Flutter'da Kullanıcı Etkileşimi ve Jest Yönetimi

Dokunuşlardan Sürüklemeye: **Uygulamalara Hayat Vermek**

Roboto
Statik arayüzlerin ötesine geçiyoruz. Bu sunumda, basit dokunuşlardan fizik tabanlı sürükleme işlemlerine kadar Flutter'ın etkileşim katmanlarını 'Aşamalı Teknik Derinleşme' (Progressive Technical Deep Dive) yöntemiyle inceleyeceğiz.

- Temel Jestler (GestureDetector)
- Kaydırarak Silme (Dismissible)
- Sürükle ve Bırak (Draggable & DragTarget)

Jest (Gesture) Nedir?

Bir jest, işaretçi (parmak) tarafından gerçekleştirilen ve anlamsal bir karşılığı olan eylemdir:

‘Dokunma’ (tap), ‘Kaydırma’ (slide) veya ‘Sürükleme’ (drag).

Bu ham veriler, Flutter tarafından işlenerek anlamlı komutlara dönüştürür.

[İşaretçi Girişi] → [Ham Veri] → [Flutter İşleme] → [Anlamlı Komut]

`JetBrains Mono`: Herhangi bir widget’ı sarmalayarak ona ‘etkileşime girebilme’ yeteneği kazandırır.

Statik Nesneleri Etkileşime Açmak

Normalde `Image` gibi widget'ların dahili bir `onPressed` özelliği yoktur. `GestureDetector` bir sarmalayıcı (wrapper) olarak burada devreye girer.

```
GestureDetector(
  onTap: () => debugPrint('Click!'),
  onDoubleTap: () => debugPrint('Double click!'),
  child: Image.asset('assets/logo.png'),
)
```

Kritik Karar: Ne Zaman GestureDetector Kullanmalı?

Tekerleği yeniden icat etmeyin. Önce Flutter'ın hazır çözümlerini kontrol edin.

DO (Doğru Kullanım)

```
IconButton(
  icon: Icon(Icons.add),
  onPressed: () {})
)
```

Neden? Dahili `onPressed`, su dalgası (ripple) efekti ve erişilebilirlik standart gelir.

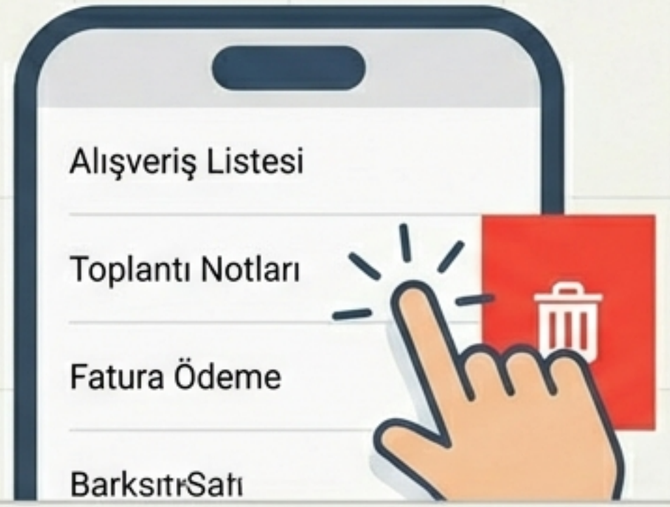
DON'T (Yanlış Kullanım)

```
GestureDetector(
  onTap: () {},
  child: Icon(Icons.add))
)
```

Neden? Çalışır ama gereksizdir; doğal tıklama hissiyatından yoksundur.

UI Deseni: Kaydırarak Silme (Swipe to Dismiss)

E-posta kutuları veya yapılacaklar listeleri... Bir öğeyi silmek için butona tıklamak yerine onu sağa veya sola fırlatmak mobil deneyimin doğal bir parçasıdır.



Bu Bölümün Hedefleri:

- Widget: Dismissible
- State: ChangeNeOifler (Veri yönetimi)
- Feedback: AlertDialog (Onay mekanizması)

Altyapı: State Yönetimi Kurulumu

Silme işlemi sadece görsel değildir; verinin kendisinin listeden atılmasını gerektirir.

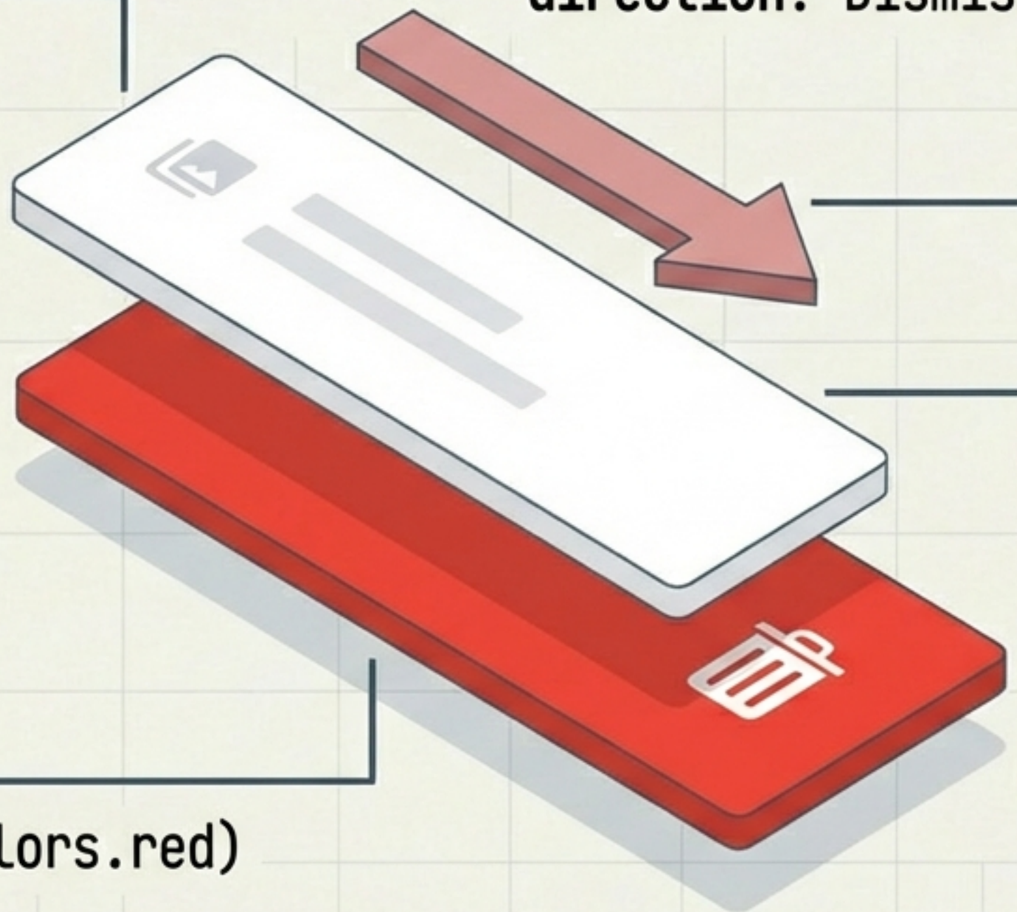
```
class SourceList with ChanageNoifler {
  final _myList = List.generate(10, (i) => 'Number $i');

  // Listeyi doğrudan dışarı açmıyoruz
  List get values => UnmodifiableListView(_myList);

  void removeItem(int index) {
    _myList.removeAt(index);
    notifyListeners(); // Kritik nokta!
  }
}
```

`notifyListeners()` olmadan UI değişikliği algılayamaz.

Dismissible Widget Anatomisi

key: Key(item)
(Zorunlu/Unique)

direction: DismissDirection.startToEnd



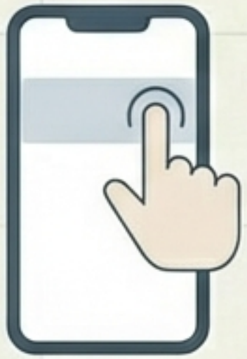
child: ListTile

background:
Container(color: Colors.red)

Aksiyon Anı: onDismissed Callback'i

Kullanıcı parmağını çektiğinde ne olur?

User Swipe



onDismissed Trigger

```
onDismissed: (direction) =>
  list.removeItem(index)
```

Logic Execution

`list.removeAt(index)` runs

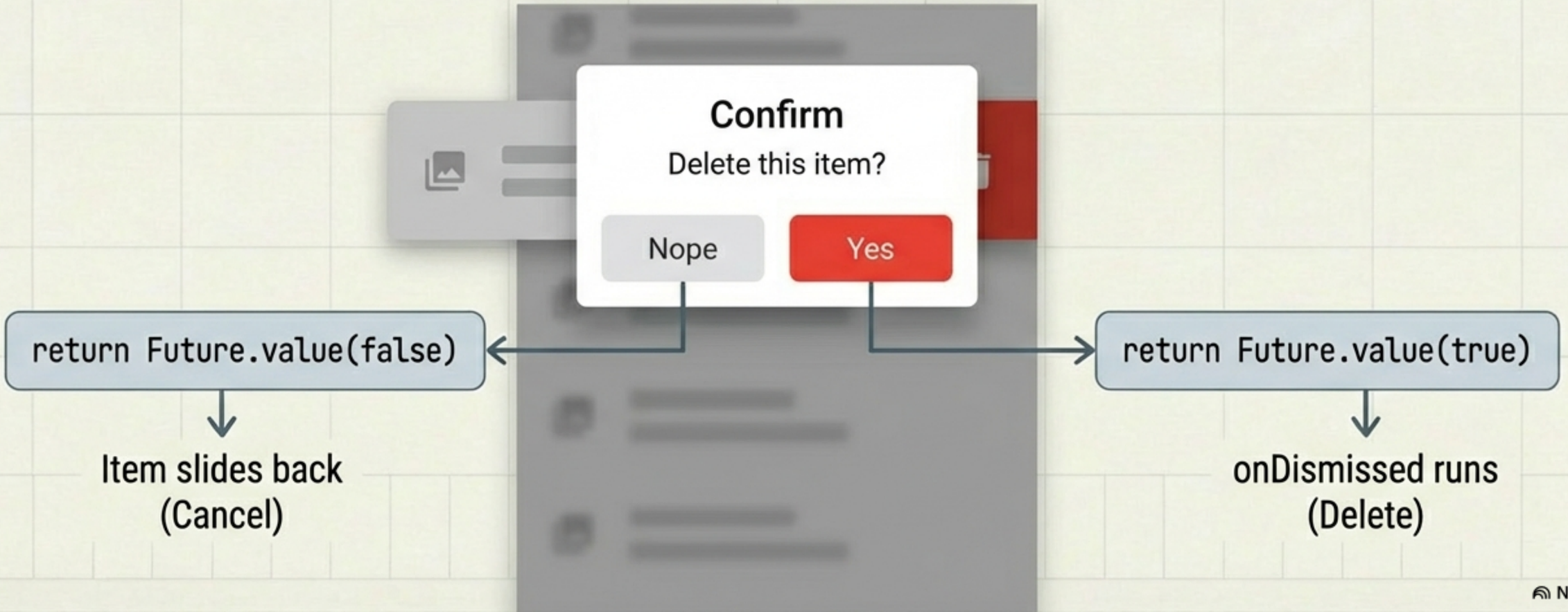
UI Rebuild

`notifyListeners()` refreshes the list

Bu döngü sayesinde görsel eylem ile veri durumu (state) senkronize kalır.

Güvenlik Ağı: İşlemi Onaylama

Kullanıcı yanlışlıkla kaydırırrsa veri kaybolur.
`confirmDismiss` ile araya girerek bir onay kutusu gösteririz.

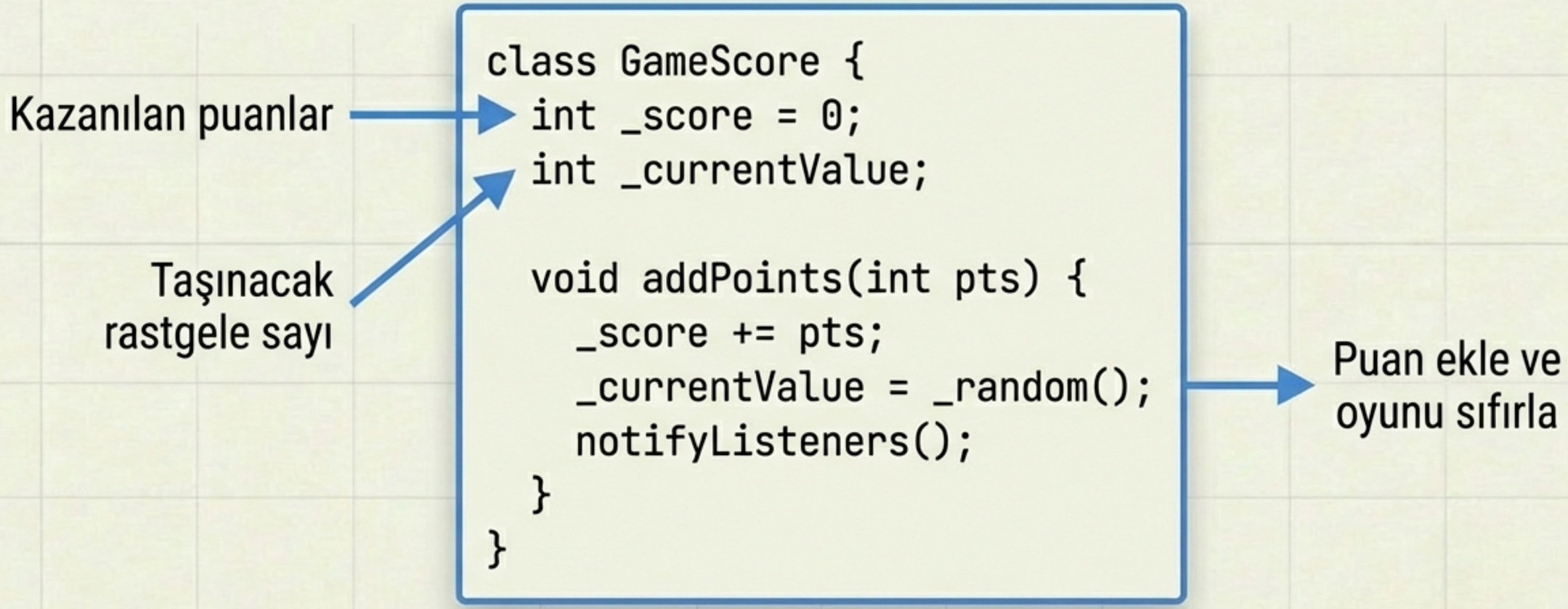


Oyunlaştırma: Sürükle ve Bırak (Drag & Drop)

Ekranda rastgele bir sayı belirir.
Tek ise 'Tek' kutusuna, çift ise 'Çift' kutusuna sürüklenir.

`Dragggable<T>` & `DragTarget<T>`

Oyun Mantığı (Game State)



Hareket Eden Parça: Draggable Widget

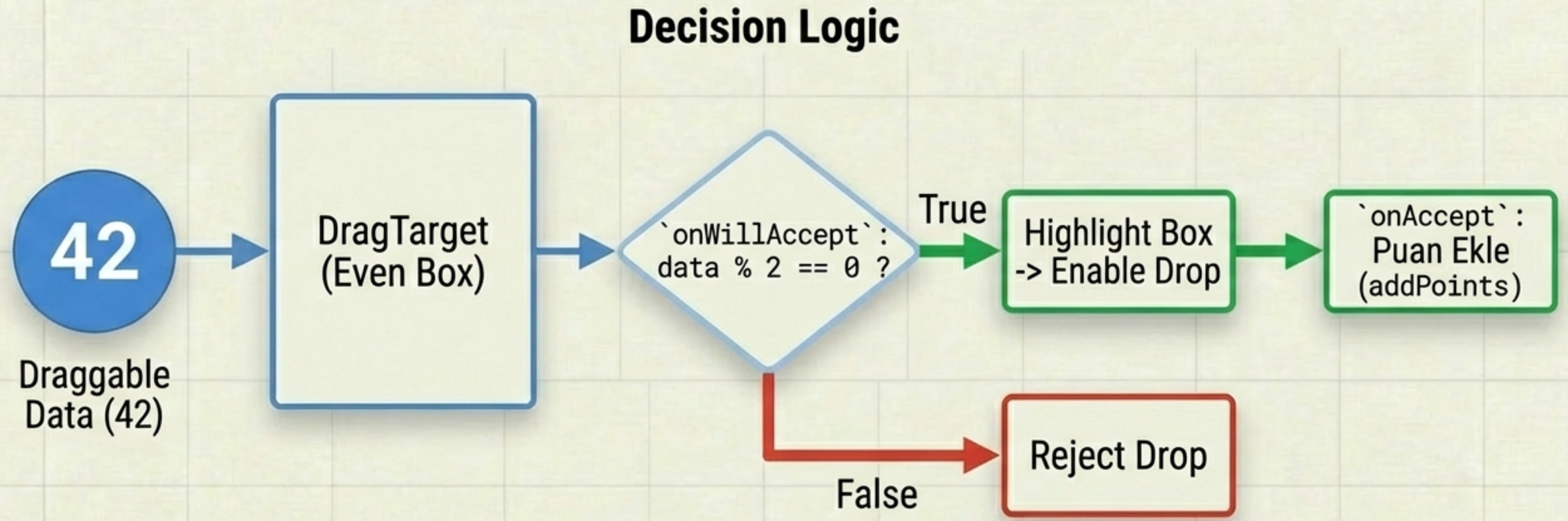
Idle
(Dururken)

Feedback
(Sürüklenirken)

Data
(Taşınan Veri)

“**Draggable**”, veriyi (data) fiziksel bir nesne gibi ekranda taşımanızı sağlar.

Hedef Bölgesi: DragTarget



DragTarget, üzerine gelen veriyi kabul edip etmeyeceğine karar veren bir filtredir.

Geri Bildirim Döngüsü ve Mixin

Kullanıcı puan kazandığında anlık bilgi vermek için SnackBar kullanımı.

Mixin Tanımı

```
mixin SnackMessage {
  void showMessage(BuildContext context, String text) {
    ScaffoldMessenger.of(context).showSnackBar(...);
  }
}
```

Özet: Etkileşim Katmanları

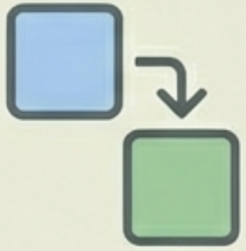
Dokunma

`GestureDetector` ile her yer tıklanabilir, ancak `IconButton` gibi standartlar önceliklidir.

Temizleme

`Dismissible` ile listeler yönetilir, `confirmDismiss` ile hatalar önlenir.

Taşıma



`Draggable` ve `DragTarget` ile fizik tabanlı, oyunlaştırılmış arayüzler kurulur.

Kullanıcı etkileşimi sadece tıklamak değildir; akıcı, doğal ve geri bildirim veren bir deneyim tasarlamaktır.