React JS

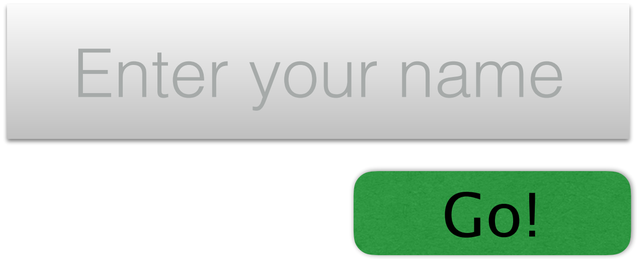
Flux Pattern: Introduction

Saikat Bhattacharya
Technology Lead | DEP

Recap

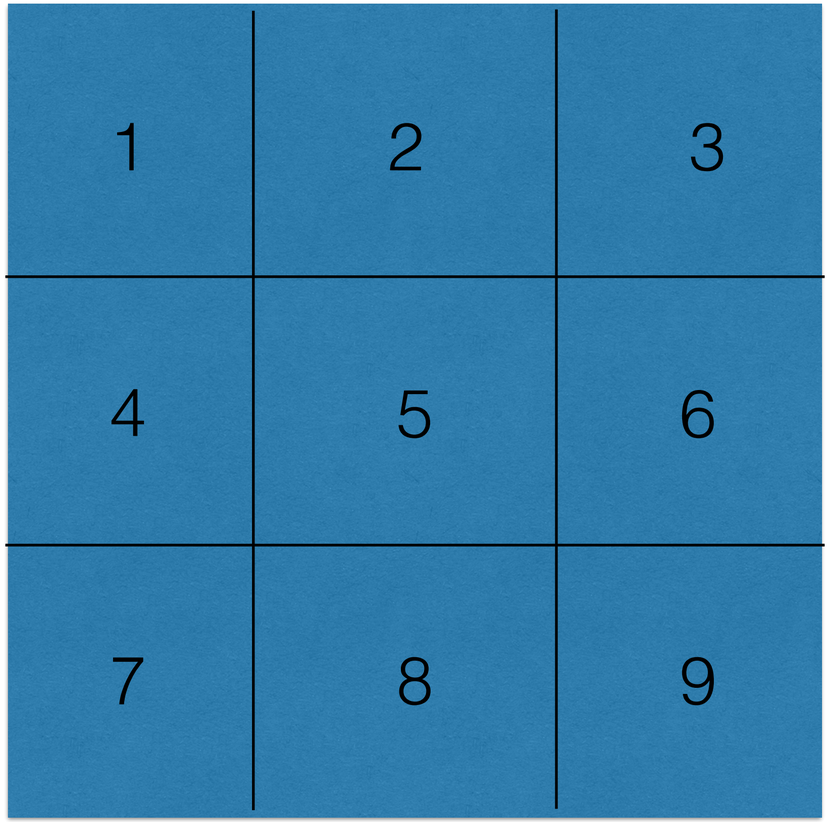
- React JS is a ~~framework~~ library
- React JS plays mostly the role of ‘View’ in MVC
- React JS uses JSX, most commonly for rendering
- React JS believes in Virtual DOM
- State and Props: the two objects that carry information
- `react-router`: A module for switching between components

Task



Task

Welcome
Saikat



Your Score:

1000

Time left:

30

Rule of game: Click the numbers to construct a prime number. If you can construct a ‘x’ digit prime number, you will be awarded 10^x points. For example, if you click 3 and press arrow, you will get 10 and if you click 3 and 7 (i.e you are constructing 37) you will be awarded 100 points. The winner will have to score maximum points within 30 secs time limit.

Task

Your score: 1000

Play again

Today's Plan

- What is Flux?
- Difference between Flux and MVC
- Action, Dispatcher, Store and View

What is ‘Flux’?

flux

/flʌks/

noun

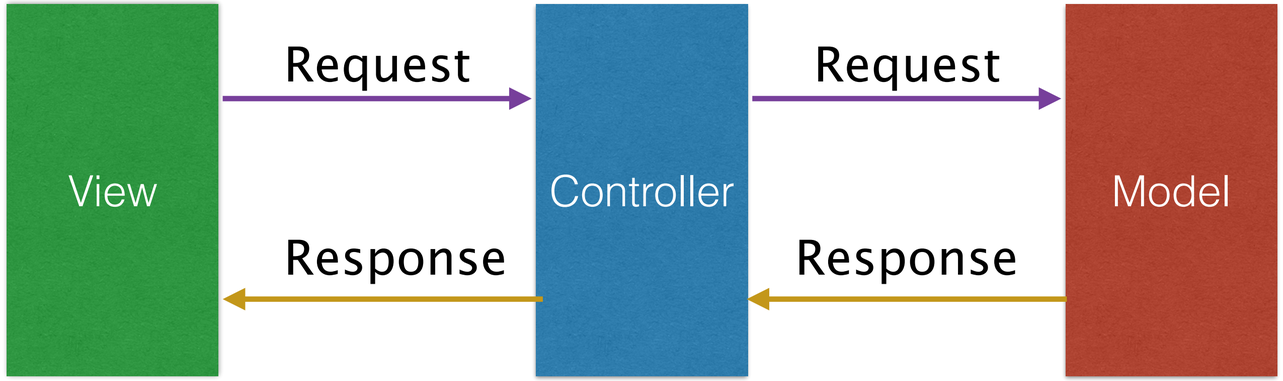
1. the action or process of flowing or flowing out.
   "the flux of ions across the membrane"

This term was introduced into field of science by Isaac Newton

What is ‘Flux’?

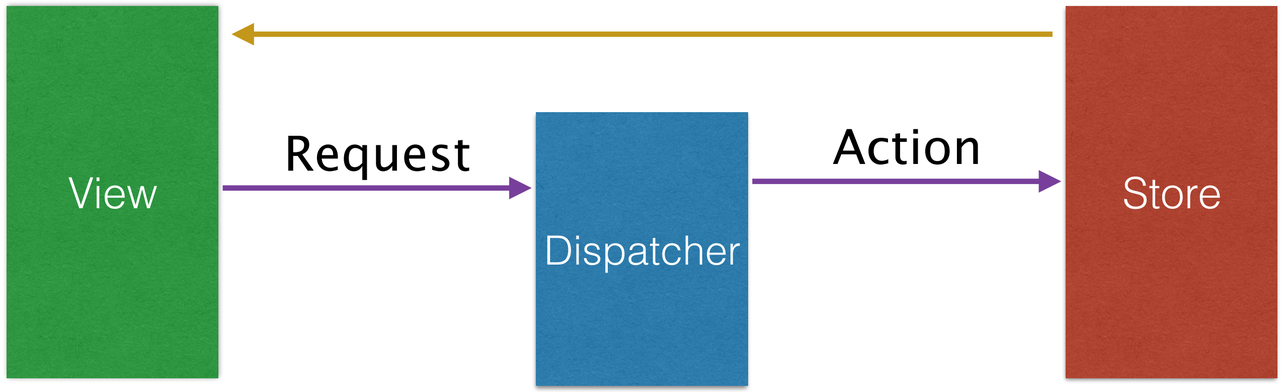
- It's more of a pattern rather than a formal framework
- It complements React's composable view components by utilizing a unidirectional data flow
- Three major parts: the dispatcher, the stores, and the views

MVC vs Flux



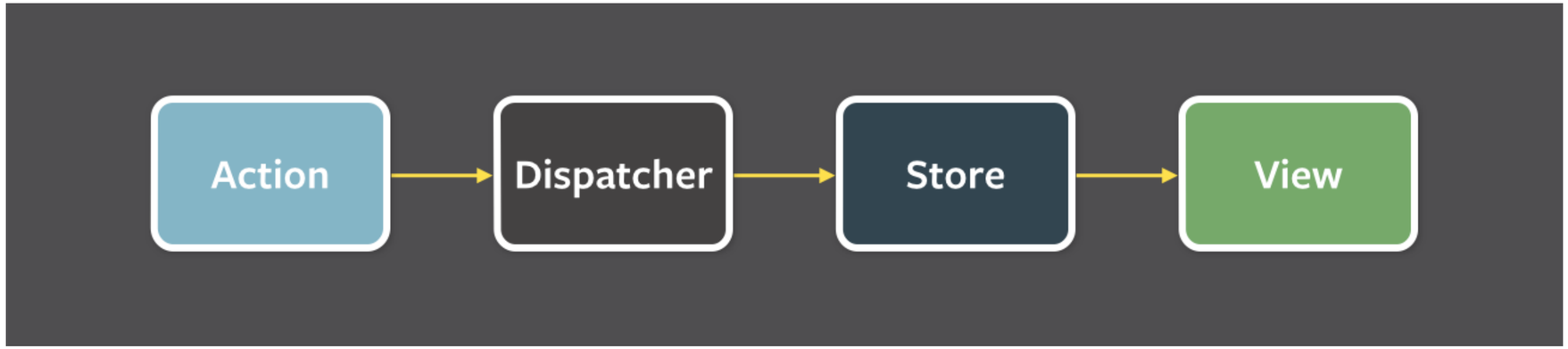
Traditional MVC data flow

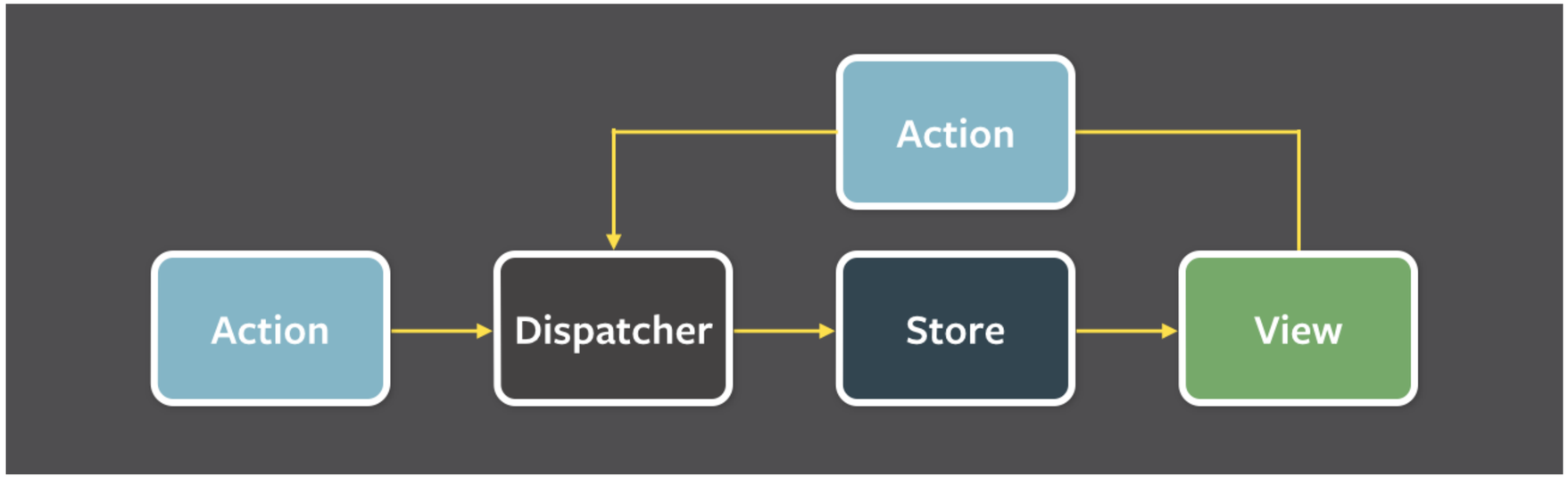
MVC vs Flux



Flux data flow

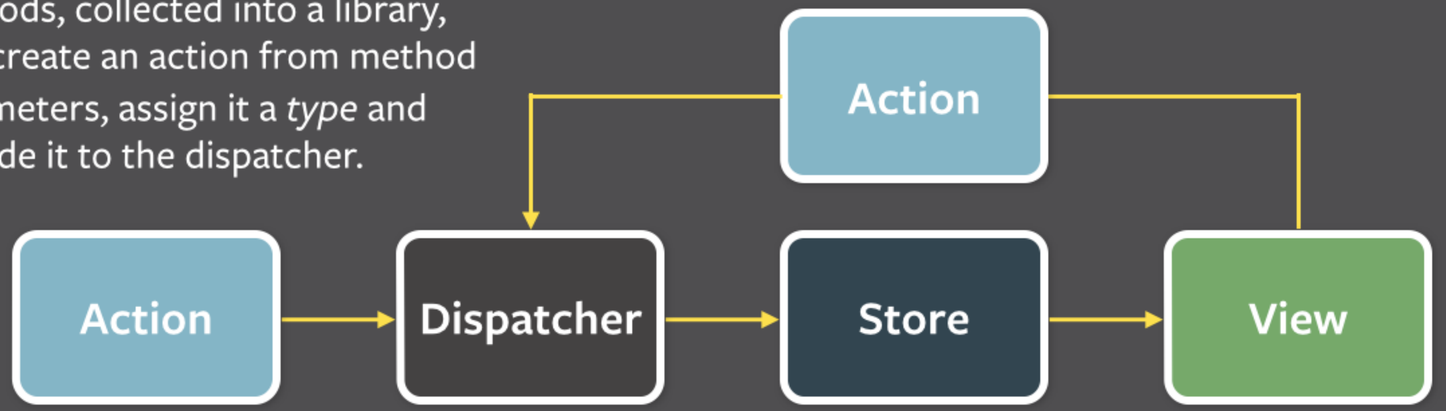
Action, Dispatcher, Store and View





Action, Dispatcher, Store and View

Action creators are helper methods, collected into a library, that create an action from method parameters, assign it a *type* and provide it to the dispatcher.



Every action is sent to all stores via the *callbacks* the stores register with the dispatcher.

After stores update themselves in response to an action, they emit a *change* event.

Special views called *controller-views*, listen for *change* events, retrieve the new data from the stores and provide the new data to the entire tree of their child views.