GH Extractor

Getting Started:

Note 1: *The primary files needed in this repository are contained within the `lib/` and `src/` directories.*

> `lib` - contains the `gson` which is used to pretty-JSONify the responses from GitHub's API, as well as the Oracle SQL driver if interaction with a database is needed.
> `src` - contains the `GHExtractor.java` class as well as a `Tests.java` class for running tests within the repository itself.

Note 2: *As this tool is used to read and download raw content from files on github, an internet connection is required to run it without any errors.*

Note 3: *In it's current form, this tool cannot read files who's raw data exists in binary form as it is primarily used to read `*.ddl` , `*.sql` , `*.js` , `*.sh` or `*.py` scripts currently.*

How to use:

The `GHExtractor` class can be instantiated using the following 3 parameters:

> 1. `tRepo (String)` : Target repository name to pull file(s) from.

Warning: if the name provided in `tRepo` is not spelled correctly, GHExtractor will attempt to find a matching name by searching a paged list of every repository owned by the provided username. While this is convenient, it is also computationally expensive and runs the risk of hitting the GitHub API's rate-limit. If this happens, GHExtractor will suspend itself until the rate-limit resets.

> 1. `uName (String)` : GitHub username; specifically the username the repository is published under.
> 2. `authT (String)` : Authentication token to access the repository, if any

is required. Otherwise, use an empty string {""}.

The `PropertiesConfig` class can be used to read from a .env file when applying sensitive information like OAuth Tokens or passwords to the GHExtractor class. It is instantiated with only one parameter...

1. `envFile (String)` : Name and path to environment variables file (".env" for example.)

... after instantiating, call the `*.set()` method to have all properties in the environment variables file loaded. These properties can then be referenced elsewhere by `System.getProperty("{PROPERTY NAME}");` .

Note: *The syntax for the environment variables file should be as follows:*

```
PropertyName1=PropertyValue1
PropertyName2=PropertyValue2
```

Once instantiated, files can be downloaded by calling the `GetFileFromGithub()` method with the following 2 parameters:

1. `fileName (String)` :
   The specific filename, including extension, to search the repository for and download. If {"*"} is used here, GHExtractor will download everything listed within the .directorymap file.
2. `inDirectory (String)` :
   In the event that you are attempting to read or download a file that is within a specific directory within a repository, you may specify that directory here, otherwise leave the field blank with an empty string (e.g. `""`).
3. `outDirectory (String)` :
   The target directory to download the file to, from the reference-point of the project root directory. If left as an empty string, or if the provided directory doesn't exist, this will download to wherever GHExtractor is located. Otherwise, it will build out a replicated directory that matches what is on github, similar to a `git clone` command.

If you need to change which repository is the current target repository, you can do so by invoking `setTargetRepository()` with the following 1 parameter:

1. `targetRepository (String)`:
   The name of the repository you would like to search within. This repository must belong to the username that the current GHExtractor object was instantiated with.

Finally, if you would like to suppress console output, you may do so by invoking the `toggleVerbose()` method, which takes no parameters.

<u>Usage Examples</u>:

```
public static void main(String[] args) {
        GHExtractor test_extractor;

        try {
            // Load properties from a .env file
            PropertiesConfig props = new PropertiesConfig(".env");
            props.Set();

            test_extractor = new GHExtractor(
                    "test-repo",
                    System.getProperty("userName"),
                    System.getProperty("OAuth"   )
            );

            // Enable a more verbose console output
            test_extractor.toggleVerbose();

            // Download an entire sub-directory within a repository...
            test_extractor.GetFileFromGithub("*", "demo.db", "files-
from-github");

            // Download a specific file from a target repository into a
nested local directory...
            test_extractor.GetFileFromGithub("dept.ddl",
"demo.db/dept", "files-from-github/anotherDirectory");

            // Download the entire target repository...
            test_extractor.GetFileFromGithub("*", "", "files-from-
github");

            // Switch to a new repository and download it to a nested
directory...
            test_extractor.setTargetRepository("GHExtractor");
            test_extractor.GetFileFromGithub("*", "", "files-from-
github/Secondary-outDirectory");

        }
        catch (IOException ioe) {
            ioe.printStackTrace();
        }
}
```