

## ■ 자바스크립트를 공부해야 하는 이유

### ● 모든 웹 개발자가 배워야 하는 3가지 언어 중 하나

- 웹 페이지의 내용을 정의하는 HTML
- 웹 페이지의 레이아웃을 지정하는 CSS
- 웹 페이지의 동작을 정의하는 JavaScript

### ● 데스크탑 프로그램 개발

- VSCode, Skype, WebTorrent, Atom, WordPress, MS Teams, WhatsApp 등

### ● 서버 프로그램 개발

- Node.js

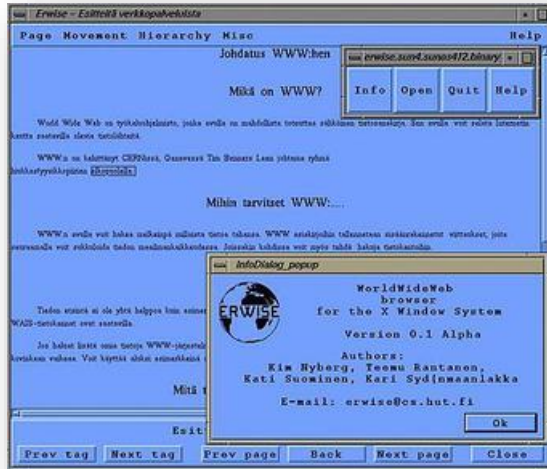
### ● 데이터베이스 제어

- MongoDB, CouchDB 등

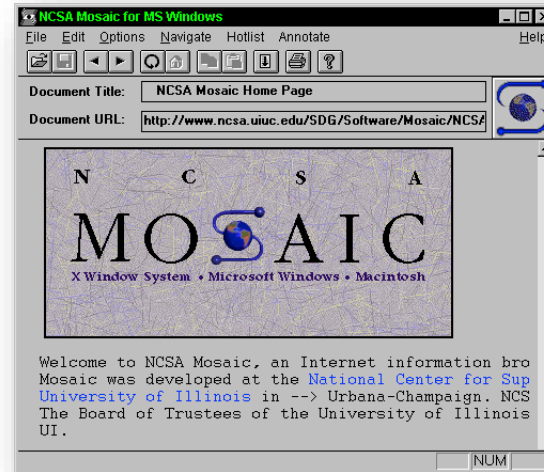
## ■ 자바스크립트 History

- 최초의 웹 브라우저 (1990년) : Nexus (팀 버너스 리)

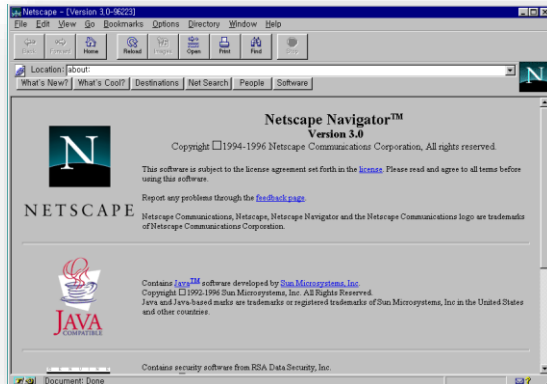
- GUI (1992년) : Erwise



- 멀티미디어 (1993년) : MOSAIC



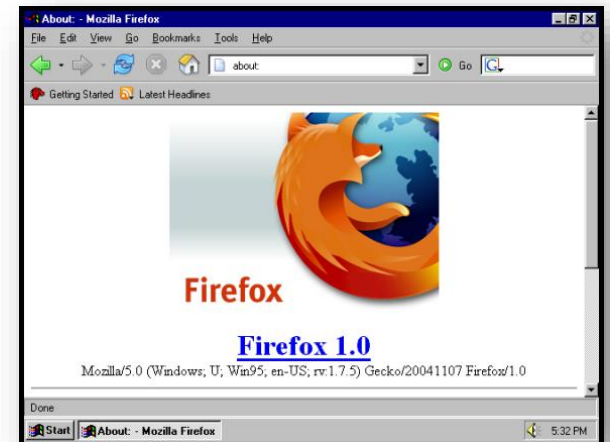
- Netscape Navigator (1994년)



- Brendan Eich가 10일만에 Mocha라는 스크립트 언어 개발
- 이후 LiveScript → JavaScript 명칭 변경

## ■ 자바스크립트 History

- Internet Explorer (1995년)
  - Netscape 소스코드 Reverse Engineering
  - JScript 내장
- ECMAScript 1 제정 (1997년)
- MS IE 점유율 95% (2000년)
- ECMAScript 4 이후로 표준화 진행 중단
- Mozilla FireFox 출시 (2004년)
  - ActionScript 내장
- AJAX 도입 (2004년)
- Google Chrome 출시 (2008년)
- 브라우저 개발사 간 커뮤니티 형성
  - WHAT WG
- ECMAScript 5 제정 (2009년)
- ECMAScript 6 제정 (2015년) ➔ 매년 새로운 버전 출시






## ■ ECMAScript

버전	공식 명칭	특징
1	ECMAScript 1 (1997)	최초 개발
2	ECMAScript 2 (1998)	-
3	ECMAScript 3 (1999)	정규식, 예외 처리
4	ECMAScript 4 (2008)	-
5	ECMAScript 5 (2009)	StrictMode, JSON, Array Functions
5.1	ECMAScript 5.1 (2011)	-
6	ECMAScript 2015	let/const, Arrow Function, Class
7	ECMAScript 2016	지수 연산(**)
8	ECMAScript 2017	Async Functions
9	ECMAScript 2018	비동기 반복, Promise.finally
	...	
11	ECMAScript 2020	

## ■ Web APIs

- 웹 브라우저에서 제공하는 기능
- 자바스크립트 문법으로 작성
- <https://developer.mozilla.org/ko/docs/Web/API>

<b>A</b> Ambient Light Events	<b>F</b> Fetch API  File System API Frame Timing API Fullscreen API	Media Session API Media Source Extensions MediaStream Recording	Storage Access API Streams 
<b>B</b> Background Tasks Battery API Beacon Bluetooth API Broadcast Channel API	<b>G</b> Gamepad API  <b>Geolocation API</b>	<b>N</b> Navigation Timing Network Information API	<b>T</b> Touch Events
<b>C</b> CSS Counter Styles CSS Font Loading API CSSOM Canvas API Channel Messaging API <b>Console API</b> Credential Management API	<b>H</b> HTML Drag and Drop API High Resolution Time History API	<b>P</b> Page Visibility API Payment Request API Performance API Performance Timeline API Permissions API Pointer Events Pointer Lock API Proximity Events Push API	<b>U</b> URL API
<b>D</b> <b>DOM</b>	<b>I</b> Image Capture API IndexedDB Intersection Observer API	<b>R</b> Resize Observer API Resource Timing API	<b>V</b> Vibration API
	<b>L</b> Long Tasks API		<b>W</b> Web Animations Web Audio API Web Authentication API Web Crypto API Web Notifications Web Storage API Web Workers API WebGL WebRTC

- 브라우저 점유율

<https://gs.statcounter.com/browser-version-market-share/desktop/south-korea>

## ■ 자바스크립트 적용 방법

### ● HTML 문서 내 <script>

```
<!DOCTYPE html>
<html>
<head></head>
<body>
  <p id="demo">text</p>
  <button type="button" onclick="change()">change</button>

  <script>
    function change() {
      document.getElementById("demo").innerHTML = "change";
    }
  </script>
</body>
</html>
```

## ■ 자바스크립트 적용 방법

### ● 외부 파일

```
<!DOCTYPE html>
<html>
<head></head>
<body>
  <p id="demo">text</p>
  <button type="button" onclick="change()">change</button>
  <script src="script.js"></script>
</body>
</html>
```

```
function change() {
  document.getElementById("demo").innerHTML = "change";
}
```

## ■ 자바스크립트 적용 위치

### ● head

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
  <script src='script.js'></script>
</head>
<body></body>
</html>
```

Parsing HTML

Block

Parsing HTML

JS가져오기

JS실행

### ● body

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
</head>
<body>
  <!-- HTML -->
  <script src='script.js'></script>
</body>
</html>
```

Parsing HTML

Block

JS가져오기

JS실행



## ■ 자바스크립트 적용 위치

### ● head + async

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
  <script async src='script.js'></script>
</head>
<body></body>
</html>
```



### ● head + defer → 가장 효율적인 방법

```
<!DOCTYPE html>
<html>
<head>
  <title>Document</title>
  <script defer src='script.js'></script>
</head>
<body></body>
</html>
```



## ■ Strict Mode

- 순수 자바스크립트(Vanilla Script)에 적용
- 자유도가 높은 자바스크립트에 엄격한 문법을 적용
  - 선언하지 않은 변수 사용 불가 등
- **자바스크립트 엔진이 더 효율적이고 빠르게 동작 (성능 개선)**

```
'use strict';
```

## ■ 출력

- `innerHTML` : HTML 요소에 출력
- `document.write()` : body에 출력
- `window.alert()` : 경고창에 출력
- `console.log()` : 브라우저 콘솔(개발자 도구)에 쓰기

### My First Web Page

My first paragraph.

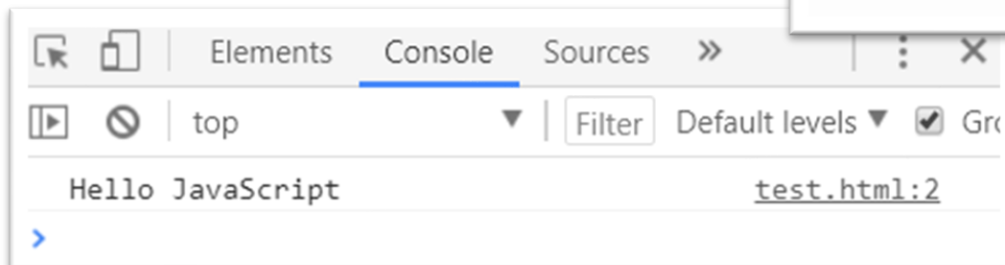
Never call `document.write` after the document has finished loading.  
It will overwrite the whole document.

11

이 페이지 내용:

11

확인



## ■ 출력

### ● innerHTML

- `document.getElementById(id)` 등을 사용하여 요소 선택 후  
`innerHTML` 을 사용하여 문자열 입력

```
<!DOCTYPE html>
<html>
<body>

  <h1>My First Web Page</h1>
  <p>My First Paragraph</p>

  <p id="demo"></p>

  <script>
    document.getElementById("demo").innerHTML = 5 + 6;
  </script>

</body>
</html>
```

**My First Web Page**

My First Paragraph.

11

## ■ 출력

### ● document.write()

- 테스트 목적으로 사용
- Popup 창에 내용을 동적으로 보여주기 위해 사용
- 기존 내용이 모두 삭제되므로 주의해서 사용

```
<h2>My First Web Page</h2>  
<p>My first paragraph.</p>  
  
<button type="button" onclick="document.write(5 + 6)">  
  Try it  
</button>
```

**My First Web Page**

My first paragraph.

Try it

11

## ■ 출력

### ● window.alert()

- 사용자에게 간단한 알림메시지 출력을 위해 경고 상자 사용
- **window** 키워드는 생략 가능

```
<h1>My First Web Page</h1>
```

```
<p>My first paragraph.</p>
```

```
<script>
```

```
  alert(5 + 6);
```

```
</script>
```

127.0.0.1:5500 내용:

11

확인

## ■ 출력

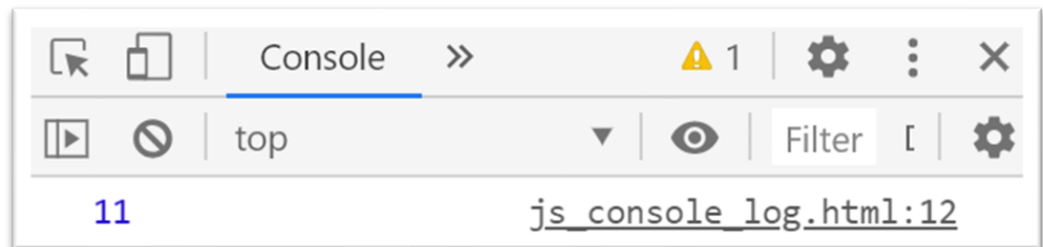
### ● console.log()

- 디버깅을 위해서 사용
- 개발자가 연산 결과를 확인하기 위해서 사용

```
<!DOCTYPE html>
<html>
<body>
  <h1>My First Web Page</h1>

  <script>
    a = 5;
    b = 6;
    c = a + b;
    console.log(c);
  </script>

</body>
</html>
```



## ■ 출력

### ● window.print()

- 현재 창의 내용을 프린터를 통해 출력

```
<!DOCTYPE html>
<html>
<body>
  <button onclick="window.print()">Print this page</button>
</body>
</html>
```

