Subject: Re: styling of widgets (animation / look and feel)
Posted by mirek on Fri, 21 Apr 2023 05:56:06 GMT
View Forum Message <> Reply to Message

dodobar wrote on Thu, 20 April 2023 22:36agreed this was to get the ball rolling:

perhaps:

```
struct Style : ChStyle<Style> {
    Value look[4];
    Color monocolor[4], textcolor[4];
    Point pressoffset;
    int   focusmargin;
    int   overpaint;
    Font  font;
    Image ok, cancel, exit;
    bool  transparent;
    bool  focus_use_ok;

    // Animation properties
  enum AnimationFinishBehavior {
        GOTO_FIRST,      // Go to the first frame when the animation is interrupted
        GOTO_LAST,       // Go to the last frame when the animation is interrupted
        COMPLETE,        // Complete the animation normally when interrupted
        COMPLETE_FAST,   // Complete the animation quickly when interrupted
        HOLD,            // Hold the current frame when the animation is interrupted
        REVERSE          // Reverse the animation when interrupted
    };

    AnimationFinishBehavior animationFinishBehavior;
    double animation_duration; // Duration of the animation in seconds
    double (*animation_easing_function)(double); // Easing function for the animation
};
```

and ChStyle perhaps:

```
template <class T>
struct ChStyle {
    // ...
    std::vector<T> animationStates;
    double animationDuration;
    EasingFunction easingFunction;
    // ...
};
```

obviously the drawing functions like ChPaint, ChPaintEdge, and ChPaintBody would need to incorporate the animation properties.

The BaseAnimation class could still handle animation logic and calculate the intermediate values

certainly the case of handling Mouse event (e.g., OnMouseEnter, OnMouseLeave)
would need to have some form of update to the animation system

```
void Button::OnMouseEnter() {
   // notify Update the animation state
   // ...

   // notify Start the animation
   // ...
}

void Button::OnMouseLeave() {
   // notify Update the animation state
   // ...

   // notify Start the animation
   // ...
}
```

That is not a good design - too much for Button code to do.

However, meanwhile I found an elegant solution. Basically the only thing really needed for the style implementing code to allow it to perform animations is a pointer to widget as another ChPaint / ChLookFn parameter. If you have that, animation engine can hook into process easily (with special Values for look, can keep track of status with those, can even post time callbacks to refresh the look).

Moreover, we can make ChPaint a Ctrl method, which solves the problem without changing a single line of code in CtrlLib.

If this all sounds a bit confusing, do not despair. I will try to implement it over weekend or soon, together with reference example.

Mirek