**PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG NÂNG CAO**

**ĐỒ ÁN NHÓM**

**Tìm hiểu & báo cáo về Recoiljs & Redux**

**MSSV1 : 1612209**

**Họ và tên : Nguyễn Hữu Hòa**

**MSSV2 : 1612210**

**Họ và tên : Phạm Ân Hòa**

**MSSV3 : 1612330**

**Họ và tên : Nguyễn Văn Lân**

**Email : 1612209@student.hcmus.edu.vn**

**SĐT : 0982327118**

C:\Users\tdqua_000\Dropbox\SS-Slides\DeCuong-CDIO\Template CDIO v4.2\Templates\Hinh anh\LogoTruong.png

Khoa Công nghệ thông tin

Đại học Khoa học tự nhiên TP HCM

# MỤC LỤC

[MỤC LỤC 1](#_Toc48411748)

[1 Tìm hiểu Recoiljs 2](#_Toc48411749)

[**1.1.** **Recoiljs là gì?** 2](#_Toc48411750)

[d. Khi nào cần sử dụng Recoiljs 2](#_Toc48411751)

[e. Yêu cầu khi học recoiljs 2](#_Toc48411752)

[**1.2.** **Các thành phần chính của Recoilj** 2](#_Toc48411753)

[a. RecoilRoot 3](#_Toc48411754)

[b. Atom 3](#_Toc48411755)

[c. Selector 4](#_Toc48411756)

[**1.3.** **Nhận xét** 4](#_Toc48411757)

[2 Tìm hiểu về Redux 5](#_Toc48411758)

[**2.1** **Redux là gì? Kiến trúc nó ra sao?** 5](#_Toc48411759)

[**2.2.** **Khi nào thì cần sử dụng redux?** 6](#_Toc48411760)

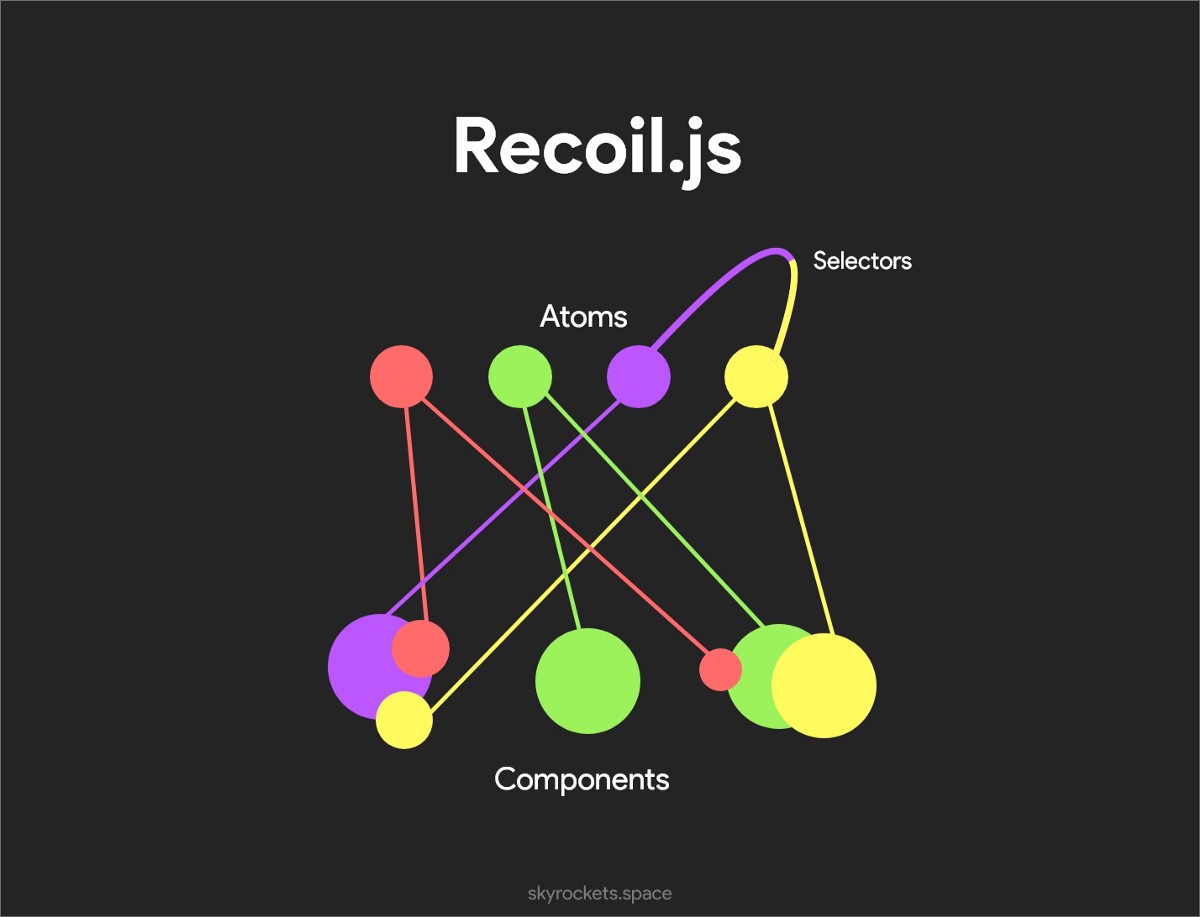
[**2.3.** **Setup Redux trong React Native với React hooks** 6](#_Toc48411761)

[3 So sánh Recoiljs & Redux 8](#_Toc48411762)

[4 Tài liệu tham khảo 9](#_Toc48411763)

# Tìm hiểu Recoiljs

* 1. **Recoiljs là gì?**
     + - 1. Khái niệm
* Về khái niệm : Recoiljs là một thư viện quản lí state dành cho React (theo recoiljs.org)
* Về hình thức: Recoijs cũng quản lí các state tương tự như ContextAPI, Redux, mobx
* Do chính Facebook phát triển.
* Cú pháp đơn giản, dễ sử dụng.
  + - * 1. Ưu điểm
* Tạo ra một công cụ quản lí state đơn giản, cú pháp trực quan giống như state đơn thuần của react.
* Hỗ trợ quản lí state bất đồng bộ.
* Cung cấp khả năng lưu trữ state lâu dài (persist state).
  + - * 1. Tại sao Recoiljs ra đời trong khi vẫn có Context hay Redux
* Prefer built-in lib hơn là external lib (Redux).
* Context API thì bị limit ở việc store được 1 giá trị và ảnh hưởng tới toàn bộ cây component ở phía dưới nó khi nó thay đổi.
* Component state thì phải truyền lên truyền xuống nhiều lần
  + - * 1. Khi nào cần sử dụng Recoiljs
* Nên dùng thử nghiệm.
* Đừng vội apply vô product, đợi xem đánh giá của cộng đồng đã.
  + - * 1. Yêu cầu khi học recoiljs
* Documents Recoiljs: <https://recoiljs.org>
* ReactJs
* ReactHook
  1. **Các thành phần chính của Recoilj**



*H1.Tổng quan về Recoiljs*

* + - * 1. RecoilRoot
* Để sử dụng được recoil, ứng dụng cần được chứa trong một component có tên là RecoilRoot. Điểm này khá là giống như Provider trong ContextAPI.
* Ví dụ:

<RecoilRoot>

<MainScreen />

</RecoilRoot>

Khi đó các component con (MainScreen và các component con của MainScreen) đều có thể truy cập được cái Atom (giống state trong react).

* + - * 1. Atom
* Recoil định nghĩa việc lưu dữ liệu bằng đơn vị riêng, gọi là atom. Nó khá giống với state trong react nhưng đặc biệt hơn là có thể truy cập ở mọi nơi miễn là nó nằm trong RecoilRoot, không cần truyền store nào cả.
* Mỗi Atom được định nghĩa bằng key và value, trong đó key là duy nhất.
* React component đều có thể truy cập và thay đổi dữ liệu chứa trong atom.
* Ví dụ:

const name= atom({

key: ‘userName’,

default: ‘tìm hiểu recoiljs’

})

Cách sử dụng:

* Get + Set:

const [key, setKey] = useRecoilState(nameState);

* Get:

const key = useRecoilValue(nameState);

* Set:

const setKey = useSetRecoilState(nameState);

* + - * 1. Selector
* Là một dạng function, trả về dữ liệu đã được tính toán.
* Ví dụ: Selectors nhận vào 1 key (là dữ liệu số điểm của từng môn chẳng hạn) và trả ra 1 giá trị là điểm tổng của các môn hoặc điểm trung bình.
  1. **Nhận xét**
* Việc sử dụng state khá đơn giản vì nó giống với useState của react, nhưng tiện hơn ở chỗ là có thể truy cập ở bất cứ đâu.
* Tương tác giữa component và state khá đơn giản và dễ hiểu.
* Khá mới, còn chưa được áp dụng rộng rãi nên chúng ta chỉ nên dừng lại ở việc tìm hiểu, áp dụng đơn giản. Những product chính hoặc lớn chưa nên áp dụng, đợi phản hồi của cộng đồng xem như thế nào đã.

# Tìm hiểu về Redux

* 1. **Redux là gì? Kiến trúc nó ra sao?**
* Redux là một thư viện js được dùng để quản lý state trong các javascript apps, mà state này có thể dự đoán được.
* Redux sử dụng kiến trúc uni-directional data flow

A picture containing clock

Description automatically generated

* Nói sơ qua về các thành phần
  + - * **Store** gồm có:
        + State: là dữ liệu hiện tại được lưu trên state.
        + Reducer: là hàm biến đổi state cũ thành state mới.
        + Dispatcher: quản lý middlewares và chuyển dữ liệu xuống reducer.
      * **Action** là plain javascrpit object mô tả hành động.
  1. **Khi nào thì cần sử dụng redux?**
* Khi 1 dữ liệu được sử dụng ở nhiều nơi.
* Có hỗ trợ các chức năng như redo/undo thì nên tích hợp redux vào project
* Cần cache dữ liệu để tái sử dụng cho những lần sau.
* …

Vậy Redux có phải chỉ để dùng với reactjs hay không?

Redux sử dụng được cho các javascript apps.

Ví dụ như:

* + ReactJS
  + Angular
  + VueJs
  + Pure javascript App
  + Other javascript App
  1. **Setup Redux trong React Native với React hooks**
* Với việc React Hook đã được sử dụng rộng rãi, khả năng xử lý state và side effect của Component đã quá thông dụng trong function component.
* Cài đặt redux yarn add redux react-redux lodash.remove
* Thêm action type và creator:

Khi Redux kiểm soát state của cả ứng dụng, state được diễn đạt như là một Javascript object. Hãy hiểu rằng object này là read-only, chúng ta không thể tự thay đổi được nó. Mà phải dùng action để làm.

* + Action là các event trong redux. Nó có thể là việc bấm nút, timer hoặc network request.
  + Để bắt đầu, hãy tạo thư mục redux trong src. Sau đó tạo file noteApp.js.
  + Vì ứng dụng này làm công việc liên quan đến note, trong file vừa mới tạo, hãy bắt đầu với 2 action như sau

// Action Types

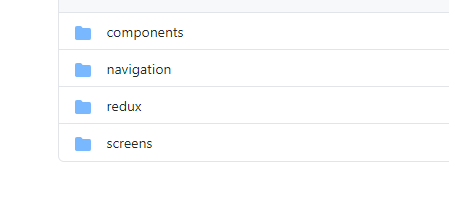
export const ADD\_NOTE = 'ADD\_NOTE'

export const DELETE\_NOTE = 'DELETE\_NOTE'

* Thêm Reducer
  + Phần nhận những Action được gọi là reducer. Bất cứ khi nào action được kích hoạt, state của ứng dụng sẽ thay đổi. Việc quản lý state này dựa vào reducer.
  + Reducer là một pure function tính toán state tiếp theo dựa vào state trước nó. Nó luôn trả về cùng một output nếu state không thay đổi. Nó có 2 tham số state và action và phải luôn trả về state.
  + State khởi tạo sẽ là một array trống. Bạn hãy import thêm remove trong lodash.remove mà bạn đã thêm trước đó.
* Tùy chỉnh Redux store

Store là một object kết nối các action với reducer. Nó cung cấp và giữ các state ở cấp độ ứng dụng thay vì trong các component.

* Cấu trúc thư mục của project



# So sánh Recoiljs & Redux

|  |  |  |
| --- | --- | --- |
|  | Recoiljs | Redux |
| Ưu điểm | - Quản lí việc lưu trữ nhanh gọn, lâu dài.  - Cú pháp đơn giản, dễ học.  - Quản lí state bất đồng bộ. | - Sử dụng được dữ liệu ở nhiều nơi  - Áp dụng tốt trong các ứng dụng cần cache dữ liệu nhiều hoặc có sử dụng redo/undo,..  - Thích hợp với mọi loại app javascript, không chỉ riêng reactjs |
| Nhược điểm | - Mới mẻ.  - Chưa có nhiều phản hồi. | - Khá là nặng. |
| Áp dụng | - Chỉ nên áp dụng thử nghiệm.  - Đợi phản hồi chung của cộng đồng & trả nghiệm cá nhân.  - Áp dụng vào các project nhỏ.  - Có kiến thức căn bản về hooks. | - Áp dụng được các project lớn & nhỏ.  - Được sử dụng rộng rãi và là một phần quan trọng của React. |

# [Tài](https://docs.unity3d.com/Packages/com.unity.package-manager-ui@2.0/manual/index.html) liệu tham khảo

* <https://viblo.asia/p/recoiljs-phan-mot-cai-dat-va-su-dung-co-ban-WAyK82enlxX>
* <https://recoiljs.org/>
* <https://recoiljs.org/docs/introduction/installation>
* <https://www.youtube.com/watch?v=CLbxzwG-yIo>
* <https://github.com/cuongnh-1139/recoil-demo>