Subject: "DoEvents" or "wxSafeYield" equivalent in Upp++
Posted by koldo on Sun, 31 Aug 2008 00:09:12 GMT
View Forum Message <> Reply to Message

Hello all

I would like to use some kind of "poor man" multithreading for long "for" loops. Like:

```
for (int i = 0; i < MANY_LOOPS; ++i) {
  ThingsToDo();
  DoEvents();
}
```

where DoEvents() lets some kind of "cooperative multitasking" by:
- disabling the user input to all program windows
- asking the Upp++ main loop to handle the pending messages in the windowing system
- re-enabling it again afterwards

Disabling user input would avoid unwanted reentrance of code.

In this moment this will be enough for me and much easier to handle than real multitasking with the gui.

Best regards

Subject: Re: "DoEvents" or "wxSafeYield" equivalent in Upp++
Posted by mirek on Sun, 31 Aug 2008 07:17:11 GMT
View Forum Message <> Reply to Message

You should be able to do this using:

```
static  Vector<Ctrl *> Ctrl::GetTopCtrls();
Vector< Ptr<Ctrl> > DisableCtrls(const Vector<Ctrl *>& ctrl, Ctrl *exclude = NULL);
static  void Ctrl::ProcessEvents();
void EnableCtrls(const Vector< Ptr<Ctrl> >& ctrl);
```

Mirek

Subject: Re: "DoEvents" or "wxSafeYield" equivalent in Upp++
Posted by koldo on Sun, 31 Aug 2008 17:53:38 GMT
View Forum Message <> Reply to Message

Hello luzr

I have tried this, but it blocks all until the loop ends.

```
for (int i = 0; i < 100; ++i) {
 Sleep(500);
 Vector<Ctrl *> topCtrls = Ctrl::GetTopCtrls();
 Vector< Ptr<Ctrl> > ctrls = DisableCtrls(topCtrls);
 Ctrl::ProcessEvents();
 EnableCtrls(ctrls);
}
```

But this simply works perfect. (taking care of avoiding the user to run this code while it is still running).

```
for (int i = 0; i < 100; ++i) {
 Sleep(500);
 Ctrl::ProcessEvents();
}
```

Thank you!
Koldo

Subject: Re: "DoEvents" or "wxSafeYield" equivalent in Upp++
Posted by mirek on Sun, 31 Aug 2008 18:33:20 GMT
View Forum Message <> Reply to Message

koldo wrote on Sun, 31 August 2008 13:53Hello luzr

I have tried this, but it blocks all until the loop ends.

```
for (int i = 0; i < 100; ++i) {
 Sleep(500);
 Vector<Ctrl *> topCtrls = Ctrl::GetTopCtrls();
 Vector< Ptr<Ctrl> > ctrls = DisableCtrls(topCtrls);
 Ctrl::ProcessEvents();
 EnableCtrls(ctrls);
}
```

But this simply works perfect. (taking care of avoiding the user to run this code while it is still running).

```
for (int i = 0; i < 100; ++i) {
 Sleep(500);
 Ctrl::ProcessEvents();
```

}

Thank you!
Koldo

I thought that "block all" was the purpose of disabling

Note that there is parameter "exclude" in DisableCtrls. Usually, you want to leave something enabled to process cancel messages, right?

Mirek
