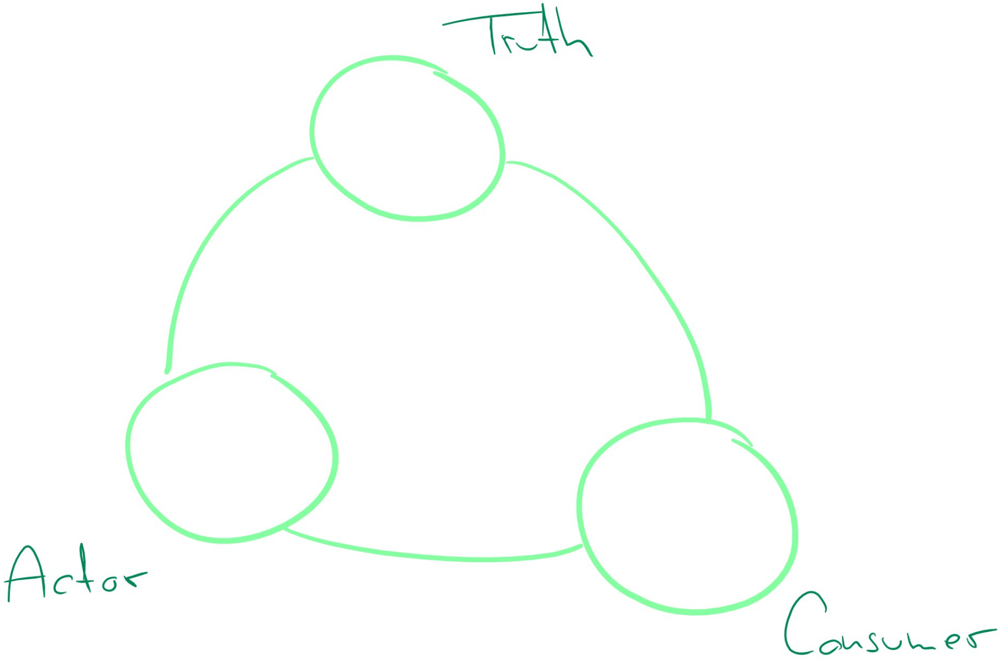
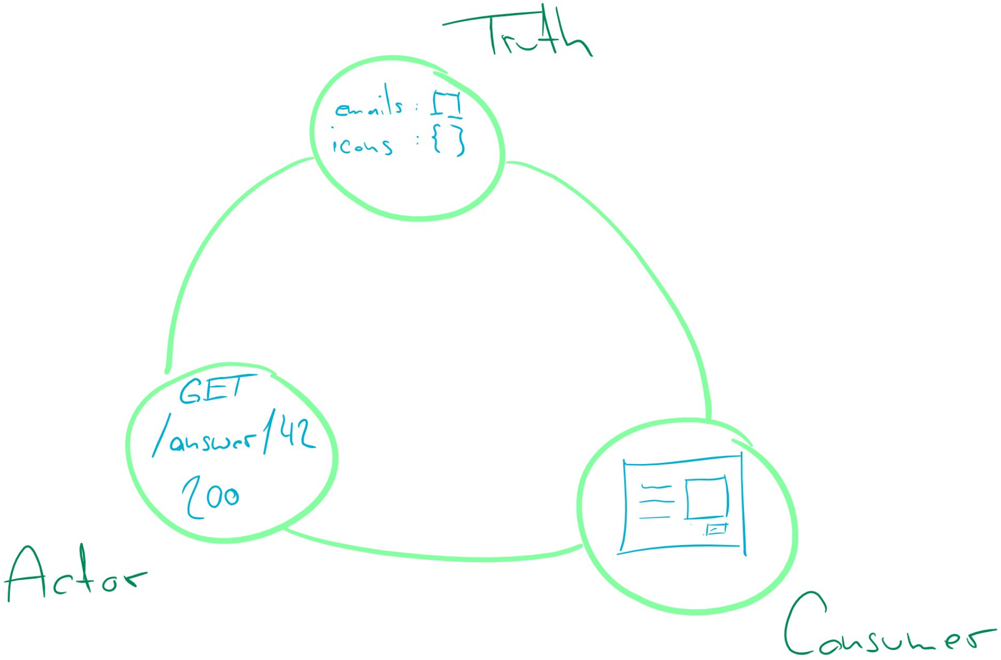
Redux and Android bouquet

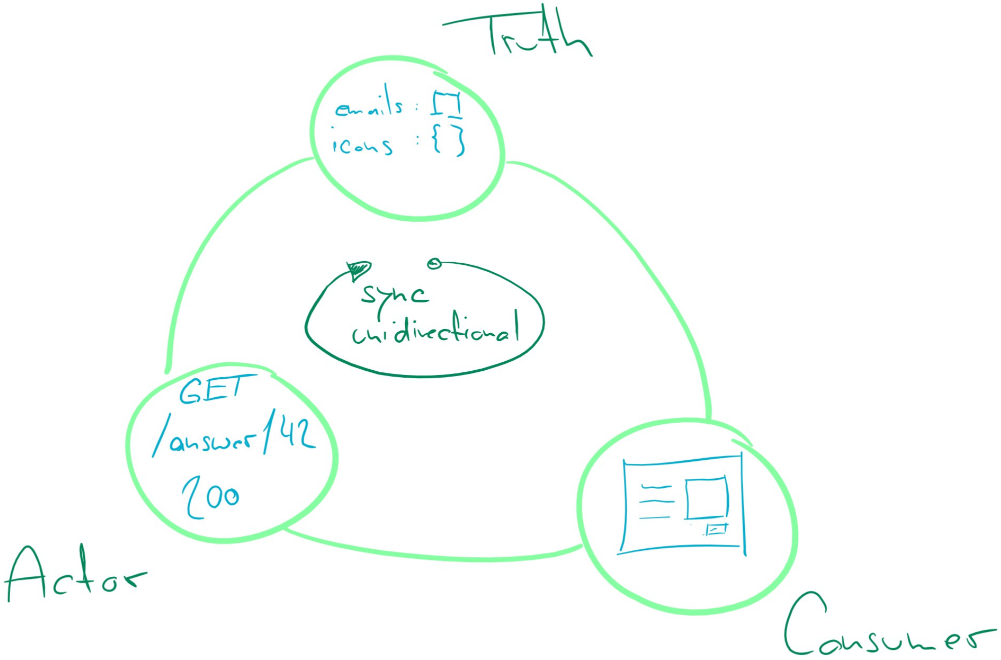
Redux



Redux



Redux


Truth
emails : []
icons : {}
sync
unidirectional
GET
/answer/42
200
Actor
Consumer

Principles

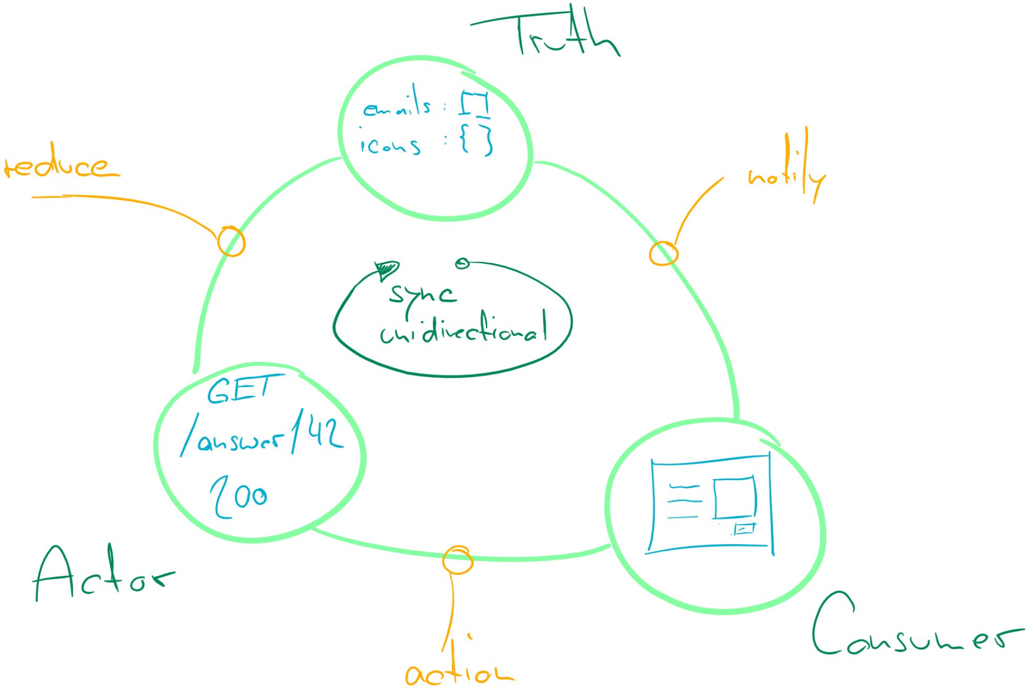
- Single Source of Truth
  - „The state of your **whole** application is stored in an object tree within a **single** store.“ redux.js.org
- State is read-only
  - „The **only** way to change the state is to emit an **action**, an object **describing** what happened.“ redux.js.org
- Changes are made with pure functions
  - „To specify **how** the state tree is transformed by actions, you write **pure** reducers.“ redux.js.org

Why

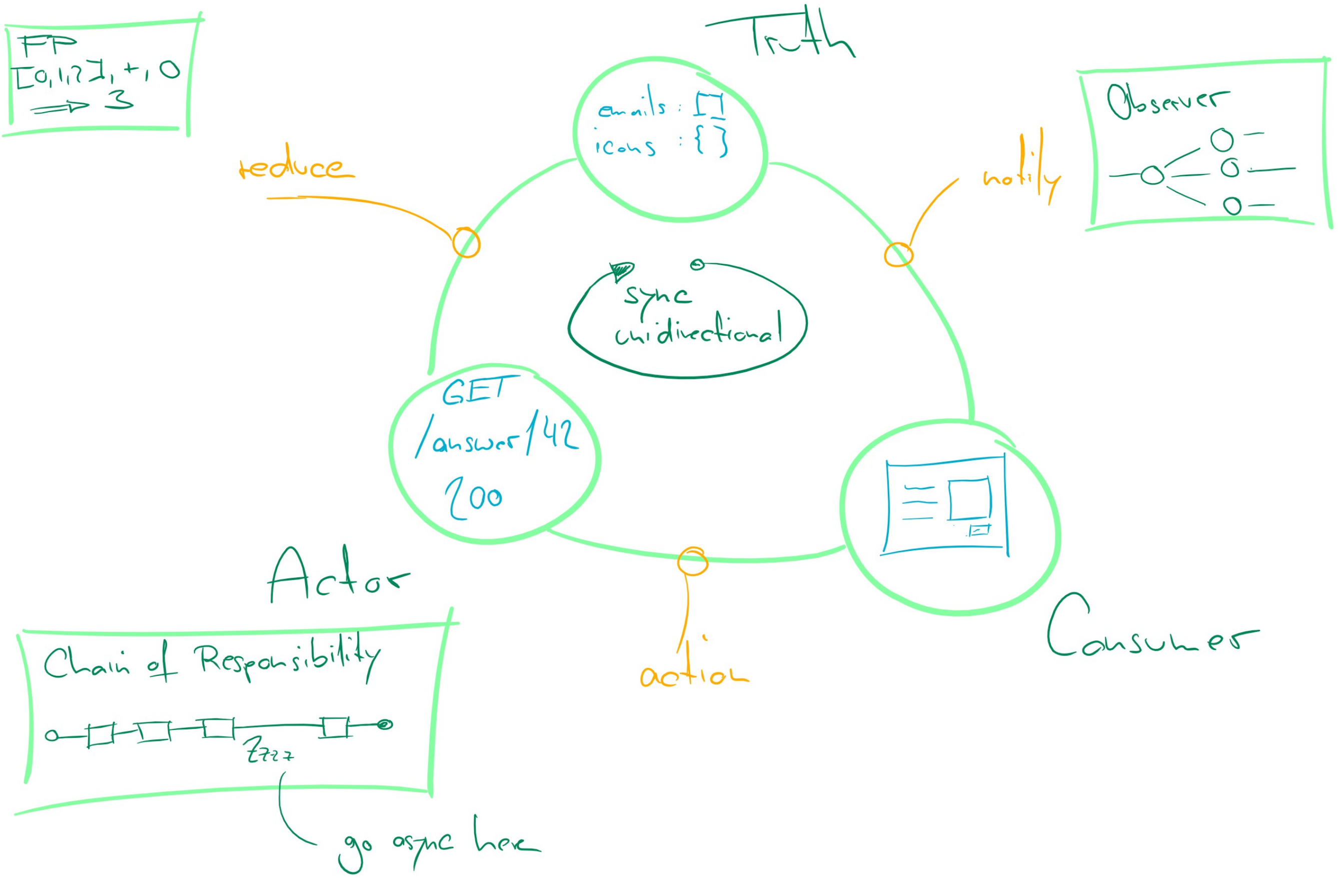
„Redux is a predictable state container for JavaScript apps."

redux.js.org

Redux


Truth
emails : []
icons : {}
notify
reduce
sync
unidirectional
GET
/answer/42
200
Actor
action
Consumer

Redux


FP
[0,1,2], +, 0
=> 3
reduce
Truth
emails : []
icons : {}
notify
Observer
sync
unidirectional
GET
/answer/42
200
action
Consumer
Actor
Chain of Responsibility
zzzz
go async here

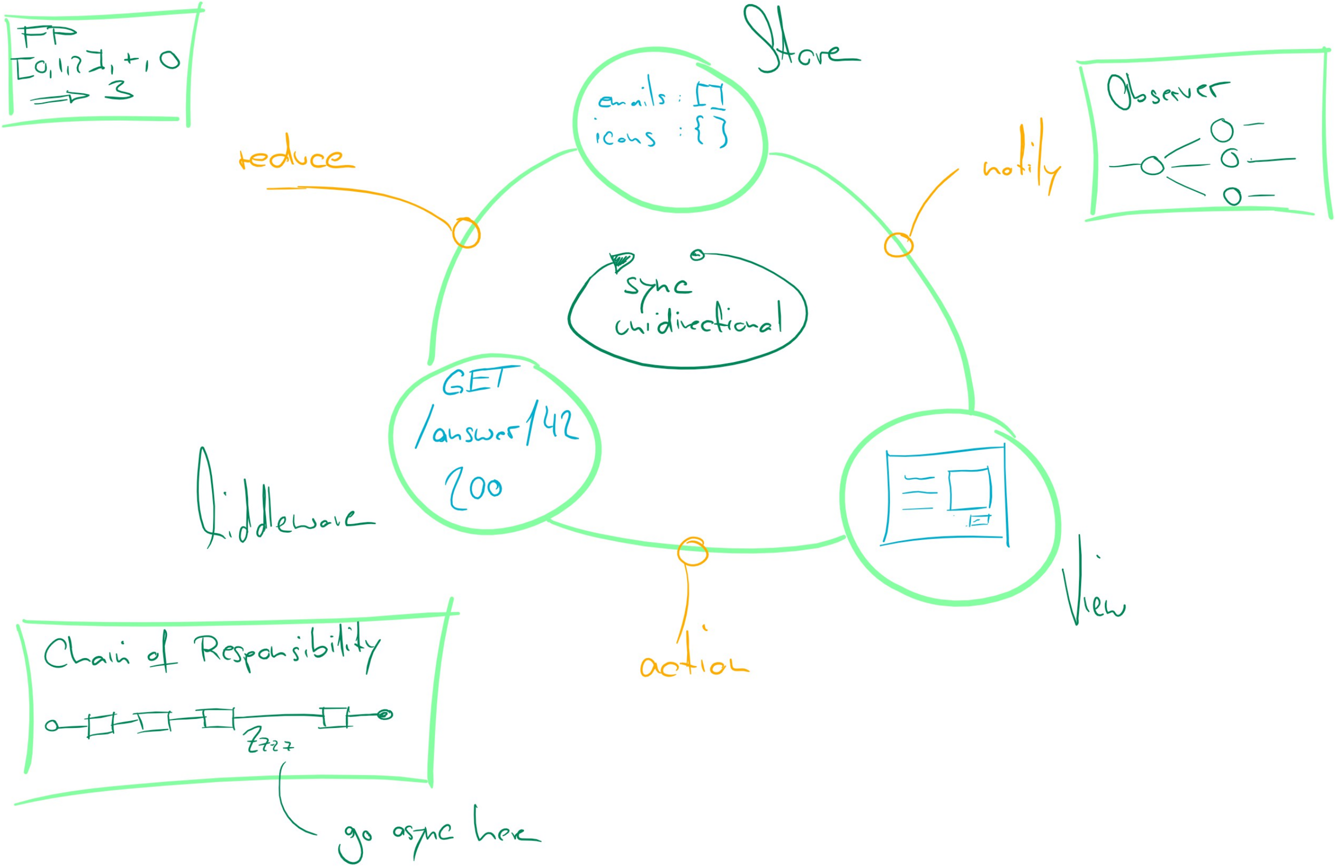
Building Block

- Action
  - „Actions **describe** the fact that *something happened*“ redux.js.org
  - „They are the **only source** of information for the store“ redux.js.org
  - „{ *type*: ADD_TODO, *text*: 'Build my first Redux app' }“ redux.js.org
- Action Creator
  - „Action creators are exactly that—functions that **create** actions“ redux.js.org
  - Should be defined in a **pure** manner
- Reducer
  - „Specify **how** the state tree is transformed by **actions**, you write **pure** reducers.“ redux.js.org
  - „Given the same arguments, it should calculate the next state and return it.
    No surprises. No side effects. No API calls. No mutations. **Just a calculation**.“ redux.js.org
  - *next*-state = [*action*].reduce (*callback*, *previous*-state)

Building Block

- Store
  - The **Store** contains the **complete** application/module's state
  - Entrypoint to **dispatch** actions → store.dispatch (todo ('welcome everyone'))
  - Entrypoint to **subscribe** to changes → layout.render (state)
- Middleware
  - **Guides** an action on the **chain** to the reducer
  - May **intercept**, **modify** or even **reject** actions
  - May contain business logic that **interpret** actions and their payloads
- Data Flow
  - „You can call **dispatch from anywhere** in your app, including components and XHR callbacks, or even at scheduled intervals.“ redux.js.org

Redux


FP
[0,1,2], +, 0
⟹ 3
reduce
Store
emails : []
icons : {}
Observer
notify
sync
unidirectional
GET
/answer/42
200
Middleware
action
View
Chain of Responsibility
Zzzz
go async here

Advanced Concepts

- Reducer **Composition**
  - Partition the storage layout into **separate parts**, each handled by a **separate reducer** function
- Normalization
  - Refer a thing by its unique ID → **consistent** and **atomic** updates for all views (complex apps)
  - Larger datasets may further effect the overall serialization performance
- Computed **Derived** Data
  - Todo-List + Filter → Todo-Filtered-List
  - Todo-Head@Id + Todo-Content@Id → Todo@Id
- **Testability**
  - Purity and Immutability → Simple In/Out Calculation
  - Testing Asynchronicity might be different → Redux Saga vs Redux Thunk Middleware

Oneway

```
function increment ()
    { type: 'inc' }
```

Oneway

```
function increment ()
    {type: 'inc'}

        store.dispatch(increment())
```

Oneway

```
function increment ()
    { type: 'inc' }


        store.dispatch (increment())

                reduce ( action, state = initial )
                    when ( action )
                        'inc'  ——▷  state.counter + 1
                        else   ——▷  state
```

Oneway

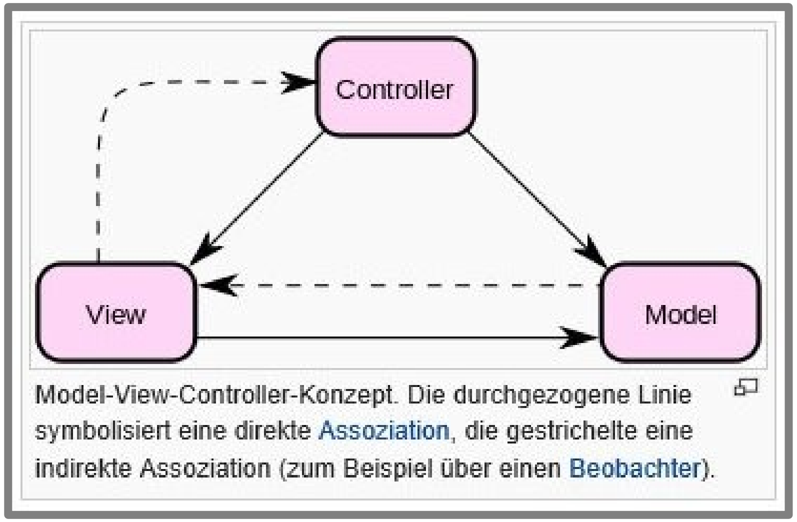
```
function increment ()
    {type: 'inc'}


        store.dispatch(increment())

            reduce ( action, state = initial)
                when (action)
                    'inc'  → state.counter + 1
                    else   → state

            notify ()
                view.render ( )
```

MVC, …

Some aspect from plain MVC

„Die Steuerung kann in manchen Implementierungen ebenfalls zu einem Beobachter des Modells werden, um bei Änderungen der Daten die Präsentation **direkt** zu **manipulieren**“ wikipedia

„Die Präsentationsschicht ist für die Darstellung der benötigten Daten aus dem Modell und die Entgegennahme von Benutzerinteraktionen zuständig. Sie **kennt sowohl** ihre Steuerung **als auch** das Modell“ wikipedia



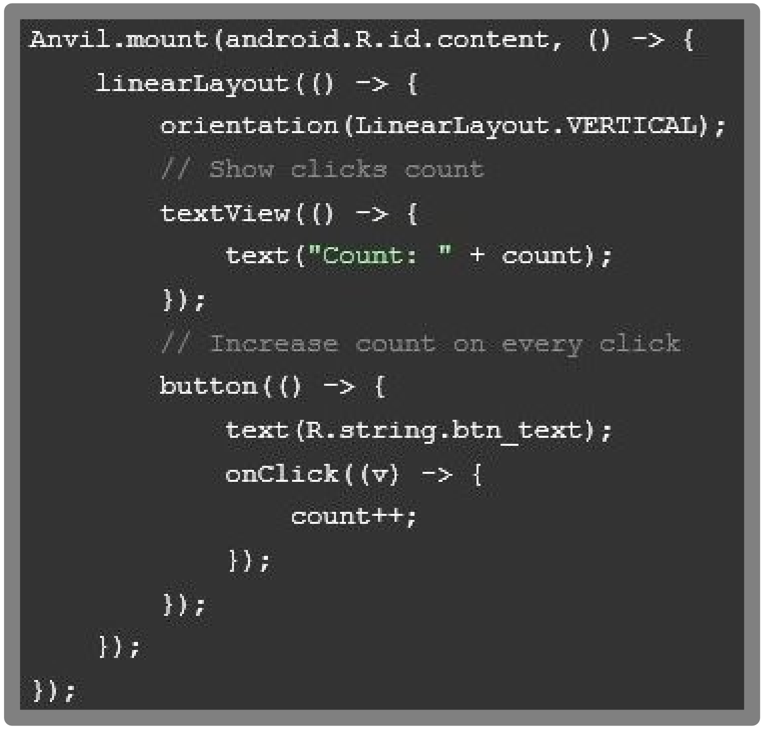
Model-View-Controller-Konzept. Die durchgezogene Linie symbolisiert eine direkte Assoziation, die gestrichelte eine indirekte Assoziation (zum Beispiel über einen Beobachter).

https://de.wikipedia.org/wiki/Model_View_Controller

Redux **decouples** those components more strictly and **enforces** this distinction by the chosen architecture, patterns and conventions.

Code

- Redux
  - Jedux → **119 LoC** (23.09) → https://github.com/trikita/jedux
- Layout
  - Code your views and data bindings instead of XML
  - Anvil → https://github.com/zserge/anvil
  - Anko → https://github.com/Kotlin/anko
- Demo
  - Calenope → https://github.com/synyx/calenope
  - Talalarmo → https://github.com/trikita/talalarmo
  - Slide → https://github.com/trikita/slide



```
Anvil.mount(android.R.id.content, () -> {
    linearLayout(() -> {
        orientation(LinearLayout.VERTICAL);
        // Show clicks count
        textView(() -> {
            text("Count: " + count);
        });
        // Increase count on every click
        button(() -> {
            text(R.string.btn_text);
            onClick((v) -> {
                count++;
            });
        });
    });
});
```

What else?

- Link Collection
  - http://redux.js.org/
  - https://developer.android.com/topic/libraries/data-binding/index.html
  - https://github.com/zserge/anvil-examples/tree/master/fragments/src/main/java/com/example/anvil/fragments
  - https://corner.squareup.com/2014/10/advocating-against-android-fragments.html
  - https://kotlinlang.org/
  - https://github.com/trikita/jedux
  - https://github.com/zserge/anvil
  - https://github.com/Kotlin/anko
  - https://en.wikipedia.org/wiki/Thunk
  - https://github.com/yelouafi/redux-saga
  - https://github.com/gaearon/redux-thunk

synyx GmbH & Co. KG
Open Source Solutions

Gartenstraße 67
76135 Karlsruhe

+49 721 203823-0
+49 721 203823-12

✉ info@synyx.de
🖥 www.synyx.de
blog.synyx.de

done (thanku ())