

# FLUTTER

## Basit Ayarları Saklama

21. hafta • Yerel Veri Saklama

36 hafta, 72 bağımsız çalışan Flutter projesi, gerçek uygulama ekranları, uygulamalı dersler ve öğrenci geri bildiriminden sonra incelenebilen çözüm kaynakları.

**Bu kitap güncel ders kaynaklarından otomatik üretilmiştir; özet ya da jenerik şablon değildir.**

# Öğrenme Haritası

| Hafta | Ders                   | Projeler                        |
|-------|------------------------|---------------------------------|
| 21    | Basit Ayarları Saklama | Tema Tercihi • Sessiz Mod Ayarı |

# 21. Hafta — Basit Ayarları Kalıcı Saklama

## Bu hafta ne yapacağız?

Bir tema anahtarını değiştirmek kolaydır; asıl problem uygulama kapatıldığında seçimin kaybolmamasıdır. Bu hafta `shared_preferences` paketiyle küçük ayarları cihazda anahtar-değer biçiminde saklayacağız. Konu uygulamasında açık/koyu tema seçimini, özgün uygulamada sessiz mod, titreşim ve bitiş saati seçeneklerini kalıcı hâle getireceğiz.

## Kazanımlar

- Kalıcı veri ile yalnız bellekte yaşayan durum arasındaki farkı açıklama
- `SharedPreferences` örneğini asenkron olarak edinme
- `bool`, `double` ve diğer basit türleri anahtarla yazma ve okuma
- Kayıt bulunmadığında güvenli varsayılan değer kullanma
- Veriyi yüklemekten önce etkileşimi devre dışı bırakma
- Kalıcılığı otomatik testte doğrulama

## Kurulum

`pubspec.yaml` dosyasına paket eklenir:

```
dependencies:  
  flutter:  
    sdk: flutter  
  shared_preferences: ^2.5.3
```

Paket eklendikten sonra `flutter pub get` çalıştırılır.

## Çözümlü uygulama — Tema Tercihi

### 1. Durum değişkenleri

```
bool koyu = false;  
bool yukleniyor = true;
```

`koyu`, arayüzde kullanılacak değeri; `yukleniyor`, cihazdaki eski tercih henüz okunurken düğmenin etkin olup olmayacağını belirtir.

### 2. Uygulama açılırken tercihi okuma

```

@override
void initState() {
  super.initState();
  tercihYukle();
}

Future<void> tercihYukle() async {
  final tercihler = await SharedPreferences.getInstance();
  if (!mounted) return;
  setState(() {
    koyu = tercihler.getBool('koyuTema') ?? false;
    yukleniyor = false;
  });
}

```

getBool anahtar daha önce yazılmadıysa null döndürür. ?? false ilk açılışta açık temayı seçer. Asenkron işlem tamamlanana kadar widget ekrandan kaldırılmış olabileceği için mounted kontrol edilir.

### 3. Yeni tercihi kaydetme

```

Future<void> temaDegistir(bool deger) async {
  final tercihler = await SharedPreferences.getInstance();
  await tercihler.setBool('koyuTema', deger);
  if (mounted) setState(() => koyu = deger);
}

```

Kaydetme tamamlandıktan sonra MaterialApp.themeMode güncellenir:

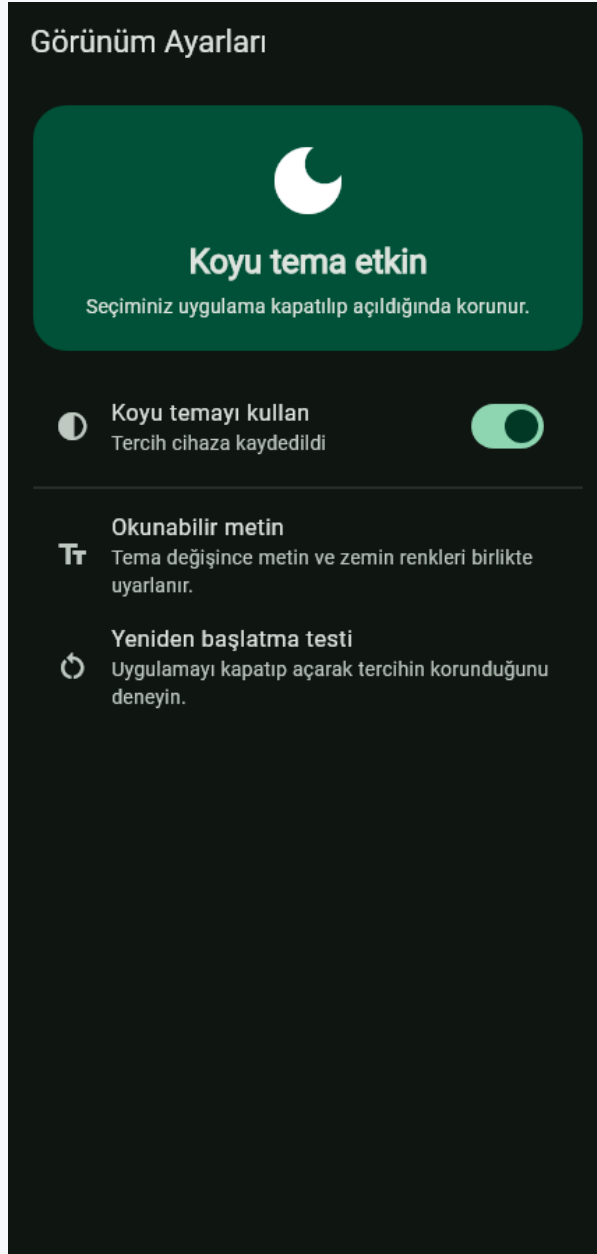
```

themeMode: koyu ? ThemeMode.dark : ThemeMode.light,

```

## Gerçek uygulama ekranı

Test boş bir depolamayla başlamış, koyu tema anahtarına basmış, arayüzün koyulaştığını ve koyuTema anahtarının true olarak yazıldığını doğrulamıştır.



Tema Tercihi gerçek Flutter ekranı

## Özgün uygulama — Ders Sessiz Modu

Bu uygulamada üç farklı tercih saklanır:

```
bool sessiz = false;  
bool titresim = true;  
double bitis = 22;
```

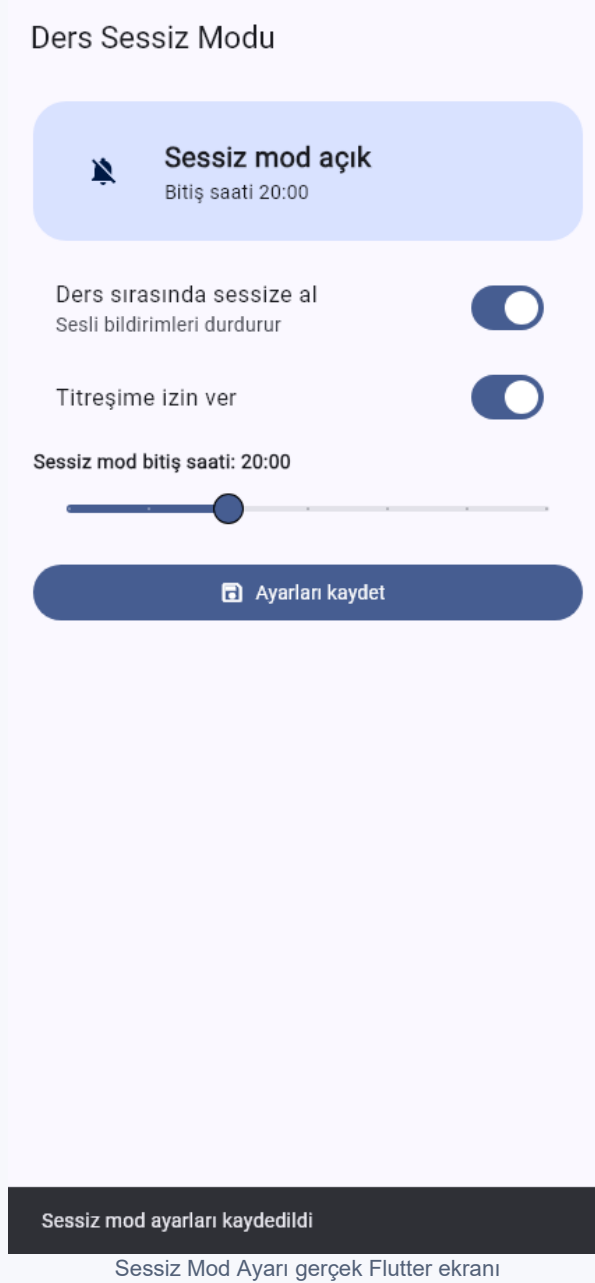
### Birden fazla ayarı okuma

```
sessiz = tercihler.getBool('sessiz') ?? false;  
titresim = tercihler.getBool('titresim') ?? true;  
bitis = tercihler.getDouble('bitis') ?? 22;
```

### Ayarları birlikte kaydetme

```
await Future.wait([
  tercihler.setBool('sessiz', sessiz),
  tercihler.setBool('titresim', titresim),
  tercihler.setDouble('bitis', bitis),
]);
```

`Future.wait`, birbirinden bağımsız kayıt işlemlerinin tamamlanmasını birlikte bekler. İşlem sonunda `SnackBar` ile kullanıcıya açık geri bildirim verilir.



Sessiz Mod Ayarı gerçek Flutter ekranı

## SharedPreferences ne zaman kullanılmalı?

Uygundur:

- tema, dil, ses ve görünüm tercihleri,
- tanıtım ekranının görülüp görülmediği,
- küçük sayaçlar veya son seçilen filtre.

Uygun değildir:

- çok sayıda ilişkili kayıt,
- büyük metin ve dosyalar,
- parola veya gizli anahtarlar,
- karmaşık sorgu gerektiren veriler.

Bu tür ihtiyaçlarda güvenli depolama, dosya veya SQLite gibi farklı çözümler seçilir.

## Uygulama görevleri

### Kolay — Yazı boyutu tercihi

Küçük, orta ve büyük seçeneklerini saklayın. Uygulama yeniden açıldığında metin boyutu geri yüklensin.

### Orta — Çalışma zamanlayıcısı

Varsayılan çalışma ve mola sürelerini iki sayı olarak kaydedin. Geçersiz süreleri engelleyin ve sıfırlama düğmesi ekleyin.

### İleri — Erişilebilirlik ayarları

Yüksek kontrast, büyük metin, animasyon azaltma ve tercih edilen renk seçeneklerini saklayan bir ekran geliştirin. Tüm seçenekleri fabrika ayarlarına döndürebilin.

## Test ve kontrol

- Anahtar hiç yokken varsayılan değer kullanılmalı.
- Kaydın ardından uygulama yeniden oluşturulduğunda seçim korunmalı.
- Bir ayarın değişmesi diğer anahtarları bozmamalı.
- Yükleme sırasında kullanıcı eski değer üzerine yazamamalı.
- Kayıt tamamlandığında görünür geri bildirim verilmeli.

## Geri bildirim ve çözüm erişimi

Öğrenci önce kendi kalıcı ayar uygulamasını gönderir. Geri bildirimden sonra İpucunu göster anahtar adları ve asenkron akış hakkında yön verir. Örnek çözümü göster gerçek `lib/main.dart` kodunu; indirme bağlantısı ise `pubspec.yaml` dâhil eksiksiz Flutter projesini sunar.

## Kaynak projeler

- Konu uygulaması: 06 - Yerel Veri Saklama/Uygulamalar/Uygulama-21-Tema Tercihi
- Özgün uygulama: 06 - Yerel Veri Saklama/Uygulamalar/Ozgun-21-Sessiz Mod Ayar■