webrx-react

Monadic Development for the Web
Using RxJS and React

Follow Along @ https://git.io/vQ10Y

Who Am I?

Pat Sissons
Senior Software Developer
at Marine Learning Systems

/patsissons

/marinels

webrx-react?

`webrx-react` is a single page application web framework written in TypeScript that aims to reduce boilerplate code by taking advantage of monadic state mutations to drive efficient page component rendering.

What is Monadic Programming?

State Encapsulation
Functional Mutations
Chainable Operations

RxJS

Monads as Observables
Asynchronous Event Management
Large Library of Operations
Ported to Many Languages

RxJS Demo

```javascript
const container = document.getElementById('container');
const baseUri = '//baconipsum.com/api/?type=all-meat&format=html';
Rx.Observable
  .timer(0, 5000)
  .take(10)
  // pick a random number of paragraphs between 1 and 5
  .map(x => Math.floor(Math.random() * 5) + 1)
  .flatMap(x => {
    return Rx.Observable
      .ajax({
        url: `${ baseUri }&paras=${ x }`,
        crossDomain: true,
        responseType: 'text',
      });
  })
  .subscribe(x => {
    container.innerHTML = x.response;
  });
```

Ideas from **WPF** and **RxUI**

MVVM Pattern
Data Binding
View Templating
Reactive Properties
Reactive Commands
Observable Composition with `whenAny`

A Lightweight Port of **webrx**

Reactive Object Framework
Smaller Footprint, More Modularity
Support for TypeScript `2.3.x`
Support for RxJS `5.x.x`
Closer Approximation to RxUI

Properties and Commands

Observable Sourced Properties
Two-Way Bound Properties
Observable Sourced Command Execution
DOM Event Command Execution

Observable Web API

Using Observable.ajax
`GET` Data
`POST` Modifications
Asynchronously Composable Results

ObservableApi Demo

```javascript
const api = new ObservableApi('//baconipsum.com/api/?type=');
Observable
  .timer(0, 5000)
  .take(10)
  // pick a random number of paragraphs between 1 and 5
  .map(x => Math.floor(Math.random() * 5) + 1)
  .flatMap(paras => {
    return api
      .getObservable<Array<string>>(
        'all-meat',
        { format: 'json', paras },
      );
  })
  .subscribe(
    x => { console.log(x); },
    e => { console.error(e); },
  );
```

Composing Observables with
whenAny

Wrapper for `combineLatest`
Automatic `startWith` for Properties

```javascript
wx.whenAny(
  Observable.of('x1'),
  Observable.timer(1000, 1000).take(2).select(x => `x2-${ x }`),
  Observable.from([ 'x3-0', 'x3-1', 'x3-2' ]),
  (x1, x2, x3) => ({ x1, x2, x3 }),
)
// { x1, x2-0, x3-0 }
.subscribe(x => { console.log(JSON.stringify(x)); });
```

React Rendering Engine

Readonly Component Attributes
Readonly Snapshots of Mutable State
Declarative Templating
Observable Results Invoke Rendering

React Demo

(Click the title for a live demo)

```jsx
class HelloThereComponent extends Component<{}, { counter: number }> {
  constructor() { super(); this.state = { counter: 0 }; }
  componentWillMount() {
    Rx.Observable
      .timer(0, 2800) // gif has a duration of ~2800ms
      .subscribe(x => this.setState(() => ({ counter: x + 1 })));
  }
  render() {
    return (
      <div style={({ textAlign: 'center' })}>
        <img src='http://gph.to/2tC4JiE' />
        <h3>Hellos There'd: { this.state.counter }</h3>
      </div>
    );
  }
}
```

View Models

Containers for Properties & Commands
Lifecycle Injection Functions
Consumers of Routing State
Search & Menu Item Injection Functions

Views

React `Component<P, S>` Wrapper
React Lifecycle Injection Functions
View Model Component Bindings
Updates Driven by View Model

Component Demo

```
class ToggleViewModel extends BaseViewModel {
  public readonly toggle = this.command<boolean>();
  public readonly enabled = this.toggle.results
    .scan(x => !x, false).toProperty(false);
}
interface ToggleProps extends BaseViewProps {}
class ToggleView extends BaseView<ToggleProps, ToggleViewModel> {
  updateOn() { return [ this.state.enabled.changed ]; }
  render() {
    const enabled = this.state.enabled.value;
    return <CommandButton
      className={ classNames('Toggle', { enabled }) }
      command={ this.state.toggle }>
        { this.props.children }
    </CommandButton>;
  }
}
```

webrx-react Components

Based on Bootstrap 3 (`react-bootstrap`)
Easy to Use Component Library
`BindableInput` & `CommandButton`
`DataGrid` & `ItemListPanel`
`ModalDialog` & `ContextMenu`
And Many More...

BindableInput & CommandButton

```
const input = wx.property<string>();
const cmd = wx.command(x => console.log(`Executed: '${ x }'`));
function render() {
  return (
    <div>
      <BindableInput property={ this.state.input }>
        <FormControl type='text' placeholder='Type Some Text In...' />
      </BindableInput>
      <CommandButton
        command={ this.state.cmd }
        commandParameter={ () => this.state.input.value }
      >
        <span>Execute!</span>
      </CommandButton>
    </div>
  );
}
```

DataGridView & ItemListPanelView

```jsx
const grid = new DataGridViewModel(Observable.of([
  { id: 1, userName: 'hmar', name: 'Hank', lastName: 'Mardukas' },
]));
function render() {
  return (
    <div>
      <DataGridView viewModel={ this.state.grid }>
        <DataGridColumn fieldName='id' header='User ID' />
        <DataGridColumn header='Name'
          renderCell={ x => `${ x.firstName } ${ x.lastName }` }
        />
      </DataGridView>
      <ItemListPanelView viewModel={ this.state.grid }>
        <DataGridColumn header='Name'
          renderCell={ x => `${ x.firstName } ${ x.lastName }` }
        />
        <DataGridColumn
          renderCell={ x => (<CommandButton command={ this.state.viewUser } />) }
          tooltip={ x => x == null ?
            (<Tooltip>Click the Button to view the user</Tooltip>) :
            (<Tooltip>{ `View User ${ x.userName }` }</Tooltip>)
          }
        />
      </ItemListPanelView>
    </div>
  );
}
```

ModalDialogView & ContextMenu

```jsx
const modal = new ModalDialogViewModel();
function render() {
  return (
    <div>
      <ContextMenu id='menu' header='Open Modal'>
        <div>Summon A Context Menu</div>
        <MenuItem onClick={ this.bindEventToCommand(this.state, x => x.showModal) }>
          Show Modal
        </MenuItem>
      </ContextMenu>
      <ModalDialogView viewModel={ this.state.modal }  header='A Wild Modal Appears'>
        <CommandButton command={ this.state.modal.hideOnExecute(this.state.cancel) }>
          Cancel
        </CommandButton>
        <CommandButton command={ this.state.modal.hideOnExecute(this.state.accept) }>
          Accept
        </CommandButton>
      </ModalDialogView>
    </div>
  );
}
```
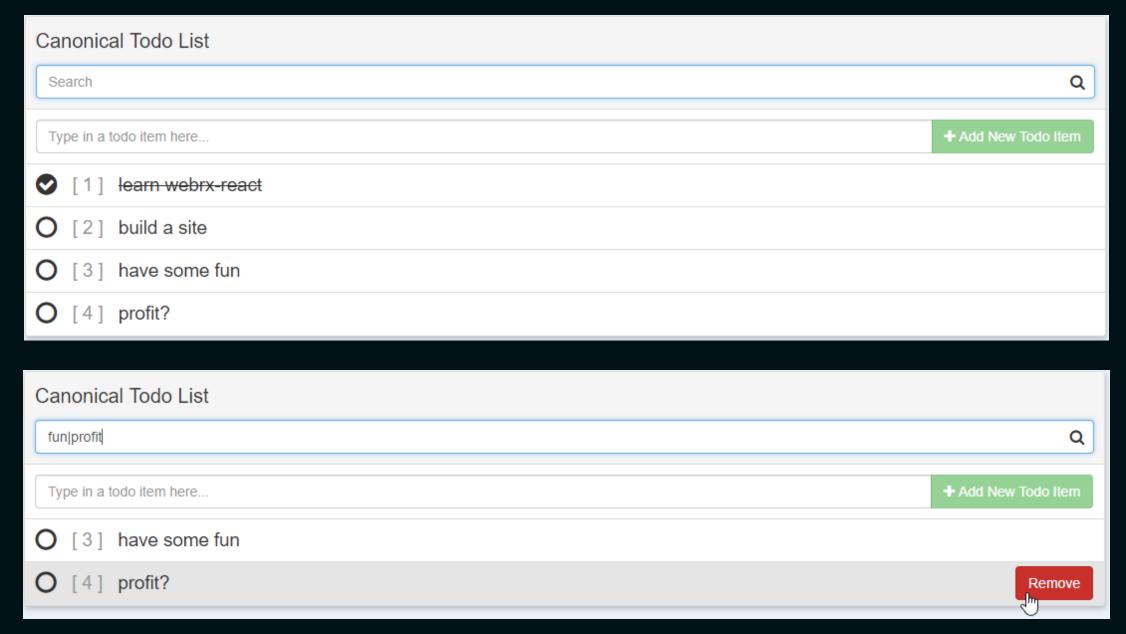
RouteHandlerView

```javascript
const routingMap = {
  '/Demo': { path: '/Demo/' }, // redirect
  '^/Demo/(.*)$': { path: '/Demo/', creator: () => new DemoComponentViewModel() },
};
const viewMap = {
  Demo: (viewModel: DemoComponentViewModel) => (
    <DemoComponentView viewModel={ viewModel }>
      <div>Routed Content: { this.state.param.value }</div>
    </DemoComponentView>
  ),
};
const router = new RouteHandlerViewModel(routingMap);
function render() {
  return (
    <div>
      <div>Header (see PageHeaderView)</div>
      <RouteHandlerView viewModel={ this.state.router } viewMap={ viewMap } />
      <div>Footer (see PageFooterView)</div>
    </div>
  );
}
```

The RouteManager

Hash-based Routing State
Use Browser `history` API
Fallback on `hashChanged` events
Routing State Decoded Automatically
`saveRoutingState` & `loadRoutingState`

Canonical Todo List

(Click the title for a live demo)

Canonical Todo List

Search 🔍

Type in a todo item here... ➕ Add New Todo Item

✓ [1] ~~learn webrx-react~~

○ [2] build a site

○ [3] have some fun

○ [4] profit?

Canonical Todo List

fun|profit 🔍

Type in a todo item here... ➕ Add New Todo Item

○ [3] have some fun

○ [4] profit? Remove

Consuming `webrx-react`

Published to npm
Available via unpkg (npmcdn)
Uglified Bundle for ES5+ (IE9 support)
Modular Imports for ES6+ & TypeScript

Why?

Modular Sharable Components
Snappy Responsive Complex Views
IE9+ Compatibility

Roadmap

First Official Non-Beta Release
Component View Abstractions
e.g., Material, Foundation, Fabric, Polymer

Questions?