
Android Material Motion

Murat Öter

Android Developer

Material Design

Material Design is an adaptable system of guidelines, components, and tools that support the best practices of user interface design.

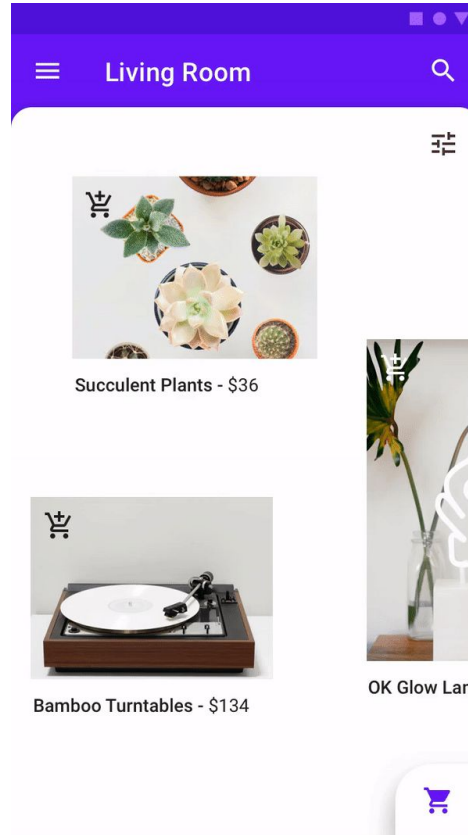
Created by Google to build high-quality digital experience for Android, iOS, Flutter and web.

Material Motion

Animasyon ve hareketler için tasarlanmış bir material design altında geliştirilmiş bir tasarım kütüphanesidir. Bu kütüphane, kullanıcı arayüzlerinin daha doğal, akıcı ve keyifli hale getirilmesi amacıyla animasyon, geçiş ve dokunma etkileşimlerini tasarlamak için kullanılır.

Bunları yaparken transition'ları kullanır. Transition animasyon ile farklı iki state'deki UI elementine animasyon vermesini sağlar.

User Education



Patterns

4 farklı pattern üzerine kuruludur.

→ **Container Transform**

→ **Shared Axis**

→ **Fade Through**

→ **Fade**

Container Transform (Persistent Container)

Listeler, cardlar, butonlar, arama kutuları gibi containeri olan iki UI elementin arasında geçiş animasyonu vermeyi sağlar. En çok kullanılan patterndir. Transition işlemi kalıcı bir element ile yapılıyor ise kullanılmalıdır.

Transition.java class'ından extend edilmiştir. Diğer patternler *Visibility.java* class'ından extend edilmiştir. Bu da MaterialContainerTransform'ı bir shared element transition'ı yapmaktadır.

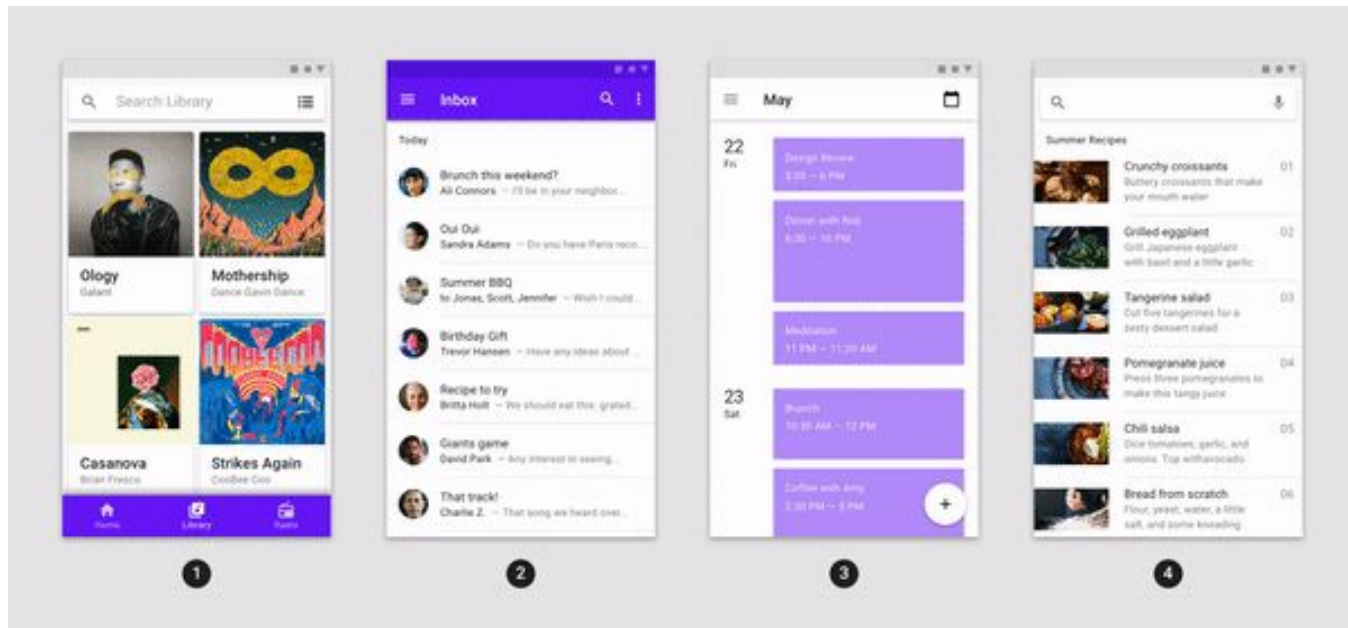


Container Transform vs Shared Element

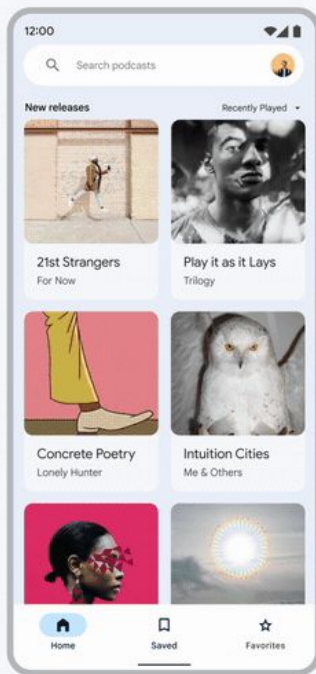
MaterialContainerTransform bir shared element transition'dir. Ancak geleneksel **shared element**te örneğin iki sahne arasındaki geçişte öğenin aynı görüntüsünü vermek için tek bir öğeyi hareket ettirir. Bir liste öğesi seçildiğinde seçilen öğenin resmi diğer sahneye aktarılır ve bu görüntü daha büyük boyutta görüntülenir.

MaterialContainerTransform da ise paylaşılan öğenin boyut ve şeklinin değiştirilerek geçişi gerçekleştirir. Örneğin, bir listenin satır View in konteyneri, tam ekran Fragment'in kök ViewGroup'unun boyut ve şekline dönüştürülür. Öğelerin hareketini sağlama şekli, esneklik ve özelleştirme sağlar.

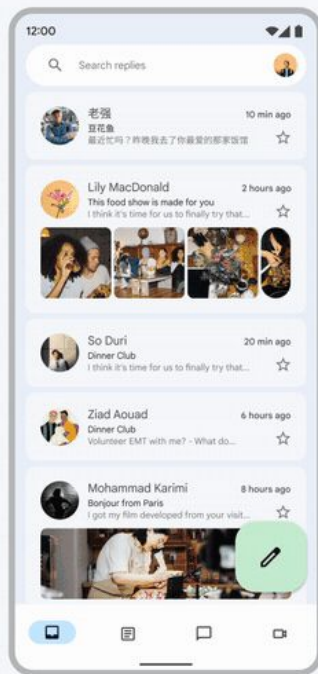
Container Transform



Container Transform



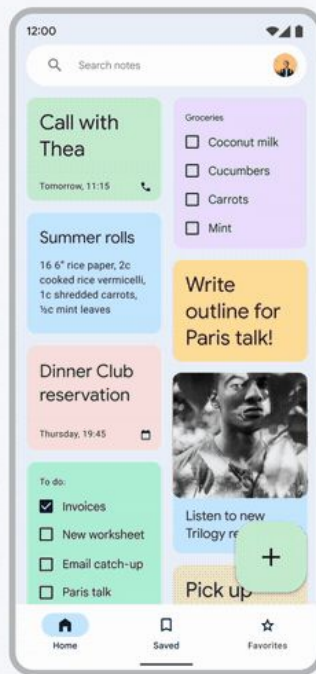
1



2



3



4

Shared Axis (Relationships)

Shared Axis animasyonları, farklı öğeler arasındaki geçişler belirli bir eksen boyunca paylaşılır. Bu, sayfa veya öğeler arasındaki geçişin daha doğal hissetmesini sağlar.

Bir visibility transition'dır. Forward ve backward direction'ları kullanarak hareket eder.

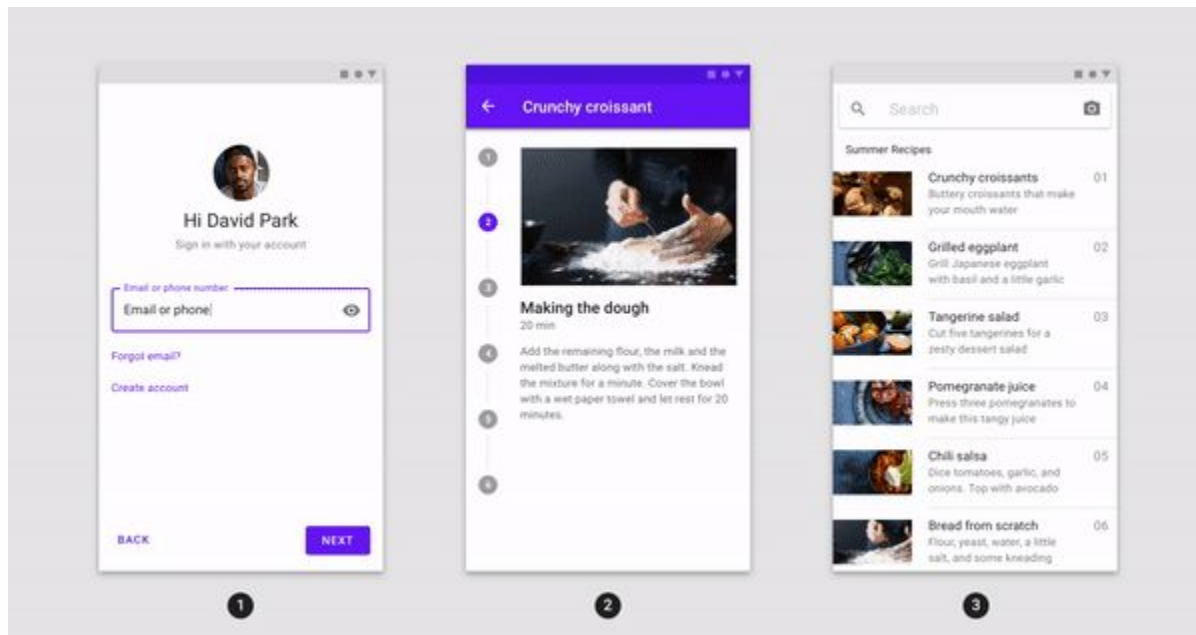
Persistent container yok ve navigational bir ilişki varsa tercih edilmeli.



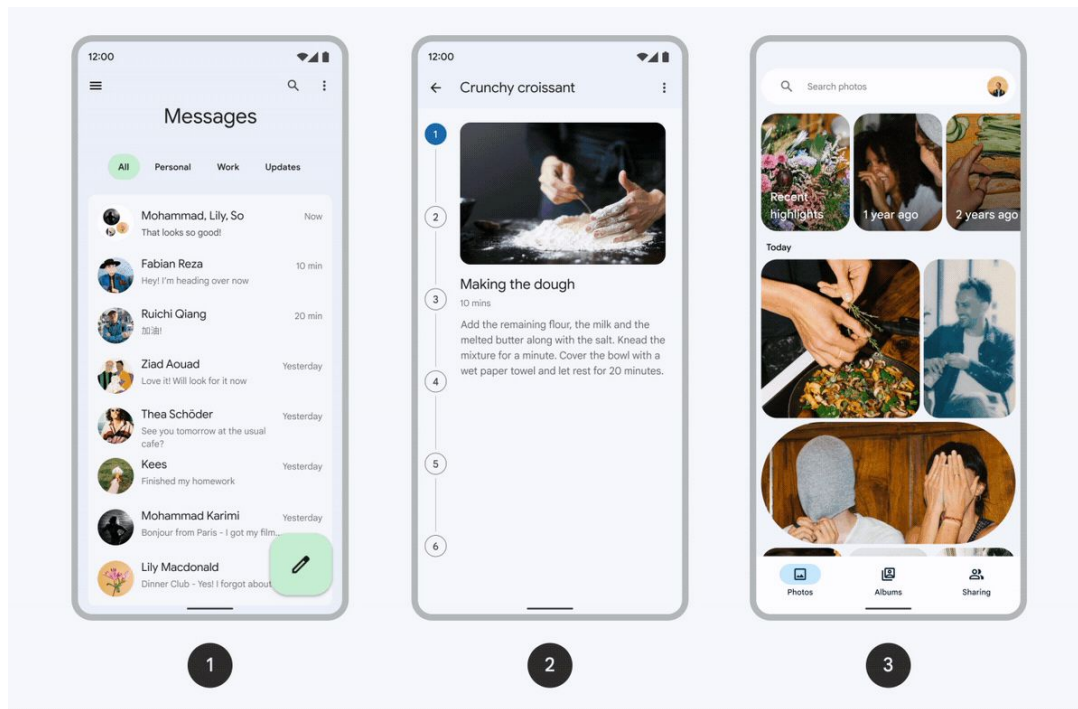
Shared Axis

Axis	Forward	Backward	Kullanımı Alanları
X	Left on x-axis	Right on x-axis	Onboarding Akışları
Y	Up on y-axis	Down on y-axis	İki component ayırma/geçiş sağlama
Z	Forward on z-axis	Backward on z-axis	Parent/Child Navigation, Derinlik

Shared Axis



Shared Axis



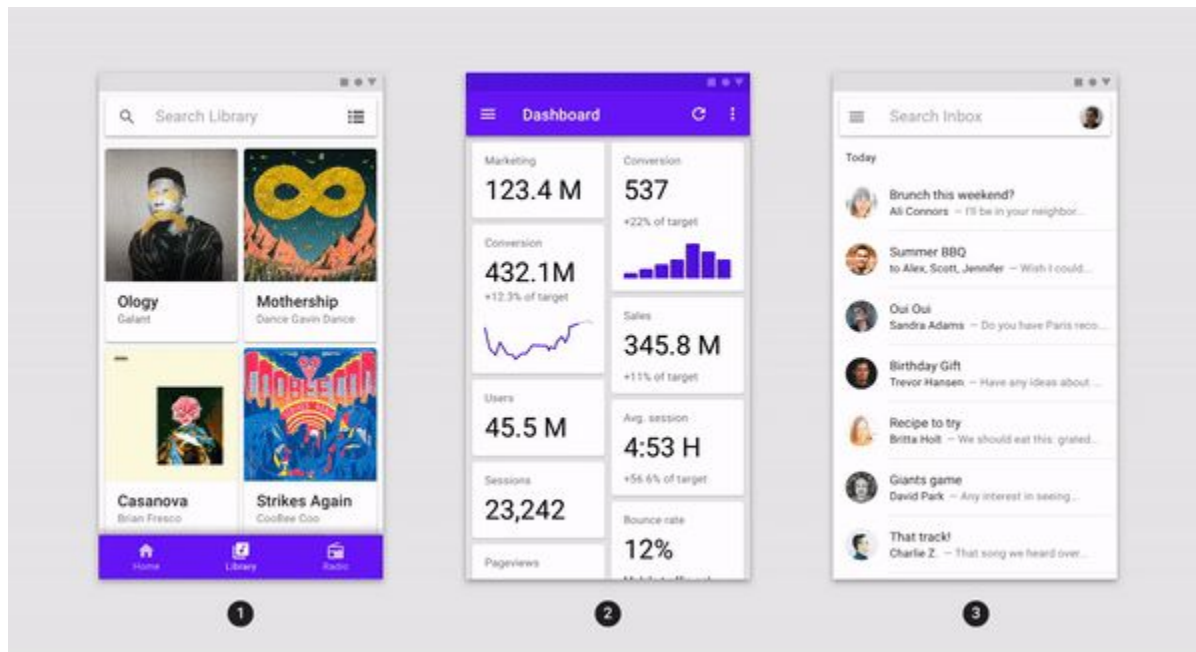
Fade Through (No Relationships or Insignificant)

Bu pattern UI elementlerin birbiriyle güçlü bir bağlantısı olmadığı durumlarda kullanılabilir. Bottom bar navigation buna güzel örnek olabilir. Tab değiştiğinde ekranda gözüken ana context değişecek. Değişimi kullanıcı daha iyi anlayacaktır.

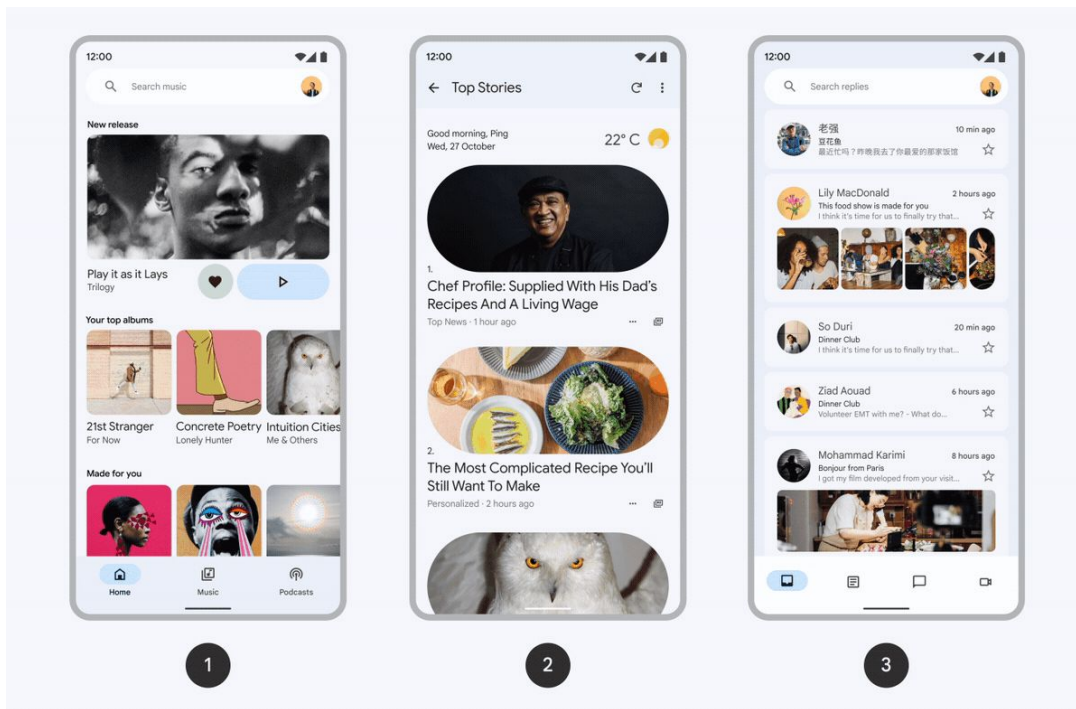
Kaybolacak view'a fade out animasyonu uygulanır, yeni gelen view'a fade in animasyonu verilir.



Fade Through



Fade Through



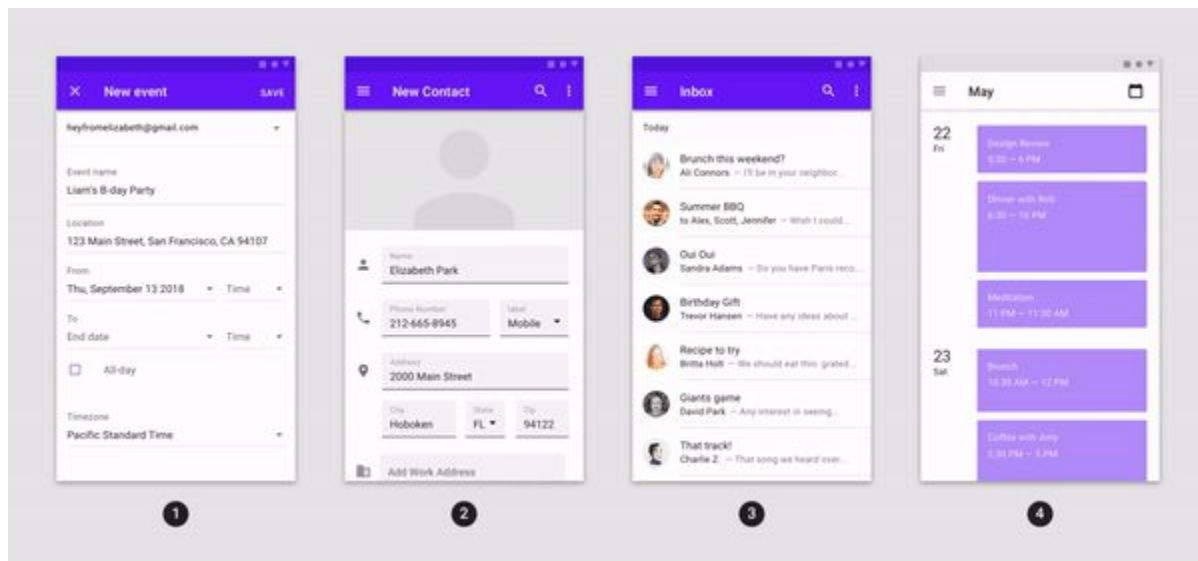
Fade

(Entering and exiting)

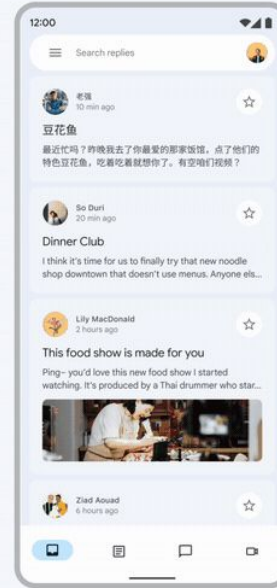
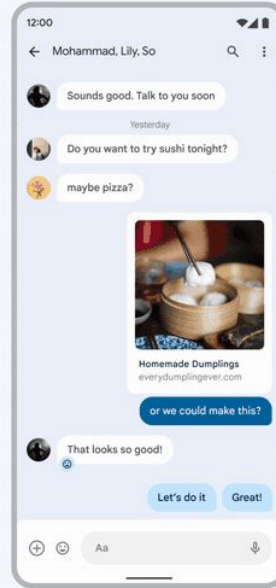
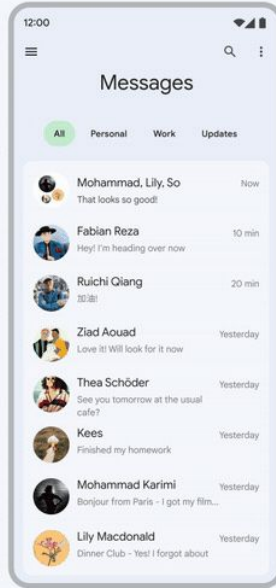
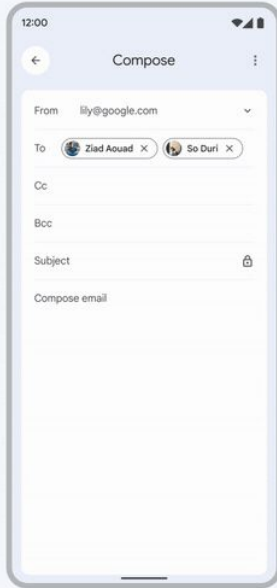
Bu pattern, Dialog, Menu, Snackbar ya da FAB gibi ekrana girip/çıkan UI elementlerin kullanılacağı durumlarda tercih edilmeli.



Fade



Fade



Speed

Oscillation

Customization

→ Speed

Transition ve Easing Süreleri

→ Motion Path

Linear ya da Arched

→ Oscillation

Bounce efekti

→ Stagger

Fade in delay

Motion Path

Stagger

