# Git Internals: 핵심 구조 및 개념 정리 보고서

# Git Internals: 핵심 구조 및 개념 정리 보고서

---

## 📁 Git 핵심 구조: 객체 지향 버전 관리 시스템

| 구성 요소 | 설명 | 저장 방식 |

|------------|-------|------------|

| `blob` | 파일 내용 객체 | SHA-1 해시 + zlib 압축 (내용만 저장, 이름 없음) |

| `tree` | 폴더 구조 스냅샷 | 여러 `blob`, `tree` 포함 + 권한 + 파일명 포함 |

| `commit` | 버전 스냅샷 | 하나의 `tree`를 참조 + author, committer, 메시지, parent 포함 |

| `HEAD` | 현재 작업 위치 | 브랜치(ref)를 가리키는 포인터 |

| `refs/heads/<branch>` | 브랜치 포인터 | 특정 커밋 해시를 가리킴 |

---

## 🔁 Git 내부 흐름

1. \*\*`git init`\*\*: `.git/` 디렉토리 생성 → Git 로컬 DB 초기화

2. \*\*`git add`\*\*:

- 파일 → `blob` 생성

- `.git/index`에 파일 경로와 blob 해시 기록

3. \*\*`git commit`\*\*:

- `index` → `tree` 객체 생성

- `commit` 객체 생성 (tree 해시 참조)

- `HEAD`가 가리키는 브랜치(ref)를 해당 커밋 해시로 업데이트

4. \*\*`git push`\*\*:

- 로컬 `.git/objects`에 있는 객체 중 원격에 없는 것만 전송

- 원격 브랜치(ref)를 새로운 커밋 해시로 업데이트

---

## 🔍 커밋 구조

commit <해시>

tree <트리 해시>

parent <이전 커밋 해시> (없을 수도 있음)

author <작성자>

committer <Git에 반영한 사람>

<커밋 메시지>

> - `author`: 실제 작성자 (변경 가능)

> - `committer`: 이 커밋을 Git DB에 등록한 사람

---

## 🌿 브랜치와 포인터

| 명령어 | 의미 | 결과 |

|--------|------|--------|

| `git branch foo` | 브랜치 생성 | `refs/heads/foo` → 현재 커밋 해시 |

| `git checkout foo` | HEAD 이동 | `.git/HEAD` → `refs/heads/foo` |

| `git reset --hard <해시>` | 포인터 이동 | 현재 브랜치를 특정 커밋으로 이동 + 워킹 디렉토리 일치 |

> ✅ Git은 "순서"를 기억하는 게 아니라, \*\*각 커밋이 부모 커밋을 가리키는 포인터 구조\*\*로 연결됨.

---

## 🔀 Merge vs Rebase

| 항목 | `merge` | `rebase` |

|------|---------|----------|

| 목적 | 브랜치 병합 | 내 커밋을 다른 브랜치 위로 재배치 |

| 이력 | 브랜치 분기, 병합 히스토리 보존 | 한 줄로 정리됨 |

| 충돌 처리 | 병합 시 1번 | 매 커밋마다 충돌 가능성 있음 |

| 커밋 해시 | 유지됨 | 새로 만들어짐 |

| 사용 상황 | 협업 병합, PR | 내 브랜치를 최신으로 업데이트할 때 |

### 📌 요약

- `merge`: "함께 만든 작업을 하나로 합친다"

- `rebase`: "내 커밋을 최신 브랜치 위로 깔끔히 재적용"

---

## ⚠️ Conflict 감지 원리

- Git은 공통 조상(`base`)을 기준으로 두 브랜치의 변경사항 비교

- \*\*같은 줄\*\*을 \*\*양쪽에서 다르게 수정\*\*한 경우 → 충돌 발생

git merge feature

# CONFLICT (content): Merge conflict in file.txt

해결 방법:

1. 충돌 파일 수동 수정

2. `git add <file>`

3. `git commit`

---

## 🧠 철학적 핵심 요약

> ✅ Git은 "내용 중심의 버전 관리 시스템"이다.

> ✅ 모든 것은 객체(blob, tree, commit)로 저장되고, 그 해시를 참조함으로써 버전을 구성한다.

> ✅ 브랜치는 단순히 커밋을 가리키는 이름 있는 포인터이다.

> ✅ 히스토리는 커밋이 부모 커밋을 가리키는 구조로 구성된다.

> ✅ `merge`, `rebase`, `reset`, `checkout`은 모두 "포인터 조작 명령"일 뿐이다.