WEB

0.WEB

web의 아버지

팀 버너스 리 경 (Sir. Tim Berners-Lee)

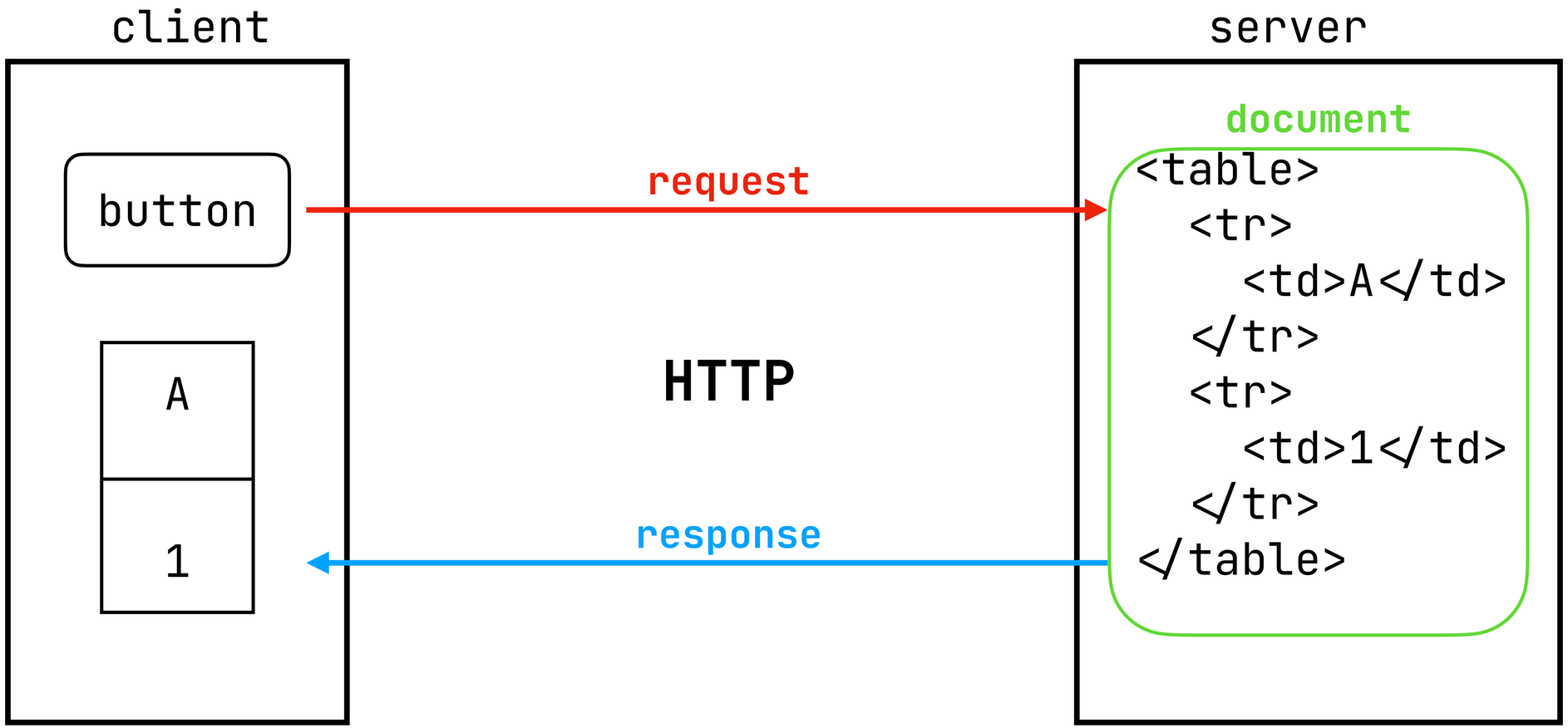
- 영국의 컴퓨터 과학자

- WWW, URL, HTTP, HTML 설계 등

- W3C (Wolrd Wide Web Consortium)

0.WEB

요청과 응답



* HTTP(HyperText Transfer Protocol) : client와 server 사이의 통신 규약

1.HTML

HyperText Markup Language

- 온라인 상의 문서(document)를 만들기 위해 데이터를 구조화 시키는 언어

```
<!DOCTYPE html>                          문서의 형식 → html
<html>
  <head>                                 문서를 정의
    <title>hello</title>
  </head>
  <body>                                 문서의 내용
    <p>Hello, World!</p>
  </body>
</html>
```

1.HTML

해당 요소의 내용

여는 태그 : 태그의 시작을 의미

닫는 태그 : 태그의 끝을 의미

속성 : 해당 태그에 적용되는 기능

요소 : <여는 태그> 내용 </닫는 태그>

```
<p style="color:red;">  Hello, World!  </p>
```

1.HTML

<!DOCTYPE>

DTD (Document Type Definition)

html4

- strict : 선언된 html 버전의 문법과 구조를 정확하게 사용

<!DOCTYPE html PUBLIC “-//W3C//DTD HTML 4.01 Strict//EN” “http://www.w3.org/TR/html4/strict.dtd”>

- transitional : 선언된 html 이외의 문법과 구조를 허용

<!DOCTYPE html PUBLIC “-//W3C//DTD HTML 4.01 Transitional//EN” “http://www.w3.org/TR/html4/transitional.dtd”>

- frameset : transitional + frame 지원

<!DOCTYPE html PUBLIC “-//W3C//DTD HTML 4.01 Frameset//EN” “http://www.w3.org/TR/html4/frameset.dtd”>

html5 : <!DOCTYPE html>

1.HTML

<!DOCTYPE>

```
<!DOCTYPE html PUBLIC “-//W3C//DTD HTML 4.01 Frameset//EN” “http://www.w3.org/TR/html4/frameset.dtd”>
```

이 문서는 **html** 로 작성하고,

국제 공용 문서 이며,

W3C 기관의 문서형식에 의해 html4.01 버전을 frameset 방식의 영어로 출력하며,

참조할 dtd 문서는 해당 링크에 있다.

1.HTML

<head>

문서를 정의하는 기본 태그

<meta> : 문서의 기본 정보 등

<title> : 문서의 제목

<script> : javascript 등

<style> : css 등

<link> : 외부 문서 연결

1.HTML

블록 요소

- 줄 바꿈
- 블록 요소 안에 텍스트와 인라인 요소 포함 가능
- 블록 요소 안에 블록 요소 포함 가능 (일부 불가)

address	article	aside	audio	blockquote	canvas	dd	div	dl	fieldset	figcpation
figure	footer	form	h1	h2	h3	h4	h5	h6	header	hgroup
hr	noscript	ol	output	p	pre	section	table	tfoot	ul	video

1.HTML

인라인 요소

- 줄 바꿈 없음
- 인라인 요소 안에 텍스트와 인라인 요소 포함 가능
- 인라인 요소 안에 블록 요소 포함 불가

a	abbr	acronym	b	bdo	big	br	button	cite	code	dfn
em	i	img	input	kbd	label	map	object	output	q	samp
script	select	small	span	strong	sub	sup	textarea	time	tt	var

1.HTML

		‘	&apos;
&	&amp;	“	&quot;
<	&lt;	©	&copy;
>	&gt;		

1.HTML

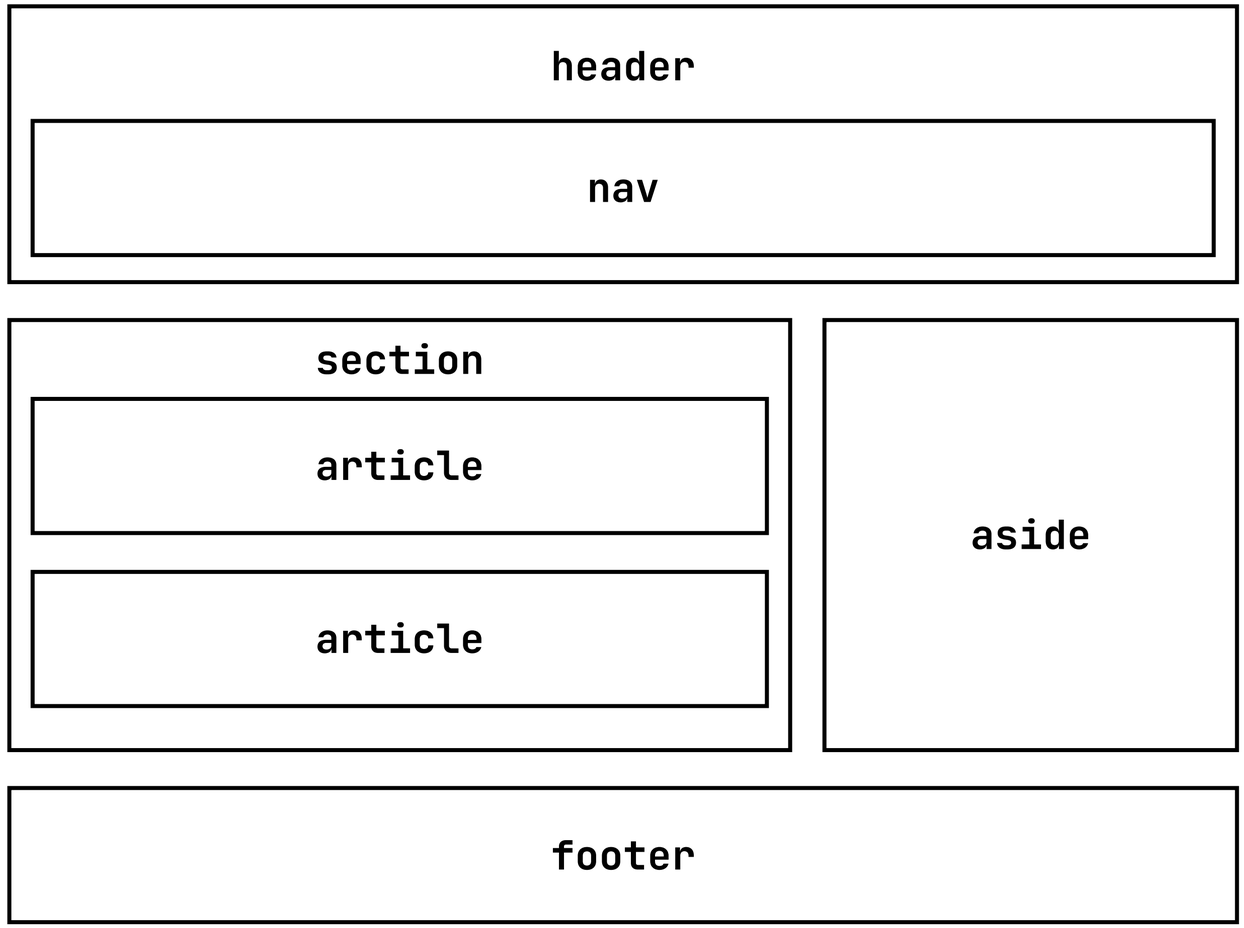
semantic tag

```
<header>
  <nav>
  </nav>
</header>

<section>
  <article></article>
  <article></article>
</section>

<aside></aside>

<footer></footer>
```



1.HTML


```
<form action='경로' method='전송 방식'>
  <input type='형태' name='name' value='value'/>
  <input type='submit' value='전송'/>
</form>
```

* 전송 방식

GET request header에 data를 담아서 전송 (queryString)

POST request body에 data를 담아서 전송

* queryString url?name=value&name=value 형태

2.CSS

Cascading StyleSheets

- 문서의 스타일을 지정해주는 언어

사용 방법

- inline : 태그 안에서 사용

```
<tag style="속성: 값;" ></tag>
```

- 내부 : head 안에서 사용

```
<style></style>
```

- 외부 : link 태그 사용

```
<link href="*.css"
      rel="stylesheet"
      type="text/css">
```

2.CSS

selector

type	태그 이름을 사용하여 선택	tag
id	태그에 지정된 id속성의 값을 사용하여 선택	#id
class	태그에 지정된 class속성의 값을 사용하여 선택	.class
전체	전체 태그를 선택	*
자식	부모 . 자식 관계에서 자식 요소 선택	tag > tag

2.CSS

selector

인접	지정한 요소 다음에 나오는 요소 선택	`tag + tag`
하위	요소 하위의 요소를 선택	`tag selector`
속성	속성이 정의된 태그를 선택	`tag[attr]`
가상 클래스	요소의 상태에 따라 선택자 지정	`tag:class`

2.CSS

[attr]	attr 속성을 가진 요소 선택
[attr=value]	attr 속성의 값이 value와 일치하는 요소 선택
[attr~=value]	attr 속성의 값이 value와 일치하는 요소 선택 (공백 포함)
[attr|= value]	value로 시작하는 요소들을 선택 (- 포함)
[attr^=value]	attr 속성의 값이 value로 시작하는 요소들을 선택
[attr$=value]	attr 속성의 값이 value로 끝나는 요소들을 선택
[attr*=value]	attr 속성의 값에 value를 포함하고 있는 요소들을 선택

2.CSS

- 스타일 선언은 위에서 아래로 순차적으로 실행되면서 마지막에 선언된 스타일이 우선순위를 가짐

- 특정 요소에 중복 선언한 경우
  `inline > id > class > type`

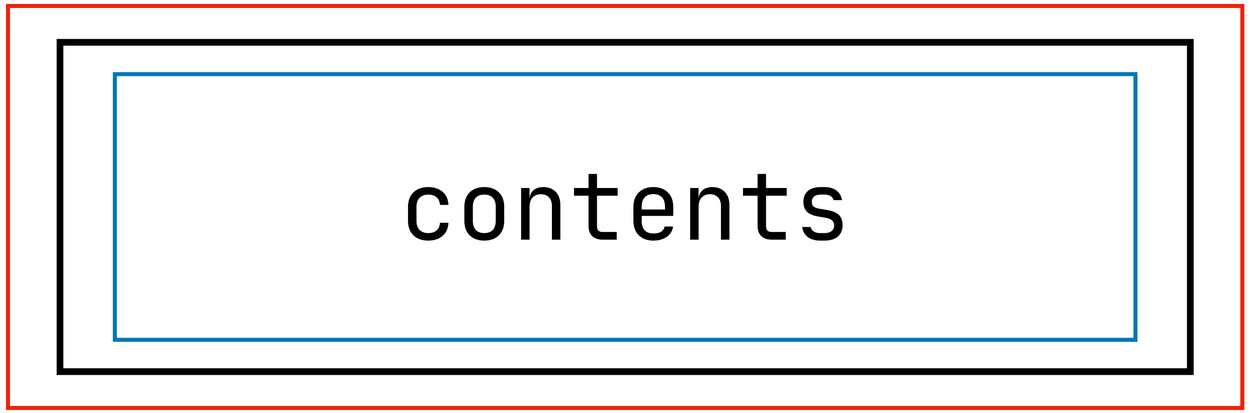
- 강제 우선순위 선언
  `!important > inline > id > class > type`

2.CSS

margin 바깥여백

border 테두리

padding 안쪽여백



* margin 병합

element 사이의 margin 간격은 겹침!

3.JavaScript

script

1. 소프트웨어에 실행시키는 처리 절차를 문자(텍스트)로 기술한 것.

Script language

응용 소프트웨어를 제어하는 컴퓨터 프로그래밍 언어를 가리킨다.

3.JavaScript

역사

1993	Mosaic web browser → Netscape
1994	Mocha → LiveScript → JavaScript
1995	Netscape에 JS interpreter 탑재 MS JScript → IE 출시
1997	ECMAScript 1
1998	ECMAScript 2
1999	ECMAScript 3
2000	ECMAScript 4 → IE 표준 선언
2004	Firefox AJAX(Asynchronous Javascript And Xml)
2008	Chrome, JIT(Just In Time compilation)
2009	ECMAScript 5 (우리가 지금 쓰는 JS)

3.JavaScript

ES6 (ECMAScript2015)

ES7 (ECMAScript2016)

ES8 (ECMAScript2017)

ES9 (ECMAScript2018)

ES10 (ECMAScript2019)

ES11 (ECMAScript2020)

ES12 (ECMAScript2021)

...

* ES6 에서 많은 변화!

3.JavaScript


```
var 변수 = 값(literal);
```

* function scope

```
let 변수 = 값;
const 상수 = 값;
```

block scope

* type inference (타입 추론) : literal에 맞는 type으로 자동 변환

* type

- string
- number
- boolean
- Null
- object
- function

3.JavaScript

명시적 함수

```
function 함수(파라미터){
    명령어;
}
```

익명 함수

```
변수 = function(파라미터){
          명령어;
      }
```

클로저 (closure) : lexical scope

```
function 외부함수(파라미터){
  명령어;
  function 내부함수(파라미터){
     명령어;
  }

}
```

3.JavaScript


```
function
function 함수(파라미터){
    this.변수;
    var 변수;
}

var 변수 = new 함수();
```

```
object literal
변수 = {
    key: value,
    key: value,
    ...
}

var 변수 = new 변수();
```

* 객체의 구성
property (속성)
function (기능)

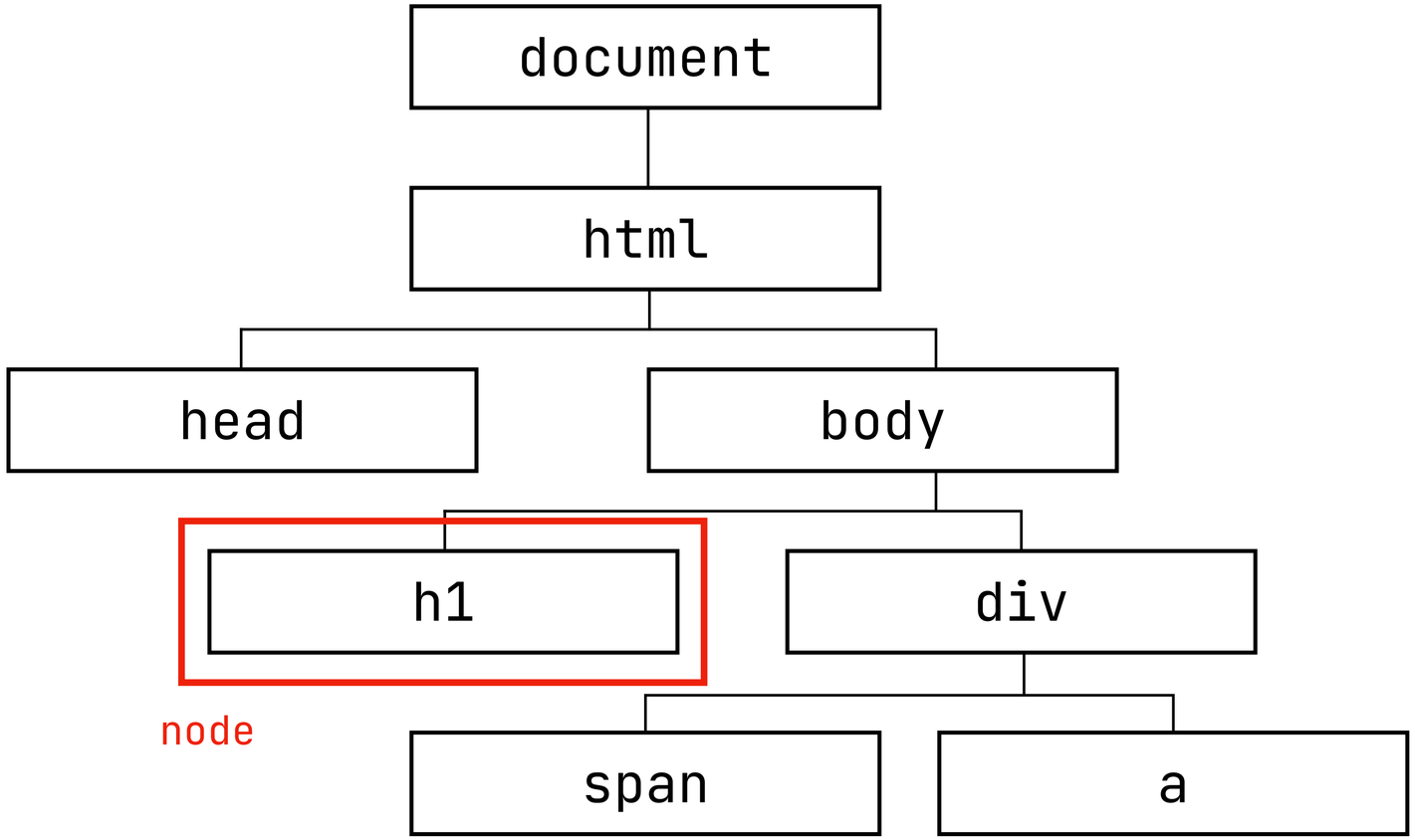
* prototype : 객체의 확장

3.JavaScript

Document Object Model

문서를 Node로 변환하여 특정 Node에 접근

```
<html>
 <head>
 </head>
 <body>
  <h1></h1>
  <div>
    <span>test</span>
    <a href=‘’>test</a>
  </div>
 </body>
</html>
```



Asynchronous Javascript And Xml

XMLHttpRequest (XHR)
http를 통한 데이터 송수신 지원 객체

* readyState

0 : uninitialized
1 : loading
2 : loaded
3 : interactive
4 : complete

* status

200 : 성공
400 : bad request
401 : unauthorized
403 : forbidden
404 : not found
415 : unsupported media type
500 : internal server error