
Subject: Ctrl responds to Language-Setting event?
Posted by [Lance](#) on Mon, 26 Mar 2012 23:35:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

If I change language, the main menu reflects it immediately. Not other Ctrl-derivatives. OnlyCtrls created afterwards will reflect the change in language setting. Is it a way to makeCtrls responds to change in language setting? For example, it is possible to defer language-translation to Paint() so that a call to Refresh() will do?

Another problem I encountered with changing language setting is with EditDate. Due to date scan format changing, it will not Accept() the date string it previously displayed.

Subject: Re: Ctrl responds to Language-Setting event?
Posted by [dolik.rce](#) on Tue, 27 Mar 2012 05:22:58 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Lance

Lance wrote on Tue, 27 March 2012 01:35

If I change language, the main menu reflects it immediately. Not other Ctrl-derivatives. OnlyCtrls created afterwards will reflect the change in language setting. Is it a way to makeCtrls responds to change in language setting? For example, it is possible to defer language-translation to Paint() so that a call to Refresh() will do?

I was fighting the very same thing few weeks ago and the solution that seemed most elegant to me is this: For each GUI class I created a method Setup() which handled all the language dependent stuff - the layout initialization (CtrlLayout call) and all the code that used t_("") macro to assign translated strings to myCtrls. If you call such function in constructor and also each time after the language changes, the labels will be translated correctly. The biggest problem I had was with GridCtrl, where I had to recreate all the contents because it contained some translated strings as well .

Lance wrote on Tue, 27 March 2012 01:35Another problem I encountered with changing language setting is with EditDate. Due to date scan format changing, it will not Accept() the date string it previously displayed.

I think simple workaround for this could be reading out the value to Date variable and assigning it back again after the change. Not tested, but should work

In past I have also used slightly different approach, where I serialized the entire state of the application and then completely restarted the GUI. But that was way to crude, inefficient and involved a pointer ... and on top of that it can't be done simply for every application.

Best regards,
Honza

Subject: Re: Ctrl responds to Language-Setting event?

Posted by [Lance](#) on Tue, 27 Mar 2012 12:51:32 GMT

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

Thank you for the suggestions, Honza! That's a lot of work though

Subject: Re: Ctrl responds to Language-Setting event?

Posted by [dolik.rce](#) on Tue, 27 Mar 2012 16:43:22 GMT

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

Lance wrote on Tue, 27 March 2012 14:51 Thank you for the suggestions, Honza! That's a lot of work though

It is not that much work Here is an example (from my not yet committed changes to HomeBudget example):

// this is a regular constructor:

```
HomeBudgetCategories::HomeBudgetCategories() {  
    Add(spl.Horz(groups, categories));  
    spl.SetPos(2000);
```

```
    groups.AddIndex(ID);  
    groups.AddColumn(NAME, t_("Category")).Edit(eg);  
    groups.Appending().Removing().Editing().Accepting().Canceling();  
    groups.RejectNullRow();  
    groups.SetToolBar();  
    groups.WhenInsertRow = THISBACK(InsertGroup);  
    groups.WhenUpdateRow = THISBACK(UpdateGroup);  
    groups.WhenRemoveRow = THISBACK(RemoveGroup);  
    groups.WhenChangeRow = THISBACK(ChangeGroup);
```

```
    categories.AddIndex(ID);  
    categories.AddColumn(NAME, t_("Name")).Edit(ec);  
    categories.AddColumn(DEFVALUE, t_("Default  
value")).Edit(defval).SetConvert(Single<ConvDouble>());  
    categories.AddColumn(PM, t_("Plus / Minus")).Edit(dlpm).SetConvert(dlpm).Default(-1);  
    categories.AddColumn(INNEWMONTH, t_("Default for a new  
month")).Edit(yesno).SetConvert(yesno).Default(0);  
    categories.WhenInsertRow = THISBACK(InsertCategory);  
    categories.WhenUpdateRow = THISBACK(UpdateCategory);  
    categories.WhenRemoveRow = THISBACK(RemoveCategory);  
    categories.WhenAcceptedRow = THISBACK(UpdateCategories);  
    categories.Appending().Removing().Editing();  
    categories.RejectNullRow();
```

```

categories.SetToolBar();

category.Resizeable(false).Header(false);
category.AddPlus(THISBACK(NewCategory));
}

// ... and here is a function that (re)sets all the translated strings:
void HomeBudgetCategories::Setup() {
    // groups and categories are GridCtrls
    groups.GetColumn(1).Name(t_("Category"));

    categories.GetColumn(1).Name(t_("Name"));
    categories.GetColumn(2).Name(t_("Default value"));
    categories.GetColumn(3).Name(t_("Plus / Minus"));
    categories.GetColumn(4).Name(t_("Default for a new month"));
    //dlpm and yesno are DropGrid instances
    dlpm.Clear();
    dlpm.Add(-1, t_("Minus")).Add(1, t_("Plus"));
    yesno.Clear();
    yesno.Add(0, t_("No")).Add(1, t_("Yes"));

    LoadGroups();
    UpdateCategories();
    category <=<= callback(expenses, &HomeBudgetExpenses::UpdateValue);
}

```

The Setup() method can be probably called from the constructor in most applications. In this example the HomeBudgetCategory is only one of three tabs more or less interconnected, so the Setup() is called from the main window constructor. Note that most of the code in Setup would be in constructor, so it is not that much more code, just some of it is moved to different place

And here is a little example how the language changing code can look:

```

void HomeBudget::Options()
{
    WithOptionsLayout<TopWindow> dlg;
    CtrlLayoutOK(dlg, t_("Options"));
    Index<int> langs = GetLngSet(); // gets list of known languages
    for(int i = 0; i < langs.GetCount(); i++){
        // we iterate and search for translated strings - "Home budget" is a
        // randomly chosen string that is probable to be translated
        // in all languages (except English of course)
        String str = GetLngString(langs[i], "Home budget");
        if (i == 0 || str != "Home budget")
            // if the string is translated we add the language to the DropList
            dlg.lang.Add(langs[i], GetNativeLangName(langs[i]));
    }
    dlg.lang.SetIndex(max(dlg.lang.Find(GetCurrentLanguage()),0));
}

```

```
dlg.clear <=<= THISBACK(ClearAll);  
// check if the dialog was canceled or OKed  
if(dlg.Execute() != IDOK)  
    return;  
// user clicked OK, lets change the language and reset all the strings  
SetLanguage(~dlg.lang);  
Setup();  
}
```

The way I search for translations is not really great, but I didn't find anything that would work better without modifying U++ sources, so for now it has to work with this hack

Best regards,
Honza

Subject: Re: Ctrl responds to Language-Setting event?

Posted by [Lance](#) on Wed, 28 Mar 2012 03:38:04 GMT

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

Very interesting. Thank you, Honza!
