



05-18강의 트랜지션, 미디어쿼리, Git



Tip

1. 속성을 하나하나 외울 필요 없습니다!

- 자주 쓰는 기능은 자연스럽게 외워집니다.
- 어떻게 하는지 잊어 버리셨다 구글링 하세요
- 속성값은 mdn에서 찾아보시면 됩니다.
- mdn에서 찾아보실 때, 브라우저 지원 여부도 봐주면 좋습니다.

2. 구글링을 습관화 하세요!

- 우리는 이미 그런 기능이 존재한다는걸 압니다.
아 css에서 마우스를 올릴때 css 바꾸는 방법이 뭐였지?
"css 마우스 올릴때 이벤트, css hover, css mouse over event" 구글링을 습관화 하세요
- 앞으로 프로젝트 만들땐 라이브러리, 프레임워크등 전혀 모르는걸 적용해서 사용해야 합니다 1. 각 기술 stack의 공식문서, 2. 구글링 튜토리얼 키워드 ex) material-ui tutorial

3. css는 명확한정답이 없습니다.

- 개개인마다 구현하는 방식이 다릅니다 그러니 문제와, 지시사항에 의존하지말고 자신만의 방식으로 구현합시다.
- 원하는 디자인과 아이디어를 활용해 html구조부터 꾸미기까지 직접 한번 만들어보세요!

4. 개발자는 게을러야 합니다.

- 개발자는 쓸데없이 반복되는 코드를 줄여야 합니다. 그래야 나중에 코드작성 시간 그리고 유지보수면에서도 훨씬 수월합니다.
- 내가 똑같은 코드를 반복하고 있는거 같다. 그럼 쉬운 방법이 없나? 줄일 수 없나? 항상 생각 하고 코드를 작성하는 버릇을 기르시다.

transform

웹사이트의 특정 요소에 회전, 크기 조절, 위치 이동등 효과를 부여할 수 있다.

속성

- rotate(45deg) : 회전
- scale(3,4) : 확대, 순서대로 width 2배, height 3배
- skew(10deg, 10deg) : 각도 변경, 순서대로 x축, y축
- translate (100px, 200px) : 위치변경 순서대로 x축, y축

순서 상관없이 한번에 사용가능

```
transform: rotate(45deg) scale(2, 3) skew(10deg, 20deg) translate(100px, 200px);
```

transform - CSS: Cascading Style Sheets | MDN

CSS transform 속성으로 요소에 회전, 크기 조절, 기울이기, 이동 효과를 부여할 수 있습니다. transform은 CSS 시각적 서식 모델의 좌표 공간을 변경합니다.

 <https://developer.mozilla.org/ko/docs/Web/CSS/transform>

 mdn web docs

transition

변화의 과정을 보기위해 사용 : 변화하는 시간, 속도, 형태 조절

- transition-property: 적용할 css 속성 선택
- transition-duration: 어느정도의 시간을 소요시킬 것인지
- transition-timing-function: 전환 속도 조절 ex) 천천히 갔다가 빨리, 빠르게, 딱딱 끊어서등
- transition-delay : 전환전, 약간의 텀을 주고싶다 ex) 1초후


한번에 사용가능

```
transition: all 3s linear 1s; /* 한번에 사용할 때 무조건 처음 s는 duration, 후에 delay */
```

transition - CSS: Cascading Style Sheets | MDN


transition CSS속성은 transition-property (en-US), transition-duration (en-US), transition-timing-function (en-US) 과 transition-delay를 위한 단축 속성입니다. 이 속성으로 엘리먼트의 두 가지 상태 사이에 변화를 줄 수 있습니다. 엘리먼트의 각 상태는 가상 클래스 를 사용해 정의된 :hover 이나 :active 또는 자바스크립트를 사용해 동적으로 만들어진 것


 <https://developer.mozilla.org/ko/docs/Web/CSS/transition>

 mdn web docs

transition-timing-function - CSS: Cascading Style Sheets | MDN

The transition-timing-function CSS property sets how intermediate values are calculated for CSS properties being affected by a transition effect.

 <https://developer.mozilla.org/en-US/docs/Web/CSS/transition-timing-function>

 mdn web docs

hover

특정 영역의 마우스를 올렸을 때, css를 수정 할 수 있다.

```
.box {
  background-color: red;
  width: 500px;
  height: 500px;
}

.box:hover {
  background-color: orange;
  transition: all 3s linear 1s;
}
```

이렇게 하면 .box class의 영역의 마우스를 올리면 1초 딜레이 후, 3초간 일정하게 배경색이 바뀐다

!! transition이랑 같이 쓰는 경우가 많고 실무에서도 매우 유용하고 많이 쓰인다 !!

반응형 웹 (@media)

모바일, 태블릿, 데스크탑등 여러 디바이스의 맞춰서 css를 꾸밉니다.
반응형 웹은 요즘 웹에는 거의 필수로 장착되어 있다고 보면 된다.

viewport

뷰포트는 현재 보고 있는 디바이스 화면의 영역을 나타낸다.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

width=device-width = 웹페이지가 디바이스의 너비 만큼 설정된다.

initial-scale=1.0 : 배율, 항상 기본값으로 사용해야한다.

@media 쿼리 란?

width를 설정하고 일정 width가 도달할 시 css가 적용된다.

```
@media (조건문) { 실행코드 }
```

반응형 웹 디자인에는 방법

모바일 우선

```
//기본으로 작성되는 CSS = 576px보다 작은 화면에서 작동

// 가로모드 모바일 디바이스 (가로 해상도가 576px 보다 큰 화면에 적용)
@media (min-width: 576px) { ... }

// 태블릿 디바이스 (가로 해상도가 768px 보다 큰 화면에 적용)
@media (min-width: 768px) { ... }

// 데스크탑 (가로 해상도가 992px 보다 큰 화면에 적용)
@media (min-width: 992px) { ... }

// 큰화면 데스크탑 (가로 해상도가 1200px 보다 큰 화면에 적용)
@media (min-width: 1200px) { ... }
```

데스크탑 우선

```
// 기본으로 작성되는 CSS = 1200px 부터 적용

// 데스크탑 (가로 해상도가 1199px 이하 적용)
@media (max-width: 1199px) { ... }

// 태블릿 디바이스 (가로 해상도가 991px 이하 적용)
@media (max-width: 991px) { ... }

// 가로모드 모바일 디바이스 (가로 해상도가 767px 이하 적용)
@media (max-width: 767px) { ... }


// 세로모드 모바일 디바이스 (가로 해상도가 575px 이하 적용)
@media (max-width: 575px) { ... }
```

각자 장단점이 있습니다. 취향의 따라 갈리고
저는 모바일 우선으로 사용합니다 더 직관적이고, 개발하기 편합니다.
모바일이 pc보다 ui가 간단해서 작은 모바일 기기부터 큰 데스크탑까지
작은것에서 부터 하나하나 추가해 가는게 좋기 때문입니다.

상속: 위에 쓴 media쿼리 css들은 none을 하지 않는이상 상속됩니다.

[CSS] 모바일 우선 vs 데스크탑 우선Breakpoint

반응형 디자인의 Breakpoint 기준을 큰 화면에서 작은 화면으로 옮겨가시나요, 아니면 작은 화면에서 큰 화면으로 옮겨가시나요?

 <https://ujeon.medium.com/css-%EB%AA%A8%EB%B0%94%EC%9D%BC-%EC%9A%B0%EC%84%A0-vs-%EB%8D%B0%EC%8A%A4%ED%81%AC%ED%83%91-%EC%9A%B0%EC%84%A0breakpoint-3ed09d033c49>



Git

Git: 프로젝트 버전 관리 시스템

Q. Git을 왜 사용해야 할까?

개발을 하다 보면 수정, 리팩토링, 유지보수가 제일 중요합니다. 프로젝트는
끝이 없다고 봐도 무방합니다. 휴대폰 앱, 게임만 봐도 계속 업데이트를 하고 버전이 올라가는걸 보실 수 있습니다. Git은 버전 관리를 쉽게 하게 만들어주고, 협업을 하기에든 편한 개발자라면 꼭 사용해야하는 버전 관리 시스템입니다.

Git의 장점

- 버전 관리, 프로젝트의 수정을하고 commit을 하면 없었던 기능이 필요하거나 알 수 없는 에러가 발생한 경우 전의 버전으로 되돌리기 가능
- commit을 한 log도 볼 수 있고, 다른 개발자의 코드가 변경된 이력을 쉽게 확인할 수 있습니다.
- 특정 시점에 저장된 버전과 비교할 수 있습니다.
- 큰 프로젝트에는 여러 명의 개발자가 역할을 나눠 개발을 진행합니다 개발을하고 코드를 합쳐야 하는데, git을 사용하면 코드를 합치는게 쉽고 다른 사람과 수정한 부분, 코드가 겹친다면 합칠 수 없도록 경고 메세지도 줍니다. 그렇기 때문에 코드를 덮어써버리는 실수를 예방할 수 있습니다.

로컬 저장소 : 우리가 사용하는 pc

원격저장소 : git의 존재하는 원격 저장소

Git 명령어

- git clone (저장소 url) : 원격 저장소에 있는 프로젝트를 가져올 수 있다.
- git remote add (원격 저장소 url) (저장소 url) : 새로운 원격 저장소 추가
- git add (파일명) : 수정된 파일을 수정 완료 처리함
- git commit -m "메세지" : 버전을 한번 올린다고 볼 수 있음 add 로 추가된 파일들을 메세지와 함께 저장
- git push -u origin dev : 지역 브랜치를 동일한 이름의 원격 브랜치에 저장

- git log : 변경 사항을 보여주는 패치와 함께 로그 표시
- git pull (원격 저장소): origin 저장소에 변경사항을 현재 내 지역 브랜치와 합치기
- git status : 수정된 파일중에, add 된 파일, 수정이됐지만 안된 파일 확인가능
- git branch (브랜치명) : 새로운 브랜치를 만든다.
- git branch -d (브랜치명) : -d 는 delete 브랜치를 삭제한다
- git checkout -b (브랜치명) : 브랜치가 없으면 만들어서 이동, 있으면 이동