कुराहरू उल्ट्याउनु (Undoing Things) ---

कुनै पनि चरणमा, तिमीले केही कुरा उल्ट्याउन चाहन सक्छौ। यहाँ, हामीले तिमीले गरेका परिवर्तनहरू उल्ट्याउनका लागि केही आधारभूत उपकरणहरूको समीक्षा गर्नेछौं। सावधान रहनू, किनभने यीमध्ये केही कुराहरू उल्ट्याउन नसकिने हुनसक्छ। यो Git को त्यस्ता केही क्षेत्रहरूमध्ये एक हो जहाँ तिमीले गल्ती गरेमा केही काम गुम्न सक्छ।

सबैभन्दा सामान्य undo तब हुन्छ, जब तिमीले commit छिटो गर्यौं र सम्भवतः केही files थप्न बिर्सियौ, या `commit message` नै गलत गर्यौं। यदि तिमीले त्यो commit लाई फेरि गर्न चाहन्छौ भने, बिर्सिएका थप परिवर्तनहरू गर, ती stage गर, अनि commit लाई फेरि `--amend` विकल्प प्रयोग गरी गर:

```
$ git commit --amend
```

यो command ले तिम्रो staging area लाई commit को लागि प्रयोग गर्छ। यदि तिमीले पछिल्लो commit पछि कुनै परिवर्तन नगरेको छौ भने (जस्तै यो command तिमीले पछिल्लो `commit` तुरुन्तै पछि चलायौ भने), तिम्रो `snapshot` बिल्कुलै पहिलेको जस्तै देखिन्छ, र तिमीले केवल commit message परिवर्तन गर्नेछौ।

त्यसै `commit-message editor` खुल्छ, तर यसमा पहिलेको commit को `message` पहिले नै हुन्छ। तिमीले `message` सधैंझैं सम्पादन गर्न सक्छौ, तर यसले तिम्रो पहिलेको `commit` लाई `overwrite` गर्छ।

उदाहरणका लागि--

यदि तिमीले commit गर्यौं र त्यसपछि एउटा file जसलाई यो commit मा थप्न चाहिएको थियो, त्यसको changes stage गर्न बिर्सियौ भने, यस्तो गर्न सक्छौ:

```
$ git commit -m 'Initial commit'
$ git add forgotten_file
$ git commit --amend
```

यसको परिणामस्वरूप तिमीले एउटा मात्र commit पाउनेछौ – दोस्रो commit ले पहिलो commit को परिणामलाई प्रतिस्थापन गर्छ।

💡 **महत्त्वपूर्ण कुरा:**

जब तिमी आफ्नो पछिल्लो commit लाई amend गर्दैछौ, तिमीले यसलाई केवल सुधार्दै होइन, पूरै नयाँ **`commit`** मा रूपान्तरण गर्दैछौ, जुन पुरानो **`commit`** लाई हटाएर नयाँ **`commit`** लाई त्यस ठाउँमा राख्छ।

यसको प्रभावले गर्दा, यसअघि भएको **`commit`** कहिल्यै भएको जस्तो देखिँदैन, र यो तिम्रो **`repository`** को **`history`** मा पनि देखिँदैन।

Commit लाई amend गर्नको फाइदा

Commits लाई amend गर्नको स्पष्ट फाइदा भनेको आफ्नो पछिल्लो commit मा साना सुधारहरू गर्नु हो, जसले गर्दा repository history “Oops, forgot to add a file” वा “Darn, fixing a typo in last commit” जस्ता **`commit messages`** ले भरिँदैन।

💡 **सावधानी:**

त्यस्ता commits मात्र amend गर, जो अझै **`local`** मा छन् र कतै **`push`** गरिएको छैन।

पहिले नै **`push`** भएका **`commits`** लाई **`amend`** गरेर branch लाई force push गर्दा, तिम्रा सहकार्यकर्ताहरू (**`collaborators`**) का लागि समस्या निम्त्याउँछ।

Staged File लाई Unstage गर्नु --

अर्का दुई खण्डहरूले तिमीलाई staging area र working directory को परिवर्तनहरूमा कसरी काम गर्ने भनेर देखाउँछन्। यसमा राम्रो कुरा के छ भने, तिमीले यी दुई क्षेत्रहरूको अवस्था पत्ता लगाउन प्रयोग गर्ने command ले तिमीलाई तिनलाई undo गर्ने तरिका पनि सम्झाउँछ।

उदाहरणका लागि, मानौं तिमीले दुई files मा परिवर्तन गर्यौं र ती दुई files लाई अलग-अलग commit गर्न चाहन्छौ। तर गल्तीले git add * टाइप गर्यौं र ती दुवै files लाई stage गर्यौं। अब, कसरी तीमध्ये एक file लाई unstage गर्ने? यसमा git status command तिमीलाई सम्झाउँछ:

```
$ git add *
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
  renamed: README.md -> README
  modified: CONTRIBUTING.md
```

"Changes to be committed" भन्ने text मुनि, "use git reset HEAD <file> to unstage" भन्ने देखिन्छ। त्यसैले, अब CONTRIBUTING.md file लाई unstage गर्न यो निर्देशन प्रयोग गरौं:

```
$ git reset HEAD CONTRIBUTING.md
Unstaged changes after reset:
M CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
  renamed: README.md -> README
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
  modified: CONTRIBUTING.md
```

Command को व्याख्या

यो command अलि अनौठो लाग्न सक्छ, तर यो काम गर्छ। अब CONTRIBUTING.md file लाई modify गरिएको छ, तर यो unstage गरिएको अवस्थामा फिर्ता आएको छ।

💡 **महत्त्वपूर्ण कुरा:**

git reset command खतरनाक हुनसक्छ, विशेष गरी यदि तिमीले --hard flag प्रयोग गर्यौं भने।

तर माथि वर्णन गरिएको अवस्थामा, तिम्रो working directory मा रहेको file untouched (नछोइएको) हुन्छ, त्यसैले यो तुलनात्मक रूपमा सुरक्षित छ।

संशोधित फाइललाई Unmodify गर्ने --

मानौं, तिमीले सोध्यौ कि अब तिमीलाई CONTRIBUTING.md फाइलमा गरेको परिवर्तन राख्न मन छैन। कसरी यो फाइललाई सजिलै unmodify गरेर फिर्ता लैजाने? अर्थात्, यो फाइललाई तिमीले पछिल्लो पटक commit गरेको, clone गरेको, वा working directory मा ल्याएको अवस्थामा फिर्ता लैजाने? भाग्यवश, git status ले तिमीलाई यसबारे पनि सल्लाह दिन्छ।

पछिल्लो उदाहरणको output मा, unstaged क्षेत्र यसरी देखिन्छ:

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)
  modified: CONTRIBUTING.md
```

यसले स्पष्ट रूपमा तिमीलाई परिवर्तनहरू कसरी हटाउने भनेर बताएको छ। अब, हामी यस निर्देशानुसार काम गरौं:

```
$ git checkout -- CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
  renamed: README.md -> README
```

अब, तिमीले देख्न सक्छौ कि परिवर्तनहरू revert भइसकेका छन्।

💡 **महत्त्वपूर्ण कुरा:**

git checkout -- <file> command खतरनाक हो।

यो command चलाएपछि, local रूपमा गरेको सबै परिवर्तनहरू हराउँछन् – Git ले फाइललाई पछिल्लो staged वा committed version सँग बदलिदिन्छ।

कुनै पनि हालतमा यो command चलाउनु अघि सुनिश्चित गर कि तिमीलाई local परिवर्तनहरू आवश्यक छैनन्।

तर, यदि तिमीले त्यो फाइलमा गरेको परिवर्तन राख्न चाहन्छौ तर अहिलेको लागि यसलाई हटाउन मात्र चाहन्छौ भने, हामी Git Branching को भागमा stashing र branching को बारेमा छलफल गर्नेछौं। यी प्रायः उत्तम तरिकाहरू हुन्।

💡 **सम्झनुहोस्:**

Git मा committed कुनै पनि डाटा लगभग सधैं पुन: प्राप्त गर्न सकिन्छ।

यहाँसम्म कि delete गरिएको branches मा भएका commits वा --amend commit को रूपमा overwrite गरिएका commits पनि recovery गर्न सकिन्छ (Data Recovery भाग हेर्नुहोस्)।

तर, कहिल्यै commit नगरिएको डाटा गुमेपछि सम्भवतः फेरि देखिँदैन।

Git restore प्रयोग गरेर कुराहरू पूर्ववत गर्नुहोस् ---

Git संस्करण 2.23.0 ले नयाँ कमाण्ड: git restore लाई प्रस्तुत गर्यो। यो git reset को विकल्प हो, जुन हामीले पहिले नै सिकिसकेका छौं। Git संस्करण 2.23.0 बाट सुरु गर्दै, Git ले धेरै undo कार्यहरूको लागि git reset को सट्टा git restore प्रयोग गर्न थाल्यो।

हामीले अघि गरेका कदमलाई फेरि पछ्याऔं र git restore प्रयोग गरेर कुराहरू undo गरौं।

Git restore प्रयोग गरेर Staged फाइललाई Unstage गर्नुहोस्

git restore प्रयोग गरेर तपाईंको staging area र working directory को परिवर्तनहरूसँग कसरी काम गर्ने भन्ने कुरा सिकाउनका लागि यो खण्ड तयार गरिएको हो। यो प्रक्रिया सजिलो बनाउने अर्को फाइदा भनेको, यो कमाण्डले तपाईंलाई के-कसरी परिवर्तनलाई undo गर्ने भनेर संकेत गर्छ।

उदाहरणका लागि, मानौं तपाईंले दुई फाइल परिवर्तन गर्नुभएको छ र तिनलाई छुट्टाछुट्टै परिवर्तनको रूपमा commit गर्न चाहनुभएको छ। तर गल्तीले git add * लेखेर दुवै फाइललाई stage गरियो। अब, ती दुई मध्ये एउटालाई कसरी unstage गर्ने त? यसमा git status कमाण्डले सहयोग पुर्याउँछः

```
$ git add *
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  modified: CONTRIBUTING.md
  renamed: README.md -> README
```

“Changes to be committed” पाठको ठीक तल यो लेखिएको छ:

`use "git restore --staged <file>..." to unstage`। त्यसैले, CONTRIBUTING.md फाइललाई unstage गर्न यो सल्लाहको उपयोग गरौंः

```
$ git restore --staged CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  renamed: README.md -> README
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
  modified: CONTRIBUTING.md
```

अब, CONTRIBUTING.md फाइल modify भए पनि फेरि unstaged भयो।

Git Restore प्रयोग गरेर Modified फाइल Undo गर्ने ---

यदि तपाईंले महसुस गर्नुभयो कि तपाईंले CONTRIBUTING.md फाइलमा गरेका परिवर्तनहरू चाहिँदैनन्, तपाईं यसलाई सजिलै Undo गर्न सक्नुहुन्छ – यसलाई पछिल्लो पटक Commit गर्दा (वा Clone गर्दा, वा जसरी तपाईंको Working Directory मा आयो) जस्तो बनाउन सक्नुहुन्छ।

भाग्यवश, git status ले तपाईंलाई यसलाई कसरी गर्ने भनेर बताउँछ। पछिल्लो उदाहरणको Output मा, Unstaged क्षेत्र यसरी देखिन्छ:

```
Changes not staged for commit:
 (use "git add <file>..." to update what will be committed)
 (use "git restore <file>..." to discard changes in working directory)
 modified: CONTRIBUTING.md
```

यसले स्पष्ट रूपमा बताउँछ कि तपाईंले गरेका परिवर्तनहरू कसरी Discard गर्न सकिन्छ। आउनुस्, यसलाई आजमाऔं:

```
$ git restore CONTRIBUTING.md
$ git status
On branch master
Changes to be committed:
 (use "git restore --staged <file>..." to unstage)
 renamed: README.md -> README
```

💡 **महत्त्वपूर्ण कुरा:**

git restore <file> एउटा खतरनाक Command हो। तपाईंले फाइलमा गरेका Local परिवर्तनहरू सबै हराउनेछन् – Git ले फाइललाई अन्तिम Staged वा Committed Version सँग प्रतिस्थापन गर्नेछ।

जबसम्म तपाईं पक्का हुनुहुन्न कि तपाईंलाई ती Unsaved Local परिवर्तनहरूको आवश्यकता छैन, यो Command प्रयोग नगर्नुहोस्।

---------- END ----------
