Subject: Cubic-Bézier Easing + AnimateProperty for U++ Animations
Posted by dodobar on Mon, 02 Jun 2025 09:52:34 GMT
View Forum Message <> Reply to Message

Hi everyone,

U++ 2025.1 shipped an awesome Animate().
Here's a tiny, header-only patch that adds cubic-Bézier curves and lets you
tween **any** property--not just rectangles.

WHAT'S INSIDE

de Casteljau cubic-Bézier evaluator

Presets: EaseLinear, EaseInQuad, EaseOutQuad,
EaseInOutQuad, EaseOutBounce ... add your own

- Core/Util.h overload
  Lerp(a, b, t, Easing::Fn) // falls back to old behaviour if Fn == nullptr

- CtrlLib/CtrlUtil.h additions
  Animate().Easing(fn) // drop-in upgrade
  template AnimateProperty<T> // animate pos, size, colour, opacity, angle...

Overhead: four multiplies per frame.
Behaviour: identical to CSS, Qt, WPF, Flutter cubic-Bézier timing.

EXAMPLE SNIPPETS

```
// 1. Ease-in-out slide
Animate(panel).Time(400)
              .Easing(Easing::EaseInOutQuad)
              .Pos(Rect(40,40,200,100))
              .Run();

// 2. Generic fade
AnimateProperty<double>(1.0, 0.0,
   [&](double a){ dlg.SetOpacity(a); }, 250, Easing::EaseLinear);

// 3. Designer curve (CSS "ease-out-expo")
auto expo = Easing::Bezier(0.19, 1.0, 0.22, 1.0);
AnimateProperty<Color>(Red, Blue,
   [&](Color c){ btn.SetInk(c); }, 500, expo);
```

WHY BOTHER?

- Industry-standard curves--instantly familiar to web/UI designers.
- Zero ABI churn--everything is inline.
- Keeps U++ lean yet modern; linear remains the default.
- Future-proof--springs or LUT curves can reuse the same `Easing::Fn` typedef.

GITHUB DISCUSSION / CODE

https://github.com/ultimatepp/ultimatepp/discussions/274

How to try:
1. Drop `Easing.h` into *CtrlLib*.
2. Add the 4-arg `Lerp` to *Core/Util.h*.
3. Extend *CtrlLib/CtrlUtil.h* with `.Easing()` and `AnimateProperty<>`.
4. Re-build TheIDE (or run `umk uppsrc theide CLANG -a`).
   Your widgets will glide.

Feedback--or extra easing presets--welcome!

Cheers,
Curt
