



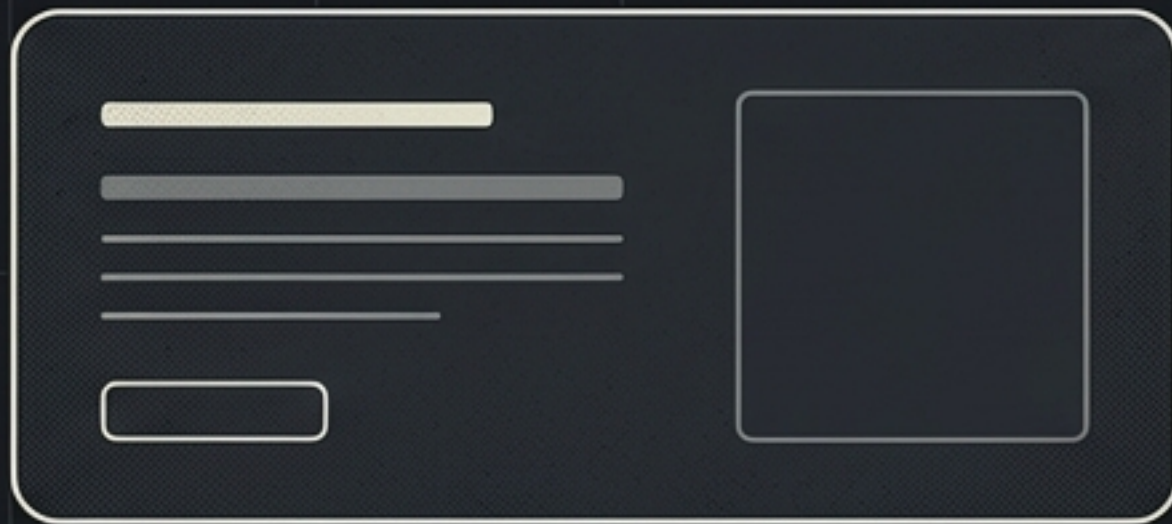
Ion Drive: The Transition Lab

Body Movin', Body Smoothin'

We've learned to pose elements. Now, we make them flow.

Transitions are the bridge between static transforms and full-blown animation. They turn static poses into intentional, readable, and expressive UI movement. This is where the web comes alive.

STATIC



DYNAMIC



Your Training Itinerary

Ion Drive will take us deeper into the core mechanics of motion.
Our mission will cover:



***Transition Basics*:** Understanding the engine.



***Timing Functions*:** Defining the feel of motion.



***Motion Recipes*:** Applying practical maneuvers.



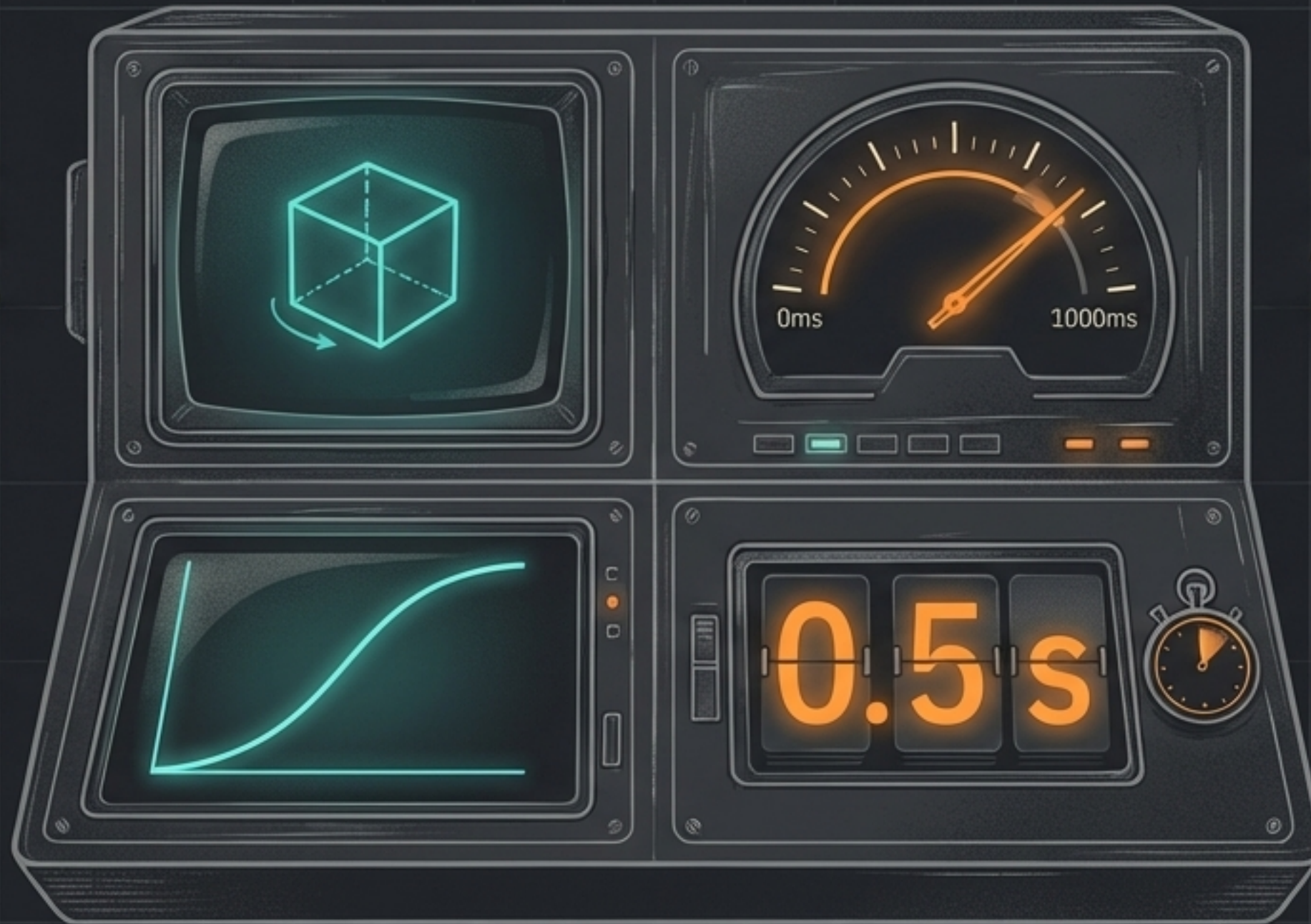
***Transition Sequencing*:** Choreographing complex movements.

The Control Panel: Deconstructing a Transition

These four pieces — property, duration, timing, and delay — are the entire transition engine. Everything else builds from here.

Property

What is allowed to interpolate?



Duration

*How long** the ride takes.

Timing Function

How the motion feels.

Delay

How long to wait before launch.

First Ignition: Hover + Transition

This is the fundamental ‘UI feels alive’ effect. A simple **hover** trigger initiates the transition, creating responsive, engaging feedback.

```
.ps--space-wabbit {  
  transform: translateY(100px) scale(1);  
  transition: transform 500ms ease-out;  
}  
  
.ps--space-wabbit:hover {  
  transform: translateY(-500px) scale(3);  
}
```



Thrust & Trajectory: Mastering Timing Functions

Same distance. Same duration. RADICALLY different vibes. Timing functions tell the browser *how* to move through time — chill, snappy, floaty, or mechanical. It's all about the curves of the timeline.



The Standard Easing Lineup

linear

No acceleration. No deceleration.
Just... math.



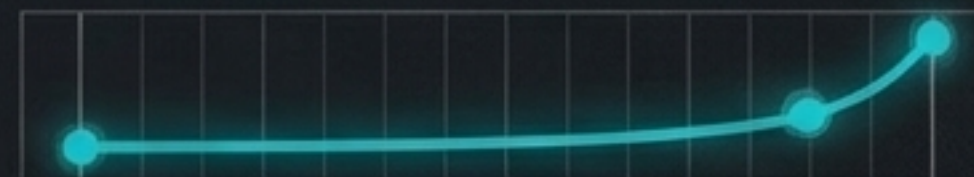
ease

Your default. Smooth start, smooth end.



ease-in

Slow at first, then rockets forward.



ease-out

Fast start, gentle finish.



ease-in-out

Slow → fast → slow. The diplomat.

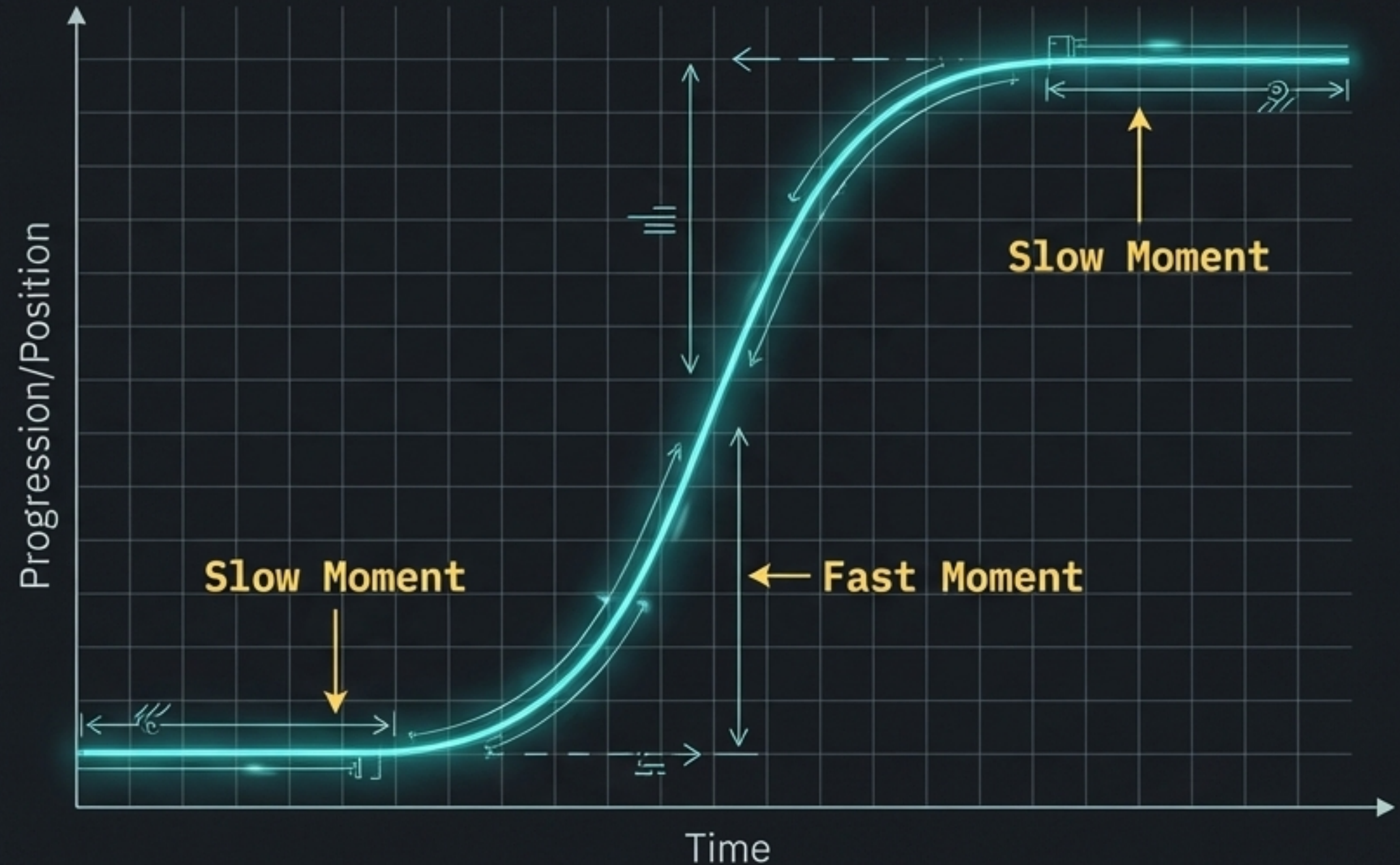


MISSION CRITICAL INTEL: Reading the Curves

A common point of confusion.

Remember: **These curves don't show position — they show time.**

The graph isn't the element's path. It's the *pace* of the motion.



Custom Trajectories: Crafting Curves with `cubic-bezier()`

Want unique personality? This is where we craft it. Every curve tells a story, from a smooth, snappy launch to a dramatic overshoot.

The Code

Technical Slide

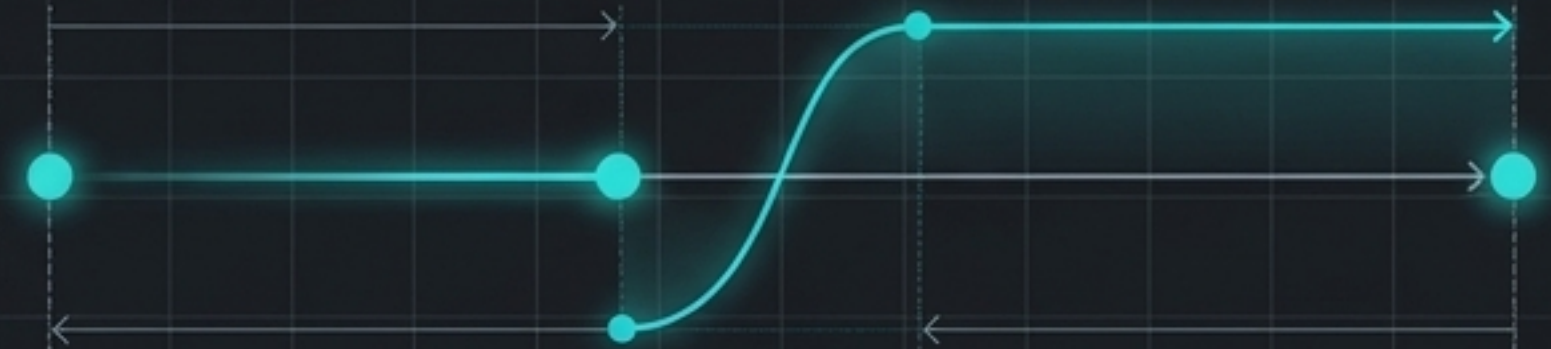
```
transition-timing-function: cubic-bezier(x1, y1, x2, y2);
```

```
/* Snappy */  
cubic-bezier(0.4, 0, 0.2, 1);  
  
/* Dramatic overshoot */  
cubic-bezier(0.7, -0.5, 0.3, 1.4);
```

The Effect

Technical Slide

Snappy



Dramatic overshoot



Quantum Jumps: Slicing Time with `steps()``

Welcome to the choppy, punchy, mechanical side of motion. While easing curves make motion smooth, `steps()` slices time into discrete chunks. The element doesn't glide — it jumps.

Use Cases

- Sprite animations
- Progress ticks
- Retro UI / Glitchy effects

Continuous Time
(`ease-in-out`)

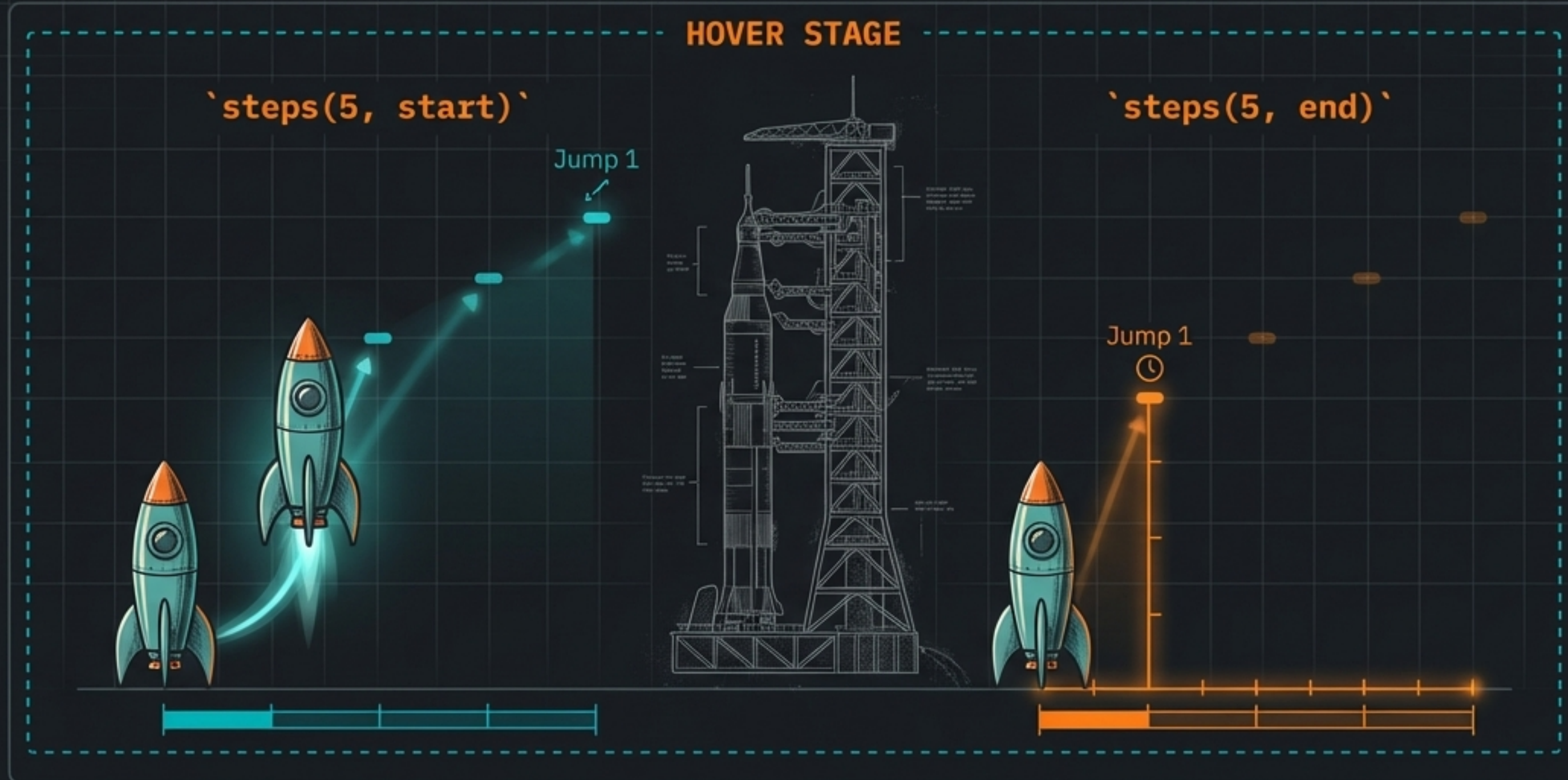


Discrete Time
(`steps(5)`)



Live Demo: `step-start` vs. `step-end`

Two transitions can have the same duration, distance, and property, but feel totally different because time was sliced differently. Hover the stage to see.



Flight Maneuvers: The Motion Recipe Book

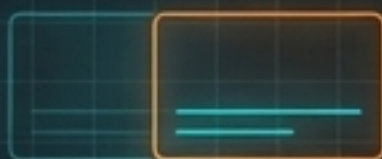
Here's where transforms and transitions finally shake hands. **Transforms** give us poses. **Transitions** give us flow. Each recipe below is a ready-to-use maneuver for your UI.

1) Glide

A soft entry from the side. Great for cards and reveals.

```
.glide-in {  
  opacity: 0;  
  transform: translateX(-20px);  
  transition:  
    opacity 0.4s ease-out,  
    transform 0.4s ease-out;  
}
```

```
.glide-in.active {  
  opacity: 1;  
  transform: translateX(0);  
}
```



2) Bounce

A playful hover hop. Perfect for buttons.

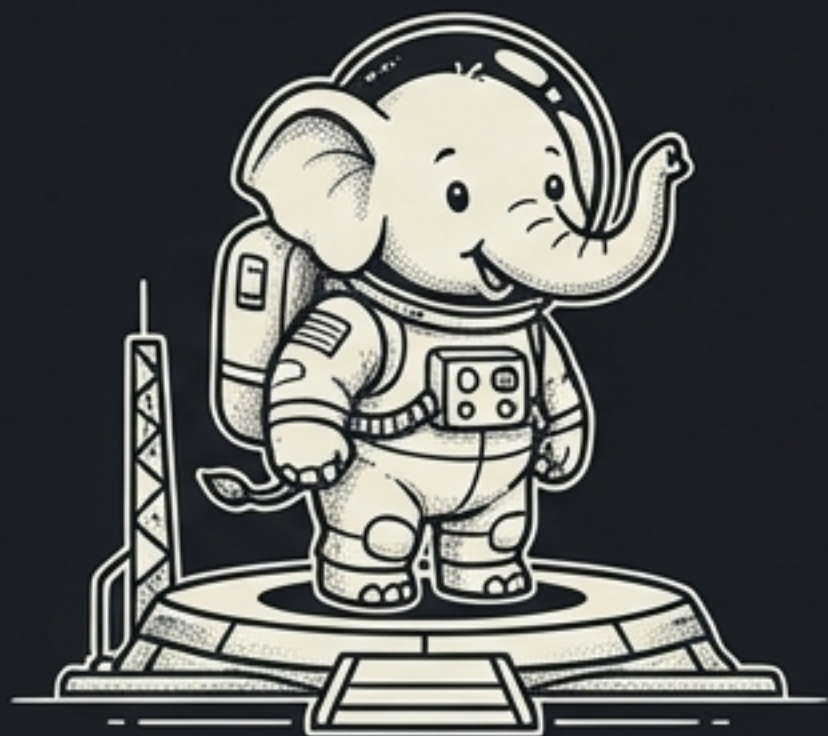
```
.bounce-button {  
  transform: translateY(0);  
  transition: transform 0.3s cubic-bezier(0.25, 1.5, 0.5, 1);  
}  
  
.bounce-button:hover {  
  transform: translateY(-10px) scale(1.05);  
}
```



Fleet Choreography: Sequencing with `transition-delay`

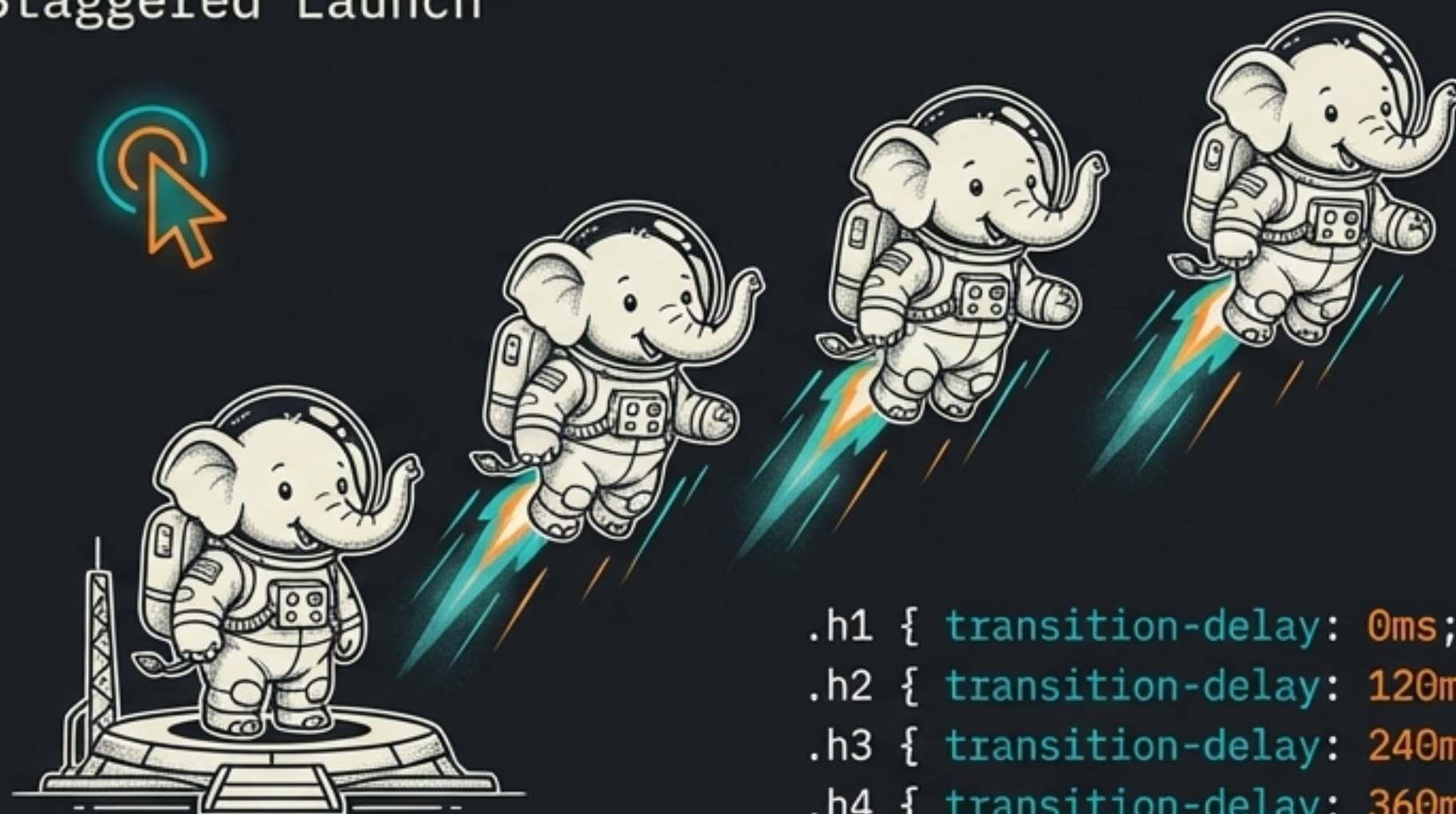
Time to make some pachyderms dance in order. Sequencing is many elements changing over time — intentionally offset. No JS, no keyframes. Just clean, predictable orchestration.

Baseline



```
transition: transform 700ms  
ease-out;
```

Staggered Launch




```
.h1 { transition-delay: 0ms; }  
.h2 { transition-delay: 120ms; }  
.h3 { transition-delay: 240ms; }  
.h4 { transition-delay: 360ms; }
```


Advanced Control: Multi-Property Sequencing

A single element can also have chained transitions. By applying different timings and delays to different properties, you can create sophisticated, layered micro-cascades.



```
/* Fades in *after* moving */  
transition: transform 400ms ease, opacity 250ms ease 180ms;  
↑  
DELAY HIGHLIGHTED
```

Mission Complete. You Have Control.

Transition sequencing gives you a clean mental model for order, a simple tool for staggering, and a powerful way to design motion beats. You now have the fundamentals to make interfaces flow.

Next up → Keyframe Animation!!