INFRASTRUCTURE · AUTOMATION · DEVOPS

Git field notes

A practical cheat sheet for reviewing and managing infrastructure changes

01 · Start safely

Confirm repository, branch, and local edits

```
git status --short --branch
git remote -v
git branch --show-current
git fetch --all --prune
```

A reviewed change, end to end

```
git switch main
git pull --ff-only origin main
git switch -c change/ssh-baseline
# edit, then inspect
git diff --check
git diff -- path/to/config
git add path/to/config
git diff --cached
git commit -m "Update SSH baseline"
git push -u origin change/ssh-baseline
```

Stage deliberately. Inspect the staged diff before committing; avoid accidental secrets.

02 · Branch & integrate

Navigate and compare

```
git branch --all
git switch -c change/name
git switch main
git log --oneline --decorate -10
git diff main...HEAD
```

Fetch and pull are different

fetch updates remote-tracking refs; pull also integrates into your branch.

```
git fetch origin
git pull --ff-only origin main
```

Merge or rebase?

merge combines histories and may add a merge commit. Rebase replays commits on a new base and changes their IDs.

Rebase your own unshared topic branch. Coordinate before rewriting shared history.

03 · Inspect & recover

Read history and repository trees

```
git log --graph --oneline --all
git show --stat --oneline HEAD
git ls-tree -r --name-only HEAD
git rev-parse 'HEAD^{tree}'
git diff -- path/to/file
```

Undo with care

```
git restore --staged path
git revert <commit>
git reflog
```

restore discards selected edits; revert adds a new undo commit. Inspect before using reset, especially --hard.

Delivery is a separate step

A reviewed merge may trigger CI/CD. GitOps controllers pull declared state and reconcile it; Git alone deploys nothing.

Confirm remote · branch · account · target

Official references: git-scm.com/docs · docs.github.com · docs.gitlab.com · learn.microsoft.com/azure/devops

Nikunj Master · nikunjmaster.org