Git & EGit beginner Workshop

Learn Git with Eclipse

EclipseCon France
5 June 2013

Matthias Sohn
SAP

Vincent Lalanne
Igor Laborie

AKKA Technologies
The team

Vincent Lalanne
Software Engineer at AKKA

Matthias Sohn
Product Owner at SAP

Igor Laborie
Software Architect at AKKA
Git enthusiast
The Raspberry Pi

- ARM processor 700 Mhz
- RAM: 512Mb
- HDMI, Composite
- Ethernet
- 2 USB port
- SD Card
- 26 Pins

Model A ~22 €
Model B ~33 €

http://elinux.org/RPi_Hub
Prepare to exercises

Raspberry Pi Wi-Fi: PiGit / EclipseCon

Or get EGit-Workshop.zip from an USB stick

=> Open index.html with a modern browser

You need an EGit version > v2.2.0

=> Install Eclipse Java Juno SR2 if needed.

Exercises are also available at http://ilaborie.github.io/Git--EGit-beginner-workshop
Agenda

- Introducing Git
- Working locally
- Branches
- Remotes
- Exam
  - => Win a Raspberry Pi B!
Introducing Git

Git, JGit and EGit
VCS

A Version Control System

- record changes
- retrieve a previous version
- provide collaborative works

Like SVN, CVS, ClearCase, ...
Central VCS Server

- Version Database
  - Version 3
  - Version 2
  - Version 1

Computer A
- Checkout
- File

Computer B
- Checkout
- File
DVCS

Computer A
- File
- Version Database
  - Version 3
  - Version 2
  - Version 1

Server Computer
- Version Database
  - Version 3
  - Version 2
  - Version 1

Computer B
- File
- Version Database
  - Version 3
  - Version 2
  - Version 1
Git

• easier offline usage
• easier to fork and merge (branches)
• speed
• scalability
• a lot of workflow
• ...

Git is developer-friendly.
Flexibility (with Workflow) is industry-friendly.
JGit & EGit

JGit is a (partial) implementation of Git
- lightweight,
- pure Java library,
- modular (OSGi-ready)

EGit is the Eclipse team provider for Git
- implemented on top of JGit
- Gerrit support
- Github support
Exercises 1 & 2

1 - Installation & Configuration
- http://192.168.42.1/exo1.html
- USB Stick: exo1.html

2 - Create Repositories
- http://192.168.42.1/exo2.html
- USB Stick: exo2.html
• Hierarchical configuration level:
  ○ system
  ○ user
  ○ project

• Authentication with Private/Public SSH keys
Working Locally

Basic operations
A commit just stores modifications (version) into the repository (local).

Be careful, make significant commits to provide a beautiful history.
The Staging Area / Index

- Intermediate zone between the working directory and the repository.
- Useful to prepare commit.
- Can be skipped.
File status into Git

● A file can be
  untracked - not managed by the repository
  tracked - managed by the repository

● A tracked file can be
  unmodified if = last commit
  modified if ≠ staging area
  staged if ≠ last commit & = staging area
Add

Never Tracked Files

Working Directory

Staging Area

Repository

add
Remove

- Untracked Files
- Working Directory
- Staging Area
- Repository

The process of 'remove' involves moving files from the Working Directory to the Repository, indicating the removal of untracked files from the Staging Area.
Ignore

- Never Tracked Files
- Working Directory
- Staging Area
- Repository

ignore
Commit

Never Tracked Files

Working Directory

Staging Area

Repository

commit
Exercices 3 & 4

3 - Basic Operations
- USB Stick: exo3.html

4 - Playing with Diffs (optional)
- USB Stick: exo4.html
Notes

- It's easy to modify the last commit (amend)
  But **Don't modify a shared commit**!

- Reset allows you to unstage (and more)
Branches

*The DAG*
Git Objects

First commit of my project

commit | size
---|---
container | 92ec2
author | Igor
committer | Igor
parents | ce23a1

commit | size
---|---
container | ce23a1

commit | size
---|---
container | ...

92ec2..

tree | size
---|---
container | 92ec2
blob | 5b1d3
blob | 911e7
blob | cba0a

blob | size
---|---
5b1d3.. | README
911e7.. | Eclipse Public License
cba0a.. | index.html

5b1d3..

98ca9..

92ec2..

911e7..

cea23a1..

...
The Directed Acyclic Graph

c0 → c1 → c2 → c3 → c4 → c5 → c6

/ \
c7 ↔ c8 ↔ c9 ↔ c10

\c11 ↔ c12
Branches

How many commits in feature A?
Create a new branch: feature_A
Create a new branch: feature_A
Create a new branch: `feature_A`
Implementing the feature A

c0 → c1 → c2 → feature_A

master
Implementing the feature A

```
c0  c1  c2  c3  c4

master

feature_A
```
Implementing the feature A
Checkout to master

c0 ← c1 ← c2 ← c3 ← c4 ← c5

master

feature_A
Checkout to \textit{master}
Working on *master*

Diagram:
- c0 → c1 → c2 → c3 → c4 → c5
- Feature A
Working on *master*
Working on *master*
Create a new branch: feature_B

- master
- feature_A
- c0 → c1 → c2 → c3
- c4 → c5 → c6 → c7
Create a new branch: feature_B
Implementing the feature B
Implementing the feature B
Implementing the feature B

Diagram:
- Master branch
  - Branch c6
    - Branch c7
      - Feature B (c9)
  - Branch c8
- Branch c3
  - Branch c4
    - Branch c5
  - Branch c2
    - Branch c1
      - Branch c0
Step back to *master*
Step back to *master*
Let's merge *master* with *feature_A*
Let's merge *master* with *feature_A*
Go to feature_B
Go to feature_B

feature_A

master

c0  c1  c2  c3

c4  c5  c6

c7

c8  c9  c10
Let's rebase on master
Let's rebase on *master*
Let's rebase on *master*
Let's rebase on master
Let's rebase on master
Merge vs Rebase
Exercices 5 & 6

5 - Branches Workflow
- [http://192.168.42.1/exo5.html](http://192.168.42.1/exo5.html)
- USB Stick: exo5.html

6 - Handle Conflicts (Optional)
- USB Stick: exo6.html
• Tag is a reference on a commit

• SHA-1 allow integrity check (fsck)

• Git rarely deletes data, reflog could save your life
Remotes

Sharing
A remote repositories is defined by

- an alias,
- an URL

  ssh://
  http://
  https://
  file://

  git://
Initially

Remote

c0

master

Local
A Local Commit

Remote

- c0
  - master

Local

- c0
- origin/master
- c1
  - master
A Remote Commit

Remote

$c_0 \rightarrow c_2$

(master)

Local

origin/master

$c_0 \rightarrow c_1$

(master)
Fetch

Remote

[Diagram showing the relationships between c0, c2, and master]

Local

[Diagram showing the relationships between c0, c1, c2, and origin/master]
Pull

Remote:
- c0
- c2
- master

Local:
- c0
- c1
- c2
- c3
- origin/master
- master
Push

Remote

Remote:
- c0
- c1
- c2
- c3

Local:
- c0
- c1
- c2
- c3

Branches:
- master
- origin/master
Exercices 7 & 8

7 - Clone Repository
● http://192.168.42.1/exo7.html
● USB Stick: exo7.html
● http://ilaborie.github.io/Git--EGit-beginner-workshop/exo7.html

8 - Remote Workflow
● http://192.168.42.1/exo8.html
● USB Stick: exo8.html
● http://ilaborie.github.io/Git--EGit-beginner-workshop/exo8.html
Notes

● Blame show commit details for each line of a file (Annotation)

● Branch tracking

● Upstream
Exam

http://git-quizz.herokuapp.com/
Thanks

Question(s)