NXTs per Bluetooth steuern

Um mehrere NXTs per Bluetooth zu steuern und auszulesen braucht man, um die Geschwindigkeit zu erhöhen (die Abfrage dauert lange, wenn das Bluetooth von einem NXT zum nächsten wechseln muss), 2 Bluetooth Dongles, die über 2 verschiedene Stacks (die Software, die die Komunikation per Bluetooth steuert) nötig. Wir haben dabei den internen Stack (Windows) und ein externes Dongle mit Widcom/Broadcom Stack verwendet. Jedes Dongle bringt sein eigenes Stack mit, das heisst man kann nicht frei wählen. Damit beide Stack gleichzeitig abgefragt werden, wird die Datei **threads.cpp** eingebunden und 2 Auslese/Schreibfunktionen gleichzeitig als Threads gestartet (In MarioFrmApp::OnInit() ;).
Zur besseren Erkennung, kann man die einzelnen NXTs mit der Software, die direkt von Lego kommt, umbennen.

Verbindungsaufbau zum COM-Port:
In der Bluetooth Software vom Dongle, das man verwenden will, wählt man den NXT, der verbunden werden soll. Danach muss auf beiden Seiten dasselbe Passwort eingegeben werden. Nun kann man vom PC aus eine „Dev-B"-Verbindung starten. In den Eigenschaften steht, auf welchem COM-Port, der NXT verbunden wurde. Dies wiederholt man am anderen Stack / an den anderen NXTs.
Software:

Unsere Software besteht aus 2 Teilen, der *GUI* und der *Kommunikationsebene* zu den NXTs. Zum kompilieren eignet sich „wxdevcpp", da dort die benötigten Dateien für die GUI bereits drin sind. Die GUI Bibiothek ist wxWidgets.

Die **serial.cpp** übernimmt die Kommunikation per COM-Port.
Mit der **mailbox.cpp** kann man strings in die Inbox des NXT schreiben / den NXT abfragen.

Eine Verbindung nimmt man so auf:

```
Serial NXT1;        //Definition der Verbindung zu NXT1
Mailbox NXT1M = Mailbox(&NXT1); //Die Mailbox von NXT1
```

In unserer verbindeBT() Funktion, wird danach die Verbindung aufgebaut und der String „Starte Verbindung" gesendet. In den marioSteuer() / gegner() Funktionen werden die Mailboxen der NXTs abgefragt. Dies geschieht mithilfe von

```
NXT1M.write("STRING",INBOX);
```

resp.

```
NXT1M.read(OUTBOX, true);
```

Da die beiden Funktionen als Threads gestartet worden sind, muss die gemeinsame Variable in die CriticalSection geschrieben werden. (Damit nicht beide Thread gleichzeitig darauf zugreifen.)
Die übrigen Funktionen, gehören zu GUI.

Programmablauf:
Die main Funktion erstellt zuertst ein Fenster der GUI und ruft dabei MarioFrmApp::OnInit auf. Von dort werden die beiden Threads gestartet. Mithilfe von „reinterpret_cast<void *>(this)" wird der Pointer auf das MarioFrmApp weitergegeben. In marioSteuer() wird das mit „MarioFrmApp * threadkill = reinterpret_cast<MarioFrmApp *>(arg);" aufgelöst. Dadurch kann von wxWidgets der Tastendruck übergeben werden. Dies geschieht in MarioFrm.cpp in MarioFrm::OnKeyDown(wxKeyEvent& event).

Auf dem NXT:
string request;

```
ReceiveMessage(INBOX, true, request); //Code um nachrichten zu erhalten
SendResponseString(OUTBOX,request); //Code zum versenden an den Laptop.
```

Viel Erfolg beim Programmieren!