



SCSC 류준범  
**WEB**  
**Warming-up Class**

# 들어가기 전에: Who Am I

- 류준범, 전기정보공학부 15'
- 15년도 컴개실을 수강하면서 코딩을 처음 접했고, 19년도에 Python을 배우면서 제대로 된 코딩 입문을 했습니다.
- 19년도부터 개인 홈페이지에 Javascript와 PHP로 작성된 짧은 코드를 삽입해 동적인 기능을 넣는 시도를 했고, 이것이 저의 첫 웹 개발입니다.
- 20년도 8월 한 달 동안 HTML, CSS, Javascript와 React를 속성 공부하여 저의 첫 상용 웹사이트를 제작해 외부에 공개했습니다.
- 그 때 만든 웹사이트에 기능 추가, 리팩토링, 기술 스택 변경 등을 하면서 23년도 10월 현재까지 이어져 오고 있습니다.

# 주의:

## 저는 전문가가 아닙니다

- 저는 웹 개발 경력이 3년 정도이고, 아직 회사에서 일해 본 적이 없습니다.
- 제대로 된 교육을 받아 본 적 없는 아마추어 엔지니어이기도 합니다.
- WEB Warming-up Class는 제목 그대로 웹에 대한 흥미를 돋울 수 있는 입문용 콘텐츠로 기획되었습니다.
- 그래서, 의도에 충실하게 웹 개발의 재미있는 부분을 보여 드리고 여러분이 웹 개발에 입문하는 데 있을 장벽을 조금이나마 내려 보려고 합니다.



# 목차

웹 개발의 기본 - 화면을 만들자! -  
(HTML, CSS, Javascript)

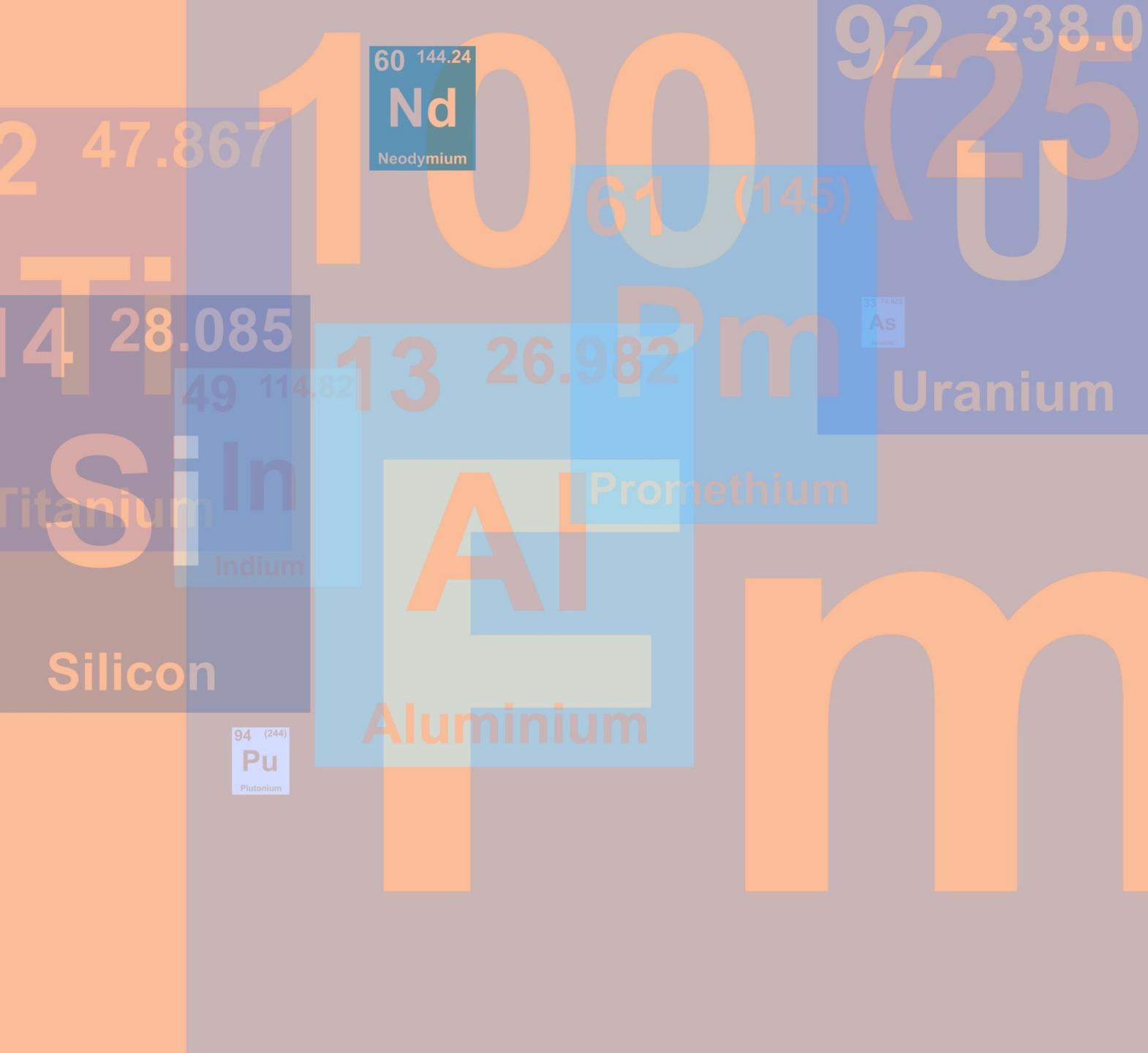
프론트엔드 프레임워크 - 더 멋진 화면을 만들자! -  
(React)

백엔드 프레임워크 - 서버와 상호작용하자! -  
(Node.js, FastAPI)

데이터베이스 - 서버에 데이터를 쌓자! -  
(MySQL, MongoDB)

# 웹 개발의 기본 - 화면을 만들자! -

(HTML, CSS, Javascript)



# 이 챕터에서 소개하는 기술을 알면!

단순한 교수님 홈페이지 같은 것을  
만들 수 있어요.

여러 크기의 텍스트, 단색 배경, 그림,  
하이퍼링크 등으로 이루어진...

좀 투박하긴 해도, 멋있지 않나요??

계승혁  
Kye, Seung-Hyeok

[Department of Mathematics](#)

[Seoul National University](#)

[Seoul](#) 151-742, Korea

Fax: 82-2-887-4694

e-mail: [kye@snu.ac.kr](mailto:kye@snu.ac.kr)

[Curriculum Vitae](#)

[Papers](#), [Talks](#), [Books](#), 기타

[Operator Alegbras: Seminar and People](#)

[강의](#), [강의 동영상](#)

[서울대 수학도서관](#)

[대한수학회](#)

Photographs: [2003](#) [2004](#) [2005](#) [2006](#) [2007](#) [2008](#)

[2009](#) [2010](#) [2011](#) [2012](#) [2013](#) [2014](#) [2015](#) [2016](#)

[2017](#) [2018](#) [2019](#) [2020](#) [2021](#) [2022](#) [2023](#)

[내가 만난 우리나라 꽃](#)

[내가 가본 우리나라 산줄기](#)

# HTML은 무엇일까요?

- HyperText Markup Language
- 웹 페이지를 **구조화하고 내용을 표현**하기 위한, 표준 마크업 언어
- 일반 텍스트에 특별한 태그를 사용해 작성돼요



# HTML은 어떻게 생겼을까요?

```
<!DOCTYPE html>
<html>
<head>
  <title>Title</title>
</head>
<body>
  <h1>Title</h1>
  <p>Paragraph</p>
  <a href="https://example.com">Link</a>
</body>
</html>
```

<!DOCTYPE html>: 문서의 유형을 선언합니다.  
HTML5에서는 <!DOCTYPE html>을 사용합니다.

<html>: HTML 문서의 루트(root) 요소입니다.

<head>: 문서의 메타데이터와 외부 리소스 링크 등을 포함합니다.

<title>: 웹 브라우저의 탭에 표시되는 문서의 제목을 설정합니다.

<body>: 실제로 브라우저에 표시되는 내용을 담습니다.

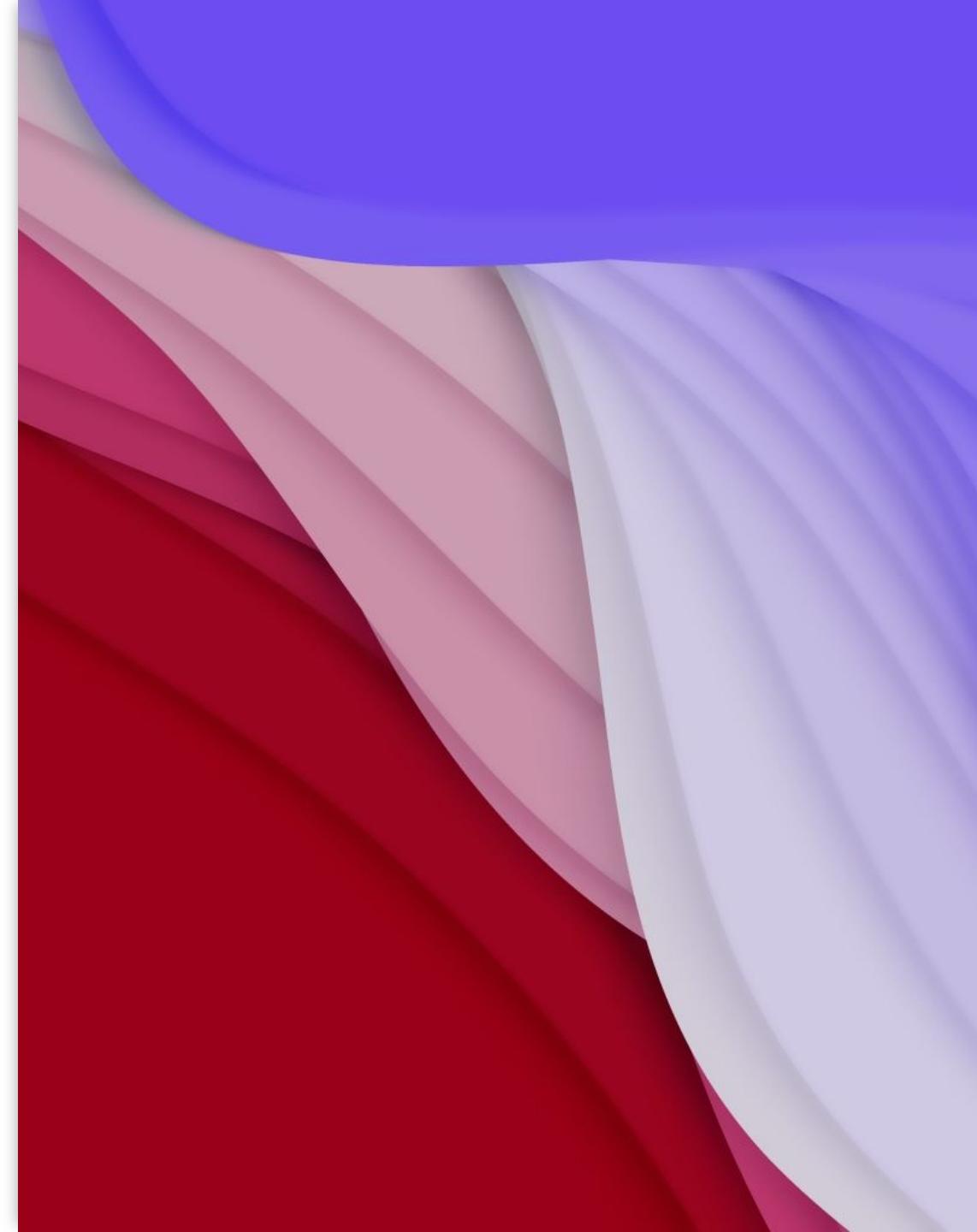
<h1>, <p>, <a> 등: 각각 제목, 문단, 하이퍼링크를 표현하는 태그입니다.

# HTML은 어떻게 배울 수 있을까요?

- 우선 HTML은 프로그래밍 언어가 아닙니다. 마크업 언어입니다.
- 문법이 간단하고, 공부해야 하는 양이 얼마 안 됩니다.
- 자주 사용하는 태그의 이름과 쓰임새만 연결지을 수 있어도 괜찮습니다.
- MDN Web Docs의 [HTML 섹션](#)
- 온라인 강의를 통해서도 쉽게 배울 수 있습니다.
- 가장 추천드리는 한국어 강의는 생활코딩 [WEB1](#)
  - 내용도 내용이지만 코딩에 대한 흥미를 증진시킨다는 점에서도 좋다고 생각합니다

# CSS는 무엇일까요?

- Cascading Style Sheets
- 문서의 **스타일**을 꾸미기 위한, 스타일시트 언어
- 레이아웃, 색상, 글꼴, 간격 등의 디자인 요소를 정의하고 적용



# CSS는 어떻게 생겼을까요?

```
/* selector {  
    property: value;  
} */
```

```
h1 {  
    color: blue;  
    font-size: 36px;  
}
```

```
p {  
    margin: 10px;  
    font-family: Arial, sans-serif;  
}
```

h1, p: 선택자(selector)로, 어떤 HTML 요소에 스타일을 적용할지 지정합니다.

color, font-size, margin, font-family: 속성(property)로, 어떤 스타일을 적용할지 지정합니다.

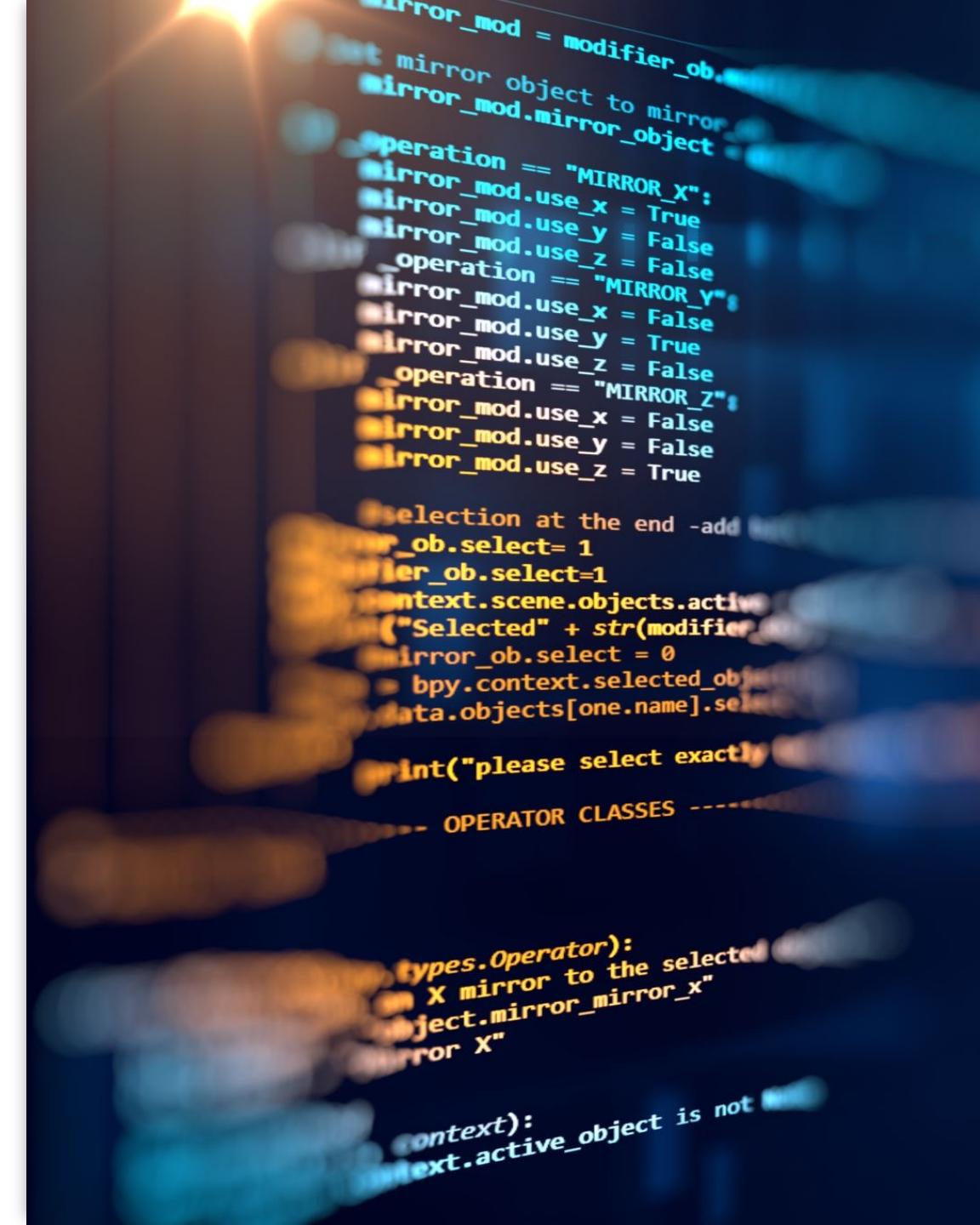
blue, 36px, 10px, Arial, sans-serif: 값(value)으로, 해당 속성에 어떤 값을 적용할지 지정합니다.

# CSS는 어떻게 배울 수 있을까요?

- CSS 역시 프로그래밍 언어가 아닙니다. 스타일시트 언어입니다.
- 제대로 알고 쓰고 싶다면 공부해야 할 양이 많습니다.
- 하지만, 인터넷 검색 능력이 좋다면 필요한 속성을 즉석에서 찾아보면서 배우고 사용할 수도 있습니다.
- MDN Web Docs의 [CSS 섹션](#)
- 생활코딩 [WEB2 - CSS](#)

# Javascript는 무엇일까요?

- 웹 브라우저에서 주로 사용되는 **프로그래밍 언어**
- 웹 페이지에 동적인 요소(**동작**)를 추가하고 특정한 **로직**을 통해 사용자와 **상호작용**하기 위해 사용
- 타 **프로그래밍 언어**와 비슷하게, **프로그래밍 언어**로서 기본적인 요소인 변수, 연산자, 함수 등을 갖추고 있습니다.



# JavaScript는 어떻게 생겼을까요?

// 변수 선언

```
let name = "John";
```

// 함수 정의

```
function greet(person) {  
    return "Hello, " + person + "!";  
}
```

// 함수 호출

```
console.log(greet(name)); // Output: "Hello, John!"
```

# Javascript는 어떤 특징을 가진 언어인가요?

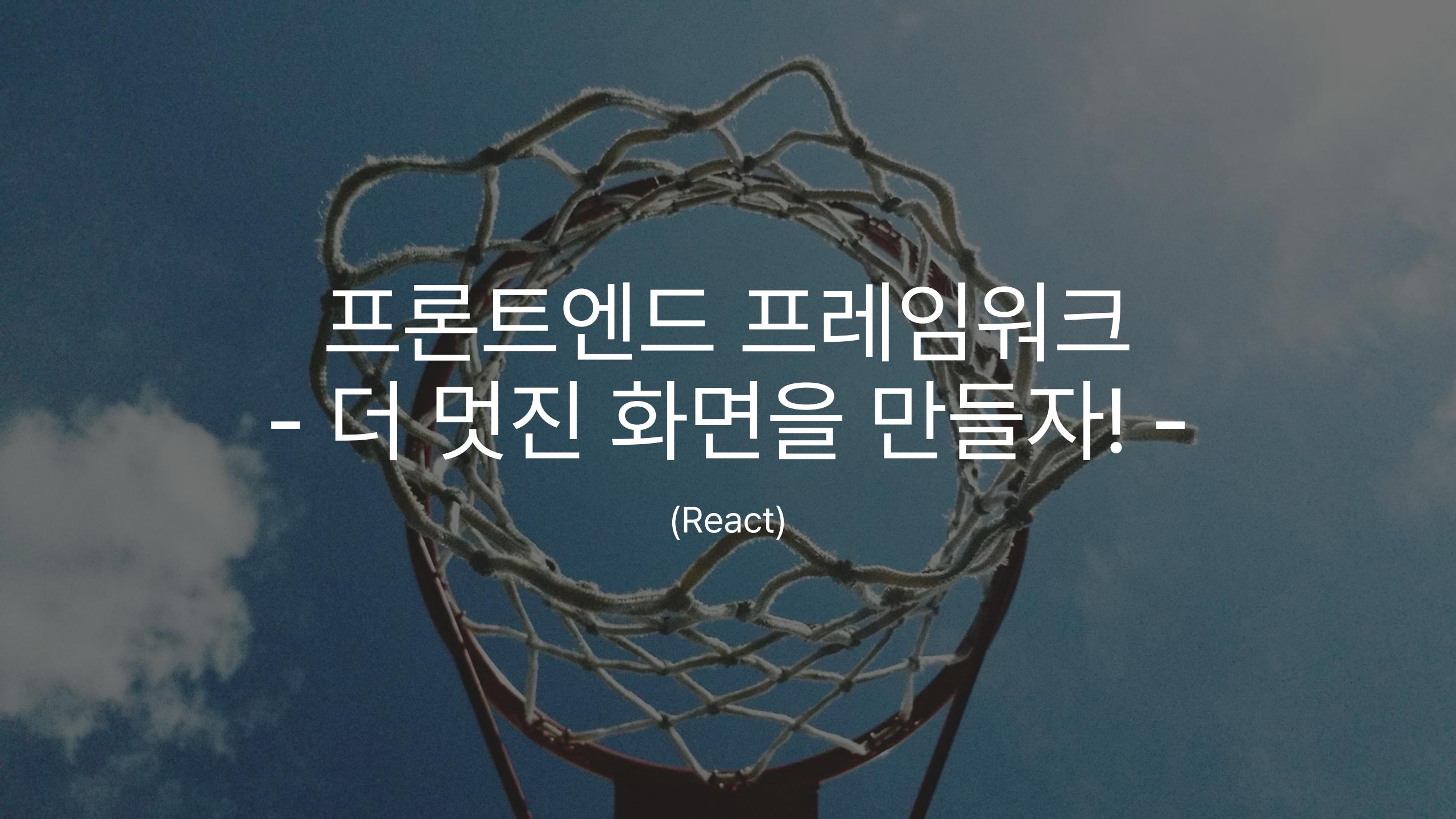
- **동적 타이핑**: 변수의 타입이 실행 시간에 결정됩니다.
- **인터프리터 언어**: (대부분의 경우) 컴파일 없이 코드를 바로 실행할 수 있습니다.
- **이벤트 처리**: 사용자의 마우스 클릭이나 키보드 입력 등의 '이벤트'에 반응 할 수 있습니다.
- **DOM 조작**: 웹 페이지의 Document Object Model (DOM)을 조작할 수 있어, 웹 페이지의 내용과 구조를 동적으로 변경할 수 있습니다.
- **객체 지향 & 함수형 프로그래밍**: 클래스 기반 또는 프로토타입 기반의 객체 지향 프로그래밍과 함수형 프로그래밍을 모두 지원합니다.

# Javascript는 어떻게 배울 수 있을까요?

- 일반적인 프로그래밍 언어를 배우듯이 Javascript를 학습할 수 있습니다.
- 후술할 React 등의 프레임워크를 배운다고 해서, 기본 Javascript의 문법과 특성 공부를 게을리하지 마세요.
- Typescript라는, 정적 타입 체크가 가능한 Javascript의 슈퍼셋 언어가 있습니다.
  - Javascript를 어느 정도 알았다면 이쪽도 공부해 보는 것이 좋습니다.
- MDN Web Docs의 [Javascript 섹션](#)
- 연습은 [백준 온라인 저지](#)를 강추합니다. (언어로 node.js를 선택해 제출)

# HTML + CSS + Javascript

- 지금까지 다룬 세 가지 요소를 합치면 **동작하는 웹 페이지**를 만들 수 있습니다.
- 사실, 이것만으로도 소규모나 중간 규모의 웹 사이트 정도라면 만들 수 있을 것입니다.
- 하지만 여러 가지 이유로 다음에 설명할 React와 같은 라이브러리/프레임워크를 찾게 될 수 있습니다.



# 프론트엔드 프레임워크 - 더 멋진 화면을 만들자! -

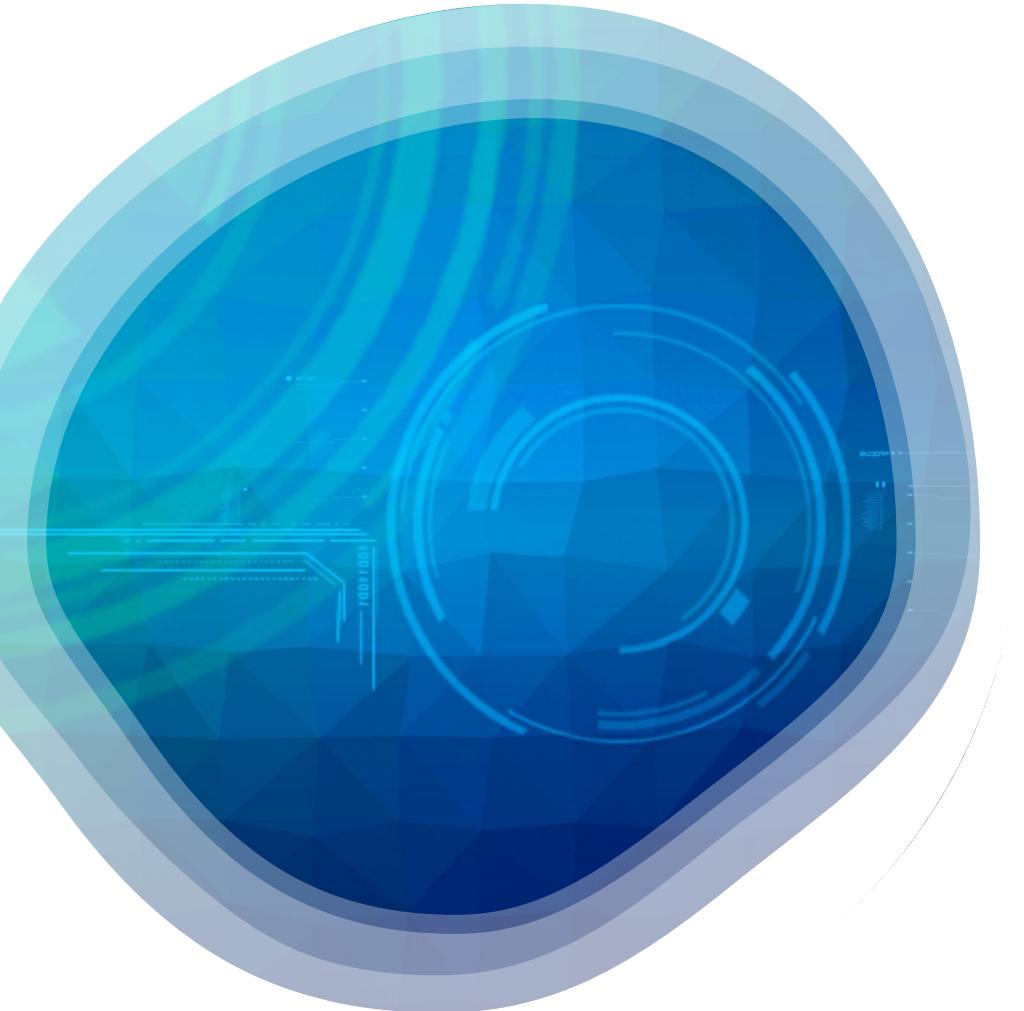
(React)

# 프론트엔드 프레임워크

- React
- Vue
- Angular
- 등등...
- 제가 아는 것이 React 뿐이라 이외의 기술은 설명이 부실합니다.
  - 사실, React가 업계 표준이기도 해요

# 라이브러리/프레임워크를 사용하는 이유?





# 1. 컴포넌트 기반 UI

- 라이브러리 및 프레임워크는 UI를 재사용 가능한 컴포넌트로 분리하는 것을 촉진합니다.
- 이를 통해 코드의 중복을 줄이고, 유지 관리를 용이하게 하며, 일관된 UI를 제공할 수 있습니다.
- 또 사용자 인터페이스를 선언적으로 설계할 수 있어, 코드가 더 읽기 쉽고 예측 가능합니다.
- 상태에 따라 UI가 어떻게 보일지 선언하면, 프레임워크가 나머지를 처리합니다.
- 다음 슬라이드부터 React, Vue, Angular에서 Hello, World!를 짹는 컴포넌트를 어떻게 만드는지 보여드리겠습니다.  
(Code generated by GPT)



# Hello, World! w/ React

```
// App.js
import React from 'react';
import HelloWorld from './HelloWorld';

function App() {
  return <HelloWorld />;
}

export default App;

// HelloWorld.js
import React from 'react';

function HelloWorld() {
  return <h1>Hello, World!</h1>;
}

export default HelloWorld;
```



# Hello, World! w/ Vue

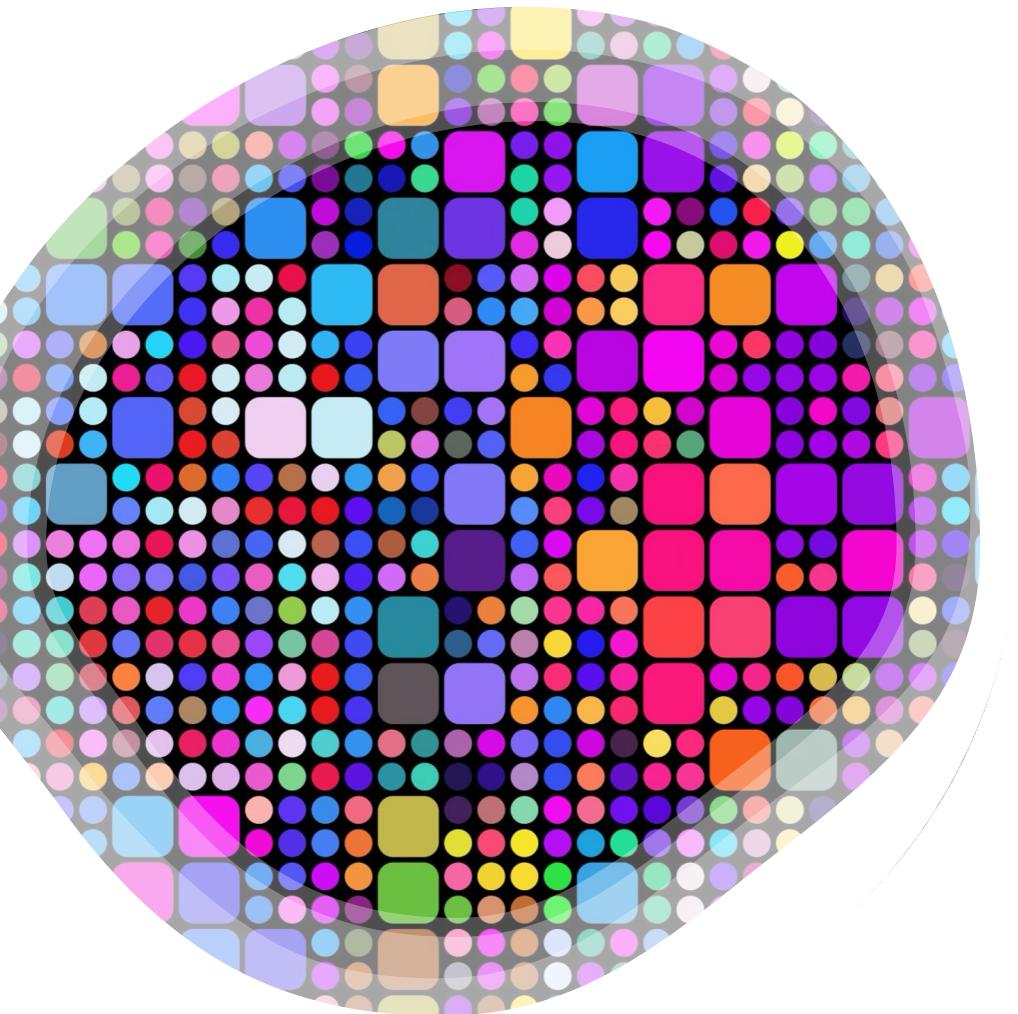
```
<!-- App.vue -->
<template>
  <HelloWorld />
</template>

<script>
import HelloWorld from './HelloWorld.vue';

export default {
  components: {
    HelloWorld
  }
};
</script>

<!-- HelloWorld.vue -->
<template>
  <h1>Hello, World!</h1>
</template>

<script>
export default {
  name: 'HelloWorld'
};
</script>
```



# Hello, World! w/ Angular

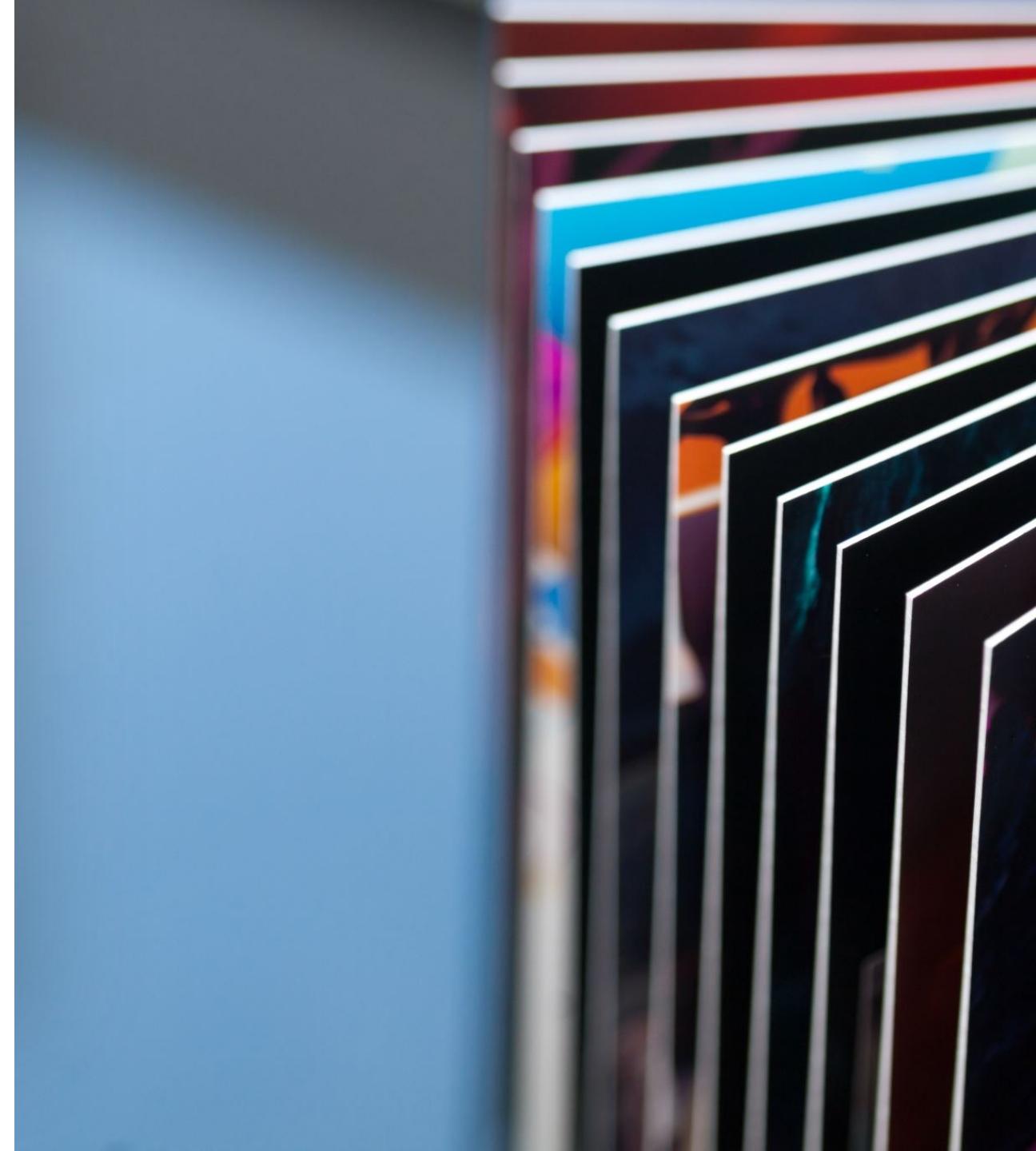
```
// app.component.ts
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: '<app-hello-world></app-hello-world>'
})
export class AppComponent {}  
  
// hello-world.component.ts
import { Component } from '@angular/core';

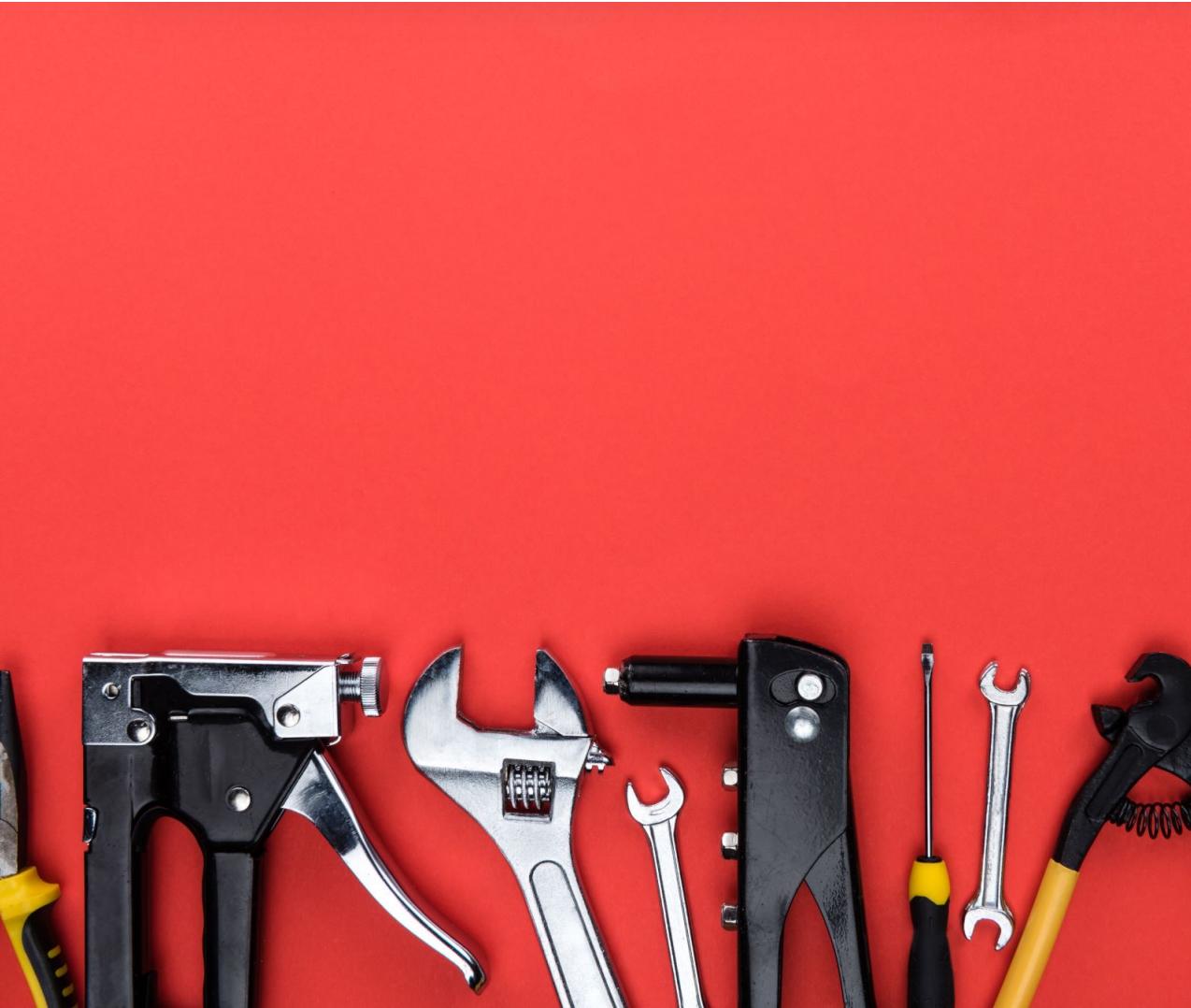
@Component({
  selector: 'app-hello-world',
  template: '<h1>Hello, World!</h1>'
})
export class HelloWorldComponent {}
```

## 2. 반응형 프로그래밍, SPA 지원

- 라이브러리나 프레임워크는 데이터 상태가 변경될 때 자동으로 UI를 업데이트합니다.
- 이를 통해 개발자는 수동으로 DOM을 업데이트하는 복잡한 로직을 작성할 필요가 없습니다.
- SPA(Single Page Application)는 페이지 전체를 다시 로드하지 않고 필요한 부분만 동적으로 업데이트하여 더 나은 사용자 경험을 제공합니다.
- 웹 애플리케이션을 이 SPA로 만들기 위해 라이브러리, 프레임워크를 많이 사용합니다.



### 3. 개발 생산성

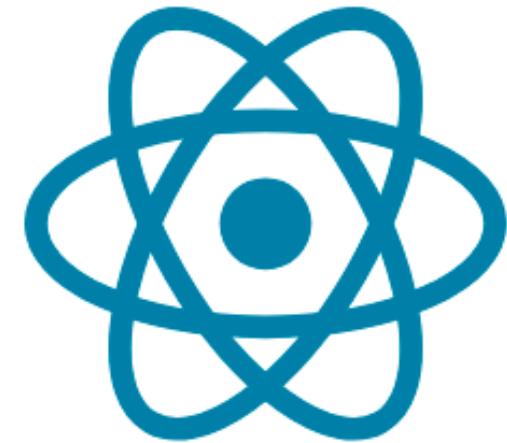


- 라이브러리 및 프레임워크는 일반적으로 개발 도구, 디버깅 도구, 테스팅 유ти리티 등과 함께 제공되어 개발 생산성을 향상시킵니다.
- 타입 체크, 테스팅 도구, 생명주기 메서드 등의 기능을 통해 코드의 안정성을 향상시키고, 유지 관리를 용이하게 합니다.
- 큰 커뮤니티와 활발한 생태계는 문제 해결, 플러그인, 확장 및 추가 리소스를 쉽게 찾을 수 있게 합니다.

# React는 어떻게 배울 수 있을까요?

---

- 먼저, Javascript를 잘 알고 있어야 React를 쉽게 배울 수 있습니다.
  - 특히 ES6 문법, 비동기 처리, 함수형 프로그래밍 개념을 잘 공부하세요.
- [React 공식 문서](#)의 내용이 매우 좋으니, 잘 알아두세요.
  - UI를 작은 컴포넌트 단위로 나누는 방법을 연습하고, 재사용 가능한 컴포넌트를 설계하는 방법을 공부하세요.
  - 상태 관리에 필요한 useState, useEffect 등의 React 내장 훅을 잘 학습하세요.
- 실습을 중심으로 지속적으로 공부하여 능숙해지세요.
  - 개인 프로젝트, 팀 프로젝트, 오픈 소스 프로젝트 참여 등에 도전해 보세요.



# React



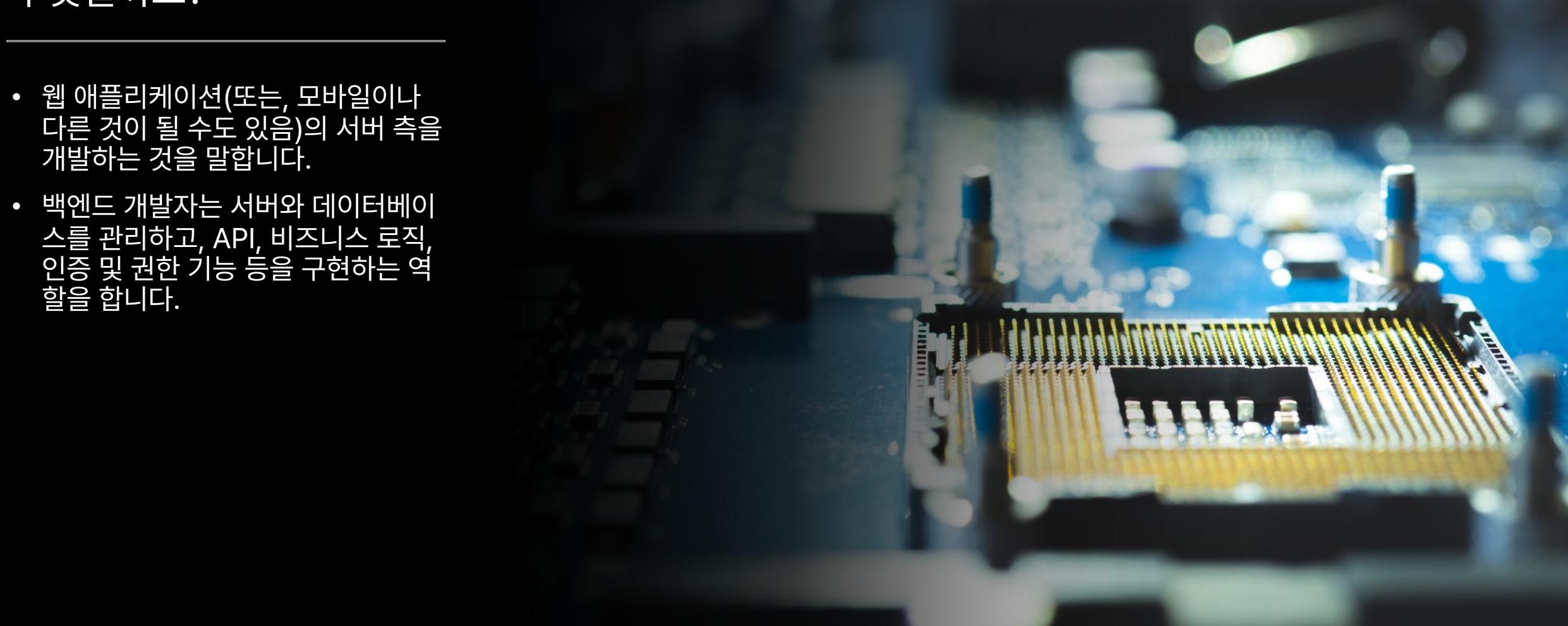
# 백엔드 프레임워크 - 서버와 상호작용하자! -

(Node.js, FastAPI)

## 백엔드 프로그래밍이란 무엇일까요?

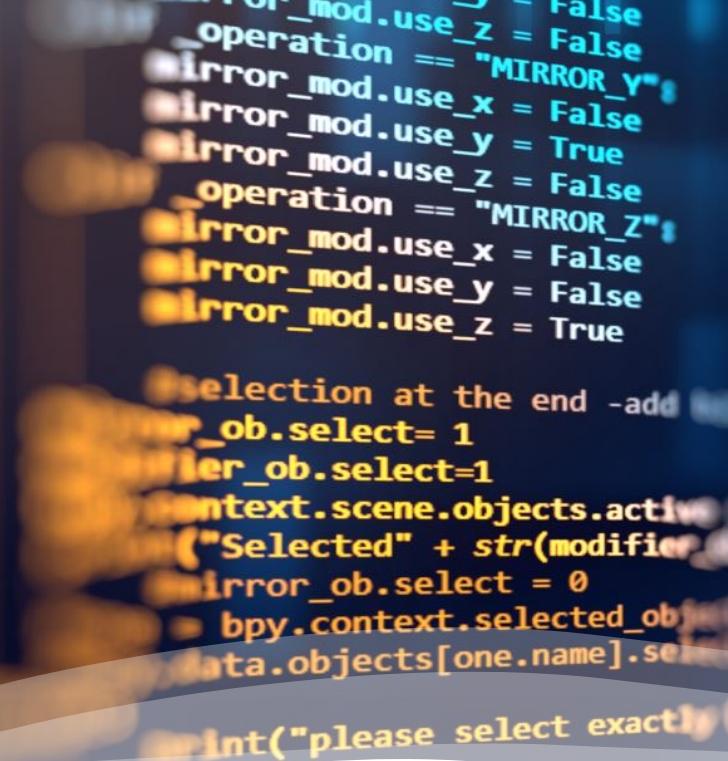
---

- 웹 애플리케이션(또는, 모바일이나 다른 것이 될 수도 있음)의 서버 측을 개발하는 것을 말합니다.
- 백엔드 개발자는 서버와 데이터베이스를 관리하고, API, 비즈니스 로직, 인증 및 권한 기능 등을 구현하는 역할을 합니다.



# API란 무엇일까요?

- 프론트엔드와 통신하기 위한 규약 및 인터페이스를 API(Application Programming Interface)라고 합니다.
- RESTful API, GraphQL 등의 방식으로 구현할 수 있습니다.



A hand is holding a smartphone, which displays a Python script. The script is part of a larger program and includes code for setting up a mirror modifier, selecting objects, and printing a message. The visible code is as follows:

```
    mirror_mod.use_z = False
    operation = "MIRROR_Y"
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
    operation == "MIRROR_Z"
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

    #selection at the end - add
    mirror_ob.select= 1
    mirror_ob.select=1
    context.scene.objects.active = bpy.data.objects[one.name]
    ("Selected" + str(modifier))
    mirror_ob.select = 0
    bpy.context.selected_objects.append(bpy.data.objects[one.name])
    print("please select exactly one object")
```

# REST API

- **GET** 요청은 데이터를 수정 없이 가져오는(**Read**) 기능을 합니다.
- **POST** 요청은 데이터를 새로 만드는(**Create**) 기능을 합니다.
- **PUT** 요청은 이미 만들어진 데이터를 수정하는(**Update**) 기능을 합니다.
- **DELETE** 요청은 데이터를 삭제하는(**Delete**) 기능을 합니다.



**CRUD?**

# 간단한 REST API의 예

---

**GET /users:** 모든 사용자의 정보를 가져옵니다.

---

**GET /users/1:** ID가 1인 사용자의 정보를 가져옵니다.

---

**POST /users:** 새로운 사용자를 생성합니다.

---

**PUT /users/1:** ID가 1인 사용자의 정보를 업데이트합니다.

---

**DELETE /users/1:** ID가 1인 사용자를 삭제합니다.

# GraphQL

---

- Facebook에서 2015년에 발표한 데 이터 쿼리 및 조작 언어입니다.
- RESTful API와의 차이점은 클라이언트 측에서 필요한 데이터의 구조와 형태를 정확하게 지정하고 요청할 수 있다는 것입니다.
- 이를 통해, 불필요한 데이터의 오버페치(over-fetching)나 언더페치(under-fetching) 문제를 효과적으로 해결할 수 있습니다.



# 간단한 GraphQL 쿼리와 응답 예



## 쿼리 (GraphQL)

```
{  
  user(id: "12345") {  
    name  
    email  
  }  
}
```



## 응답 (JSON)

```
{  
  "data": {  
    "user": {  
      "name": "John Doe",  
      "email": "john.doe@example.com"  
    }  
  }  
}
```

# 백엔드 프레임워크



## Java

Spring, Spring Boot



## Javascript

Node.js: Express, Nest.js 등



## Python

Django

Flask

FastAPI



백엔드 프레임워크를  
사용하는 이유



## 1. 개발 생산성과 보안

---

- 프레임워크는 많은 기본 기능과 구조를 제공하므로, 개발자는 처음부터 모든 것을 구축할 필요가 없습니다. 이로 인해 개발 시간이 크게 단축됩니다.
- 프레임워크는 일반적으로 특정한 구조나 패턴을 따르게 되어 있어, 코드의 일관성과 유지 관리성이 향상됩니다.
- 대부분의 유명한 프레임워크는 보안 문제에 대한 해결책을 내장하고 있습니다. 예를 들어, SQL 인젝션, CSRF, XSS 공격 등에 대한 방어 메커니즘이 포함되어 있을 수 있습니다.



## 2. 데이터베이스, 미들웨어, 배포의 용이성

- 많은 프레임워크는 데이터베이스와의 통합을 쉽게 만들어주는 ORM(Object-Relational Mapping) 도구나 데이터베이스 드라이버를 제공합니다.
- 사용자 인증, 로깅, 캐싱, 세션 관리 등의 기능을 쉽게 추가할 수 있도록 미들웨어나 플러그인 시스템을 제공합니다.
- 프로덕션 환경에서의 성능 최적화, 로깅, 에러 핸들링 등의 기능을 제공하여 실제 서비스 환경에 쉽게 배포할 수 있게 돋습니다.

### 3. 확장성과 커뮤니티

---

- 프레임워크는 대부분 확장성을 고려하여 설계되어 있습니다. 추가 모듈이나 미들웨어를 쉽게 통합할 수 있습니다.
- 프레임워크를 사용하면 공통적으로 사용되는 기능을 컴포넌트나 미들웨어로 쉽게 재사용할 수 있습니다.
- 프레임워크는 테스팅과 디버깅을 위한 도구와 라이브러리를 제공하거나 통합하기 쉽게 설계되어 있습니다.
- 대부분의 인기 있는 프레임워크는 활발한 커뮤니티를 가지고 있어, 문제 해결, 플러그인 개발, 최적화 전략 등에 대한 지원을 받을 수 있습니다.



# Node.js Express 코드는 어떻게 생겼을까요?

```
const express = require('express');
const app = express();
const PORT = 3000;

app.get('/', (req, res) => {
    res.send('Hello, World!');
});

app.listen(PORT, () => {
    console.log(`Server is running on
http://localhost:${PORT}`);
});
```

# FastAPI 코드는 어떻게 생겼을까요?

```
from fastapi import FastAPI

app = FastAPI()

@app.get("/")
def read_root():
    return {"message": "Hello, World!"}
```



## 백엔드 프레임워크 중 무엇을 배워야 하나요?

- 백엔드 개발자 국내 취업에 관심이 있다면, 무.조.건. Java Spring을 배우세요!!
  - 훌륭한 아키텍처를 가지고 있고, 국내 기업들이 Spring 인재를 많이 필요로 합니다.
- 프론트엔드 프로그래밍에도 관심이 있다면 Javascript로 개발할 수 있는 Node.js를 공부해 보세요.
- Python을 이미 알고 있다면, Django와 FastAPI(또는 Flask) 중 하나를 공부해 보세요.

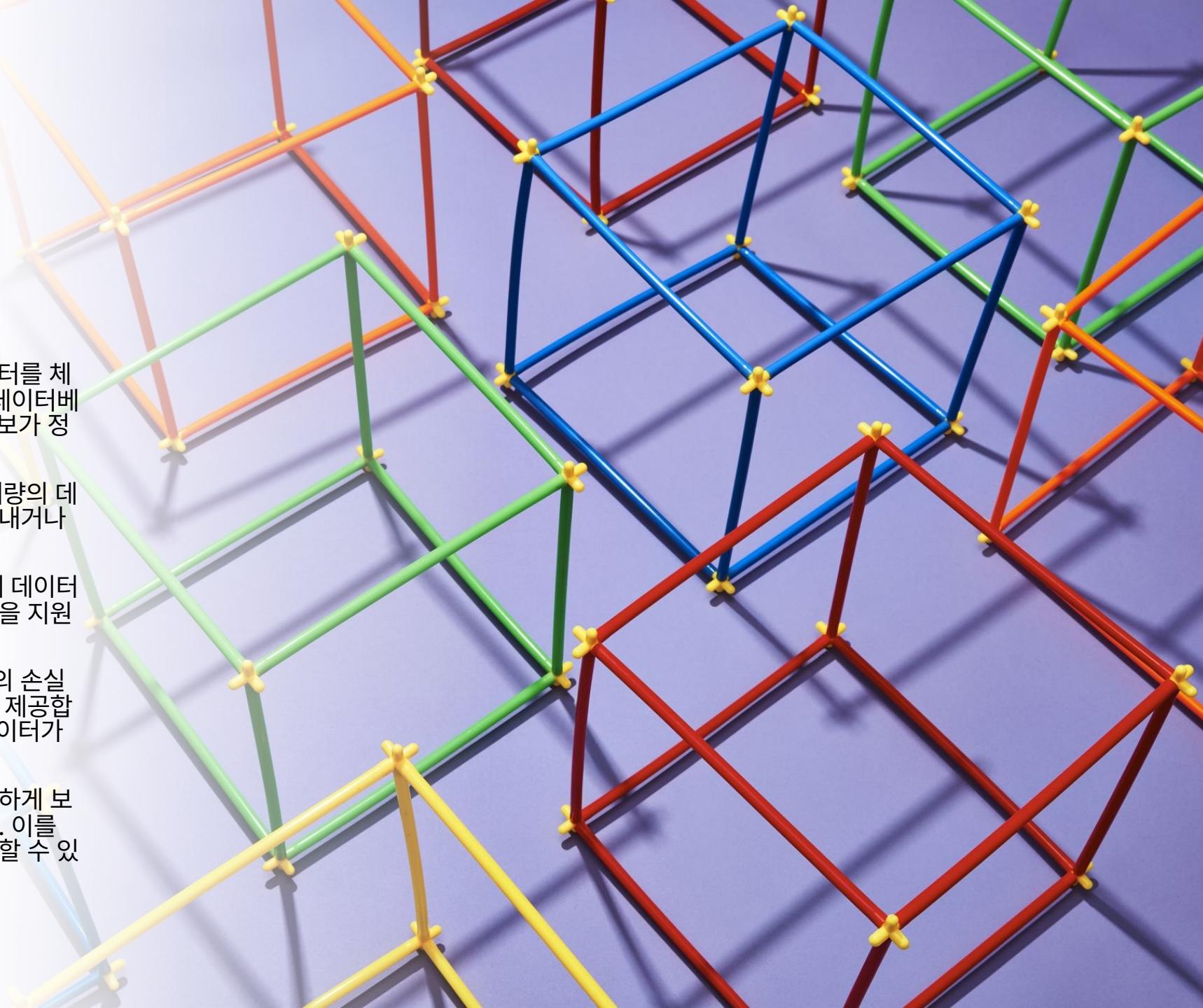


# 데이터베이스 - 서버에 데이터를 쌓자! -

(MySQL, MongoDB)

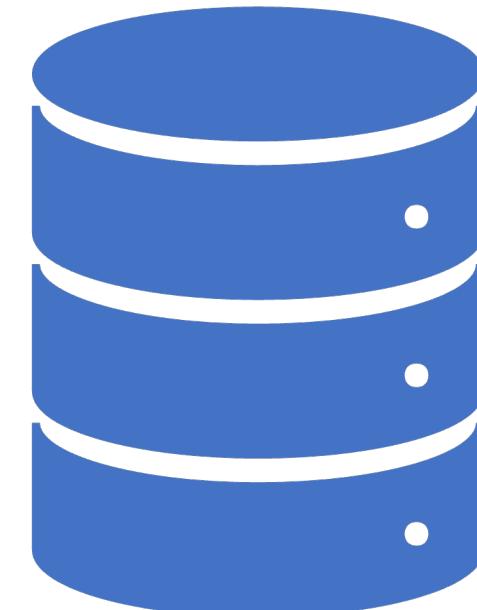
# 데이터베이스란 무엇일까요?

- **정리된 데이터 저장:** 데이터베이스는 데이터를 체계적으로 저장합니다. 예를 들어, 학교의 데이터베이스에는 학생, 선생님, 수업 등에 관한 정보가 정리되어 저장될 수 있습니다.
- **빠른 검색 및 업데이트:** 데이터베이스는 대량의 데이터 중에서도 원하는 정보를 빠르게 찾아내거나 수정할 수 있게 도와줍니다.
- **동시 접근:** 여러 사람이나 시스템이 동시에 데이터베이스에 접근하여 정보를 읽거나 쓰는 것을 지원합니다.
- **데이터의 안정성:** 데이터베이스는 데이터의 손실이나 오류를 방지하기 위한 다양한 기능을 제공합니다. 예를 들어, 전원이 갑자기 꺼져도 데이터가 안전하게 보호됩니다.
- **보안:** 데이터베이스는 중요한 정보를 안전하게 보호하기 위한 보안 기능을 가지고 있습니다. 이를 통해 무단 접근이나 데이터의 유출을 방지할 수 있습니다.



# 상용 데이터베이스의 종류

- **RDBMS (Relational DataBase Management System)**
  - MySQL, PostgreSQL, Oracle DB, MariaDB, SQLite 등등
- **NoSQL**
  - MongoDB, Cassandra 등등



# MySQL 쿼리는 어떻게 생겼을까요?

- 예시 **students** 테이블: 이 테이블은 학생들의 정보를 저장하며, 각 학생은 id, name, age, grade 컬럼을 가집니다.
- 테이블 생성

```
CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    age INT,
    grade VARCHAR(50)
);
```

# MySQL 쿼리는 어떻게 생겼을까요?

- 데이터 입력

```
INSERT INTO students (name, age, grade) VALUES ('John Doe', 20, 'Sophomore');
```

- 데이터 조회

```
SELECT * FROM students;
```

```
SELECT * FROM students WHERE age >= 21;
```

- 데이터 수정

```
UPDATE students SET grade = 'Junior' WHERE name = 'John Doe';
```

- 데이터 삭제

```
DELETE FROM students WHERE name = 'Jane Smith';
```

# MongoDB는 어떻게 생겼을까요?

- 예시 **students** 컬렉션: 이 컬렉션은 학생들의 정보를 저장하며, 각 학생은 name, age, grade 필드를 가집니다.

- 데이터 입력

```
db.students.insert({name: "John Doe", age: 20, grade: "Sophomore"});
```

- 데이터 조회

```
db.students.find();
```

```
db.students.find({age: {$gte: 21}});
```

# MongoDB는 어떻게 생겼을까요?

- 데이터 수정

```
db.students.update({name: "John Doe"}, {$set: {grade: "Junior"}});
```

- 데이터 삭제

```
db.students.remove({name: "Jane Smith"});
```

# 데이터베이스는 어떻게 배울 수 있을까요?

- 우선, SQL에 대해서 공부하세요.
  - 여유가 된다면, [SQL 개발자 시험 \(SQLD\)](#)을 준비해보는 것도?
- 다음으로, MySQL, Oracle DB, PostgreSQL 중 하나 이상의 RDBMS를 공부하세요.
- MongoDB와 같은 NoSQL 데이터베이스도 공부하면 좋습니다.
- 컴퓨터공학부에서 '데이터베이스' 과목을 수강해 이론적 지식을 강화하는 것도 좋습니다.



# My Story: Pokémoem



- [웹사이트](#)
- [포트폴리오](#)

# 문제의식

- 포켓몬스터 랭크배틀 코어 유저였던 저는 사람들이 편하게 더 많은 배틀 관련 데이터를 확인할 수 있으면 좋겠다는 생각을 가지게 되었습니다.
- 우선 공식 앱에서 제공하는 데이터의 가공을 시도 했는데, 어떻게 하는지 전혀 알지 못했지만 꼭 해 보고 싶었기 때문에 지인 개발자에게서 API를 크롤링 하는 코드를 제공받고 이를 기반으로 원하는 데이터를 시각화하는 Python 코드를 작성하여 Google Colab을 통해 온라인 상에 공유하였습니다.
- 이로써 배틀 데이터의 시각화와 분석이 가능해졌지만 더 해 보고 싶은 것이 많았습니다.



# 초기 기획과 구현

- 기획 단계에서 가장 중요하게 생각한 것은 사용자들의 요구에 맞는 기능을 구현하고 기본에 충실하자는 것이었습니다. 그래서 첫 기획 의도였던 시각화와 분석 기능에 가장 신경썼습니다.
- 또 포켓몬의 육성 형태(속칭 '샘플')를 SNS와 유사한 형태로 쉽게 공유할 수 있도록 기획했습니다.
- 웹 사이트를 실제로 구현하기 위해, 각종 웹 기술을 배워야 했는데 아무것도 모르는 상태였기 때문에 Udemy의 "[The Web Developer Bootcamp](#)" 강의를 일주일 간 수강하면서 HTML, CSS, Javascript의 기초를 공부했습니다
- 다음으로 Express, MySQL, React 등의 기술을 공부했는데, 모든 게 처음인지라 굉장히 어렵게 느껴졌습니다. 구현하기 어려운 부분은 나중으로 미루는 대신, 지금 당장 구현할 수 있는 부분을 빠르게 만들어보자는 마인드로 작업을 진행했습니다.
- 웹 사이트의 프로토타입은 개발을 시작한 지 한 달 만에 AWS EC2로 배포하였습니다.



# 운영 과정과 임팩트

- 국내에 정보 소스가 부족해 여러 해외 사이트들에 의존해야 했던 포켓몬 배틀 씬의 상황에서 포케모음 웹 사이트의 등장은 성공적인 것으로 평가받았습니다. 특히 SNS 형식의 샘플 공유 기능이 많은 호평을 받았습니다.
- 데미지 계산기를 개발하고, 디스코드 서버를 만들고 디스코드를 활용해 게임 대회를 진행하는 등의 시도를 했습니다.
- 2022년 9월부터는 기술 스택을 변경해 리팩토링 작업을 하여, 완전히 새로운 UI와 백엔드를 가진 사이트로 이전하였습니다.
- 포케모음은 매일 1만 2천 조회수 이상을 기록하는 웹 사이트이며, SNS 연동으로 포케모음 계정을 만든 이용자 수는 2200명 이상, 사용자 모임 디스코드 서버 인원은 600명 이상이 되었습니다. 포켓몬 배틀 유저 풀이 적은 국내에서 손에 꼽히는 규모로 성장하였으며, 이는 현재 진행형입니다.



# 이 버튼을 누르면 무슨 일이 일어날까요?

- 웹 브라우저와 웹 서버의 상호작용이 어떻게 일어나는지 좀 더 자세히 알려드릴게요
- 지금부터는 실제 웹 서비스 코드를 살펴볼 거예요  
(1도 이해하지 못해도 됩니다)

 pokemoem ver 2.1 

메인 도감 샘플 파티 계산 대회 유저 로그인

 (2/2) 포켓몬 도감이 스칼렛 바이올렛 포켓몬 홈 배틀 데이터와 대응합니다!

## 포켓몬 도감

SV 시즌 10 배틀 데이터 업데이트 시간: 2023-09-20 PM 2:59 (GMT+0900)

 배틀 데이터 재다운로드

포켓몬은 속도를 위해 세�이 유지되는 동안은 배틀 데이터를 다시 다운로드하지 않아요. 오류가 생겼거나 시간이 많이 지난 경우 버튼을 눌러 다운로드해주세요.

도감 상세 검색 영칭/일칭 검색

정렬 기준

오늘의 싱글  오늘의 더블  시즌을 선택하세요 ▾  원하는 시즌 싱글 - SV  원하는 시즌 더블 - SV  원하는 시즌 싱글 - SWSH  원하는 시즌 더블 - SWSH  전국도감순

포켓몬의 이름으로 검색하세요

도감 리스트 PNG로 저장

	망나뇽	#1(0)		드래곤 비행	정신력 멀티스케일	91 134 95 100 100 80 600
	날개치는머리	#2(0)		고스트 페어리	고대활성	55 55 55 135 135 135 570
	타부자고	#3(0)		강철 고스트	황금몸	87 60 95 133 91 84 550

# React 버튼 컴포넌트 코드

버튼 클릭은  
onClick을 트리거

```
<Button variant="outlined" sx={{margin: 5, width: 160}}
    onClick={() => {
        downloadBattleData(0, 0, 'sc', false).then((data) => {
            setTodaySeason(data.ranking.season);
            setTodayUpdateTime(data.ranking.created);
            setTodaySingle(data.ranking.data);
            setTodaySingleDetails(data.details.data);
        });
        downloadBattleData(1, 0, 'sc', false).then((data) => {
            setTodayDouble(data.ranking.data);
            setTodayDoubleDetails(data.details.data);
        });
        setSliced(30);
    })>
    배틀 데이터 재다운로드
</Button>
```

# downloadBattleData

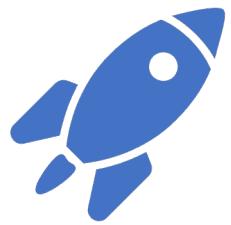
JS의 fetch API로  
백엔드 서버에  
GET 요청을 보냄

```
import { loadJsonSession, saveJsonSession } from "./json";
import { BattleData, Details } from "./types";

export const downloadBattleData = async (rule: number, season: number | string = 0, version: string = "sc", rankingOnly: boolean = true) => {
  const ruleString = rule === 0 ? 'single-' : 'double-';
  fetch(`https://api.pokemoem.com/battlestat/ranking/${season === 0 ? "today" : `past/${season}`}?rule=${rule}&version=${version}`)
    .then(response => {
      if (!response.ok) throw Error();
      return response.json();
    })
    .then(response => {
      saveJsonSession(`${ruleString}${version}-${season}-ranking`, response);
    })
    .catch(console.error);

  if (!rankingOnly) await fetch(`https://api.pokemoem.com/battlestat/details/${season === 0 ? "today" :
`past/${season}`}?rule=${rule}&version=${version}`)
    .then(response => {
      if (!response.ok) throw Error();
      return response.json();
    })
    .then(response => {
      saveJsonSession(`${ruleString}${version}-${season}-details`, response);
    })
    .catch(console.error);

  const battledata: BattleData = {
    ranking: loadJsonSession(`${ruleString}${version}-${season}-ranking`),
    details: rankingOnly ? [] as Details[] : loadJsonSession(`${ruleString}${version}-${season}-details`)
  };
  return battledata;
}
```



# api.pokemoem.com

- 저희 웹 사이트 서버는 Ubuntu 인스턴스로서 3개의 서로 다른 포트에 React, FastAPI, MongoDB 서버가 실행 중입니다.
- Ubuntu에 Nginx가 설치되어 있어, 각 포트를 서로 다른 웹 URL에 대응시켜 줍니다.
- (www.)pokemoem.com은 React로,  
api.pokemoem.com은 FastAPI로 프록시되고 있습니다.



# FastAPI 코드

## 데이터베이스에서 데이터를 받아와 원하는 대로 가공한 뒤 반환

```
@router.get("/ranking/today", response_description="Read Battle Rankings of today", response_model=PokemonRanking)
def today_pokemonranking(rule: int, request: Request):
    if len(today_ranking := list(request.app.database["pokemon_ranking_sc"].find({"rule": rule, "season": {"$type": 16}, "data": {"$gt": {}}}))\
        .sort([("season", -1), ("created", -1)]).limit(1)) > 0:
        today_ranking = today_ranking[0]
    if type(today_ranking["created"]) is int:
        time = datetime.fromtimestamp(today_ranking["created"])
    else:
        time = datetime.strptime(today_ranking["created"], '%Y-%m-%d %H:%M:%S')
    yesterday = time - timedelta(days=1)
    if len(yesterday_ranking := list(request.app.database["pokemon_ranking_sc"]\
        .find({"season": today_ranking["season"], "rule": rule, "data": {"$gt": {}}, "created": {"$lt": int(yesterday.timestamp())}}))\
        .sort("created", -1).limit(1)) > 0:
        yesterday_ranking = yesterday_ranking[0]
    else:
        yesterday_ranking = request.app.database["pokemon_ranking_sc"] \
            .find({"season": today_ranking["season"] - 1, "rule": rule, "data": {"$gt": {}}}) \
            .sort("created", -1).limit(1)[0]
    ranking_array = today_ranking['data']
    for index, pokemon in enumerate(ranking_array):
        try:
            last_ranking = yesterday_ranking['data'].index(pokemon)
        except:
            last_ranking = -1
        ranking_array[index]["previous"] = last_ranking
    today_ranking['data'] = ranking_array
    return today_ranking
    raise HTTPException(status_code=status.HTTP_404_NOT_FOUND, detail="Battle Ranking not found")
```

# 1만 개가 넘는 도큐먼트 중 필요한 것만 쿼리!

pokemoem.pokemon\_ranking\_sc

10.9k 1  
DOCUMENTS INDEXES

Documents Aggregations Schema Indexes Validation

Filter Type a query: { field: 'value' } Explain Reset Find Options ▾

+ ADD DATA ▾ EXPORT DATA ▾ 1 – 20 of 10938 ⏪ ⏴ ⏵ ⏴ ⏵

```
_id: ObjectId('63da6dd68510f2b4c0e0fd07')
season: 1
date: "2023-01-01"
rule: 0
data: Array (150)
  0: Object
    id: 1000
    form: 0
  1: Object
    id: 887
    form: 0
  2: Object
    id: 635
    form: 0
  3: Object
    id: 908
    form: 0
  4: Object
    id: 149
    form: 0
  5: Object
    id: 778
    form: 0
```

MongoDB  
데이터베이스 컬렉션  
pokemon\_ranking\_sc

# 웹 브라우저가 응답을 받았어요!

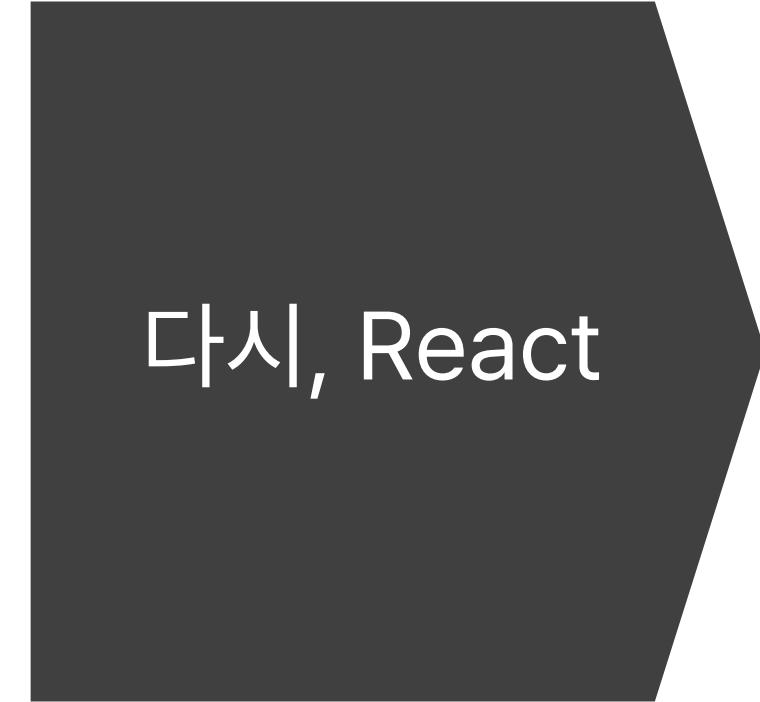
- Chrome 같은 브라우저라면 [검사] - [네트워크]에서 서버로부터 어떤 응답을 받았는지 확인할 수 있어요.

The screenshot shows the Chrome DevTools Network tab with the following details:

- Network Tab Headers:** Shows network activity with a green checkmark indicating success. It includes filters for Fetch/XHR, JS, CSS, Img, 미디어, 글꼴, 문서, WS, Wasm, and 기타.
- Timeline:** A horizontal timeline at the bottom shows time points: 5000 ms, 10000 ms, 15000 ms, and 20000 ms.
- Request Details:** The main pane displays a request for "today?rule=0&version=sc".
  - Headers:** Includes "Content-Type: application/json; charset=UTF-8" and "Accept: \*/\*".
  - Response:** Status is 200 OK, with a size of 1.6 MB and a timestamp of 2023-09-20T08:06:00+09:00.
  - Content:** The response body is a JSON object:

```
{<span>season: 10, date: "2023-09-20", rule: 0, id: 149</span>, <span>created: 1695207961</span>, <span>data: [<span>{id: 149, form: 0, previous: 0, next: 150}</span>]</span>, <span>date: "2023-09-20"</span>, <span>rule: 0</span>, <span>season: 10</span>}
```
- Console Tab:** Shows a single error message: "Failed to load resource: net::ERR\_BLOCKED\_BY\_CLIENT pagead2.googlesyndic...".

# 다시, React



업데이트된 데이터를 바탕으로  
화면의 일부분을 갱신

pokemoem ver 2.1

메인 도감 샘플 파티 계산 대회 유저 로그인

(2/2) 포켓몬 도감이 스칼렛 바이올렛 포켓몬 흡 배틀 데이터와 대응합니다!

## 포켓몬 도감

SV 시즌 10 배틀 데이터 업데이트 시작: 2023-09-20 PM 8:06 (GMT+0900)

도감 상세 검색 영칭/일칭 검색

정렬 기준

오늘의 싱글 ○ 오늘의 더블 ○ 시즌을 선택하세요 ○ 원하는 시즌 싱글 - SV ○ 원하는 시즌 더블 - SV ○ 원하는 시즌 싱글 - SWSH ○ 원하는 시즌 더블 - SWSH ○ 전국도감순

포켓몬의 이름으로 검색하세요

도감 리스트 PNG로 저장

포켓몬 이미지	이름	#	등급	강체	속성	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	
	망나뇽	#1(0)		드래곤 비행	정신력 멀티스케일	91	134	95	100	100	80	600
	날개치는머리	#2(0)		고스트 페어리	고대활성	55	55	55	135	135	135	570
	타부자고	#3(0)		강철 고스트	황금몸	87	60	95	133	91	84	550
	파오젠	#4(0)		악	재앙의검	80	120	80	90	65	135	530

# 마무리: 그래서 WEB에 입문하려면 뭐부터 해야 할까요?

- HTML, CSS, Javascript부터 배우세요.
- 그 다음에 프론트엔드나 백엔드 중 하나를 선택해서 개인 프로젝트를 진행하세요.
- 개인 프로젝트를 하다 보면 프론트와 백 양쪽에 대해서 어느 정도는 지식이 쌓이게 됩니다.
- 그렇다고 현업에서 프론트와 백 양쪽 모두를 다루는 '풀스택 개발자'의 역할을 할 필요는 없습니다.
- 웹 개발자가 되고 싶다면, 다양한 프로젝트를 하면서 한쪽에 특화된 사람이 되도록 노력해 보세요.



이제부터 질문 타임입니다.

들어주셔서 감사합니다!

