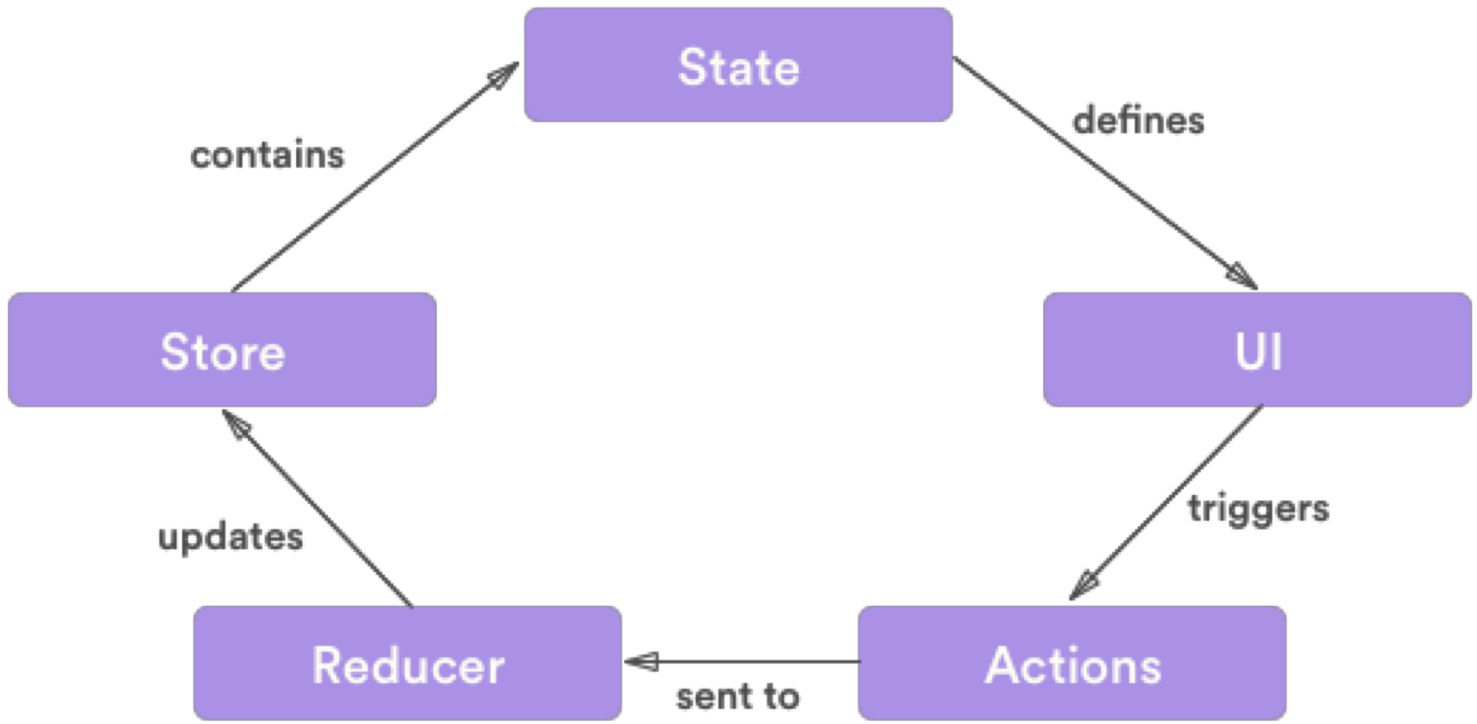
Lab07
Redux Review

WebApp

DataLab, CS, NTHU

2019 Spring

Redux Workflow



From React To Redux

```
render() {
    const {unit} = this.props;
    const {group, city, masking, posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} unit={unit} onQuery={this.handleWeatherQuery}/>
                <WeatherDisplay {...this.state} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(lab-react-weathermood-post)

```
render() {
    const {city, group, description, temp, unit, masking} = this.props;
    const {posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} defaultUnit={unit} submitAction={getWeather} />
                <WeatherDisplay {...{group, description, temp, unit, masking}} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(redux-weathermood-post)

UI

- Use props rather then state

```
render() {
    const {unit} = this.props;
    const {group, city, masking, posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} unit={unit} onQuery={this.handleWeatherQuery}/>
                <WeatherDisplay {...this.state} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(lab-react-weathermood-post)

```
render() {
    const {city, group, description, temp, unit, masking} = this.props;
    const {posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} defaultUnit={unit} submitAction={getWeather} />
                <WeatherDisplay {...{group, description, temp, unit, masking}} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(redux-weathermood-post)

UI

- Dispatch actions

```
render() {
    const {unit} = this.props;
    const {group, city, masking, posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} unit={unit} onQuery={this.handleWeatherQuery}/>
                <WeatherDisplay {...this.state} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(lab-react-weathermood-post)

```
import {getWeather} from 'states/weather-actions.js';

render() {
    const {city, group, description, temp, unit, masking} = this.props;
    const {posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} defaultUnit={unit} submitAction={getWeather} />
                <WeatherDisplay {...{group, description, temp, unit, masking}} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx
(redux-weathermood-post)

Action

- You can dispatch multiple (asynchronous) actions in one action

```
export function getWeather(city, unit) {
    return (dispatch, getState) => {
        dispatch(startGetWeather(city, unit));

        dispatch(maskTodayBg());
        setTimeout(() => {
            dispatch(unmaskTodayBg());
        }, 600);

        return getWeatherFromApi(city, unit).then(weather => {
            const {city, code, group, description, temp, unit} = weather;
            dispatch(endGetWeather(city, code, group, description, temp));
            dispatch(setUnit(unit));
        }).catch(err => {
            console.error('Error getting weather', err);
            dispatch(resetWeather());
        });
    };
};
```

weather-action.js

Action

- You can dispatch multiple (asynchronous) actions in one action

```
export function getWeather(city, unit) {
    return (dispatch, getState) => {
        dispatch(startGetWeather(city, unit));

        dispatch(maskTodayBg());
        setTimeout(() => {
            dispatch(unmaskTodayBg());
        }, 600);

        return getWeatherFromApi(city, unit).then(weather => {
            const {city, code, group, description, temp, unit} = weather;
            dispatch(endGetWeather(city, code, group, description, temp));
            dispatch(setUnit(unit));
        }).catch(err => {
            console.error('Error getting weather', err);
            dispatch(resetWeather());
        });
    };
};
```

weather-action.js

Action

- In a simple action, return the type of action and parameters (which will be sent to reducer later)

```
function startGetWeather(city, unit) {
    return {
        type: '@WEATHER/START_GET_WEATHER',
        city,
        unit
    };
}
```

weather-action.js

Reducer

- In a reducer, return the new state for redux store

```
export function weather(state = initWeatherState, action) {
    switch (action.type) {
        case '@WEATHER/START_GET_WEATHER':
            return {
                ...state,
                city: action.city, // set city state immediately to prevent input text
                weatherLoading: true
            };
```

weather-reducers.js

Connect Redux store to React

- Use connect function

```
import {connect} from 'react-redux';
```

- Pass a function to connect
  - Input: The state of redux store
  - Output: The props of the react component

```
export default connect((state) => {
    return {
        ...state.weather,
        unit: state.unit
    };
})(Today);
```

Today.jsx

Back To UI

- Use the props of the react component to render

```
render() {
    const {city, group, description, temp, unit, masking} = this.props
    const {posts, postLoading} = this.state;

    document.body.className = `weather-bg ${group}`;
    document.querySelector('.weather-bg .mask').className = `mask ${masking ? 'masking' : ''}`;

    return (
        <div className='today'>
            <div className='weather'>
                <WeatherForm city={city} defaultUnit={unit} submitAction={getWeather} />
                <WeatherDisplay {...{group, description, temp, unit, masking}} day='today'/>
            </div>
            <div className='posts'>
                <PostForm onPost={this.handleCreatePost} />
                <PostList posts={posts} onVote={this.handleCreateVote} />{
                    postLoading &&
                    <Alert color='warning' className='loading'>Loading...</Alert>
                }
            </div>
        </div>
    );
}
```

Today.jsx

Assignment - Redux Weathermood Post

Grading

1. (10%) Readme
   List your action types and reducers in README.md, and simply explain how they works (you can use Chinese or English)
   Please add another readme file. **Do Not Modify This README** .
2. (45%) Implement post-actions.js
3. (45%) Implement post-reducers.js

We will grade your implementation based on your code structure and readability.

Bonus: lab-react-weathermood-post

1. (5%) Modify the code to let user can vote to a post only once.
2. (5%) If user votes different mood on the same post, the vote mood will change to the new mood.
3. (5%) If user votes the same mood on the same post, the vote mood will be deleted.

Complete above requirements **with Redux** to get bonus points.

Deadline

Submit your work before 2019/04/25 (Thu.) 23:59:59