
Subject: Re: [Proposal] Adding a GLLock struct to GLCtrl

Posted by [Klugier](#) on Thu, 03 Sep 2020 21:46:23 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello Xemuth and Mirek,

We should think about the drawback of this addition:

- maintenance (current implementation delivered by Xemuth can be simplify, however still the implementation will need to be provided when adding new platform).
- concurrent solution (one thing can be done using two different approach - my solution is better than yours (code review problem))

Drawbacks:

- no need to open new additional layer of indentation
- more...

Backing to the "new layer of indentation", you could always call new function (which will make original one smaller):

```
bool LoadSTL(){ //This function is used when a button is pressed
    bool rendered;
    canvas.ExecuteGL([&] { rendered = renderSTL(); });
    return rendered;

    // Maybe with template magic we could simplify to
    // return canvas.ExecuteGL([&] -> bool { return renderSTL(); });
}
```

```
bool renderSTL() {
    // All methods calls from here could call to OpenGL... The same is true with lock
    approach...
    try {
        //Context is here...
        //Shaders Stuff
        //OpenGL buffer filling and loading
        //Definition of draw
    } catch (Exc e) {
        Exclamation(DeQt(e));
    }
    return false;
}
```

I do not have strong opinion and the solution seems to have more drawbacks than pluses. Would be nice investigating, how we can extend ExecuteGL to return custom types.

Klugier
