Subject: Dynamic skin changes...
Posted by mirek on Wed, 13 Nov 2024 22:53:57 GMT
View Forum Message <> Reply to Message

I am now working on the option of changing the skin of running application, as followup to the discussion here:

https://www.ultimatepp.org/forums/index.php?t=msg&th=123 29&start=20&

It is developed in the "skin" branch and so far I was able to "convert" examples/UWord and it IMO it looks good. Please test. Details will follow...

Mirek

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Thu, 14 Nov 2024 15:32:03 GMT
View Forum Message <> Reply to Message

Hi Mirek!

Great stuff! I just briefly tested this on a couple of my main apps and it worked right out of the box. :)

Simply added "Ctrl::SkinChangeSensitive();" to the beginning of my main function.

Is it also possible to have some function to call to select between:

- Dynamically changing theme (just this 'follow system theme' that we now get with this)
- Force Light theme regardless of system theme
- Force Dark theme regardless of system theme

and have it realized on the fly?

I guess having this option in GUI apps settings would really please some customers...

Thanks and best regards,

Tom

EDIT: BTW: Can you pump up the UPP_VERSION for this branch so that I can access this new feature only when available?

```
#if (UPP_VERSION>0x20240900)
 Ctrl::SkinChangeSensitive();
#endif
```

Subject: Re: Dynamic skin changes...
Posted by mirek on Fri, 15 Nov 2024 08:06:23 GMT
View Forum Message <> Reply to Message

TheIDE should be now sensitive (in skin branch). Needs more testing.

Some preliminary info:

- Problem of derivative colors is resolved with new SColor class, which is principally Color defined by function. Instance has to be static/global.
- Skin virtual method is called on skin changes, the order is from children to parents. So far, I have found that it is necessary to use Skin to reset toolbar so that icons are updated to correct variants. You can also use it to call e.g. ArrayCtrl::EvenRowColor if you want to use custom color that is not SColor.
- As you have guessed, it has to be activated with Ctrl::SkinChangeSensitive().
- To simplify testing, in debug mode Ctrl+Shift+Num[-] flips between dark and light modes.
- And yes, you can use Ctrl::SetSkin to change the theme. Test it with TheIDE Settings....
- One thing where I am reluctant to go is changes of default GUI font - that requires resizing windows and that is too scary for now. So font is fixed on startup

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Sun, 17 Nov 2024 11:36:10 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Looks very nice!

So far only thing I have noticed is that theide file tabs don't get updated when theme is changed.

Best regards,

Tom

EDIT: It would surely be nice to have two separate color palettes of user configurable syntax highlighting colors: One for light and one for dark mode(s). And also have the appropriate palette loaded automatically when theme is changed.

Subject: Re: Dynamic skin changes...
Posted by mirek on Mon, 18 Nov 2024 13:28:20 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Sun, 17 November 2024 12:36Hi Mirek,

So far only thing I have noticed is that theide file tabs don't get updated when theme is changed.

EDIT: It would surely be nice to have two separate color palettes of user configurable syntax highlighting colors: One for light and one for dark mode(s). And also have the appropriate palette loaded automatically when theme is changed.

Pls check.

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Mon, 18 Nov 2024 20:04:08 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Yes, now the tabs look right.

I think TheIDE would need "Ctrl::SkinChangeSensitive();" in main.cpp to properly follow "Host platform" selection automatically when theme changes.

Additionally, it would be nice to have just a minimal reference example for switching themes through a menu with following choices:

"Host platform (dynamic)"
"Host platform (forced dark)"
"Host platform (forced light)"
"Custom theme (static)"

The two middle choices, i.e.:

"Host platform (forced dark)"
"Host platform (forced light)"

are of interest on Windows due to the synthetic nature of dark mode there, but probably irrelevant on other platforms.

Thanks and best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Tue, 19 Nov 2024 06:44:22 GMT
View Forum Message <> Reply to Message

Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Tue, 19 Nov 2024 07:42:54 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Mon, 18 November 2024 22:04Hi Mirek,

Yes, now the tabs look right.

I think TheIDE would need "Ctrl::SkinChangeSensitive();" in main.cpp to properly follow "Host platform" selection automatically when theme changes.

Additionally, it would be nice to have just a minimal reference example for switching themes through a menu with following choices:

"Host platform (dynamic)"
"Host platform (forced dark)"
"Host platform (forced light)"
"Custom theme (static)"

The two middle choices, i.e.:

"Host platform (forced dark)"
"Host platform (forced light)"

are of interest on Windows due to the synthetic nature of dark mode there, but probably irrelevant on other platforms.

Thanks and best regards,

Tom

Mirek,

I mean this is sort of example...

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct MainWindow : public TopWindow{
 MenuBar menu;

 MainWindow() {
  Title("Dynamic Theming Sample").Sizeable();
  AddFrame(menu);
  menu.Set([=](Bar& bar) {
   bar.Sub("Theme",[=](Bar& bar) {
    bar.Add("Host platform (dynamic)",[=] { Ctrl::SkinChangeSensitive(true);
SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced dark)",[=] { Ctrl::SkinChangeSensitive(false);
SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced light)",[=] { Ctrl::SkinChangeSensitive(false);
SetDarkThemeEnabled(false); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Custom theme (static)",[=] { Ctrl::SkinChangeSensitive(false);
Ctrl::SetSkin(ChStdSkin); });
   });
  });
 }

};


GUI_APP_MAIN{
 Ctrl::SkinChangeSensitive(true);

 MainWindow().Run();
}
```

Additionally, this requires the following change:

```
void Ctrl::SkinChangeSensitive(bool set){
 s_skin_change_sensitive = set;
}
```

Best regards,

Tom

EDIT: Fixed sample code...

Subject: Re: Dynamic skin changes...
Posted by mirek on Tue, 19 Nov 2024 09:14:15 GMT

View Forum Message <> Reply to Message

Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.

Subject: Re: Dynamic skin changes...
Posted by mirek on Tue, 19 Nov 2024 09:15:45 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Tue, 19 November 2024 08:42Tom1 wrote on Mon, 18 November 2024 22:04Hi Mirek,

Yes, now the tabs look right.

I think TheIDE would need "Ctrl::SkinChangeSensitive();" in main.cpp to properly follow "Host platform" selection automatically when theme changes.

Additionally, it would be nice to have just a minimal reference example for switching themes through a menu with following choices:

"Host platform (dynamic)"
"Host platform (forced dark)"
"Host platform (forced light)"
"Custom theme (static)"

The two middle choices, i.e.:

"Host platform (forced dark)"
"Host platform (forced light)"

are of interest on Windows due to the synthetic nature of dark mode there, but probably irrelevant on other platforms.

Thanks and best regards,

Tom

Mirek,

I mean this is sort of example...

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct MainWindow : public TopWindow{
 MenuBar menu;

 MainWindow() {
  Title("Dynamic Theming Sample").Sizeable();
  AddFrame(menu);
  menu.Set([=](Bar& bar) {
   bar.Sub("Theme",[=](Bar& bar) {
    bar.Add("Host platform (dynamic)",[=] { Ctrl::SkinChangeSensitive(true);
SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced dark)",[=] { Ctrl::SkinChangeSensitive(false);
SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced light)",[=] { Ctrl::SkinChangeSensitive(false);
SetDarkThemeEnabled(false); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Custom theme (static)",[=] { Ctrl::SkinChangeSensitive(false);
Ctrl::SetSkin(ChStdSkin); });
   });
  });
 }

};


GUI_APP_MAIN{
 Ctrl::SkinChangeSensitive(true);

 MainWindow().Run();
}
```

Additionally, this requires the following change:

```
void Ctrl::SkinChangeSensitive(bool set){
 s_skin_change_sensitive = set;
}
```

Best regards,

Tom

EDIT: Fixed sample code...

SetDarkThemeEnabled is now sort of deprecated, I am not going to change CtrlCore just because of single reference example and I think the reference example should demonstrate much more than this (Skin, SColor). But yep, it is coming.

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Tue, 19 Nov 2024 09:23:13 GMT

View Forum Message <> Reply to Message

mirek wrote on Tue, 19 November 2024 11:15SetDarkThemeEnabled is now sort of deprecated, I am not going to change CtrlCore just because of single reference example and I think the reference example should demonstrate much more than this (Skin, SColor). But yep, it is coming.
Absolutely, but I wish to know how this exact behavior is supposed to be realized in apps. That's just what I found necessary to realize the desired behavior, but I'm sure your deeper

understanding of the internals yields better reference example for the same... and more.

Thanks,

Tom

EDIT: After all, I wish to give my clients a 'Theme' menu with at least three choices: 'System (dynamic)', 'Light (forced)' and 'Dark (forced)'. All based on 'Host platform' on Windows.

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Tue, 19 Nov 2024 13:22:10 GMT

View Forum Message <> Reply to Message

mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 20 Nov 2024 09:29:33 GMT

View Forum Message <> Reply to Message

Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).

Subject: Re: Dynamic skin changes...

Posted by Tom1 on Wed, 20 Nov 2024 11:23:45 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 20 November 2024 11:29Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If

Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).
I mean, there is a user selectable color available at: "Settings/Editor/Paint line at column [96]". Currently this is not changed along with theme, although it is visible in the editing area. This color is not set within "Syntax highlighting" either, so it is sort of a fixed color without theme dependence.

Just to test, change the "Settings/Editor/Paint line at column [96]" color to pure white and see how it disappears with light theme and is shown with high contrast on dark theme. I think it should be some way connected to the theme to keep it viewable with any theme while avoiding too much contrast.

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 20 Nov 2024 11:40:34 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Wed, 20 November 2024 12:23mirek wrote on Wed, 20 November 2024 11:29Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).
I mean, there is a user selectable color available at: "Settings/Editor/Paint line at column [96]". Currently this is not changed along with theme, although it is visible in the editing area. This color is not set within "Syntax highlighting" either, so it is sort of a fixed color without theme dependence.

Just to test, change the "Settings/Editor/Paint line at column [96]" color to pure white and see how it disappears with light theme and is shown with high contrast on dark theme. I think it should be some way connected to the theme to keep it viewable with any theme while avoiding too much contrast.

Best regards,

Tom

If you change it to pure white, you have custom colors. Do you suggest to throw away user choices?

Mirek

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Wed, 20 Nov 2024 11:57:37 GMT

View Forum Message <> Reply to Message

mirek wrote on Wed, 20 November 2024 13:40Tom1 wrote on Wed, 20 November 2024 12:23mirek wrote on Wed, 20 November 2024 11:29Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).
I mean, there is a user selectable color available at: "Settings/Editor/Paint line at column [96]". Currently this is not changed along with theme, although it is visible in the editing area. This color is not set within "Syntax highlighting" either, so it is sort of a fixed color without theme dependence.

Just to test, change the "Settings/Editor/Paint line at column [96]" color to pure white and see how it disappears with light theme and is shown with high contrast on dark theme. I think it should be some way connected to the theme to keep it viewable with any theme while avoiding too much contrast.

Best regards,

Tom

If you change it to pure white, you have custom colors. Do you suggest to throw away user choices?

Mirek

I do not think I first changed it at all, but I just observed its high contrast on one theme and low contrast on the other, when switching between light and dark on Windows. How do I reset it back to default to get it theme sensitive again?

BR, Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 20 Nov 2024 14:37:50 GMT

View Forum Message <> Reply to Message

Tom1 wrote on Wed, 20 November 2024 12:57mirek wrote on Wed, 20 November 2024 13:40Tom1 wrote on Wed, 20 November 2024 12:23mirek wrote on Wed, 20 November 2024 11:29Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).
I mean, there is a user selectable color available at: "Settings/Editor/Paint line at column [96]". Currently this is not changed along with theme, although it is visible in the editing area. This color is not set within "Syntax highlighting" either, so it is sort of a fixed color without theme dependence.

Just to test, change the "Settings/Editor/Paint line at column [96]" color to pure white and see how it disappears with light theme and is shown with high contrast on dark theme. I think it should be some way connected to the theme to keep it viewable with any theme while avoiding too much contrast.

Best regards,

Tom

If you change it to pure white, you have custom colors. Do you suggest to throw away user choices?

Mirek

I do not think I first changed it at all, but I just observed its high contrast on one theme and low contrast on the other, when switching between light and dark on Windows. How do I reset it back to default to get it theme sensitive again?

BR, Tom

There is "Custom colors" option. If active, we do not touch editor colors on switch.

Of course, it is possible you have hit some cornercase. But when I am testing it, it looks fine...

Mirek

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Wed, 20 Nov 2024 15:16:47 GMT
View Forum Message <> Reply to Message

mirek wrote on Wed, 20 November 2024 16:37Tom1 wrote on Wed, 20 November 2024 12:57mirek wrote on Wed, 20 November 2024 13:40Tom1 wrote on Wed, 20 November 2024 12:23mirek wrote on Wed, 20 November 2024 11:29Tom1 wrote on Tue, 19 November 2024 14:22mirek wrote on Tue, 19 November 2024 11:14Tom1 wrote on Tue, 19 November 2024 07:44Hi,

Another finding in theide: The color of the 'cross hairs' (one vertical and two horizontal) that show the caret location in the code editor are not updated when theme changes.

Best regards,

Tom

Thank you, hopefully fixed, please keep testing.
Thanks, it works fine now.

Maybe the "Settings/Editor/Paint line at column [96]" color setting should be automatically converted between light and dark modes. Depending on the user's color selection, the line may draw too much attention after theme change.

Best regards,

Tom

Not sure what you mean. If Custom colors is not active, it is changed according to the theme. If Custom colors is active, you would be changing color selected by user. Did I misundertood the request?

Oh, BTW, dark theme flip key in debug mode is now "Ctrl + Num[*]" (previously selected key was causing problems).
I mean, there is a user selectable color available at: "Settings/Editor/Paint line at column [96]". Currently this is not changed along with theme, although it is visible in the editing area. This color is not set within "Syntax highlighting" either, so it is sort of a fixed color without theme dependence.

Just to test, change the "Settings/Editor/Paint line at column [96]" color to pure white and see how it disappears with light theme and is shown with high contrast on dark theme. I think it should be some way connected to the theme to keep it viewable with any theme while avoiding too much contrast.

Best regards,

Tom

If you change it to pure white, you have custom colors. Do you suggest to throw away user choices?

Mirek

I do not think I first changed it at all, but I just observed its high contrast on one theme and low contrast on the other, when switching between light and dark on Windows. How do I reset it back to default to get it theme sensitive again?

BR, Tom

There is "Custom colors" option. If active, we do not touch editor colors on switch.

Of course, it is possible you have hit some cornercase. But when I am testing it, it looks fine...

Mirek
Hi,

I can't really figure this out: I see, the "Custom colors" option is not checked here. I also note that changing any color in "Syntax highlighting" automatically turns on the "Custom colors" option. However, it's worth noting that "Paint line at column [96]" color selection button is not on "Syntax highlighting" tab, but instead on "Editor" tab. Also, changing "Paint line at column [96]" color does not activate "Custom colors" on "Syntax highlighting" tab. But never mind, this is not a critical issue from my point of view: I can always select transparent color for the line if it really seriously starts bugging me.

I will keep testing... And looking forward to the reference example with Host platform (dynamic) / Light (forced) / Dark (forced) / ++ selections. :)

BR, Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Thu, 21 Nov 2024 19:30:04 GMT
View Forum Message <> Reply to Message

OK, finally figured out that I misunderstood which line we are talking about, should be now fixed... :)

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Fri, 22 Nov 2024 07:40:38 GMT

View Forum Message <> Reply to Message

mirek wrote on Thu, 21 November 2024 21:30OK, finally figured out that I misunderstood which line we are talking about, should be now fixed... :)
Thanks!

Looks good now. :)

BTW: I added "Ctrl::SkinChangeSensitive();" to GUI_APP_MAIN in ide/main.cpp. Works here.

Best regards,

Tom

EDIT: What is the proper way to add "Host platform Dark (forced)" and "Host platform Light (forced)" to the list of available themes, while keeping current "Host platform" dynamic as is?

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Fri, 22 Nov 2024 15:11:44 GMT
View Forum Message <> Reply to Message

Mirek,

In TheIDE, the "Help and Topics (in window)" icon is better with Light theme than with Dark theme. Coloring is about the same, so maybe just enable the Light icon for both themes. (?)

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Sat, 23 Nov 2024 17:10:28 GMT
View Forum Message <> Reply to Message

Hi,

Just updated to latest on skin... and now ASSERT failed on Color.cpp line 124.

BR, Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Mon, 25 Nov 2024 14:43:02 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Just noticed on current 'skin' that some widgets no longer look like 'Host platform' on Windows 11. E.g. options (check box) and drop down lists have changed their appearance...

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 27 Nov 2024 10:28:26 GMT

View Forum Message <> Reply to Message

So I think I am done with writing the code, two new features/changes:

- After a long deliberation, I have exploited the fact that Image is reference counted and changed all iml Images to "logical constants". That means that even a copy of Image that is from iml changes its appearance accordingly to dark/light theme. Maybe a bit tricky, but the practical result is that e.g. toolbars work without resetting them in Skin.

- Added a new logical Color kind - AColor. This one is defined with its light theme color and gets "adjusted" if used in dark theme.

Everything is now demonstrated in "reference/ThemeChangeSensitive" example.

Mirek

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Wed, 27 Nov 2024 11:14:32 GMT

View Forum Message <> Reply to Message

Hi Mirek,

Thanks for the update. :)

I took a quick look at the example (and ide too) and found some issues...

First, the release mode compilation fails with DDUMP in Color.cpp:

```
void SColor::Write(Color c, Color val)
{
 int ii = c.GetRaw() & VBITS;
 if(ii == 3)
  DDUMP(val);
```

Second, as I mentioned recently, the Option (check mark) button and Drop list buttons do not anymore have Windows 11 look by default, but some other.

Third, it seems that the reaction to 'Toggle dark' button does not retain Windows 11 Host platform skins, but instead switches between ChHostSkin, ChStdSkin and ChDarkSkin. How do I switch between "dynamic Windows 11 Host platform", "static Windows 11 Host platform Light theme" and "static Windows 11 Host platform Light theme"?

Best regards,

Tom

EDIT: Further noted that radio buttons and menu check marks also have wrong appearance in Win11. The default style comes from some other theme.

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 27 Nov 2024 11:43:20 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Wed, 27 November 2024 12:14Hi Mirek,

Thanks for the update. :)

I took a quick look at the example (and ide too) and found some issues...

First, the release mode compilation fails with DDUMP in Color.cpp:

```
void SColor::Write(Color c, Color val)
{
 int ii = c.GetRaw() & VBITS;
 if(ii == 3)
  DDUMP(val);
```

Ops, fix pushed.

Quote:
Second, as I mentioned recently, the Option (check mark) button and Drop list buttons do not anymore have Windows 11 look by default, but some other.

Is this different from master?

Quote:
Third, it seems that the reaction to 'Toggle dark' button does not retain Windows 11 Host platform skins, but instead switches between ChHostSkin, ChStdSkin and ChDarkSkin.

It is debugging feature. I cannot (in general) switch Host theme, so I am toggling between the one set by "SetSkin" and one of internal themes. But that does not really matter, it is not meant for user experience, but for developer so that he can switch dark / light quickly to test that everything reacts accordingly.

Quote:
How do I switch between "dynamic Windows 11 Host platform", "static Windows 11 Host platform Light theme" and "static Windows 11 Host platform Light theme"?

There is no such thing as ""static Windows 11 Host platform Light theme" and "static Windows 11 Host platform Dark theme". If you mean "static Windows 11 Host platform theme", then just do not call SkinChangeSensitive.

Well, if you insist, I can add bool parameter to SkinChangeSensite, I just did not consider that very useful. Either your app reacts or not...

Best regards,

Tom

EDIT: Further noted that radio buttons and menu check marks also have wrong appearance in Win11. The default style comes from some other theme.[/quote]

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 27 Nov 2024 11:45:33 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Wed, 27 November 2024 12:14
Second, as I mentioned recently, the Option (check mark) button and Drop list buttons do not anymore have Windows 11 look by default, but some other.

Ah, sorry, now I can see it in Win10 too. Will fix. (Sorry for ignoring this, I thought it is some Win11 related problem)

Mirek

Subject: Re: Dynamic skin changes...
Posted by mirek on Wed, 27 Nov 2024 12:06:11 GMT
View Forum Message <> Reply to Message

- problem with options fixed
- SkinChangeSensitive now has bool b = true parameter. I wonder what good that will do :)

Mirek

Subject: Re: Dynamic skin changes...

Posted by Tom1 on Wed, 27 Nov 2024 13:37:30 GMT

View Forum Message <> Reply to Message

Thanks Mirek!

Looks very good now! :)

As to your question, now we can do this:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct MainWindow : public TopWindow{
 MenuBar menu;

 MainWindow() {
  Title("Dynamic Theming Sample").Sizeable();
  AddFrame(menu);
  menu.Set([=](Bar& bar) {
   bar.Sub("Theme",[=](Bar& bar) {
    bar.Add("Host platform (dynamic)",[=] { Ctrl::SkinChangeSensitive(true); SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced dark)",[=] { Ctrl::SkinChangeSensitive(false); SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced light)",[=] { Ctrl::SkinChangeSensitive(false); SetDarkThemeEnabled(false); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Custom theme (static)",[=] { Ctrl::SkinChangeSensitive(false); Ctrl::SetSkin(ChStdSkin); });
   });
  });
 }

};

GUI_APP_MAIN{
 Ctrl::SkinChangeSensitive(true);
 Ctrl::SetDarkThemeEnabled(true);

 MainWindow().Run();
}
```

This new automatic theme responsiveness added with manual control keeps maritime users happy: Many of them like to have the entire desktop with all apps dark at night and light during daytime. Night at sea is really dark and it requires dimming each and every light on the ship's bridge to its minimum to be able to see much anything ahead. (There's even a utility available for Windows that automatically switches between light and dark themes based on Sun rise and Sun set times.) But as always, some users wish to have it exactly their way with different themes on different applications. So, now everyone (excluding some even more difficult users) can have it their way. :)

Thanks and best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Thu, 28 Nov 2024 09:31:58 GMT
View Forum Message <> Reply to Message

Hi Mirek,

It seems that currently the built-in 'Flat' themes in TheIDE look exactly the same as their non-flat counterparts. They used to look different in some way... maybe corner rounding. (?)

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Thu, 28 Nov 2024 18:31:40 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Thu, 28 November 2024 10:31Hi Mirek,

It seems that currently the built-in 'Flat' themes in TheIDE look exactly the same as their non-flat counterparts. They used to look different in some way... maybe corner rounding. (?)

Best regards,

Tom

All seems fine here - I am testing with theide...

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Thu, 28 Nov 2024 19:48:43 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 28 November 2024 20:31Tom1 wrote on Thu, 28 November 2024 10:31Hi Mirek,

It seems that currently the built-in 'Flat' themes in TheIDE look exactly the same as their non-flat counterparts. They used to look different in some way... maybe corner rounding. (?)

Best regards,

Tom

All seems fine here - I am testing with theide...
Hi,

You're right. It works here at home. I noticed the issue at the office, so I must have sources out of sync there.

Sorry for the false alarm!

BTW: I have been playing around with U++ theming the whole day and it has been a very interesting journey. I have yet to figure out how to adjust color and shape of many widgets, but I think I'm getting there gradually. E.g. I have not been able to figure out yet how LabelBox line color is changed... (?)

Also noticed that exposing "void ChMakeSkin(int roundness, Color button_face, Color thumb, int *adj);" allows creating custom skins -- e.g. ChMyCustomSkin() -- at app level without need to change U++ internals. a very nice feature! :)

Thanks and best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Didier on Thu, 28 Nov 2024 21:46:51 GMT
View Forum Message <> Reply to Message

Hi Mirek and Tom,

I recently noticed that Theide sometimes needs "help" to compile properly : erase all cache directory and all build directory.
I never had to do this since until quite recently (i've been using upp since more than 15 years now) ==> all compiles OK but execution isn't at all what I expect ... until cleaning up cache and build dir.

I compile with clang under windows.

Maybe you are in the same situation ?

Subject: Re: Dynamic skin changes...
Posted by Didier on Thu, 28 Nov 2024 21:56:55 GMT
View Forum Message <> Reply to Message

Quote:I have not been able to figure out yet how LabelBox line color is changed...

here are some examples I figured out how to do it:

```
struct sChLook_LabelBox {
 Image img;
};

Value Make_ChLook_LabelBox(const Image& img)
{
 sChLook_LabelBox x;
 x.img = MakeButton(8, img, 1, SBlack());
 return RawToValue(x);
}

Value ChLookFn_LabelBox(Draw& w, const Rect& r, const Value& v, int op, Color ink) {

 if( v.Is<sChLook_LabelBox>() )
 {
  const sChLook_LabelBox& e = v.To<sChLook_LabelBox>();
  ChPaint(w, r, e.img);
  return 1;
 }
 return Null;
}

INITBLOCK {
 ChLookFn(ChLookFn_LabelBox);
}

GUI_APP_MAIN
{
 LabelBox::SetLook(White());
 LabelBox::SetLook(WithHotSpots(MakeButton(8, MyImg::BACKGND1(), 5, Blue()), DPI(8),
DPI(8), 0, 0));
 LabelBox::SetLook( MakeButton(4, MyImg::BACKGND1(), 2, Null) );


 static const Value myLabelBoxLook = Make_ChLook_LabelBox(MyImg::BACKGND2);
 LabelBox::SetLook( myLabelBoxLook );


        .....
}
```

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Fri, 29 Nov 2024 08:00:56 GMT
View Forum Message <> Reply to Message

Didier wrote on Thu, 28 November 2024 23:46Hi Mirek and Tom,

I recently noticed that Theide sometimes needs "help" to compile properly : erase all cache directory and all build directory.
I never had to do this since until quite recently (i've been using upp since more than 15 years now)
==> all compiles OK but execution isn't at all what I expect ... until cleaning up cache and build dir.

I compile with clang under windows.

Maybe you are in the same situation ?
Hi Didier,

Thanks! Actually I was at 17486 while I should have been at 17490... Now it works properly here at the office too. Don't know the root cause of this, but it's fine now.

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Fri, 29 Nov 2024 08:03:09 GMT
View Forum Message <> Reply to Message

Didier wrote on Thu, 28 November 2024 23:56Quote:I have not been able to figure out yet how LabelBox line color is changed...

here are some examples I figured out how to do it:

```
struct sChLook_LabelBox {
 Image img;
};

Value Make_ChLook_LabelBox(const Image& img)
{
 sChLook_LabelBox x;
 x.img = MakeButton(8, img, 1, SBlack());
 return RawToValue(x);
}

Value ChLookFn_LabelBox(Draw& w, const Rect& r, const Value& v, int op, Color ink) {

 if( v.Is<sChLook_LabelBox>() )
 {
  const sChLook_LabelBox& e = v.To<sChLook_LabelBox>();
  ChPaint(w, r, e.img);
  return 1;
 }
 return Null;
}

INITBLOCK {
 ChLookFn(ChLookFn_LabelBox);
}

GUI_APP_MAIN
{
 LabelBox::SetLook(White());
 LabelBox::SetLook(WithHotSpots(MakeButton(8, MyImg::BACKGND1(), 5, Blue()), DPI(8), DPI(8), 0, 0));
 LabelBox::SetLook( MakeButton(4, MyImg::BACKGND1(), 2, Null) );


 static const Value myLabelBoxLook = Make_ChLook_LabelBox(MyImg::BACKGND2);
 LabelBox::SetLook( myLabelBoxLook );

	.....
}
```

Thanks! I will look into this...

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Fri, 29 Nov 2024 10:57:33 GMT
View Forum Message <> Reply to Message

Finished documentation as well. Will merge after 2024 release.

Mirek

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Sat, 30 Nov 2024 12:26:27 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Wed, 27 November 2024 15:37Thanks Mirek!

Looks very good now! :)

As to your question, now we can do this:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct MainWindow : public TopWindow{
 MenuBar menu;

 MainWindow() {
  Title("Dynamic Theming Sample").Sizeable();
  AddFrame(menu);
  menu.Set([=](Bar& bar) {
   bar.Sub("Theme",[=](Bar& bar) {
    bar.Add("Host platform (dynamic)",[=] { Ctrl::SkinChangeSensitive(true); SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced dark)",[=] { Ctrl::SkinChangeSensitive(false); SetDarkThemeEnabled(true); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Host platform (forced light)",[=] { Ctrl::SkinChangeSensitive(false); SetDarkThemeEnabled(false); Ctrl::SetSkin(ChHostSkin); });
    bar.Add("Custom theme (static)",[=] { Ctrl::SkinChangeSensitive(false); Ctrl::SetSkin(ChStdSkin); });
   });
  });
 }

};

GUI_APP_MAIN{
 Ctrl::SkinChangeSensitive(true);
 Ctrl::SetDarkThemeEnabled(true);

 MainWindow().Run();
}
```

This new automatic theme responsiveness added with manual control keeps maritime users happy: Many of them like to have the entire desktop with all apps dark at night and light during daytime. Night at sea is really dark and it requires dimming each and every light on the ship's bridge to its minimum to be able to see much anything ahead. (There's even a utility available for Windows that automatically switches between light and dark themes based on Sun rise and Sun set times.) But as always, some users wish to have it exactly their way with different themes on different applications. So, now everyone (excluding some even more difficult users) can have it their way. :)

Thanks and best regards,

Tom
Mirek,

Can you hold merging this a little while longer? It turned out that I still cannot 'Force Dark' if Windows is currently in 'Light' mode. I will look into this...

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Sat, 30 Nov 2024 12:56:51 GMT

View Forum Message <> Reply to Message

Hi,

As IsDarkThemeEnabled() is only used in Windows and only below, so I have repurposed it here to actually mean 'dark theme forced'.

In 'void ChHostSkin()' in ChWin32.cpp:

```
 //sEmulateDarkTheme = Ctrl::IsDarkThemeEnabled() && IsSystemThemeDark() &&
!IsDark(Color::FromCR(GetSysColor(COLOR_WINDOW)));
 sEmulateDarkTheme = ((Ctrl::IsDarkThemeEnabled()&&!Ctrl::IsSkinChangeSensitive()) ||
(IsSystemThemeDark()&&Ctrl::IsSkinChangeSensitive())) &&
!IsDark(Color::FromCR(GetSysColor(COLOR_WINDOW)));
```

Also 'Ctrl' in Ctrl.cpp:

```
static bool Ctrl::IsSkinChangeSensitive()
{
 return s_skin_change_sensitive;
}
```

And the corresponding header change in CtrlCore.h:

```
static bool IsSkinChangeSensitive();
```

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by mirek on Tue, 03 Dec 2024 19:22:37 GMT

View Forum Message <> Reply to Message

Does not make sense at all.... I guess you want those function to mean something else than I intended...

Anyway, just for you (I guess), I have added Win32 only

void ChHostSkinLight();
void ChHostSkinDark();

Use with Ctrl::SetSkin ... I guess this should solve your problem.

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Wed, 04 Dec 2024 06:48:57 GMT

View Forum Message <> Reply to Message

> mirek wrote on Tue, 03 December 2024 21:22Does not make sense at all.... I guess you want those function to mean something else than I intended...
>
> Anyway, just for you (I guess), I have added Win32 only
>
> void ChHostSkinLight();
> void ChHostSkinDark();
>
> Use with Ctrl::SetSkin ... I guess this should solve your problem.

Hi Mirek!

And thank you very much! This is exactly what I needed. Early Christmas this year :)

Now the dynamic/static theming sample becomes:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

struct MainWindow : public TopWindow{
 MenuBar menu;

 MainWindow() {
  Title("Dynamic Theming Sample").Sizeable();
  AddFrame(menu);
  menu.Set([=](Bar& bar) {
   bar.Sub("Theme",[=](Bar& bar) {
    bar.Add("System",[=] { Ctrl::SetSkin(ChHostSkin); });
#ifdef WIN32
    bar.Add("Light",[=] { Ctrl::SetSkin(ChHostSkinLight); });
    bar.Add("Dark",[=] { Ctrl::SetSkin(ChHostSkinDark); });
#endif
    bar.Add("Custom",[=] { Ctrl::SetSkin(ChStdSkin); });
   });
  });
 }

};

GUI_APP_MAIN{
 Ctrl::SkinChangeSensitive(true);
 Ctrl::SetDarkThemeEnabled(true);

 MainWindow().Run();
}
```

Best regards,

Tom

Subject: Re: Dynamic skin changes...
Posted by Tom1 on Wed, 04 Dec 2024 11:48:58 GMT

View Forum Message <> Reply to Message

Hi,

For those looking for rapid theme switching on Windows, I composed a tray utility to does just that. You may wish to put it in auto start, so it will be always available...

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

#ifndef WIN32
#error Sorry, but this program is only available for Windows at this time.
#endif

struct App : TrayIcon {
    typedef App CLASSNAME;

 bool IsSystemThemeDark(){
  return !GetWinRegInt("AppsUseLightTheme",
"Software\\Microsoft\\Windows\\CurrentVersion\\Themes\\Personalize",
HKEY_CURRENT_USER);
 }

 void SetDarkTheme(){
  SetWinRegInt(0, "SystemUsesLightTheme",
"Software\\Microsoft\\Windows\\CurrentVersion\\Themes\\Personalize",
HKEY_CURRENT_USER);
  SetWinRegInt(0, "AppsUseLightTheme",
"Software\\Microsoft\\Windows\\CurrentVersion\\Themes\\Personalize",
HKEY_CURRENT_USER);
```

```
  SetWinRegInt(0, "AccentColorMenu",
"Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Accent", HKEY_CURRENT_USER);
  Sleep(500);
  SetWinRegInt(0xff1d3f58, "AccentColorMenu",
"Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Accent", HKEY_CURRENT_USER);
 }

 void SetLightTheme(){
  SetWinRegInt(1, "SystemUsesLightTheme",
"Software\\Microsoft\\Windows\\CurrentVersion\\Themes\\Personalize",
HKEY_CURRENT_USER);
  SetWinRegInt(1, "AppsUseLightTheme",
"Software\\Microsoft\\Windows\\CurrentVersion\\Themes\\Personalize",
HKEY_CURRENT_USER);
  SetWinRegInt(0, "AccentColorMenu",
"Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Accent", HKEY_CURRENT_USER);
  Sleep(500);
  SetWinRegInt(0xff1d3f58, "AccentColorMenu",
"Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Accent", HKEY_CURRENT_USER);
 }

 virtual void LeftUp() {
  if(IsSystemThemeDark()) SetLightTheme();
  else SetDarkTheme();
 }

 virtual void Menu(Bar& bar) {
  bool dark = IsSystemThemeDark();
  bar.Add("Light", [=](){ SetLightTheme(); }).Check(!dark);
  bar.Add("Dark", [=](){ SetDarkTheme(); }).Check(dark);
  bar.Separator();
  bar.Add("Exit", [=](){ Break(); });
 }

 App() {
  Icon(Image::Hand());
 }
};

GUI_APP_MAIN
{
 Ctrl::SetDarkThemeEnabled(true);
#if (UPP_VERSION>=0x20241100)
 Ctrl::SkinChangeSensitive(true);
#endif
 App().Run();
}
```

Feel free to include this in U++ or use wherever you need it.

Best regards,

Tom

EDIT: You may also wish to give it a nice tray icon instead of "Image::Hand()" :)

Subject: Re: Dynamic skin changes...
Posted by Lance on Thu, 02 Jan 2025 03:30:46 GMT

View Forum Message <> Reply to Message

Great! It works beautifully with virtually no effort on u++ library users.

IMHO though,

```
 Ctrl::SkinChangeSensitive(true);
 Ctrl::SetDarkThemeEnabled(true);
```

deserves to be default. For example, a native gnome app will do exactly that. With these the default behavior, the dynamic color/theme is truly effortless to u++ library users.

Subject: Re: Dynamic skin changes...
Posted by mirek on Thu, 02 Jan 2025 03:40:15 GMT

View Forum Message <> Reply to Message

Lance wrote on Thu, 02 January 2025 04:30Great! It works beautifully with virtually no effort on u++ library users.

IMHO though,

```
 Ctrl::SkinChangeSensitive(true);
 Ctrl::SetDarkThemeEnabled(true);
```

deserves to be default. For example, a native gnome app will do exactly that. With these the default behavior, the dynamic color/theme is truly effortless to u++ library users.

Dark theme IS default.

I am hesitant to make SkinChangeSensitive default as that really requires quite a bit more coding and testing in each app.

Subject: Re: Dynamic skin changes...
Posted by Lance on Thu, 02 Jan 2025 14:12:35 GMT
View Forum Message <> Reply to Message

I see. So one line is required at the moment. That's half the work I thought was required. I am happy with that. Now everything (including Assist++ documentation) fits together naturally and smooth as silk. Great job, Mirek!

Subject: Re: Dynamic skin changes...
Posted by mirek on Thu, 02 Jan 2025 17:10:55 GMT
View Forum Message <> Reply to Message

Lance wrote on Thu, 02 January 2025 15:12I see. So one line is required at the moment. That's half the work I thought was required. I am happy with that. Now everything (including Assist++ documentation) fits together naturally and smooth as silk. Great job, Mirek!

No, actually, that is the problem. Many many lines are required for it to work properly.... :)

Subject: Re: Dynamic skin changes...
Posted by Lance on Thu, 02 Jan 2025 17:58:56 GMT
View Forum Message <> Reply to Message

I understand. A good library takes a great deal of effort from the library develpers' side in order to make the library users' life easier. The dark theme(color/image) integration into u++ certainly is one of the example.

Subject: Re: Dynamic skin changes...
Posted by mirek on Thu, 02 Jan 2025 19:37:49 GMT
View Forum Message <> Reply to Message

Lance wrote on Thu, 02 January 2025 18:58I understand. A good library takes a great deal of effort from the library develpers' side in order to make the library users' life easier. The dark theme(color/image) integration into u++ certainly is one of the example.

Oh, I did not mean library developer effort. I mean that your code has to be adjusted and tested for skin changes, otherwise it will have wrong colors after the change. Which means a lot of changes and code added to your code (in addition to that one line activation).

Subject: Re: Dynamic skin changes...
Posted by Lance on Fri, 03 Jan 2025 13:58:31 GMT
View Forum Message <> Reply to Message

I see now. I am the lucky one as I don't go deep enough to have lots of Blend(...) and Style with customized Color.