



Redux in 10 minutes

Yut

Growth Session #21 - January 24-25 2019

- What is Redux?
- The Redux concepts
 - View
 - Action Creator
 - Action and Payload
 - Store
 - Middleware
 - Reducer
 - Selector



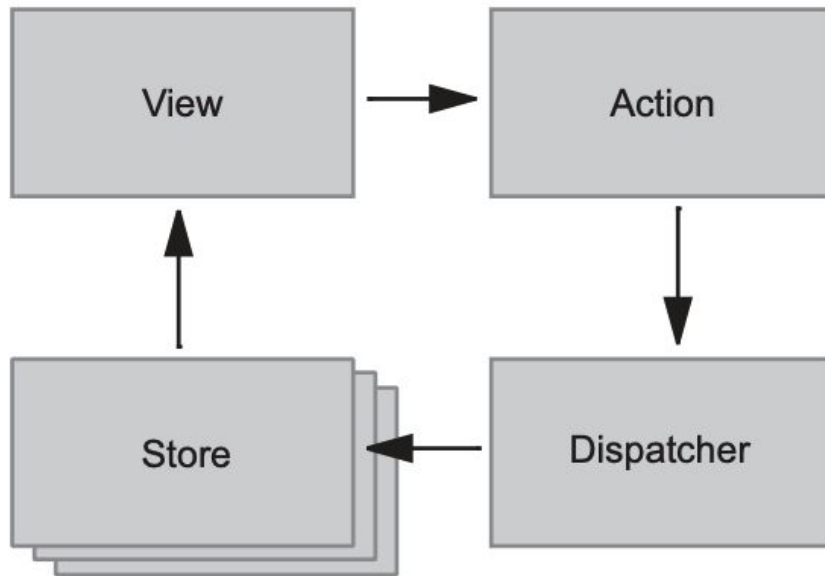
What is Redux?

What is Redux?

- Redux is a state management library.
- When working on the UI, especially if you follow a component-based architecture, you need to think and design your component in terms of “state”.
- The component state or UI state is an object that determines what component will look like and how it will behave.
- When you need to learn Redux, you need to know also the concepts of the Flux architecture but Redux is NOT 100% Flux architecture.
- Redux is easier than Flux.

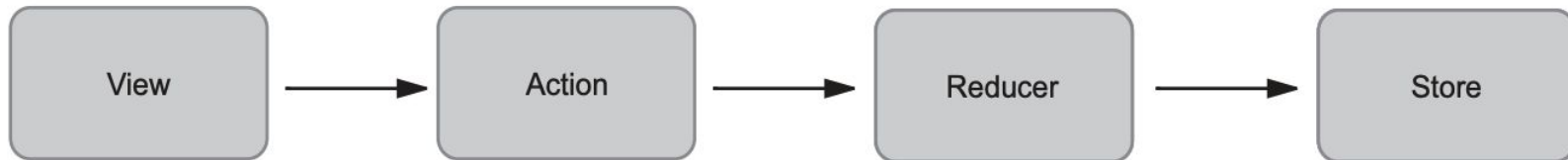
The Flux architecture

The Flux architecture is a design pattern that solve the complexity of the UI state with concept of “one-way” data flow.



Redux is a Flux-inspired library.

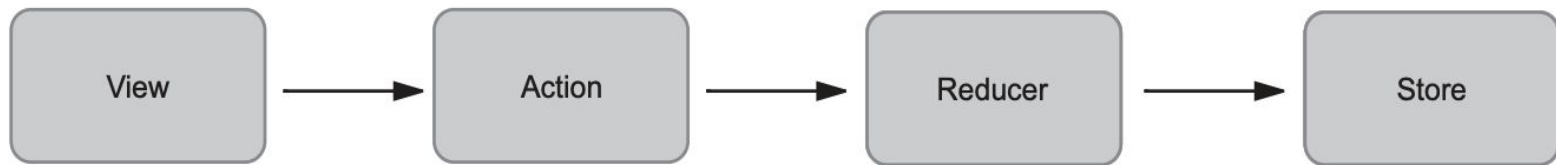
- Redux is a Flux-inspired library. It does NOT cover 100% Flux architecture patterns.
- It does NOT have dispatcher or many stores. It has only one store and replace the Dispatcher with Reducer.



The Redux concepts

The Redux concepts

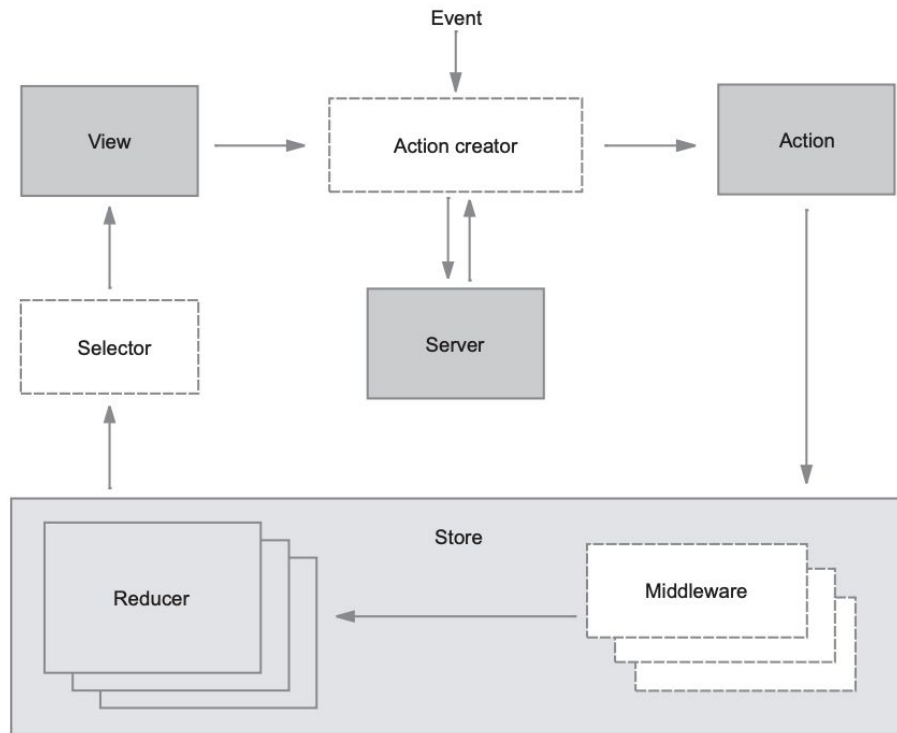
The basic concepts in Redux:



The Redux concepts

The complete concepts in Redux:

- View
- Action creator
- Action and Payload
- Store
 - Middleware
 - Reducer
- Selector



View

The Redux concepts: View

- The view in Redux is just the “View”.
- It may be plain HTML or vanilla JavaScript which listens to the user interaction.
- It may be a React component.
- It may be a Vue component.
- View is just a UI view.

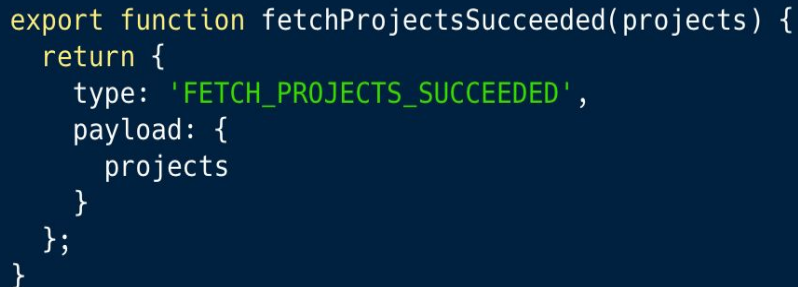
Action Creator

The Redux concepts: Action Creator

- The action creator is just a function. It is like a controller in the MVC architecture.
- The action creator is a place to perform business logic and change the previous state to the next state.
- The action creator has two types:
 - Synchronous action creator
 - Asynchronous action creator

The Redux concepts: Synchronous Action Creator.

A synchronous action creator is just a function that return the “Action” object.




```
export function fetchProjectsSucceeded(projects) {  
  return {  
    type: 'FETCH_PROJECTS_SUCCEEDED',  
    payload: {  
      projects  
    }  
  };  
}
```

The Redux concepts: Asynchronous Action Creator.

- The asynchronous action creator is a function that returns another function.
- This returned function will receive the “dispatch” method as an argument.
- This returned function will handle the asynchronous process e.g. call asynchronously to the server.
- The returned function will be used by the redux-thunk middleware.

The Redux concepts: Asynchronous Action Creator.



```
export function createTask({ title, description, status = 'Unstarted' }) {  
  return function(dispatch) {  
    api.createTask({ title, description, status }).then((response) => {  
      dispatch(createTaskSucceeded(response.data));  
    });  
  }  
}
```


Action and Payload

The Redux concepts: Action and Payload

- The action is just an object that contain only two attributes:
 - type
 - payload
- The action object will returned from the “Synchronous action creator” only.
- The action object will used by Reducer for update the next state in the store.

Store

The Redux concepts: Store

- Store is a place for keep the all states of the app.
- In Redux, the app will have only one store.

```
{
  "projects": [
    {
      "id": 1,
      "name": "Short-Term Goals"
    },
    {
      "id": 2,
      "name": "Long-Term Goals"
    }
  ],
  "tasks": [
    {
      "id": 1,
      "title": "Learn Redux",
      "description": "The store, actions, and reducers, oh my!",
      "status": "Unstarted",
      "timer": 86,
      "projectId": 1
    },
    {
      "id": 2,
      "title": "Peace on Earth",
      "description": "No big deal.",
      "status": "Unstarted",
      "timer": 132,
      "projectId": 2
    },
    {
      "id": 3,
      "title": "Create Facebook for dogs",
      "description": "The hottest new social network",
      "status": "Completed",
      "timer": 332,
      "projectId": 1
    }
  ]
}
```

Middleware

The Redux concepts: Middleware

- Redux middleware is code that sits between an action object being dispatched and the store passing the action to the reducer.
- The middleware helps you do things that do NOT fit within the Redux concepts e.g. send analytics to a third party to track the user interaction per action, helper to handle async functions or execute a long running-process task

Reducer

The Redux concepts: Reducer

- Reducer is a function to update the application state in the store.
- Reducer will check on the type attribute from the action object.
- The type will specify which part of the store will update.
- The new updated state will come from the payload attribute in the action object.
- A Reducer is a pure function: same input <> same output, no side effects.

Selector

The Redux concepts: Selector

- Selectors are functions that accept a state from the Redux store and compute data that will be passed to the view.
- The biggest benefit of a selector is that it can transform the data structure that we get from the store to a different structure that the view requires.

Conclusion

- Redux is just a design pattern library.
- It does NOT work only React. You can use Redux even if you use only vanilla JavaScript.
- The Redux concepts look very easy but when you implement the real thing, you need take time to learn it fully. Especially when you are working on asynchronous processes or long running-process tasks.
- It decouples the business logic from the view.
- People complain about Redux as it requires lots of boilerplate. But they forgot about its benefit. I think this is fair trade-off.

Next Steps

- Convert the auction system in PTT-PM project to React/Redux component.



Managing the UI state is both “HARD” and an “ART”.

Learning Redux is a good thing.

You can apply this idea to all major frameworks today: React, Vue or the next standard candidate of the future, the “Web component”.

Thanks!

Contact Nimble

nimblehq.co

hello@nimblehq.co

Bangkok

399 Interchange 21 Sukhumvit Road, Unit
#2402-03, Klong Toei, Wattana, Bangkok
10110, Thailand

Singapore

28C Stanley St, Singapore 068737

Hong Kong

20th Floor, Central Tower
28 Queen's Road, Central, Hong Kong

