



ID	...	Creation date	26 January 2009
Platform	S60 3rd Edition FP1, S60 3rd Edition FP2, S60 5th Edition	Tested on devices	Emulator
Category	Qt for Symbian	Subcategory	

Keywords (APIs, classes, methods, functions): QByteArray

Overview

This code snippet demonstrates how to handle individual bits in [Qt for S60](#).

Qt provides a QByteArray class which can be use to create an array of the bits. Each bit of this array can be handle saperately. It also supports bitwise AND, OR, NOT, XOR operation similar to the c++.

Preconditions

- Download and Install latest version [Qt for Symbian - Installation packages](#) which has links on how to install the latest version

Various Function

- Replaces len bytes from index position pos with the byte array after, and returns a reference to this byte array.

```
QByteArray x("Say yes!");
QByteArray y("no");
x.replace(4, 3, y);
```

- Sets the byte array to the printed value of n in base base (10 by default) and returns a reference to the byte array. The base can be any value between 2 and 36.

```
QByteArray ba;
int n = 63;
```

How_to_handle_individual_bits_using_QBitArray_in_Qt_for_Symbian

```
ba.setNum(n);
ba.setNum(n, 16);
```

- Returns the number of bytes in this byte array.

```
QByteArray ba("Hello");
int n = ba.size();
ba.data()[0];
ba.data()[4];
ba.data()[5];
```

Source File

More About QBitArray visit:<http://pepper.troll.no/s60prereleases/doc/qbytearray.html>

```
#include <QApplication>
#include <QBitArray>
#include <QWidget>
int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QWidget new QWidget;
    QBitArray a; // Create a bit array x of the size 15 and set all the bits. Second argument
// Second way to initialize a bit array
    QBitArray a
    resized(2);
    [0] = false;
    [1] = true;
// Third way to initialize a bit array
    QBitArray a
    setBit(0, false);
    setBit(1, true);
    QBitArray x
    setBit(2, true);
// x: [ 0, 0, 1, 0 ]

    QBitArray y
    setBit(3, true);
// y: [ 0, 0, 0, 1 ]

    |= y; // XOR operation
// x: [ 0, 0, 1, 1 ]

    &= y; // AND operation
// x: [ 0, 0, 0, 1 ]

    ^= y; // XOR operation
// x: [ 0, 0, 0, 0 ]

    = ~x; // NOT operation
// x: [ 1, 1, 1, 1 ]
}
```

More About QBitArray

Various Function