







- 1. SW의 버전 방법
 - v1.45.3
- 2. 버전을 부여할 수 있는 태그 기능
 - 일반태그와 주석태그



- 1. 커밋에 태그를 붙일 수 있다.
- 2. 일반 태그와 주석 태그를 활용할 수 있다.
- 3. 태그 목록을 확인하고 특정 태그의 자세한 정보를 확인할 수 있다.
- 특정 태그를 삭제할 수 있다.







| 버전과태그개요|

실제 SW 버전

- 對 버전(version)
 - 프로그램을 수정하거나 개선할 때마다 코드를 구분하려고 부여된 식별자를 의미
 - 보통 숫자를 사용
 - '년.월'을 사용하기도 함
- ☑ 보통 두 자리 또는 세 자리 형태의 숫자로 작성
 - 1.0
 - **2.1.4**
- **M** Git SW 버전
 - 2.39.0

버전과태그개요



SemVer(Semantic Versioning) 방식

- major.minor.patch
 - 세 자리 숫자 형태로 표기하는 버전
 - 메이저(major) 번호
 - 첫 자리가 0으로 시작하면 아직 초기 개발 중인 제품이라는 의미
 - 정식 버전은 1부터 시작하는데 이를 메이저(major) 버전
 - 마이너(minor) 번호
 - 메이저 버전에서 기능을 추가하거나 변경 사항이 있을 때 바꿈
 - 패치(patch) 번호
 - 버그 수정 등 미미한 변화가 있을 때 바꿈

버전과태그개요





- 특정 커밋(해시 값)에 버전 번호나 다른 이름을 부여하는 기능
 - 보통 v1.4.0

₩ 태그 2가지 종류

- 주석 태그(Annotated tag)
 - 태그 이름 + 정보(태그 작성자 이메일, 태그 시각, 태그 메시지) 포함
- 일반(가벼운) 태그(Lightweight tag)
 - 태그 이름만 포함









주석태그생성





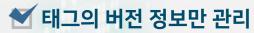
- 주석(누가, 언제, 태그 메시지 등의 정보)이 있는 태그
- 태그 버전 이름 중복 불가능
- ★ \$ git tag -a v1.0.0 -m 'first version'
 - 작성한 사람의 이메일, 날짜, 메시지 등 정보 포함
- **★** \$ git tag −a v1.0.0
 - 기본 설정된 편집기로 메시지 편집
 - \$ git config --global init.defaultBranch 'code --wait'
- **★** \$ git tag −a v1.1.0 commitID
 - 특정 커밋에 태그를 붙임
 - -a가 아니어도 가능
 - \$ git tag v1.1.0 HEAD~











- \$ git tag v1.0.1
- 태그 버전 이름 중복 불가능
 - -a -m 사용 불가능





2 주석태그생성





- 예전 태그부터 표시
 - \$ git tag
 - v0.0.1
 - v1.0.0



○ 최신 커밋부터 표시









2 주석태그생성





- **★** \$ git show v1.0.0
 - Annotated 태그의 여러 정보 표시
 - 태그 이름, 태그 작성자 이메일, 태그 시각, 태그 메시지

```
$ git show v1.0.0
tag v1.0.0
Tagger: ai7dnn <ai7dnn@gmail.com>
Date: Tue Jan 17 22:29:55 2023 +0900
first version
commit 09b8fda7237ef8de960eb4c9eef164f58e94d8bc (HEAD -> main, tag: v1.0.0)
Author: ai7dnn <ai7dnn@gmail.com>
Date: Tue Jan 17 22:28:22 2023 +0900
C
diff --git a/f b/f
index dbee026..1802a74 100644
--- a/f
+++ b/f
@@ -1,2 +1,3 @@
aaa
bbb
+CCC
```

2 주석태그생성





\$ git tag -d v1.0.0
Deleted tag 'v1.0.0' (was 09b8fda)





2 주석태그생성





순서	내용	명령
1	저장소 gtag 생성과 이동	<pre>\$ git init gtag \$ cd gtag</pre>
2	파일 f 생성과 커밋	<pre>\$ echo aaa > f \$ git add f</pre>
		\$ git commit -m A
3	현재 커밋에 일반 태그 붙이기	\$ git log
		\$ git tag v0.0.1
		\$ git logoneline
4	파일 f 수정 후 커밋 연속 2회	<pre>\$ echo bbb >> f</pre>
		\$ git commit -am B
		<pre>\$ echo ccc >> f</pre>
		\$ git commit -am C



2 주석태그생성



▼ 주석 태그 활용과 태그 목록, 태그 삭제

순서	내용	명령
⑤	현재 커밋에 주석 태그 붙이기	<pre>\$ git logoneline \$ git tag -a v1.0.0 -m 'first version' \$ git logoneline \$ git show \$ git show v1.0.0</pre>
6	태그 목록 보기	<pre>\$ git tag \$ git logoneline</pre>
7	태그 삭제	<pre>\$ git tag -d v0.0.1</pre>



2 주석태그생성



☑ 이전 커밋에 일반 태그 활용

순서	내용	명령
8	파일 f 수정 후 커밋 연속 2회	<pre>\$ echo ddd >> f</pre>
		\$ git commit -am D
		<pre>\$ echo eee >> f</pre>
		\$ git commit -am E
9	태그 확인 후 이전 커밋에 일반 태그 붙이기	<pre>\$ git logoneline</pre>
		\$ git tag
		\$ git tag v1.0.1 HEAD~
100	태그 확인	\$ git tag
		<pre>\$ git logoneline</pre>
		\$ git show v1.0.1
		\$ git show v1.0.0





Summary

>>> Sementic Versioning 방식

- 3.34.5
 - Major.minor.patch

>>> 태그 생성

- \$ git tag v1.0.1
- \$ git tag -a v1.0.1 -m 'msg'

>> 태그 자세한 정보 확인

- ◆ \$ git show v1.0.1
- >>> 태그 목록 확인
 - \$ git tag
- >>> 태그 삭제
 - \$ git tag v1.0.1