React

18.08.2016

Fatih Şimşek - Software Infrastructure

React

Open source Javascript Library

Created by Jordan Walke, a software engineer at Facebook

First deployed on Facebook's newsfeed in 2011

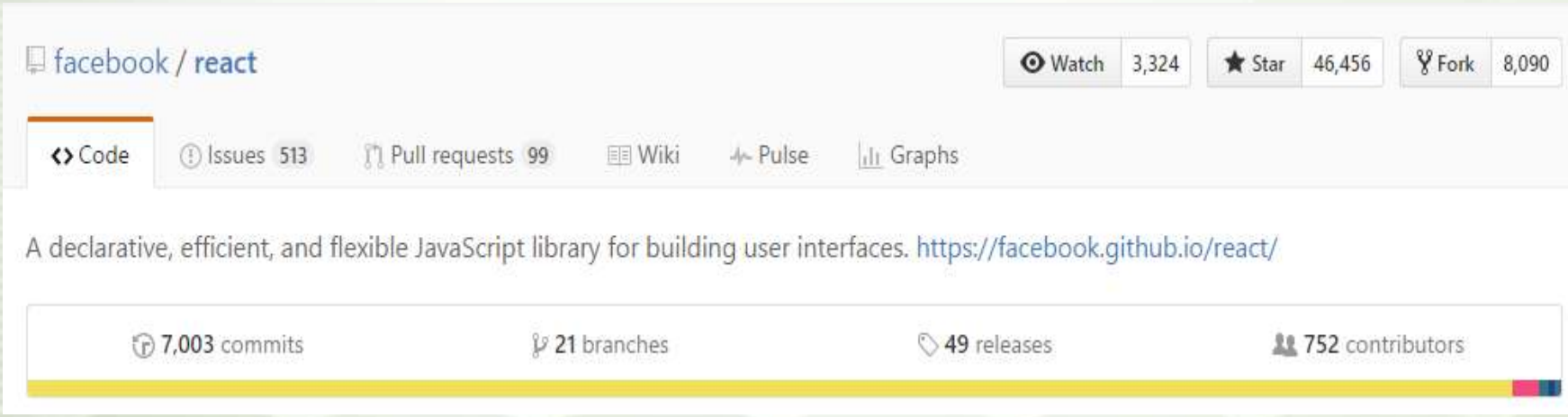
Maintained by **Facebook**, **Instagram** and a community of individual developers and corporations

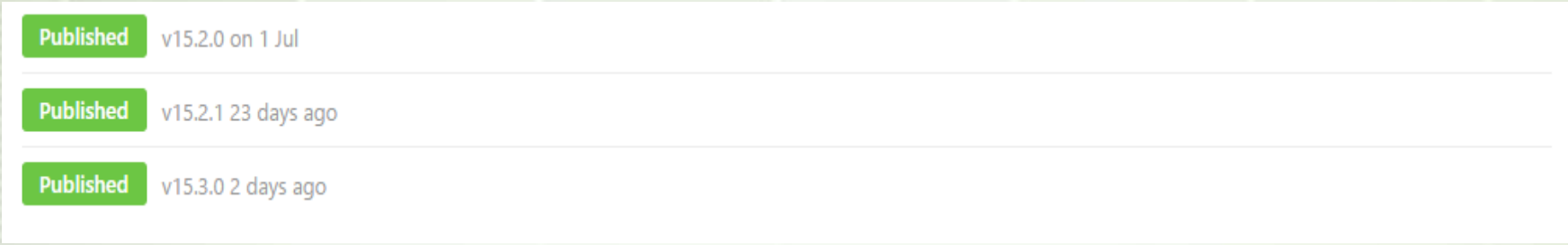
Who is using React?

9flats - Airbnb - Alipay - **Atlassian** - **BBC** - Box - Capital One - **Coursera** -**Dailymotion** - Deezer - **Docker** - Expedia - **Facebook** - Fotocasa - HappyFresh - **IMDb** - Instacart - **Instagram** - Khan Academy - Klarna - Lyft - NBC - **Netflix** - NFL - Paypal - Periscope - Ralph Lauren - **Reddit** – **Salesforce - Sberbank** - **Stack Overflow** - **Tesla** - Tmall - The New York Times - **Twitter Fabric** - **Twitter Mobile** - **Uber** - **WhatsApp** - Wired - **Yahoo** - Zendesk

https://github.com/facebook/react/wiki/Sites-Using-React

Github


facebook / react
Watch 3,324
Star 46,456
Fork 8,090
Code
Issues 513
Pull requests 99
Wiki
Pulse
Graphs
A declarative, efficient, and flexible JavaScript library for building user interfaces. https://facebook.github.io/react/
7,003 commits
21 branches
49 releases
752 contributors


Published
v15.2.0 on 1 Jul
Published
v15.2.1 23 days ago
Published
v15.3.0 2 days ago

React

Creating user interface(V in MVC model)

Speed

Declarative

Composable

Learn Once, Write Anywhere

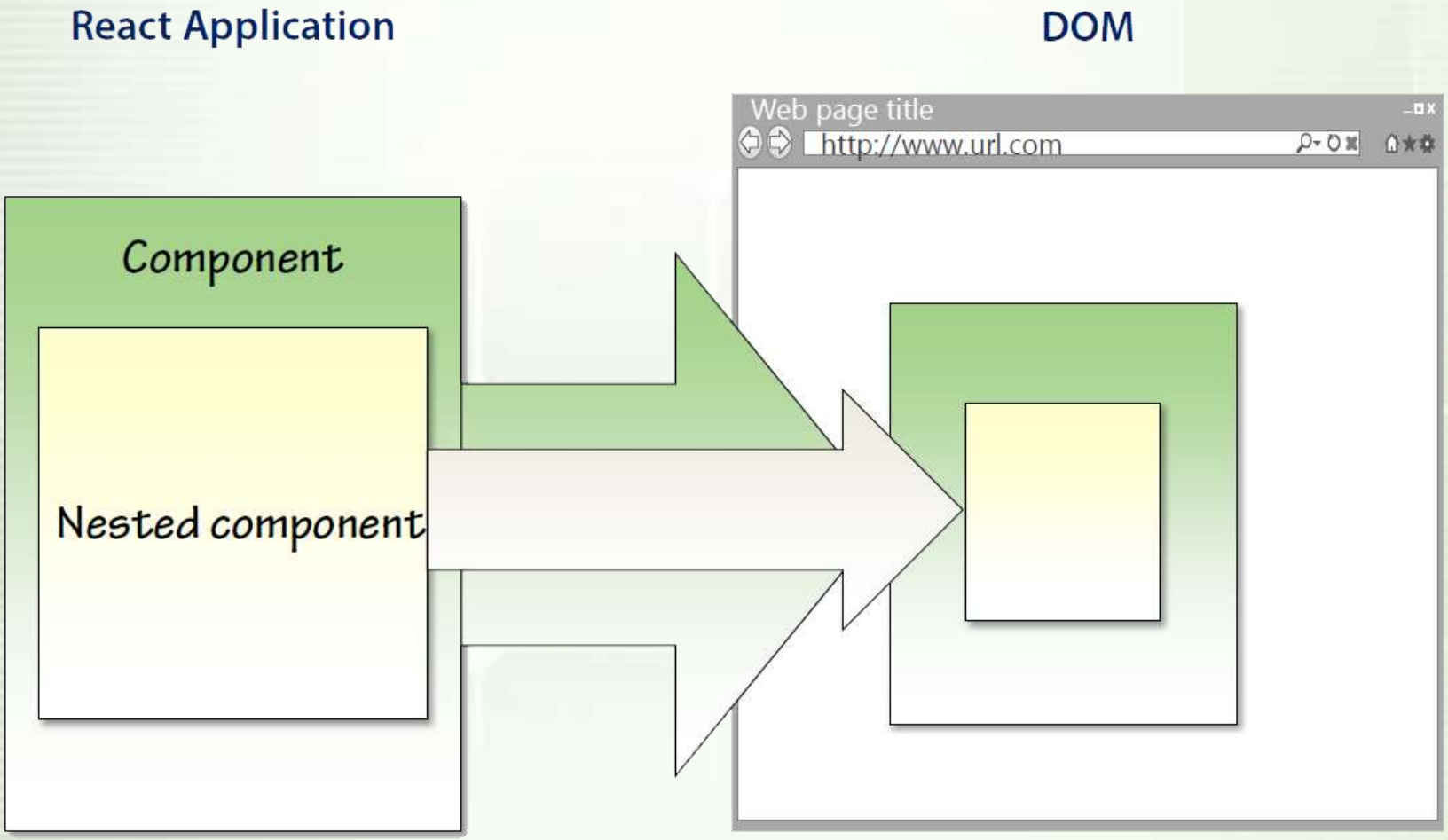
Support ES6

Testable

React

Write once, Run Everywhere

Architecture



Architecture

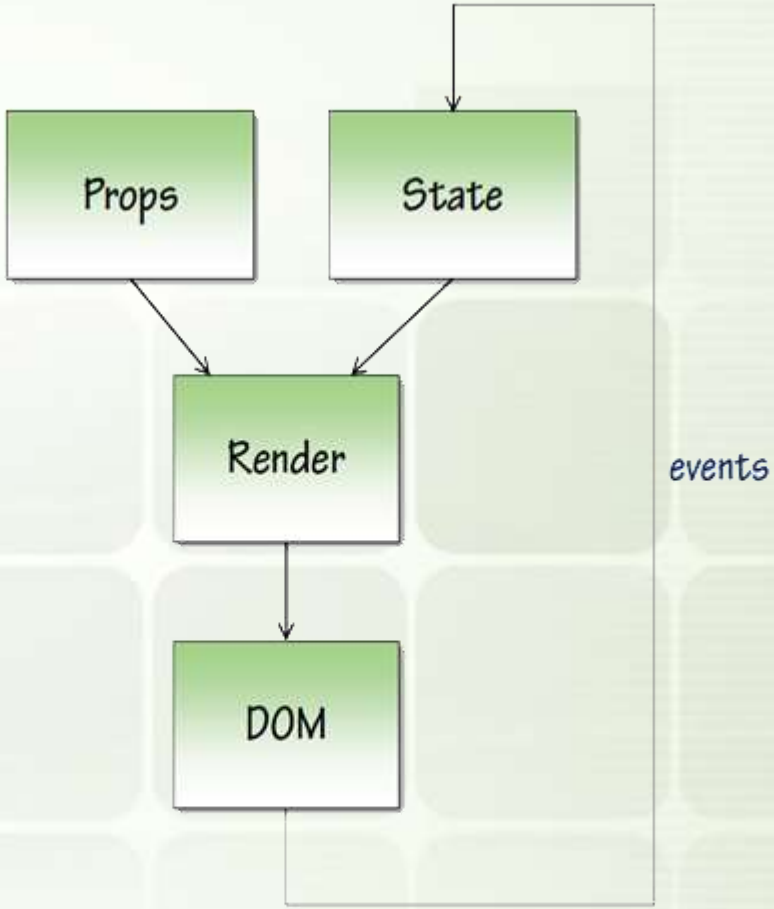
Component manage own state

One way binding

JSX

Virtual DOM

Component Lifecycle



Model+ component = DOM

JSX

Supports xml-like syntax inline in JavaScript

Each element is transformed into a Javascript function call

- <Hello /> => Hello(null)
- => React.DOM.div(null)

Virtual DOM

Problem:

- DOM manipulation is expensive
- Touching DOM is hard to test
- Re-render all parts of DOM make your app slowly

Virtual DOM

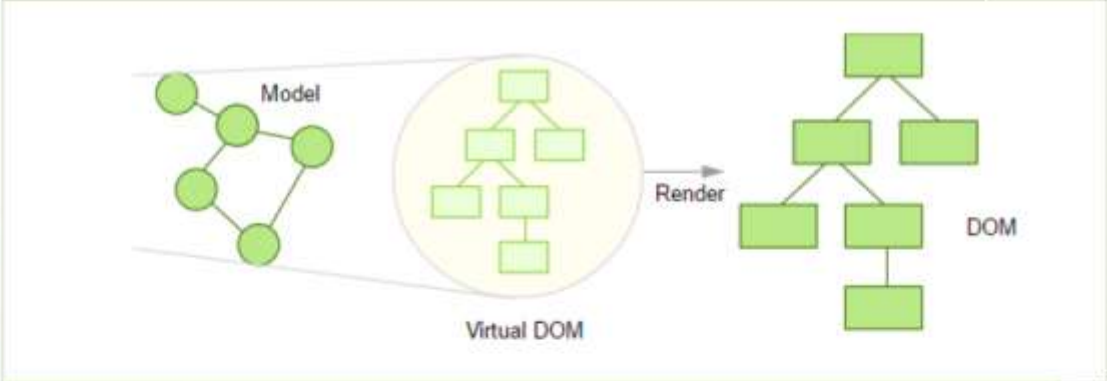
Solution:

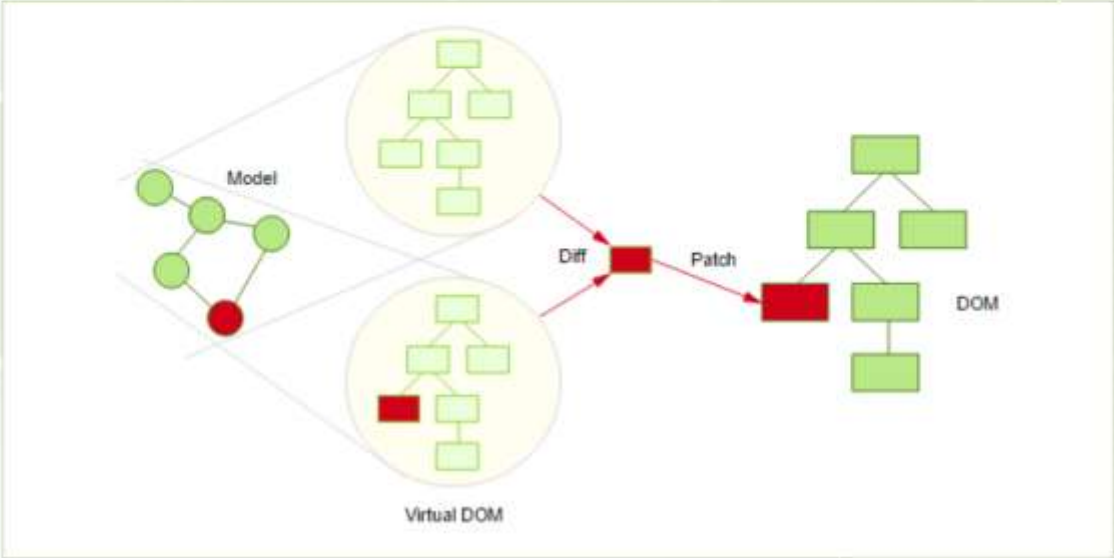
When the component's state is changed, React

- Use different algorithm with the browser DOM tree to identify the changes
- Instead of creating new object, React just identify what change is took place and once identify update that state
- Render the subtree of DOM elements into the rendering of the DOM

Virtual DOM

Only diff changes from the two V-DOMs are applied to real DOM





DEMO

Prop / State

Prop:

- Used to pass parameter from parent to children
- PropTypes used to validate properties
- getDefaultProps

State:

- Used to manage state inside component
- getInitialState
- setState

Prop / State

Features	props	state
Can get initial value from parent Component?	Yes	Yes
Can be changed by parent Component?	Yes	No
Can set default values inside Component?	Yes	Yes
Can change inside Component?	No	Yes
Can set initial value for child Components?	Yes	Yes
Can change in child Components?	Yes	No

Mixins / Statics

Mixins:

- Different components may share some common functionality
- Several mixins can be defined
- Methods defined on mixins run in the order mixins were listed

Statics:

- Like .Net / Java static method
- Define static methods in statics block of component

Events

componentWillMount: Invoked once before the initial render

componentDidMount: Invoked once, only on the client

componentWillReceiveProps: Invoked when a component is receiving new props

shouldComponentUpdate: Invoked before rendering when new props or state are being received

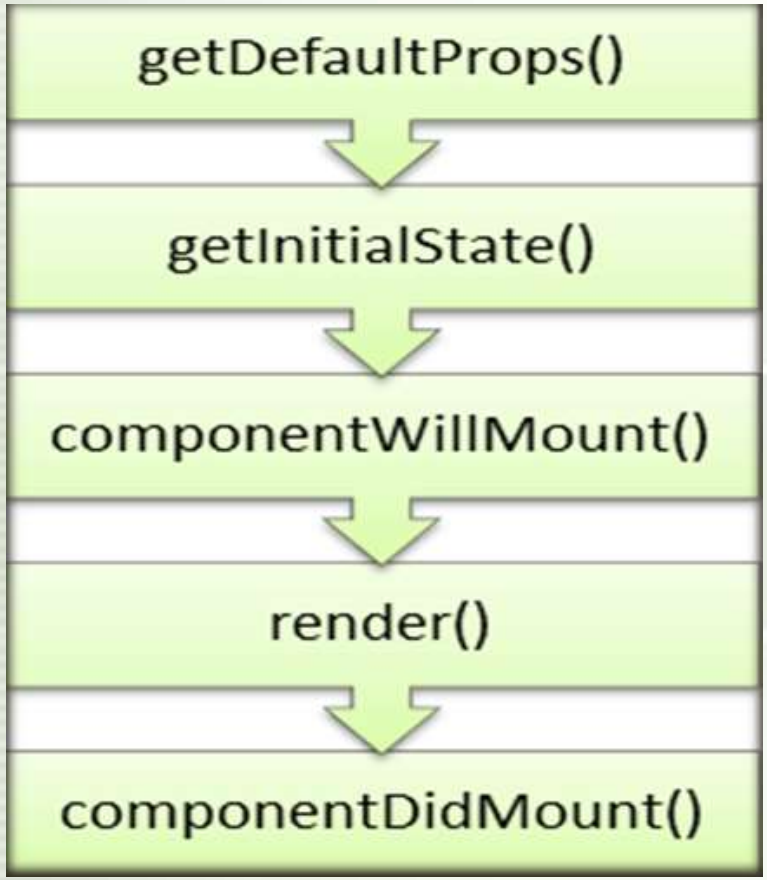
componentWillUpdate: Invoked immediately before rendering when new props or state are being received

componentDidUpdate: Invoked immediately after the component's updates are flushed to the DOM

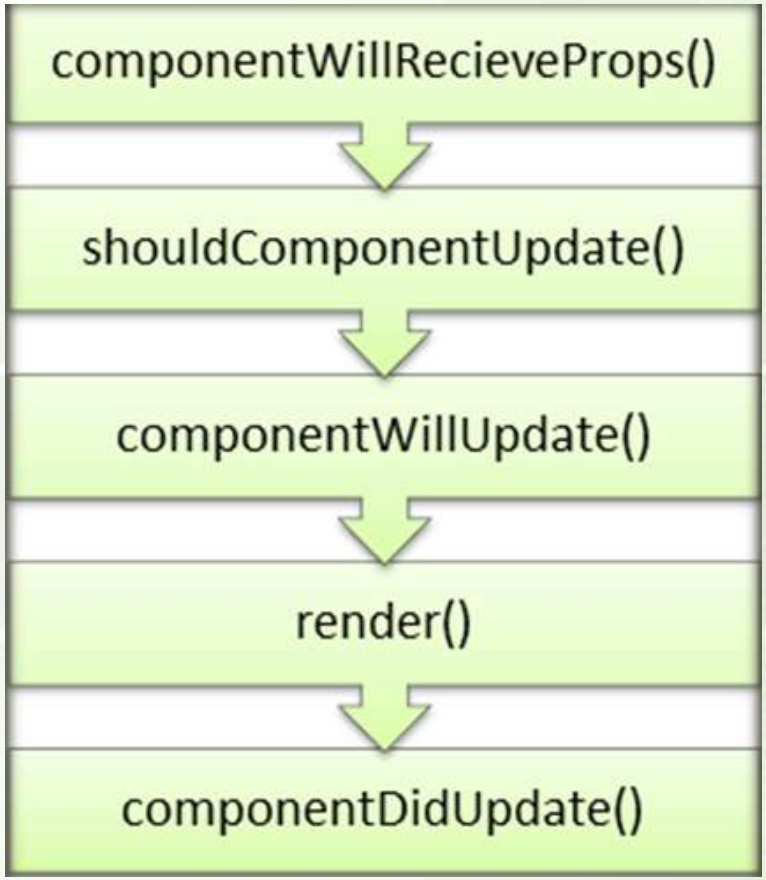
componentWillUnmount: Invoked immediately before a component is unmounted from the DOM

Lifecycle

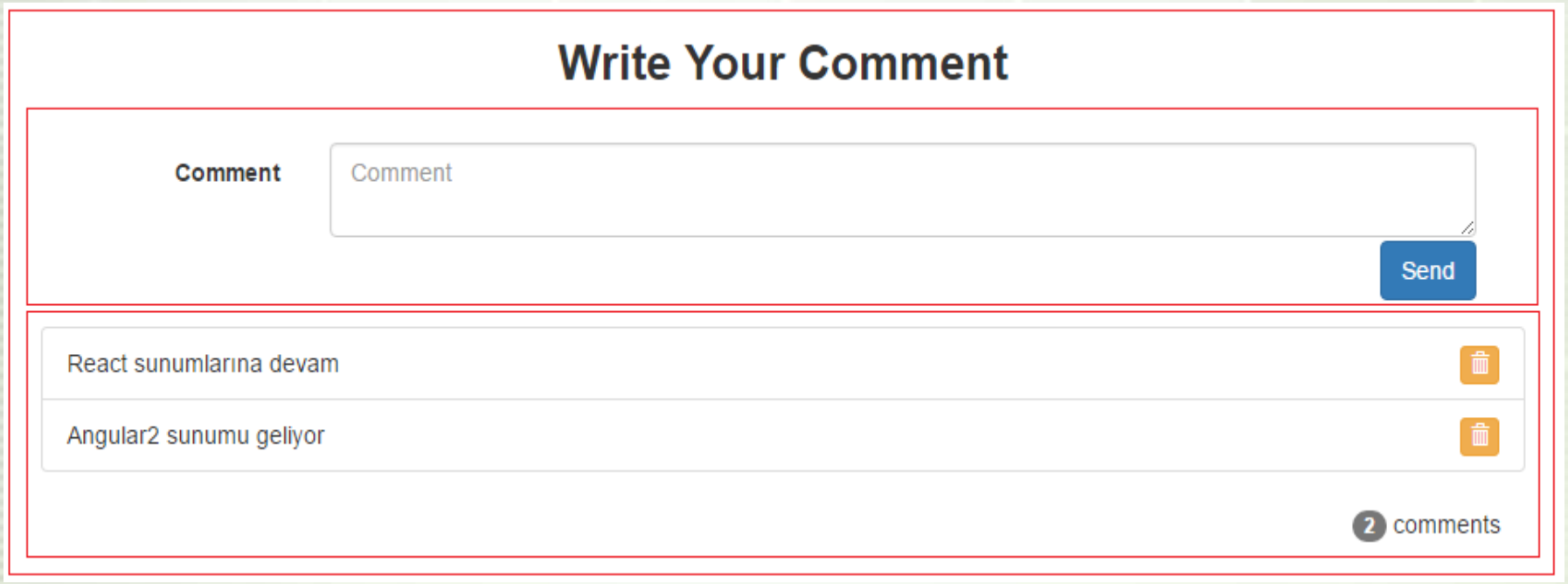
Initial Phase



Update Phase



DEMO


Write Your Comment
Comment
Comment
Send
React sunumlarına devam
Angular2 sunumu geliyor
2 comments


10
min

Flux

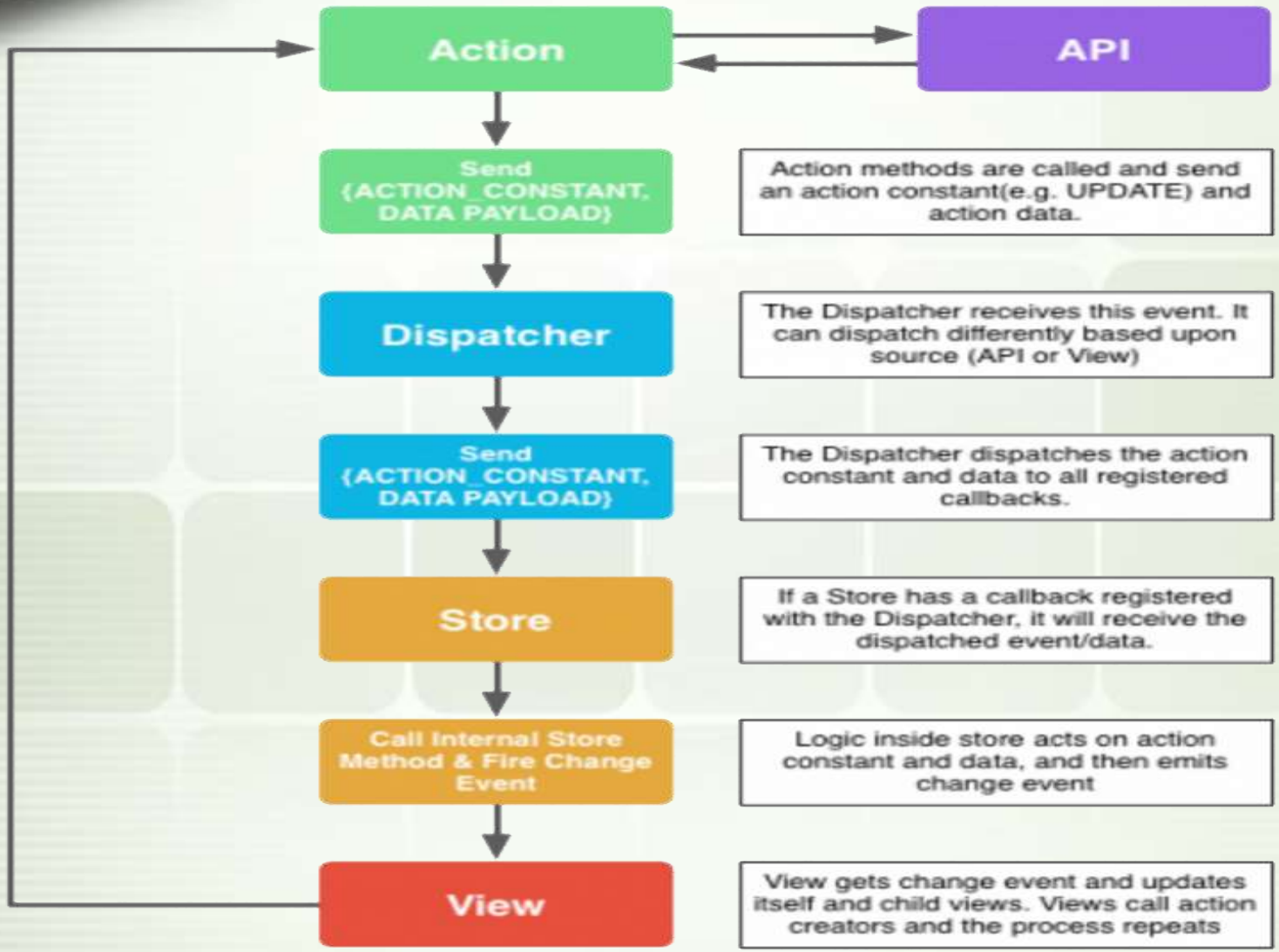
Action: Helper method that facilitate passing data to the Dispatcher

Dispatcher: Receive action and broadcast payload to registered callback

Store: Containers for application state & logic that have callbacks registered to the dispatcher

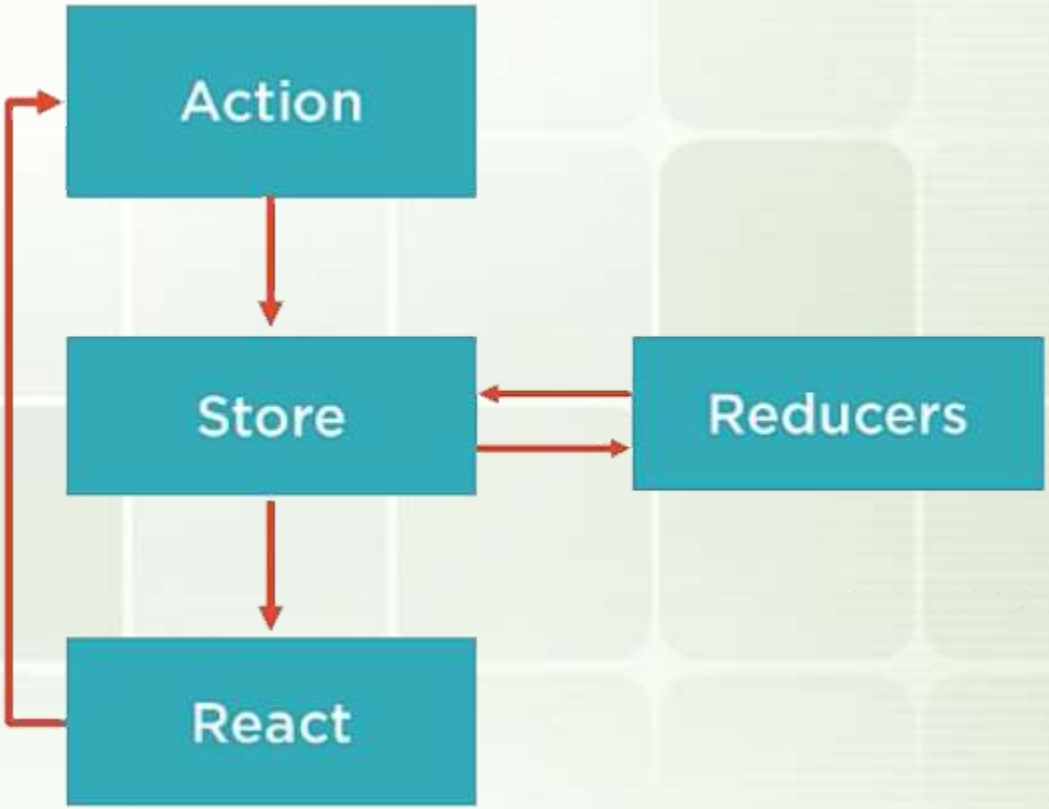
View: State from Stores and pass it down via props to child components

Flux


Action
API
Send {ACTION_CONSTANT, DATA PAYLOAD}
Action methods are called and send an action constant(e.g. UPDATE) and action data.
Dispatcher
The Dispatcher receives this event. It can dispatch differently based upon source (API or View)
Send {ACTION_CONSTANT, DATA PAYLOAD}
The Dispatcher dispatches the action constant and data to all registered callbacks.
Store
If a Store has a callback registered with the Dispatcher, it will receive the dispatched event/data.
Call Internal Store Method & Fire Change Event
Logic inside store acts on action constant and data, and then emits change event
View
View gets change event and updates itself and child views. Views call action creators and the process repeats

Redux

- **Single source** of truth
- State is **read-only**
- Mutations are written as **pure function**s (reducers)



Redux

- Important difference to Flux: no dispatching inside the action
- There is no Dispatcher at all; pure functions do not need to be managed, they need to be composed
- Mutations are written as pure functions

Benchmark

Duration in milliseconds (Slowdown = Duration / Fastest)

	angular v1.5.7	angular v2.0.0-rc4	aurelia v1.0.0-rc1.0.0	cyclejs v6.0.3	cyclejs v7.0.0	ember v2.6.1	inferno v0.7.13	mithril v0.2.5	plastiq v1.30.1	preact v4.8.0	ractive v0.7.3	react-lite v0.15.14	react v0.14.8	react v15.2.0	react v15.2.0-mobX-v2.3.3	tsers v1.0.0	vidom v0.3.6	vue v1.0.26	vanillajs
create rows Duration for creating 1000 rows after the page loaded.	249.56	192.38	220.08	808.74	181.13	747.01	153.6	322.62	180.23	207.68	548.84	169.04	251.61	184.48	201.17	289.39	144.65	259.77	126.18
replace all rows Duration for updating all 1000 rows of the table (with 5 warmup iterations).	262.91	210.64	131.69	725.63	106.59	583.99	160.18	278.42	195.87	194.53	96.8	230.35	248.28	197.82	211.2	135.03	157.08	261.84	70.13
partial update Time to update the text of every 10th row (with 5 warmup iterations).	16.41	11.87	11.71	584.41	48.02	37.67	13.7	97.05	20.35	21.77	35.9	29.1	16.2	18.78	20.75	42.65	18.51	15.76	11.51
select row Duration to highlight a row in response to a click on the row. (with 5 warmup iterations).	7.85	5.08	53.89	557.97	38.85	42.24	6.31	75.71	8.38	5.6	10.89	17.81	6.43	7.07	6.51	30.13	12.52	7.33	5.98
swap rows Time to swap 2 rows on a 1K table. (with 5 warmup iterations).	50.53	51.76	53.8	555.23	43.85	74.69	48.09	138.3	51.49	60.01	32.02	65.85	53.37	53.05	55.71	35.51	58.97	52.96	7.83
remove row Duration to remove a row. (with 5 warmup iterations).	70.72	133.52	98.35	584.52	72.84	92.95	61.91	138.56	69.57	65.9	199.34	79.76	68.67	64.89	66.28	79.13	73.33	70.61	63.33
create many rows Duration to create 10,000 rows	2573.01	1842	1857.53	59304.35	1806.99	5905.1	1488.96	3258.88	1870.98	2656.55	5994.12	2192.9	3002.65	1858.08	1941.14	2937.51	1415.71	2705.86	1265.22
append rows to large table Duration for adding 1000 rows on a table of 10,000 rows.	826.88	679.57	700.63	59569.98	729.01	1211.8	295.41	1881.17	294.75	459.71	1431.26	1755.8	494.86	326.41	344.25	701	319.6	743.44	225.3
clear rows Duration to clear the table filled with 10.000 rows.	840.63	436.54	604.56	253.33	240.71	1182.97	227.16	294.11	247.11	358.09	2295.3	394.49	385.48	2000.85	2027.88	310.89	223.47	434.58	222.16
clear rows a 2nd time Time to clear the table filled with 10.000 rows. But warmed up with only one iteration.	1644.59	411.59	615.63	274.29	236.9	1617.69	227.97	296.14	238.25	345.9	1500	342.1	366.03	4117.91	4127.69	317	773.1	435.33	193.3
slowdown geometric mean	2.41	1.89	2.17	16.1	1.78	4.35	1.31	3.46	1.47	1.77	3.40	2.14	1.84	2.42	2.53	2.00	1.56	1.97	1.00

Memory allocation in MBs

	angular v1.5.7	angular v2.0.0-rc4	aurelia v1.0.0-rc1.0.0	cyclejs v6.0.3	cyclejs v7.0.0	ember v2.6.1	inferno v0.7.13	mithril v0.2.5	plastiq v1.30.1	preact v4.8.0	ractive v0.7.3	react-lite v0.15.14	react v0.14.8	react v15.2.0	react v15.2.0-mobX-v2.3.3	tsers v1.0.0	vidom v0.3.6	vue v1.0.26	vanillajs
ready memory Memory usage after page load.	4.86	17.05	21.59	4.72	3.32	10.1	3	2.68	2.68	2.64	4.09	3	4	4.25	5.17	6.67	2.89	3.36	2.43
run memory Memory usage after adding 1000 rows.	13.37	27.53	27.95	9.47	6.65	37.17	6.35	8.89	8.77	6.16	22.24	13.78	11.02	9.71	11.22	13.8	6.13	12.85	3.13