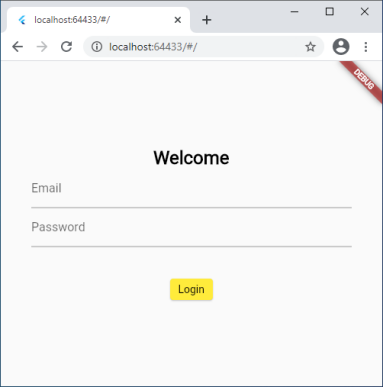
# Login



# 0 配置主题

theme.dart

final appTheme = ThemeData(

  primarySwatch: Colors.yellow,

  textTheme: TextTheme(

    headline1: TextStyle(

      fontFamily: 'Corben',

      fontWeight: FontWeight.w700,

      fontSize: 24,

      color: Colors.black,

    ),

  ),

);

调用：main.dart

child: MaterialApp(

        title: '',

        theme: appTheme,

# 1 建立MVVM provider

1.0 配置

  provider: '>=5.0.0-nullsafety.5 <6.0.0'

1.1 changetNotifier

DataModel.dart

所有数据在这里存贮

class DataModel extends ChangeNotifier {

1.2 初始化main.dart

Widget build(BuildContext context) {

    return MultiProvider(

      providers: [

        ChangeNotifierProvider<DataModel>(

          create: (context) => DataModel(),

        ),

      ],

      child: MaterialApp(

1.3 widgets用consumer包裹

main.dart根据token，email状态来决定走到哪个页面

return Consumer<DataModel>(builder: (context, datamodel, child) {

      if (datamodel.token == null) {

        //login

        //return login.logInFunc();

        return new MyForm("login");

      } else if (datamodel.formLists["login"]["items"][0]["email"]

              ["defaultValue"] ==

          "") {

        //reset password

        return Text('reset password');

      }

      return new MyMain();

# 2 国际化 参考国际化

# 3 构造Form组件

根据dataModel中的配置决定Form的样式

如login form

3.1 dataModel.dart

form定义放在\_formLists中

 Map<String, Map<String, dynamic>> \_formLists = {

    "login": {

      "title": S.current.welcome,

      "submit": S.current.login,

      "screenHeight": 120,

      "items": [

        {

          "email": {

            "hint": "xxx@xxxxx.xxx",

            "id": "email",

            "isrequired": true,

            "label": S.current.email,

            "minLength": 2,

            "type": "email",

            "unit": S.current.character,

            "defaultValue": "",

            "newValue": ""

          }

        },

        {

          "password": {

            "hint": "password",

            "id": "password",

            "isrequired": true,

            "label": S.current.password,

            "minLength": 8,

            "type": "password",

            "unit": S.current.character,

            "defaultValue": "",

            "newValue": ""

          }

        },

      ]

    }

  };

3.2 组件MyForm

调用：new MyForm(‘login’);

MyForm.dart内容,根据login明细内容，生成listview, 并添加submit按钮，调用统一saveForm方法

cclass MyForm extends StatelessWidget {

  final String \_formName;

  MyForm(this.\_formName);

  @override

  Widget build(BuildContext context) {

    return Consumer<DataModel>(

      builder: (context, datamodel, child) {

        final GlobalKey<FormState> \_formKey = GlobalKey<FormState>();

        Map<String, dynamic> formDefine = datamodel.formLists[\_formName];

        List<Map<String, Map<String, Object>>> **items** = formDefine['items'];

        return Scaffold(

          body: Container(

            padding: EdgeInsets.all(40.0),

            child: Form(

              key: \_formKey,

              child: Column(

                mainAxisAlignment: MainAxisAlignment.center,

                children: [

                  Text(

                    formDefine['title'],

                    style: Theme.of(context).textTheme.headline1,

                  ),

                  SizedBox(

                    height: formDefine['screenHeight'],

                    child: ListView.builder(

                      itemCount: items.length,

                      itemBuilder: (context, i) => TextFormField(

                        decoration: InputDecoration(

                          labelText: items[i].entries.first.value['label'],

                          hintText: items[i].entries.first.value['hint'],

                        ),

                        obscureText: (items[i].entries.first.value['type'] ==

                            "password"),

                        validator: (String value) {

                          if (items[i].entries.first.value['isrequired'] &&

                              value.isEmpty) {

                            return S.of(context).isrequired(

                                items[i].entries.first.value['label']);

                          }

                          if (items[i].entries.first.value['type'] == "email" &&

                              !RegExp(r"^[a-zA-Z0-9.a-zA-Z0-9.!#$%&'\*+-/=?^\_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+")

                                  .hasMatch(value)) {

                            return S

                                .of(context)

                                .invalidname(S.of(context).email);

                          }

                          if (items[i].entries.first.value["minLength"] !=

                                  '0' &&

                              value.length <

                                  items[i].entries.first.value["minLength"]) {

                            return S.of(context).mininput(

                                items[i].entries.first.value["minLength"],

                                items[i].entries.first.value["unit"]);

                          }

                          return null;

                        },

                        initialValue:

                            items[i].entries.first.value["defaultValue"],

                        onSaved: (String value) {

                          items[i].entries.first.value["newValue"] = value;

                        },

                      ),

                    ),

                  ),

                  SizedBox(

                    height: 24,

                  ),

                  ElevatedButton(

                    child: Text(formDefine['submit']),

                    onPressed: () {

                      if (!\_formKey.currentState.validate()) {

                        return;

                      }

                      \_formKey.currentState.save();

                      datamodel.saveForm(\_formName);

                    },

                    style: ElevatedButton.styleFrom(

                      primary: Colors.yellow,

                    ),

                  ),

                ],

              ),

            ),

            //),

          ),

        );

      },

    );

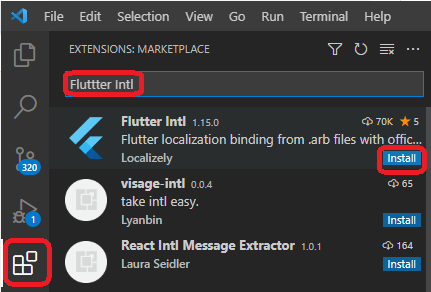
  }

}

# Appendix

## 国际化

1 IDE中安装Fluttere intl插件



2 pubspec.yaml

dependencies:

  flutter:

    sdk: flutter

  # Import the provider package.

  provider: '>=5.0.0-nullsafety.5 <6.0.0'

  # The following adds the Cupertino Icons font to your application.

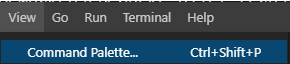
  # Use with the CupertinoIcons class for iOS style icons.

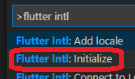
  cupertino\_icons: ^1.0.2

**flutter\_localizations:**

**sdk: flutter**

3 初始化





初始化执行成功后，会自动在 pubspec.yaml 末尾增加：

**CODE**

|  |  |
| --- | --- |
| 1 2 | flutter\_intl:  enabled: true |

lib下生成generated和i10n文件夹

4 添加语言

View->Command Palette...->输入fluter intl->选择Flutter Intl: Add Locale=>输入：zh

generated下多了一个messages\_zh.dart

i10n下多了一个intl\_zh.arb

5 修改MaterialApp

 child: MaterialApp(

        title: '',

        theme: appTheme,

        home: MyHomePage(title: 'Flutter Demo Home Page'),

        localizationsDelegates: [

          S.delegate,

          GlobalMaterialLocalizations.delegate,

          GlobalWidgetsLocalizations.delegate,

          GlobalCupertinoLocalizations.delegate,

        ],

        // 将 en设置为第一项, 没有适配语言时, 英语为首选项

        supportedLocales: [

          const Locale('en', ''),

          ...S.delegate.supportedLocales

        ],

        // 插件目前不完善手动处理简繁体

        localeResolutionCallback: (locale, supportLocales) {

          // 中文 简繁体处理

          if (locale?.languageCode == 'zh') {

            // zh-CN：地区限制匹配规范，表示用在中国大陆区域的中文。

            // 包括各种大方言、小方言、繁体、简体等等都可以被匹配到。

            if (locale?.scriptCode == 'Hant') {

              // zh-Hant和zh-CHT相同相对应;

              return const Locale('zh', 'HK'); //繁体

            } else {

              // zh-Hans：语言限制匹配规范，表示简体中文

              return const Locale('zh', 'CN'); //简体

            }

          }

          return null;

        },

      ),

6 添加内容：

intl\_en.arb

{"password":"Password"}

intl\_zh.arb

{"password":"密码"}

7 使用：

普通使用

**CODE**

|  |  |
| --- | --- |
| 1 2 | {"content": "内容"} S.of(context).content |

占位

**CODE**

|  |  |
| --- | --- |
| 1 2 | {"welcome": "welcome {name}"} S.of(context).welcome("phoenixsky") |

重复占位

**CODE**

|  |  |
| --- | --- |
| 1 2 | {"goodStudy": "{good}{good}学习,{day}{day}向上"} S.of(context).goodStudy("好", "天") |

复数形式

**CODE**

|  |  |
| --- | --- |
| 1 2 | {"getMessageTips": "{howMany, plural, zero{You have no message} one{You have 1 message} other{You have {howMany} messages}}"} S.of(context).getMessageTips(2) |

8 选择语言



8.1组件

getLocalComponents() {

    return [

      ElevatedButton(

        child: Text('中文'),

        onPressed: () {

          setLocale(Locale('zh'));

        },

      ),

      ElevatedButton(

        child: Text('English'),

        onPressed: () {

          setLocale(Locale('en'));

        },

      )

    ];

  }

8.2

Locale \_locale = const Locale('en', '');

  Locale get locale => \_locale;

setLocale(value) {

    \_locale = value;

    S.load(\_locale);

    notifyListeners();

  }