WEATHER APP

Carlos Martinez

Jessenia Piza

Nuestra app…

- Nuestra app contiene:
  - Nombre de la ciudad
  - Temperatura
  - Descripción del clima
  - Sensación térmica
  - Humedad
  - Velocidad del viento
  - Presión atmosférica
- Nuestra app tiene la intención de actualizar estos valores en tiempo real a base de una API.
- Así mismo actualizar el background dependiendo de la descripción del clima

myhomepage.dart


```dart
@override
Widget build(BuildContext context) {
  Size size = MediaQuery.of(context).size;
  WeatherController weatherController = Get.find();
  dynamic id = 3688689;
  weatherController.updateWeather(id);
  Timer.periodic(const Duration(hours: 1), (Timer t) => weatherController.updateWeather(id));
  return Obx(()=>Scaffold(
    body: Container(
      decoration: const BoxDecoration(
        image: DecorationImage(
          image: AssetImage('assets/backgrounds/clear_sky.png'),
          fit: BoxFit.fill,
        ), // DecorationImage
      ), // BoxDecoration
```

myhomepage.dart


```
child:
Column(
  children: [
    SizedBox(height: size.height*0.05,),
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceAround,
      children: [
        Text('Feels Like', textAlign: TextAlign.center,
        style: GoogleFonts.rubik(color: Colors.white.withOpacity(0.7), fontSize: 18,
        )), // Text
        Text('Humidity', textAlign: TextAlign.center,
        style: GoogleFonts.rubik(color: Colors.white.withOpacity(0.7), fontSize: 18,
        )), // Text
      ],
    ), // Row
    Row(
      mainAxisAlignment: MainAxisAlignment.spaceAround,
      children: [
        Text('${weatherController.feelsLike}°C', textAlign: TextAlign.center,
        style: GoogleFonts.rubik(color: Colors.white, fontSize: 30,
        )), // Text
        Text('  ${weatherController.humidity}%', textAlign: TextAlign.center,
        style: GoogleFonts.rubik(color: Colors.white, fontSize: 30,
        )), // Text
      ],
    ), // Row
```

weather.dart

```dart
class Weather {
  dynamic id;
  String? name;
  double? temp;
  String? description;
  double? feelsLike;
  int? humidity;
  dynamic windSpeed;
  int? pressure;
  String? main;

  Weather({required this.id,
           required this.name,
           required this.temp,
           required this.description,
           required this.feelsLike,
           required this.humidity,
           required this.windSpeed,
           required this.pressure,
           required this.main});
```

```dart
  Weather.fromJson(Map<String, dynamic> json) {
    id = json['id'];
    name = json['name'];
    temp = json['temp'];
    description = json['description'];
    feelsLike = json['feels_like'];
    humidity = json['humidity'];
    windSpeed = json['wind_speed'];
    pressure = json['pressure'];
    main = json['main'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = <String, dynamic>{};
    data['id'] = id;
    data['name'] = name;
    data['temp'] = temp;
    data['description'] = description;
    data['feels_like'] = feelsLike;
    data['humidity'] = humidity;
    data['wind_speed'] = windSpeed;
    data['pressure'] = pressure;
    data['main'] = main;
    return data;
  }
}
```

Weather_controller.dart

```dart
class WeatherController extends GetxController {
  final _id = 0.obs;
  final _temp = 0.0.obs;
  final _city = ''.obs;
  final _description = ''.obs;
  final _feelsLike = 0.0.obs;
  final _humidity = 0.obs;
  final _windSpeed = 0.0.obs;
  final _pressure = 0.obs;
  final _listCities = [].obs;
  final _background = 'assets/backgrounds/clear_sky.png'.obs;

  dynamic get id => _id.value;
  double get temp => _temp.value;
  String get city => _city.value;
  String get description => _description.value;
  double get feelsLike => _feelsLike.value;
  int get humidity => _humidity.value;
  dynamic get windSpeed => _windSpeed.value;
  int get pressure => _pressure.value;
  List<String> get listCities => [... _listCities];
  String get background => _background.value;
```

```dart
// var timer = Timer.periodic(const Duration(seconds: 5), (Timer t) => updateWeather(_id.value));

Future<void> initDB() async {
  WeatherDB database = WeatherDB.instance;
    final String response = await rootBundle.loadString('assets/json/city_list.json');
    var jsonResponse = convert.jsonDecode(response);

    for (var i in jsonResponse){
      Weather weatherCity = Weather(id: i['id'], name: '', temp: 0.0, description: '', feelsLike:
      try{
        database.insertWeatherCity(weatherCity);
      }on Exception catch (_){
        // ignore: avoid_print
        print('Error');
      }
    }
}

Future<List<String>> listCity() async {
  Database database = await WeatherDB.instance.database;
  List<Map> res = await database.rawQuery(
                                    "SELECT name FROM weather_city");

  return List<String>.generate(res.length, (i) {for (var q in res){q['name'];}return '';});
```

Weather_controller.dart

```dart
if (weatherCity.main == 'Clear'){
  _background.value = 'assets/backgrounds/clear_sky.png';
}
else if(weatherCity.main == 'Clouds'){
  _background.value = 'assets/backgrounds/clouds.png';
}
else if(weatherCity.main == 'Drizzle'){
  _background.value = 'assets/backgrounds/drizzle.png';
}
else if(weatherCity.main == 'Rain'){
  _background.value = 'assets/backgrounds/rain.png';
}
else if(weatherCity.main == 'Snow'){
  _background.value = 'assets/backgrounds/snow.png';
}
else if(weatherCity.main == 'Thunderstorm'){
  _background.value = 'assets/backgrounds/thunderstorm.png';
}
else{
  _background.value = 'assets/backgrounds/clouds.png';
}
```

```dart
Future<void> updateWeather(dynamic newId) async {
  WeatherDB database = await WeatherDB.instance;
  WeatherClient weatherClient = WeatherClient(newId);
  Weather weatherCity = await weatherClient.getWeather();
  _id.value = newId;
  _temp.value = weatherCity.temp!;
  _city.value = weatherCity.name!;
  _description.value = weatherCity.description!;
  _feelsLike.value = weatherCity.feelsLike!;
  _windSpeed.value = weatherCity.windSpeed!;
  _humidity.value = weatherCity.humidity!;
  _pressure.value = weatherCity.pressure!;
```

Weather_client.dart

```dart
class WeatherClient {
  static const baseUrl = "https://api.openweathermap.org/data/2.5/weather";
  static const appid = "d5abe3d816a237c5f52019701508dd84";
  final dynamic id;
  WeatherClient(this.id);
  Future<Weather> getWeather() async {
    var uri = Uri.parse(baseUrl)
        .resolveUri(Uri(queryParameters: {
      "id": id.toString(),
      "units": "metric",
      "appid": appid
    }));
    try {
      final response = await http.get(uri);
      if (response.statusCode == 200) {
        var jsonResponse = convert.jsonDecode(response.body);
        dynamic id = jsonResponse['id'];
        String name = jsonResponse['name'];
        var weatherDescription = jsonResponse['weather'];
        String main = weatherDescription[0]['main'];
        var description = weatherDescription[0]['description'];
        double temp = jsonResponse['main']['temp'];

        double feelsLike = jsonResponse['main']['feels_like'];
        int humidity = jsonResponse['main']['humidity'];
        dynamic windSpeed = jsonResponse['wind']['speed'];
        int pressure = jsonResponse['main']['pressure'];

        Weather weather = Weather(id: id, name: name, description: descripti
        return weather;
      } else {
        return Future.error([]);
```

weather_db.dart

```dart
Future<Database> _openDB() async{
  return openDatabase(
    join(await getDatabasesPath(), 'weather_database.db'),
    onCreate: (db, version) async {
      // return await db.transaction((txn) async {
      //       await txn.execute('''
      //       CREATE TABLE weather_city(id INT PRIMARY KEY, name TEXT)
      //       ''');
      // });
      return db.execute(
        "CREATE TABLE weather_city(id INT PRIMARY KEY, name TEXT)",
      );
    },
    version:1,
  );
}

Future<void> insertWeatherCity(Weather weatherCity) async {
  final db = await instance.database;
  await db.insert(
    'weather_city',
    weatherCity.toMap(),
    conflictAlgorithm: ConflictAlgorithm.replace,
  );
}
```

```dart
Future<List<String>> weatherCity() async {
  final db = await instance.database;
  final List<Map<String, dynamic>> maps = await db.query('weather_city');

  return List.generate(maps.length, (i) {
    return  maps[i]['name'];}
  );
}

Future<void> updateWeatherCity(Weather weatherCity) async {
  final db = await instance.database;
  await db.update(
    'weather_city',
    weatherCity.toMap(),
    where: "id = ?",
    whereArgs: [weatherCity.id],
  );
}

Future<void> deleteWeatherCity(int id) async {
  final db = await instance.database;
  await db.delete(
    'weather_city',
    where: "id = ?",
    whereArgs: [id],
  );
}

Future close() async {
```