図6.2 **スタート画面**

画像選択画面

スマートフォンの画像ライブラリから画像を選択する画面です(**図6.3**)。「画像を選ぶ」ボタンをタップすると画像ライブラリから画像を選択でき、選択した画像はプレビューされます。画像ライブラリへのアクセスはimage_pickerというパッケージを使用します。また、画像のプレビューのためにimageパッケージを使用します。「画像を編集する」ボタンをタップすると画像編集画面に遷移します。

図6.3 **画像選択画面**

画像編集画面

選択した画像を回転、反転させる画面です（**図6.4**）。画像の回転は90度単位で行います。画像の回転、反転にはimageパッケージを使用します。

図6.4 **画像編集画面**

開発の土台づくり

第4章で解説した「開発の土台づくり」の要素です。アプリは日本語にローカライズし、アセットはflutter_genパッケージを使用して管理します。環境変数は特に設定しません。

導入するパッケージは**表6.1**のとおりです。

表6.1　**導入するパッケージ一覧**

パッケージ名	用途
intl	アプリのローカライズ
image_picker	画像ライブラリへのアクセス
image	画像データの加工
flutter_svg	SVG画像の表示
build_runner	flutter_genのコード生成
flutter_gen_runner	アセットの管理

テーマと画面遷移の方針

テーマはMaterial Design 3のテーマを使用します。テーマの変更は`ColorScheme`クラスの`seedColor`のみを変更するにとどめます。

画面遷移はNavigator 1.0のAPIのみを使用します。

6.2 プロジェクトを作成する

新たにプロジェクトを作成します。第1章の「1.1 プロジェクトの作成」で示した手順に従って、プロジェクトを作成してください。プロジェクト名は「edit_snap」としましょう。プロジェクト作成直後は`lib/main.dart`にテンプレートになるアプリコードが書かれていますので、不要なコードを削除してしまいましょう。

./lib/main.dart
```
import 'package:flutter/material.dart';

void main() {
```

```
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      // Androidのタスクマネージャーに表示されるアプリ名
      // iOSでは使用されません
      title: 'Edit Snap',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: Container(),
    );
  }
}
```

開発の土台づくり

本来であれば、開発をスタートするこの段階でアプリのローカライズやアセット管理のしくみを整えておくことが望ましいです。しかし、本章では各種しくみの導入前後の違いを比較しやすいよう、後工程で導入することにします。

6.3 アプリ起動後のスタート画面を作成する

アプリ起動後に表示されるスタート画面を作成します。スタート画面のコードを記述するファイルを作成します。Android Studioの左側にあるProjectビューで`lib`フォルダを右クリックし、「New」➡「Dart File」を選択します。ファイル名を入力するダイアログが表示されるので、「start_screen」と入力して「OK」ボタンをクリックします。

ひとまず、スタート画面は中央に文字を表示するだけのシンプルなものにしておきます。

./lib/start_screen.dart

```
import 'package:flutter/material.dart';

class StartScreen extends StatelessWidget {
  const StartScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: Text('Start Screen'),
      ),
    );
  }
}
```

続いて、アプリ起動後にスタート画面が表示されるようにmain.dartを修正します。

./lib/main.dart

```
import 'package:edit_snap/start_screen.dart'; ❶
import 'package:flutter/material.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      // Androidのタスクマネージャーに表示されるアプリ名
      // iOSでは使用されません
      title: 'Edit Snap',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const StartScreen(), ❷
    );
  }
}
```

先ほど実装したStartScreenウィジェットを参照するため、start_screen.

dartをインポートしました(❶)。MaterialAppウィジェットのhomeパラメータにStartScreen画面を渡しました(❷)。ここでアプリを実行すると、**図6.5**のようにスタート画面が表示されます。

図6.5 **スタート画面**

レイアウトを作成する

スタート画面のレイアウトを作成していきます。Android Studioでstart_screen.dartを開き、StartScreenクラスのbuildメソッドを以下のように修正します。

```
./lib/start_screen.dart
import 'package:flutter/material.dart';

class StartScreen extends StatelessWidget {
  const StartScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: const Text('Edit Snap'),
      ),
      body: Center(
```

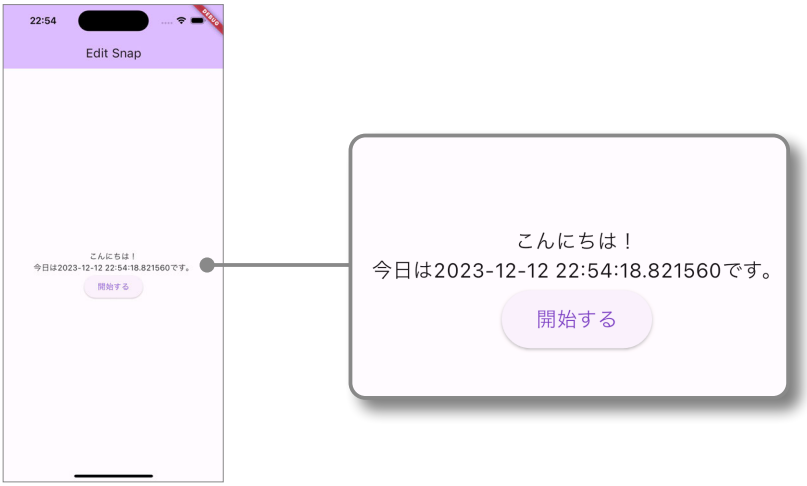
❶

```
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(
              'こんにちは！\n今日は${DateTime.now()}です。',
              textAlign: TextAlign.center,
            ),
            ElevatedButton(
              child: const Text('開始する'),
              onPressed: () {},
            ),
          ],
        ),
      ),
    );
  }
}
```

❷
❸

上部にAppBarウィジェット(❶)があり、Textウィジェット(❷)とElevatedButtonウィジェット(❸)が垂直方向に並ぶ画面です。❷と❸を垂直方向に並べるためにColumnウィジェットで囲みました。これを実行すると、iOS Simulator上で**図6.6**のように表示されます。

図6.6 **レイアウト作成後のスタート画面**



6.4 テーマをアレンジする

アプリのテーマを好みにアレンジしてみましょう。MaterialAppウィジェットに渡すthemeパラメータを変更します。

./lib/main.dart

```
// 省略
class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Edit Snap',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.teal), ❶
        useMaterial3: true,
      ),
      home: const StartScreen(),
    );
  }
}
```

ColorSchemeクラスのコンストラクタに渡すseedColorパラメータを変更しました(❶)。再びアプリを実行するとテーマが変更され、AppBarウィジェットやElevatedButtonウィジェットの色が変わります。

6.5 アプリを日本語化する

アプリ内で表示する文字列を日本語化し、arbファイルでメッセージを管理します。各コードのより詳しい説明は第4章を参照してください。

パッケージを導入する

まずはパッケージを2つ(flutter_localizationsとintl)を導入します。プロジェクトのディレクトリで、ターミナルから以下のコマンドを実行してください。