

REACT & RXJS

REACTIVE PROGRAMMING USING REACT WITH RXJS.

ReactJS Utah - July 28, 2015

Seth House @whiteinge (Twitter & GitHub)

REACTIVE EXTENSIONS



OBSERVERS

```
var Rx = require('rx');

var myobsv = Rx.Observer.create(
  function(x) {
    console.log('onNext: ', x);
  },
  function(err) {
    console.log('onError: ', err);
  },
  function() {
    console.log('onCompleted: ', x);
  });

myobsv.onNext(1);
myobsv.onNext(2);
myobsv.onCompleted();
```

```
var keyups = Rx.Observable.fromEvent($input, 'keyup')
```

OBSERVABLES

```
myobsv.subscribe(  
  function(x) {  
    console.log('onNext: ', x);  
  },  
  function(err) {  
    console.log('onError: ', err);  
  },  
  function() {  
    console.log('onCompleted: ', x);  
  });
```

EVENT LISTENRS

```
var keysource = Rx.dom.keydown(window)
    .pluck('keyCode')
    .filter(x => x === 27);

var clicksource = Rx.dom.click(window)
    .skip(1) // ignore first click that opens the dropdown
    .filter(ev => document.querySelector('#thing')
        .contains(ev.target));

keysource.merge(clicksource)
    .subscribe(closeModal);
```

FETCHING USERNAMES FROM GITHUB

```
var github_users = Rx.DOM.ajax({
    method: 'GET',
    url: 'https://api.github.com/users' });
    .filter(x => x.status === 200)
    .map(JSON.parse)
    // Cache the deserialized response.
    .shareReplay(1)
    // Explode items in JSON response into stream items.
    .flatMap(x => Rx.Observable.from(x));

var usernames_list = github_users
    .pluck('login')
    .subscribe(x => console.log('GitHub user:', x));
```

COMBINE USERS WITH CLICK EVENTS

```
var clicks = Rx.DOM.click(document.querySelector('#thelink'));  
  
// Combine each click with a user.  
clicks.zip(usernames_list, (click, user) => user)  
    // When the list is exhausted the event handler is removed.  
    .subscribe(x => console.log('GitHub user:', x));
```


FLUX

- Unidirectional data flow.
- "Immutable" data structures.
- Minimization of state tracking.

DISPATCHER

```
var Dispatcher = new Rx.Subject();
```

STORE

```
var myStore = Dispatcher
    .filter(x => x.fooEvents === true);

var myAjax = myStore
    .startWith({action: 'refresh'})
    .filter(x => action === 'refresh')
    .flatMap(() => Rx.DOM.get('/some/url'))
    .shareReplay(1);

var myAjaxSummarized = myAjax
    .map(summarizeData);
```

VIEW

```
var app = myAjaxSummarized
    .map(function(summaryData) {
        return (
            <h3 onClick={Dispatcher.onNext({
                fooEvents: true, action: 'refresh',
            })}>Refresh</h3>

            <ul>
                {summaryData.map(x => <li>x</li>)}
            </ul>
        );
    });
```

SIDE-EFFECTS

```
app.subscribe(function(content) {  
  React.render(  
    React.createElement('div', content),  
    document.querySelector('#content'));  
});
```

MORE SIDE-EFFECTS

```
Dispatcher.subscribe(function(event) {  
  console.log('Dispatching event', event);  
});
```