Working with GIT

Tim Sutton, 2011

Contents

1 Working with GIT

This document provides a simple introduction to working with git. GIT is a distributed source code management (SCM) system. Although this is a source code management system, you can use it for **any** type of documents that you want to version control and / or work on collaboratively with others.

If you are familiar with SVN, GIT is quite similar but adds some new concepts. In particular, GIT is distributed, which means there doesnt have to be one single repository that you work against. GIT also is ideal for offline work where you still want to do version control. We will explore this more as we go on.

1.1 Installation

Installing GIT is easy. Under linux, install like this:

```
sudo apt-get install git meld gitg
```

The latter two are not actually needed but will prove useful later.

Under windows you can use msysgit - notes on using msys git are provided further down in this document.

1.2 Quick Start

I would like to plunge in with a quick start to using git and then come back to a more systematic coverage of the tool.

1.2.1 Create a local repository

```
cd dev
mkdir git-sandbox
cd git-sandbox
git init
```

Ok now we have a repository, lets do some work in it.

1.2.2 Adding a file to the repository

```
gedit README
```

or

```
vim README
```

Write a little text to the file.

```
Hello world from my sandbox
```

Now save and close the file.

First we can use the git status command to see the status of our repository:

```
git status
```

Which will return something like this:

```
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
# README
nothing added to commit but untracked files present (use "git add" to track)
```

So it tells us that we have one untracked file - the README file we just created. So how do we start tracking the file?

```
git add README
```

Now ‘`git status`‘ will show the file as tracked:

```
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
# new file:   README
#
```

1.3 Creating a new repository

You can create your repository in one of two ways:
initialise a new one # clone an existing one

1.3.1 Creating a local repository

Lets first play with making our own repository:

```
mkdir git-sandbox
cd git-sandbox
git init
```

Thats it, you can now work in this directory, commit, branch, merge etc. as it is a complete repository.

Often you will want to work with others to build your software. Typically the

1.3.2 Git under windows

Here are some generic notes on using git under windows you should install msys git app. In windows explorer go to c:\Documents and Settings\<your user>\

Make a directory called .ssh

In that directory create a text file called ’config’ (note it has no extension) and put the following content into it:

```
Host <host name>
  User <user name>
  HostName <host name>
  Port <port>
```

Replaces items in angle brackets above as appropriate.

Now copy your id_dsa into this directory (it should be a unix style one so you may need to convert from putty style private key, though just try with your existing one first).

Open the msys git shell then go to the directory where you want to check out your project to. For example to check it out to c:\dev\foo do

```
cd /c/
mkdir dev
cd dev
```

Now clone the directory:

```
git clone git@foo:bar.git bar
```

Make sure to type 'yes' in full when it asks you if you are sure you want to continue connecting.

Then enter your passphrase when prompted.

Wait a few minutes while it checks out.

Thereafter you use the git commands from the msys shell as normal. There is also a tortoisegit explorer integration for windows you can try but I havent used it and don't know how well it works.