

---

Subject: Re: Can a ctrl remove itself?

Posted by [Oblivion](#) on Mon, 16 Jul 2018 10:11:16 GMT

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

---

Hello Giorgio,

Quote: Is there a way for a ctrl to remove itself?

Yes, you can have a ctrl remove itself, by calling `Ctrl::Remove()`.

E.g.

```
class Foo : public TopWindow {  
  
    Button bt;  
  
public:  
    Foo() {  
        SetRect(0,0, 640, 480);  
        CenterScreen();  
  
        bt.SetLabel("Press to remove me!") << [=] { bt.Remove(); };  
        Add(bt.HCenterPosZ(120).VCenterPosZ(24));  
    }  
};
```

This works, because:

Quote:  
Ctrl Tree

Ultimate++ uses a linked list for all the child Ctrl's that have been `Add()`ed to it, partaking of its drawing space. The Ctrl does NOT own its children, but simply references them (`Ptr<Ctrl>`). They should be owned by your application, somewhere in a U++ container, i.e. `Array<Label>` or they are already made members of your application when using Layout files. If a Ctrl is added to another, it is ensured to be properly removed from its previous parent, thus a Ctrl can't be part of 2 trees.

[https://www.ultimatepp.org/srcdoc\\$CtrlCore\\$CtrlDesignConcept\\_s\\_en-us.html](https://www.ultimatepp.org/srcdoc$CtrlCore$CtrlDesignConcept_s_en-us.html)

Best regards,

