Code Documentation

The source of this application is split into components, each with its own purpose. The main components are :

- *main.cpp*
- *chessSource.cpp*
- *singlePlayer.cpp*

Apart from *main.cpp*, all other source files have a **header**. Therefore *chessSource.cpp* will be linked to *main.cpp* via **chess.h**, And so on.

The other sources the application uses are :

- *buttonSource.cpp*
- *labelSource.cpp*
- *translatorSource.cpp*
- *gameSource.cpp*

As mentioned above, each source has its own purpose.

In *main.cpp* you can find the GUI, and the mechanics of the application.

In *chessSource.cpp*, chess rules are defined.

In *singlePlayer.cpp*, the computer's algorithms can be found.

gameSource.cpp is responsible for encoding / decoding into and from PGN format when saving / loading games.

Apart from *main.cpp* and *singleplayerSource.cpp* I don't recommend changing the other sources. If you desire however to add more languages to the application, you will need to make changes in *translatorSource.cpp*.

If you change something into the sources that aren't *main.cpp*, make sure you also make the required change into the header file (if necessary).

When changing the GUI, **button** and **label** are the two main classes you will use. To create a new button simply type in the following code :

*"**button [buttonName];***

[buttonName].create(posX , posY, sizeX, sizeY);

***[buttonName].update(hover);** "*

The second line of code will create a new button at posX , posY with the size sizeX , sizeY. The third one will change its color when *hover* Boolean is changed.

To create a label, all u have to do is type in :

*"**label [labelName];***

[labelName].create(posX , posY, CharacterSize, text, font);"

All the fonts that can be used are stored in **res / fonts**.

Should you want to work on the singleplayer computer, you need to know the following:

The computer makes its moves based on a Min-Max algorithm. Increasing its parameter *depth*, will add more complexity to it.

Each position is evaluated via *GetScore()*. Changing that will also affect its decisions.

Scheme of how the decisions are taken.

Get all possible moves > For each move set its score via min-max >

Get the highest score moves > Make a random move out of all final moves.