

# 스타트업 개발자와 함께 공부하는 Node.js

## 02. 노드 소개

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

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



# 목차

1. 노드 소개
2. 노드 설치
3. 개발환경 설치
4. 첫 번째 웹서버
5. HTTP 소개
6. 배열과 관련 함수

Seoul woman up 

# 노드 소개

# Node.js 소개

## ❑ 노드(Node.js)

- ❑ 서버측 애플리케이션 개발에 사용되는 오픈 소스 런타임 환경
- ❑ 2009년 Ryan Dahl 에 의해 처음 개발

## ❑ 주요 특징

- ❑ 비동기 I/O
- ❑ 싱글 스레드
- ❑ NPM(Node Package Manager)

## ❑ 주요 사용 사례

- ❑ 웹 서버
- ❑ API 서버
- ❑ 실시간 애플리케이션

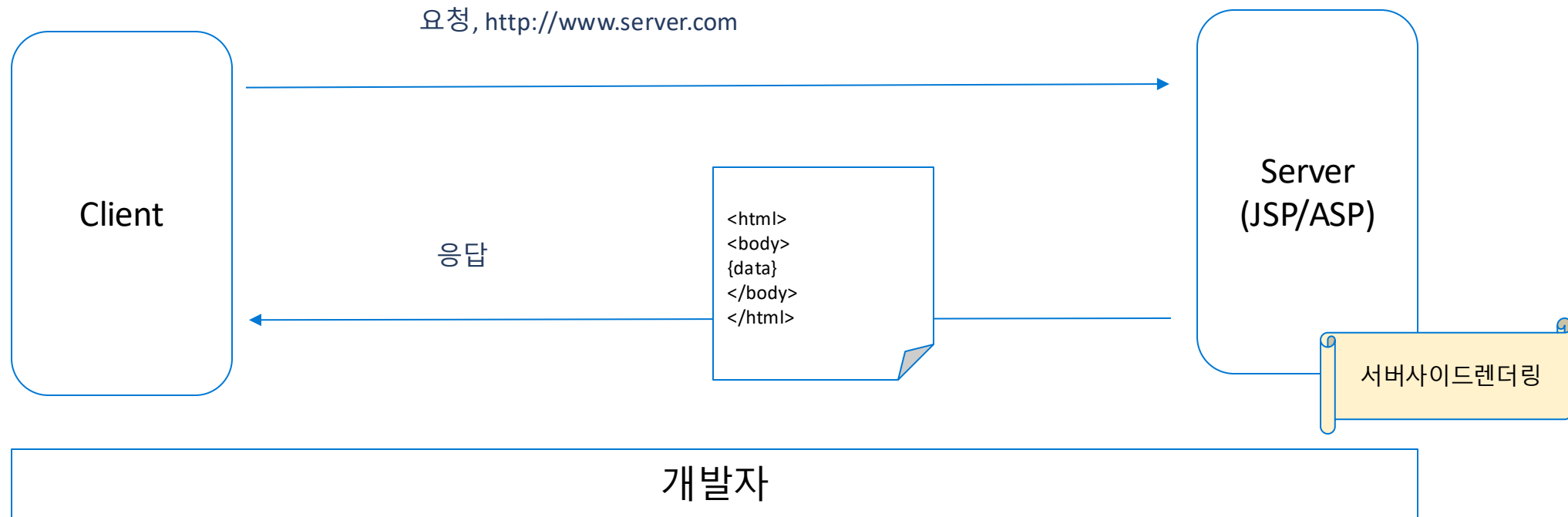
## Node.js 장 단점

- 빠른 성능
  - 확장성
  - 풍부한 모듈 생태계
  - 자바스크립트 기반
- 싱글 스레드의 한계
  - 콜 백(Callback) 지옥
  - 메모리 누수

# 노드가 등장하기 전

## Web 1.0

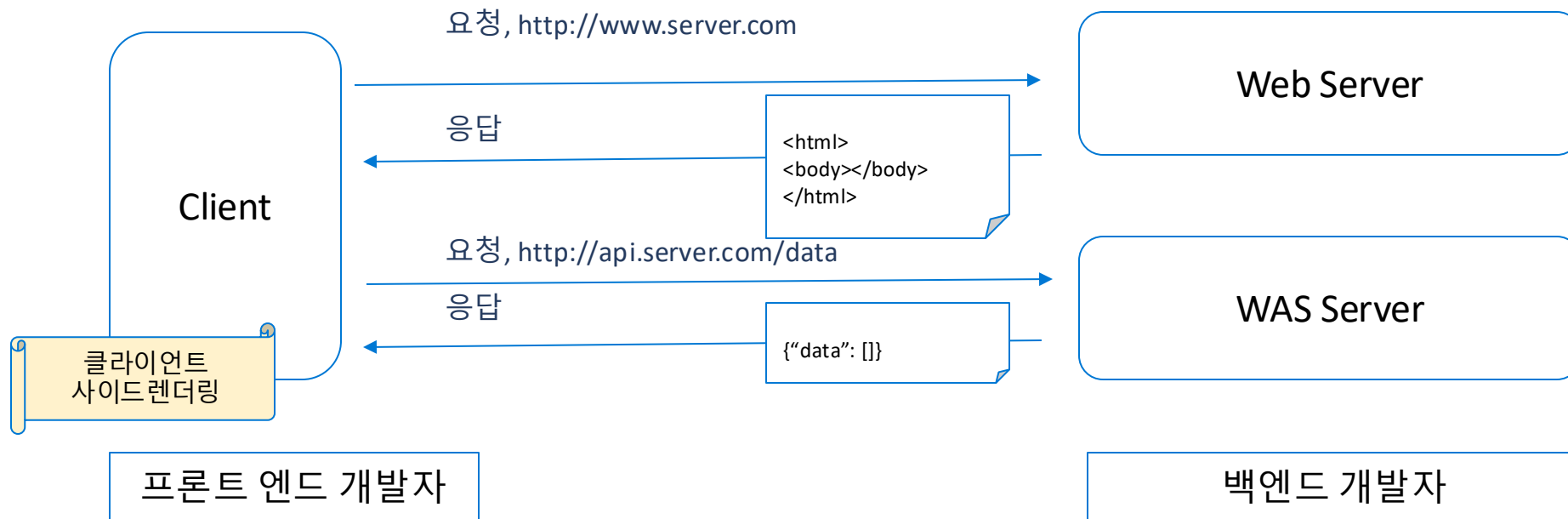
- 2000년 대 초반 부터 Web 개발은 Java, JSP, ASP 등이 주류
- 화면 UI 코드와 데이터가 같이 처리됨



# 노드가 등장하기 전

## Web 2.0

- 2005년 부터 구글과 아마존에서는 자바스크립트 기반의 리치 클라이언트를 제공
- Web1.0과 가장 큰 기술적 특징은 AJAX를 이용하여 화면 UI코드와 데이터를 분리

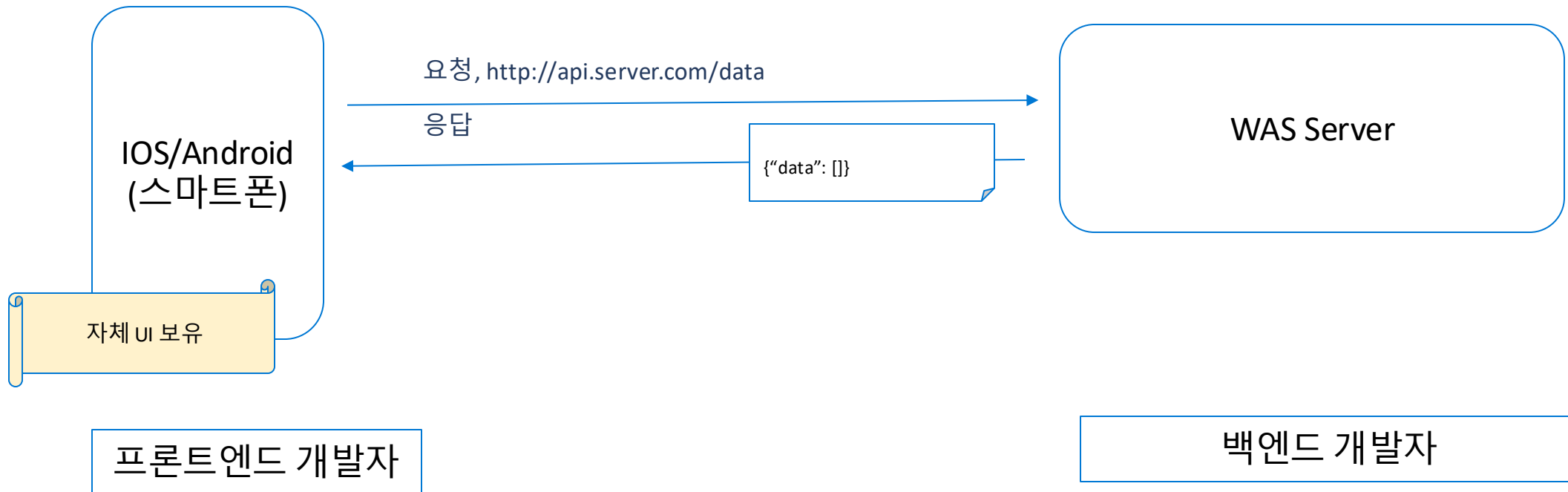




# 노드가 등장하기 전

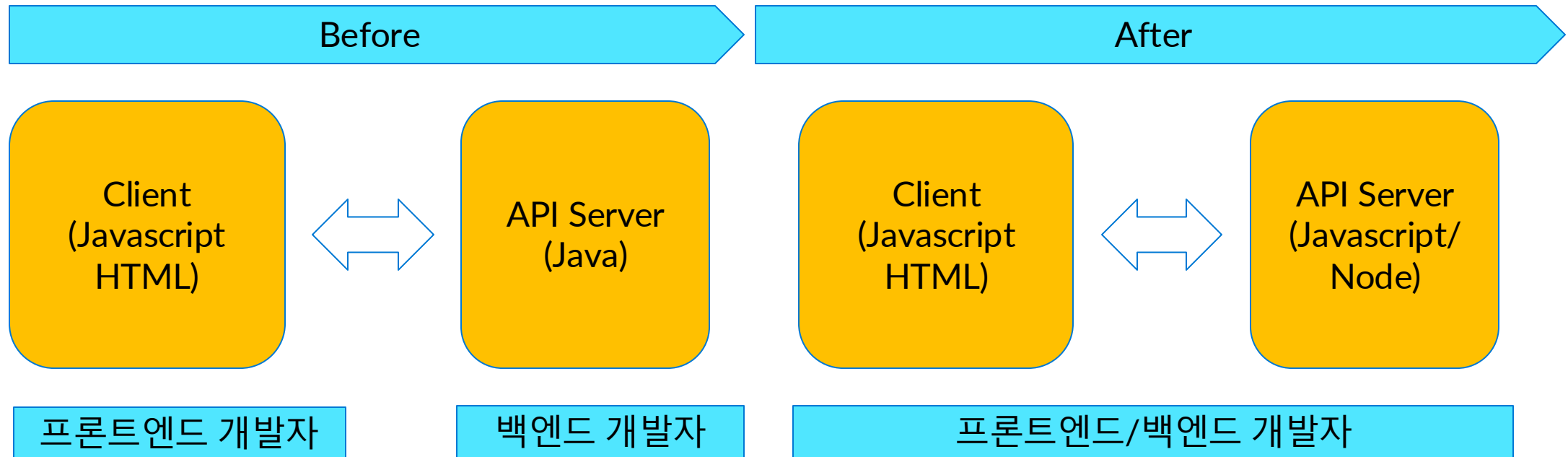
## Smart Phone

- 2008년 부터 아이폰, 안드로이드 폰의 등장



# 페이팔(Paypal)

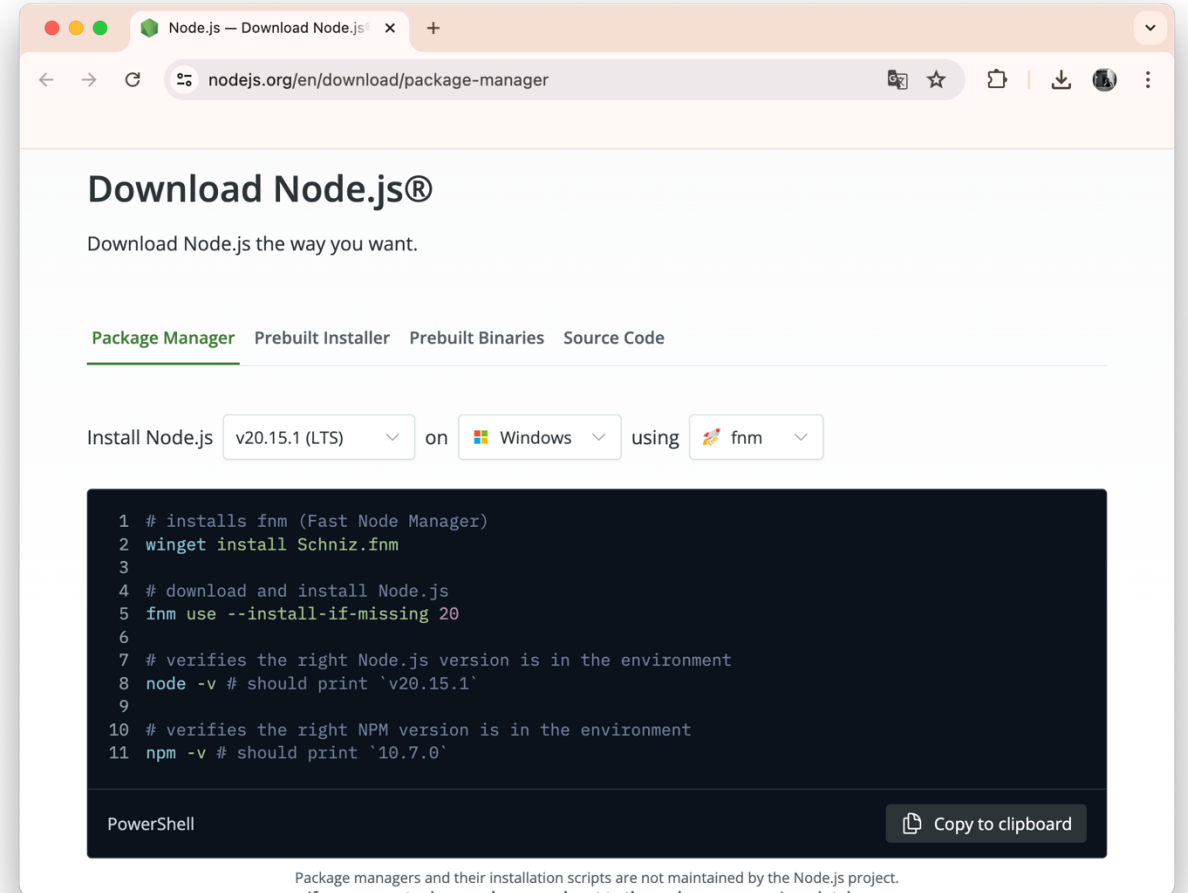
- 2013년 페이팔은 서버 환경을 Java 에서 Node.js 로 변경



# 드 설치

# 노드 설치

- ❑ [nodejs.org/en/download](https://nodejs.org/en/download) 로 이동
- ❑ Package Manager 에서 Windows 선택
- ❑ 파워셸을 열고 가이드 대로 타이핑



# 개발 도구 설치(Visual Studio Code)

- ❑ <https://code.visualstudio.com/docs/setup/windows>
- ❑ 가이드 대로 설치
- ❑ 이하 vscode 로 칭함

## Visual Studio Code on Windows

[Edit](#)

### Installation

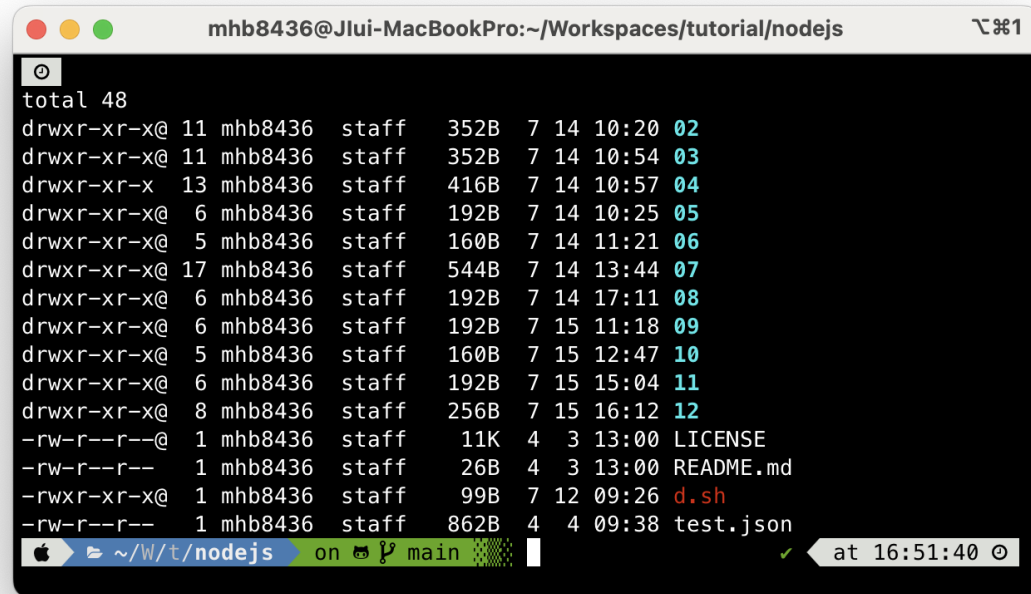
1. Download the [Visual Studio Code installer](#) for Windows.
2. Once it is downloaded, run the installer (VSCodeUserSetup-{version}.exe). This will only take a minute.
3. By default, VS Code is installed under `C:\Users\{Username}\AppData\Local\Programs\Microsoft VS Code`.

Alternatively, you can also download a [Zip archive](#), extract it and run Code from there.

**Tip:** Setup will add Visual Studio Code to your `%PATH%`, so from the console you can type 'code .' to open VS Code on that folder. You will need to restart your console after the installation for the change to the `%PATH%` environmental variable to take effect.

# 실습 디렉토리 만들기

- 강의 자료는 <http://github.com/mhb8436/nodejs/starter.zip> 에서 다운로드
- 배포된 starter.zip 파일을 [C]-[nodejs]-[project] 밑에 복사 후 압축을 풀
- [C]-[nodejs]-[project] 디렉토리의 내용은 다음과 같음



```
mhb8436@Jlui-MacBookPro:~/Workspaces/tutorial/nodejs
total 48
drwxr-xr-x@ 11 mhb8436  staff   352B  7 14 10:20 02
drwxr-xr-x@ 11 mhb8436  staff   352B  7 14 10:54 03
drwxr-xr-x  13 mhb8436  staff   416B  7 14 10:57 04
drwxr-xr-x@  6 mhb8436  staff   192B  7 14 10:25 05
drwxr-xr-x@  5 mhb8436  staff   160B  7 14 11:21 06
drwxr-xr-x@ 17 mhb8436  staff   544B  7 14 13:44 07
drwxr-xr-x@  6 mhb8436  staff   192B  7 14 17:11 08
drwxr-xr-x@  6 mhb8436  staff   192B  7 15 11:18 09
drwxr-xr-x@  5 mhb8436  staff   160B  7 15 12:47 10
drwxr-xr-x@  6 mhb8436  staff   192B  7 15 15:04 11
drwxr-xr-x@  8 mhb8436  staff   256B  7 15 16:12 12
-rw-r--r--@  1 mhb8436  staff    11K  4  3 13:00 LICENSE
-rw-r--r--  1 mhb8436  staff     26B  4  3 13:00 README.md
-rwxr-xr-x@  1 mhb8436  staff     99B  7 12 09:26 d.sh
-rw-r--r--  1 mhb8436  staff     862B  4  4 09:38 test.json
```

# Hello World

```
console.log('hello world');
```

- ❑ 윈도우 탐색기에서 [C]-[nodejs]-[ch02] 오른쪽 마우스 클릭 'vscode로 열기' 메뉴 클릭
- ❑ 소스 파일명 : ch02\_01.js
- ❑ 좌측과 같이 코드 입력
- ❑ vscode 메뉴에서 새로운 터미널을 열고
- ❑ node ch02\_01.js

# 확인 문제

```
1 for(var i=0;i<10;i++) {  
2   console.log('file' + (i+1));  
3 }  
4  
5 console.log('-----');  
6  
7 const arr = [1,2,3,4,5,6,7,8,9,10];  
8 arr.forEach(i=> {  
9   console.log(`file${i}`)  
10 });  
11  
12 console.log('-----');  
13  
14 Array.from({length: 10}, (_,i) => {  
15   console.log('file' + (i+1))  
16 });  
17  
18 console.log('-----');  
19  
20 const arr2 = [...Array(10).keys()];  
21 arr2.forEach(i=> {  
22   console.log(`file${i+1}`)  
23 });
```

- ❑ file1, file2, ... file10 까지 라인단위로 출력하는 프로그램을 만들어보세요
- ❑ 소스파일명 : ch02\_02.js



# 첫 번째 웹서버

```
1 const http = require('http');  
2  
3 http.createServer((req, res) => {  
4     res.statusCode = 200;  
5     res.setHeader("Content-Type", "text/plain");  
6     res.write('Hello World');  
7     res.end();  
8 }).listen(4500);  
-
```

- ❑ 첫 번째 웹 서버를 만들어 보세요
- ❑ 소스파일명 : ch02\_03.js

# 화살표 함수

- ❑ ES6에서 도입
- ❑ function 키워드 대신 => 사용

```
const 함수이름 = (매개변수) => {  
    // 함수 본문  
}
```

# 화살표 함수

## □ 예제

```
// 매개변수가 없는 경우
const sayHello = () => {
  console.log('Hello, world!');
};
```

```
// 매개변수가 하나인 경우
const greet = name => {
  console.log(`Hello, ${name}!`);
};
```

```
// 매개변수가 여러 개인 경우
const add = (a, b) => {
  return a + b;
};
```

# 화살표 함수

## □ 단일 표현식

```
const add = (a, b) => a + b;
```

## □ 객체 리터럴 반환

```
const getUser = () => ({ name: 'John', age: 30 });
```

# 화살표 함수

```
1 // 일반 함수
2 function multiply(a, b) {
3   |   return a * b;
4 }
5 console.log(multiply(2, 3));
6
7
8 // 애로우 함수
9 const multiply2 = (a, b) => a * b;
10 console.log(multiply2(4, 5));
11
```

- ❑ 소스파일명 : ch02\_04.js
- ❑ 좌측의 코드를 넣어 보고 실행해 보세요

# 확인 문제

```
--  
13 function calc(x,y,z) {  
14     let result = x * y + z;  
15     return result  
16 }  
17
```

- ❑ 소스파일명 : ch02\_04.js
- ❑ 좌측의 함수를 화살표 함수로 만들어보세요

# HTTP

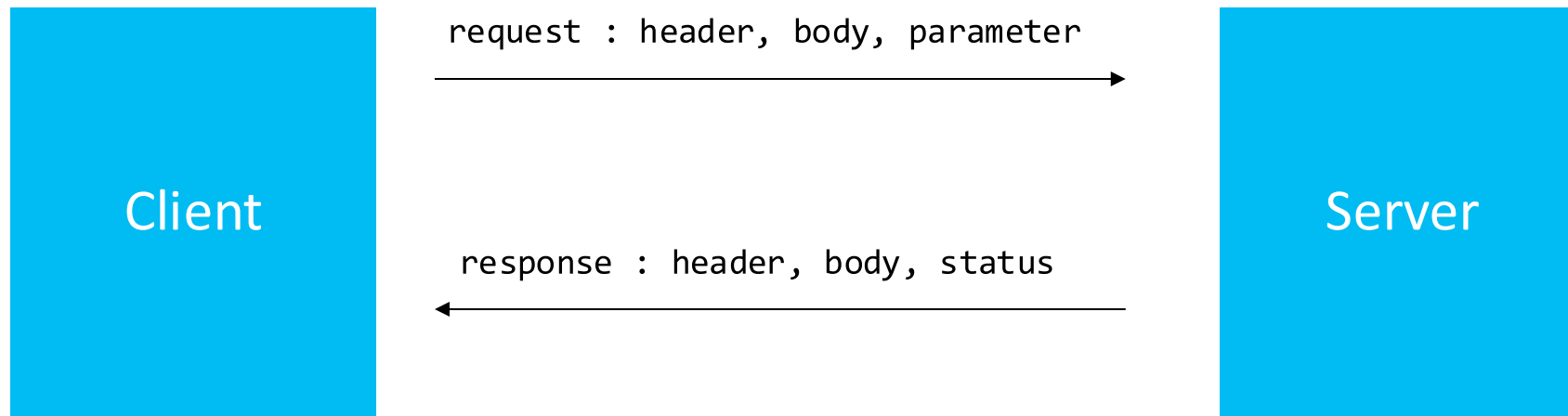
<http://127.0.0.1:3000/user>

- ❑ http : 프로토콜 이름
- ❑ 127.0.0.1 : 서버 이름
- ❑ 3000 : 포트 번호
- ❑ user : URI → 웹 서버의 라우팅 주소

# HTTP

## Request and Response

□ HTTP 는 요청과 응답 구조로 이루어짐





# HTTP

## Response code

- ❑ 2XX : 성공

  - ❑ 200

  - ❑ 201

- ❑ 3XX : 리다이렉션

- ❑ 4XX : 클라이언트 에러

- ❑ 5XX : 서버 에러

# HTTP

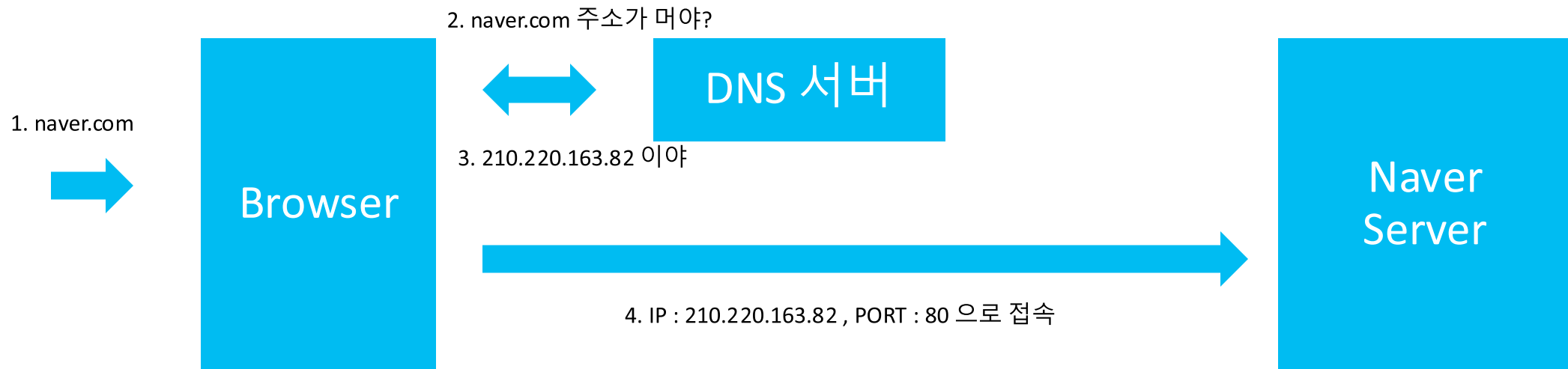
## Header

- ❑ 클라이언트와 서버가 요청 또는 응답으로 부가정보 전송
- ❑ 대표적인 요청 헤더
  - ❑ Authorization : 인증 정보
- ❑ 대표적인 응답 헤더
  - ❑ Content-Type : text/plain, application/json

# HTTP

## 도메인 (Domain)

- 컴퓨터가 인식할 수 있는 IP 주소
- 사람이 인식할 수 있는 도메인 주소



# 확인 문제

```
1 const http = require('http');
2
3 http.createServer((req, res) => {
4   res.statusCode = 200;
5   res.setHeader("Content-Type", "text/plain");
6   const arr = [...Array(10).keys()];
7   console.log(arr);
8   const arr2 = arr.map(x=> {
9     return 'Hello World ' + (x+1);
10  });
11  const content = arr2.join('\n');
12  res.write(content);
13  res.end();
14 }).listen(5000);
```

- ❑ 소스코드 파일명 : ch02\_05.js
- ❑ 브라우저에서 접속 하면 Hello Web Server를 10줄 출력하는 웹 서버를 만들어보세요. (포트는 3000번)
- ❑ 포트를 5000 번으로 바꾸어 보세요
- ❑ 터미널을 하나 더 열고 ext05.js 실행 해보세요

# 배열 및 관련 함수

```
1 let arr = [1,2,3]
2
3 // push
4 arr.push(4)
5 console.log('arr', arr)
6
7 // map
8 arr2 = arr.map(x=>{
9   return x+1;
10 });
11 console.log('arr2', arr2);
12
13 // filter
14 arr3 = arr.filter(x=> {
15   return x%2 == 0; // boolean
16 });
17 console.log('arr3', arr3)
18
19 // forEach
20 arr.forEach((v, i)>= {
21   console.log('arr4', v+2)
22 });
23
24 // sort
25 arr5 = arr.sort((a,b) => {
26   return b - a;
27 });
28 console.log('arr5', arr5)
29
```

❑ 소스파일명 : ch02\_06.js

❑ push

❑ map

❑ filter

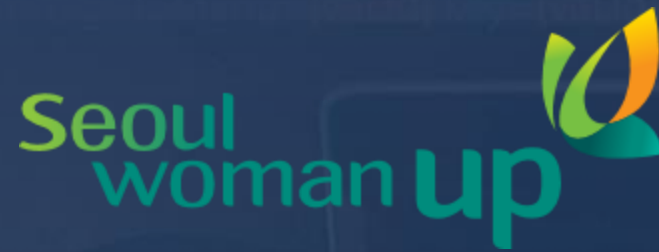
❑ forEach

❑ sort

# 확인 문제

```
1 let data = [1,2,3,4,5,6,7,8,9,10]
2
3 for(var i=11;i<21;i++) {
4   |   data.push(i)
5 }
6
7 const data2 = data.map(x => x*2);
8 console.log(data2)
9
10 const data3 = data.filter(x => x%2 ==0);
11 console.log(data3)
12
13 data.forEach(x=> {
14   |   if(x >=5 && x <= 15){
15   |     |   console.log(x);
16   |   }
17 })
```

- ❑ 소스 파일명: ch02\_07.js
- ❑ data 배열 생성과 동시에 1~10까지 할당합니다.
- ❑ data 배열에 11부터 20까지 추가 합니다.
- ❑ map 함수를 이용해 각 원소에 2를 곱한 새로운 배열을 생성하고 출력합니다.
- ❑ filter 함수를 이용해 홀수 원소만 있는 새로운 배열을 생성하고 출력합니다.
- ❑ forEach 함수를 이용해 데이터 배열 중 5 ~ 15 값만 출력 합니다.



백문이 불여일타