스타트업 개발자와 함께 공부하는

Node.js

04. 익스프레스

강의 내용은 강사가 별도로 명시하지 않는 한 비공개로 간주합니다. 녹음이나 사진 촬영를 허락하지 않으며 콘텐츠를 블로그, SNS 등에 게시하거나 공개적으로 공유하지 마세요.

콘텐츠 공유 가능 여부에 대해 궁금한 점이 있는 경우 강사에게 문의하시기 바랍니다.

목차

1. 라우팅 과 파라미터 다루기
2. 익스프레스

라우팅

라우팅

URL과 URI

http://www.naver.com/path/to/file.html?key1=value1&key2=value2

라우팅

HTTP 서버에 라우팅 추가

```
1  const http = require('http');
2  const url = require('url');
3
4  http.createServer((req, res) => {
5      const path = url.parse(req.url, true).pathname;
6      res.setHeader('Content-Type', 'text/html');
7
8      if(path == '/hello'){
9          res.end('<h1>hello</h1>');
10     }else if(path == '/world') {
11         res.end('<h1>world</h1>');
12     }else{
13         res.end('<h1>hi</h1>');
14     }
15
16 }).listen(4500, ()=> console.log('Add Routing'));
```

- 소스코드 파일명:ch04_01.js
- /hello
- /world

라우팅

JSON 응답 라우팅 추가

```
1  const http = require('http');
2  const url = require('url');
3
4  http.createServer((req, res) => {
5      const path = url.parse(req.url, true).pathname;
6      res.setHeader('Content-Type', 'application/json');
7
8      const data = {'name':'hello', 'value':'world'};
9      if(path == '/json'){
10         res.end(JSON.stringify(data));
11     }else{
12         res.end('');
13     }
14
15 }).listen(4500, ()=> console.log('Add Routing 2'));
16
```

- JSON 응답 라우팅 추가
- 소스코드 파일명: ch04_02.js
- 요청 URI : /json
- 응답 : JSON 문자열

라우팅

확인 문제

```
1  const http = require('http');
2  const url = require('url');
3  const fs = require('fs')
4
5  http.createServer((req, res) => {
6      const path = url.parse(req.url, true).pathname;
7      res.setHeader('Content-Type', 'application/json');
8
9      if(path == '/list'){
10         const data = fs.readFileSync('test.json', 'utf-8');
11         const result = JSON.parse(data);
12         res.end(JSON.stringify(result));
13     }else{
14         res.end('');
15     }
16
17 }).listen(4500, ()=> console.log('Add Routing 3'));
18
```

- 소스코드 파일명 : ch04_03.js
- 요청 URI : /list
- 응답: test.json 파일을 읽어서 내용을 JSON 포멧 으로 응답

라우팅

라우팅 리팩토링

```
1  const http = require('http');
2  const url = require('url');
3  const fs = require('fs')
4
5  http.createServer((req, res) => {
6      const path = url.parse(req.url, true).pathname;
7      res.setHeader('Content-Type', 'application/json');
8
9      if(path == '/list'){
10         list(req, res);
11     }else{
12         res.end('');
13     }
14
15 }).listen(4500, ()=> console.log('Refactoring Routing 1'));
16
17
18 const list = (req, res) => {
19     const data = fs.readFileSync('test.json', 'utf-8');
20     const result = JSON.parse(data);
21     res.end(JSON.stringify(result));
22 }
```

- 소스코드 파일명 : ch04_04.js
- URI 요청 별로 별도의 함수로 분리

라우팅

라우팅 일원화

```
1  const http = require('http');
2  const url = require('url');
3  const fs = require('fs');
4
5  http.createServer((req, res) => {
6      const path = url.parse(req.url, true).pathname;
7      res.setHeader('Content-Type', 'application/json');
8
9      if(path in urlMap){
10         urlMap[path](req,res)
11     }
12
13 }).listen(4500, ()=> console.log('Refactoring Routing 2'));
14
15 const list = (req, res) => {
16     const data = fs.readFileSync('test.json', 'utf-8');
17     const result = JSON.parse(data);
18     res.end(JSON.stringify(result));
19 }
20
21 const urlMap = {
22     '/' : (req, res) => res.end('HOME'),
23     '/list': list
24 }
25
```

- 소스코드 파일명:ch04_05.js
- 맵(Map)을 이용하여 URI 통합관리

라우팅

파라미터 관리

```
1  const http = require('http');
2  const url = require('url');
3  const fs = require('fs')
4
5  http.createServer((req, res) => {
6      const path = url.parse(req.url, true).pathname;
7      const param = url.parse(req.url, true).query;
8
9      res.setHeader('Content-Type', 'application/json');
10
11     if(path == '/list'){
12         const data = fs.readFileSync('test.json', 'utf-8');
13         let result = JSON.parse(data);
14         if(param.title && param.content) {
15             result['result'].push({
16                 'title': param.title, 'content': param.content
17             });
18         }
19
20         res.end(JSON.stringify(result));
21     }else{
22         res.end('');
23     }
24
25 }).listen(4500, ()=> console.log('manage parameter'));
```

- 소스코드 파일명 : ch04_06.js
- 파라미터 값을 응답에 추가

익스프레스

익스프레스

프로젝트 생성

```
npm init
npm i express
```

- [C]-[nodejs]-[project]-[04]

익스프레스

첫 번째 익스프레스 웹 서버

```
1  const express = require('express')
2  const app = express()
3  const port = 3000;
4
5  app.get('/', (req,res) => {
6      res.send(
7          'Hello World'
8      );
9  });
10
11 app.listen(port, () => {
12     console.log(`Example app
13         listening on port ${port}`);
14 });
```

❑ 파일명 : ch04_07.js

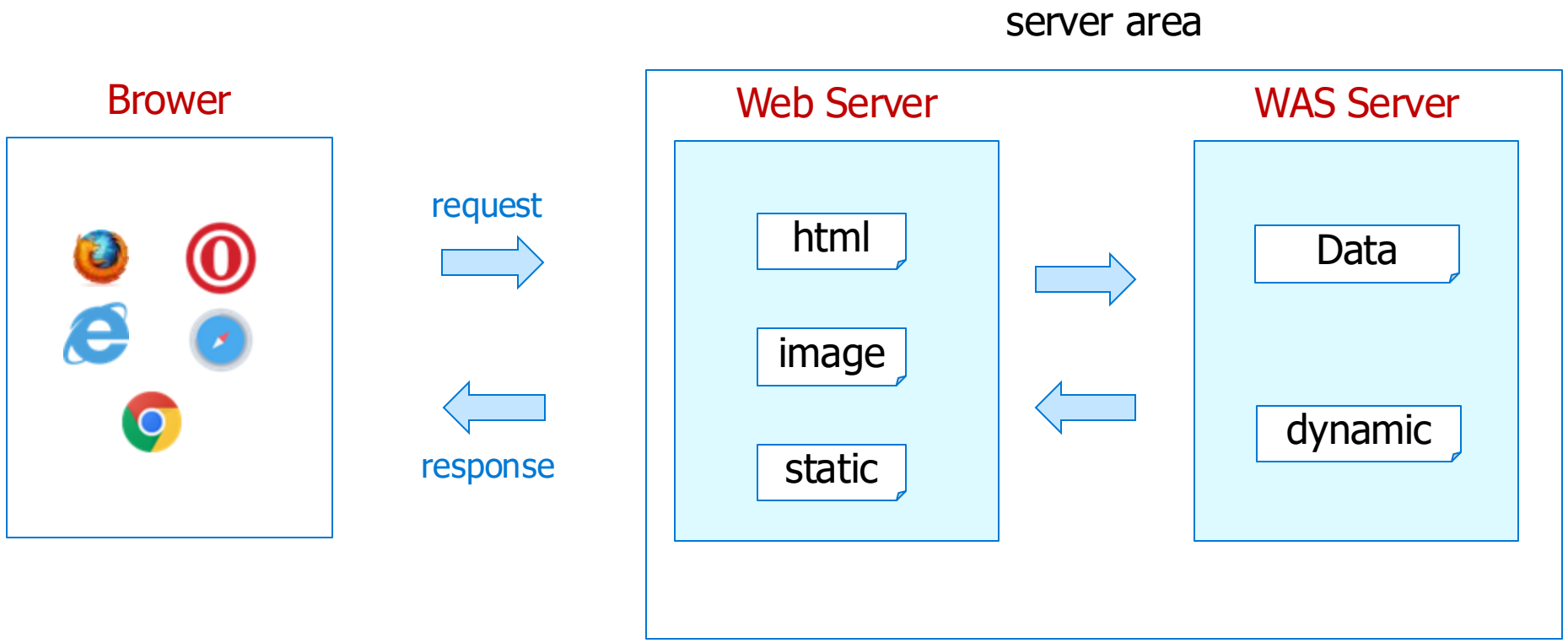
익스프레스

익스프레스 소개

- 노드를 위한 빠르고 개방적인 웹 프레임워크 → 웹(Web)서버와 와스(Was)서버 역할 동시에 수행
- 라우팅
- 정적 파일 서비스
- 템플릿 엔진
- Request와 Response 객체 다루기
- 파일 업로드
- 쿠키 및 세션 지원
- 리다이렉트
- 미들웨어

익스프레스

웹 서버와 와스 서버



익스프레스

JSON 응답 라우팅 추가

```
1  const express = require('express')
2  const fs = require('fs');
3  const app = express()
4  const port = 3000;
5
6  app.get('/list', (req,res) => {
7      list(req, res)
8  });
9
10 const list = (req, res) => {
11     const data = fs.readFileSync(
12         'test.json', 'utf-8');
13     const result = JSON.parse(data);
14     res.json(result);
15 }
16
17 app.listen(port, () => {
18     console.log(`Example app
19         listening on port ${port}`);
20 });
```

- 파일명 : ch04_08.js
- URI : /list
- 응답 : test.json 파일 내용

익스프레스

웹 페이지 출력

```
1  const express = require('express')
2  const fs = require('fs');
3  const app = express()
4  const port = 3000;
5
6  app.get('/home', (req,res) => {
7      home(req, res)
8  });
9
10 const home = (req, res) => {
11     res.send(`<h1>Welcome</h1>
12         <h1>Home</h1>`)
13 }
14
15 app.listen(port, () => {
16     console.log(`Example app
17         listening on port ${port}`);
18 });
19
```

- 파일명 : ch04_09.js
- URI : /home
- 응답 : Welcome Home

익스프레스

<table> 태그 이용 리스트 출력

```
1  const express = require('express')
2  const fs = require('fs');
3  const app = express()
4  const port = 3000;
5
6  app.get('/list', (req,res) => {
7      list(req, res)
8  });
9
10 const list = (req, res) => {
11     const data = fs.readFileSync('test.json',
12         'utf-8');
13     const result = JSON.parse(data);
14     let table = [];
15     let body = [];
16
17     result['result'].forEach(x=> {
18         body.push(`<tr><td>${x['title']}</td>
19             <td>${x['content']}</td></tr>`);
20     });
21
22     table.push(`<table><th><td>Ttitle</td>
23         <td>Content</td></th>${body.join('')}</table>`);
24     res.send(table.join(''));
25 }
26
27 app.listen(port, () => {
28     console.log(`Example app listening on port ${port}`);
29 });
30
```

- 파일명: ch04_10.js
- URI : /list
- 응답 : test.json 내의 배열을 html 포멧으로 출력

백문이 불여일타