

Fastcampus

Web Programming & Frontend Dev SCHOOL

Web, REST API, Socket Programming

Database Advanced

SQL - 데이터 정의언어

데이터를 정의

CREATE - DB 개체 정의

DROP - DB 개체 삭제

ALTER - DB 개체 정의 변경

SQL - 데이터 조작언어

데이터 검색, 등록, 삭제, 갱신

INSERT - 행, 테이블 데이터 삽입

UPDATE - 테이블 업데이트

DELETE - 특정 행 삭제

SELECT - 테이블 검색 결과 집합

SQL - 데이터 제어언어

데이터 액세스 제어

GRANT - 작업 수행권한 부여

REVOKE - 권한 박탈

RDBMS vs MongoDB

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
Column	Field
Primary Key	Primary Key

How to Design Database?

Schema

- Database의 구조와 제약조건에 대한 전반적인 명세 기술
- Database의 Blueprint
- 외부(서브)스키마, 개념스키마, 내부스키마로 구성

외부(서브) 스키마

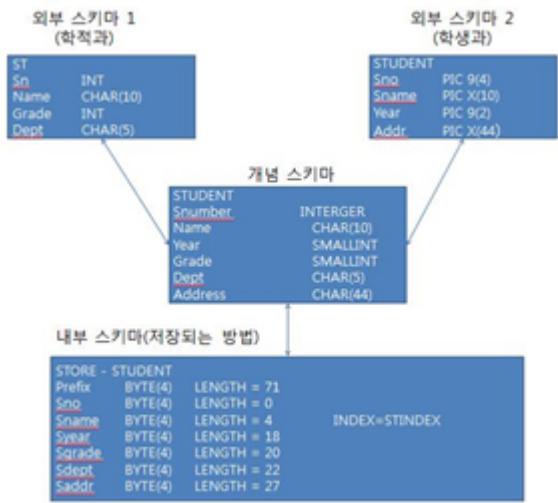
- 프로그램 사용자가 필요로 하는 데이터베이스의 논리적인 구조를 정의
- 겉으로 보이는 테이블의 형태 view

개념 스키마

- 조직 전체의 관점에서의 구조와 관계를 정의
- 외부 스키마의 합과 그 사이의 데이터의 관계 등등
- 일반적인 스키마의 정의
- 테이블의 구성, 속성, 릴레이션 등등

내부 스키마

- 저장장치의 입장에서 데이터베이스가 저장되는 방법을 기술
- DBMS가 관리하는 부분, 이 데이터는 디스크에 어떻게 저장하고, 메모리를 어떻게 점유하고 등등



Design Database

Primary Key	Attribute1	Attr2	Attr3	Attr4
Tuple1				
Tuple2				
Tuple3				
Tuple4				

Web Programming

웹 개발 패턴의 변화

- 1991 ~ 1999: Sir Timothy John "Tim" Berners-Lee가 하이퍼텍스트 기반의 프로젝트를 제안한 이후 정적인 컨텐츠들을 중심으로 한 웹 기술이 발달
- 1999 ~ 2009: Linux, Apache, Mysql, Php 중심의 동적인 서버, 정적인 클라이언트 모델이 지속됨
- 2010 ~ 현재: javaScript!! (Dynamic Web Client)

웹 개발 패턴의 변화

```
<html>
<head></head>
<body>
<h1>Static Header</h1>
<div>Static Contents</div>
</body>
</html>
```

- 1991 ~ 1999: Sir Timothy John "Tim" Berners-Lee가 하이퍼텍스트 기반의 프로젝트를 제안한 이후 정적인 컨텐츠들을 중심으로 한 웹 기술이 발달

웹 개발 패턴의 변화

```
<html>
<head></head>
<body>
<h1>{% Dynamic Header %}</h1>
<div>{% Dynamic Contents %}</div>
</body>
</html>
```

- 1999 ~ 2009: Linux, Apache, Mysql, Php 중심의 동적인 서버, 정적인 클라이언트 모델이 지속됨

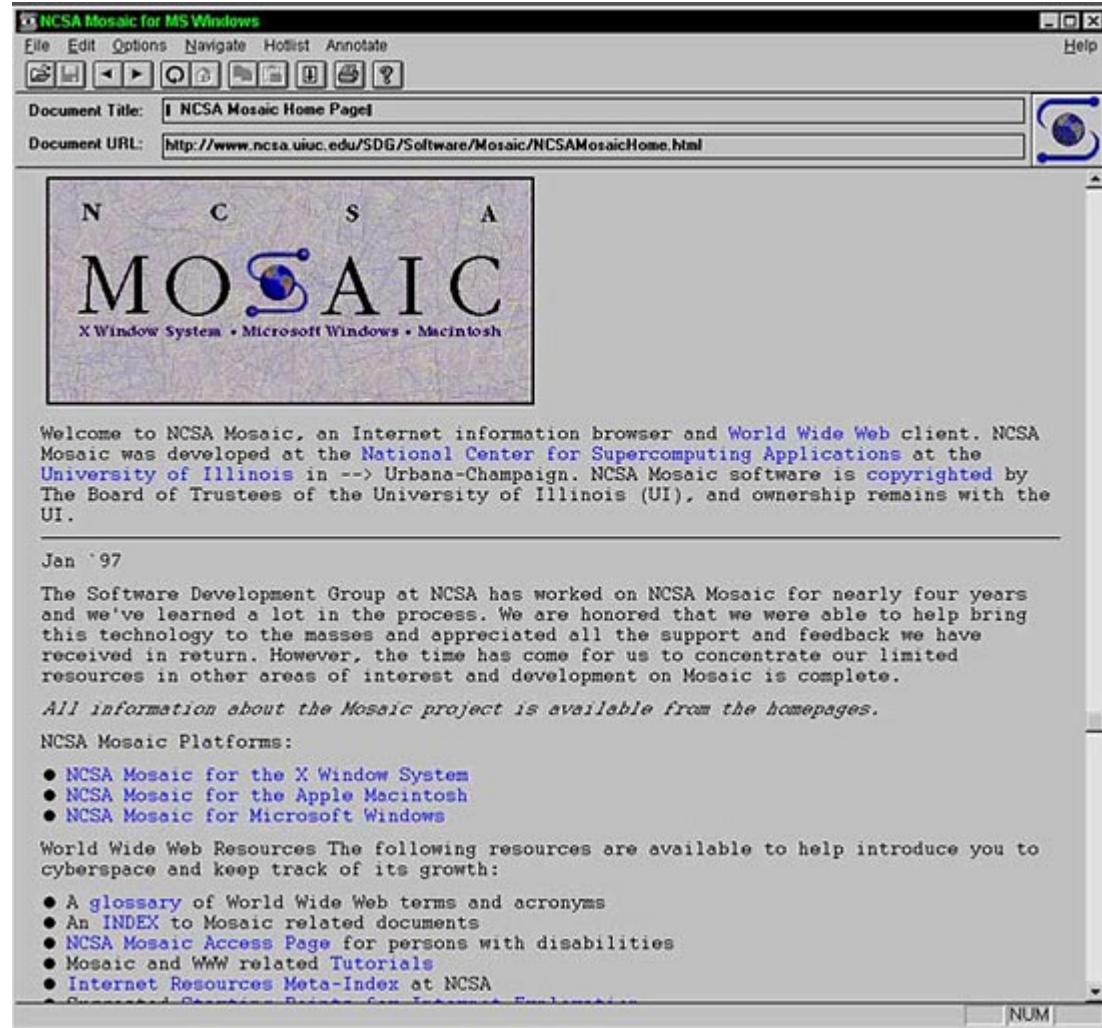
웹 개발 패턴의 변화

```
<html>
<head>
<script src="https://unpkg.com/vue"></script>
</head>
<body>
<h1>{{ header }}</h1>
<div id="app">
  {{ message }}
</div>
<script>
var app = new Vue({
  el: '#app',
  data: {
    message: '안녕하세요 Vue!'
  }
})
</script>
</body>
</html>
```

- 2010 ~ 현재: JavaScript!! (Dynamic Web Client)

Web Browser

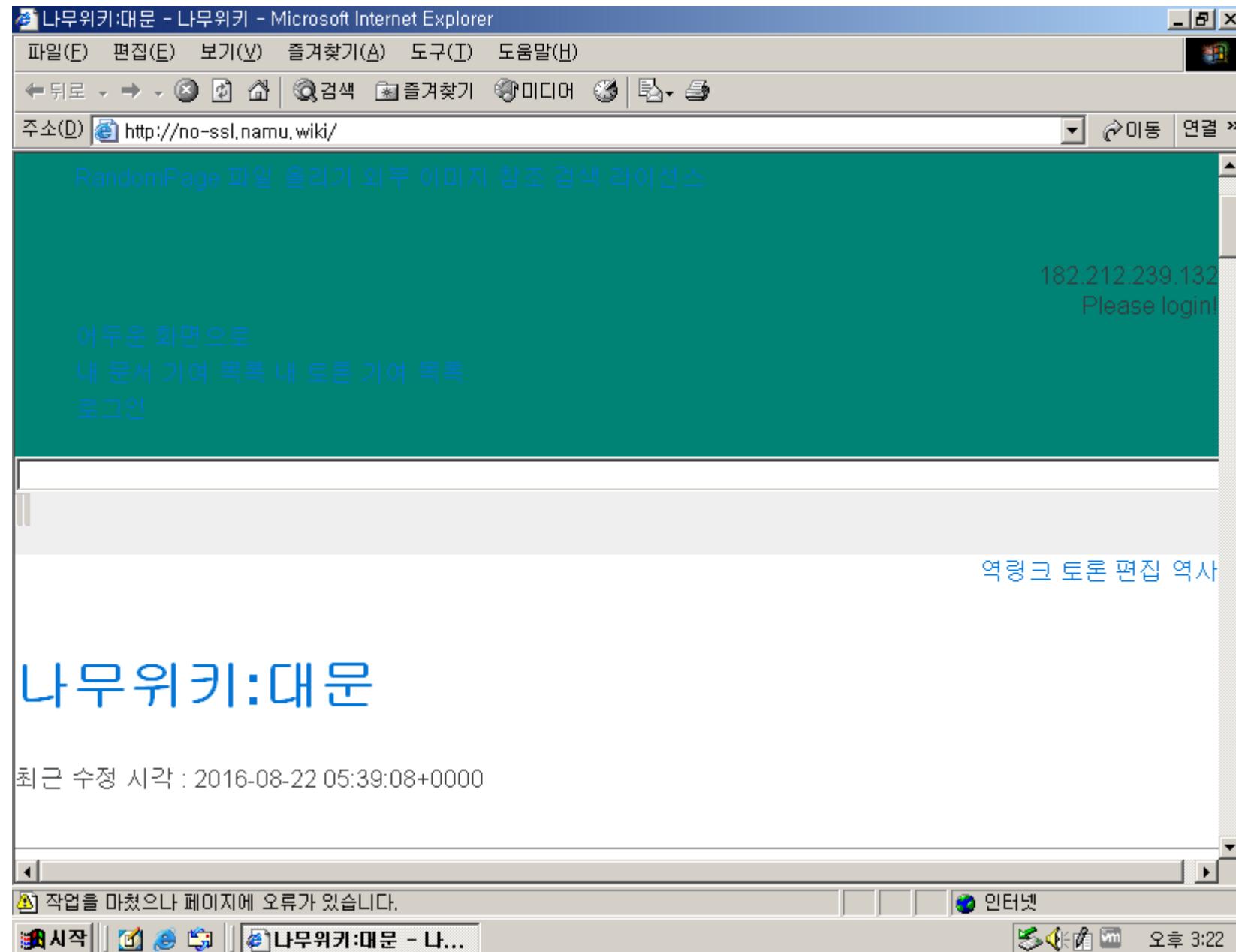
Mosaic(1993)



Netscape(1994)



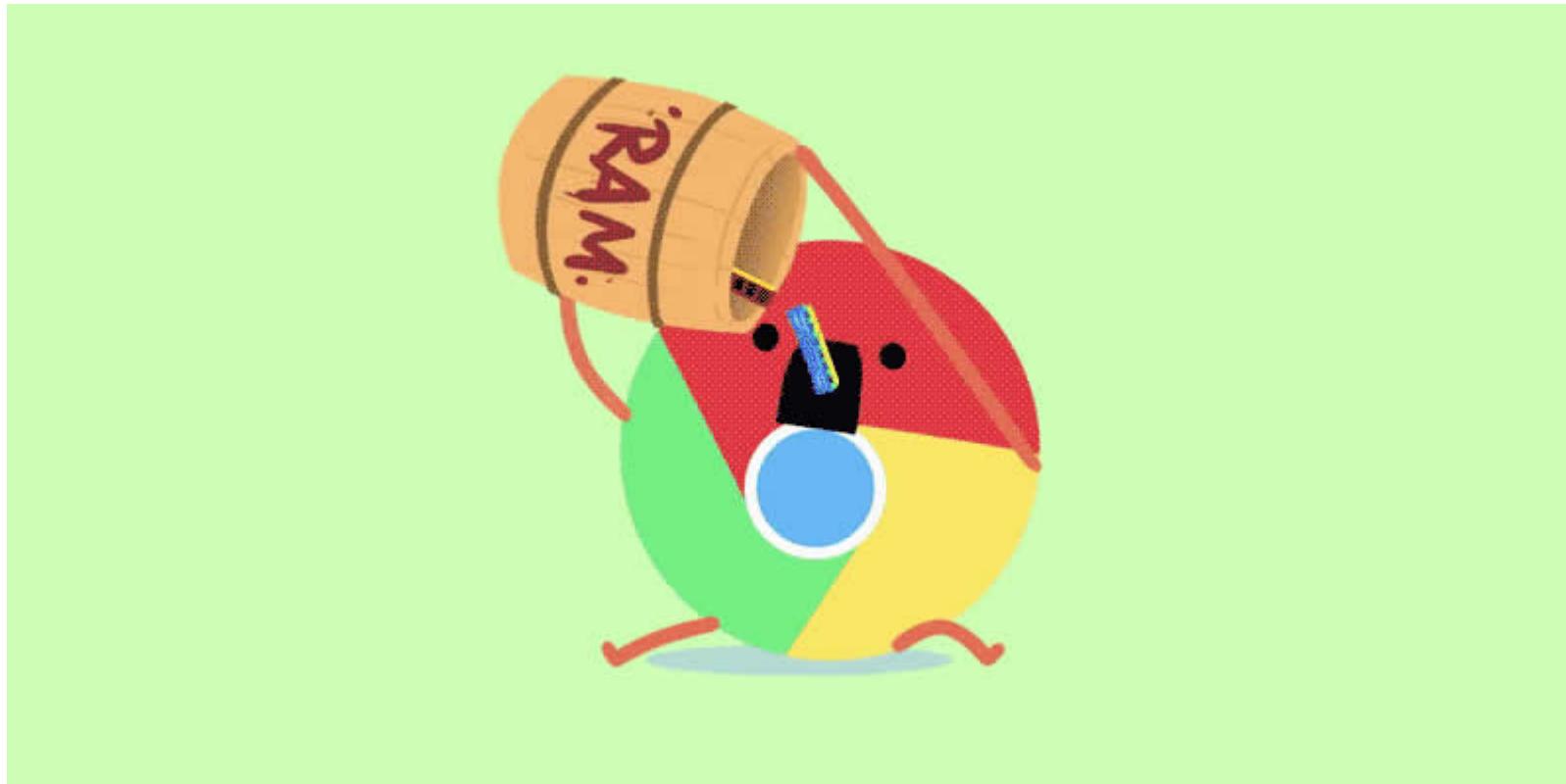
Internet Explorer(1995)



FireFox(2004)

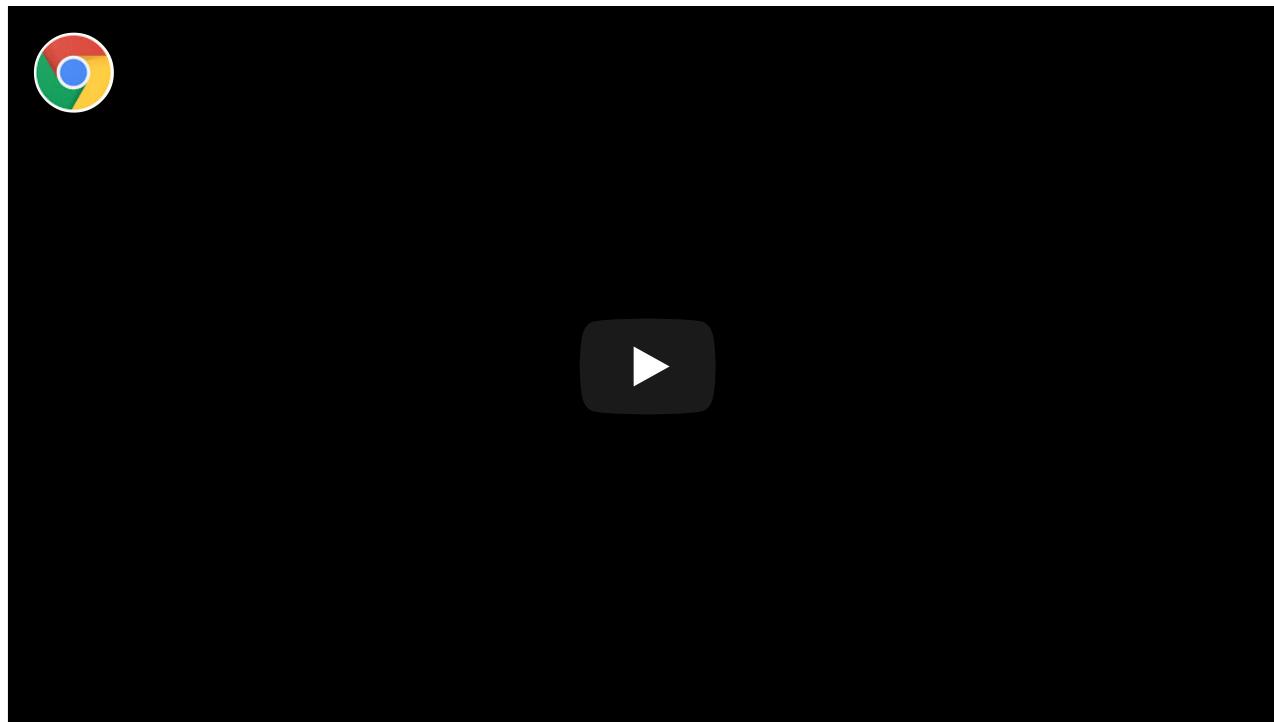


Chrome(2008)

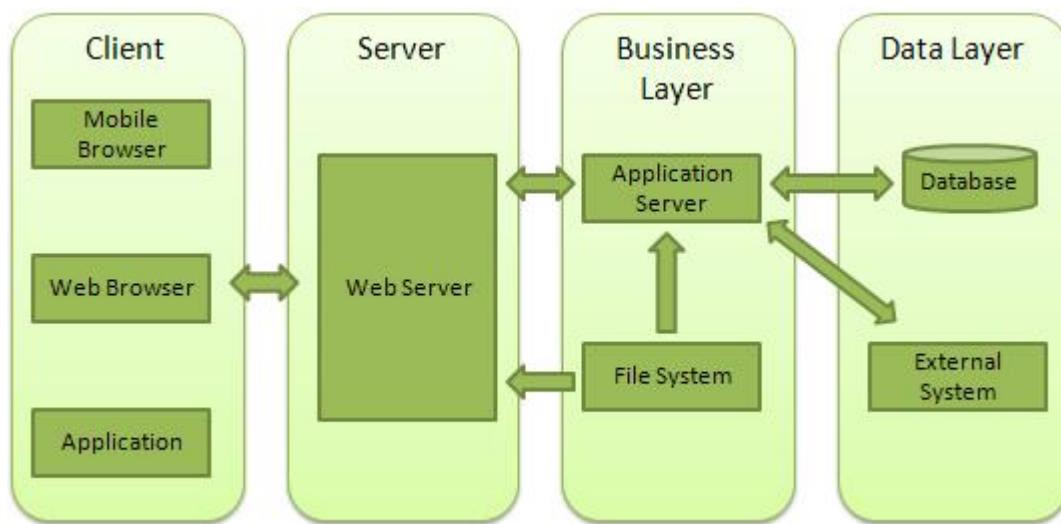


PWA in Chrome Dev Summit 2017

[schedule](#)



Web architecture



웹 개발의 현재

javaScript

Client-side

- HTML/CSS, javaScript
- jQuery, AJAX
- Front-end Web Framework
 - AngularJS
 - React.js
 - Vue.js
- CSS Framework
 - Bootstrap
 - Foundation

Server-side

- Depends on Language
 - PHP: Laravel
 - javaScript: Node.js(Express.js)
 - Java: Spring
 - C++, C#: [ASP.net](#)
 - Python: Django, Flask
 - Golang: itself
 - Ruby: Ruby on Rails

Database

- RDBMS
 - MySQL
 - PostgreSQL
 - MariaDB
- noSQL
 - MongoDB
 - CouchDB
 - Redis

etc

- celery (for Distributed Task Queue)
- github, Bitbucket, gitlab (for SCM)
- travis CI or jenkins (for Continuous Integration)
- slack, trello

URI, URL, URN

URI

- Uniform Resource Information
- `https://www.example.com/post/how-to-make-url`

URL

- Uniform Resource Locator
- `https://www.example.com/post/`

URN

- Uniform Resource Name
- `www.example.com/post/how-to-make-url`

API

API

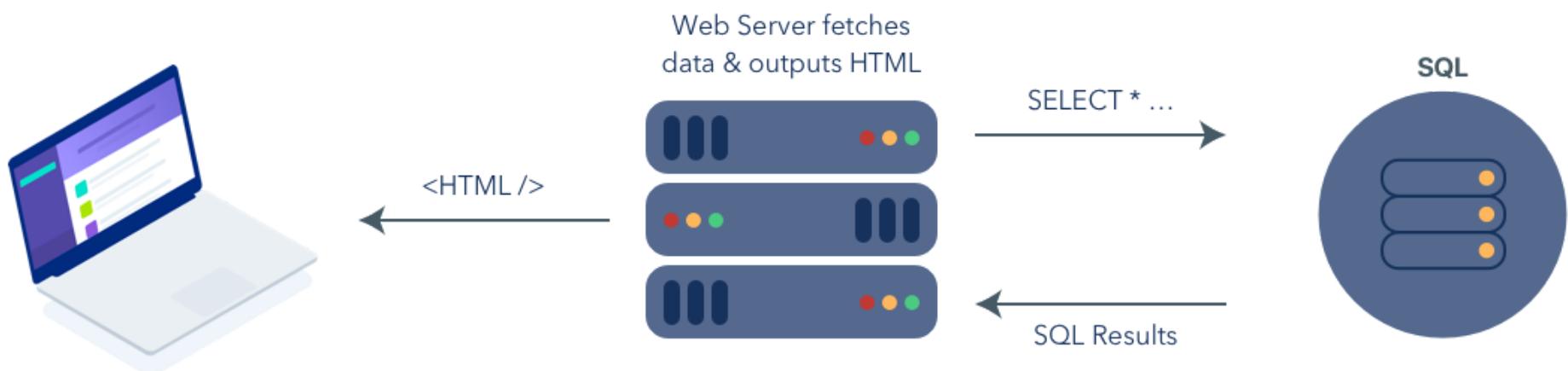
Application Program Interface

- 응용프로그램에서 사용할 수 있도록 운영체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스
- [Windows API](#)
- [python/C API](#)

Web API

웹서버 혹은 웹브라우저를 위한 API

Before REST



Before REST

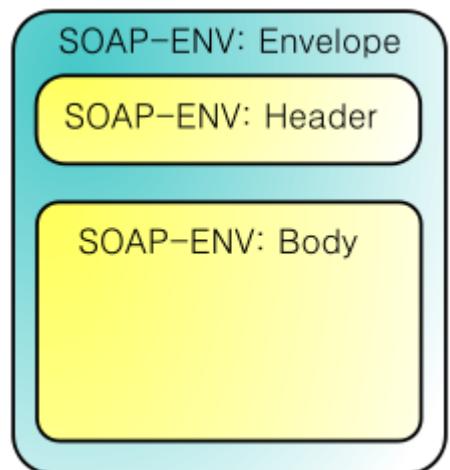


SOAP

Simple Object Access Protocol

XML 파일 포맷을 활용해 데이터를 주고 받기 위한 시도

xml architecture



xml

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
<food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
        Two of our famous Belgian Waffles with plenty of real maple s
    </description>
    <calories>650</calories>
</food>
<food>
</food>
</breakfast_menu>
```

REST API

RE presentational S tate T ransfer

A pplication P rogramming I nterface

Resource - URI

Verb - HTTP method

Representations - 표현

So, REST is

HTTP URI + HTTP method

[Yahoo Finance](#)

[json api](#)

Roy Fielding



- 2000년 UC Irvine의 박사 학위 논문 "Architectural Styles and the Design of Network-based Software Architectures" 발표

Characteristics of REST

- 범용성(HTTP가 가능하면 OK)
- 리소스 중심 API 명세(URI를 읽는 것으로 이해 가능)
- Stateless(클라이언트의 상태를 신경쓰지 않음)

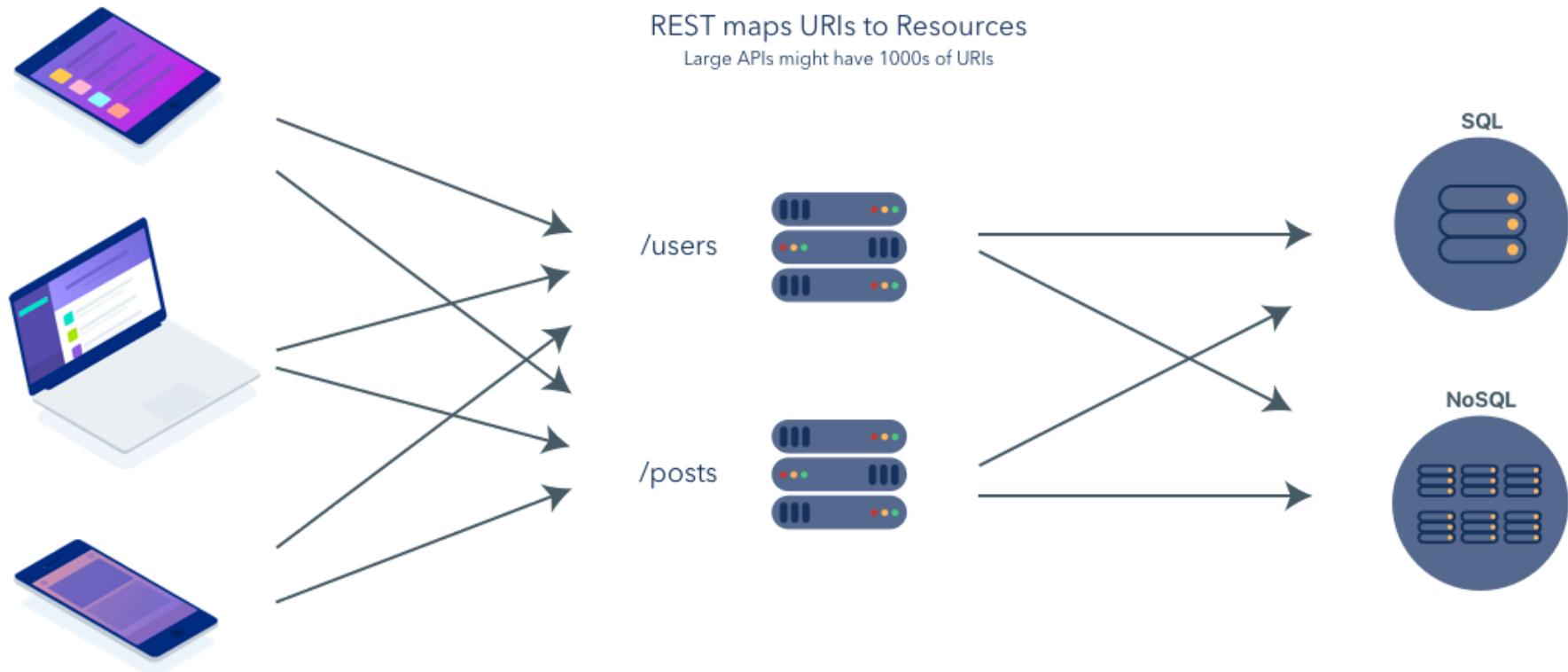
pros and cons of REST

pros:

- 스펙없이 기존의 HTTP를 이용해 요청을 처리할 수 있다.

cons:

- 사용할 수 있는 메소드가 4개다
- 표준이 없다



CRUD

Create

Read

Update

Delete

HTTP Response code

200, 201 - Success

400, 404 - Not found

403 - Forbidden

500 - Server error

HTTP	REST	Status Code
GET	read	200 OK
POST	create	201 CREATED
PUT	update	200 OK
DELETE	delete	200 OK

404 Not Found

500 Internal Server Error

```
app.get('/users', (req, res)=>{
    res.json(db.users);
});

app.get('/users/:id', (req, res)=>{
    let user = db.users.find(user => user.id == req.params.i
    if (user) {
        res.json(user);
    } else {
        res.status(404).end();
    }
});

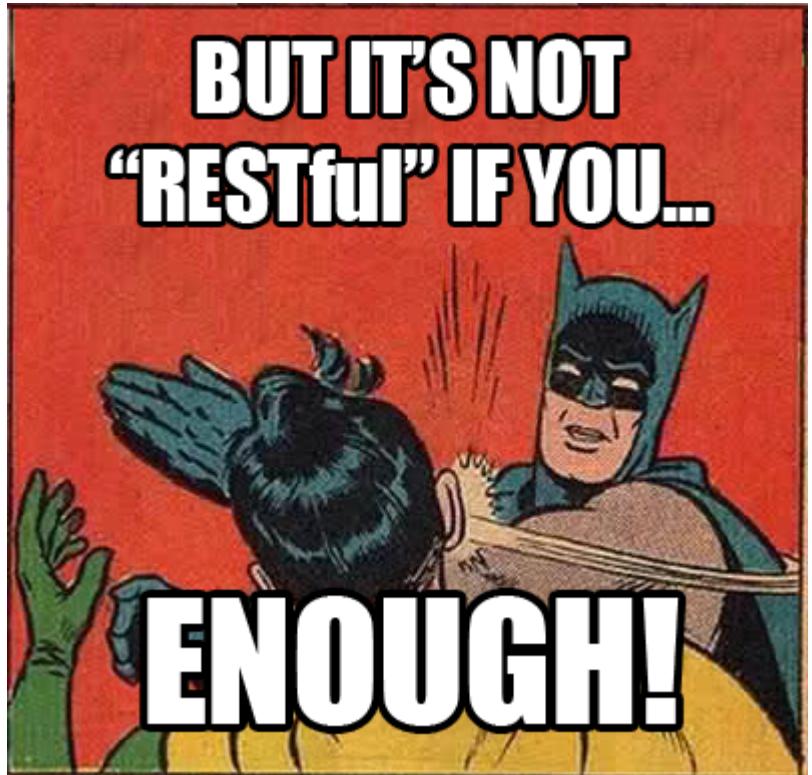
app.put('/users', ...);

app.post('/users', ...);

app.delete('/users/:id', ...);
```

REST API 설계시 주의할 점

- 버전관리 <https://api.foo.com/v1/bar>
- 명사형 사용 <https://foo.com/showid/> --> <https://foo.com/user/>
- 반응형 <https://foo.com/m/user/>, <https://m.foo.com/user/> (x)
- 언어코드 <https://foo.com/kr/>, <https://kr.foo.com/> (x)
- 응답상태 코드 (200, 400, 500)



RESTful API Documentation

- 기존의 정의된 메소드를 사용하세요(200, 403, ..)
- 사용자가 정의하는 방식으로 작성하세요
- URI를 너무 강조하지 마세요
- 문서화도구(swagger 등)을 너무 믿지 마세요

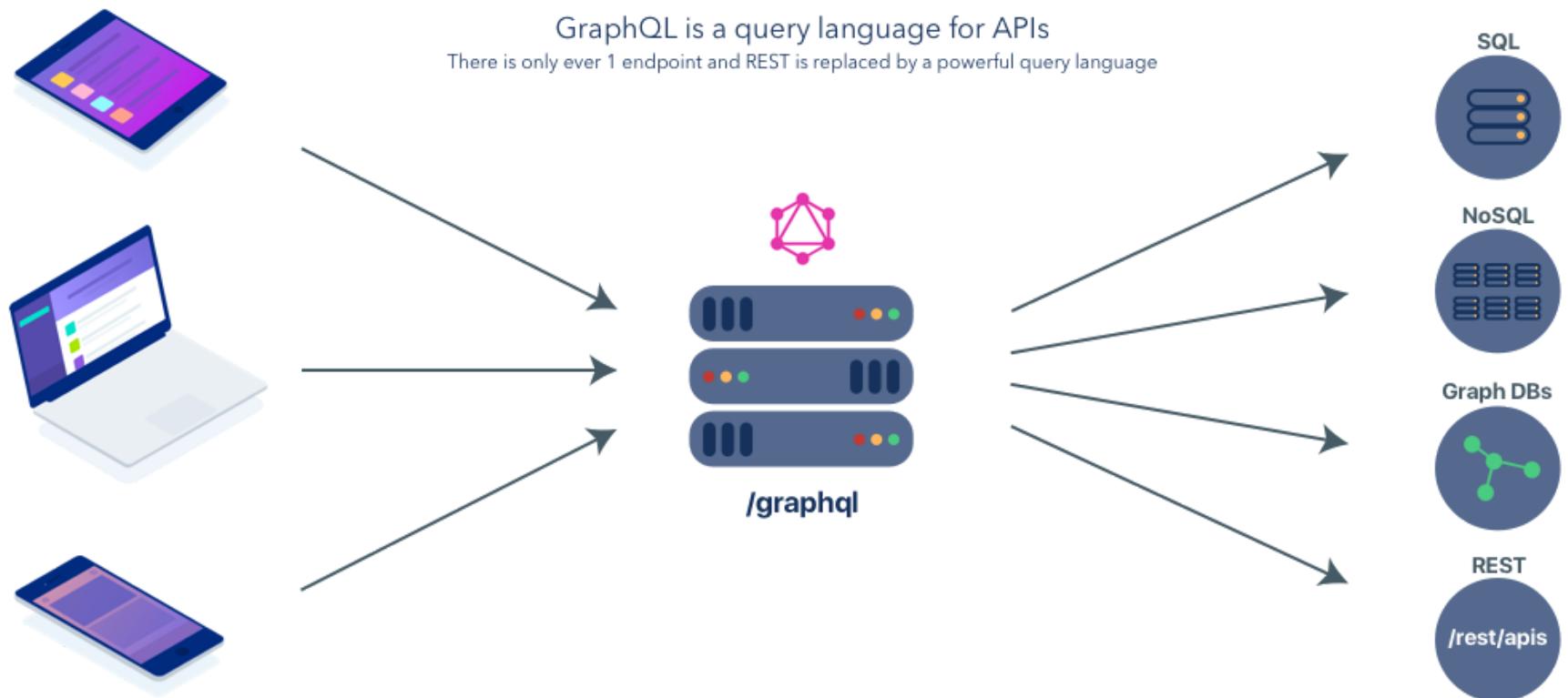
ex) <https://www.vmware.com/support/developer/vas/rest-api-1.1.0.RELEASE/index.html#4>

After REST API

GraphQL

- Open-sourced by Facebook
- Alternative to REST for building APIs
- create strong contract between Client and Server

GraphQL



Querying with GraphQL

```
query MoviesAndActors {  
  movies {  
    title  
    image  
    actors {  
      image  
      name  
    }  
  }  
}
```

schema of GraphQL

```
schema {
  query: Query
}

type Query {
  movies: [Movie]
  actors: [Actor]
  movie(id: Int!): Movie
  actor(id: Int!): Actor
  searchMovies(term: String): [Movie]
  searchActors(term: String): [Actor]
}
```

```
type Movie {
    id: Int
    title: String
    image: String
    release_year: Int
    tags: [String]
    rating: Float
    actors: [Actor]
}

type Actor {
    id: Int
    name: String
    image: String
    dob: String
    num_credits: Int
    movies: [Movie]
}
```

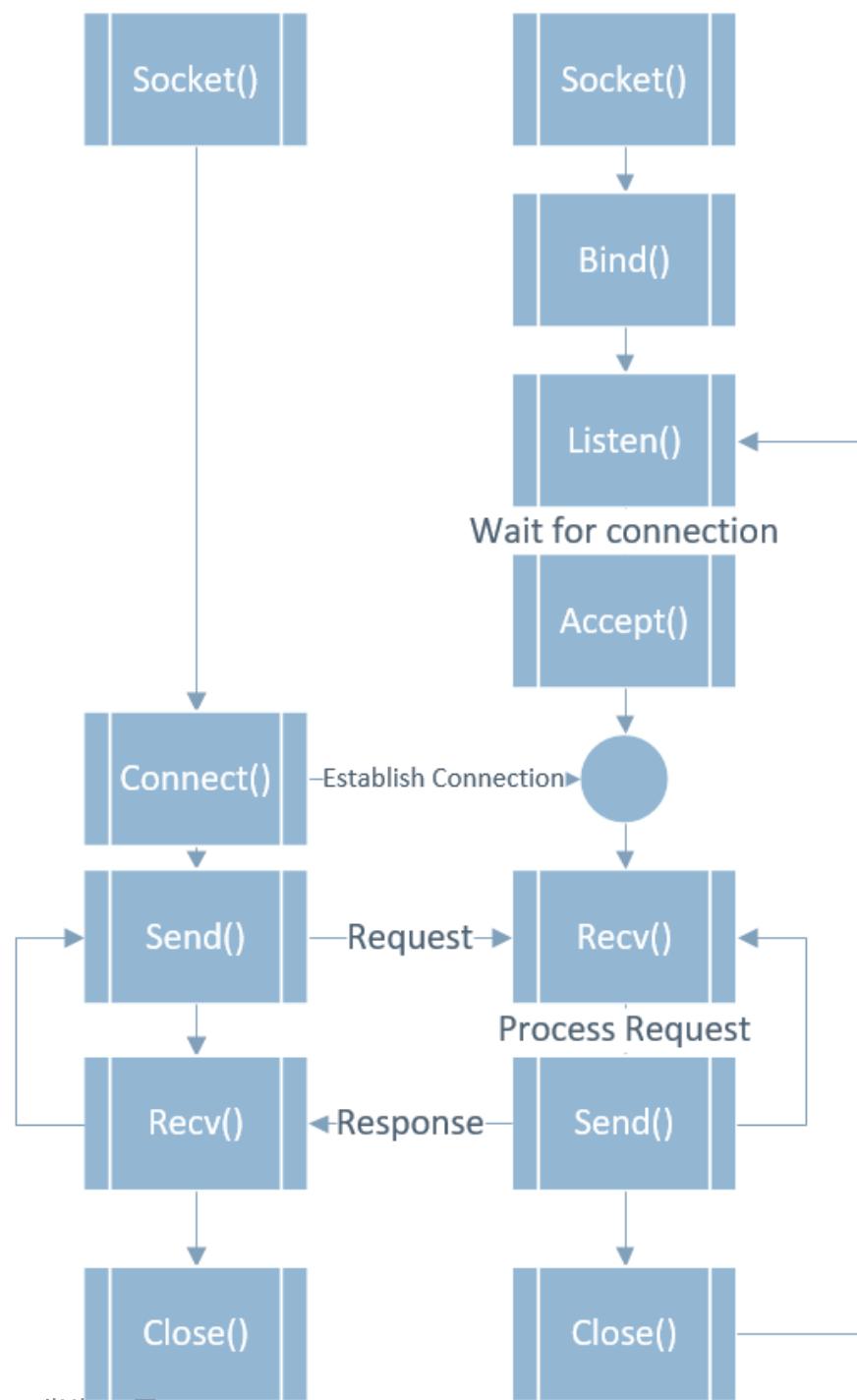
Socket

Socket

- Virtual End Point where entities can perform inter-process communication.

So, Socket is ..

떨어져 있는 두 컴퓨터를 연결해주는 과정



parameters

socket family(family) - AF_INET, AF_UNIX, AF_BLUETOOTH

socket type(type) - SOCK_STREAM, SOCK_DGRAM

socket 통신도 통신이므로.. 통신보안!!

TLS(Transport Layer Security) - 프로토콜에 의한 암호화

SSL(Secure Socket Layer) - 포트에 의한 암호화