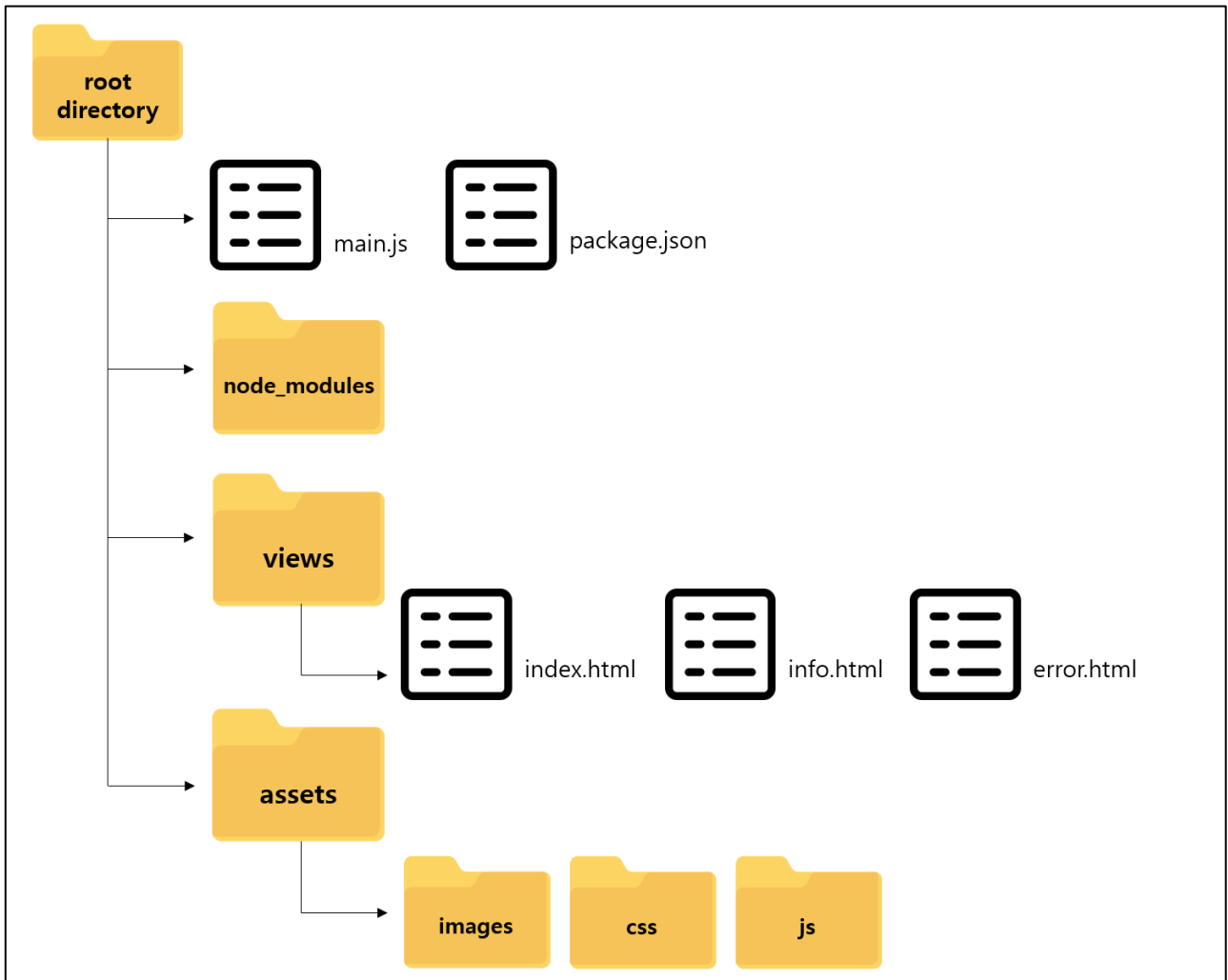


# 라우트와 에셋

## 에셋(Assets)

- 에셋이란 App에서 제공하는 이미지(jpg, png.), 스타일시트(css), JavaScript파일을 뜻하며 Client측에서 뷰 페이지와 함께 동작한다.
- .jpg, .png, .css, .js등의 에셋들도 각자 고유의 라우트가 필요하다.
- 예제를 진행 전 4챕터(웹 서버 만들기)에서 진행하였을 때 만들었던 웹 서버의 루트 디렉터리를 복사하여서 아래와 같은 구조가 될 수 있도록 디렉터리를 생성해주도록 한다.



- /views/index.html : 홈 페이지 html
- /views/info.html : 정보 html
- /views/error.html : HTTP 404 html
- /assets/images : 이미지 파일 에셋 저장 디렉터리
- /assets/css : 스타일시트 파일 에셋 저장 디렉터리
- /assets/js : 자바스크립트 파일 에셋 저장 디렉터리

✓ index.html, info.html, error.html, images, css, js등의 콘텐츠는 무엇이 되었든 작성 및 생성하도록 한다.

## 예제1) 파일 확장자에 따른 라우팅(에셋응답)

1. 전 챕터(4)에서 만들었던 HTTP 웹 서버의 루트 디렉토리를 복사하여 새로운 디렉토리 생성
2. 복사된 디렉토리내 main.js의 이름을 main\_assets.js로 변경  
: mv main.js main\_assets.js
3. html 파일이 저장될 views디렉토리 생성  
: mkdir views
4. views디렉토리로 이동하여 index.html, info.html, error.html 작성 (내용은 무엇이든 가능하다.)  
: cd views  
: vi index.html info.html error.html
5. 에셋들이 저장될 assets디렉토리 생성  
: mkdir assets
6. 에셋의 종류에 따라 assets디렉토리 구조화 진행  
: mkdir assets/images assets/css assets/js  
✓ 여기까지 진행하였을 시 위의 디렉토리 구조와 같이 웹 디렉토리가 완성되었다.
7. main\_assets.js에 아래의 내용과 같이 입력

```
"use strict";
const http = require("http");
const httpStatus = require("http-status-codes");
const fs = require("fs");
const app = http.createServer();

const getJSONString = function(obj){
    return JSON.stringify(obj, null, 2);
};

// Client에서 요청하는 파일이 있을 시 응답헤더 작성 함수
const setOkResponse = function(res, content_type){
    res.writeHead(httpStatus.OK, {
        "Content-Type" : content_type
    });
    return res;
};

// 에러 응답 함수
const sendErrorResponse = function(res){
    res.writeHead(httpStatus.NOT_FOUND, {
        "Content-Type" : "text/html"
    });
    customReadFile("./views/error.html", res);
};
```

```
// 파일경로에 따른 응답
const customReadFile = function(file_path, res){
    if(fs.existsSync(file_path)){
        // file_path에 해당하는 파일이 있을 시
        fs.readFile(file_path, function(err, data){
            // 파일 읽기
            if(err){
                // 파일 읽기 에러
                console.log(err);
                sendErrorResponse(res);
                return;
            }
            // 파일 응답
            res.end(data);
        });
    }
    else{
        // file_path에 해당하는 파일이 없을 시
        // HTTP 404 에러 응답
        sendErrorResponse(res);
    }
};

// 다음 페이지에서 이어서 작성
```

## // 전 페이지부터 이어서 작성

```
app.on("request", function(req, res){
    let url = req.url;

    console.log("—Request—");
    console.log("Method : ", req.method);
    console.log("URL : ", req.url);
    console.log("Headers : ", getJSONString(req.headers));

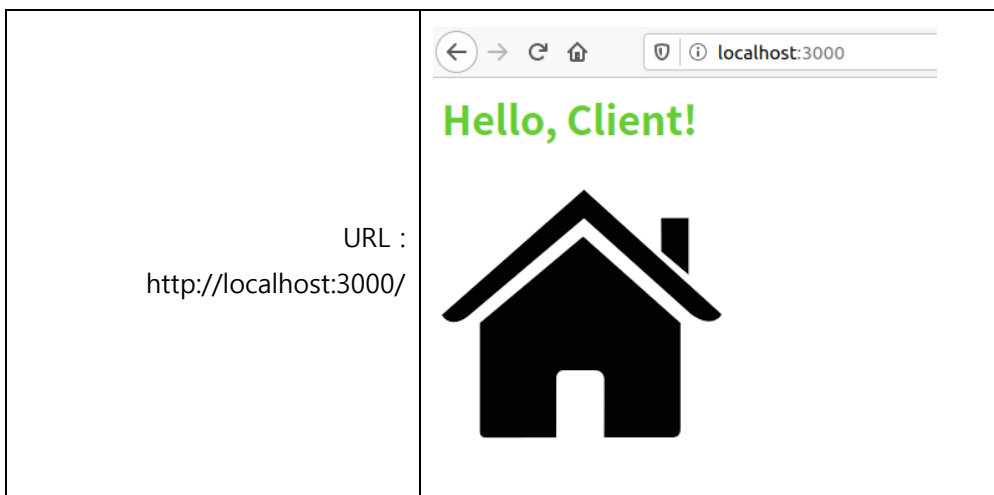
    // 요청 파일 확장자에 따라 다른 경로로 응답
    if(url == "/" )
    {
        // 루트 접근 시 index.html로 응답
        res = setOkResponse(res, "text/html; charset=utf8");
        customReadFile("./views/index.html", res);
    }
    else if(url.indexOf(".html") >= 0) // .html
    {
        res = setOkResponse(res, "text/html; charset=utf8");
        customReadFile("./views"+url, res);
    }
    else if(url.indexOf(".js") >= 0) // .js
    {
        res = setOkResponse(res, "text/javascript");
        customReadFile("./assets/js"+url, res);
    }
    else if(url.indexOf(".css") >= 0) // .css
    {
        res = setOkResponse(res, "text/css");
        customReadFile("./assets/css"+url, res);
    }
    else if(url.indexOf(".jpg") >= 0)
    {
        res = setOkResponse(res, "image/jpg");
        customReadFile("./assets/images"+url, res);
    }
    else if(url.indexOf(".png") >= 0)
    {
        res = setOkResponse(res, "image/png");
        customReadFile("./assets/images"+url, res);
    }
    else
    {
        // App에서 제공하지 않은 확장자일 시
        // HTTP 404 에러 응답
        sendErrorResponse(res);
    }

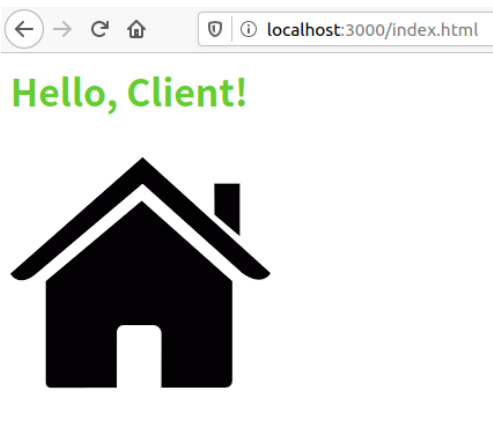

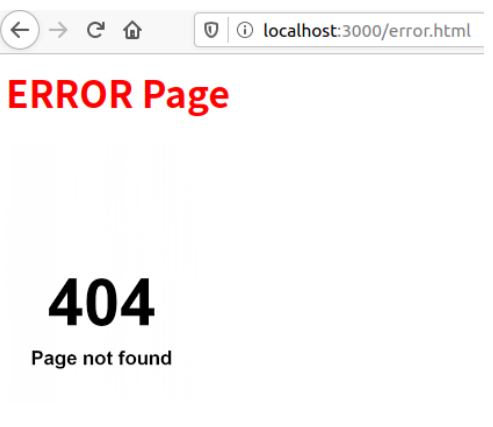
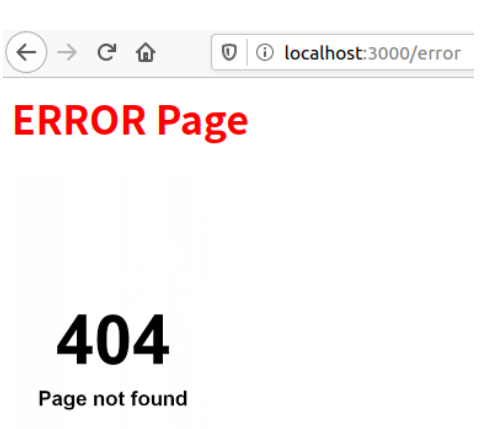
    console.log();
});

app.listen(port);
console.log("The Server has started and is listening on port
number : %d", port);
```

8. main\_assets.js실행  
: node main\_assets.js

✓ 클라이언트 웹 브라우저 실행화면



<p>URL : http://localhost:3000/index.html</p>	 <p>The screenshot shows a web browser window with the address bar displaying 'localhost:3000/index.html'. The page content includes the text 'Hello, Client!' in green and a large black silhouette of a house.</p>
<p>URL : http://localhost:3000/info.html</p>	 <p>The screenshot shows a web browser window with the address bar displaying 'localhost:3000/info.html'. The page content includes the text 'This is info page.' in blue and a large grey information icon (a lowercase 'i' inside a circle).</p>
<p>URL : http://localhost:3000/error.html</p>	 <p>The screenshot shows a web browser window with the address bar displaying 'localhost:3000/error.html'. The page content includes the text 'ERROR Page' in red and '404 Page not found' in black.</p>
<p>URL : http://localhost:3000/error</p> <p>(서버에 존재하지 않는 파일로 고의적 에러 유발)</p>	 <p>The screenshot shows a web browser window with the address bar displaying 'localhost:3000/error'. The page content includes the text 'ERROR Page' in red and '404 Page not found' in black.</p>

✓ 서버 콘솔 실행화면

```
wlgns12www@nodejs-test:~/nodejs/7_assets/simple_assets$ ls
assets  main_assets.js  node_modules  package-lock.json  package.json  views
wlgns12www@nodejs-test:~/nodejs/7_assets/simple_assets$ node main_assets.js
The Server has started and is listening on port number : 3000

--Request--
Method : GET
URL : /
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "upgrade-insecure-requests": "1"
}

--Request--
Method : GET
URL : /font.css
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/css,*/*;q=0.1",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/"
}

--Request--
Method : GET
URL : /home.png
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "image/webp,*/*",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/"
}

--Request--
Method : GET
URL : /index.html
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "upgrade-insecure-requests": "1"
}

--Request--
Method : GET
URL : /font.css
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/css,*/*;q=0.1",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/index.html"
}

--Request--
Method : GET
URL : /home.png
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "image/webp,*/*",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/index.html"
}
```

App 시작

Request  
URL  
>> /

Request  
URL  
>>  
/index.html

```
--Request--
Method : GET
URL : /info.html
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "upgrade-insecure-requests": "1"
}

--Request--
Method : GET
URL : /font.css
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/css,*/*;q=0.1",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/info.html"
}

--Request--
Method : GET
URL : /info.png
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "image/webp,*/*",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/info.html"
}

--Request--
Method : GET
URL : /error.html
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "upgrade-insecure-requests": "1"
}

--Request--
Method : GET
URL : /font.css
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/css,*/*;q=0.1",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/error.html"
}

--Request--
Method : GET
URL : /not_found.png
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "image/webp,*/*",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/error.html"
}
```

Request

URL

>>

/info.html

Request

URL

>>

/error.html

```
--Request--
Method : GET
URL : /error
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "upgrade-insecure-requests": "1"
}

--Request--
Method : GET
URL : /font.css
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "text/css,*/*;q=0.1",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/error"
}

--Request--
Method : GET
URL : /not_found.png
Headers : {
  "host": "localhost:3000",
  "user-agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0",
  "accept": "image/webp,*/*",
  "accept-language": "ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3",
  "accept-encoding": "gzip, deflate",
  "connection": "keep-alive",
  "referer": "http://localhost:3000/error"
}
```

Request  
URL  
>> /error

Ex) index.html 내용

```
<html>
  <head>
    <meta charset="utf-8"/>
    <link rel="stylesheet" type="text/css" href="font.css"/> <!-- /font.css로 서버로 URL을 보낸다. -->
    <title>sample</title>
  </head>
  <body id="home">
    <h1>Hello, Client</h1>
     <!-- /home.png로 서버로 URL을 보낸다. -->
    <!-- 서버는 /home.png URL을 보고 정의된 라우트에 따라 라우팅한다. -->
  </body>
</html>
```

- ✓ 해당 예제에서는 index.html만이 아닌 info.html, error.html, font.css, home.png등 다양한 파일을 생성해서 실행한 예제 임으로 해당 예제를 실행할 때에는 자신이 직접 html과 css로 페이지를 생성하도록 한다.

## 참고문헌

- 조나단 웨슬러, 김성준. (2020.01.31). NODEJS로 프로그래밍 시작하기. 105-110.