HTML / CSS

Web과 HTTP

Front End / Back End

- 웹 개발은 Front End 개발과 Back End 개발로 나누어 진다.

- **Front End**
  - 사용자가 보고 상호작용(UI/UX)하는 부분.
    - 브라우저, 앱에서 실행되는 클라이언트 로직
    - UX 디자이너 – User Experience (사용자 경험) 설계
    - UI 디자이너 – 화면 구성과 시각적 요소 디자인
    - Front End 개발자 – 브라우저/앱에서 실행되는 **사용자** 측 애플리케이션 구현

- **Back End**
  - 사용자 요청을 처리하는 서버 부분을 개발한다.
    - 서버 프로그램 개발자 - API, 인증, 비즈니스 로직, 데이터 처리
    - DB 관리자(DBA) - 데이터 저장 구조 설계, 최적화, 보안, 백업
    - DevOps/인프라 엔지니어 - CI/CD, 배포 자동화, 모니터

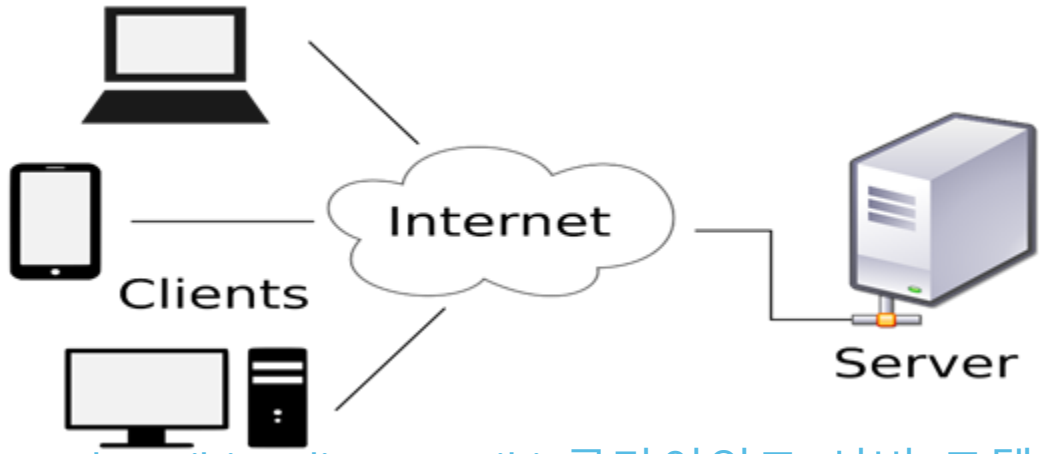
- Full Stack
  - Front End와 Back End를 모두 설계 구현 할 수 있는 사람을 Full Stack 개발자라고 한다.

인터넷

- **네트워크** (Network)
  - 컴퓨터와 컴퓨터를 연결한 것
- **인터넷** (Internet)
  - 전세계 컴퓨터 들을 연결한 통신망
- **프로토콜** (Protocol)
  - 네트워크 통신 규약
  - 네트워크로 연결된 서로 다른 컴퓨터들끼리 데이터를 주고 받을 때 지켜야 하는 약속/규약이다.
  - 대표적인 프로토콜
    - TCP: 전송방법을 정의한 프로토콜로 신뢰성있는 데이터 전송을 보장한다.
      - HTTP, FTP 등(응용계층) 의 기반 프로토콜
    - UDP: 전송방법을 정의한 프로토콜로 빠른 데이터 전송을 목적으로 한다.

서버(Server)/클라이언트(Client)

- 서버 (Server)
  - 네트워크 상에서 서비스 제공자
- 클라이언트 (Client)
  - 네트워크 상에서 서비스 요청자
- 서버-클라이언트 구조
  - 서버와 클라이언트 간의 작업을 분리해 주는 네트워크 아키텍처(구조)

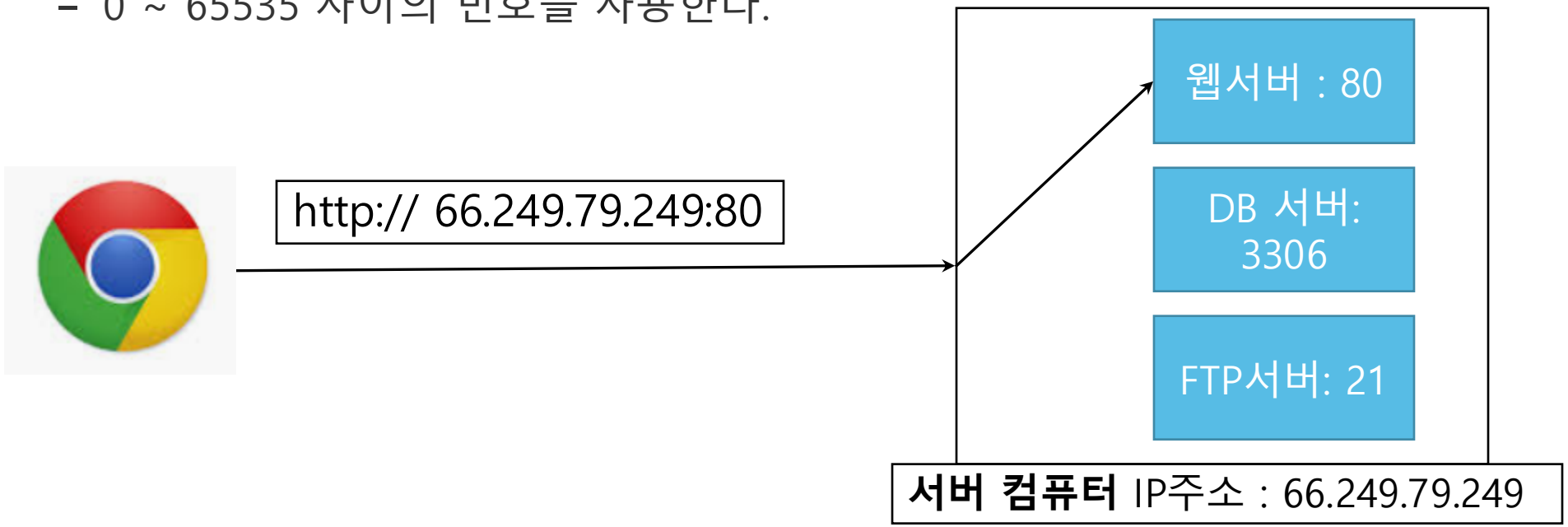


https://ko.wikipedia.org/wiki/클라이언트_서버_모델

IP 주소, Port 번호, URL

- **IP 주소**
  - 인터넷 상에 연결된 컴퓨터의 고유한 주소
  - 32비트 주소공간으로 구성된 IPv4 체계로 구축되어 왔으나 주소가 소진되고 있어 그 대안으로 128 비트 주소공간을 가지는 IPv6 프로토콜이 제안되어 적용되고 있다.
  - **Domain 주소:** 숫자로 된 IP 주소를 문자 기반에 기억하기 쉬운 이름으로 표현한 주소

- **PORT(포트) 번호**
  - 서버 컴퓨터내에서 서비스하는 네트워크 (서버) 프로그램들을 구분하기 위한 번호
  - 0 ~ 65535 사이의 번호를 사용한다.



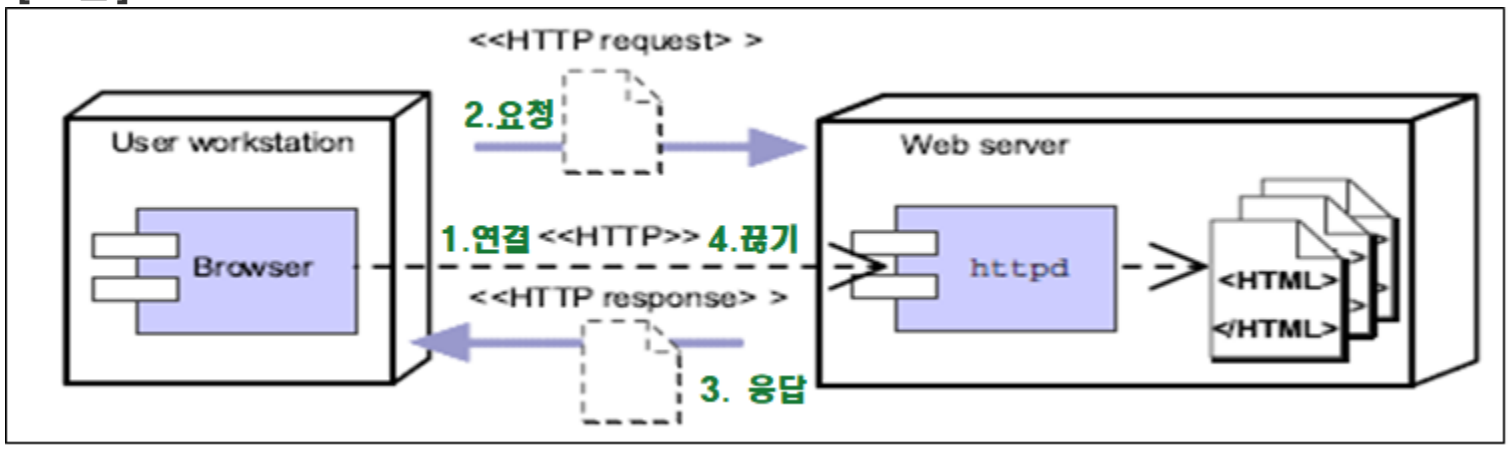
IP 주소, Port 번호, URL

- URL (Uniform Resource Locator)
  - 네트워크상의 자원이 어디에 있는지 표시하기 위한 규약
  - 도메인/IP 주소를 포함해서 프로토콜, 자원의 위치까지 포함한 전체 주소
  - 구조
    - scheme://user:password@host:port/path?query#fragment
      - **schema**: 자원 접근 방식. 프로토콜 지정
      - **user:password@**: 인증을 위한 사용자 계정과 비밀번호. 생략가능
      - **host**: 접속할 서버 IP 또는 Domain 주소
      - **port**: 서버의 포트번호
      - **path**: 서버내 자원의 위치
      - **query**: 자원에 전달할 파라미터. QueryString 이라고 한다.
      - **fragment**: 문서 내의 특정 위치
    - https://www.google.com/search?q=파이썬

HTTP 프로토콜 – Web 프로토콜

- 개요
  - **H**yper**T**ext **T**ransfer **P**rotocol
  - HTML 문서 제공을 목적으로 만들어진 TCP/IP 기반 프로토콜
  - HTML 이외에도 다양한 형태의 자원(동영상,음악, 일반파일 등)들을 제공할 수 있다.
  - 자원들을 구분하기 위해 MIME 타입을 사용한다.
  - Stateless (무상태) 특징을 가진다.
    - **요청시 연결하고 응답이 끝나면 연결을 종료한다.** → 연결이 유지 되지 않는다.
    - 서버는 클라이언트의 상태(정보)를 유지 하지 않는다.

[흐름]



1. 연결 (Client -> Server)
2. 요청 (Client -> Server)
3. 응답 (Server -> Client)
4. 연결 끊기

HTTP 프로토콜 – 구성요소

- HTTP Client (Program)
  - 웹 브라우저
    - Internet Explorer, Chrome, FireFox 등
- HTTP Server (Program)
  - 웹 서버
    - Apache httpd, NGINX, IIS 등
- HTML (**H**yper **T**ext **M**arkup **L**anguage)
  - HTML은 웹 문서의 구조와 의미를 기술하는 Markup 언어이다.
  - HTTP 프로토콜을 통해 **서버와 클라이언트 간에 교환되는 문서**(데이터)로 웹 브라우저가 해석하고 랜더링한다.

HTTP 프로토콜 – HTTP 요청방식(HTTP Method)

- **HTTP 요청 방식(HTTP Method)**
  - 클라이언트가 서버에 요청하는 목적에 따라 다음과 같은 8가지 방식을 제공한다.
  - GET, POST, PUT, DELETE, HEADER, OPTIONS, TRACE, CONECT
    - Web은 GET과 POST 방식 지원

- **GET방식**
  - HTTP 요청의 기본방식
  - 목적 : 서버가 가진 데이터 요청
  - 서버로 전달하는 값
    - 문자열만 가능하며 URL뒤에 붙여서 보낸다.(QueryString 이라고 한다.)
      - Query String(쿼리 스트링) : URL**?**name=value&name=value

- **POST방식**
  - 목적: 서버에 데이터 전송
  - <form method="post"> 로 설정
  - 문자열 뿐 아니라 파일도 전송할 수 있다.
  - HTTP 요청 정보의 Body를 통해 요청파라미터 전달

요청파라미터(Request Parameter) 란

- 사용자가 일처리를 위해 서버로 전송하는 값으로 name=value 쌍 형식으로 전송된다.

- 값이 여러 개일 경우 & 로 연결된다.

- ex) id=abc&password=1111&name=김철수

HTTP 프로토콜 – HTTP 요청방식(HTTP Method)

- **PUT**
  - 기존 Resource를 변경
- **DELETE**
  - 기존 Resource를 삭제
- 그 외
  - HEAD, OPTIONS, TRACE, CONNECT

HTTP 프로토콜 - 요청정보 및 요청방식

- HTTP 요청정보
  - Web Browser가 Web Server로 요청할 때 만드는 정보
  - HTTP 프로토콜에 정해진 형식대로 요청한다.

- HTTP 요청정보 구성
  - 요청라인
    - "요청방식 요청경로 HTTP 버전" 으로 구성된 첫번째 라인.
    - GET 방식의 경우 요청파라미터가 요청 경로 뒤에 QueryString으로 전송된다.
  - 헤더
    - 요청 클라이언트의 정보를 name-value 쌍 형태를 가진다.
  - 요청 Body
    - POST 방식의 경우 요청파라미터가 저장된다.
    - GET방식은 요청파라미터가 요청라인의 URL 추가되어(QueryString) 전송되므로 빈 요청 body가 전달됨.

HTTP 프로토콜 - HTTP 요청 정보

- GET 방식 요청정보

요청 라인	GET /search?keyword=python HTTP/1.1
요청 헤더(Header)	Connection: Keep-Alive User-Agent: Modzilla/4.76 [en] (x11; U, SunOS 5.8 sun4u) Host: localhost:8088 Accept:image/gif,image/x-bitmap, image/jpeg, */* Accept-Charset: utf-8
요청 Body	

- POST 방식 요청정보

요청 라인	POST /member/loginHTTP/1.1
요청 헤더(Header)	Connection: Keep-Alive User-Agent: Modzilla/4.76 [en] (x11; U, SunOS 5.8 sun4u) Host: localhost:8088 Accept:image/gif,image/x-bitmap, image/jpeg, */* Accept-Charset: utf-8
요청 Body	custom_id=id-1010&password=pwd1234

HTTP 프로토콜 – HTTP 응답정보

- HTTP 응답 정보
  - Web Server가 Web Browser(Client)에게 응답할 때 만드는 정보
- HTTP 응답 정보 구성
  - 응답라인 : 응답 처리 결과를 코드(상태코드) 로 전송 (https://ko.wikipedia.org/wiki/HTTP_%EC%83%81%ED%83%9C_%EC%BD%94%EB%93%9C)
  - 응답 Header : 응답에 관련된 다양한 정보를 담는다.
    - 응답 내용의 타입, 응답내용의 크기, Cookie값 등
  - 응답 Body : 응답 내용(처리결과)을 담는다.

응답 라인	HTTP/1.1 200 OK
응답 헤더(Header)	Content-Type: text/html;charset=utf-8 Date: Tue, 10 Apr 2000 01:01:01 GMT Server: Apache Tomcat/8 (HTTP/1.1 Connector) Connection: close
응답 Body	<html> <head> <title>응답메세지</title> </head> <body> 안녕하세요 </body> </html>

HTML
(HyperText Markup Language)

참조

- http://www.w3schools.com
  - HTML, CSS, JavaScript 등 튜토리얼 사이트
- www.w3.org
  - World Wide Web Consortium

HTML, CSS, JavaScript

- HTML 문서란
  - **HTML 문서(HTML Document)**는 웹에서 정보를 표현하기 위해 HTML 언어로 작성된 **텍스트 기반 파일**이며, 브라우저가 해석(파싱)하여 화면에 렌더링할 수 있는 구조를 가지고 있다.
  - 초창기 HTML 문서는 모든 요소들을 HTML 문법을 이용해 작성했으나 **현재는 HTML, CSS, Javascript 세 가지 언어를 이용해 작성**한다.
- HTML
  - **문서의 내용과 구조**를 정의
  - Markup 언어
- CSS
  - 문서의 스타일(디자인)을 정의
- JavaScript
  - 문서내에서 동작을 정의

Hyper Text Markup Language

- Markup Language란
  - 문서내 내용(content)에 **의미를 표시하기 위한 언어**
    - 컴퓨터에게 텍스트의 의미(내용)을 알려주기 위해 그 의미를 표시한다.
    - HTML, XML 이 대표적인 markup 언어이다.
  - 시맨틱 마크업(semantic markup)
    - 의미 기반 마크업으로 정보의 의미를 명확히 표시 한다는 의미임.
    - 특히 웹문서에서 시각적 효과와 같은 의미와 관련 없는 표시는 하지 않도록 한다.
- HTML
  - 웹 문서를 만들기 위한 markup 언어로 1991년 팀 버너스리(웹 창시자)가 발표
  - 인터넷 상에서 문서공유를 위해 만들어짐.
  - Hyper Text
    - 문서간의 이동을 쉽게 처리하기 위한 방법

HTML 태그 개요

- HTML문서는 Text 기반으로 작성하며 '파일명.html' 로 저장한다.
- HTML언어는 대소문자를 구분하지 않는다.
  - 단 통일해서 쓰는 것이 좋다.
- **Tag 개요**
  - 태그(Tag)는 정보(내용)의 의미나 기능을 표시하는 방법으로 다음과 같은 형태를 가진다.

```
<tag>내용</tag>
<tag>
<tag 속성="값">내용</tag>
```

```
<h1>제목: HTML</h1>
<br>
<img src='tree.png'>
```

  - 태그는 여는 태그(open tag)와 닫는 태그(close tag) 의 구조를 가진다.
  - 열고 닫는 태그는 그 안의 내용(content)에 대한 범위를 지정하는 역할을 한다.
  - 태그는 기능을 확장하기 위해 속성을 가질 수 있다.
    - 속성은 **속성명 = 값** 형태를 가지며 값은 " "로 감싸준다.
  - 내용이 없는 태그를 빈 태그(empty tag)라고 하며 여는 태그만 작성한다.

HTML 태그 개요

- 부모 / 자식 태그
  - 태그의 내용으로 태그가 들어오면 계층관계가 만들어 진다.

```
<section>
  <ul>
     <li>사과</li>
     <li>귤</li>
     <li>배</li>
  </ul>
</section>
```

    - 위의 예에서 <ul> 을 **부모 태그**라고 하고 를 **자식 태그** 라고 한다.
  - <section>과 <li> 의 경우는 **조상-자손 관계**(**Ancestor–Descendant**)라고 한다.
  - 전체 문서는 하나의 태그로 부터 파생되며 그 태그를 **Root Tag/Root element라고** 한다**.**
  - HTML의 태그 간의 계층관계는 웹 브라우저가 Tree 구조로 만들어 관리한다.
    - 이 구조를 DOM Tree 라고 한다.

HTML 문서의 기본 구조

```
<!doctype html>
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

- doctype 선언 - Web Browser에 문서의 HTML Version을 알려주는 역할
  - HTML 4
    - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
  - HTML 5
    - **<!DOCTYPE html>**

HTML 문서의 구조

- <html>
  - HTML 문서 전체 범위를 지정한다. 문서의 모든 내용은 html 태그의 내용으로 들어와야 한다.
  - **Root 태그**
- <head>
  - 문서에 대한 정보를 담는 태그
  - 하위 태그들
    - <title> : 문서제목
    - <meta> : 문서에 대한 meta data
      - <meta name='정보종류' content='정보값'>
        - 정보종류: author, description, keywords 등
      - <meta charset='인코딩타입'>
        - 인코딩 타입: UTF-8, EUC-KR

HTML 문서의 구조

- <head> 이어서
  - <link>
    - 외부문서를 연결할 때 사용. 주로 CSS 파일 불러올때 사용한다.
    - <link rel='가져올문서 종류' href='가져올 문서의 url'>
  - <style>
    - CSS 를 직접 HTML 문서내에 작성할 때 사용.
    - 위치는 어디든 올 수 있지만 주로 <head> 의 자식요소로 작성한다.
  - <script>
    - JavaScript 코드를 작성하는 태그
    - 위치는 어디든 올 수 있지만 주로 <head> 의 자식요소로 작성한다.
- <body>
  - 문서의 내용을 담는태그. 웹브라우저에서 보여지는 부분을 작성한다.

HTML 주요 기본 태그들

- <!-- 주석 -->
- <hX> : 제목 태그
  - X는 1 ~ 6
-
 : 다음줄
  - HTML 문서에서는 엔터와 space는 하나의 공백으로 처리
  -
은 내용이 없으므로 단독으로 쓰거나
 로 쓴다.
  - 참고 : 공백은   사용
- <p> : 문단(paragraph)
  - 내용 아래위 한 줄의 공백이 들어간다.
- <b>, : 볼드체
- <i>, : 이태릭체

Link, Image

- Hyperlink - 다른 문서로 이동하기 위한 방법
  - 태그구문
    - 링크구문||이미지
    - 예다음
- Image
  - 태그 구문
    - <img src="url" alt="설명">
- URL
  - 전체 경로 : http://domain:port/문서경로/문서명
  - 로컬 경로(같은 도메인내의 자원으로 이동 시)
    - 절대 경로 - Domain root 경로에서 부터 찾는다.
      - / 로 시작
    - 상태 경로 – 현재 보는 페이지가 있던 경로를 기준으로 찾는다.
      - . : 현재 경로
      - .. : 상위 경로

Table - 표

- <table> 표를 정의하는 태그
  - 하위 태그 - <thead><tbody><tfoot><tr>
- <thead> 표의 상단 행들을 묶는 태그
  - 표의 상단에 나올 행(tr)들을 정의
- <tbody> 표의 내용 행들을 묶는 태그
  - 표의 중단에 나올 행(tr) 들을 정의
- <tfoot> 표의 하단 행들을 묶는 태그
  - 표의 하단에 나올 행(tr) 들을 정의
- <tr> 하나의 행을 정의
  - 하위 태그 <td> 또는 <th>
- <td>, <th> 행 내의 cell을 정의 하는 태그로 내용(값)을 가진다.
  - 속성 : rowspan="정수" (두 행의 cell을 합친다.), colspan="정수" (두 열의 cell을 합친다.)

List

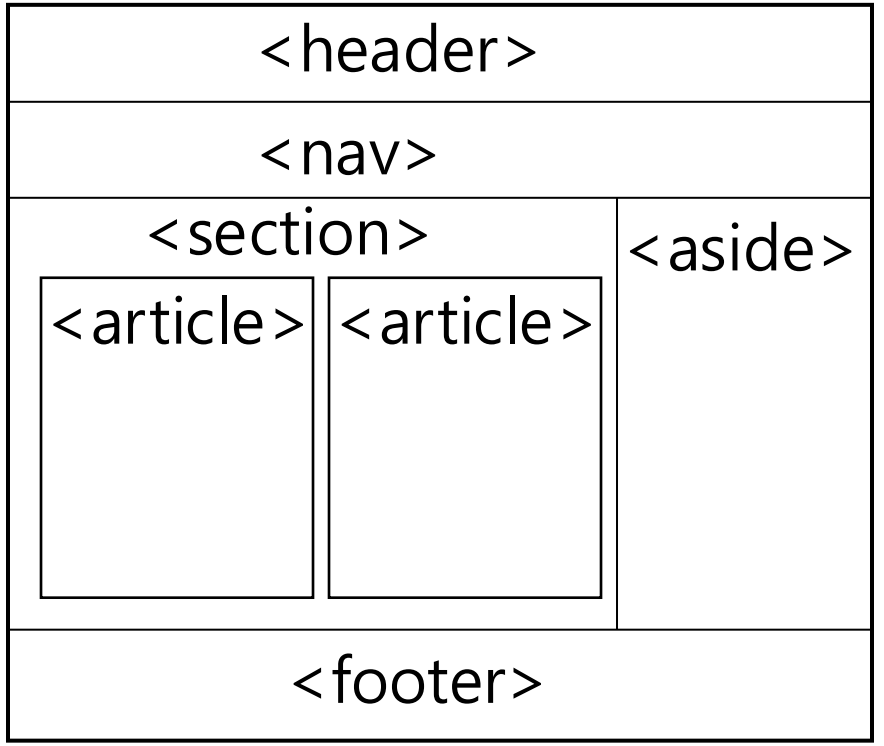
- - unordered list : 순서가 없는 목록
  - 속성
    - type – 아이템 앞에 붙을 기호 타입 지정
      - 값 : cicle(기본), square, none
- - ordered list : 아이템들에 순번을 붙이는 목록
  - 속성
    - type – 아이템 앞에 붙일 순번의 타입
      - 값 : 1(기본-숫자), A-알파벳대문자, a-알파벳소문자, I-로만숫자(대문자), i-로만숫자(소문자)
- - 리스트내에 들어갈 아이템을 정의하는 태그
  - 의 하위태그로 사용
  - 내용(content)
    - 아이템에 보여질 항목(문자열)
    - || - 리스트 내에 하위 리스트를 넣는 경우

Block

- block-level 태그와 inline 태그
  - block-level 태그 : 새로운 라인에서 내용을 보여주는 태그
    - ex) <h1>, <p>, <table>, <ul> 등
  - inline 태그 : 라인을 변경하지 않고 내용을 보여주는 태그
    - ex) <b>, <i>, <a>, <img> 등
- Block 태그
  - 구역 지정이 목적인 태그
  - CSS나 JavaScript와 연되어 많이 쓰인다.
  - **HTML 4**
    - : block-level 태그
    - : inline 태그
  - **HTML 5**
    - 다양한 시맨틱(semantics) 태그들 추가

Block – HTML 5 의 시맨틱(Semantics) 태그

- 시맨틱 태그 – 태그가 내용의 의미를 표현하는 태그.
  - 검색 엔진 등에서 문서의 내용 검색이 용의해 진다.
- HTML 5에서 block 내의 내용을 표현할 수 있는 block 태그들이 추가됨
- 주요 태그
  - <header> : 문서의 제목이나 소개의 내용등을 담는다.
    - 문서의 헤더로 사용 : 사이트 소개, 로고, 메뉴 등을 담는다.
    - section이나 article의 헤더로 사용
  - <nav> : 네비게이터 부분. 사이트 메뉴등을 담는다.
  - <section> : 문서의 한 테마에 대한 내용을 묶어준다.
  - <article> : 독립적인 하나의 글을 담는다.



Block – HTML 5 의 시맨틱(Semantics) 태그

- <aside> : 문서의 내용의 주제와 다른 내용을 보여줄 경우.(ex: 사이드바)
- <footer> : 전체 문서나 각 섹션의 하단부에 나올 내용을 담는다.
  - 전체 문서의 footer : 회사전화번호, 주소 등
  - section이나 article 의 footer : 작성자, 연락처 등.
- <figure>
  - 이미지+이미지 하단 설명을 담는 태그
  - <figcaption> : 이미지 설명

```
<figure>
    <img src="run_image.jpg" alt="실행 예시 이미지">
    <figcaption>그림 4-1 실행 시 화면</figcaption>
</figure>
```

Form

- 요청 파라미터
  - 사용자로부터 입력 받아 서버로 전송하는 값
  - name=value 쌍 형태를 가진다.
    - name=value, name=value&name=value
- <form>
  - 사용자부터 값을 입력 받기 위한 입력 태그들을 묶어 주는 역할
  - 속성
    - method : 서버로 입력 받은 값을 전송할 때 사용할 HTTP 방식 지정 – get, post
    - action : 입력 받은 값을 전송할 서버의 url
    - enctype : 서버로 전송할 값들의 encoding 타입
      - 기본 : application/x-www-form-urlencoded
      - 파일 업로드 : multipart/form-data

Form – 입력 태그

- 입력 태그들
  - <input> 다양한 형태의 입력 폼 제공
  - <select> Drop-Down 목록에서 선택하도록 처리
  - <textarea> 여러 줄 입력 폼 제공
- 공통 속성
  - name : 전송할 요청 파라미터 값에 붙일 이름. 서버에서는 이 이름을 통해 값을 읽는다.
  - value : 입력 양식에 설정할 기본 값

Form – input 입력태그

- <input>
  - type 속성의 값으로 다양한 입력 형태를 만든다.
- type
  - text : 한 줄 text 입력 폼
  - password : 패스워드를 입력 받기 위한 한 줄 입력 폼
  - radio : 선택 입력 폼으로 여러 개 중 하나 선택 시 사용
    - 사용자가 값을 입력 받는 것이 아니라 지정된 값들 중 하나를 선택하도록 할 때 사용
    - 같은 이름의 radio들이 하나의 group으로 묶인다.
  - checkbox : 선택 입력 폼으로 yes/no를 선택 하거나 여러 개 중 0개 이상을 선택 시 사용
    - 사용자가 값을 입력 받는 것이 아니라 지정된 값들 중 0개 이상을 선택하도록 할 때 사용
  - hidden : 사용자에게 보여주지 않고 전송 시 전송될 값을 코드상 설정할 때 사용
  - file : 사용자 컴퓨터에 있는 파일을 선택해 서버로 전송할 때 사용

Form – input 입력태그

- type (이어서)
  - submit : 전송버튼
  - reset : 초기화 버튼
  - button : 일반 버튼
  - image : 전송 버튼. 이미지를 버튼으로 사용한다.
- HTML 5 추가 type (웹브라우저에 따라 지원 안 할 수 있음)
  - number : 숫자만 입력 받는 한 줄 입력 태그
    - 속성 : min – 선택할 수 있는 최소 숫자 지정, max – 선택 할 수 있는 최대 숫자 지정
  - email : 이메일 형식의 문자열을 입력 받는 한 줄 입력 태그
    - 값@값 형식이어야 한다.
  - range : 숫자의 범위를 지정해 슬라이드 바 형태로 숫자를 선택하는 태그
    - min : 최소 숫자, max : 최대 숫자
  - date, time : 날짜, 시간 입력

Form – input 입력태그

- 주요 속성
  - readonly : 사용자가 바꿀 수 없도록 처리.
  - size : 입력 폼의 너비 지정(보여지는 글자 수로 지정)
  - maxlength : 입력할 수 있는 최대 글자 수
  - checked : radio와 checkbox의 속성으로 체크된 상태로 나오도록 처리.
  - selected : select태그의 option속성으로 지정. 목록에 처음 나올 항목 지정
- HTML 5 추가 속성
  - autofocus : 페이지 로딩 시 포커스를 얻을 입력 폼 지정
  - placeholder : 입력 폼에 넣을 값에 대한 힌트를 설정.
  - required : 필수 입력 항목 지정(사파리 지원 안 함)

Form – select, textarea

- <select>
  - 사용자가 지정된 값(item) 중 하나를 선택하도록 하며 drop-down 형태를 가짐.
  - 속성 : name
  - 하위 태그
    - <option>
      - 사용자가 선택할 아이템들을 설정
      - 속성 : value – 전송할 값 (생략 시 값(text) 가 전송됨)
- <textarea>
  - 여러 줄의 text를 입력 받기 위한 입력 폼.
  - 속성
    - cols : 열 수 지정
    - rows : 행 수 지정
    - 기본 값은 value 가 이나라 태그 내에 넣는다. ex)<textarea>기본값</textarea>

CSS
(Cascade Style Sheet)

개요

- Cascading Style Sheets 의 약자로 HTML 문서를 꾸밀 때 사용하는 언어
- 문서의 구조와 문서의 스타일을 분리한다.
  - HTML로 문서의 구조를 만들고 CSS 로 문서를 꾸민다.

HTML에 Style 적용

- CSS를 HTML문서에 적용하는 3가지 방법
  - 태그의 속성을 이용해 적용 (Inline Style)
  - HTML 문서 내에 style 태그를 만들어 분리해 적용 (Internal Style Sheet)
  - CSS 파일을 따로 만들어 HTML 문서에 적용(External Style Sheet)
- 적용 순서
  - 동일한 html 요소에 적용 된 경우
  - Inline style - Internal – External 순서로 적용된다.

HTML에 Style 적용

- External Style Sheet
  - 외부에 파일로 정의한 뒤 참조하도록 정의

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

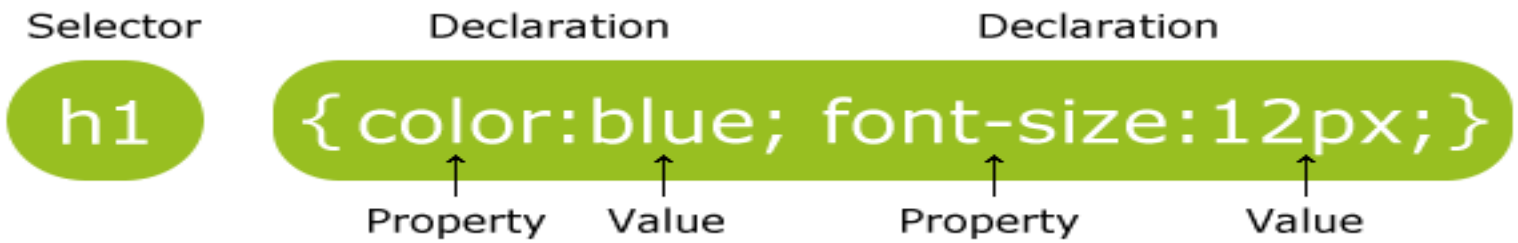
- Internal Style Sheet
  - HTML내에 구문과 분리하여 구현

```
<head>
  < style type="text/css">
      p {margin-left:20px;}
      body {background-image:url("images/back40.gif");}
  < /style>
< /head>
```

- Inline Style
  - HTML 태그에 직접 정의

```
<p style="color:sienna;margin-left:20px">This is a paragraph.</p>
```

기본구문



- Selector
  - Style을 적용할 대상을 선택하는 문법
  - 태그이름, id 속성, class 속성 등 이용
- Declaration
  - Style 정의
    - **Property** : 적용할 Style 종류
    - **Value** : Style에 대한 값
- 주석 (Comment)

```
/*
  주석 입니다....
*/
```

ID와 Class 선택자(selector)

- 태그 식별자 속성
  - HTML 문서의 태그를 CSS나 JavaScript 에서 지정할 때 사용하는 식별자 역할
- ID
  - 요소(element) 중 하나를 지정할 때 사용
  - 구문 : **#id**

```
<input type="text" name="name" id="nameTF"/>

#nameTF { 스타일 지정}
```

- Class
  - 요소 그룹(element group) 을 지정할 때 사용
  - 구문 : **.className**

```
<input type="text" name="name" class="inputTF"/>
<input type="text" name="age" class="inputTF"/>

.inputTF { 스타일 지정}
```

주요 CSS Selector (https://www.w3.org/TR/selectors/)

Selector	설명	Ex)
*	모든 태그	
E	태그 명 E와 일치하는 모든 element node	`div` `p`

E.C	E 요소들 중 class속성값이 C인 모든 element node	`div.name`
		`.name`
		`.name1.name2`

E#I	E 요소들 중 id속성값이 I인 모든 element node	`div#id` `#id`
E F	E 로 선택된 요소의 자손인 모든 F element node	`div p`
E>F	E 로 선택된 요소의 자식인 모든 element node	`div > p`
E+F	E 로 선택된 요소의 **바로 다음** 형제 element node	`div+p`
E~F	E 로 선택된 요소 다음에 나오는 **모든 형제** element node	`div~p`
E[A]	태그명이 E인 것중 속성으로 A를 가지는 모든 element node	`form[target]` `[target]`
E[A=V]	태그명이 E인 것중 속성 A의 값이 V인 모든 element node	`form[target=blank]`
E[A^=V]	태그명이 E인 것중 속성 A의 값이 V로 시작하는 모든 element node	`a[href^="https"]`
E[A$=V]	태그명이 E인 것중 속성 A의 값이 V로 끝나는 모든 element node	`a[href$="com"]`
E[A*=V]	태그명이 E인 것중 속성 A의 값에 V가 들어가는 모든 element node	`a[href*="google"]`

주요 CSS Selector

Selector	설명	Ex)
a:link	방문 전 링크	
a:visited	방문 한 링크	
a:active	마우스 클릭했을 때 링크	
a:hover	마우스 오버(커서를 올렸을 때) 했을 때 링크	
E:first-child	E 들 중 첫번째 자식 요소(태그)	`p:first-child`
E:last-child	E 들 중 마지막 자식 요소(태그)	`p:last-child`
E:nth-child(정수)	E 들 중 정수번째 자식 요소(태그) – **1부터 시작**	`div:nth-child(3)`
E:nth-child(정수n)	E 들 중 정수배 번째 자식 요소(태그)	`div:nth-child(3n)` `div:nth-child(3n+2)`
E:nth-last-child(정수)	E 들 중 뒤에서 정수번째 자식요소	`p:nth-last-child(3)`
E:nth-last-child(정수n)	E 들 중 뒤에서 부터 정수배 번째 자식 요소	`p:nth-last-child(2n)` `p:nth-last-child(2n+1)`
E:nth-type-of(정수)	E요소와 **같은 태그들** 중 자식태그로 n번째 있는 것	`div:nth-type-of(2)`
E:nth-type-of(정수n)	E요소와 **같은 태그들** 중 자식태그로 정수배 번째 있는 것	`div:nth-type-of(2n)` `div:nth-type-of(2n +3)`

주요 CSS Framework

- 웹 페이지 스타일링을 빠르고 일관되게 개발할 수 있도록 미리 만들어진 CSS 코드 라이브러리이다.

- 대표적인 CSS Framework
  - Bootstrap
    - 다양한 UI 컴포넌트를 제공하는 가장 대중적인 CSS Framework
  - Tailwind CSS
    - 유틸리티 클래스들을 조합해서 자유로운 디자인을 할 수 있어 자유도가 높은 최근 가장 인기 있는 CSS 프레임워크
  - Materialize
    - 구글의 머터리얼 디자인을 구현하는 CSS Framework.