const q = [1, 2, 3, 4, 5]

q.reduce((acc,n)=>acc\*n, 1)

q.reduce((acc,n)=>acc+n, 0)

myReduce(q,(acc,n)=>acc+n,0)

myReduce(q, (acc, n) => acc \* n, 0)

1.......

function myReduce(arr, fn, init) {

let acc = init

arr.forEach((element, idx) => {

acc = fn(acc, element)

})

return acc

}

2...............

q.myReduce((acc, n) => acc + n , 0) //NOt Work

Array.prototype.myReduce = function myReduce( fn, init) {

let acc = init

// arr will be undefined, so need to fix it, you cannot reference `arr` in this case

this.forEach((element, idx) => {

acc = fn(acc, element)

})

return acc

}

10 secs

expected usage case: delay(10).then(() => console.log('10secs later...'))

implement `delay` function so that I can use it like this

delay(10).then(() => console.log('10secs later...'))

use Promise and setTimeout to return promise object

function delay(ns) {

Return new Promise(resolve=>setTimeout(resolve,ns \* 1000))

}

PureComponent vs Component

virtual dom

server side rendering

seo

Redux?