HANSUNG
UNIVERSITY

Express.js

Web Programming

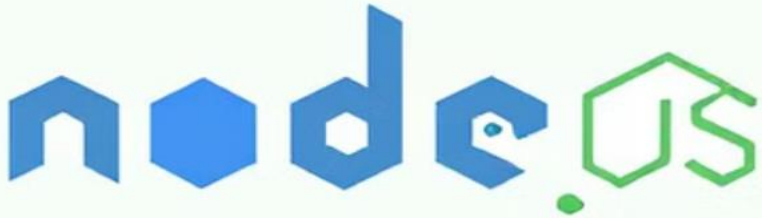
Sung-Dong Kim,
School of Computer Engineering,
Hansung University

학습 내용

HANSUNG UNIVERSITY

- Introduction
- 시작하기

Introduction



Express.js 소개

- Nodejs 기반 웹 애플리케이션 프레임워크
- 서버측 애플리케이션 개발을 단순화, 효율화
- 백앤드 개발에 널리 사용됨
- RESTful API, web application 구축에 적합

주요 특징

- 간결한 문법
- 미들웨어 아키텍처
  - 요청, 응답 처리
  - 요청/응답 흐름 조작
  - 로깅, 인증, 오류 처리 등의 기능 추가 가능
- 라우팅 기능: URL과 메소드 (GET, POST, PUT, DELTE)에 따라 요청을 적절한 핸들러로 분배
- Node.js와 통합
- 폭 넓은 커뮤니티와 생태계

활용 사례

HANSUNG UNIVERSITY

- RESTful API 구축
- 실시간 애플리케이션 (채팅, 알림, ...)
- 간단한 웹사이트 및 백엔드 서버
- MEAN (MongoDB, Express, Angular, Node.js) 스택의 핵심 요소

시작하기

시작하기

- Node.js 설치
- 프로젝트 초기화
  - 프로젝트 디렉토리 생성
  - npm init: 초기화 -> package.json 생성

```
mkdir my-express-app
cd my-express-app
npm init -y
```

- Express.js 설치

```
npm install express --save
```

웹 서버 만들기

HANSUNG UNIVERSITY

- app1.js

```
const express = require('express');

const app = express();
app.set('port', 3000);
app.use((req, res, next) => {
        res.status(200).send('<h1>Welcome !</h1>');
});
app.listen(app.get('port'), () => {
        console.log('Server listening on port 3000');
});
```

HANSUNG UNIVERSITY

- app 객체 = express server object

method	설명
set(name, value)	서버 속성 설정
get(name)	속성 값 확인
use([path,] function [, function...])	미들웨어 함수 사용
get([path] function)	특정 패스로 요청된 정보를 처리

미들웨어로 클라이언트에 응답 보내기

HANSUNG UNIVERSITY

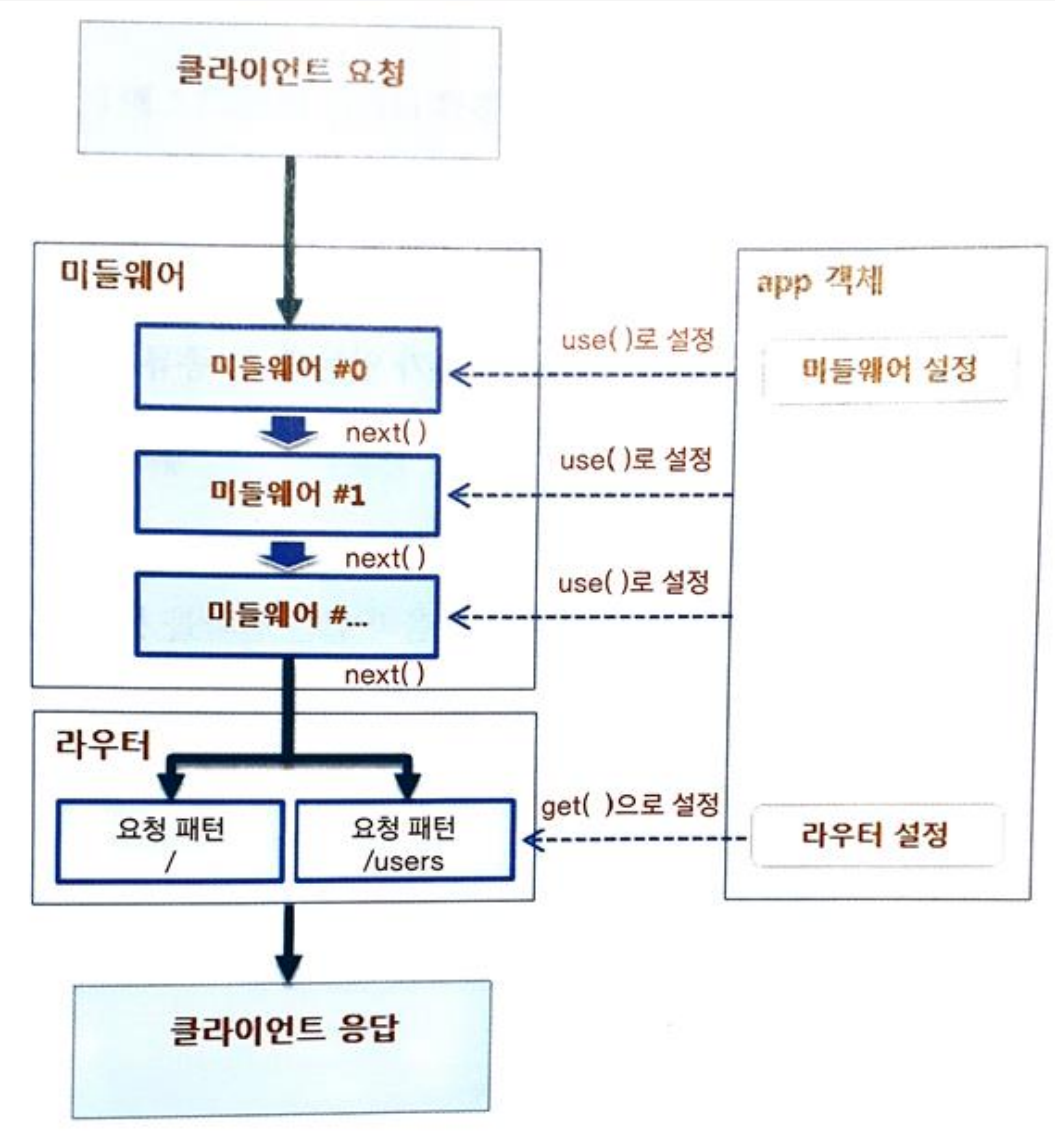
미들웨어 (middleware)

- 웹 요청과 응답사이에 필요한 처리를 수행하는 독립된 함수
- 필요한 기능을 순차적으로 실행
- use() 메소드로 미들웨어 설정 → 요청을 처리하여 응답을 보냄

라우터 (router)

- 클라이언트의 요청 패스를 보고 처리할 수 있는 곳으로 기능을 전달하는 역할
- /users 패스로 요청 → 응답 처리 함수 호출
- 응답 처리 함수: get() method로 미리 등록

미들웨어로 클라이언트에 응답 보내기



미들웨어로 클라이언트에 응답 보내기

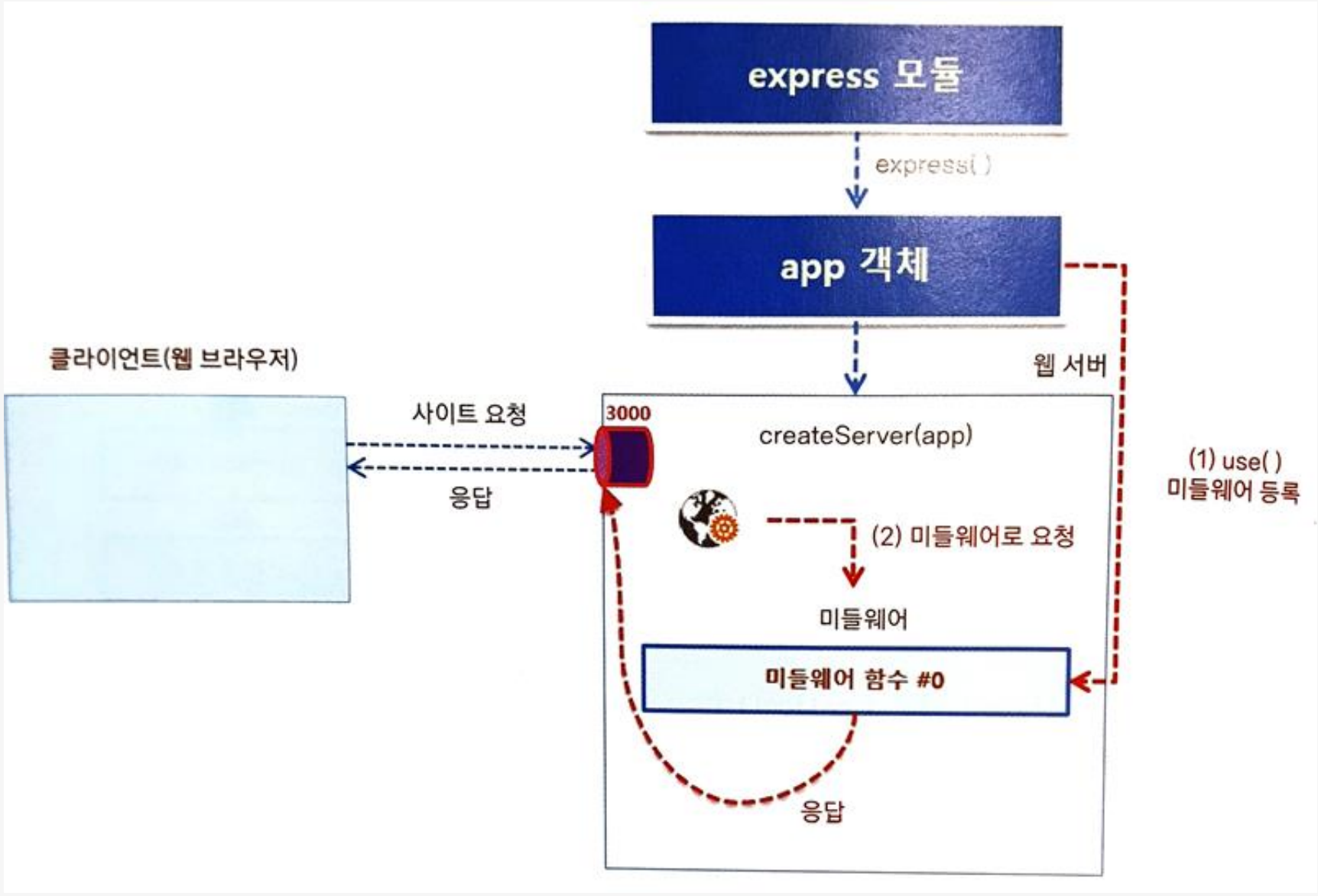
HANSUNG UNIVERSITY

- app2.js

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

app.use(function(req, res, next) {
  res.writeHead(200, {'Content-Type':'text/html;charset=utf8'});
  res.end('<h1>Express Server에서 응답한 결과</h1>');
});
app.listen(3000, () => {
        console.log('Server listening on port 3000');
});
```

미들웨어로 클라이언트에 응답 보내기



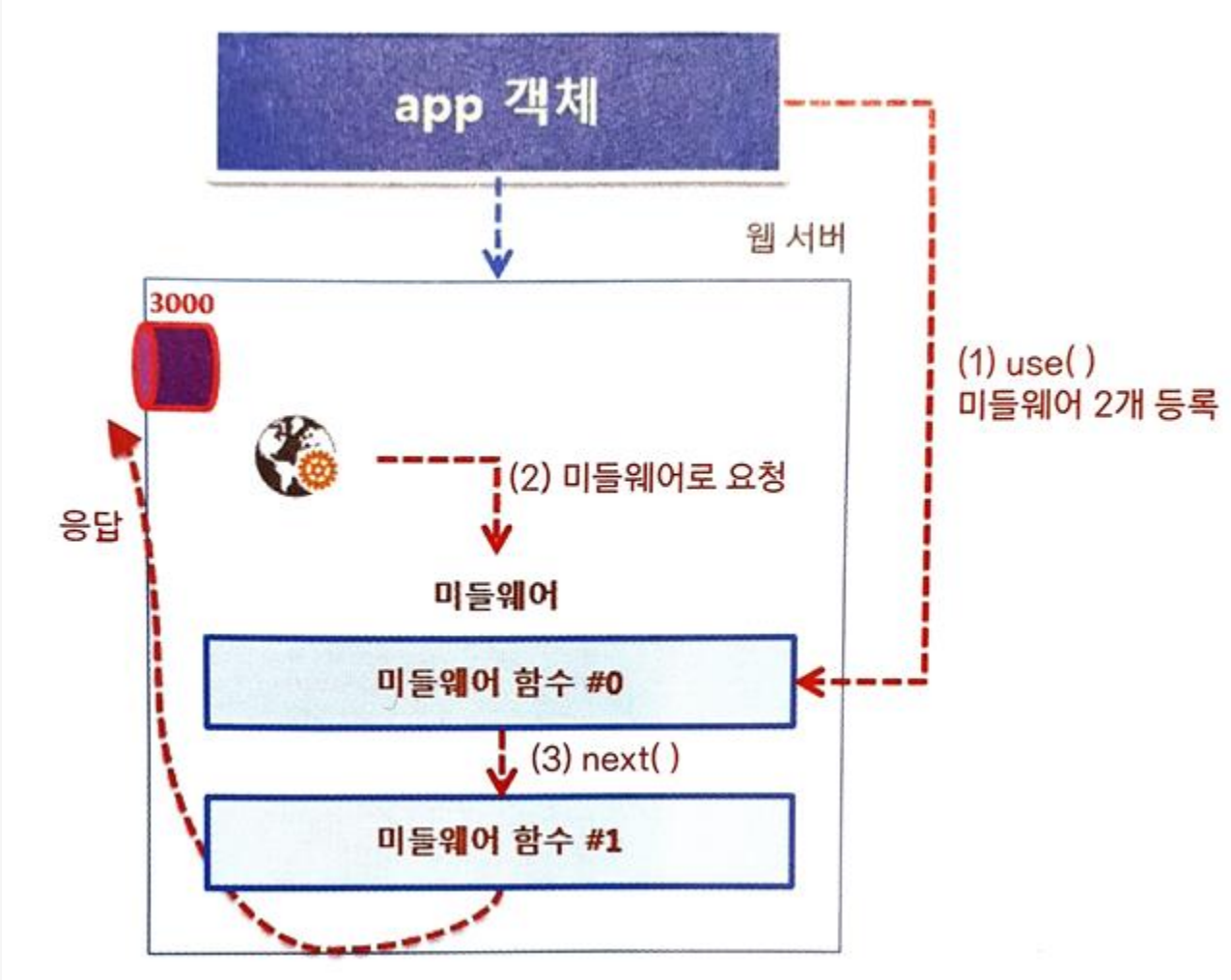
여러 개의 미들웨어를 등록하여 사용하는 방법

HANSUNG UNIVERSITY

- app3.js: next() 호출

```
app.use(function(req, res, next) {
  console.log('First Middleware ...');
  req.user = 'KSD';
  next();
});
app.use(function(req, res, next) {
  console.log('Second Middleware ...');
  res.status(200).send(`<h1>${req.user} responds at Express Server</h1>`);
});
```

여러 개의 미들웨어를 등록하여 사용하는 방법



미들웨어로 클라이언트에 응답 보내기

HANSUNG UNIVERSITY

- app4.js

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

app.get('/', (req, res) => {
        res.status(200).send('<h1>Welcome !!</h1>');
});
app.get('/about', (req, res) => {
        res.status(200).send('<h2>Here is KSD home. </h2>');
});
app.listen(3000);
```

요청 객체와 응답 객체

HANSUNG UNIVERSITY

- http 객체 + 추가

method	설명
send([body])	응답 데이터 보내기 (html 문자열, Buffer 객체, JSON 객체, JSON 배열)
status(code)	HTTP 상태 코드 반환. end()나 send() 이용
sendStatus(statusCode)	HTTP 상태 코드 반환. 상태 메시지와 함께 전송됨
redirect([status,] path)	웹 페이지 경로를 강제로 이동시킴
render(view [, locals][, callback])	뷰 엔진을 사용해 문서를 만든 후 전송

요청 객체와 응답 객체

HANSUNG UNIVERSITY

- app5.js: send() method

```
app.use(function(req, res, next) {
  console.log('첫 번째 미들웨에어서 요청 처리');
  res.send({name: 'KSD', age: 30});
});
```

- app6.js: redirect() method

```
app.use(function(req, res, next) {
  console.log('첫 번째 미들웨에어서 요청 처리');
  res.redirect('http://hansung.ac.kr');
});
```

요청 객체에 추가한 헤더와 파라미터

HANSUNG UNIVERSITY

- 추가 헤더와 파라미터

추가한 정보	설명
query	클라이언트에서 GET 방식으로 전송한 요청 파라미터 확인 예: req.query.name
body	클라이언트에서 POST 방식으로 전송한 요청 파라미터 확인 (body-parser 같은 외장 모듈 필요) 예: req.body.name
header(name)	헤더 확인

요청 객체에 추가한 헤더와 파라미터

HANSUNG UNIVERSITY

- GET 방식 요청: app7.js

```
app.use(function(req, res, next) {
  const userAgent = req.header('User-Agent');
  const paramName = req.query.name;

  res.writeHead(200, {'Content-Type' : 'text/html;charset=utf-8'});
  res.write('<h1>Express 서버에서 응답한 결과</h1>');
  res.write(`<div><p>User-Agent: ${userAgent} </p></div>`);
  res.write(`<div><p>Param name: ${paramName} </p></div>`);
  res.end();
});
```

학습 정리

학습 정리

HANSUNG UNIVERSITY

- Introduction: what, 특징, 활용 사례
- 시작하기
  - 프로젝트 초기화
  - 웹 서버 만들기
  - 미들웨어 사용
  - 요청, 응답 객체