

웹 프로그래밍 4주차

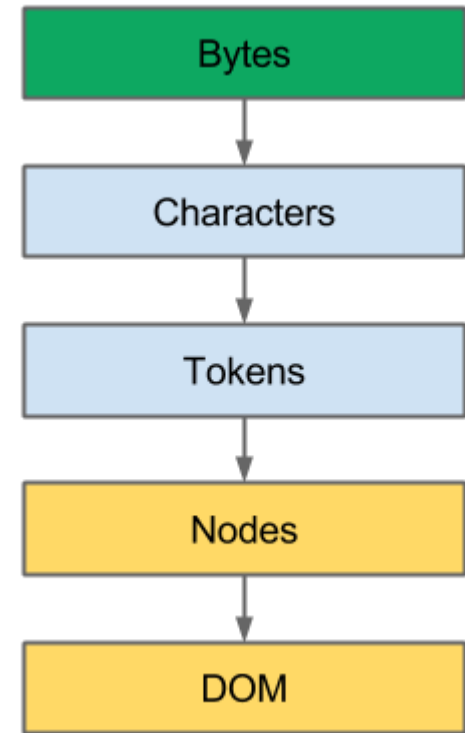
브라우저 렌더링 내부 훑아보기

1. 브라우저 렌더링

- 브라우저 렌더링 – Critical Rendering Path
 - 웹 브라우저가 HTML, CSS 파일을 로딩하여 브라우저 화면에 그래픽요소로 표현하는 과정
 - 실제로 각 픽셀 자리에 어떤 내용 (색상) 정보가 들어가야 하는지 계산하는 과정
- Parsing -> Style -> Layout -> Paint -> Composite 5단계로 구성

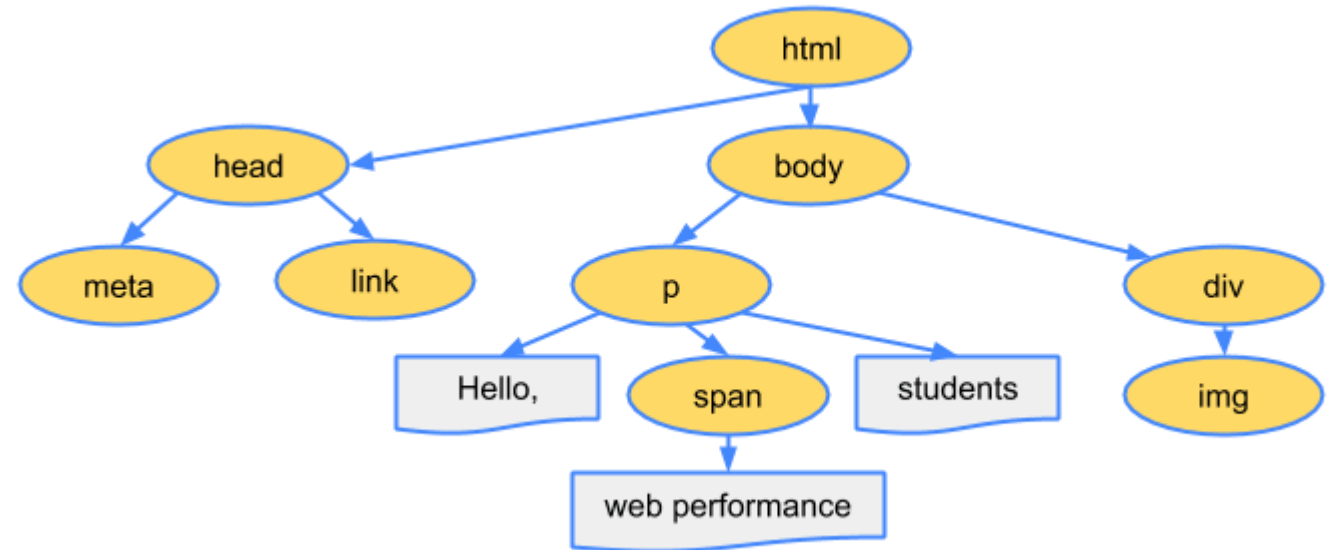
1. 브라우저 렌더링

- Parsing: Object Model 구축
 - HTML 문서를 바탕으로 Document Object Model(DOM)으로, CSS 내용을 가지고 CSS Object Model (CSSOM) 구축하는 과정
 - Tokenizing: HTML/CSS 문서를 가지고 태그, 스타일의 시작, 끝 단위, 내부 텍스트 등으로 나눠줌
 - Lexing: 나눠진 토큰들을 노드 단위로 만들어줌
 - DOM 생성: 각 노드 간의 관계를 트리 모양으로 구축함



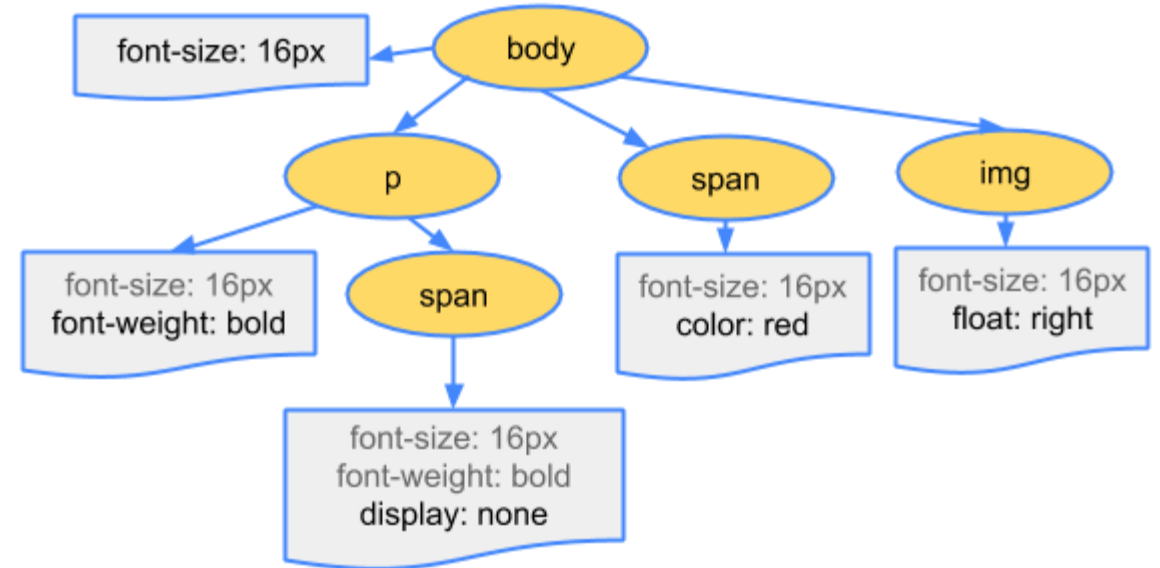
1. 브라우저 렌더링

- 구축된 DOM 트리
 - 계층 구조로 이루어진 마크업 문서를 트리 형식으로 변환하고, 각 노드 내용을 부여함
 - 각 노드에는 어떤 태그인지, 각 태그에 부여된 어트리뷰트, 하위 노드에 대한 정보를 가짐



1. 브라우저 렌더링

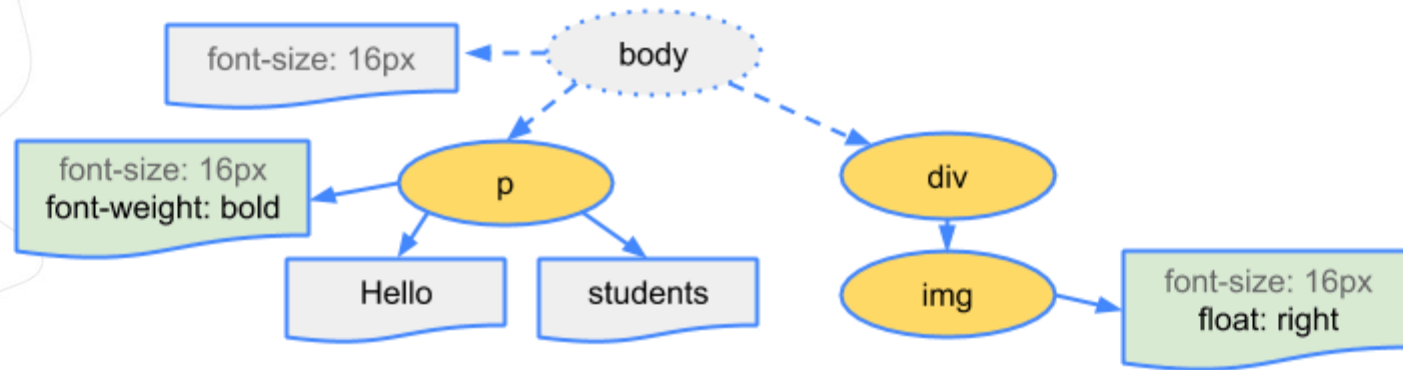
- 구축된 CSSOM 트리
 - DOM 트리 안에는 각 노드에 포함된 내용을 가지고 있고, CSSOM 에는 각 노드에 적용될 스타일 정보를 포함함.



1. 브라우저 렌더링

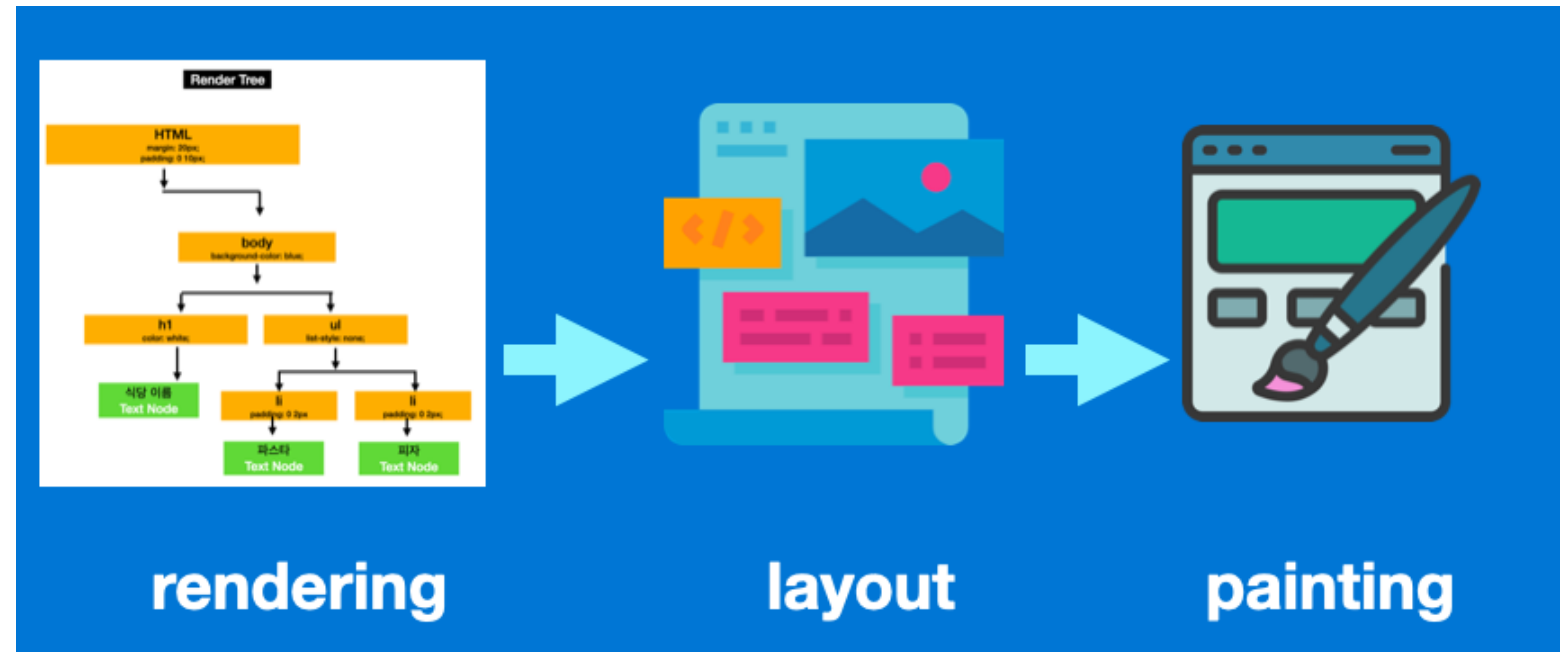
- Style: Rendering Tree 구축
 - DOM Tree + CSSOM Tree = Rendering Tree
 - 실제 브라우저 화면에 보여지는 노드들만 포함시킴
 - 화면에 보이지 않는 요소는 렌더링 성능에 영향을 주지 않음

Render Tree



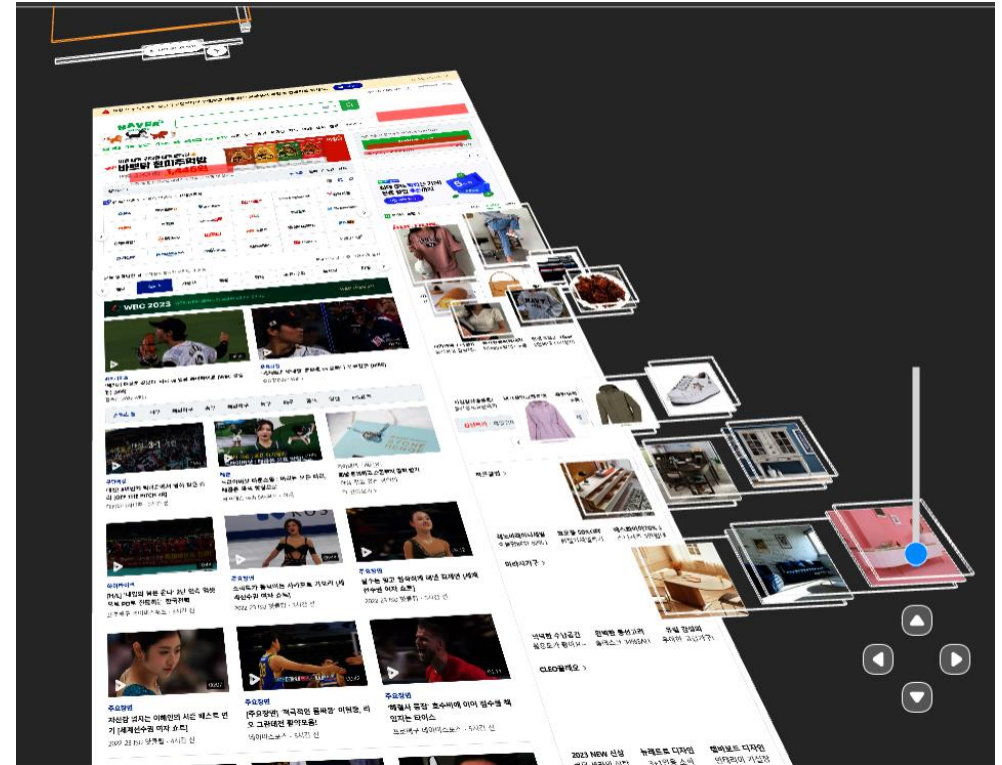
1. 브라우저 렌더링

- Layout: 각 노드들의 화면에 배치될 위치를 **CPU**를 이용해 계산
 - 브라우저 화면 크기 (viewport), 폰트 크기에 따른 크기 요소들까지 전부 계산
 - Layout Tree 구축



1. 브라우저 렌더링

- 각 Layer 단위로 나누어서 최적화 여부를 결정할 수 있음
 - 화면은 브라우저 3D View

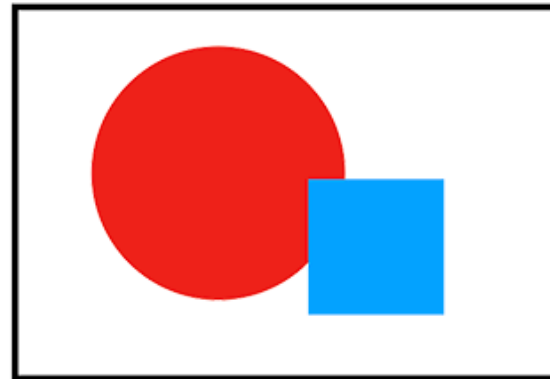


1.브라우저 렌더링

- Reflow: 브라우저의 Layout 정보가 변경되어 재계산하는 동작
 - viewport 변경 (브라우저 크기 변경)
 - 폰트 크기 변경
 - 스크롤 (일부분)
 - 화면 확대/축소
 - 스크립트에 의한 크기 정보 변경
 - 새로운 DOM 노드 추가

1. 브라우저 렌더링

- Paint: Layout 에서 계산해놓은 각 위치를 가지고, 실제로 그려야 하는 Layer 를 쌓아서 그리는 정보(Paint Record)를 구축한다

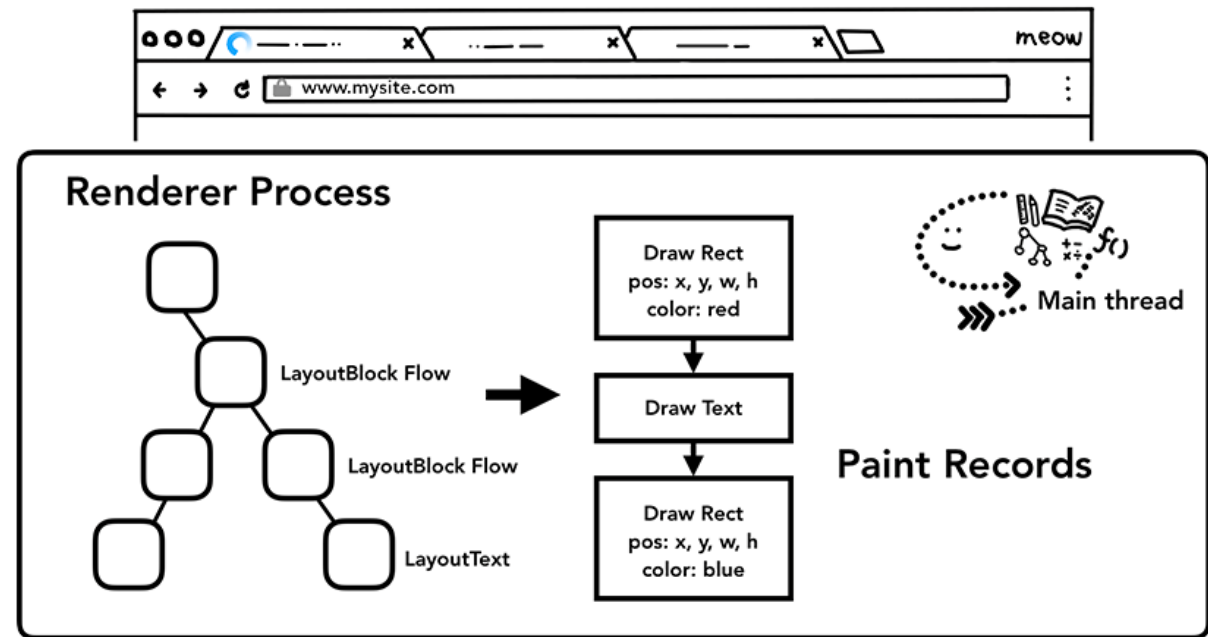


Draw Circle first
or
Square first?



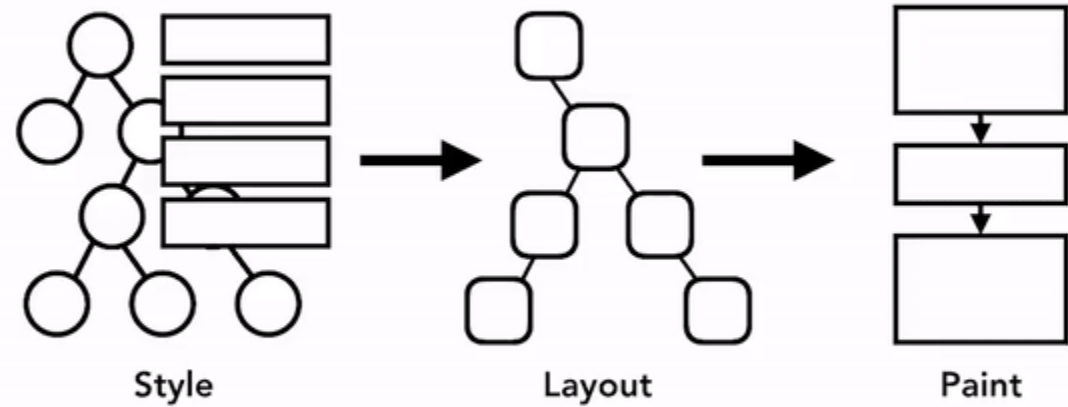
1. 브라우저 렌더링

- Paint Record: 레이아웃 트리를 가지고 그림을 그려야 하는 순서와 해당 순서에서 그릴 내용에 대해 순서대로 가지고 있는 레코드



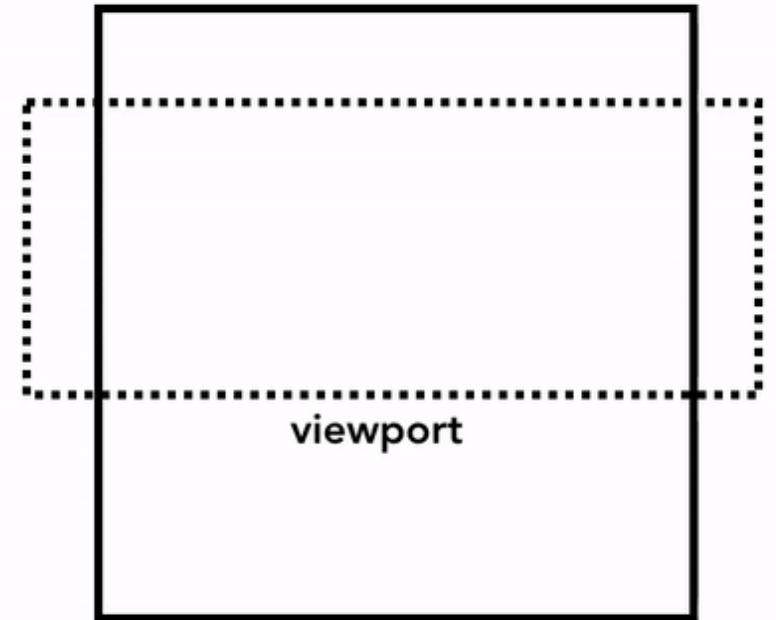
1. 브라우저 렌더링

- 브라우저 렌더링 과정은 모두 초당 60프레임이 매번 일어나야 함.
 - 1프레임 당 약 14-16ms 안에 모든 내용들이 처리되어야 함
 - 만약 16ms 초과해서 렌더링 되면 사람이 체감하는 버벅임이 발생함



1. 브라우저 렌더링

- Composite: 브라우저가 알고있는 모든 layout, paint 정보를 활용해서 모니터 화면에 픽셀단위에 표현하는 작업. rasterizing 이라고도 함.
- 모니터 화면이 옮겨지면, 경우에 따라 추가로 그림
- 경우에 따라서 **GPU**를 활용하여 최적화 함
 - 가장 짧게 걸리는 작업 중 하나

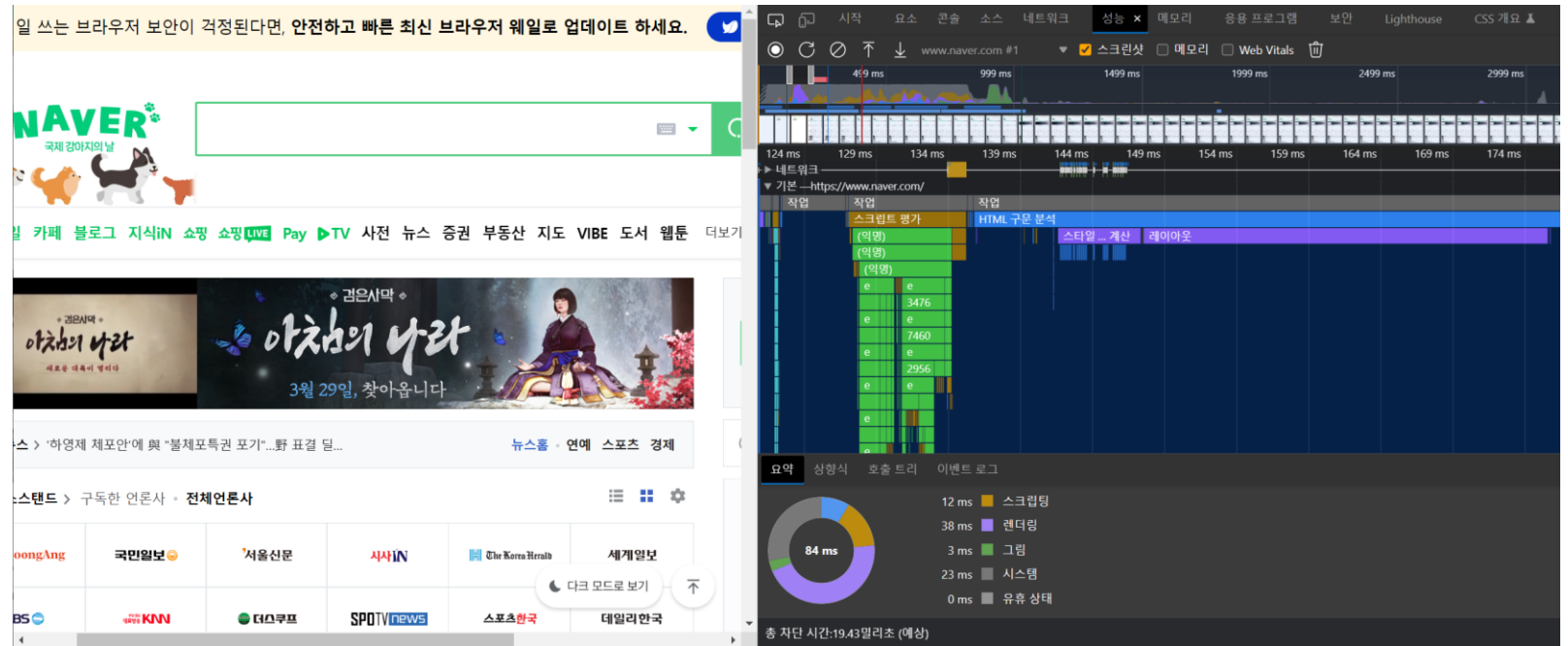


1. 브라우저 렌더링

- reflow vs repaint
- reflow: Render Tree나 Layout Tree 정보가 변경되어 화면 전체의 레이아웃 구성요소를 다시 계산하는 과정
- repaint: 이미 계산한 layout 영역 과 Paint Record 안에서 화면에 그릴 요소들을 일부분 수정하는 과정
 - transform
 - filter
 - 등의 요소를 변경/추가할 때 repaint 가 일어남

1. 브라우저 렌더링

- 브라우저 렌더링에 소요된 시간 측정하기
 - 브라우저 F12 -> 성능(Performance)탭 -> 새로고침



2. Javascript 소개

- 웹 페이지에서 복잡한 기능이나 동적인 기능 구성을 처리하기 위해 만들어진 프로그래밍 언어.
 - 최근에는 node / deno 같이 웹 브라우저 없이도 자바스크립트를 실행할 수 있음
- 1995년에 Brendan Eich에 의해 만들어짐
 - 초기 Netscape, Internet Explorer 에 이식됨

2. Javascript 소개

- 2006년 jQuery가 출시됨. 대부분의 웹 사이트에서 jQuery 채택.
 - Netscape, Internet Explorer 마다 자바스크립트의 동작이 달라 브라우저를 고려하지 않고 자바스크립트를 작성할 수 있게 도와주는 라이브러리
 - 15년간 주류 웹 사이트 개발에 채택되어 사용됨
 - 아직도 많은 사이트들은 jQuery기반으로 동작

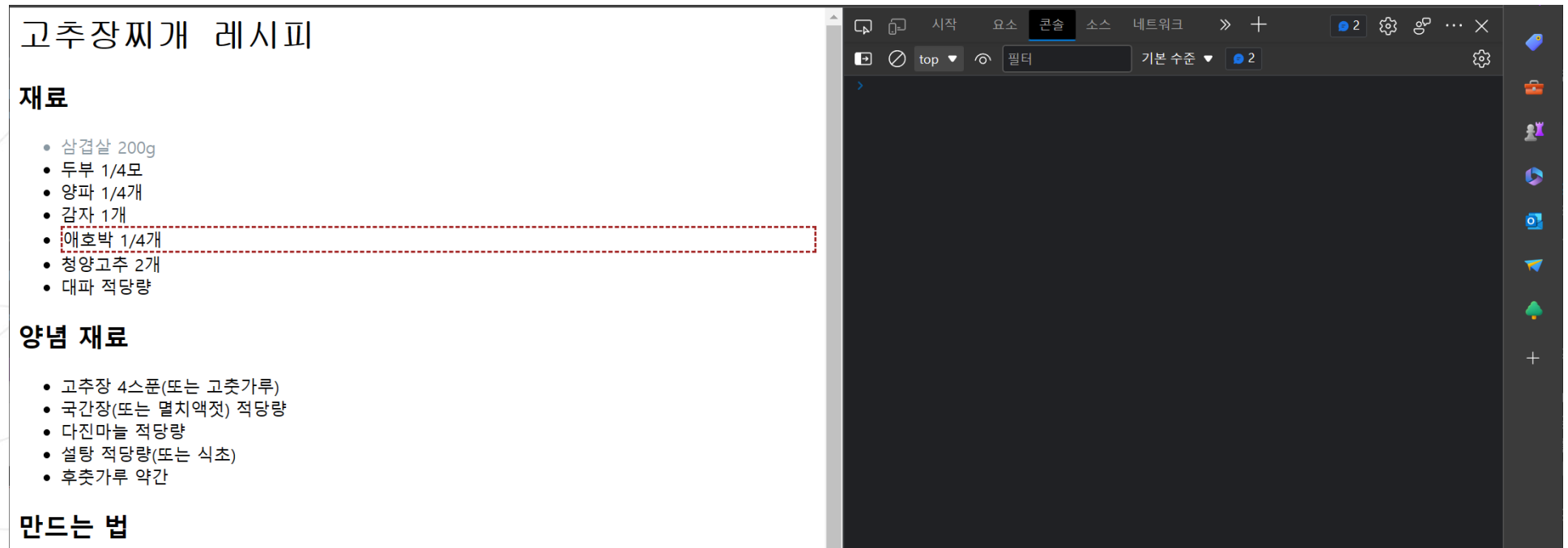


2. Javascript 소개

- jQuery를 활용한 웹 사이트 개발의 어려움을 느낀 Facebook(현 Meta)에서 브라우저에 표현할 정보를 저장하면 화면에 최적화하여 화면에 표시할 방법을 구현하기 위해 React를 개발, 오픈소스로 발표.
 - 이외 다양한 컨셉의 자바스크립트 프레임워크/라이브러리가 출시됨
 - backbone, knockout, prototype
 - Angular, vue, Preact, svelte 등

3. Javascript 실습

- 브라우저 콘솔을 활용한 실습
 - 웹 사이트가 열려있는 상태에서 F12 -> 콘솔(Console) 메뉴



3. Javascript 실습

- 브라우저 콘솔창은 자바스크립트 REPL (Read-evaluate-print loop)
 - 타이핑 하는대로 자바스크립트 코드를 한 줄 한 줄 실행해줌
 - 자바스크립트 기능이 헛갈리면 바로 열어서 코드를 실행해서 결과 확인 가능



3. Javascript 실습

- \$0 : 브라우저 Inspector 로 선택한 태그
- \$0.style : 선택한 태그의 스타일을 직접 지정
- **중요** : 자바스크립트를 활용해 태그를 변경하는 것은 영구적인 변경 X

3. Javascript 실습

- window 하위에서 기본으로 지원하는 함수
 - alert: 알림 메시지
 - confirm: 확인/취소 함수
- document 에서 기본으로 지원하는 기능
 - append: 태그를 추가함
 - removeChild: 태그 아래에 노드를 삭제함

3. Javascript 실습

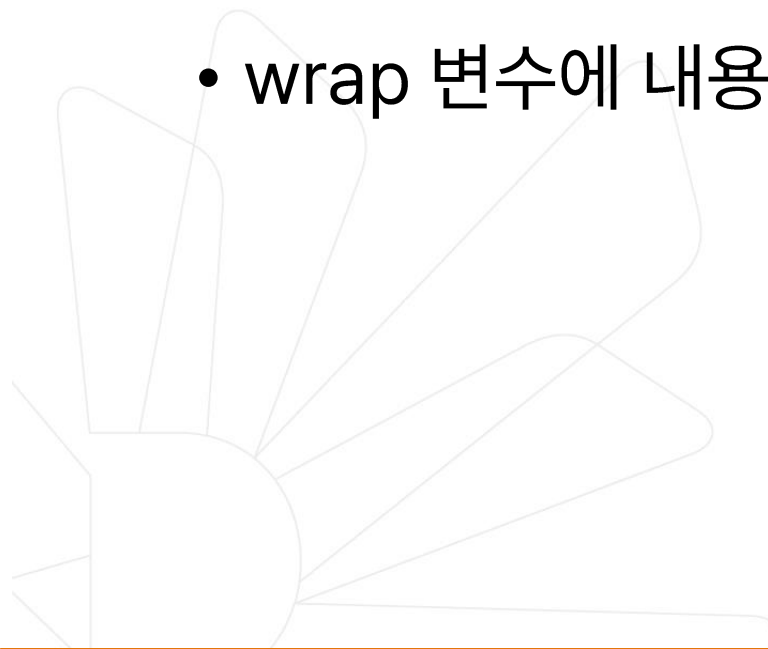
- document.getElement^sByTagName('태그이름')
 - 태그명을 기준으로 모든 태그를 가져옴
- document.getElementById
 - 태그 id 를 기준으로 하나의 태그를 가져옴
- document.getElement^sByClassName
 - 태그에 부여된 class 를 기준으로 모든 태그를 가져옴

3. Javascript 실습

- document.querySelector
 - CSS 선택자를 활용하여 하나의 태그를 선택
- document.querySelectorAll
 - CSS 선택자를 활용하여 모든 태그를 선택

3. Javascript 실습

- 선택한 태그를 wrap 이라는 변수에 저장
- wrap 변수에 내용 수정 가능



```
> document.querySelector('div#wrap')
< <div id="wrap" style="display: block;" class>...
  </div>

> const wrap = document.querySelector('div#wrap')
< undefined

> wrap.style
< CSSStyleDeclaration {0: 'display', accentColor: '',
  ▼additiveSymbols: '', alignContent: '', alignItems:
    '', alignSelf: '', ...} i
  0: "display"
  accentColor: ""
```

3. Javascript 실습

- 버튼을 클릭했을 때 javascript 코드를 실행하기
 - button onclick 어트리뷰트 안에 있는 자바스크립트 코드를 실행함

```
<button onclick="alert('hi')">  
|   버튼입니다  
</button>  
.
```

4. 실습 과제

- 레시피.html 에 다음 기능을 하는 버튼을 각각 추가한다
 - 클릭 할 때마다 문서의 너비가 50px 씩 늘어나는 버튼
 - 클릭 할 때마다 문서의 배경색이 흰색 -> 검은색 -> 흰색 -> 검은색... 토글
 - 클릭 할 때마다 “헤더를 키우시겠습니까?” 확인 / 취소를 물어보고 확인을 누르면 헤더의 높이를 10px 씩 늘리는 코드.
 - 헤더가 없다면 header 영역도 새로 만들 것