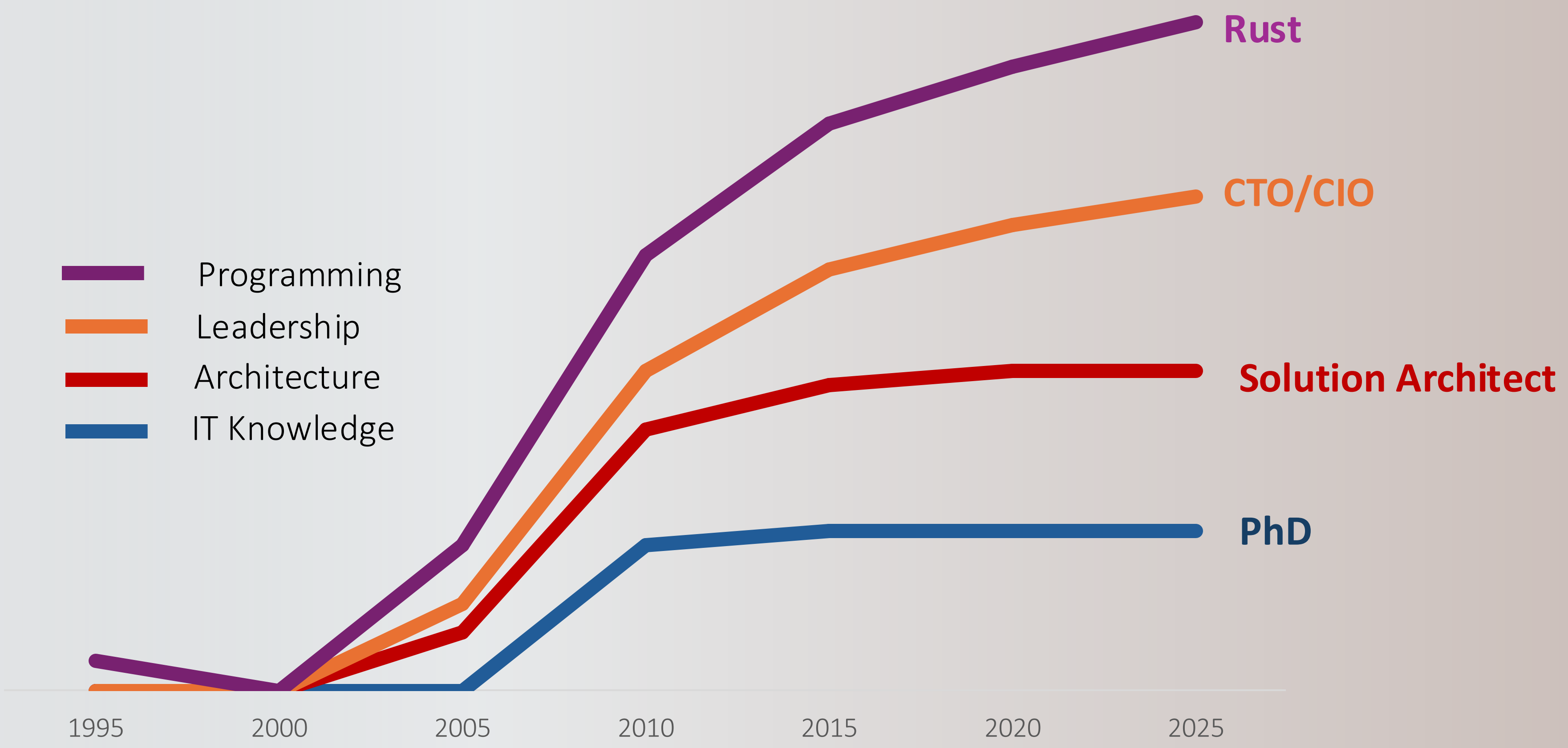
FLUTTER STATE MANAGEMENT (AND BUSINESS LOGIC) IN RUST

Patrick Mukherjee
patrick.mukherjee@gmail.com www.linkedin.com/in/patmuk

whoami


Programming
Leadership
Architecture
IT Knowledge
Rust
CTO/CIO
Solution Architect
PhD
1995
2000
2005
2010
2015
2020
2025

What I will show you ...

01 Why Flutter & Rust

Why Flutter & Rust?

- Java/.net →VM not as efficient
- C++ → requires (more) platform-specific code

→ Flutter & Rust compile native

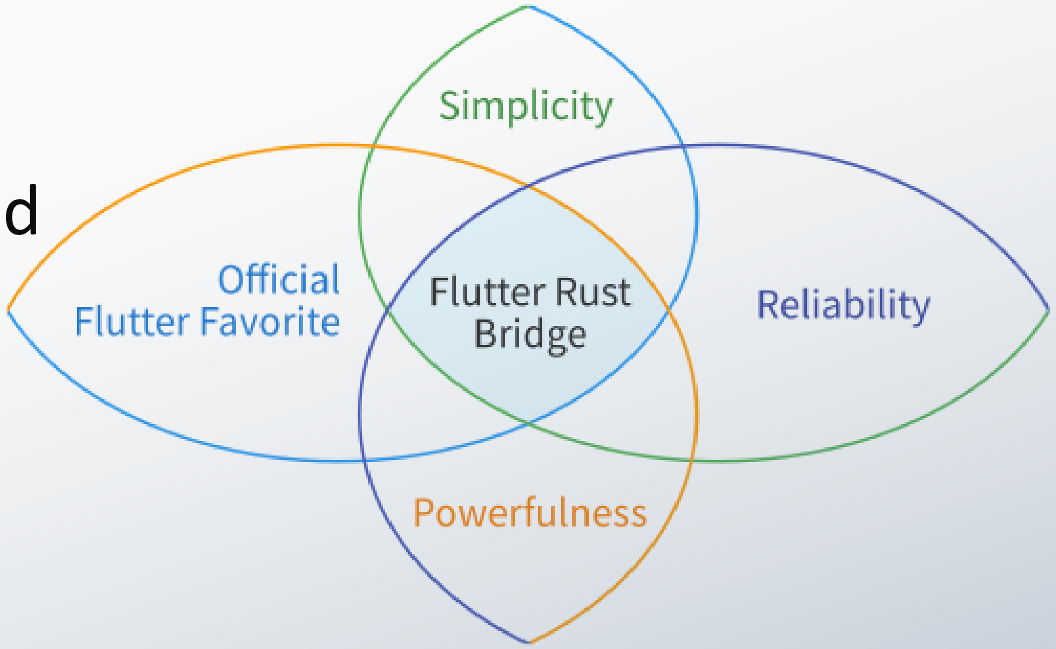
write once
run anywhere

02 How Flutter & Rust (FRB, FFI)

How Flutter & Rust (FFI)

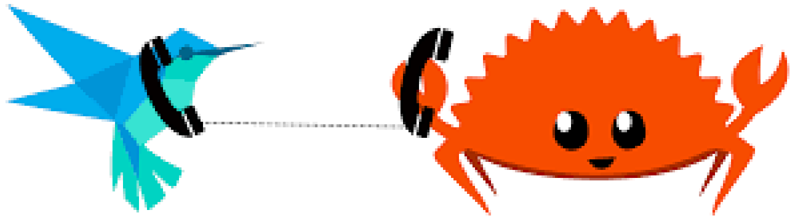
- Dart and Rust communication through generated code.
- Flexibility without enforcing an opinionated structure.

- Rust-Written (boilerplate) Code Generator
  - FFI (Dart's ffigen[1])
  - SerDe



1) https://pub.dev/packages/ffigen

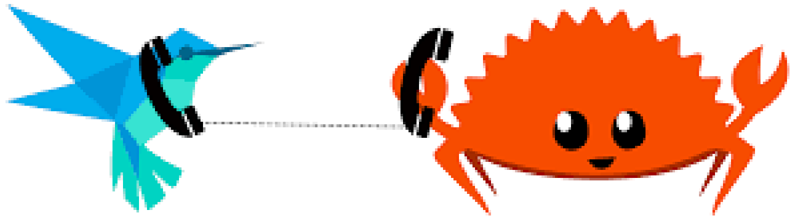
Flutter Rust Bridge (FRB)[1]



- Simple Rust...
  - `pub fn f(a: String, b: Vec<MyEnum>) -> MyStruct { ... }`
- ...called from Dart
  - `print(f(a: 'Hello', b: [MyEnum.c('Tom')]));`

1) https://github.com/fzyzcjy/flutter_rust_bridge

Flutter Rust Bridge (FRB)[1]

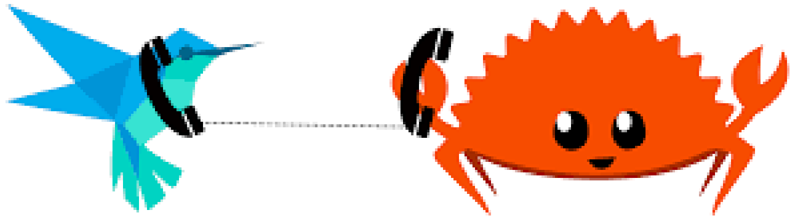


- And vice-versa:

  -
    ```
    pub fn rust_fn(dart_callback: Fn(String) -> String) {
        dart_callback("Tom"); // Will get `Hello, Tom!`
    }
    ```
- ... Dart closure
  - `rustFn(dartCallback: (name) => 'Hello, $name!');`

1) https://github.com/fzyzcjy/flutter_rust_bridge

Flutter Rust Bridge (FRB)[1]



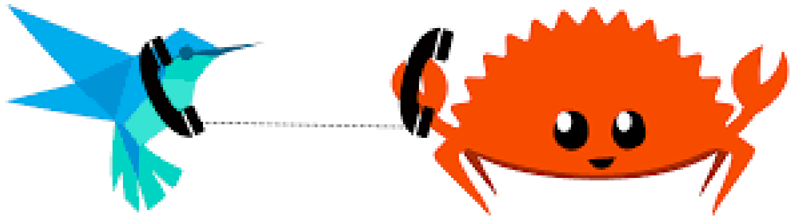
- And vice-versa:

  -
    ```
    pub fn rust_fn(dart_callback: Fn(String) -> String) {
        dart_callback("Tom"); // Will get `Hello, Tom!`
    }
    ```
- ... Dart closure
  - `rustFn(dartCallback: (name) => 'Hello, $name!');`

Adapted Style!

1) https://github.com/fzyzcjy/flutter_rust_bridge

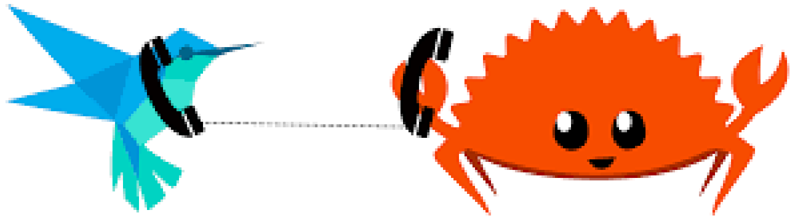
Flutter Rust Bridge (FRB)[1]



- Async out-of-the box (can be sync)
- Steam (Dart) → Iterator (Rust)
- Zero-Copy Rust → Dart
  - External typed data[2] & Dart_PostCObject[3]
  - Only for Vec<u8> → Uint8List
- Not Dart → Rust

1) https://github.com/fzyzcjy/flutter_rust_bridge
2) https://github.com/dart-lang/sdk/blob/6fcd15c1aa024bd42056487374a146be492277a2/runtime/include/dart_native_api.h#L93
3) https://github.com/dart-lang/sdk/blob/6fcd15c1aa024bd42056487374a146be492277a2/runtime/include/dart_native_api.h#L127

Flutter Rust Bridge (FRB)[1]



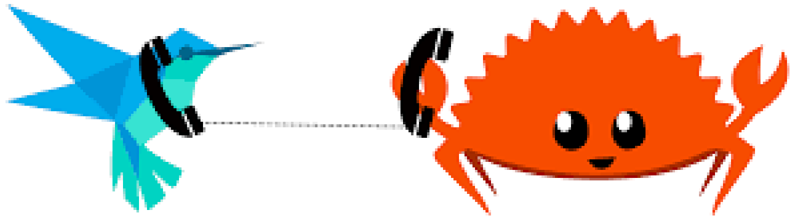
- References:
  - FRB keeps alife as needed
  - GC'ed in Dart (or call Foo.dispose)
- Limitations:
  - No Return References: (`fn f<'a>(foo: &'a Foo) -> &'a Bar { .. }`)
  - MOVE → can be wrapped:

```
pub struct BarReference<'a>(&'a Bar);
fn f<'a>(foo: &'a Foo) -> BarReference<'a> { .. }
```

  - ! YOU handle concurrent access (e.g. inner & / mut &) !

1) https://github.com/fzyzcjy/flutter_rust_bridge

Flutter Rust Bridge (FRB)[1]

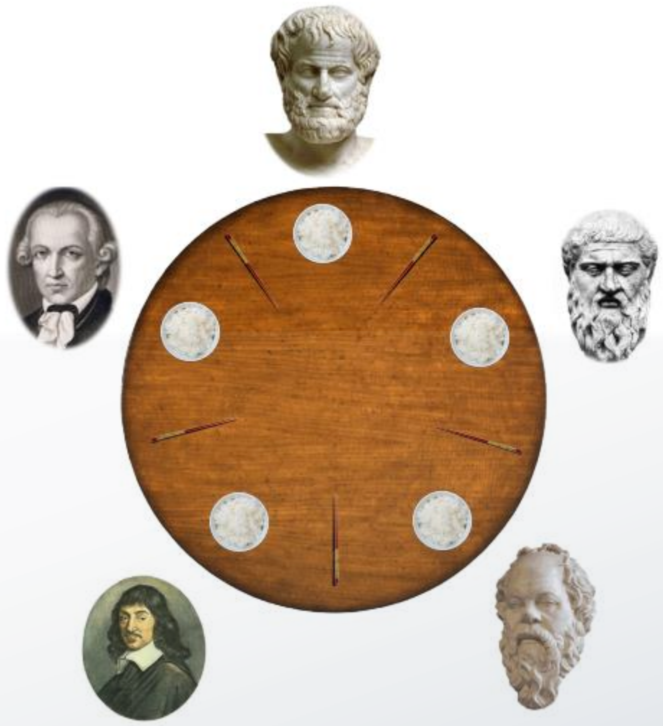


Want to know more?

Tom offers giving a talk!

1) https://github.com/fzyzcjy/flutter_rust_bridge

How to handle concurrency?



- Wrapper:

```
pub struct TodoListModelLock {
        pub model: RustAutoOpaque<TodoListModel>,
}
```

- Is simplified:

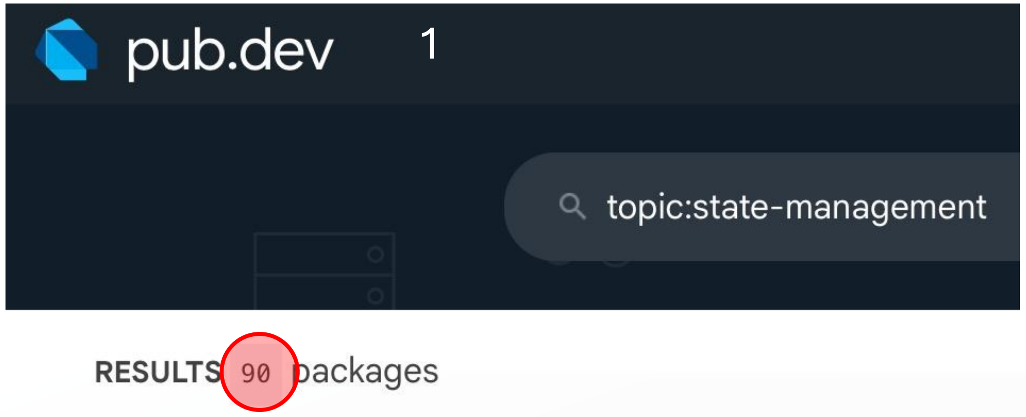
```
RustAutoOpaque<TodoListModel> ≈ Arc<RWLock<TodoListModel>>
```

- (Simplified) Getters & Setters for TodoListModelLock:

```
pub fn query_get_todo(&self, todo_pos: usize) -> TodoListItem {
        &self.model.blocking_read().items[todo_pos - 1].clone()
}
```

03 State Management

State Management: MVC, MVVM und MfG

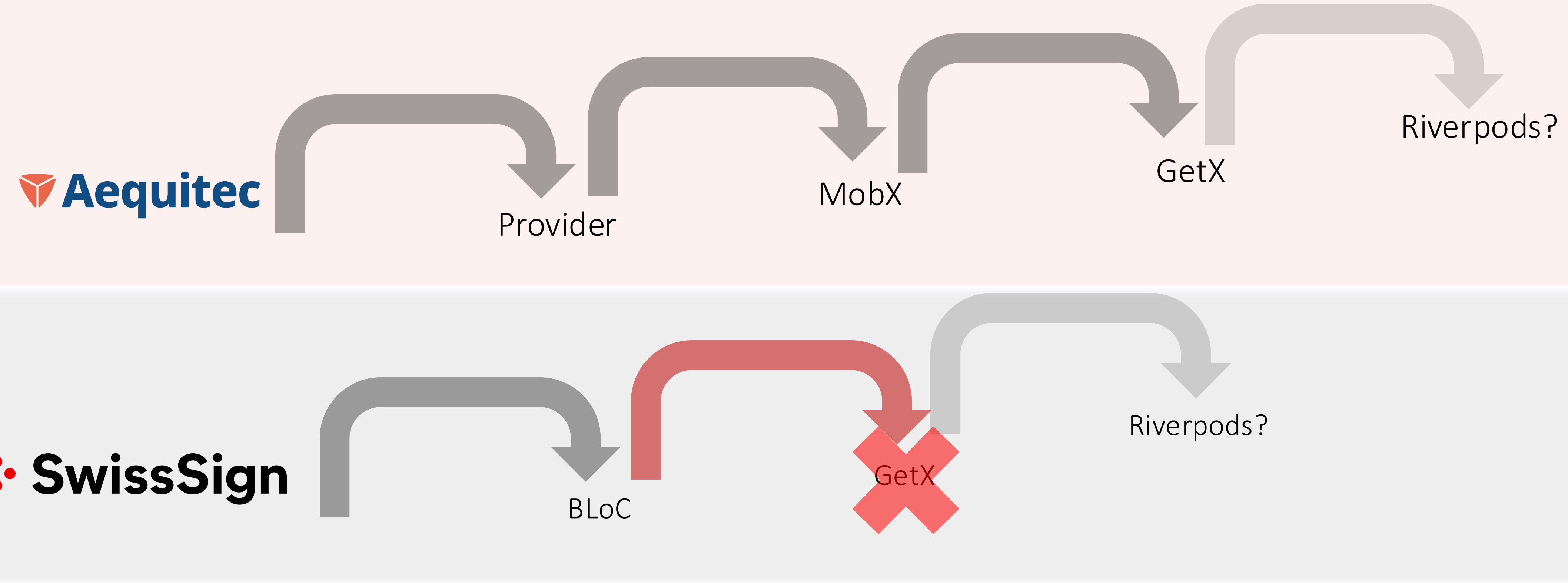


"State management is a complex topic. If you feel that some of your questions haven't been answered, or that the approach described on these pages is not viable for your use cases, you are probably right."
-- Flutter documentation, State Management Options[2]

Endorsed/Popular:
Provider[3] .... (too) basic
BLoC[4] .... suits medium size
Riverpods[5]........ for complex projects

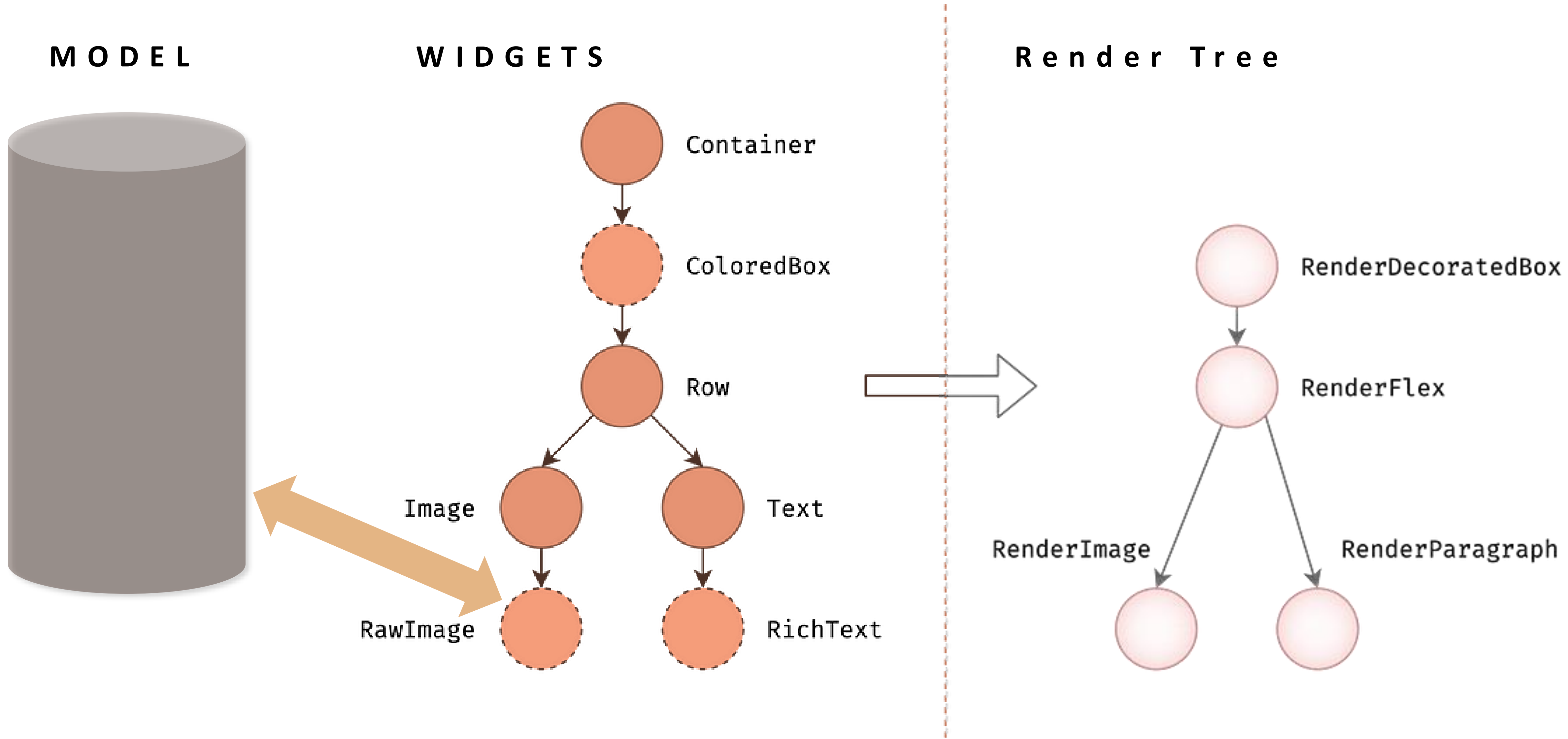
1) https://pub.dev/packages?q=topic%3Astate-management
2) https://docs.flutter.dev/data-and-backend/state-mgmt/options
3) https://pub.dev/packages/provider
4) https://bloclibrary.dev
5) https://riverpod.dev

State Management I used (to refactor)

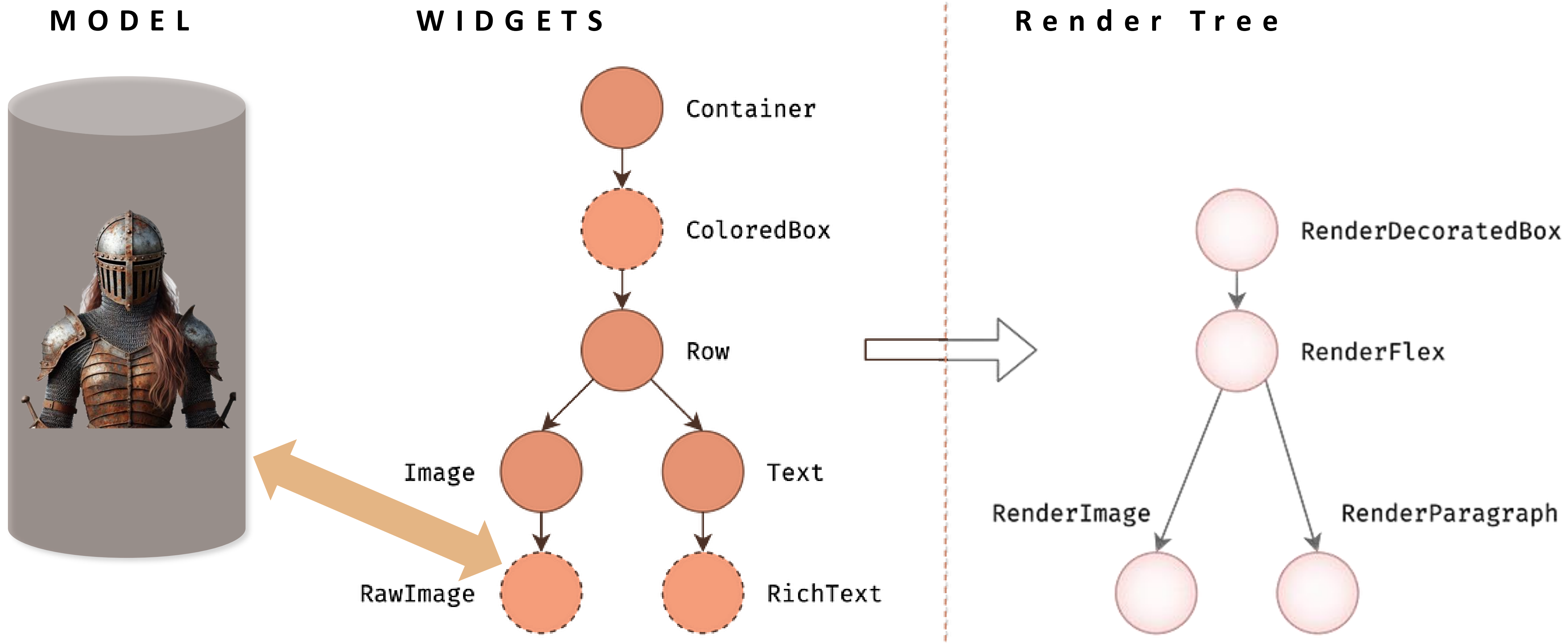


- Boilerplate or too dissimilar
- Basic Principle: **Global Variable** and **Publish/Subscribe**

State Management: In Flutter?



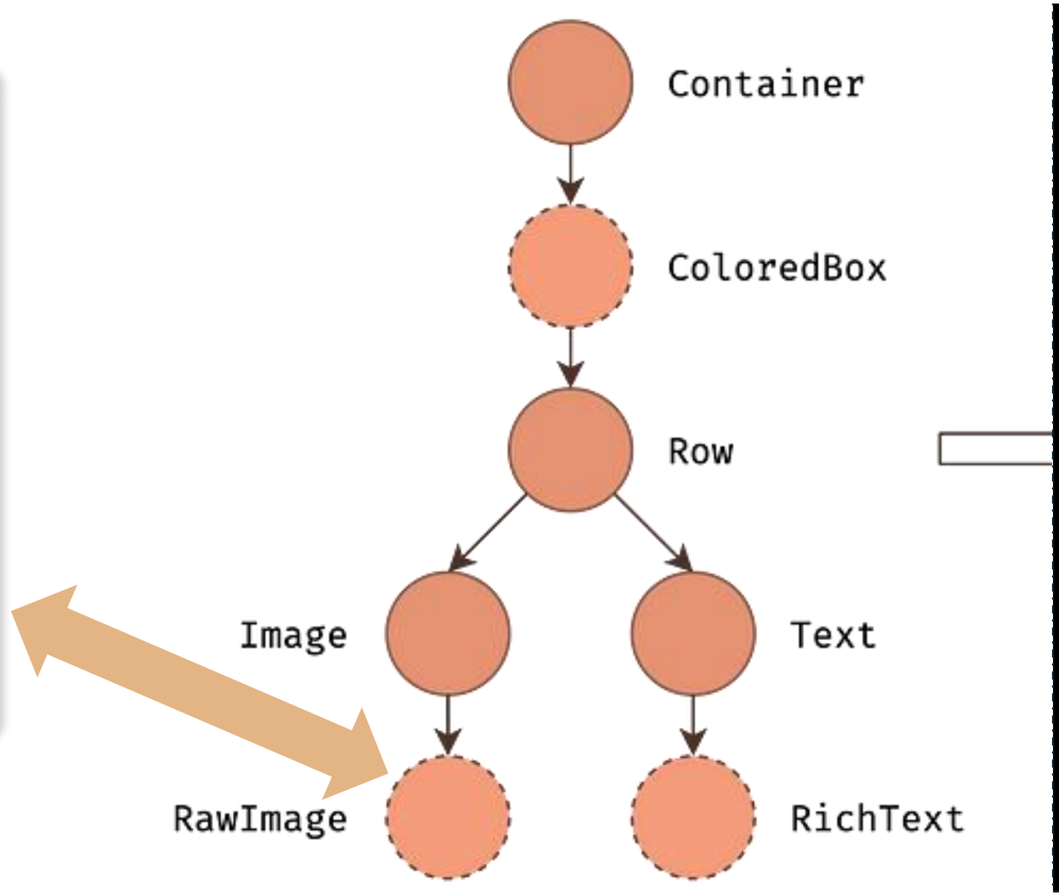
State Management: In Rust!



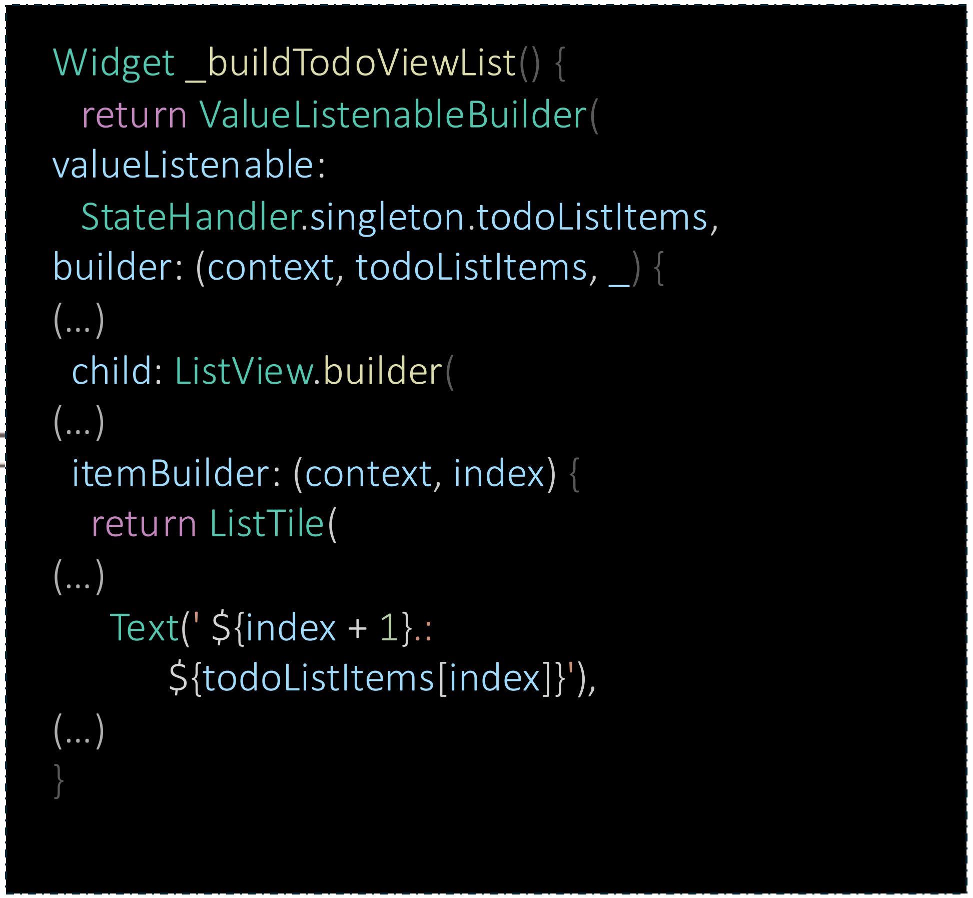
State Management: In Rust!

MODEL

WIDGETS



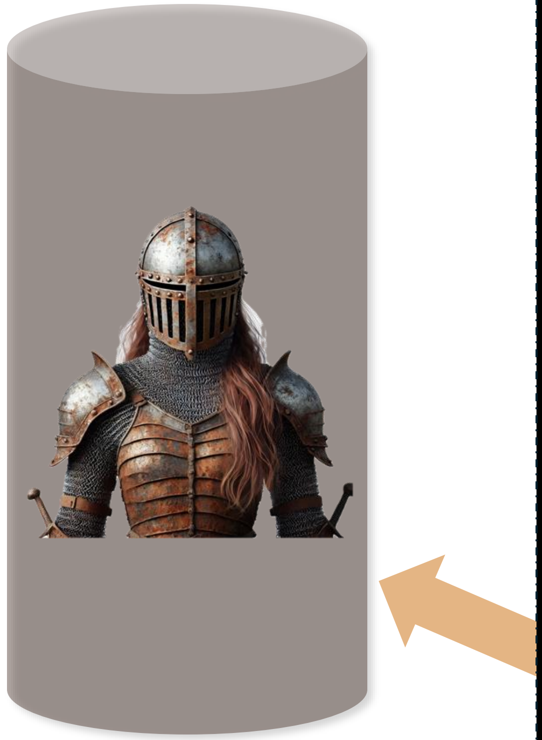
Render Tree



```
Widget _buildTodoViewList() {
  return ValueListenableBuilder(
valueListenable:
  StateHandler.singleton.todoListItems,
builder: (context, todoListItems, _) {
(...)
 child: ListView.builder(
(...)
 itemBuilder: (context, index) {
   return ListTile(
(...)
     Text(' ${index + 1}.:
         ${todoListItems[index]}'),
(...)
}
```

State Management: In Rust!

MODEL



WIDGETS

```
class StateHandler {
StateHandler._singletonConstructor();

/// ViewModels, observed by the UI
final ValueNotifier<List<String>>
  todoListItems =
    ValueNotifier(List.empty());

final ValueNotifier<String>
  todoListTitle =
    ValueNotifier("Click here to
                             set a title");
```

Render Tree

```
Widget _buildTodoViewList() {
  return ValueListenableBuilder(
valueListenable:
  StateHandler.singleton.todoListItems,
builder: (context, todoListItems, _) {
(...)
 child: ListView.builder(
(...)
 itemBuilder: (context, index) {
   return ListTile(
(...)
    Text(' ${index + 1}.:
         ${todoListItems[index]}'),
(...)
}
```

State Management: In Rust!

MODEL

WIDGETS

```
class StateHandler {
StateHandler._singletonConstructor();

/// ViewModels, observed by the UI
final ValueNotifier<List<String>>
  todoListItems =
    ValueNotifier(List.empty());

final ValueNotifier<String>
  todoListTitle =
    ValueNotifier("Click here to
                              set a title");
```

Render Tree

```
Widget _buildTodoViewList() {
  return ValueListenableBuilder(
valueListenable:
  StateHandler.singleton.todoListItems,
builder: (context, todoListItems, _) {
(...)
  child: ListView.builder(
(...)
  itemBuilder: (context, index) {
    return ListTile(
(...)
      Text(' ${index + 1}.:
           ${todoListItems[index]}'),
(...)
}
```

04 Architecture

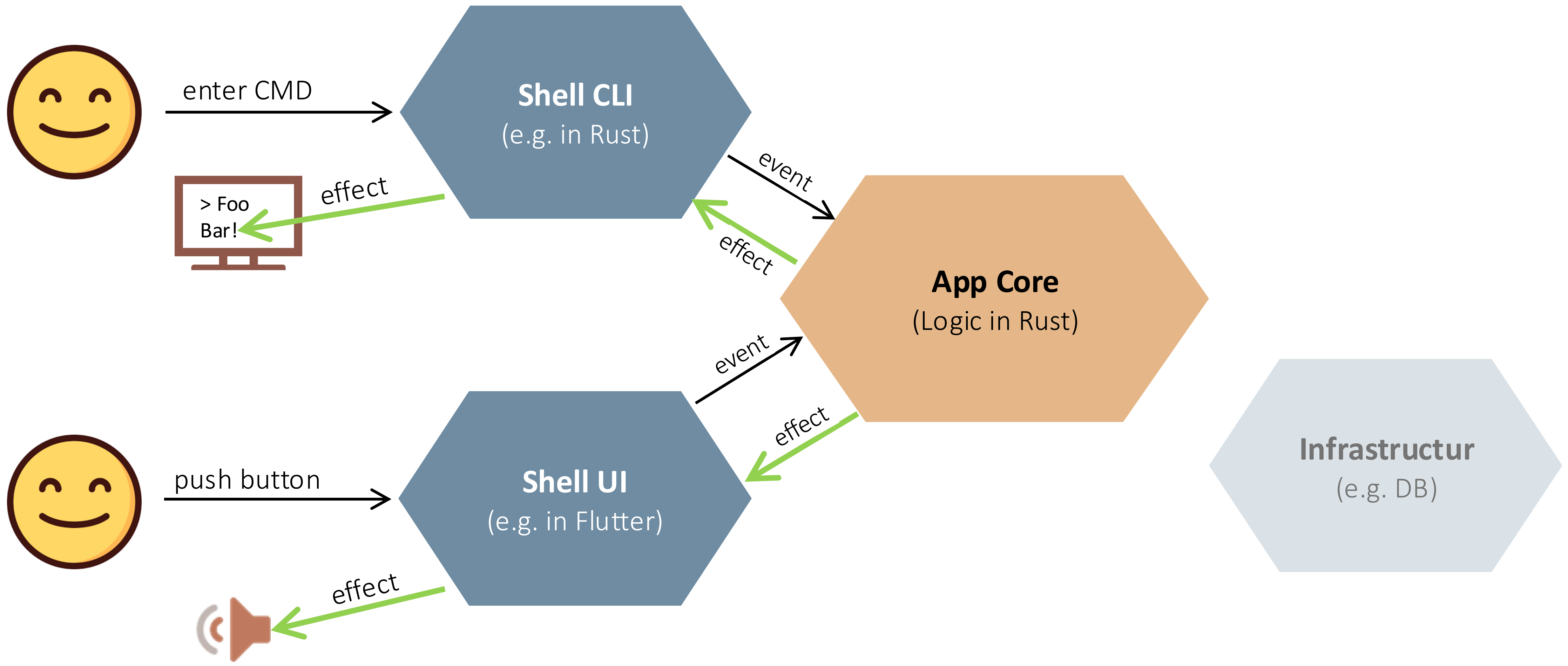
- Crux architecture overview
- Event-driven / CQRS
- Implementation in Rust

Event-Driven, clean architecture

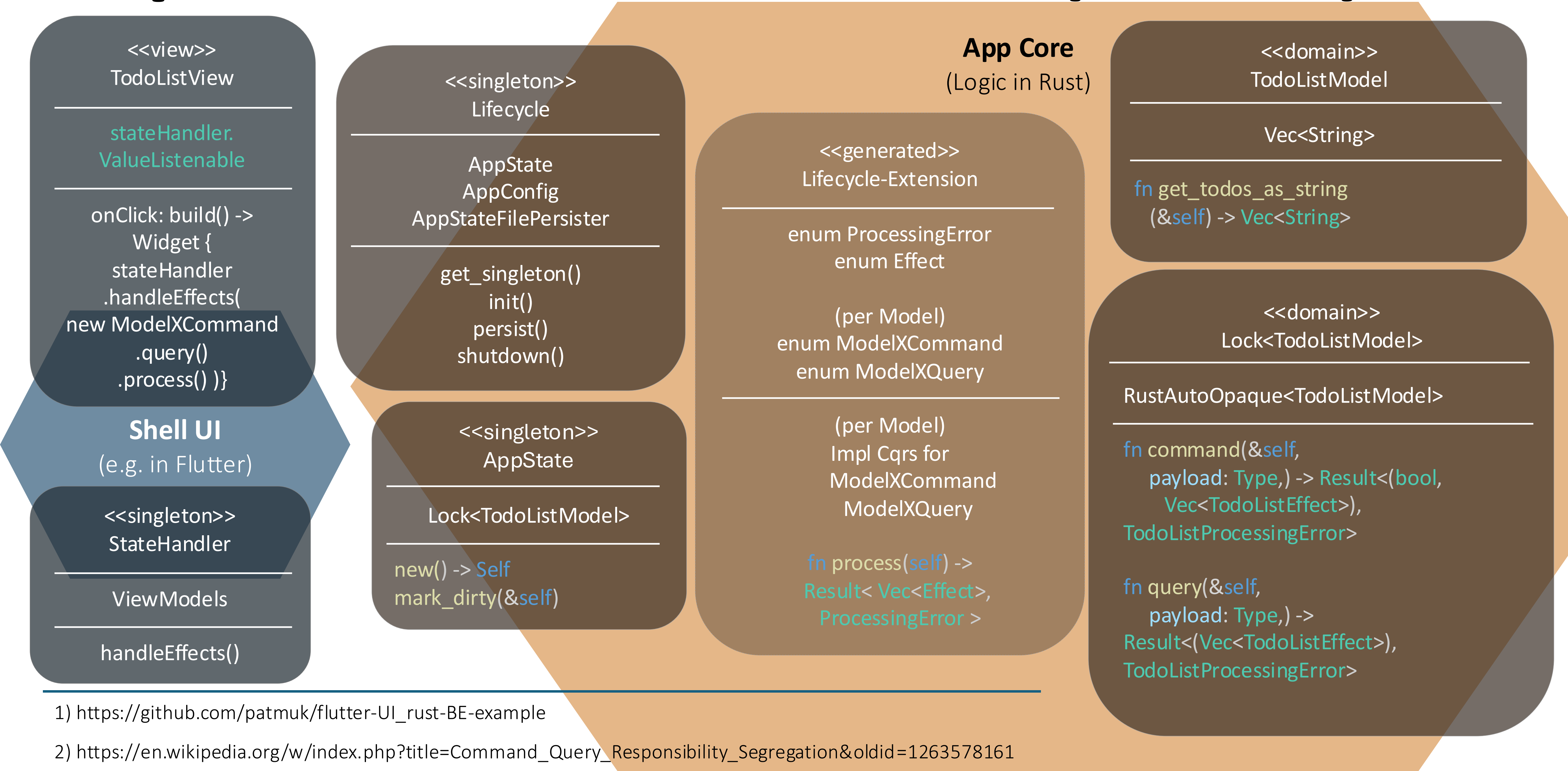
- Inspired by CRUX[1], which is inspired by
  - Elm, event-sourcing, event-driven and hexagonal architecture
  - is a Rust-to-X framework
  - X = &[Swift, Kotlin, TypeScript]; x ≠ Dart
  - X = &[SwiftUI, Jetpack Compose, React/Vue, WASM (Yew)]
  - X ≠ Flutter

1) https://github.com/redbadger/crux

Event-Driven, clean architecture



Implementation[1] in Rust (CQRS[2])



Shell UI
(e.g. in Flutter)

<<view>>
TodoListView

stateHandler.
ValueListenable

onClick: build() ->
Widget {
stateHandler
.handleEffects(
new ModelXCommand
.query()
.process())}

<<singleton>>
StateHandler

ViewModels

handleEffects()

<<singleton>>
Lifecycle

AppState
AppConfig
AppStateFilePersister

get_singleton()
init()
persist()
shutdown()

<<singleton>>
AppState

Lock<TodoListModel>

```
new() -> Self
mark_dirty(&self)
```

App Core
(Logic in Rust)

<<generated>>
Lifecycle-Extension

enum ProcessingError
enum Effect

(per Model)
enum ModelXCommand
enum ModelXQuery

(per Model)
Impl Cqrs for
ModelXCommand
ModelXQuery

```
fn process(self) ->
Result< Vec<Effect>,
ProcessingError >
```

<<domain>>
TodoListModel

Vec<String>

```
fn get_todos_as_string
  (&self) -> Vec<String>
```

<<domain>>
Lock<TodoListModel>

RustAutoOpaque<TodoListModel>

```
fn command(&self,
    payload: Type,) -> Result<(bool,
      Vec<TodoListEffect>),
TodoListProcessingError>

fn query(&self,
    payload: Type,) ->
Result<(Vec<TodoListEffect>),
TodoListProcessingError>
```

1) https://github.com/patmuk/flutter-UI_rust-BE-example

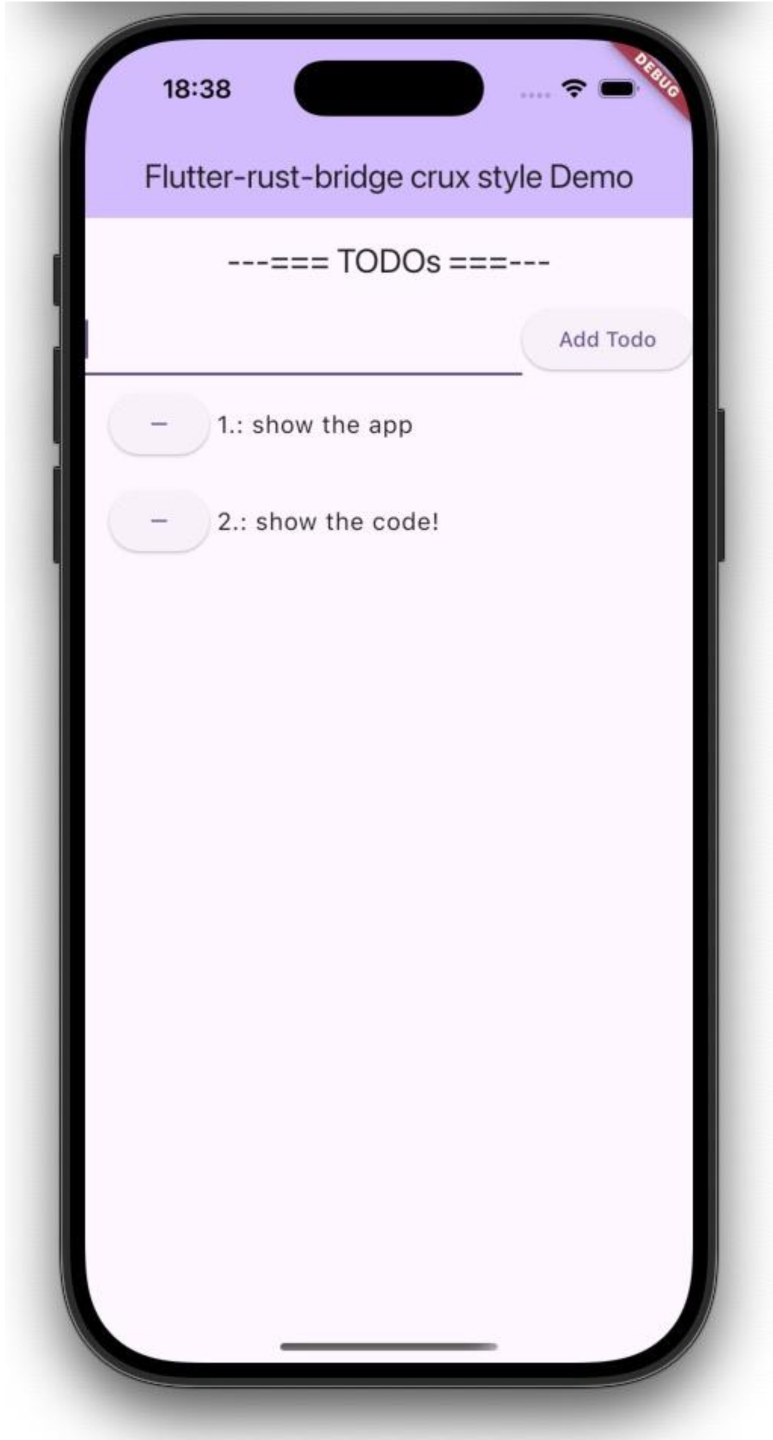
2) https://en.wikipedia.org/w/index.php?title=Command_Query_Responsibility_Segregation&oldid=1263578161

Implementation: Walkthrough

https://github.com/patmuk/flutter-UI_rust-BE-example

https://github.com/patmuk/generate_cqrs_api_macros

https://github.com/fzyzcjy/flutter_rust_bridge



Questions

https://github.com/patmuk/flutter-UI_rust-BE-example

https://github.com/patmuk/generate_cqrs_api_macros

https://github.com/fzyzcjy/flutter_rust_bridge

Patrick Mukherjee
patrick.mukherjee@gmail.com www.linkedin.com/in/patmuk