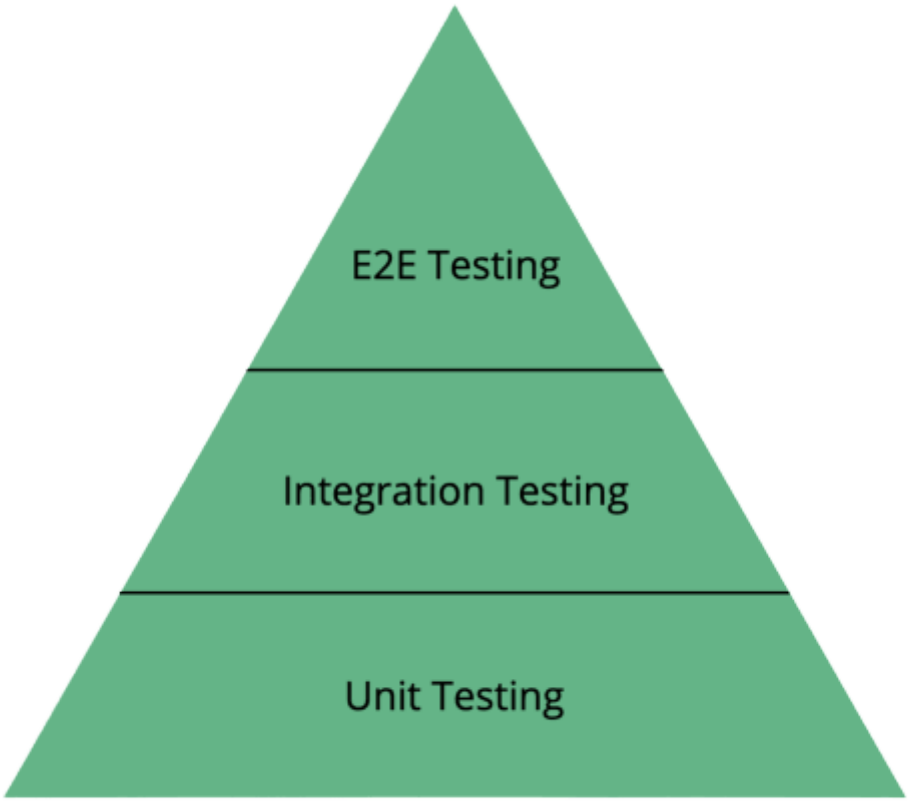
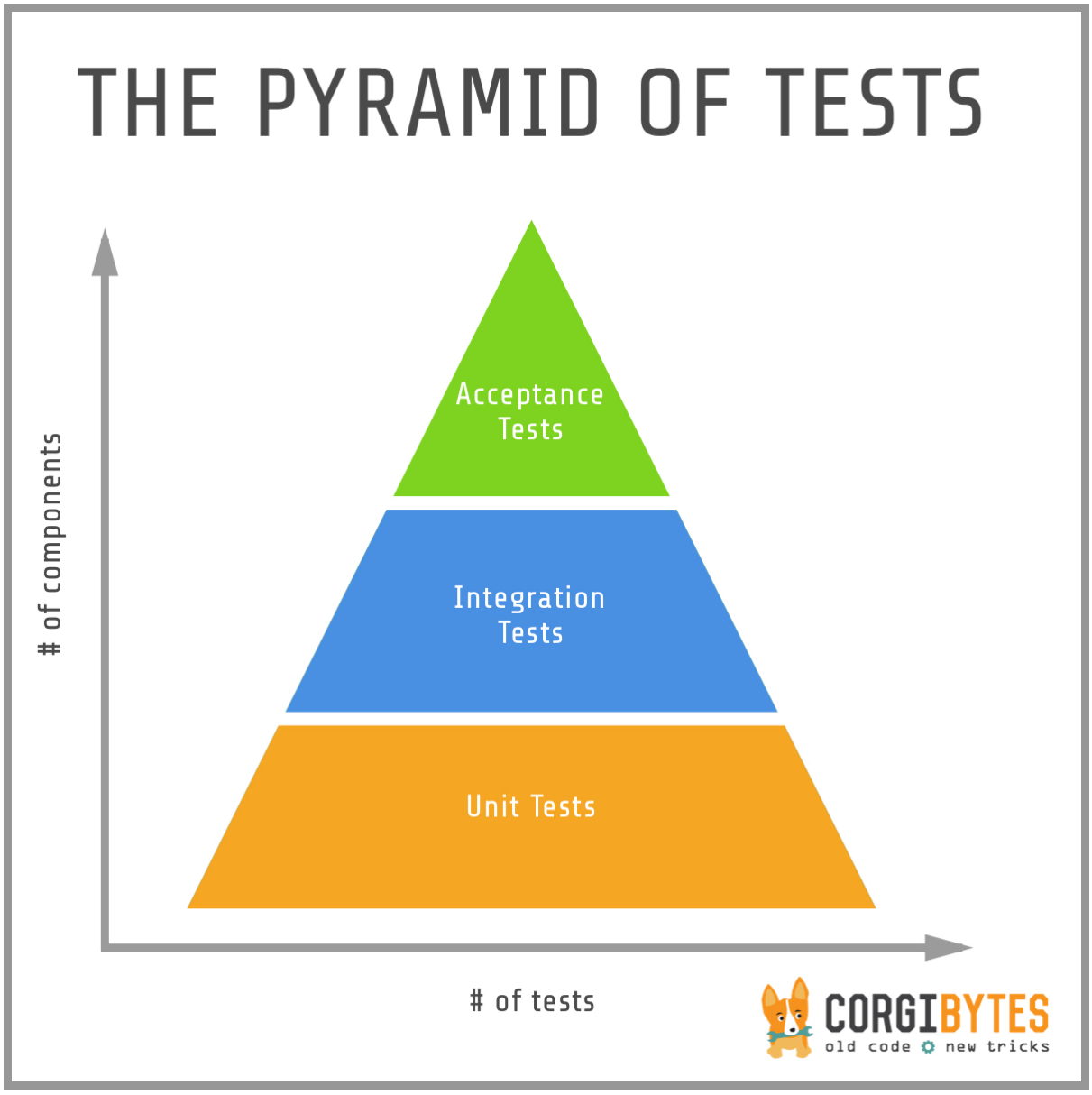
Test 조사

→



혹은



- Unit Tests : 기능 (함수)별 잘 동작하는지
- Integration Tests: 여러 개의 단위 테스트가 잘 동작하고 있는지 검증.
- E2E Tests: 전체적인 동작이 잘 되는지 (사용자 관점에서)

▼ **Difference Between Testing and Debugging**

조사중…

React Testing Library

Behavior Driven Test(행위 주도 테스트) 방법론이 대두 되면서 주목받기 시작됨.

Implementation Driven Test(구현 주도 테스트)	**Behavior Driven Test(행위 주도 테스트)**
- React가 만들어내는 가상 DOM을 기준으로 테스트를 작성	- 실제 DOM 기준 테스트
테스트 로 구현되어있을때 테스트가 h2 태그로 사용되어졌는지	브라우저에서 랜더링되는 컴포넌트가 어떤 모습인지
lib: Airbnb에서 만든 Enzyme	lib: RTL

RTL vs Jest

- React 내에서 테스트를 진행할 때 같이 사용되기에 상호 보완 관계.
  (비지니스 로직은 jest, 컴포넌트 테스트는 RTL)

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

  const decrement = () => setCount(count - 1);

  const increment = () => setCount(count + 1);

  return (
    <>
      <div>{count}</div>
```

```
      <button onClick={decrement}>-</button>
      <button onClick={increment}>+</button>
    </>
  );
};

export default Counter;


--------------------------------- Test

// Counter.test.js
import { render } from '@testing-library/react';

import Counter from './Counter';

describe('Counter test', () => {
  it('should render Counter', () => {
    render(<Counter />);
  });
});

/*
  describe: 그룹화 할수있는 키워드
  it: 개별 테스트 수행
  render: 테스트할 컴포넌트
*/
```

RTL API

Query: 렌더링 된 DOM 노드에 접근하여 엘리먼트를 가져오는 메서드.

- `get`(쿼리 타입)
- `All`(타겟의 개수)
- `ByRole`(타겟 유형)

```
it('should render Counter', () => {
  render(<Counter />);

  // Query Api
  const target = screen.getByRole('button', { name: '+' });

  // Action Api
  fireEvent.click(target);
```

```
  userEvent.click(target);
});
```

API Mocking Test

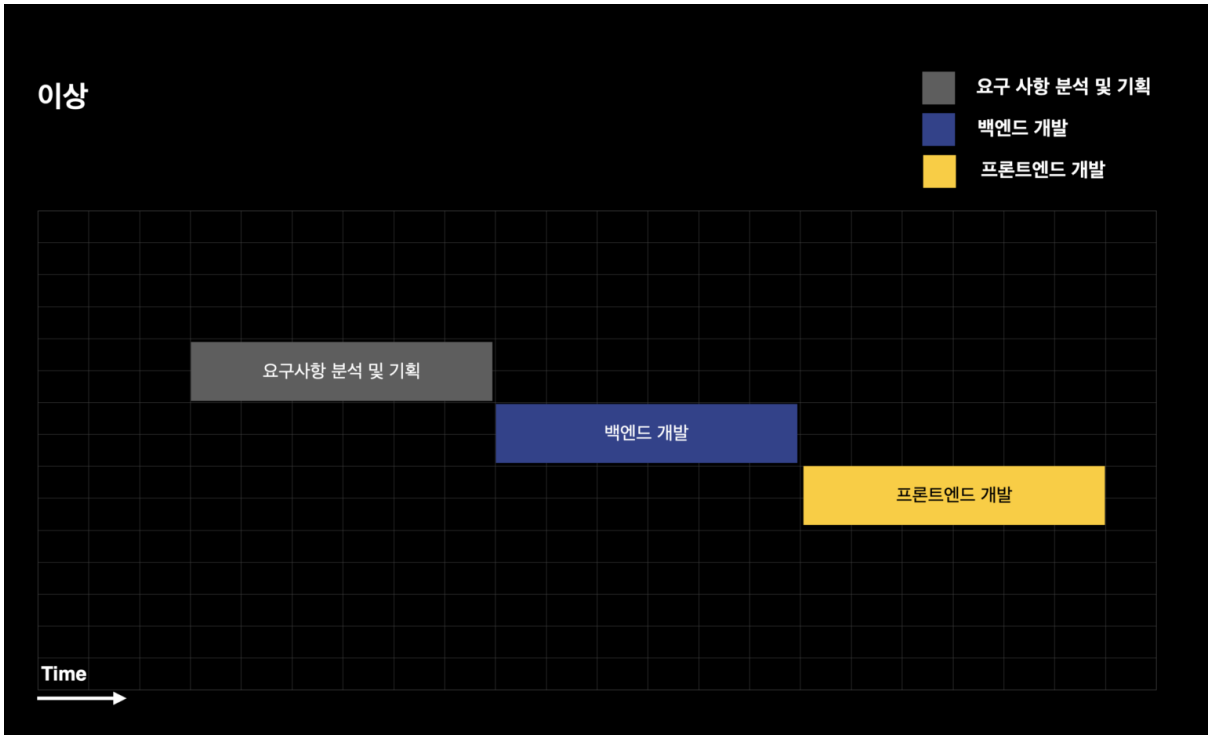
MSW(Mock Service Worker)는 **서비스 워커**(Service Worker)를 사용하여 네트워크 호출을 가로채는 API 모킹(mocking) 라이브러리

→ **서비스 워커**
개발자에게 오프라인 환경을 통제할 수 있는 권한을 부여함으로써, **오프라인 환경에서 캐시(cache)와의 상호작용, 백그라운드 동기화, 푸시 알림 등의** 기능을 가능.

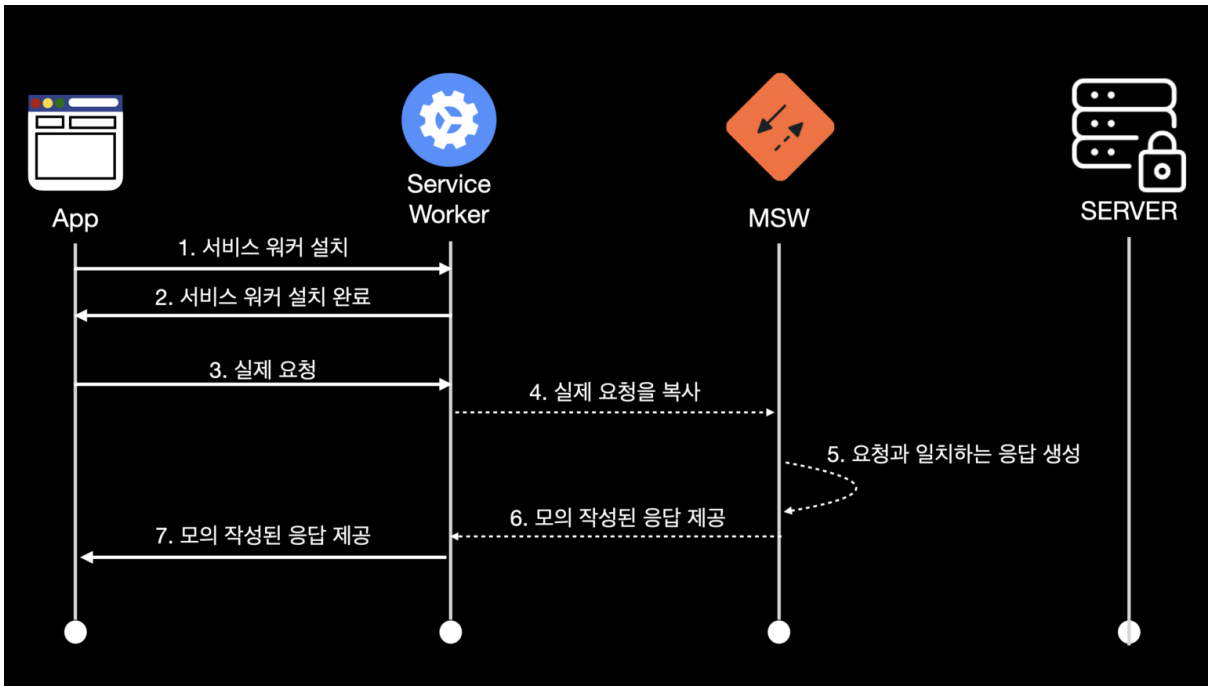
→ **서비스 워커의 특징**

- 서비스 워커는 **사용자가 요청해야 동작**하고 할 일을 **마칠 때까지 꺼지지 않는다**.
- 서비스 워커는 **웹페이지 밖에서 동작**하기 때문에 웹브라우저의 열고 닫힘과 무관하게 동작한다.(웹 개발자는 이러한 서비스 워커의 **lifecycle을 제어할 수 없다**.)
- 웹페이지 밖에서 동작하기 때문에 **DOM요소에 접근할 수 없다**.
- **네트워크 요청을 가로채는 행위도 수행**

문제


이상
요구 사항 분석 및 기획
백엔드 개발
프론트엔드 개발
요구사항 분석 및 기획
백엔드 개발
프론트엔드 개발
Time

MSW 흐름도


App
Service Worker
MSW
SERVER
1. 서비스 워커 설치
2. 서비스 워커 설치 완료
3. 실제 요청
4. 실제 요청을 복사
5. 요청과 일치하는 응답 생성
6. 모의 작성된 응답 제공
7. 모의 작성된 응답 제공

해결

```
// src/mocks/handlers.js
import { rest } from 'msw'

const handlers = [ HTTP GET 메서드 + API URL
    rest.get('/api/products', (req, res, ctx) => {
        return res(
            ctx.status(200), 응답 상태 설정
            ctx.delay(4000), 응답 시간 설정
            ctx.json({
                "items": [
                    { "name": "product-1" },
                    { "name": "product-2" }
                ]
            })
        );                                  응답 결과 설정
    })
]

export default handlers
```