

# EGit

Eclipse plug-in for git

Tomasz Zarna

Eclipse Platform Workspace committer  
IBM Poland

2009-11-28 / Eclipse DemoCamp



- 1 git
- 2 EGit
- 3 Links



- 1 git
- 2 EGit
- 3 Links



# What is git?

Characteristic, pros and cons

Git is a free distributed revision control



# What is git?

Characteristic, pros and cons

Git is a free distributed revision control

- distributed
- high performance
- reliable
- simple design



# What is git?

Characteristic, pros and cons

Git is a free distributed revision control

- distributed
- high performance
- reliable
- simple design
- difficult concepts
- no full control over the history
- bypassing official channels
- no dedicated versions at one time

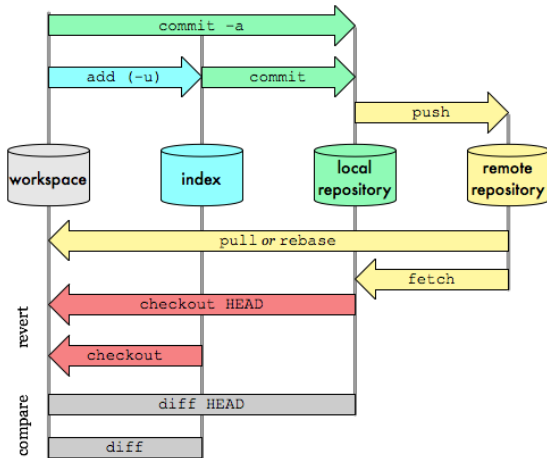


# Transport commands

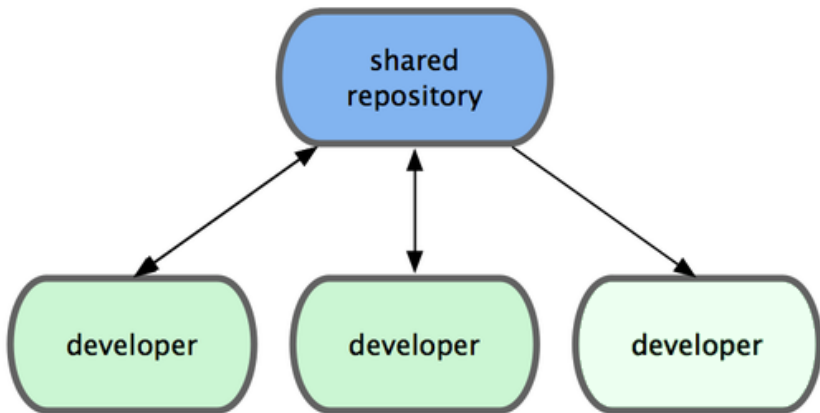
Add, push, pull, fetch... WTF?

## Git Data Transport Commands

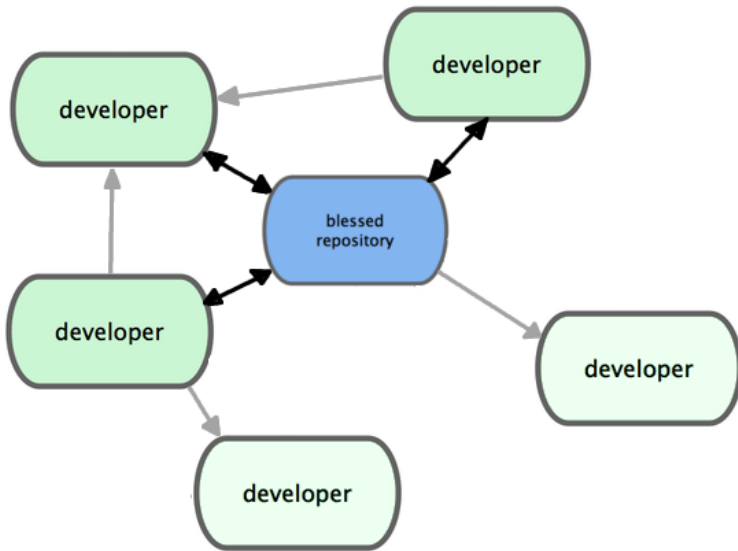
<http://osteele.com>



# Centralized

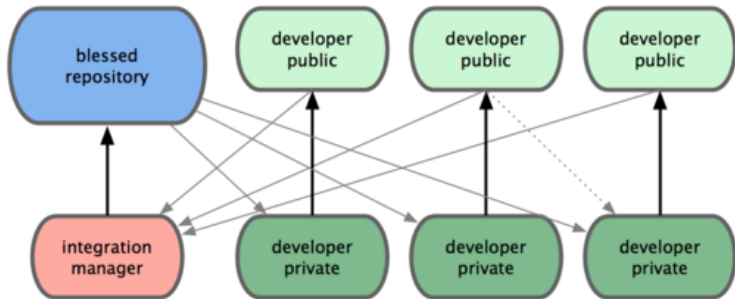






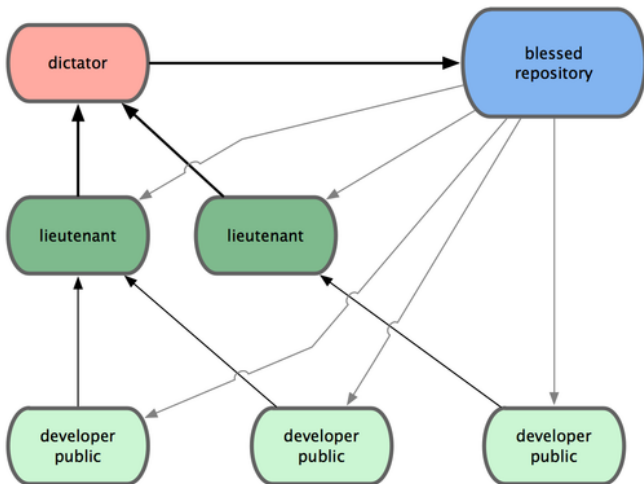
# Distributed

Integration manager



# Distributed

## Dictator and lieutenants



# git and git gui

Not only console

The screenshot shows the Git console interface. At the top, there's a commit history table with columns for commit hash, author, and date. Below the table, there's a search bar and a file browser showing the contents of `lib/git/git.rb`. The file content is displayed in a monospaced font, showing code for the `git` class.

Commit Hash	Author	Date
3099-01-30 17:49:40	Josh Blackius <josh@catnook.com>	2009-01-30 17:49:40
2008-08-26 04:48:51	Josh Goebel <dreamer3@gmail.com>	2008-08-26 04:48:51
2008-08-29 10:55:31	Scott Chacon <-s@chacoconsult.com>	2008-08-29 10:55:31
2008-08-26 10:09:45	Scott Chacon <-s@chacoconsult.com>	2008-08-26 10:09:45
2008-08-23 11:00:10	Hans Engel <engel@engel.uk.to>	2008-08-23 11:00:10
2008-08-23 09:51:15	Hans Engel <engel@engel.uk.to>	2008-08-23 09:51:15
2008-08-22 14:51:56	Scott Chacon <-s@chacoconsult.com>	2008-08-22 14:51:56
2008-08-11 10:05:57	Scott Chacon <-s@chacoconsult.com>	2008-08-11 10:05:57
2008-08-10 15:04:59	Scott Chacon <-s@chacoconsult.com>	2008-08-10 15:04:59
2008-08-07 15:35:31	Scott Chacon <-s@chacoconsult.com>	2008-08-07 15:35:31
2008-08-06 11:46:51	Scott Chacon <-s@chacoconsult.com>	2008-08-06 11:46:51
2008-07-31 13:47:19	Scott Chacon <-s@chacoconsult.com>	2008-07-31 13:47:19
2008-07-31 12:52:25	Byron Kaylor <-byronk@worldnet.att.net>	2008-07-31 12:52:25
2008-07-29 19:39:49	Byron Kaylor <-byronk@worldnet.att.net>	2008-07-29 19:39:49

```
Make the number of bytes to read from $stdout configurable.
Signed-off-by: Scott Chacon <-s@chacoconsult.com>

lib/git/git.rb
index f417d5e..f638d5 100644
--- lib/git/git.rb
+++ lib/git/git.rb
@@ -22,11 +22,19 @@ module Git
   include Enumerable

   class << self
     attr_accessor :git_binary, :git_timeout
     attr_accessor :git_binary, :git_timeout, :git_max_size
     end

     self.git_binary = "usr/bin/git"
```

The screenshot shows the Git GUI interface. It displays the repository path, the current branch (master), and a list of staged changes for `src/document.tex`. The changes are previewed in a monospaced font. At the bottom, there are options for the commit message and actions like Rescan, Stage Changed, Sign Off, Commit, and Push.

Repository: `C:\workspace\private\com.ibm.demos\...`

Current branch: master

File: src/document.tex
@@ -12,11 +12,19 @@
%end{center}
%end{block}
%\begin{multicols}{2}
%
%\begin{itemize}
%\item high performance
%\item distributed
%\item reliable: SHA1 hash
%\item simple design
%\end{itemize}
%\begin{itemize}
%\item simple design: blob, tree, commit, tag
%\end{itemize}
%\begin{itemize}
%\item difficult concepts
%\end{itemize}
%\item no full control over the binary

Commit Message: `[New Commit]` `[Amend Last Commit]`  
Signed-off-by: unknown <name@.none>

Ready.



- Amarok
- Android
- Digg
- Fedora
- GIMP
- GNOME
- GTK
- Linux kernel
- Perl
- Qt
- Ruby on Rails
- Samba
- VLC
- Wine
- ...



- 1 git
- 2 EGit
- 3 Links



## JGit

a pure Java implementation of Git's internals

## EGit

git plug-in for Eclipse



The screenshot shows the Eclipse IDE interface with the EGit Commit Changes dialog box open. The dialog has a "Commit Message:" field, "Author:" and "Committer:" fields, and checkboxes for "Amend previous commit" and "Add Signed-off-by". A table shows the status of files to be committed:

Status	File
<input checked="" type="checkbox"/> Modified	com.ibm.democamp.slides.2009.egit: src/document.tex
<input checked="" type="checkbox"/> Mod., not staged	com.ibm.democamp.slides.2009.egit: .gitignore

The Package Explorer on the right shows a project structure for "com.ibm.democamp.slides.2009.egit" with subfolders like bin, img, src, tmp, .cvsignore, .project, and .textlipse. The Git log at the bottom shows the commit history:

```
HEAD master iteration 2
● iteration 1
● Initial commit

commit 6835c81d24548bfa5dfe887ee7b4d8710698f1a6
Author: unknown <tzarna@.none> 2009-11-23 18:38:08
Committer: unknown <tzarna@.none> 2009-11-23 18:38:08
Parent: 8b61a4e90aacc5ebef5b38f0c29f1f2116e3f0cf (iteration 1)

iteration 2
```

A table on the right lists the changes in the commit:

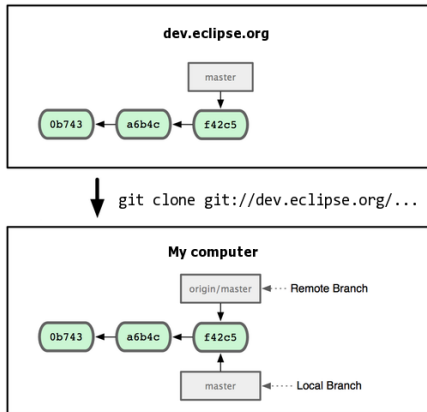
Path
A img/distribution.jpg
A img/git-github.png
M src/document.tex



# My scenario

## Working with Eclipse git repos

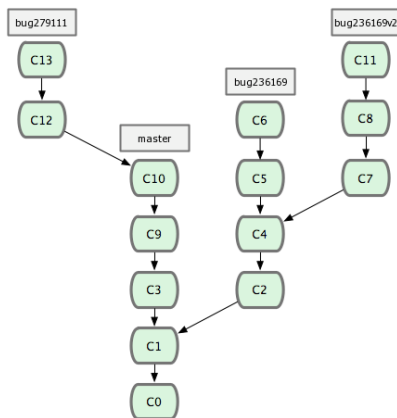
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

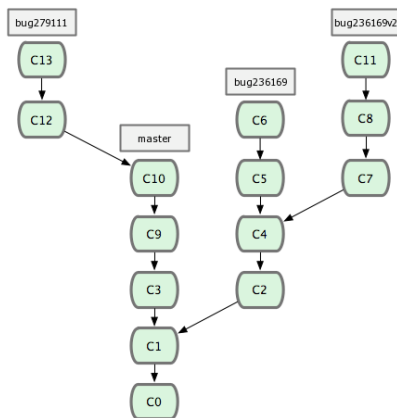
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

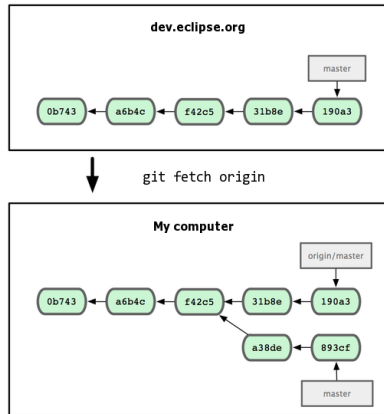
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

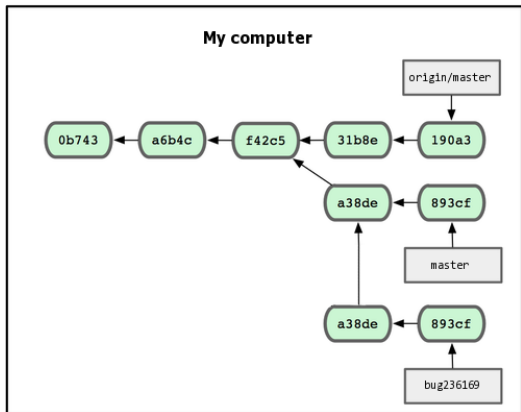
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

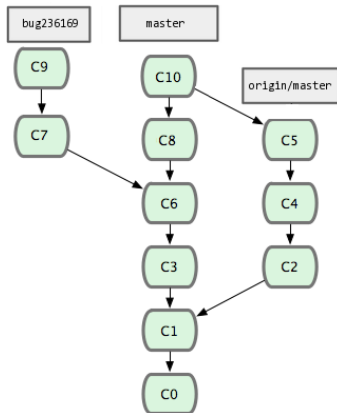
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

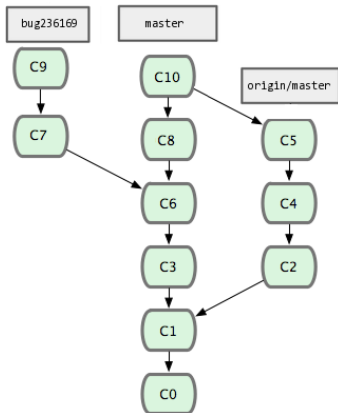
- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.



# My scenario

## Working with Eclipse git repos

- 1 clone an Eclipse git repo/project
- 2 branch
- 3 commit, commit, commit
- 4 fetch/pull origin
- 5 merge/rebase
- 6 create patch:  
format-patch diff
- 7 submit the patch
- 8 go to 2.





- 1 git
- 2 EGit
- 3 Links**



# Links

EGit links, git links

- Home page
- Update site
- Contribution guide
- Deprecate old VCS tools bug
- Eclipse git repositories



# Links

EGit links, git links

- Home page
- Update site
- Contribution guide
- Depracate old VCS tools bug
- Eclipse git repositories
- git home
- Pro Git
- msysgit, git on Windows
- GitHub
- git community book
- git ready



Thank you

Thank you

