```
# flutter_localizationsパッケージを導入
$ flutter pub add flutter_localizations --sdk=flutter
# intlパッケージを導入
$ flutter pub add intl:any
```

次にpubspec.yamlを開き、以下のように追記します。

./pubspec.yaml

```
# 省略

flutter:
  uses-material-design: true
  generate: true ——❶
```

コードジェネレータの設定のためflutterキーの下にgenerate: trueを追加しました(❶)。生成されたコードをプロジェクトから参照できるように以下のコマンドを実行します。

```
$ flutter pub get
```

ローカライズの構成ファイルを作成する

次にローカライズの構成ファイルを作成します。左側のProjectビューでプロジェクトルートを右クリックし、「New」➡「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、「l10n.yaml」と入力して「OK」ボタンをクリックします。

構成ファイルは以下のように記述します。各パラメータの詳細は第4章を参照してください。

./l10n.yaml

```
template-arb-file: app_ja.arb
output-class: L10n
nullable-getter: false
```

arbファイルを作成する

メッセージを管理するarbファイルを作成します。左側のProjectビューでlibフォルダを右クリックし、「New」➡「Directory」を選択します。フォルダ名を入力するダイアログが表示されるので、「l10n」と入力して「OK」ボタンを

クリックします。

次にl10nフォルダを右クリックし、「New」➡「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、「app_ja.arb」と入力して「OK」ボタンをクリックします。arbファイルは以下のように記述します。

./lib/l10n/app_ja.arb

```
{
  "startScreenTitle": "Edit Snap",
  "helloWorldOn": "こんにちは！\n今日は{date}です",
  "@helloWorldOn": {
    "placeholders": {
      "date": {
        "type": "DateTime",
        "format": "MEd"
      }
    }
  },
  "start": "開始する"
}
```

❶

スタート画面のメッセージには日付を埋め込みます。キーhelloWorldOnにプレースホルダを設定し、日付のフォーマットを指定しました(❶)。typeは日付型としてDateTime、formatはMEd(月、日、短縮系の曜日で構成されるフォーマット)です。

arbファイルを作成したら、以下のコマンドでコードジェネレータを実行します。

```
$ flutter gen-l10n
```

ローカライズされたメッセージを適用する

コードジェネレータが完了したら、生成されたメッセージをコードに反映させましょう。main.dartを以下のように修正します。

./lib/main.dart

```
import 'package:edit_snap/start_screen.dart';
import 'package:flutter/material.dart';
import 'package:flutter_gen/gen_l10n/app_localizations.dart'; ❶

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

```
}

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      localizationsDelegates: L10n.localizationsDelegates, ——❷
      supportedLocales: L10n.supportedLocales, ——❸
      title: 'Edit Snap',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.teal),
        useMaterial3: true,
      ),
      home: const StartScreen(),
    );
  }
}
```

生成されたコードをインポートし(❶)、対応言語(今回は日本語のみ)の翻訳データと対応言語のリストを渡します(❷、❸)。

start_screen.dartは以下のように修正します。

./lib/start_screen.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_gen/gen_l10n/app_localizations.dart'; ——❶

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

  @override
  Widget build(BuildContext context) {
    final l10n = L10n.of(context); ——❷
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(l10n.startScreenTitle), ——❸
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(
              l10n.helloWorldOn(DateTime.now()), ——❹
              textAlign: TextAlign.center,
            ),
```

```
            ElevatedButton( // 「開始する」ボタン
              child: Text(l10n.start), ——❺
              onPressed: () {},
            ),
          ],
        ),
      ),
    );
  }
}
```

こちらも生成コードをインポートし(❶)、各メッセージを生成コードから取得します(❷、❸、❹、❺)。

App Storeでの表示言語を設定する

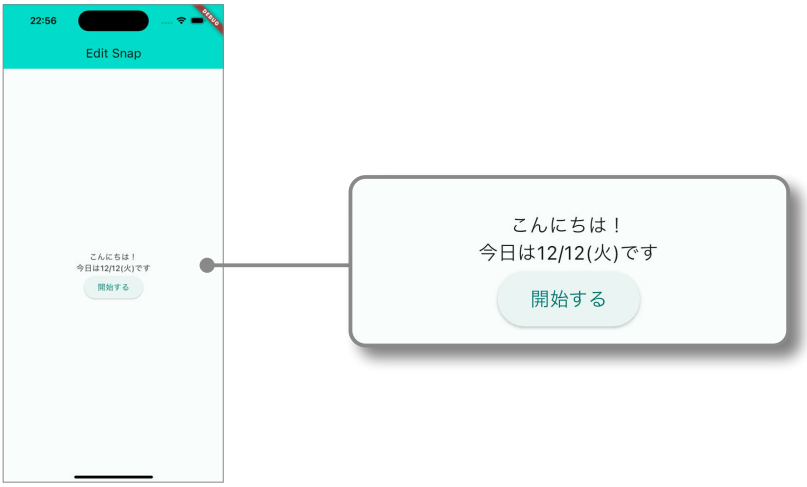
最後にiOSネイティブの対応言語を設定します。これはApp Storeに表示されるアプリの対応言語に影響します。`ios/Runner/Info.plist`を開き、`CFBundleLocalizations`キーの下に`ja`を追加します。

./ios/Runner/Info.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <!-- 省略 -->
    <key>CFBundleLocalizations</key>
    <array>
        <string>ja</string>
    </array>
</dict>
</plist>
```

これで日本語化が完了しました。アプリを実行してみましょう。**図6.7**のように日付のフォーマットが日本語になっていることが確認できます。

図6.7　日本語化したスタート画面



6.6 画像選択画面を作成する

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を作成します。

メッセージを追加する

画像選択画面で使用するメッセージをarbファイルに追加します。

./lib/l10n/app_ja.arb

```
{
  // 省略
  // （一つ上の行の末尾にカンマを追加してください）
  "imageSelectScreenTitle": "画像を選択",
  "imageSelect": "画像を選ぶ",
  "imageEdit": "画像を編集する"
}
```

arbファイルに追加したらコードジェネレータを実行します。

```
$ flutter gen-l10n
```

レイアウトを作成する

次に画像選択画面のコードを記述するファイルを作成します。左側の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、「image_select_screen」と入力して「OK」ボタンをクリックします。

画像選択画面はStatefulWidgetを継承したクラスで実装します。

./lib/image_select_screen.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_gen/gen_l10n/app_localizations.dart';

class ImageSelectScreen extends StatefulWidget {
  const ImageSelectScreen({super.key});

  @override
  State<ImageSelectScreen> createState() => _ImageSelectScreenState();
}

class _ImageSelectScreenState extends State<ImageSelectScreen> {

  @override
  Widget build(BuildContext context) {
    final l10n = L10n.of(context);
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(l10n.imageSelectScreenTitle),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            ElevatedButton( //  「画像を選ぶ」ボタン
              onPressed: () {},
              child: Text(l10n.imageSelect),
            ),
            ElevatedButton( //  「画像を編集する」ボタン
              onPressed: () {
              },
              child: Text(l10n.imageEdit),
            ),
          ],
        ),
      ),
```

```
    );
  }
}
```

レイアウトはスタート画面とおおむね同じです。上部にAppBarウィジェットがあり、「画像を選ぶ」ボタンと「画像を編集する」ボタンが垂直方向に並びます。

続いて、スタート画面から画像選択画面に遷移する処理を実装します。

./lib/start_screen.dart

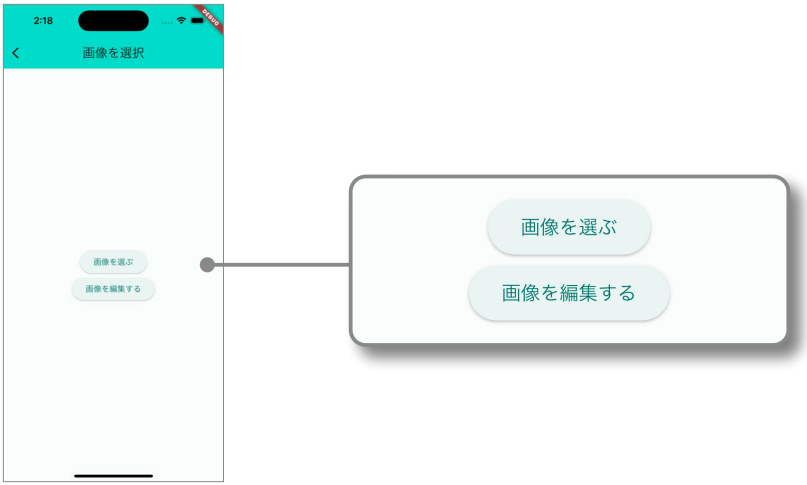
```
import 'package:edit_snap/image_select_screen.dart'; ❶
import 'package:flutter/material.dart';
import 'package:flutter_gen/gen_l10n/app_localizations.dart';

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

  @override
  Widget build(BuildContext context) {
    final l10n = L10n.of(context);
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
        title: Text(l10n.startScreenTitle),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text(
              l10n.helloWorldOn(DateTime.now()),
              textAlign: TextAlign.center,
            ),
            ElevatedButton(  // 「開始する」ボタン
              child: Text(l10n.start),
              onPressed: () {
                Navigator.of(context).push(
                  MaterialPageRoute(
                    builder: (context) => const ImageSelectScreen(), ❷
                  ),
                );
              },
            ),
          ],
        ),
      ),
```

```
    );
  }
}
```

画像選択画面のクラスを参照するため`image_select_screen.dart`をインポートしました(❶)。「開始する」ボタンの`onPressed`コールバックで画像選択画面に遷移する処理を実装しました(❷)。今回はシンプルにNavigator 1.0のAPIを採用し、`MaterialPageRoute`を使用した画面遷移です。

これで画像選択画面に遷移する処理が実装できました。アプリを実行し、スタート画面の「開始する」ボタンをタップしてみましょう。**図6.8**のように画像選択画面に遷移します。

図6.8 **画像選択画面**



画像ライブラリから画像を取得する

スマートフォンの画像ライブラリから画像を取得する処理は、image_pickerパッケージを使用します。また画像データを取り扱うためにimageパッケージも使用します。

パッケージを導入する

パッケージを導入するために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。