Moving Ultimate Datetime to GitHub

Created a free GitHub account – [jay@55penwood.com](mailto:jay@55penwood.com). May need to upgrade to a Developer account to get private repository. Cost $7 / month.

Installed GitHub extension in Visual Studio on the Pavilion dm4 laptop. After installation, go to Team-> Manage Connections to connect to GitHub. Just click the link in Team Explorer when it appears.

Attempt to add the Ultimate Datetime Solution to source control:

1. Right-click the solution and select Add Solution to Source Control. Generates an error indicating not all of the projects are in the solutions directory.
2. Determined TZDatabase is at the same level as the solution directory. Remove this project from the solution by right-clicking the project then selecting Remove. The Solution now contains 4 projects. A Rebuild of the solution is successful.
3. Now attempt to add the solution to source control. The operation is successful. Under Team Explorer, Ultimate Datetime now appears as a local Git Repository.
4. Upgrade to Developer’s version of GitHub to support private repositories.
5. There will be a ‘remote’ repository on the local machine and the repository on Github. The commits will be to the local repository, which then will be synced to the Github repository.
6. From Team Explorer, elect to Synch the repository. Click Publish to Github. Indicates this repository does not have a remote. Fill out the form and check Private Repository. Then click Publish. Indicates there is already a repository named Ultimate Datetime for the current account. Logon to Github then delete the repository. Attempt to publish again. Succeeds.
7. On github, the Ultimate-Datetime repository now exists. All of the files have been uploaded to the project in the repository.
8. Now add the Julia files and the Document files to the Ultimate Datetime project. When these changes were committed and synched, only the project file was updated – the files did not get copied. This problem can be solved by moving the Julia Integration directories and Documents directories under the Ultimate Datetime Datatype \ Ultimate Datetime directory, thereby putting it in the same folder as the solution. This is the same issue as in point 1. When this was done, all of the files became part of the repository.
9. The repository on github was created from the laptop. To instantiate the repository on the rackstation for use by the desktop, need to clone the repository. To accomplish this:
   1. Install github extension for Visual Studio
   2. Run Visual Studio. From Team Explorer, connect to github
   3. Under the Ultimate Datetime Datatype folder, create an Ultimate Datetime git directory.
   4. From Team Explorer, elect to clone the repository. Select the Ultimate Datetime repository. For the local folder, select Ultimate Datetime git.
   5. A new folder, Ultimate Datetime git/Ultimate-Datetime is created with all of the directories.
   6. The solution appears in Team Explorer. Double click to open the solution.
10. Solution opens. Attempt to Rebuild the solution.
    1. Rebuild fails because include files required to build TZ Localtime, which are contained in the TZDatabase directory cannot be found.
    2. In the Ultimate Datetime Git folder, create a TZ Database directory, then create a TZ Database subdirectory. Copy private.h and tzfile.h into this subdirectory. Build succeeds and tests run with no errors.
    3. While the above works, since the new directories are not under the Ultimate-Datetime directory, they are not included in the repository.
    4. Move the two header files to the Ultimate Datetime directory (where the main.c test file is located). Change the TZ Localtime asctime.c and localtime.c references to point to that directory, as well as the main.c reference.