

# React Session 04

## useEffect, Routing

00

# 지난 세션 복습

# useState

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

state

state를 변경 ?

Initial Value

구조 분해 할당

[a, b] = [10, 20];

# useState

```
import React, { useState } from "react";

const StatePractice = () => {
  const [value, setValue] = useState(0);
  const handleClick = () => {
    setValue(1);
  };
  return <div onClick={handleClick}>{value}</div>;
};

export default StatePractice;
```

**State의 값이 변경될 때마다 (setState를 통해)  
화면이 리렌더링 될거야**

# useState 작동 원리 \_ 단순 버전

```
// react 모듈

let _value

export useState(initialValue){

  if(_value === undefined){
    _value = initialValue
  }

  const setState = (newValue) =>{
    _value = newValue
  }

  return [_value, setState]
}
```

setState는  
리렌더링을  
발생시킨다.

setState는 state를 변경 시키는 게 아니었다

# useState 작동 원리 \_ 단순 버전

```
import React, { useState } from "react";

const StatePractice = () => {
  const [value, setValue] = useState(0);
  const handleClick = () => {
    setValue(value + 1);
    console.log(value);
  };
  return <div onClick={handleClick}>{value}</div>;
};

export default StatePractice;
```

1. `_value`에  
0 담겨서  
return

3. `console.log(0)`

5. 화면이 리렌더링 됨으로써  
1이 제대로 그려짐

2. 버튼 클릭,  
`handleClick`  
함수 실행,  
`setValue()`  
1을 담아 호출

4. `_value`에  
1 담겨서 return,  
`setValue`에 의해  
화면 리렌더링  
(비동기)

```
// react 모듈

let _value

export useState(initialValue){

  if(_value === undefined){
    _value = initialValue
  }

  const setState = (newValue) =>{
    _value = newValue
  }

  return [_value, setState]
}
```

# setState는 왜 비동기 ?

## Batch Update

React 는 batch update를 16ms 단위로 진행한다.

16ms 동안 변경된 상태 값들을 모아서 단 한번의 리렌더링을 진행한다.

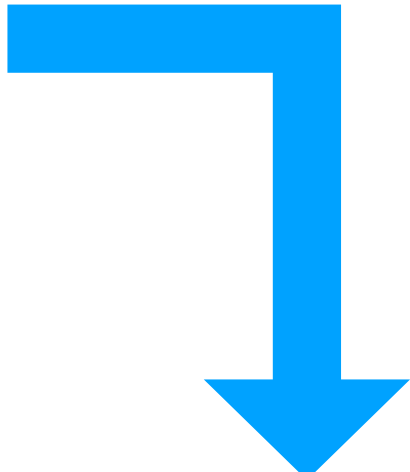
이러한 행동은 웹 페이지 렌더링 횟수를 줄여 좀 더 빠른 속도로 동작하게끔 만든다.

```
import React, { useState } from "react";

const StatePractice = () => {
  const [value, setValue] = useState(0);
  const handleClick = () => {
    setValue(value + 1);
    setValue(value + 1);
    setValue(value + 1);
  };
  return <div onClick={handleClick}>{value}</div>;
};

export default StatePractice;
```

setState의 인자로  
함수를 전달해 해결



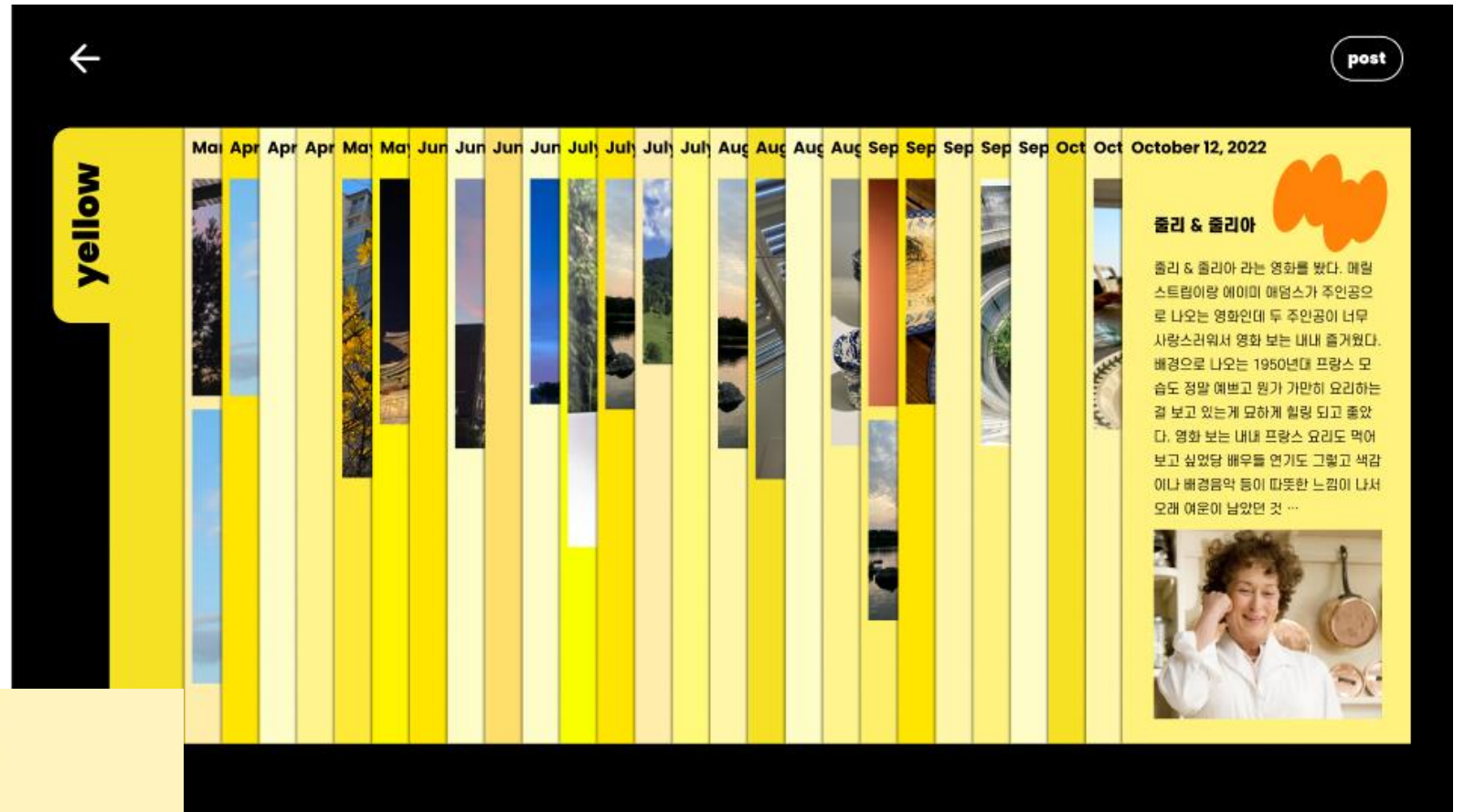
```
import React, { useState } from "react";

const StatePractice = () => {
  const [value, setValue] = useState(0);
  const handleClick = () => {
    setValue((value) => value + 1);
    setValue((value) => value + 1);
    setValue((value) => value + 1);
  };
  return <div onClick={handleClick}>{value}</div>;
};

export default StatePractice;
```



# 01 라우팅이란?



posts.css  
posts.js  
posts.html  
글 목록 페이지



post\_detail.css  
post\_detail.js  
post\_detail.html  
글 상세 페이지

github.com/yunseonyeong

github.com/yunseonyeong

seonyeong Yun  
yunseonyeong  
CAU CSE 18 / LikeLion-CAU

Pinned

LonelyAlgorithm Public  
난 나랑 싸운다 벗어나자 알린이  
Python ☆ 3

1,082 contributions in the last year

Contribution settings

2022  
2021  
2020

github.com

github.com

yunseonyeong

Recent Repositories

Find a repository...

GitchoTantan/Gardener\_Client  
yunseonyeong/ML-assignment  
yunseonyeong/Gardener\_Client  
yunseonyeong/LonelyAlgorithm  
LikeLion-CAU-9th/DoMain  
yunseonyeong/nodeBook  
LikeLion-CAU-9th/Algorithm

Show more

Following For you Beta

euije started following yymin1022 1 hour ago

Yongmin Yoo 유용민 yymin1022  
Military Service Ongoing : ROKAF Korean Solo Developer  
대학생 1인 개발자 defcon.or.kr / dev-lr.com  
104 repositories 73 followers

Follow

shb03323 forked shb03323/Havit-Server from TeamHavit/Havit-Server  
12 hours ago

TeamHavit/Havit-Server  
HAVIT의 귀여움 담당, 효식 정아의 HAVIT SERVER  
JavaScript ☆ 30 Updated Jul 7

Star

Latest changes

12 hours ago  
Code scanning enterprise-level REST API

3 days ago  
Dependabot alerts are now ranked by most important priority

6 days ago  
Zuplo is now a GitHub secret scanning partner

7 days ago  
Sendinblue is now a GitHub secret scanning partner

View changelog →

Explore repositories

# 라우팅이란?

사용자가 요청한 URL에 따라 알맞은 화면을 보여주는 것을 의미

React Router > 사용자가 입력한 주소를 감지하는 역할

상황에 맞는 여러 종류의 라우터 컴포넌트 제공

```
npm install react-router-dom@6
```

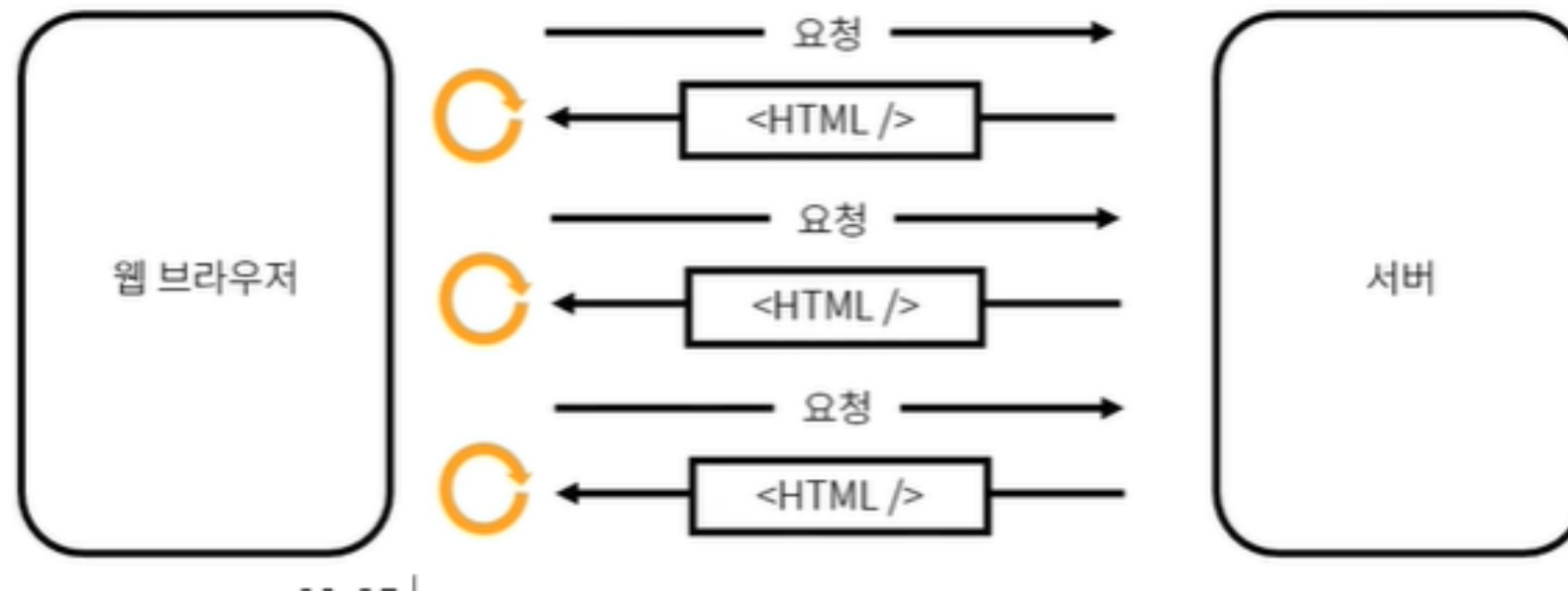
```
yarn add react-router-dom@6
```

MPA  
SPA  
CSR

**집 중 집 중 !!!**

# MPA

## (Multiple Page Application)



여러 개의 페이지로 이루어진 애플리케이션  
매 요청마다 서버에서 완성된 html을  
클라이언트에 응답하는 방식

# SPA

## (Single Page Application)

React, Angular, Vue



**한 개의 페이지**로 이루어진 애플리케이션  
첫 로딩 시 서버로부터 하나의 페이지를 받고,  
그 후로는 데이터만 동적으로 업데이트 하는 방식

# CSR

## (Client Side Rendering)

SPA가 정보를 렌더링하는 방식

화면 깜빡임 없이 페이지 이동이 가능함

1. 사용자가 페이지 요청을 보낸다.
2. 서버가 클라이언트에게 1개의 썩데기 HTML 파일과 JS에 접근할 수 있는 링크를 보내준다.
3. 클라이언트는 화면을 그려 주기 위해 JS 파일을 내려 받는다.
4. 그 이후의 모든 변화에 대해서는 동적으로 브라우저 단에서 대응한다.



# 그래서 ?

리액트가 SPA를 구현하기 위해서는  
동적으로 컴포넌트를 갈아 치워 주는 것

즉, 우리는 url에 따라 서로 다른 “page”를  
로딩 시키는 것이 아니라

url 요청에 따라 보여줄 컴포넌트를 불러  
다른 “화면”을 보여주는 것이다.

## 우리가 주로 하고 싶은 것들

클릭 시 원하는 페이지로 이동

NavBar는 페이지 어디든 있었으면 좋겠어

[github.com](https://github.com) 요청 시, 깃허브 홈을 보여줘

[github.com/yunseonyeong](https://github.com/yunseonyeong) 요청 시, 윤선영 깃허브 보여줘

[github.com/LikeLion-at-CAU-10th](https://github.com/LikeLion-at-CAU-10th) 요청 시, 멋사 10기 깃허브 보여줘

그러나 세 화면 모두 NavBar는 있어야 돼

# 02

## React Router 사용법

# 프로젝트에 라우터 적용

react-router-dom에 내장되어 있는  
BrowserRouter라는 컴포넌트를 사용하여 감싸기

src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import { BrowserRouter } from 'react-router-dom';

ReactDOM.render(
  <BrowserRouter>
    <App />
  </BrowserRouter>,
  document.getElementById('root')
);
```

# Routes / Route

사용자의 브라우저 url에 따라 원하는 컴포넌트를 보여줄래요.

path : 여기로 요청하면 ~  
element : 이걸 보여줄게 ~

```
<Routes>  
  <Route path="/" element={<Home />} />  
  <Route path="/about" element={<About />} />  
  <Route path="/love" element={<Love />} />  
</Routes>
```

Route는 Routes안에서 사용한다.

# Link

클릭 시 다른 페이지로 이동하도록 할래요.

<a> 태그 ?

to : 여기로 가줘

```
import { Link } from "react-router-dom";

const Home = () => {
  return (
    <>
      <div>Home</div>
      <Link to="/about">About</Link>
    </>
  );
};
```

# useNavigate

페이지 이동을 할 수 있게 해주는 함수를 반환  
함수 호출을 통해 페이지 이동을 하기 때문에 조건문을 달아  
특정 조건에 따라 페이지 이동을 하게끔 할 수 있음

useNavigate + useLocation 사용하여  
Navigate와 동시에 Props 전달 가능

```
<Button  
  btnColor="purple"  
  onClick={() => {  
    clickBtn("/");  
  }}  
>  
  Home  
</Button>
```

```
const navigate = useNavigate();  
const clickBtn = (url) => {  
  navigate(url);  
};
```

# 중첩 라우팅

똑같은 UI에 특정 콘텐츠만 갈아 끼우고 싶어요!  
Ex) about/user1, about/user2

## 2가지 방식 사용

1. Outlet
2. path \*

```
<Route path="/about" element={<Outlet />}>  
  <Route path="yunseonyeong" element={<div>윤선영 상세 페이지</div>}/>  
  <Route path="likelion" element={<div>멋사 상세 페이지</div>} />  
</Route>
```



# React 에서의 동적 라우팅

**URL Parameter** : /about/yunseonyeong

특정 데이터를 보여줄 경우.

예1) user 목록 > user 상세 페이지

예2) 블로그 글 목록 > 블로그 글 상세 페이지

**useParams()** Hook 사용

**Query String** : /search?key=react&page=1

데이터를 정렬/필터링 해서 보여줄 경우.

예) 검색 결과 페이지

**useSearchParams()** Hook 사용

## 동적 라우팅 (URL 파라미터)

User 상세 페이지 보여주고 싶은데  
User가 100명이다 ?

그동안은 “정해진 url”에 대해서 라우팅함.

위와 같은 상황에서는 `userId` 등을 매개변수로 지정해  
동적으로 라우팅해야 한다.

```
<Route path="/about" element={<UserNav />}>  
  <Route path=":nickname" element={<About />} />  
</Route>
```

동적 라우팅 + 중첩 라우팅

localhost:3000/about/yunseonyeong

localhost:3000/about/likelion

localhost:3000/about/tiger

여기서 끝 ㄴㄴ About 컴포넌트를 보여주는 할건데  
3개 다 조금씩 다른 상세페이지를 보여줘야겠죠 ?

# 동적 라우팅 실습

1. useState 사용
2. useParams 사용
3. 컴포넌트 조건부 렌더링
4. Styled-component 사용
5. Routes/Route 사용
6. useNavigate() 사용
7. Link 사용
8. 중첩 라우팅 사용

# 03

## useEffect

# useEffect

컴포넌트 렌더링 시, 특정 작업을 실행할 수 있도록 하는 Hook

두번째 인자에 어떤 값이 들어가는지 에 따라  
언제, 어떻게 작업을 실행할 지 제어할 수 있다.

```
const UseEffectTest = () => {  
  const [count, setCount] = useState(0);  
  
  useEffect(() => {  
    console.log(count);  
  }, [count]);  
  
  const handleClick = () => {  
    setCount((c) => c + 1);  
  };  
  
  return (  
    <>  
      <div>UseEffectTest</div>  
      <Button onClick={handleClick}>Up</Button>  
      <p>Count : {count}</p>  
    </>  
  );  
};
```

두번째 인자 : count  
count값이 바뀔 때마다  
console.log(counter) 실행해주세요

## useEffect 예시 (2)

서버에서 데이터를 받아와  
사용하는 작업을 한다고 할 때 ,

useEffect 를 이용해 첫 렌더링 시에만 실행되도록 해서  
불필요하게 데이터를 받아오지 않도록 한다.

Data Fetching 세션에서 더 자세히 다룰 예정

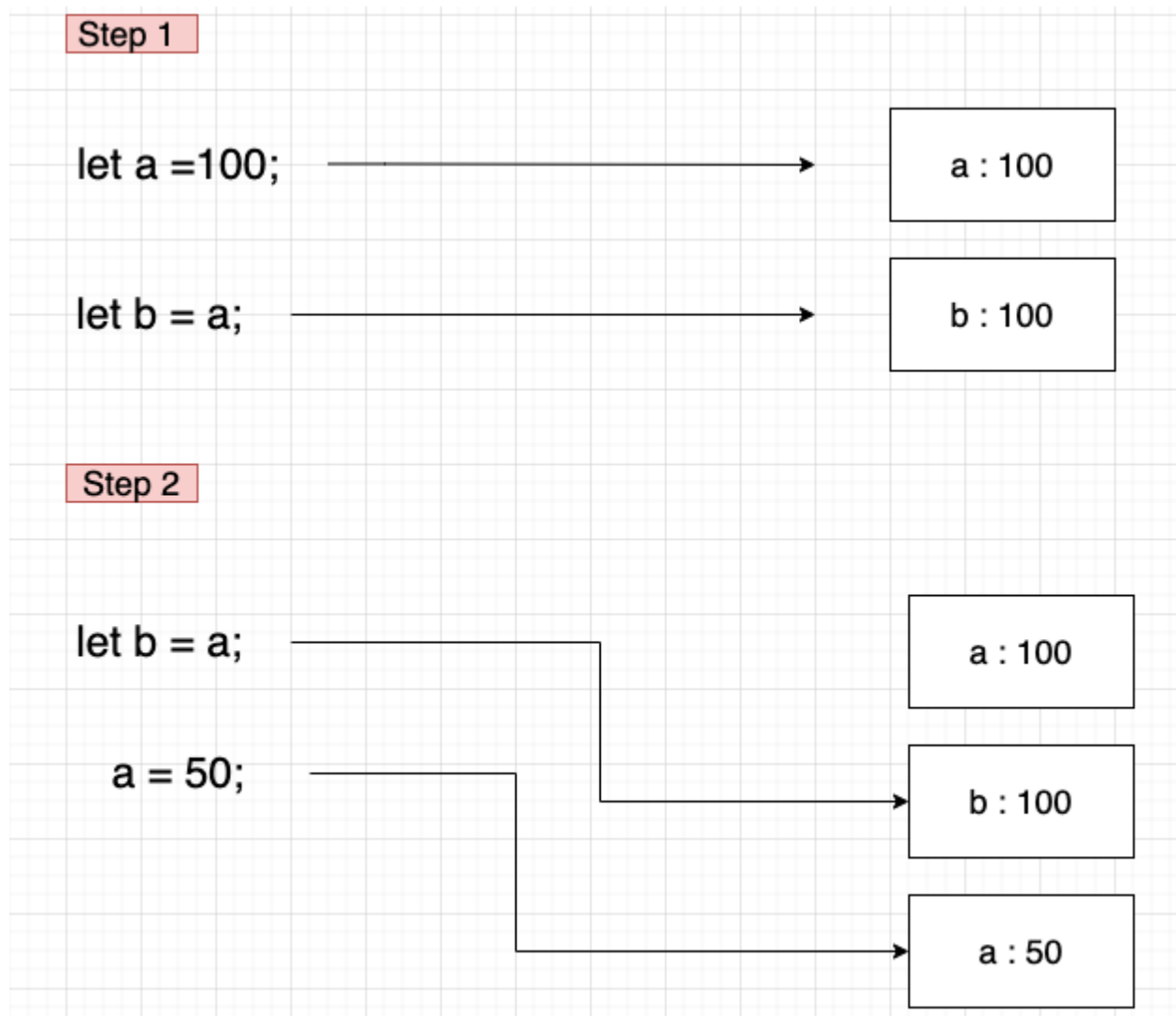
# useEffect 실습

# 04

## React와 불변성

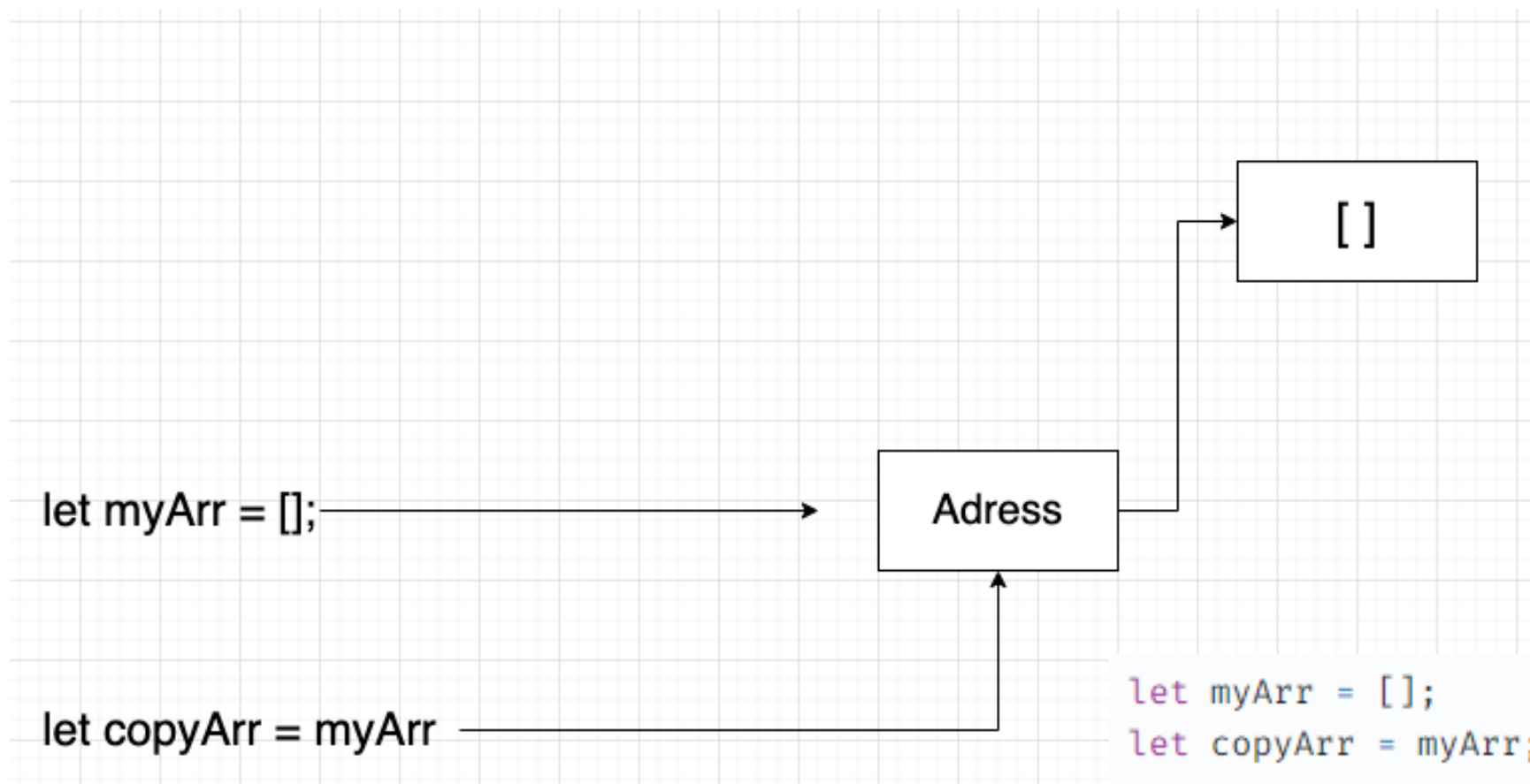


## 원시형



```
let a = 100;  
let b = a;  
a = 50;  
  
console.log(b) // 100
```

## 참조형



```
let myArr = [];  
let copyArr = myArr;  
  
myArr.push("hello");  
  
console.log(copyArr); // ["hello"]
```

리액트 입장에서는 state가 참조하는  
주소값이 여전히 똑같이 때문에 state가  
바뀌었다고 생각하지 않는다.

## 이게 왜 문제가 될까?

```
import React, { useState, useEffect } from "react";

const StatePractice = () => {
  const [value, setValue] = useState([1, 2, 3, 4]);

  const handleClick = () => {
    const copy = value;
    copy.push(5);
    setValue(copy);
    console.log(copy);
    console.log(value);
  };

  return (
    <>
      <div onClick={handleClick}>{value}</div>
    </>
  );
};

export default StatePractice;
```

리엑트는 state를 참조값으로 비교해 변화를 감지한다.

참조값에 변화가 없다 ?

State 변화가 없다고 느낀다 ?

화면 리렌더링이 안된다..

리액트에서는 state 값을 변경할 때 불변성을 지켜야 한다.

변경하는 대신 복사 등의 방법을 통해 새로운 값을 만들어야 한다.

# Spread Operator (...)

```
let a = [1,2,3,4,5]
let b = a;
a.push(5);
console.log(b) // [1,2,3,4,5]
let c = [...a]
a.push(6)
console.log(c) // [1,2,3,4,5]
```

# Spread Operator (...)

```
const StatePractice = () => {  
  const [value, setValue] = useState([1, 2, 3, 4]);  
  
  const handleClick = () => {  
    const copy = [...value];  
    copy.push(5);  
    setValue(copy);  
  };  
  
  return (  
    <>  
      <div onClick={handleClick}>{value}</div>  
    </>  
  );  
};
```

05

과 ㅏ ㅑ ㅓ ㅕ 제 ㅖ ㅗ ㅛ

## 과제 (~7/26 오후 6시)



🔗 **(필수)** 당근마켓 상품 별 상세 페이지를 만들어주세요.

단, 정적 라우팅이 아닌, URL parameter 를 이용해 동적 라우팅 해주세요.

상품의 상세 데이터가 담긴 파일은 따로 json or js 형식의 파일로 만들어 import 하여 사용해주세요.

🔗 **(선택)** 당근마켓 메인 페이지에 상품을 검색할 수 있는 input 태그를 만들고, 추후 검색 기능에 활용할 수 있도록 input 태그의 value값이 타이핑 한대로 즉각 반영되는지 화면에 출력해주세요.



감사합니다

정말 고마워!