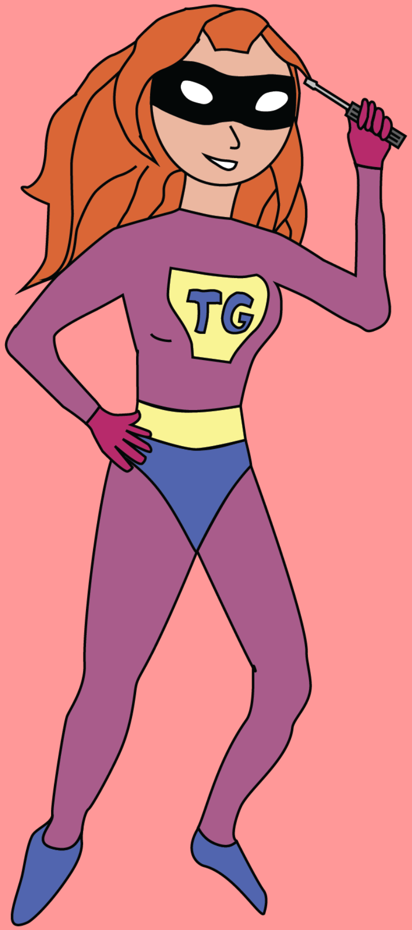
HANNAH HOWARD #ABOUTME



@techgirlwonder
she/her

hannah@carbon
five.com

Personal
Anecdote: I have
a dog

REACTIVE PROGRAMMING:

A Better Way to Write Frontend Applications

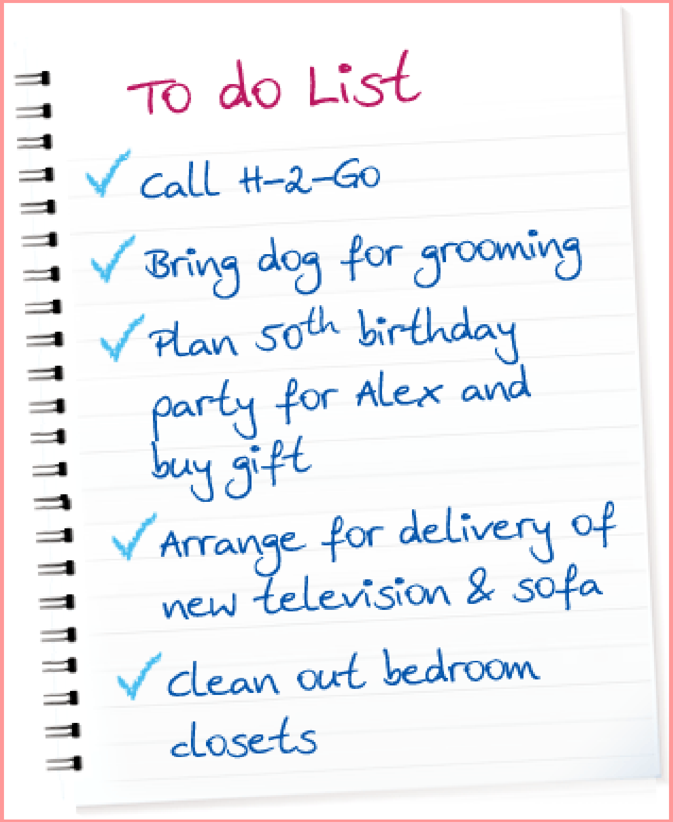
1.
PROBLEM STATEMENT

WHAT IS A COMPUTER PROGRAM?

A computer program is a sequence of instructions for performing a task designed to solve specific problems.

- Wikipedia

'SEQUENCE OF INSTRUCTIONS'



To do List

✓ Call H-2-Go

✓ Bring dog for grooming

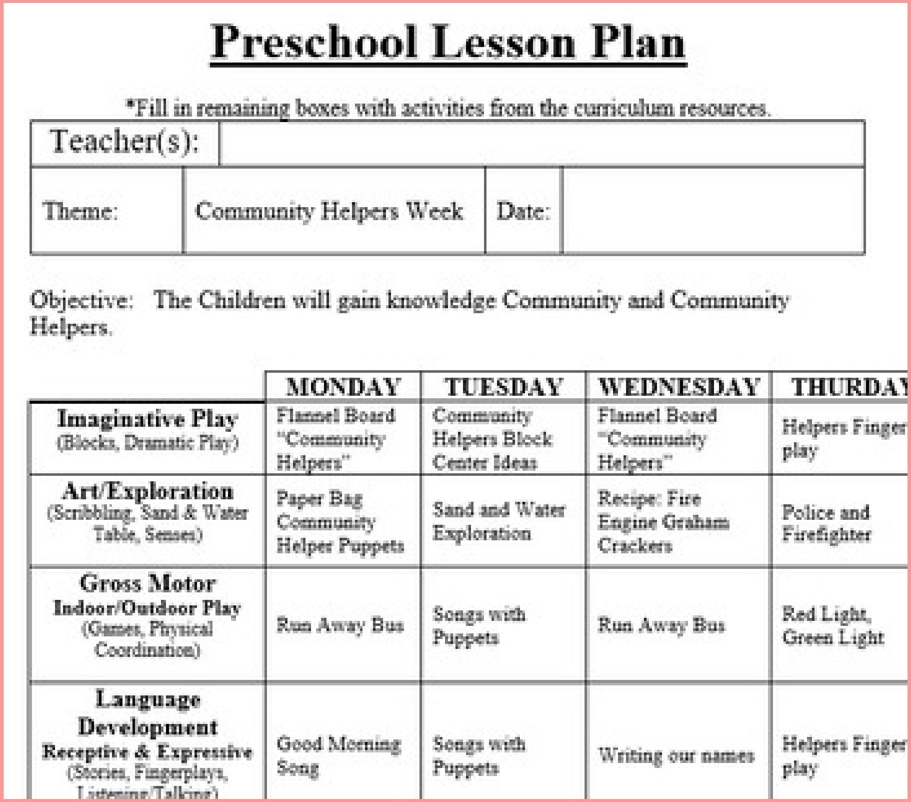
✓ Plan 50th birthday party for Alex and buy gift

✓ Arrange for delivery of new television & sofa

✓ Clean out bedroom closets

Program = Todo List?

LESSON PLAN



Preschool Lesson Plan

*Fill in remaining boxes with activities from the curriculum resources.

Teacher(s):			
Theme:	Community Helpers Week	Date:	

Objective: The Children will gain knowledge Community and Community Helpers.

	MONDAY	TUESDAY	WEDNESDAY	THURDAY
Imaginative Play (Blocks, Dramatic Play)	Flannel Board "Community Helpers"	Community Helpers Block Center Ideas	Flannel Board "Community Helpers"	Helpers Finger play
Art/Exploration (Scribbling, Sand & Water Table, Senses)	Paper Bag Community Helper Puppets	Sand and Water Exploration	Recipe: Fire Engine Graham Crackers	Police and Firefighter
Gross Motor Indoor/Outdoor Play (Games, Physical Coordination)	Run Away Bus	Songs with Puppets	Run Away Bus	Red Light, Green Light
Language Development Receptive & Expressive (Stories, Fingerplays, Listening/Talking)	Good Morning Song	Songs with Puppets	Writing our names	Helpers Finger play

HOW COMPUTER PROGRAMS ACTUALLY RUN

INTERRUPTIONS:

the heart of frontend programming

2.
A BRIEF HISTORY OF INTERRUPTIONS

Technique 1:

GLOBAL EVENT BUS

In The Beginning... C!

```c
#define BYTE unsigned char
#define NUM_SCAN_QUE 256 // this MUST be 256, using BYTE roll-over for \
                         // q code
BYTE gb_scan;
BYTE gb_scan_q[NUM_SCAN_QUE];
BYTE gb_scan_head;
BYTE gb_scan_tail;

static void interrupt(far *oldkb)(void); /* BIOS keyboard handler */

/* ---------------------- get_scan() --------------------- April 17,1993 */
void interrupt get_scan(void)
{
  /* read the scan code from the keyboard */
```

Windows event loop

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
  MSG msg;
  BOOL bRet;

  while (1)
  {
    bRet = GetMessage(&msg, NULL, 0, 0);

    if (bRet > 0) // (bRet > 0 indicates a message that must be processed.)
    {
      TranslateMessage(&msg);
      DispatchMessage(&msg);
```

```
LRESULT CALLBACK MainWndProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam) // second message parameter
{
  switch (uMsg)
  {
  case WM_CREATE:
    // Initialize the window.
    return 0;

  case WM_PAINT:
    // Paint the window's client area.
    return 0;

  case WM_SIZE:
    // Set the size and position of the window
```

Window procedure = read message, update state

```
  case WM_DESTROY:
    // Clean up window-specific data objects.
    return 0;
  //
```

1999?

```
LRESULT CALLBACK MainWndProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam) // second message parameter
{
  switch (uMsg)
  {
  case WM_CREATE:
    // Initialize the window.
    return 0;

  case WM_PAINT:
    // Paint the window's client area.
    return 0;

  case WM_SIZE:
    // Set the size and position of the window.
    return 0;

```

Or did we?

```
    return 0;
  //
  // Process other messages.
  //
  default:
```

```
function todoApp(state = initialState, action) {
  switch (action.type) {
    case SET_VISIBILITY_FILTER:
      return { ...state,
        visibilityFilter: action.filter
      };
    case ADD_TODO:
      return { ...state,
        todos: [
```

NO SHADE TO REDUX

Technique 2:

OBSERVER PATTERN

A SHORT DIGRESSION…

VERY IMPORTANT CONTENT CREATOR

HOW WILL PEOPLE SEE MY CONTENT?

OLD SCHOOL WAY

I will make content

- influencer

I will subscribe to your content

- adoring fan

I made new content

- influencer

I am notified about your content, and can watch it

- adoring fan

I will emit events

- Subject

I will subscribe to your events

- Observer

An event happened

- Subject

I am notified about the event, and can handle it

- Observer

Real World Example

```javascript
// Function to change the content of t2
const modifyText = () => {
  const toggle = document.getElementById("toggle");
  toggle.firstChild.nodeValue = t2.firstChild.nodeValue == "on" ? "off" : "on";
}

// add event listener to table
const el = document.getElementById("toggle-switch");
el.addEventListener("click", modifyText, false);
```

OBSERVER PATTERN VS GLOBAL EVENT BUS

- (+) Way simpler than global event bus
- (+) Localized scope
- (-) Have To Setup Subscriptions

Take home quiz:

TRY DRAG AND DROP

MIXING CONCERNS

1. Handling events
2. Subscribing observers

REDUX ORIGIN STORY

Is this what happened?

IS THERE A BETTER WAY?

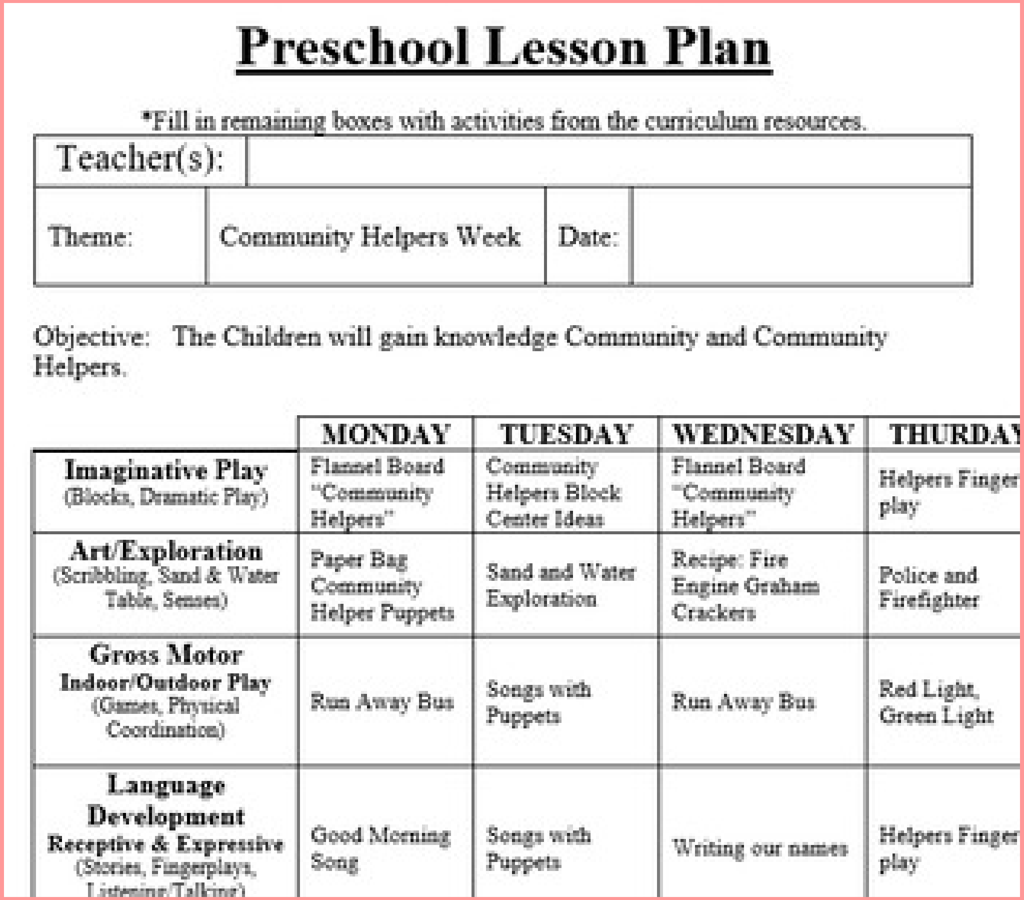
3.
FUNCTIONAL REACTIVE PROGRAMMING

ONCE UPON A TIME...

I taught middle school

GOOD TEACHERS = JEDI

DON'T START WITH A PLAN...



Preschool Lesson Plan

*Fill in remaining boxes with activities from the curriculum resources.

Teacher(s):			
Theme:	Community Helpers Week	Date:	

Objective: The Children will gain knowledge Community and Community Helpers.

	MONDAY	TUESDAY	WEDNESDAY	THURDAY
Imaginative Play (Blocks, Dramatic Play)	Flannel Board "Community Helpers"	Community Helpers Block Center Ideas	Flannel Board "Community Helpers"	Helpers Finger play
Art/Exploration (Scribbling, Sand & Water Table, Senses)	Paper Bag Community Helper Puppets	Sand and Water Exploration	Recipe: Fire Engine Graham Crackers	Police and Firefighter
Gross Motor Indoor/Outdoor Play (Games, Physical Coordination)	Run Away Bus	Songs with Puppets	Run Away Bus	Red Light, Green Light
Language Development Receptive & Expressive (Stories, Fingerplays, Listening/Talking)	Good Morning Song	Songs with Puppets	Writing our names	Helpers Finger play

AND GET INTERRUPTED.

PLAN FOR INTERRUPTIONS AND REACT!

PSA:

PAY TEACHERS

HOW COULD WE WRITE PROGRAMS REACTIVELY?

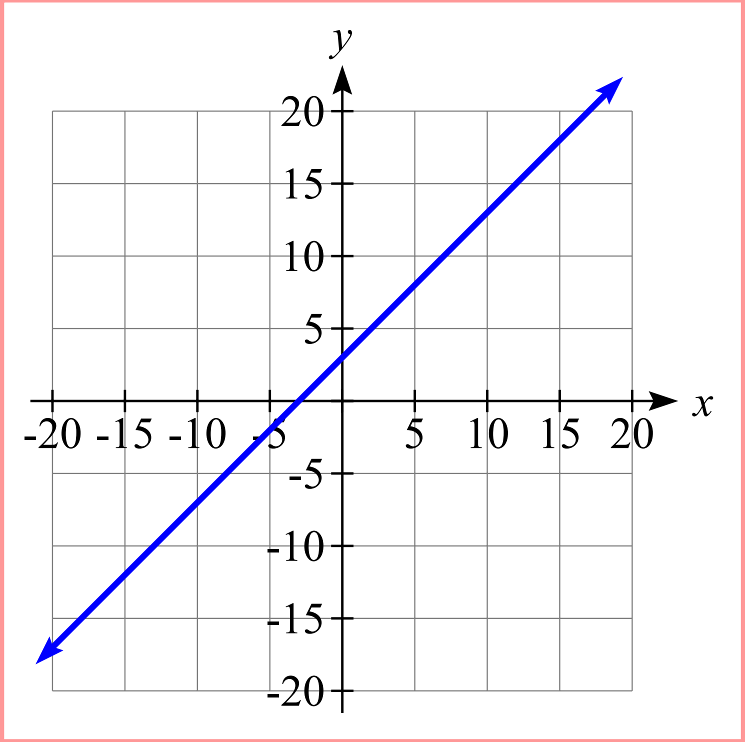
$$Y = X + 3$$

Consider this statement

IMPERATIVE MEANING:

Assign once the value for y by adding 3 to x

MATH MEANING

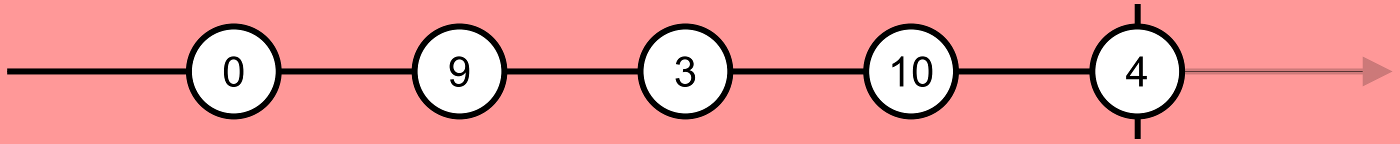


An equation

REACTIVE MEANING:

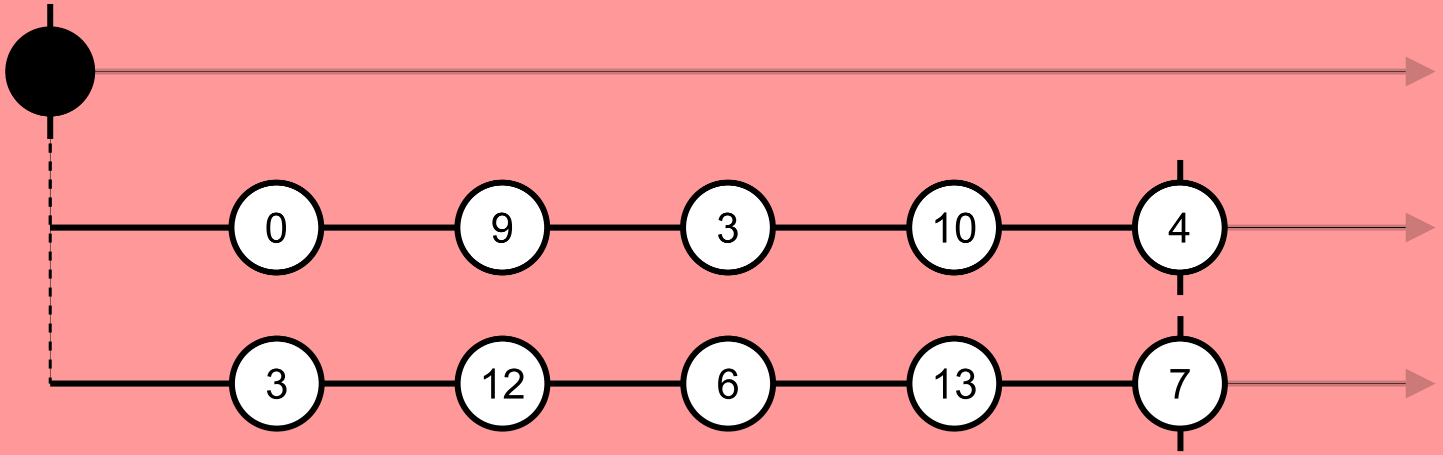
X is a value that can change over time. Y is always the value 3 greater than X

X VALUES OVER TIME



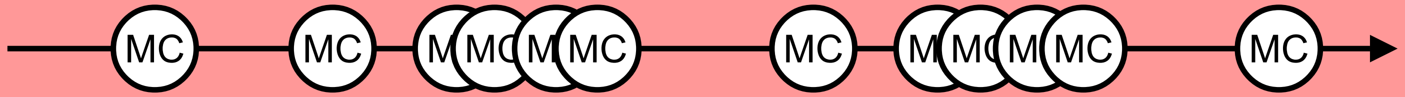
a data stream of numbers over time

Y VALUES OVER TIME



a data stream of numbers derived from another stream

MOUSE CLICKS OVER TIME



Stream of user input events

REACTIVE PROGRAMMING IN THE REAL WORLD?

OBSERVER PATTERN!

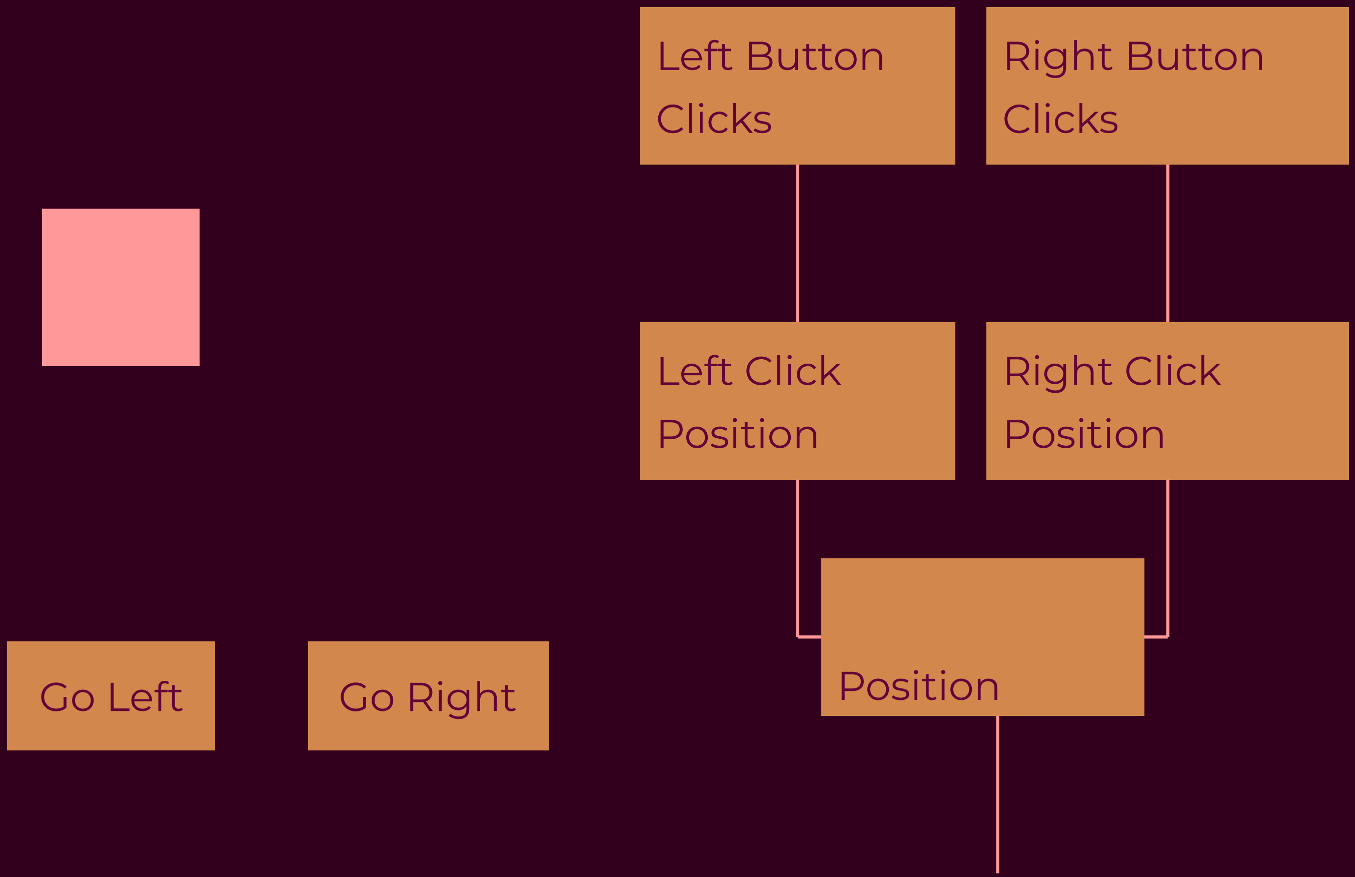
Only better...

OBSERVABLE:

A value that changes over time, that we can listen to changes on

Value as 'Observable'

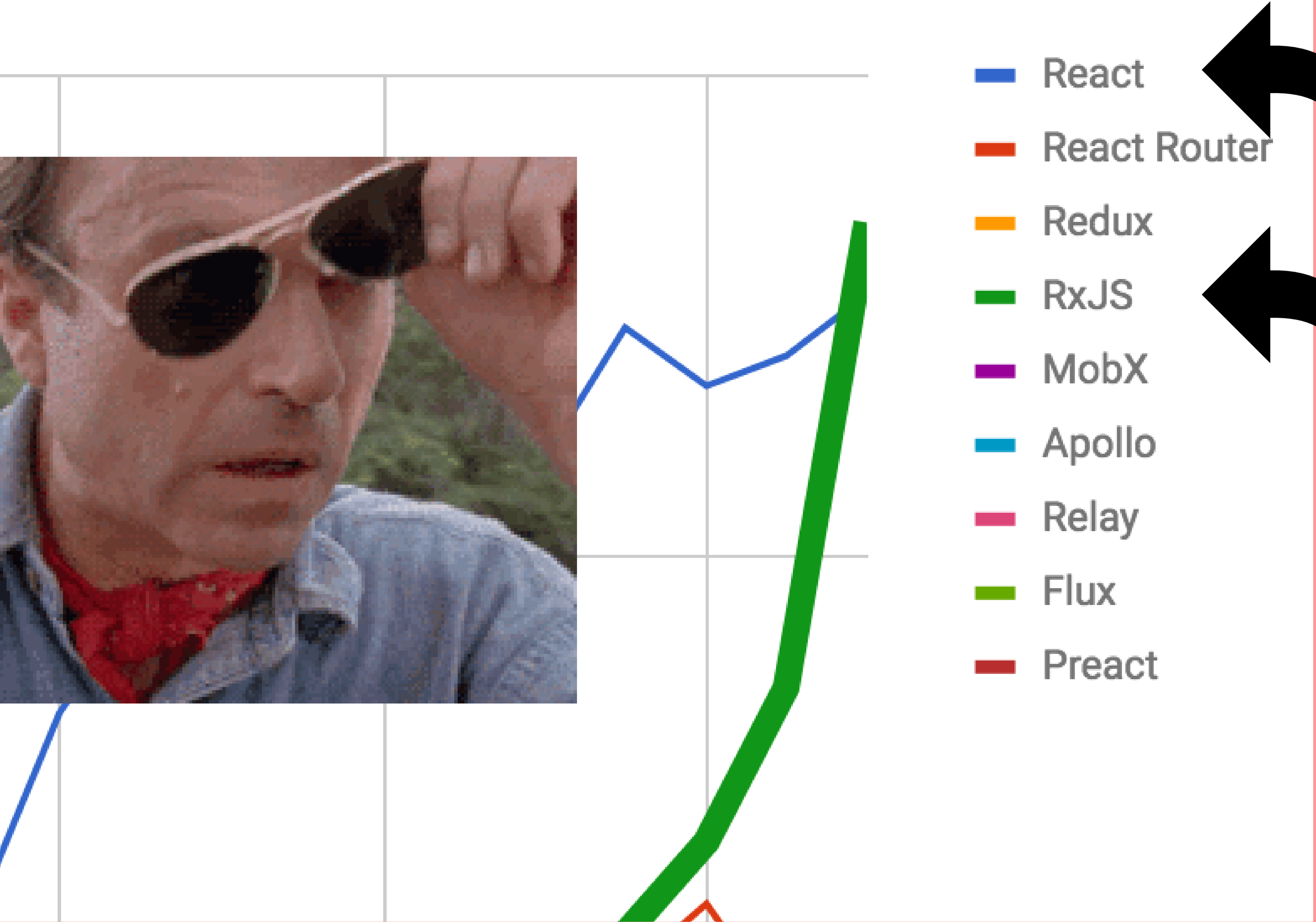
```
x.subscribe(nextX => console.log(nextX))

x.subscribe(nextX => console.log(nextX + 3))

y = "?";

x = [0, 9, 3, 10, 4]

y = x.map(nextX => nextX + 3)

x = Observable.of(0, 9, 3, 10, 4);

y = x.map(nextX => nextX + 3)
```


Left Button Clicks
Right Button Clicks
Left Click Position
Right Click Position
Position
Go Left
Go Right

How This Works

```
import {
  fromEvent,
  merge
} from "rxjs";
```

4. HOW DO I ACTUALLY USE THIS?


React
React Router
Redux
RxJS
MobX
Apollo
Relay
Flux
Preact

YOU'RE ALREADY USING IT.

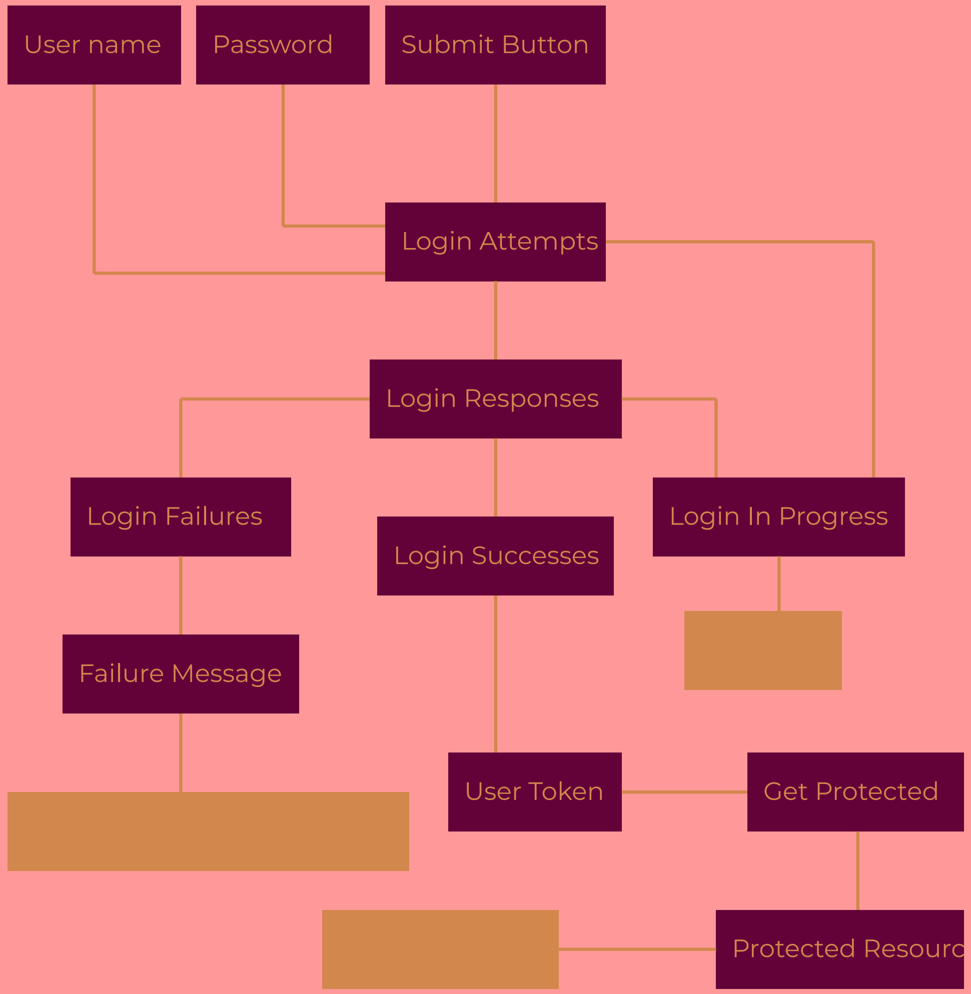
TWO QUESTIONS FOR USING RXJS

- How to architect applications with RxJS?
- How do I integrate this in my application, today?

User name:

Password:

Submit



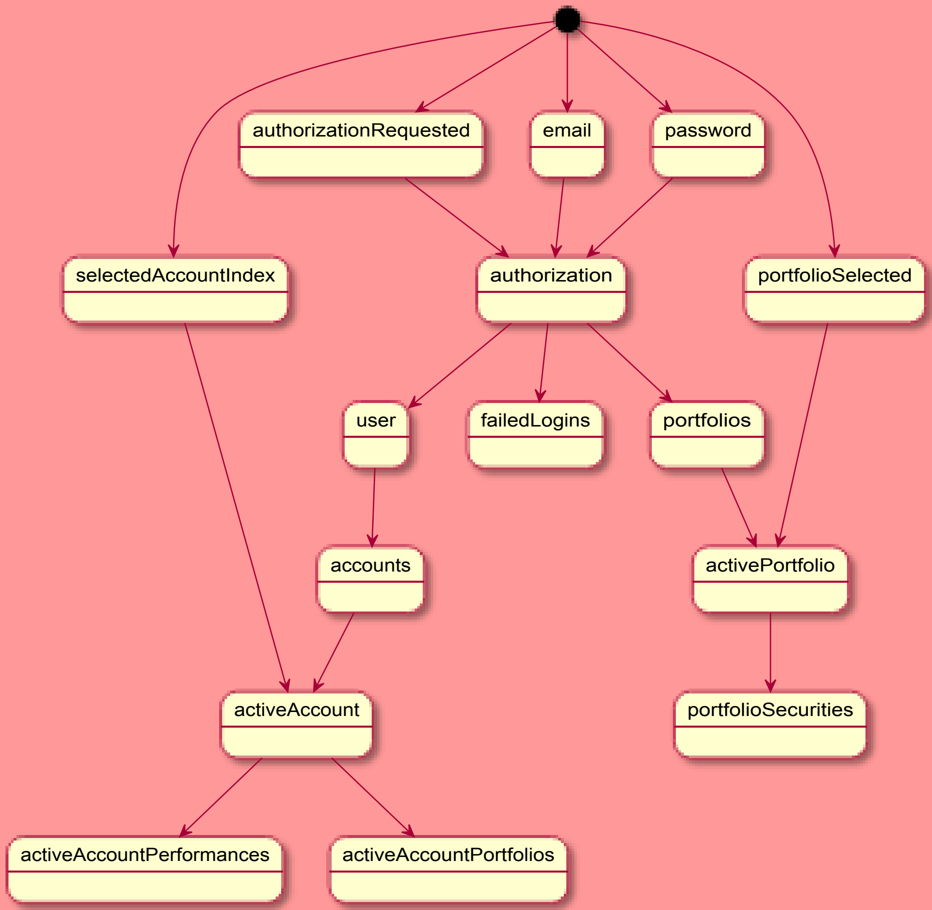
But how tho?

```
const api = {
  login: (username, password) => Promise,
  protectedResource: {
    get: (userToken) => Promise
  }
};

const username$ = new Subject();
const password$ = new Subject();
const submitButton$ = new Subject();

const loginAttempts$ = submitButton$.pipe(withLatestFrom(username$, password$));

const loginResponses$ = loginAttempts$.pipe(
  mergeMap(([_, username, password]) => api.login(
    username,
    password
```

SIGNAL GRAPH

How data propogates through your program

ACTUAL SIGNAL GRAPH FROM REAL APP



PRODUCTION CONCERNS

1. How do I test?
2. How do I make sure my graph is sound?
3. Ack RxJs idiosyncracies!
4. One big graph or lots of smaller ones?
5. Diagramming is hard

I liked Signal Graphs so much I bought the company!

- me, 2018

SIGNAL:

A library for frontend state management using signal graphs

Signal!

```
const signalGraph = new SignalGraphBuilder()
  .define(
    addPrimary('username$'),
    addPrimary('password$'),
    addPrimary('submitButton$'),
    addDerived(
      'loginAttempts$',
      makeLoginAttempts,
      'submitButton$',
      'username$',
      'password$'
    ),
    addDerived('loginResponses$', makeLoginResponses,
'loginAttempts$', 'api'),
    addDerived(
      'loginInProgress$',
      makeLoginInProgress,
      'loginAttempts$',
```

Available Now(ish):

@RXREACT/SIGNAL

Coming Soon:

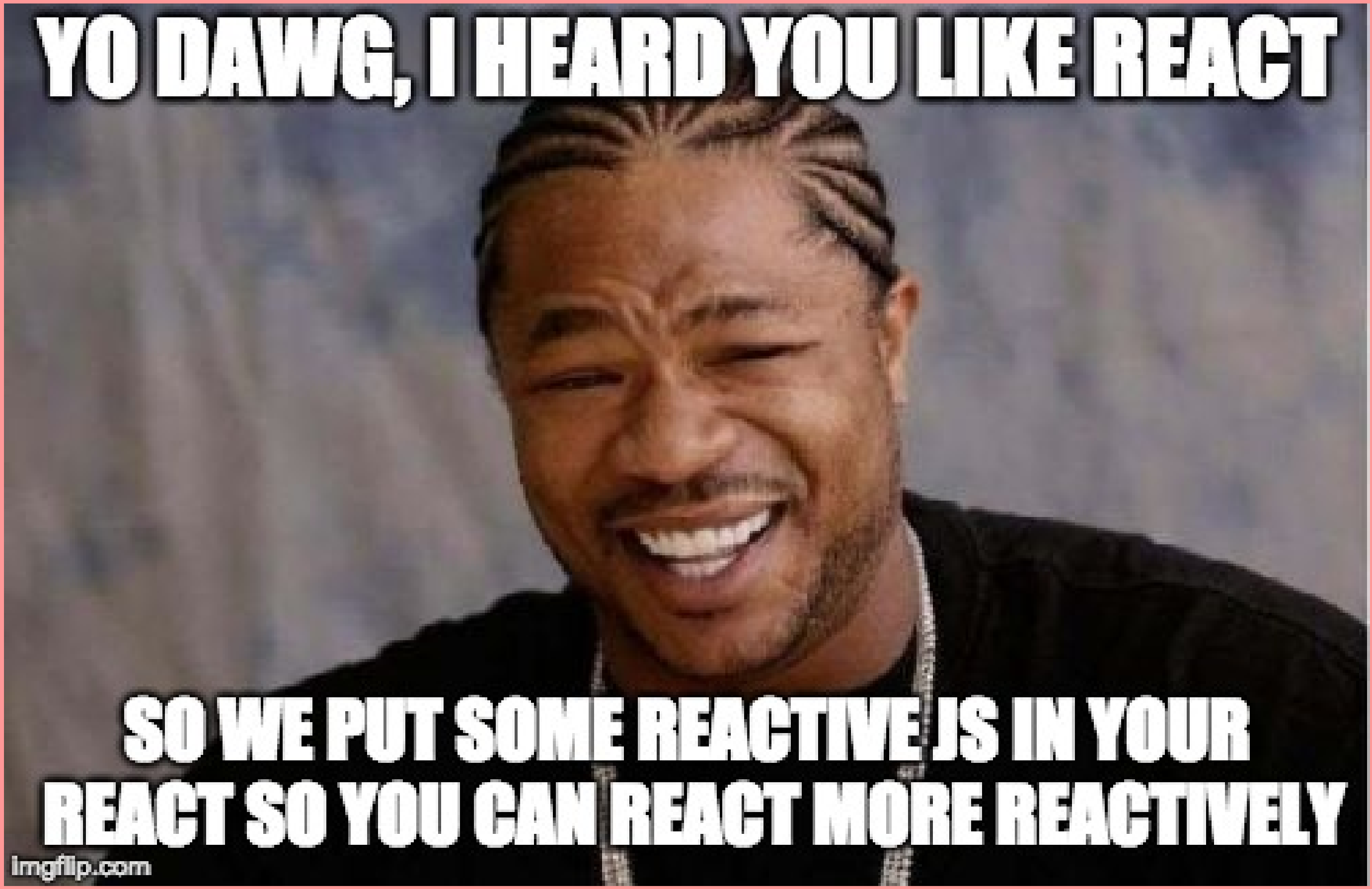
AUTOMATIC GRAPH VISUALIZATION

INTEGRATION

FRAMEWORK = ANGULAR

1. You're done
2. Check out NgRx

BUT WHAT ABOUT REACT?


YO DAWG, I HEARD YOU LIKE REACT
SO WE PUT SOME REACTIVE JS IN YOUR REACT SO YOU CAN REACT MORE REACTIVELY
Imgflip.com

GOOD NEWS!

RXREACT:

Tools for integrating react with RxJs!

Signal-connect

```
import { withViewModel } from '@rxreact/signal-connect'

const PositionedBox = ({ position }) => <RedBox pose={position} />

const ballSignalGraph = new SignalGraphBuilder().define(/*...*/).build()

// connect(graph, mapGraphOutputsToProps, mapGraphInputsToProps = {})
const BoundBox = connect(
  ballSignalGraph,
  {
    position: 'position$'
  }
)(PositionedBox)

const LeftButton = connect(
```

5.
USED CAR SALES PORTION

@RXREACT/CORE:

RxJs+React on it's own

RxReact Demo

```
import { withViewModel } from '@rxreact/core'

const PositionedBox = ({ position }) => <RedBox pose={position} />

const boxVm = {
  inputs: {
    position: position$
  }
}

const BoundBox = withViewModel(boxVm)(PositionedBox)

const LeftButton = withViewModel({
  outputs: {
    onClick: leftClick$
  }
```

WHAT ABOUT TYPESCRIPT?

I thought you might be worried... about the security... of your shit.

RXREACT 💖TYPESCRIPT

VIEW MODEL AS REDUCER?

```
let viewModel = viewModelFromReducer({
  initialState: {
    count: 2,
    fruit: 'bananas',
    extra: 'applesauce'
  },
  reducer(state, action) {
    switch (action.type) {
      case ActionType.SET_COUNT:
        return ReducerResult.Update({ ...state,
          count: action.payload
```

@RXREACT/PROCESS

THAT'S ALL FOLKS!

reactivex-talk.techgirlwonder.com
github.com/hannahhoward/reactivex-talk