Activitat d'avaluació 1.3 - Crea una pantalla a partir del disseny

Fitness Time

CIFO L'Hospitalet
Desenvolupament d'Aplicacions mòbils per iOS i Android amb Flutter

Alumne
Carles Lázaro Costa

Tutor
Eduard Carreras

4 d'octubre de 2024

Índex

Fitness Time

Repositori del codi: cifo_flutter_fitness_time

Repositori d'aplicacions: cifo_flutter

Activitat de creació de pantalles a partir del disseny d'una aplicació esportiva.

Una primera pantalla d'inici mostra un missatge de benvignuda a l'aplicació, les darreres activitats esportives d'una persona, un indicador de progrés del seu objectiu mensual i una barra de navegació interactiva a la part inferior.

En fer click a la imatge de perfil de la part superior dreta es mostra una segona pantalla amb la imatge més gran i altres dades de la persona, juntament amb estadístiques de totes les seves activitats. També es pot accedir al perfil mitjançant el menú lateral que s'obre fent click a la icona de la part superior esquerra de la pantalla d'inici.

Instal·lació

1. S'ha d'haver instal·lat el Flutter SDK.
2. Clonar el repositori:

```
git clone https://github.com/Carleslc/cifo_flutter_fitness_time.git
# GitHub CLI: gh repo clone Carleslc/cifo_flutter_fitness_time

cd cifo_flutter_fitness_time
```

3. Instal·lar les dependències:

```
flutter pub get
```

4. Executar l'aplicació amb `flutter run` o desde l'IDE.

Estructura de l'aplicació

```
lib
  main.dart
  models
      activity.dart
      distance.dart
      user.dart
  screens
      home_page.dart
      home_screen.dart
      profile_screen.dart
      template_page.dart
  styles
      app_styles.dart
      color_styles.dart
      text_styles.dart
  utils
      date_utils.dart
      duration_utils.dart
      number_utils.dart
  widgets
       activity_card.dart
       card_attribute.dart
       goal_indicator.dart
       slider_attribute.dart
       text_link.dart
       user_avatar.dart
```

L'inici de l'aplicació és a `main.dart`.

A la carpeta `models` es troben les classes de domini, com `User`, `Activity` i `Distance`.

Les diferents pantalles de l'aplicació es troben a la carpeta `screens`.

A la carpeta `styles` es troben els estils de l'aplicació, que es centralitzen i s'accedeixen mitjançant la classe `AppStyles` definida a `app_styles.dart`. Les altres classes d'estils són per organitzar millor el codi.

A `widgets` hi ha els widgets propis que no es corresponen amb una pantalla determinada, com per exemple `ActivityCard` o `GoalProgressIndicator`, entre d'altres.

S'ha afegit una carpeta `utils` per utilitats i extensions que ajudin al formateig de diferents dades, com la classe `DateTimeUtils` i l'extensió `CustomDateTimeFormatter` dins de `date_utils.dart` per facilitar el formateig i tractament de dates, l'extensió `CustomNumberFormatter` a `number_utils.dart` per donar format a un número decimal o l'extensió `CustomDurationFormatter` a `duration_utils.dart` per donar format a una duració.

Notes del desenvolupament

Per començar l'aplicació he llegit el pdf Activitat d'avaluació 1.3 - Crea una pantalla a partir del disseny per conèixer alguns dels recursos utilitzats (colors, icones, font, imatges...) així com alguns dels widgets que s'han d'utilitzar.

S'han definit els estils a la classe estàtica `AppStyles` (`styles/app_styles.dart`). La classe és `abstract` per que no pugui ser instanciada (l'aternativa seria utilitzar un constructor privat `const AppStyles._()`) i `final` per que no pugui ser heredada (no tindria sentit ja que una subclasse a Dart no té accéss als atributs estàtics de la classe pare).

Per organitzar millor les diferents parts de la classe d'estils, com els colors i els textos, s'han definit les classes `ColorStyles` (`styles/color_styles.dart`) i `TextStyles` (`styles/text_styles.dart`) que s'accedeixen mitjançant getters a la classe d'estils principal `AppStyles`, per exemple: `AppStyles.color.primary`. Tenen accés privat perque només `AppStyles` pugui instanciar-les ja que formen part de la mateixa llibreria utilitzant les directives `part` i `part of`. Altra forma és possar aquestes classes dins del mateix fitxer `app_styles.dart` però crec que en aquest cas queda millor organitzat en fitxers separats.

La meva idea inicial era crear subclasses estàtiques per accedir amb `AppStyles.Colors.primary` però he optat per l'implementació actual degut a que a Dart no es poden crear static nested classes de la mateixa manera que a altres llenguatges com Java, i les convencions de noms també són diferents.

Dins de la classe `AppStyles` es defineix un mètode per generar el tema de l'aplicació a partir d'un tema per defecte i les constants d'estils del disseny, i s'utilitza a `main.dart` amb `AppStyles.theme(context)`, que s'assigna al paràmetre `theme` de `MaterialApp`.

S'ha investigat la documentació de la llibreria `percent_indicator` per utilitzar el widget `CircularPercentIndicator` a la pantalla d'inici. S'ha agrupat el seu ús al widget

GoalProgressIndicator (widgets/goal_indicator.dart).

S'han organitzat les diferents pantalles a la carpeta screens.

Primer vaig crear la pantalla principal home_screen.dart per mostrar el missatge de benvinguda i les darreres activitats de la persona que utilitza l'aplicació.

El codi després de diverses iteracions i després d'afegir un PageView s'ha dividit finalment en dos widgets, HomeScreen per controlar l'estat de la vista actual i tot el layout exterior de la pantalla (appBar, drawer, bottomNavigationBar) i HomePage, definit a screens/home_page.dart, pel disseny del body de la vista principal.

Després s'ha creat la pantalla profile_screen.dart per mostrar la foto de perfil, així com les altres dades de la persona i les estadístiques de les seves activitats. S'ha utilitzat el widget Hero per animar la imatge de perfil durant la navegació entre pantalles. El widget UserAvatar definit a widgets/user_avatar.dart es reutilitza al botó de la AppBar i a la pantalla del perfil. A més, des de la pantalla de perfil es pot fer click a la imatge per tancar la pantalla de perfil i tornar enrere.

Per enriquir l'aplicació i organitzar millor el codi s'han creat diverses classes de domini a la carpeta models: User per modelar el perfil de l'usuari, Activity per modelar les activitats de l'usuari amb una classe abstracta i una classe concreta RunningActivity que és el tipus d'activitat que s'utilitza al disseny proporcionat, i Distance que modela la distància d'una activitat de *running*.

Al fitxer models/user.dart es defineix l'usuari plantilla del disseny, i al fitxer models/activity.dart es defineixen les seves activitats que es mostren al disseny a la pantalla d'inici.

La idea d'aquestes classes és simular una mica la llògica real que tindria l'aplicació, per tal de que no sigui només el disseny amb totes les dades fixes com valors de texte. D'aquesta manera l'ús dels models s'aproxima millor a com seria aquesta aplicació en producció, per exemple si s'agafessin les dades de l'usuari i les seves activitats d'un backend.

Les classes Activity i RunningActivity s'utilitzen als widgets ActivityCard i RunningActivityCard definits a widgets/activity_card.dart, que corresponen a les tarjetes de les "Darreres activitats" de la pantalla d'inici.

Per mostrar les estadístiques de l'usuari (*Time, Km, Activities*) a la pantalla del perfil s'utilitza el widget CardAttribute definit a widgets/card_attribute.dart, i per mostrar els valors de *Height* i *Weight* s'utilitza el widget SliderAttribute definit a widgets/slider_attribute.dart. Per evitar que el slider sigui interactiu es passa un callback onChanged que no fa res i per evitar que apareguin animacions quan es prem a sobre del slider s'utilitza el widget AbsorbPointer.

S'ha creat el fitxer utils/date_utils.dart per implementar el formateig de les dates i hores, utilitzant la classe DateFormat de la llibreria intl per crear els formats propis tal i com es mostren al disseny. La llibreria s'inicialitza al carregar l'aplicació al fitxer main.dart. També s'ha afegit el format de la distància a la classe Distance, utilitzant l'extensió de utils/number_utils.dart i la classe NumberFormat de la llibreria intl, i per últim el format de la duració, en una extensió a utils/duration_utils.dart. Això m'ha servit per aprendre com formatejar diversos tipus de dades amb Dart i Flutter, per tal de que quedin les dades amb el mateix format que al disseny.

Després he fet el `BottomNavigationBar` interactiu afegint un `PageView` que canvia horitzontalment entre els widgets `HomePage` i `TemplatePage` definits a la carpeta `screens`. Per fer-ho he seguit aquest article proporcionat a la pàgina del curs. He afegit la classe `PageScreen` al final del fitxer `home_screen.dart` per organitzar les pàgines navegables mitjançant `PageView` a la pantalla principal.

També he afegit un `Drawer` a la pantalla principal per completar la funcionalitat del botó del menú. Aquest menú es una altra manera d'accedir a la pantalla del perfil o de navegar entre les diferents pàgines de la pantalla.

Com a nota personal per altres aplicacions, segons la documentació de `BottomNavigationBar`, un widget similar més actual per Material *3* és `NavigationBar`. També, segons he vist a la documentació del `Drawer`, un widget més idoni per Material *3* és `NavigationDrawer`, però no l'he fet servir per què ho he vist després i sembla una mica més enrevesat, així que m'ho apunto per un altre projecte. El `Drawer` actual funciona prou bé per aquesta aplicació.

Més tard he vist que es mostra un tooltip en anglès per la icona de navegació del menú o de tornar enrere, amb una pulsació llarga del botó al mòvil o quan es passa per sobre a la web. Com que l'aplicació està principalment en català, he buscat a la documentació com canviar la localització d'aquests missatges dels widgets propis de Flutter, utilitzant la llibreria del sdk `flutter_localizations` i modificant alguns atributs de `MaterialApp` al fitxer `main.dart`.

També he investigat sobre els temes de Flutter i les especificacions Material *3* sobre els colors i els textos, i he documentat i refactoritzat les classes `AppStyles`, `ColorStyles` i `TextStyles` de la carpeta `styles` per afegir un tema adeqüat als colors i textos del disseny i que sigui prou complert i útil per que em serveixi d'exemple per futures aplicacions, però potser ha quedat una mica difícil d'entendre i segurament existeixen millors solucions.

Per últim he fet que l'enllaç "Més detalls" sigui clickable i s'obri una pàgina web, utilitzant la llibreria url_launcher i els widgets `RichText` i `TextSpan`. La implementació està al widget `TextLink` definit a `widgets/text_link.dart`.

Les principals dificultats que he trobat han sigut el modelat i formateig de les dades per que quedin igual que al disseny sense possar els textos a mà i la creació de les classes per organitzar els estils i crear el tema, així com cuadrar els colors i la font amb el disseny, tenint en compte que els colors visualment canvien una mica segons la pantalla i segons el dispositiu on s'executi. El widget més difícil ha sigut `ActivityCard`, que involucra l'ús de diverses classes com `Activity` i `RunningActivity` i ha de formatejar la data, hora i distància.

He utilitzat l'IDE *Visual Studio Code* durant pràcticament tot el desenvolupament, utilitzant principalment un mòvil físic Pixel 8 amb Android 14 (API *35*). També he provat l'aplicació amb un emulador amb Android 10 (API 29) i mitjançant l'IDE web Project IDX.

Imatges

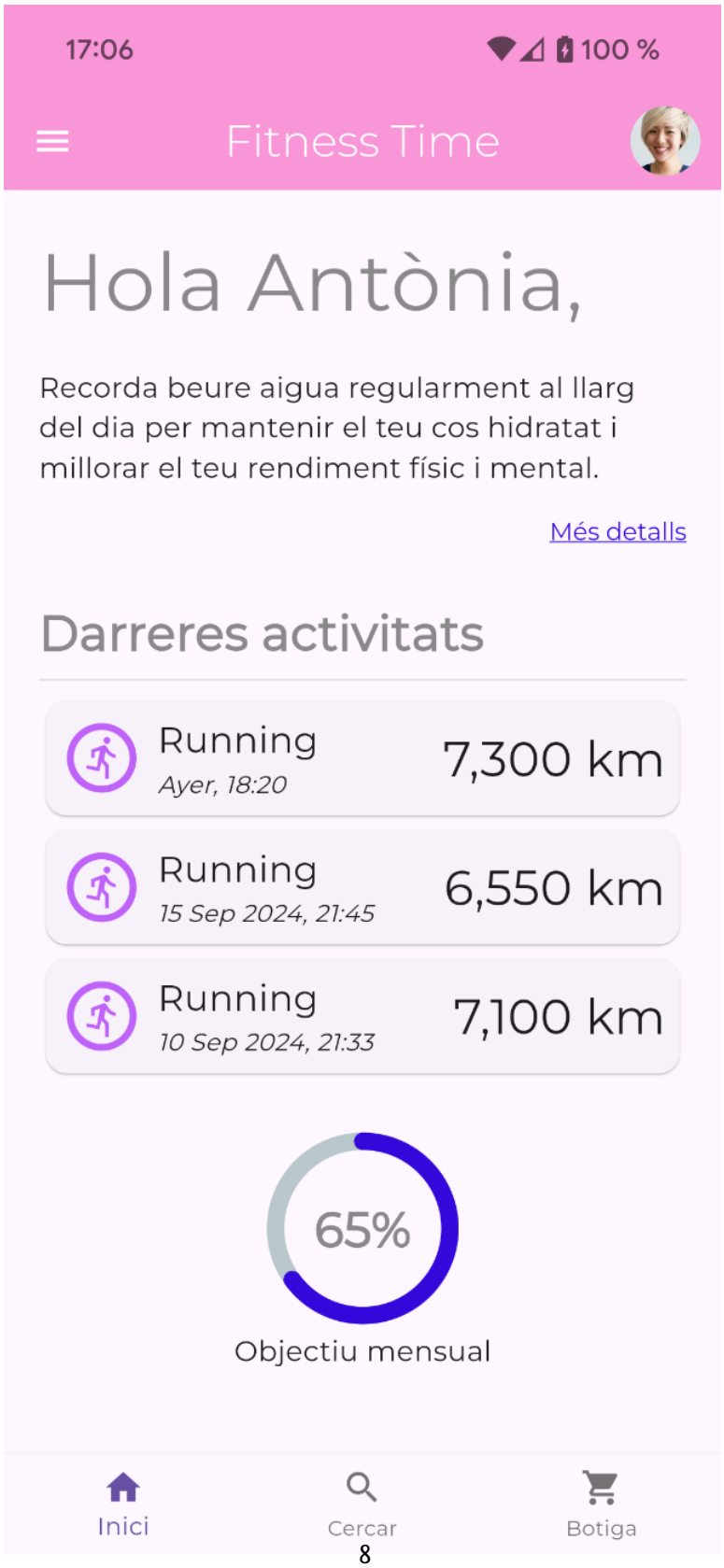

Figura 1: fitness_time_1.png

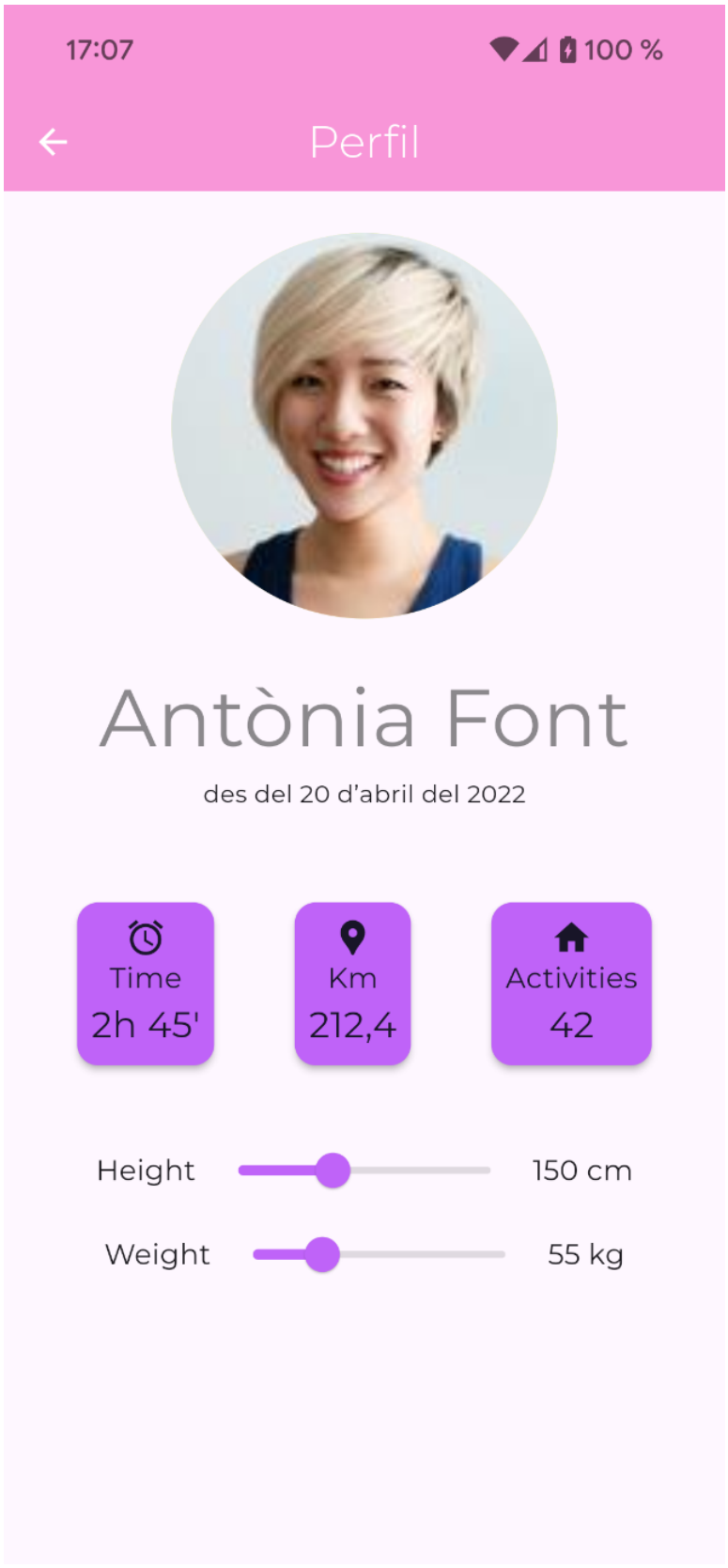

Figura 2: fitness_time_2.png

Recursos

Relactionats amb el disseny de l'aplicació:

- Imatge de perfil
- Paleta de colors
- Font: Montserrat

Relacionats amb Flutter:

- Material Widgets
- Themes
- `ThemeData` (`ColorScheme`, `TextTheme`)
- `SingleChildScrollView`
- `Card`
- `Hero`
- `Duration`
- `PageView`
- `BottomNavigationBar` / `NavigationBar`
- `Drawer` / `NavigationDrawer`
- `AbsorbPointer`
- `RichText`
- `MediaQuery`
- `CircularPercentIndicator` (percent_indicator)
- `DateFormat` (intl)
- `NumberFormat` (intl)
- Internationalization

Llibreries externes

- google_fonts
- percent_indicator
- url_launcher
- collection
- intl