Qt Topics

27.12.2014

Multidimensional Arrays in QSettings

I started a thread here:
http://www.qtforum.de/forum/viewtopic.php?f=1&t=16671

Worker-Threads

QThread
Wenn man eine GUI hat, in der dann rechenintensieve Routinen laufen, dann sollten diese unbedingt in einem eigenen Thread laufen!

Grundprinzip:

- Es gibt den GUI-Thread (z.B. MainWindow) und den Arbeiter-Thread.
- Innerhalb des mainWindows wird der Arbeiter-Thread mit

  ```
  QThread workerThread;
  ```

  angelegt.

- Weiters muss man die rechenaufwändigen Sachen in eine eigene QObject-Klasse packen, also von QObject ableiten.

  ```
  class Worker : public QObject
  {
    ...
  }
  ```

- In der GUI wird dann zur Laufzeit ein Objekt von Worker anglegt, und dieses dann in den Thread geschoben:

  ```
  worker = new Worker();
  worker->moveToThread(&workerThread);
  ```

- Hat man alle Signals und Slots vom mainWindow und worker verbunden, startet man den Thread:

  ```
      connect(&workerThread, &QThread::finished, worker,
  &QObject::deleteLater);
      connect(this, MainWindow::operate, worker, &Worker::process);
      connect(worker, &Worker::finished, this, &MainWindow::finished);
      connect(worker, &Worker::currentFileNameChanged, this,
  &MainWindow::updateFileName);
  ```

```
    workerThread.start();
```

Somit läuft das Objekt nun in einem eigenen Thread, und wir mit einem

```
emit operate();
```

angestartet. Von dem Thread aus kann man dann ganz normal seine Signals an mainWindow schicken, und dort entsprechend die GUI updaten, z.B. einen QProgressbar.

software, deutsch, c++