React Fundamentals

Lifecycle hooks

Peter Kassenaar –
info@kassenaar.com

Lifecycle hooks

Executing code at specific points in time of the component lifecycle

What are lifecycle hooks?

“Every Component follows a cycle from when it’s created and mounted on the DOM to when it is unmounted and destroyed. This is what we refer to as the ***Component lifecycle****.”*

https://blog.pusher.com/beginners-guide-react-component-lifecycle/

Available lifecycle hooks

- Three stages in the lifecycle
  1. Creating and Mounting
  2. Updating
  3. Unmounting

- *We now cover the lifecyle hooks of class based components*
  - Function based components have a generic `useEffect()` hook

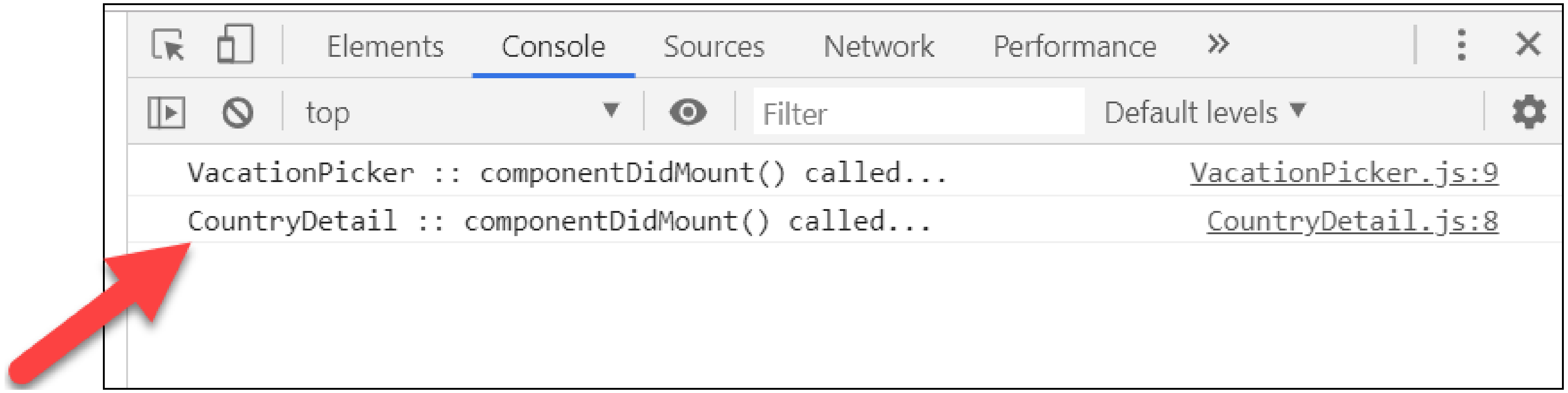
1. Mounting lifecycle hooks

- `constructor()`
  - The default JavaScript constructor of the function or class
- `render()`
  - Your render function
- `componentDidMount()` – used for:
  - Make API calls
  - Set up timers
  - Add event listeners…
- OLD: `componentWillMount()` – don't use anymore!

componentDidMount()

```
// Mounting lifecycle hook
componentDidMount() {
    console.log('VacationPicker :: componentDidMount() called...');
}
```

```
componentDidMount() {
    console.log('CountryDetail :: componentDidMount() called...');
}
```

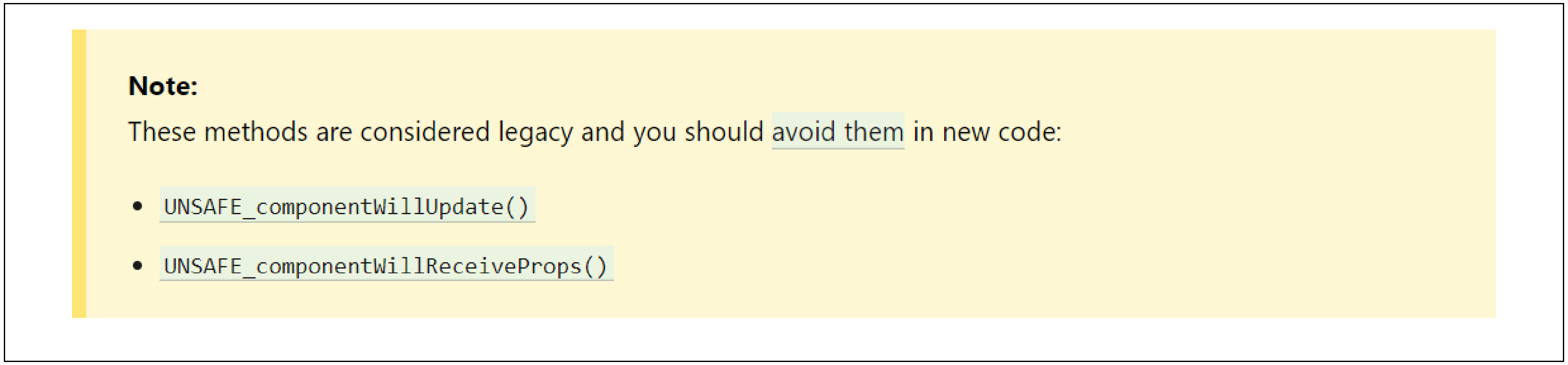


```
VacationPicker :: componentDidMount() called...          VacationPicker.js:9
CountryDetail :: componentDidMount() called...            CountryDetail.js:8
```

2. Updating lifecycle hooks

- `shouldComponentUpdate()`
  - Called before the component re-renders after receiving new props or there's a new state.
  - Two parameters: `nextProps` and `nextState`
  - Not used very often – may harm performance
- `componentDidUpdate()`
  - Called after any rendered HTML has finished loading
  - Two parameters: `prevProps` and `prevState`
  - Used very often

- Deprecated – don’t use anymore
  - `componentWillReceiveProps()`
  - `componentWillUpdate()`
- https://reactjs.org/docs/react-component.html


Note:

These methods are considered legacy and you should avoid them in new code:

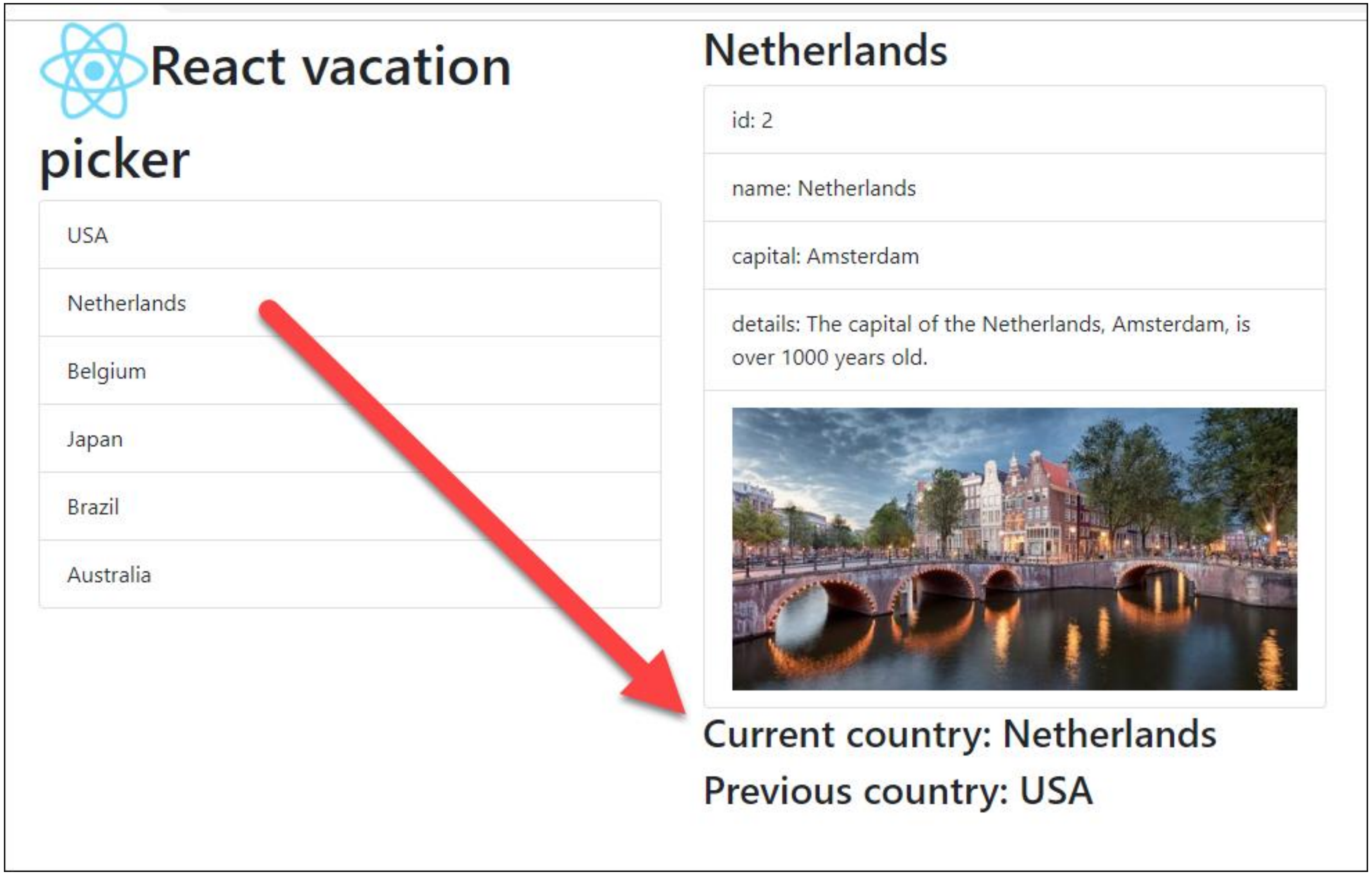
- `UNSAFE_componentWillUpdate()`
- `UNSAFE_componentWillReceiveProps()`

Result

```
state={
    currentCountry:{},
    prevCountry:{}
};
```

```
// 2. Update lifecycle hook
componentDidUpdate(prevProps, prevState, snapshot) {
    console.log('CountryDetail updated!');
    console.log({prevProps}, {prevState});
    // store previous and current country in state
    if(prevProps.country.name !== this.props.country.name) {
        this.setState({
            currentCountry: this.props.country,
            prevCountry: prevProps.country
        })
    }
}
```

Result



React vacation picker

USA

Netherlands

Belgium

Japan

Brazil

Australia

Netherlands

id: 2

name: Netherlands

capital: Amsterdam

details: The capital of the Netherlands, Amsterdam, is over 1000 years old.

Current country: Netherlands

Previous country: USA

3. Unmounting lifecycle hooks

Only one hook available: `componentWillUnmount()`

"`componentWillUnmount()` is called right before a component is removed from the DOM"

Result

CountryDetail.js

```
// 3. Unmount lifecycle hook
componentWillUnmount() {
    console.log('CountryDetail :: Component unmounted...');
    // flush caches, close db-connections, update localStorage, etc.
}
```

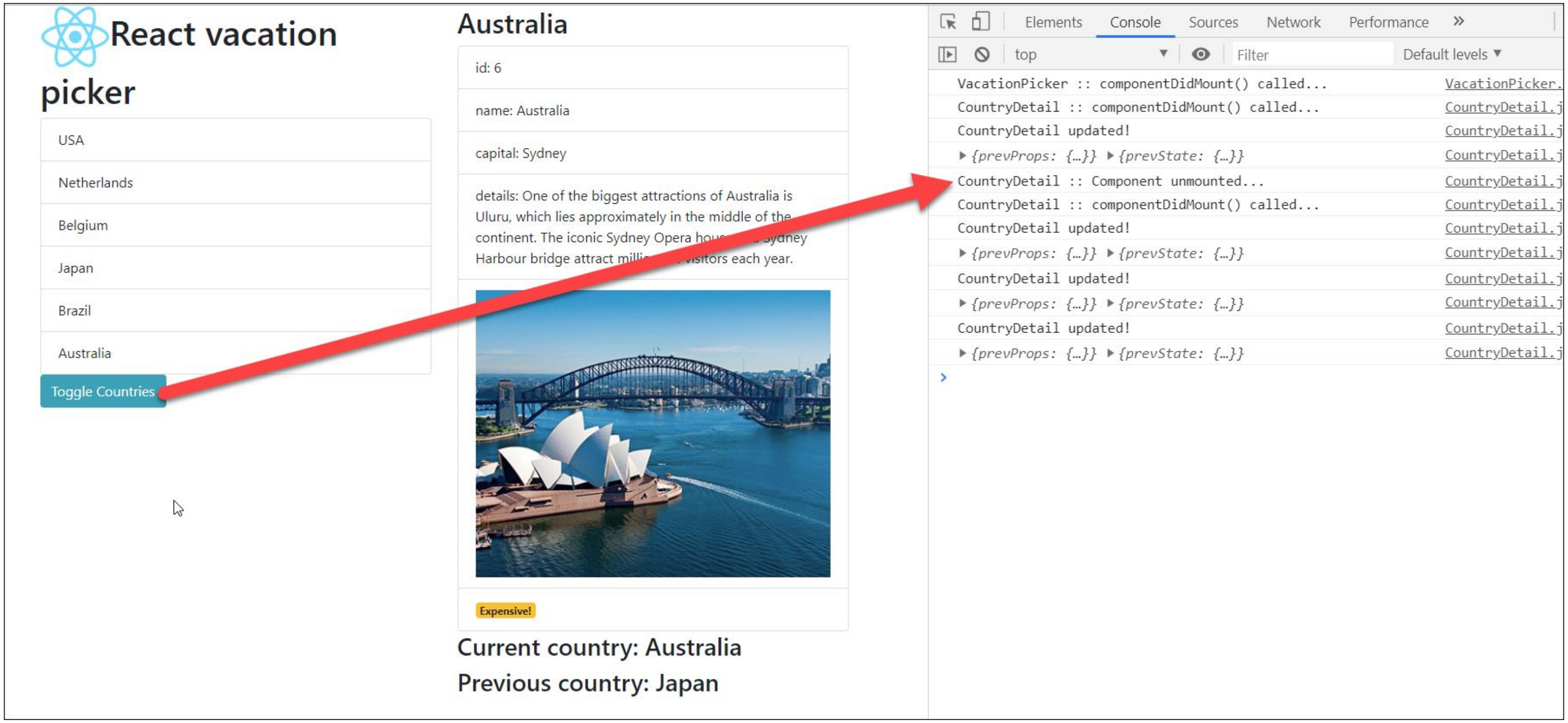
```
// Show or hide country details in the UI
toggleCountries() {
    this.setState({
        showCountries: !this.state.showCountries
    })
}
```

App.js

App.js

```
{
    // conditional rendering
    this.state.showCountries &&
    <CountryDetail country={this.state.currentCountry}/>
}
```

…/240-lifecycle-hooks



Australia

id: 6

name: Australia

capital: Sydney

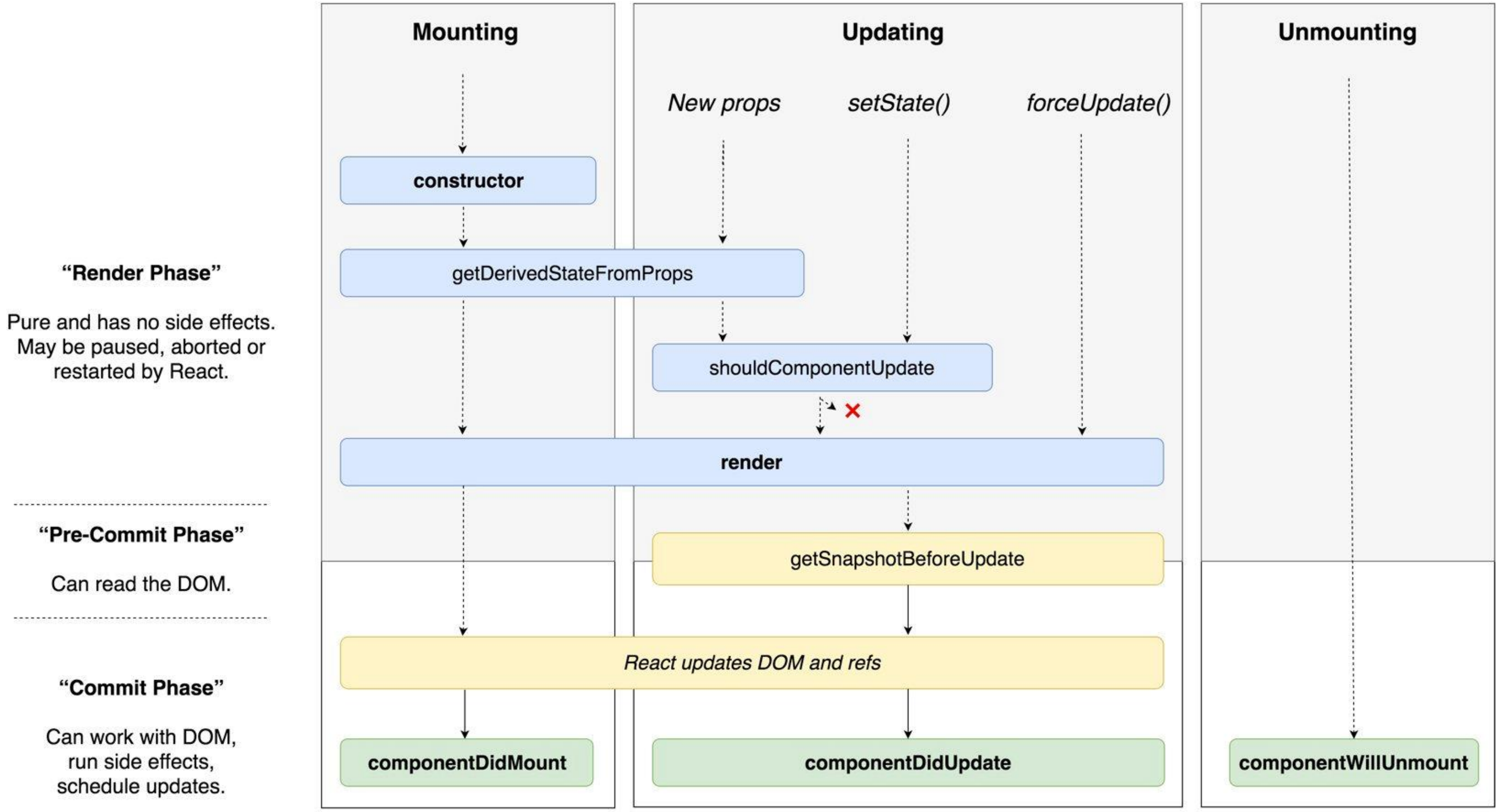
details: One of the biggest attractions of Australia is Uluru, which lies approximately in the middle of the continent. The iconic Sydney Opera house and Sydney Harbour bridge attract millions of visitors each year.

Current country: Australia

Previous country: Japan

```
VacationPicker :: componentDidMount() called...        VacationPicker.
CountryDetail :: componentDidMount() called...         CountryDetail.j
CountryDetail updated!                                 CountryDetail.j
▸ {prevProps: {…}} ▸ {prevState: {…}}                  CountryDetail.j
CountryDetail :: Component unmounted...                CountryDetail.j
CountryDetail :: componentDidMount() called...         CountryDetail.j
CountryDetail updated!                                 CountryDetail.j
▸ {prevProps: {…}} ▸ {prevState: {…}}                  CountryDetail.j
CountryDetail updated!                                 CountryDetail.j
▸ {prevProps: {…}} ▸ {prevState: {…}}                  CountryDetail.j
CountryDetail updated!                                 CountryDetail.j
▸ {prevProps: {…}} ▸ {prevState: {…}}                  CountryDetail.j
>
```

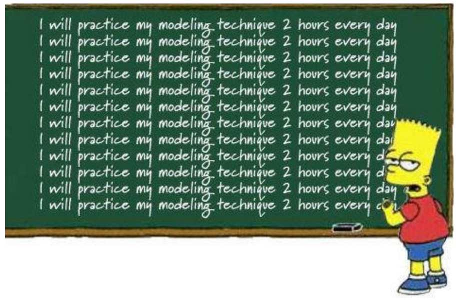
Summary - visual overview



https://twitter.com/dan_abramov/status/981712092611989509

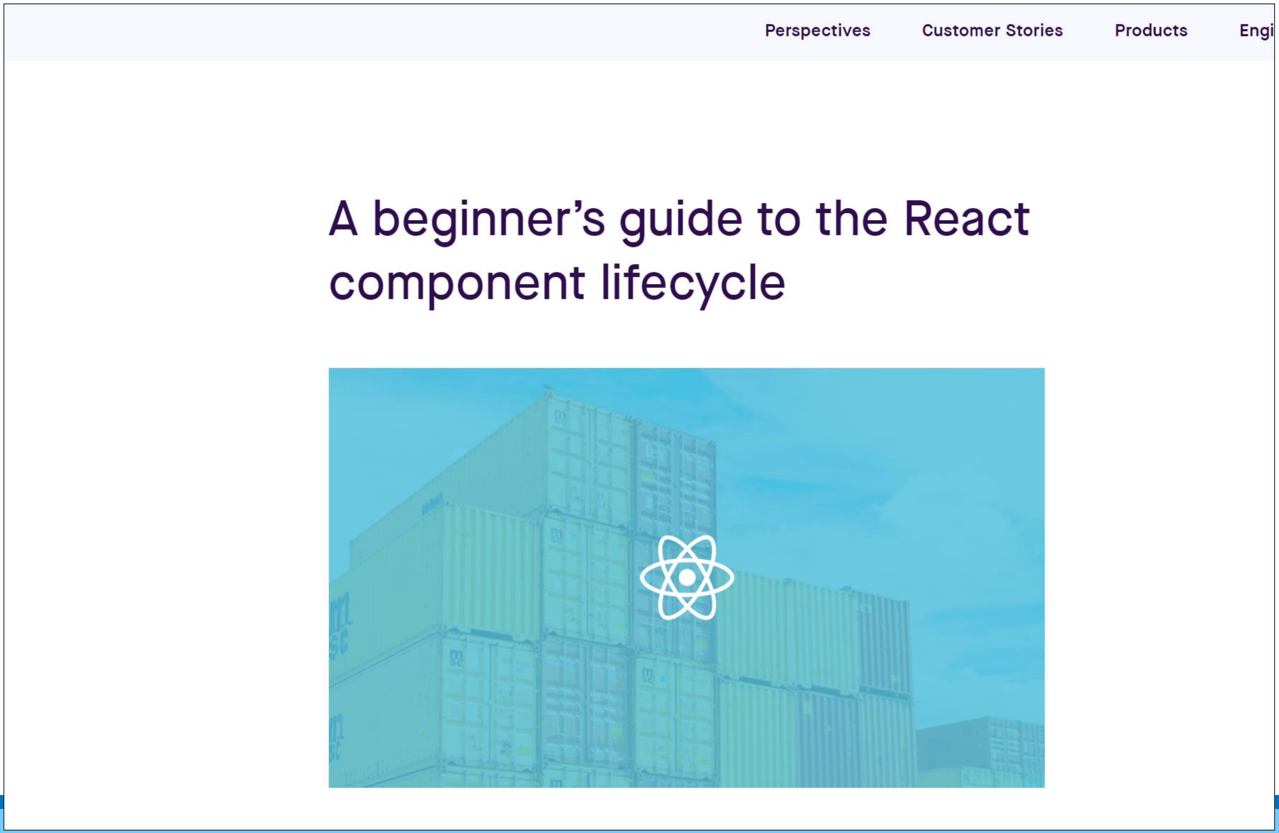
Workshop

- Create a component with a `button` and a `counter`
- In a lifecycle hook, log how many times `counter` is updated.
- After initial rendering it should read '`you updated 0 times`'
- After that, '`you updated 1 time, 2 times, 3 times`', etc.
- Embed the counter in another component where you can show/hide it.
- Make sure the lifecycle hooks still work as expected
- Example `../240-lifecycle-hooks`


I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day
I will practice my modeling technique 2 hours every day

More info on lifecycle

- https://blog.pusher.com/beginners-guide-react-component-lifecycle/



A beginner’s guide to the React component lifecycle

Checkpoint

- You know about component lifecycle hooks in class based components
- There are three important hooks:
  - `componentDidMount()`
  - `componentDidUpdate()`
  - `componentWillUnmount()`
- Previous versions of React had additional hooks that are now deprecated
- Lifecycle hooks can only be used in class based components