

hi guys

in my struggle with a 'scriptable' layout for OSC interfaces i came along the need / idea to make the controls more open to manipulation in terms of their properties (not the hardwired things like callbacks).

so one could imagine a uniform interface like

```
void Ctrl::SetProperty(const String& name, const Value& v);  
void Ctrl::GetProperty(const String& name, Value& v);
```

now this would enable / map things like

```
c.SetProperty("background", Black);  
c.GetProperty("background", v);  
  
c.SetProperty("min", 123);  
c.GetProperty("max", v);  
  
c.SetProperty("xalign", "left");  
c.SetProperty("xpos", 123);  
c.SetProperty("yalign", "center");  
c.SetProperty("ypos", 45);  
  
c.SetProperty("pattern", "%.g");  
c.SetProperty("grid" ValueArray(Vector<int>() << 5 << 6));  
  
//would map to GetData / SetData directly i.e.  
c.SetProperty("data", 345);  
c.GetProperty("data", v);  
  
c.GetProperty("count", v);  
  
c.SetProperty("enable", true);  
c.GetProperty("enable", v);  
  
c.SetProperty("show", false);  
c.SetProperty("title", "MyTitle");  
c.SetProperty("tip", "GoHelp");  
  
//query all gettable properties  
c.GetProperty("rprops", v);  
//query all settable properties
```

```
c.GetProperty("wprops", v);
```

..  
and many more..

this would ease things like a custom gui creation / live edit of controls..

this is just an idea, where std Ctrl could already map to a lot of things by itself.. and derived Ctrl's simply extending the 'dictionary' of properties and managing the properties..

i imagine to extend it on template base if not desired in upp, but having a uniform polymorphic interface already from Ctrl level would help a great deal in this.

i might provide an example implementation soon..  
just wanted to know your opinion.

cheers

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 18 Nov 2010 14:12:47 GMT  
[View Forum Message](#) <> [Reply to Message](#)

got something working, almost the way desired, and without manipulating Ctrl..

it's Callback map based approach to handlers.. using public interface of the controls. the lack of virtualisation made it necessary to pay attention to the derive hierarchy, to invoke things from base first.

attached is a test  
comments/ideas..

#### File Attachments

1) [CtrlProp.rar](#), downloaded 396 times

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [koldo](#) on Thu, 18 Nov 2010 17:28:10 GMT  
[View Forum Message](#) <> [Reply to Message](#)

Hello kohait00

This is the window. How is it used?

## File Attachments

1) [dib.PNG](#), downloaded 1361 times

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 18 Nov 2010 19:03:29 GMT

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

---

sorry, the 'test' case was just veeeeery brief to be able to look at the 'api'.. it was just to show that the respective calls to SetData() has arrived successfully..

the api is very simple

```
bool SetProperty(Ctrl& c, const String& n, const Value& v);  
bool GetProperty(const Ctrl&, const String& n, Value& v);
```

both return false if no property with name n could be found for Ctrl& c. or if in case of SetProperty the Value v had incompatible type..

this is just a very bried first - shot idea, without need of manipulating Ctrl at all but still yielding full support to all top level Ctrls. in this case it is an EditInt beeing passed as Ctrl&, and still can set Min and Max values...so this is a uniform api.

my point is to have a feedback on wheather stuff like that is of use or if there is a more intelligent way to export and access properties of controls in a uniform way other than the pure compile time c++ api, but also as runtime manipulation api..

cheers

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 18 Nov 2010 22:20:47 GMT

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

---

here comes a new Test Case, which i think shows where the idea was going to..

a 'dynamic' property list editor...

i am still not happy with the 'listget' and 'listset' option, but i cant imagine how to filter them out in an effective way..

ideas..

## File Attachments

1) [CtrlPropTest.rar](#), downloaded 431 times

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Wed, 24 Nov 2010 16:51:09 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

hi guys, is there any ideas on that one?  
i'd need to know if it's worth to go in that direction, especially in terms of setting up an OSC  
controls layout  
hope not to come harsh..

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Sat, 27 Nov 2010 17:31:05 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Perhaps I could express my opinion if I knew what is "osc control"

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Sun, 28 Nov 2010 10:01:59 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

in fact, i started a post about osc, which has drifted towards audio interface things in u++

osc is actually a parameters manipulating language / message protocol. there are applications,  
where one can put together control elements, and assign them osc patterns (the parameters they  
will be operating on) and thus have remote control interfaces..

these interfaces can be created in layout editors, where one can edit the controls' properties, like  
size, position etc..

see

<http://hexler.net/software/touchosc>

there is also an editor to download...

[http://opensoundcontrol.org/spec-1\\_0](http://opensoundcontrol.org/spec-1_0)

the specification

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Tue, 30 Nov 2010 20:12:48 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

any idea if/how this can be done better?

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl

---

Posted by [mirek](#) on Wed, 01 Dec 2010 08:00:35 GMT

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

---

I would use some nice macros to automate the most of process...

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl

Posted by [kohait00](#) on Wed, 01 Dec 2010 09:04:46 GMT

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

---

i think macros wont do in my case..since i need it at runtime. macros would fix everything at compile time..

or did i get you wrong? some example?

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl

Posted by [mirek](#) on Wed, 01 Dec 2010 17:00:26 GMT

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

---

kohait00 wrote on Wed, 01 December 2010 04:04i think macros wont do in my case..since i need it at runtime. macros would fix everything at compile time..

or did i get you wrong? some example?

Oh, no, I mean macros to automate what you are already doing, just with less code.

I believe that something like

```
CTRL_PROPERTIES(EditInt)
    PROPERTY(int, Min)
    PROPERTY_SET(bool, "NotNull", NotNull)
    PROPERTY_GET(bool, "NotNull", IsNotNull)
END_CTRL_PROPERTIES
```

should be possible.

Mirek

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl

Posted by [kohait00](#) on Wed, 01 Dec 2010 19:22:44 GMT

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

---

that is, ofcourse better in terms of API, lets see how to fit that in. my problem is that i actually dont want to mess with Ctrl stuff. trying to extend it from public interface point of view, idea is to have typeid() do the entrance point from where to start parsing the properties..and need to resemble a bit the derive hierarchy if meaningfull..  
thanks for the hint.

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 02 Dec 2010 07:44:15 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

still not what i am proud of, but first scores are done  
i think this is quite helpful for dynamic control editing..

#### File Attachments

1) [CtrlPropTest.rar](#), downloaded 369 times

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 02 Dec 2010 08:24:08 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

BTW: found 2 const correctness issues in CtrlCore.h

CtrlCore.h:1199

```
bool  IsInitFocus() const          { return initfocus; }
```

CtrlCore.h:1249

```
int   GetBackPaint() const         { return backpaint; }
```

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Thu, 02 Dec 2010 10:03:52 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Wed, 01 December 2010 14:22that is, ofcourse better in terms of API, lets see how to fit that in. my problem is that i actually dont want to mess with Ctrl stuff. trying to extend it from public interface point of view, idea is to have typeid() do the entrance point from where to start parsing the properties..and need to resemble a bit the derive hierarchy if meaningfull..  
thanks for the hint.

I would rather go with dynamic\_cast... Makes more sense.

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 02 Dec 2010 10:18:00 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

what would be the benefit in terms of organisation of the callback structures? you mean to automate the checking for even unregistered types? that would probably be an advantage, but then, i'd have to check for a whole bunch of types (performance) for each property. i couldnt use mapping anymore..

a help would be so kind of a

```
virtual String Ctrl::PropertyHook() const { return "Upp::Ctrl"; }
```

which i could use for querying maps..

then, derived controls, that dont extend property stuff would still be mapped easily (because their base specifies the hook correctly)..but this means extending the Ctrl API. but only this one single method..what do you think about it?

a EditInt i.e would, then, need to

```
virtual String EditInt::PropertyHook() const { return "Upp::EditInt"; }
```

to announce, that it wants own property entrance in the map

also, String(typeid(CLASSNAME).name()) could be used..

then, a #define would do it to announce

```
#define HASPROPERTY \  
virtual String PropertyHook() const { return String(typeid(CLASSNAME).name()); }
```

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Thu, 02 Dec 2010 10:22:21 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Thu, 02 December 2010 05:18 what would be the benefit in terms of organisation of the callback structures? you mean to automate the checking for even unregistered types? that would probably be an advantage, but then, i'd have to check for a whole bunch of types (performance) for each property. i couldnt use mapping anymore..

a help would be so kind of a

```
virtual String Ctrl::PropertyHook() const { return "Upp::Ctrl"; }
```

which i could use for querying maps..

then, derived controls, that dont extend property stuff would still be mapped easily..but this means extending the Ctrl API..what do you think about it?

a EditInt i.e would, then, need to

```
virtual String EditInt::PropertyHook() const { return "Upp::EditInt"; }
```

to announce, that it wants own property entrance in the map

also, String(typeid(CLASSNAME).name()) could be used..

then, a #define would do it to announce

```
#define HASPROPERTY \
```

```
virtual String PropertyHook() const { return String(typeid(CLASSNAME).name()); }
```

Overengineering.

Mirek

---

---

Subject: Re: SetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 02 Dec 2010 10:24:47 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

sure, it's not easy to think simple  
so what do you mean with dynamic\_cast? how to use that here?

---

---

Subject: Re: SetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Thu, 02 Dec 2010 12:33:24 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Thu, 02 December 2010 05:24: sure, it's not easy to think simple  
so what do you mean with dynamic\_cast? how to use that here?

```
CTRL_PROPERTIES(EditInt)  
    PROPERTY(int, Min)
```



```

    PROPERTY_SET(bool, "NotNull", NotNull)
    PROPERTY_GET(bool, "NotNull", IsNotNull)
END_CTRL_PROPERTIES

```

```

static void GetSet_uniquebyline(Ctrl& c, Value& v, const String& prop, bool set)
{
    EditInt *c = dynamic_cast<EditInt>(&x);
    if(!c) return;
    if(prop == "Min" && set)
        c->Min(v);
    if(prop == "Min" && !set)
        v = c->GetMin();
    ....
}

```

```

INITBLOCK { RegisterCtrlGetSet(GetSet_uniquebyline); }

```

(Will take some effort to really squeeze to macros, but should be possible to do so).

---



---

Subject: Re: SetProperty() / SetProperty() for Ctrl  
 Posted by [kohait00](#) on Thu, 02 Dec 2010 12:39:13 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

thanks, thats a hint...i'll think about it

EDIT: the property definition is not as much of a problem as to organize the 'jump in to search the property of an arbitrary ctrl'

how would you do the property handling so that the entrance point is not relaying on the type information..?

i mean the following:

```

//needs type information to go forward
//and access the corresponding properties table
template<class C>
void SetProp(C& c, const String& name, const Value& v);

////

//typeless, will search for type table of properties and
//there for the convenient property
void SetProp(Ctrl& c, const String& name, const Value& v);

```

this is basicly the question..

also, it may have get only properties, and set only properties

but i see your point to simplify it and handle everything in one function, to register only one function...

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 02 Dec 2010 22:50:38 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

i finally came up with sth like this, what do you think about it?

```
//for Ctrl
```

```
CTRL_PROPERTIES(Ctrl)
PROPERTY(Ctrl, "data", PropSetData, PropGetData)
PROPERTY(Ctrl, "enable", PropEnable, ProplsEnabled)
PROPERTY(Ctrl, "show", PropShow, ProplsVisible)
PROPERTY(Ctrl, "editable", PropEditable, ProplsEditable)
PROPERTY(Ctrl, "logpos", PropSetLogPos, PropGetLogPos)
PROPERTY(Ctrl, "focus", PropFocus, PropHasFocus)
PROPERTY(Ctrl, "modify", PropModify, ProplsModified)
PROPERTY(Ctrl, "tip", PropSetTip, PropGetTip)
PROPERTY(Ctrl, "wantfocus", PropWantFocus, ProplsWantFocus)
PROPERTY(Ctrl, "initFocus", PropInitFocus, ProplsInitFocus)
PROPERTY(Ctrl, "backpaint", PropBackPaint, ProplsBackPaint)
PROPERTY(Ctrl, "transparent", PropTransparent, ProplsTransparent)
PROPERTY_SET(Ctrl, "refresh", PropRefresh)
END_CTRL_PROPERTIES(Ctrl, RecurseDone)
```

```
//for EditInt
```

```
DEC_CTRL_PROPERTIES(Ctrl)

CTRL_PROPERTIES(EditInt)
PROPERTY(EditInt, "min", PropSetMin, PropGetMin)
PROPERTY(EditInt, "max", PropSetMax, PropGetMax)
END_CTRL_PROPERTIES(EditInt, Ctrl)
```

this already works and is a more slim solution as before. take a look in header.

note how one can specify properties even separated (see Ctrl definition)..

a good option is, that properties could be reset to sth different (overridden), by specifying again:

```
PROPERTY(Ctrl, "data", MyPropSetData, MyPropGetData)
```

i think this is simple enough..what do you think?

## File Attachments

1) [CtrlPropTest.rar](#), downloaded 349 times

---

---

Subject: Re: SetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Fri, 03 Dec 2010 08:31:30 GMT

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

---

BTW: Edit Controls inconsistency  
EditIntNotNullSpin not present  
and  
EditIntSpin not in same manner as EditDoubleSpin

add:  
EditCtrl.h:326

```
class EditIntNotNullSpin : public EditIntSpin
{
public:
    EditIntNotNullSpin(int inc = 1) : EditIntSpin(inc) { NotNull(); }
    EditIntNotNullSpin(int min, int max, int inc = 1) : EditIntSpin(min, max, inc) { NotNull(); }
};
```

change to:  
EditCtrl.h:321

```
EditIntSpin(int inc = 1);
EditIntSpin(int min, int max, int inc = 1);
```

EditField.cpp:1052

```
void EditIntSpin::Init()
{
    sb.inc.WhenAction = sb.inc.WhenRepeat = callback(this, &EditIntSpin::Inc);
    sb.dec.WhenAction = sb.dec.WhenRepeat = callback(this, &EditIntSpin::Dec);
    AddFrame(sb);
}
```

```
EditIntSpin::EditIntSpin(int inc) : inc(inc) { Init(); }  
EditIntSpin::EditIntSpin(int min, int max, int inc) : EditInt(min, max), inc(inc) { Init(); }
```

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Thu, 09 Dec 2010 05:59:08 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Still could be better IMO. Too much code to be written. What about something like:

```
PROPERTY(EditInt, "min", IsNumber(param), MinMax(param, o.GetMax()), GetMin())
```

also

```
REGISTERPROPS(EditInt);
```

is not needed, you can squeeze that to CTRL\_PROPERTIES

And with a little effort, you can move base class parameter to CTRL\_PROPERTIES too and make END\_CTRL\_PROPERTIES parameter-less.

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 09 Dec 2010 06:03:45 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

thanks, i'll try

EDIT: question remains: am i using the dynamic\_cast in the way you thought it?

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [mirek](#) on Thu, 09 Dec 2010 07:15:21 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Thu, 09 December 2010 01:03: thanks, i'll try

EDIT: question remains: am i using the dynamic\_cast in the way you thought it?

Yes.

I would perhaps rather use simple ifs to test for property name instead of map of callbacks.

BTW, after more thinking, maybe:

```
PROPERTY(EditInt, "min", if(IsNumber(param)) MinMax(param, o.GetMax()), GetMin())
```

---

---

Subject: Re: SetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Thu, 09 Dec 2010 07:36:30 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

i'd prefer if's as well, if one could override them externally..  
but it is questionable, if it is usefull to derive/override properties. if yes, than callbacks are the only option so far.  
so a still stayed with the explicit naming of the handlers, so full control in handlers remains.  
everything else is done as you said, this is the result so far.

```
CTRL_PROPERTIES(Ctrl, RecurseDone)
PROPERTY("data", PropSetData, PropGetData)
PROPERTY("enable", PropEnable, ProplsEnabled)
PROPERTY("show", PropShow, ProplsVisible)
PROPERTY("editable", PropEditable, ProplsEditable)
PROPERTY("logpos", PropSetLogPos, PropGetLogPos)
PROPERTY("focus", PropFocus, PropHasFocus)
PROPERTY("modify", PropModify, ProplsModified)
PROPERTY("tip", PropSetTip, PropGetTip)
PROPERTY("wantfocus", PropWantFocus, ProplsWantFocus)
PROPERTY("initFocus", PropInitFocus, ProplsInitFocus)
PROPERTY("backpaint", PropBackPaint, ProplsBackPaint)
PROPERTY("transparent", PropTransparent, ProplsTransparent)
PROPERTY_SET("refresh", PropRefresh)
END_CTRL_PROPERTIES
```

..

```
CTRL_PROPERTIES(EditInt, Ctrl)
PROPERTY("min", PropSetMin, PropGetMin)
PROPERTY("max", PropSetMax, PropGetMax)
END_CTRL_PROPERTIES
```

if c++ had delegates like in c# that'd be the best.

BTW: how to install a MouseHook (which is no \_\_thiscall) to operate on a application local TopWindow? i want to start an edit properties window on arbitrary controls upon a mouse+key combination..

EDIT: solution found, just tell me if there are pitfalls i dont see:

```
bool MyMouseHook(Ctrl *ctrl, bool inframe, int event, Point p,
                 int zdelta, dword keyflags)
```

```

{
  if(event & (Ctrl::MOUSEMOVE | Ctrl::MOUSEENTER | Ctrl::MOUSELEAVE |
Ctrl::CURSORIMAGE)) return false;
  if((keyflags & K_MOUSERIGHT))
  if((keyflags & K_SHIFT_CTRL))
  {
    CallbackArgTarget<int> m;
    MenuBar menu;
    menu.Add("List Properties",m[0]);
    menu.Add("Edit Properties",m[1]);
    menu.Execute();
    if(IsNull(m)) return true;
    switch(m)
    {
      case 0:
        { PropList& p = Single<PropList>(); p.PopUp(Ctrl::GetActiveWindow(), *ctrl); }
        break;
      case 1:
        { PropEdit& p = Single<PropEdit>(); p.PopUp(Ctrl::GetActiveWindow(), *ctrl); }
        break;
    }
    return true;
  }
  return false;
}

```

---

## File Attachments

1) [CtrlPropTest2.rar](#), downloaded 345 times

---



---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
 Posted by [kohait00](#) on Fri, 10 Dec 2010 07:55:41 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

i've got a prolem with the mousehook..

it does not work with, i.e., StaticText, which has IgnoreMouse().  
 how could i grab arbitrary controls for editing? with the mouse hook?

---



---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
 Posted by [dolik.rce](#) on Fri, 10 Dec 2010 09:24:13 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Fri, 10 December 2010 08:55 i've got a problem with the mousehook..

it does not work with, i.e., StaticText, which has IgnoreMouse().

how could i grab arbitrary controls for editing? with the mouse hook?

One workaround could be the method that Sc0rch used in his FormEditor (have a look at it in the bazaar). It uses transparent Ctrl on top of all other Ctrl's, grabs the mouse position and then finds the Ctrl that lies at the given location IIRC. It introduces some additional processing, but works for any Ctrl.

Honza

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Fri, 10 Dec 2010 09:32:42 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

great idea thanks a lot  
i used transparent clickable controls elsewhere, but never used it as a entire coverage to a layout..cool.

```
int FormView::ObjectFromPt(Point p)
{
    if (!IsLayout()) return -1;

    for (int i = GetObjectCount() - 1; i >= 0; --i)
        if (Zoom(Offseted( (*GetObjects())[i].GetRect() )).Contains(p))
            return i;
    return -1;
}
```

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [kohait00](#) on Wed, 26 Jan 2011 12:55:44 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

please have a look at bazaar/CtrlProp:

plans are to have a 'live workspace' where i can switch to manipulating the gui elements, store the layout, load it from elsewhere and ofcourse edit the properties of the listed controls.

the current solution is not very satisfying though.

i think/imagine to release the potential for Value and have each control have a set (list or array) of Value's as exposed interface of its properties. much like properties in C#. that'd be great. this would make really dynamic gui cases possible.

plans are, to have kind of a 'dictionary' of such properties, applicationwide, to be able to setup arithmetical functions based on values from the dictionary, and to feed other properties.

this would introduce a new Void() derive with some callbacks to feed the destination controls. it's much like a scripting environment, with all the controls and variables accessible.

it would also make some reactional channels nessecary..i.e. when having arithmetical functions calculating something depending on some properties and resulting in setting another property.

ideas? on how to do that?

attached is a mini demo..

LiveWorkTest, needs current svn rev 3101..

ps: i am thinking of things like

[http://www.jazzmutant.com/lemur\\_overview.php](http://www.jazzmutant.com/lemur_overview.php)

[http://www.jazzmutant.com/download\\_lemur\\_soft.php](http://www.jazzmutant.com/download_lemur_soft.php)

check out the JazzEditor, where one can setup values and have the OSC messages be generated based on other values/properties in the system..

---

## File Attachments

1) [LiveWork.rar](#), downloaded 364 times

---

---

Subject: properties like in C#

Posted by [kohait00](#) on Tue, 01 Feb 2011 13:45:29 GMT

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

---

see

<http://www.codeproject.com/KB/cpp/cppproperties.aspx>

which shows an implementation idea..

thanks to Callback we already can have stuff like that:

```
template<class T>
struct Property
{
    typedef Callback1<const T&> S;
    typedef Callback1<T&> G;
```

```
    Property(const S& s, const G& g)
```



```

: set(s), get(g) {}

const T& Set(const T& a) { ASSERT(set); set(a); return a; }
T Get() const { ASSERT(get); T t; get(t); return t; }

inline T operator= (const T& a) { return Set(a); }
inline operator T() const { return Get(); }

public:
    const S set;
    const G get;
};

typedef Property<Value> PropertyValue;

...

//.h
class PropertyTest : public WithPropertyTestLayout<TopWindow> {
public:
    typedef PropertyTest CLASSNAME;
    PropertyTest();

    void GetD(Value& a) { a = "abc"; }
    void SetD(const Value& a) { RLOG(a); }

    PropertyValue vp;
};

//.cpp
PropertyTest::PropertyTest()

#pragma warning(push)
#pragma warning(disable:4355)
: vp( THISBACK(SetD), THISBACK(GetD) )
#pragma warning(pop)

{
    CtrlLayout(*this, "Window title");

    //setting, will call SetD
    vp = 123;

    //getting, will call GetD
    Value v = vp;
    RLOG(v);
}

```

```
GUI_APP_MAIN
{
    PropertyTest().Run();
}
```

now what's the benefit of it? imagine theCtrls beeing able to be parametrized like

```
EditInt ei;
```

```
ei.min = 100;
ei.max = 200;
ei.data = 150;
ei.show = true;
ei.rect = Rect(0,0, 100,200);
...
```

while i know it's not much better than current design rule of daisy chaining methods, it makes code even bit more clean

---

Subject: Re: properties like in C#  
Posted by [dolik.rce](#) on Tue, 01 Feb 2011 15:13:31 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

kohait00 wrote on Tue, 01 February 2011 14:45now what's the benefit of it? imagine theCtrls beeing able to be parametrized like

```
EditInt ei;
```

```
ei.min = 100;
ei.max = 200;
ei.data = 150;
ei.show = true;
ei.rect = Rect(0,0, 100,200);
...
```

while i know it's not much better than current design rule of daisy chaining methods, it makes code even bit more clean

Hi kohait

Actually it is not better in any way... The underlaying code is more complex and the efficiency of chaining is better (both performance-wise and typing-wise). And the readability is IMHO exactly the same:

```
EditInt ei;
```

```
ei.SetMin(100)
  .SetMax(200)
  .SetData(150)
  .Show()
  .SetRect(Rect(0,0, 100,200));
```

Those are the basic reasons why all the previous proposals to add properties were vetted... You might find some more details by searching the forum.

Best regards,  
Honza

PS: Plus having both properties and SetX()/GetX() functions leads to a unnecessary duplicity.

---

---

Subject: Re: properties like in C#  
Posted by [kohait00](#) on Tue, 01 Feb 2011 17:51:37 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

i'm aware of the lesser usefullness

my problem is, actually, i'm still in search of exposing a uniform way of changing a set properties through a uniform interface.

basicly need it for a live workspace, where controls are placed, resized, edited etc..

any other idea?

sth like the following is planned...

---

#### File Attachments

1) [lemur.JPG](#), downloaded 876 times

---

---

Subject: Re: properties like in C#  
Posted by [koldo](#) on Tue, 01 Feb 2011 21:34:58 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Cute!

---

---

Subject: Re: properties like in C#  
Posted by [kohait00](#) on Wed, 02 Feb 2011 08:30:18 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

dont be too excited.. this is \*not\* my project but i am planning to have sth like that

---

---

Subject: Re: GetProperty() / SetProperty() for Ctrl  
Posted by [luluxiu](#) on Fri, 01 Jul 2011 02:18:38 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

It uses the top of all other Ctrl's transparent Ctrl key, to seize the position of the mouse, and then find the Ctrl, IIRC in a given location where the. It introduces some additional processing, but for any Ctrl key works.

#### File Attachments

1) [cool.gif](#), downloaded 396 times

---