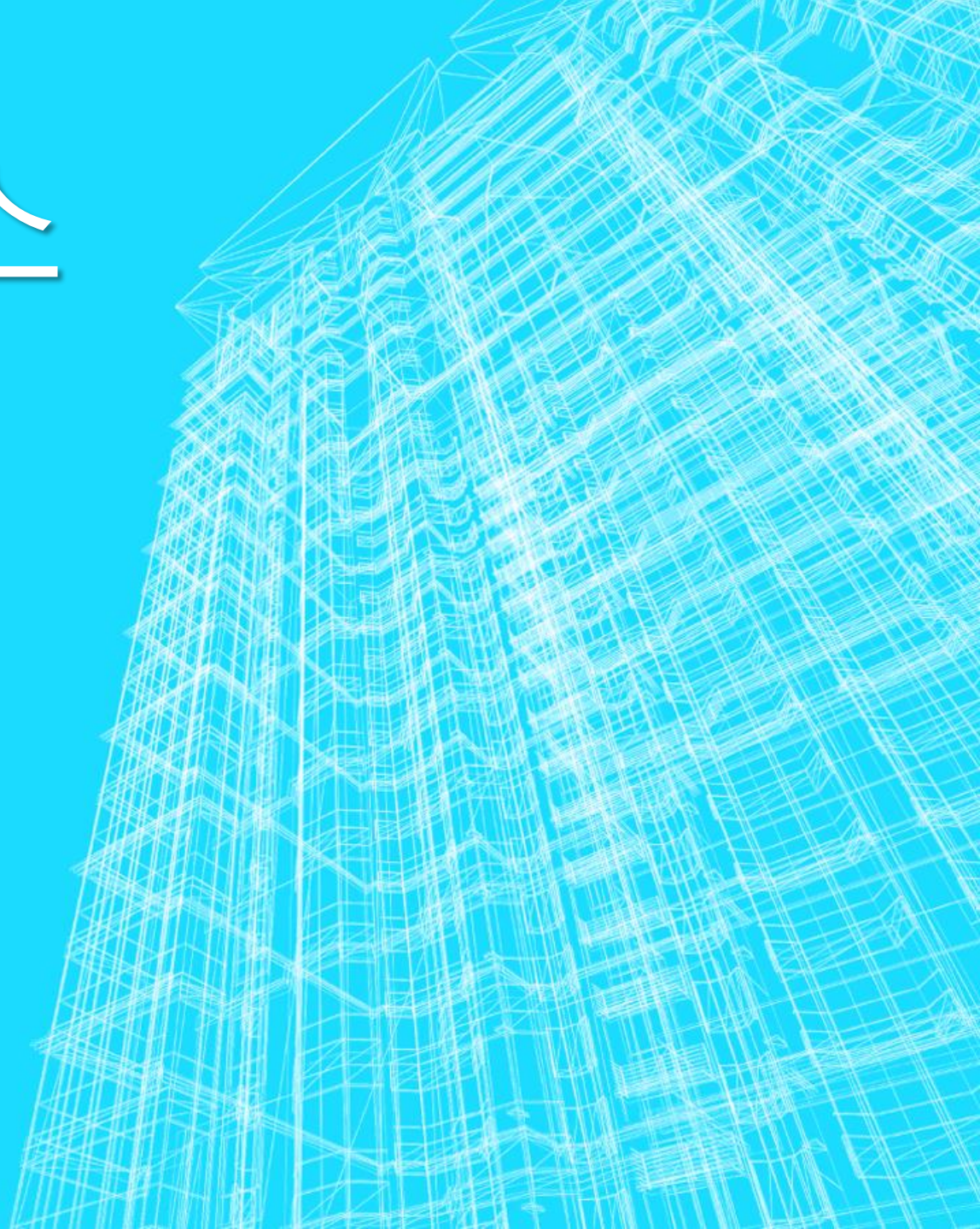


# 네트워크 서비스 프로토콜

## 2주차

소프트웨어학부  
김형균 교수



# 주차별 수업 계획

01주차	9월 3일	교과목 소개	09주차	10월 29일	몽고디비 1
	9월 5일	교과목 소개		10월 31일	몽고디비 2
02주차	9월 10일	노드 시작하기	10주차	11월 5일	익스프레스로 SNS 서비스 만들기 1
	9월 12일	추석 휴무		11월 7일	익스프레스로 SNS 서비스 만들기 2
03주차	9월 17일	알아두어야 할 자바스크립트 1	11주차	11월 12일	웹 API 서버 만들기 1
	9월 19일	알아두어야 할 자바스크립트 2		11월 14일	웹 API 서버 만들기 2
04주차	9월 24일	노드 기능 알아보기 1	12주차	11월 19일	웹 소켓으로 실시간 데이터 전송하기 1
	9월 26일	노드 기능 알아보기 2		11월 21일	웹 소켓으로 실시간 데이터 전송하기 2
05주차	10월 1일	개천절 휴무	13주차	11월 26일	실시간 경매 시스템 만들기 1
	10월 3일	패키지 매니저		11월 28일	실시간 경매 시스템 만들기 2
06주차	10월 8일	익스프레스 웹 서버 만들기 1	14주차	12월 3일	과제발표 1
	10월 10일	익스프레스 웹 서버 만들기 2		12월 5일	과제발표 2
07주차	10월 15일	MySQL 1	15주차	12월 10일	기말고사
	10월 17일	MySQL 2		12월 12일	보충수업
08주차	10월 22일	중간고사			
	10월 24일	보충수업			



# 수업에 들어가며

- 학습자 분석평가 실시
  - 자바스크립트 이해도 테스트- 문제 3문항
- 이번 시간 학습목표
  - Node.js 개념 이해하기
  - Node.js 설치
  - Node.js 런타임 실행
  - 객체 리터럴로 객체 생성하기
  - DOM을 사용해 이벤트처리기 등록



# 학습자 분석평가 실시

- 평가시간 20분
- 평가지 다운로드후 해당 문제의 정답작성 란에 코드 작성해서 업로드
  - 파일형태 : pptx
  - 파일이름 : 성명학번
- 코드 테스트 방법
  - 문제 1,2 번은 크롬의 개발자도구(ctrl+shift+J) 상태에서 테스트
  - 문제 3번은 메모장을 통해 제시된 코드를 활용해 완성 저장(\*.html)후 웹브라우저로 실행결과 확인
  - 문제 3번 사용함수 : `Math.sqrt( )`





# 노드의 정의

- 공식 홈페이지 설명
  - Node.js®는 크롬 V8 자바스크립트 엔진으로 빌드된 자바스크립트 런타임입니다.
  - Node.js는 이벤트 기반, 논블로킹 I/O 모델을 사용해 가볍고 효율적입니다.
  - Node.js의 패키지 생태계인 npm은 세계에서 가장 큰 오픈 소스 라이브러리 생태계이기도 합니다.
- 노드는 서버 아닌가요?
  - 서버의 역할도 수행할 수 있는 자바스크립트 런타임



# 런타임

## 노드: 자바스크립트 런타임

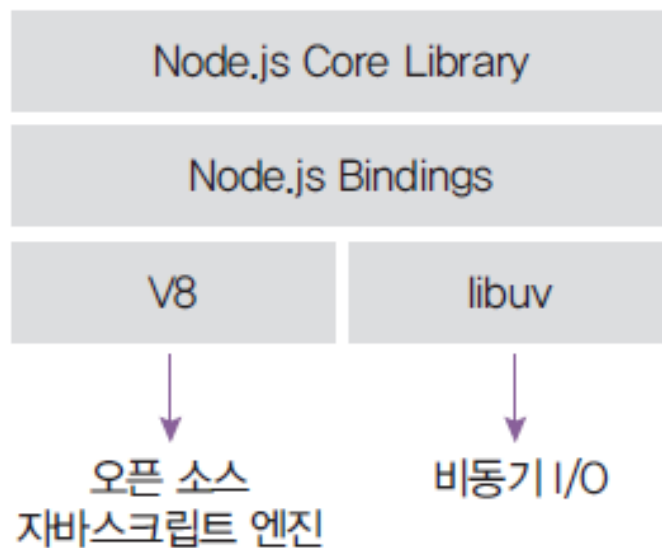
- 런타임: 특정 언어로 만든 프로그램들을 실행할 수 있게 해주는 가상 머신 (크롬의 V8 엔진 사용)의 상태
- ∴ 노드: 자바스크립트로 만든 프로그램들을 실행할 수 있게 해 줌
- 다른 런타임으로는 웹 브라우저가 있음
- 노드 이전에도 자바스크립트 런타임을 만들기 위한 많은 시도
- But, 엔진 속도 문제로 실패

# 내부 구조

2008년 V8 엔진 출시, 2009년 노드 프로젝트 시작

노드는 V8과 libuv를 내부적으로 포함

- V8 엔진: 오픈 소스 자바스크립트 엔진] -> 속도 문제 개선
- libuv: 노드의 특성인 이벤트 기반, 논블로킹 I/O 모델을 구현한 라이브러리





# 노드 설치하기

윈도(10 기준), 맥(하이 시에라 기준)

- <https://nodejs.org> 접속
- LTS 버전인 10버전 설치
- LTS는 안정된 버전, Current는 최신 버전(실험적)

Download for Windows (x64)

10.16.3 LTS

Recommended For Most Users

12.9.1 Current

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

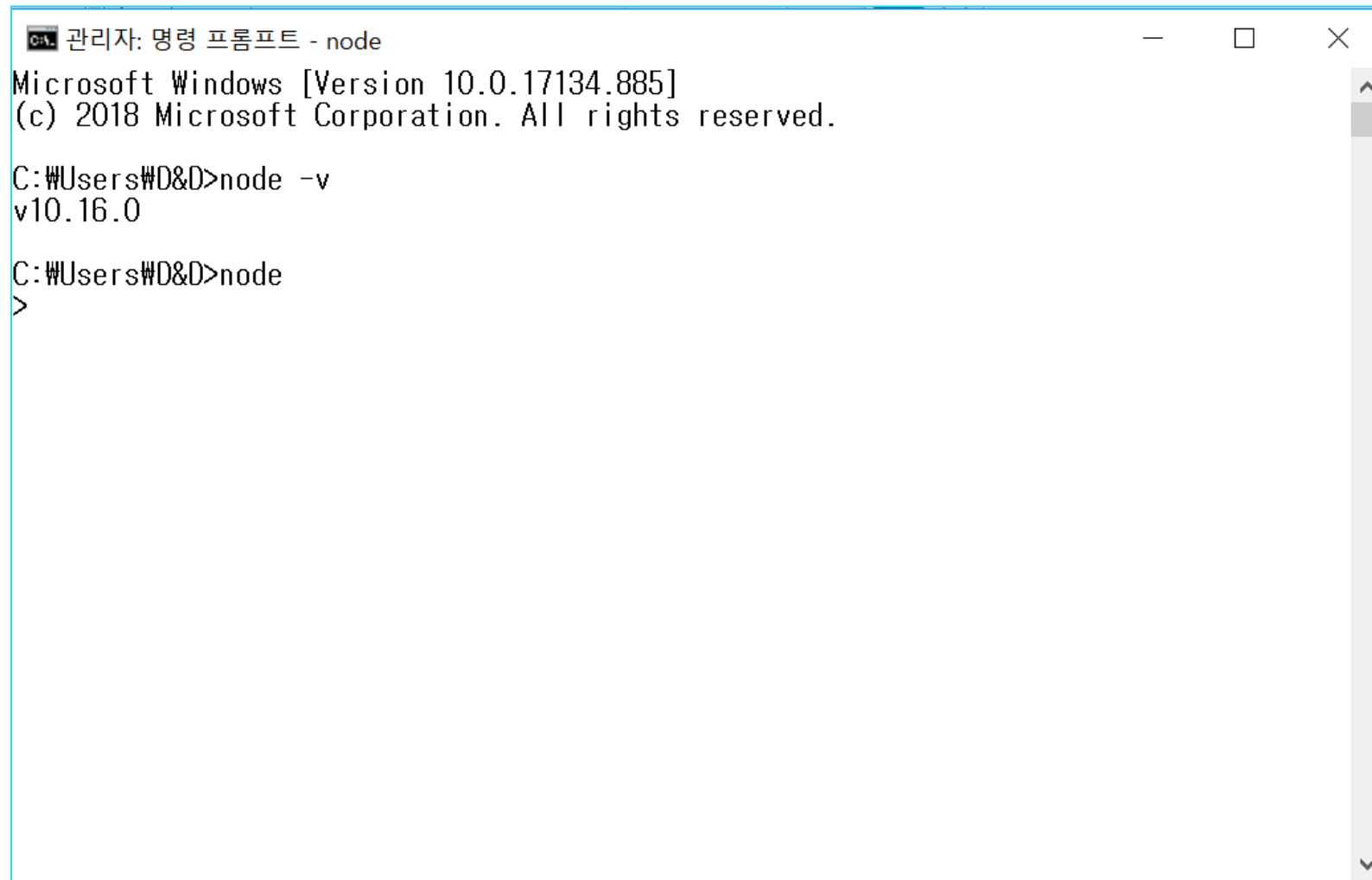
[Other Downloads](#) | [Changelog](#) | [API Docs](#)

Or have a look at the [Long Term Support \(LTS\) schedule](#).

Sign up for [Node.js Everywhere](#), the official Node.js Monthly Newsletter.



# 설치확인



A screenshot of a Windows Command Prompt window titled "관리자: 명령 프롬프트 - node". The window shows the following text:

```
Microsoft Windows [Version 10.0.17134.885]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\Users\D&D>node -v  
v10.16.0  
  
C:\Users\D&D>node  
>
```

The window has a standard Windows title bar with minimize, maximize, and close buttons. A vertical scrollbar is visible on the right side of the command prompt area.

# 실행확인 종료하기 CTRL+C, CTRL+C

```
관리자: 명령 프롬프트 - node
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

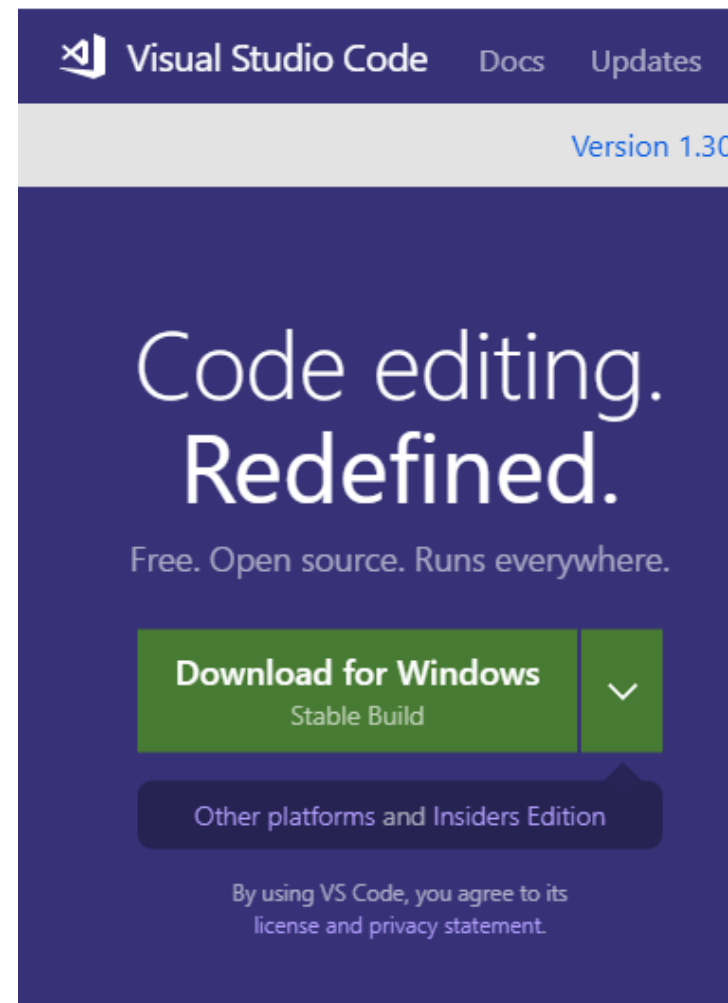
C:\Users\WD&D>node -v
v10.16.0

C:\Users\WD&D>node
> console.log(1+1)
2
undefined
>
(To exit, press ^C again or type .exit)
>
```

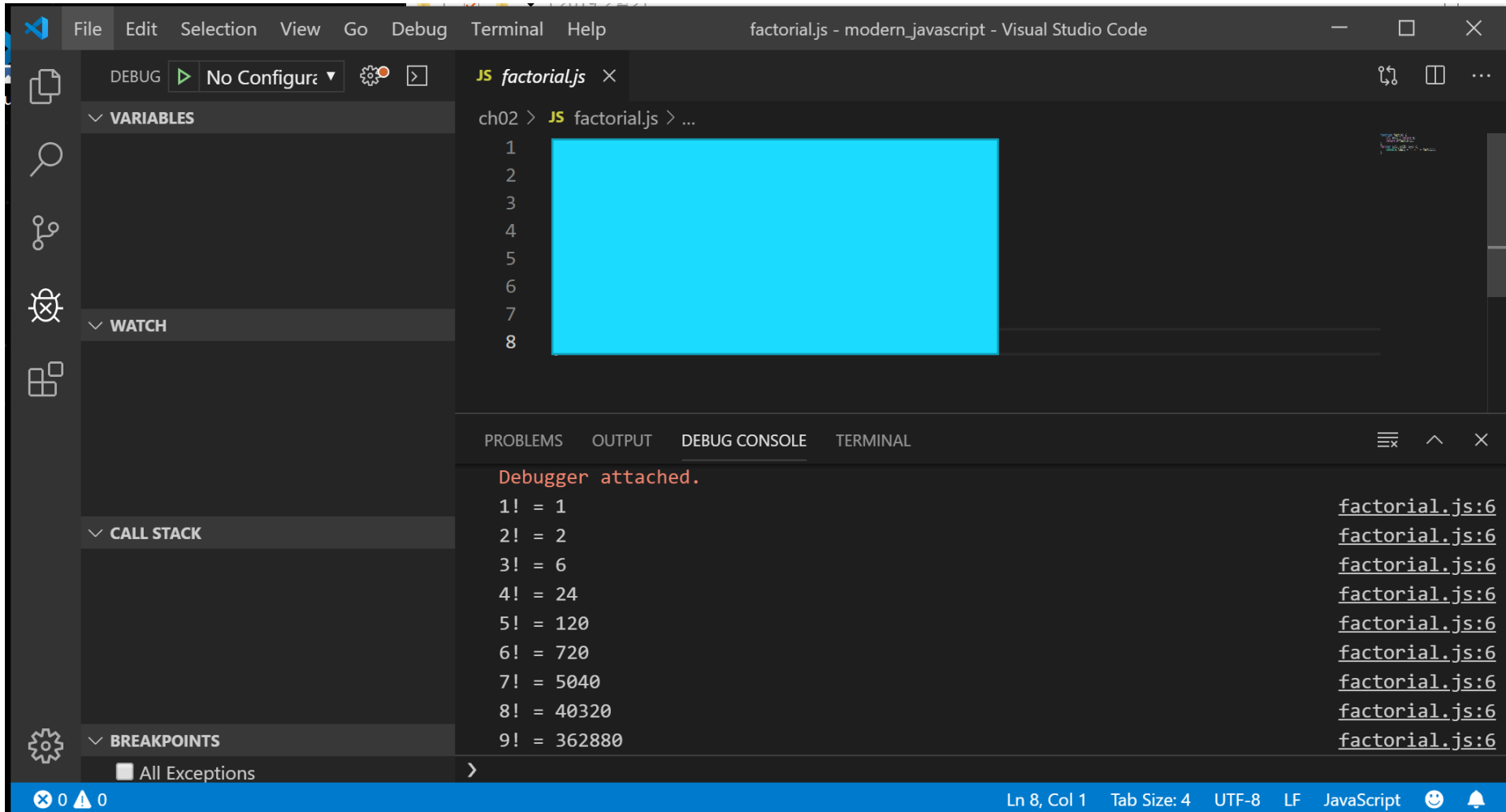
# VS CODE 설치하기

VS Code: 마이크로소프트에서 제공하는  
오픈 소스 코드 에디터

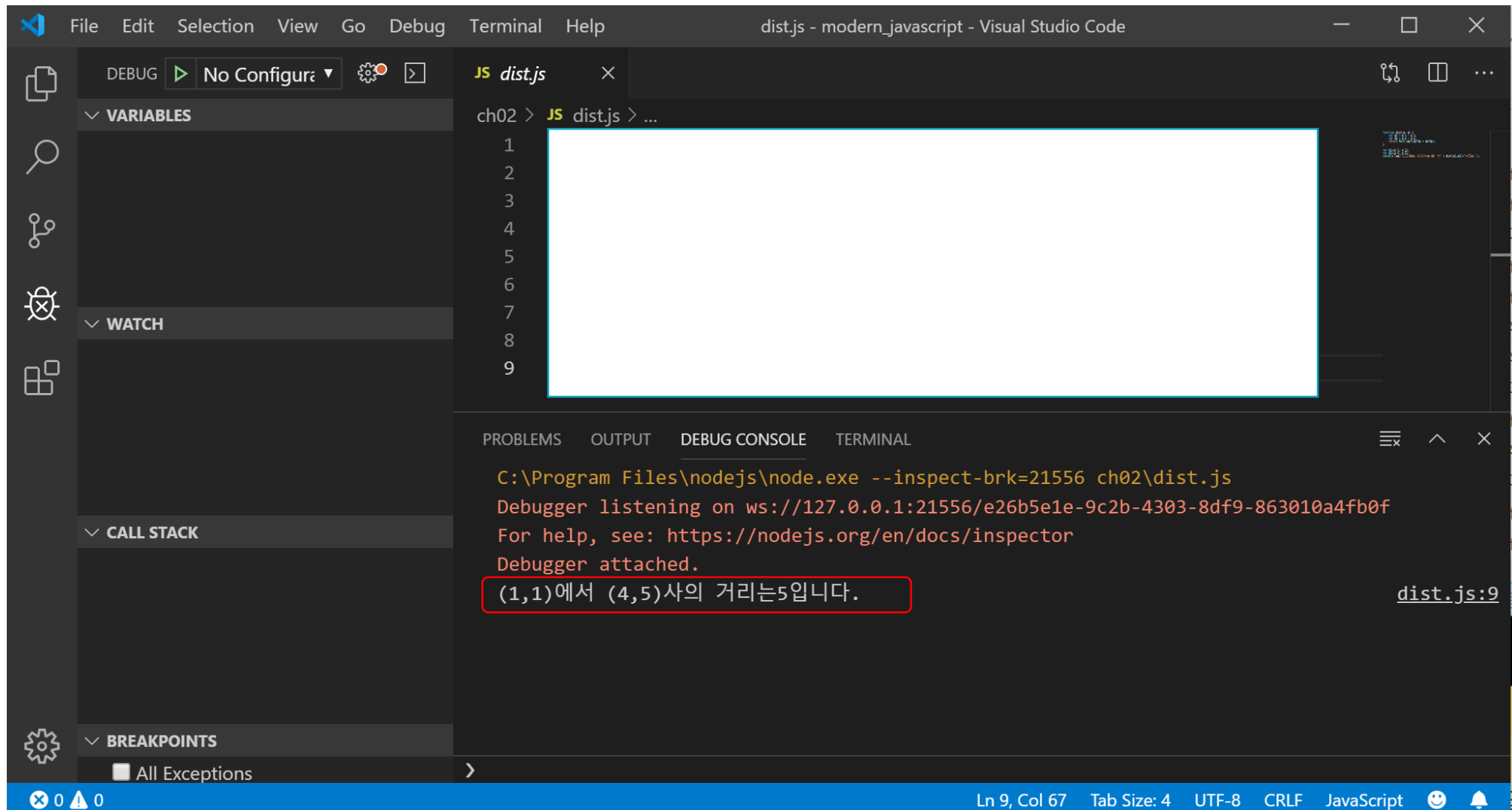
- 윈도우, 맥, 리눅스(GUI) 모두  
<https://code.visualstudio.com> 접속
- 운영체제에 맞는 파일 설치
- VS Code 외에도 취향에 맞는 코드 에디터 사용해도 됨



# 문제 1-함수



## 문제 2. 함수-객체 인수



The screenshot shows the Visual Studio Code interface with the file `dist.js` open. The editor has a dark theme. On the left sidebar, the **DEBUG** tab is active, showing the **VARIABLES**, **WATCH**, **CALL STACK**, and **BREAKPOINTS** panels. The **DEBUG CONSOLE** at the bottom displays the output of a Node.js command: `C:\Program Files\nodejs\node.exe --inspect-brk=21556 ch02\dist.js`. The output text includes "Debugger listening on ws://127.0.0.1:21556/e26b5e1e-9c2b-4303-8df9-863010a4fb0f", "For help, see: https://nodejs.org/en/docs/inspector", and "Debugger attached.". A red rectangular box highlights the Korean text `(1,1)에서 (4,5)사의 거리는5입니다.` on line 9 of `dist.js`. The status bar at the bottom indicates the current position is `Ln 9, Col 67` with a `Tab Size: 4`, `UTF-8` encoding, `CRLF` line endings, and `JavaScript` language.



# 자바스크립트에서 객체 생성법

- 객체리터럴 이용
  - { ... }
  - var p1 = {x:1, y:1};
- new 연산자 이용
  - 자바스크립트에는 클래스가 없고 생성자 함수로 선언하고
  - new 연산자로 객체 생성

```
> function Card(suit, rank){  
    this.suit = suit;  
    this.rank = rank;  
}  
  
var card1 = new Card("하트", "A");  
console.log(card1);  
  
▶ Card {suit: "하트", rank: "A"}  
  
◀ undefined  
  
> |
```

# NEW 연산자 이용해 객체 생성하기

```
function Card(suit, rank){  
    this.suit = suit;  
    this.rank = rank;  
}
```

```
var card1 = new Card("하트","A");  
console.log(card1);
```

```
> function Card(suit, rank){  
    this.suit = suit;  
    this.rank = rank;  
}  
  
var card1 = new Card("하트","A");  
console.log(card1);  
  
▶ Card {suit: "하트", rank: "A"}  
◀ undefined  
> |
```

# 객체 리터럴로 객체 생성하기

```
> var card2 = {suit:"하트", rank:"A"};
< undefined
> console.log(card2.suit);
하트
< undefined
> card2.value = 14;
< 14
> console.log(card2);
▶ {suit: "하트", rank: "A", value: 14}
```

- 객체리터럴을 이용해 card2객체 변수 생성
- 객체안의 프로퍼티 접근 연산자 : .
- 없는 프로퍼티 이름에 값을 대입하면 새로운 프로퍼티가 추가됨

[카드를 표현하는 객체]

이름	값	
suit	"하트"	<- 프로퍼티
rank	"A"	<- 프로퍼티
value	14	<- 뉴 프로퍼티 추가

# 프로퍼티 삭제

- 자바스크립트 객체는 실행 중에 프로퍼티를 자유롭게 추가하거나 삭제가 가능(자바 등에서는 불가능)
- delete 연산자 사용

```
delete card2.rank;  
console.log(card2);
```

```
14 delete card2.rank;  
15 console.log(card2);  
16  
> Object {suit: "하트", rank: "2", value: 14}  
> Object {suit: "하트", value: 14}
```

# 프로퍼티 확인

- 객체에 특정 프로퍼티가 있는지 확인 : in 연산자
  - 존재여부에 따라 true or false 반환

형식) **프로퍼티 이름을 뜻하는 문자열 in 객체명**

```
var card = {suit:"하트", rank:"A"};  
console.log("suit" in card);  
console.log("color" in card);
```

```
1  var card = {suit:"하트", rank:"A"};  
2  console.log("suit" in card);  
3  console.log("color" in card);  
4  |
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
C:\Program Files\nodejs\node.exe --inspect-br  
Debugger listening on ws://127.0.0.1:43597/d4  
For help, see: https://nodejs.org/en/docs/ins  
Debugger attached.  
true  
false
```



# 복습문제

- 다음과 같은 속성을 갖는 객체를 생성하자.
- 다음과 같이 콘솔 출력을 해보자.

이름 : 홍길동

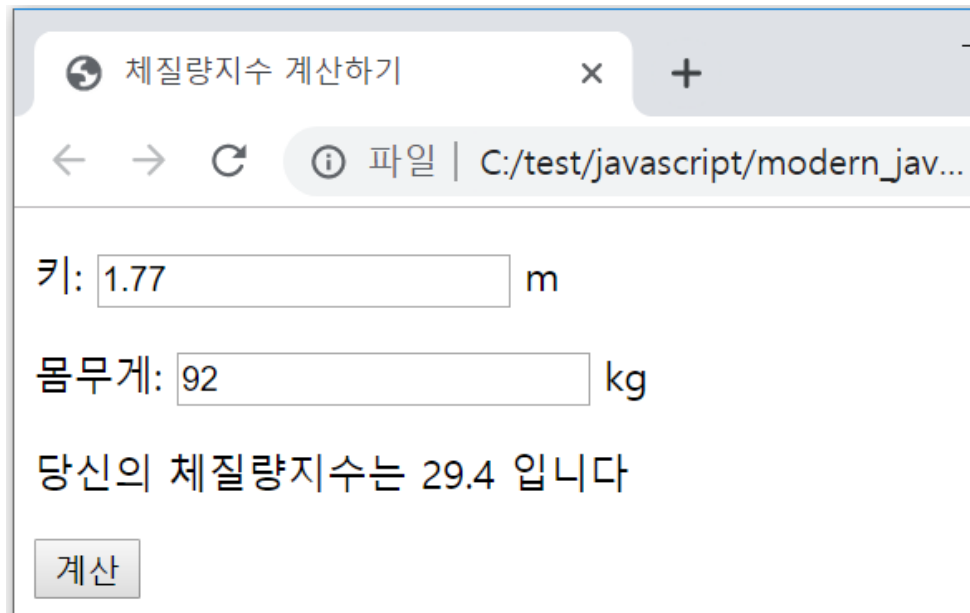
성별 : 남

홍길동 님의 5년후 나이는 28 입니다.

- 결혼여부를 확인하는 프로퍼티를 추가하자.(홍길동은 미혼)
- 최종 수정된 person객체를 출력해보자.

[person 객체]	
이름	값
name	홍길동
age	23
sex	남

# 문제3-DOM을 사용해 이벤트처리

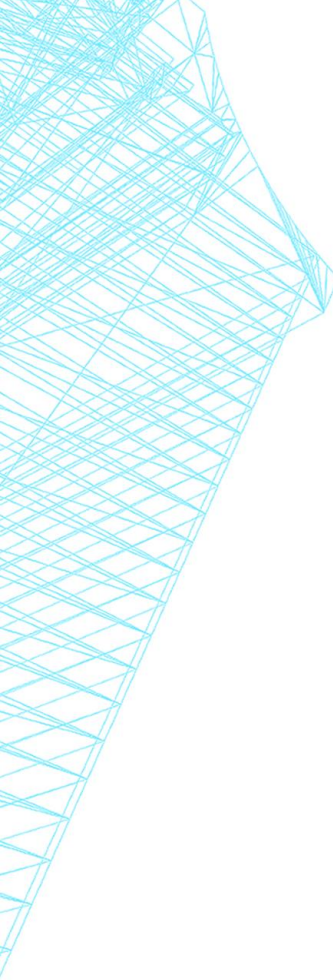


A screenshot of a web browser window. The browser has a single tab titled '체질량지수 계산하기' (BMI Calculator). The address bar shows the file path 'C:/test/javascript/modern\_jav...'. The page content includes two input fields: '키: 1.77 m' (Height: 1.77 m) and '몸무게: 92 kg' (Weight: 92 kg). Below these fields, a text line states '당신의 체질량지수는 29.4 입니다' (Your BMI is 29.4). At the bottom left, there is a button labeled '계산' (Calculate).



# DOM을 사용해 이벤트처리기 등록

- DOM(Document Object Model)은 자바스크립트 등의 프로그램이 HTML 요소를 조작할 수 있게 하는 인터페이스
- DOM을 사용해 이벤트처리기 등록하는 전형적인 방법
  - 1) `window.onload` 를 사용해 HTML문서를 다 읽어들인 후 2)와3)을 실행
  - 2) `document.getElementById` 메서드를 사용해 특정 id속성 값을 가진 HTML요소의 객체를 가져온다.
  - 3) 요소 객체의 이벤트 처리기 프로퍼티에 이벤트 처리기로 동작할 함수를 등록한다.



```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>시각을 콘솔에 표시하기</title>
  <script>
    function displayTime() {
      var d = new Date();
      console.log("현재 시각은 " + d.toLocaleString() + " 입니다.") ;
    }
    // ① Window 객체의 onload 프로퍼티에 함수를 저장한다
    window.onload = function() {
      // ② input 요소의 객체 가져오기
      var button = document.getElementById("button");
      // ③ input 요소를 클릭했을 때 동작하는 이벤트 처리기를 등록한다
      button.onclick = displayTime;
    };
  </script>
</head>
<body>
  <input type="button" value="click" id="button">
</body>
</html>
```

# HTML요소의 innerHTML 프로퍼티로 읽고 쓰기

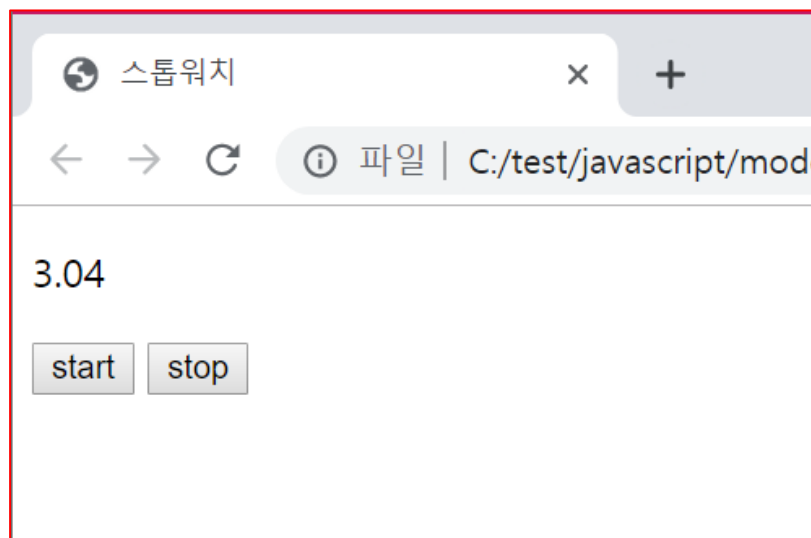
- 요소객체의 **innerHTML** 프로퍼티
  - 그 HTML요소의 내용을 가리키며
  - 이로써 HTML요소의 내용을 읽거나 쓸수 있다.





# 복습문제

- start버튼을 누르면 0.01 초마다 경과한 시간을 표시
- 타이머함수 활용
  - setInterval()함수 : 지정한 시간마다 반복실행하는 함수
  - clearInterval()함수 : 타이머를 중지하는 함수



# 타이머 함수 예제

5초마다 경고 메시지를 콘솔에 출력하는 예제

```
var time = 0;
playAlert = setInterval(function() {
    time+=5;
    console.log(time,"초 경과");
    if(time==20) clearInterval(playAlert);
}, 5000);
```

```
C:\Program Files\nodejs\node.exe --ir
Debugger listening on ws://127.0.0.1:
For help, see: https://nodejs.org/en/
Debugger attached.
5 초 경과
10 초 경과
15 초 경과
20 초 경과
```