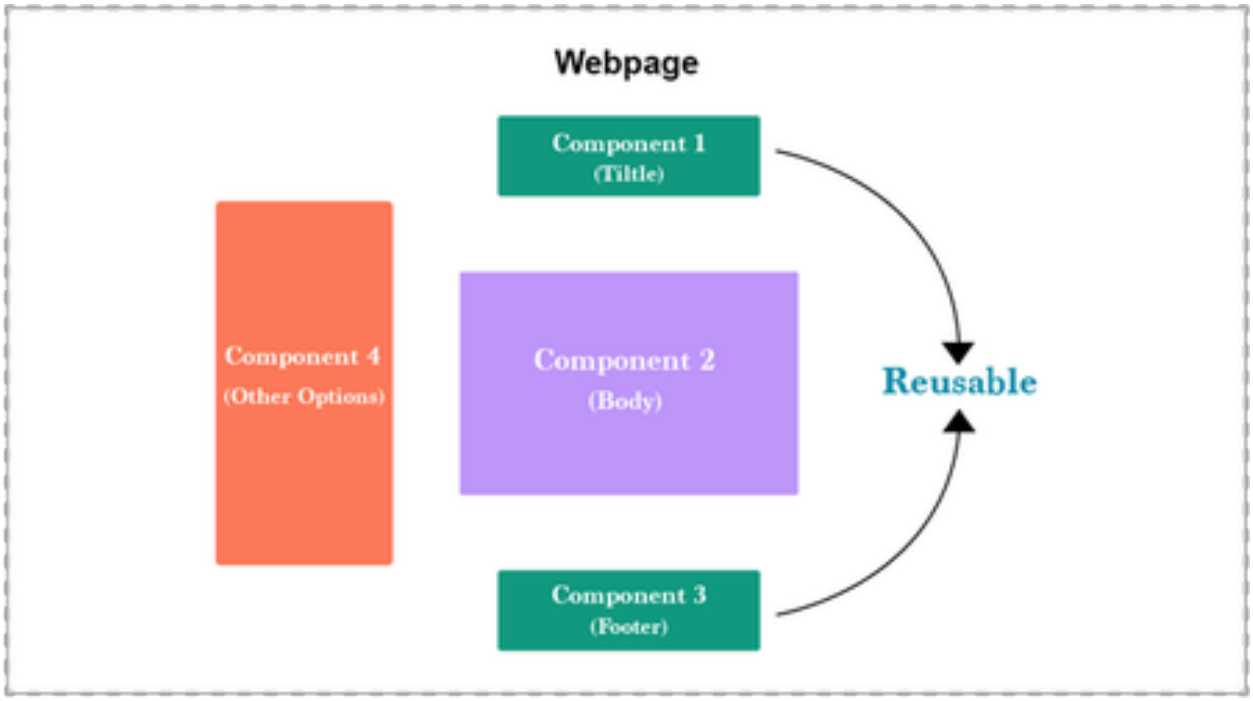
Reactjs - Redux saga for beginners

Tóm tắt lý thuyết

Giới thiệu Reactjs:

- Reactjs là một thư viện javascript được phát triển bởi đội ngũ Facebook dùng để thiết kế giao diện website theo hướng single page application.
- Sử dụng Reactjs theo hướng component và có thể tái sử dụng lại được.
- 1 component là một class hoặc một function chứa các hàm xử lý và một hàm trả về giao diện của nó.
- 1 trang view gồm 1 hoặc nhiều component kết hợp lại với nhau.



Ưu và nhược điểm:

Ưu điểm:

- Sử dụng cây DOM ảo để dánh dấu cập nhật rồi sau đó cập nhật trực tiếp lên DOM thật => Không cập nhật lại toàn bộ => Cải thiện hiệu suất.
- Kết hợp sử dụng cú pháp ES6 và JSX để code dễ dàng hơn.
- Nhiều tool hỗ trợ, debug dễ dàng, cộng đồng reactjs dev cực lớn.
- Dễ viết testcase.

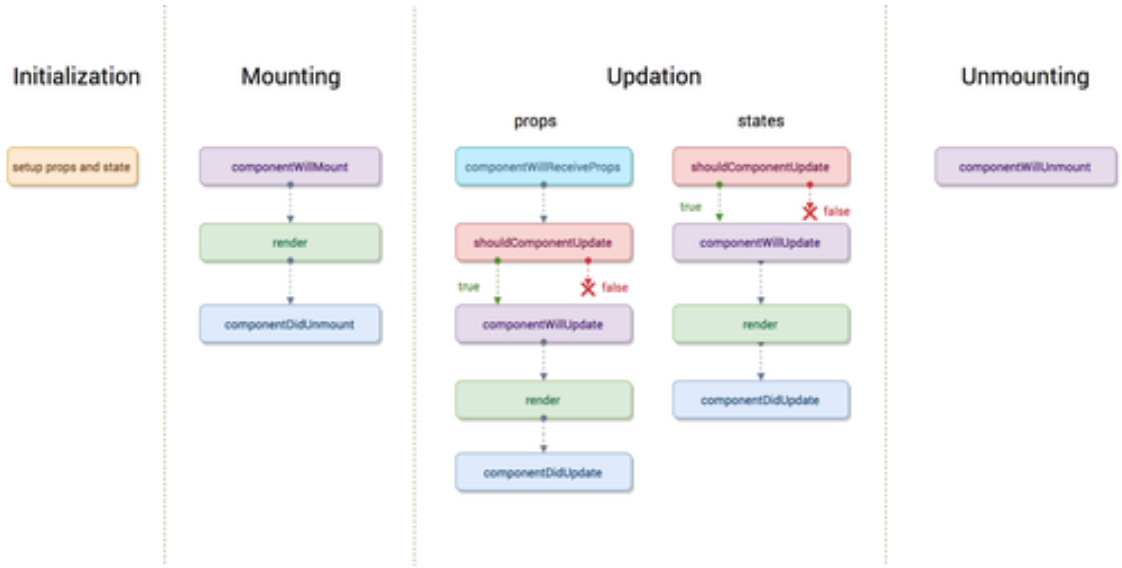
- Đã nhiều ứng dụng lớn thành công nhất định với Reactjs.

Nhược điểm:

- Dễ tiếp cận nhưng không dễ để đạt level master.
- Vì là library không phải framework hoàn chỉnh nên phải kết hợp với các library khác.
- Phải config nhiều, tuy nhiên có các starter kit đã hỗ trợ việc config khi init project.

Các khái niệm cơ bản:

- Reactjs làm việc với STATE và PROPS để quyết định component có render lại hay không, lưu ý là chỉ component sử dụng STATE và PROPS đó mới check để render lại (đèn nhà ai nấy sáng).
- STATE của một component là một object có thể thay đổi được thông qua hàm setState của class đó.
- PROPS của một component là một giá trị bất biến và được truyền vào từ parent component (có thể là 1 STATE của parent component và tất nhiên muốn thay đổi giá trị này thì parent component phải truyền vô 1 callback).
- Hình bên dưới là life cycle của 1 component.



- Theo kinh nghiệm:
    1. Call các API ban đầu ở componentDidMount
    2. Nếu là một container render dùng để render detail thì phải gọi lại API load detail trong componentWillReceiveProps khi **current location key** khác **next location key** hoặc **current id** khác **next id**

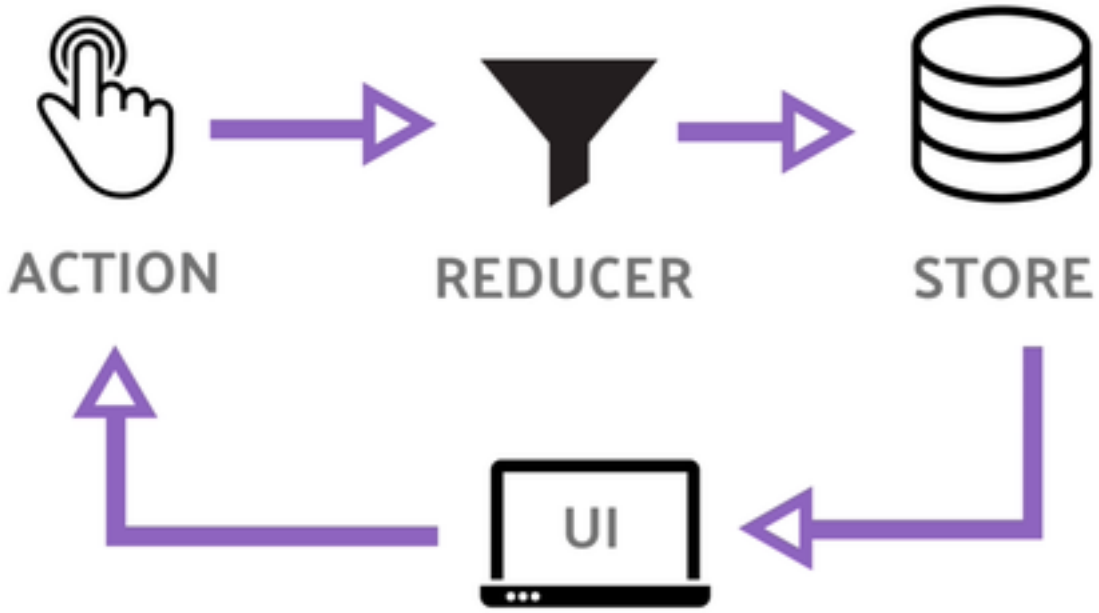
```
class App extends React.PureComponent {
    componentDidMount() {
            this.props.getDetail(this.getStuffId());
    }

    componentWillReceiveProps(nextProps, nextState) {
```

```
9.            if (this.props.location.key !== nextProps.location.key
   ||
10.                           this.getStuffId() !==
   this.getStuffId(nextProps)) {
11.
      this.props.getDetail(this.getStuffId(nextProps));
12.                   }
13.           }
14.
15.           getStuffId(props = this.props) {
16.                   return props.match.params.stuffId;
17.           }
18.
19.           render() {
20.                   return (
21.                           <div>Hello World</div>
22.                   );
23.           }
   }
```

24. Hạn chế sử dụng state (vì đã có redux)

Giới thiệu về Redux:

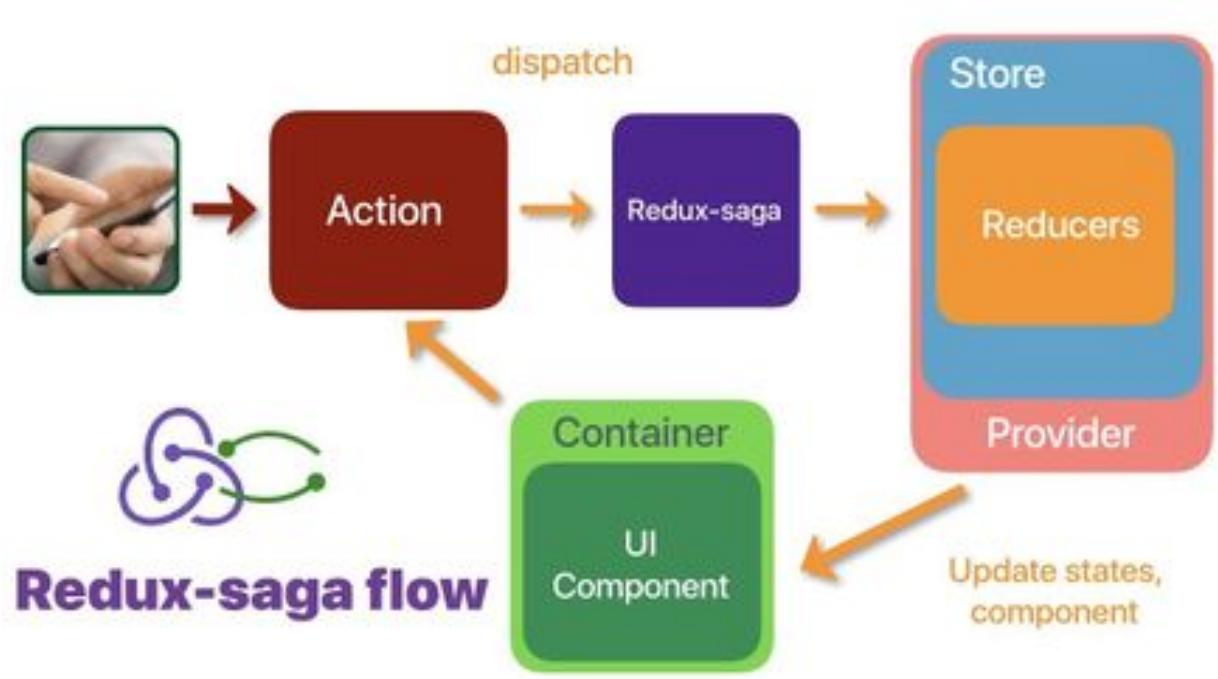
- Việc mỗi component chứa state của nó là quá nhiều và cồng kềnh, ví dụ muốn load 1 cái list user ở 2 component khác nhau khải khởi tạo 2 state và các hàm xử lý api liên quan => duplicate code => cách khắc phục là cần 1 chỗ tập trung để lưu dữ liệu như là 1 global state (gọi nó là recuder).
- Reducer là 1 đơn vị trong store, 1 store chứa nhiều reducer được phân biệt bởi các key.



- component sử dụng reducer (lấy từ trong store) như là một props, theo lý thuyết props không thể đổi giá trị state của parent component => component cần dispatch action event để thay đổi giá trị của reducer.

Bất đồng bộ trong redux:

- Redux là cơ chế đồng bộ, nếu sử dụng promise call api thì sẽ xảy ra nhiều callback lồng với nhau (callback hell). Nên có nhiều thư viện xử lý bất đồng bộ cho redux để code clean hơn và dễ dàng xử lý logic hơn như Redux Thunk, Redux Saga, Redux Observable ...
- Redux Saga tích hợp vào redux với vai trò là 1 middleware, nó nằm sau dispatch action và cũng có thể gọi lại dispatch action.
- 1 saga được tạo phải được subcribe tại store thì mới sử dụng được saga đó, giống như 1 reducer được tạo phải gắn vào store (cơ chế observable).



Lưu ý: *Ngoài ra cần sử dụng react router để định tuyến cho các containers trong react và kết hợp với redux để quản lý location, history trong store.*

Thực hành với CREATE REACT APP

Yêu cầu cơ bản:

- Có một máy tính
- Hiểu cơ bản lập trình
- Hiểu biết về ngôn ngữ javascript ES6,7,8
- Thông thạo html css

Tham khảo tutorial sau:

https://medium.com/@lavitr01051977/make-your-first-call-to-api-using-redux-saga-15aa995df5b6