

Software life cycle management

Michael Gustavsson

michael.gustavsson@softtech-dev.se

Mål

- Versionshantering (local & remote)
- Continuous Deployment & Integration
- Molnlösningar
- Applikations design
- Optimering av webb applikationer
- Datasäkerhet & regler

Software Life Cycle Management (SLCM)?

- Kallas i bland för SLM eller ALM (Application Life Cycle Management).
- De metoder/processer, människor och verktyg som är inblandade i hanteringen av ett system ifrån idé till pensionering.

SLM

- Definierar 4 olika faser
 - Request (En begäran om ett system)
 - Provision (Inköp, installation samt licenshantering)
 - Manage (Underhåll, kontroll av systemens beteende och efterlevnad)
 - Change (Möjlighet att se över vilka licenser som behövs samt se över vilka programvaror som verkligen behövs och eventuellt avinstallera och få tillbaka licenser).

Versions- hantering

Versions hantering

- Git (Vårt fokus)
- SVN
- CVS
- TFS (Microsoft)
- ClearCase

Version Hantering (Remote)

- GitHub
- Microsoft DevOps

Git?

- Var är Git?
 - Git är ett version hanterings verktyg för lokal lagring av olika versioner av filer.
- Varför använda versions hantering ?
 - Varför borstar vi tänderna minst 2 gånger om dagen?
- Varför använda Git?
 - Enkelt
 - Gratis
 - Oerhört snabbt

Komma igång med Git

- Installera på Windows
 - Ladda ner git ifrån <https://git-scm.com/downloads>
 - Kör installationsprogrammet.
- Installera på Mac OS X
 - Enkelt, bara skriv `git --version` i terminal fönstret.
 - Om git inte är installerat kommer MacOS att starta installation av git.

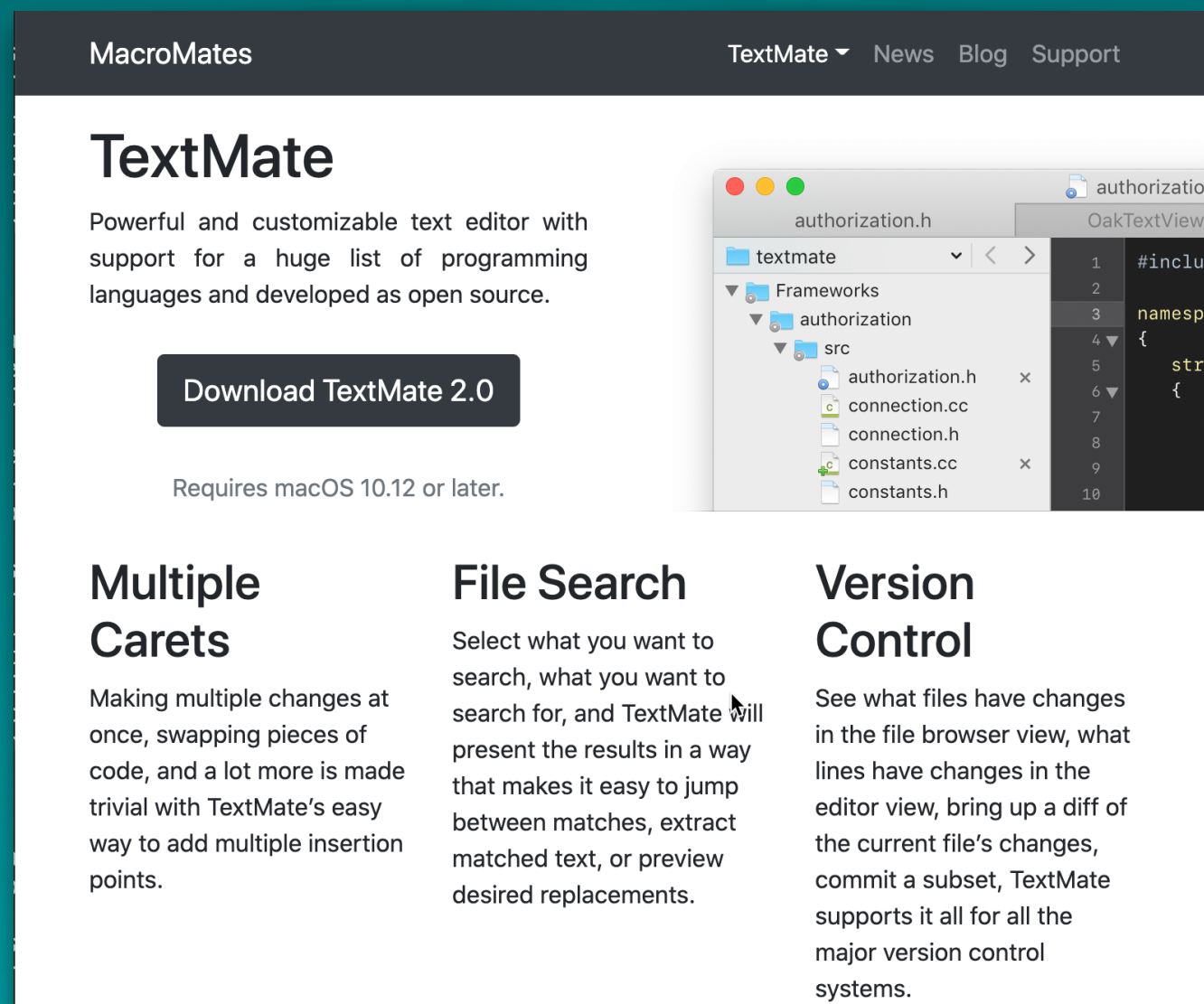
Konfigurera Git

```
git config --global user.name "Ditt namn"  
git config --global user.email "e-post adress"
```

Installera text editor

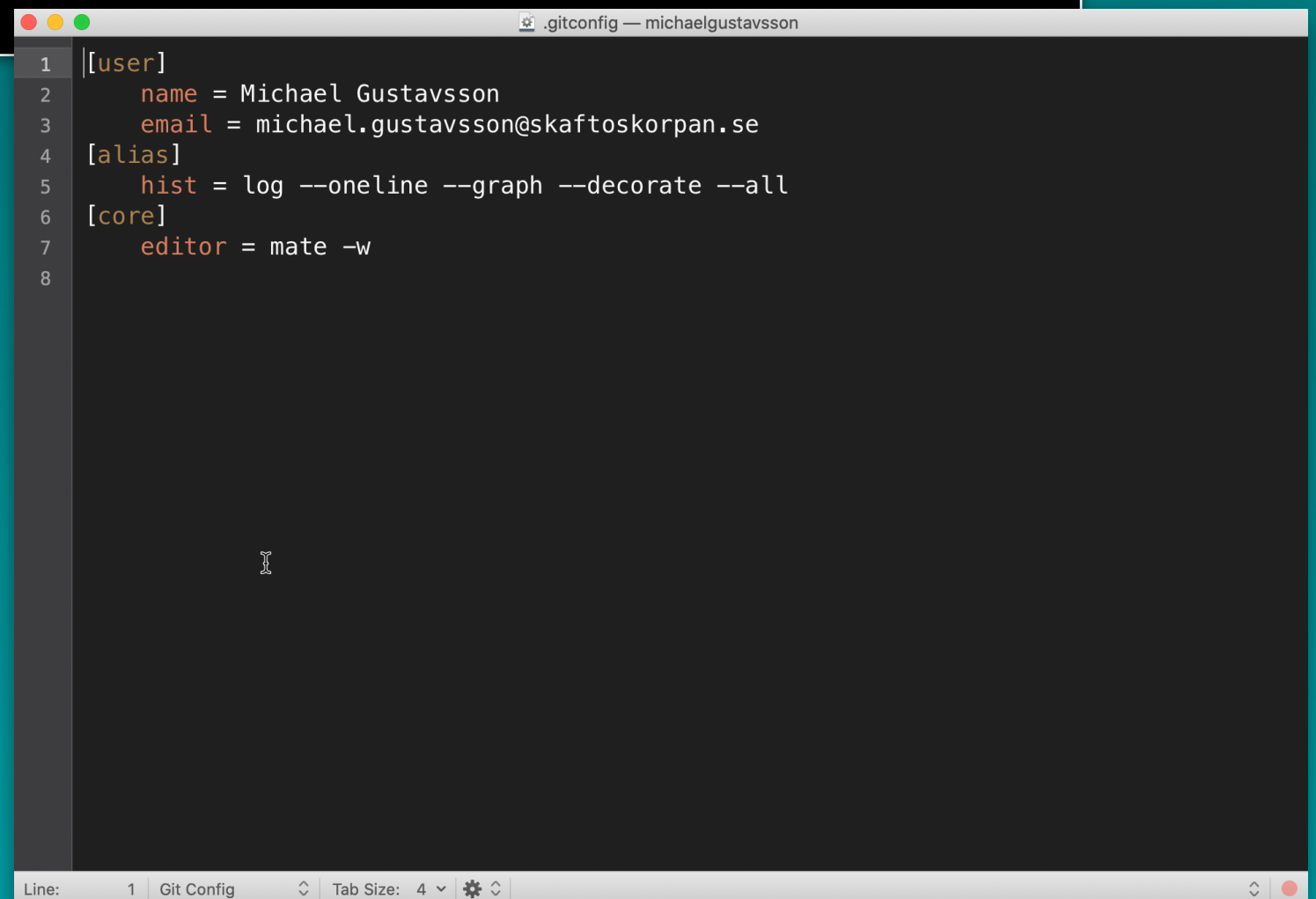
- För Mac OS X

- TextMate (<https://macromates.com/>)



Konfigurera TextMate

```
git config --global core.editor "mate -w"  
git config --global -e
```

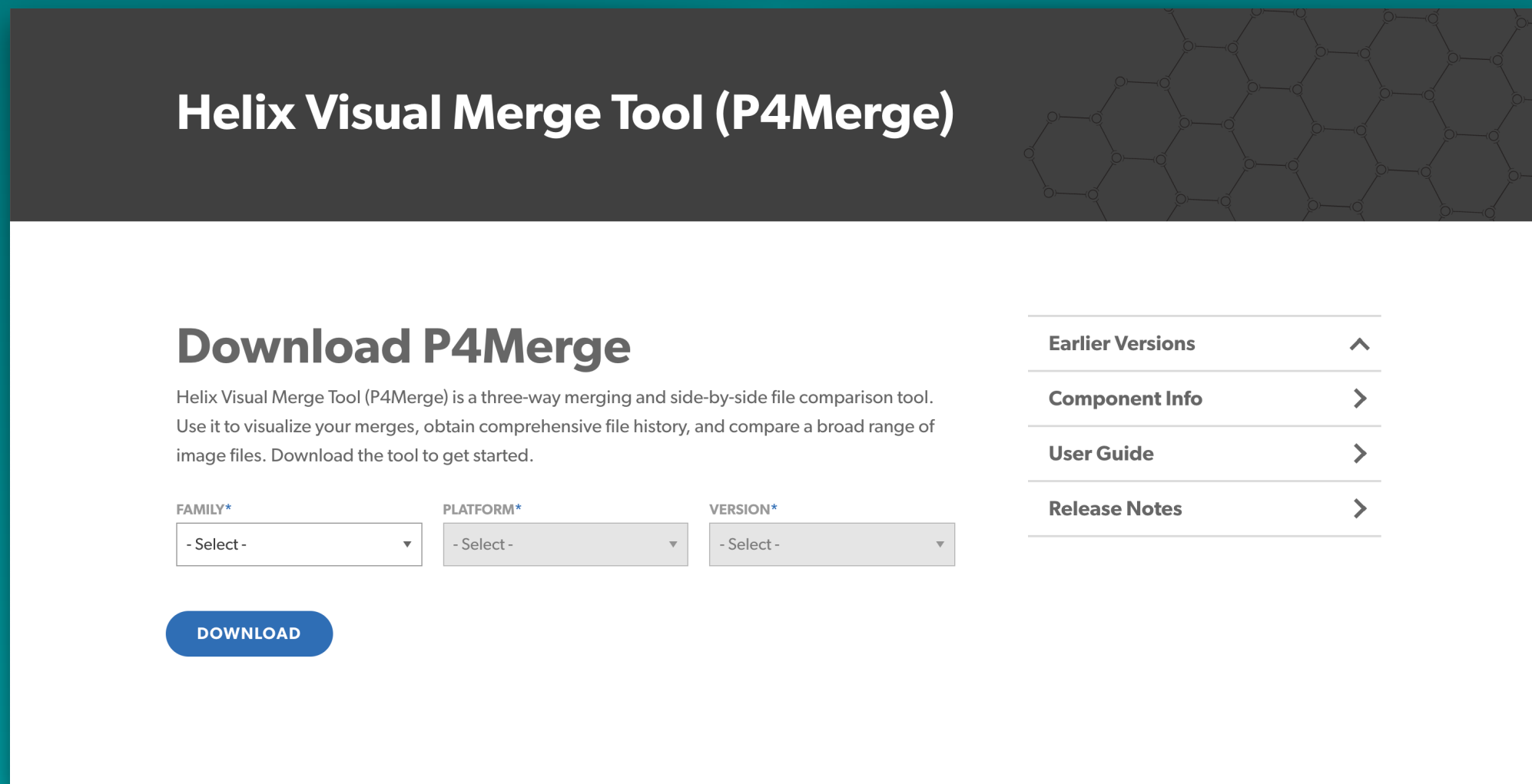
A screenshot of a text editor window titled ".gitconfig — michaelgustavsson". The editor shows the contents of a .gitconfig file. On the left, a line number column ranges from 1 to 8. The file content is as follows:
1 | [user]
2 | name = Michael Gustavsson
3 | email = michael.gustavsson@skaftoskorpan.se
4 | [alias]
5 | hist = log --oneline --graph --decorate --all
6 | [core]
7 | editor = mate -w
8 |
The status bar at the bottom indicates "Line: 1", "Git Config", "Tab Size: 4", and a settings icon.

Versionshanterings verktyg

- Git har inbyggda verktyg för hantering av versioner av kod
 - Tyvärr är det text baserat
- P4Merge Mac OS X är ett grafiskt verktyg för att kunna upptäcka skillnader i olika kod versioner.
- <https://www.perforce.com/downloads/visual-merge-tool>

P4Merge

- Gå till deras hemsida
- Registrera dig, sedan är det bara att ladda ner



Helix Visual Merge Tool (P4Merge)

Download P4Merge

Helix Visual Merge Tool (P4Merge) is a three-way merging and side-by-side file comparison tool. Use it to visualize your merges, obtain comprehensive file history, and compare a broad range of image files. Download the tool to get started.

FAMILY* PLATFORM* VERSION*

- Select - - Select - - Select -

DOWNLOAD

- Earlier Versions ^
- Component Info >
- User Guide >
- Release Notes >

Konfigurera P4Merge för Git

1.

```
git config --global diff.tool p4merge
```

2. På en rad.

```
git config --global difftool.p4merge.path "/Applications/p4merge.app/  
Contents/MacOS/p4merge"
```

3.

```
git config --global difftool.prompt false
```

Låt oss börja
använda
Git

Placera filer under git's ansvar

- Två sätt att placera filer under versions hantering
 - Använda *git init* utan argument
 - Placerar alla filer i aktuell katalog under version hantering
 - Använd *git init* kommandot med katalog namn
 - En ny katalog med angivet namn kommer att skapas och alla filer i den katalogen kommer att bli version hanterade.

```
demo — -zsh — 98x27
[→ projects mkdir demo
[→ projects cd demo
[→ demo git init
Initialized empty Git repository in /Users/michaelgustavsson/Documents/Dev/Courses/ITHS/Software-life-cycle-management/projects/demo/.git/
[→ demo git:(master) ls -al
total 0
drwxr-xr-x@ 3 michaelgustavsson  staff   96  9 0kt 21:52 .
drwxr-xr-x@ 4 michaelgustavsson  staff  128  9 0kt 21:51 ..
drwxr-xr-x@ 9 michaelgustavsson  staff  288  9 0kt 21:52 .git
[→ demo git:(master) |
```

```
demo — -zsh — 98x27
[→ projects git init demo
Initialized empty Git repository in /Users/michaelgustavsson/Documents/Dev/Courses/ITHS/Software-life-cycle-management/projects/demo/.git/

[→ projects cd demo

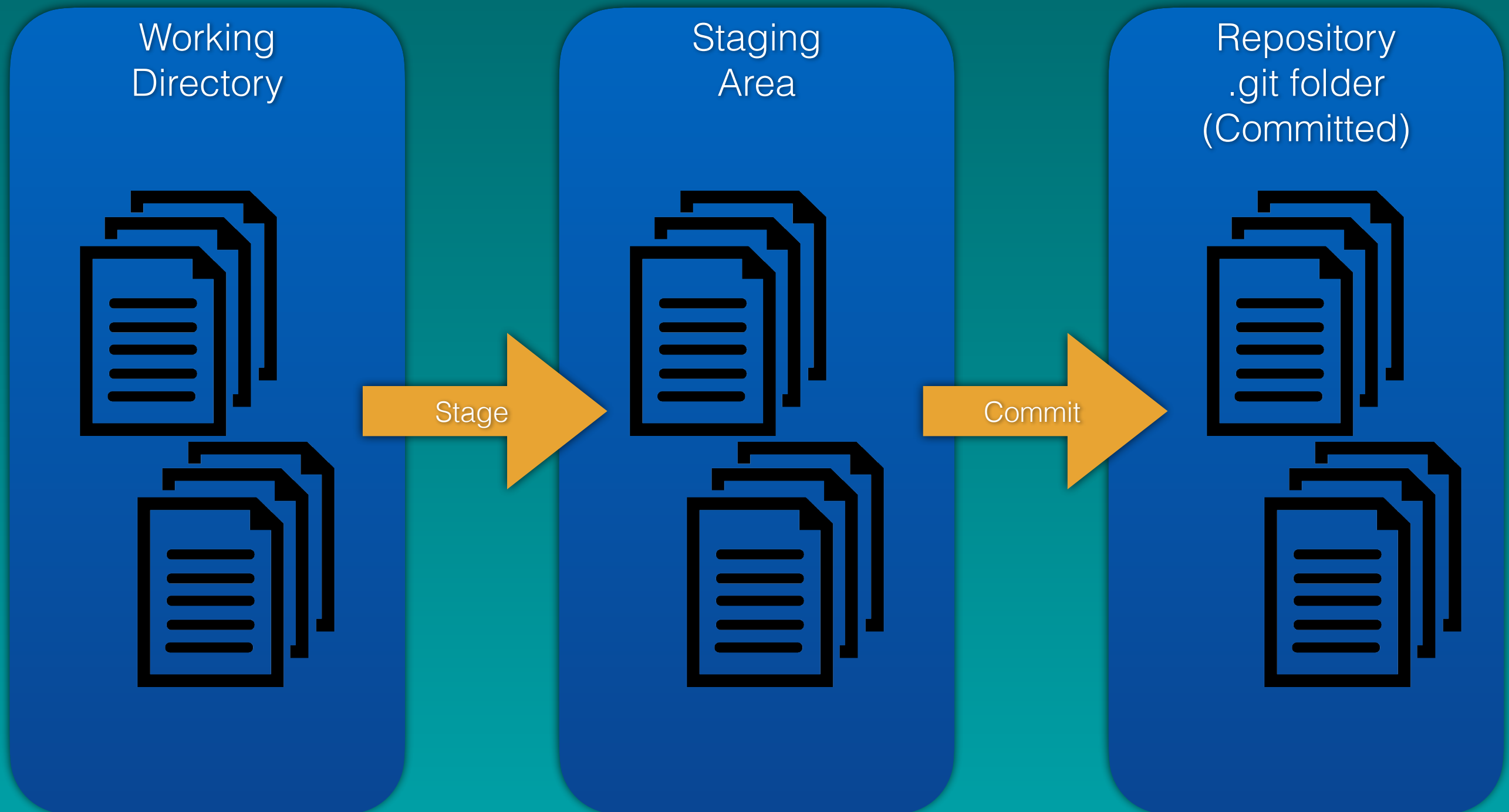
[→ demo git:(master) ls -al
total 0
drwxr-xr-x  3 michaelgustavsson  staff   96  9 0kt 21:54 .
drwxr-xr-x@ 4 michaelgustavsson  staff  128  9 0kt 21:54 ..
drwxr-xr-x  9 michaelgustavsson  staff  288  9 0kt 21:54 .git

→ demo git:(master) |
```

Grunderna

Git's tre tillstånd

Git States



Git States forts.

- Working directory
 - Våra filer direkt under katalogen
 - Dvs de filer som vi arbetar med.

Git States forts.

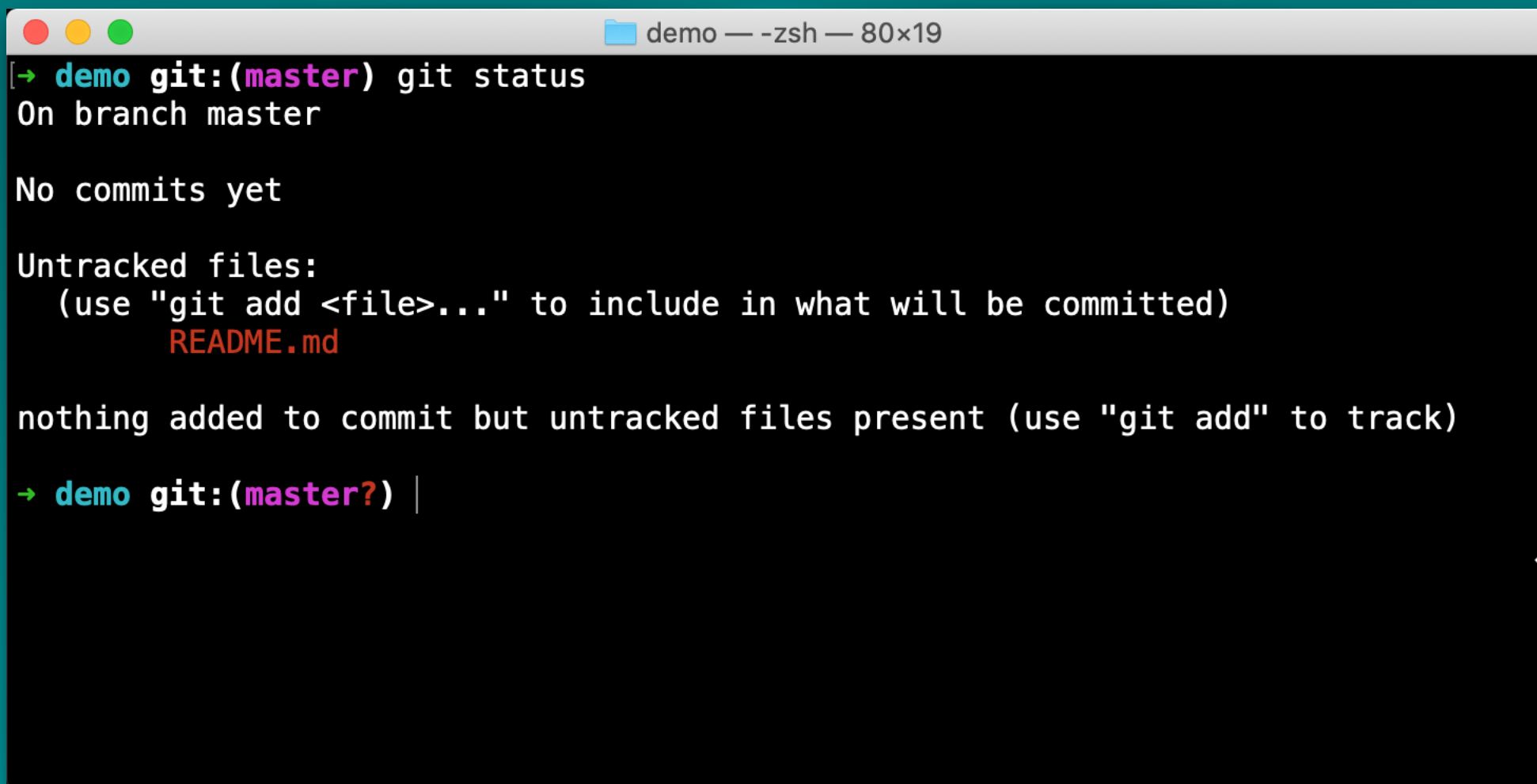
- Staging Area
 - Här hamnar filerna när vi skapar nya, ändrar eller raderar filer.

Git States forts.

- Git Repository (.git mappen)
- Här hamnar våra filer när vi har gjort en "commit" på vår katalog.

Git kommandon

- För att ta reda på status använd "git status"

A terminal window titled "demo — -zsh — 80x19" with a dark background and light-colored text. The window shows the output of the "git status" command. The output indicates that the user is on the master branch, there are no commits yet, and there is an untracked file named "README.md". It also provides instructions on how to use "git add" to track the file. The prompt shows the user has entered "git status" and is now at the "demo git:(master?)" prompt.

```
demo git:(master) git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
       README.md

nothing added to commit but untracked files present (use "git add" to track)
demo git:(master?) |
```

Flytta till Staging Area

- Använda git kommandot "add" för att flytta en fil eller filer ifrån "Working Directory" till "Staging Area".
 - Två version av kommandot kan användas:
 - `git add <filename>`
 - Flyttar namngiven fil ifrån working directory till Staging Area.
 - `Git .`
 - Flyttar alla filer ifrån working Directory till Staging Area.

```
demo — -zsh — 80x19
[→ demo git:(master+) git add README.md ]
[→ demo git:(master+) git status ]
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README.md

→ demo git:(master+) |
```

Flytta filer till Repository

- För att flytta en fil eller flera filer ifrån Staging Area till Repository använd följande kommandon:
- `git commit -m "Commit meddelande"`

```
demo — -zsh — 80x19
[→ demo git:(master+) git commit -m "My First Commit to the project"
[master (root-commit) f5e5317] My First Commit to the project
1 file changed, 3 insertions(+)
create mode 100644 README.md

[→ demo git:(master) git status
On branch master
nothing to commit, working tree clean

→ demo git:(master) |
```