

”

Node.js

부제 : 서버 만들기

발표시작 >

발표 순서

”

01

Node.js

간략 소개

02


서버1

http 모듈

03

서버2

express



Node.js

소개

Node.js와 특징

”

01

Node.js ?

Node.js는 Chrome V8
Javascript 엔진으로 빌드된
Javascript 런타임이다.

자바스크립트 실행기

웹 서버 내장

이벤트 기반 event-driven

이벤트가 발생할 때 미리
지정해둔 작업을 수행하는
방식

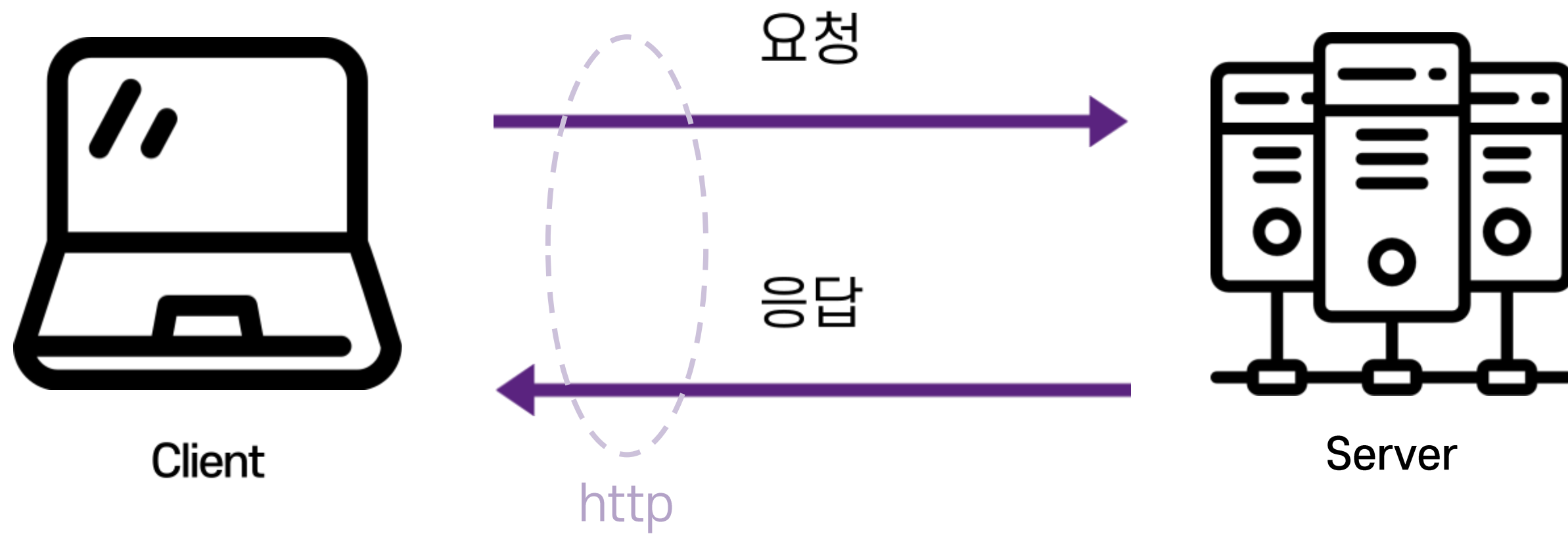
논 블로킹 I/O

Non-Blocking I/O

이전 작업이 완료될 때까지
대기하지 않고 다음 작업을
수행

싱글 스레드

클라이언트와 서버



”



서서1

(http 모듈)

02

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

http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/html; charset=utf-8' });
  res.write('<h1>Hello Node!</h1>');
  res.end('<p>서버 연결 성공!</p>');
})

.listen(8080, () => { // 서버 연결
  console.log('8080번 포트에서 서버 대기 중입니다!');
});
```

8080번 포트에서 서버 대기 중입니다!

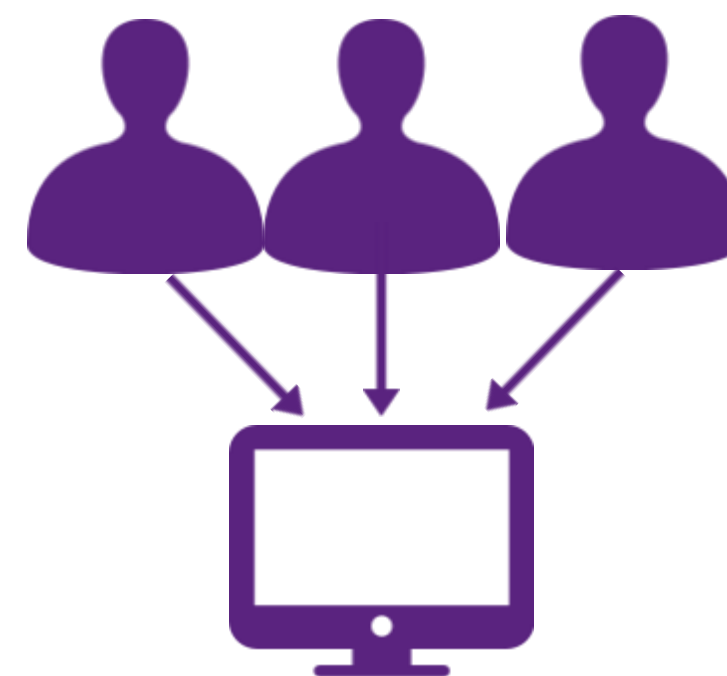
← → ↻ ⓘ localhost:8080

Hello Node!

서버 연결 성공!



 로그인



쿠키 : 카-값



쿠키 삭제



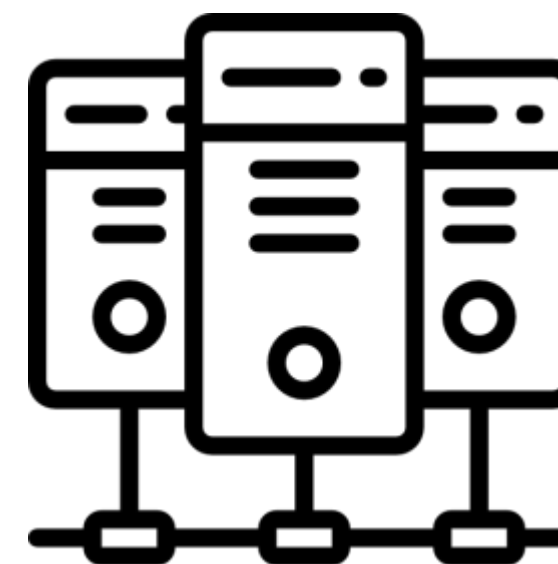
클라이언트

요청

쿠키 & 응답

쿠키 & 요청

응답



서버

```
} else {  
  try {  
    const data = await fs.readFile('./cookie.html');  
    res.writeHead(200, { 'Content-Type': 'text/html; charset=utf-8' });  
    res.end(data);  
  } catch (err) {  
    res.writeHead(500, { 'Content-Type': 'text/plain; charset=utf-8' });  
    res.end(err.message);  
  }  
}
```

프라미스가 처리될 때까지 기다림

← → ↻ ⓘ localhost:8085

Request URL: <http://localhost:8084/login?name=%EB%A1%9C%EA%B7%B8%EC%9D%B8>

Request Method: GET

Status Code: 302 Found

Remote Address: [::1]:8084

Referrer Policy: strict-origin-when-cross-origin

```
http.createServer(async (req, res) => {
  const cookies = parseCookies(req.headers.cookie);
  if (req.url.startsWith('/login')) {
    const { query } = url.parse(req.url);
    const { name } = qs.parse(query);
    const expires = new Date();
    expires.setMinutes(expires.getMinutes() + 5);
    res.writeHead(302, {
      Location: '/',
      'Set-Cookie': `name=${encodeURIComponent(name)}; Expires=${expires.toGMTString()}; HttpOnly; Path=/`,
    });
    res.end();
  }
});
```

Response Cookies

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Priority
name	%EB%A1%9C%EA%B7%B8%EC...	localhost	/	2021-01-25T20:57:48....	89	✓			Medium

```
else if (cookies.name) {  
  res.writeHead(200, { 'Content-Type': 'text/plain; charset=utf-8' });  
  res.end(`${cookies.name}님 안녕하세요`);  
}
```

session 브라우저에 전달하지
않고 중요한 정보는
서버에서 관리.

```
const session = {}; // 세션 객체 생성
```

```
http.createServer(async (req, res) => {  
  const cookies = parseCookies(req.headers.cookie);  
  if (req.url.startsWith('/login')) {  
    const { query } = url.parse(req.url);  
    const { name } = qs.parse(query);  
    const expires = new Date();  
    expires.setMinutes(expires.getMinutes() + 5);  
    const uniqueInt = Date.now();  
    session[uniqueInt] = {  
      name,  
      expires,  
    };  
    res.writeHead(302, {  
      Location: '/',  
      'Set-Cookie': `session=${uniqueInt}; Expires=${expires.toGMTString()}; HttpOnly; Path=/`,  
    });  
    res.end();  
  }  
});
```

Response Cookies

Name	Value
name	%EB%A1%9C%EA%B7%B8%EC...



Response Cookies

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpO...	Secure	SameS...	Priority
session	1611609661961	localhost	/	2021-01-25T21:26:...	78	✓			Medium

```
else if (cookies.session && session[cookies.session].expires > new Date()) {  
  res.writeHead(200, { 'Content-Type': 'text/plain; charset=utf-8' });  
  res.end(`${session[cookies.session].name}님 안녕하세요`);  
}
```

”

서버 2

[익스프레스]

03

express

npm에서 서버를 제작하는 과정에서의 불편함을 해소하고 편의 기능을 추가한 웹 서버 프레임 워크.

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

const app = express();
app.set('port', process.env.PORT || 3000);

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

http 모듈

res.write

res.end


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

const app = express();
app.set('port', process.env.PORT || 3000);

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

app.post('/', (req, res) => {
  res.send('post, Express');
});

app.get('/about', (req, res) => {
  res.send('about, Express');
});

app.listen(app.get('port'), () => {
  console.log(app.get('port'), '번 포트에서 대기 중');
});
```

← → ↻ ⓘ localhost:3000

Hello, Express

← → ↻ ⓘ localhost:3000/about

about, Express

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

const app = express();
app.set('port', process.env.PORT || 3000);
```

```
app.use((req, res, next) =>{
  console.log('모든 요청에 실행');
  next();
}, (req, res, next) =>{
  throw new Error('에러 발생');
});
```

```
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, '/index.html'));
});


app.listen(app.get('port'), () => {
  console.log(app.get('port'), '번 포트에서 대기 중');
});
```

미들웨어

요청에 대한 응답 과정 중간에 끼서 어떠한 동작을 해주는 프로그램

Error: 에러

```
at C:\Users\zxcvb\Desktop\express-server\app.js:11:11
at Layer.handle [as handle_request] (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:317:13)
at C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:284:7
at Function.process_params (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:335:12)
at next (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:275:10)
at C:\Users\zxcvb\Desktop\express-server\app.js:9:5
at Layer.handle [as handle_request] (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\layer.js:95:5)
at trim_prefix (C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:317:13)
at C:\Users\zxcvb\Desktop\express-server\node_modules\express\lib\router\index.js:284:7
```

Name	Status	Type	Initiator	Size	Time
 localhost	500	docum...	Other	1.5 kB	11 ms



정보 노출

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

const app = express();
app.set('port', process.env.PORT || 3000);

app.use((req, res, next) =>{
  console.log('모든 요청에 실행');
  next();
}, (req, res, next) =>{
  throw new Error('에러 발생');
});

app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, '/index.html'));
});

app.listen(app.get('port'), () => {
  console.log(app.get('port'), '번 포트에서 대기 중');
});
```

추가

```
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, '/index.html'));
});

app.use((err, req, res, next) =>{
  console.error(err);
  res.send('에러 발생, 잠시 후에 다시 시도해주세요');
});

app.listen(app.get('port'), () => {
  console.log(app.get('port'), '번 포트에서 대기 중');
});
```

← → ↺ ⓘ localhost:3000/ab



에러 발생, 잠시후에 다시 시도해주세요

실제 에러

 localhost	500	docum...	Other	259 B	39 ms
---	-----	----------	-------	-------	-------

```
app.use((err, req, res, next) =>{  
  console.error(err);  
  res.status(200).send('에러 발생 , 잠시 후에 다시 시도해 주세요');  
})
```

브라우저 상

Name	Status	Type	Initiator	Size	Time
 lie	200	docum...	Other	282 B	15 ms
 lie	404	docum...	Other	289 B	24 ms

cookie-parser

Response Cookies

Name	Value
name	%EB%A1%9C%EA%B7%B8%EC...

name = gildong

```
req.cookies
```

```
{ name : 'gildong'}
```

```
app.get('/login', (req, res) => {
  res.cookie('cookie', 'cookiesecret', { signed : true });
  app.use(cookieParser('password'));
  res.sendFile(path.join(__dirname, '/index.html'));
});
```

Response Cookies

Name	Value	Domain	Path	Expires / Max...	Size	Http...	Sec...	Sam...	Prio...
cookie	s%3Acookiesecret.x...	localhost	/	Session	77				Med...

static

```
app.use(express.static());  
app.use('/', express.static(__dirname, 'public'));
```

```
app.use('요청 경로', express.static(__dirname, '실제 경로'));
```

localhost:3000/index.html

public/index.html

요청 경로와 실제 경로가 달라
서버 구조 파악이 어려워 보안에 도움

”

발표를 들어주셔서
감사합니다.

발표 끝!