

How to actually use those compo lifecycle methods

Once initially

mount

componentDidMount

componentDidUpdate

componentWillUnmount

functional

Class Component methods

useEffect Hook

every time when state will update

we can use all lifecycle methods of class components inside functional comp

once

similar to componentDidMount

```
useEffect(() => {  
  // put 'run once' code here  
}, [])
```

pass empty array

on props change

similar to componentDidUpdate

```
function YourComponent({ someProp }) {  
  useEffect(() => {  
    // code to run when someProp changes  
  }, [someProp]);  
}
```

include all monitored props

after every render

similar to componentDidUpdate

```
useEffect(() => {  
  // put 'every update' code here  
})
```

no second argument

on state change

similar to componentDidUpdate

```
function YourComponent() {  
  const [state, setState] = useState(  
    // code to run when state changes  
  ), [state])  
}
```

include all state vars to watch

on unmount

similar to componentWillUnmount

```
useEffect(() => {  
  return () => {  
    // put unmount code here  
  }  
})
```

return the cleanup function

oavesecodia.com

with help of 'useEffect Hook'

what is use of Component will unmount

→ clearing Times, Intervals.

W4 API

setInterval

calls a function
at specific intervals
of time

vs

setTimeout

only
once

calls a funⁿ after
some specified time.

task 1 - print Hello every 20 seconds

Ans → setInterval

task 2 → print hello after 20 seconds
from your app & then

Ans → setTimeout

[] — only once useEffect —

componentDidMount

useEffect(() => {
 // code
}, [])
only once

```
useEffect(() => {  
  setInterval(() => {  
    console.log('useEffect called after components are mounted')  
    axios.get('https://jsonplaceholder.typicode.com/posts')  
      .then(res => {  
        setData(res.data)  
      })  
      .catch(err => {  
        console.log(err)  
      })  
  }, 5000)  
}, [ ])
```

web api
every 5 sec
start executing
the code
that is present
only inside
setInterval
5000

only once

5000