Front End Software Development

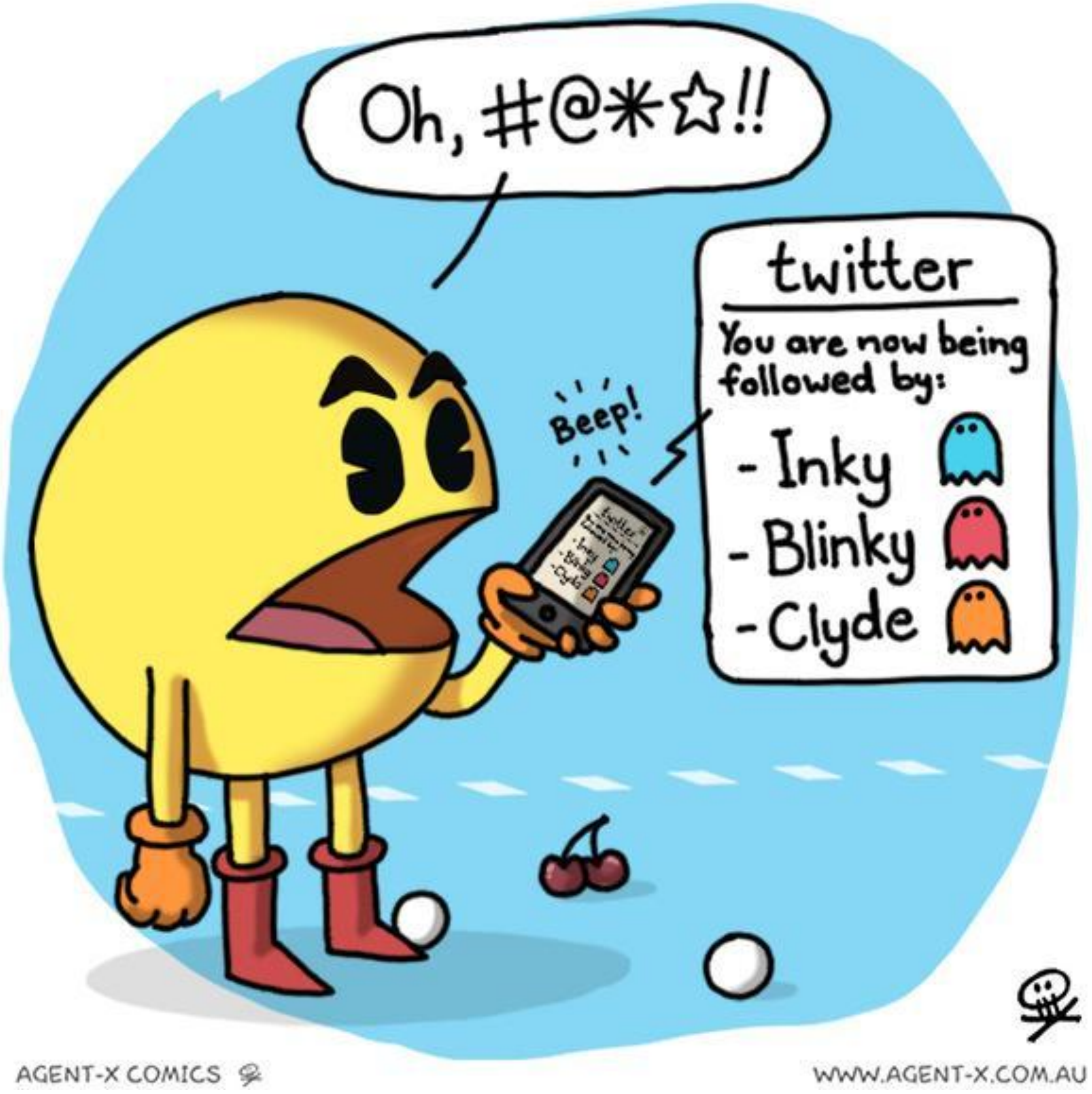
Web App Design w/React (weeks 13 - 18)

Week 05

Agenda

- Questions
- Higher Order Components
- Redux
  - Installation
  - Store
  - Actions
  - Reducers
- Final Project

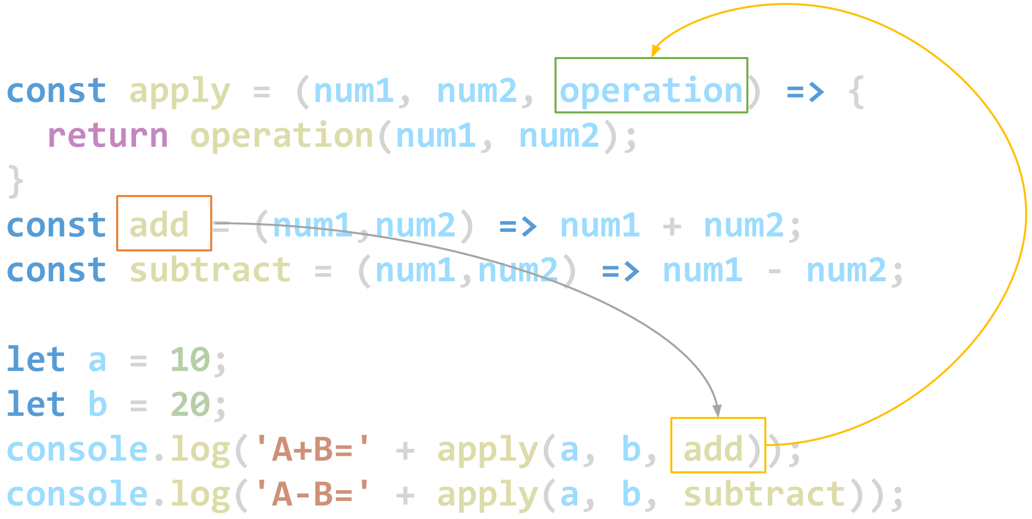
Questions


Oh, #@*☆!!
Beep!
twitter
You are now being followed by:
- Inky
- Blinky
- Clyde
AGENT-X COMICS
WWW.AGENT-X.COM.AU

Higher Order Components

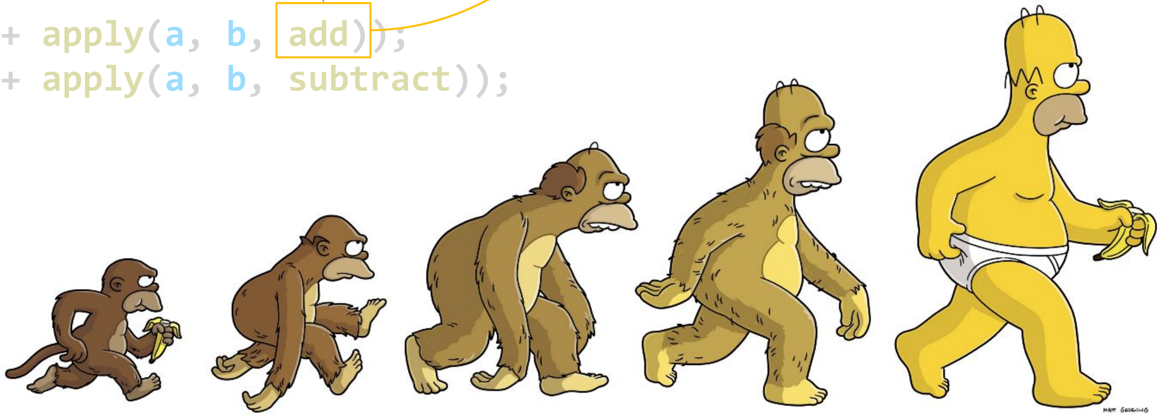
(Functions)

- Higher-Order Function
  - Simple definition: Function that takes a function as a parameter or returns a function.
    - Common jargon: Lambas, expressions, delegates, arrow functions



```
const apply = (num1, num2, operation) => {
  return operation(num1, num2);
}
const add = (num1,num2) => num1 + num2;
const subtract = (num1,num2) => num1 - num2;

let a = 10;
let b = 20;
console.log('A+B=' + apply(a, b, add));
console.log('A-B=' + apply(a, b, subtract));
```

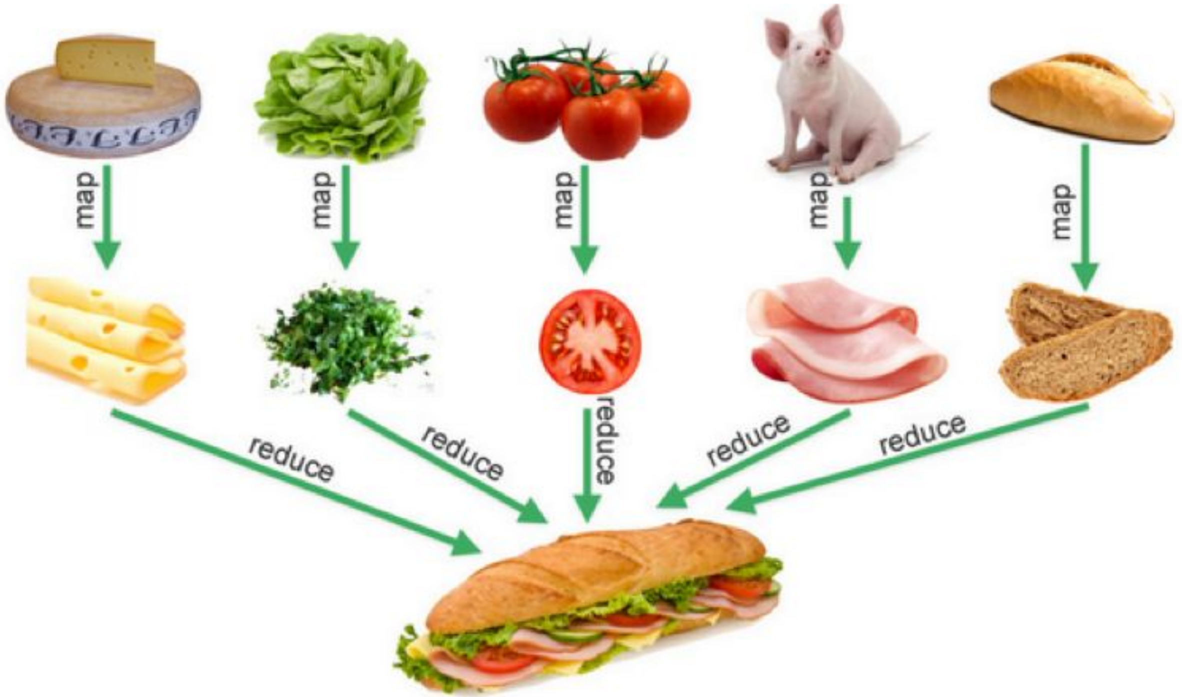


Higher Order Components

(Functions)

- Built-in examples:
  - `Array.forEach()`
  - `Array.filter()`
    `Array.map()`
  - `Array.reduce()`

Note: Map/reduce is a core concept associated with big data. Takes large amounts of data, transforms it (map) and reduces to a single value (reduce).



Higher Order Components

- Component
  - Takes a Component as a parameter and returns a new Component.
  - Does not mutate (change) original component.
- Best Practice(s)
  - Place in `components/` folder
  - Prefix name of component **`with`**`<ComponentName>`
  - Label parameter as `WrappedComponent`

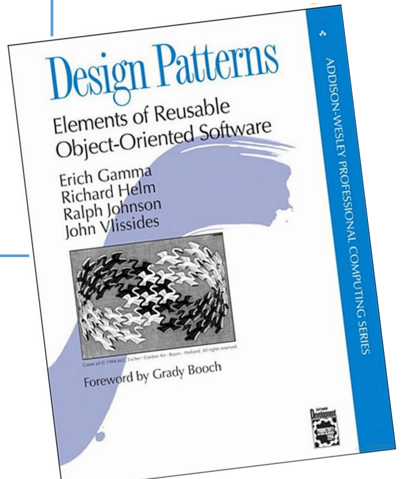
Higher Order Components

```
function App() {
  const ResultWithAlert = withAlert( Result );
  return (
    <div>
      <ResultWithAlert>
        This is an alert
      </ResultWithAlert>
    </div>
  );
}
```

```
const Result = (props) => {
  return (
    <>
      { props.children }
    </>
  );
};
```

```
const withAlert = (WrappedComponent) => {
  return class extends React.Component {
    render() {
      return(
        <div className="alert alert-primary" role="alert">
          <WrappedComponent {...this.props} />
        </div>
      );
    }
  }
};
```

Also Known As: *Decorator* Pattern

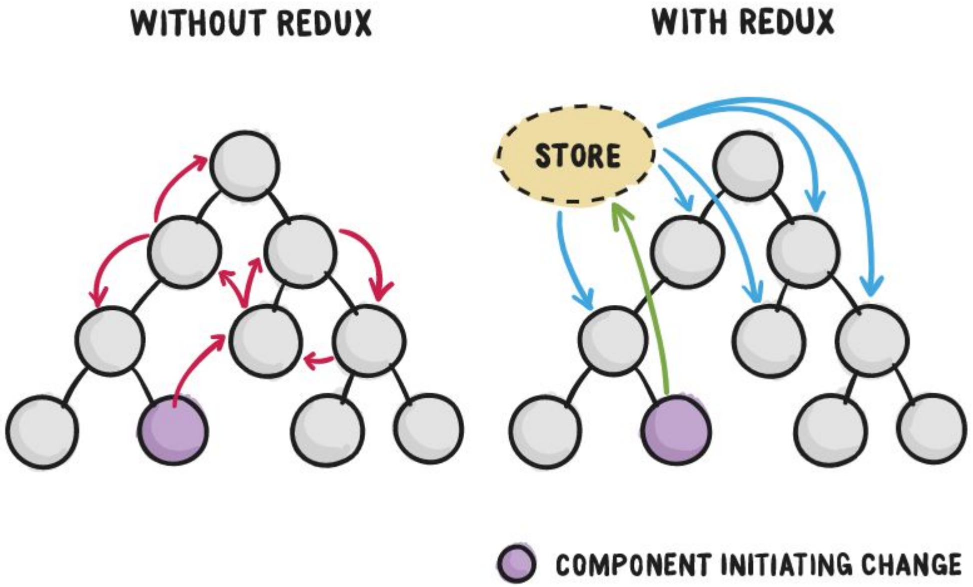

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Redux

- Lifting & Pushing `state`
  - Allows state to be accessed centrally
- Others: Vuex, MobX, Immer, etc.

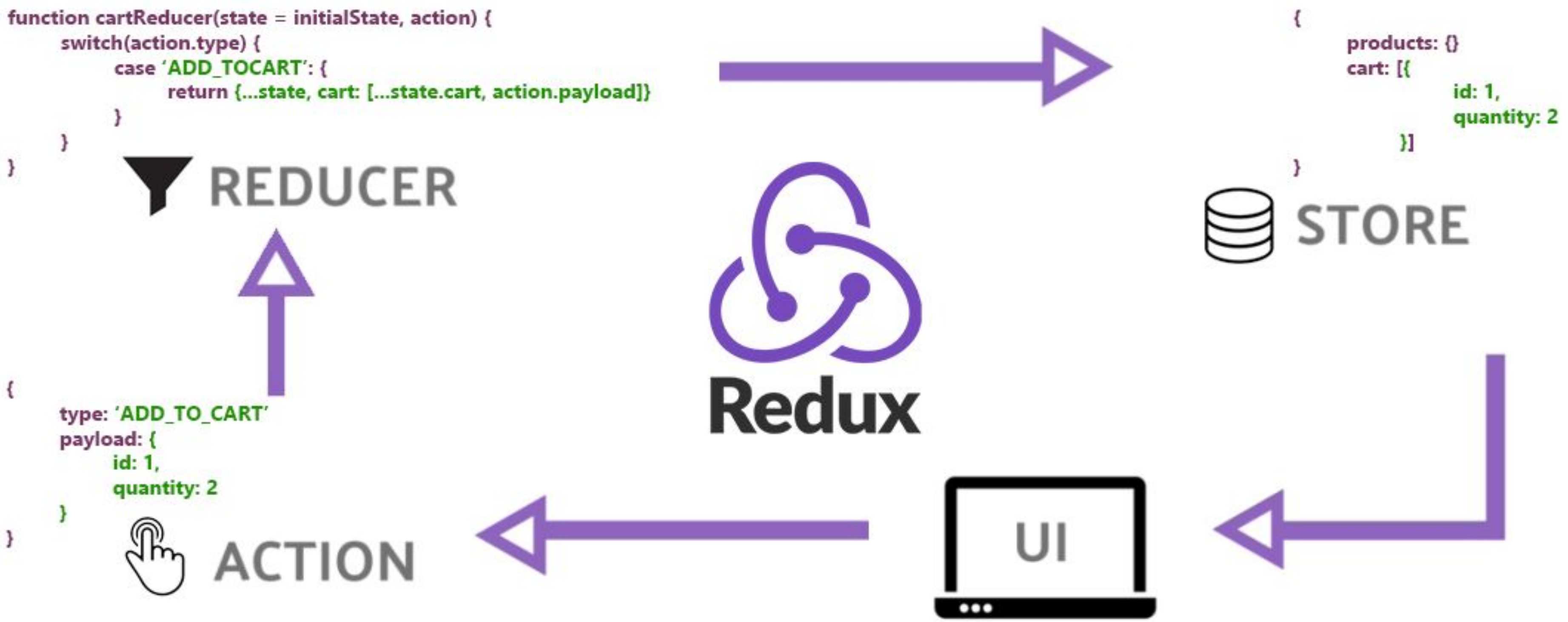
"Don't use Redux until you have problems with vanilla React."

- Principles
  - Store
  - Actions
  - Reducers



Redux
(Installation)

```
npm install redux react-redux
```



```
function cartReducer(state = initialState, action) {
    switch(action.type) {
        case 'ADD_TOCART': {
            return {...state, cart: [...state.cart, action.payload]}
        }
    }
}
```

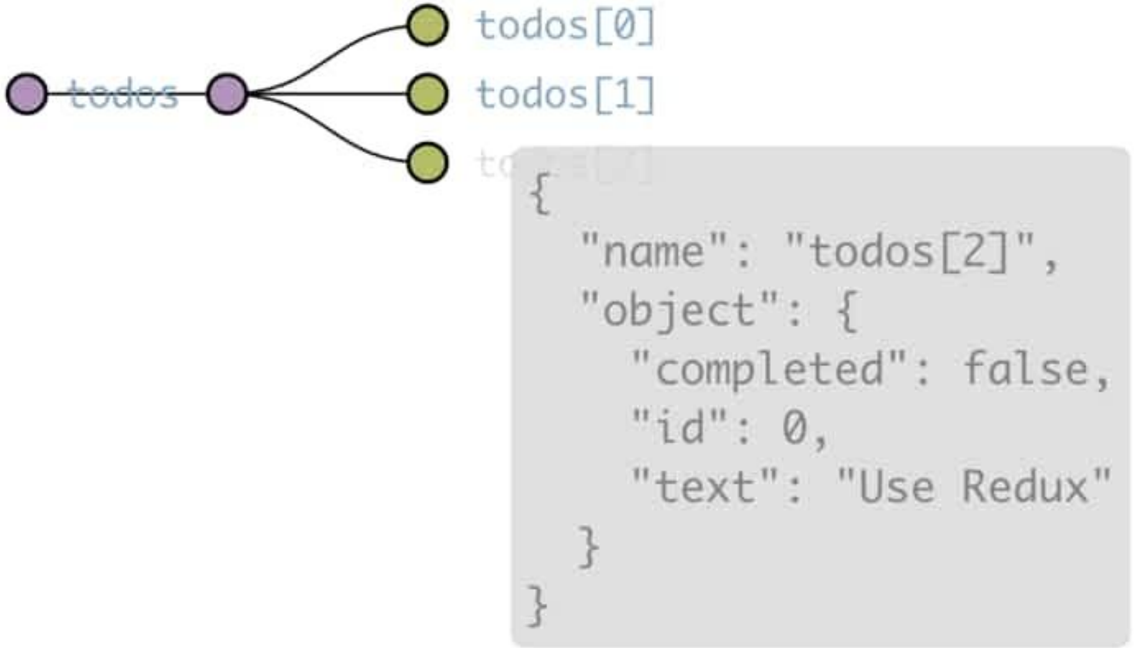
```
{
    products: {}
    cart: [{
            id: 1,
            quantity: 2
        }]
}
```

```
{
    type: 'ADD_TO_CART'
    payload: {
        id: 1,
        quantity: 2
    }
}
```

Redux

(Store)

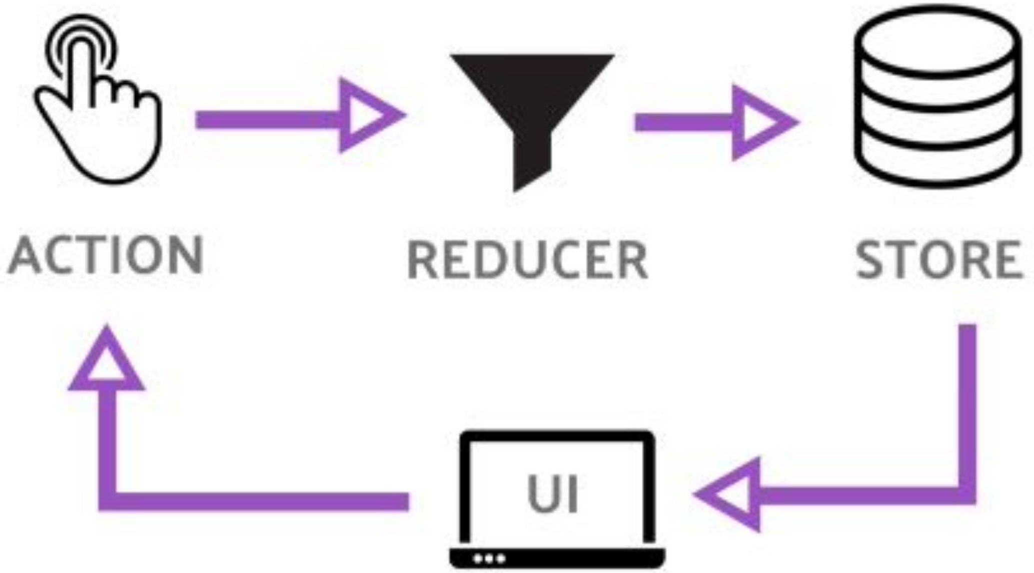
- Store
  - Single, centralized location of application State
  - One Store per application
  - Centralized container
  - Read-only



```
{
  "name": "todos[2]",
  "object": {
    "completed": false,
    "id": 0,
    "text": "Use Redux"
  }
}
```

Redux
(Actions)

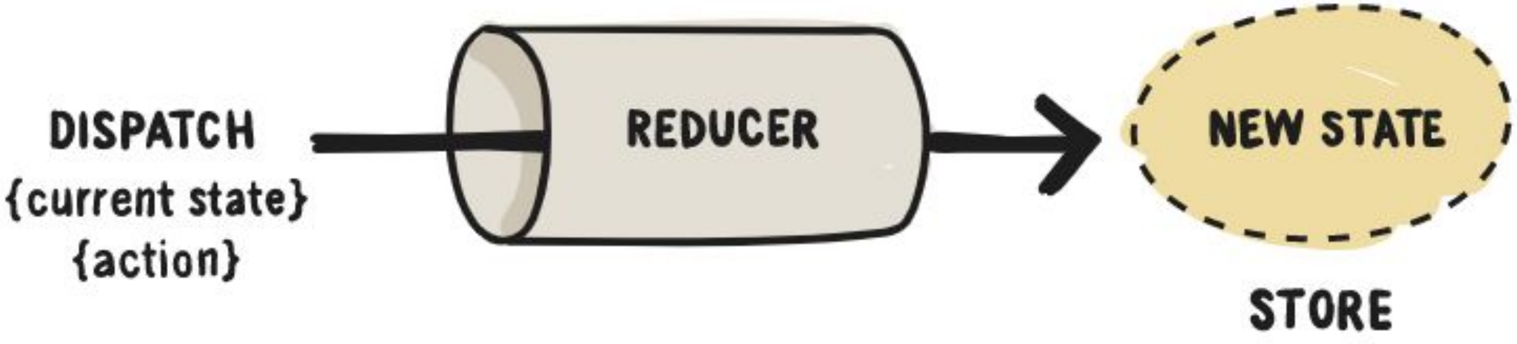
- Actions
  - Modify the store
  - JavaScript objects
  - Typically created or associated with AJAX & DOM events *(i.e. click, fetch, etc.)*



Redux

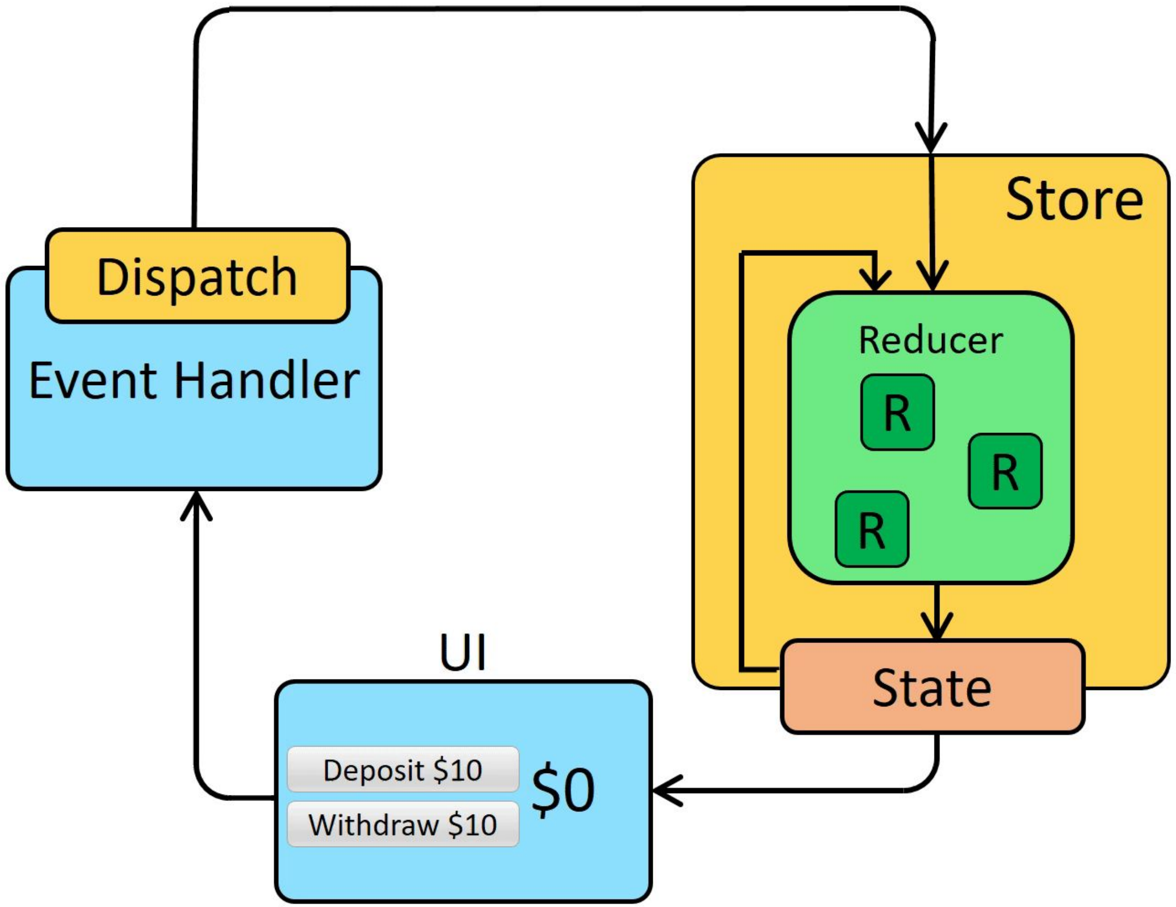
(Reducers)

- Reducers
  - Pure function
  - Takes current state as input
  - Returns new / updated state
  - State updated based on Actions
  - Typically, single Reducer per application



Redux

(Dataflow)



Final Project

Using an online **API** of your choice create a React project of your choice.

If no API is available, use an array for in-memory storage. **Note**: *crudcrud.com* is a popular choice.

- **Requirements**
  - Use React Router and have at least 3 pages
  - Use React Bootstrap (or an alternative styling library)
  - Allow for **ALL** CRUD operations (Create, Read, Update & Delete).

  - Must contain at least 10 components

HANDS ON

Sample Project

(Continued)

The end…

is just the beginning!