
Flutter

ติดตามผู้เขียน
ผ่านช่องทางยูทูป

https://www.youtube.com/channel/UCQ1r_4x-P-fETLIU4pqf98w

https://www.facebook.com/KongRuksiamTutorial/

หรือสแกน QRCODE

เรียนเนื้อหา Flutter & Database ได้ที่

https://bit.ly/362jX29

Phase 2 - เรียนอะไรบ้าง

- Navigation & Route และการใช้ Widget อื่นๆ
- แบบฟอร์มบันทึกข้อมูล
- ตรวจสอบความถูกต้องของข้อมูล
- รู้จักกับ Provider & Consumer
- ระบบฐานข้อมูล

พื้นฐานที่ต้องมี

- Flutter เบื้องต้น
  - เข้าเรียนได้ที่ : https://bit.ly/3aVAMiu
- Dart เบื้องต้น
  - เข้าเรียนได้ที่ : https://bit.ly/384jcHe

สร้าง Icon บน AppBar

```
actions:[
    IconButton:(icon:ชื่อไอคอน),
    onpressed:(){
    }
}
```

สำหรับเชื่อมโยงหน้าแอพแต่ละหน้าผ่านระบบ Navigation และระบบนำทาง (Route)

Navigation และ Route

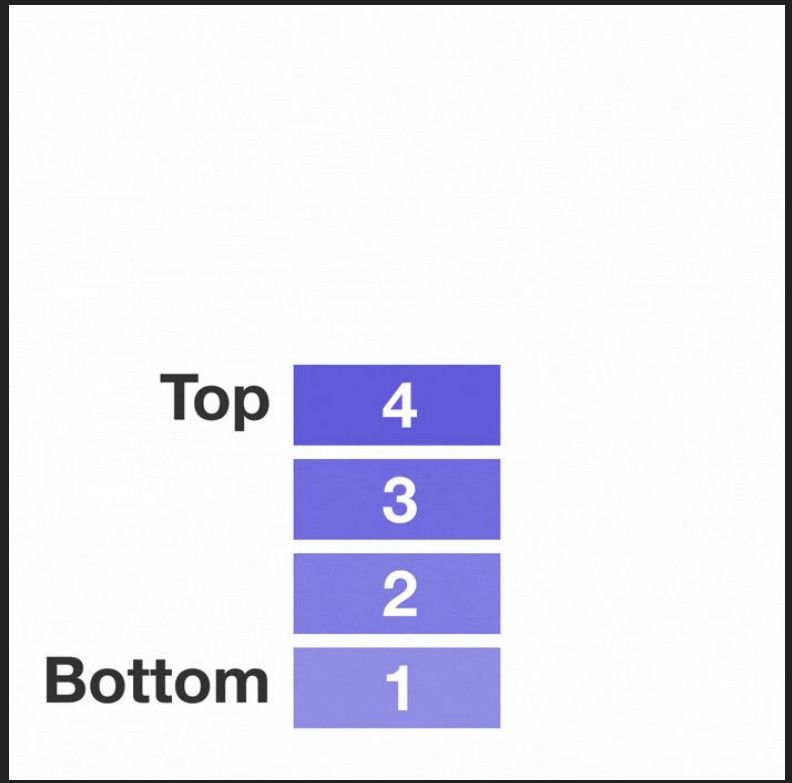
- ถ้าต้องการให้แอพเรามีหน้าจอหลายๆหน้าจอจะทำอย่างไร ?
- จะเชื่อมโยงการทำงานของแอพแต่ละหน้าจออย่างไร ?

- ใช้ความรู้ในการสร้าง Widget มาสร้างออกแบบหน้าจอแอพ
- เรียกหน้าจอแต่ละหน้าว่า Widget ย่อย หรือหน้าแอพย่อย
- เชื่อมโยงหน้าแอพแต่ละหน้าด้วย Navigator และ Route

Navigator Widget

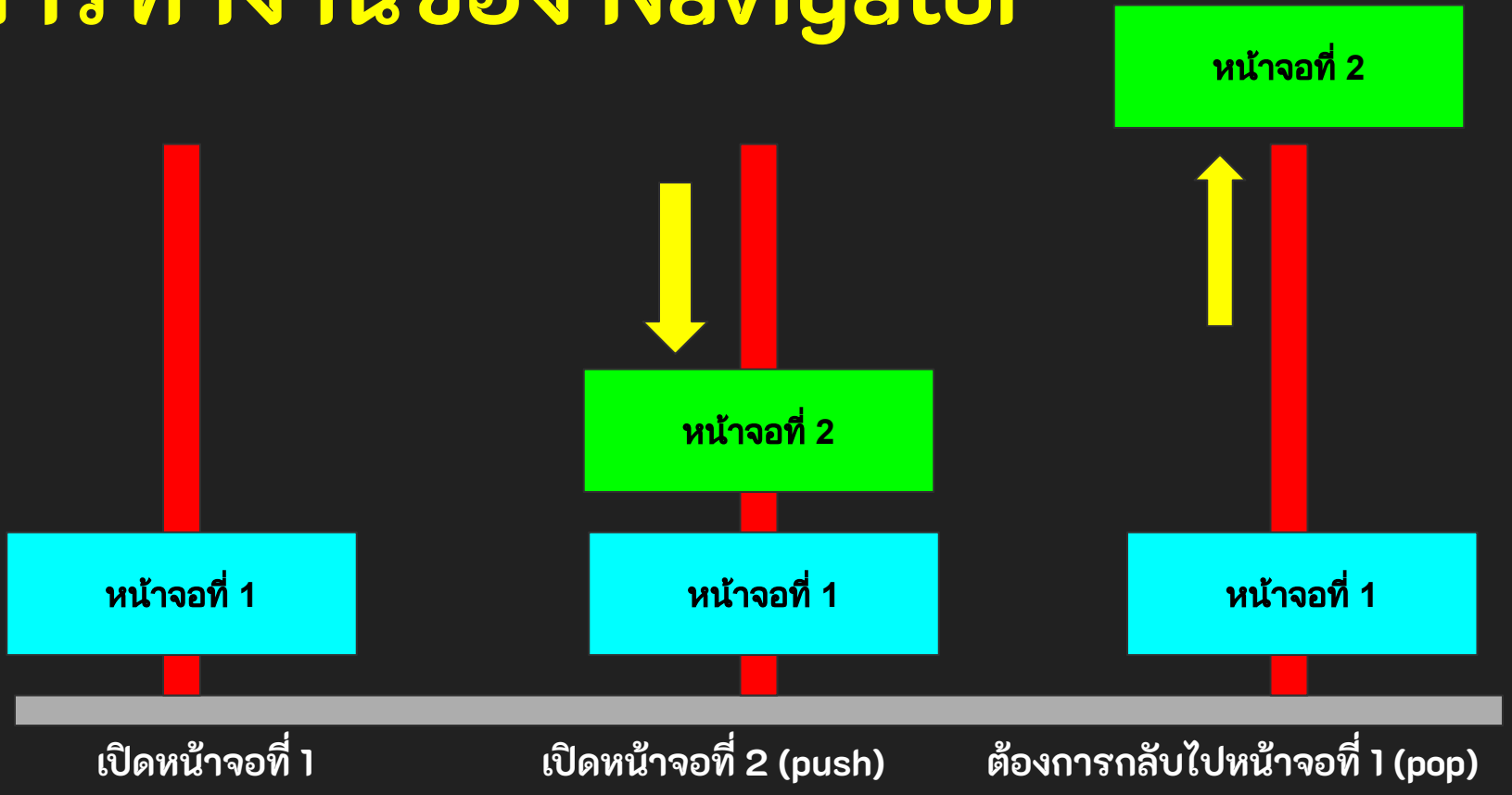
คือ กลุ่มของ Widget หรือ คลาสที่ใช้ร่วมกับ Route สำหรับจัดการ Widget ย่อยในแอพ โดยมีการ*จัดวางตามโครงสร้างข้อมูลแบบ Stack* คือการวางแผ่นซ้อนทับกันตามลำดับจากล่างขึ้นบน โดยแผ่นที่นำมาเรียงกันก็คือส่วนของ Widget ย่อยนั่นเอง แผ่นที่อยู่บนสุดจะถูกแสดงผลและทับหน้าอื่นๆเอาไว้ที่เปิดใช้งานก่อนหน้านี้

การทำงานของ Stack



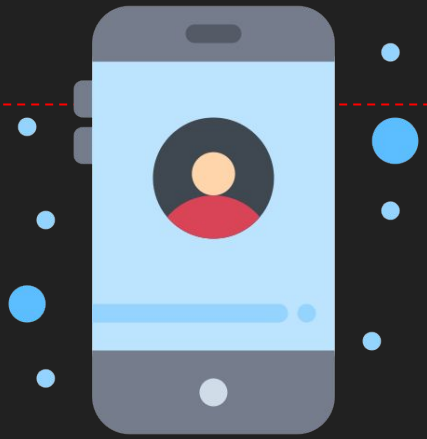
- Push การนำสมาชิกมาใส่ไว้บนสุดของ Stack (Top Stack)
- Pop การนำสมาชิกบนสุดออกไปจาก Stack

การทำงานของ Navigator

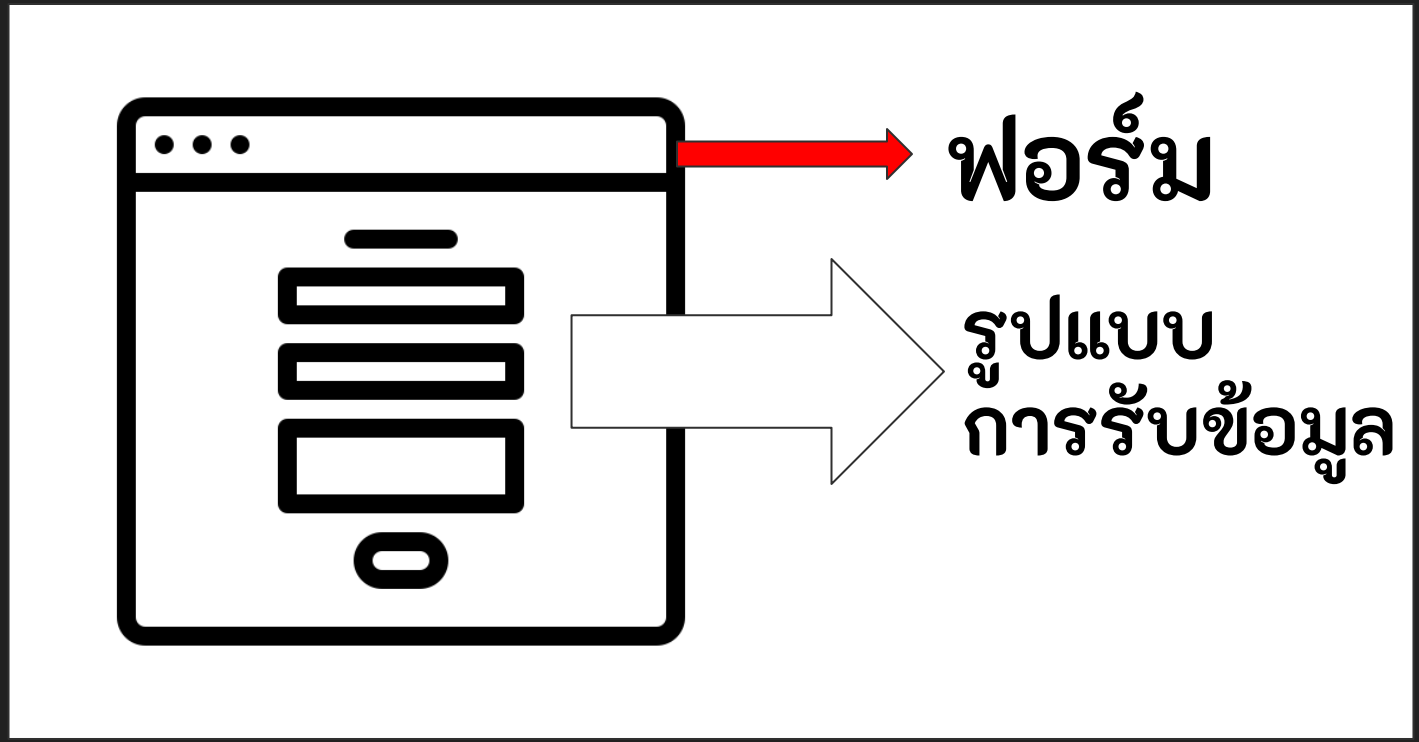


การสร้างแบบฟอร์ม

คือ การรับข้อมูล (Input) จากผู้ใช้ เช่น ข้อความ ตัวเลข วันเวลา หรือตัวเลือกต่างๆ ซึ่งมีโครงสร้างอยู่ 2 ส่วน คือ แบบฟอร์ม และ ส่วนควบคุมการทำงานของแบบฟอร์ม (Controller)



โครงสร้างของแบบฟอร์ม



การจัดรูปแบบฟอร์ม

- autoFocus – สั่งโฟกัสที่ช่องรับข้อมูลในตอนเริ่มต้น
- keyboardtype – กำหนดรูปแบบของช่องรับข้อมูล

แสดงข้อมูลด้วย Card Widget

- คือการแสดงผลแบบเป็นการ์ด (Card = ไพ่ หรือ บัตร) มีการกำหนด Attribute ชื่อว่า elevation ในการกำหนดเงาให้การ์ด

Margin

คือ การกำหนดระยะห่างของ Widget ออกจาก**ขอบนอก**ของ Layout

Padding

คือ การกำหนดระยะห่างของ Widget ออกจาก**ขอบใน**ของ Layout

State Management

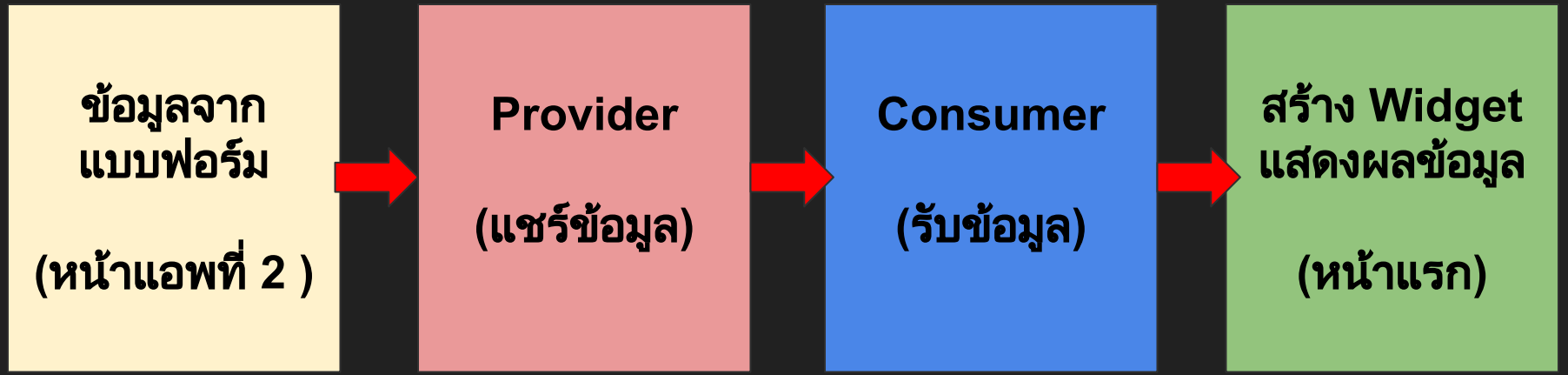
ในกรณีที่เรามีหน้าแอพมากกว่า 1 หน้า การจัดการ State ก็จะมีความยุ่งยากมากยิ่งขึ้น เนื่องจากเราต้องนำเอา State หรือชุดข้อมูลต่างๆไปทำงานกับหน้าแอพที่เราได้กำหนดขึ้นมา พูดง่ายๆคือถ้าเรามีชุดข้อมูลแล้วอยากจะนำไปใช้งานในหน้าแอพแต่ละหน้าที่แตกต่างกันออกไปจะทำอย่างไร ?

State Management

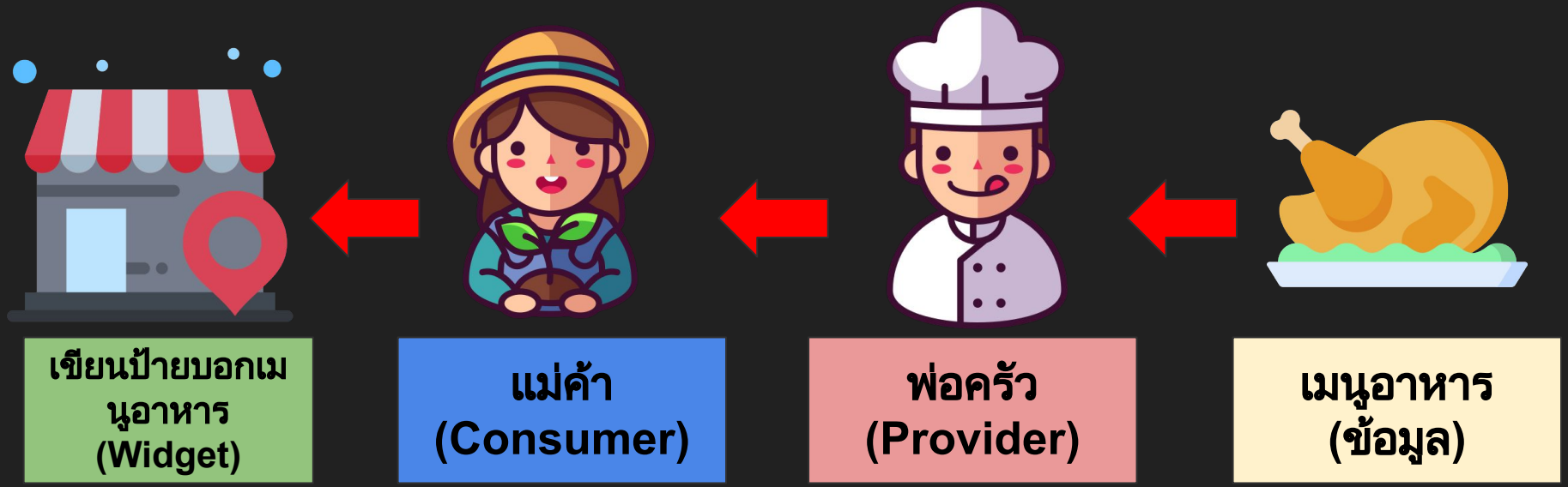
การจัดการปัญหาดังกล่าวเราจะใช้ส่วนที่เรียกว่า การจัดการ State หรือ Statemanagement นั่นเอง ซึ่งแบ่งออกเป็น 2 ส่วน คือ

- Provider ดูแลและจัดการข้อมูลแล้วนำไปส่งให้ Consumer
- Consumer นำข้อมูลที่ได้จาก Provider ไปสร้างหรือแสดงผลใน หน้าแอพหรือ Widget

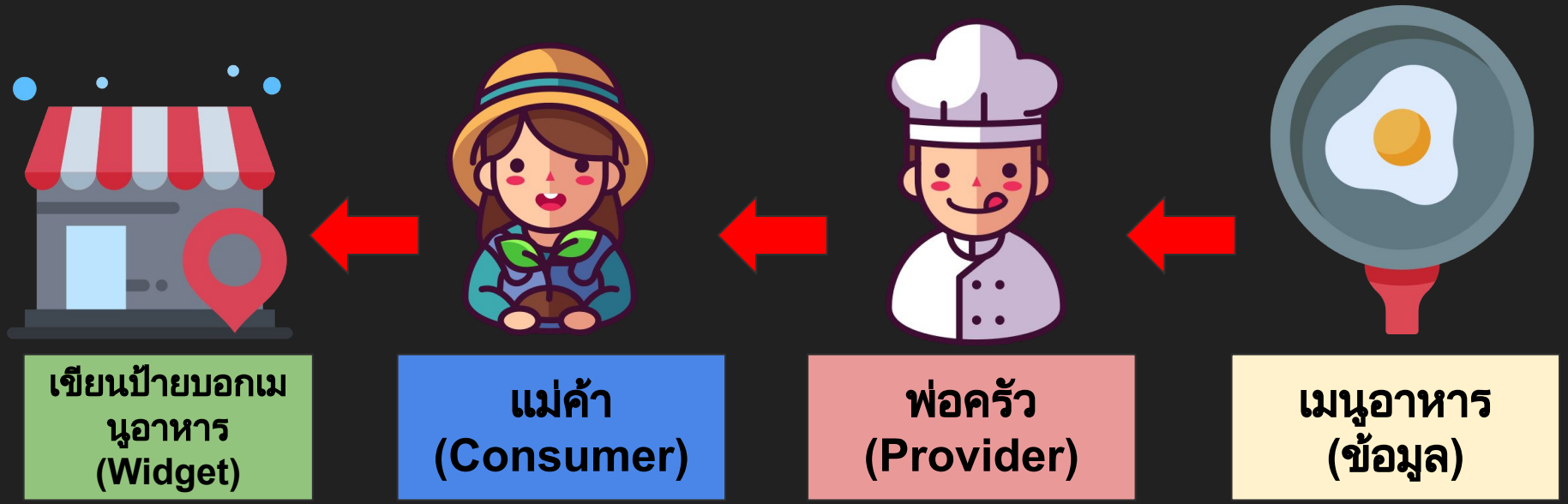
แผนภาพการทำงาน



แผนภาพการทำงาน



แผนภาพการทำงาน



ตัวเชื่อมการทำงาน Provider / Consumer

แม่ค้า
(Consumer)

Multiple
Provider

พ่อครัว
(Provider)

เมนูอาหาร
(ข้อมูล)

ตัวเชื่อมการทำงาน (Multiple Provider)

แม่ค้า
(Consumer)

Multiple
Provider

Provider 1

Provider 2

Provider

```
Provider<Something>(
  create: (_) => Something(),
  child: Provider<SomethingElse>(
    create: (_) => SomethingElse(),
    child: Provider<AnotherThing>(
      create: (_) => AnotherThing(),
      child: someWidget,),
  ),),
```

Multiple Provider

```
MultiProvider(
        providers: [
          Provider<Something>(create: (_) => Something()),
          Provider<SomethingElse>(create: (_) => SomethingElse()),
          Provider<AnotherThing>(create: (_) => AnotherThing()),
        ],
        child: someWidget,
)
```

Local Database

การเก็บข้อมูลในพื้นที่เก็บข้อมูลของเครื่องนั้นๆ และเรียกใช้เมื่อต้องการข้อมูล แต่ถ้าผู้ใช้ทำการลบแอพหรือล้างเครื่องข้อมูลก็จะสูญหายไปด้วย

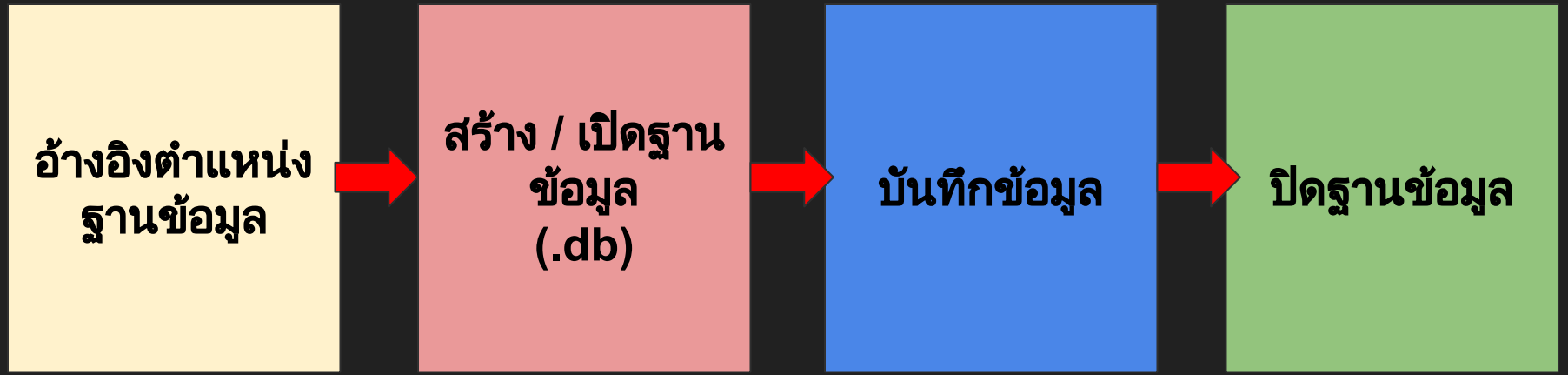
รูปแบบการเก็บข้อมูล

Relational Database	Non-Relational Database
SQL	NOSQL

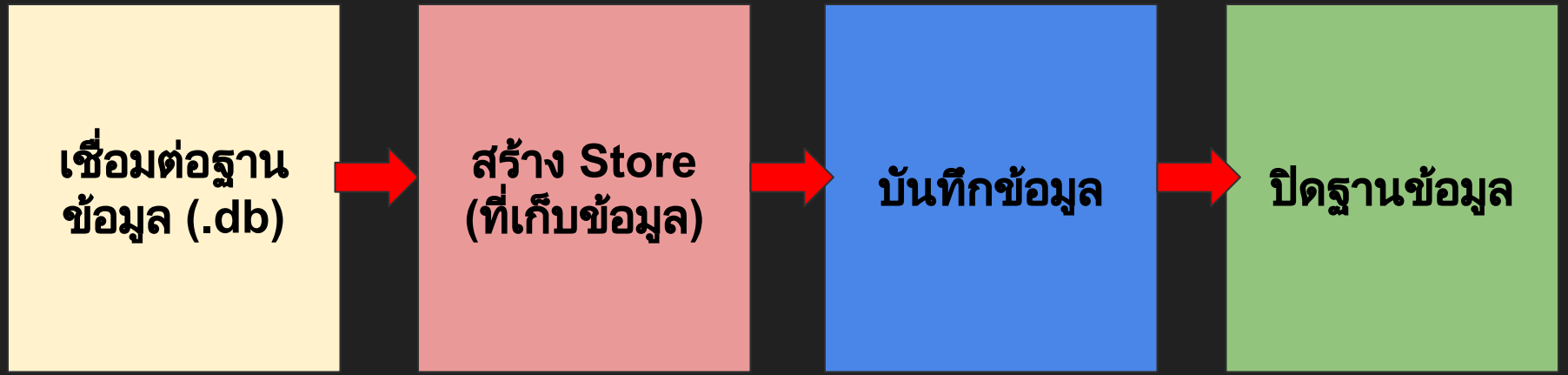
รูปแบบการเก็บข้อมูล

SQL	NOSQL
Table	Collections
Row	Document
Column	Field
Primary Key	ObjectID

ภาพรวมของระบบ



ขั้นตอนการทำงาน



ติดตั้ง Package

- Sambast - จัดการฐานข้อมูล
- Path_Provider - ดึงตำแหน่งฐานข้อมูลของเครื่อง
- Path - อ้างอิงตำแหน่งที่เก็บฐานข้อมูล

นำไปใส่ใน pubspac.yaml

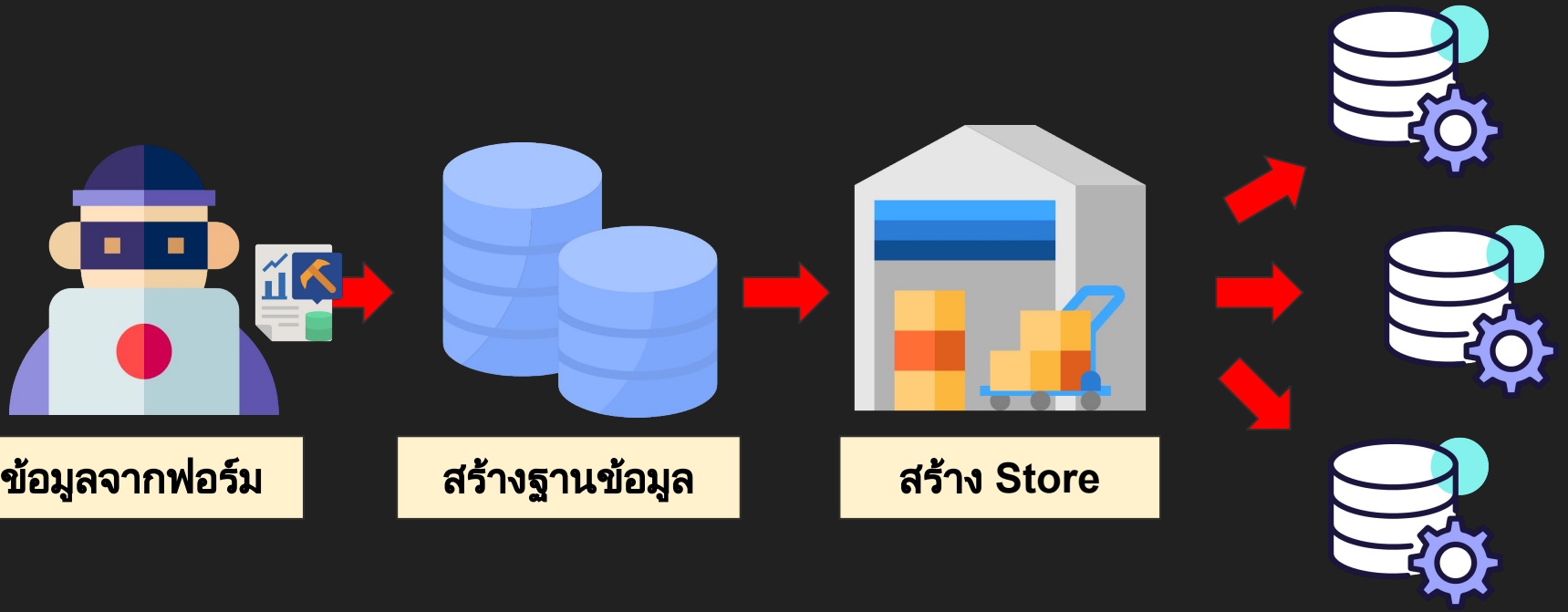
สร้างฐานข้อมูลในเครื่องผู้ใช้

- สร้าง class สำหรับจัดการฐานข้อมูล
- เปิดและปิดฐานข้อมูล
- ดำเนินการเพิ่มและดึงข้อมูลมาใช้ในแอพถ้าปิดแอพไปข้อมูลก็ยังคงอยู่ที่เครื่องเหมือนเดิม

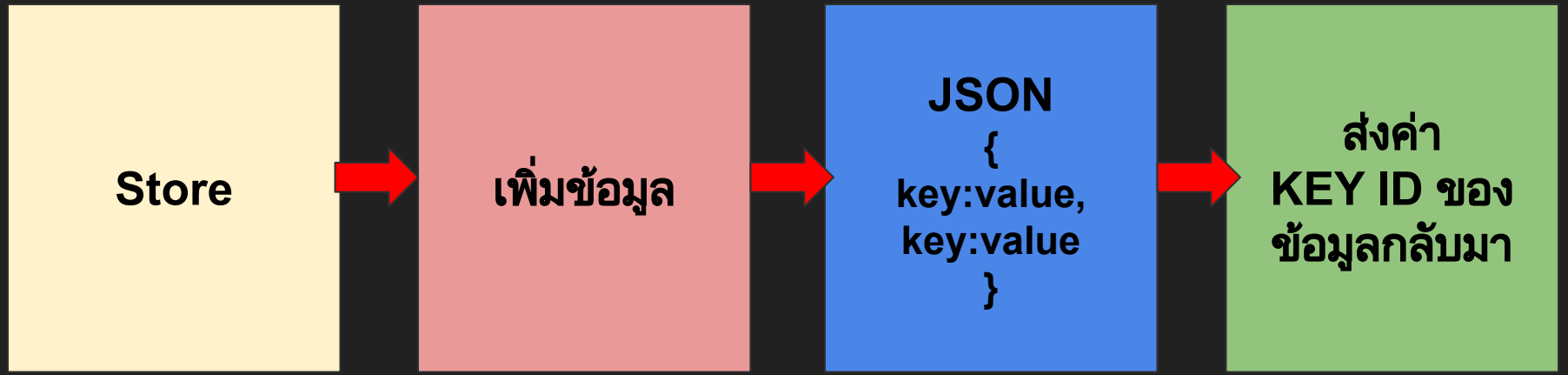
สร้าง Store

การสร้าง Store คือการระบุที่จัดเก็บข้อมูลในแอพว่าชื่ออะไร มีรูปแบบการจัดเก็บข้อมูลแบบใด (คล้ายๆกับการสร้างตารางในฐานข้อมูล) โดยใช้ intMapStoreFactory

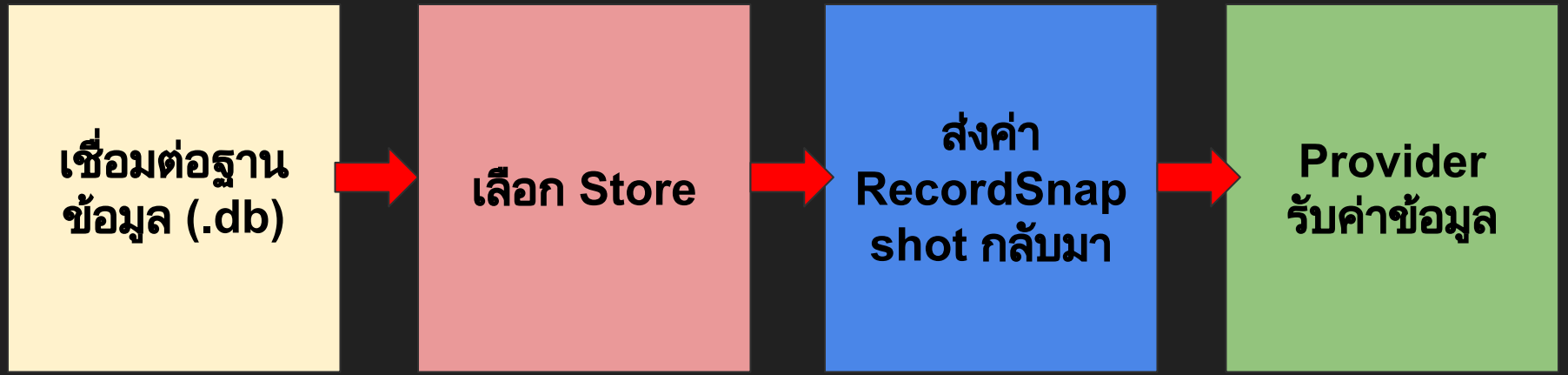
ขั้นตอนการทำงานของ Store



แผนภาพการบันทึกข้อมูล



แผนภาพการดึงข้อมูล (1)



แผนภาพการดึงข้อมูล (2)

การทำงานของ Snapshot

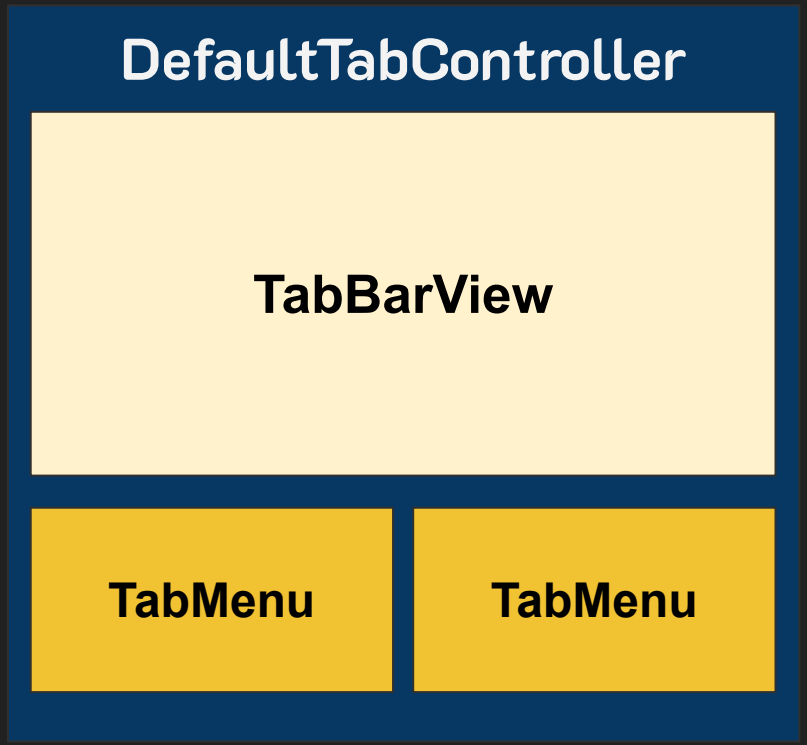
ปัญหาเกิดจากการเก็บข้อมูลที่มีสภาพแวดล้อมต่างกัน เช่น เก็บข้อมูลใน Android หรือ iOS สภาพแวดล้อมของแต่ละ Platform แตกต่างกันโดยสิ้นเชิง ส่งผลให้การจัดการข้อมูลมีความยุ่งยากไปด้วย จึงมีแนวคิดในการสร้างประเภทข้อมูลที่สามารถทำให้ข้อมูลทำงานได้ในสภาพแวดล้อมต่างกันเราเรียกส่วนนี้ว่า

RecordSnapshot

การทำงานของ Snapshot

การทำงานของ Snapshot

การทำเมนูแบบ Tab



องค์ประกอบอยู่ 3 ส่น คือ

- DefaultTabController ควบคุมระบบโดยรวมของ Tab Menu ทั้งหมด เช่น กำหนดจำนวนเมนู เป็นต้น
- TabBarView แสดง Widget หน้าตาของแอพตรงส่วนของ Tab ที่เรากำลังทำงาน
- TabMenu ส่วนของเมนูที่แสดงใน Tab