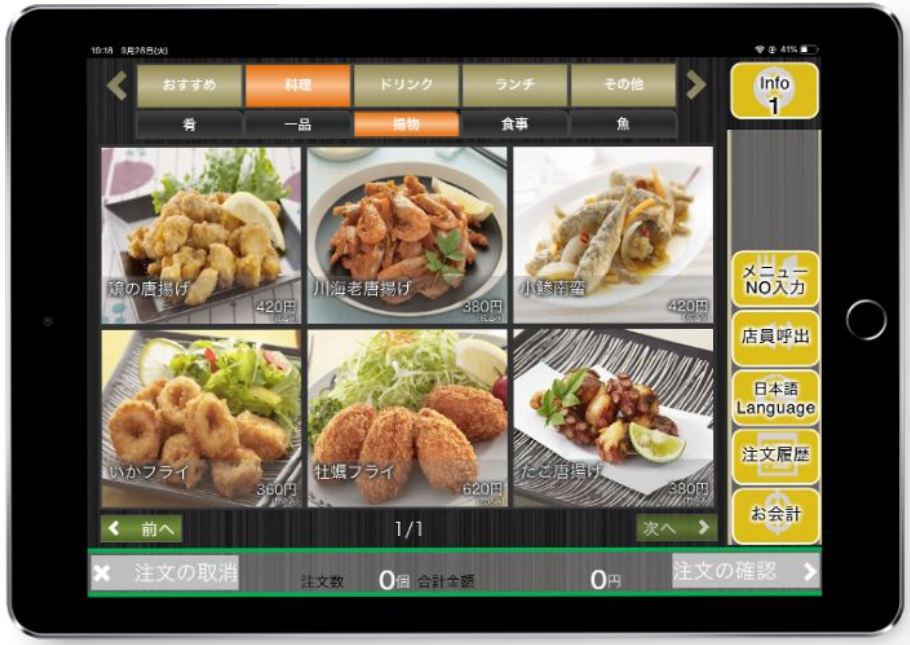
自主製作
タブレットメニュー

2024年2月28日　小林 司

■アジェンダ

1. なぜ作ろうと思ったのか
2. 使用した言語・ライブラリ
3. 機能要件
4. 構成
5. 反省点とこの経験から得た学び

1、なぜ作ろうと思ったのか



よくあるタブレットのメニューはこんな感じ

しかし私の近所にあるファミレスは…

◀️このメニューを見て

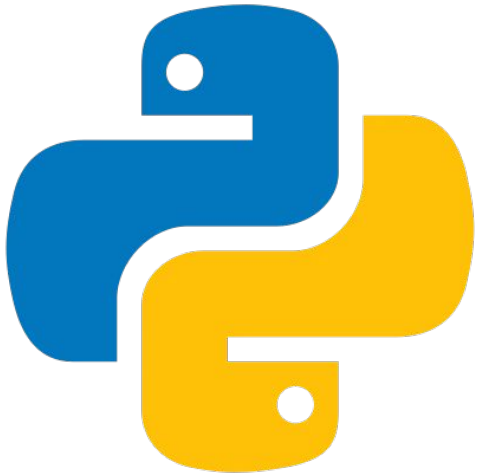
ここに番号を入力▶️

なぜタブレットと旧式メニュー表をハイブリッドで運用しているんだ…

▶️自分で作って
・タブレットで完結させることの大変さ
・メニューが変わる際の修正工数

検証してみる

2、使用した言語・ライブラリ



言語：Python

ライブラリ：Kivy

■Kivyとは

Kivy(キビー)は、PythonのNUI(Natural User Interface)でのマルチタッチアプリケーション開発のためのオープンソースライブラリである。動作環境はAndroid、iOS、Linux、Mac OS X、Windowsである。MITライセンスで配布されているフリーのオープンソースソフトウェアである。Kivyは様々なプラットフォームで動作するように開発が進められており、Raspberry Piもサポートされた。

特徴としては以下のとおりである。

- マウス、キーボード、TUIO、OS固有のマルチタッチイベントなど広範な入力への対応
- OpenGL ES 2のVertext Buffer Objectとshadersを使用したグラフィックライブラリ
- マルチタッチ対応のさまざまなウィジェット
- カスタムウィジェットを容易にデザインするための中間言語Kv

Pythonを使って

クロスプラットフォームのアプリ開発ができる

ライブラリ

開発には**kvファイル**を使える

クロスプラットフォームについて

一つの開発言語で、

Windows、Mac、Android、iOSなど

複数のOSで稼働するプログラム

３、機能要件

- メニュー表示機能
  - 写真付きでメニューを表示する
  - カテゴリー別にメニューを絞り込む
- 注文機能
  - メニューを選択して注文する
  - 数量を変更する
  - 複数の注文をまとめて行う
  - 注文履歴を確認する
  - カートを確認する
  - 注文の時間を記録する
- その他機能
  - 人数を記録する
  - 店員を呼ぶボタンを実装する
  - 会計と共に、注文情報をデータベースへ記録する

3、機能要件

- customerテーブル
  - 顧客ID
  - 人数
- menuテーブル
  - 商品コード
  - 商品名
  - 金額
- orderテーブル
  - 顧客ID
  - 日時
  - 商品コード
  - 商品名
  - 数量
  - 金額

最終的にこのテーブルにデータを残す

４、構成

■Appクラス（アプリを実行するクラス）

- MainApp

■Buttonクラス（機能をカスタマイズしたボタン）

- MyButton

■BoxLayoutクラス（複数のボタンを持つレイアウト）

- FooterLayout

４、構成

■Screenクラス（表示する画面）

- TopScreen
- MainScreen
- DetailScreen
- CartScreen
- ConfirmScreen
- OrderLogScreen
- CallWaiterScreen
- CheckoutScreen

４、構成

■ScrollViewクラス（スクロールする画面）

- ScrollView
- OrderView

■Databaseクラス（データベースとの接続）

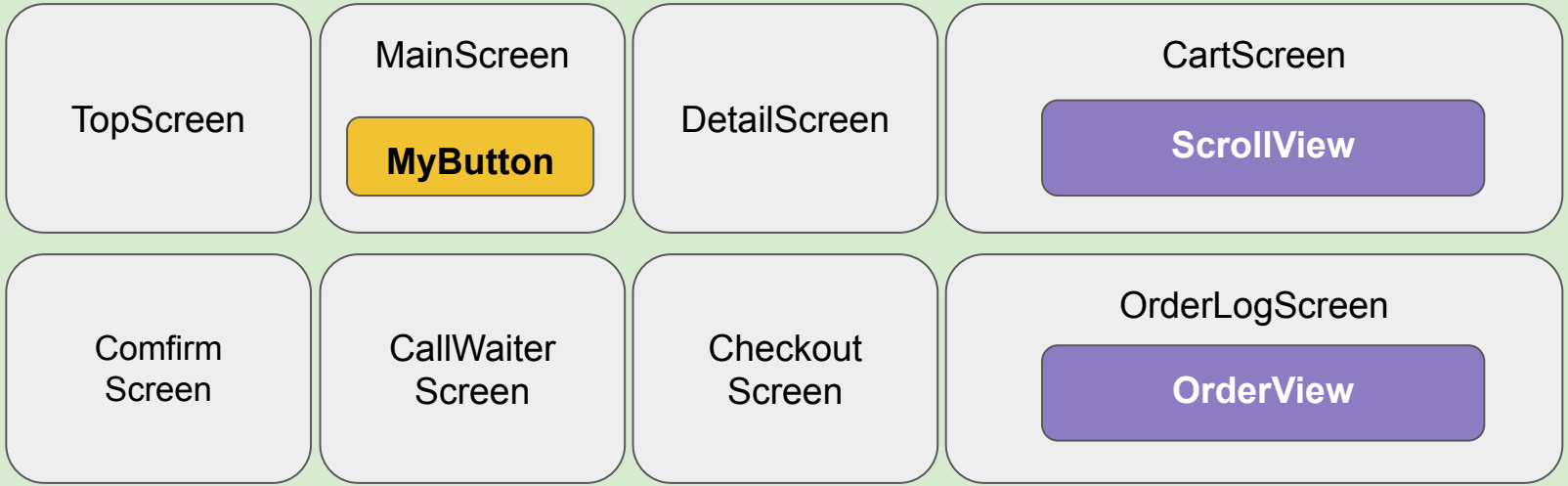
- Database

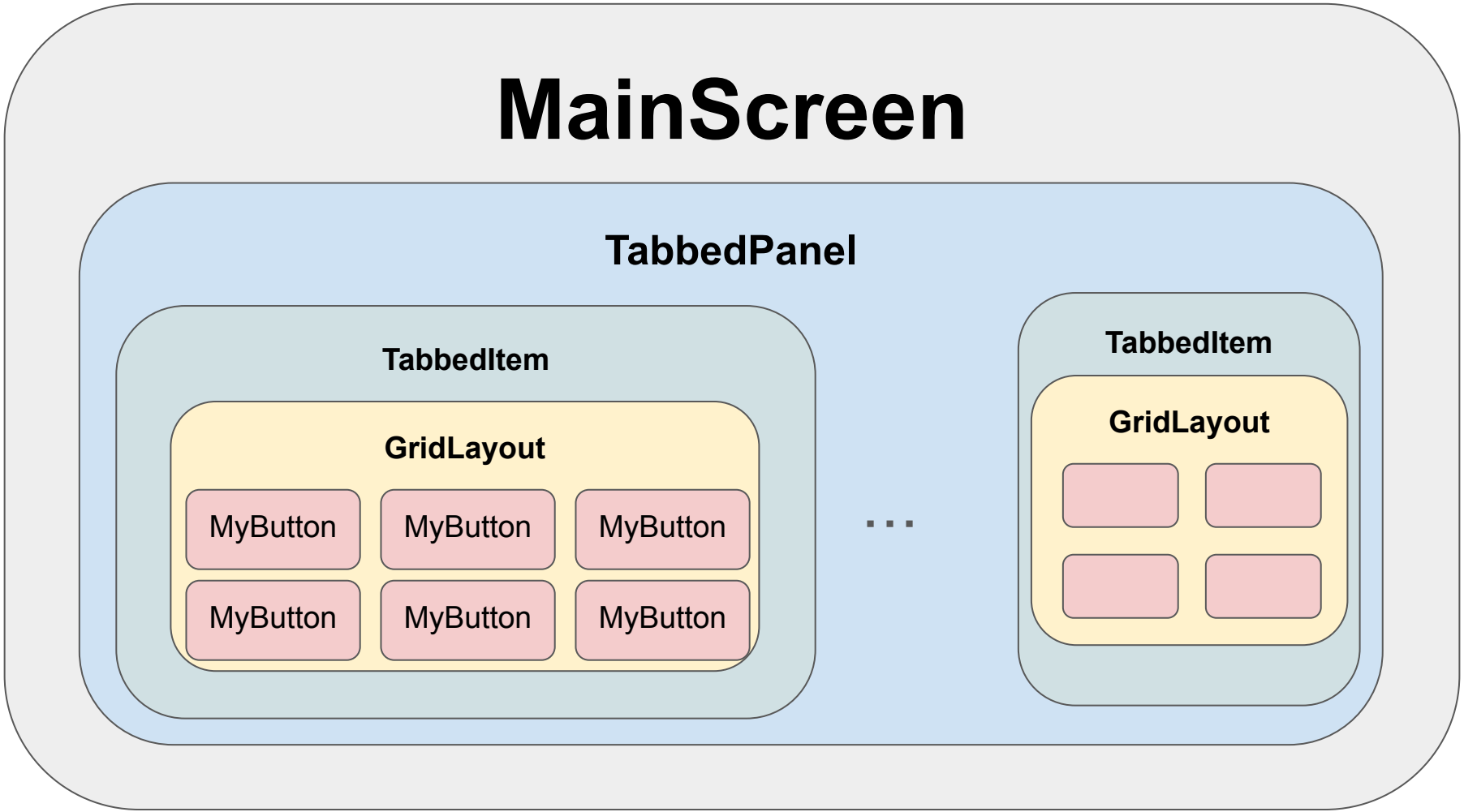
■kvファイル（アプリ内の表示機能をもつファイル）

- Menu.kv

ScreenManeger

※これはデフォルトの機能のまま利用しています




MainScreen
TabbedPanel
TabbedItem
GridLayout
MyButton
MyButton
MyButton
MyButton
MyButton
MyButton
. . .
TabbedItem
GridLayout

実際の動作をご覧ください

反省点

企画段階でどの言語、ライブラリを利用するかはよく調べるべき

- よく調べたらKivyは相当ニッチなライブラリだったので
  経験値としてあまり有用ではないかもしれない
  ※ReactNativeやflutter、少なくともSwiftにしておけば良かった
- kivyにデフォルトで搭載されている機能を把握するのに
  非常に時間がかかるし、エラーの原因がわからないことが
  多々あった。

チャットGPTの使いすぎに気を付けるべし

- チャットGPTを使うと大まかなアルゴリズムを作ってくれるので、
  自身でアルゴリズムを考える訓練にならない。
- エラーの原因を丸投げすると、なぜエラーが発生しているのか
  把握できず、自身の成長につながらない。
- 意外と間違っていることがよくある。
  kivyがマイナーなモジュールだからなのか、
  このように記述してくださいという通りに書いても動かず、
  ネットで調べるとその書き方では動かないことがわかる。

この経験から得た学び

自身で調べながら
新たなライブラリを利用できた

- 全く知らないモジュールを一から調べて、自分の希望している機能を最低限実装したアプリが作れたのは自信になりました。
- ライブラリにデフォルトで搭載されているクラスや関数を把握していくことで、後半は割とスムーズに開発ができたと思う。

Swiftの本を読んだらLabel、Button、BoxLayoutの記述があり、

もしかしたら全く無駄というわけでもなかったのかもしれない

実際に動くことの喜びが
次の挑戦へのモチベーションになる

- 実装したい機能が全くうまく動かず、製作が難航していても、

  諦めずに挑戦し続けて意図した通りに動いた時の嬉しさが尋常ではない。その喜びがあって次の機能を実装する気力が生まれるし、自身の成長も感じられるので、開発は非常に楽しんで進められた。

最後に

今回のように、

”この店はなぜこのメニューを採用しているんだ”

という疑問や、

”もっとこうなっていれば”

という日常のふとした気付きをメモにして、

開発に挑戦することは成長につながると思いました。