



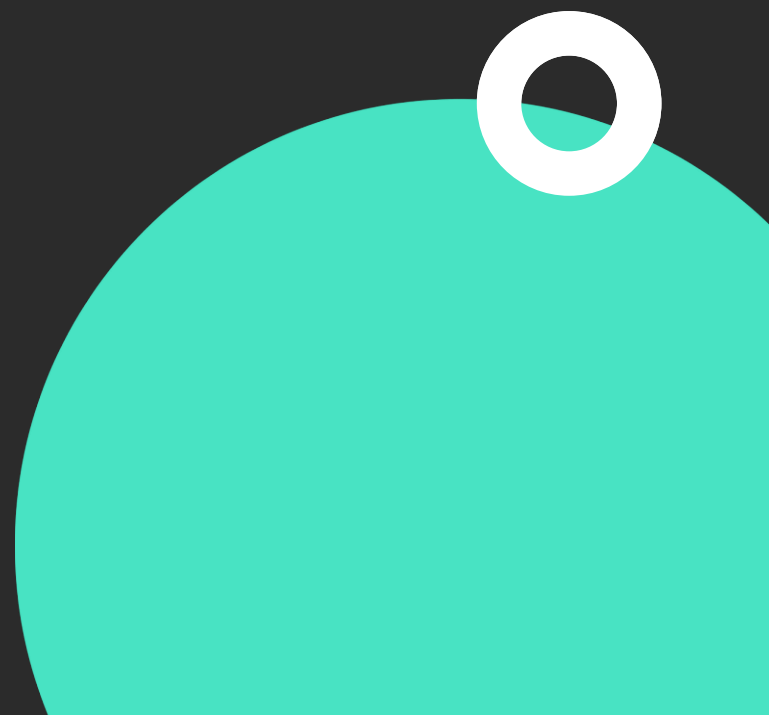
# Internationalization and Localization in Flutter Apps



Dheeraj Singh Bhadoria

 @bhadoriadheeru

 in/dheeraj-singh-bhadoria-android-developer



# Internationalization and Localization in Flutter

- Definition and importance of internationalization and localization in app development
- Benefits of creating a multilingual app
- Goals of the project

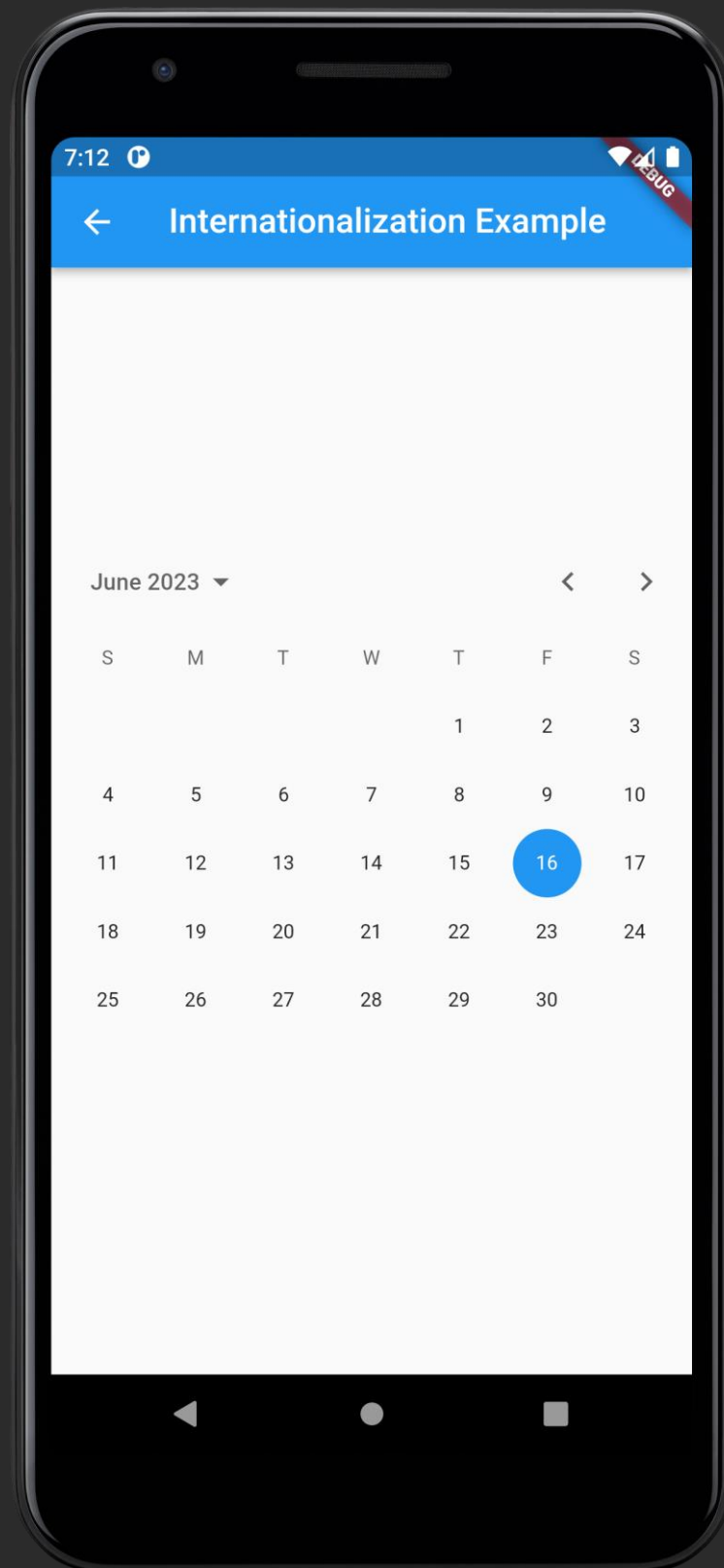


# Importance of a Multilingual App

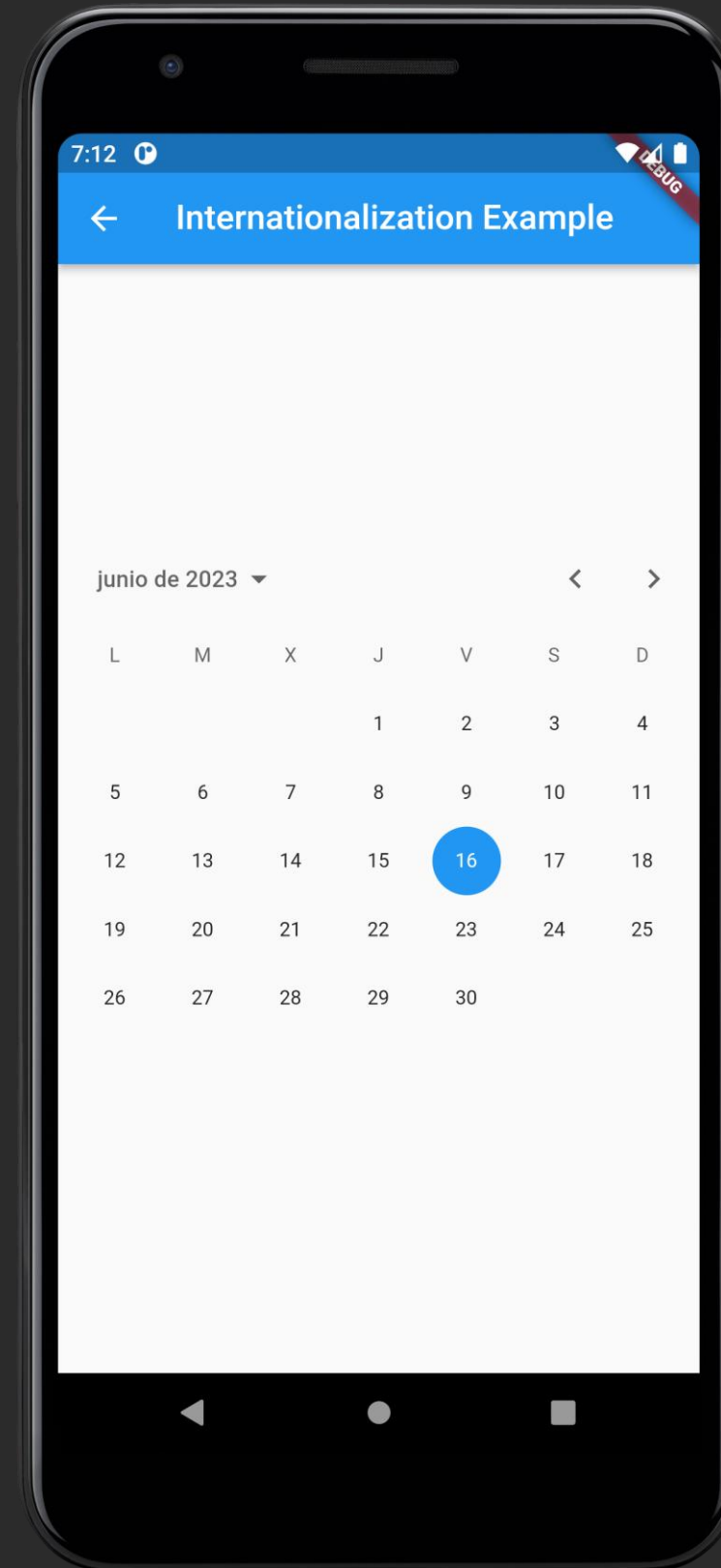
- Expanding the user base and reaching a global audience
- Enhancing user experience by providing content in their preferred language
- Boosting user engagement and increasing app adoption



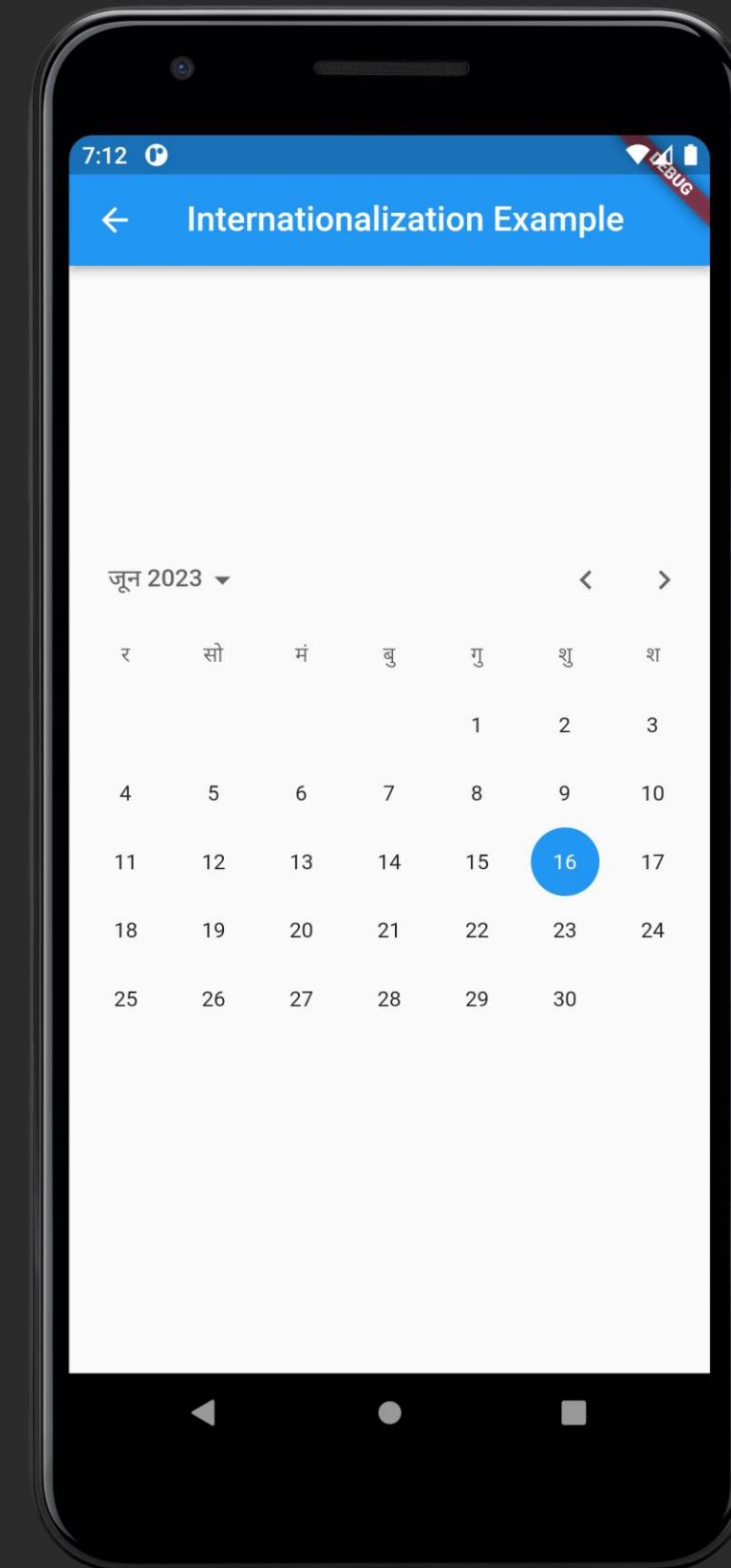
# Internationalization Screen - Overview



English



Spanish

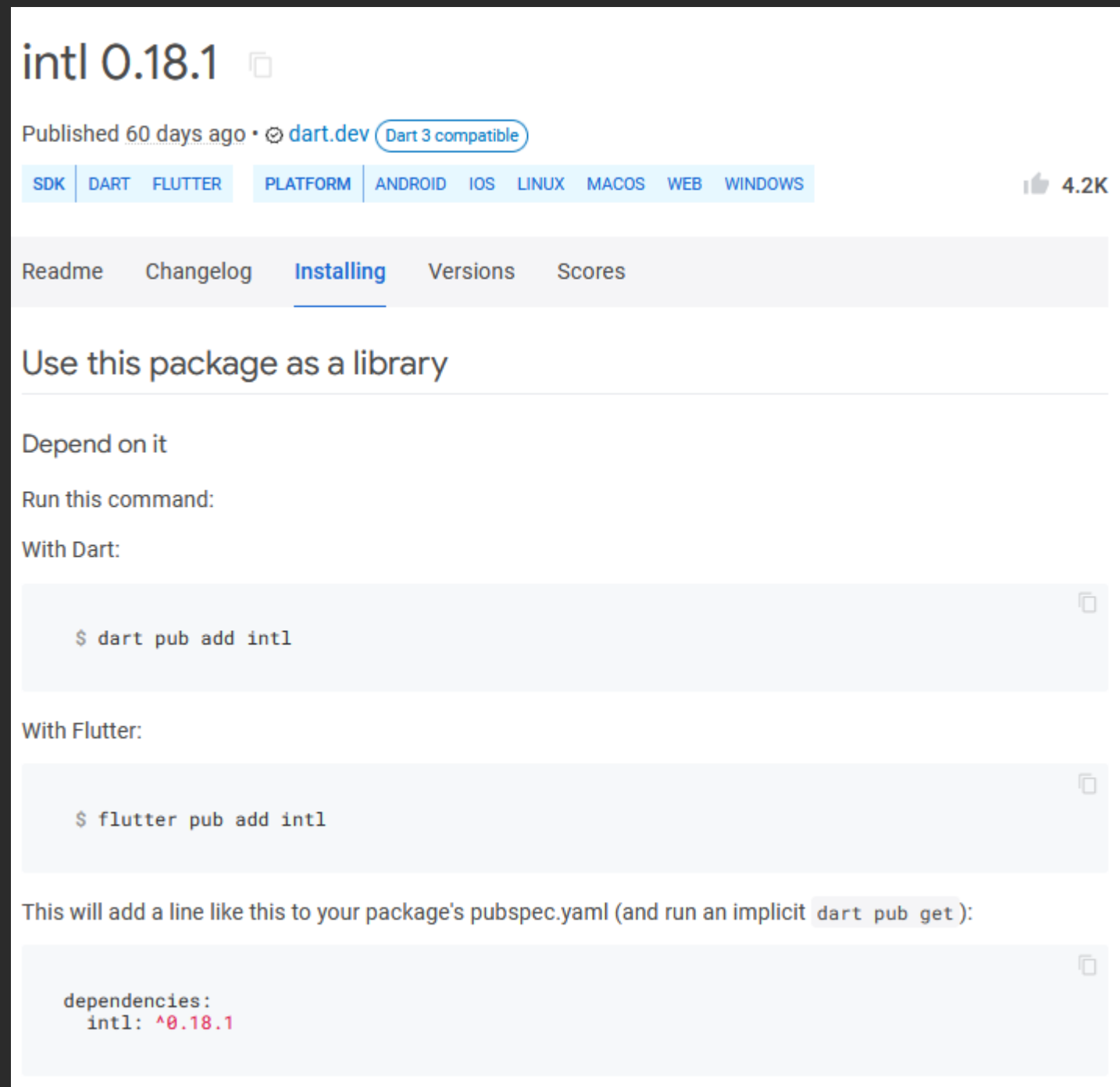


Hindi




# Implementing Internationalization in Flutter


- Step 1: Adding the necessary package to enable internationalization



The screenshot shows the package page for `intl` version `0.18.1` on the `pub.dev` website. The page includes a header with the package name and version, a 'Dart 3 compatible' badge, and a list of supported platforms: SDK, DART, FLUTTER, PLATFORM, ANDROID, IOS, LINUX, MACOS, WEB, and WINDOWS. The 'Installing' tab is selected, showing instructions on how to use the package as a library, how to depend on it, and the commands to run with Dart and Flutter. It also shows the resulting entry in the `pubspec.yaml` file.

intl 0.18.1 

Published 60 days ago • [dart.dev](#) Dart 3 compatible

[SDK](#) [DART](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [LINUX](#) [MACOS](#) [WEB](#) [WINDOWS](#)  4.2K

[Readme](#) [Changelog](#) [Installing](#) [Versions](#) [Scores](#)

Use this package as a library

Depend on it

Run this command:

With Dart:

```
$ dart pub add intl
```

With Flutter:

```
$ flutter pub add intl
```

This will add a line like this to your package's pubspec.yaml (and run an implicit `dart pub get`):

```
dependencies:  
  intl: ^0.18.1
```



# Implementing Internationalization in Flutter

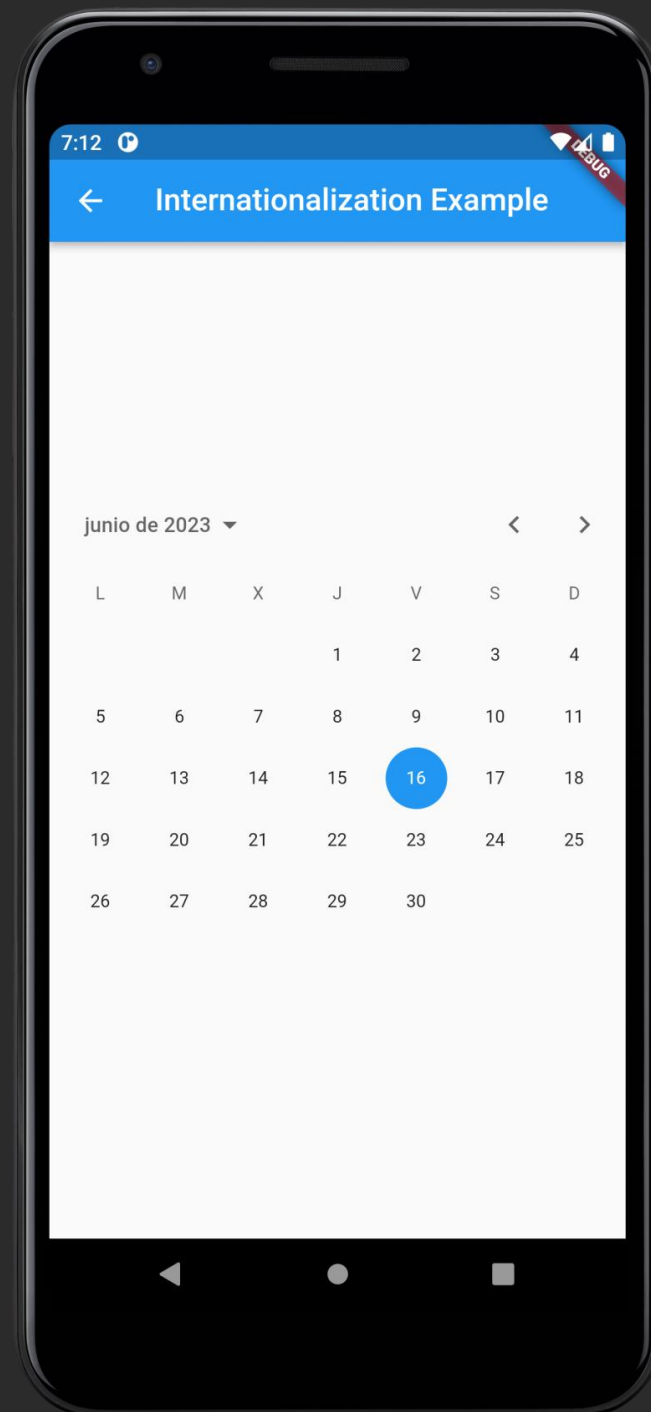
- Step 2: Implementing code changes to support multiple languages

```
Localizations.override(  
  context: context,  
  locale: const Locale('es'),  
  // Using a Builder to get the correct BuildContext.  
  // Alternatively, you can create a new widget and Localizations.override  
  // will pass the updated BuildContext to the new widget.  
  child: Builder(  
    builder: (context) {  
      // A toy example for an internationalized Material widget.  
      return CalendarDatePicker(  
        initialDate: DateTime.now(),  
        firstDate: DateTime(1900),  
        lastDate: DateTime(2100),  
        onChanged: (value) {},  
      );  
    },  
  ),  
)
```

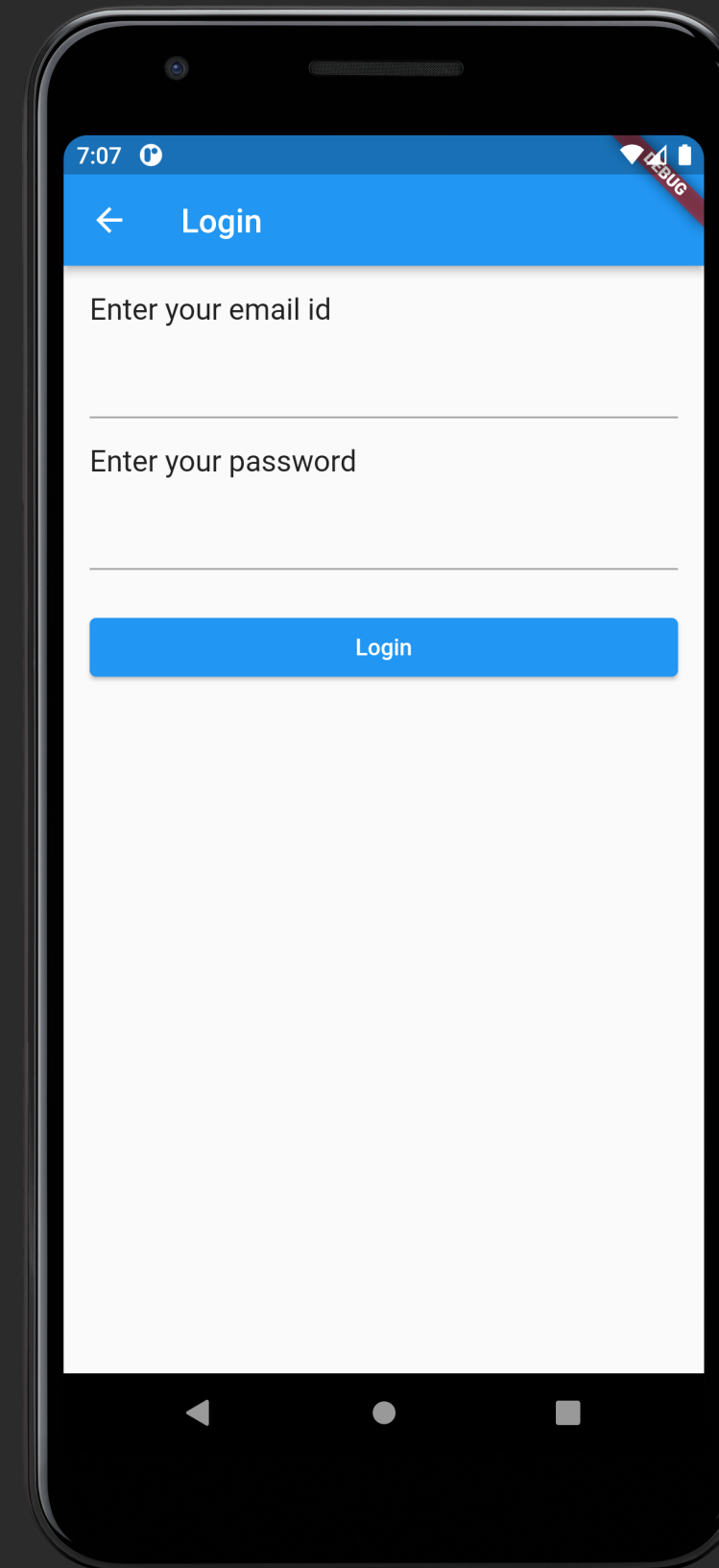


# Implementing Internationalization in Flutter

- Step 3: Showcasing the integration of a date picker widget with language support



# Localization Screen - Overview





# Implementing Localization in Flutter

- Step 1: Adding the necessary localization package at pubspec.yaml

```
dependencies:
  flutter:
    sdk: flutter

  flutter_localizations:
    sdk: flutter
  intl: any

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2

dev_dependencies:
  flutter_test:
    sdk: flutter

  # The "flutter_lints" package below contains a set of recommended lints to
  # encourage good coding practices. The lint set provided by the package is
  # activated in the `analysis_options.yaml` file located at the root of your
  # package. See that file for information about deactivating specific lint
  # rules and activating additional ones.
  flutter_lints: ^2.0.0

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec

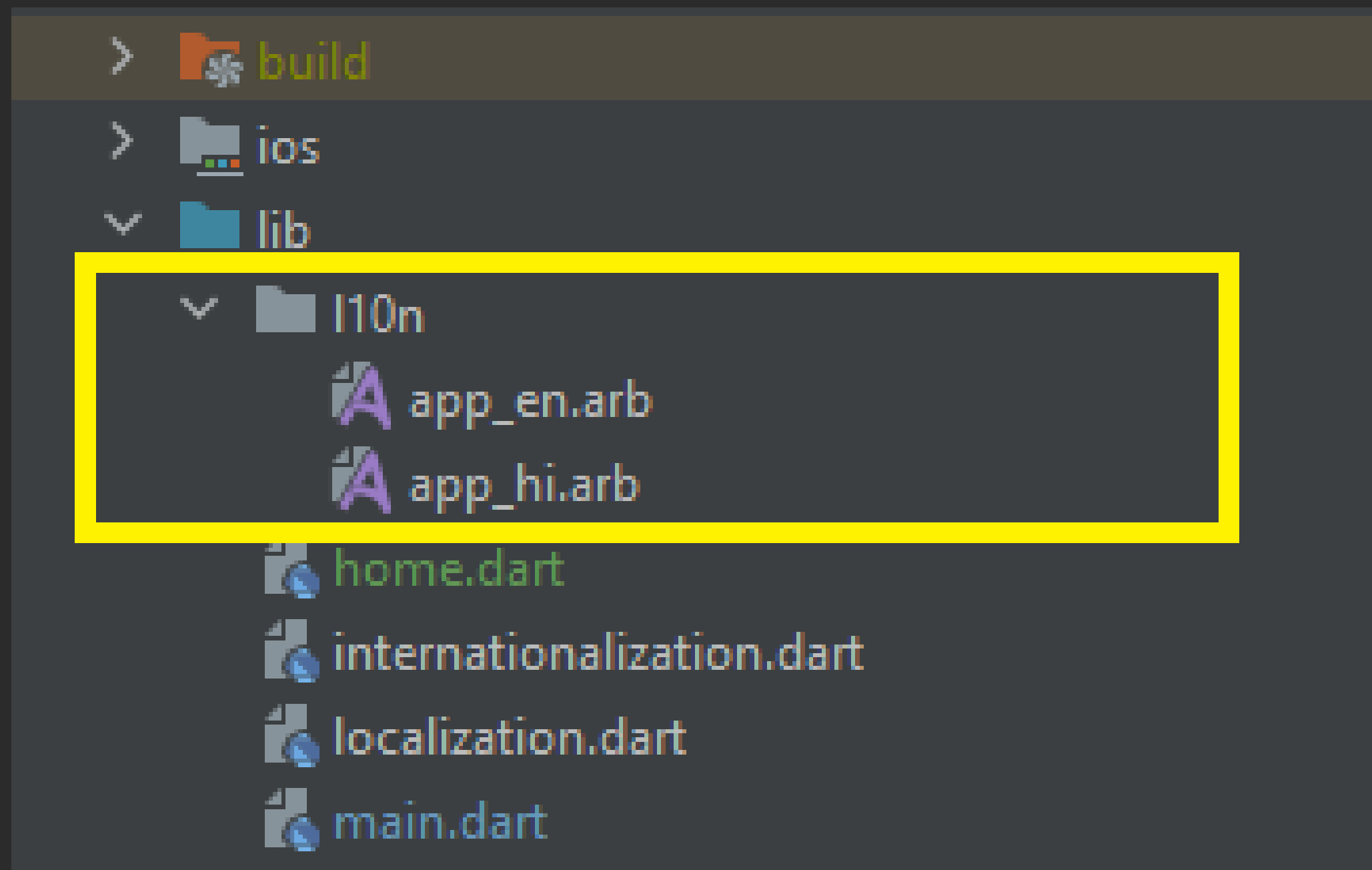
# The following section is specific to Flutter packages.
flutter:

  generate: true # Add thi line for localization
  # The following line ensures that the Material Icons font is
  # included with your application, so that you can use the icons in
  # the material Icons class.
  uses-material-design: true
```



# Implementing Localization in Flutter

- Step 2: Creating the l10n folder structure and files (display a screenshot)



# Implementing Localization in Flutter

- Step 3: Adding resources files for different languages (display files)

```
app_en.arb x
1  {
2    "helloWorld": "Hello World",
3    "loginScreenTitle": "Login",
4    "enterEmail": "Enter your email id",
5    "enterPassword": "Enter your password",
6    "loginButton": "Login"
7  }
```

```
app_hi.arb x
1  {
2    "helloWorld": "हैलो वर्ल्ड",
3    "loginScreenTitle": "लॉगिन करें ",
4    "enterEmail": "अपना ईमेल आईडी टाइप करें",
5    "enterPassword": "अपना पासवर्ड टाइप करें ",
6    "loginButton": "लॉगिन "
7  }
```



# Implementing Localization in Flutter

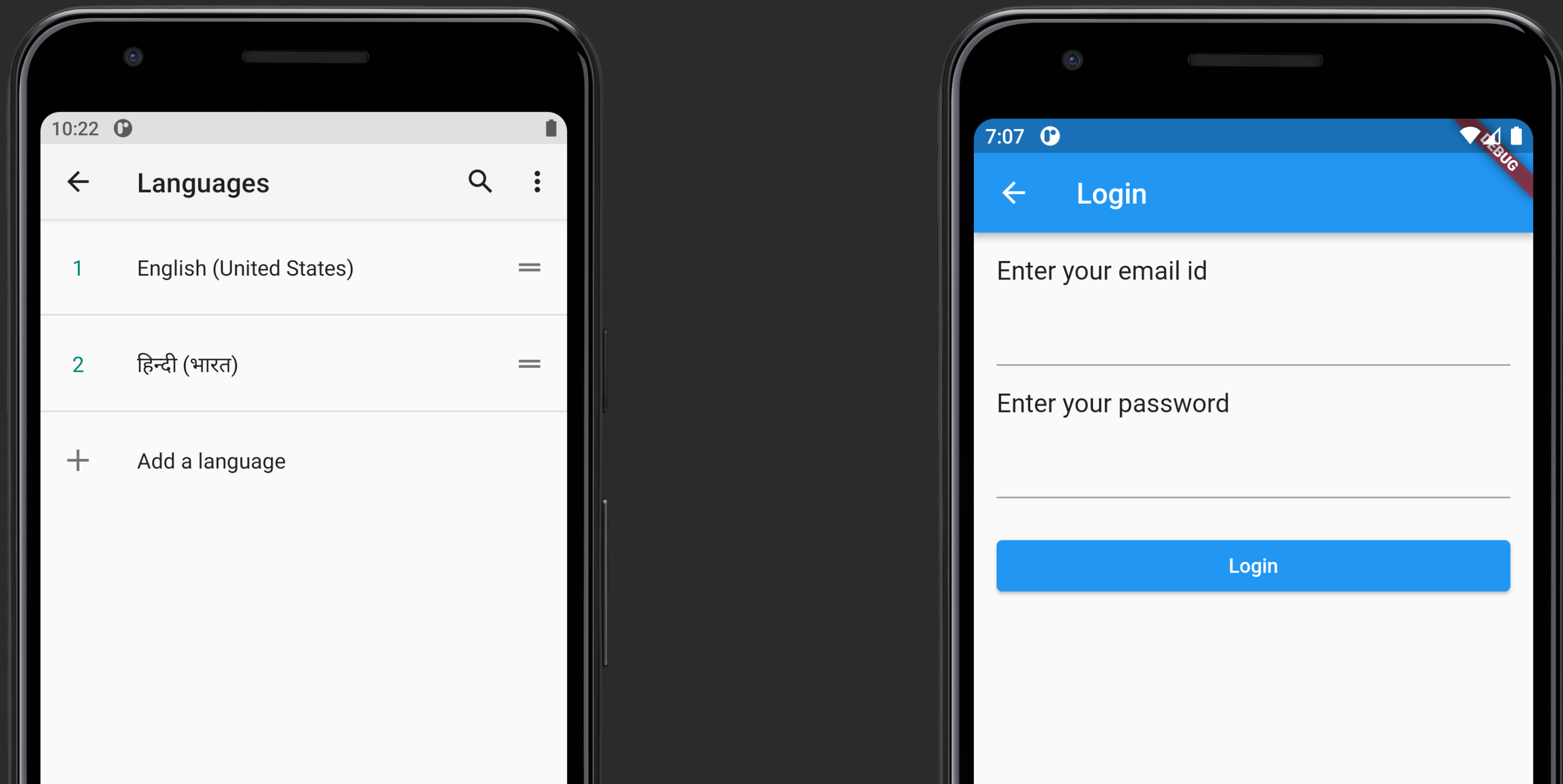
- Step 4: Displaying text from resources files in the app (display files)

```
appBar: AppBar(  
  title: Text(AppLocalizations.of(context)!.loginScreenTitle),  
) , // AppBar  
body: Padding(  
  padding: EdgeInsets.all(16.0),  
  child: Column(  
    crossAxisAlignment: CrossAxisAlignment.stretch,  
    children: [  
      Text(  
        AppLocalizations.of(context)!.enterEmail,  
        style: TextStyle(fontSize: 18.0),  
      ) , // Text  
      SizedBox(height: 8.0),  
      TextFormField(  
        onChanged: (value) {  
          setState(() {  
            email = value;  
          });  
        },  
      ) , // TextFormField  
      SizedBox(height: 16.0),  
      Text(  
        AppLocalizations.of(context)!.enterPassword,  
        style: TextStyle(fontSize: 18.0),  
      ) , // Text  
      SizedBox(height: 8.0),  
      TextFormField(  
        obscureText: true,  
        onChanged: (value) {  
          setState(() {  
            password = value;  
          });  
        },  
      ) , // TextFormField  
    ],  
  ),  
) , // Column
```



# Implementing Localization in Flutter

- Step 5: Demonstrating how changing the device language affects the app's text



# Implementing Localization in Flutter

- Step 5: Demonstrating how changing the device language affects the app's text

