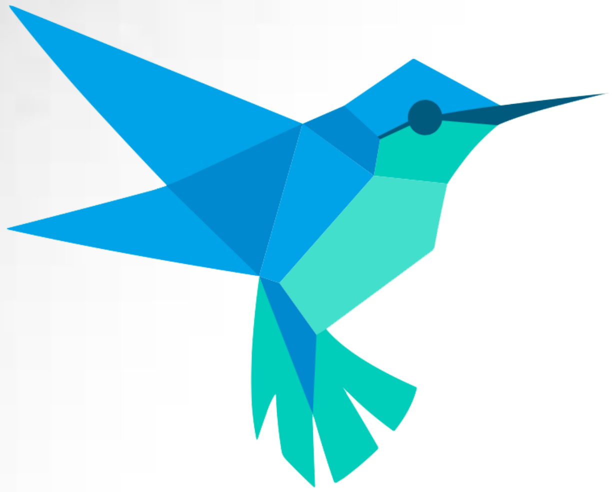
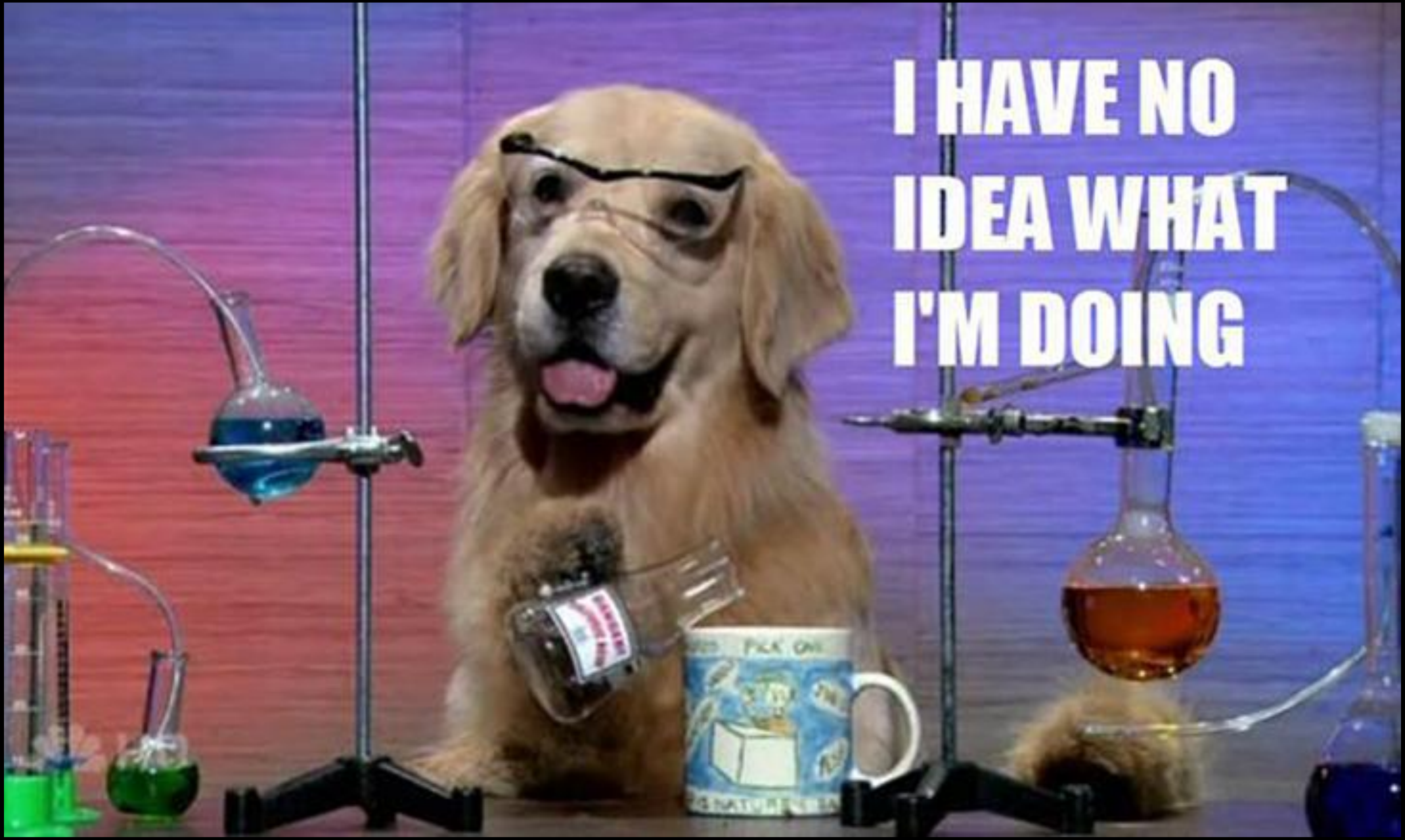


FRP & Architectures

CIKLUM
EMPOWERING COLLABORATION

How we think we develop

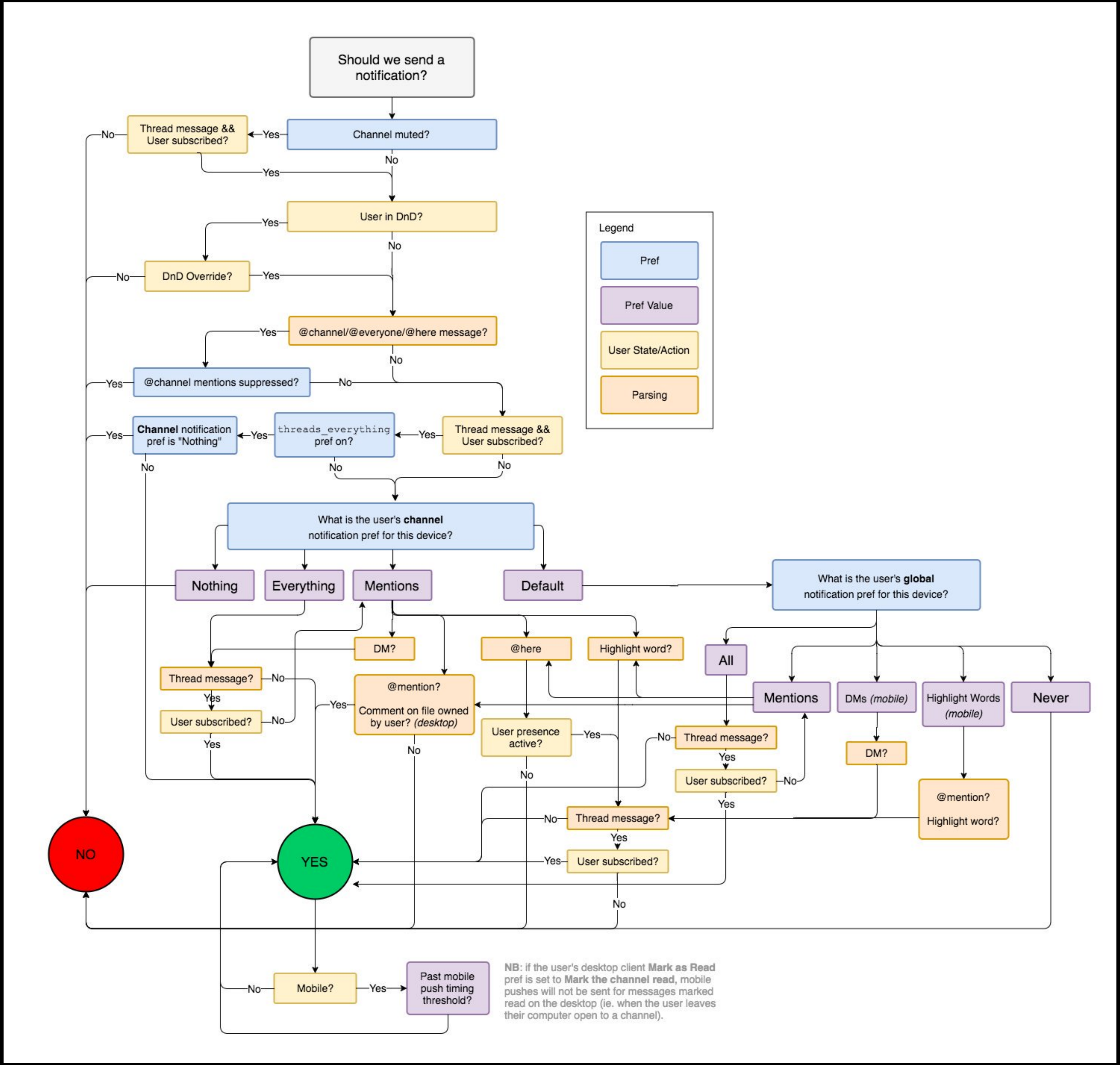
In real life



Why is that?

When Slack notification should be sent?

When Slack notification should be sent?


Should we send a notification?
Channel muted?
Thread message && User subscribed?
User in DnD?
DnD Override?
@channel/@everyone/@here message?
@channel mentions suppressed?
Thread message && User subscribed?
threads_everything pref on?
Channel notification pref is "Nothing"
What is the user's channel notification pref for this device?
Nothing
Everything
Mentions
Default
What is the user's global notification pref for this device?
DM?
@here
Highlight word?
All
Mentions
DMs (mobile)
Highlight Words (mobile)
Never
Thread message?
User subscribed?
@mention? Comment on file owned by user? (desktop)
User presence active?
Thread message?
User subscribed?
DM?
@mention? Highlight word?
Thread message?
User subscribed?
NO
YES
Mobile?
Past mobile push timing threshold?
Yes
No
Legend
Pref
Pref Value
User State/Action
Parsing

NB: if the user's desktop client **Mark as Read** pref is set to **Mark the channel read**, mobile pushes will not be sent for messages marked read on the desktop (ie. when the user leaves their computer open to a channel).

How to fight the complexity?

- Study
- Practice

Patterns simplify studying

- Provide a way to solve issues with a proven solution.
- Make communication more efficient

Architecture Pattern is just a pattern

Architecture Pattern != Architecture

Software architecture is about making fundamental structural choices that are costly to change once implemented.

Implementation matters

Before we dig into architecture

- Object-oriented Programming
- Functional Programming
- Reactive Programming

When simple is complex

```dart
class MySimpleClass {
  Object data;

  Type getDataType() =>
    data.runtimeType;
}
```

How to fix this?

```
class MySimpleClass {
  Type getDataType(Object data) => data.runtimeType;
}
```

```
  Type getDataType(Object data) => data.runtimeType;
```

This is a pure function

```dart
  Type getDataType(Object data) => data.runtimeType;
```

In other words

- No side effects
- No dependency or affect to the outside the scope
- Same output for the same input

Math is pure & predictable

```dart
int add(int a, int b) => a + b;
```

There is a trick to make complex apps with math

Partial evaluation demo

Partial evaluation

```
int add(int a, int b) => a + b;

int add2(int a) => add(a, 2);
```

```
final add = (int a, int b) => a + b;
final add2 = (int a) => add(a, 2);
```

When your app is math

- Everything is a function
- A lot of functions
- A mean **A LOT**

Scoping functions into correct locations helps keep maintainability

Architecture Patterns help us with scoping rules

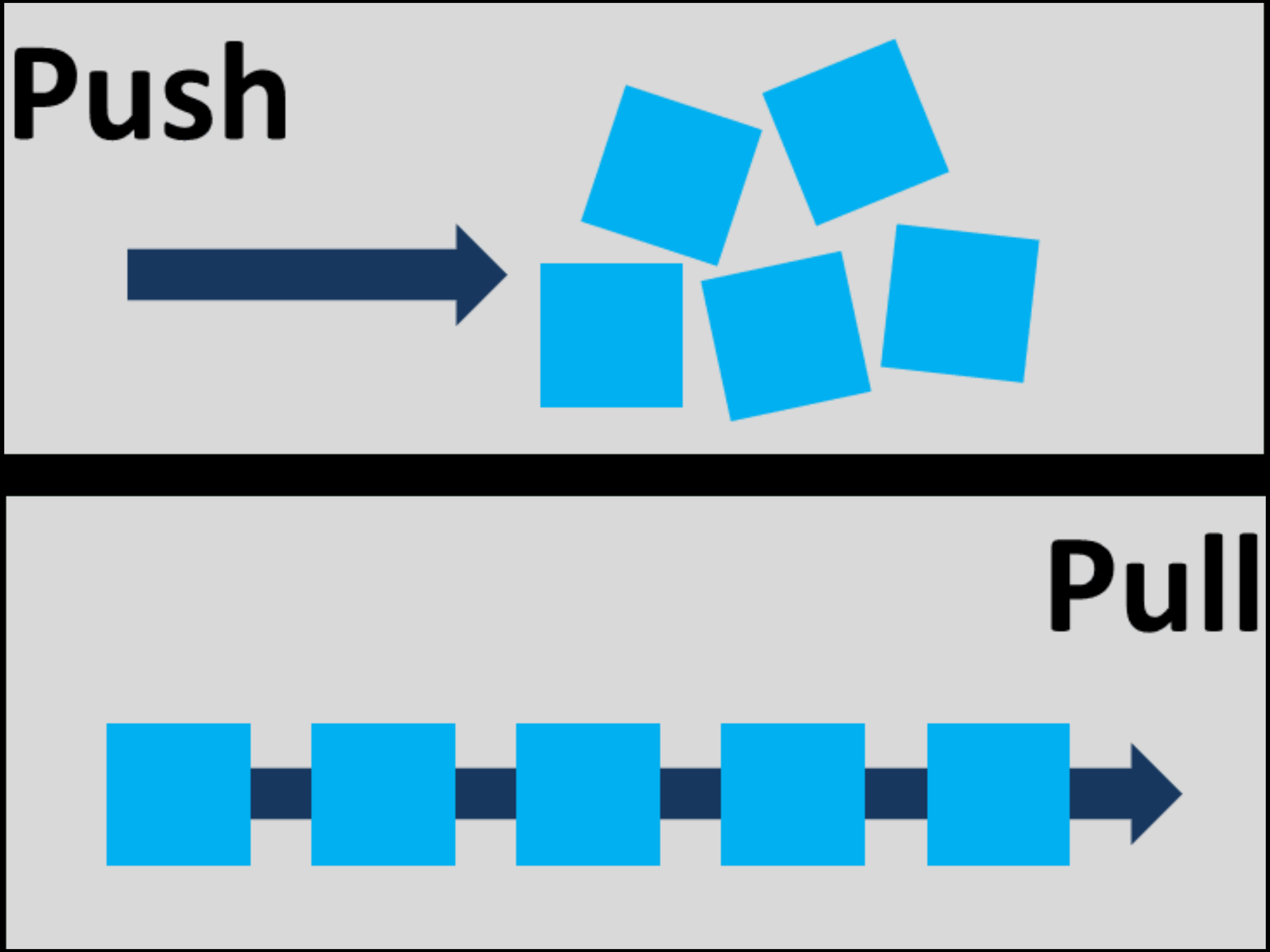
One more thing...

“Math” needs a trigger

- User
- Multiple Async operations (geolocation)
- OS events

Trigger is always outside, that is why we can not create “pure-math-apps”

How can we handle triggers?


Push
Pull

Flutter is Reactive (Pull)

But setState() is kind of half-Push

Demo

What is wrong with setState() ?

5 min break ⌚

One more widget
InheritedWidget

Provider is a better setState()

Demo

What is wrong with Provider ?

Redux as a global state

Demo

What is wrong with Redux ?

BLoC — many reactive Redux’es

Demo

What is wrong with BLoC ?

Architecture Patterns are neither good nor bad

There are problems you don’t understand with solutions for those problems, which you don’t like.

How to chose an architecture pattern?

- Try to avoid architecture decisions (architecture != architecture pattern)
- Understand the pattern implementation
- Solve your problem with the simplest solution, but remember — “you can not robber a bank with agile”

Unfortunately, intuition and experience matters

So many questions

- How to manage dependencies?
- How to navigate?
- How to test?
- …

In case pattern is flexible and app is testable — you’re good

IMPLEMENTATION MATTERS!

Homework

Make this app better

https://github.com/olexale/flutter_exam_app

Links

- https://fluttersamples.com
- https://github.com/rrousselGit/provider
- https://github.com/brianegan/flutter_redux
- https://bloclibrary.dev
- https://github.com/felangel/bloc