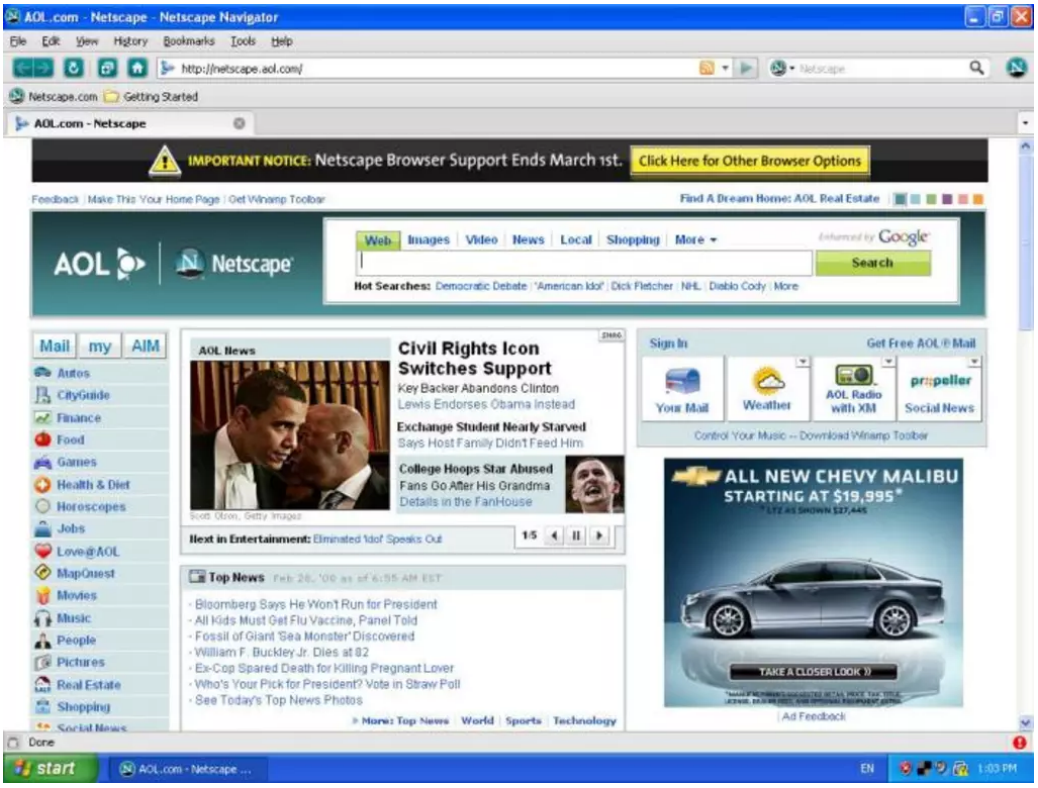
Chap2 자바스크립트란?

2-1. 자바스크립트의 탄생

- **1995년 넷스케이프 커뮤니케이션즈**는 웹페이지의 보조적인 기능을 수행하기 위해 브라우저에서 동작하는 경량 프로그래밍 언어, 현재의 **자바스크립트**를 개발했다.
- 처음 출시되었을 때 넷스케이프 내비게이터라는 웹 브라우저에 탑재되었다.
- 이름은 1996년 3월 **'모카'**라는 이름으로 출시되었고 이후 9월에 **'라이브스크립트'**로 바뀌고 최종적으로 12월에 **'자바스크립트'**라는 이름으로 명명되었다.



| 넷스케이프 내비게이터

2-2. 자바스크립트의 표준화

- 1996년 8월 마이크로소프트는 자바스크립트의 파생 버전인 JScript를 인터넷 익스플로러3에 탑재했다.
- 이로인해 브라우저에 따라 **크로스 브라우징** 이슈가 발생하기 시작
- 해결방안

넷스케이프 커뮤니케이션즈는 컴퓨터 시스템의 표준을 관리하는 비영리 표준화 기구인 **ECMA 인터내셔널**에 자바스크립트이 표준화를 요청.

1997년 7월 표준화 된 자바스크립트 초판 완성되었고 상표권 문제로 **ECAMScript**로 명명되었다.

> **크로스 브라우징** : 웹페이지가 웹 브라우저의 종류에 구애 받지 않고 제작자의 의도에 맞게 보여지거나 동작할 수 있게 하는 작업. (즉 크로스 브라우징 이슈라 함은 브라우저마다 화면이 다르게 보이는 현상이 발생함)

2-3. 자바스크립트의 성장의 역사

▼ Ajax

1999년 등장한 자바스크립트를 이용해 서버와 브라우저가 비동기 방식으로 데이터를 교환할 수 있는 통신 기능이다.

AJAX는 전체 페이지가 다시 로드되지 않고 HTML 페이지 일부 DOM만 업데이트하는 좀 더 복잡한 웹페이지를 만들 수 있게 해준다. 면 웹페이지 일부가 리로드 되는 동안에도 코드가 계속 실행되어, 비동기식으로 작업할 수 있다.

▼ JQuery

JavaScript 라이브러리로 DOM 제어를 더 쉽고 편리하게 해준다.

▼ 개인적인 궁금점

- Ajax와 JQuery 차이

ajax는 서버와의 비동기 통신을 다루는 기술을 의미하고 jquery는 ajax뿐만 아니라 여러 자바스크립트 관련 작업을 쉽게 할 수 있도록 도와주는 자바스크립트 라이브러리이다.

그리고 JQuery 같은 경우에는 현재 fetch API, Axios, react, vue가 나와서 선호도가 떨어진다고 한다.

▼ V8 자바스크립트 엔진

더욱 빠르게 동작하는 자바스크립트 엔진의 필요성이 대두되면서 2008년 등장하였다.

V8자바스크립트 엔진으로 촉발된 자바스크립트의 발전으로 과거 웹 서버에서 수행되던 로직들이 대거 클라이언트(브라우저)로 이동했고 이는 웹 애플리케이션 개발에서 프론트엔드 영역이 주목 받는 계기로 작용했다.

▼ Node.js

- V8 자바스크립트 엔진으로 빌드된 자바스크립트 런타임 환경이다.
- 브라우저의 자바스립트 엔진에서만 동작하던 자바스크립트를 브라우저 이외의 환경에서도 동작할 수 있도록 자바스크립트 엔진을 독립시킨 자바스크립트 실행 환

경. → 프론트엔드와 백엔드 영역에서 자바스크립트를 사용할 수 있어졌다.

- 비동기 I/O를 지원하며 단일 스레드 이벤트 루프 기반으로 동작하여 SPA에 적합하다. 단 CPU 사용률이 높은 애플리케이션에는 권장하지 않는다.

▼ SPA 프레임워크

개발 규모가 크고 복잡해지면서 패턴과 라이브러리가 출현했다. 이 덕분에 많은 도움 이 되었지만 변경에 유연하면서 확장하기 쉬운 애플리케이 아키텍처 구출을 어렵게했고 필연적으로 프레임워크가 등장하게 되었다.

요즘의 프레임워크 : Angular, React, Vue.js, Svelte

▼ 개인적으로 궁금

Svelte ?

스벨트(Svelte)는 2016년 출시한 오프노스 프론트엔드 프레임워크이다.

특징

- Write less code(적은코드)
- No Virtual DOM(가상 돔 없음)
- Trult reacive(진정한 반응성)

```
<script>
  let count = 0
  const increment = () => {
    count += 1
  }
</script>

<button on:click={increment}>
  Clicks: {count}
</button>
```

```
import React,{useState} from 'react;

export default()=>{
const [a,setA]=useState(1);
```

```
const [b,setB]=useState(2);

function handleChangeA(event){
  setA(+event.target.value);
}
function handleChangeB(event){
  setB(+event.target.value);
}
return(
<div>
  <input type="number" vlaue={a} onChange{handleChangeA
  <input type="number" vlaue={b} onChange{handleChangeB

  <p>{a}+{b}={a+b}</p>
<div>
);
};
```

스벨트는 별도의 클래스나 객체를 따로 정의하지 않고도 상태(state)를 추적할 수 있도록 해준다.

2-4. 자바스크립트와 ECMAScript

ECMAScript

- 자바스크립트 표준 사양인 ECMA-262를 뜻한다.
- 값. 타입, 객체와 프로퍼티, 함수, 표준 빌트인 객체 등 핵심 문법 규정

자바스크립트

- ECMAScript + 브라우저 별도 지원 요소(클라이언트 사이드 Web API, 즉 DOM,BOM, Canvas, XMLHttpRequest, fetch, requestAnimationFrame, SVG, Web Storage, Web Component, Web Worker) 합친 개념

2-5 자바스크립트 특징

- **인터프리터 언어**이다. (그러나 대부분의 모던 자바스크립트 엔진은 컴파일러와 인터프리터 장점 결합해 사용한다.)

자바스크립트는 일반적으로 인터프리터 언어로 구분한다. 전통적인 컴파일러 언어와 인터프리터 언어를 비교하면 다음과 같다.

컴파일러 언어	인터프리터 언어
코드가 실행되기 전 단계인 컴파일 타임에 소스코드 전체를 한번에 머신 코드[26]로 변환한 후 실행한다.	코드가 실행되는 단계인 런타임에 문 단위로 한 줄씩 중간 코드intermediate code인 바이트코드[27]로 변환한 후 실행한다.
실행 파일을 생성한다.	실행 파일을 생성하지 않는다.
컴파일 단계와 실행 단계가 분리되어 있다. 명시적인 컴파일 단계를 거치고, 명시적으로 실행 파일을 실행한다.	인터프리트 단계와 실행 단계가 분리되어 있지 않다. 인터프리터는 한 줄씩 바이트코드로 변환하고 즉시 실행한다.
실행에 앞서 컴파일은 단 한번 수행된다.	코드가 실행될 때마다 인터프리트 과정이 반복 수행된다.
컴파일과 실행 단계가 분리되어 있으므로 코드 실행 속도가 빠르다.	인터프리트 단계와 실행 단계가 분리되어 있지 않고 반복 수행되므로 코드 실행 속도가 비교적 느리다.

- 명령형, 함수형, 프로토타입 기반 객체지향 프로그래밍을 지원하는 **멀티 패러다임 프로그래밍 언어**이다.