Understanding and Using Git at Eclipse

http://eclipse.org/egit

Chris Aniszczyk (Red Hat)
zx@redhat.com
http://aniszczyk.org

Shawn Pearce (Google)
sop@google.com
http://spearce.org
Agenda

- Introduction
- How Git Works
- Demos
- Conclusion
- Q&A
Git and Eclipse

**Git** is a Distributed Version Control System (DVCS, GPL)

**EGit** is an Eclipse Team provider for Git (EPL).

**JGit** is a lightweight Java library implementing Git (EDL).

Technology sub projects (incubation)
- [http://www.eclipse.org/egit](http://www.eclipse.org/egit)
- [http://www.eclipse.org/jgit](http://www.eclipse.org/jgit)

The goal is to **build a community** around Git at Eclipse
EGit and JGit Diverse Committership

- Chris Aniszczyk (Red Hat)
- Christian Halstrick (SAP)
- Mik Kersten (Tasktop)
- Mathias Kinzler (SAP)
- Stefan Lay (SAP)
- Mykola Nikishov (Independent)
- Shawn Pearce (Google)
- Robin Rosenberg (Dewire)
- Matthias Sohn (SAP)
- Remy Suen (IBM)
- Gunnar Wagenknecht (Ageto)
Git at Eclipse

Eclipse is moving to Git...

http://git.eclipse.org

Eclipse.org Git repositories

To use Git in Eclipse, check out the EGit project.

This is a web interface for Eclipse Git repositories. You can also browse our CVS and SVN repositories. For more information about Git, please see the Git Wiki page.

The contents of the Git repositories are made available under the terms and conditions of the Eclipse.org Software User Agreement.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Owner</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>bpm2</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Antoine Toulme</td>
<td>20 hours</td>
</tr>
<tr>
<td>bpmnmodeler</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Antoine Toulme</td>
<td>10 days</td>
</tr>
<tr>
<td>emf/org.eclipse.emf.mwe.git</td>
<td>Gemini Web - OSGi web container implementation</td>
<td>root</td>
<td></td>
</tr>
<tr>
<td>gemini.web/org.eclipse.gemini.web.gemini-web-container.git</td>
<td>Grynex = Equinox on servers. Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Glyn Normington</td>
<td>13 hours</td>
</tr>
<tr>
<td>gynexmain.git</td>
<td>Gemini Web - OSGi web container implementation</td>
<td>Gunnar Wagenknecht</td>
<td>8 weeks</td>
</tr>
<tr>
<td>mpc/org.eclipse.mpc.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Steffen Pingel</td>
<td>5 weeks</td>
</tr>
<tr>
<td>mylyn/org.eclipse.mylyn.reviews.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Mario Bernhart</td>
<td>4 days</td>
</tr>
<tr>
<td>scalamodules/org.eclipse.scala.modules.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Heiko Seeberger</td>
<td>3 months</td>
</tr>
<tr>
<td>scalamodules/org.eclipse.scala modules.reposname.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Heiko Seeberger</td>
<td></td>
</tr>
<tr>
<td>sketch/org.eclipse.sketch.git</td>
<td>Eclipse Sketch project. Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Ugo Braga Sangiorgi</td>
<td>8 weeks</td>
</tr>
<tr>
<td>swtbot/org.eclipse.swtbot.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>Ketan Padegaonkar</td>
<td>5 months</td>
</tr>
<tr>
<td>tmmorg.eclipse.xtext.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>root</td>
<td>11 hours</td>
</tr>
<tr>
<td>tmf/org.eclipse.xtext.releng.git</td>
<td>Unnamed repository; edit this file <code>description</code> to name the repository.</td>
<td>root</td>
<td>11 hours</td>
</tr>
<tr>
<td>virgo/org.eclipse.virgo.apps.git</td>
<td>Virgo Applications - Virgo-supplied applications</td>
<td>Glyn Normington</td>
<td>6 days</td>
</tr>
</tbody>
</table>
History

2005  Linus Torvalds starts Git
2006  Proof-of-concept, quite unusable
2007  Index reader, quickdiff
2008  Add history view, commit, push/fetch
2009  Eclipse decides on Git; moved to Eclipse.org

3/2010  Released 0.7
  Diff/Merge Algorithms, Automatic IP Logs
6/2010  Released 0.8 (Helios)
  Usability Improvements, Git Repositories View, Tagging

9/2010  Planned 0.9 (Helios SR1)
  Merge, Synchronize View, Stashing, Staging

12/2010 Planned 1.0
Git, Eclipse and Rome

EGit is still beta (incubation by Eclipse terms) and we want to establish a feedback loop to improve the tooling

“Rome wasn’t built in a day”
Features in EGit 0.8

- **Supported**
  - git init / git clone
  - git add
  - git status
  - git commit
  - git diff
  - git fetch
  - git log
  - git merge
  - git remote

- **Partially supported**
  - git pull
  - git push

- **Not yet supported**
  - git stash
  - git branch
  - git tag
  - git checkout
  - git config
  - git format-patch
  - git mv / git rm
  - git reset

* planned for 0.9 in September ’10
No Free Lunch

The best way to learn Git is to use Git
Git vs. CVS/SVN

- **Distributed (git)**
  - Full local history
  - Work off-line
  - Fast
  - Rebase patches easily
  - Forks happen, deal with it
  - Powerful merging, allows many branches

- **Centralized (CVS)**
  - No
  - No
  - Slow
  - Patches go stale
  - Forks are painful
  - Merging is painful
Agenda

• Introduction
• How Git Works
• Demos
• Conclusion
• Q&A
How Git works?

Staging directory

Images taken from Pro Git book (http://progit.org)
How Git works?

Typical setup

Images taken from Pro Git book (http://progit.org)
How Git works?

Any workflow...

Images taken from Pro Git book (http://progit.org)
How Git works?
Saves state not deltas.

Images taken from Pro Git book (http://progit.org)
How Git works?

Objects

Images taken from Pro Git book (http://progit.org)
How Git works?

Branches and tags

Images taken from Pro Git book (http://progit.org)
How Git works?

Structure of .git

- **extensions**
- **global exclude file**
- **record branch changes**
- **object storage**
- **local branches**
  - default main branch
  - remote tracking branches, divided by repository
    - default remote repository
    - default remote branch
- **configuration for this repository**
- **tip of current branch**
- **cache for staging changes for next commit**
Merging

Merges are just the weaving together of two (or more) local branches into one

However, unlike CVCS, you don't have to specify anything about where you're merging from and to; the trees automatically know what their split point was in the past, and can work it out from there.

Merging is much easier in a DVCS like Git
Agenda

- Introduction
- How Git Works
- **Demos**
- Conclusion
- Q&A
Demo
A tour of EGit and Gerrit Code Review
“How the EGit and JGit teams work”
Agenda

• Introduction
• How Git Works
• Demos
• **Conclusion**
• Q&A
Conclusion

DVCS like Git are powerful

Git supports convenient branching and merging
Git is very fast and scales well
Gerrit enables a nice code review workflow

Git is the future SCM of Eclipse
Resources

Ask questions on the EGit/JGit forums

http://git-scm.com/documentation is your friend

If you want comedy, watch Linus' talk at Google:
http://www.youtube.com/watch?v=4XpnKHJAAok8

Read Pro Git: http://progit.org/book/

http://whygitisbetterthanx.com/

Read the EGit User Guide
Thanks!

Q & A