

# 웹의 개요

# 목차

- 웹의 역사
- 웹 프로토콜의 구조
- 브라우저의 동작 구조
- 프론트엔드 프레임워크의 비교

# 마크업의 개념

- 웹의 시대
- 국가간 전자문서교환
- 포맷이 맞지않음?

# 마크업(Markup) 이란 ?

- 전통적인 마크업
  - 전통적인 출판 단계
    - 원고 작성자
      - 원고 작성
    - 서식 설계사
      - 원고의 출판 형태를 정의(서체 종류 및 크기 등)  
이러한 수작업을 마크업이라 한다
    - 출판 전문가
      - 마크업된 지시대로 원고 데이터를 표현



# 마크업(Markup) 이란 ?

- 절차적(Procedural) 마크업
  - (예) troff 절차적 마크업

```
.bp.  
.ps 20.  
.ft I  
.ce.  
문서 기술 언어 SGML.  
.sp 2.  
.ps 10.  
.ft B  
SGML 은 "Standard generalized Markup Language"의 약자로, "문서 기  
술 언어"로 번역할 수 있다.  
.sp 2.
```

## 문서 기술 언어 SGML

SGML 은 "Standard generalized Markup Language" 의 약자로, "문서 기술 언어"로 번역할 수 있을 것이다.

- 인쇄 제어 명령 마크업들이 특정 분리된 라인에 위치
- 라인 맨 앞에 "."으로 시작하여 순수 텍스트와 구분
- .bp : 새로운 페이지로 개행
- .ps 20 :폰트 크기를 20
- .ft I : 글자를 이탤릭체
- .ce :텍스트를 중앙정렬
- .sp 2 : 두 행 띄우기

# 마크업(Markup) 이란 ?

HTML(HyperText Markup Language)

W3C의 명세

웹상에서 Hypertext 문서를 생성할 수 있는 간단한 마크업 언어

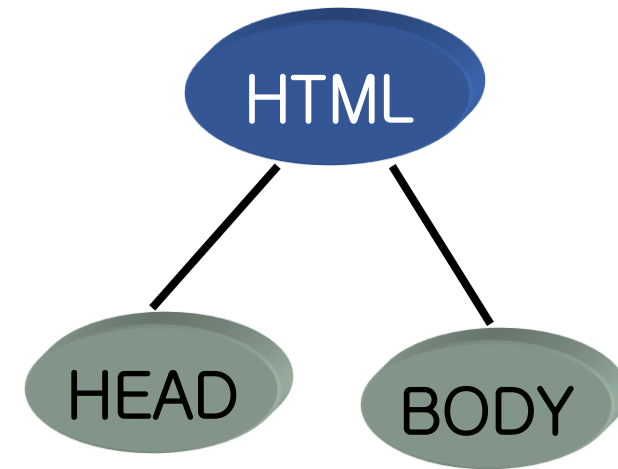
SGML의 subset

ASCII Text 양식의 문서

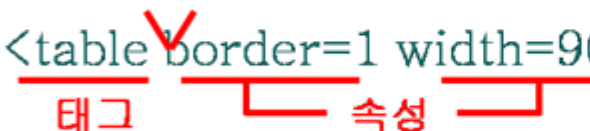
(일반 텍스트 편집기 작성 가능)

HTML = SGML 선언 + a DTD

HTML                  SGML



# HTML

- Hyper Text Markup Language
- 웹(www)문서를 작성하기 위해 사용되는 언어
- Http 프로토콜 사용
- 한 쌍의 태그(tag)명령으로 이루어짐 – 한쌍이 아닌경우도 있음
  - `<font>...</font>`, `<br>`
- 태그 – 속성(attribute)로 옵션을 줌
  - `<table border=1 width=900>.....</table>`  

- 대소문자 구별하지 않음
- 2 이상의 공백 혹은 enter 무시 – 별도의 특수문자 기호 필요
  - Enter-`<br>`, 공백-`&nbsp;`

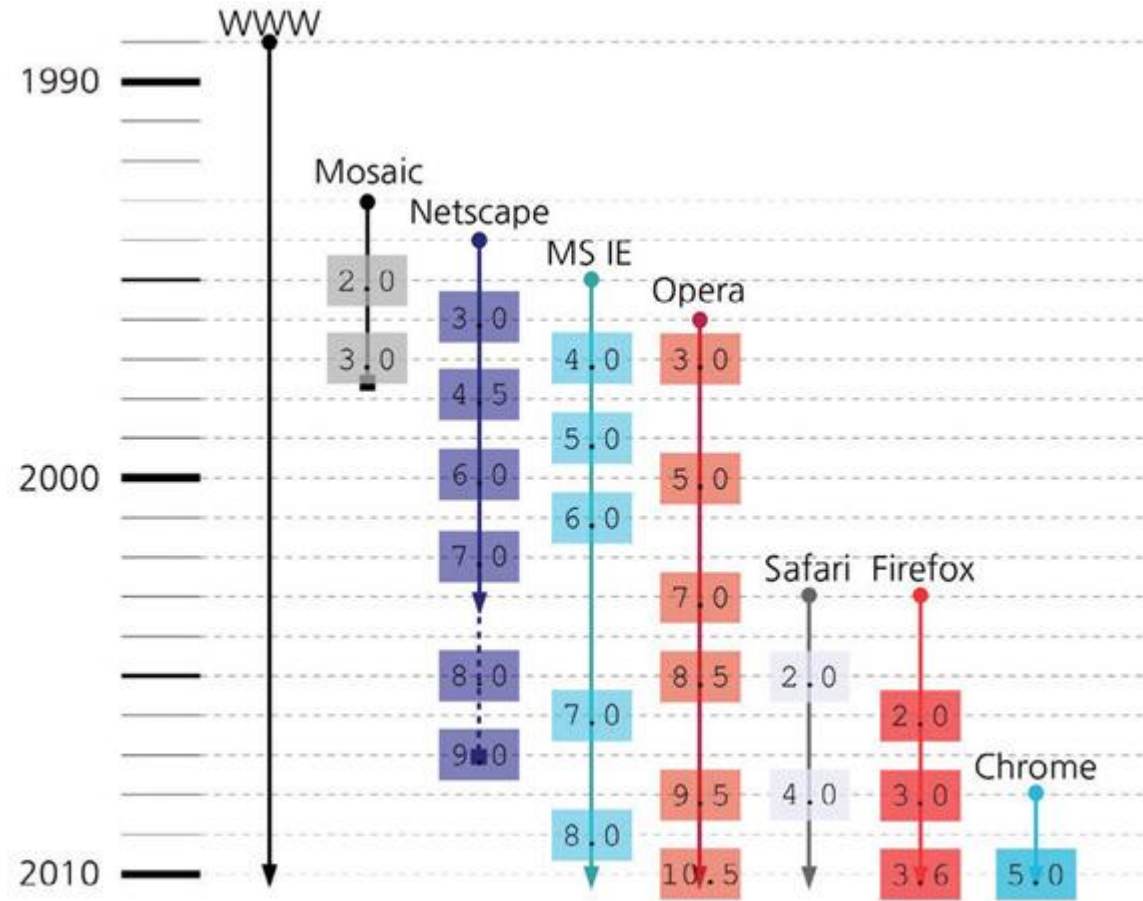
# HTML문서의 기본 구성

머릿글	<HTML>	:HTML문서의 시작
	<HEAD>	:문서의 헤더 시작
	<TITLE>	:문서의 제목 시작
	문서의 제목...	:문서의 제목 내용
	</TITLE>	:문서의 제목 끝
본문	</HEAD>	:문서의 헤더 끝
	<BODY>	:문서의 본문 시작
	본문 내용	:문서의 본문 내용
	</BODY>	:문서의 본문 끝
	</HTML>	:HTML문서의 끝



# 웹의 역사

# 웹브라우저 발달역사



출처: <http://evolutionofweb.appspot.com/>

# 웹브라우저 역사

- 최초의 웹브라우저는 1990년 12월 25일 영국의 소프트웨어 공학자인 팀 버너스리(Tim Berners-Lee)가 정보 공유 시스템용으로 개발한 월드와이드웹(WorldWideWeb)이었다. 처음에는 웹브라우저의 이름과 정보 공유 방법을 모두 월드와이드웹이라고 했으나, 후에 두 기능의 혼동을 방지하기 위하여 웹브라우저의 이름은 넥서스(Nexus)로, 정보 공유 방법은 월드 와이드 웹(World Wide Web) 또는 간단히 웹(web)으로 구분하였다.
- 넥서스는 단순한 웹브라우저로 그 기능이 텍스트를 처리하는 정도였다
- 모자이크(Mosaic)은 미국 일리노이 대학 슈퍼컴퓨터 응용 연구소(National Center for Supercomputing Applications: NCSC)의 마크 앤드리슨에 의하여 개발되었다.
- 윈도 환경에서 사용할 수 있도록 멀티미디어 그래픽 사용자 인터페이스를 채용하였다.
- 마크 앤드리슨이 회사를 설립하고 모자이크를 개량하여 네스케이프 내비게이터(Netscape Navigator)를 개발
- 1995년에 MS사는 스파이글래스 모자이크을 인터넷 익스플로러(Internet Explorer: IE)라는 이름으로 출시하였다. 이것이 바로 윈도95에 끼워 제공하던 IE이다.
- 윈도98과 함께 배포될 때에는 대폭적으로 성능이 개선되었다. 이 시점 이후로 IE는 네스케이프를 밀어내고 웹브라우저의 절대 강자로 군림하게 되었다.
- 1996년에 개발된 오페라(Opera)는 작고 빠른 웹브라우저라는 모토 하에 꾸준히 명맥을 이어왔다.
- 2003년 네스케이프의 몰락과 함께 모질라(Mozilla) 재단은 독자적으로 파이어폭스(FireFox)를 개발하여 지금에 이르고 있다.
- 2003년 애플사는 자체적으로 사파리(Safari)를 개발하여 매킨토시에서 아이폰에 이르는 모든 제품의 웹브라우저로 제공해오고 있다.
- 2008년에 들어와 구글 크롬(Chrome)이 출현하였다. 크롬의 가장 큰 특징은 페이지 탭별로 멀티태스킹이 가능하고 빠르다는 점이다. 이에 힘입어 크롬은 지금까지 꾸준히 점유율을 높여오고 있다

# html역사

연월	사양	설명
1993년 6월	HTML 1.0	IETF Internet Draft
1995년 11월	HTML 2.0	RFC 1866
1997년 1월	HTML 3.2	W3C 권고
1997년 12월	HTML 4.0	W3C 권고
1998년 2월	XML 1.0	W3C 권고
1998년 12월	HTML 4.01	W3C 권고
2000년 1월	XHTML 1.0	W3C 권고

# HTML 의 발달사

넷스케이프 → W3C → WHATWG →

W3C: 팀버너스 리가 1994년 설립한 비영리 단체로  
HTML4.01을 권고한 것을 1999년 일로 10년간 HTML 최  
신버전으로 있었음

WHATWG(2004년) :모질라 재단과 오페라 소프트웨어  
(모든 웹사이트에 플러그인(액티브엑스) 이 들어가면서 웹  
사이트가 점점 무거워짐  
익스플로러를 제외한 독자적으로 새로운 웹 표준기관을 설  
립함

2004년 6월 HTML5 표준을 제정하는 WHATWG가 설립됨



## HTML5 배경

- 웹이 정적 문서에서 동적 프로그램으로의 변화
- XHTML2.0의 기존 웹과의 비호환성과, 엄격한 XML 규칙의 적용으로 인한 제어의 어려움

### WHATWG (Web Hypertext Application Technology Working Group)

모질라재단, 애플, 오페라 소프트웨어의 세 회사가 모여 현재의 HTML4.01 기술과 호환되면서 웹의 기능과 표현 범위를 확장하고자 하는 기술 표준을 작성

# html5 의 역사



**W3C**

**VS**

**WHATWG**

**xhtml을 제안**

**html5을 제안**  
**apple, mozilla, opera**

# html5 의 역사



W3C



WHATWG

XHTML 사양 폐기  
HTML5를 표준으로 채택  
2009년 10월 WHATWG 모든 사양을 W3C 에 이관

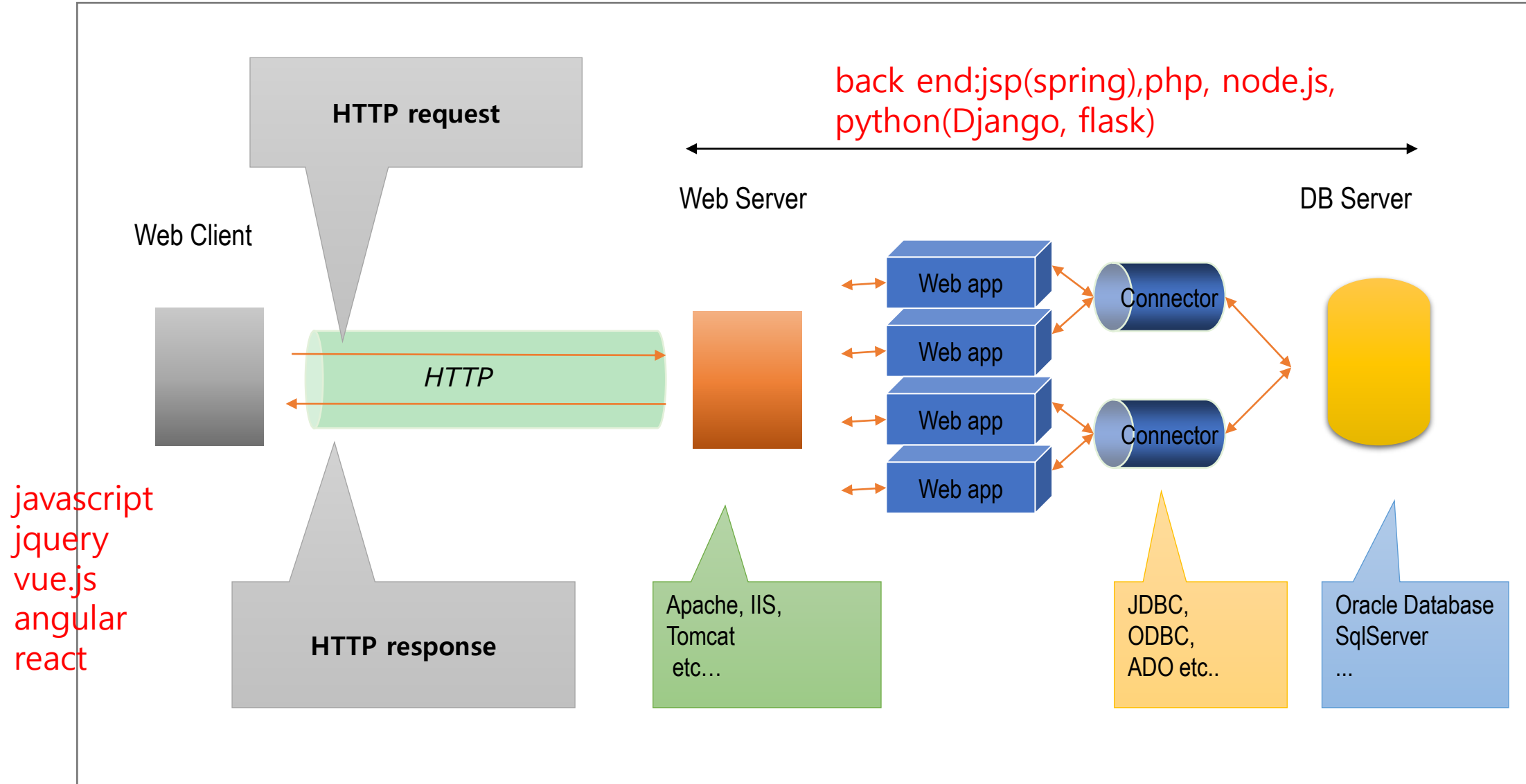


# html5 역사

연월	사양	설명
2011년 5월	HTML5 최종 초안	IETF Internet Draft
1995년 11월	HTML5 후보 표준안	RFC 1866
1997년 1월	HTML5 제안 표준안	W3C 권고
1997년 12월	HTML5 최종 표준안	W3C 권고

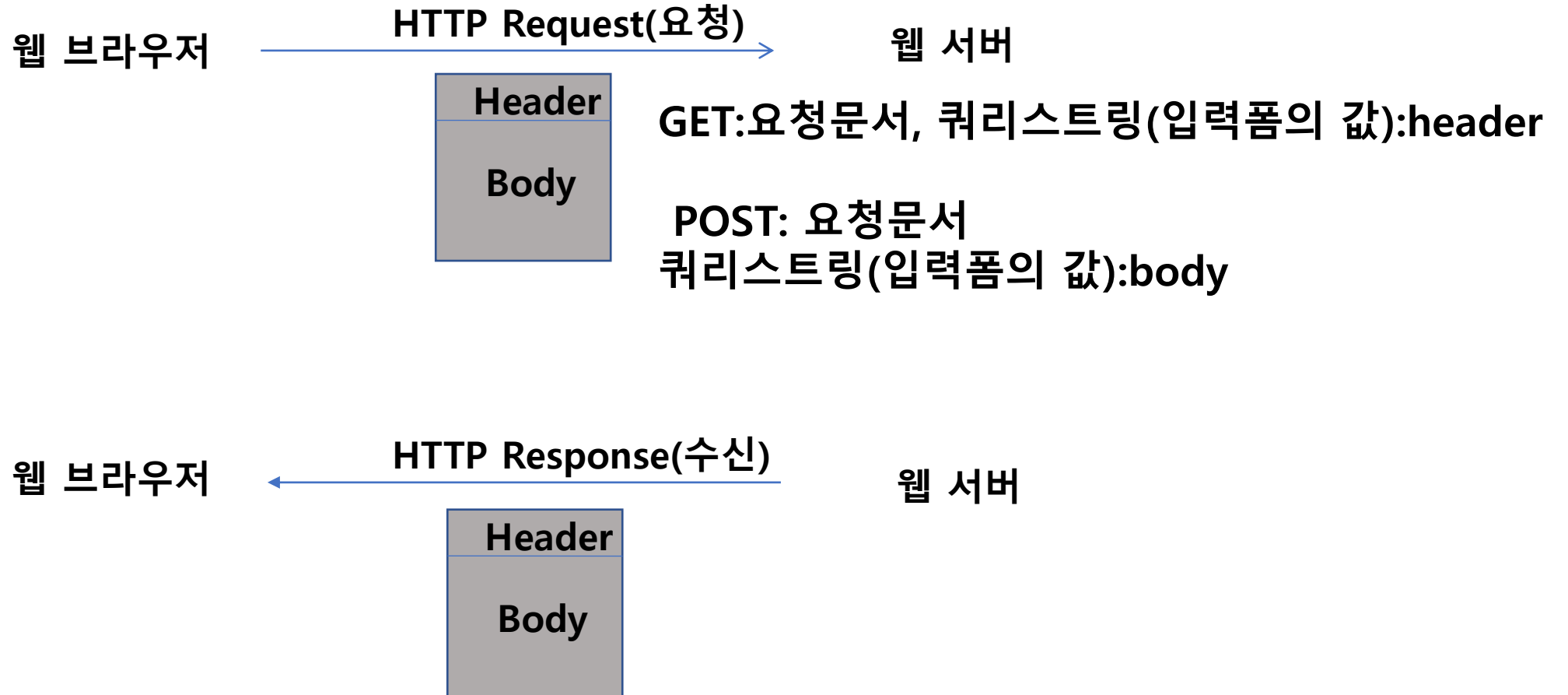
- html5
- css
- javascript
- ECMA

# 웹어플리케이션 아키텍처



# 웹프로토콜의 구조

## 웹 프로토콜 – HTTP Request



# HTTP Request

## ○ HTTP Request에 포함된 상세 정보

① GET /index.html HTTP/1.1

// 요청 URL 정보(메소드, 페이지) 및 HTTP 버전

② user-agent: MSIE 6.0; Windows NT 5.0

// 사용자 웹 브라우저 종류

③ accept: text/html; \*/\*

// 요청 데이터 타입

④ cookie: name = value

// 쿠키(인증 정보)

⑤ referer: <http://www.bbb.com>

// 경유지 URL

⑥ host: www.aaa.kr

// 요청 도메인

## 웹 프로토콜 – HTTP Request

### GET Method

2,083 정도의 길이 데이터만을 처리( 게시판 글 입력 처리 불가 등 )

Method	구조	설 명
GET	GET [request-uri]?query_string HTTP/1.1 Host:[Hostname] 혹은 [IP]	GET 요청 방식은 요청 URI(URL)가 가지고 있는 정보를 검색하기 위해 서버 측에 요청하는 형태

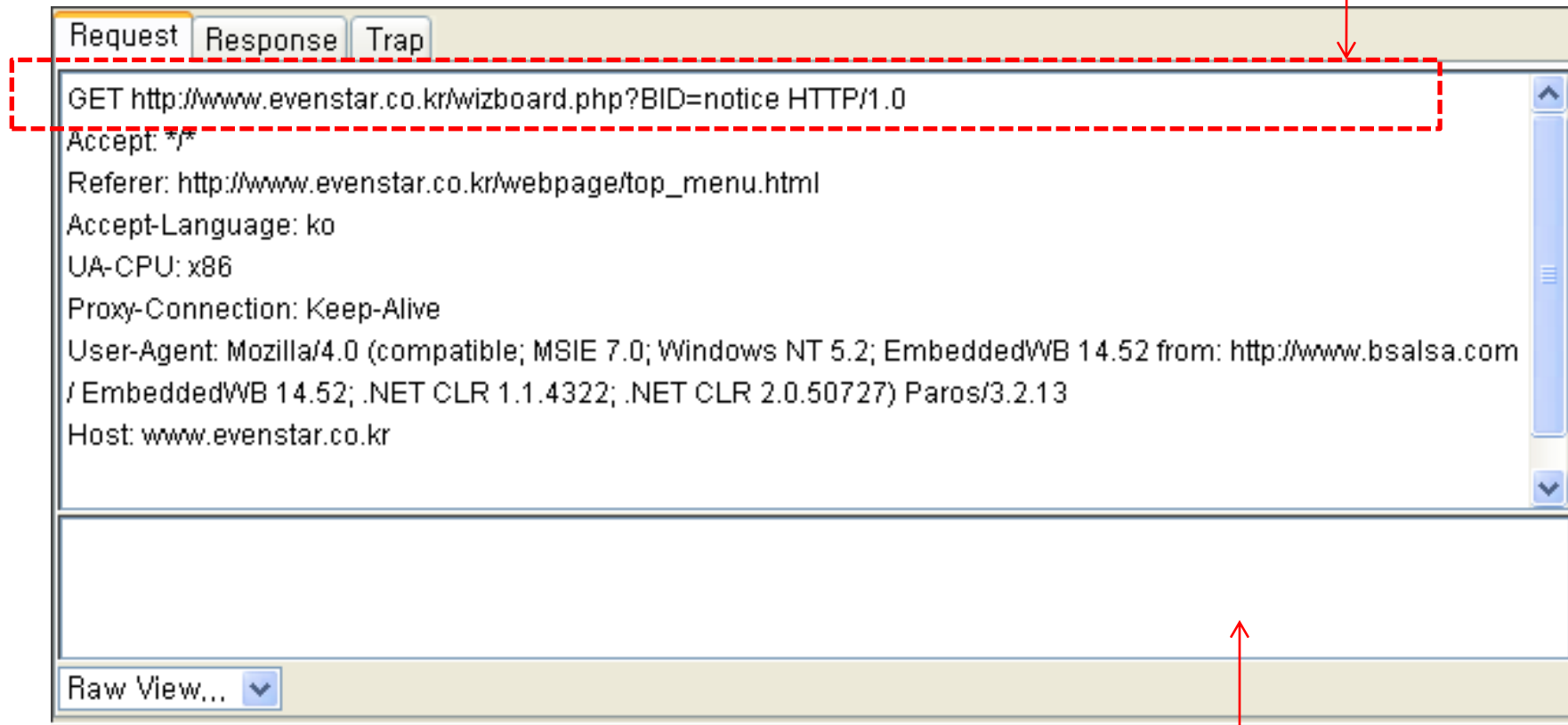
### ○ HTTP GET 구조 (URI + Query String)

<http://www.test.co.kr/a.html> ( http header에 포함 )  
URI

<http://www.test.co.kr/a.jsp?name=kim> ( http header에 포함 )  
URI      Query String

## ○ GET Method

GET 요청



메시지(Body)는 없음



**GET /home/index.html HTTP/1.1**

**Host:www.evenstar.co.kr**

**Accept:text/html, text/plain**

**Accept-Encoding:gzip, compress**

**Accept-Language:ko**

**If-Modified-Since:Sat,31 Jan 2004 12:00:00 GMT**

**User-Agent:Internet Explorer6.0**

-----  
-----

본문 ( GET인 경우는 빈 공백임 )

HTTP Header

요청 Header에는 메소드 종류 및 서버에 전달하는 클라이언트의 정보가 포함된다.

헤더와 본문의 경계 ( 1줄 빈 공백 )

HTTP Body

## 웹 프로토콜 – HTTP Request

### ○ POST Method

길이 제한이 없어 많은 입력 데이터를 처리( 게시판 입력 글 처리 가능 )

Method	전송 형태	설 명
POST	POST [request-uri] HTTP/1.1 Host:[Hostname] 혹은 [IP] Content-Length:[Bytes] Content-Type:[Content Type]	게시판 등과 같은 폼 데이터 페이지를 위해 처리하기 위해 POST 방식으로 전송하게 되며, 웹 브라우저와 시스템 간 데이터 처리로 웹 브라우저에는 페이지 정보만을 확인할 수 있다.
	[query-string] 혹은 [데이터]	

### ○ HTTP POST 구조 (URI + Query String)

http://www.test.co.kr/a.jsp ( http header에 포함 )  
URI

name=kim ( http Body에 포함 )  
Query String

## 웹 프로토콜 – HTTP Response

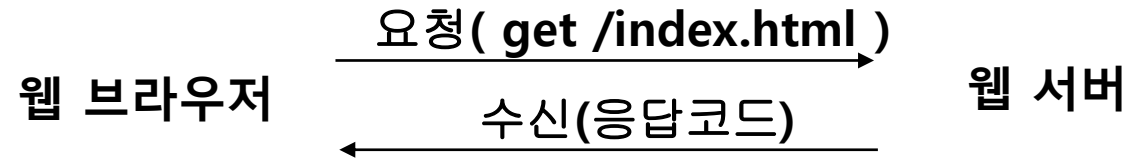
### ○ HTTP Response에 포함된 상세 정보

① HTTP/1.1 OK 200	// 프로토콜 버전 및 응답코드	Header
② Server: NCSA/1.4.2	// 웹 서버 배너 정보	
③ Content-type: text/html	// MIME타입	
④ Content-length: 107	// HTTP Body 사이즈	
빈 공백 1줄		
⑤ <html> <head> </head> <Title>test</Title> <body> hello web </body> </html>	// 페이지 구성 정보(HTML태그 등)	Body

- HTTP Header 포함 범위 : (1), (2), (3), (4)
- HTTP Body 포함 범위 : (5)

# 웹 프로토콜 – HTTP Response

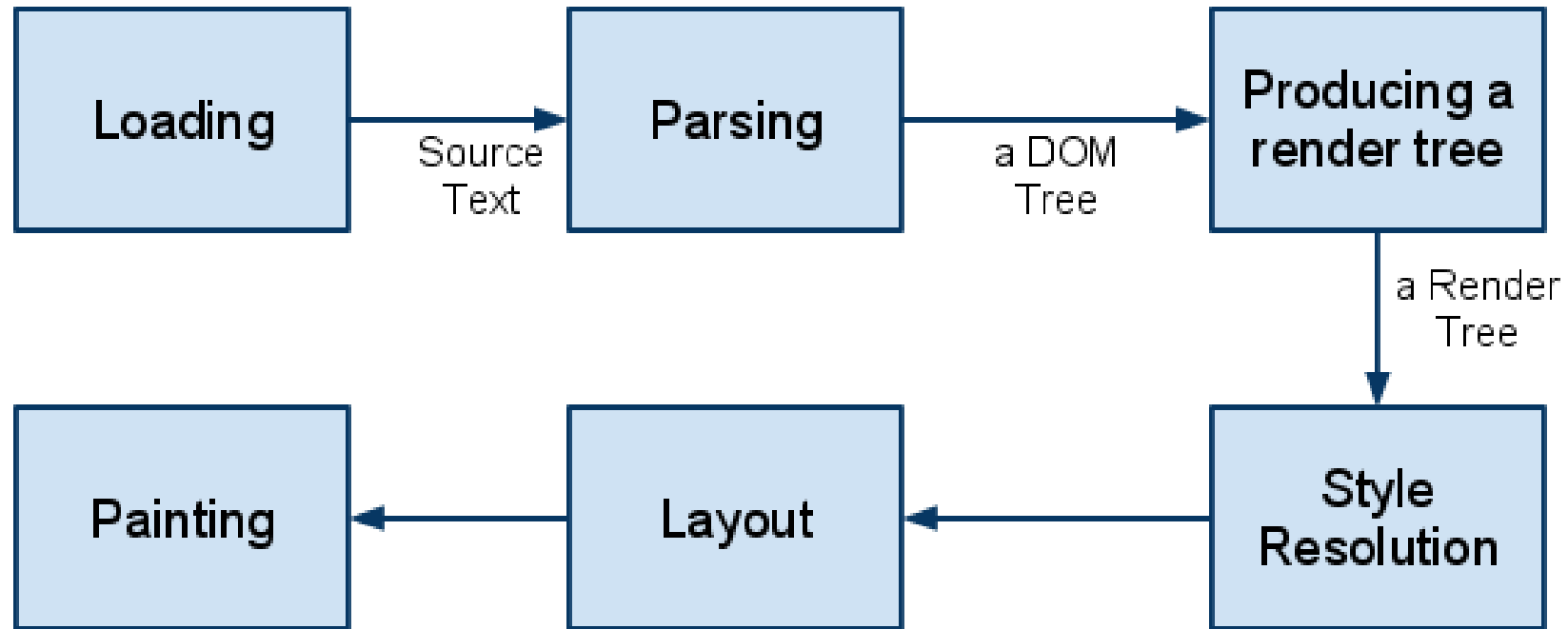
## HTTP Status Code(응답코드) 종류



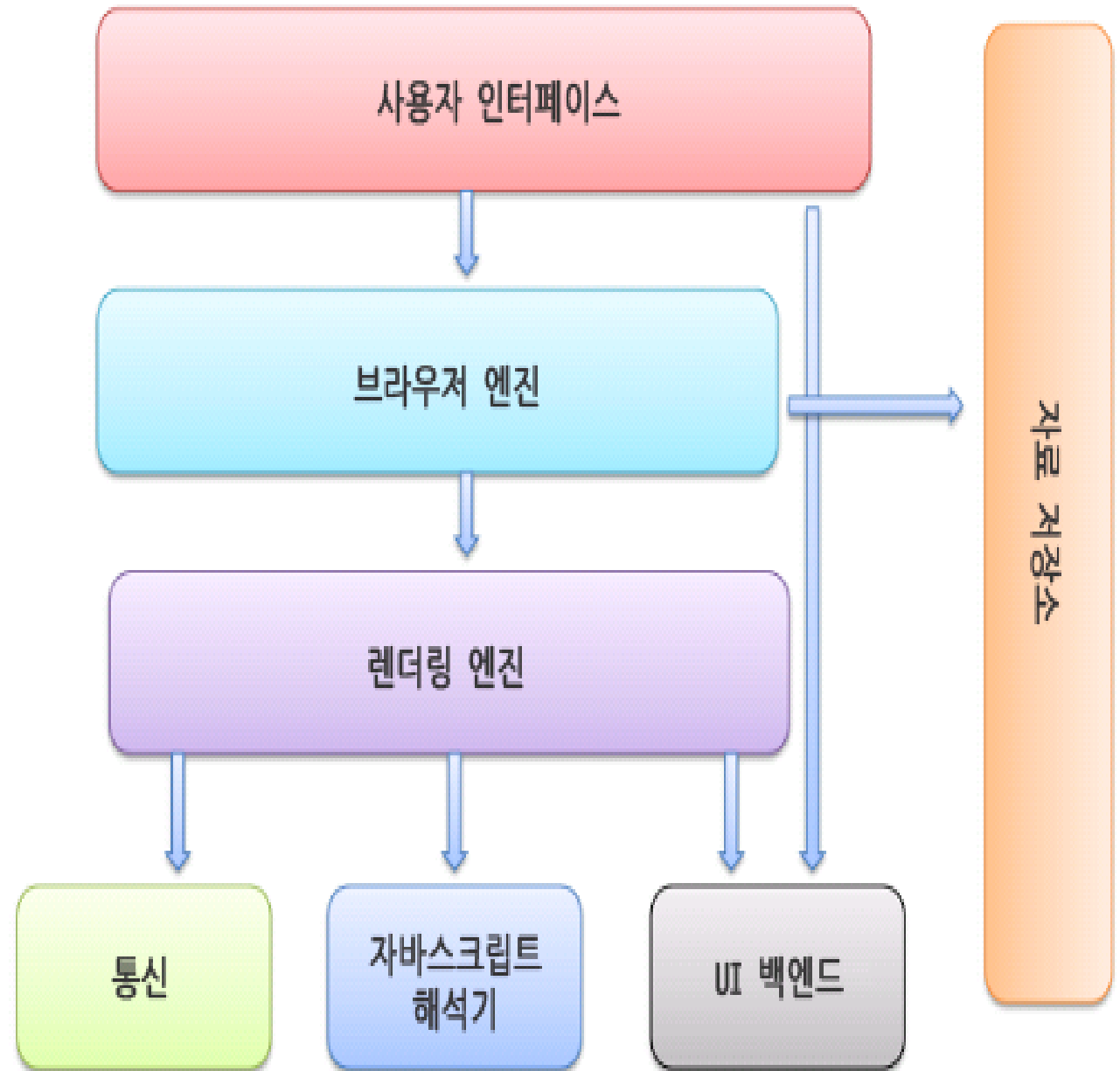
- 200번(요청성공)
- 201번(원격지서버에 파일 생성)
- 302번(페이지이동)
- 304번(로컬 캐쉬정보이용)
- 401번(인증실패)
- 403번(접근금지)
- 404번(페이지없음)
- 500번(서버에러)

# 브라우저의 동작구조

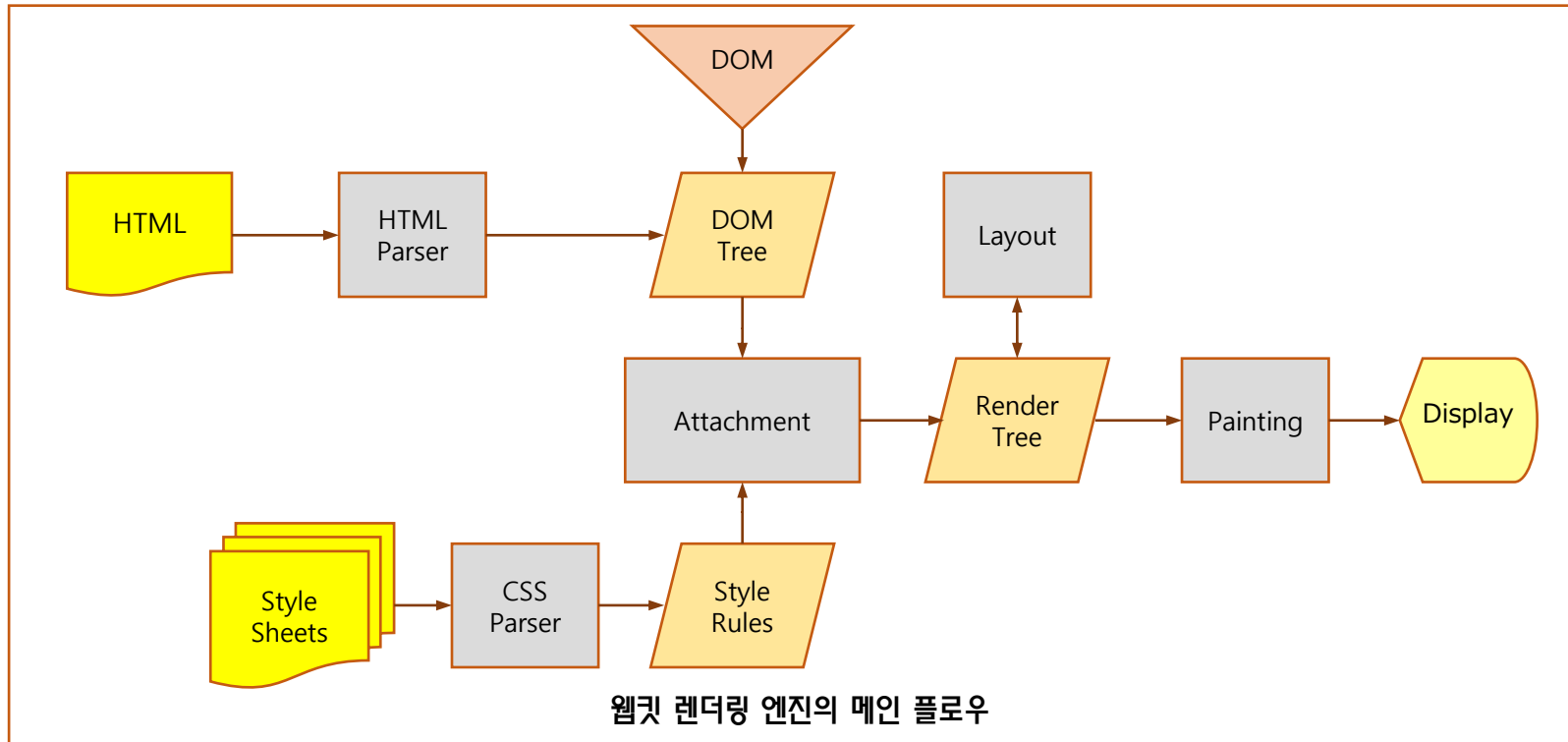
# 브라우저의 렌더링엔징



IE : Trident  
크롬 : Webkit  
파이어폭스 : Gecko  
사파리 : Webkit  
오페라 : Presto



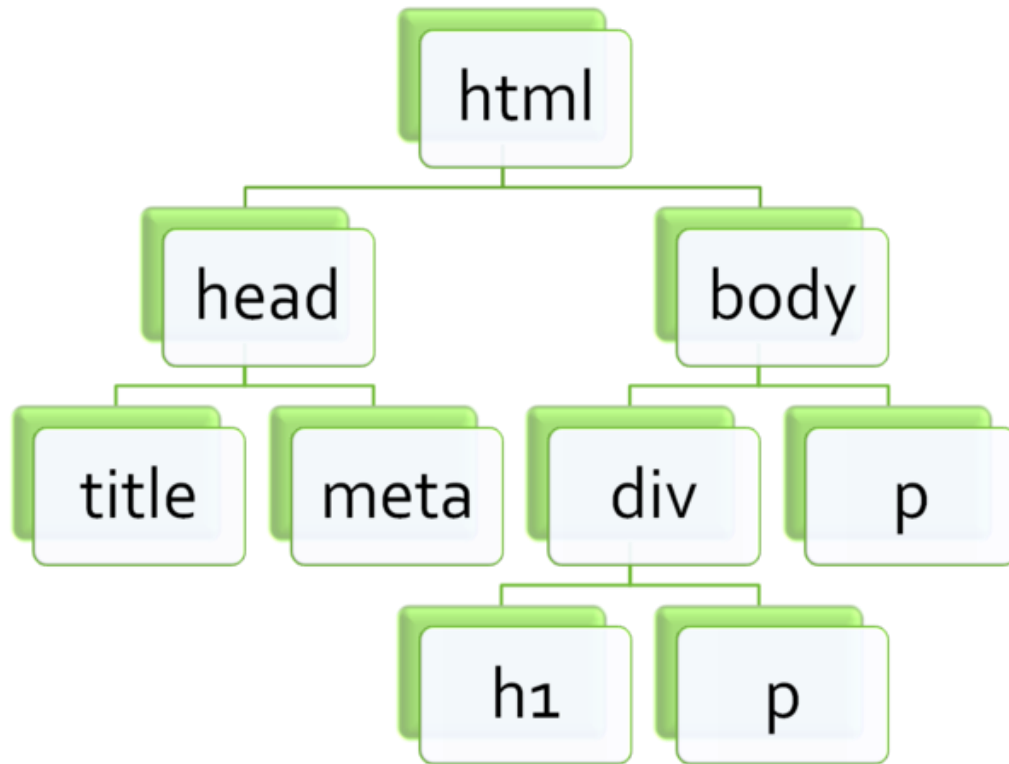
# 렌더링 엔진의 메인 플로우





# DOM(브라우저 렌더링에 의해 트리형태로 구성된 파싱문서)

객체: 속성, 메소드  
객체.속성  
객체.메소드



자바스크립트  
(DOM 트리 접근자)

front end 종류

# EcmaScript version 5, 6

European Computer Manufacturers Association (ECMA)

웹표준 프론트엔드 언어

[https://www.w3schools.com/js/js\\_es6.asp](https://www.w3schools.com/js/js_es6.asp)

# jquery



- 2006년에 도입된 jQuery는 초기 프론트엔드 프레임워크
- 현재도 광범위하게 사용하는 인기있는 프레임워크
- jQuery는 간단하고 사용하기 쉬울 뿐만 아니라 광범위한 JavaScript 코드를 작성할 필요성도 최소화
- 수년 동안 존재한 덕분에 솔루션을 위한 상당한 jQuery 커뮤니티가 있음
- CSS와 DOM을 조작하고 웹 사이트의 기능과 상호 작용을 최적화하기 위해 사용
- 모든 브라우저를 지원

<https://www.w3schools.com/jquery/>

[cheatsheet https://oscarotero.com/jquery/](https://oscarotero.com/jquery/)

# jquery chaining



# jquery 장점 단점

## 장점:

DOM은 요소를 추가하거나 제거하기 위해 유연  
HTTP 요청 전송이 단순화됨  
동적 콘텐츠 촉진

## 단점:

상대적으로 느린 작업 능력  
유지보수가 어렵다(대형프로젝트에 적합하지 않음)

## 사용 시기:

jQuery는 데스크톱 기반 javascript 애플리케이션을 개발하는 데 사용  
코드를 간결하고 매우 단순하게 유지  
이벤트 처리 및 애니메이션 수행에 사용.

## 사용하지 않을 경우:

대규모 애플리케이션을 개발할 때는 javascript 코드를 많이 추가하여 애플리케이션을 무겁게 하기 때문에 jQuery를 사용할 수 없다.  
자바스크립트의 진보된 촉진, 낮은 코딩 및 구성요소의 재사용성을 가지고 현대의 프레임워크와 경쟁할 수 없다.

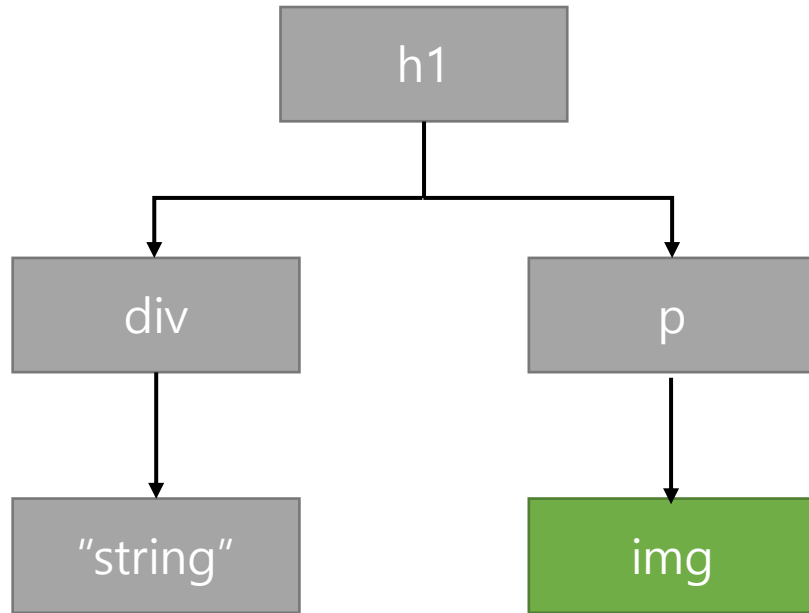
# vue.js



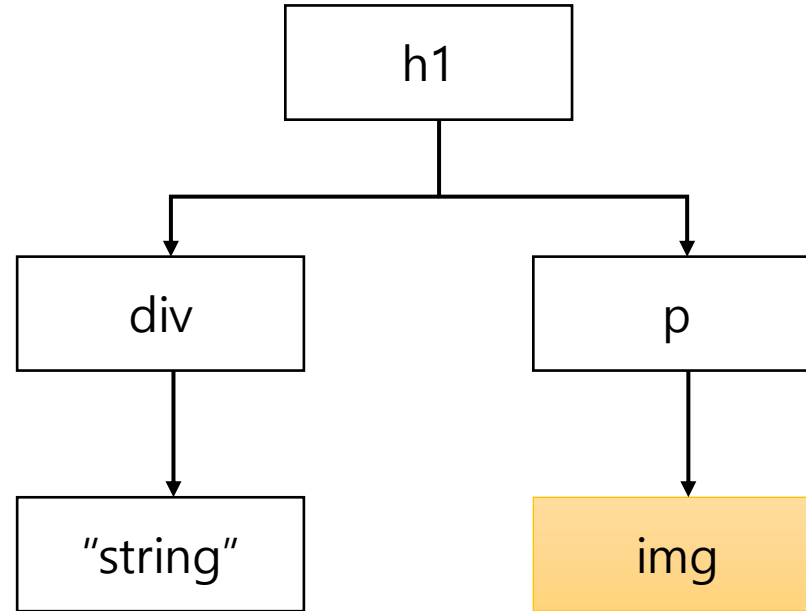
- 오늘날 가장 인기 있는 프론트 엔드 프레임워크 중 하나
- Vue는 간단한 프레임워크이다.
- 앵글러 개발자들이 겪는 복잡성을 없애는 데 좋다.
- 크기는 더 작으며 시각적 DOM과 구성요소 기반이라는 두 가지 주요 이점을 제공합니다. 또한
- 2방향 바인딩
- 웹 애플리케이션과 모바일 애플리케이션 구축부터 진보적인 웹 애플리케이션까지 간편하고 역동적인 프로세스를 모두 손쉽게 처리할 수 있다.
- 앱 성능을 최적화하고 복잡성을 타개하기 위해 구축
- 알리바바, 로이터, 샤오미 등은 이 프레임워크 사용중

[https://www.w3schools.com/whatis/whatis\\_vue.asp](https://www.w3schools.com/whatis/whatis_vue.asp)

Virtual DOM



DOM





# vue.js 장단점

## 장점:

- 광범위하고 상세한 설명서
- 간단한 구문 - 자바스크립트 배경을 가진 프로그래머들은 Vuejs로 쉽게 시작할 수 있다.
- 애플리케이션 구조 설계의 유연성

## 단점:

- 구성 요소의 안정성 결여
- community 가 아직까지 적다
- 플러그인 및 구성 요소를 포함한 언어 장벽(대부분의 플러그인은 중국어로 작성됨)

## 사용:

유연한 설계 구조에는 vue를 사용하는 것이 좋습니다.

모든 것을 처음부터 설계할 수 있게 해주고 대형 프로젝트 개발에도 좋음

## 사용하지 않을 경우:

복잡성에 대한 답을 얻기 위한 지원 커뮤니티가 부족하다

또한 안정적인 구성 요소가 필요한 애플리케이션은 프레임워크가 구성 요소의 안정성에 문제를 보여왔기 때문에 Vuejs로 구축하기에 적합하지 않다.

# angular

- Angular는 2010년 Google에 의해 처음 개발된 TypeScript 기반의 JavaScript 프레임워크
- AngularJS는 자바스크립트로 만든 client 측 MVC/MVVM 프레임워크로 모던 단일 페이지 웹 애플리케이션 개발의 정수
- 자바스크립트로 작성할 코드량을 줄여준다.
- Dom을 선택하고 조작하는 자바스크립트 코드를 작성하지 않아도 된다.
- 양방향 데이터 바인
- 모델의 데이터와 뷰 데이터가 양방향 데이터 바인딩이 되어, 모델이 바뀌면 뷰 데이터도 같이 변경
- HTML, CSS, 로직 등의 개발 영역을 명확하게 분리해줍니다.
- 기존 자바스크립트에서는 Dom 조작과 이벤트 처리를 위해 HTML을 잘 알고 있어야 했으나, AngularJS는 뷰 코드와 로직 코드가 명확히 분리

참고:<https://www.w3schools.com/angular/>

참고: <https://ithub.tistory.com/68>

장점:

- 이원 데이터 바인딩과 같은 대부분의 주요 기능이 기본적으로 제공되므로 코드 양을 줄여줌
- 구성 요소를 외부 요소를 정의하여 종속성에서 분리합니다.
- 구성 요소를 재사용할 수 있으며 종속성 주입을 사용하여 관리하기 쉽습니다.
- 학습 및 지원을 위한 광범위한 커뮤니티

단점 :

- 앵글롤러는 완전한 동적 솔루션이기 때문에 작업을 수행할 수 있는 여러 가지 방법이 있으므로 학습이 쉽지 않다.
- 동적 앱은 구조와 크기가 복잡하기 때문에 성능이 좋지 않다.

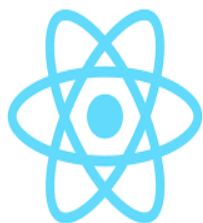
**angular 사용:**

양방향 데이터 바인딩을 사용하기 때문에 콘텐츠를 순식간에 동적으로 업데이트하여 브라우저 기반 애플리케이션의 성능을 각도로 향상시킵니다. 엔터프라이즈 기반 애플리케이션과 동적 웹 애플리케이션의 경우 angular 사용하는 것이 가장 좋습니다.

**사용 안 할 경우**

angular 는 프론트엔드 프레임워크로서의 완전한 솔루션이다. 제한된 범위의 애플리케이션을 빌드하려면 앵글러가 제공하는 리소스를 사용할 수 없다. 또한 더 작은 크기의 팀이 있는 경우 더 적은 복잡성과 간단한 구문으로 더 작은 프레임워크를 선택.

# react



- 가장 간단한 프레임워크 중 하나인 React는 페이스북에서 앱의 기능이 지속적으로 추가되어 코드 유지관리성 문제를 해결하기 위해 개발
- 오픈 소스 프레임워크인 React는 뛰어난 기능을 제공하는 가상 문서 객체 모델(DOM)을 사용한다.
- 높은 트래픽이 예상되며 이를 처리하기 위한 안정적인 플랫폼이 필요한 사람에게 이상적인 프레임워크.
- 기술 컨설턴트로서, 단일 [페이지](#) 웹 [애플리케이션](#) 및 PWA [구축](#)과 관련된 프로젝트에 대해 React를 추천.

장점:

- 구성요소를 재사용할 수 있으므로 응용프로그램의 다른 부분에서 구성요소를 쉽게 협업하고 재사용할 수 있다.
- 가상 DOM 사용으로 일관되고 원활한 성능 제공
- React 후크에서 구성 요소를 쓰는 가장 좋은 대안으로서, 클래스 없이 구성 요소를 작성할 수 있으며 React를 보다 쉽게 학습할 수 있습니다.
- React Dev 도구는 고급이며 매우 유용.

## **단점:**

- 프레임워크의 다중적이고 지속적인 업데이트로 인해 적절한 설명서를 만드는 것이 어렵고 초보자가 학습시 어렵다.
- JSX의 복잡성을 이해하기 어렵다.

## **React 사용 :**

- React는 특히 단일 페이지 응용프로그램을 개발하려는 경우 사용자 인터페이스를 구축하는 데 사용됩니다.
- 구성요소를 재사용할 수 있으므로 시간을 단축하고 대화형 인터페이스를 개발하고자 할 때 가장 강력한 프론트엔드 프레임워크

## **React를 사용하지 않을 경우:**

- Javascript에 대한 실제 경험이 없는 경우 React는 권장되는 옵션이 아닙니다.
- 또한 경험이 부족한 개발자들에게는 JSX 학습이 어렵다.

# ember.js



- 2011년에 개발된 Eberjs는 구성요소 기반이며 앵글러와 유사한 이원 데이터 바인딩을 제공
- 현대 기술의 증가하는 수요를 원활하게 처리하도록 설계되됨.
- Eberjs를 사용하여 복잡한 모바일 및 웹 애플리케이션을 개발할 수 있으며, 효율적인 아키텍처로 문제를 해결할 수 있다.
- 학습이 어렵다.
- 개발자 커뮤니티는 작다.

### **장점:**

- 잘 조직된
- 가장 빠른 프레임워크
- 이원 데이터 바인딩
- 적절한 문서

### **단점:**

- 작은 커뮤니티, 덜 인기 있는
- 복잡한 구문 및 느린 업데이트
- 하드러닝곡선
- 소규모 애플리케이션을 위한 무거운 프레임워크

### **사용 시기:**

풍부한 사용자 인터페이스를 갖춘 Linkedin과 같은 현대적인 애플리케이션을 구축하고자 한다면, Eberjs는 Eberjs가 제공하는 뛰어난 라우팅 덕분에 다양한 애플리케이션 상태를 보는 것과 같은 모든 기술적 프론트엔드 축진을 갖춘 프레임워크입니다.

이 프레임워크는 필요에 따라 페이지를 렌더링할 수 있는 준비된 구성, 유용한 바인딩 및 사용자 지정 속성을 제공하기 때문에 대규모 프로젝트를 위한 완벽한 프론트엔드 솔루션입니다.

### **사용하지 않을 경우:**

엠버즈는 복잡한 문제를 해결하기 위해서는 경험과 비즈니스 논리가 필요하기 때문에 소규모 개발 팀에서는 올바른 선택이 아닙니다. 초기 비용은 엠버지가 더 높을 수 있다. 또한 간단한 ajax 기능을 작성하고 간단한 사용자 인터페이스를 구현하는 경우 프레임워크가 올바른 선택이 아닐 수 있습니다.

# backbone.js



BACKBONE.JS

- 가장 쉬운 프레임워크 중 하나인 backbone.js를 사용하면 단일 페이지 애플리케이션을 신속하게 개발할 수 있다.
- MVC 아키텍처를 기반으로 하는 프레임워크이다.
- 컨트롤러와 비슷하게 MVC 아키텍처의 뷰는 요소 로직을 구현할 수 있다.



## 장점 :

- 가장 빠른 자바스크립트 프레임워크 중 하나
- 배우기 쉽다
- 경량 프레임워크

## 단점:

- 앱 구조를 만들기 위한 기본 도구 제공(프레임이 준비된 구조를 제공하지 않음)
- 뷰를 모델 및 모델로 전달하기 위해 보일러 플레이트 코드를 작성해야 합니다.

## 사용 시기:

백본js는 Trello와 같은 동적 용도에 사용됩니다. 이를 통해 개발자는 클라이언트 측 모델을 구축하고, 업데이트를 더 빠르게 하며, 코드를 재사용할 수 있습니다. 따라서 업데이트를 동적으로 처리하고 클라이언트를 유지하며 서버와 지속적으로 동기화하는 데 효율적입니다.

## 사용하지 않을 경우:

backbone.js는 다른 MVC 클라이언트 측 프레임워크에 비해 웹 애플리케이션을 구축하기 위한 최소한의 요구 사항을 제공합니다. 그러나 플러그인 및 확장을 지원하여 기능을 확장할 수 있습니다. 따라서 개발 팀이 하나의 프레임워크에서 완전한 솔루션을 목표로 하는 상황에서 backbone.js를 목표로 해서는 안 된다

# frontend UI

jquery ui: <https://jqueryui.com/>

jquery mobile: <https://jquerymobile.com/>

bootstrap: <https://getbootstrap.com/>  
<https://www.w3schools.com/bootstrap4/>

semantic ui: <https://semantic-ui.com/>