FUNCTIONAL PROGRAMMING IN JAVASCRIPT

Sarah Ransohoff

August 20, 2015

BrooklynJS

WHAT IS FUNCTIONAL PROGRAMMING

» based on mathematical notion of functional composition

» list of characteristics (immutable data, first class functions, tail call optimization)

» mentions of mapping, reducing, recursing, currying, etc

WHAT IS FUNCTIONAL PROGRAMMING

» the absence of side effects

EXAMPLE OF NON FUNCTIONAL, WITH SIDE EFFECTS

```
var a = 0;
var addOne = function() {
    console.log(a);
    return a += 1;
}
```

EXAMPLE OF FUNCTIONAL, SANS SIDE EFFECTS

```
var addOne = function(a) {
    return a += 1;
}
```

WE DID IT!

TO GET THE FULL PICTURE, WHAT ELSE DO WE NEED TO KNOW ABOUT FUNCTIONAL PROGRAMMING

1. DON'T ITERATE OVER LISTS

use map and reduce

WHY USE MAP AND REDUCE OVER ITERATING?

» less chance for side effects

» encourages immutable data

» code in a loop can affect variables outside of it

MAP

» takes a function and a collection of items

» runs function on each item in the original collection

» inserts each return val into a new collection

» returns new collection

MAP - AN EXAMPLE

```
<tbody>
    {this.props.apps.map(this.renderApp)}
</tbody>
```

MAP - AN EXAMPLE

```
<tbody>
    {this.props.apps.map(this.renderApp)}
</tbody>
```

Inbox (11) - sarah.ransohof | Inbox (11) - sranso@gmail.c | Venmo | Venmo - Share Payments

dev.venmo.com/account/settings/profile

venmo

 Andrew Kortina | Contact Support | Log out

Settings

Profile

Banks & Cards

Notifications

Privacy

Security

Developer

Profile Photo

 Edit photo

First Name

Andrew

Last Name

Kortina

Username

@kortina

www.venmo.com/kortina

Save Settings

venmo

Andrew Kortina Contact Support Log out

Settings

Profile
Banks & Cards
Notifications
Privacy
Security
Developer

Apps

These are the apps that you have created.

Application	ID	Secret
Local Developer Portal	1000	PVnBy4BPCkdjXg4Us6eKGKkazV4scyrG

REDUCE

» takes a function and a collection of items

» returns the value that is created by combining the items

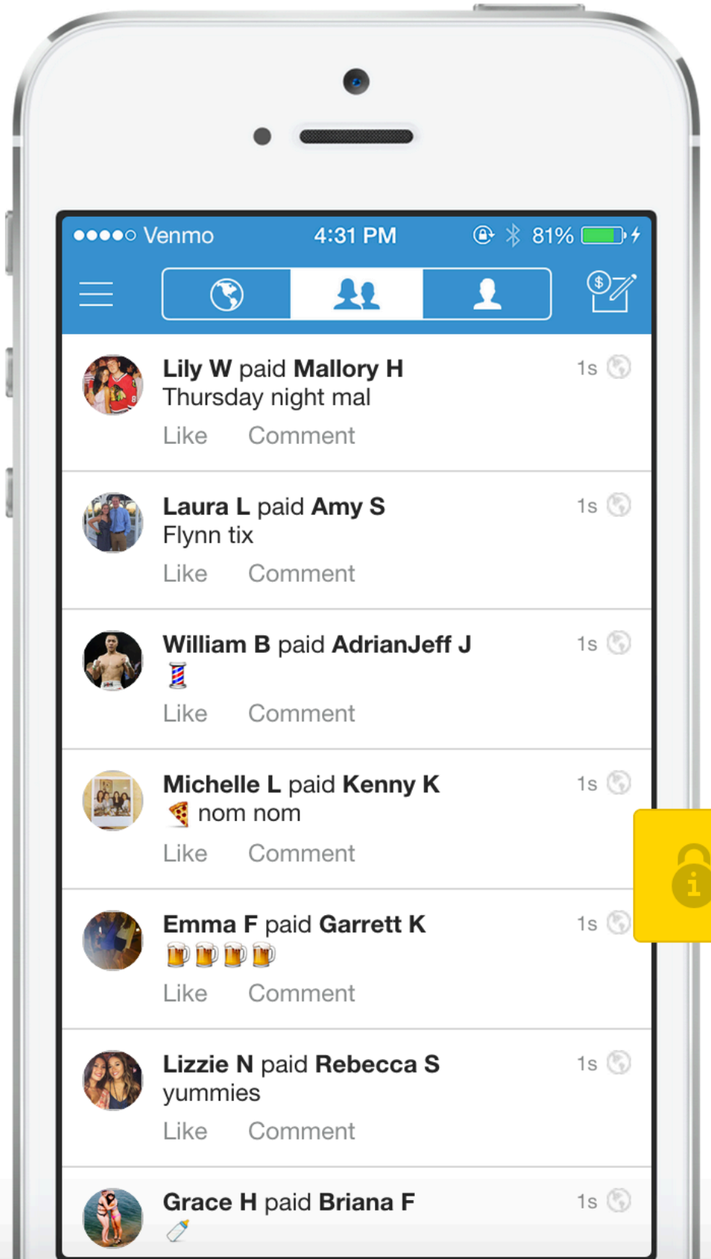
REDUCE - AN EXAMPLE

```
errors = this.state.errors.reduce(function(acc, err) {
    var orientation = 'left';
        if ( err.field === 'lastName' ) {
            orientation = 'right';
        }

        acc[err.field] = <Tooltip message={err.error} orientation={orientation} />;
        return acc;
}, {});
```

Inbox (11) - sarah.ransohof | Inbox (11) - sranso@gmail.c | Venmo | Venmo - Share Payments

https://venmo.com



Welcome.

Enter a first name | First name | Last name | Enter a last name

Email is required | Email

Enter a phone number | Phone

Create a password that is unique to Venmo. Don't use your email password.

Create password

Complete signup

By signing up, you are agreeing to Venmo Terms and Privacy.

2. WRITE DECLARATIVELY, NOT IMPERATIVELY

» describes what to do rather than how to do it
(tell comp the problem, not how to solve it)

DECLARATIVE FEELS SORT OF LIKE THERAPY

REACT IS DECLARATIVE

```
render: function() {
    return (
        <div className="sign-in" onClick={this.handleClick}>
            <a className="icon-lock-blue">Sign in</a>
            { this.state.showForm ? <SignInForm /> : null }
        </div>
    );

}
```

WHEREAS JQUERY IS NOT (IT'S IMPERATIVE)

```
$('.sign-in-form').show();
```

AKA NOT A GOOD THERAPIST

CAN ALSO MAKE PROGRAM MORE DECLARATIVE (AND THEREFORE MORE FUNCTIONAL) BY USING FUNCTIONS

A RELATIVELY CONVOLUTED STATEMENT I KNOW BUT BEAR WITH ME

MORE DECLARATIVE = MORE FUNCTIONS = MORE FUNCTIONAL

```
loginAsGroup(groupId) {
    return this.apiClient.loginAsGroup(groupId)
        .then((resp) => {
            return this.apiClient.loginAsGroupDjango(resp.access_token);
    }).then(this.goToHomePage);
}
```

3. REMOVE STATE

» function should perform every time as if for the first time


BOY

REMOVE STATE - AN EXAMPLE

```
statics: {
    requestSendToken: function(secret, via) {
        return VenmoAPI.twoFactorAuth.sendToken(secret, via);
    }
}
SendTokenButton.requestSendToken(this.props.secret, this.props.via)
    .then(function() {
        this.setState({ spinning: false });
        // code code code
    });
```

WHY FUNCTIONAL PROGRAMMING IS GREAT

- allows programs to be broken down into smaller, simpler pieces
  - makes them more reliable, testable, understandable, readable
- co-exists very well with code written in other styles
- often tends to be more readable

USING FUNCTIONAL PROGRAMMING

» functional languages: Haskell, Clojure, Scala

» libs: underscore.js, bacon.js

» can write functional code in any language that supports first-class functions

WHAT DID WE LEARN

- functional programming is essentially the absence of side effects
- other good notes
  - use map and reduce
  - write declaratively, not imperatively
  - remove state

WE DID IT!?

YEAH, WE DID IT.

SOURCES

- http://maryrosecook.com/blog/post/a-practical-introduction-to-functional-programming
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/
- http://www.smashingmagazine.com/2014/07/dont-be-scared-of-functional-programming/

THANK YOU.
I'M SRANSO
EVERYWHERE.