



<b>ID</b>	CS001427	<b>Creation date</b>	June 16, 2009
<b>Platform</b>	S60 3rd Edition, FP1, FP2 S60 5th Edition	<b>Tested on devices</b>	Nokia 5800 XpressMusic
<b>Category</b>	Qt for Symbian	<b>Subcategory</b>	Base/System

**Keywords (APIs, classes, methods, functions):** QThread

## Overview

This code snippet demonstrates how to implement platform-independent threads in Qt by using the `QThread` class. Instead of starting in `main()`, the execution of QThreads begins in `run()`. By default, `run()` starts the event loop by calling `exec()`.

**Note:** In order to use this code, you need to have Qt for S60 installed on your platform.

## Preconditions

- Install latest Qt for S60 see [Qt for S60 - Installation packages](#)
- Check this link for installation guide: [How to install the package.](#)
- Go through this article: [Getting started with Qt for S60](#)

## Header (mythread.h)

```
#include <QObject>
#include <QThread>

class MyThread : public QThread
{
    Q_OBJECT

public:
    MyThread(QObject* parent = 0);
    virtual ~MyThread();
};
```

```

public: // From QThread
    void run();

signals:
    void dataChanged();

public slots:
    void setData(int someData);

private:
    int data;
};

```

## Source (mythread.cpp)

```

#include "mythread.h"

MyThread::MyThread(QObject* parent) : QThread(parent)
{
}

MyThread::~MyThread()
{
}

void MyThread::run()
{
    // TODO: there you can run some part of your code in
    // different thread that rest of the application

    // You can create needed classes here or also in MyThread construction.

    // Thread enters the event loop and waits until exit() is called
    exec();
}

void MyThread::setData(int someData)
{
    data = someData;

    emit dataChanged();
}

```

## Starting thread

```

thread = new MyThread(this);
thread->start();

```

## See also

- [CS001428 - Using your own class as a signal and slot parameter in QThread](#)
- For more information about QThread, see [QThread Class Reference](#).

## Postconditions

A part of the code is executed in a separate thread.