

METAPHYSICS AND JAVASCRIPT

@rich_harris • Aug 2019

SVELTE IS BETTER THAN REACT

go on, tweet this slide without context

i live for the drama

SVELTE IS OBJECTIVELY BETTER THAN REACT



Svelte, as a compiler, results in apps that are much smaller and faster to start than traditional runtime frameworks

“Computer, Build Me an App”
JSConf EU, June 2018, Berlin

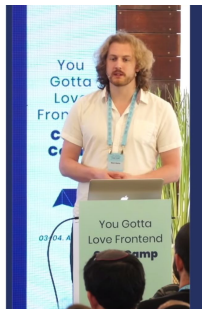
SVELTE IS OBJECTIVELY BETTER THAN REACT

Does it matter? (Yes.)

stone



YGLF

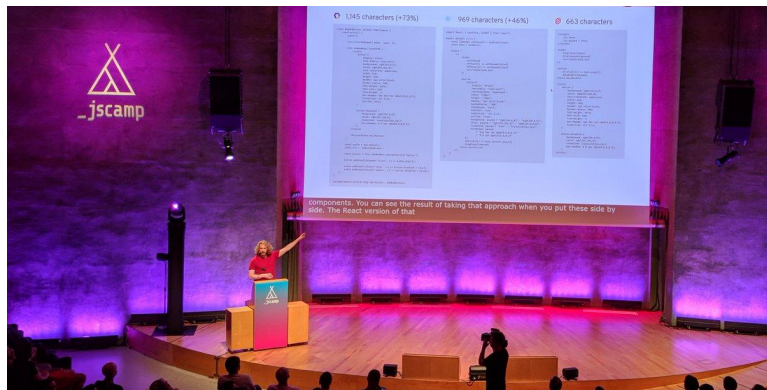


You Gotta Love Frontend
Code Camp 2019

By eliminating the virtual DOM, apps become many times faster – especially critical on low-powered devices

“Rethinking Reactivity”
YGLF, April 2019, Israel

SVELTE IS OBJECTIVELY BETTER THAN REACT



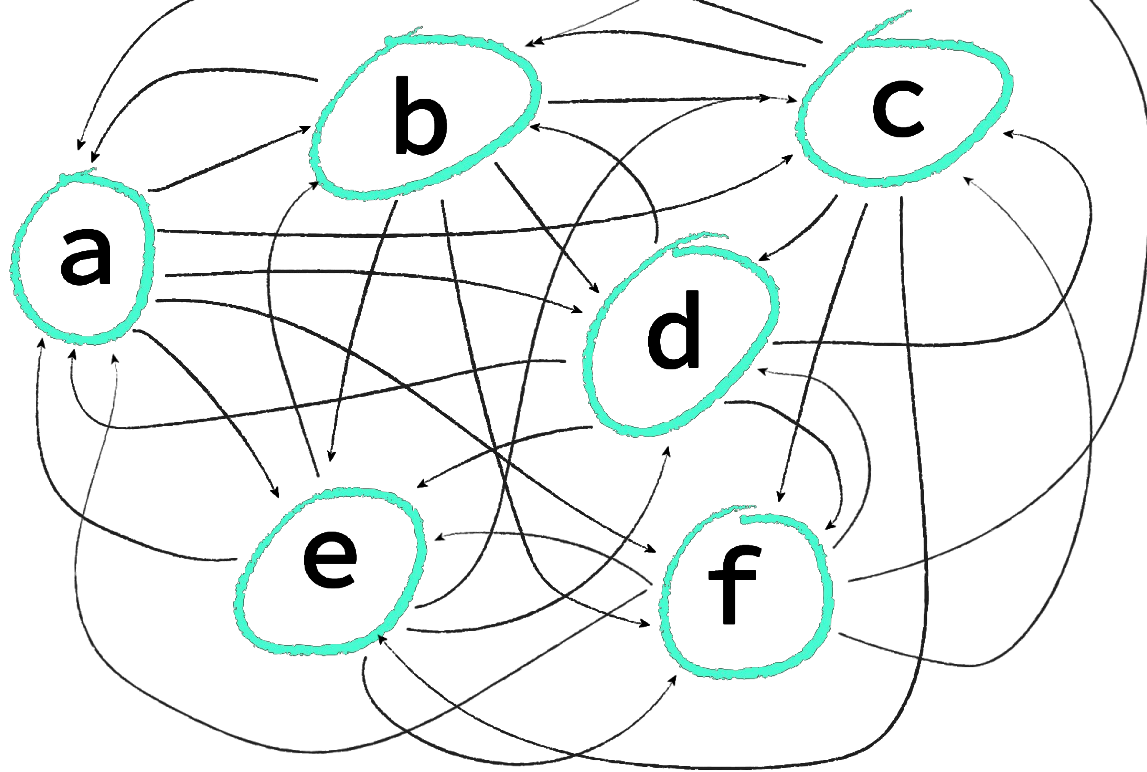
Designing a component language from first principles allows us to write leaner, clearer, more robust code

“The Return of ‘Write Less, Do More’”
JSCamp, July 2019, Barcelona

SVELTE IS... SUBJECTIVELY BETTER THAN REACT?

$$UI = F(STATE)$$

STATE-DRIVEN UI IS EASIER TO REASON ABOUT





PURE UI

“With this model in place, the programmer is thus relieved from the burden of specifying the transition between states (or transformation) of the UI over time. No need to specify how to go from A to B: **just describe what A looks like and what B looks like, in a discrete way.**”

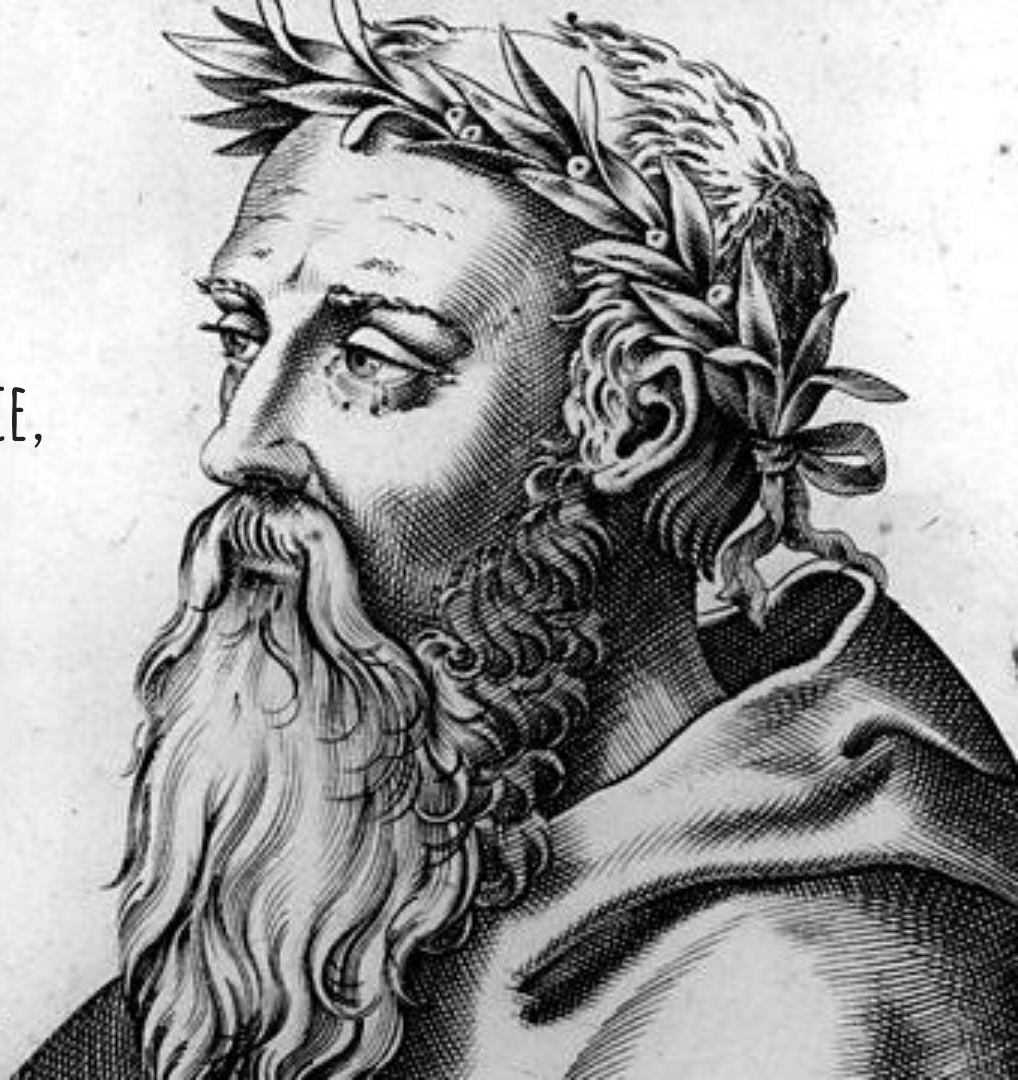
-Guillermo Rauch, rauchg.com/2015/pure-ui

FUNCTIONAL UI !== STATE-DRIVEN UI

PEOPLE ARE HORNY FOR FUNCTIONS.

NO MAN STEPS IN THE SAME RIVER TWICE,
FOR IT IS NOT THE SAME RIVER
AND HE IS NOT THE SAME MAN

—Heraclitus



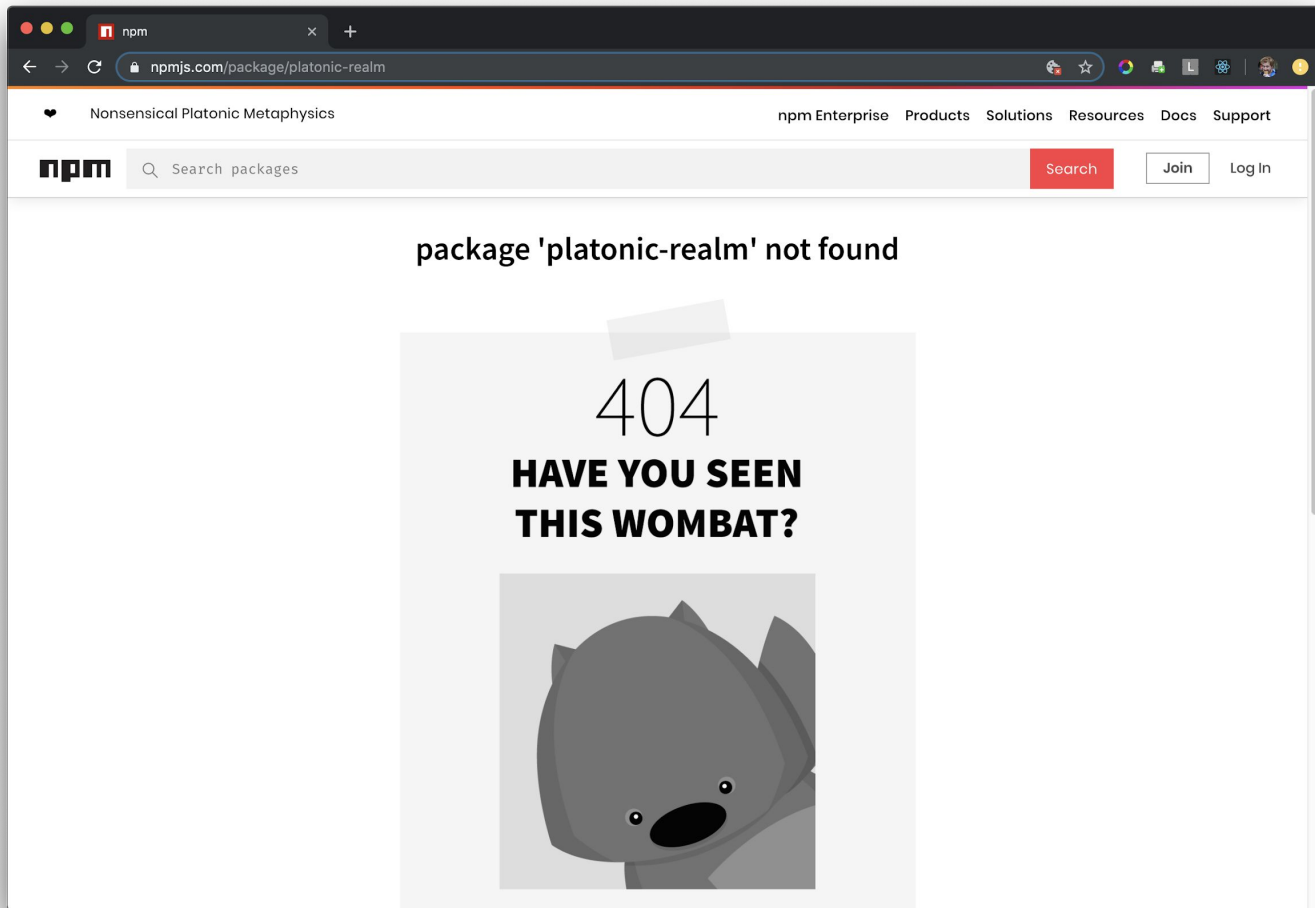


THE THEORY OF FORMS

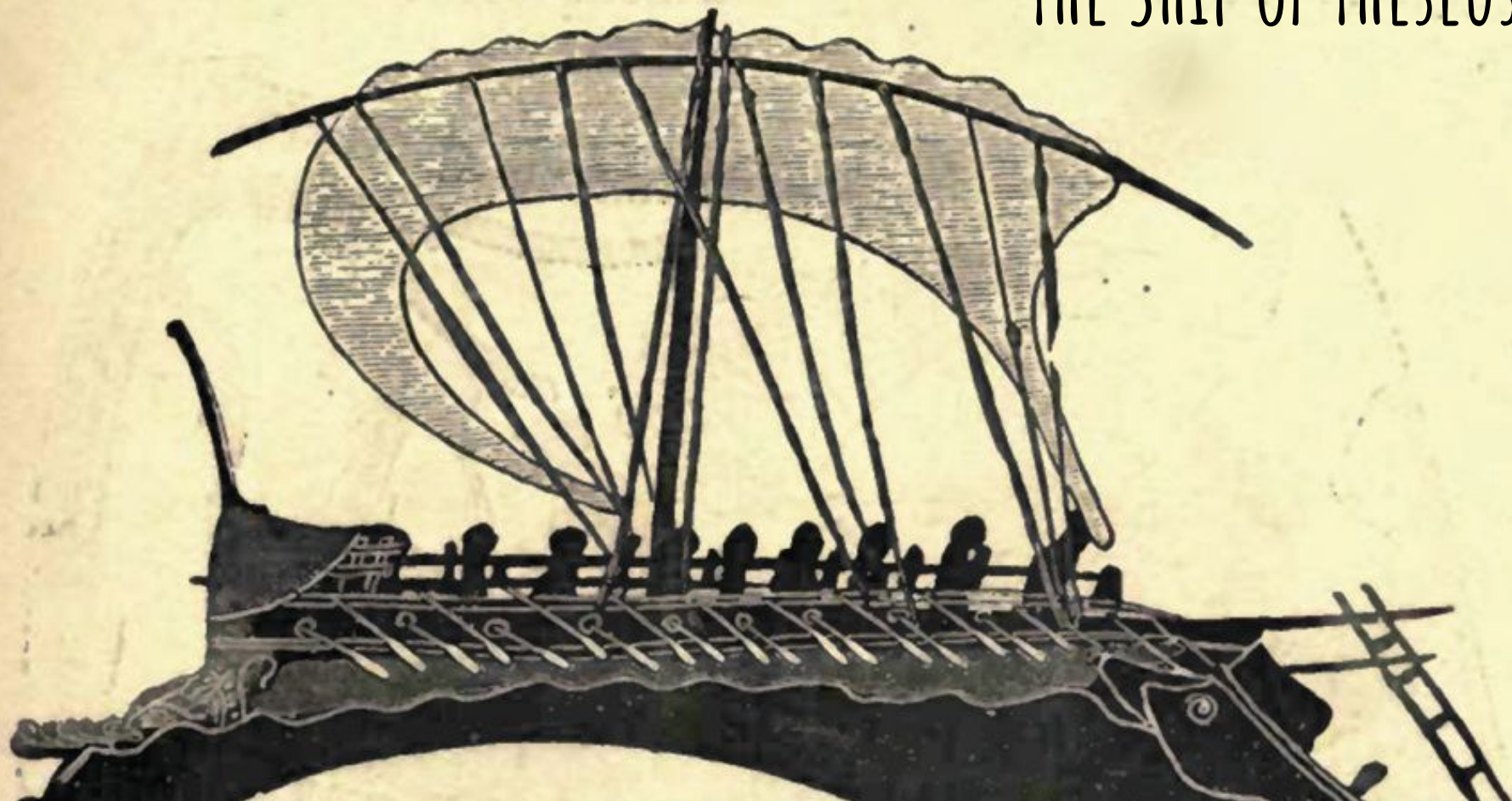
THE ALLEGORY OF THE CAVE



```
<form></form>
```

THE SHIP OF THESEUS



$$UI = F(STATE)$$

```
ui1 = <App state={state1}/>
```

```
ui2 = <App state={state2}/>
```

```
ui1 !== ui2
```



formulanimations :: happy jumping

4,911 views

405

1

SHARE

SAVE

...

**Inigo Quilez**

Published on 24 Jul 2019

SUBSCRIBE 9.8K

A mathematical animation. There are no meshes, 3D poly models, no skeletons, no rigs, no textures, no program or software package, no renderer and no global illumination. Instead, mathematical expressions define the shapes, placement, color (and light), shadows, movement, camera lens simulation, etc. It is a very big mathematical expression, basically.

The source code and realtime version are here: <https://www.shadertoy.com/view/3lsSzf>

Category

[Science & Technology](#)

SHOW LESS

Shader Inputs

```

115   vec3 sq = vec3( abs(r.x), r.yz );
116
117   // head
118   vec3 h = r;
119   float hr = sin(0.791*atime);
120   hr = 0.7*sign(hr)*smoothstep(0.5,0.7,abs(hr));
121   h.xz = mat2x2(cos(hr),sin(hr),-sin(hr),cos(hr))*h.xz;
122   vec3 hq = vec3( abs(h.x), h.yz );
123   float d = sdEllipsoid( h-vec3(0.0,0.20,0.02), vec3(0.08,0.2,0.15) );
124   float d2 = sdEllipsoid( h-vec3(0.0,0.21,-0.1), vec3(0.20,0.2,0.20) );
125   d = smin( d, d2, 0.1 );
126   res.x = smin( res.x, d, 0.1 );
127
128   // belly wrinkles
129   {
130     float yy = r.y-0.02-2.5*r.x*r.x;
131     res.x += 0.001*sin(yy*120.0)*(1.0-smoothstep(0.0,0.1,abs(yy)));
132   }
133
134   // arms
135   {
136     vec2 arms = sdStick( sq, vec3(0.18-0.06*hr*sign(r.x),0.2,-0.05), vec3(0.3+0.1*p2,-0.2+0.3*p2,-0.15), 0.03, 0.06 );
137     res.xz = smin( res.xz, arms, 0.01+0.04*(1.0-arms.y)*(1.0-arms.y)*(1.0-arms.y) );
138   }
139
140   // ears
141   {
142     float t3 = fract(atime+0.9);
143     float p3 = 4.0*t3*(1.0-t3);
144     vec2 ear = sdStick( hq, vec3(0.15,0.32,-0.05), vec3(0.2+0.05*p3,0.2+0.2*p3,-0.07), 0.01, 0.04 );
145     res.xz = smin( res.xz, ear, 0.01 );
146   }
147
148   // mouth
149   {
150     d = sdEllipsoid( h-vec3(0.0,0.15+4.0*hq.x*hq.x,0.15), vec3(0.1,0.04,0.2) );
151     res.w = 0.3+0.7*clamp( d*150.0,0.0,1.0);
152     res.x = smax( res.x, -d, 0.03 );
153   }
154
155   // legs
156   {
157     float t6 = cos(6.2831*(atime*0.5+0.25));
158     float ccc = cos(1.57*t6*sign(r.x));
159     float sss = sin(1.57*t6*sign(r.x));
160     vec3 base = vec3(0.12,-0.07,-0.1); base.y -= 0.1/sy;
161     vec2 legs = sdStick( sq, base, base + vec3(0.2,-ccc,sss)*0.2, 0.04, 0.07 );
162     res.xz = smin( res.xz, legs, 0.07 );
163   }
164
165   // eye
166

```

SO WHY WOULDN'T WE EMBRACE THIS IDEOLOGY?

“An ideology is a collection of normative beliefs and values that an individual or group holds for other than purely epistemic reasons.”

—Wikipedia

ideological statement



$$UI = F(STATE)$$

```
const render = state => {  
  document.body.innerHTML = `  
    <h1>Hello ${state.name}!</h1>  
    <input value="${state.name}">  
  `;  
  
  const input = document.querySelector('input');  
  
  input.oninput = () => {  
    render({  
      name: input.value  
    });  
  };  
};  
  
render({ name: 'world' });
```

Hello world!

 ⓘ

```
const render = state => {  
  document.body.innerHTML = `  
    <h1>Hello ${state.name}!</h1>  
    <input value="${state.name}">  
  `;  
  
  const input = document.querySelector('input');  
  
  input.oninput = () => {  
    render({  
      name: input.value,  
      focus: document.activeElement === input  
    });  
  };  
  
  if (state.focus) input.focus();  
};  
  
render({ name: 'world' });
```

Hello world!

 ⓘ


```
const render = state => {
  document.body.innerHTML = `
    <h1>Hello ${state.name}!</h1>
    <input value="${state.name}">
  `;

  const input = document.querySelector('input');

  input.oninput = () => {
    render({
      name: input.value,
      focus: document.activeElement === input,
      selectionStart: input.selectionStart,
      selectionEnd: input.selectionEnd,
      selectionDirection: input.selectionDirection
    });
  };

  if (state.focus) input.focus();

  input.setSelectionRange(
    state.selectionStart || 0,
    state.selectionEnd || 0,
    state.selectionDirection || 'none'
);
};

render({ name: 'world' });
```

Hello world!

world ⓘ

```
const render = state => {
  document.body.innerHTML = `
    <style>
      h1 {
        animation: fade-in 1s;
      }

      @keyframes fade-in {
        from { opacity: 0 }
        to { opacity: 1 }
      }
    </style>

    <h1>Hello ${state.name}!</h1>
    <input value="${state.name}">
  `;

  const input = document.querySelector('input');

  input.oninput = () => {
    render({
      name: input.value,
      focus: document.activeElement === input,
      selectionStart: input.selectionStart,
      selectionEnd: input.selectionEnd,
      selectionDirection: input.selectionDirection
    });
  };

  if (state.focus) input.focus();
}
```

```
const App = () => {  
  const [name, setName] = useState('world');  
  
  return (  
    <>  
      <h1>Hello {name}!</h1>  
      <input onChange={e => setName(e.target.value)}>  
    </>  
  );  
};  
  
render(<App/>, document.body);
```



Sebastian Markbåge

@sebmarkbage



Render should be pure. It should have no external side-effects. The biggest challenge for React going forward is that we really meant it. "Yes, but..." As an ecosystem we'll have to find and teach patterns to replace the ones that break this rule. This is going to be frustrating.

10:06 PM · Aug 13, 2019 · [Twitter Web App](#)

43 Retweets **280** Likes

```
class Clock extends React.Component {
  state = {
    count: 0
  }

  componentDidMount() {
    this.interval = setInterval(() => {
      this.setState({
        count: this.state.count + 1
      });
    }, 1000);
  }

  componentDidUpdate() {
    console.log(this.state.count);
  }

  componentWillUnmount() {
    clearInterval(this.interval);
  }

  render() {
    return (
      <p>Count: {this.state.count}</p>
    );
  }
}
```



Ryan Florence
@ryanflorence



Replying to [@ryanflorence](#) [@dan_abramov](#) and 2 others

The question is not "when does this effect run" the question is "with which state does this effect synchronize with"

`useEffect(fn) // all state`

`useEffect(fn, []) // no state`

`useEffect(fn, [these, states])`

10:14 AM · May 5, 2019 · [Twitter Web App](#)

266 Retweets **1.1K** Likes

```
function Clock() {
  const [count, setCount] = useState(0);

  // componentDidMount / componentWillUnmount
  useEffect(() => {
    const interval = setInterval(() => {
      setCount(count + 1);
    }, 1000);

    return () => {
      clearInterval(interval);
    };
  }, []);

  // componentDidUpdate
  useEffect(() => {
    console.log(count);
  }, [count]);

  return (
    <p>Count: {count}</p>
  );
}
```



Kent C. Dodds @kentcdodds · Jun 15



Help wanted:

What are some pitfalls you've had using/adopting React Hooks?

(Please favorite replies of people who mention one you've experienced rather than tweeting the same ones at me).



130



57



306



Dave

@davecporter



Replying to [@kentcdodds](#)

Stale closures stale closures stale closures stale closures


```

7 function App() {
8   const [count, setCount] = useState(0);
9
10  useEffect(() => {
11    const timeout = setTimeout(() => {
12      setCount(count + 1);
13    }, 1000);
14
15    return () => {
16      clearTimeout(timeout);
17    };
18  }, [count]);
19
20  return (
21    <p>Count: {count}</p>
22  );
23 }

```

```

7 function App() {
8   const [count, setCount] = useLocalStorage('count', 0);
9
10  useEffect(() => {
11    const timeout = setTimeout(() => {
12      const [count, setCount] = useLocalStorage('count', 0);
13      const count: any
14
15      React Hook useEffect has a missing dependency: 'setCount'.
16      Either include it or remove the dependency array. (react-
17      hooks/exhaustive-deps) eslint
18      Quick Fix... Peek Problem
19    }, [count]);
20
21    return (
22      <p>Count: {count}</p>
23    );
24  });
25 }

```

In the words of Ryan Florence:

I've had a lot of people point to setInterval with hooks as some sort of egg on React's face

Honestly, I think these people have a point. It is confusing at first.

But I've also come to see it not as a flaw of Hooks but as a mismatch between the React programming model and `setInterval`. Hooks, being closer to the React programming model than classes, make that mismatch more prominent.

Disclaimer: this post focuses on a *pathological case*. Even if an API simplifies a hundred use cases, the discussion will always focus on the one that got harder.

—Dan Abramov, overreacted.io



Yehuda Katz  

@wycats



Honestly, React became popular because it was fun to use and productive.

Making React less fun and feel less productive in the name of "purity" is a weird direction to take.



Sebastian Markbåge @sebmarkbage · Aug 13

Render should be pure. It should have no external side-effects. The biggest challenge for React going forward is that we really meant it. "Yes, but..." As an ecosystem we'll have to find and teach patterns to replace the ones that break this rule. This is going to be frustrating.

[Show this thread](#)

10:29 PM · Aug 13, 2019 · [Twitter for Android](#)

17 Retweets 115 Likes



Jason Miller 🦊 ⚙️
@_developit




I keep writing this useMount() hook.

It's easier to scan for in code than useEffect.

Basically my brain can't remember this:

```
useEffect(fn) // run after each render
```

```
useEffect(fn, []) // runs after mount
```



```
function useMount(fn) {  
  useEffect(fn, []);  
}
```

10:03 PM · May 4, 2019 · [Twitter Web App](#)

116 Retweets 759 Likes



Michel Weststrate @mweststrate · Feb 4

Ok, I think I need a `useFinally` [@reactjs](#) hook that get's called when component is either unmounted, or discarded without ever being committed. I'm still in time for such a feature, right 😊



3



12



Michel Weststrate @mweststrate · Feb 4

Q: Wait, what, so your render has side effects?

A: Yes 🤔👉



2



3



Sebastian Markbåge

@sebmarkbage

Replying to [@mweststrate](#) and [@reactjs](#)

This is by far the most requested feature for concurrent mode but also by far the least likely that we can add.

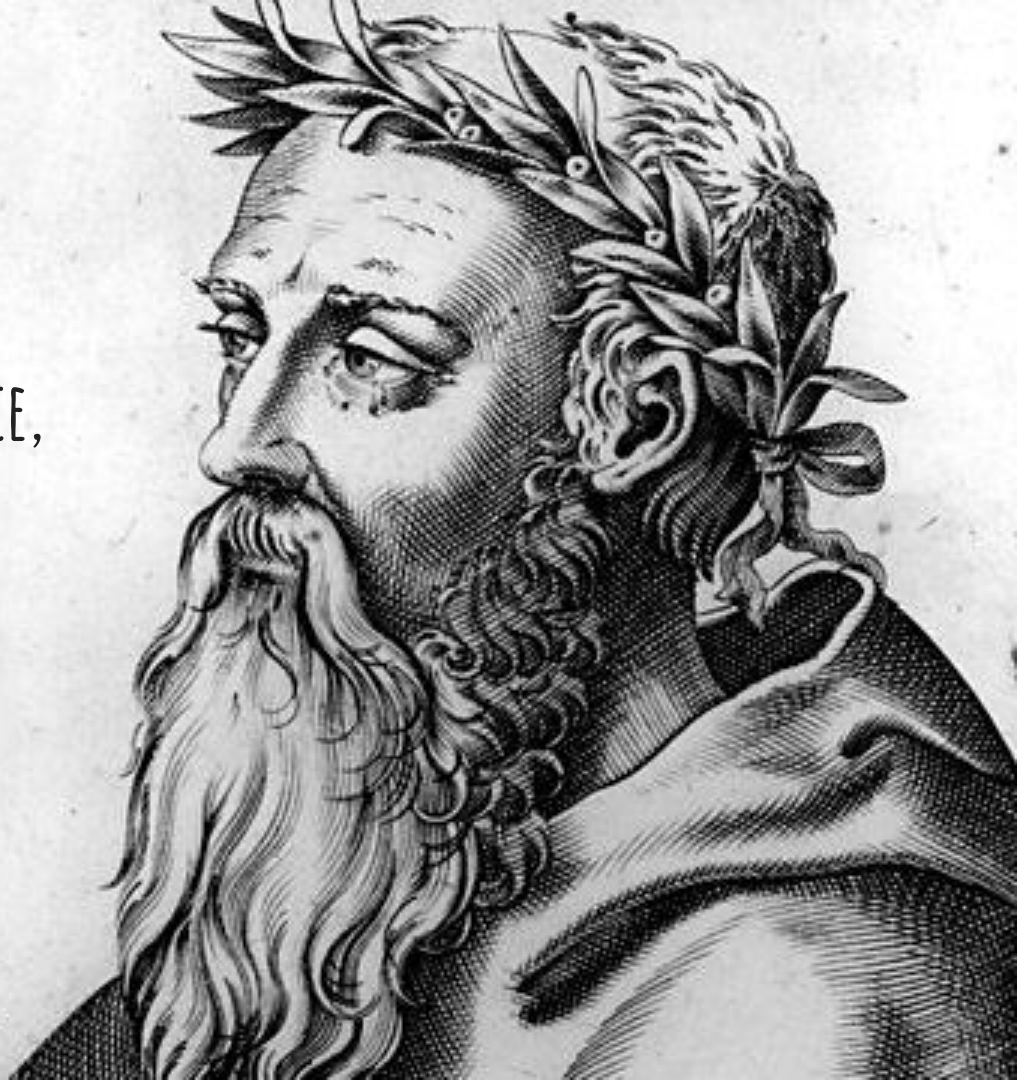
- **React expert**  @react_expert · Aug 16
Check out this cool thing I did with hooks!
 11  4  36  
- **React core team member** @react_core · Aug 16
That will break for the following incredibly subtle reasons:
 2   
- **React expert**  @react_expert · Aug 16
Ah, you're right. Honestly, this was so much more intuitive with classes.
 2    
- **React core team member**
@react_core

Replying to [@react_expert](#)

Don't worry, it'll make sense! You just need to completely reconfigure your brain

NO MAN STEPS IN THE SAME RIVER TWICE,
FOR **IT** IS NOT THE SAME RIVER
AND **HE** IS NOT THE SAME MAN

—Heraclitus



DEMO TIME!!!

THANKS!

- <https://svelte.dev>
- <https://twitter.com/sveltejs>
- https://twitter.com/rich_harris