


# Javascript 공부

## 자바스크립트의 역사

1993- Mosaic Web Browser (Marc Andreessen)

### HTML 시작하기 - Web 개발 학습하기 | MDN

이 문서는 HTML의 기본적인 내용에 대한 글입니다. 이 글에서는 HTML에 관련된 용어들(Element, Attribute ..)의 정의에 대해 설명할 것입니다. 또한 HTML이 무엇으로 이루어져 있는지(구성요소), 어떻게

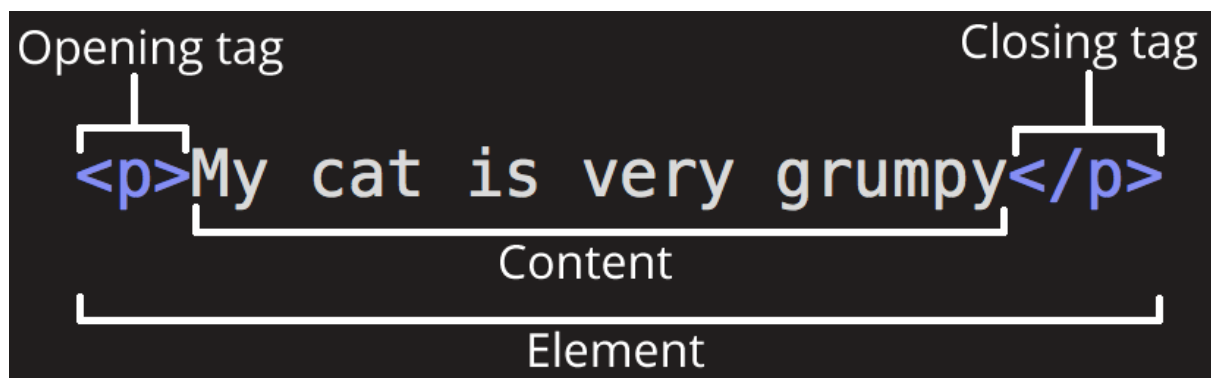
 [https://developer.mozilla.org/ko/docs/Learn/HTML/Introduction\\_to\\_HTML/Getting\\_started](https://developer.mozilla.org/ko/docs/Learn/HTML/Introduction_to_HTML/Getting_started)



MDN Web Docs  
moz://a

HTML (Hypertext Markup Language) 프로그래밍 언어는 아니고, 우리가 보는 웹페이지가 어떻게 구조화되어 있는지 브라우저로 하여금 알 수 있도록 하는 마크업 언어이다.

-HTML은 elements로 구성되어 있으며, 이들은 적절한 방법으로 나타내고 실행하기 위해 각 콘텐츠의 여러 부분들을 감싸고 마크업 합니다.



[Figure 01. tag, content, element]

\*마크업 언어 (markup language)란?

- 태그 등을 이용하여 문서나 데이터의 구조를 명기하는 언어의 한 가지이다. 위의 그림과 같이

(역사적으로 태그는 원래 텍스트와는 별도로 원고의 교정부호와 주석을 표현하기 위한 것이었으나 용도가 점차 확장되어 문서의 구조를 표현하는 역할을 하게 됨)

1994 - Netscape Navigator는 점유율 80%를 자랑하며 고공행진을 하게 된다.

동시에 Marc Andreessen은 어떻게 하면 동적인 웹사이트를 만들 수 있을까? 라는 고민을 하게 된다. 결론은 Scripting 언어를 추가하자! 라고 했으며 (왜 Scripting언어를 추가하면 동적이게 되는건데?)

동적언어를 뭘로 만들지?

처음으로 고려되었던 것은 Java언어였는데 Website한테 좀 무거워서

두 번째는 Brendan Eich를 스카웃해 와서 기존에 있었던 Scheme Script언어를 조금 변형하는 것이었다. Netscape가 새로운 언어를 만들자고 했다. 기존의 Scheme Script언어의 틀을 유지하면서 문법은 좀 Java스럽게 해보자구. (시간이 없어 10일 안에 만들어보자)

1994, 9 - 내부적으로는 Mocha라는 새로운 언어가 탄생하게 됨 → LiveScript로 이름이 바뀜 (Interpreter)가 포함되어 브라우저가 출시

(당시 Java언어의 인기가 치솟고 있었는데 어떻게 하면 그 인기에 살짝 얹혀갈 수 있을까 고민하다 LiveScript를 JavaScript로 바꾸자고 함)

1995년 - 공식적으로 Netscape Navigator 브라우저 위에 JavaScript와 그 언어를 이해할 수 있는 엔진(Interpreter)가 포함되어 드디어 출시가 된다.

이때 Microsoft가 호시탐탐 노리고 있었다.

MS는 Netscape Navigator에서 출시한 브라우저를 바로 Reverse Engineering하게 됨 (Binary Code를 분석해서 소스코드를 복원)

조금만 변경해서 베껴와서 JScript로 이름을 변경해서 시장에 내놓게 됨.

이 시점이 바로 개발자들이 모두 고통을 받기 시작한 시발점이 되었다. 브라우저마다 웹사이트의 최적화가 다르게 돌아가서 각 브라우저를 신경써야 했기때문에.

참지 못한 Netscape는 (1996,11)에 ECMA International에 찾아가서 우리가 Javascript를 만들었는데 이걸로 표준화를 만들어보자고 함

1997.7 - ECMAScript1 language specification이 드디어 시장에 나오게 되었다. (ECMA Script는 브라우저에서 동작하는 언어를 만들때 그 언어를 이해할수 있는 엔진을 이해하기 위해서는 변수, 함수 등의 문법들을 잘 정리한 문서이다)

ECMA Script 1 in 1997,

ECMA Script 2 in 1998,

ECMA Script 3 in 1999, - (error handling)

ECMA Script 4 in 2000 - (class도 넣어보자 등)

ECMA Script는 점차적으로 다양한 아이디어와 함께 논의 되고 있었다.

2000년에는 MS의 Internet Explorer로 95%시장 점유율을 확보하게 됨.

(점점 건방져짐 ECMA Script한테 아니 우리는 개발자들 쉽게 할려고 Script언어를 만들었지 class등을 포함시키면 더 이상 Script언어가 아니지 않느냐! 그래 맘대로해 어차피 사람들이 우리 언어를 많이 사용하고 있기때문에 우리가 표준이야! 이에 따라 MS는 더 이상 ECMA Script 표준 안에 참가하지 않게 됨) We don't care anymore

이런 이유로 ECMAScript 4부터 표준화 진행이 더디게 된 것이다.

4년뒤 2004년에서 MOZ://A사에서 Fireforx Browser가 출시됨

→ 다시 ECMAScript4에 찾아가서 우리가 ActionScript3라는 언어가 있고 타마린 (Tamarin)이라는 엔진이 있는데 이걸로 다시 표준화를 검토해보자.

이런 와중에 2004년에 AJAX (Aynchronous Javascript and XML) 비동기적으로 데이터를 서버에서 받아오고 처리할수 있도록 도와주는 AJAX가 들어옴

하지만 여전히 3사는 신경전을 벌이고 있음

동시에 개발자들은 커뮤니티가 형성이 되게 되는데 여기서 jQuery, dojo, mootools와 같은 많은 Library가 형성됨. 이게 생긴 목적은 개발자들이 브라우저에 따라 각자 개발해야 하는 수고로움을 덜어주기 위함이었다. 우리 APIs만 쓰면 다른 function만 쓰면 밑에 알아서 다른 브라우저에서 동작할수 있게 신경써줄게 (jQuery가 독보적이었다)

이 웹시장을 급격하게 변하는 정말 진취적인 사건이 하나가 있었다.

2008 - Chrome (google) 브라우저를 출시하게 됨. JIT (just-in-time comilation)이 포함된 엄청나게 강한 브라우저임 - javascript를 엄청나게 빠르게 읽어들이 수 있었다

따라서 2008년 7월에 4사 구글, 모질라, MS, Netscape가 모여서 생산적인 대화를 하기 시작함.

2009년 ECMAScript 5 → 2015 ECMAScript 6가 나옴 (class 등 여기서 생김) 통합됨

그리고 매년 새로운 버전이 나온다.

Javascript는 잘 정착된 언어가 되었다

이후에 모든 브라우저들이 표준사항을 잘 따라가고 있었기 때문에 더 이상 jQuery나 dojo 등의 라이브러리들의 도움 없이도 충분히 Javascript API등만으로도 모든 브라우저에서 돌릴 수 있었다.

각 브라우저마다 ECMAScript 표준을 따르는 엔진들이 존재한다

Chrome - V8, Firefox - SpiderMonkey, Safari-JSCore, MS Edge - Chakra, Opera-Carakan, Adobe Flash-Tamarin

V8 Js Engine은 node.js, Electron에도 사용된다.

SPA (Single Page Application) → 웹사이트를 만드는 것은 더 이상 충분하지 않다. 필요한 부분만 업데이트 하는 것이 유행한다 → SPA 구현을 더 쉽게 해주는 것이 React, Angular, Vue 같은 애들이 나오기 시작함

WA (Web assembly 잠재력이 굿)