2020년 6월 19일

**버전 컨트롤**

3가지 기억하면 된다.

Repository : 나의 소스를 저장하는 폴더. Git은 repository를 추적하는 시스템

Commit : 파일 변경 기록함 Git이 기록한 변경사항 기록함.(프로젝트 상황창)

Branch : 나뭇가지 같은 것. 마스터의 복사판. 예를 들어 마스터는 건드리지 않고 마스터파일과 똑 같은 코드로 실험하고 싶을 때 만드는 파일.

브랜치를 만들고 (2) 마스터는 그대로 두고 (3) 브랜치에서 새로운 작업 완료하고 (4) 브랜치를 마스터에 결합

**Git vs Github**

Git : 코드의 변경사항을 추적하는 시스템. 누구에 의해 변경 되었는지 기록하는 것.

Github : 깃에서 변경한 기록파일들을 업로드하는곳.

다른 서비스도 있다. Bitbucket : 사전인 깃 관리가 무료로 가능. (깃은 사적인 관리 유료.)

**HTML**

Hyper Text Markup Language의 약자. 하이퍼 텍스트는 중요치 않음. Markup은 내가 공부하다 중요한 것을 형광 펜으로 밑줄 긋는것과 같음

**CSS**

디자인 하는 것

6월 20일

HTML

태그의 구조 : <이름 속성 = “값”>내용</이름>



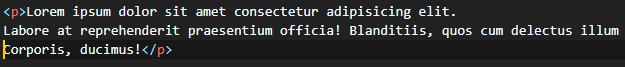
Human = 태그 이름 gender = 속성 “male”=속성값

제목의 경우



<title>제목</title>으로 구성

문단의 경우



<p>내용</p>

링크의 경우

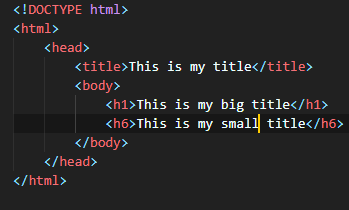


<a href=”주소”>내용</a>

만약 새 창에서 웹 페이지가 열리게 하고 싶다면



Target=”\_blank” 추가해주면 된다.



**<!DOCTYPE html>**

웹 브라우저한테 html문서라는 것을 알리는 태그 혼자 열고 닫고 한다.

<html></html> 사이에 있는 태그는 모두 html 문서라고 보면 된다.

문서는 크게 2가지로 나누어진다

**Head, body**

Head : 유저에게 보이지 않음. 웹 브라우저에게 내 웹사이트에 필요한 정보만 제공해줄 뿐.

Body : 우리가 보는 웹사이트는 body영역.

H1 : 큰 제목을 쓰고 싶을 때 <h1>제목</h1> 1부터 6까지 있고, 갈수록 작아짐.



**Meta : 추가정보**

Meta name description : 웹 페이지에서 검색했을 때 제목 밑에 나오는 간략한 설명

Meta name author : 홈페이지 작성자

Semantic : 의미가 있는 태그 ex)h1,

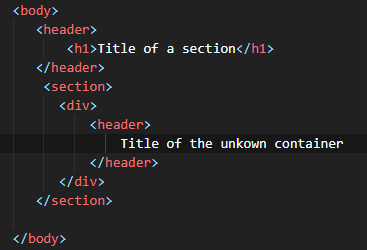
Non-semantic : 의미가 없는 태그. 제목, 문단, 네비게이션 등. ex) div , span

박스 같은 것이 필요할 때 dvi 태그를 쓸 것임

**태그 이름**

Div를 많이 쓰면 뭐가 뭔지 헷갈린다.

Ex)



헤더가 두 개 있는데 뭐가 뭔지 모른다.

헤더에 이름을 짓는건 중요하다. 나중에 CSS로 명령을 해야하기 때문에.



ID는 고유한 것. Class는 여러 개 존재가능.

Ex) ID = 여권번호 Class = 국적

6월 21일

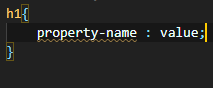
**CSS**

뒤에 확장자 .css 파일 생성

CSS는 두 가지 파트로 구성되어있다

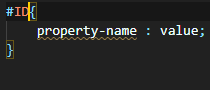
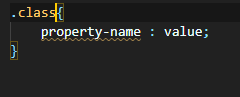
1. Selector : property를 html에 태그에 연결할 때 사용하는 것
2. Property : 무조건 소문자, 공백 없음, 그리고 value 마지막에 세미클론(;)

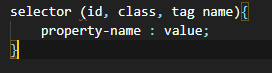
Ex) property-name: value;



h1{}은 selector 괄호안에 들어간게 property

뭐든지 selector로 작성 가능한데, ID를 작석하려면 #ID라고 작성 Class는 .class



이런 식으로 작성 가능

**CSS link**

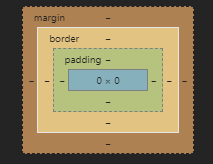
CSS 파일을 연결 하는 방법 두 가지가 있음

1. Inline : 여러 페이지 적용하려면 일일이 복붙 해야함
2. 외부연결 : CSS파일을 따로 만들고 헤더부분에 CSS파일을 링크 걸어주면 된다.

그럼 모든 페이지에 적용된다. (링크 걸어줬을시)

**Box model**

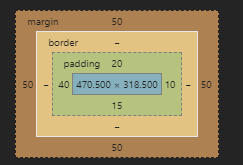
박스는 4가지로 구성되어있음



1. 컨텐츠
2. 보더(경계)
3. 패딩(보더 보다 더 안쪽에 있는 경계)
4. 마진(보더 보다 바깥쪽에 있는 경계)

Padding 단축키 : 상 우 하 좌 순으로 바뀜

Ex) padding : 20 10 20 10; 입력하면 패딩간격이 상 : 20 우 : 10 하 :20 좌 : 10으로 적용된다.



Border 단축키 : 폭, 스타일, 색상

Ex) border : 5px solid red; 5px 실선 빨강색



**Inline vs Block vs Inline Block**

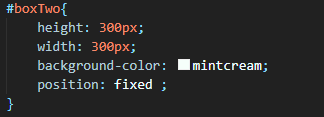
Inline : 서로 옆에 붙는 형태. 넓이, 높이 설정 불가능. 줄 바꿈 기능 없음.

내용이 차지하는 만큼의 넓이 확보.

Inline Block : 박스들이 서로 옆으로 붙음. ◼◼◼형태를 유지하고 싶을 때

Block : 밑으로 박스가 붙음

**Positon property**



static : 속성 값은 position 속성의 default 값 입니다. static으로 지정하게 되면 각 요소들의 고유 특징에 따라 위치가 결정된다고 생각하시면 됩니다.

fixed : 고정 스크롤 내리고 올려도 따라온다

relative : 속성 값은 자신을 기준으로 위치가 결정됩니다. 자신이 부모가 된다고 생각하시면 이해하기 쉽다.

absolute : 속성 값은 자신의 현재 상위태그에 해당하는 요소에서 특정 위치만큼 떨어진 거리를 설정할 수 있다.

**6월 22일**

Fluid layouts with Flexbox

Flex는 부모 클래스에다가만 적용.

Flex-direction을 row로 하면 justiy-content는 수평으로, align-item은 수직으로 적용

Flex-direction을 column으로 하면 justiy-content는 수직으로, align-item은 수평으로 적용

<https://flexboxfroggy.com/#ko> 참고