Versionshantering med Git och GitHub

ZoCom

Vim

Om man inte skriver något commit-meddelande (git commit -m *"meddelande"*) så kommer Git att öppna en texteditor.

Git är gjort för att kunna användas helt och hållet från terminalen. Det finns en texteditor som heter Vim, som man använder direkt i terminalen. För att avsluta Vim, skriv:

`:q!` `+enter`

På datorer som har *Nano* kan man använda den i stället för *Vim*. Skriv:
`git config --global core.editor "nano -w"`

Ångra

Det finns ingen *git undo*. Man kan ångra sina misstag på flera sätt:

1. Ångra "git add" med `git reset HEAD` *`file`*
2. Ångra "git commit" med `git reset SHA-1` *`file`*
3. Ångra "git push" genom att göra en motsatt commit med `git revert`

Vi kan använda *git reset* för att backa till en tidigare revision.

Reset - för *lokal* ångra

Syntax: `git reset` *`target [file] [--hard]`*

Target är vilken revision man vill återvända till. Kan vara antingen HEAD eller revisionens SHA-1 kod.

```
$ git log --oneline
ce9a71b (HEAD -> master)
afc4fb6 Näst senaste revisionen
```

HEAD är en referens till den senaste revisionen. Om man återgår till den kommer allt som ligger löst i *working directory* och *staging area* att slängas.

Exempel git reset

`git reset HEAD`

Slänga alla ändringar. En "dirty" working directory blir "clean".

`git reset ce9a71b --hard`

Återvänd till revisionen med SHA-1 kod "ce9a71b". De commits man gjort efter den finns kvar tills man gör en ny commit.

Flaggan `--hard` talar om att vi vill återställa alla filer i working directory. Eventuella lokala ändringar kommer att tas bort. Det är en "farlig" operation eftersom man kan förlora arbete om man gör fel. Därför är den inte standard.

Revert - för *remote* ångra

Syntax: `git revert target [file]`

Target är vilken revision man vill "ångra".

HEAD är en referens till den senaste revisionen. Om man använder den så kommer git att skapa en ny commit som är motsatsen.

Om vi använder en tidigare revision så kommer git att skapa en ny commit som bara ångrar just den revisionen. Om man vill ångra flera revisioner så måste man göra *git revert* flera gånger!

Revert - exempel

Vi har gjort en commit med följande ändring:

```
+ function fun{} [ console.log('A fun function'); ]
```

Åh nej! Vi har använt fel sorters parenteser. Nu använder vi revert:

```
$ git revert HEAD
```

Git skapar en ny commit med följande ändring:

```
- function fun{} [ console.log('A fun function'); ]
```

Exempel git revert

`git revert HEAD`

Skapar en commit som gör motsatsen till den senaste committen.

`git revert HEAD --no-commit`

Lägger till ändringar i staging area, som motsvarar motsatsen till den senaste committen. Om vi skriver *git commit* efter detta så blir det samma som kommandot ovan.

Case study: push fail (en commit)

Zod har precis gjort en commit och push, för att ladda upp de senaste ändringarna till servern. Men han kommer på att han gjort ett fel.

För att skapa en motsats-commit skriver han:

```
$ git revert HEAD
```

(Gamla commit-meddelandet var *"Added file index.html"* så Zod skriver som nytt meddelande *"Revert: Added file index.html"*.)

För att ladda upp den nya revisionen till servern skriver han:

```
$ git push
```

Case study: push fail (flera commits)

Zod har gjort två commits i stället för en! Nu måste man skapa en commit som är motsats till flera revisioner.

```
$ git log -2 --oneline
aaaaaaa (HEAD -> master) Whoops 2
bbbbbbb Whoops 1

$ git revert aaaaaaa --no-commit
$ git revert bbbbbbb --no-commit
$ git commit -m "Revert: Whoops 2, Whoops 1"
$ git push
```

Öva på ångerfull git

1. Skapa ett nytt repository i en mapp. Gör en första commit där du lägger till minst en fil med lite textinnehåll.
2. Gör minst två nya commits med ändringar av filen. (git log för att kontrollera)
3. Använd *reset* för att gå tillbaka till en tidigare version av filen. Vad händer när du skriver *git status*? Blir det skillnad om du använder *--hard* med *git reset*?
4. Använd *reset* igen, för att återvända till din senaste commit.
5. Använd reset för att backa minst en commit och gör en ny commit. Vad händer med commit-historiken? (git log)
6. Se till att det finns minst en commit som vi ska ångra. I stället för *reset* ska du använda *revert* för att skapa en ny commit, som ångrar den gamla.

Om det blir fel, skapa ett nytt repo och börja om!