Tool Buddy

Developer Documentation

Overview:

- Installation
- Framework
  - Qt API
  - Qt Resources
- Program Structure
  - Project File
  - Resource File
- Languages
  - C++
  - SQL
  - HTML/CSS
- Code Convention
  - Naming Convention
  - Self-Documenting Code
- Other Software
  - SQLite3
  - ZXing

Installation

Downloading the Files

You will need to download all of the files from the repository located at https://github.com/stmu-eng/final-project-laurenesco. Store them all in the same directory to prevent any compilation errors.

Downloading SQLite3

Since SQLite3 is the database engine that this application is programmed for, you will need it installed to run the program. However, if you choose, you can simply continue to use the database found in the repository, which the application is already set up to use.

If you choose to create another, you can download the version appropriate for your machine at https://www.sqlite.org/download.html. For my Windows 10 laptop, this was the download 'sqlite-tools-win32-x86-3380300.zip', which includes the command line tool we will use to create the database. You will need to extract the contents of this folder to your desired location.

Once you have this done, you can create your database. Run SQLite3 and a command shell should appear. SQLite3 does not support normal commands such as just dir, cd, pwd. Instead you must use the syntax .shell <command> or .cd <directory>. For example, '.cd <directory/name>' will take you to the desired directory. To print the working directory, use '.shell cd'.

Navigate to your desired directory (usually the folder containing the rest of the applications files) and create the new database.

Downloading SQLite Manager

SQLite Manager is not necessary to run this application, but is extremely useful to check on database submissions and perform SQL commands outside of the program. It is a Firefox extension which can easily be downloaded from the themes and add ons

menu. To test the database after downloading - Once in SQLite Manager, navigate to File->Open a Database, then select the database you wish to open. You will not get feedback, so to make sure that the database is open you can execute a SQL statement such as ‘select * from tools’, which will return all the data from the table tools in the test database test.db.

Execution

Once all is done, you can compile and run the program through Qt. There is not a standalone installer at this point.

Framework

QT API

Qt is an API which supports C++ and creates UI files based off of the code you create. In the designer, you can visually curate the UI files to create the GUI of your dreams. This API works almost entirely off of object-oriented programming, allowing for a lot of flexibility when creating your code. Qt is well documented, with its own documentation website, and there are also numerous forums which detail various problems and solutions in Qt.

QT Resources

As previously mentioned, there are many resources to help you work with Qt. The official Qt documentation website is located at: https://doc.qt.io/ and the official Qt forums are located at https://forum.qt.io/. Aside from those two resources, there are numerous other forums and videos outlining tutorials regarding various Qt features.

Project File

The project file (.pro) is the file containing all the information qmake will use to compile and run your program. In this file the programs headers, source files, configurations, Qt modules, and more will be stored. A lot of this is created dynamically - the only thing

you will need to change is adding or removing modules based on how this application evolves. The full documentation can be found here: https://doc.qt.io/qt-5/qmake-project-files.html.

Resource File

The resource file (.qrc) is where various resources for the program are stored. In the case of this application, that is merely some .png images. This file can be used to store small resources that will be called on in the application. The official documentation can be found here: https://doc.qt.io/qt-5/resources.html.

Languages

C++

This program utilizes C++, SQL, and HTML in order to bring the application to life. The majority of the program is written in C++ (86.9% according to GitHub). Qt uses C++ because, as previously mentioned, the entire framework runs on object oriented programming. C++ resources can be found here: https://www.cplusplus.com/ on the official C++ website.

SQL

Thanks to the sql module, we can also utilize SQL in the program. This will mainly be found in queries (in the seach page, edit item, and add item file), and since the database is a SQLite3 database, that is the standard which you can follow. More information about SQLite3 can be found here: https://www.sqlite.org/docs.html.

HTML

There is a small amount of HTML used to create the Data Dictionary. This file is used to present a table to the user which describes the schema of the table. It displays the table name, column names, and variable types. This file can be updated as the application grows.

Code Convention

Naming Convention

The naming convention followed in the application is as follows:

File names:	All lowercase, using _ for spaces
Class Names:	Capitalized, using _ for spaces
Variable Names:	No spaces, lowercase first word, capitalize the first letter of every subsequent word.

This naming convention should be followed at all times to ensure readability and maintainability of the application.

Self-Documentation

The application is also self documenting, meaning there are comments heading each file, above every function, and in any place where the code may be ambiguous or hard to understand. This is especially important to prevent future developers from removing important code that may seem to not serve much purpose, as well as to help future developers easily understand and improve the program.

Other Software

SQLite3

SQLite3 is the database engine of choice for this application because it is very easy to set up and can be easily manipulated from within Qt. It is a C based library that is stable and can perform across various platforms. The great thing about SQLite is that it doesn't require any other machines, mechanisms, or even a server to function. This

makes the databases easily accessible, manipulatable, and portable. The official documentation can be found here: https://www.sqlite.org/index.html and the engine can be downloaded here: https://www.sqlite.org/download.html.

ZXing

ZXing (Zebra Crossing) is an open source barcode/qr-code processing library. It ports to C++, and there are other open source libraries such as QZXing dedicated to helping you decode QR codes with ZXing in Qt. The QZXing Wiki contains detailed instructions on how to implement the code and get started. This can be found at: https://sourceforge.net/p/qzxing/wiki/Home/?version=3.