INTRO TO FUNCTIONAL REACTIVE PROGRAMMING

(FRP)

FUNCTIONAL PROGRAMMING

+

REACTIVE PROGRAMMING

FUNCTIONAL PROGRAMMING

TREATS COMPUTATION AS THE EVALUATION OF MATHEMATICAL FUNCTIONS AND AVOIDS CHANGING-STATE AND MUTABLE DATA.

SAME INPUT ALWAYS EQUALS SAME OUTPUT

EXAMPLE

FIND ALL PRIME NUMBERS IN AN ARRAY?

```
function isPrime(number) {
    let squareroot = Math.sqrt(number)
    for (var i = 2; i <= squareroot; i++) {
        if ((number % i) == 0) {
            return false
        }
    }
    return true
}

let numbers = [1, 52, 4, 90, 17, 42, 72, 101, 55, 3]
let primeNumbers = numbers.filter(isPrime) //1, 17, 101, 3
```

TOO SIMPLE?

TAKE AN ARRAY OF NAMES AND AN ARRAY OF EMAILS. CREATE MODEL OBJECTS AND LIST ALL VALID MODELS

```
class Person {
    constructor(name, email) {
        this.name = name;
        this.email = email;
    }

    static getEmail(person) {
        return person.email
    }
}

function zip(arrays) {
    return arrays[0].map(function (_, i) {
        return arrays.map(function (array) { return array[i] })
    });
}

function contains(sequence) {
    return function (values) {
        return values.indexOf(sequence) != -1
    }
}
```

```
let emails = ["test@mail.com", "garbage", "email@"]
let names = ["Jimmy", "Dave", "Tyler"]

let people = zip([names, emails])
    // [
    //   ["Jimmy", "test@mail.com"],
    //   ["Dave", "garbage"],
    //   ["Tyler", "email@"]
    // ]
    .map(item => new Person(item[0], item[1]))
    // [
    //   Person {name: "Jimmy", email: "test@mail.com"},
    //   Person {name: "Dave", email: "garbage"},
    //   Person {name: "Tyler", email: "email@"}
    // ]
    .filter(person => contains(".com")(Person.getEmail(person)))
    // [
    //   Person {name: "Jimmy", email: "test@mail.com"}
    // ]
```

REACTIVE PROGRAMMING

CHANGES IN DATA WILL BE AUTOMATICALLY PROPAGATED THROUGH THE DATA FLOW

```
c = a + b
```

A OR B CHANGES AND C WILL AUTOMATICALLY BE CHANGED (EXCEL)

FUNCTIONAL REACTIVE PROGRAMMING

COMBINES FUNCTIONAL CONCEPTS WITH PROPAGATION OF CHANGE

- STREAMS OF DATA OVER TIMES
- TRANSFORMATIONS OF DATA THROUGH FUNCTIONS
- BINDING OF DATA TO COMPONENTS

EXAMPLE

SIMPLE LOGIN PAGE

SIMPLE LOGIN PAGE

- ABILITY TO ENTER EMAIL
- ABILITY TO ENTER PASSWORD
- ABILITY TO SUBMIT EMAIL AND PASSWORD

SIMPLE? LOGIN PAGE

- VALIDATE THAT EMAIL IS VALID
- VALIDATE THAT PASSWORD IS VALID
- CHECK THAT BOTH EMAIL AND PASSWORD IS PRESENT AND VALID
- PREVENT SUBMITTING MULTIPLE TIMES IN A ROW

...

IMPERATIVE WAY

HOW WOULD YOU APPROACH THIS NOW?

FRP WAY

DEMO