

네트워크서비스 프로토콜

11주차 2

» 소프트웨어학부

» 김형균 교수



수업에 들어가며

- 오늘 학습할 내용
 - 10:40~11:10 학습참여도 평가 – 배점 3점
 - 출제된 문제를 각자 해결해서 완성 후 실행화면 노트북을 담당교수에게 확인
 - 선착 20명 – 3점
 - 다음 선착 20명 – 2점
 - 최종 10명 - 1점

초기 설정



- » E-캠퍼스 자료실에서 Nodejs-master.zip 다운로드 후 압축풀기
- » 에디터에 프로젝트 추가하기

A screenshot of the Visual Studio Code editor interface. On the left, the Explorer sidebar shows a project structure with folders like 'test', 'nodebird', 'node.js-mysql', and 'express'. The 'express' folder is expanded, showing subfolders like 'data', 'lib', 'nodejs', 'syntax', and files like '1.html', '2.html', '3.html', 'checkbox.html', 'coding.jpg', 'index.html', 'main.js', 'package-lock.json', 'package.json', and 'password.js'. The 'package.json' file is selected. On the right, the editor window shows the content of 'package.json' with line numbers 1 through 25. The JSON content is as follows:

```
1 {  
2   "name": "web2-nodejs",  
3   "version": "1.0.0",  
4   "description": "",  
5   "main": "main.js",  
6   "directories": {  
7     "lib": "lib"  
8   },  
9   "scripts": {  
10    "test": "echo \"Error: no test specified\" && exit 1"  
11  },  
12  "repository": {  
13    "type": "git",  
14    "url": "git+https://github.com/web-n/Nodejs.git"  
15  },  
16  "author": "",  
17  "license": "ISC",  
18  "bugs": {  
19    "url": "https://github.com/web-n/Nodejs/issues"  
20  },  
21  "homepage": "https://github.com/web-n/Nodejs#readme",  
22  "dependencies": {  
23    "sanitize-html": "^1.18.2"  
24  }  
25 }
```

초기 설정



» 콘솔에서 `npm install` 실행

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

새로운 크로스 플랫폼 PowerShell 사용 <https://aka.ms/pscore6>

```
PS C:\express\express> npm install
npm WARN web2-nodejs@1.0.0 No description
```

```
added 34 packages from 52 contributors and audited 48 packages in 0.917s
found 1 high severity vulnerability
```

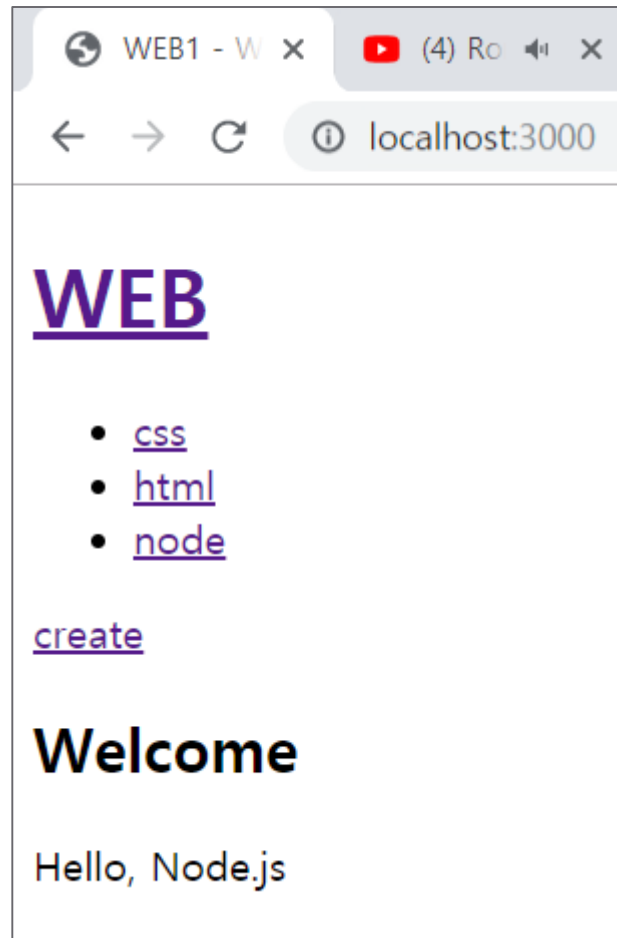
```
  run `npm audit fix` to fix them, or `npm audit` for details
```

```
PS C:\express\express> █
```

초기 설정



» 웹서버 실행 확인



express 설치

» 프로젝트 폴더에서 설치

» 에디터 콘솔에서 다음과 같이 명령

PS C:\express\express> npm install express --save

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

found 1 **high** severity vulnerability

run `npm audit fix` to fix them, or `npm audit` for details

PS C:\express\express> npm install express --save

npm **WARN** web2-nodejs@1.0.0 No description

+ express@4.17.1

added 48 packages from 36 contributors and audited 174 packages in 2.033s

found 1 **high** severity vulnerability

run `npm audit fix` to fix them, or `npm audit` for details

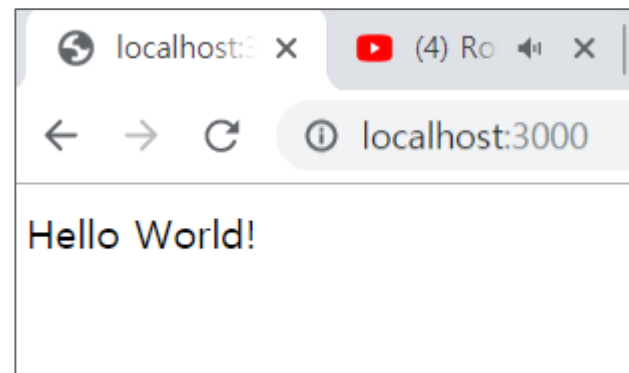
PS C:\express\express> █



Hello World

» main.js 기존 코드 전체 주석 처리 후 express 코드 입력

```
1  const express = require('express')
2  const app = express()
3
4  app.get('/', (req, res) => res.send('Hello World!'))
5
6  app.listen(3000, () => console.log('Example app listening on port 3000!'))
7
8  /*
9  var http = require('http');
```





라우팅

- » 클라이언트가 서버로 접속할때는 특정한 URL를 통해 접속한다.
- » 서버에서는 이 URL에 해당하는 자원을 클라이언트로 보내준다. 혹은 POST요청일 경우는 자원을 만들기도 한다.
- » 이러한 클라이언트 요청을 위한 **URL 스키마를 라우트**라고 한다.
- » 서버에서는 **라우팅 작업을 통해 클라이언트와 통신의 인터페이스를 제공**해 준다.
- » 익스프레스에서 중요한 것 중 하나가 이 **라우팅 모듈**이다.



라우팅 정의

» 기본적인 라우팅 설정 코드는 아래와 같다.

```
1  const express = require('express')
2  const app = express()
3
4  app.get('/', (req, res) => res.send('Hello World!'))
```

» 익스프레스 객체를 담고있는 app변수는 HTTP 메소드 명에 해당하는 함수를 가지고 있다.

- app.post()
- app.get()
- app.put()
- app.delete()

» 각 함수의 첫번째 파라미터에는 서버자원을 가리키는 URI 문자열을 지정한다.

» 두번째 파라미터는 라우팅 로직 함수를 콜백 형태로 구현한다.

» 즉 설정한 URI의 메소드로 요청이 들어오면 두번째 파라미터에 구현한 함수가 동작하는 것이다.



익스프레스 응답 메서드

» 익스프레스가 메서드를 추가함

- `res.send`(버퍼 또는 문자열 또는 HTML 또는 JSON): 기본 응답 메서드
- `res.sendFile`(파일 경로): 파일 전송
- `res.json`(JSON 데이터): 제이슨 데이터 응답
- `res.redirect`(주소): 리다이렉트
- `res.render`('템플릿 파일 경로', { 변수 }): 템플릿 엔진 렌더링

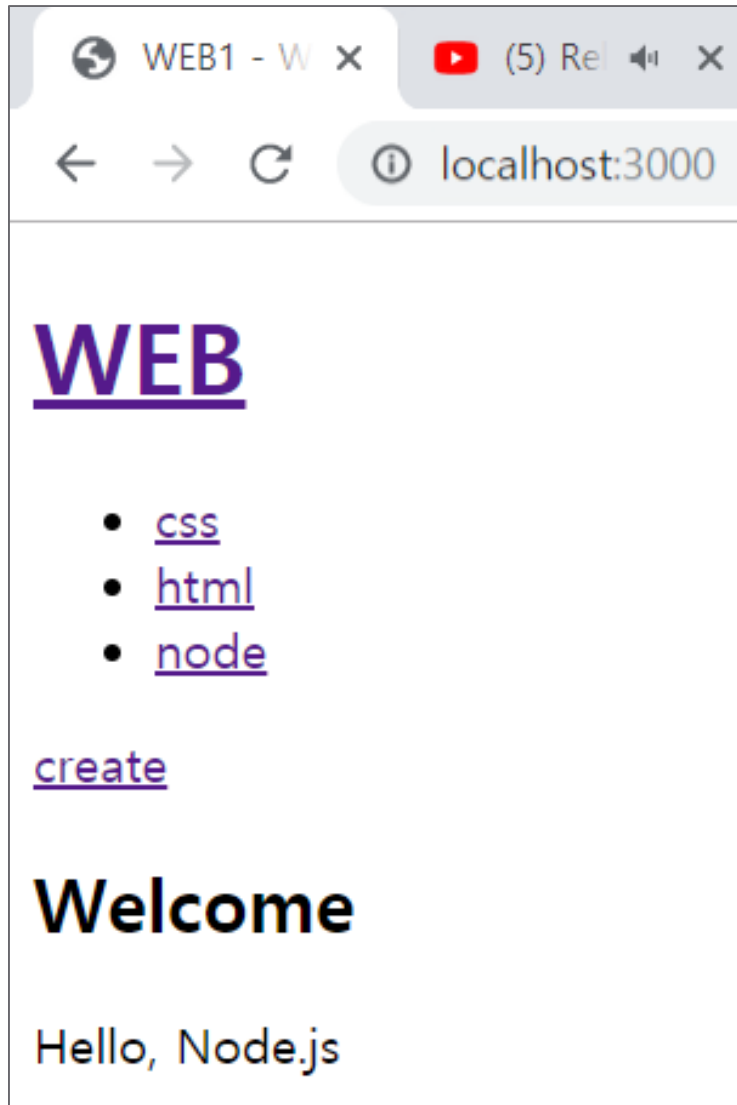


라우팅 실습

express > JS main.js > ...

```
1  var express = require('express')
2  var app = express()
3
4  app.get('/', function(req, res) {
5    |   return res.send('/');
6  });
7
8  app.get('/page', function(req, res) {
9    |   return res.send('/page');
10 });
11
12 app.listen(3000, function() {
13   |   console.log('Example app listening on port 3000!')
14 });
```

홈페이지 구현





홈페이지 구현

» 기존 코드 확인

```
var app = http.createServer(function(request, response){  
  var _url = request.url;  
  var queryData = url.parse(_url, true).query;  
  var pathname = url.parse(_url, true).pathname;  
  if(pathname === '/'){  
    if(queryData.id === undefined){  
      fs.readdir('./data', function(error, filelist){  
        var title = 'Welcome';  
        var description = 'Hello, Node.js';  
        var list = template.list(filelist);  
        var html = template.HTML(title, list,  
          `

## ${title}</h2>${description}`, ` ); response.writeHead(200); response.end(html); }); } } });


```

라우팅 처리

본문에서 사용

응답 메서드



express > JS main.js > ...

```
1  var express = require('express')
2  var app = express()
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17  });
```