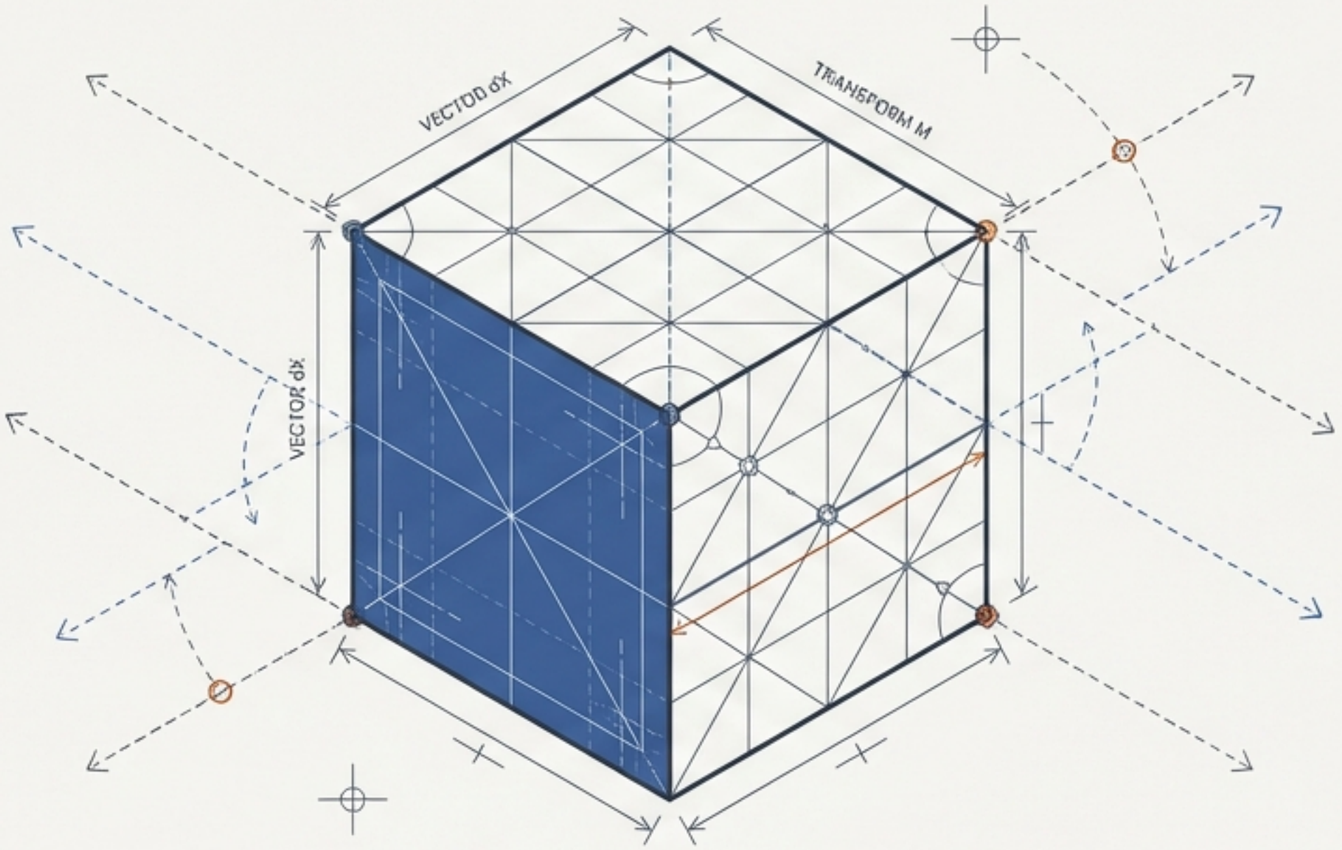


Flutter ile Özel Animasyonlar

Sınırları Aşmak: Matrisler, 3D ve Matematik

Bu sunum, standart widget'ların ötesine geçmek isteyen geliştiriciler için hazırlanmıştır.

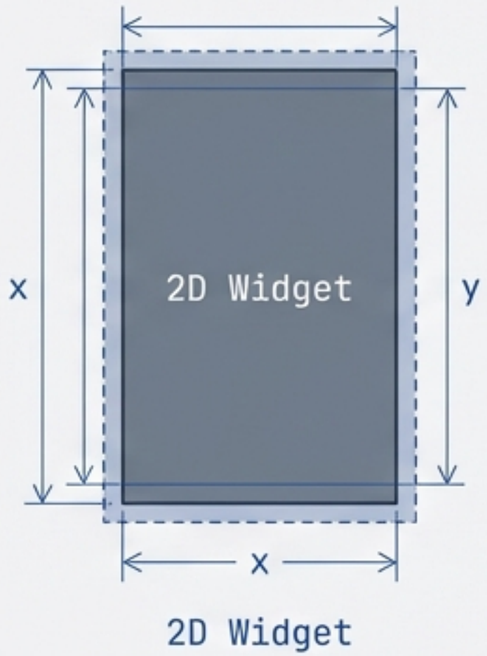
Standartların Ötesine Geçmek

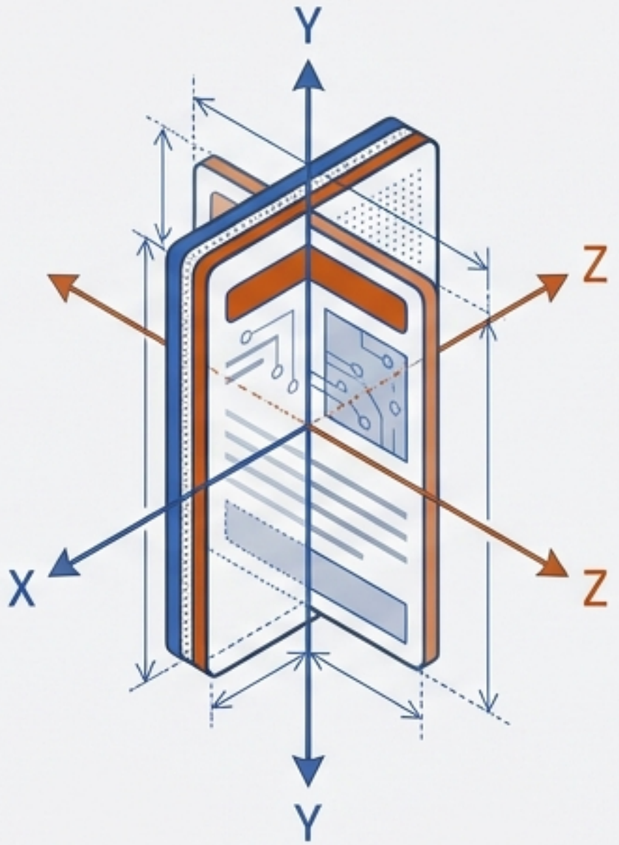
Belirli animasyon türleri (örneğin 3D hareketler) Flutter'ın standart widget'ları tarafından doğrudan desteklenmez.

Bu noktada matematik devreye girer: Matrisler ve Trigonometri.

Amaç: 2 boyutlu ekranımızda 3 boyutlu uzayda çalışmak.

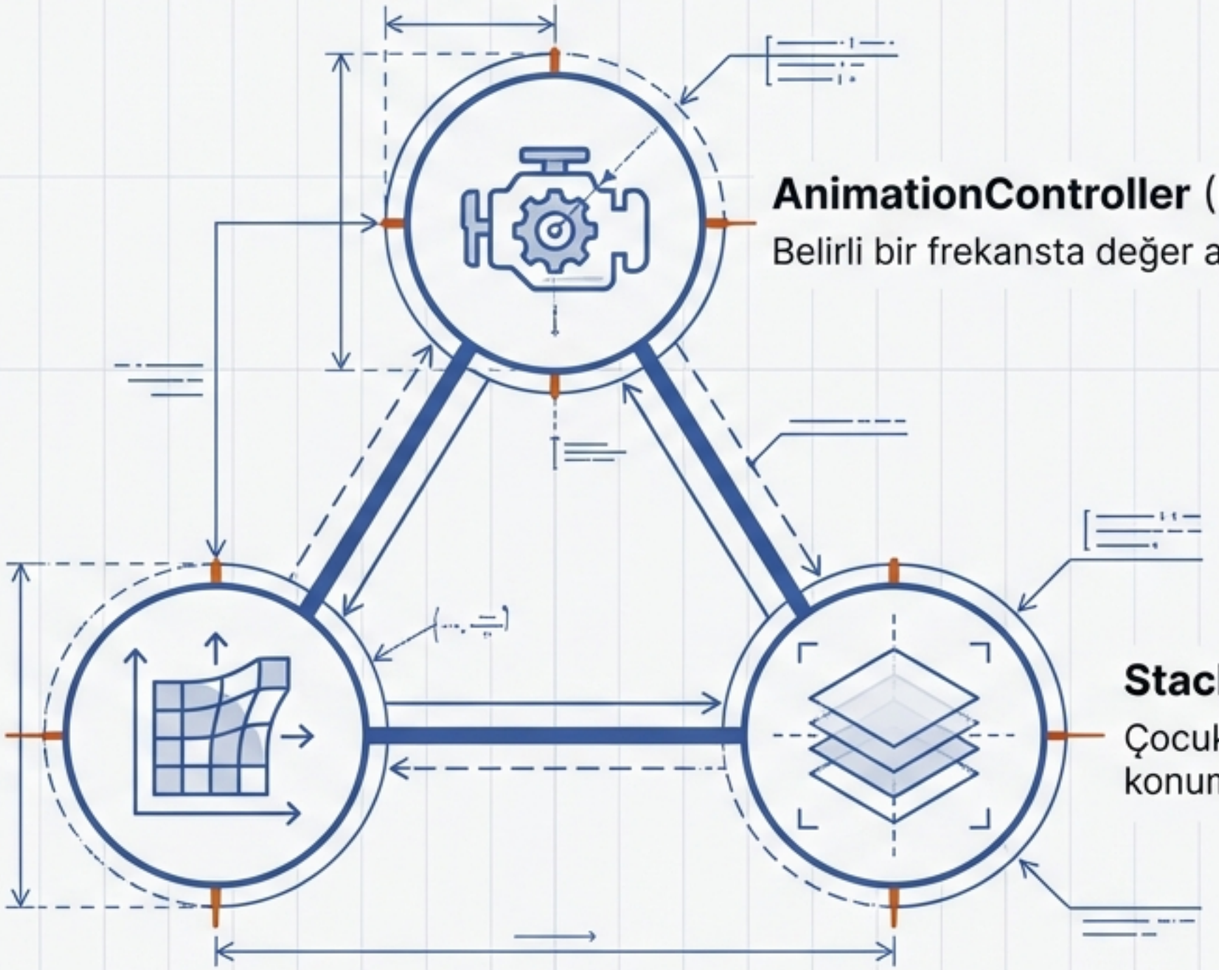
"Burada 3 boyutlu uzayda çalışıyoruz, bu yüzden matematik gerçekten gereklidir."





Çözüm: Sadece 3 Sınıf

Karmaşık animasyonlar oluşturmak, aslında sadece üç temel sınıfın kombinasyonunda ustalaşmaktan geçer:



AnimationController (Motor)

Belirli bir frekansta değer aralığı yayar.

Transform (Şekil Değiştirici)

Matrisleri kullanarak widget'ı ölçekler, döndürür veya eğer.

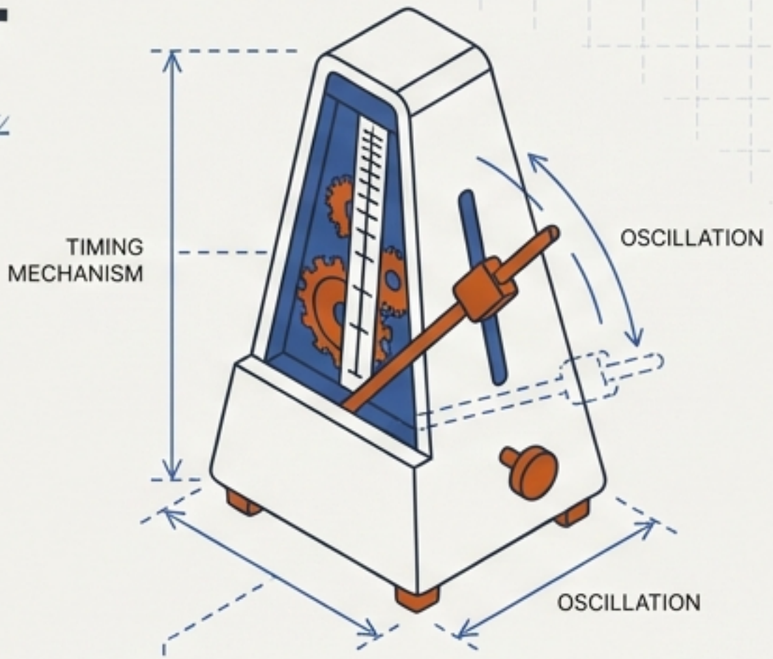
Stack (Tuval)

Çocukları üst üste bindirir ve özel konumlandırmaya izin verir.

1. Motor: AnimationController

Animasyonu yönlendiren değerleri üretir.

- **Gereksinim:** `SingleTickerProviderStateMixin` (vsync için).
- **Yaşam Döngüsü:** `initState` içinde tanımlanır, `dispose` içinde temizlenir.
- **Süre:** Animasyonun ne kadar süreceğini belirler.




```
_controller = AnimationController(
  vsync: this,
  duration: const Duration(seconds: 15),
)..repeat();
```

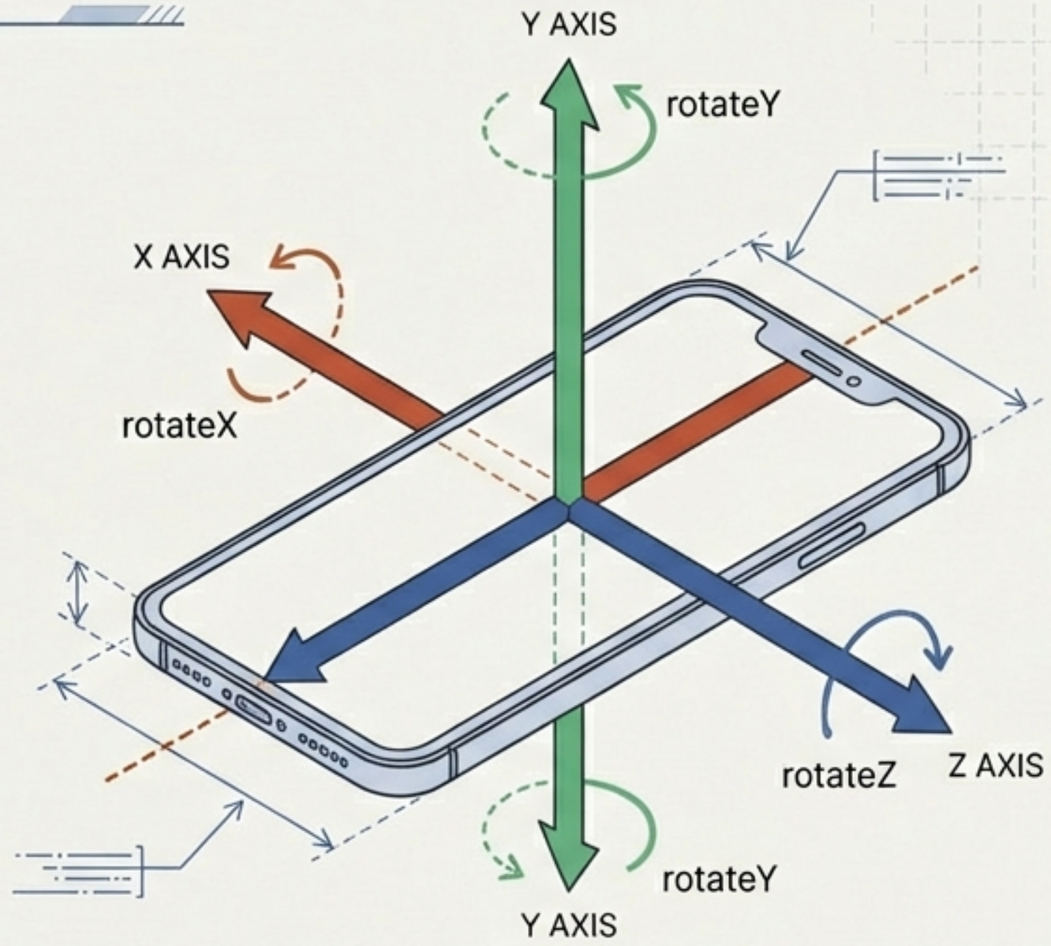
Hız kontrolü

2. Şekil Değiştirici: Transform ve Matrix4

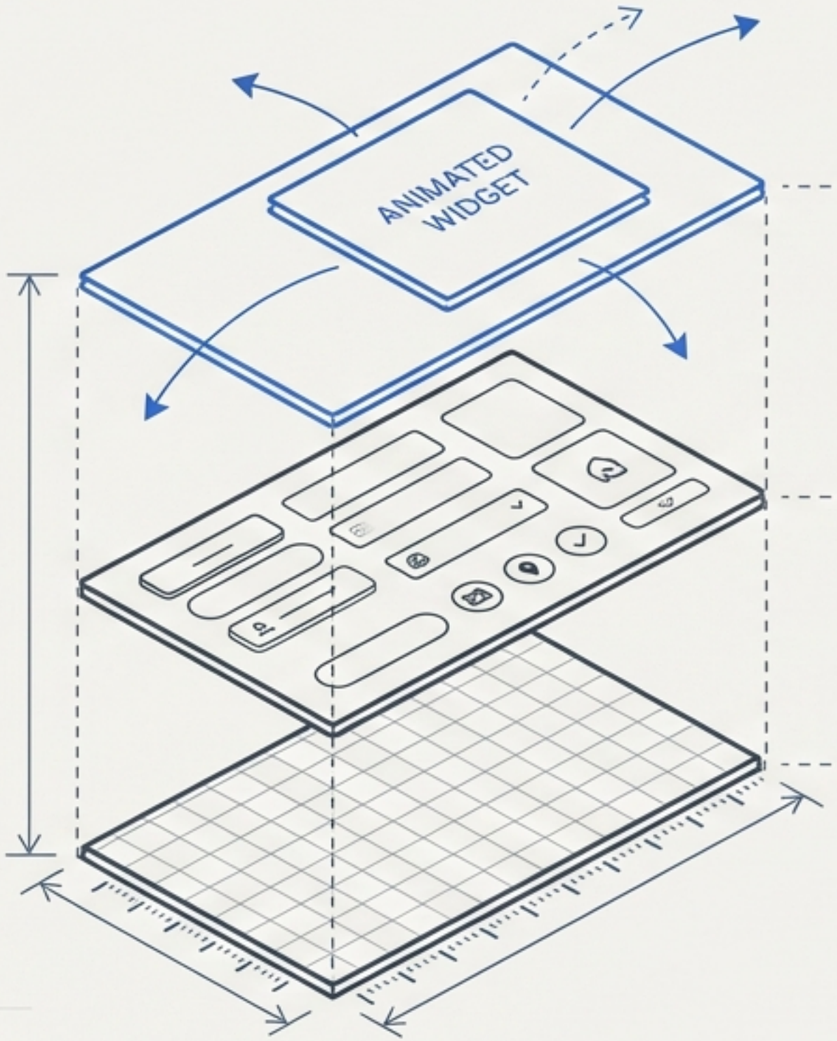
Matrix4: 3D koordinatları temsil etmek ve hesaplamalar yapmak için kullanılan 4x4'lük bir matristir.

Düz bir ekranda derinlik algısı yaratmamızı sağlar.

- **Döndürme Eksenleri:**
  1. `rotateX: Yatay eksende dönüş.
  2. `rotateY: Dikey eksende dönüş (3D efekti için kritik).
  3. `rotateZ: Düzlem üzerinde dönüş.



3. Tuval: Stack



Top Layer:
Floating Card
(Animated)

Middle Layer:
Static UI Elements

Bottom Layer:
Base Grid

Hareket eden widget'ların, diğer widget'ların konumunu bozmasını engeller.

Positioned: Stack içindeki öğelerin koordinatlarını hassas bir şekilde kontrol etmeyi sağlar.

Üst üste binen 'katmanlar' mantığıyla çalışır; widget'lar birbirine müdahale etmeden özgürce hareket edebilir.

Uygulama 1: 3D Skew (Eğme) Efekti

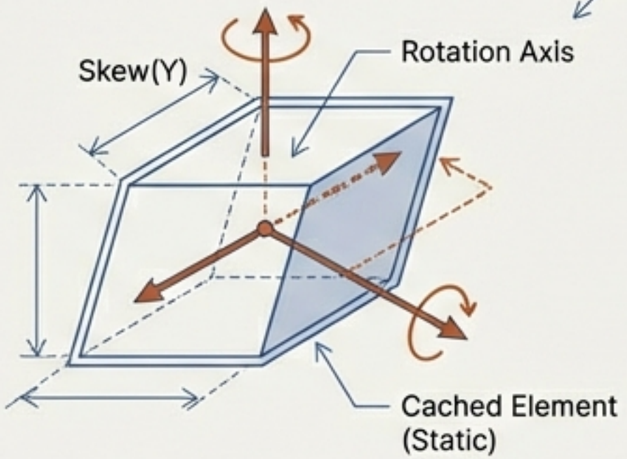
Bir kutunun Y ekseni etrafında dönerken hafifçe eğildiği (skew) bir animasyon kuralım.

Performans İpucu:
AnimatedBuilder kullanın.

Bu yapı, animasyonlu olmayan çocuğu (Container/Text) önbelleğe alır ve sadece Transform'u yeniden oluşturur.

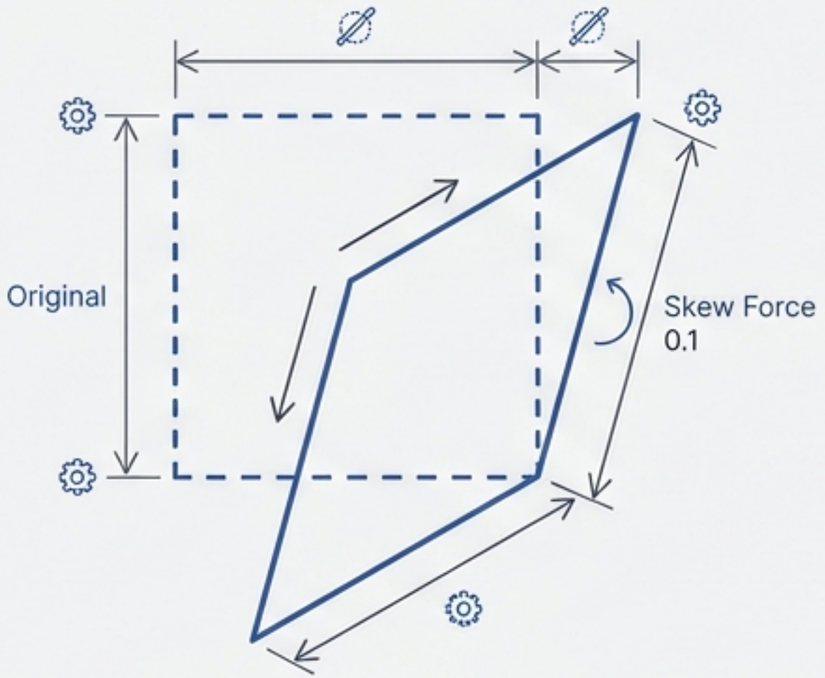
Statik:
Sadece bir kez çizilir

```
AnimatedBuilder(
  animation: _controller,
  builder: (context, child) {
    return Transform(
      // Matrix işlemleri buraya gelecek
      child: child,
    );
  },
  child: Container(...), // Önbelleğe alınan kısım
);
```



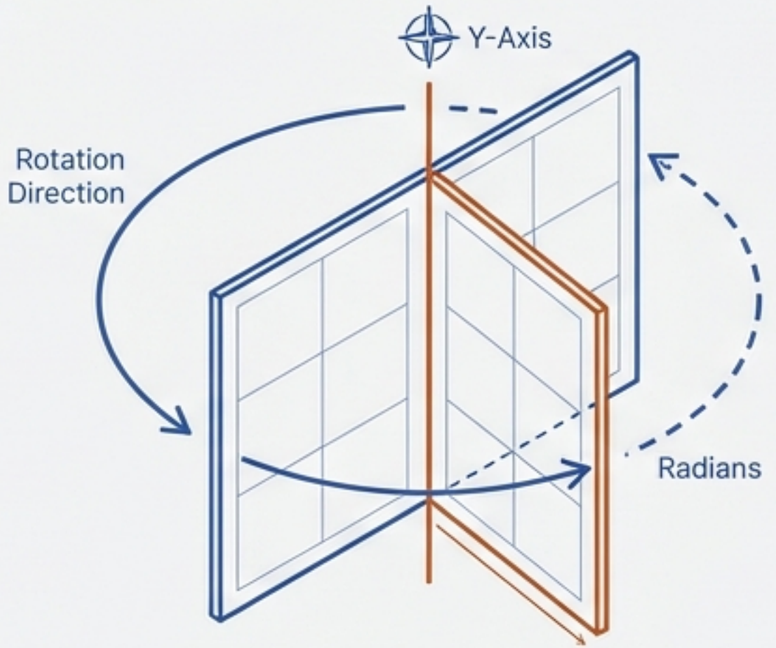
Matematiği Çözmek

`Matrix4.skewY(0.1)`



Kenarları paralel tutarken widget'ı belirli bir yönde eğer.

`rotateY(radians)`



Çocuğu Y ekseni etrafında döndürür.

Formül: `_controller.value * 2 * math.pi` (Tam bir tur)

Akışı Kontrol Etmek

Animasyonun nasıl tekrarlanacağını `AnimationController` üzerinden yönetiriz.

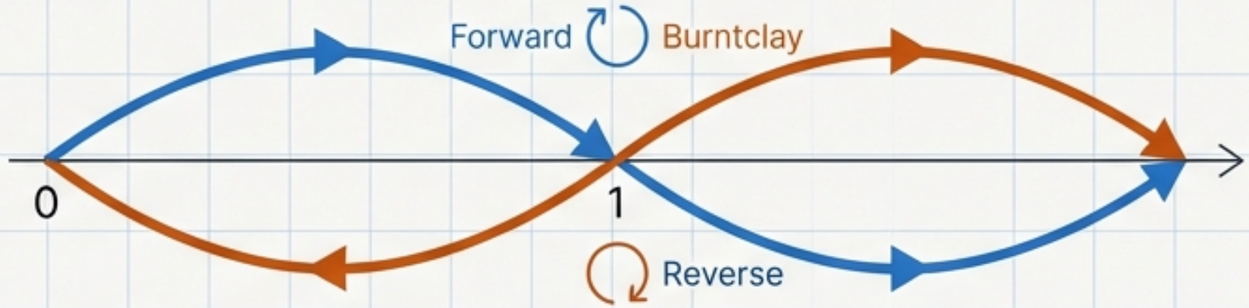
Repeat

```
JetBrains Mono
_controller.repeat();
```

Sürekli İleri: Animasyon tek bir yönde sürekli akar. Dönme efektleri için idealdir.

Reverse

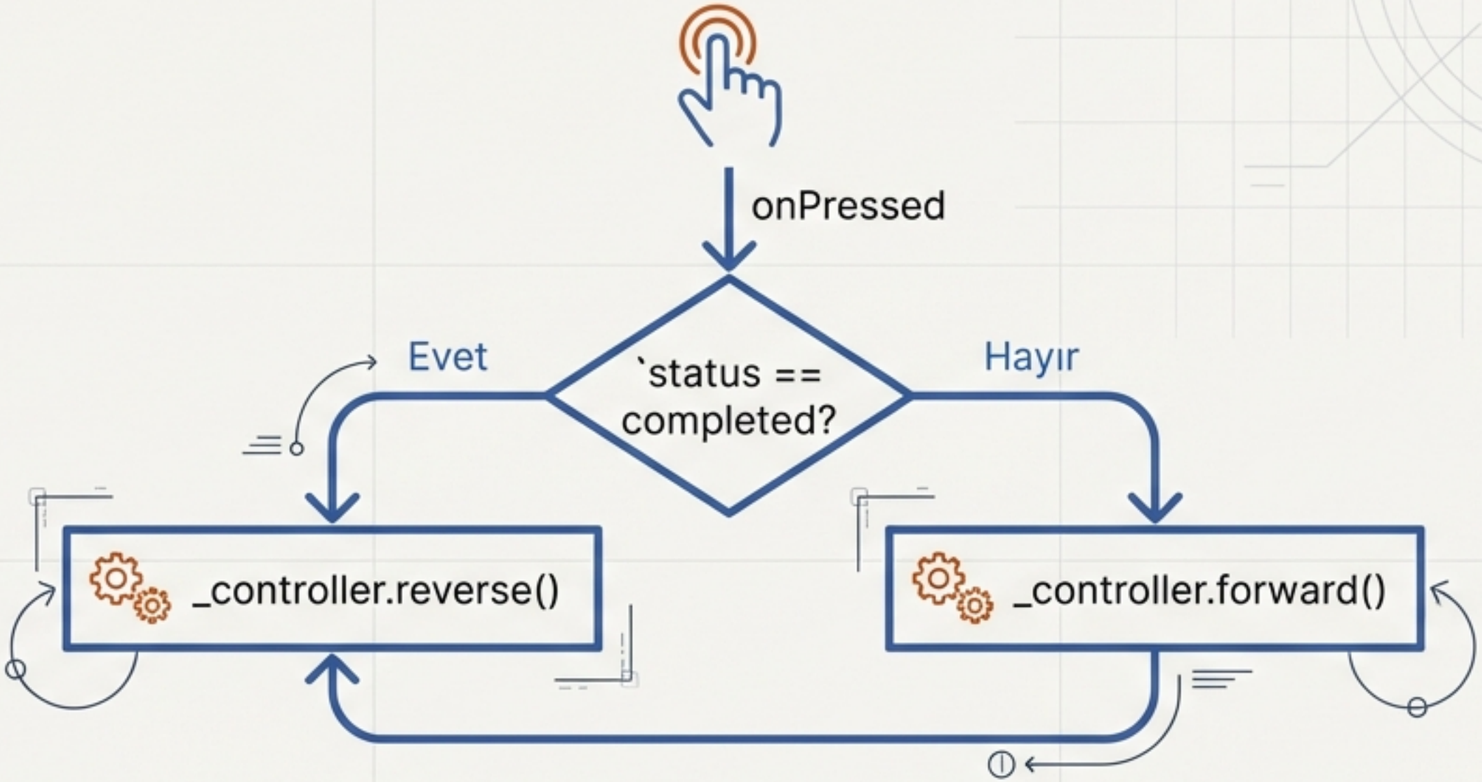
```
JetBrains Mono
_controller.repeat(reverse: true);
```



Git-Gel (Yo-Yo): İleri gider ve sonra geri döner. 'Nefes alma' efektleri için kullanılır.

Uygulama 2: Etkileşimli Hareket

Bir düğmeye tıklandığında küçülüp başka bir konuma kayan bir widget yapalım.



JetBrains Mono

```
if (_controller.status == AnimationStatus.completed)
  _controller.reverse();
else
  _controller.forward();
```

Blueprint Version 1.8

Hareketin Mantığı: Ölçekle ve Taşı

Başlangıç:

```
JetBrains Mono
Matrix4.identity()
```

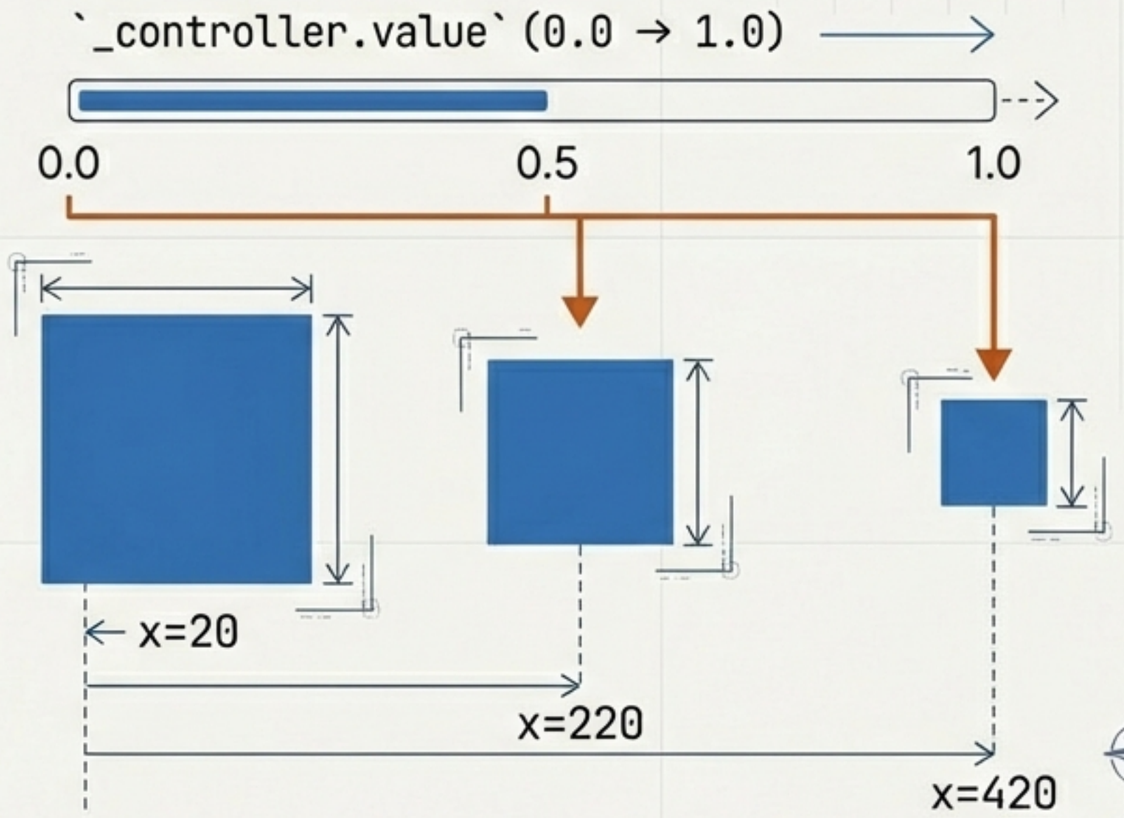
(Değişim öncesi temiz durum).

Ölçekleme (Scale):

- Formül: `1 - (_controller.value * 0.5)`
- Animasyon ilerledikçe widget küçülür.

Pozisyon (Translate):

- Formül: `20 + (_controller.value * 400)`
- Değer değiştikçe widget ekranda kayar.



Blueprint Version 1.9

Güç Sizin Elinizde

İhtiyacınız olan sadece bir **Controller**, bir **Stack** ve **Transform** widget'ı.

Animasyon dünyasında kaybolmaktan korkmayın.
Bu üç sınıfı birleştirerek Flutter'da büyük bir güce sahip olursunuz. Denemeye başlayın!

Yaratıcılığınız tek sınırınızdır.