GIT Cheat Sheet

### Basic Commands

- **`git init`**: Initializes a new Git repository in the current directory.

```
git init
```

- **`git clone`**: Clones an existing repository into a new directory.

```
git clone https://github.com/user/repo.git
```

- **`git status`**: Shows the status of changes as untracked, modified, or staged.

```
git status
```

- **`git add`**: Stages changes for the next commit.

```
git add <file>
git add .   # Adds all changes in the current directory
```

- **`git commit`**: Records the staged changes in the repository.

```
git commit -m "Commit message"
git commit -a -m "Commit message"   # Skips staging area and commits all changes
```

- **`git push`**: Updates the remote repository with local commits.

```
git push origin main
```

- **`git pull`**: Fetches changes from the remote repository and merges them into the current branch.

```
git pull origin main
```

### Branching and Merging

- **`git branch`**: Lists all branches or creates a new branch.

```
git branch        # Lists all branches
git branch <name> # Creates a new branch
```

- **`git checkout`**: Switches to a different branch or restores working directory files.

```
git checkout <branch>        # Switch to the specified branch
git checkout -b <new-branch>  # Create and switch to a new branch
```

- **`git merge`**: Merges changes from one branch into the current branch.

```
git merge <branch>
```

- **`git rebase`**: Reapplies commits on top of another base tip.

```
git rebase <branch>
```

### Remote Repositories

- **`git remote`**: Manages remote repository connections.

```
git remote -v            # Lists remote connections
git remote add origin URL   # Adds a new remote repository
```

- **`git fetch`**: Downloads objects and refs from another repository.

```
git fetch origin
```

- **`git push`**: Pushes local changes to the remote repository.

```
git push origin <branch>
```

- **`git pull`**: Fetches and integrates with another repository or a local branch.

```
git pull origin <branch>
```

### Stashing and Cleaning

- **`git stash`**: Temporarily saves changes that are not ready to be committed.

```
git stash         # Stashes changes
git stash apply   # Applies stashed changes
git stash pop     # Applies and removes stashed changes
```

- **`git clean`**: Removes untracked files from the working directory.

```
git clean -f     # Forcefully remove untracked files
git clean -fd    # Remove untracked files and directories
```

### History and Logs

- **`git log`**: Shows the commit history.

```
git log
git log --oneline        # Condensed log view
git log --graph --all    # Graphical log view
```

- **`git diff`**: Shows changes between commits, commit and working tree, etc.

```
git diff                # Shows unstaged changes
git diff --staged       # Shows staged changes
git diff <commit1> <commit2> # Shows changes between two commits
```

- **`git show`**: Shows various types of objects.

```
git show <commit>       # Shows details of a specific commit
```

### Undoing Changes

- **`git reset`**: Resets current HEAD to the specified state.

```
git reset --soft <commit>  # Keeps changes in the working directory
git reset --hard <commit>  # Discards all changes
```

- **`git revert`**: Creates a new commit that undoes the changes from a previous commit.

```
git revert <commit>
```

- **`git checkout`**: Restores working directory files.

```
git checkout -- <file>   # Discards changes in the working directory
```

### Tags

- **`git tag`**: Creates, lists, deletes or verifies a tag object.

```
git tag            # Lists all tags
git tag -a v1.0 -m "Version 1.0"   # Creates an annotated tag
git push origin v1.0 # Pushes a specific tag to remote
```

### Collaboration and Review

- **`git fork`**: Copies a repository from a user’s GitHub account to your own.

- **`git pull request`**: Proposes changes to the upstream repository.

### Configuration

- **`git config`**: Gets and sets repository or global options.

```
git config --global user.name "Your Name"
git config --global user.email "your.email@example.com"
git config --list          # Lists all settings
```

### Aliases

- **`git alias`**: Creates shortcuts for commonly used Git commands.

```
git config --global alias.co checkout
git config --global alias.br branch
git config --global alias.ci commit
git config --global alias.st status
```