

Git with Eclipse (EGit) - Tutorial

Lars Vogel

Version 2.7

Copyright © 2009, 2010, 2011 Lars Vogel

09.11.2011

Revision History

Revision 0.1	11.12.2009	Lars Vogel
Created		
Revision 0.2 - 2.7	12.12.2009 - 09.11.2011	Lars Vogel
bug fixes and enhancements		

Git with Eclipse (EGit)

This tutorial describes the usage of EGit; an Eclipse plugin to use the distributed version control system Git. This tutorial is based on Eclipse 3.7 (Indigo).

Table of Contents

- 1. Overview
- 2. EGit Installation
- 3. Prerequisites for this tutorial
- 4. Putting projects under version control
 - 4.1. Put a new project under version control
 - 4.2. Clone existing project
 - 4.3. Repository view
- 5. Using EGit
 - 5.1. Basic operations
 - 5.2. Merge
 - 5.3. View the resource history
- 6. Create a Git repository for multiple projects
- 7. Using EGit with Github
 - 7.1. Github
 - 7.2. Create Repository in Github
 - 7.3. Push project to Github
 - 7.4. Clone repository from Github
 - 7.5. Mylyn integration with Github
- 8. Hacking EGit - Getting the source code
- 9. Thank you
- 10. Questions and Discussion
- 11. Links and Literature
 - 11.1. EGit and Git Resources

11.2. vogella Resources

1. Overview

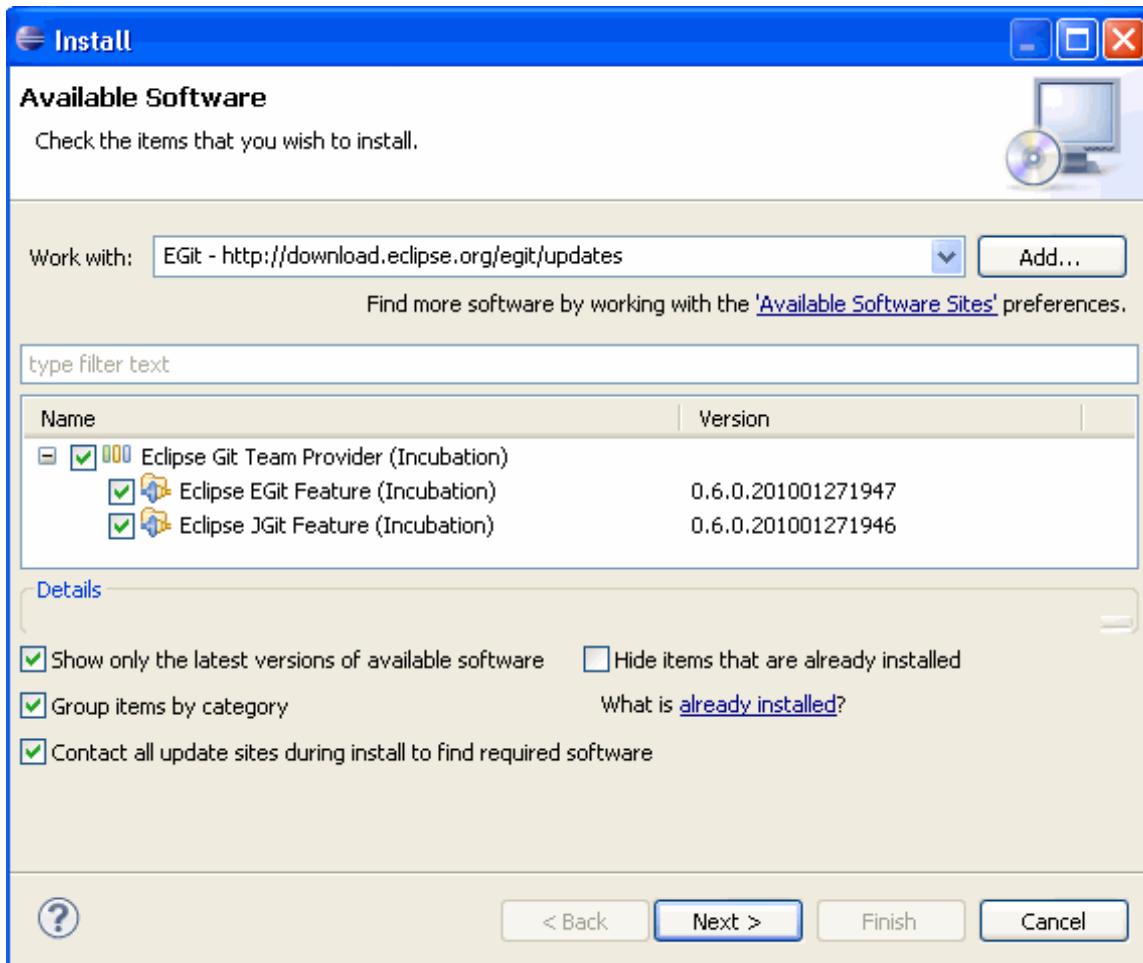
Git is a distributed version control system written in C. JGit is a library which implements the Git functionality in Java. EGit is an Eclipse Team provider for Git (using JGit) and makes Git available in the Eclipse IDE.

This tutorial describes the usage of EGit. If you want to learn about the usage of the Git command line you can use the [Git Tutorial](#) as a reference.

2. EGit Installation

Eclipse 3.7 contains EGit in its default configuration. So typically you do not need to install anything.

If you use an older version of Eclipse you can use the [Eclipse Update manager](#) to install the EGit plugin from <http://download.eclipse.org/egit/updates>. The latest features can be found in the nightly build but this build is not as stable as the official update site target <http://download.eclipse.org/egit/updates-nightly>.



3. Prerequisites for this tutorial

This article assumes what you have basic understanding of development for the Eclipse platform. Please see [Eclipse RCP Tutorial](#) or [Eclipse Plugin Tutorial](#).

4. Putting projects under version control

The following section explains how to create a repository for one project and shows how to checkout an exiting projects from a remote repository.

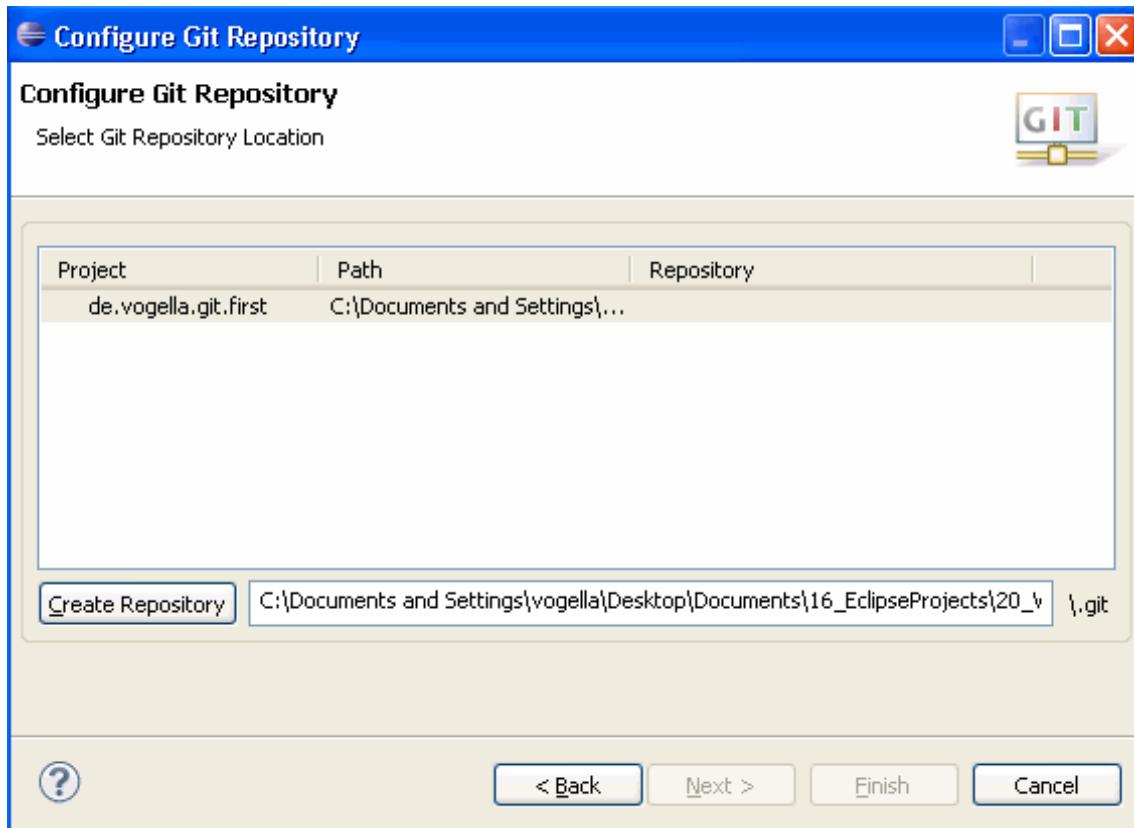
4.1. Put a new project under version control

Create a new Java project "de.vogella.git.first" in Eclipse. Create the following class.

```
package de.vogella.git.first;

public class GitTest {
    public static void main(String[] args) {
        System.out.println("Git is fun");
    }
}
```

Right click your project, select Team -> Share Project -> Git. Select the proposed line and press "Create repository". Press finish.



You have created a local Git repository. The git repository is in this case directly stored in the project in a ".git" folder.

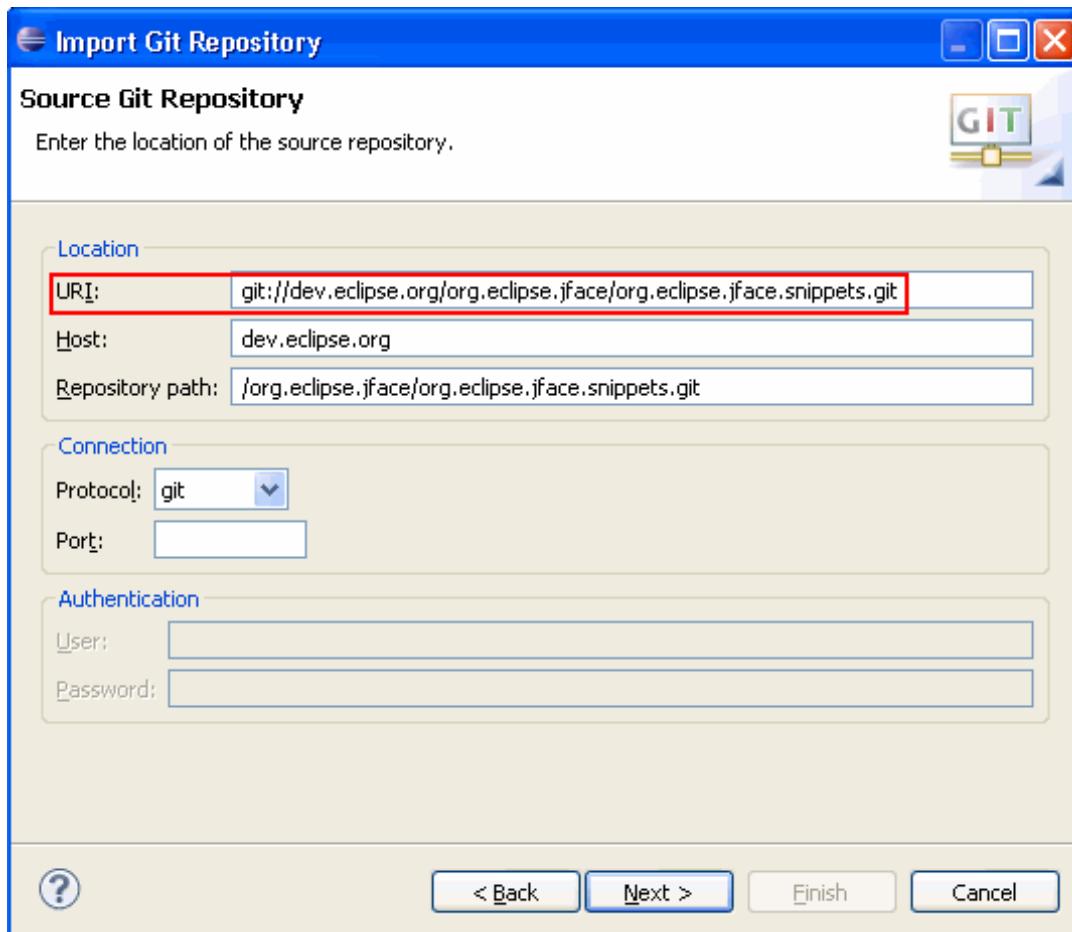
Create the file ".gitignore" in your project with the following content. All files / directories which apply to the pattern described in this file will be ignored by git. In this example all files in the "bin" directory will be ignored.

```
bin
```

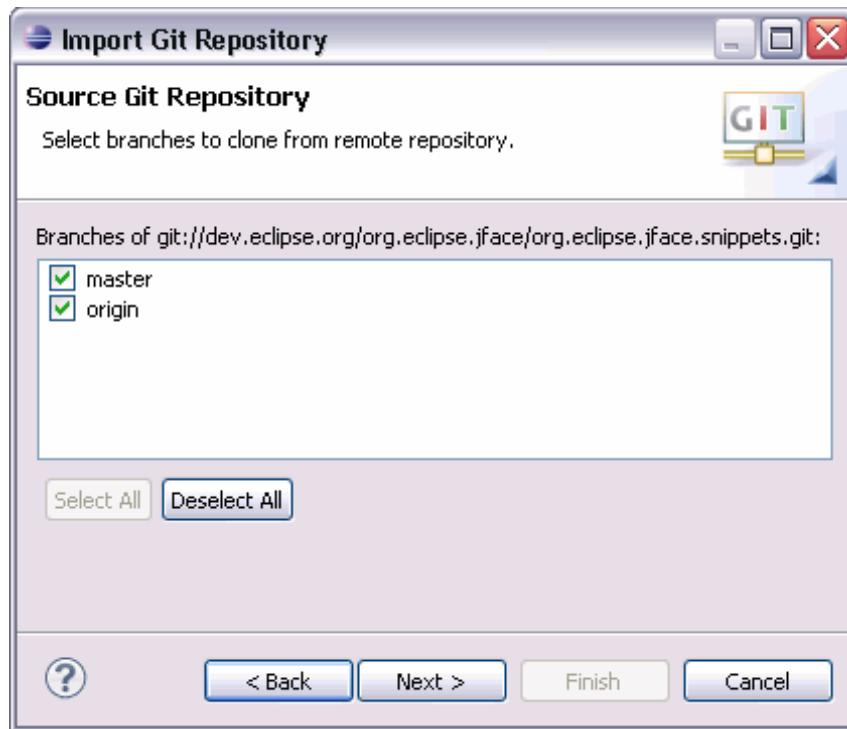
4.2. Clone existing project

EGit allows to clone an existing project. We will use an Eclipse plugin as example. To learn more about accessing standard Eclipse coding please see [Eclipse Source Code Guide](#).

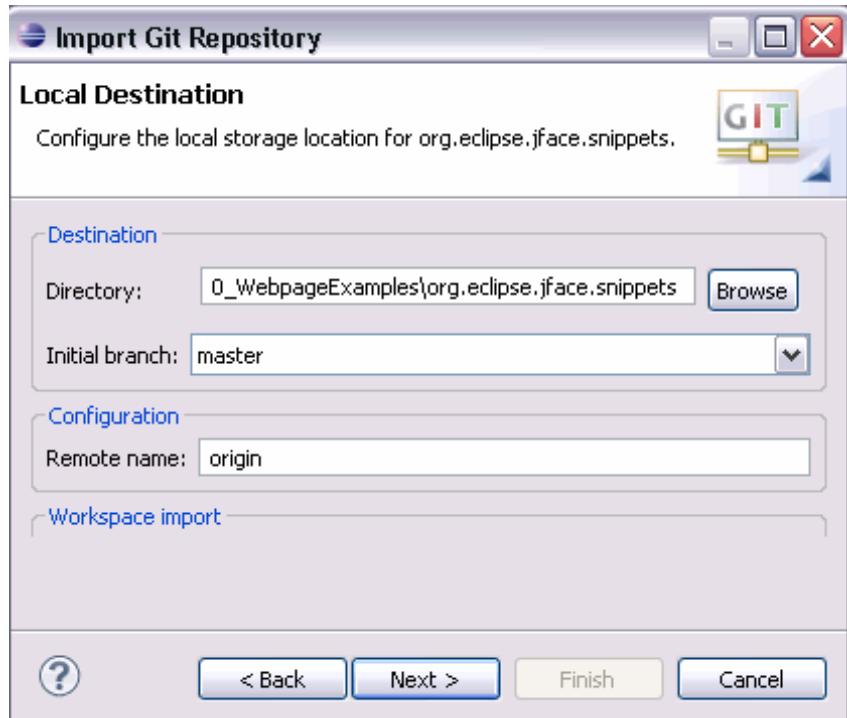
Select File -> Import -> Git -> Git Repository. Press clone and maintain the git repository "git://dev.eclipse.org/org.eclipse.jface/org.eclipse.jface.snippets.git". You only have to paste the URL to the first line of the dialog, the rest will be filled out automatically.



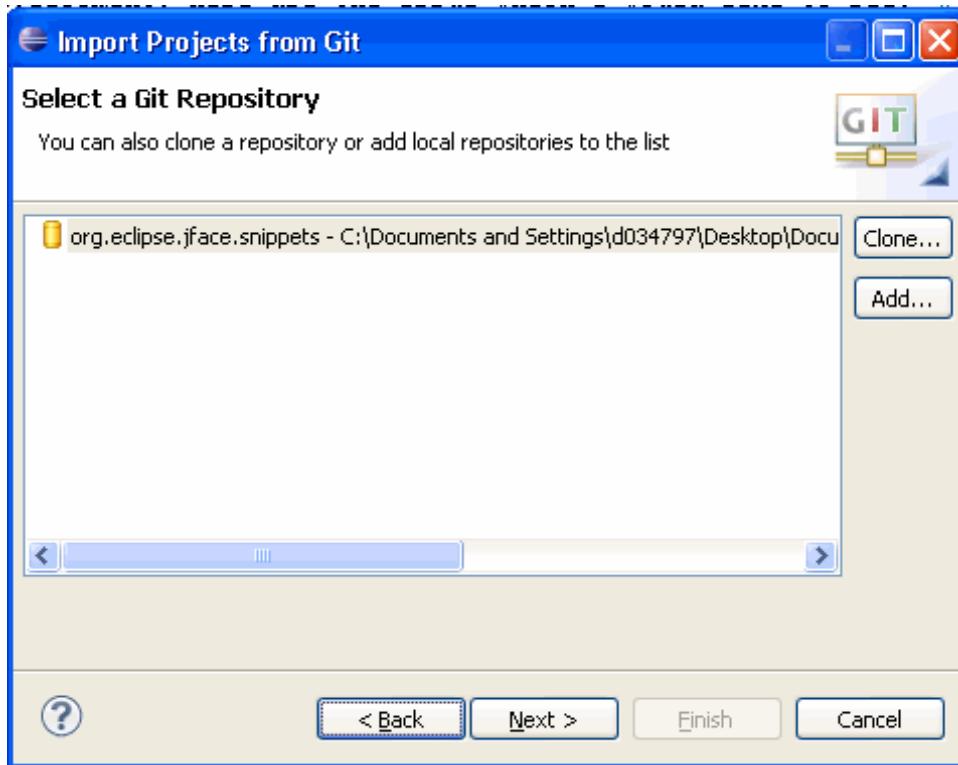
After pressing next the system will allow you to import the existing branches. You should select at least "master".



The next dialog allow you to specify where the project should be copied to and which branch should be initially selected.



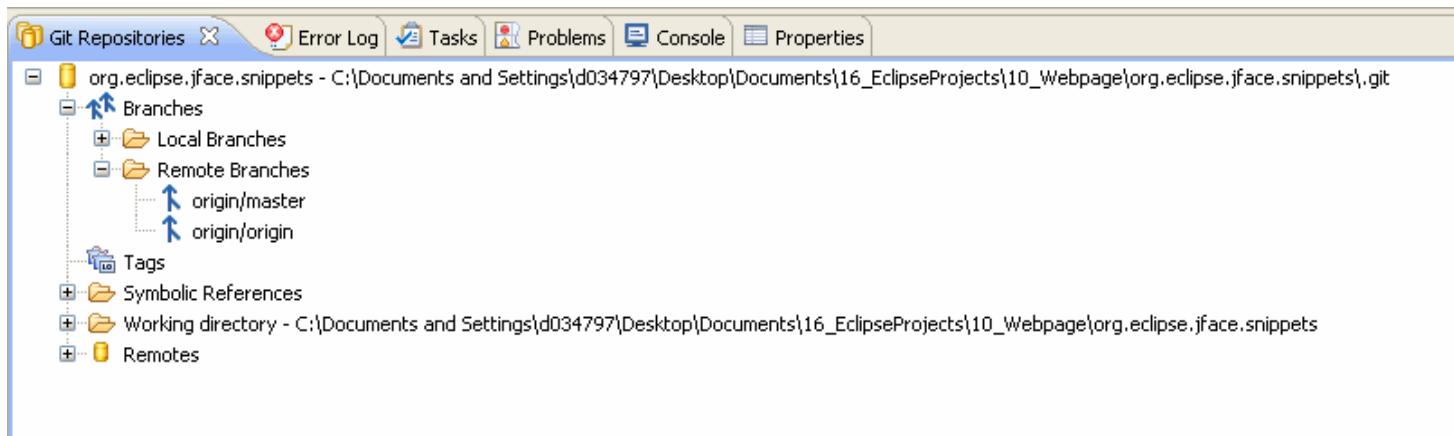
You get an additional import dialog.



You have checked to the project and can use the team Git operation on your project.

4.3. Repository view

Egit has a "Git repository" view which allow you to browse your repositories, checkout projects and manage your branches.



5. Using Egit

5.1. Basic operations

Once you have placed a project under version control you can start using team operations on your project. The team operations are available via right mouse click on your project. You can:

- Select "Team" -> "Add", on the project node to add all files to version control.

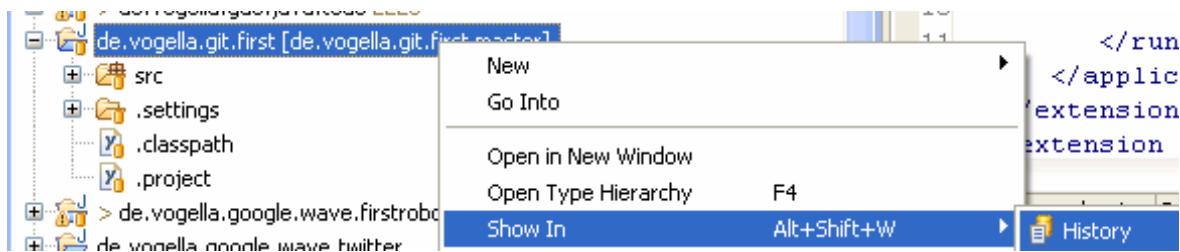
- Select "Team" -> "Commit", to commit your changes to the local Git repository.
- Select "Team" -> "Branch" to create and to switch between branches.
- Select "Team" -> "Push" to push your change to a remote Git repository (see the GitHub chapter).
- "Team" -> "Tag" allows you to create and manage tags.

5.2. Merge

EGit supports currently fast-forward merge. Fast-forward merge allows to add the changes of one branch into another if this branch has not been changed. Select "Team" -> "Merge" to perform this operation.

5.3. View the resource history

Select a resource and select Show in (Alt+Shift+W) -> History to see the commit history of this resource.



	Author	Date
HEAD [master] Added history view	Lars Vogel <Lars.Vogel@gmail.com>	2010-06-07 14:34:29
Updated EGit and Git tutorial	Lars Vogel <Lars.Vogel@gmail.com>	2010-05-25 06:07:07
Updated EGit article to new plugin	Lars Vogel <Lars.Vogel@gmail.com>	2010-05-21 10:12:01
Initial commit with the content	Lars Vogel <Lars.Vogel@gmail.com>	2010-05-21 08:36:35

```

commit b012d65a14a25439a0b24683add8e551f7281315
Author: Lars Vogel <Lars.Vogel@gmail.com> 2010-05-21
10:12:01
Committer: Lars Vogel <Lars.Vogel@gmail.com>
2010-05-21 10:12:01
Parent: 0e5e7e79c8e0b018e7a7141569081d87f4e83c71
(Initial commit with the content)
Child: b7004bc67c8b8b78cddd1f171cc60a9cedebf630
(Updated EGit and Git tutorial)

Updated EGit article to new plugin
-----
```

6. Create a Git repository for multiple projects

To put several plugins into the same git repository you should create them in a subfolder under your workspace and then create via EGit a repository for this subfolder.

To test this create two new Java projects "de.vogella.egit.multi.java1" and "de.vogella.egit.multi.java2". Do not use the default location (which would be the workspace) but in a subfolder "gitmulti".



Create a Java Project

Create a Java project in the workspace or in an external location.

Project name:

Use default location

Location: [Browse...](#)

JRE

Use an execution environment JRE:



Use a project specific JRE:



Use default JRE (currently 'java-6-openjdk')

[Configure JREs...](#)

Project layout

Use project folder as root for sources and class files

Create separate folders for sources and class files

[Configure default...](#)

Working sets

Add project to working sets

Working sets:

[Select...](#)



< Back

[Next >](#)

Cancel

Finish



Create a Java Project

Create a Java project in the workspace or in an external location.



Project name:

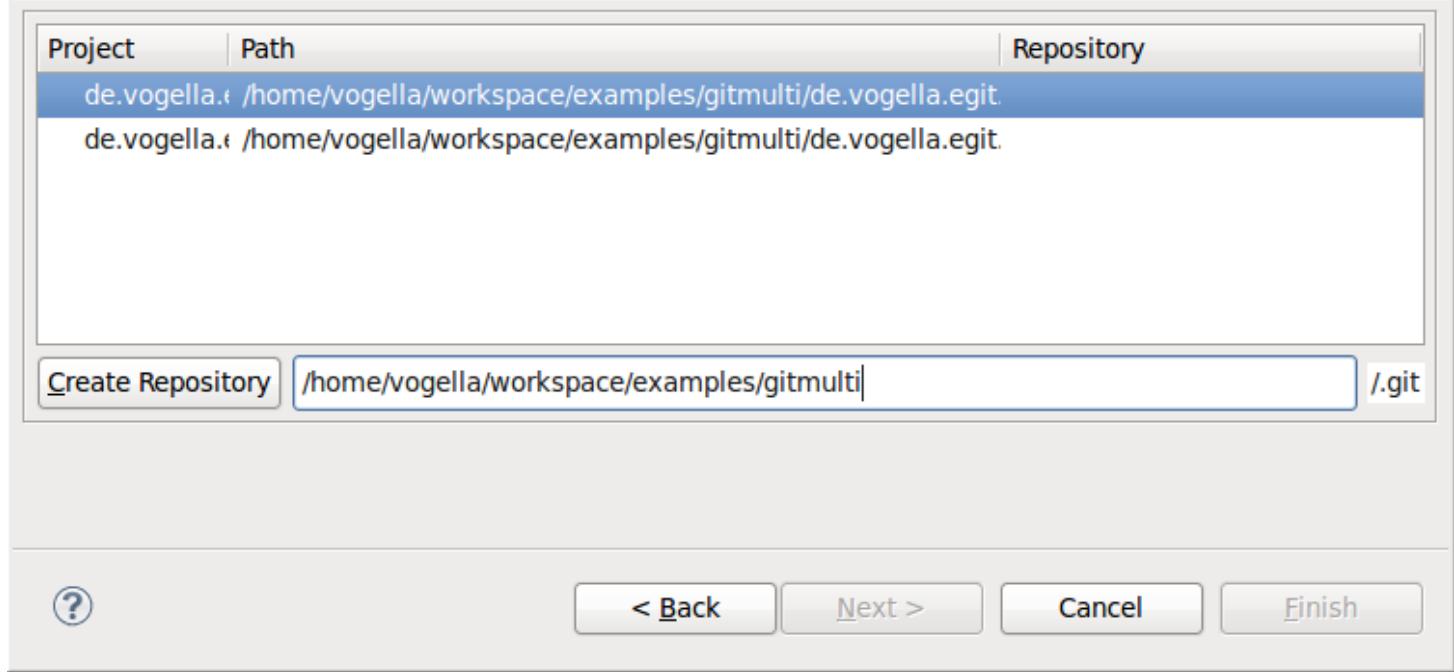
Use default location

Location: [Browse...](#)

Create a few classes, as git is not able to save empty folders. Now select both projects, right click on them, select Team-> Share-> Git.

Configure Git Repository

Select Git Repository Location



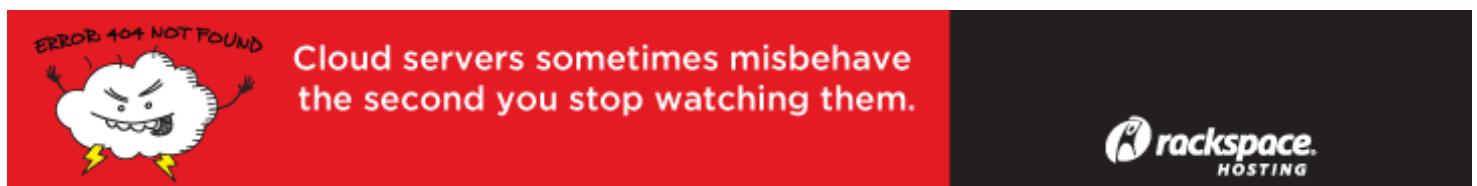
Project	Path	Repository
de.vogella.egitmulti.java1	/home/vogella/workspace/examples/gitmulti/de.vogella.egitmulti.java1	gitmulti
de.vogella.egitmulti.java2	/home/vogella/workspace/examples/gitmulti/de.vogella.egitmulti.java2	gitmulti

Create Repository /home/vogella/workspace/examples/gitmulti/.git

Buttons: ? < Back Next > Cancel Finish

Done. Both projects are now in the same git repository.

- ▷  de.vogella.egitmulti.java1 [gitmulti master]
- ▷  de.vogella.egitmulti.java2 [gitmulti master]



7. Using EGit with Github

7.1. Github

[Github](#) is a popular hosting provider for Git projects and if your repository is public to everyone Github hosting is free. To use GitHub create an account on the [Github Website](#). During the sign-up you will be asked to provide a "passphase". This "passphase" is later needed to push to Github from your local repository.

7.2. Create Repository in Github

Create a new repository on Github, e.g. "de.vogella.git.github".

Create a New Repository

Create a new empty repository into which you can push your local git repo.

NOTE: If you intend to push a copy of a repository that is already hosted on GitHub, please [fork it instead](#).

Project Name

Description

Homepage URL

Who has access to this repository? (You can change this later)

Anyone ([Learn more about public repos](#))

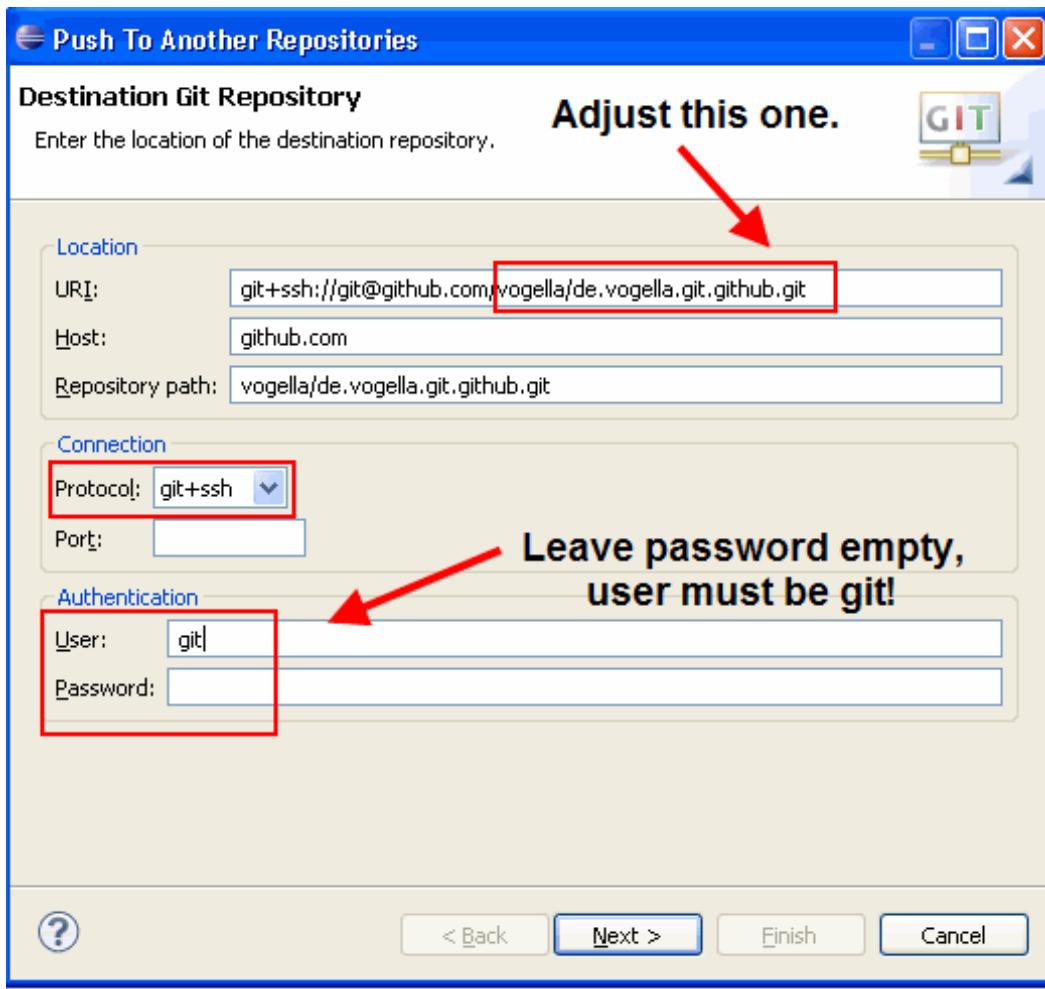
[Upgrade your plan to create more private repositories!](#)

After creation of your new repository Github tells you what to do if you would import via the command line. As we are going to use EGit we can ignore this information.

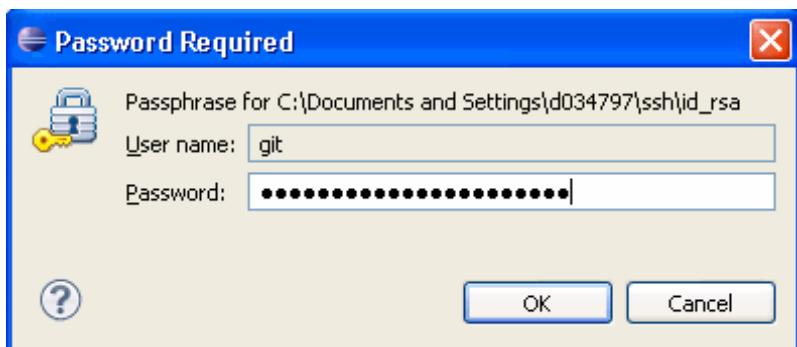
7.3. Push project to Github

Create a new project "de.vogella.git.github" and put it under version control. Right-mouse click on your project and select "Team" -> "Push". A dialog pops up. Maintain the following data. Adjust the highlighted line so that you are using your user and your project name.

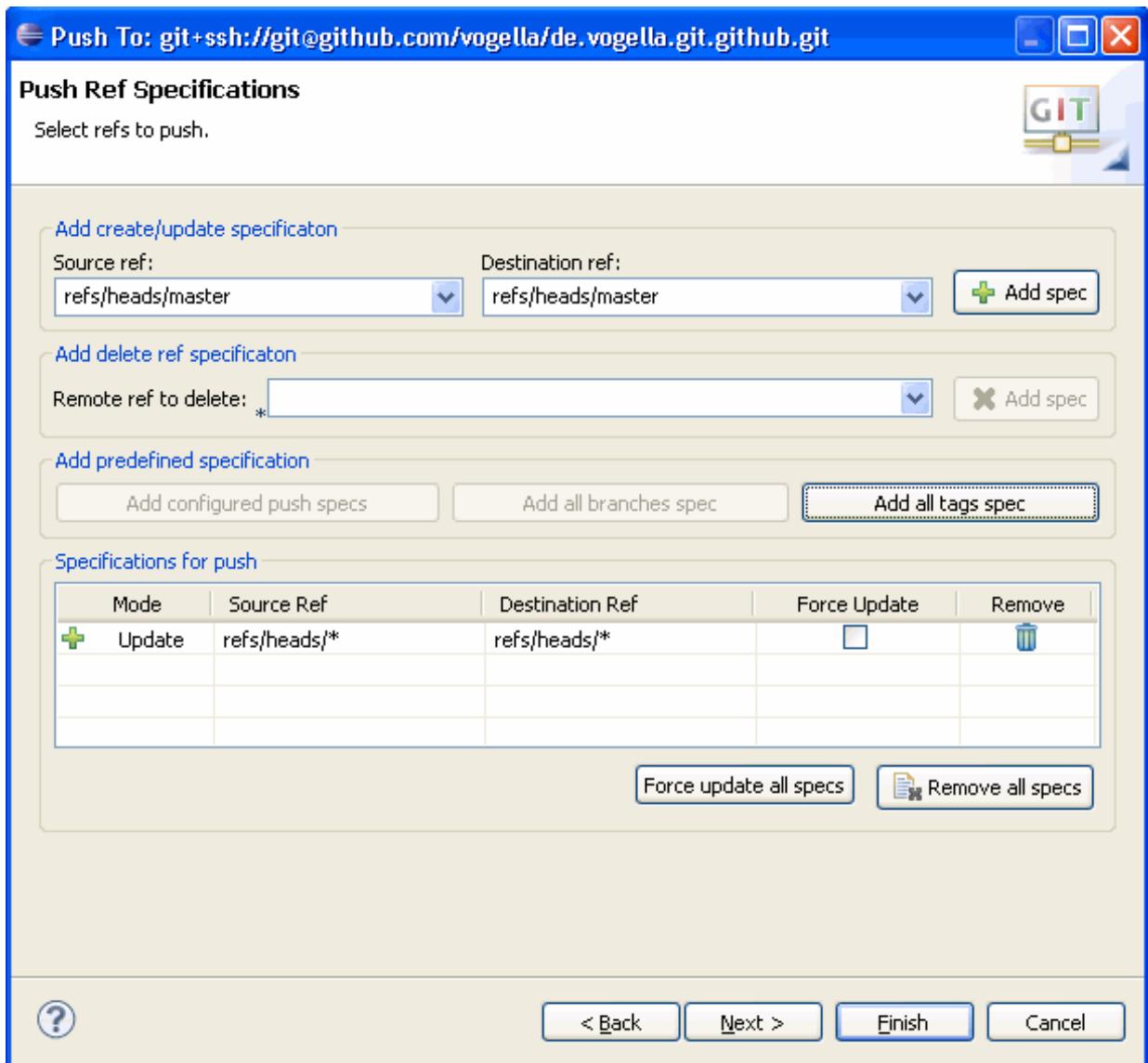
```
git+ssh://git@github.com/vogella/de.vogella.git.github
```



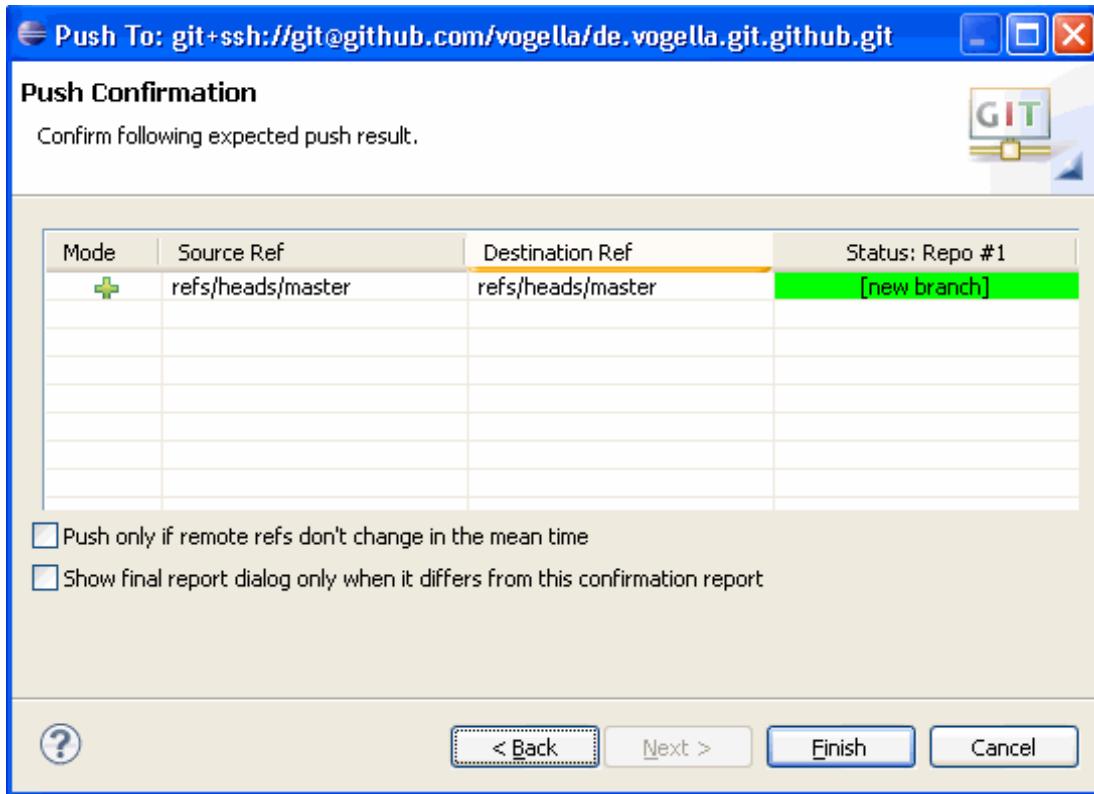
Maintain your passphrase which you maintained during the Github setup.



Select your branch (you should currently only have master if you followed my tutorial), press "Add all branches spec" and select next.



Press finish.



If you now select your github Dashboard and then your project you should see the committed files.

vogella Social Coding

Dashboard Inbox 0 Account Settings Log Out

Explore GitHub Gist Blog Help Search GitHub...

vogella / de.vogella.git.github

Admin Unwatch Pull Request Download Source 1 1

Source Commits Network (1) Fork Queue Issues (0) Downloads (0) Wiki (1) Graphs Branch: master

Branches (1) Tags (0)

First Github project via EGit
www.vogella.de

pledge Enable Donations

Private Read-Only HTTP Read-Only git@github.com:vogella/de.vogella.git.github.git This URL has Read+Write access

First commit
vogella (author) 3 minutes ago

commit: 8c5a3d51caac42bf1b90b7c526bc24bcad98077e
tree: bc5fb0b2428312841cf51fa78ed76f6162b3faf0

de.vogella.git.github /

name	age	message	history
.classpath	3 minutes ago	First commit [vogella]	
.project	3 minutes ago	First commit [vogella]	
.settings/	3 minutes ago	First commit [vogella]	

We couldn't find a README for this repository, we strongly recommend adding one.

7.4. Clone repository from Github

Use the same URI you use to push to Github to clone the project into another workspace.

7.5. Mylyn integration with Github

[Eclipse Mylyn](#) is a productivity tool for programmers. There is a GitHub connector for Mylyn available, please see <http://wiki.eclipse.org/EGit/GitHub/UserGuide> for details. .

8. Hacking EGit - Getting the source code

EGit is self-hosted on <git://egit.eclipse.org>. You can clone the EGit code from the repository using EGit using the following URL <git://egit.eclipse.org/igit.git> and <git://egit.eclipse.org/egit.git>.

You also need some libraries from Orbit. See [Libraries from Orbit](#) for getting these libraries.

9. Thank you

Please help me to support this article:



10. Questions and Discussion

Before posting questions, please see the [yogella FAQ](#). If you have questions or find an error in this article please use the [www.vogella.de Google Group](#). I have created a short list [how to create good questions](#) which might also help you.

11. Links and Literature

11.1. EGit and Git Resources

<http://www.vogella.de/articles/Eclipse/article.html> EGit IDE Tutorial

http://wiki.eclipse.org/EGit/User_Guide EGit User Guide

http://wiki.eclipse.org/EGit/Contributor_Guide EGit contributor guide

[Git Tutorial](#)

11.2. vogella Resources

[Eclipse RCP Training](#) (German) Eclipse RCP Training with Lars Vogel

[Android Tutorial](#) Introduction to Android Programming

[GWT Tutorial](#) Program in Java and compile to JavaScript and HTML

[Eclipse RCP Tutorial](#) Create native applications in Java

[JUnit Tutorial](#) Test your application

[Git Tutorial](#) Put everything you have under distributed version control system