HTML과 CSS, Git

2 / HTML과 CSS, Git

목차

01

레이아웃 기초

width, height 속성

<크기>	너비나 높이의 값을 px이나 em 단위로 지정
<백분율>	박스 모델을 포함하는 부모 요소를 기준으로 너빗값이나 높이값을 백분율(%)로 지정
auto	기본값, 박스 모델의 너빗값과 높이값이 콘텐츠 양에 따라 자동으로 결정됨

자주 사용하는 속성

box-sizing

- 박스 모델의 크기를 계산하는 속성
- 종류
  - content-box: 기본값, 너비와 높이가 콘텐츠 영역만을 포함
  - border-box: 너비와 높이가 안쪽 여백과 테두리까지 포함

display 속성

display 속성은 요소를 블록과 인라인 요소 중 어떤 형태로 처리할지를 정의

- 대표 속성
  - inline: 인라인 속성으로 변경
  - block: 블록 속성으로 변경
  - inline-block: 인라인 레벨 요소와 블록 레벨 요소의 속성을 모두 가지고 있으며 마진과 패딩 지정 가능
  - flex: 1차원 레이아웃을 정렬하기 위한 방법
  - grid: 2차원 레이아웃을 정렬하기 위한 방법
  - none: 디스플레이(표시)하지 않음. (요소에서 사라지진 않고, 화면에서 표시되지만 않음)

자주 사용하는 속성

position

- 문서 상에서 요소를 배치하는 방법을 정의합니다.
- position 속성값 종류
  - static: 문서의 흐름에 맞춰 배치. 기본값
  - relative: 일반적인 문서 흐름에 따라 배치하되, 상하좌우 위치 값에 따라 오프셋을 적용
  - absolute: 요소를 일반적인 문서 흐름에서 제거하고, 상위 요소 중 가장 가까운 position 지정 요소에 대해 상대적으로 오프셋을 적용한다. position 지정 요소란 position 속성에 속성값이 정의되어 있는 요소

자주 사용하는 속성

웹 요소의 위치를 정하는 left, right, top, bottom 속성

position이 요소의 배치방법을 정의하면 top, left, bottom, right가 최종 위치를 결정하는 방식으로 필요에 따라 선택적으로 사용가능

- left: 기준 위치와 왼쪽으로 얼마나 떨어져 있는지
- right: 기준 위치와 오른쪽으로 얼마나 떨어져 있는지
- top: 기준 위치와 위쪽으로 얼마나 떨어져 있는지
- bottom: 기준 위치와 아래쪽으로 얼마나 떨어져 있는지

자주 사용하는 속성

의사 클래스

의사클래스(가상클래스)는 선택자에 추가하는 키워드로, 요소가 어떤 특정한 상태가 되었을 때 요소를 선택하겠다는 의미

```
선택자:의사클래스{
    속성명: 속성값;
}
```

```
h1:hover{
    color: red;
}
```

=> h1 요소에 마우스 커서가 올라오면(hover) 글자를 빨간색으로 하겠다!

자주 사용하는 속성

의사 클래스

의사클래스	의미
hover	마우스 포인터가 요소에 올라가 있다.
active	사용자가 요소를 활성화했다. (예를 들면, 마우스로 누르기와 같은)
focus	요소가 포커스를 받고 있다.
disabled	비활성 상태의 요소이다.
nth-child()	형제 사이에서의 순서에 따라 요소를 선택한다.

https://developer.mozilla.org/ko/docs/Web/CSS/pseudo-classes

02

트랜스폼, 애니메이션, 트랜지션

transform

transform 속성은 HTML 요소를 회전, 크기 조절, 기울이기, 이동 효과를 나타낼 때 사용

transform 속성 값으로 특수한 함수를 넣어주면 됨, 단 해당 요소의 display 속성이 block 또는 inline-block 이여야함

```
/* x축(가로)으로 20px 이동 */
transform: translateX(20px);

/* y축(세로)으로 40px 이동 */
transform: translateY(40px);

/* x축(가로)으로 20px, y축(세로)으로 40px 이동 */
transform: translate(20px, 40px);
```

transform

- translate(x, y)
  - 지정한 크기만큼 x, y 축으로 이동
- scale(x, y)
  - 지정한 크기만큼 x, y축으로 확대 및 축소 됨
- rotate(각도)
  - 지정한 각도만큼 회전(+ 시계방향, - 시계반대방향)
- skew(x, y)
  - 지정한 각도만큼 x, y축으로 왜곡함

Transition

트랜지션은 웹 요소의 배경색을 바꾸거나 도형의 테두리를 사각형에서 원형으로 바꾸는 거처럼 스타일 속성이 바뀌는 것을 말함

즉, 웹 요소의 스타일 속성이 시간에 따라 바뀌는 것을 트랜지션이라고 함

Transition

트랜지션 속성

- transition-property: 트랜지션 대상 지정 ex) width, height
- transition-duration: 트랜지션 실행 시간 지정 ex) 2s
- transition-timing-function: 트랜지션 실행 형태 지정 ex) ease-in
  - ease: 처음에는 천천히 시작하고 점점 빨라지다가 마지막엔 천천히 끝냄, 기본값
- transition-delay: 트랜지션 지연시간을 지정 ex) 3s
- transition: 위 속성들을 한꺼번에 지정 ex) transition: all 2s easn-in 0

Animation

애니메이션은 CSS가 움직이거나 변화하는 것을 말함

이때 CSS 스타일이 변화하는 지점을 키프레임(keyframe)이라고 함

https://coolcssanimation.com/

Animation

애니메이션 규칙

```
@keyframes 애니메이션명 {
    선택자 { 스타일 }
    …
}
```

➔

```
@keyframes 애니메이션명 {
    from { 스타일 }
    to { 스타일 }
}
```

Animation

애니메이션 속성

- animation-name: 애니메이션의 중간 상태를 지정하기 위한 이름을 정의
- animation-duration: 한 싸이클의 애니메이션이 얼마에 걸쳐 일어날지 지정
- animation-delay: 엘리먼트가 로드되고 나서 언제 애니메이션이 시작될지 지정
- animation-iteration-count: 애니메이션이 몇 번 반복될지 결정
- animation-play-state: 애니메이션을 멈추거나 다시 시작할 수 있음
- animation-timing-function: 중간 상태들의 전환을 어떤 시간간격으로 진행할지 지정
- animation-fill-mode: 애니메이션이 시작되기 전이나 끝나고 난 후 어떤 값이 적용될지 지정

03

반응형 웹

반응형 웹

반응형 웹은 다양한 기기에서 웹 사이트를 이용할 때, 화면의 크기에 맞추어 적절하게 조정되는 웹사이트를 말합니다. 이를 위해서는 다음과 같은 기술과 방법들이 사용됩니다.

- 미디어 쿼리(Media Queries)
  - 미디어 타입
  - 조건에 대한 물음(쿼리)
- 뷰포트
  - 단위: vw, vh, vmin, vmax

반응형 웹

```
@media 미디어_타입 and (조건에_대한_물음) {
    /*
    미디어 타입과 조건을
    모두 만족할 때 덮어씌울
    스타일 선언문
    */
}
```

반응형 웹

조건에 대한 물음

속성명	정의
min-width	디스플레이 영역의 최소 너비
max-width	디스플레이 영역의 최대 너비
min-height	디스플레이 영역의 최소 높이
max-height	디스플레이 영역의 최대 높이

속성명	정의
orientation	portrait 또는 landscape 감지
color	기기의 색상당 비트 수
color-index	출력 기기의 색상 테이블 수
aspect-ratio	디스플레이 영역의 너비와 높이의 비율

반응형 웹

```
@media screen and (max-width: 768px) {
    /*
    화면(screen)의
    너비가 768px 이하일 경우에
    여기에 정의된 스타일 선언문을
    추가 적용할 것이다!
    */
}
```

반응형 웹

- min-width: 현재크기 >= min-width로 지정된 값
- max-width: 현재크기 <= max-width로 지정된 값
- @media screen and (min-width: 768px) and (max-width: 1439px)
  - 768px <= 현재크기 <= 1439px

display 속성

플렉스박스

플렉스박스는 행 또는 열을 주축으로 설정하여 웹 요소를 배치 및 정렬하는 1차원 레이아웃 방식을 말합니다.

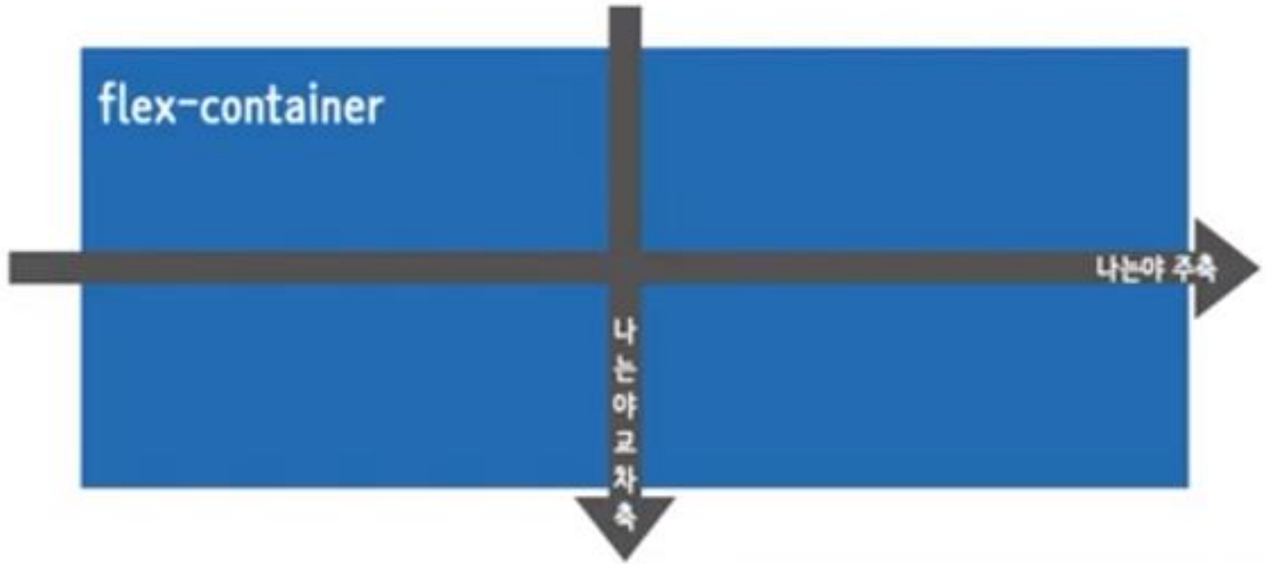
플렉스박스 방식에서, 요소의 배치와 정렬은 플렉스 컨테이너와 플렉스 아이템 간의 상호작용을 통해 결정됩니다.

플렉스 컨테이너 : 플렉스박스 방식으로 레이아웃을 결정할 요소
플렉스 아이템 : 플렉스 컨테이너 내부에서 플렉스박스 방식으로 배치되는 요소

display 속성

플렉스박스

플렉스박스 방식은 두 개의 축을 제공합니다. 그 중 하나의 축을 주축삼아 요소를 배치하는데, 주축의 기본값은 가로 방향(왼쪽에서 오른쪽)입니다.



display 속성

플렉스박스

종류	설명
justify-content	주축 방향의 정렬 방법
align-items	교차축 방향의 정렬 방법
align-self	교차축에 있는 개별 항목의 정렬 방법
align-content	교차축에서 여러 줄로 표시된 항목의 정렬 방법

display 속성

그리드 레이아웃

- 그리드 컨테이너: 그리드 방식으로 레이아웃을 결정할 요소
- 그리드 아이템: 그리드 컨테이너 내부에서 그리드 방식으로 배치되는 요소

display 속성

- grid-template-columns
- grid-template-rows
- gap(grid-gap)
- 트랙 관련 함수
- grid-column & row
- grid-template-areas
- grid-area
- align-items
- align-self
- justify-items
- justify-self
- align-content
- justify-content

display 속성

grid-template-columns

- 그리드 컨테이너의 트랙 중 열(column) 트랙 내 아이템들의 크기를 지정할 수 있는 속성입니다. 그리드 컨테이너에서 트랙이란 행 또는 열을 뜻합니다.

속성값	의미
none	기본값. 명시한 값이 없으므로 암묵적으로 값이 정해집니다.
수치	길이를 나타내는 음수가 아닌 값을 지정합니다.
그 외	다양한 키워드나 CSS 함수를 사용해 지정하기도 합니다.

display 속성

grid-template-rows

- 그리드 컨테이너의 트랙 중 행(row) 트랙 내 아이템들의 크기를 지정할 수 있는 속성입니다.

속성값	의미
none	기본값. 명시한 값이 없으므로 암묵적으로 값이 정해집니다.
수치	길이를 나타내는 음수가 아닌 값을 지정합니다.
그 외	다양한 키워드나 CSS 함수를 사용해 지정하기도 합니다.

display 속성

gap(grid-gap)

그리드 아이템 사이의 간격을 지정하는 속성입니다. 행에서의 간격과 열에서의 간격을 똑같이 지정할 수도 있고, 각자 따로 지정할 수도 있습니다.
row-gap과 column-gap의 단축속성입니다.

속성값	의미
normal	기본값. 명시한 값이 없는 기본적인 상태.
수치	길이를 나타내는 값을 지정합니다. 다양한 단위 사용 가능.

display 속성

트랙 관련 함수

그리드 컨테이너의 트랙(행과 열) 크기를 지정할 때 사용할 수 있는 유용한 함수들이 있습니다.

함수	기능
repeat()	반복되는 값을 자동으로 처리할 수 있는 함수.
minmax()	최솟값과 최댓값을 각각 지정할 수 있는 함수.
auto-fill & auto-fit	반응형을 고려해 사용할 수 있는 키워드들 (함수X)

display 속성

grid-column & row

grid-column과 grid-row 속성을 이용하면 그리드 컨테이너의 줄 번호를 이용해 아이템을 배치할 수 있습니다.

grid-container 행, 열 각각의 줄 번호는 이렇게

display 속성

grid-template-areas

그리드 영역(아이템)의 이름을 이용해 레이아웃의 형태를 정의할 수 있습니다.

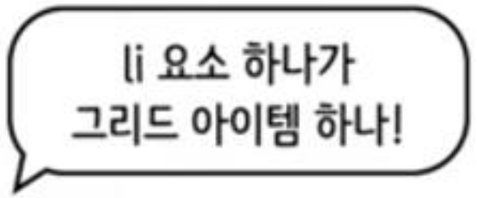
```
grid-template-areas:
  "거 거 호"
  "다 다 청"
  "고 고 고";
```

display 속성

grid-template-areas

그리드 영역(아이템)의 이름을 지정할 때 사용하는 속성입니다.

```
li:nth-child(1){ grid-area: 거; }
li:nth-child(2){ grid-area: 호; }
li:nth-child(3){ grid-area: 다; }
li:nth-child(4){ grid-area: 청; }
li:nth-child(5){ grid-area: 고; }
```



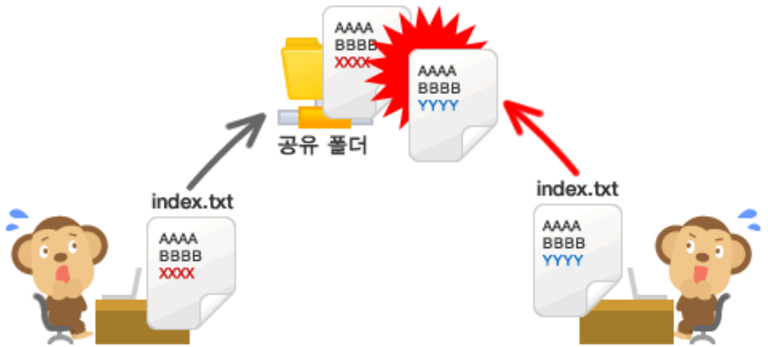
04

Git 개념

Git

아래와 같은 상황을 봅시다.

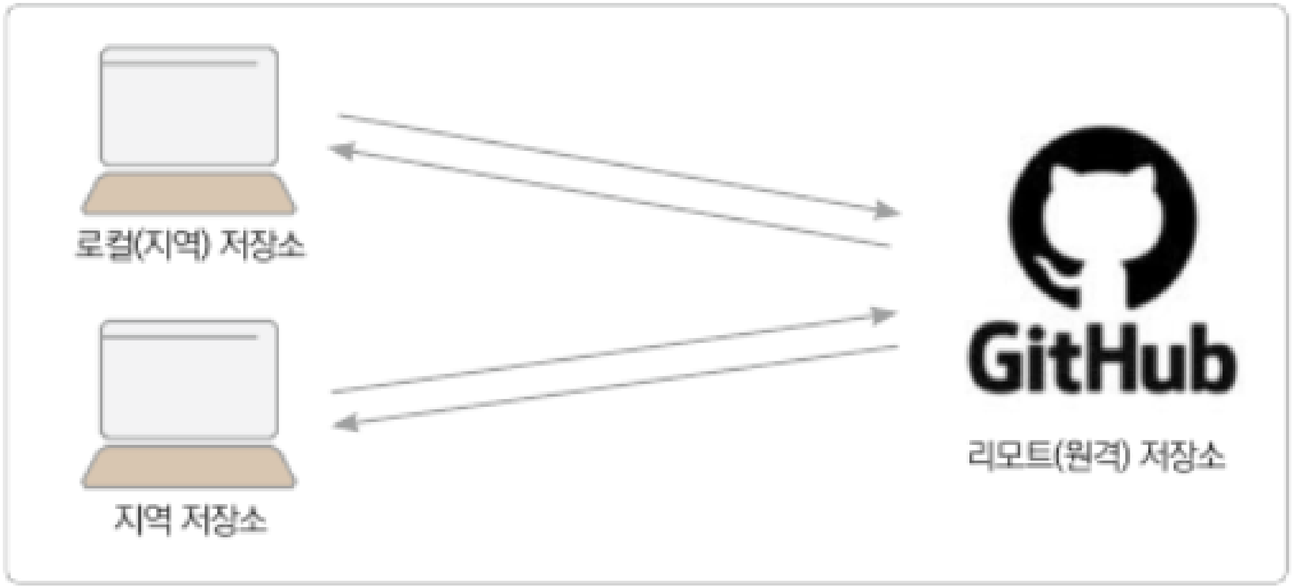
뭐가 최신판인지 알 수 없다



여러 명이 공유하는 파일을 수정해버리면서
다른 사람이 쓴 내용과 충돌이 발생

Git

Git은 분산형 버전 관리 시스템이다



Git

Git 을 설치해봅시다!

Git

Git 이 어떻게 동작하는지 살펴봅시다.

Git

원격 저장소에 커밋 날리는 과정

1. git init
2. 사용자 설정 작업
3. git commit –m …
4. git push …

Git 기본 용어

- 저장소(Repository): Git에서 버전 관리 대상이 되는 디렉토리 또는 파일들의 모음
- 커밋(Commit): Git에서 변경 사항을 저장하는 작업을 의미
- 브랜치(Branch): 독립적인 작업 영역을 의미함, 여러 작업자가 같은 저장소에 동시에 작업할 때, 각자의 작업 내용을 서로 간섭하지 않게끔 할 수 있음
- 푸시(Push): 로컬 저장소에 저장된 변경 사항을 원격 저장소에 업로드하는 작업을 의미
- 풀(Pull): 원격 저장소에서 변경된 사항을 로컬 저장소에서 다운로드하는 작업을 의미

Git 로그 확인

- `git log` 명령어를 사용해 커밋 로그를 확인할 수 있음

브랜치 생성 및 전환

- `git branch` 명령어를 사용하여 브랜치를 생성
- `git checkout` 명령어를 사용하여 브랜치를 전환

원격 저장소 생성 및 푸시

- Github, Gitlab 등의 원격 저장소를 이용하여 Git 저장소를 생성
- `git remote add` 명령어를 사용해 원격 저장소를 추가
- `git push` 명령어를 사용해 로컬 저장소에 저장된 변경 사항을 원격 저장소에 업로드

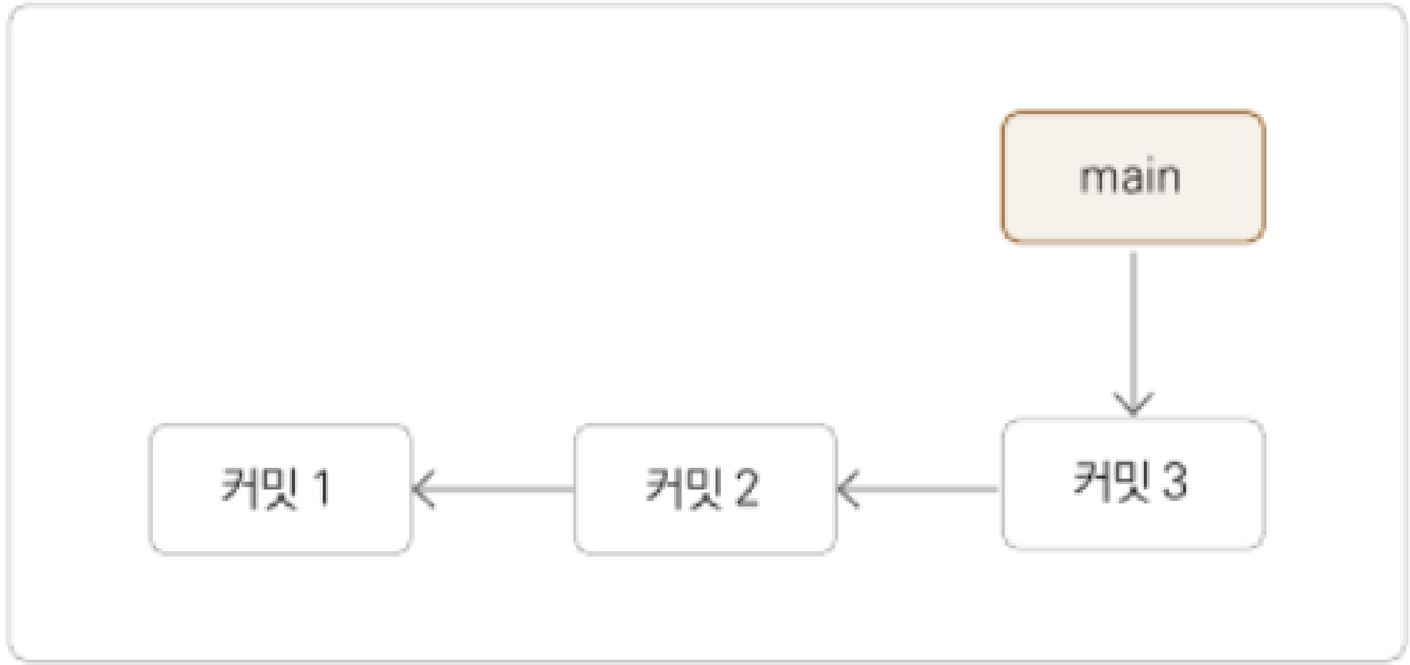
.gitignore

- `.gitignore` 파일을 생성하여 Git에서 추적하지 않을 파일을 설정함
- 이 파일에 등록된 파일들은 로컬 저장소 및 원격 저장소에서 제외됨
- node_modules 같이 용량이 큰 파일, 보안상 민감한 파일등에 사용됨

Git의 협업 기능

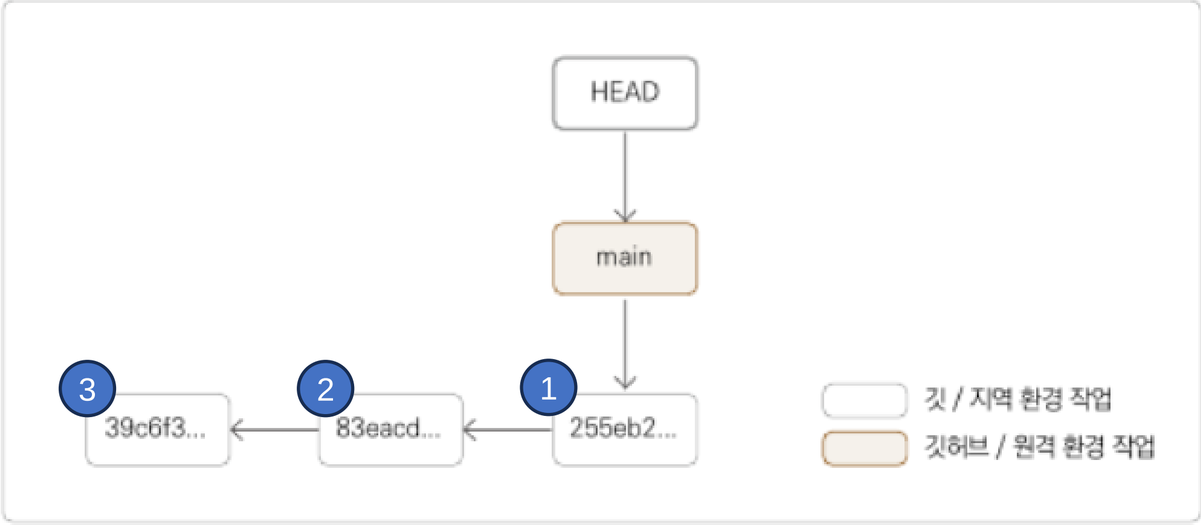
- Git은 여러 명의 작업자가 함께 작업할 때 효율적으로 협업할 수 있는 기능을 제공함
- 풀 리퀘스트(Pull request)를 이용하여 브랜치를 비교하고 변경 사항을 검토하고 병합할 수 있음
- 이슈 트래킹(Issue Tracking)을 이용하여 개발 과정에서 발생한 이슈를 관리하고 해결할 수 있음

브랜치

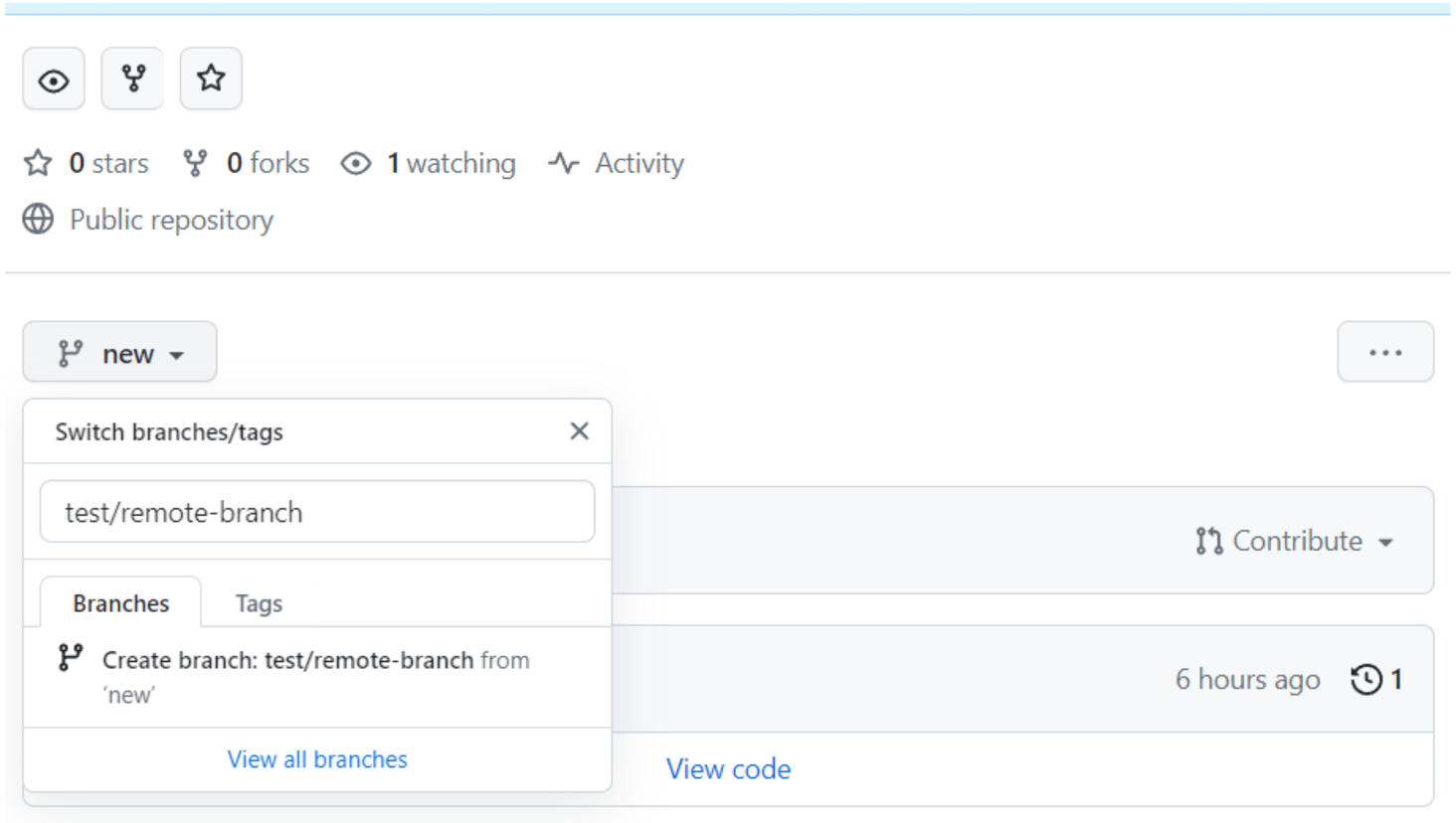


브랜치

```
mastering-git-github % git log --pretty=oneline --graph
* 255eb26c90bce40d348eeee7d1ebc8f71565115b (HEAD -> main, origin/main) Add
hotline to main page 1
* 83eacd81833cca3a5be10313009655b4da21028e Change the title of main page 2
* 39c6f390d12a489c55f21cd318cbf22897ecc4d6 Add initial files and .gitignore 3
```

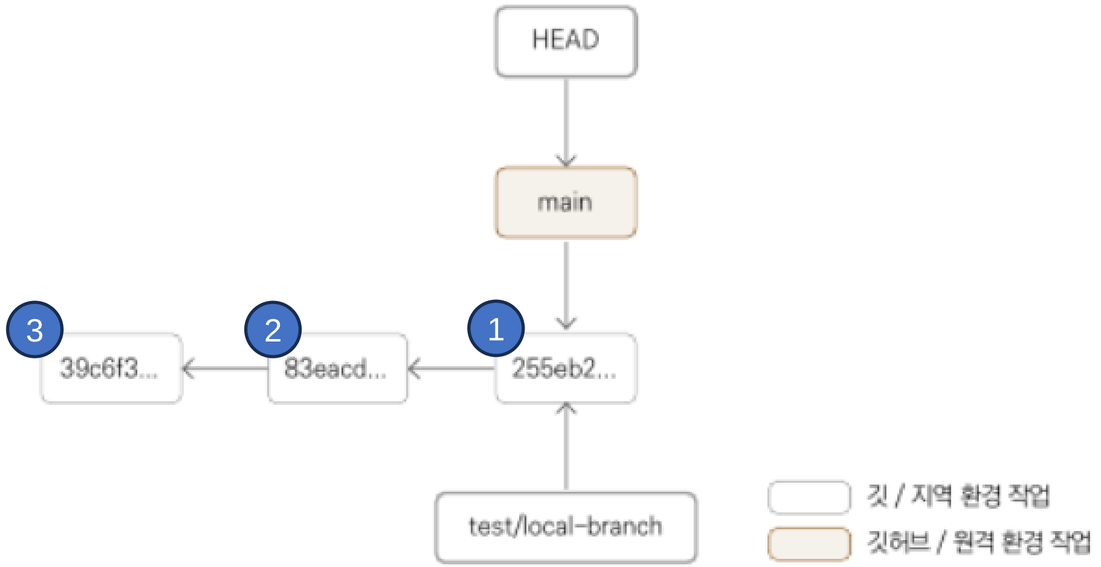


브랜치 실습

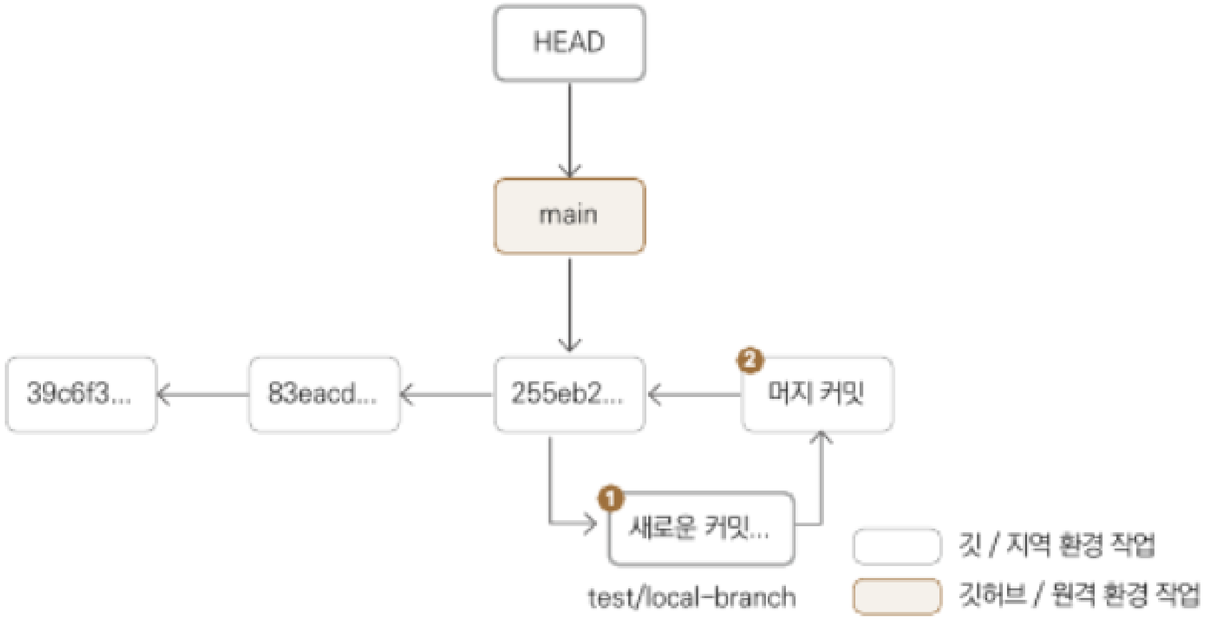

0 stars
0 forks
1 watching
Activity
Public repository
new
Switch branches/tags
test/remote-branch
Branches
Tags
Create branch: test/remote-branch from 'new'
View all branches
Contribute
6 hours ago
1
View code

브랜치 실습

```
mastering-git-github % git log --pretty=oneline --graph
* 255eb26c90bce40d348eeee7d1ebc8f71565115b (HEAD -> main, origin/test/remote-branch, origin/test/local-branch, origin/main, test/remote-branch, test/local-branch) Add hotline to main page (1)
* 83eacd81833cca3a5be10313009655b4da21028e Change the title of main page (2)
* 39c6f390d12a489c55f21cd318cbf22897ecc4d6 Add initial files and .gitignore (3)
```



브랜치 실습


HEAD
main
39c6f3...
83eacd...
255eb2...
2
머지 커밋
1
새로운 커밋...
test/local-branch
깃 / 지역 환경 작업
깃허브 / 원격 환경 작업

브랜치 실습

빨리감기 병합: fast forward

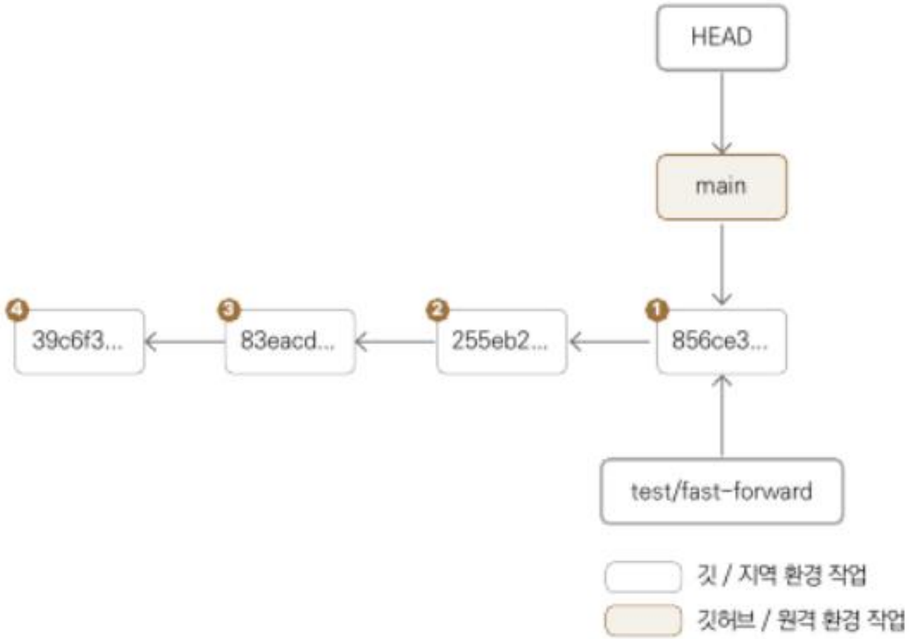
기준 브랜치에 새로운 커밋이 없을 경우 빨리감기 병합으로 진행이 됨

```
mastering-git-github % git log --pretty=oneline --graph
* 856ce3d46f12bb631b2d343346d960ea9315750a (HEAD -> main, test/fast-forward)
Change title 1
* 255eb26c90bce40d348eeee7d1ebc8f71565115b (origin/test/remote-branch,
origin/test/local-branch, origin/main, test/remote-branch, test/local-
branch) Add hotline to main page 2
* 83eacd81833cca3a5be10313009655b4da21028e Change the title of main page 3
* 39c6f390d12a489c55f21cd318cbf22897ecc4d6 Add initial files and .gitignore 4
```

브랜치 실습

빨리감기 병합: fast forward

기준 브랜치에 새로운 커밋이 없을 경우 빨리감기 병합으로 진행이 됨



브랜치 실습

충돌 해결하기

- 충돌이란 두 개 이상의 브랜치에서 같은 파일의 같은 부분을 수정하고 병합하려고 할 때 발생하는 문제입니다. 충돌이 발생하면 Git은 충돌 부분을 표시해줍니다. 예를 들어 다음과 같은 형식으로 나타납니다

```
<<<<<<< HEAD
This is some content from the current branch.
=======
This is some content from another branch.
>>>>>>> another-branch
```

브랜치 실습

충돌 해결하기

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>시스템 v1</title>
</head>
<body>
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
<<<<<<< HEAD (Current Change)
    Git
=======
    Git conflict check
>>>>>>> feature/conflict (Incoming Change)
</body>
</html>
```

브랜치 실습

충돌 해결하기

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>시스템 v1</title>
</head>
<body>
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
<<<<<<< HEAD (Current Change)
    Git
=======
    Git conflict check
>>>>>>> feature/conflict (Incoming Change)
</body>
</html>
```

Git의 기본 개념과 사용법 이해

- 짐코딩
- 유노코딩
- Do it! 한권으로 끝내는 웹 기본 교과서 HTML + CSS + 자바스크립트 웹 표준의 정석
- 박미정의 깃&깃허브 입문