScript, Python, and others are available.

On macOS, you can install H3 using `brew`:

```
brew install h3
```

Otherwise, to build H3 from source, please see the following instructions.

Building from source

Still here? To build the H3 C library, you'll need a C compiler (tested with `gcc` and `clang`), CMake, and Make. If you intend to contribute to H3, you must have clang-format installed and we recommend installing ccmake and LCOV to configure the `cmake` arguments to build and run the tests and generate the code coverage report. We also recommend using `gcc` for the code coverage as some versions of `clang` generate annotations that aren't compatible with `lcov`. Doxygen is needed to build the API documentation.

Install build-time dependencies

- Alpine

```
# Installing the bare build requirements
apk add cmake make gcc libtool musl-dev
```

- Debian/Ubuntu

```
# Installing the bare build requirements
sudo apt install cmake make gcc libtool
# Installing useful tools for development
sudo apt install clang-format cmake-curses-gui lcov doxygen
```

- macOS (using `brew`)

First make sure you have the developer tools installed and then

```
# Installing the bare build requirements
brew install cmake
# Installing useful tools for development
brew install clang-format lcov doxygen
```

- Windows (Visual Studio)

You will need to install CMake and Visual Studio, including the Visual C++ compiler. For building on Windows, please follow the Windows build instructions.

- FreeBSD

```
# Installing the build requirements
sudo pkg install bash cmake gmake doxygen lcov
```

Compilation   When checking out the H3 Git repository, by default you will check out the latest development version of H3. When using H3 in an application, you will want to check out the most recently released version:

```
git checkout v$(<VERSION)
```

From the repository root, you can compile H3 with:

```
mkdir build
cd build
cmake -DCMAKE_BUILD_TYPE=Release ..
make
```

All subsequent `make` commands should be run from within the `build` directory.

Note: There are several ways to build H3 with CMake; the method above is just one example that restricts all build artifacts to the `build` directory.

You can install system-wide with:

```
sudo make install
```

If using the method above, from the repository root, you can clean all build artifacts with:

```
rm -rf build
```

Testing   After making the project, you can test with `make test`. You can run a faster test suite that excludes the most expensive tests with `make test-fast`.

Coverage   You can generate a code coverage report if `lcov` is installed, and if the project was built with the `CMAKE_BUILD_TYPE=Debug` and`ENABLE_COVERAGE=ON` options. For example, from a clean repository, you could run:

```
mkdir build
cd build
cmake -DCMAKE_BUILD_TYPE=Debug -DENABLE_COVERAGE=ON ..
make
make coverage
```

You can then view a detailed HTML coverage report by opening `coverage/index.html` in your browser.

Benchmarks You can run timing benchmarks by building with the `CMAKE_BUILD_TYPE=Release`, and running `make benchmarks`:

```
mkdir build
cd build
cmake -DCMAKE_BUILD_TYPE=Release ..
make
make benchmarks
```

Documentation You can build developer documentation with `make docs` if Doxygen was installed when CMake was run. Index of the documentation will be `dev-docs/_build/html/index.html`.

After making the project, you can build KML files to visualize the hexagon grid with `make kml`. The files will be placed in `KML`.

To build the documentation website, see the website/ directory.

Usage

From the command line

To get the H3 index for some location:

```
./bin/latLngToCell --resolution 10 --latitude 40.689167 --longitude -74.044444
```

10 is the H3 resolution, between 0 (coarsest) and 15 (finest). The coordinates entered are the latitude and longitude, in degrees, you want the index for (these coordinates are the Statue of Liberty). You should get an H3 index as output, like `8a2a1072b59ffff`.

You can then take this index and get some information about it, for example:

```
./bin/cellToBoundary --index 8a2a1072b59ffff
```

This will produce the vertices of the hexagon at this location:

```
8a2a1072b59ffff
{
   40.690058601 -74.044151762
   40.689907695 -74.045061792
   40.689270936 -74.045341418
```

```
    40.688785091 -74.044711031
    40.688935993 -74.043801021
    40.689572744 -74.043521377
}
```

You can get the center coordinate of the hexagon like so:

```
./bin/cellToLatLng --index 8a2a1072b59ffff
```

This will produce some coordinate:

```
40.6894218437 -74.0444313999
```

From C

The above features of H3 can also be used from C. For example, you can compile and run examples/index.c like so:

```
cc -lh3 examples/index.c -o example
./example
```

You should get output like:

```
The index is: 8a2a1072b59ffff
Boundary vertex #0: 40.690059, -74.044152
Boundary vertex #1: 40.689908, -74.045062
Boundary vertex #2: 40.689271, -74.045341
Boundary vertex #3: 40.688785, -74.044711
Boundary vertex #4: 40.688936, -74.043801
Boundary vertex #5: 40.689573, -74.043521
Center coordinates: 40.689422, -74.044431
```

Contributing

Pull requests and Github issues are welcome. Please see our contributing guide for more information.

Before we can merge your changes, you must agree to the Uber Contributor License Agreement.

Legal and Licensing

H3 is licensed under the Apache 2.0 License.

DGGRID Copyright (c) 2015 Southern Oregon University