Saros Developer Documentation

COLLABORATORS			
	TITLE : Saros Developer Documentation		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		September 3, 2012	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Saros is using the Git version control system to manage the sourcecode, Gerrit for code review and Jenkins for Continuous Integration. We assume you want to help to improve Saros and contribute your modifications. This little guide should help you to get started, setup your environment, install Eclipse plugins and make your first modification and put it up for review. This guide provides the information in both text and pictures, pick the one you prefer.

Usage of Images
You might have eyed at the number of pages of this guide, don't be afraid. We are using a lot of pictures through out this guide and they take a lot of space.

1.2 About this manual

This manual is created with DocBook. The sources are managed with Git and are available from the same server as the main Saros plugin (*git clone* from *http://saros-build.imp.fu-berlin.de/gerrit/saros-developer-guide*), the review is done using Gerrit and Jenkins is used to automatically update the website.

Chapter 2

Rules

2.1 Code Rules

We have some simple rules for coding convention and commit messages. The current version can be found here. Please make sure the Saros Formatter is enabled for the Saros projects.

2.2 Review Rules

Every change needs to be reviewed. This is for catching errors before they enter the tree and for allowing your peers to learn and discover the code through review. As a courtsey to others, test your changes before asking them to be reviewed. We have unit tests and subset of end-to-end tests that can be executed quickly.

Chapter 3

Installing and Configuring Plugins

3.1 Installing the SVN Plugin

Saros can integrate with the SVN Team support and needs to have access to the plugin when compiling and being executed. We are using version 1.6.x of the Subclipse plugin.

3.1.1 Installation of Subclipse (textual)

1. OPEN THE NEW SOFTWARE DIALOG

   Help → Install New Software

2. PRESS THE ADD BUTTON TO ADD A SITE

   Add to add the site.

3. ADD SUBCLIPSE REPOSITORY

   Pick a name for the Name field, enter the following http://subclipse.tigris.org/update_1.6.x into the Location field and press Ok.

4. SELECT SUBCLIPSE

   Select the Subclipse group and press Next.

5. REVIEW THE INSTALLATION DETAILS

   Review the installation details and continue by pressing Next.

6. ACCEPT LICENSES

   Review and accept the license and continue by pressing Finish. This will start the installation

7. INSTALLATION PROCESS

   You will be presented the progress of the installation

8. UNSIGNED PACKAGES

   The packages are not cryptographically signed, nothing we can change right now.

9. RESTART ECLIPSE

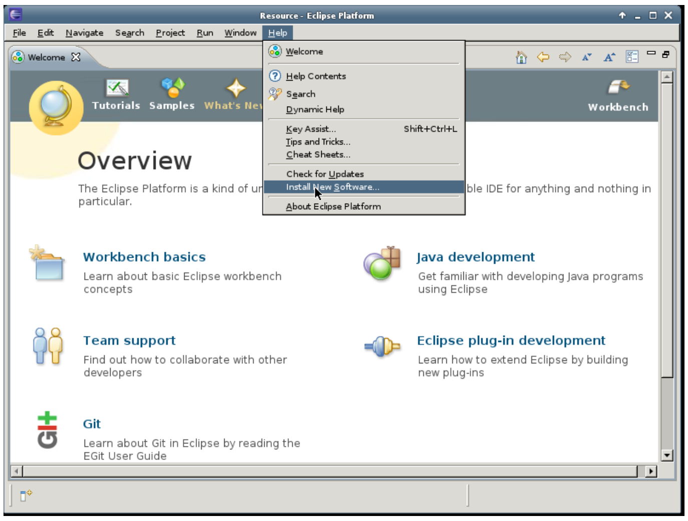
   You will be asked to restart Eclipse at the end of the installation.

10. FEEDBACK

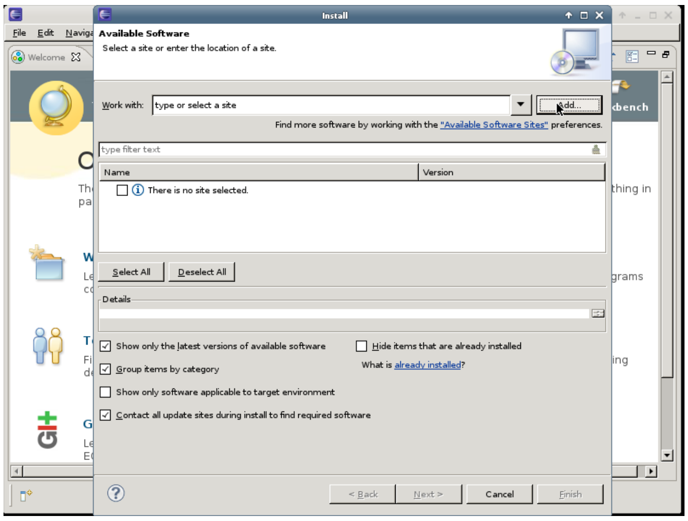
    You are asked if you want to participate in providing feedback, the choice is yours.

3.1.2 Installation of Subclipse (graphical)

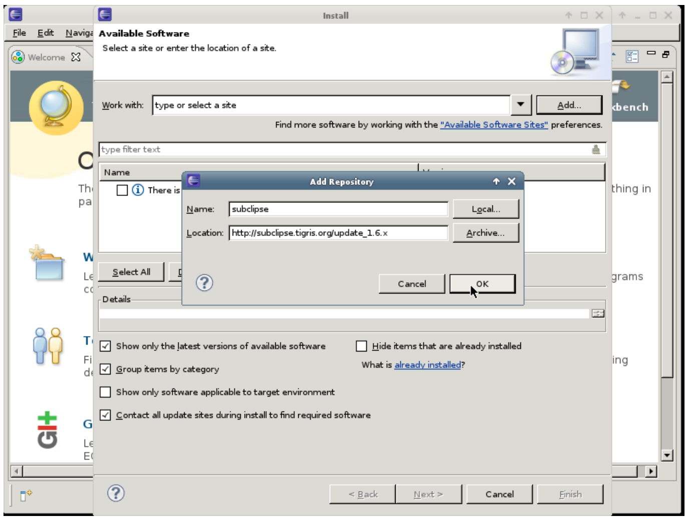
1. Open the New Software Dialog



2. Press the Add Button to add a site


Install
Available Software
Select a site or enter the location of a site.
Work with:
type or select a site
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
There is no site selected.
Select All
Deselect All
Details
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back
Next >
Cancel
Finish

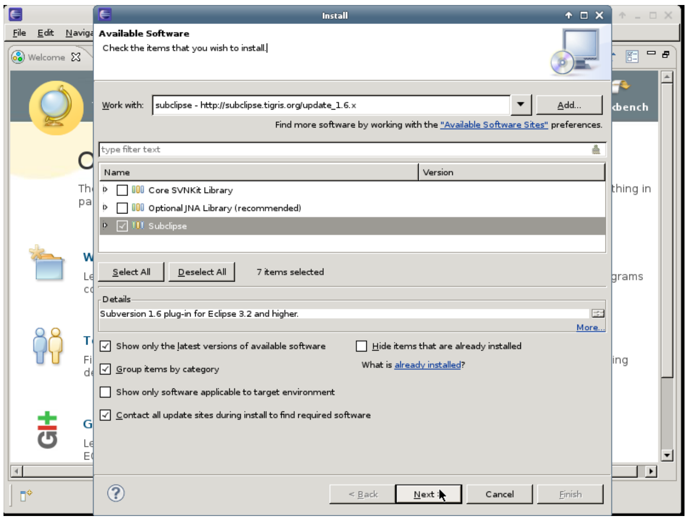
3. Add Subclipse repository



Name: subclipse

Location: http://subclipse.tigris.org/update_1.6.x

4. Select Subclipse


Available Software

Check the items that you wish to install

Work with: subclipse - http://subclipse.tigris.org/update_1.6.x

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☐ Core SVNKit Library	
☐ Optional JNA Library (recommended)	
☑ Subclipse	

Select All   Deselect All   7 items selected

Details

Subversion 1.6 plug-in for Eclipse 3.2 and higher.

☑ Show only the latest versions of available software

☐ Hide items that are already installed

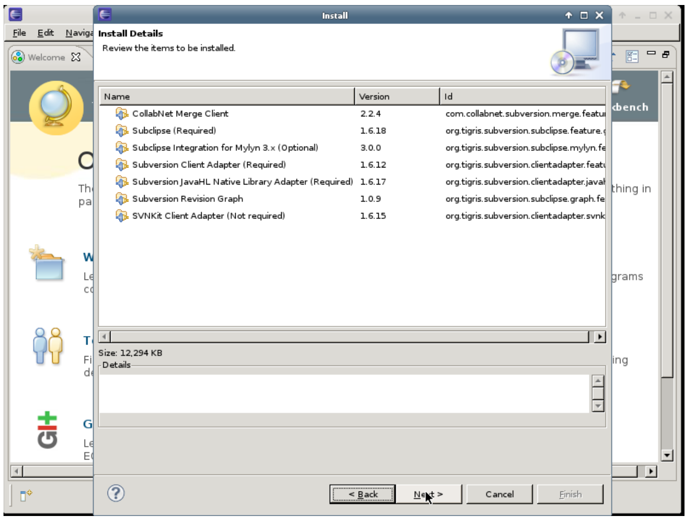
☑ Group items by category

What is already installed?

☐ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

5. REVIEW THE INSTALLATION DETAILS


Install Details

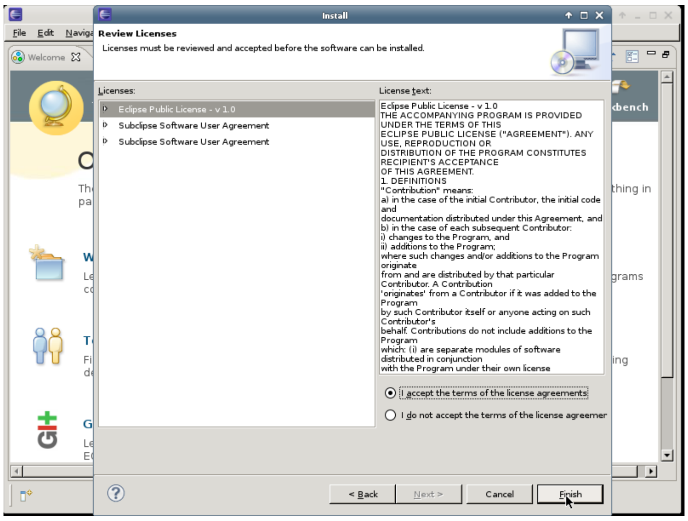
Review the items to be installed.

Name	Version	Id
CollabNet Merge Client	2.2.4	com.collabnet.subversion.merge.featu
Subclipse (Required)	1.6.18	org.tigris.subversion.subclipse.feature.
Subclipse Integration for Mylyn 3.x (Optional)	3.0.0	org.tigris.subversion.subclipse.mylyn.fe
Subversion Client Adapter (Required)	1.6.12	org.tigris.subversion.clientadapter.featu
Subversion JavaHL Native Library Adapter (Required)	1.6.17	org.tigris.subversion.clientadapter.javal
Subversion Revision Graph	1.0.9	org.tigris.subversion.subclipse.graph.fe
SVNKit Client Adapter (Not required)	1.6.15	org.tigris.subversion.clientadapter.svnk

Size: 12,294 KB

Details

6. ACCEPT LICENSES


Review Licenses

Licenses must be reviewed and accepted before the software can be installed.

Licenses:

- Eclipse Public License - v 1.0
- Subclipse Software User Agreement
- Subclipse Software User Agreement

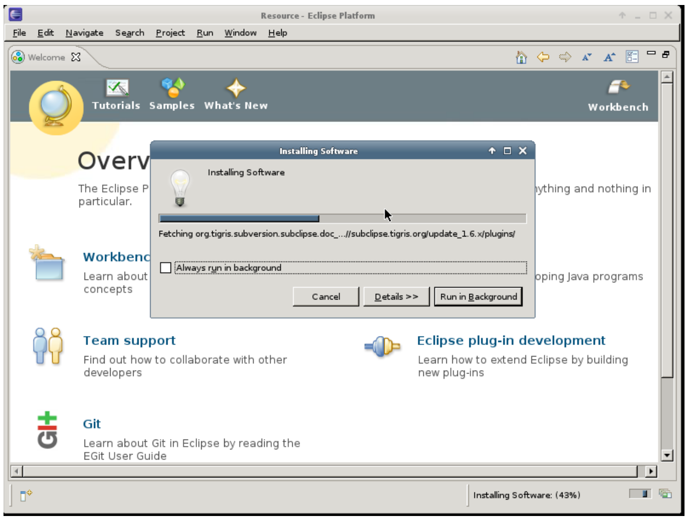
License text:

Eclipse Public License - v 1.0
THE ACCOMPANYING PROGRAM IS PROVIDED
UNDER THE TERMS OF THIS
ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY
USE, REPRODUCTION OR
DISTRIBUTION OF THE PROGRAM CONSTITUTES
RECIPIENT'S ACCEPTANCE
OF THIS AGREEMENT.
1. DEFINITIONS
"Contribution" means:
a) in the case of the initial Contributor, the initial code
and
documentation distributed under this Agreement, and
b) in the case of each subsequent Contributor:
i) changes to the Program, and
ii) additions to the Program;
where such changes and/or additions to the Program
originate
from and are distributed by that particular
Contributor. A Contribution
'originates' from a Contributor if it was added to the
Program
by such Contributor itself or anyone acting on such
Contributor's
behalf. Contributions do not include additions to the
Program
which: (i) are separate modules of software
distributed in conjunction
with the Program under their own license

[✓ I accept the terms of the license agreements]

I do not accept the terms of the license agreemer

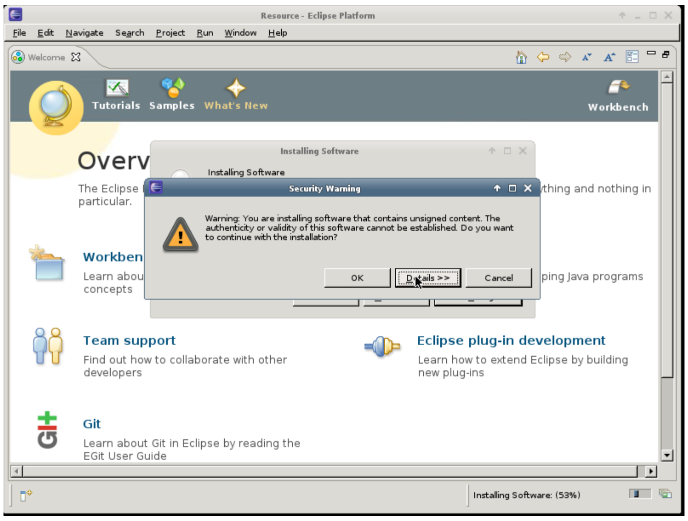
7. INSTALLATION PROCESS


Resource - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Welcome
Tutorials Samples What's New Workbench
Installing Software
Always run in background
Cancel
Details >>
Run in Background
Installing Software: (43%)

Installing Software

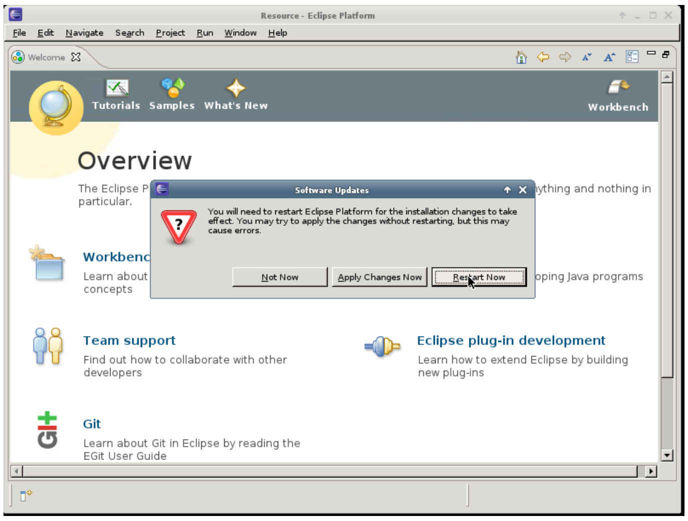
Fetching org.tigris.subversion.subclipse.doc_...//subclipse.tigris.org/update_1.6.x/plugins/

8. Unsigned packages


Resource - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Welcome
Tutorials Samples What's New Workbench
Installing Software
Security Warning
OK
Details >>
Cancel
Installing Software: (53%)

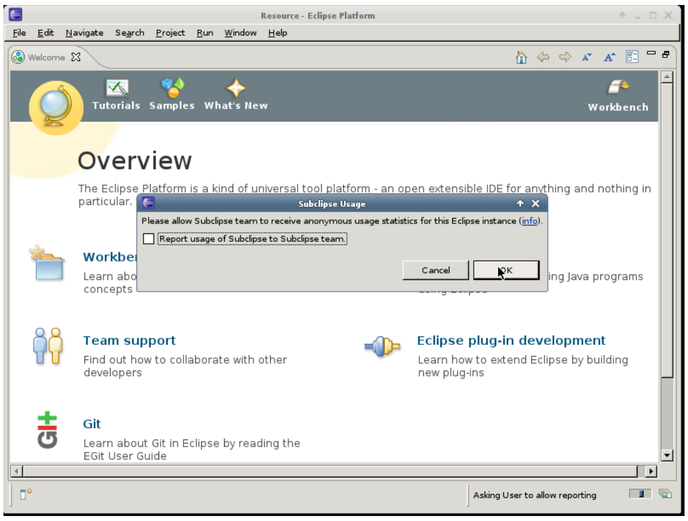
Warning: You are installing software that contains unsigned content. The authenticity or validity of this software cannot be established. Do you want to continue with the installation?

9. Restart Eclipse


Resource - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Welcome
Tutorials Samples What's New Workbench
Software Updates
Not Now
Apply Changes Now
Restart Now

You will need to restart Eclipse Platform for the installation changes to take effect. You may try to apply the changes without restarting, but this may cause errors.

10. Feedback


Resource - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Welcome
Tutorials
Samples
What's New
Workbench
Overview
Workbe
Team support
Eclipse plug-in development
Git
Subclipse Usage
Cancel
OK
Asking User to allow reporting

The Eclipse Platform is a kind of universal tool platform - an open extensible IDE for anything and nothing in particular.

Please allow Subclipse team to receive anonymous usage statistics for this Eclipse instance (info).

Report usage of Subclipse to Subclipse team.

Learn abo concepts

ing Java programs

Find out how to collaborate with other developers

Learn how to extend Eclipse by building new plug-ins

Learn about Git in Eclipse by reading the EGit User Guide

3.2 Installing the Git Plugin

The Saros Team is using Git to manage the source code. You will need to install the EGit plugin to get a copy of the Saros source code and start working on it.

3.2.1 Installation of EGit (textual)

1. Open the New Software Dialog

   Help → Install New Software

2. Press the Add Button to add a site

   Add to add the site.

3. Add EGit repository

   Pick a name for the Name field, enter the following http://download.eclipse.org/egit/updates into the Location field and press Ok.

4. Select EGit

   Select the *Eclipse Git Team Provider* and press Next button.

5. Review the Installation details

   Review the installation details and continue by pressing the Next.

6. Accept Licenses

   Review and accept the license and continue by pressing Finish. This will start the installation

7. Installation Process

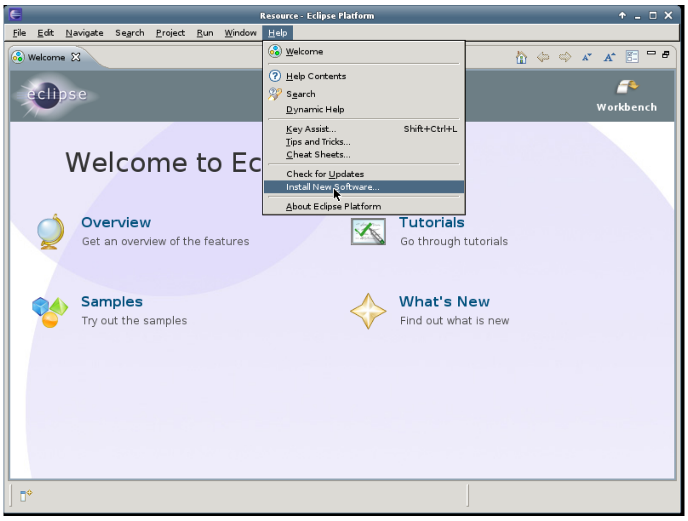
   You will be presented the progress of the installation

8. Restart Eclipse

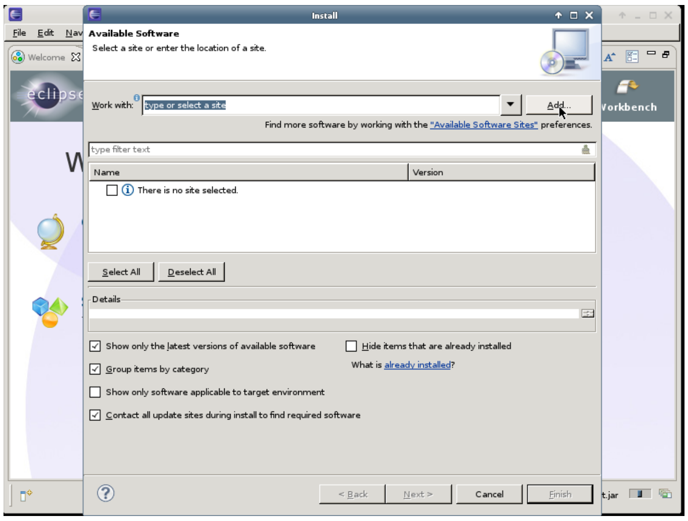
   You will be asked to restart Eclipse at the end of the installation.

3.2.2 Installation of EGit (graphical)

1. Open the New Software Dialog



2. Press the Add Button to add a site



Available Software

Select a site or enter the location of a site.

Work with: type or select a site

Find more software by working with the "Available Software Sites" preferences.

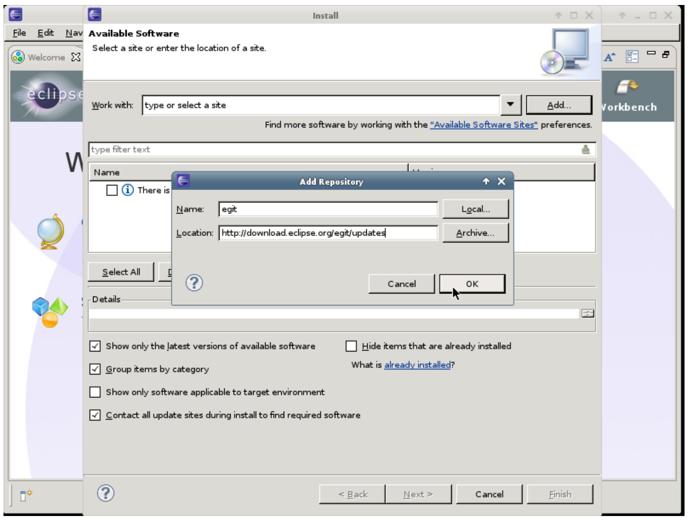
type filter text

Name	Version
There is no site selected.	

Details

- [✓] Show only the latest versions of available software
- [ ] Hide items that are already installed
- What is already installed?
- [✓] Group items by category
- [ ] Show only software applicable to target environment
- [✓] Contact all update sites during install to find required software

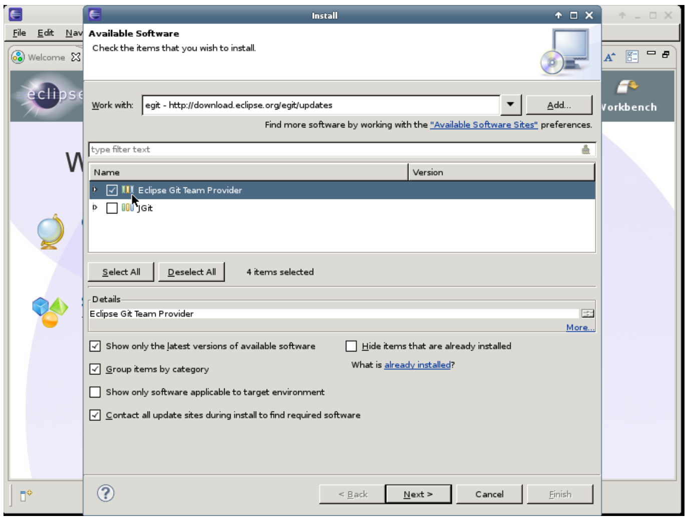
3. Add EGit repository



Name: egit

Location: http://download.eclipse.org/egit/updates

4. Select EGit



Available Software

Check the items that you wish to install.

Work with: egit - http://download.eclipse.org/egit/updates

Find more software by working with the "Available Software Sites" preferences.

Name	Version
[✓] Eclipse Git Team Provider	
[ ] JGit	

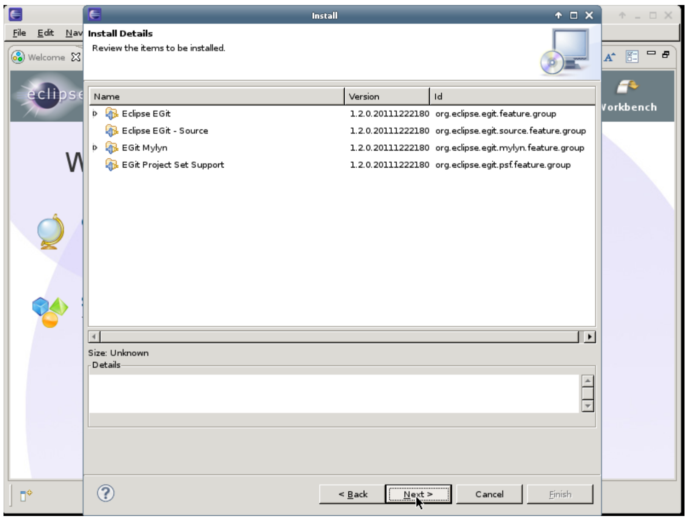
4 items selected

Details: Eclipse Git Team Provider

- [✓] Show only the latest versions of available software
- [ ] Hide items that are already installed
- [✓] Group items by category
- [ ] Show only software applicable to target environment
- [✓] Contact all update sites during install to find required software

What is already installed?

5. Review the Installation details



Install Details

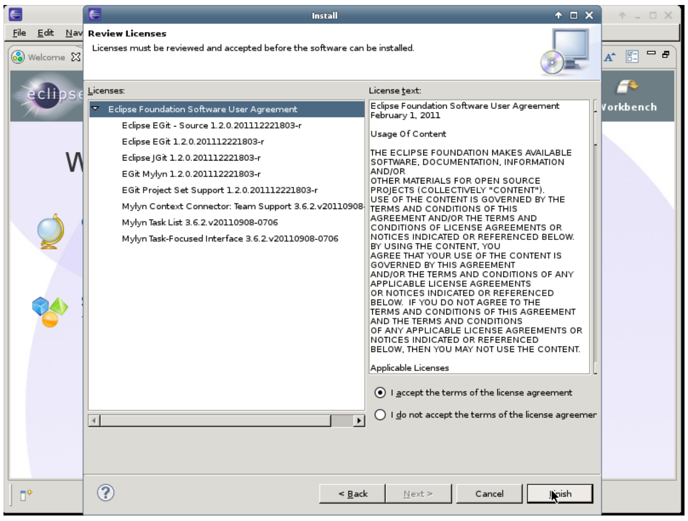
Review the items to be installed.

Name	Version	Id
Eclipse EGit	1.2.0.20111222180	org.eclipse.egit.feature.group
Eclipse EGit - Source	1.2.0.20111222180	org.eclipse.egit.source.feature.group
EGit Mylyn	1.2.0.20111222180	org.eclipse.egit.mylyn.feature.group
EGit Project Set Support	1.2.0.20111222180	org.eclipse.egit.psf.feature.group

Size: Unknown

Details

6. Accept Licenses



Review Licenses

Licenses must be reviewed and accepted before the software can be installed.

Licenses:

- Eclipse Foundation Software User Agreement
  - Eclipse EGit - Source 1.2.0.201112221803-r
  - Eclipse EGit 1.2.0.201112221803-r
  - Eclipse JGit 1.2.0.201112221803-r
  - EGit Mylyn 1.2.0.201112221803-r
  - EGit Project Set Support 1.2.0.201112221803-r
  - Mylyn Context Connector: Team Support 3.6.2.v20110908-
  - Mylyn Task List 3.6.2.v20110908-0706
  - Mylyn Task-Focused Interface 3.6.2.v20110908-0706

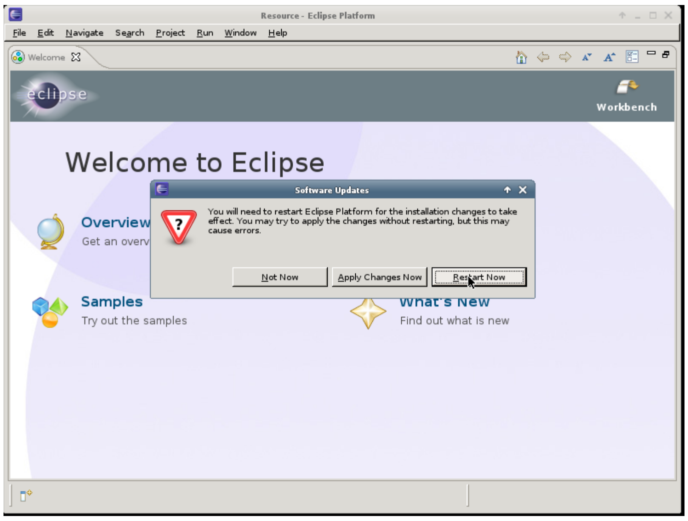
License text:

```
Eclipse Foundation Software User Agreement
February 1, 2011

Usage Of Content

THE ECLIPSE FOUNDATION MAKES AVAILABLE
SOFTWARE, DOCUMENTATION, INFORMATION
AND/OR
OTHER MATERIALS FOR OPEN SOURCE
PROJECTS (COLLECTIVELY "CONTENT").
USE OF THE CONTENT IS GOVERNED BY THE
TERMS AND CONDITIONS OF THIS
AGREEMENT AND/OR THE TERMS AND
CONDITIONS OF LICENSE AGREEMENTS OR
NOTICES INDICATED OR REFERENCED BELOW.
BY USING THE CONTENT, YOU
AGREE THAT YOUR USE OF THE CONTENT IS
GOVERNED BY THIS AGREEMENT
AND/OR THE TERMS AND CONDITIONS OF ANY
APPLICABLE LICENSE AGREEMENTS
OR NOTICES INDICATED OR REFERENCED
BELOW.  IF YOU DO NOT AGREE TO THE
TERMS AND CONDITIONS OF THIS AGREEMENT
AND THE TERMS AND CONDITIONS
OF ANY APPLICABLE LICENSE AGREEMENTS OR
NOTICES INDICATED OR REFERENCED
BELOW, THEN YOU MAY NOT USE THE CONTENT.

Applicable Licenses
```

[✓ I accept the terms of the license agreement]

I do not accept the terms of the license agreemer

7. Restart Eclipse



You will need to restart Eclipse Platform for the installation changes to take effect. You may try to apply the changes without restarting, but this may cause errors.

3.2.3 Known issues

3.2.3.1 Password for the private key is not accepted

This issue is described in Bug 326526 and is caused by EGit/JSCH not being able to read AES encrypted private keys. The available workarounds are to use 3DES encrypted keys, set the *GIT_SSH=/usr/bin/ssh* before launching Eclipse or upgrade JSCH to version 0.1.45.

3.3 Installing the JTourbus Plugin

JTourBus allows to describe aspects of the code as route. Each route has a numbered list of stops inside the code. The route can be browsed from within eclipse using this plugin. JTourBus is used in Saros to document the basics and the following will inform you on how to install the plugin.

3.3.1 Installation of JTourBus (textual)

1. OPEN THE NEW SOFTWARE DIALOG

   Help → Install New Software

2. PRESS THE ADD BUTTON TO ADD A SITE

   Add to add the site.

3. ADD JTOURBUS REPOSITORY

   Pick a name for the Name field, enter the following http://saros-build.imp.fu-berlin.de/update-jtourbus into the Location field and press Ok.

4. SELECT JTOURBUS

   Select *JTourBus* and press Next button.

5. REVIEW THE INSTALLATION DETAILS

   Review the installation details and continue by pressing the Next.

6. ACCEPT LICENSES

   Review and accept the license and continue by pressing Finish. This will start the installation

7. ACCEPT UNSIGNED CONTENT

   Press Ok to install unsigned content.

8. INSTALLATION PROCESS

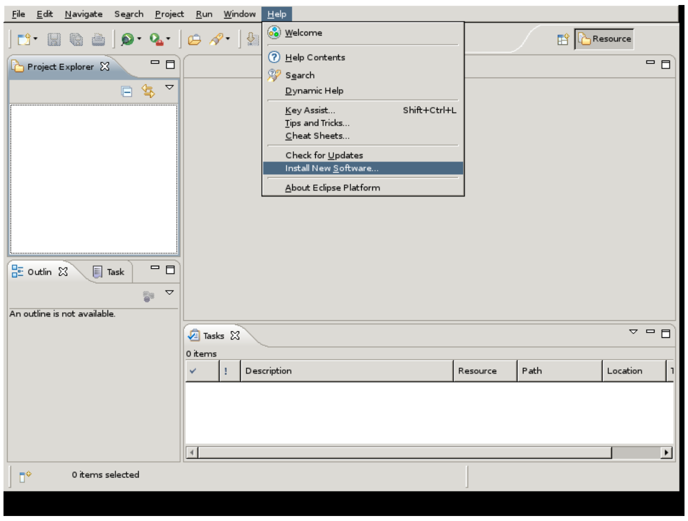
   You will be presented the progress of the installation

9. RESTART ECLIPSE

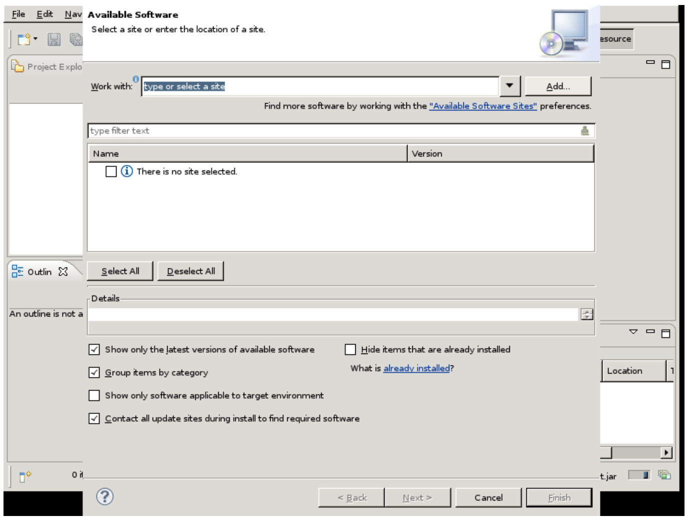
   You will be asked to restart Eclipse at the end of the installation.

3.3.2 Installation of JTourBus (graphical)

1. OPEN THE NEW SOFTWARE DIALOG



2. PRESS THE ADD BUTTON TO ADD A SITE


Available Software

Select a site or enter the location of a site.

Work with: type or select a site   Add...

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☐ There is no site selected.	

Select All   Deselect All

Details

☑ Show only the latest versions of available software ☐ Hide items that are already installed

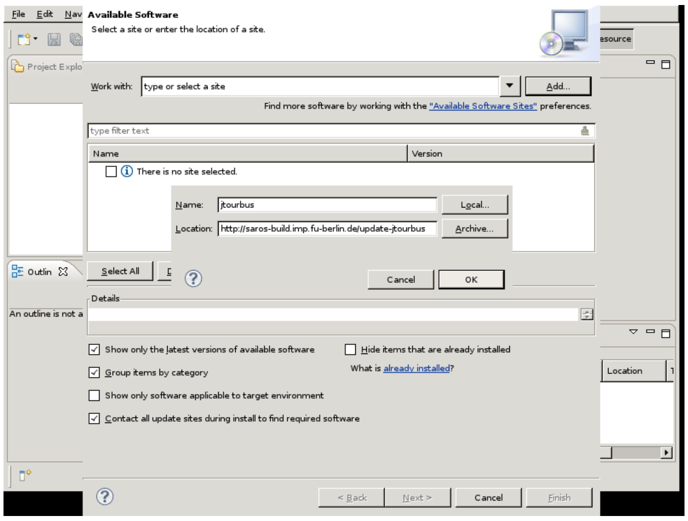
☑ Group items by category What is already installed?

☐ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

< Back   Next >   Cancel   Finish

3. ADD JTOURBUS REPOSITORY



Available Software

Select a site or enter the location of a site.

Work with: type or select a site [Add...]

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☐ ⓘ There is no site selected.	

Name: jtourbus [Local...]

Location: http://saros-build.imp.fu-berlin.de/update-jtourbus [Archive...]

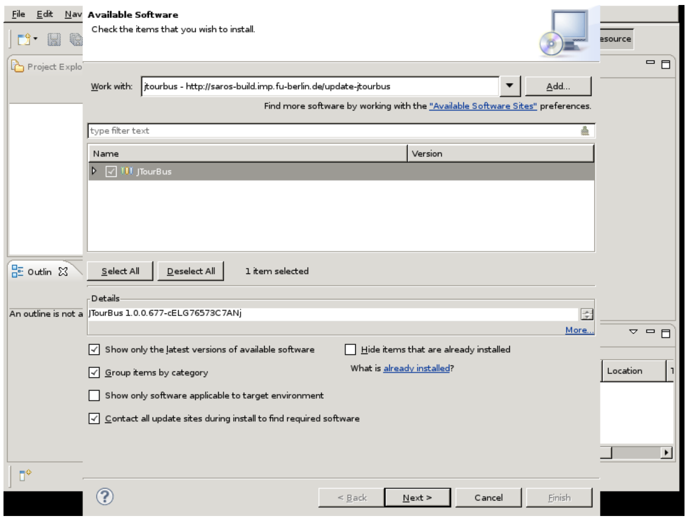
[Select All] [Cancel] [OK]

Details

☑ Show only the latest versions of available software
☐ Hide items that are already installed
What is already installed?
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software

[< Back] [Next >] [Cancel] [Finish]

4. SELECT JTOURBUS



Available Software

Check the items that you wish to install.

Work with: jtourbus - http://saros-build.imp.fu-berlin.de/update-jtourbus [Add...]

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
☑ JTourBus	

[Select All] [Deselect All] 1 item selected

Details

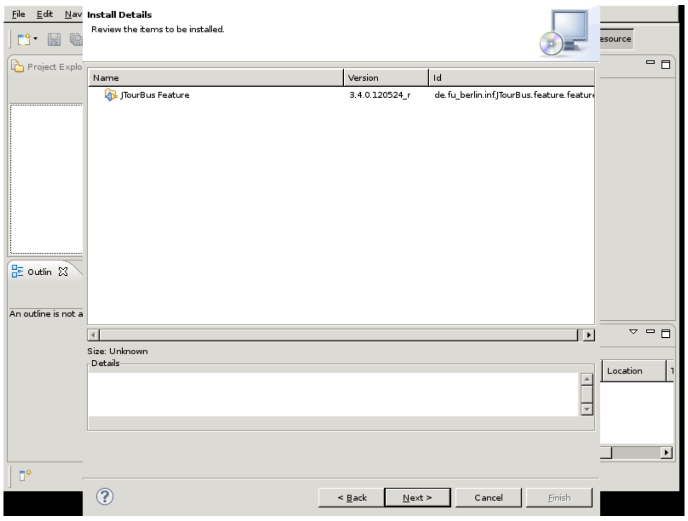
JTourBus 1.0.0.677-cELG76573C7ANj

More...

☑ Show only the latest versions of available software
☐ Hide items that are already installed
What is already installed?
☑ Group items by category
☐ Show only software applicable to target environment
☑ Contact all update sites during install to find required software

[< Back] [Next >] [Cancel] [Finish]

5. REVIEW THE INSTALLATION DETAILS



Install Details

Review the items to be installed.

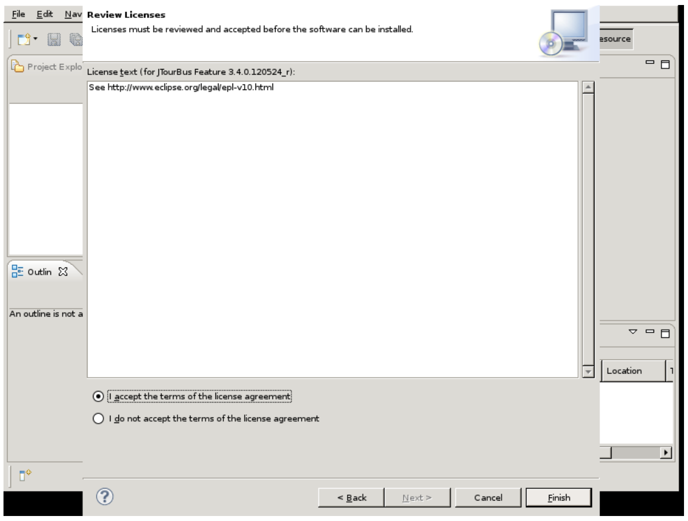
Name	Version	Id
JTourBus Feature	3.4.0.120524_r	de.fu_berlin.inf.jTourBus.feature.featur

Size: Unknown

Details

[< Back] [Next >] [Cancel] [Finish]

6. ACCEPT LICENSES



Review Licenses

Licenses must be reviewed and accepted before the software can be installed.

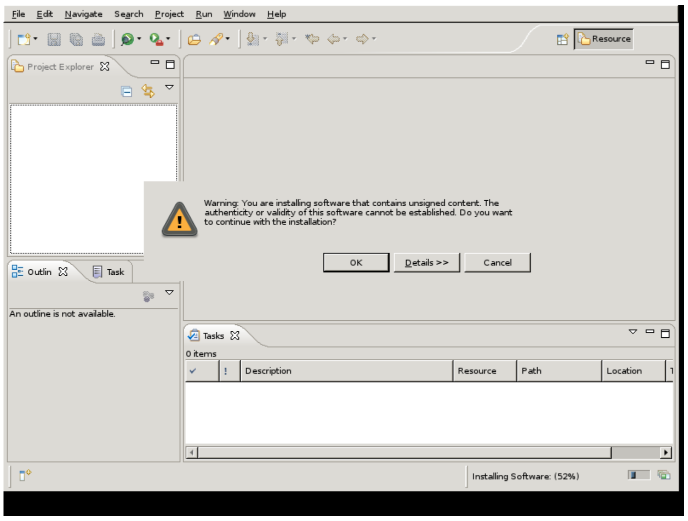
License text (for JTourBus Feature 3.4.0.120524_r):

See http://www.eclipse.org/legal/epl-v10.html

[✓ I accept the terms of the license agreement]

I do not accept the terms of the license agreement

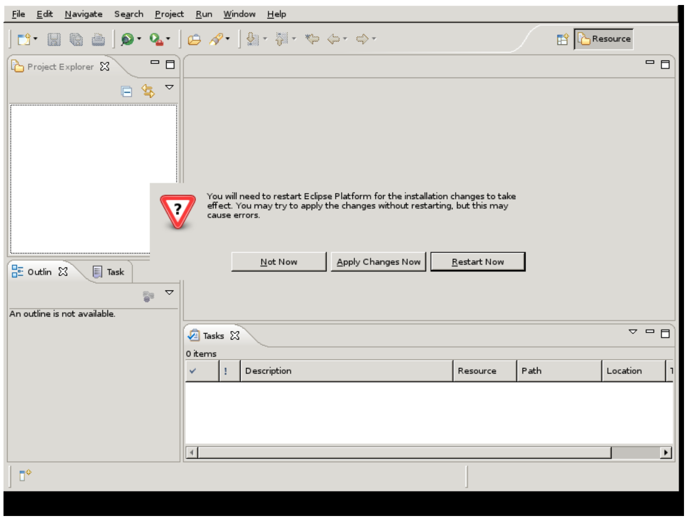
7. Accept unsigned



Warning: You are installing software that contains unsigned content. The authenticity or validity of this software cannot be established. Do you want to continue with the installation?

Installing Software: (52%)

8. Restart Eclipse



You will need to restart Eclipse Platform for the installation changes to take effect. You may try to apply the changes without restarting, but this may cause errors.

Chapter 4

Sourcecode and Review

We are using Git to manage our source code, Jenkins for continuous integration and Gerrit for the code review. We assume that you understand the benefits of version control, continuous integration and code review but you might not know the specific utilities. This chapter should help you find out how they are integrated and how you are going to use them.

You will need to get the source code from our main git repository. After you have completed modifications to the source you can create local commits. Your commits can be pushed to a repository you own or you can push them into the Gerrit code review system. This will result in Gerrit creating a *Chase* and will lead to the Jenkins server being triggered and asked to compile and test your commits. The result will be returned to Gerrit. Your teammates are asked to make reviews of your. You are asked to modify your change until both the Jennkins result and the feedback of your teammates is positive. Your commits will be automatically integrated into the main repository.

4.1 Version Control with Git

4.1.1 Introduction to Git

Git is a modern and widely used version control system. We assume that you have already worked with another version control system and will solely focus on some Git basics and how this is can be used in the context of Saros. The central concept is a *Git Repository*, it is located in the `.git/` subdirectory and holds the entire history of the project (branches, commits, all versions of files). The *Git checkout* command takes a specific revision from your local repository and updates, adds and removes files from your *Git working directory* to match the state of that revision. *Git commit* allows you to create a local commit that will be stored in your local *Git Repository*. The *Git Index* is a unique concept that represents the state that will be used for the next commit. You will mostly use it to add a new file, remove an existing one. An advanced usage of the *Git Index* is to partially add modifications of a file to the index, in the beginning you will probably not want to use that specific feature though. *Git* allows you to create local branches that are branched off any *Git commit* that is already in the local *Git Repository*. In case you created a *Git Commit* too quickly you can always *amend* to the last one. Once you want to share your changes you will need to *Git push*, this will move your local changes to the remote *Git Repository*. To get changes from a remote *Git Repository* you will need to use the *Git pull*. This will transfer all commits from the remote repository to your local one and it will try to move your changes to the top of the history. The last operation is called a *Git rebase*.

There are plenty of Git introductions available on the web, some describe the technical details, some are for people coming from Subversion, some show how to use the Eclipse integration. The Git website has a documentation section with links to good documentation. If the following guide is not enough you should be able to find information there.

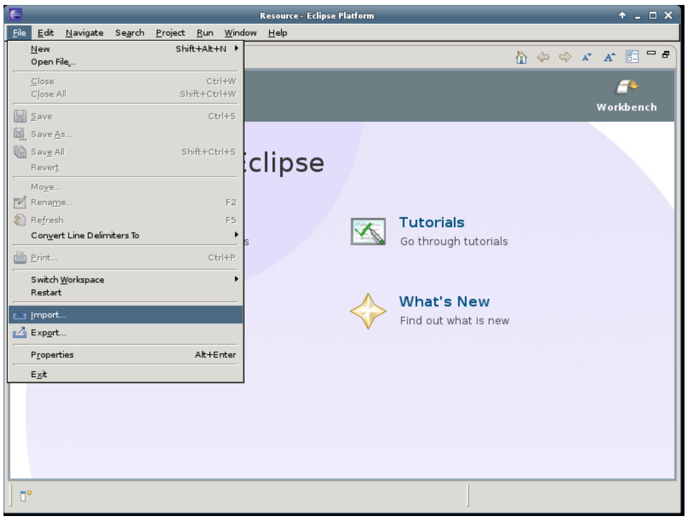
4.1.2 Git clone with EGit (textual)

EGit Plugin versions
The below procedure was tested with Eclipse 3.7.1 and EGit plugin version 1.3.0.201202151440-r.

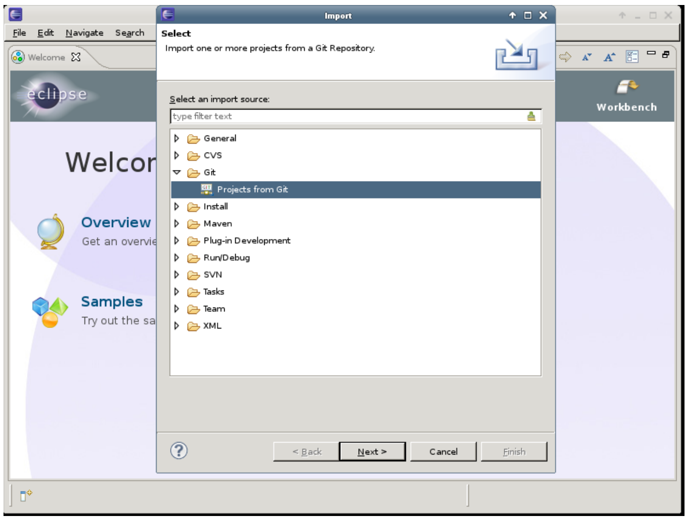
1. BEGIN
   File → Import
2. SELECT IMPORT FROM GIT
   Select Git → Projects from Git and press Next.
3. SELECT REPOSITORY SOURCE
   Select URI and press Next.
4. SOURCE GIT REPOSITORY
   Fill in *dpp.git.sourceforge.net* as Host, use *gitroot/dpp/saros* as repository path and select *git* as the protocol and finish by pressing the Next button.
5. SELECT THE BRANCH
   Select the *master* branch and continue by pressing Next.
6. SELECT LOCAL DESTINATION
   Decide where to store the cloned Git repository. The proposed directory should work and press Next.
7. CLONING THE REPOSITORY
   Progress of downloading the repository is shown.
8. SELECT PROJECT DIRECTORY
   Select *Import existing projects* and import the top level directory *Working Directory* and continue using the Next button.
9. IMPORT PROJECTS
   Select at least the *de.fu_berlin.inf.nebula* and *Saros* to import.
10. SWITCH TO *Git Perspective*
    Clicking on Window+Open Perspective → Other and select the *Git Perspective*.
11. ENTER GERRIT CONFIGURATION
    Unfold the *Saros*, *Remotes* and *origin* folder. Use the context menu and select *Gerrit Configuration*.
12. CONFIGURE GERRIT
    Use *ssh://saros-build.imp.fu-berlin.de:29418/saros.git* as the *Push URI* and *refs/for/master* as the *Destination branch*.
13. VERIFY EGIT IS ENABLED
    Switch to the *Java Perspective* and use the context menu in the *Package Explorer* and verify that the Team contains options for commit.

4.1.3 Git clone with EGit (graphical)

1. BEGIN



2. Select import from Git

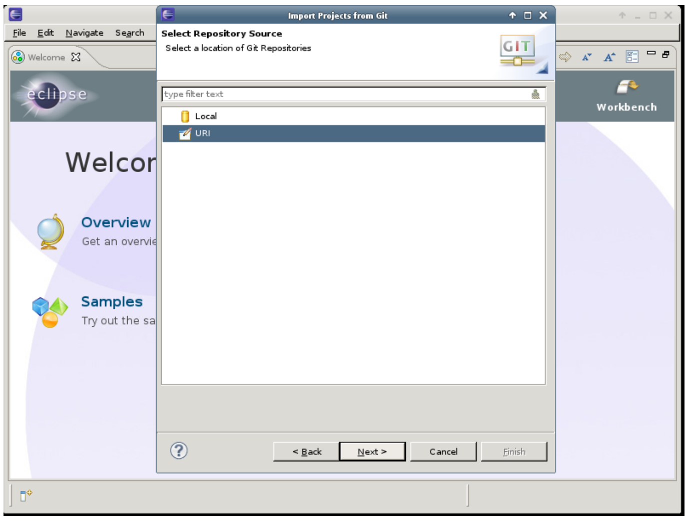


Select

Import one or more projects from a Git Repository.

Select an import source:

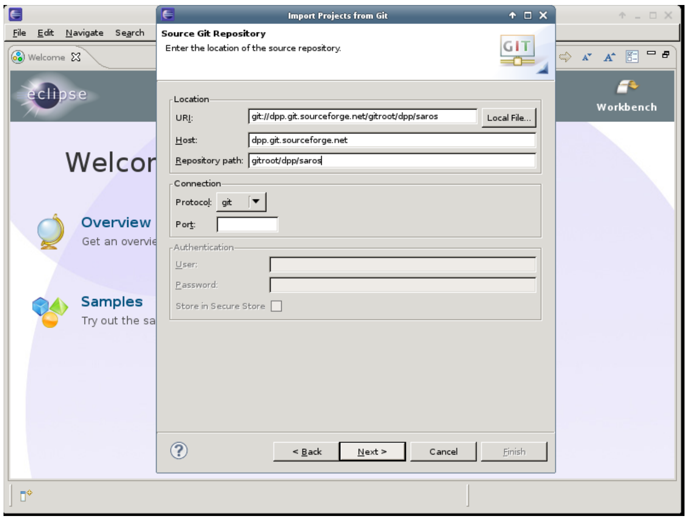
3. Select Repository Source



Select Repository Source

Select a location of Git Repositories

4. Source Git Repository



Import Projects from Git

Source Git Repository

Enter the location of the source repository.

Location

URI: git://dpp.git.sourceforge.net/gitroot/dpp/saros

Host: dpp.git.sourceforge.net

Repository path: gitroot/dpp/saros

Connection

Protocol: git

Port:

Authentication

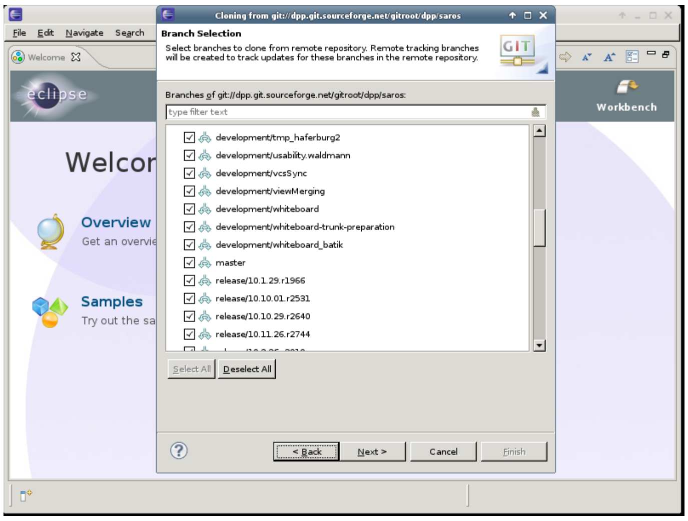
User:

Password:

Store in Secure Store ☐

Host: *dpp.git.sourceforge.net*, Path: *gitroot/dpp/saros*

5. SELECT THE BRANCH



Cloning from git://dpp.git.sourceforge.net/gitroot/dpp/saros

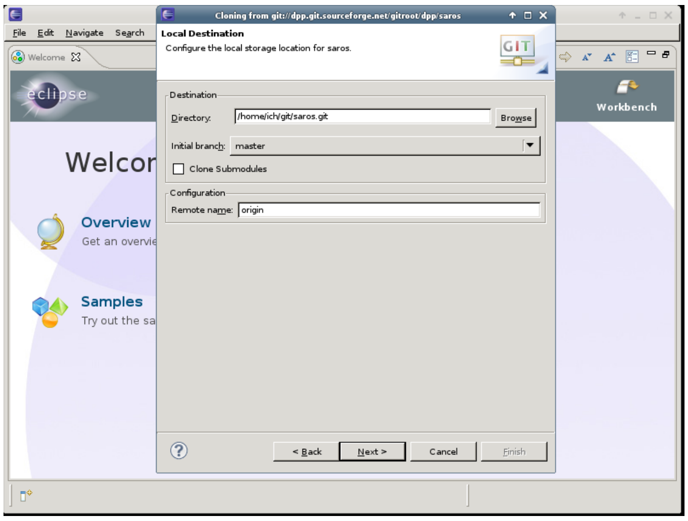
Branch Selection

Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.

Branches of git://dpp.git.sourceforge.net/gitroot/dpp/saros:

- ☑ development/tmp_haferburg2
- ☑ development/usability.waldmann
- ☑ development/vcsSync
- ☑ development/viewMerging
- ☑ development/whiteboard
- ☑ development/whiteboard-trunk-preparation
- ☑ development/whiteboard_batik
- ☑ master
- ☑ release/10.1.29.r1966
- ☑ release/10.10.01.r2531
- ☑ release/10.10.29.r2640
- ☑ release/10.11.26.r2744

6. SELECT LOCAL DESTINATION



Cloning from git://dpp.git.sourceforge.net/gitroot/dpp/saros

Local Destination

Configure the local storage location for saros.

Destination

Directory: /home/ich/git/saros.git

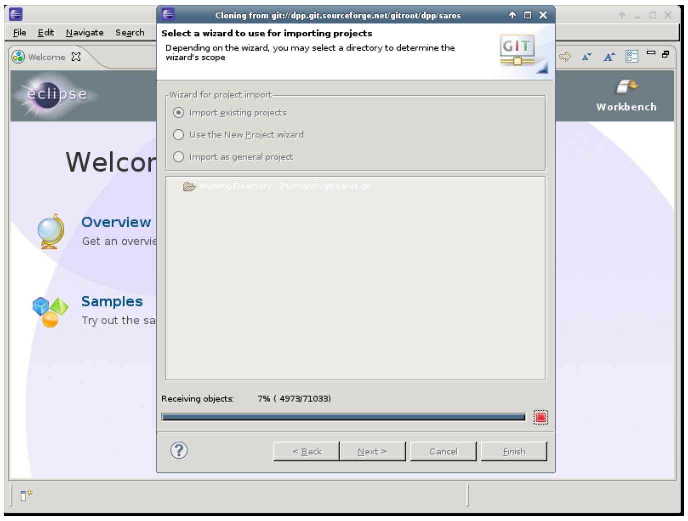
Initial branch: master

☐ Clone Submodules

Configuration

Remote name: origin

7. CLONING THE REPOSITORY



Cloning from git://dpp.git.sourceforge.net/gitroot/dpp/saros

Select a wizard to use for importing projects

Depending on the wizard, you may select a directory to determine the wizard's scope

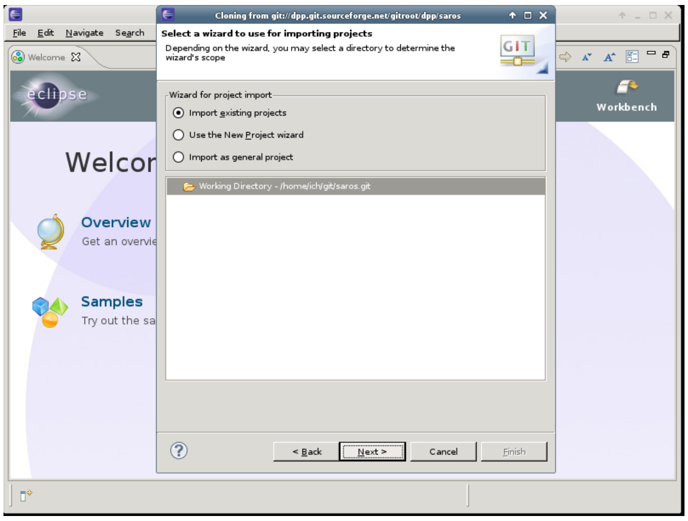
Wizard for project import

(●) Import existing projects
( ) Use the New Project wizard
( ) Import as general project

Working Directory - /home/ich/git/saros.git

Receiving objects: 7% (4973/71033)

8. Select project directory



Cloning from git://dpp.git.sourceforge.net/gitroot/dpp/saros

Select a wizard to use for importing projects

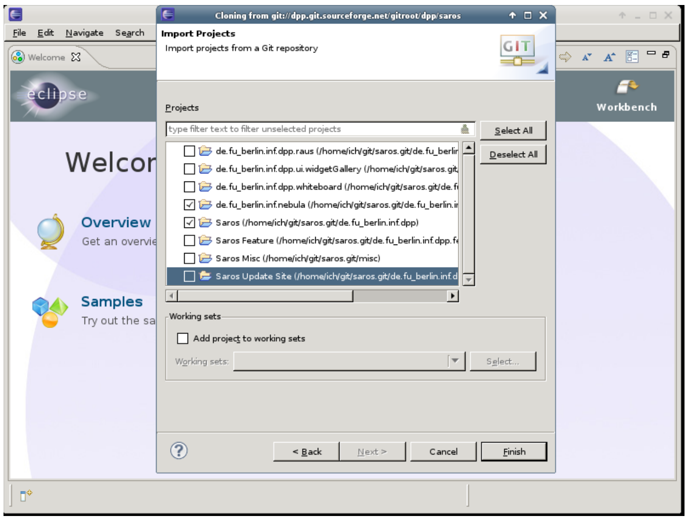
Depending on the wizard, you may select a directory to determine the wizard's scope

Wizard for project import

(●) Import existing projects
( ) Use the New Project wizard
( ) Import as general project

Working Directory - /home/ich/git/saros.git

9. Import projects



Cloning from git://dpp.git.sourceforge.net/gitroot/dpp/saros

Import Projects

Import projects from a Git repository

Projects

type filter text to filter unselected projects

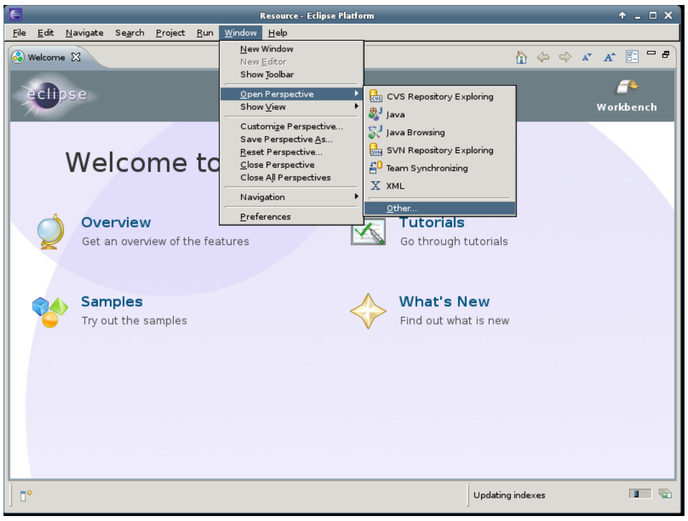
- [ ] de.fu_berlin.inf.dpp.raus (/home/ich/git/saros.git/de.fu_berlin
- [ ] de.fu_berlin.inf.dpp.ui.widgetGallery (/home/ich/git/saros.git/
- [ ] de.fu_berlin.inf.dpp.whiteboard (/home/ich/git/saros.git/de.f
- [x] de.fu_berlin.inf.nebula (/home/ich/git/saros.git/de.fu_berlin.i
- [x] Saros (/home/ich/git/saros.git/de.fu_berlin.inf.dpp)
- [ ] Saros Feature (/home/ich/git/saros.git/de.fu_berlin.inf.dpp.f
- [ ] Saros Misc (/home/ich/git/saros.git/misc)
- [ ] Saros Update Site (/home/ich/git/saros.git/de.fu_berlin.inf.d

Working sets

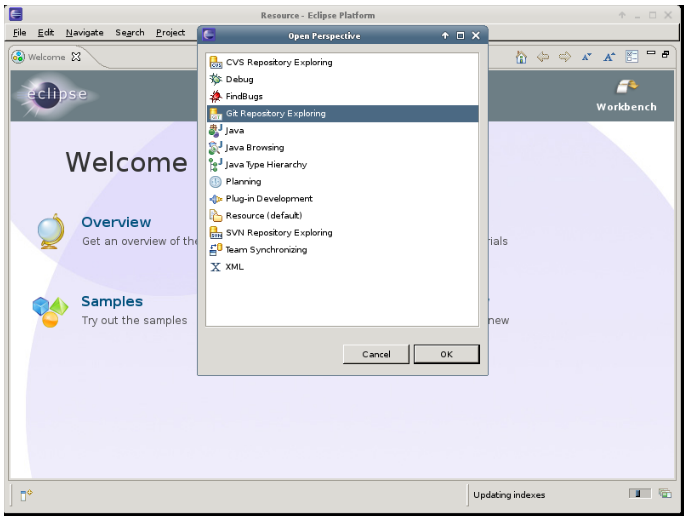
- [ ] Add project to working sets

Working sets:

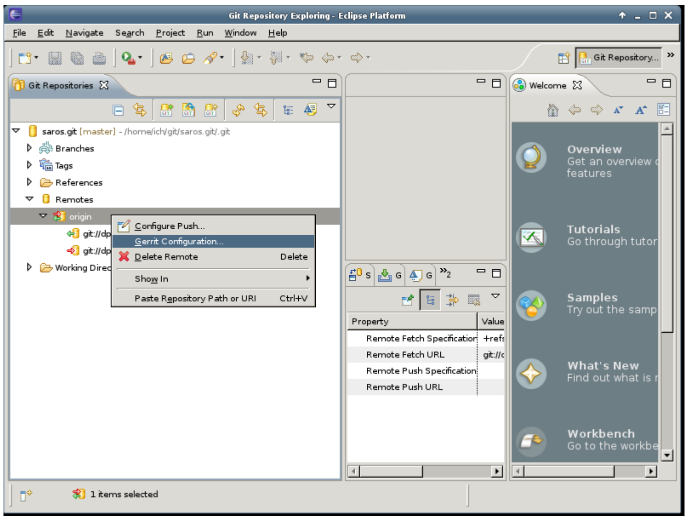
10. Switch to the Git Perspective



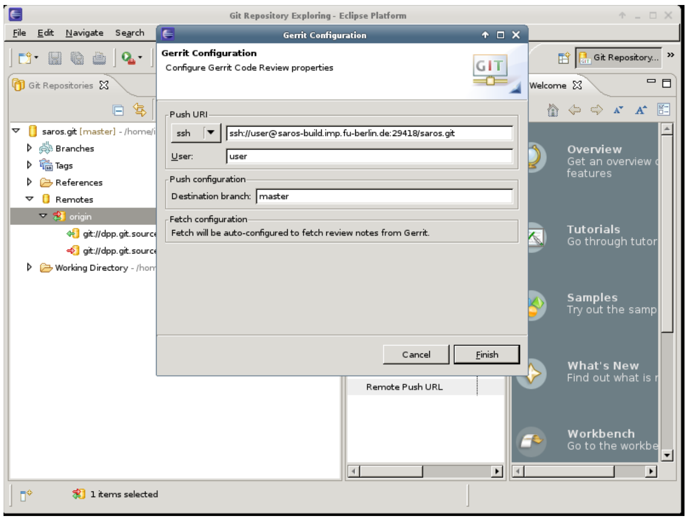
11. Switch to the Git perspective



12. Enable Gerrit



13. Configure Gerrit


Gerrit Configuration

Configure Gerrit Code Review properties

Push URI

ssh ssh://user@saros-build.imp.fu-berlin.de:29418/saros.git

User: user

Push configuration

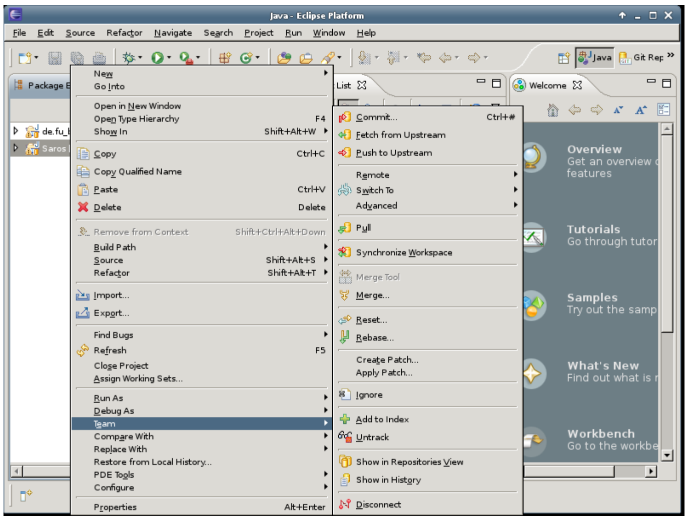
Destination branch: master

Fetch configuration

Fetch will be auto-configured to fetch review notes from Gerrit.

Use *saros-build.imp.fu-berlin.de* and *refs/for/master*

14. VERIFY EGIT BEING USED IN THE JAVA PERSPECTIVE



4.2 Configuring Git for Gerrit

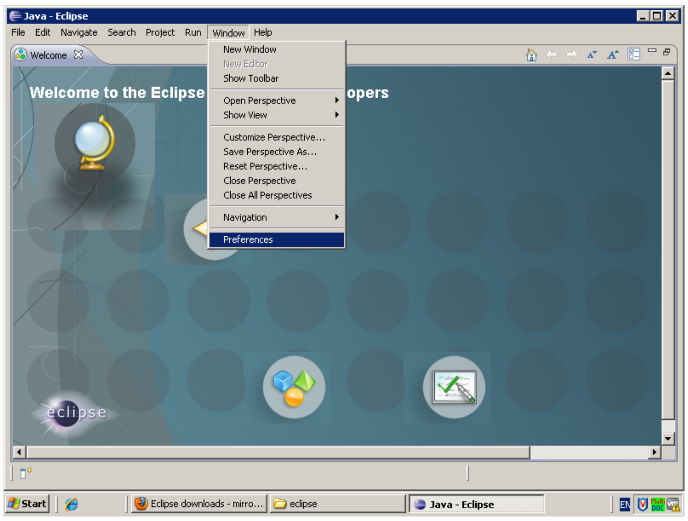
So far you managed to clone a remote *Git repository* and have checked out the default branch. To make local commits you will need to set a name and email address. If you do not do this Gerrit might reject your commits. If you want to make your changes available and ask for review you will need to push your local commits into the *Gerrit Git repository*. *Git* is using SSH to communicate with remote *Git repositories*, you will authenticate with your private SSH key. The following sections will show you how to do it.

4.2.1 Creating an SSH key

You can create SSH keys using Eclipse. The following procedure will explain the process.

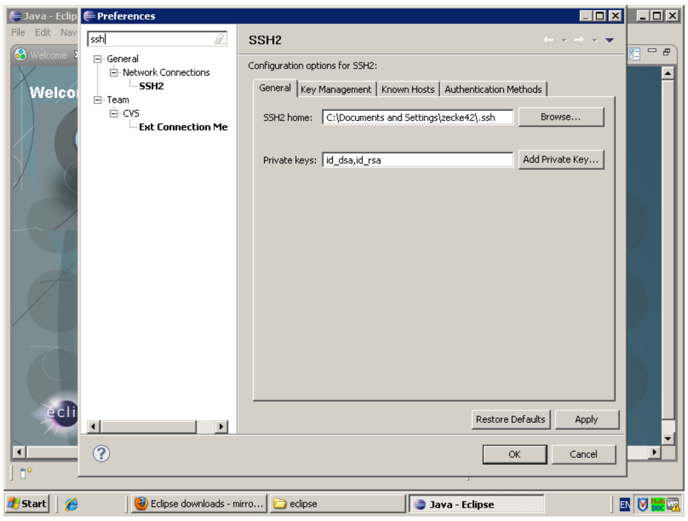
1. OPEN THE PREFERENCES

   Window → Preferences



2. Select the SSH Configuration

   Enter SSH into the search field and select *SSH2*.

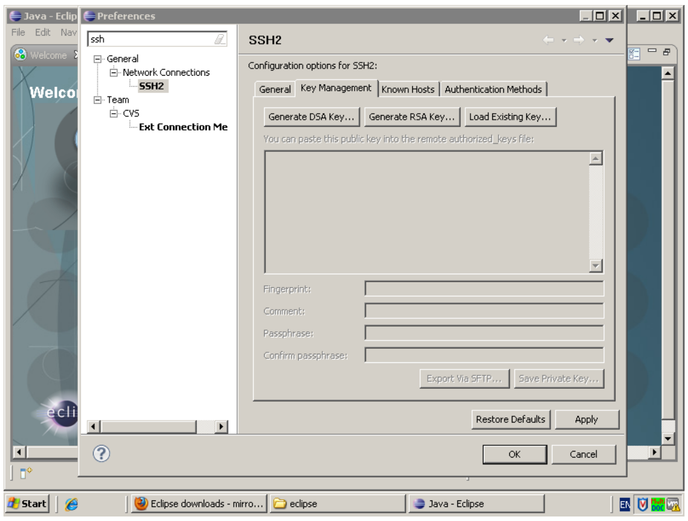


Configuration options for SSH2:

SSH2 home: C:\Documents and Settings\zecke42\.ssh

Private keys: id_dsa,id_rsa

3. Switch to the Key Management tab

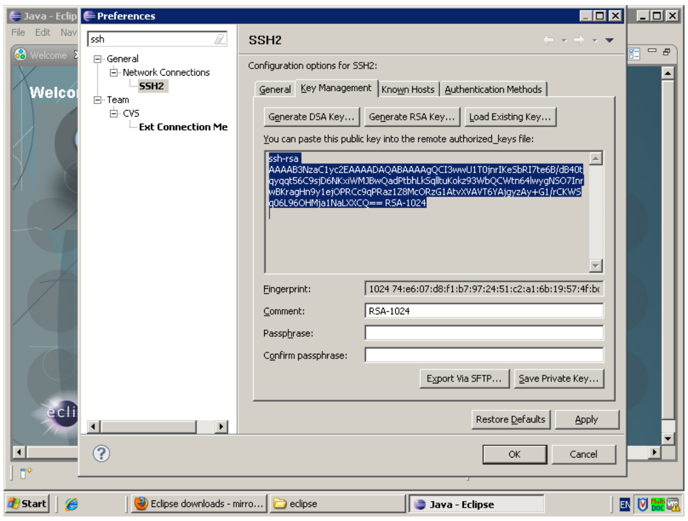


Configuration options for SSH2:

You can paste this public key into the remote authorized_keys file:

4. Generate a RSA key

   Select the Generate RSA Key to generate a new key pair. Optionally set a password. The selected text is the public key and needs to be used for Gerrit.



SSH2

Configuration options for SSH2:

General | Key Management | Known Hosts | Authentication Methods

You can paste this public key into the remote authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAAAgQCI3wwU1T0jnrIKeSbRI7te6B/dB40t
qyqqt56C9sjD6NKxiWMJBwQadPtbhLkSqlltuKokz93WbQCWtn64lwygNSO7Inr
wBKragHn9y1ejOPRCc9qPRaz1Z8McORzG1AtvXVAVT6YAjgyzAy+G1/rCKWS
q06L96OHMja1NaLXXCQ== RSA-1024
```

Fingerprint: 1024 74:e6:07:d8:f1:b7:97:24:51:c2:a1:6b:19:57:4f:bc

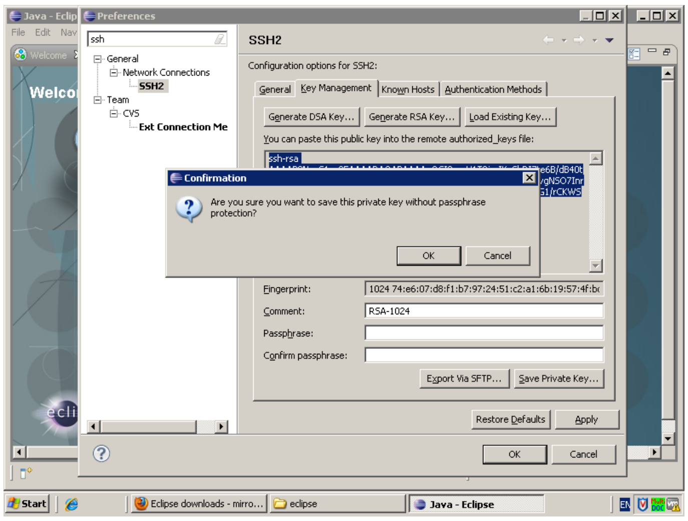
Comment: RSA-1024

Passphrase:

Confirm passphrase:

5. Save the Key

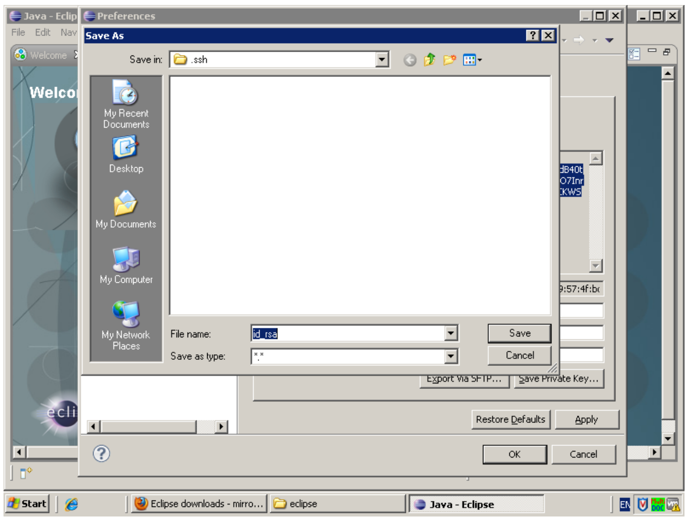
Press Save Private Key to store the new keypair. This will require clicking a couple of times.



Confirmation

Are you sure you want to save this private key without passphrase protection?

OK | Cancel



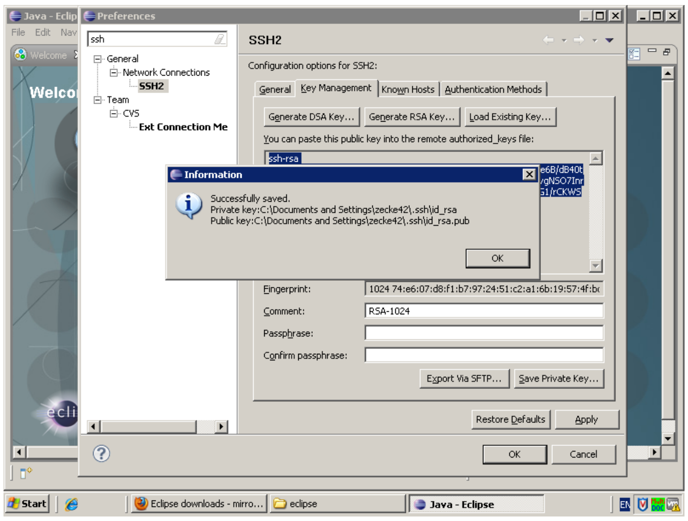
Save As

Save in: .ssh

File name: id_rsa

Save as type: *.*

Save | Cancel



Configuration options for SSH2:

You can paste this public key into the remote authorized_keys file:

ssh-rsa

Information

Successfully saved.
Private key:C:\Documents and Settings\zecke42\.ssh\id_rsa
Public key:C:\Documents and Settings\zecke42\.ssh\id_rsa.pub

OK

Fingerprint: 1024 74:e6:07:d8:f1:b7:97:24:51:c2:a1:6b:19:57:4f:bc

Comment: RSA-1024

Passphrase:

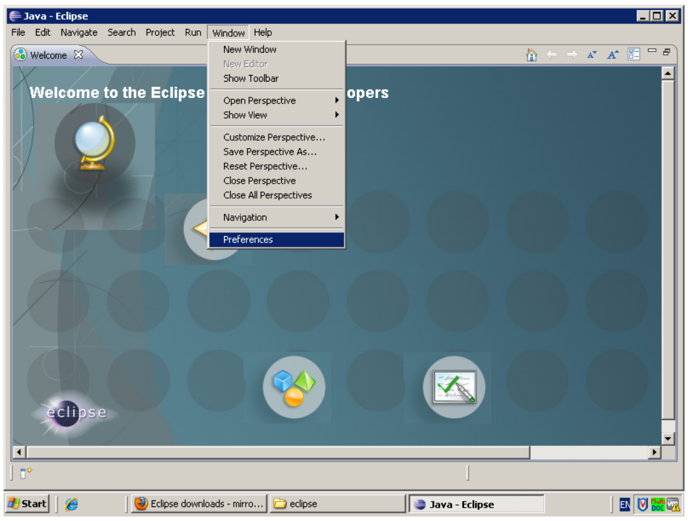
Confirm passphrase:

4.2.2 Setting your name and email address

You will need to set your name and email address. Git has a very simple configuration format. You will need to set the keys *user.name* and *user.email* to the names of your choice. The below procedure will show you how to do it and then you are ready to make local commits.

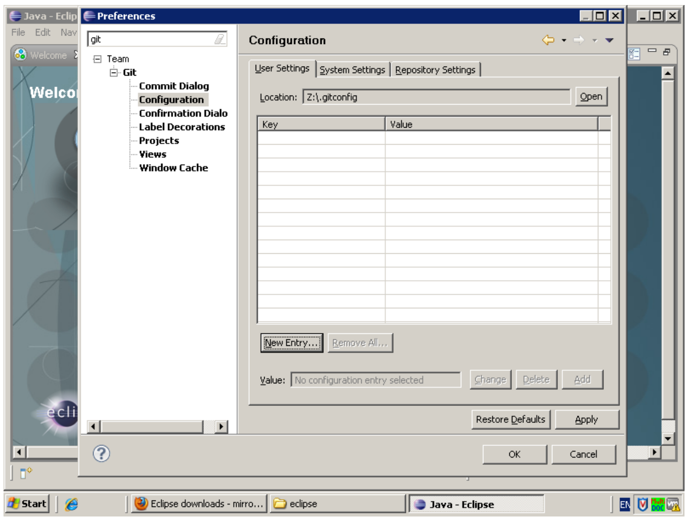
1. OPEN THE PREFERENCES

   Window → Preferences



2. OPEN THE GIT CONFIGURATION

   Search for *Git* and select the Configuration page.

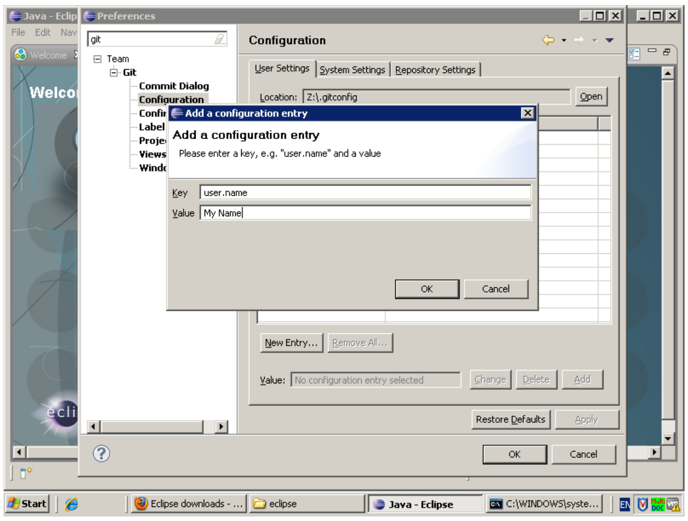


Location: Z:\.gitconfig

Value: No configuration entry selected

3. ADD AN ENTRY FOR YOUR NAME

   Press the New Entry button and use *user.name* as the key and set your name as the value.



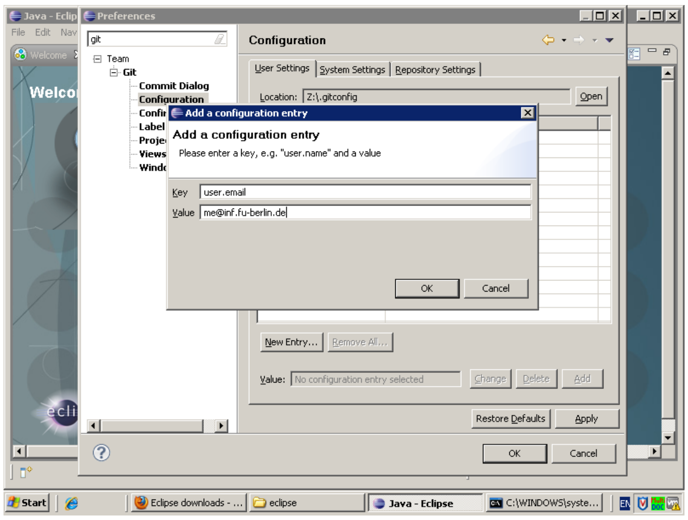
Please enter a key, e.g. "user.name" and a value

Key: user.name

Value: My Name

4. ADD AN ENTRY FOR YOUR EMAIL ADDRESS

   Press the New Entry button and use *user.email* as the key and set your email address as the value.



Please enter a key, e.g. "user.name" and a value

Key: user.email

Value: me@inf.fu-berlin.de

4.3 Register on Gerrit

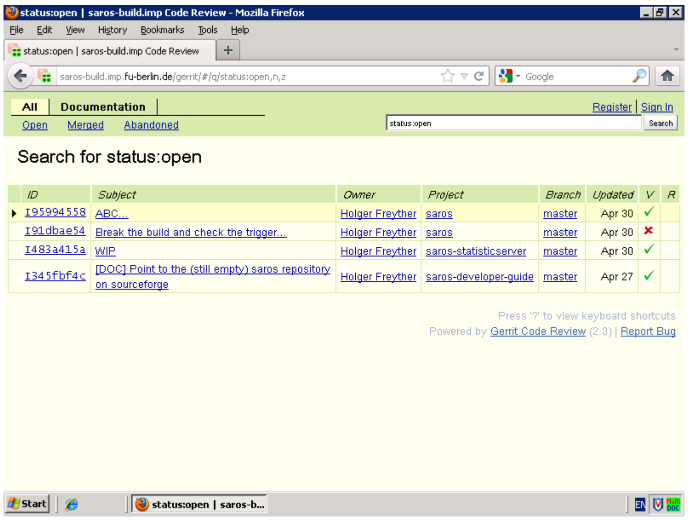
We are using Gerrit for code review and automatic integration of reviewed patches. You will need to create an account, pick a user name, register your email addresses and add your public SSH key. Gerrit will compare the committer email address in the Git commits with the registered email addresses for your SSH key.

4.3.1 Signing up using OpenID

Our Gerrit is doing authentication based on OpenID. This means you will need to have an account with an OpenID provider. Google and Yahoo are well known providers. After you signed up you need to pick a user name, add your email addresses and SSH keys. The first time you login all this can be done from the landing page.

1. NAVIGATE TO GERRIT

   Go to http://saros-build.imp.fu-berlin.de/gerrit and press *Sign In*.


status:open | saros-build.imp Code Review - Mozilla Firefox
File Edit View History Bookmarks Tools Help
saros-build.imp.fu-berlin.de/gerrit/#/q/status:open,n,z
All
Documentation
Open
Merged
Abandoned
Register | Sign In
Search
Start
status:open | saros-b...

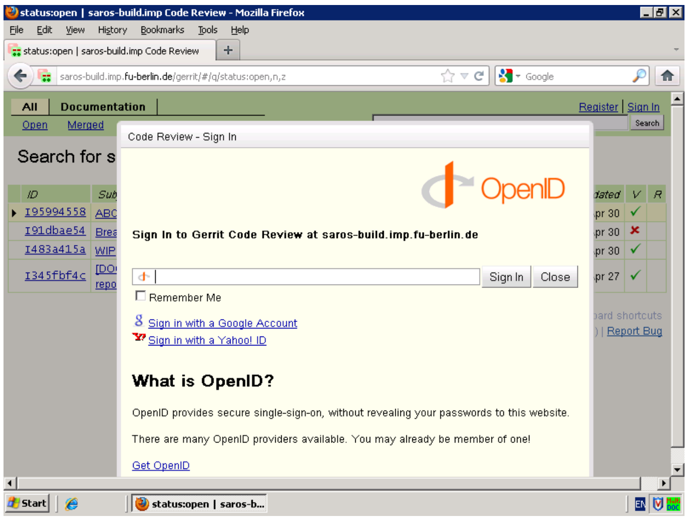
Search for status:open

ID	Subject	Owner	Project	Branch	Updated	V	R
I95994558	ABC...	Holger Freyther	saros	master	Apr 30	✓	
I91dbae54	Break the build and check the trigger...	Holger Freyther	saros	master	Apr 30	✗	
I483a415a	WIP	Holger Freyther	saros-statisticserver	master	Apr 30	✓	
I345fbf4c	[DOC] Point to the (still empty) saros repository on sourceforge	Holger Freyther	saros-developer-guide	master	Apr 27	✓	

Press '?' to view keyboard shortcuts
Powered by Gerrit Code Review (2.3) | Report Bug

2. SELECT AN OPENID PROVIDER

   Select your OpenID provider, e.g. Yahoo, Google, Livejournal, MyOpenID.



Sign In to Gerrit Code Review at saros-build.imp.fu-berlin.de

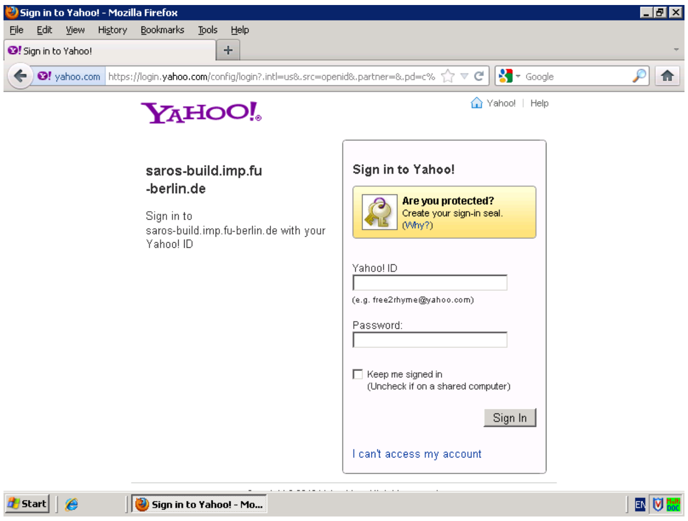
What is OpenID?

OpenID provides secure single-sign-on, without revealing your passwords to this website.

There are many OpenID providers available. You may already be member of one!

3. SIGN INTO YOUR OPENID PROVIDER

   You will need to sign into your account.


Sign in to Yahoo! - Mozilla Firefox

https://login.yahoo.com/config/login?.intl=us&.src=openid&.partner=&.pd=c%

saros-build.imp.fu-berlin.de

Sign in to saros-build.imp.fu-berlin.de with your Yahoo! ID

Sign in to Yahoo!

Are you protected? Create your sign-in seal. (Why?)

Yahoo! ID

(e.g. free2rhyme@yahoo.com)

Password:

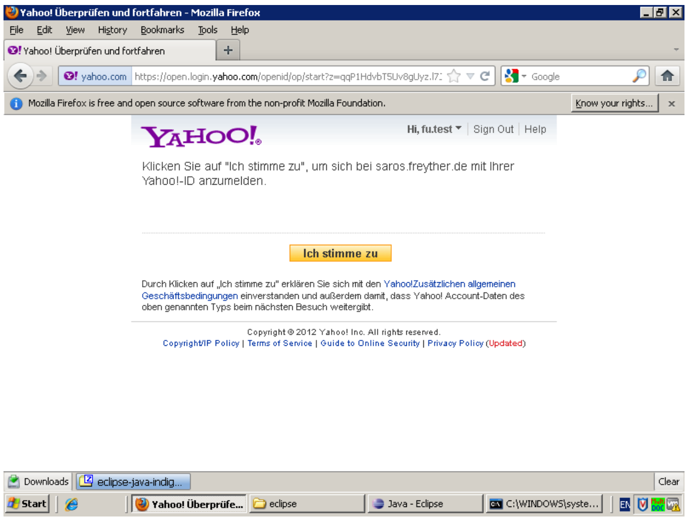
Keep me signed in (Uncheck if on a shared computer)

Sign In

I can't access my account

4. Authorize the usage by Gerrit

   Allow to use your identity by Gerrit.


Yahoo! Überprüfen und fortfahren - Mozilla Firefox

https://open.login.yahoo.com/openid/op/start?z=qqP1HdvbT5Uv8gUyz.I7:

Mozilla Firefox is free and open source software from the non-profit Mozilla Foundation.

Hi, fu.test | Sign Out | Help

Klicken Sie auf "Ich stimme zu", um sich bei saros.freyther.de mit Ihrer Yahoo!-ID anzumelden.

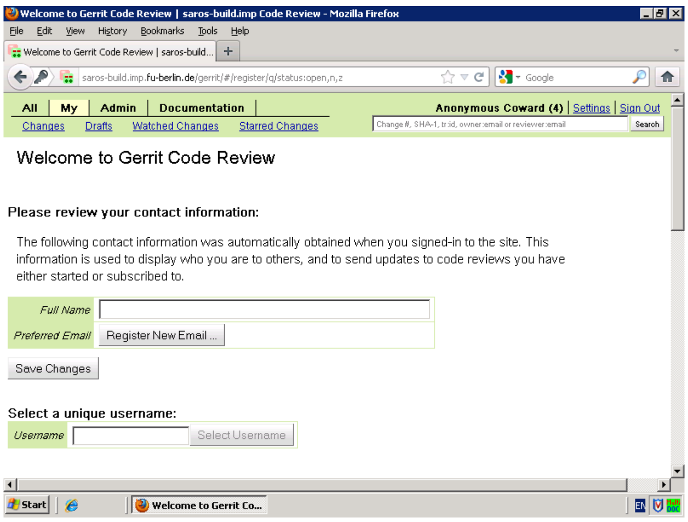
Ich stimme zu

Durch Klicken auf „Ich stimme zu" erklären Sie sich mit den Yahoo!Zusätzlichen allgemeinen Geschäftsbedingungen einverstanden und außerdem damit, dass Yahoo! Account-Daten des oben genannten Typs beim nächsten Besuch weitergibt.

Copyright © 2012 Yahoo! Inc. All rights reserved.
Copyright/IP Policy | Terms of Service | Guide to Online Security | Privacy Policy (Updated)

5. Configure the Gerrit account

   Your browser will return to our Gerrit and you will need to set your name, pick a user name, add your SSH keys. You should ask your supervisor to add you to the *Reviewer* group to be allowed to make reviews and submit code.


Welcome to Gerrit Code Review | saros-build.imp Code Review - Mozilla Firefox

saros-build.imp.fu-berlin.de/gerrit/#/register/q/status:open,n,z

All | My | Admin | Documentation — Anonymous Coward (4) | Settings | Sign Out

Changes | Drafts | Watched Changes | Starred Changes

Welcome to Gerrit Code Review

Please review your contact information:

The following contact information was automatically obtained when you signed-in to the site. This information is used to display who you are to others, and to send updates to code reviews you have either started or subscribed to.

Full Name	
Preferred Email	Register New Email ...

Save Changes

Select a unique username:

Username — Select Username

4.3.2 Concepts and Terminology

Change

Every time you push a commit with a new *Change-Id* Gerrit allocates a change. The *change* contains a number of *patchsets*, comments on the patchsets and ratings in various *categories*. Each change has a dedicated page that shows information about it. This includes dependencies between different changes, patchsets and the review comments.

Category

A *category* has a name and a rating scale. This is used to allow Users, Jenkins and other scripts to provide review. Right now we have the *Verified*, *Review* and *Sanity categories* are configured. The rating can range from -2 to +2.

Patchset

A *patchset* is a git commit that has been pushed to Gerrit. Gerrit uses the *Change-Id* of the commit meesage to identify the change. Each *patchset* can receive inline comments.

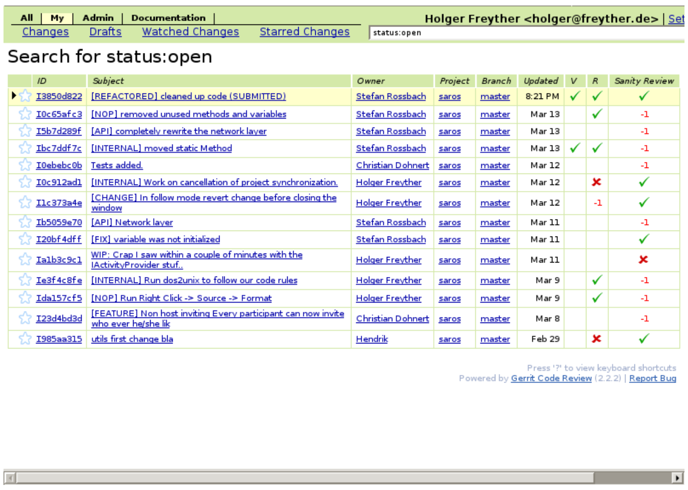
Submit

Once a *change* has received a +2 in the *Review category* and no negative voting in the other *categories* the last patchset can be *submitted*. This means Gerrit will now try to merge your patchset and mark the change as merged.

4.3.3 Making a review

Once you have opened the Gerrit page you can move to the list of changes that require review by clicking *All*. This will show you the list of all changes that require action. Clicking on any of these changes will bring you to the change overview. This page contains information about the change, already received voting, dependencies of the change with other changes and allows you to look at the current patchset and leave inline comments. Once you went through all the touched files you can click the Review button, leave a vote and provide a comment. The easiest way to start the review is by clicking on the *Commit Message* and then jump to the next file. Inside the *Differences* view you can modify the display by clicking on the *Side-by-Side*, *Unified Preferences*, *Patch Sets* links.

1. SWITCH TO THE *All* PAGE


All | My | Admin | Documentation
Changes
Drafts
Watched Changes
Starred Changes
Holger Freyther <holger@freyther.de> | Set
status:open

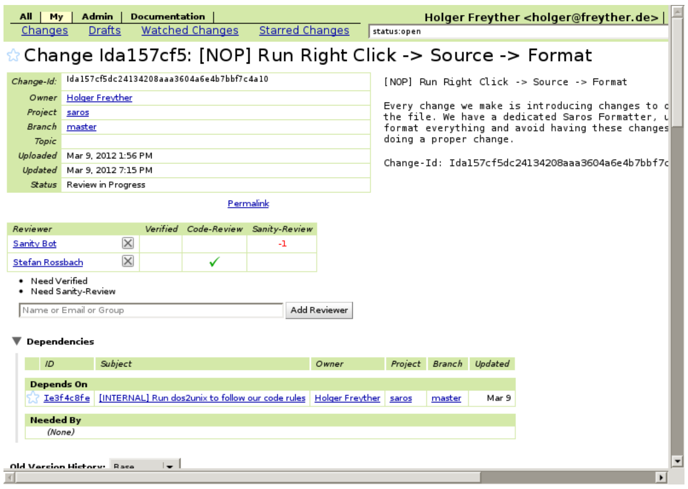
Search for status:open

	ID	Subject	Owner	Project	Branch	Updated	V	R	Sanity Review
▸ ☆	I3850d822	[REFACTORED] cleaned up code (SUBMITTED)	Stefan Rossbach	saros	master	8:21 PM	✓	✓	✓
☆	I0c65afc3	[NOP] removed unused methods and variables	Stefan Rossbach	saros	master	Mar 13		✓	-1
☆	I5b7d289f	[API] completely rewrite the network layer	Stefan Rossbach	saros	master	Mar 13			-1
☆	Ibc7ddf7c	[INTERNAL] moved static Method	Stefan Rossbach	saros	master	Mar 13	✓	✓	-1
☆	I0ebebc0b	Tests added.	Christian Dohnert	saros	master	Mar 12			-1
☆	I0c912ad1	[INTERNAL] Work on cancellation of project synchronization.	Holger Freyther	saros	master	Mar 12		✗	✓
☆	I1c373a4e	[CHANGE] In follow mode revert change before closing the window	Holger Freyther	saros	master	Mar 12		-1	✓
☆	Ib5059e70	[API] Network layer	Stefan Rossbach	saros	master	Mar 11			-1
☆	I20bf4dff	[FIX] variable was not initialized	Stefan Rossbach	saros	master	Mar 11			✓
☆	Ia1b3c9c1	WIP: Crap I saw within a couple of minutes with the IActivityProvider stuf..	Holger Freyther	saros	master	Mar 11			✗
☆	Ie3f4c8fe	[INTERNAL] Run dos2unix to follow our code rules	Holger Freyther	saros	master	Mar 9		✓	-1
☆	Ida157cf5	[NOP] Run Right Click -> Source -> Format	Holger Freyther	saros	master	Mar 9		✓	-1
☆	I23d4bd3d	[FEATURE] Non host inviting Every participant can now invite who ever he/she lik	Christian Dohnert	saros	master	Mar 8			-1
☆	I985aa315	utils first change bla	Hendrik	saros	master	Feb 29		✗	✓

Press '?' to view keyboard shortcuts
Powered by Gerrit Code Review (2.2.2) | Report Bug

Select the change you want to review

2. INDIVIDUAL *Change* PAGE


All | My | Admin | Documentation |
Changes Drafts Watched Changes Starred Changes
Holger Freyther <holger@freyther.de> |
status:open
Permalink
Add Reviewer
Old Version History: Base

Change Ida157cf5: [NOP] Run Right Click -> Source -> Format

Change-Id:	Ida157cf5dc24134208aaa3604a6e4b7bbf7c4a10
Owner	Holger Freyther
Project	saros
Branch	master
Topic	
Uploaded	Mar 9, 2012 1:56 PM
Updated	Mar 9, 2012 7:15 PM
Status	Review in Progress

```
[NOP] Run Right Click -> Source -> Format

Every change we make is introducing changes to c
the file. We have a dedicated Saros Formatter, u
format everything and avoid having these changes
doing a proper change.

Change-Id: Ida157cf5dc24134208aaa3604a6e4b7bbf7c
```

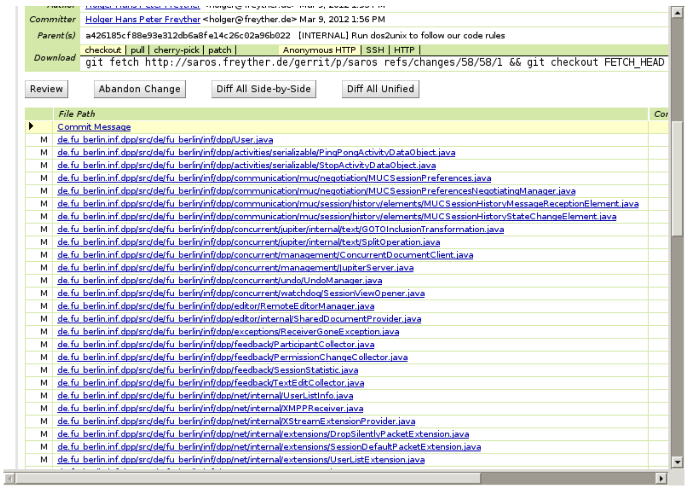
Reviewer	Verified	Code-Review	Sanity-Review
Sanity Bot			-1
Stefan Rossbach		✓	

- Need Verified
- Need Sanity-Review

Name or Email or Group

Dependencies

ID	Subject	Owner	Project	Branch	Updated
Depends On					
Ie3f4c8fe	[INTERNAL] Run dos2unix to follow our code rules	Holger Freyther	saros	master	Mar 9
Needed By					
(None)					



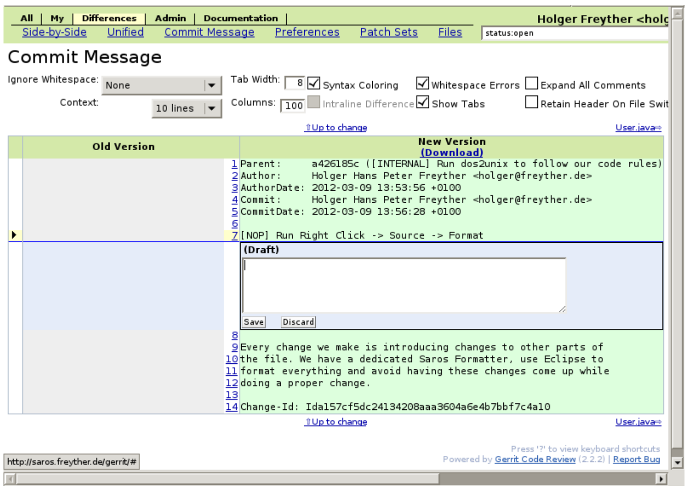
Committer	Holger Hans Peter Freyther <holger@freyther.de> Mar 9, 2012 1:56 PM
Parent(s)	a426185cf88e93e312db6a8fe14c26c02a96b022 [INTERNAL] Run dos2unix to follow our code rules
Download	checkout | pull | cherry-pick | patch | Anonymous HTTP | SSH | HTTP |

```
git fetch http://saros.freyther.de/gerrit/p/saros refs/changes/58/58/1 && git checkout FETCH_HEAD
```

	File Path	Com
	Commit Message	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/User.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/activities/serializable/PingPongActivityDataObject.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/activities/serializable/StopActivityDataObject.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/communication/muc/negotiation/MUCSessionPreferences.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/communication/muc/negotiation/MUCSessionPreferencesNegotiatingManager.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/communication/muc/session/history/elements/MUCSessionHistoryMessageReceptionElement.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/communication/muc/session/history/elements/MUCSessionHistoryStateChangeElement.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/jupiter/internal/text/GOTOInclusionTransformation.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/jupiter/internal/text/SplitOperation.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/management/ConcurrentDocumentClient.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/management/JupiterServer.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/undo/UndoManager.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/concurrent/watchdog/SessionViewOpener.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/editor/RemoteEditorManager.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/editor/internal/SharedDocumentProvider.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/exceptions/ReceiverGoneException.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/feedback/ParticipantCollector.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/feedback/PermissionChangeCollector.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/feedback/SessionStatistic.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/feedback/TextEditCollector.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/UserListInfo.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/XMPPReceiver.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/XStreamExtensionProvider.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/extensions/DropSilentlyPacketExtension.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/extensions/SessionDefaultPacketExtension.java	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/internal/extensions/UserListExtension.java	

Check existing reviews, dependencies and scroll down to the patchsets and start your review by pressing on the *Commit Message*.

3. Provide Comments


All | My | Differences | Admin | Documentation |
Side-by-Side Unified Commit Message Preferences Patch Sets Files
Holger Freyther <holg
status:open
⇧Up to change
User.java⇒
Save
Discard
http://saros.freyther.de/gerrit/#
Press '?' to view keyboard shortcuts
Powered by Gerrit Code Review (2.2.2) | Report Bug

Commit Message

Ignore Whitespace: None Tab Width: 8 [✓ Syntax Coloring] [✓ Whitespace Errors] Expand All Comments
Context: 10 lines Columns: 100 Intraline Difference [✓ Show Tabs] Retain Header On File Swit

Old Version	New Version (Download)
	Parent: a426185c ([INTERNAL] Run dos2unix to follow our code rules)
	Author: Holger Hans Peter Freyther <holger@freyther.de>
	AuthorDate: 2012-03-09 13:53:56 +0100
	Commit: Holger Hans Peter Freyther <holger@freyther.de>
	CommitDate: 2012-03-09 13:56:28 +0100
	[NOP] Run Right Click -> Source -> Format
	(Draft)
	Every change we make is introducing changes to other parts of
	the file. We have a dedicated Saros Formatter, use Eclipse to
	format everything and avoid having these changes come up while
	doing a proper change.
	Change-Id: Ida157cf5dc24134208aaa3604a6e4b7bbf7c4a10

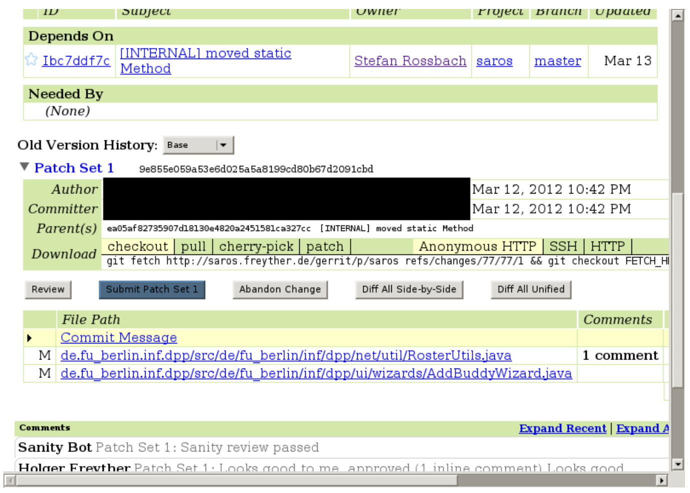
Provide inline comments, or go to the *Preferences* to change the way the patch is displayed.

4. Finishing the Review

Press the *Up to Change* to return to the change page and then the Review to finish your review and leave some final comments and rate the change.

4.3.4 Submitting a change

Once a patchset of a change has been approved the Submit Patchset will appear on the screen. Press the button to try to integrate your patch. The most common reasons that a patchset will not integrate is that a dependency is not integrated yet or the patchset can not be cleanly merged. For the later the following chapter will explain how you can rebase it and re-push it.



ID	Subject	Owner	Project	Branch	Updated
Depends On					
Ibc7ddf7c	[INTERNAL] moved static Method	Stefan Rossbach	saros	master	Mar 13
Needed By					
(None)					

Old Version History: Base

Patch Set 1 9e855e059a53e6d025a5a8199cd80b67d2091cbd

Author	[redacted]	Mar 12, 2012 10:42 PM
Committer		Mar 12, 2012 10:42 PM
Parent(s)	ea05af82735907d18130e4820a2451581ca327cc [INTERNAL] moved static Method	
Download	checkout | pull | cherry-pick | patch | Anonymous HTTP | SSH | HTTP |	

```
git fetch http://saros.freyther.de/gerrit/p/saros refs/changes/77/77/1 && git checkout FETCH_H
```

	File Path	Comments
	Commit Message	
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/net/util/RosterUtils.java	1 comment
M	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/ui/wizards/AddBuddyWizard.java	

Comments — Expand Recent | Expand A

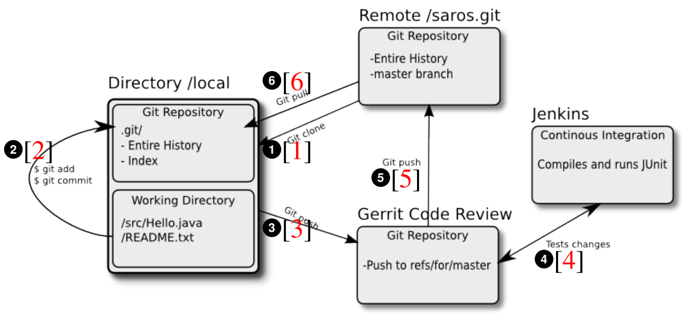
Sanity Bot Patch Set 1: Sanity review passed

Holger Freyther Patch Set 1: Looks good to me, approved (1 inline comment) Looks good

4.4 Summary

4.4.1 System Overview

In the previous sections we looked at the EGit and Gerrit integration and will now connect the dots between those systems. Once you have installed EGit you will need to *Git clone* the Saros repository, you can then make local changes, create branches, move files, etc. The local commits can be pushed from your local *Git repository* to the Gerrit system. The changes can now be reviewed in Gerrit by the Saros team. The Jenkins CI system will be triggered and will test your change and report back to Gerrit. Once the change is approved it can be submitted and will be merged into the official *Git repository*.



1. Clone a remote repository and checkout the default branch. This will create a local directory and populate the working directory.
2. Create local branches, add new files and make local commits. This will create local history in your Git repository.
3. Push your local branch to Gerrit for review. Gerrit will assign you a change number. The change can now be reviewed.
4. Jenkins will be notified about your change and will execute the test cases. The result of the tests will be reported back to Gerrit.
5. After a change is approved it can be submitted and will be integrated into the repository. Gerrit will push the changes to Github.
6. This will copy the current state of a remote repository into your local repository. Your current changes will move from the old last revision of the remote repository to the new one.

Figure 4.1: Code, Review and Test workflow

Chapter 5

Making changes to Saros

This chapter will show you how to work with EGit and Gerrit in the most common cases. This includes testing the code of someone else, making your first change, editing your change after having received review, updating your change to apply against the latest version of Saros. This will conclude with more advanced usage of having a chain of changes that build on top of each other and being asked to modify a change that is in the middle. This does not replace taking a look at the rather lengthy Eclipse EGit Wiki.

5.1 Cleaning your working copy

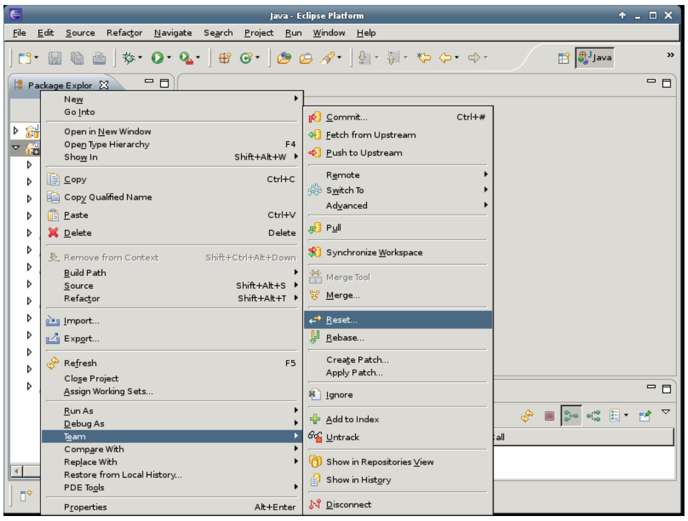
You might have created new classes and they are not under version version control yet and now you would like to switch the branch. Files not under version control will not be removed when switching branches. In some cases this might not be what you want and you could create a work in progress commit with the new files and later amend the commit with more changes and a better *Git commit message*. The other option is to remove files and this is what will be handled now.

5.1.1 Reverting local modifications

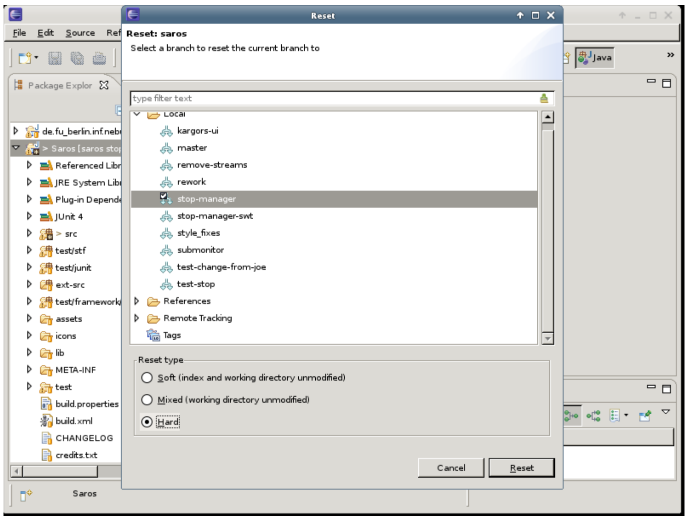
The easiest way to revert your un-comitted modifications is to use the *Git reset* feature. You want to use the *hard* option. This will reset the last commit of your history to the selected commit and will modify your working directory to match this commit.

1. Enter the context menu and select Team → Reset.
2. Select the branch/last commit. The default is the last commit. Select *hard* as the *reset type*.
3. Confirm the reset option.
4. The package explores does not indicate any modification.

1. ENTER CONTEXTMENU



2. Select reset type


Reset: saros

Select a branch to reset the current branch to

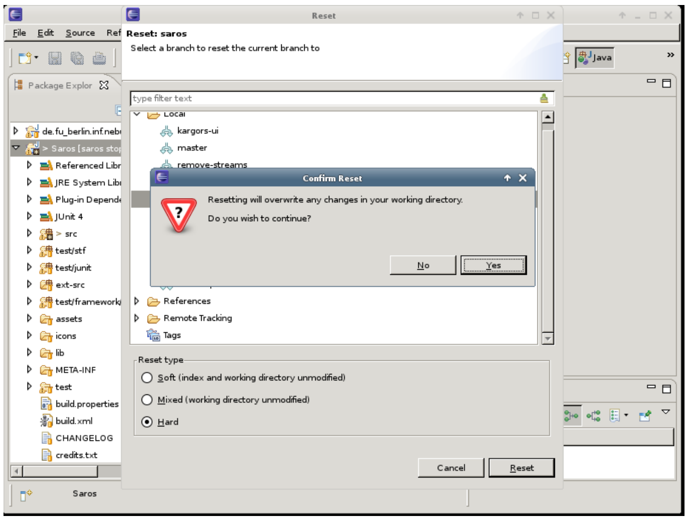
type filter text

- Local
  - kargors-ui
  - master
  - remove-streams
  - rework
  - stop-manager
  - stop-manager-swt
  - style_fixes
  - submonitor
  - test-change-from-joe
  - test-stop
- References
- Remote Tracking
- Tags

Reset type

- ( ) Soft (index and working directory unmodified)
- ( ) Mixed (working directory unmodified)
- [✓ Hard]

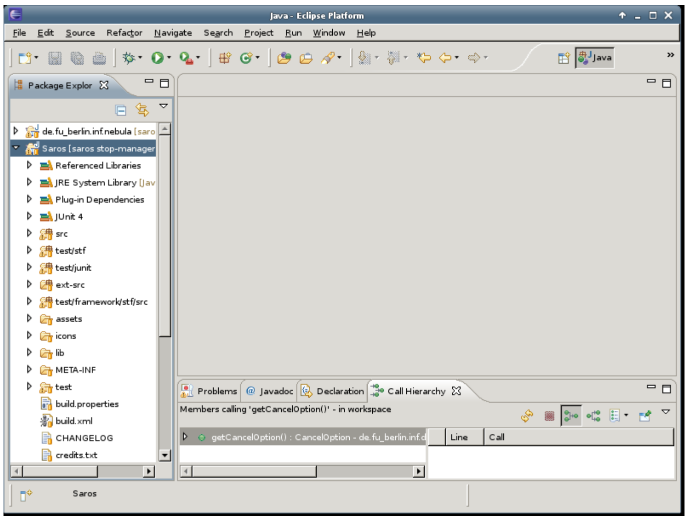
3. Confirm


Confirm Reset

Resetting will overwrite any changes in your working directory.

Do you wish to continue?

4. Clean tree

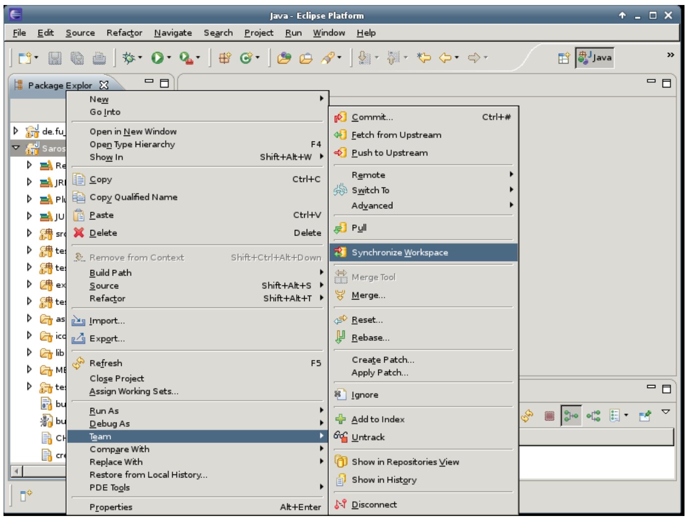


5.1.2 Deleting untracked/new files

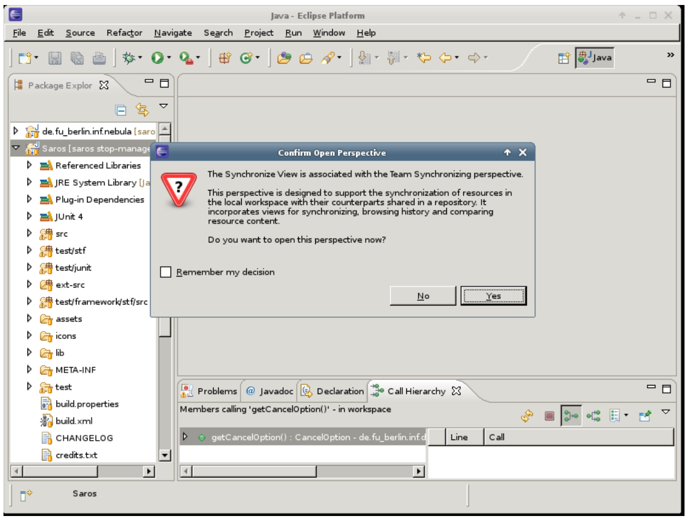
The *Git reset* feature does not remove files that are not under version control. The *Team Synchronizing* perspective will show you files not under version control. You can easily delete these files in this perspective.

1. Use the Contextmenu and then Team → Synchronize Workspace.
2. Confirm moving to the *Team Synchronizing* perspective.
3. Select the files you want to remove and use the Contextmenu to delete them.

1. ENTER CONTEXTMENU



2. SWITCH PERSPECTIVE


Java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explor
Problems Javadoc Declaration Call Hierarchy
Line Call
Saros

Confirm Open Perspective

The Synchronize View is associated with the Team Synchronizing perspective.

This perspective is designed to support the synchronization of resources in the local workspace with their counterparts shared in a repository. It incorporates views for synchronizing, browsing history and comparing resource content.

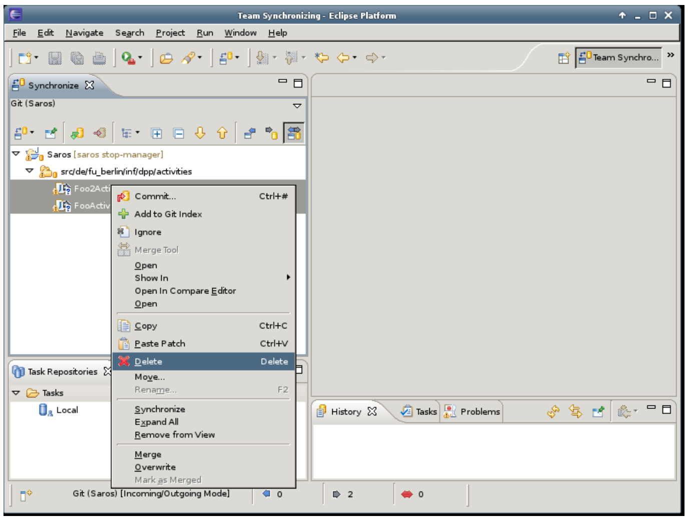
Do you want to open this perspective now?

Remember my decision

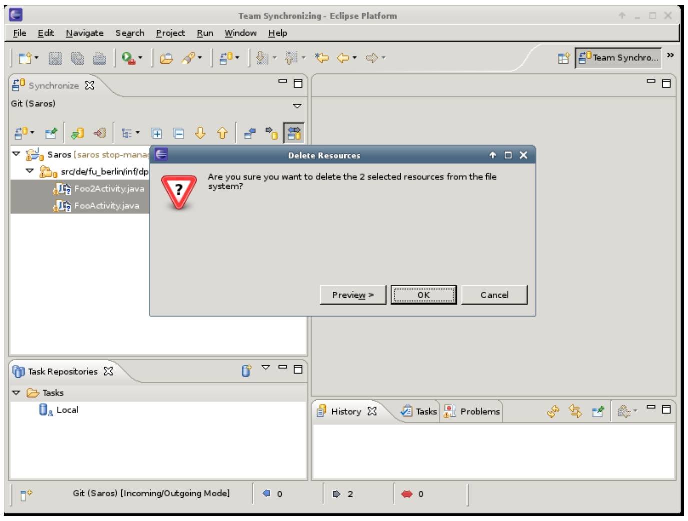
No Yes

Members calling 'getCancelOption()' - in workspace

3. Delete files



4. Confirm

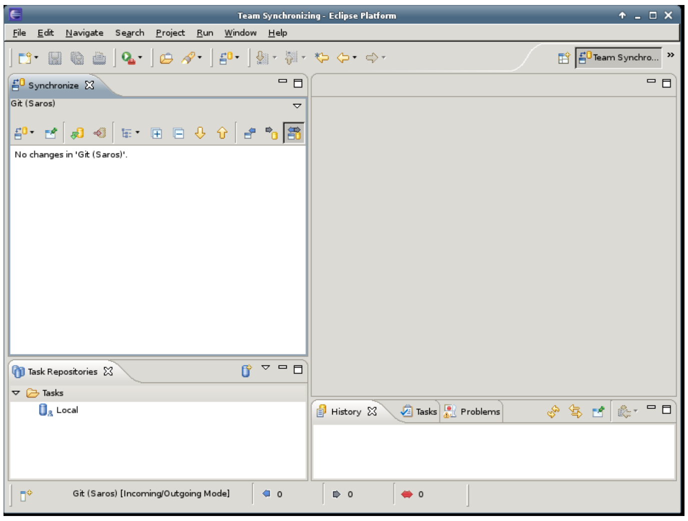

Team Synchronizing - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Synchronize
Git (Saros)
Saros [saros stop-manager]
src/de/fu_berlin/inf/dp
Foo2Activity.java
FooActivity.java
Task Repositories
Tasks
Local
History Tasks Problems
Git (Saros) [Incoming/Outgoing Mode]

Delete Resources

Are you sure you want to delete the 2 selected resources from the file system?

Preview > OK Cancel

5. No changes left


Team Synchronizing - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Team Synchro...
Synchronize
Git (Saros)
Task Repositories
Tasks
Local
History
Tasks
Problems
Git (Saros) [Incoming/Outgoing Mode]
0
0
0

No changes in 'Git (Saros)'.

5.2 Downloading a change

A member of your team has made some changes and uploaded them to Gerrit. You might want to test his change and the EGit plugin has support to make that very easy. The below will go through the necessary steps to fetch a change from Gerrit into a new local branch. The procedure assumes that you are in the *Git Perspective*.

5.2.1 Downloading a change (textual)

1. SELECT FETCH FROM GERRIT

   Enter the context menu on the *Git Repository* and select the Fetch From Gerrit.

2. SELECT A CHANGE BY CTRL+I

   You can enter the change directly or use CTRL+I to fetch a list of available changes an select it from a dropdown menu or enter it directly.

3. SELECT A CHANGE FROM THE DROPDOWN

   Select the change and the patchset you want. If you enter it directly use e.g. *refs/changes/47/47/3*, the format is to give the last two digits of the change, followed by the complete change number and then select the patchset you want.

4. CREATE A LOCAL BRANCH

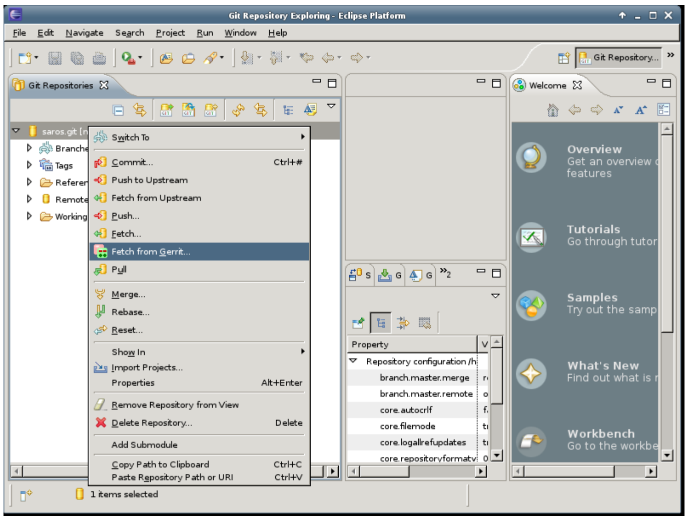
   Pick a local name and put it into the *Branch name* lineedit. This way you will have a local branch with the chosen name.

5. SWITCHED TO LOCAL BRANCH

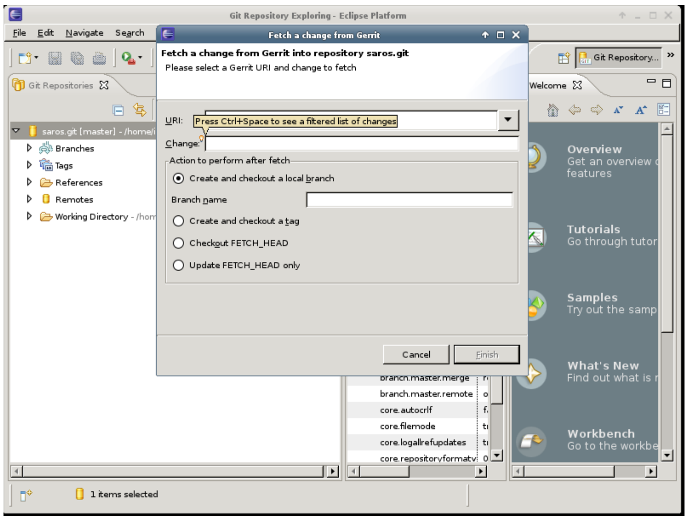
   EGit will indicate that it switched to your new branch.

5.2.2 Downloading a change (graphical)

1. SELECT FETCH FROM GERRIT



2. Select a change by CTRL+I


Fetch a change from Gerrit

Fetch a change from Gerrit into repository saros.git

Please select a Gerrit URI and change to fetch

URI: Press Ctrl+Space to see a filtered list of changes

Change:

Action to perform after fetch

(●) Create and checkout a local branch

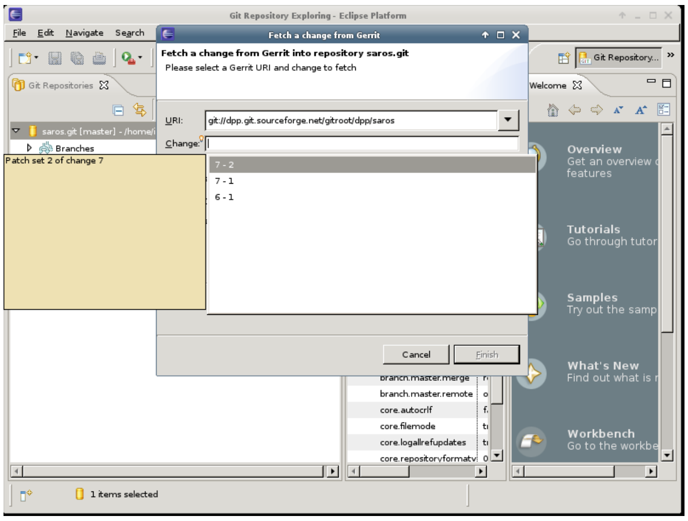
Branch name

( ) Create and checkout a tag

( ) Checkout FETCH_HEAD

( ) Update FETCH_HEAD only

3. Select a change from the dropdown


Fetch a change from Gerrit

Fetch a change from Gerrit into repository saros.git

Please select a Gerrit URI and change to fetch

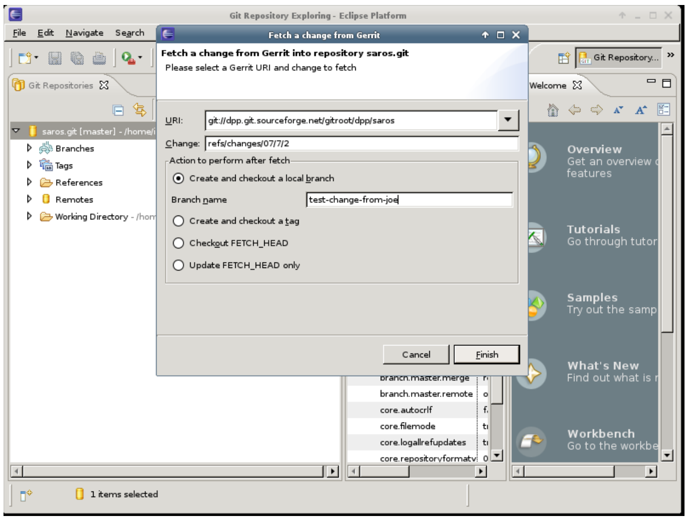
URI: git://dpp.git.sourceforge.net/gitroot/dpp/saros

Change:

7 - 2
7 - 1
6 - 1

Patch set 2 of change 7

4. Create a local branch



Fetch a change from Gerrit

Fetch a change from Gerrit into repository saros.git

Please select a Gerrit URI and change to fetch

URI: git://dpp.git.sourceforge.net/gitroot/dpp/saros

Change: refs/changes/07/7/2

Action to perform after fetch

[✓ Create and checkout a local branch]

Branch name: test-change-from-joe

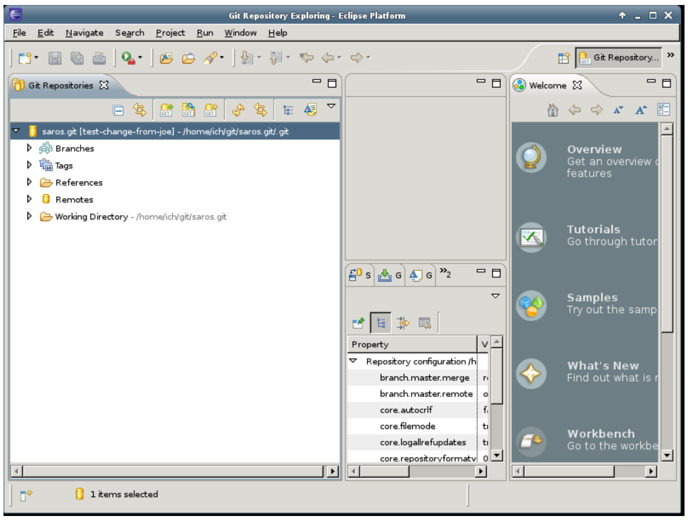
Create and checkout a tag

Checkout FETCH_HEAD

Update FETCH_HEAD only

Cancel | Finish

5. Switched to local branch



Git Repository Exploring - Eclipse Platform

saros.git [test-change-from-joe] - /home/ich/git/saros.git/.git

5.3 Doing your first change

Hopefully sooner than later you are ready to send your first contribution. You will need to create a Git commit and post it for review to our Gerrit installation. General information about using EGit can be found in the Eclipse EGit Wiki.

5.3.1 The first change (textual)

1. Open any file

   E.g. open Utils.java file.

2. Make a modification

   E.g. remove the *getFreePort* method.

3. Add change to the index

   Use the context menu of the file to access Team → Add to Index. This moves the change into the index and marks the new version as to be committed.

4. Commit

   Commit all changes that are in the index.

5. WRITING A COMMIT MESSAGE

   Make sure that the Gerrit option is enabled and that a line with *Change-Id* is in the editor field. Follow our Commit Guidelines for writing the message.

6. SWITCH TO THE REPOSITORY VIEW

   Use the context menu and Team → Show in Repository View to switch to another view to be able to push your changes.

7. PUSH

   Use the context menu on the repository and select the Push item.

8. SELECT TARGET

   Select the pre-configured Gerrit repository as the target for your changes and continue by pressing the Next button.

9. SELECT BRANCH

   Select what and where to push things. The default Destination Ref should be *refs/for/master* and press Next to continue.

10. PUSH CONFIRMATION

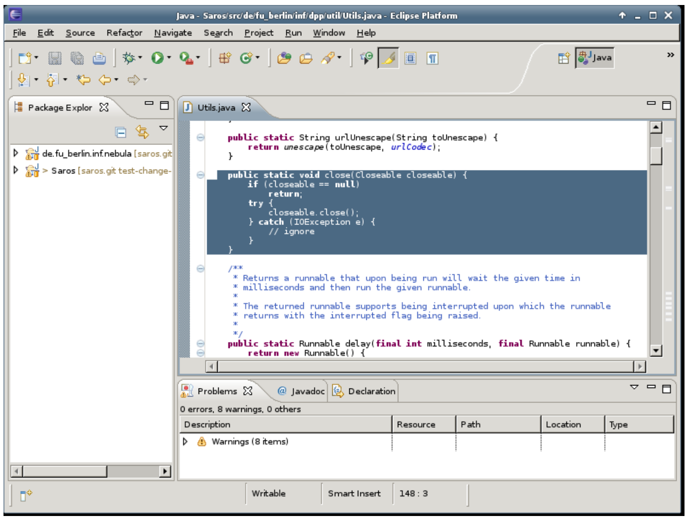
    Review what and where things are pushed and Finish .

11. PUSH RESULT

    The result should include a set of change numbers of the Gerrit system.

5.3.2 The first change (graphical)

1. OPEN A UTILS.JAVA

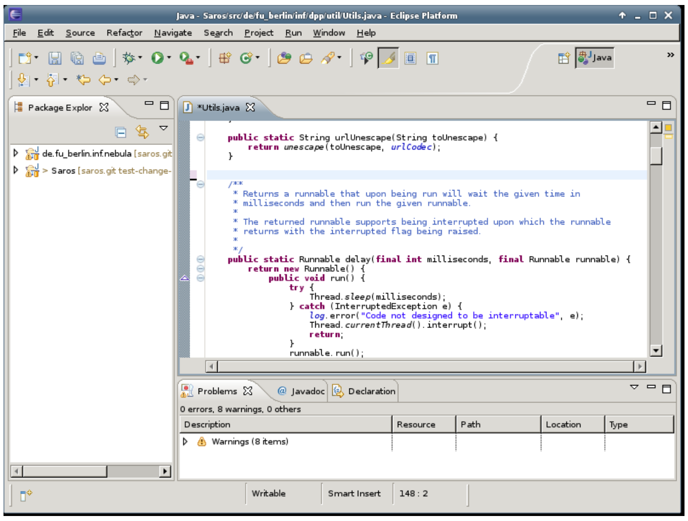

Java - Saros/src/de/fu_berlin/inf/dpp/util/Utils.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Package Explor
de.fu_berlin.inf.nebula [saros.git
> Saros [saros.git test-change-
Utils.java
Problems
Javadoc
Declaration
Writable
Smart Insert
148 : 3

```
    public static String urlUnescape(String toUnescape) {
        return unescape(toUnescape, urlCodec);
    }

    public static void close(Closeable closeable) {
        if (closeable == null)
            return;
        try {
            closeable.close();
        } catch (IOException e) {
            // ignore
        }
    }

    /**
     * Returns a runnable that upon being run will wait the given time in
     * milliseconds and then run the given runnable.
     *
     * The returned runnable supports being interrupted upon which the runnable
     * returns with the interrupted flag being raised.
     *
     */
    public static Runnable delay(final int milliseconds, final Runnable runnable) {
        return new Runnable() {
```

0 errors, 8 warnings, 0 others

Description	Resource	Path	Location	Type
Warnings (8 items)				

2. MAKE A MODIFICATION

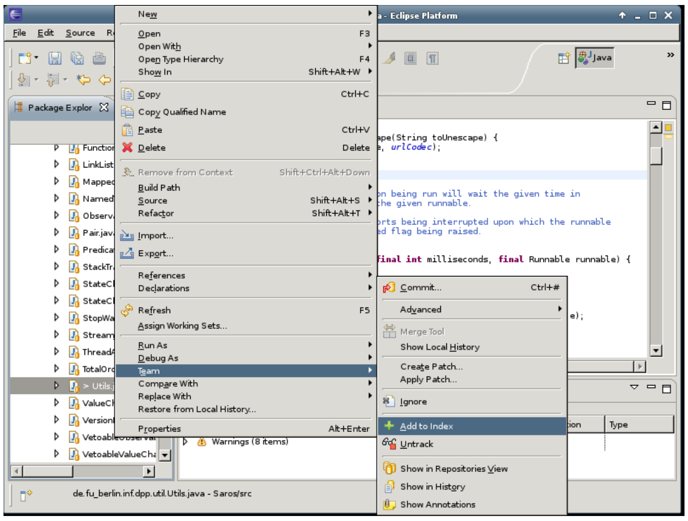


```
    public static String urlUnescape(String toUnescape) {
        return unescape(toUnescape, urlCodec);
    }


    /**
     * Returns a runnable that upon being run will wait the given time in
     * milliseconds and then run the given runnable.
     *
     * The returned runnable supports being interrupted upon which the runnable
     * returns with the interrupted flag being raised.
     *
     */
    public static Runnable delay(final int milliseconds, final Runnable runnable) {
        return new Runnable() {
            public void run() {
                try {
                    Thread.sleep(milliseconds);
                } catch (InterruptedException e) {
                    log.error("Code not designed to be interruptable", e);
                    Thread.currentThread().interrupt();
                    return;
                }
                runnable.run();
```

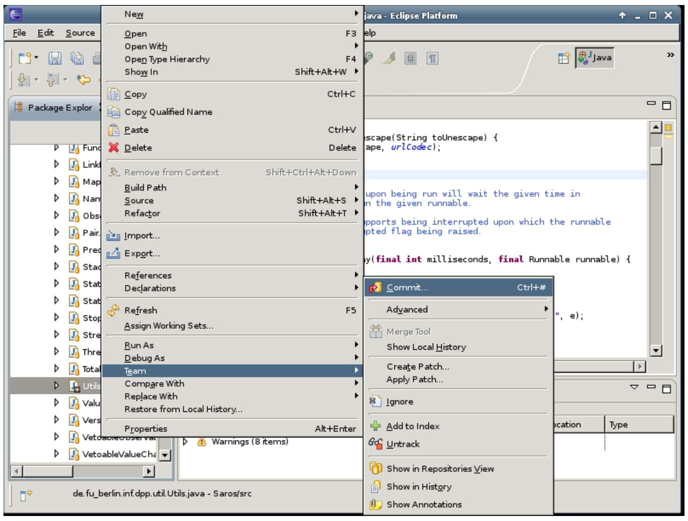
0 errors, 8 warnings, 0 others

Description	Resource	Path	Location	Type
Warnings (8 items)				

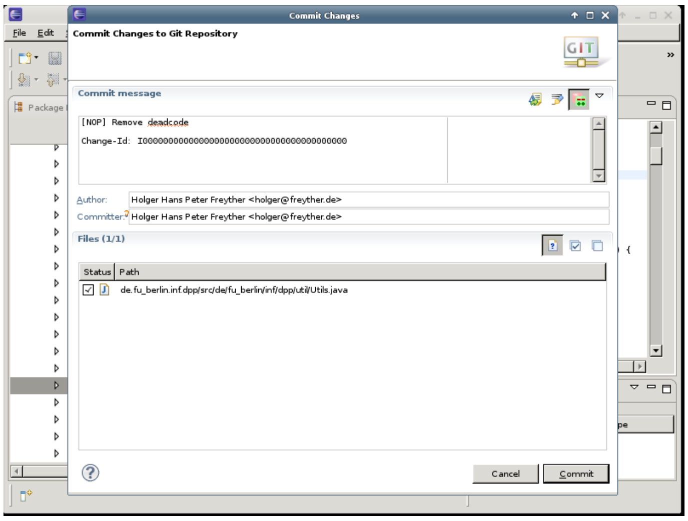
3. ADD CHANGE TO THE INDEX



4. COMMIT



5. WRITING A COMMIT MESSAGE

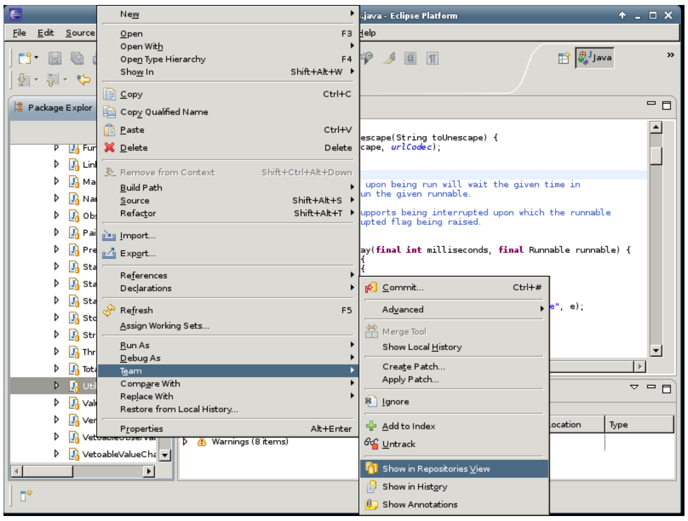


```
[NOP] Remove deadcode

Change-Id: I0000000000000000000000000000000000000000
```

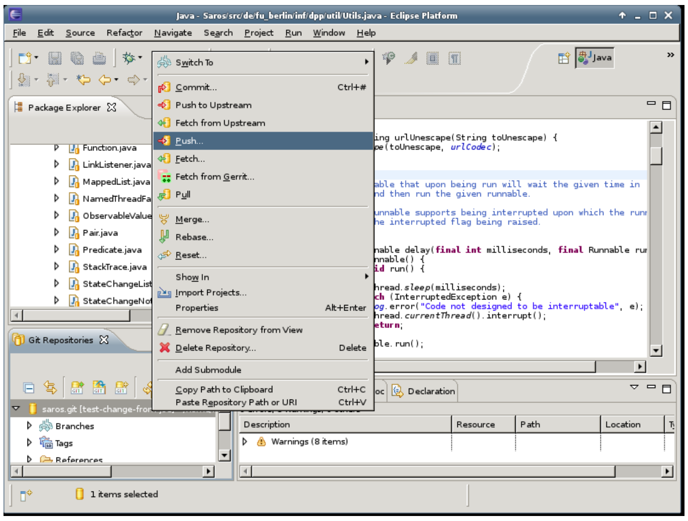
Author: Holger Hans Peter Freyther <holger@freyther.de>
Committer: Holger Hans Peter Freyther <holger@freyther.de>

Status	Path
✓	de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/util/Utils.java

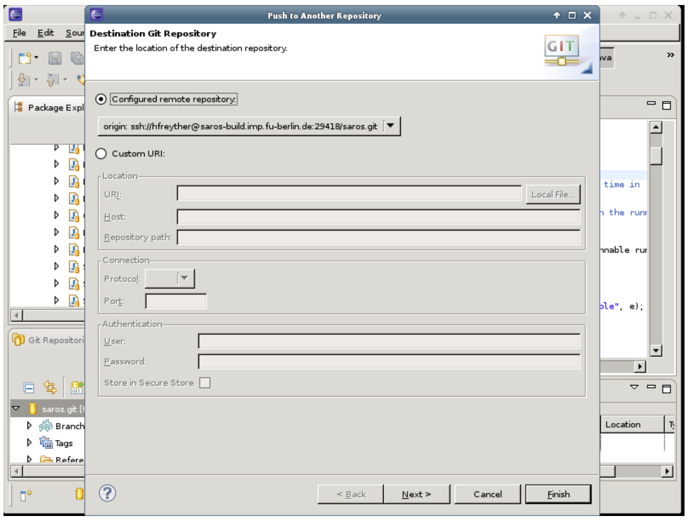
6. SWITCH TO THE REPOSITORY VIEW



7. PUSH



8. SELECT TARGET


Push to Another Repository

Destination Git Repository

Enter the location of the destination repository.

[✓ Configured remote repository:]

origin: ssh://hfreyther@saros-build.imp.fu-berlin.de:29418/saros.git

Custom URI:

Location

URI:

Host:

Repository path:

Connection

Protocol:

Port:

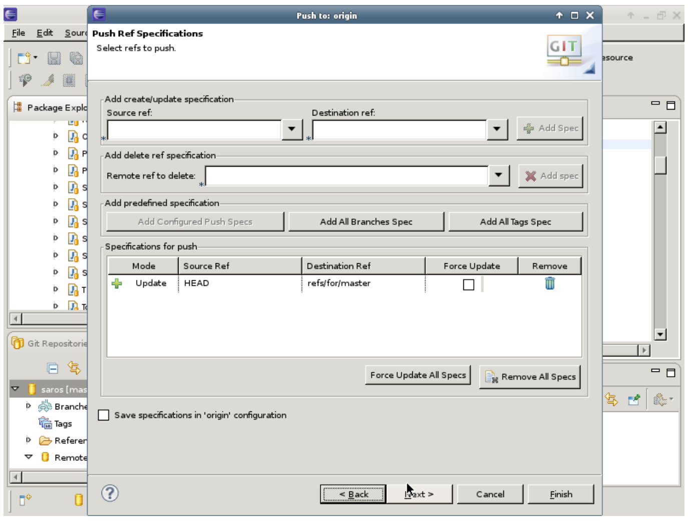
Authentication

User:

Password:

Store in Secure Store

9. Select Branch


Push to: origin

Push Ref Specifications

Select refs to push.

Add create/update specification

Source ref:

Destination ref:

Add delete ref specification

Remote ref to delete:

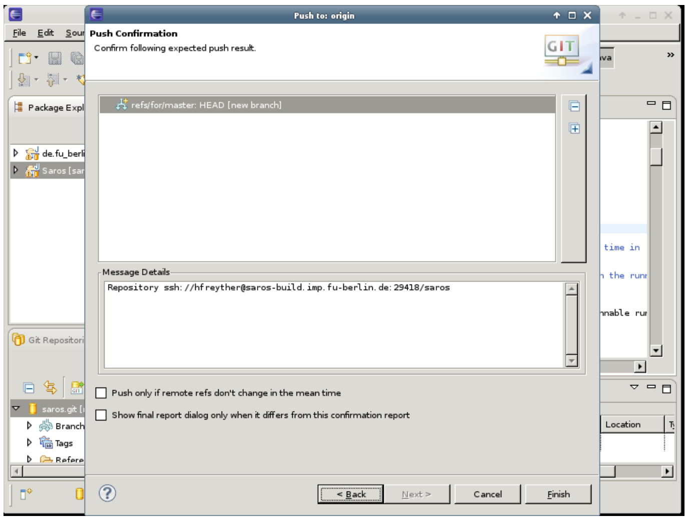
Add predefined specification

Specifications for push

Mode	Source Ref	Destination Ref	Force Update	Remove
Update	HEAD	refs/for/master	☐	

Save specifications in 'origin' configuration

10. Push Confirmation


Push to: origin

Push Confirmation

Confirm following expected push result.

refs/for/master: HEAD [new branch]

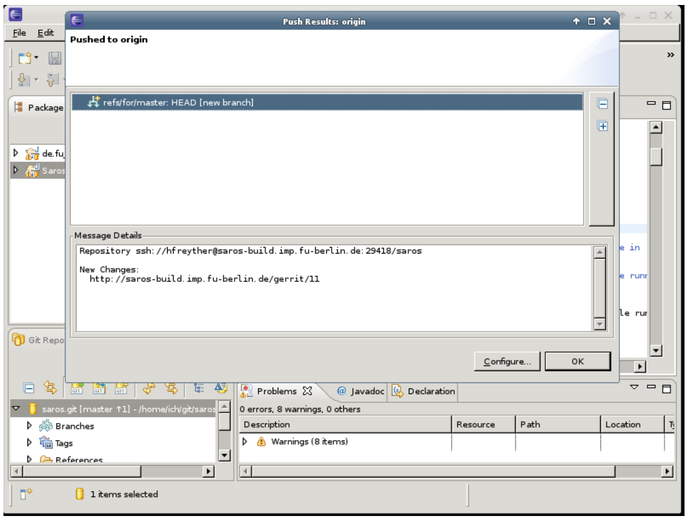
Message Details

```
Repository ssh://hfreyther@saros-build.imp.fu-berlin.de:29418/saros
```

Push only if remote refs don't change in the mean time

Show final report dialog only when it differs from this confirmation report

11. Push Result



```
Repository ssh://hfreyther@saros-build.imp.fu-berlin.de:29418/saros

New Changes:
  http://saros-build.imp.fu-berlin.de/gerrit/11
```

5.3.3 Understanding build failures

After you pushed your changes to Gerrit, the Saros-Gerrit job will be scheduled on the Jenkins CI system. The execution of this job might fail and the most common errors are missing files, build failures due building on an older OSGI/Eclipse version or test failures because the tests are run without SWT/GUI being initialized. The below will help you to identify the reason of the failure.

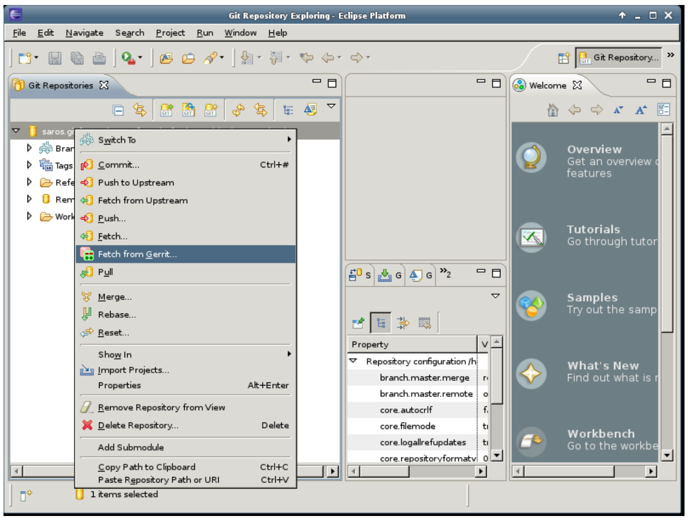
In case the build has failed it can be difficult to find the error in the ant log. The easiest is to search for . *ERROR* in the console log of the job. An example for a build failure is Job 24, try to find the error.

5.3.4 Dealing with feedback

Imagine you made three commits on top of each other and have pushed them to Gerrit. Now either you or a reviewer has found something that should be modified in any of the separate changes. The straightforward way is to use the built-in Gerrit support to create a new branch with these changes, make the desired modifications, *amend* to the old commit and push it to Gerrit again.

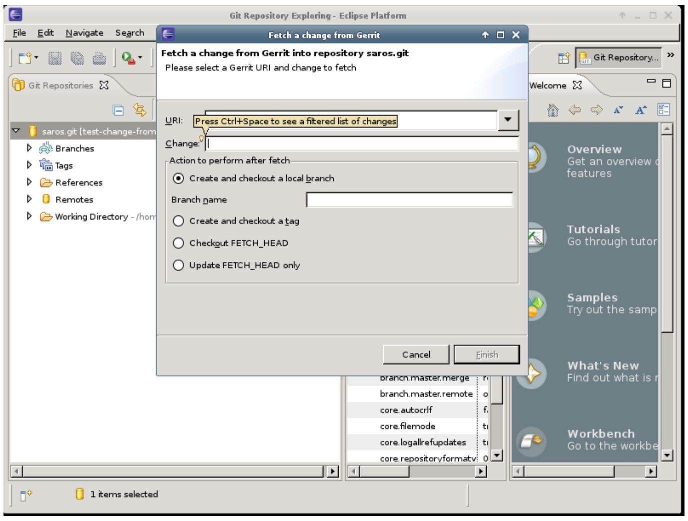
1. FETCH FROM GERRIT

   From within the *Git repository* view select the context menu and press Fetch from Gerrit



2. SELECT A CHANGE

   You will need to select a change or use Control+Space to search for changes.

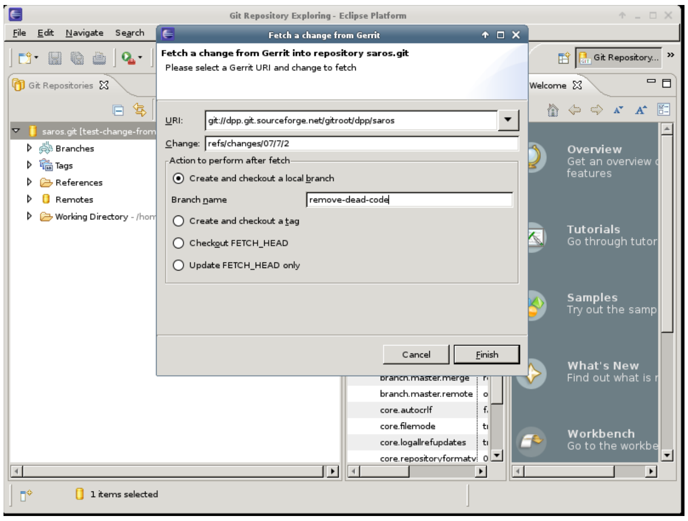


Fetch a change from Gerrit into repository saros.git
Please select a Gerrit URI and change to fetch

URI: Press Ctrl+Space to see a filtered list of changes
Change:
Action to perform after fetch
(●) Create and checkout a local branch
Branch name
( ) Create and checkout a tag
( ) Checkout FETCH_HEAD
( ) Update FETCH_HEAD only

3. Pick a local branch name

   Select a name for the local branch.

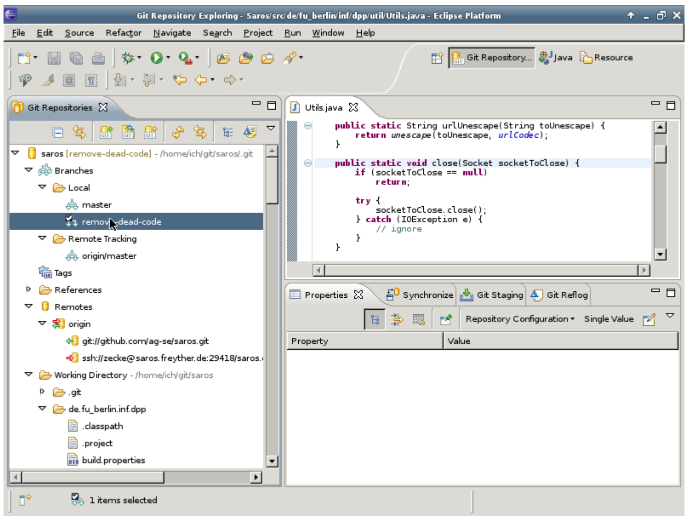


Fetch a change from Gerrit into repository saros.git
Please select a Gerrit URI and change to fetch

URI: git://dpp.git.sourceforge.net/gitroot/dpp/saros
Change: refs/changes/07/7/2
Action to perform after fetch
(●) Create and checkout a local branch
Branch name remove-dead-code
( ) Create and checkout a tag
( ) Checkout FETCH_HEAD
( ) Update FETCH_HEAD only

4. Start your modification

   You are on a branch that contains your previous change. You are now able to do the desired modification.

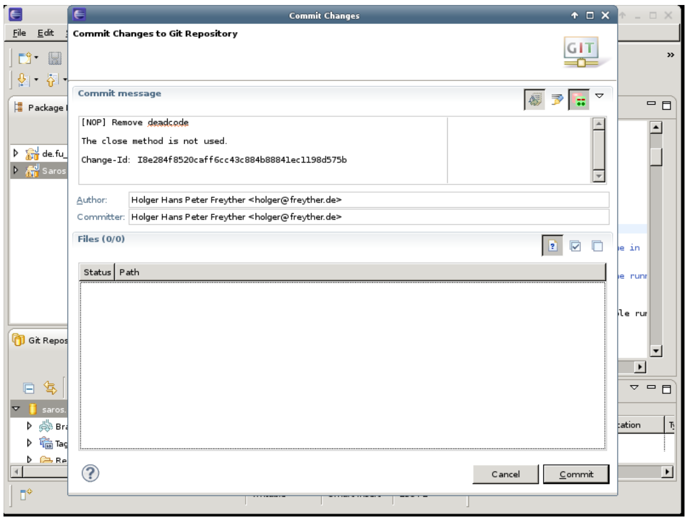


Git Repository Exploring - Saros/src/de/fu_berlin/inf/dpp/util/Utils.java - Eclipse Platform

```
public static String urlUnescape(String toUnescape) {
    return unescape(toUnescape, urlCodec);
}

public static void close(Socket socketToClose) {
    if (socketToClose == null)
        return;

    try {
        socketToClose.close();
    } catch (IOException e) {
        // ignore
    }
}
```

5. Amending your commit

Once you are done with your modification, enter the commit editor and select the amend button. This will combine your new and old change. The commit dialog will automatically pick the commit message from the previous commit.

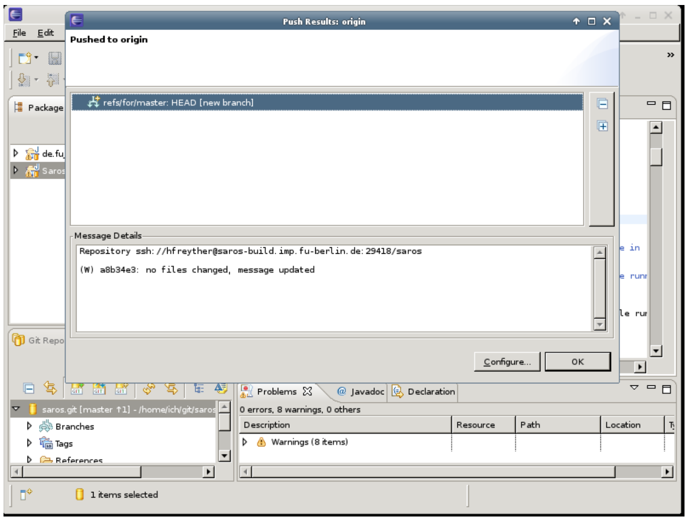


```
[NOP] Remove deadcode

The close method is not used.

Change-Id: I8e284f8520caff6cc43c884b88841ec1198d575b
```

Author: Holger Hans Peter Freyther <holger@freyther.de>
Committer: Holger Hans Peter Freyther <holger@freyther.de>

6. PUSHING YOUR COMMITS

   If you push your current branch to Gerrit again, Gerrit will update the patches in the changes.



refs/for/master: HEAD [new branch]

```
Repository ssh://hfreyther@saros-build.imp.fu-berlin.de:29418/saros

(W) a8b34e3: no files changed, message updated
```

7. DELETING A BRANCH

   You can now checkout a different branch and delete the old one.

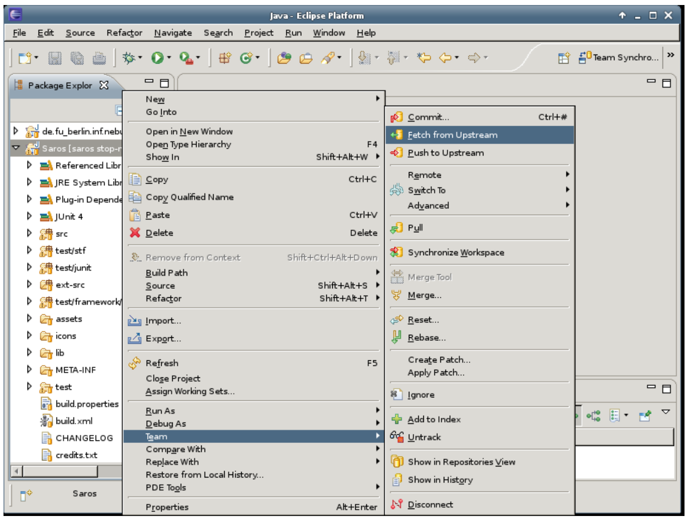
5.4 Updating your changes

5.4.1 Fetch, Rebase and Push

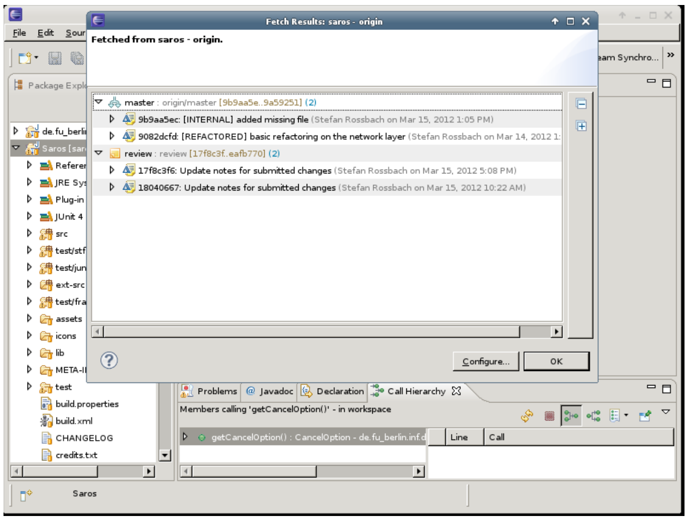
You have started your work, created a local branch based on the last commit of *origin/master*. During the time you made your changes and now some other changes were integrated into *origin/master* and you want to update. This can be done by combining two operations. The first is *Git fetch* to fetch all changes from the *origin* into your local *Git repository*. This operation will update the *origin/master* branch to the state of the remote *Git repository*. The second operation is *Git rebase*. This will move any local commits to the new *origin/master*. The operation is called *Git rebase* because your local changes will be moved from the old state of *origin/master* to the current state of it. During the *Git rebase* local commits can vanish as they are already included, or you can end with a conflict. In that case you will be presented with multiple options, to abort the rebase, to skip (remove) the commit that caused the conflict or to attempt to merge it using the mergetool. Once you have resolved the conflicts use Team → Rebase → Continue to continue with the next local *Git commit*..

1. Enter the Contextmenu, select Team → Fetch from Upstream.
2. An overview with changes appear
3. Enter the Contextmenu, select Team → Rebase.
4. Select the branch you want to rebase against.
5. On conflict a dialog will ask you to make a decision on how to resolve it. Once it is resolved selectTeam → Rebase → Continue in the contextmenu to continue with the next commit.
6. Your branch is now based on the new *origin/master*

1. FETCH FROM UPSTREAM



2. RESULT

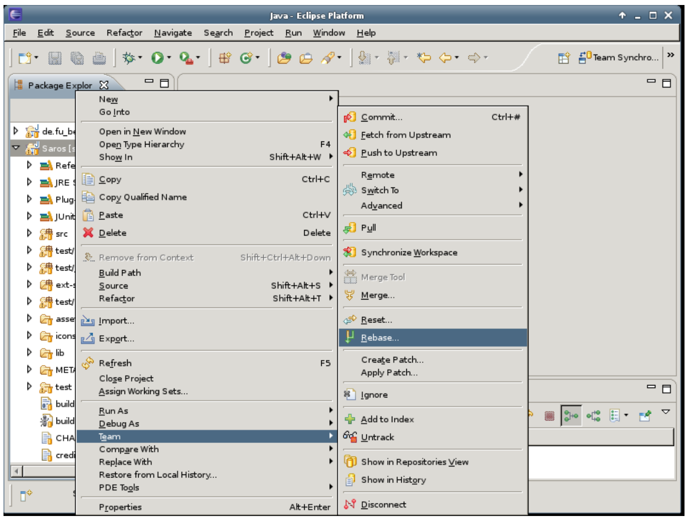

Fetch Results: saros - origin

Fetched from saros - origin.

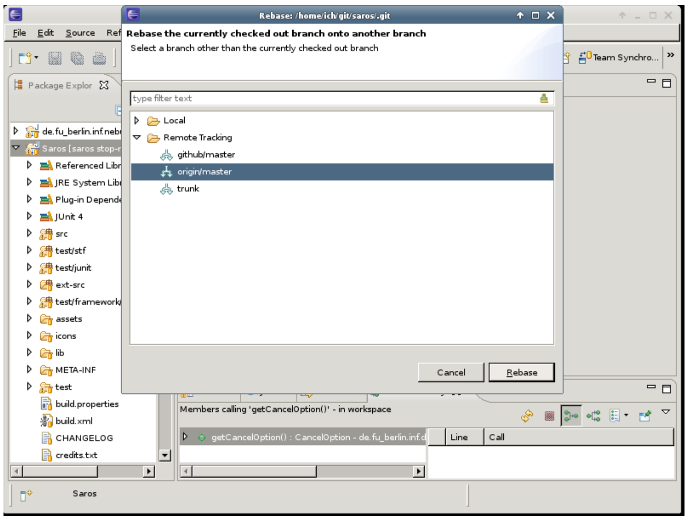
```
master : origin/master [9b9aa5e..9a59251] (2)
    9b9aa5ec: [INTERNAL] added missing file (Stefan Rossbach on Mar 15, 2012 1:05 PM)
    9082dcfd: [REFACTORED] basic refactoring on the network layer (Stefan Rossbach on Mar 14, 2012 1:
review : review [17f8c3f..eafb770] (2)
    17f8c3f6: Update notes for submitted changes (Stefan Rossbach on Mar 15, 2012 5:08 PM)
    18040667: Update notes for submitted changes (Stefan Rossbach on Mar 15, 2012 10:22 AM)
```

Configure...
OK
Problems
Javadoc
Declaration
Call Hierarchy
Members calling 'getCancelOption()' - in workspace
getCancelOption() : CancelOption - de.fu_berlin.inf.d
Line
Call
Saros

3. BEGIN REBASE



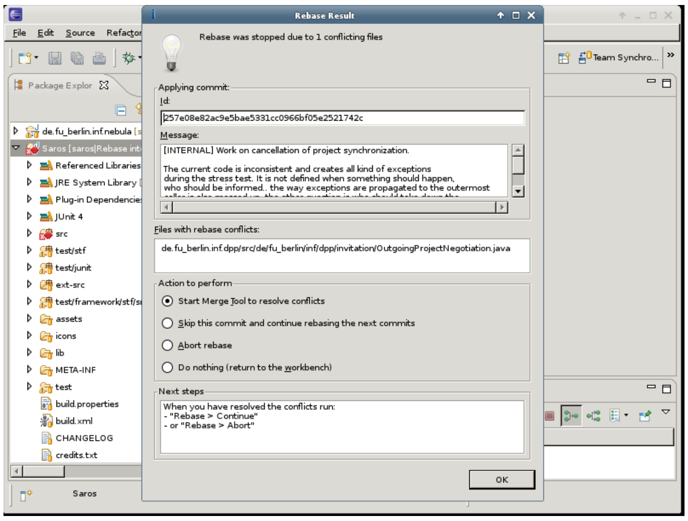
4. SELECT BRANCH


Rebase: /home/ich/git/saros/.git

Rebase the currently checked out branch onto another branch

Select a branch other than the currently checked out branch

5. CONFLICT


Rebase Result

Rebase was stopped due to 1 conflicting files

Applying commit:

Id: 257e08e82ac9e5bae5331cc0966bf05e2521742c

Message:

[INTERNAL] Work on cancellation of project synchronization.

The current code is inconsistent and creates all kind of exceptions
during the stress test. It is not defined when something should happen,
who should be informed.. the way exceptions are propagated to the outermost

Files with rebase conflicts:

de.fu_berlin.inf.dpp/src/de/fu_berlin/inf/dpp/invitation/OutgoingProjectNegotiation.java

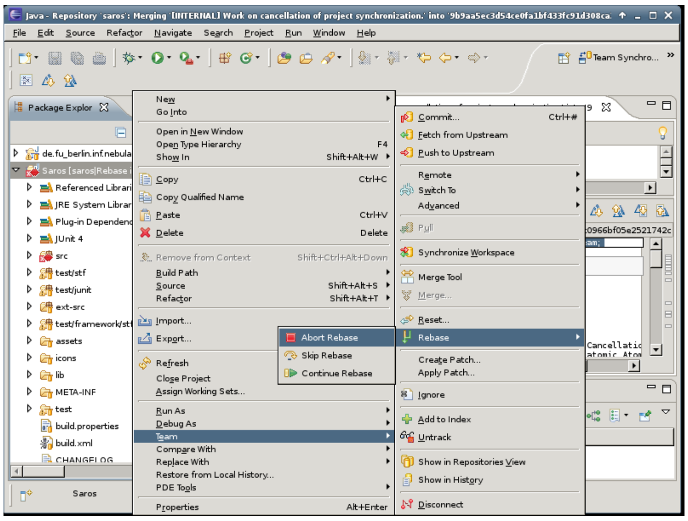
Action to perform

[✓ Start Merge Tool to resolve conflicts]
Skip this commit and continue rebasing the next commits
Abort rebase
Do nothing (return to the workbench)

Next steps

When you have resolved the conflicts run:
- "Rebase > Continue"
- or "Rebase > Abort"

6. ABORT



When the rebase goes wrong
When the rebase goes wrong you have two options. You can either select Team → Rebase → Abort or if you notice after the rebase the *Git Reflog* contains your changes before the *Git rebase* and you can checkout this version again.

External Git mergetool
On OSX and Linux you can use *git mergetool* on the command line to use opendiff/kdiff3/meld to the merging.

5.5 Advanced Git Topics

You might have started to use *Git* beyond the featureset of SVN and have a chain of local commits. You have asked to review your chain of changes and a peer has found an issue in of the changes in the middle of the chain. From the previous chapters you have seen how to the modification of a single change and this is what you need to do first, the next thing is to rebase the commits that followed the original one. And below there will be two alternatives for doing that.

5.5.1 Cherry-pick

TODO: describe cherry-pick

5.5.2 Staging Changes

TODO

5.6 Tasks during a release

For the release two additional tasks are required. The first is to create the branch and the second is to push bugfixes to the right branch. In both cases this is done by selecting another branch in the destination branch when pushing your changes. In case of creating the release branch the right destination branch is *refs/heads/release/DATE*, asking for review is done through the *refs/for/release/DATE* branch.

Task	Destination Reference
Create Release Branch	refs/*heads*/release/DATE
Change for Release	refs/*for*/release/DATE

Table 5.1: Destination Reference

5.6.1 Pushing to a non default branch (textual)

1. SELECT PUSH IN THE CONTEXT MENU

   Inside the Git repository view activate the context menu and select the Push item.

2. SELECT THE CONFIGURED REMOTE REPOSITORY

   The configured remote repository is selected by default, move to the next page by pressing the Next button.

3. SPECIFY THE BRANCH NAME

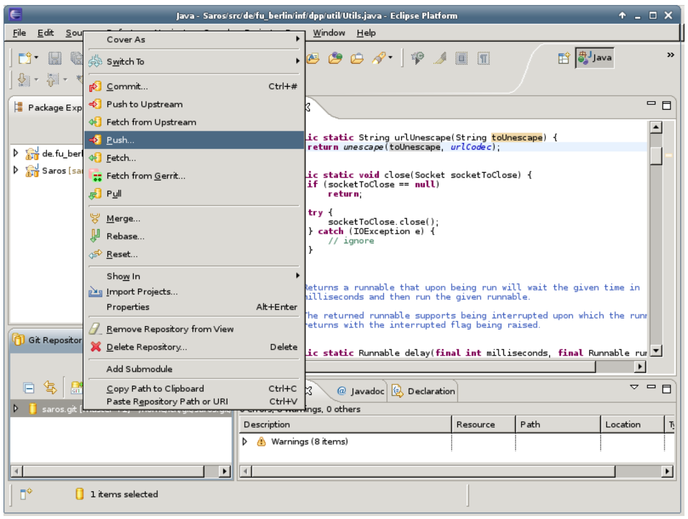
   Inside the *Specifications for push* area change *refs/for/master* to the branch name mentioned above and continue to the next page.

4. PUSH CONFIRMATION

   Confirm the push and finish the dialog

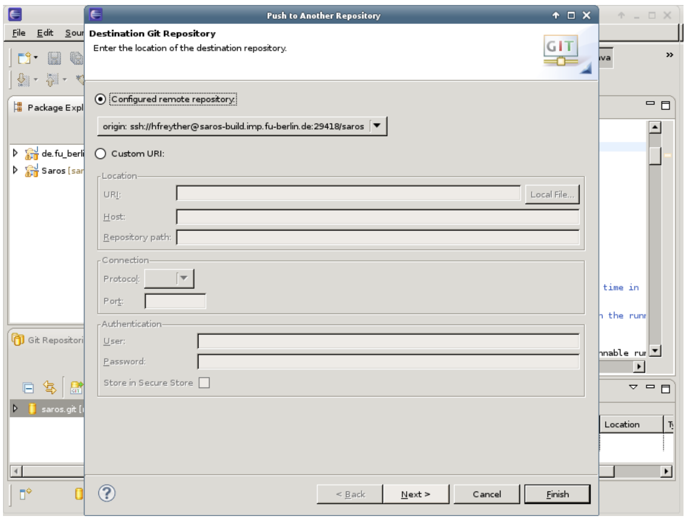
5.6.2 Pushing to a non default branch (graphical)

1. SELECT PUSH IN THE CONTEXT MENU



```
ic static String urlUnescape(String toUnescape) {
return unescape(toUnescape, urlCodec);

ic static void close(Socket socketToClose) {
if (socketToClose == null)
    return;

try {
    socketToClose.close();
} catch (IOException e) {
    // ignore
}

Returns a runnable that upon being run will wait the given time in
illiseconds and then run the given runnable.

The returned runnable supports being interrupted upon which the runr
eturns with the interrupted flag being raised.

ic static Runnable delay(final int milliseconds, final Runnable rur
```

2. SELECT THE CONFIGURED REMOTE REPOSITORY



Push to Another Repository

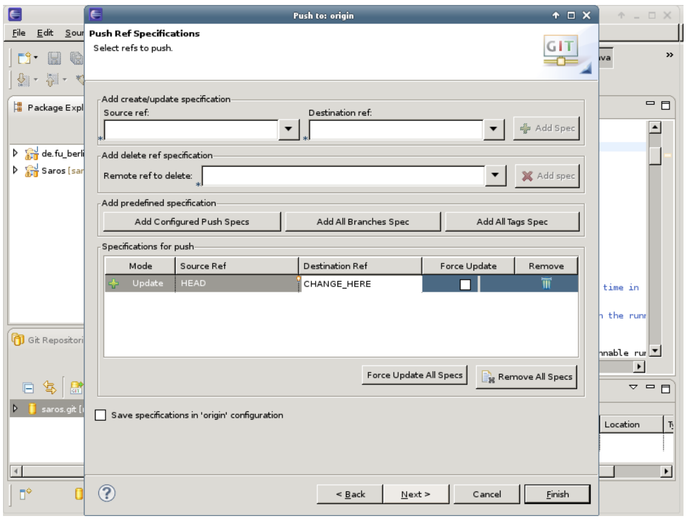
Destination Git Repository

Enter the location of the destination repository.

[✓ Configured remote repository:] origin: ssh://hfreyther@saros-build.imp.fu-berlin.de:29418/saros

Custom URI:

3. SPECIFY THE BRANCH NAME



Push to: origin

Push Ref Specifications

Select refs to push.

Mode	Source Ref	Destination Ref	Force Update	Remove
Update	HEAD	CHANGE_HERE	☐	

4. PUSH CONFIRMATION

Confirm the push and finish the dialog

Chapter 6

Patterns and Anti-Patterns

There are various patterns we want to be used and there have been various anti-patterns we don't want to see again. This section provides an illustration of what is good and what should be avoided and give reasons about it.

Under Construction
Not completed yet.

6.1 Anti-Patterns

6.1.1 PicoContainer usage

The PicoContainer will resolve dependencies between objects. For this to work properly one needs to have all dependencies as parameters of the constructors. One common misuse is to get a dependency with an indirection.

```
public class SkypeManager {
    private final SarosNet sarosNet;
    private final IPreferenceStore preferences;


    public SkypeManager(Saros saros) {
        this.sarosNet = saros.getSarosNet();
        this.preferences = saros.getPreferenceStore();
    }
}
```

The above code has dependencies on SarosNet and IPreferenceStore and they are resolved through the Saros instance. This is a problem when it comes to testing, there might not be a mock for the SarosNet or the IPreferenceStore store and one might have an NullPointerException. The below code shows how to properly express the dependencies.

```
public class SkypeManager {
    private final SarosNet sarosNet;
    private final IPreferenceStore preferences;


    public SkypeManager(SarosNet sarosNet, IPreferenceStore preferences) {
        this.sarosNet = sarosNet;
        this.preferences = preferences;
    }
}
```

6.1.2 Use Interfaces

TODO: E.g. ISarosSession vs. Session, talk about testability

6.1.3 Splitting Work

Describe.. splitting work across different threads and synch. issues

6.1.4 ...

MORE

Chapter 7

Configuration of Gerrit

This is mostly a guide for the project owner. Our Gerrit has some project specific configuration that is explained in this chapter.

7.1 Grant push access to Gerrit

Users in the Approved group are allowed to push. Add new users to this group.

7.2 Granting +2/-2 and the right to submit

Users in the Approvers group are allowed to set +2/-2 in the review category, override the verified ranking and are allowed to submit approved changes.

7.3 Granting branch creation

Users in the Release-Managers group are allowed to push merges for review and to create new branches.