# React data flow and lifecycle

adapt

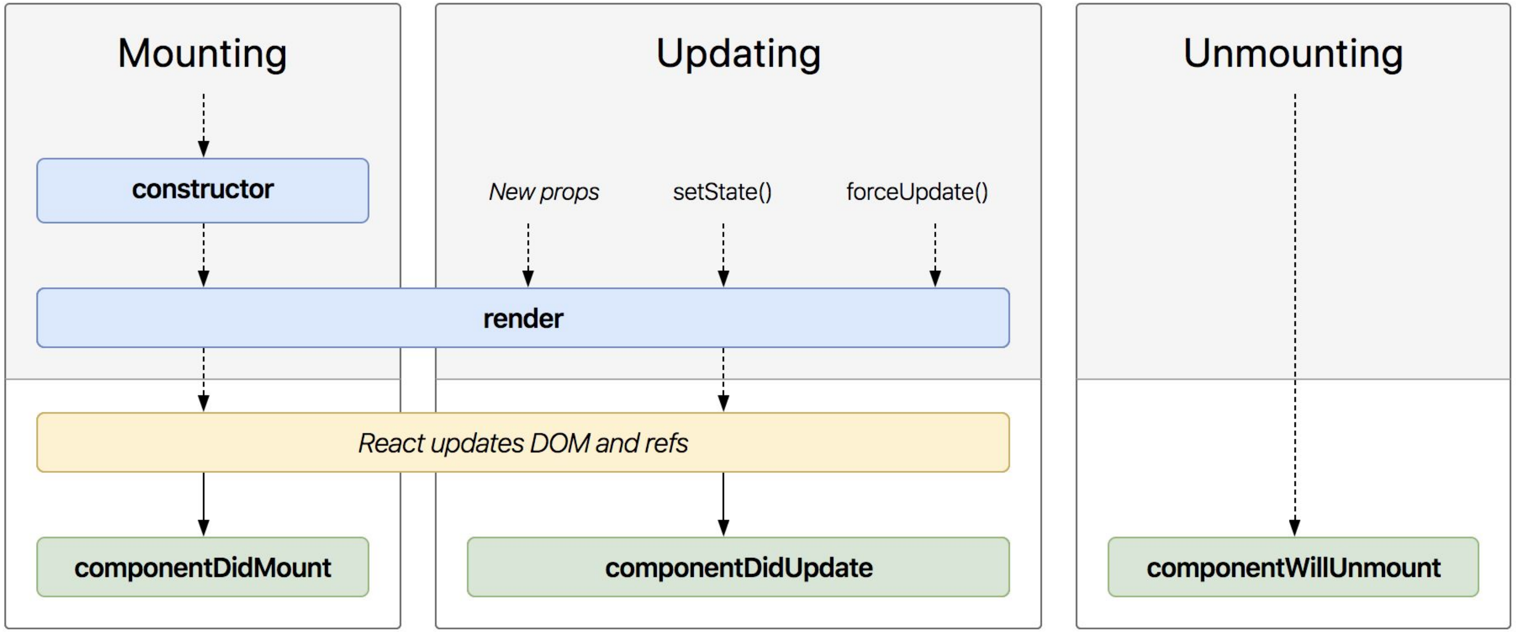
# Common react life-cycle

“Render phase”

Pure and has no side effects. May be paused, aborted or restarted by React.

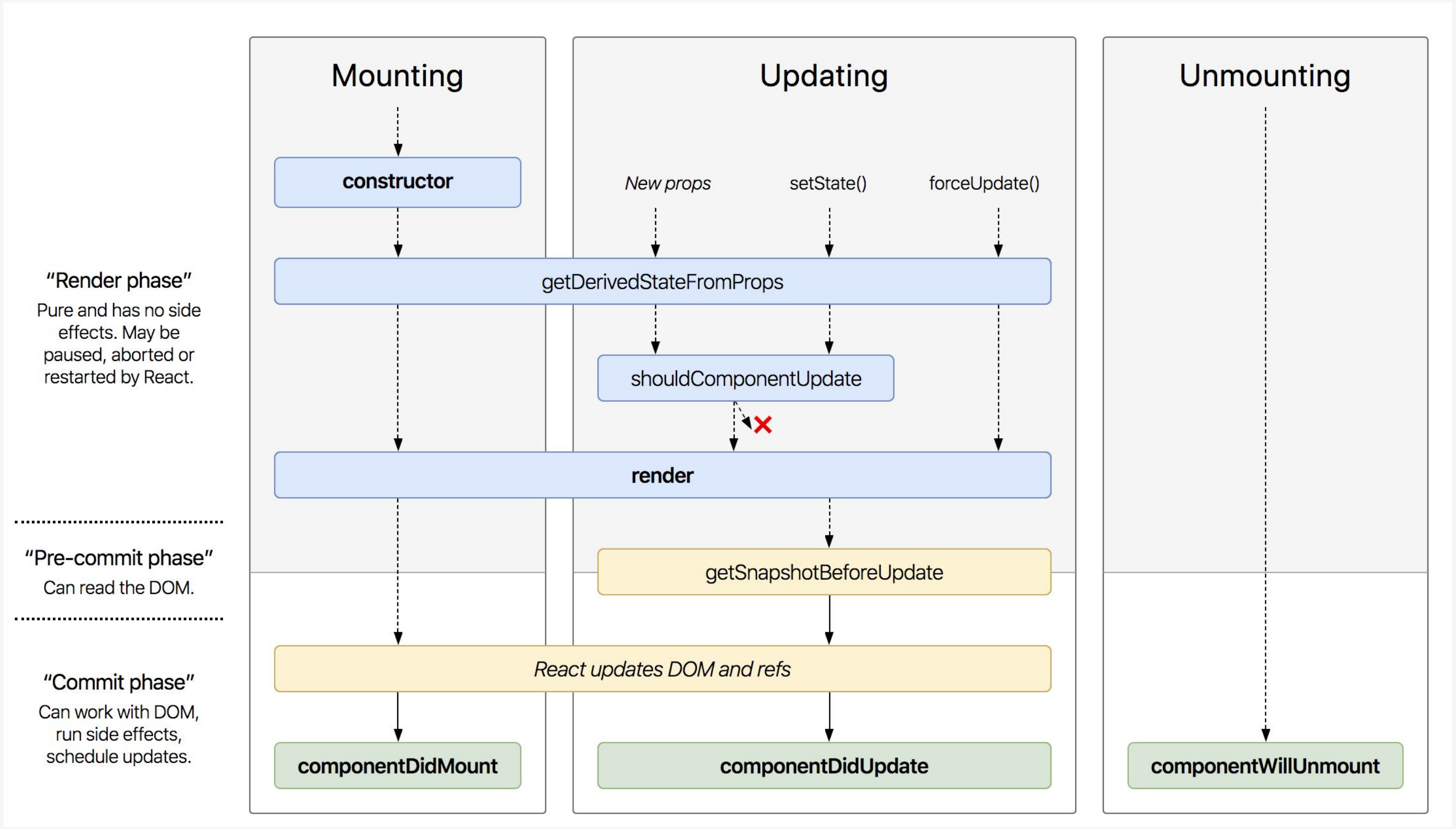
“Commit phase”

Can work with DOM, run side effects, schedule updates.



adapt

# All-inclusive react life-cycle

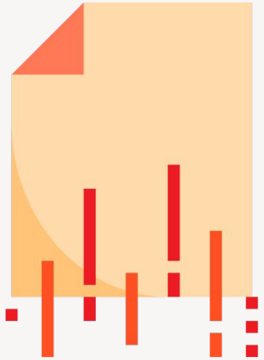


adapt

## Mounting

These methods are called in the following order when an instance of a component is being created and inserted into the DOM:

- constructor()
- static getDerivedStateFromProps()
- render()
- componentDidMount()



## constructor()

constructor(props)

Used for:

- Initializing state.
- Binding event handlers.

Not used for:

- Doing state changes or calling external service.

Don’t forget:

- User super(props);

## static getDerivedStateFromProps()

static getDerivedStateFromProps(props, state)

Invoked right before calling the render method, both on the initial mount and on subsequent updates. It should return an object to update the state, or null to update nothing.

Used for:

- Control state depending on props.

Don’t forget:

- You probably don’t need it.
- It does not have access to component instance.

## componentDidMount()

Used for:

- Subscribing to event.
- Changing state or calling external services.

## Updating

An update can be caused by changes to props or state. These methods are called in the following order when a component is being re-rendered:

- static getDerivedStateFromProps()
- shouldComponentUpdate()
- render()
- getSnapshotBeforeUpdate()
- componentDidUpdate()

adapt

## shouldComponentUpdate()

shouldComponentUpdate(nextProps, nextState)

Used for:

- Manually control component render circle.
- Improving performance.

Don’t forget:

- It should be used only when it really needed.
- Use PureComponent instead.
- Don’t do deep checks, it hurts performance.
- Don’t “prevent” rendering - it could lead to bugs.

## getSnapshotBeforeUpdate()

getSnapshotBeforeUpdate(prevProps, prevState)

Invoked right before the most recently rendered output is committed to e.g. the DOM.

Used for:

- Capturing information from DOM.
- Passing values to componentDidUpdate().

Don’t forget:

- It’s rarely used and in very specific UI cases.

## componentDidUpdate()

componentDidUpdate(prevProps, prevState, snapshot)

Used for:

- Changing state or calling external services if changes requires that.

Don’t forget:

- Do conditioning to prevent endless loop.
- It is not called when shouldComponentUpdate() returns false.

## Unmounting

This method is called when a component is being removed from the DOM:

- componentWillUnmount()

## componentWillUnmount()

componentWillUnmount()

Used for:

- Unsubscribing to events.
- Doing cleanups.
- Canceling timers.
- Canceling network requests.

Not used for:

- Manipulating component in any way.

## Error Handling

These methods are called when there is an error during rendering, in a lifecycle method, or in the constructor of any child component.

- static getDerivedStateFromError()
- componentDidCatch()

## componentDidCatch()

componentDidCatch(error, info)

- error - The error that was thrown.
- info - An object with a componentStack key containing information about which component threw the error.

Used for:

- Logging errors.
- Changing state to render fallback component.

## Legacy Lifecycle Methods

UNSAFE_componentWillMount() - constructor().

UNSAFE_componentWillReceiveProps() -  componentDidUpdate().

UNSAFE_componentWillUpdate() - componentDidUpdate().

## Hooks vs States? It depends

There is no big reason to refactor existing code to hooks because besides consistency there is no real value

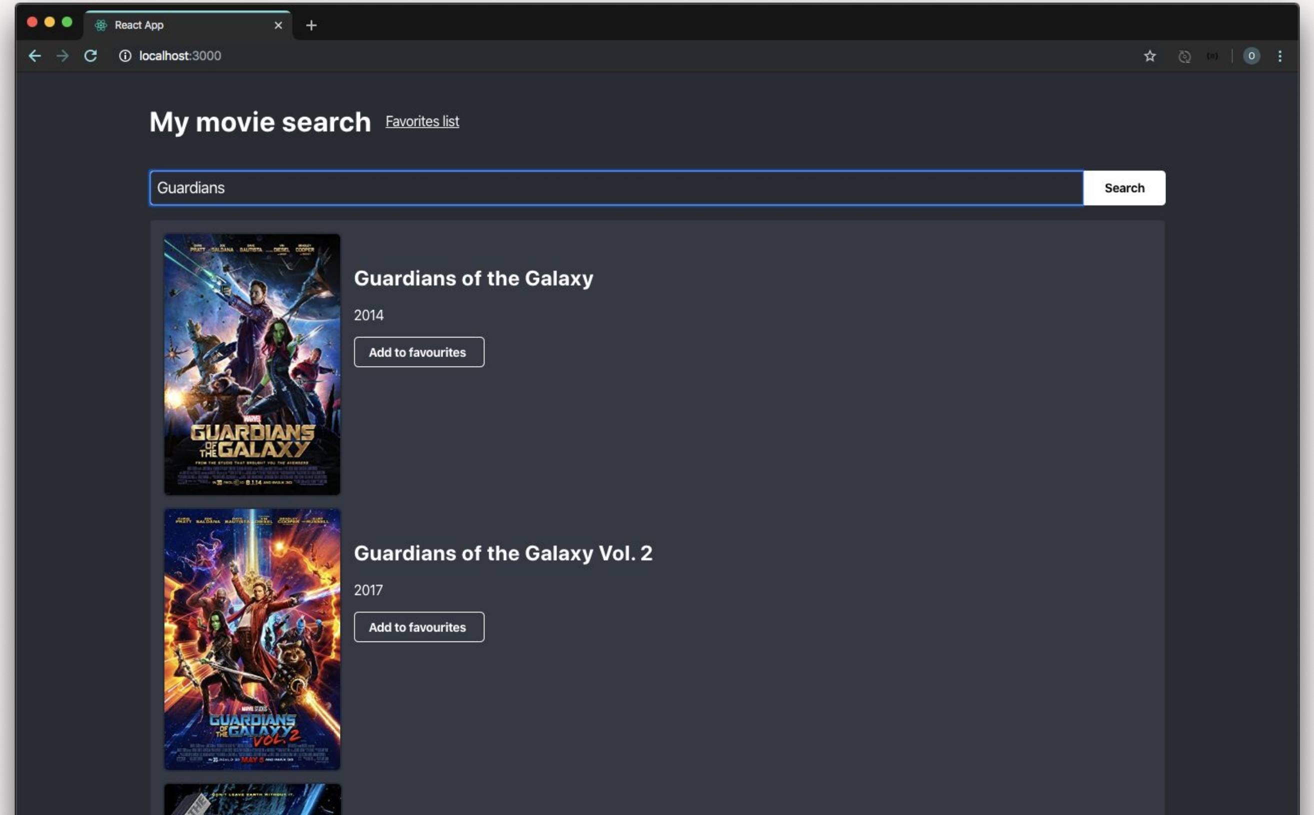
For new projects I would recommend to use hooks

adapt

## Let’s make something out of it

The principles are hard to understand if we don’t start to build something with it.

# Let’s make something out of it



My movie search Favorites list

Guardians Search

Guardians of the Galaxy

2014

Add to favourites

Guardians of the Galaxy Vol. 2

2017

Add to favourites

adapt