

Görsel Programlama

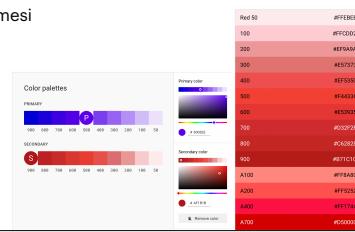
Temalar

Emir ÖZTÜRK

Uygulama Teması Oluşturma

Renkler

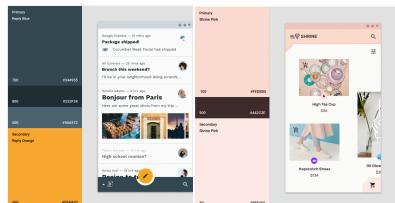
- Renk paletleri
- Uygulamanın renklerinin belirlenmesi
- Material Palette Generator
- Renklerin karşılık gelen anlamları
- Logo rengine uygunluk



Uygulamanın teması genellikle farklı renklerin kombinasyonu olarak belirlenebilir. Bu tarz gruplar renk şeması olarak adlandırılabilir. Genellikle seçilen renklerin arasında bir uyum bulunması uygulamanın daha profesyonel gözükmesi adına önemlidir. Renk paletlerine erişmek için online belirli araçlar bulunmaktadır.

Uygulama Teması Oluşturma

- primaryColor
- accentColor (Secondary)
- Floating Button



Flutter'da uygulamanın ana rengini belirleyen özellik `primaryColor` olarak adlandırılmaktadır. Alt rengi ise `accentColor` belirler. Uygulamanın daha az dikkat çeken ya da ek işlev sahip elementleri bu `accentColor` özelliğini kullanır.

Uygulama Teması Oluşturma

- Font
- ttf
- Google Fonts
- Font ailesi
- Font ağırlığı
- Uygulamanın formatına uygunluk



Uygulamada font seçimi de uygulamanın kabul edilebilirliği için önemlidir. Font dosyaları ttf uzantısıyla bulunabilmektedir. Google fonts üzerinden belirli fontların ailesi indirilebilmekte ve bunların ağırlık kalınlık veya boyut özellikleri uygulama içerisinde belirlenebilmektedir.

Uygulama Teması Oluşturma

Uygun fontun seçilmemesi



Doğru font seçiminin önemi

Stil

- Bir çok widget style ile değiştirilebilir
- Renk
- Text görünüsü
- Çerçevevler
- Arkaplan resimleri

Uygulamanın parçalarının renk ve şekilleri çoğunlukla style parametresi ile belirlenir.

Text Stilleri

- style:
 - color
 - decoration
 - fontSize
 - fontStyle
 - fontWeight
 - fontFamily

Bir text bileşeninin rengi, font büyüğlüğü ve türü, font boyutu gibi özellikle TextStyle sınıfı ile belirlenir

Text Stili Font ekleme

- flutter/fonts klasörüne font dosyalarının atılması
- pubspec.yaml
 - flutter:
 - fonts:
 - family: FontAilesi
 - fonts:
 - asset: fonts/font-dosya-adi.ttf
 - pubspec.yaml altında boşlukların önemi

Bir font eklenmek isteniyorsa ana klasöre fonts klasörü açılabilir. Bu klasörün altına atılacak ttf dosyaları pubspec.yaml'da tanımlanmalıdır. Daha sonra bu fontlar uygulama içerisinde kullanılabilmektedir.

Text Stili Google Fonts

- flutter pub add google_fonts
- flutter pub get
- import 'package:google_fonts/google_fonts.dart';

```
style: GoogleFonts.poppins(
  fontSize: 30,
  fontWeight: FontWeight.bold,
  color: Colors.blueAccent,
),
```

Bir font eklenmek isteniyorsa ana klasöre fonts klasörü açılabilir. Bu klasörün altına atılacak ttf dosyaları pubspec.yaml'da tanımlanmalıdır. Daha sonra bu fontlar uygulama içerisinde kullanılabilmektedir.

Container Stili

Dekorasyon

- decoration: BoxDecoration
- color
- boxShadow -> BoxShadow()
- border -> Border()
- Border.all()
- borderRadius -> BorderRadius()
- shape -> BoxShape

Uygulamada yalnızca text bileşenleri bulunmamaktadır ve diğer bileşenlerin de uygulama ile bütünlük göstermesi önemlidir. Bu sebeple belirli widget stilleri de kullanılmaktadır. Örneğin bazı bileşenlerin stilini belirlemek adına dışına container ekledikten sonra decoration özelliği kullanılabilir.

Temalar

- Stillerin birbirinden farklı olması
- Widget stillerinin takibi
- Uygulamanın dizayn bütünlüğü

Uygulamanın her sayfasının birbirine benzer olması gerekmektedir. Her sayfanın font ve renklerinin aynı kaynaktan çekilebilmesi için theme özelliği kullanılabilir.

Temalar

- CSS
 - Tüm bileşenlerin otomatik olarak dizayn elementlerini alması
 - Stilin miras alınması
 - Takip zorluğu
- Flutter
 - Bileşenlerin ortak temayı kullanacağının belirtilmesi
 - Açık belirtme

Css üzerinde bileşenler belirlenen sınıf ya da tür etiketlerine göre özelliklerini alır. Bu durum kodlama esnasında hız sağlasa da belirli özelliklerin diğerlerinden sonra gelip ezme işlemi yapması sebebiyle istenmeyen çıktıların oluşması ve hata olasılığı artmaktadır. Flutter'da ise bileşenlerin tümünün nasıl bir tema kullanacağı açık bir şekilde belirtilmelidir.

Uygulama Teması Oluşturma

- MaterialApp
 - theme:
- ThemeData()
 - brightness
 - fontFamily
 - textTheme

Theme özelliğine verilen bir ThemeData ile uygulamanın ana teması belirlenebilir.

Temanın kullanılması

- Theme.of(ctx)
- Renk
 - Theme.of(ctx).colorScheme.primary
- Text
 - Theme.of(ctx).textTheme.headline

Daha sonra bu temanın özelliklerine erişmek için herhangi bir widget'in atanmak istenen özelliği context üzerinden belirlenebilir. Örneğin bir textbox'ın color özelliği değiştirilmek isteniyorsa Theme.of(context) üzerinden renk şemasına erişilebilir ve buradan ana ve yan renkler elde edilebilmektedir.

Temanın kullanılması

- <https://m3.material.io/get-started>
- <https://docs.flutter-dev.translate.goog/ui/design/cupertino>

Daha sonra bu temanın özelliklerine erişmek için herhangi bir widget'in atanmak istenen özelliği context üzerinden belirlenebilir. Örneğin bir textbox'ın color özelliği değiştirilmek isteniyorsa Theme.of(context) üzerinden renk şemasına erişilebilir ve buradan ana ve yan renkler elde edilebilmektedir.