

삼육대학교 SW융합교육원 김진호

# 자바스크립트 기초교육

# 01. JavaScript 개요

1-1. JavaScript의 역사

1-2. JavaScript의 성장

1-3. JavaScript의 특징

1-4. 개발환경구축

# ❖ 1. JavaScript 개요

## 1-1. JavaScript의 역사

JavaScript는 웹 페이지의 보조적인 기능을 수행하기 위해 브라우저에서 동작하는 경량 프로그래밍 언어로서 1996년 넷스케이프 커뮤니케이션즈에서 처음 개발되었다. 이어 마이크로소프트에서도 JavaScript의 파생 버전인 JScript를 만들었는데 두 회사는 자사 브라우저의 시장 점유율을 높이기 위해 자사 브라우저에서만 동작하는 기능을 경쟁적으로 추가했다. 이는 **크로스 브라우징 이슈**-브라우저에 따라 웹 페이지가 정상적으로 동작하지 않는 현상-을 야기했고 표준화된 자바스크립트의 필요성이 대두되었다.

컴퓨터 시스템의 표준을 관리하는 비영리 표준화 기구인 ECMA 인터네셔널에서 1997년 ECMA-262라 불리는 표준화 된 자바스크립트 초판(ECMAScript 1) 사양이 완성되었고, 상표권 문제로 자바스크립트는 ECMAScript로 명명되었다. 이후 1999년 ECMAScript 3(ES3)이 공개 되고, 10년만인 2009년에 출시된 **ECMAScript 5(ES5)**는 **HTML5**와 함께 출현한 표준 사양이다. 2015년에 공개된 **ECMAScript 6(ECMAScript 2015, ES6)**는 **let/const** 키워드, 화살표 함수, 클래스 등과 같이 범용 프로그램이 언어로서 갖춰야 할 기능들을 대거 도입하는 큰 변화가 있었다.

ECMAScript는 자바스크립트 표준 사양으로 프로그래밍 언어의 값, 타입, 객체와 프로퍼티, 함수, 표준 빌트인 객체 등 핵심 문법을 규정한다. 각 브라우저 제조사는 ECMAScript 사양을 준수해서 브라우저에 내장되는 자바스크립트 엔진을 구현한다. 자바스크립트는 일반적으로 프로그래밍 언어로서 기본 뼈대를 이루는 ECMAScript와 브라우저가 별도 지원하는 클라이언트 사이드 Web API(DOM, BOM, XMLHttpRequest 등등)를 아우르는 개념이다.

# ❖ 1. JavaScript 개요

## 1-2. JavaScript의 성장

### ■ Ajax(1999년)

- 자바스크립트를 이용해 서버와 브라우저가 비동기 방식으로 데이터를 교환할 수 있는 통신 기능
- 전체 웹 페이지 렌더링-HTML, CSS, JavaScript로 작성 된 문서를 해석해서 브라우저에 시각적으로 출력하는 것-이 아닌 필요한 데이터만 한정적으로 렌더링하여 빠른 성능과 부드러운 화면 전환이 가능해졌다.

### ■ jQuery(2006년)

- 번거롭던 DOM을 쉽게 제어할 수 있게 되었고 크로스 브라우징 이슈에도 많은 도움을 주었다.

### ■ V8 자바스크립트 엔진(2008년)

- 더욱 빠르게 동작하는 구글 V8 자바스크립트 엔진의 개발로 데스크톱 애플리케이션과 유사한 사용자 경험을 제공할 수 있는 웹 어플리케이션 프로그래밍 언어로 정착하게 되었다.
- 웹 서버에서 수행되던 로직이 클라이언트(브라우저)로 이동하는 등 프론트엔드 영역이 주목받는 계기가 되었다.

# ❖ 1. JavaScript 개요

## 1-2. JavaScript의 성장

### ■ Node.js(2009년)

- 브라우저의 자바스크립트 엔진에서만 동작하던 자바스크립트를 브라우저 이외의 환경에서도 동작할 수 있도록 자바스크립트 엔진을 브라우저에서 독립시킨 자바스크립트 런타임 환경이다.
- 서버 사이드 애플리케이션 개발에 주로 사용되며 이에 필요한 내장 API를 제공한다.
- 이제 자바스크립트는 프론트엔드는 물론 백엔드 영역까지 아우르는 웹 프로그래밍 언어의 표준으로 자리잡고 있다.

### ■ SPA 프레임워크

- 모던 웹 어플리케이션은 개발 규모와 복잡도가 날로 상승했으며 복잡해진 개발 과정 수행을 위해 많은 프레임워크가 등장했다.
- CDB(Component based development) 방법론을 기반으로 하는 SPA(Single Page Application)가 대중화 되면서 Angular, React, Vue.js 등 다양한 SPA 프레임워크/라이브러리가 많은 사용층을 확보하고 있다.

# ❖ 1. JavaScript 개요

## 1-3. JavaScript의 특징

- 웹 브라우저에서 동작하는 유일한 프로그래밍 언어이다.
  - 기본 문법은 C, Java와 유사하지만 프로토타입 기반, 일급 함수의 개념 등은 다른 언어에서 차용했다.
- 개발자가 별도의 컴파일 작업을 수행하지 않는 인터프리터 언어이다.
  - 인터프리터 언어 : 코드가 실행되는 단계인 런타임에 문 단위로 한 줄씩 중간 코드인 바이트코드로 변환한 후 실행하는 언어
  - 대부분의 모던 자바스크립트 엔진은 인터프리터와 컴파일러의 장점을 결합해 비교적 처리 속도가 느린 인터프리터의 단점을 해결했다.
- 클래스 기반 객체지향 언어보다 효율적이면서 강력한 프로토타입 기반의 객체지향 언어이다.

# ❖ 1. JavaScript 개요

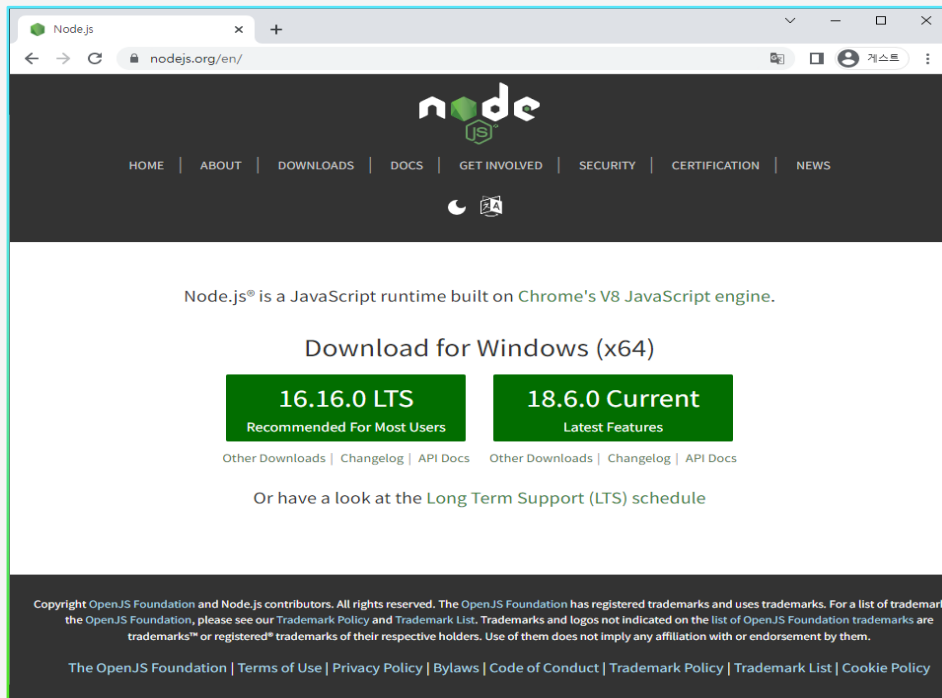
## 1-4. 개발환경구축

- 자바스크립트는 브라우저 환경 또는 Node.js 환경에서 실행할 수 있다.
  - 두 환경 모두 자바스크립트의 코어인 ECMAScript는 실행할 수 있다.
  - 브라우저는 파싱 된 HTML 요소를 선택하거나 조작하는 기능의 집합인 DOM API를 기본적으로 제공하지만 브라우저 외부에서 자바스크립트 개발 환경을 제공하는 것이 주 목적인 Node.js는 DOM API를 제공하지 않는다.
  - 반대로 Node.js에서는 파일 시스템을 기본 제공하지만 브라우저는 보안상의 이유로 이를 제공하지 않는다.
- 다양한 웹 브라우저 환경 중 ECMAScript 사양을 준수하며 시장 점유율도 높은 구글 Chrome 브라우저를 사용한다.
  - 개발자 도구, 콘솔, 디버깅 기능 등을 활용한다.

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- 간단한 개발은 브라우저만으로도 가능하지만 프로젝트 규모가 커 다양한 프레임워크/라이브러리 도입이나 여러 도구를 사용이 필요하다면 Node.js 환경을 사용한다.
  - Node.js(<https://nodejs.org/>) 접속
  - LTS(Long Term Support) 버전 다운로드 후 기본 설정으로 설치





# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- 설치 완료 후 cmd 창에서 `node -v`, `npm -v` 입력하여 Node.js와 함께 설치 된 자바스크립트 패키지 매니저인 npm(node package manager)의 버전을 출력해 정상적으로 설치되었는지 확인

```
C:\W>node -v
v16.16.0

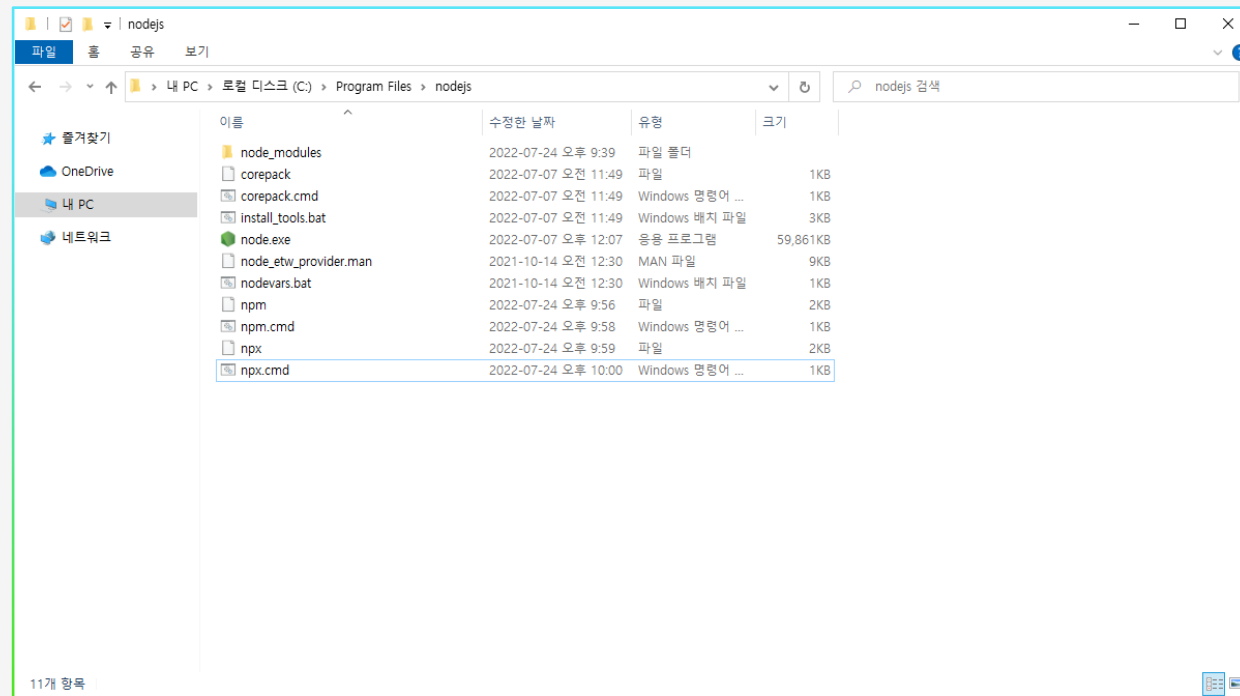
C:\W>npm -v
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.
8.11.0

C:\W>
```

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

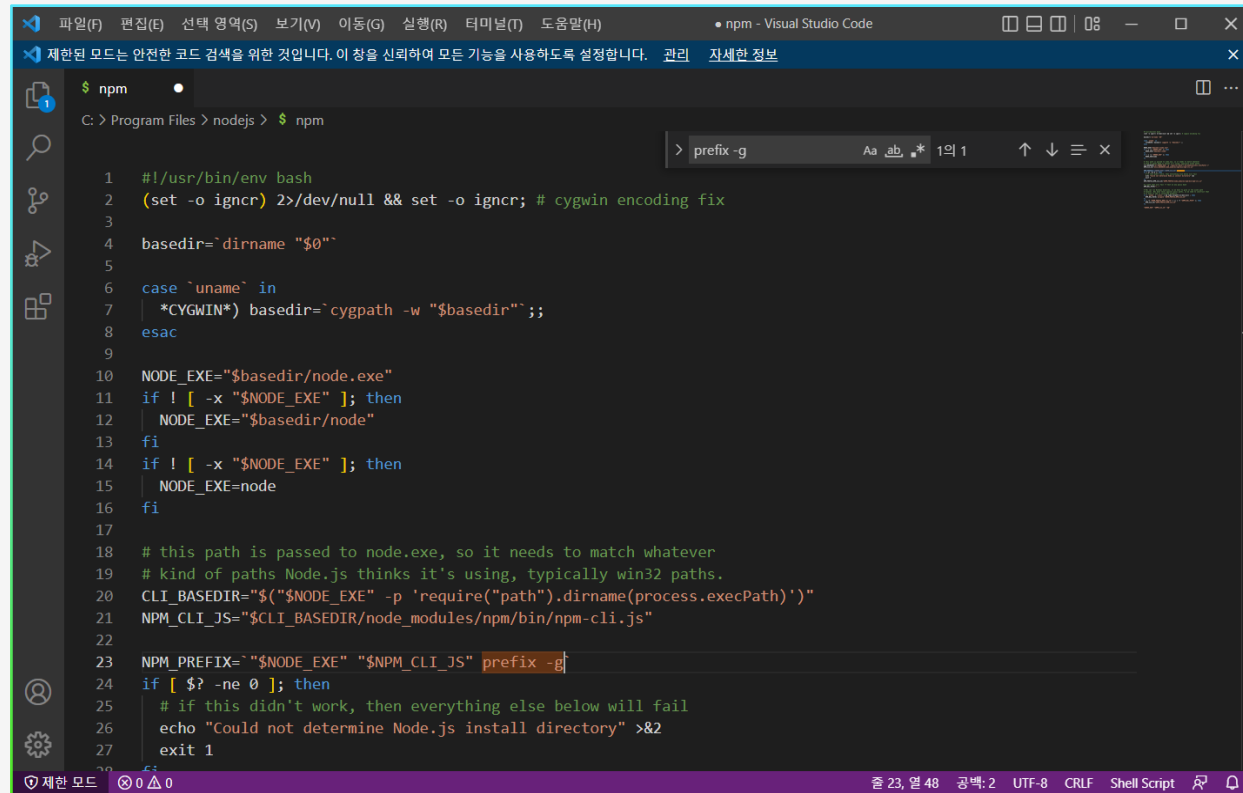
- npm WARN config global **--global**, **--local** are deprecated. Use **--location=global** instead.  
와 같은 오류를 해결하려면 C:\Program Files\nodejs 에 있는 설정 파일  
npm, npm.cmd, npx, npx.cmd을 수정



# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- ▣ VS Code 편집기로 해당 파일을 열고 **prefix-g** 를 검색하여 해당 부분을 **prefix --location=global** 로 수정

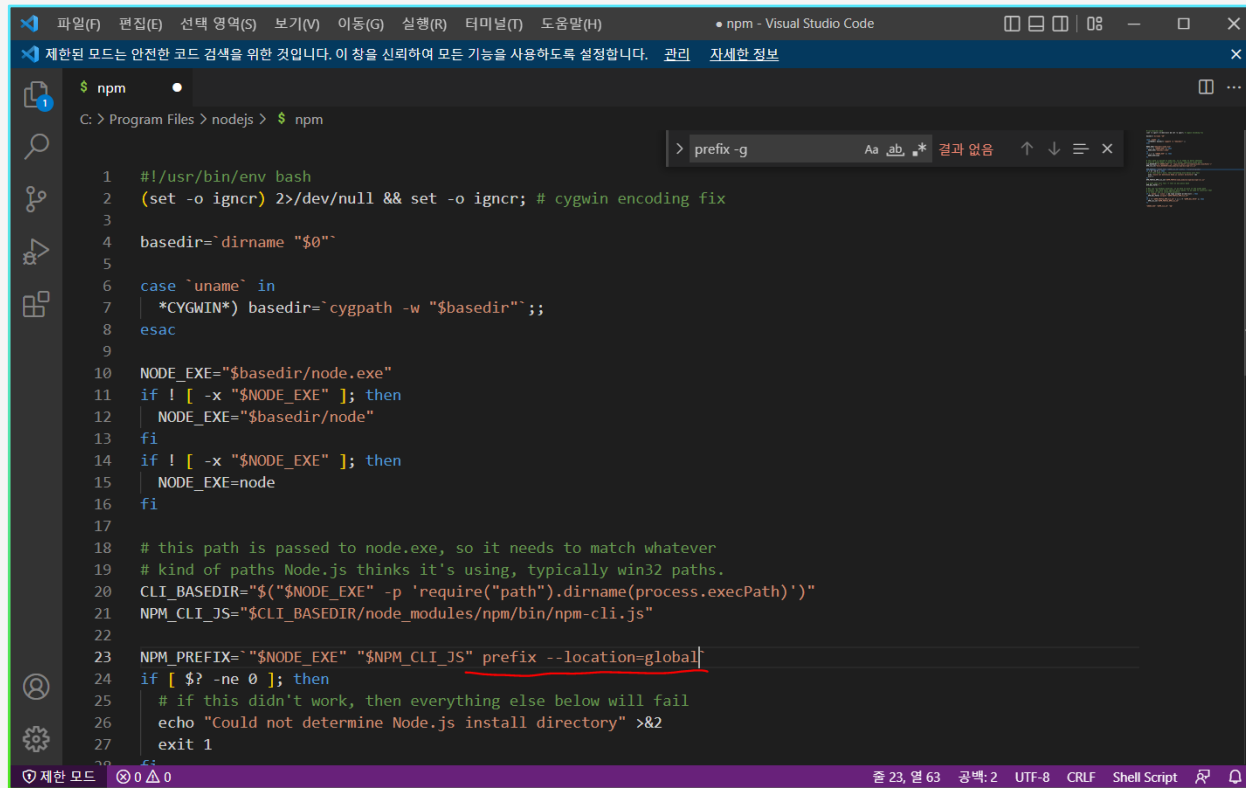


```
1  #!/usr/bin/env bash
2  (set -o igncr) 2>/dev/null && set -o igncr; # cygwin encoding fix
3
4  basedir=`dirname "$0"`
5
6  case `uname` in
7  *CYGWIN*) basedir=`cygpath -w "$basedir"`;;
8  esac
9
10 NODE_EXE="$basedir/node.exe"
11 if ! [ -x "$NODE_EXE" ]; then
12   NODE_EXE="$basedir/node"
13 fi
14 if ! [ -x "$NODE_EXE" ]; then
15   NODE_EXE=node
16 fi
17
18 # this path is passed to node.exe, so it needs to match whatever
19 # kind of paths Node.js thinks it's using, typically win32 paths.
20 CLI_BASEDIR="$("$NODE_EXE" -p 'require("path").dirname(process.execPath)')
21 NPM_CLI_JS="$CLI_BASEDIR/node_modules/npm/bin/npm-cli.js"
22
23 NPM_PREFIX=`"$NODE_EXE" "$NPM_CLI_JS" prefix -g`
24 if [ $? -ne 0 ]; then
25   # if this didn't work, then everything else below will fail
26   echo "Could not determine Node.js install directory" >&2
27   exit 1
28 fi
```

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- ▣ 저장 실패 시 하단의 관리자 모드로 저장 클릭하여 저장



```
1  #!/usr/bin/env bash
2  (set -o igncr) 2>/dev/null && set -o igncr; # cygwin encoding fix
3
4  basedir=`dirname "$0"`
5
6  case `uname` in
7  *CYGWIN*) basedir=`cygpath -w "$basedir"`;;
8  esac
9
10 NODE_EXE="$basedir/node.exe"
11 if ! [ -x "$NODE_EXE" ]; then
12   NODE_EXE="$basedir/node"
13 fi
14 if ! [ -x "$NODE_EXE" ]; then
15   NODE_EXE=node
16 fi
17
18 # this path is passed to node.exe, so it needs to match whatever
19 # kind of paths Node.js thinks it's using, typically win32 paths.
20 CLI_BASEDIR="$("$NODE_EXE" -p 'require("path").dirname(process.execPath)')"
21 NPM_CLI_JS="$CLI_BASEDIR/node_modules/npm/bin/npm-cli.js"
22
23 NPM_PREFIX="$NODE_EXE" "NPM_CLI_JS" prefix --location=global
24 if [ $? -ne 0 ]; then
25   # if this didn't work, then everything else below will fail
26   echo "Could not determine Node.js install directory" >&2
27   exit 1
28 fi
```

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- ▣ 수정 후 다시 cmd 창에서 확인하면 오류 발생하지 않음

```
명령 프롬프트
C:\W>node -v
v16.16.0

C:\W>npm -v
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.
8.11.0

C:\W>npm -v
8.11.0

C:\W>
```

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- cmd에서 **node**라는 명령어를 실행하면 프롬프트가 **>**로 변경되고 자바스크립트 코드를 실행해볼 수 있다.

```
선택 명령 프롬프트 - node
C:\W>node -v
v16.16.0

C:\W>npm -v
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.
8.11.0

C:\W>npm -v
8.11.0

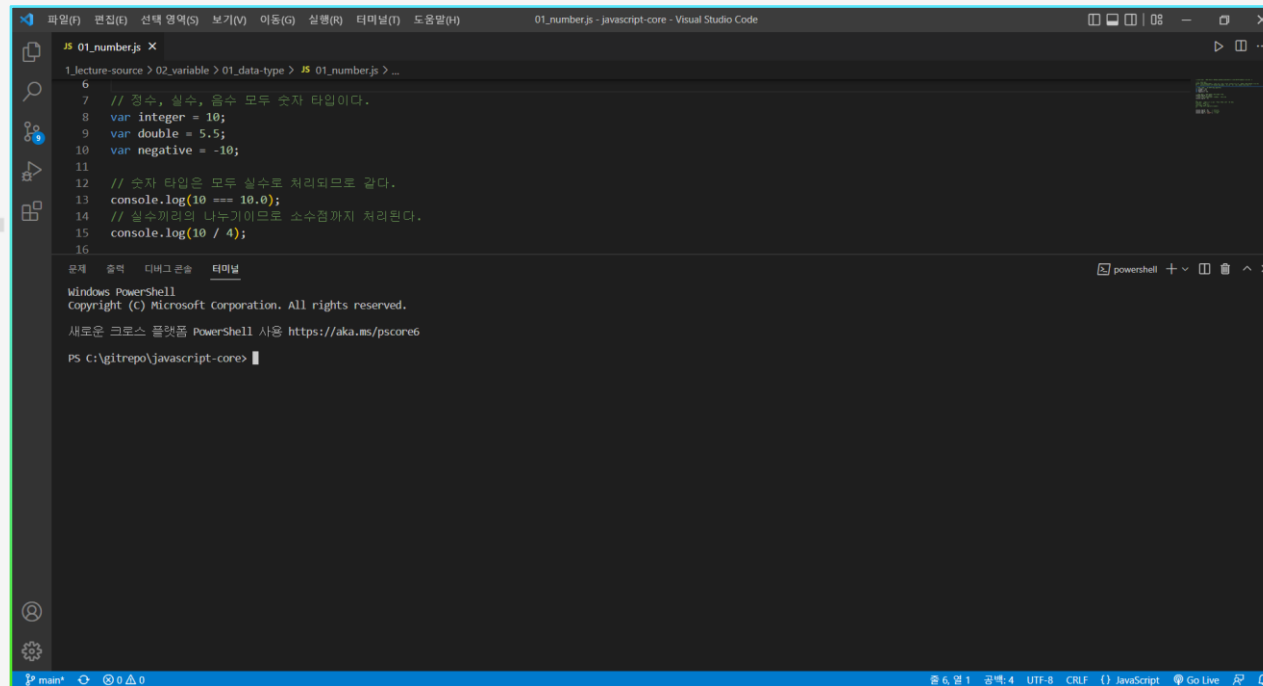
C:\W>node
Welcome to Node.js v16.16.0.
Type ".help" for more information.
> 1 + 2
3
>
```

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

### ■ VS Code

- 브라우저의 콘솔, Node.js의 REPL(Read Eval Print Loop)에서 자바스크립트 코드를 실행할 수 있지만 애플리케이션 개발을 위해서는 코드 에디터의 기능이 필요하므로 VS Code를 통해 작업한다.
- VS Code에서 제공하는 내장 터미널은 Ctrl + `를 누르면 열린다.



The screenshot shows the Visual Studio Code interface. The editor window displays a file named `01_number.js` with the following content:

```
1 // 정수, 실수, 음수 모두 숫자 타입이다.
2
3 var integer = 10;
4 var double = 5.5;
5 var negative = -10;
6
7 // 숫자 타입은 모두 실수로 처리되므로 같다.
8 console.log(10 === 10.0);
9 // 실수끼리의 나누기이므로 소수점까지 처리한다.
10 console.log(10 / 4);
11
```

Below the editor, the integrated terminal is open, showing the Windows PowerShell prompt. The terminal text includes the Windows PowerShell copyright notice and the current directory path:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/powershell

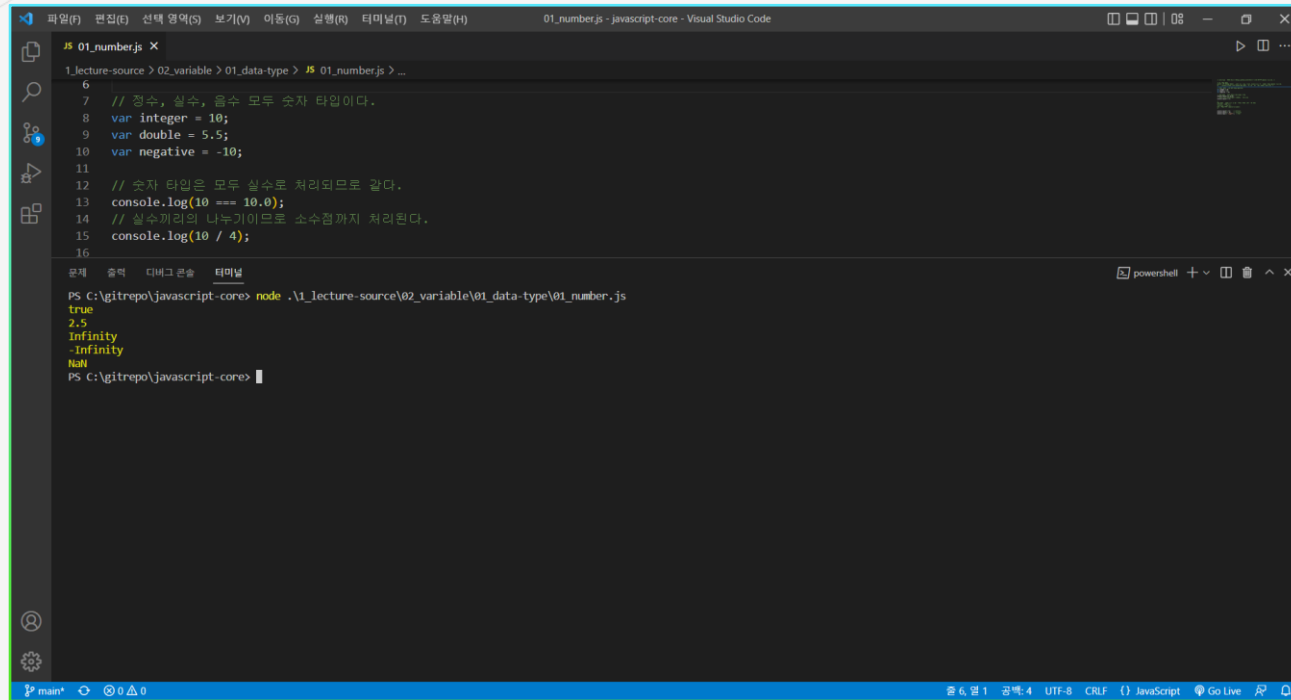
PS C:\gitrepo\javascript-core>
```

The status bar at the bottom indicates the file is on the `main` branch, using `UTF-8` encoding, and the file type is `JavaScript`.

# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- 프롬프트에 **node** 파일명을 입력하여 자바스크립트 파일을 실행할 수 있다.



The screenshot shows the Visual Studio Code interface with a file named `01_number.js` open. The code in the editor is as follows:

```
1_lecture-source > 02_variable > 01_data-type > JS 01_number.js > ...
6
7 // 정수, 실수, 음수 모두 숫자 타입이다.
8 var integer = 10;
9 var double = 5.5;
10 var negative = -10;
11
12 // 숫자 타입은 모두 실수로 처리되므로 같다.
13 console.log(10 === 10.0);
14 // 실수끼리의 나누기이므로 소수점까지 처리된다.
15 console.log(10 / 4);
16
```

Below the editor, the terminal window shows the command `node .\1_lecture-source\02_variable\01_data-type\01_number.js` being executed, with the following output:

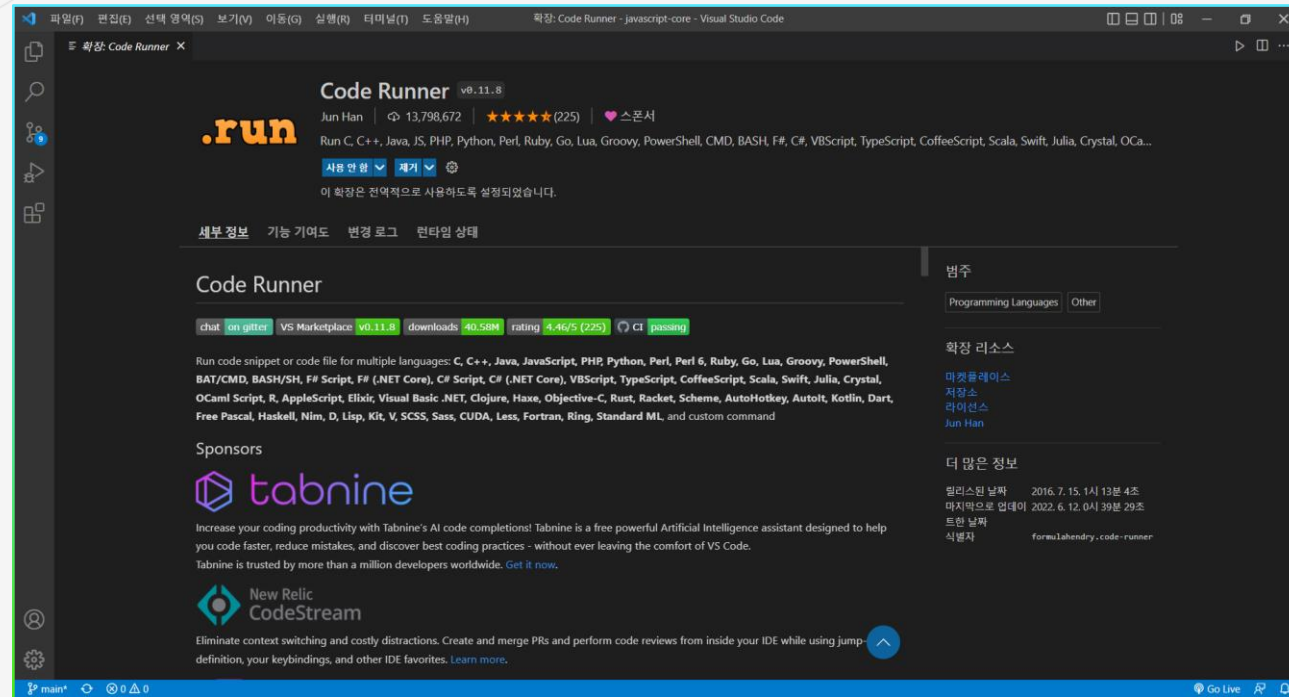
```
PS C:\gitrepo\javascript-core> node .\1_lecture-source\02_variable\01_data-type\01_number.js
true
2.5
Infinity
-Infinity
NaN
PS C:\gitrepo\javascript-core>
```



# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

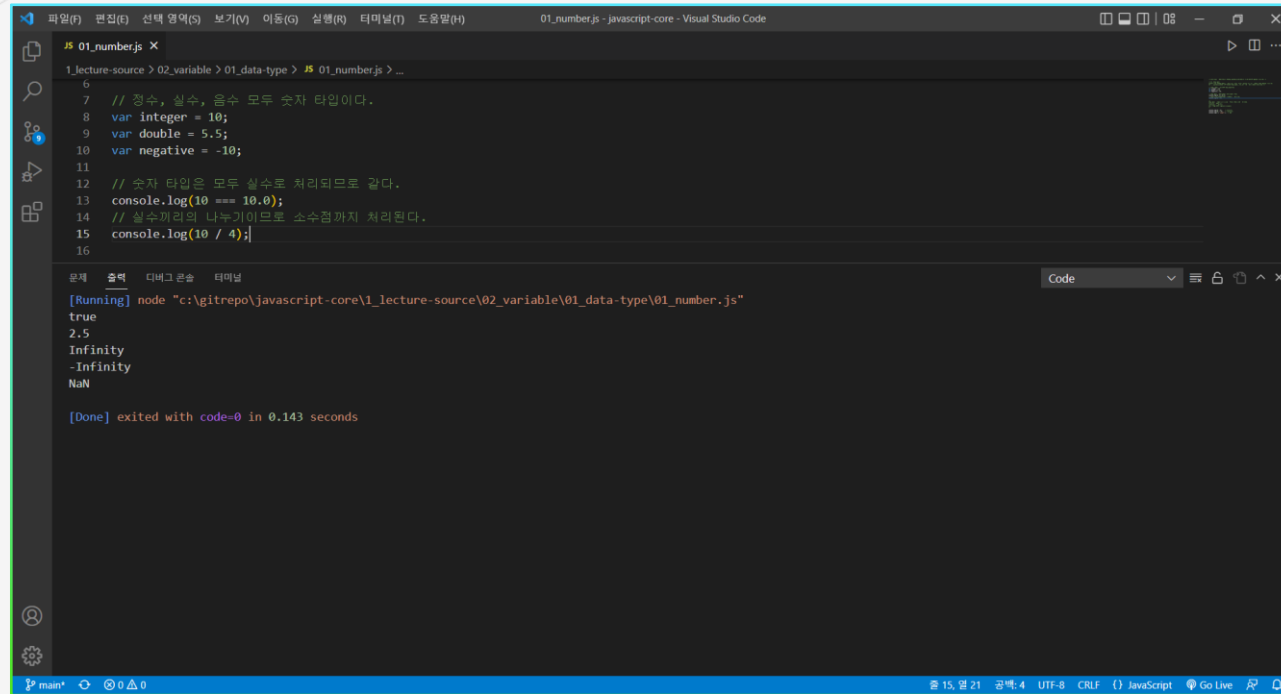
- ▣ 확장(Extension)에서 Code Runner를 Install



# ❖ 1. JavaScript 개요

## 1-4. 개발환경구축

- Ctrl + Alt + N 을 누르면 현재 표시 중인 자바스크립트 파일을 실행



```
01_number.js
1_lecture-source > 02_variable > 01_data-type > 01_number.js > ...
6
7 // 정수, 실수, 음수 모두 숫자 타입이다.
8 var integer = 10;
9 var double = 5.5;
10 var negative = -10;
11
12 // 숫자 타입은 모두 실수로 처리되므로 같다.
13 console.log(10 === 10.0);
14 // 실수끼리의 나눗기이므로 소수점까지 처리된다.
15 console.log(10 / 4);
16

문서  출력  디버그 콘솔  터미널
[Running] node "c:\gitrepo\javascript-core\1_lecture-source\02_variable\01_data-type\01_number.js"
true
2.5
Infinity
-Infinity
NaN

[Done] exited with code=0 in 0.143 seconds
```

# ❖ 1. JavaScript 개요

## 수업 참고 자료

▣ <https://github.com/2024-sahmyook-javascript-video-lecture>