

# Многопоточность в C/C++

## Native C/C++

- Многопоточное программирование в C++ на YouTube канале [#SimpleCode](#)

## Qt-library

- Лев Алексеевский. Многопоточность в Qt:
  - Пример с `moveToThread()`;
  - Наследование от `QThread` с переопределением метода `run()`;
  - [Фоновое задание](#) и [остановка потока](#).
- [Поддержка потоков в Qt 4.5.2](#) по материалам сайта [CrossPlatform.Ru](#)
- Краткая реализация подхода с `moveToThread()`.  
Предполагается, что `execute()` по завершении работы вызовет `emit finished()`:

```
#include <QThread>
.
.
// В конструкторе основной формы:
// а) создание членов класса основной формы;
QThread *thread = new QThread;
Worker *worker = new Worker();
// б) перемещение worker'а в созданный поток
worker->moveToThread(thread);
// в) установка необходимые соединения
// в.1) сигнал finished с флагом успешности.
connect(worker, &Worker::finished, this, [](bool state){
    if (state) {
        std::cout << "Worker отработал успешно\n";
    }
    else {
        std::cout << "Worker столкнулся с ошибкой\n";
    }
});
// в.2) выполнение полезной работы
// при запуске потока начинается выполнение Worker::execute()
connect(thread, &QThread::started, worker, &Worker::execute);
// в.3) при сигнале finished worker'а выход из потока и его закрытие
connect(worker, &Worker::finished, thread, &QThread::quit);
// в.4) удаление объекта worker
connect(worker, &Worker::finished, worker, &QObject::deleteLater);
// в.5) удаление, ставшего ненужным созданного ранее потока thread
connect(thread, &QThread::finished, thread, &QObject::deleteLater);
// г) Запуск созданного и настроенного потока на выполнение
thread->start();
```

Last update: 2024/06/05  
00:11

itechnology:cpp:multithread <https://www.jurik-phys.net/itechnology:cpp:multithread>

---

From:

<https://www.jurik-phys.net/> - **Jurik-Phys.Net**

Permanent link:

<https://www.jurik-phys.net/itechnology:cpp:multithread>

Last update: **2024/06/05 00:11**

