3/18 Web 과목평가 준비

Web 1일차 교재

World Wide Web
인터넷으로 연결된 컴퓨터들이 정보를 공유하는 거대한 정보 공간

Web
Web site, Web application 등을 통해 사용자들이 정보를 검색하고 상호 작용하는 기술

Web site
인터넷에서 여러 개의 Web page가 모인 것으로, 사용자들에게 정보나 서비스를 제공하는 공간

Web page
HTML, CSS 등의 웹 기술을 이용하여 만들어진, Web site를 구성하는 하나의 요소
(HTML, CSS, Javascript로 구성된다.)

HTML (HyperText Markup Language)
Web page의 의미와 **구조**를 정의하는 언어

Hypertext
Web page를 다른 페이지로 연결하는 링크
참조를 통해 사용자가 한 문서에서 다른 문서로 즉시 접근할 수 있는 텍스트

Markup Language
Tag 등을 이용하여 문서 & 데이터의 구조를 명시하는 언어
(HTML, Markdown)

HTML 구조
<!DOCTYPE html>
해당 문서가 html로 문서라는 것을 나타냄

<html></html>
전체 페이지의 콘텐츠를 포함

<title></title>
브라우저 탭 및 즐겨찾기 시 표시되는 제목

<head></head>
HTML 문서에 관련된 설명, 설정 (사용자에게 보이지 않음)

<body></body>
페이지에 표시되는 모든 콘텐츠

HTML Element (요소)
요소 = <여는 태그> 내용 </닫는 태그>

요소는 '여는 태그'와 '닫는 태그' 그리고 그 안에 '내용'으로 구성된다.
닫는 태그는 태그 이름 앞에 /를 포함한다.
닫는 태그가 없는 태그도 존재한다.

HTML Attributes (속성)
속성 : 여는 태그 안에 있는 class=""

속성의 규칙

1. 속성은 요소 이름과 속성 사이에 공백이 있어야 함
2. 하나 이상의 속성들이 있는 경우 속성 사이에 공백으로 구분
3. 속성값은 ""로 감싸야 함

속성의 목적

1. 나타내고 싶지 않지만 **추가적인 기능&내용**을 담고 싶을 때 사용
2. CSS에서 해당 **요소를 선택**하기 위한 값으로 활용

HTML Text structure
HTML의 주요 목적 중 하나는 **텍스트 구조와 의미**를 제공하는 것

<h1></h1>
현재 문서의 **최상위 제목**이라는 의미를 부여한다.

종류
Heading & Paragraphs
h1~h6, p

Lists
ol, ul, li

Emphasis & Importance
em, strong

```
<body>
    <h1>Main Heading</h1>
    <h2>Sub Heading</h2>
    <p>This is my page</p>
    <p>This is <em>emphasis</em></p>
    <p>Hi <strong>my name</strong> is Air</p>
    <ol>
        <li>파이썬</li>
        <li>알고리즘</li>
        <li>웹</li>
    </ol>
</body>
</html>
```

CSS (Cascading Style Sheet)
Web page의 **디자인**과 **레이아웃**을 구성하는 언어

```
h1 {
    color: red;
    font-size: 30px
}
```

h1 : 선택자 (Selector)
속성 (Prorerty) : color, font-size
값 (Value) : red, 30px
선언 (Declaration) : color: red;, font-size: 30px

CSS 적용 방법

1. **인라인(Inline) 스타일**
   HTML 요소 안에 style 속성 값으로 작성
   <h1 style="color: blue;">Hello World!</h1>

2. **내부(Internal) 스타일 시트**
   head 태그 안 style 태그에 작성
   <head> <style> h1 { color: blue; } </style> </head>

3. **외부(External) 스타일 시트**
   별도의 CSS 파일 생성 후 HTML link 태그를 사용해 불러오기
   <head><link rel="stylesheet" href="style.css"></head>

CSS Selectors
HTML 요소를 선택하여 스타일을 적용할 수 있도록 하는 선택자

종류
I. 기본 선택자
전체 선택자 : HTML 모든 요소를 선택
* { }

요소 선택자 : 지정한 모든 태그를 선택
h2 { }, h3, h4 { }

클래스 선택자 : 주어진 클래스 속성을 가진 모든 요소를 선택
.class { }

아이디 선택자 : 주어진 아이디 속성을 가진 요소를 선택
(문서에는 주어진 아이디를 가진 요소가 하나만 있어야 함)
#id { }

속성 선택자

II. 결합자 (Combinators)
자손 결합자
첫번째 요소의 자손 요소들 선택
p span은 <p> 안에 있는 모든 을 선택 (하위 레벨 상관없이)

자식 결합자
첫 번째 요소의 직계 자식만 선택
ul > li은 <ul> 안에 있는 모든 를 선택 (한단계 아래 자식들만)

명시도 (Specificity)
결과적으로 요소에 적용할 CSS 선언을 결정하기 위한 알고리즘

CSS Selector에 가중치를 계산하여 어떤 스타일을 적용할지 결정

동일한 요소를 가르키는 2개 이상의 CSS 규칙이 있는 경우 가장 높은 명시도를 가진 Selector가 승리하여 스타일이 적용

즉, 스타일 적용의 우선순위를 의미함

!important : 다른 우선순위 규칙보다 우선하여 적용하는 키워드
(Cascade의 구조를 무시하고 강제로 스타일 적용, 권장X)

명시도(우선순위) 순서

1. !important
2. Inline 스타일
3. id 선택자
4. class 선택자
5. 요소 선택자
6. 소스 코드 선언 순서

Casade
한 요소에 동일한 가중치를 가진 선택자가 적용될 때 CSS에서 마지막에 나오는 선언이 사용됨 (계단식)

CSS 상속
기본적으로 CSS는 상속을 통해 부모 요소의 속성을 자식에게 상속해 재사용성을 높임

CSS 속성 2가지 분류
상속되는 속성
Text 관련 요소(font, color, text-align), opacity, visibility 등

상속되지 않는 속성
Box model 관련 요소 (width, height, border, box-sizing)
position 관련 요소 (position, top, bottom, right, left, z-index 등)

HTML 관련 사항
요소(태그) 이름은 "소문자" 사용을 권장
속성의 따옴표는 "큰 따옴표" 사용을 권장
(둘 모두 대문자, 작음 따옴표를 사용해도 문제는 없음)
HTML은 프로그래밍 언어와 달리 에러를 반환하지 않기 때문에 작성 시 주의

CSS 인라인(Inline) 스타일 사용 지양
CSS와 HTML 구조 정보가 혼합되어 작성되기 때문에 코드를 이해하기 어렵게 만듦

속성은 class만 사용할 것
id, 요소 선택자들과 함께 사용할 경우 우선순위 규칙에 따라 규칙이 적용되기에 유지보수가 어려워짐
문서에서 단 한번 유일하게 적용될 스타일에만 "id 선택자" 사용을 고려

Web 2일차 교재

CSS Box Model
모든 HTML 요소를 사각형 박스로 표현하는 개념
내용(content), 안쪽 여백(padding), 테두리(border), 외부 간격(margin)으로 구성된다.
원은 네모 박스를 깎은 것이다. (기본은 네모 박스로 시작한다는 뜻)

Margin
가장 바깥쪽 영역, 이 박스와 다른 요소 사이의 공백

Border
콘텐츠와 패딩을 감싸는 테두리 영역

Padding
콘텐츠 주위에 위치하는 공백 영역

Content
콘텐츠가 표시되는 영역

Box 구성의 방향별 명칭
Content
너비 : width, 높이 : height

Margin, Border, Padding
상하좌우 : top, bottom, left, right

width & height 속성
요소의 너비와 높이를 지정
이때 지정되는 요소의 너비와 높이는 "콘텐츠 영역"을 대상으로 함
-> 실제 박스가 차지하는 너비와 높이는 더 길고 높다는 뜻
(Padding, Border, Margin의 크기만큼)

box-sizing 속성
box-sizing: content-box;
-> content 영역의 크기를 결정

box-sizing: border-box;
-> border를 포함한 영역의 크기를 결정

박스 타입
Block & Inlin 타입

Noraml flow
CSS를 적용하지 않았을 경우 웹페이지 요소가 기본적으로 배치되는 방향
Block : 위에서 아래로
Inline : 왼쪽에서 오른쪽으로

block 타입 특징
항상 새로운 행으로 나뉨
width와 height 속성을 사용하여 너비와 높이를 지정
기본적으로 width 속성을 지정하지 않으면 박스는 inline 방향으로 사용 가능한 공간을 모두 차지
대표적인 block 타입 태그 : h1~h6, p, div

inline 타입 특징
새로운 행으로 나뉘지 않음
width와 height 속성 사용 불가능
padding, margin, borders 적용 가능
-수직방향 : 다른 요소를 밀어낼 수는 없음
-수평방향 : 다른 요소를 밀어낼 수 있음
대표적인 inline 타입 태그 : a, img, span

inline-block
inline과 block 요소 사이의 중간 지점을 제공하는 display 값
block 요소의 특징을 가짐
-width & height 속성 사용 가능
-padding, margin, borde 적용 및 다른 요소를 밀어냄
요소가 줄 바꿈 되지않고 너비와 높이를 적용하고 싶은 경우 사용

none
요소를 화면에 표시하지 않고, 공간조차 부여하지 않음

shorthand 속성
여러 속성을 하나의 선언으로 축약하여 사용하는 기법

shortand - border
border의 width, style, color를 한 번에 설정하는 속성
border: 2px solid black;

shortand - margin & padding
4방향 속성을 각각 지정하지 않고 한 번에 지정할 수 있는 속성
4개 : 상우하좌
3개 : 상 / 좌우 / 하
2개 : 상하 / 좌우
1개 : 공통

Margin collapsing (마진 상쇄)
두 block 타입 요소의 margin top와 margin bottom이 만나면 더 큰 margin으로 결합되는 현상
각 요소에 대한 상/하 margin을 각각 설정하지 않고 한 요소에 대해서만 설정하기 위함

CSS Layout
각 요소의 **위치**와 **크기**를 조정하여 Web page의 디자인을 결정하는 것
(Display, Position, Float, Flexbox 등)

CSS Position
요소를 Normal Flow에서 제거하여 다른 위치로 배치하는 것
(다른 요소 위에 올리기, 화면 특정 위치에 고정시키기 등)

Position 유형
static
기본값
요소를 Normal Flow에 따라 배치

relative
요소를 Normal Flow에 따라 배치
자기자신을 기준으로 이동
요소가 차지하는 공간은 static일 때와 같음

absolute
요소를 Normal Flow에서 제거
가장 가까운 relative 부모 요소를 기준으로 이동
(relative 부모 요소가 없으면 body를 기준으로 이동)
문서에서 요소가 차지하는 공간이 없어짐

fixed
요소를 Normal Flow에서 제거
현재 화면영역(viewport)을 기준으로 이동
문서에서 요소가 차지하는 공간이 없어짐

sticky
요소를 Normal Flow에 따라 배치
요소가 일반적인 문서 흐름에 따라 배치되다가 스크롤이 특정 임계점에 도달하면 그 위치에 고정됨 (fixed)
만약 다음 sticky 요소가 등장하면 다음 sticky 요소가 이전 sticky 요소의 자리를 대체
-> 이전과 다음 sticky 요소가 고정되어야 할 위치가 겹치기 때문에

z-index
요소가 겹쳤을 때 어떤 요소 순으로 위에 나타낼지 결정

z-index 특징
정수 값을 사용해 Z축 순서를 지정
더 큰 값을 가진 요소가 작은 값의 요소를 덮음

Position의 역할
전체 페이지에 대한 레이아웃을 구성하는 것이 아닌 **페이지 특정 항목의 위치를 조정**하는 것

CSS Flexbox
요소를 행과 열 형태로 배치하는 1차원 레이아웃 방식 -> 공간 배열 & 정렬

main axis
flex item들이 배치되는 기본축 (가로)
main start에서 시작해서 main end 방향으로 배치 (기본값)
(main start, main end는 각각 Flex Container의 왼쪽 끝, 오른쪽 끝)

cross axis
main axis에 수직인 축 (세로)
cross start에서 시작해 cross end 방향으로 배치 (기본값)
cross start, cross end는 각각 Flex Container의 제일 위, 아래

Flex Container
display: flex; 혹은 display: inline-flex; 가 설정된 부모요소
컨테이너의 1차 자식 요소들이 flex item이 됨
flexbox 속성값들을 사용해 자식요소 flex item들을 배치하는 주체

flex item
Flex Container 내부에 레이아웃 되는 항목

Flexbox 레이아웃 구성

1. **Flex Container 지정**
   flex item은 기본적으로 행으로 나열 (주 축의 기본값인 가로방향)
   flex item은 주 축의 시작 선에서 시작
   flex item은 교차 축의 크기를 채우기 위해 늘어남

2. **flex-direction**
   flex item이 나열되는 방향을 지정
   column으로 지정할 경우 주 축이 변경됨
   -reverse로 지정하면 flex item 배치의 시작선과 끝선이 서로 바뀜

3. **flex-wrap**
   flex item 목록이 flex container의 한 행에 들어가지 않을 경우 다른 행에 배치할지 여부 설정

4. **justify-content**
   주 축을 따라 flex item과 주위에 공간을 분배

5. **align-content**
   교차 축을 따라 flex item과 주위에 공간을 분배
   flex-wrap이 wrap 또는 wrap-reverse로 설정된 여러 행에만 적용됨
   한줄짜리 행에는 효과 없음 (flex-wrap이 nowrap으로 설정된 경우)

6. **align-items**
   교차 축을 따라 flex item 행을 정렬

7. **align-self**
   교차 축을 따라 개별 flex item을 정렬

8. **flex-grow**
   남는 행 여백을 비율에 따라 각 flex item에 분배
   -아이템이 컨테이너 내 확장하는 비율을 지정
   flex-grow의 반대는 flex-shrink

9. **flex-basis**
   flex item의 초기 크기 값을 지정
   flex-basis와 width 값을 동시에 적용한 경우 flex-basis가 우선

속성명 Tip
주축 : justify
교차축 : align

Flex Container 관련 속성
display, flex-direction, flex-wrap, justify-content, align-items, align-content

Flex Iten 관련 속성
align-self, flex-grow, flex-basis, order

목적에 따른 속성 분류
배치 : flex-direction, flex-wrap
공간 분배 : justify-content, align-content
정렬 : align-items, align-self

반응형 레이아웃
다양한 디바이스와 화면 크기에 자동으로 적응하여 콘텐츠를 최적으로 표시하는 Web layout 방식

반응형 레이아웃 작성
flex-wrap을 사용해 반응형 레이아웃 작성 (flex-grow, flex-basis 활용)

justify-items 및 justify-self 속성이 없는 이유
필요 없기 때문이다. (margin auto를 통해 정렬 및 배치가 가능)

Web 3일차 교재

Bootstrap
CSS 프론트엔드 프레임워크 (Toolkit)
미리 만들어진 다양한 디자인 요소들을 제공하여 Web site를 빠르고 쉽게 개발할 수 있도록 함

CDN
Content Delivery Network
지리적 제약 없이 빠르고 안전하게 콘텐츠를 전송할 수 있는 전송 기술
서버와 사용자 사이의 물리적인 거리를 줄여 콘텐츠 로딩에 소요되는 시간을 최소화
(Web page 로드 속도를 높임)
지리적으로 사용자와 가까운 CDN 서버에 콘텐츠를 저장해서 사용자에게 전달

Bootstrap CDN
온라인 CDN 서버에 업로드 된 css 및 js 파일을 불러와서 사용하는 것

Bootstrap 기본 사용법
{property}{sides}-{size}로 구성된다.
<예시>
Hello, World!
m = property, t = sides, 5 = size

Property
m : margin
p : padding

Sides
t : top
b : bottom
s : left
e : right
y : top, bottom
x : left, right
blank : 4 sides

Size
0 : 0 rem, 0px
1 : 0.25 rem, 4px
2 : 0.5 rem, 8px
3 : 1 rem, 16px
4 : 1.5 rem, 24px
5 : 3 rem, 48px
auto : auto, auto

Bootstrap에는 특정한 규칙이 있는 클래스 이름으로 스타일 및 레이아웃이 미리 작성되어 있음

Reset CSS
모든 HTML 요소 스타일을 일관된 기준으로 재설정하는 간결하고 압축된 규칙 세트
HTML Element, Table, List 등의 요소들에 일관성 있게 스타일을 적용시키는 기본 단계

Reset CSS 사용 배경
모든 브라우저는 각자의 'user agent stylesheet'를 가지고 있음 (Web site를 보다 읽기 편하게 하기 위해)
문제 : 해당 설정이 브라우저마다 상이하다. 개발자 입장에서는 매우 골치 아픈 일
-> 모두 똑같은 스타일 상태로 만들고 스타일 개발을 시작하기 위해 사용함

User-agent stylesheets
모든 문서에 기본 스타일을 제공하는 기본 스타일 시트

Normalize CSS
Reset CSS 방법 중 대표적인 방법
웹 표준 기준으로 브라우저 중 하나가 불일치한다면 차이가 있는 브라우저를 수정하는 방법

Bootstrap에서의 Reset CSS
bootstrap은 bootstrap-reboot.css 라는 파일명으로 normalize.css를 자체적으로 커스텀해서 사용하고 있음

Typography
제목, 본문 텍스트, 목록 등

Display headings
기존 Heading보다 더 눈에 띄는 제목이 필요할 경우 (더 크고 약간 다른 스타일)

Inline text elements
HTML inline 요소에 대한 스타일

Lists
HTML list 요소에 대한 스타일

Bootstrap Color system
Bootstrap이 지정하고 제공하는 색상 시스템

Colors
Text, Border, Background 및 다양한 요소에 사용하는 Bootstrap의 색상 키워드
종류 : Text colors, Background colors

Bootstrap Component
Bootstrap에서 제공하는 **UI 관련 요소**
(버튼, 네비게이션 바, 카드, 폼, 드롭다운 등)

Component 이점
일관된 디자인을 제공하여 Web site의 구성 요소를 구축하는데 유용하게 활용

대표 Component 종류
Alerts, Badges, Buttons, Cards, Navbar

Bootstarp을 사용하는 이유
가장 많이 사용되는 CSS 프레임워크
사전에 디자인된 다양한 컴포넌트 및 기능 (빠른 개발과 유지보수)
손쉬운 반응형 웹 디자인 구현
커스터마이징(customizing)이 용이
크로스 브라우징(Cross browsing) 지원
-모든 주요 브라우저에서 작동하도록 설계되어 있음

Semantic Web
웹 데이터를 의미론적으로 구조화된 형태로 표현하는 방식
-> "이 요소가 가진 목적과 역할은 무엇일까?"

HTML Semantic Element
기본적인 모양과 기능 이외에 의미를 가지는 HTML 요소
-> 검색엔진 및 개발자가 Web page 콘텐츠를 이해하기 쉽도록

Semantic Element 종류
header, nav, main, article, section, aside, footer

CSS 방법론
CSS를 효율적이고 유지보수가 용이하게 작성하기 위한 일련의 가이드라인

OOCSS (Object Oriented CSS)
객체 지향적 접근법을 적용하여 CSS를 구성하는 방법론

OOCSS 기본 원칙

1. **구조와 스킨을 분리**
   구조와 스킨을 분리함으로써 재사용 가능성을 높임
   모든 버튼의 **공통** 구조를 정의 + **각각**의 스킨을 정의

2. **컨테이너와 콘텐츠를 분리**
   객체에 직접 적용하는 대신 객체를 둘러싸는 컨테이너에 스타일을 적용
   스타일을 정의할 때 위치에 의존적인 스타일을 사용하지 않도록 함
   콘텐츠를 다른 컨테이너로 이동시키거나 재배치할 때 스타일이 깨지는 것을 방지

책임과 역할
HTML : 콘텐츠의 구조와 의미
CSS : 레이아웃과 디자인

의미론적인 마크업이 필요한 이유
검색엔진 최적화 (SEO)
-검색엔진이 해당 Wen site를 분석하기 쉽게 만들어 검색순위에 영향을 줌

웹 접근성 (Web Accessibility)
-Web site, 도구, 기술이 고령자&장애를 가진 사용자들이 사용할 수 있도록 설계 및 개발하는 것
(스크린 리더 등)

Web 4일차 교재

Bootstrap Grid system
Web page의 레이아웃을 조성하는데 사용되는 **12개의 컬럼**으로 구성된 시스템

Grid system 목적
반응형 디자인을 지원해 Web page를 모바일, 태블릿, 데스크탑 등 다양한 기기에서 적절하게 표시할 수 있도록 도움

반응형 웹 디자인 (Responsive Web Design)
디바이스 종류나 화면 크기에 상관없이, 어디서든 일관된 레이아웃 및 사용자 경험을 제공하는 디자인 기술

Grid system 기본요소
Container
Column들을 담고 있는 공간

Column
실제 컨텐츠를 포함하는 부분
1개의 row안에 12개의 column 영역이 구성
-> 각 요소는 12개 중 몇 개를 차지할 것인지 저장됨

Gutter
컬럼과 컬럼 사이에 여백 영역
Grid system에서 column 사이에 여백 영역으로 x축은 **padding**, y축은 **margin**으로 여백 생성

The Grid System
CSS가 아닌 편집 디자인에서 나온 개념으로 구성 요소를 잘 배치해서 시각적으로 좋은 결과물을 만들기 위함
기본적으로 안쪽에 있는 요소들의 오와 열을 맞추는 것을 기인
정보구조와 배열을 체계적으로 작성하여 정보의 질서를 부여하는 시스템

Grid system으로 Responsive Web Design 구현
12개의 column과 **6개의 breakpoints**를 사용하여 반응형 웹 디자인을 구현

breakpoints
Wen page를 다양한 화면 크기에서 적절하게 배치하기 위한 분기점
화면 너비에 따라 6개의 분기점 제공 (xs, sm, md, lg, xl, xxl) 각 breakpoints마다 설정된 최대 너비 값 **이상으로** 화면이 커지면 grid system 동작이 변경됨

Grid cards
row-cols 클래스를 사용하여 행당 표시할 열(카드) 수를 손쉽게 제어할 수 있음