Bitbucket 101

If you are new to hosting your code, code management with distributed version control systems (DVCS), or either Git or Mercurial, this Bitbucket 101 tutorial gives you a taste all of them. In this tutorial, you'll first install both Git and Mercurial for your operating system. You'll do some work using both Git and then Mercurial. Throughout, you'll use the hosted code management system that is Bitbucket. The tutorial consists of the following pages:

- Set up Git and Mercurial on Windows (or Mac OSX or Linux)
- Create an Account and a Git Repo
- Clone Your Git Repo and Add Source Files
- Fork a Repo, Compare Code, and Create a Pull Request
- Add Users, Set Permissions, and Review Account Plans
- Set up a Wiki and an Issue Tracker
- Set up SSH for Git
- Set up SSH for Mercurial
- Mac Users: SourceTree a Free Git and Mercurial GUI
- Sourcetree Git GUI on Windows

The tutorial teaches you some simple DVCS workflows using basic Git and Mercurial commands.

How to work through the tutorial

If you are totally new to DVCS and/or Bitbucket, you should work through each page sequentially as each new page builds on the material from the previous pages. If you prefer to learn the tools before working through this tutorial, you can learn more about Git here. To learn more about Mercurial, visit Mercurial's website. To learn more about DVCS in general, see Getting Started with Git and Mercurial.

At the end of each page is a **Next** section directs you to the next tutorial page. If you get lost you can use the navigation bar (to your left) to locate the next page. If you feel confident skipping pages or just going to the pages you need, feel free to do that too.

Next

Start the tutorial with Set up Git and Mercurial. If you want to skip the tutorial all together, You might be interested in:

- Frequently asked questions and answers
- Bitbucket Google group

- How to use your repository
- Bitbucket blog

Like 2 people like this

37 Comments

Lars Vogel

See also here: Git Tutorial

Anonymous

Great Tutorial Lars, Thanks for sharing.

Anonymous

Good tutorial. Makes it easy to use bitbucket. Thanks!

manthony

Thank you! Always glad to hear this kind of comment. 

Anonymous

thank u

Muhammad Moosa

very good tutorial, I am switching from github and enjoying bitbucket and this tutorial. Very smooth and clean wording really amazing way. Thanks for producing such a detailed and easy tutorial specially for beginners.

Anonymous

Great tutorial, I spent a lot of time on it, but got pretty confused as my boss has me learning Mercurial and Bitbucket... learning git only confused me. One of the pages says something to the effect if you want to learn Mercurial with it (other than a few nods at Mercurial), "you're on your own."

Would you consider having an alternate tutorial for Mercurial on it's own. For me, time is money and I wasted a bunch of time on Git.

Otherwise, I am very impressed by the overall tutorial. Thanks for that!

Kevin

manthony

Kevin,

We have talked about having two tutorials yes. We aren't planning one for the near future. There are many more things you can do with Mercurial and with Git than our covered by our tutorial – which is really teaching users about bitbucket and not any particular DVCS. We often talk about expanding and creating tutorials with more in depth material about Git commands and process — or Mercurial commands and processes . Would you have liked this as well?

Mary

Anonymous

So I can go throught 101 for Mercurial? I do not want to download, use, learn Git. Just Mercurial.
Or do I have to go to the Mercurial site?lket

manthony

Hi,

This tutorial is for Bitbucket's features and how to use them. Since Bitbucket supports both Git and Mercurial, the tutorial has you getting a taste of both. The guide isn't intended to be a tutorial for Mercurial or Git as DVCS packages. If you want to learn the ins and outs of Mercurial, yes, you should use the tutorial the makers of Mercurial recommend.

Cheers,

Mary

Brian Anderson

Love it!

Andrew Pennebaker

I've imported a GitHub project into BitBucket. Since then, the GitHub project has received changes. What's the easiest way to import the GitHub changes into the BitBucket version?

Mary Anthony [Administrative Account]

The easiest way just depends on how big a change. One or two files, you could just clone the Bitbucket version and make the change manually. If you expect to try an maintain these both simultaneously, you should investigate using push hooks or some other automation feature to keep them in sync.

Andrew Pennebaker

Thanks for the quick response! Yeah, I guess those are standard options for importing changes. I hear talk of BitBucket adding a feature similar to Import, where BitBucket can from the web browser import changes from the original repository. That would be cool!

Anonymous

Something that would be very helpful in this documentation is a list of terms and definitions (i.e. Repo, Fork, Pull Request, SSH for Git, etc.). I'd include a brief explanation of what each term means/is and how it's used in relation to other tools and Bitbucket. This dictionary would be very helpful for developers moving from Microsoft products or even SVN.

manthony

Thank you for the suggestion. I weigh the pros and cons of this a lot; a brief explanation is some times not possible – in particular if I'm comparing and contrasting tools. So, I'm not sure I want to put that into the tutorial. Glossaries don't typically get a lot of users; but maybe I can pitch another way that would draw more users.

Anonymous

I have a team account.

And here is the problem: how to check previous version of my source code after one member of my team has made some changes and merged them?

thank you!!!

manthony

Hi,

I need more information. By "check" do you mean review the changes, test the changes, or something else?

Mary

Anonymous

use

(updating own repository)

git pull origin master

(push own locally changes to git repository)

(command) (first commit then push)

git commit -a -m "add changes"

git push origin master

manthony

The easiest way to do this is for you and your team mates each to work in branches. That way, you don't pull his code directly into yours. You can learn some basics workflows here: http://www.atlassian.com/git/workflows

Anonymous

what do you means ? (That way, you don't pull his code directly into yours.)

manthony

When you work on a repository, you are always working on a branch. To see which branch you are on use the `git status` command.

If you pull another user's changes directly into your working branch, git attempts to merge for you. Depending on what you are doing, you might not want to run the risk of clobbering your current work.

Git Branching is a great resource on this topic.

Guy Hilsman

Can I create a 'bare' Git repository? I tried to ssh via command line to bitbucket to do this, but it is not allowed.

manthony

You can create an empty repo on Bitbucket and clone it, yes. Did I understand your question?

Guy Hilsman

Thanks for the reply! Actually, can I ssh into my repo on BitBucket and use the command line there?

Guy Hilsman

Now I realize 'ssh' login is NOT allowed. I'm trying to create a 'bare' repo of an existing production website.

Is it correct that by using:

```
> git remote add origin ssh://git@bitbucket.org/username/project.git
```

this will create a 'bare' repo?

Thank you!

manthony

Guy,

Correct, you cannot `ssh` directly to Bitbucket and use a command line. Also, you can't create a repo on Bitbucket with a `git` command. You *can* do it with a REST call.

https://api.bitbucket.org/1.0/repositories --data "name=mynewrepo"

See the repository Resource for more information on this.

Mary

Anonymous

There is a difference between a repo and a bare repo. How do you create a bare repro?

manthony

Yes, a bare repo doesn't contain a working code tree. A bare repo is contains only the contents of the .git subdirectory. You can't create one on Bitbucket. You can only create them on your local machine. To create one, you supply the `--bare` flag when you clone.

```
git clone --bare repository_url
```

This net resource has more information on them: http://gitolite.com/concepts/bare.html

---

Anonymous

گنج تو وجودت است؛ جای دیگر به دنبالش نگرد! همه قصر ها و همه ی پل هایی که به قصر ختم میشوند مهمل و بی معنی اند، تو باید پل خود را در درون وجود خود خلق کنی! قصر آنجاست؛ گنج هم آنجاست...

Anonymous

嗯， 很好！

manthony

谢谢!

Anonymous

ola

Anonymous

In case you haven't noticed already, the link to the Max OSX version of SourceTree incorrectly links to the Windows version.

manthony

Nope, I hadn't noticed. Thanks for the catch. 😊

Anonymous

Great idea but hopelessly difficult site to use. How can any busy author wish to spend time learning all these technicalities? Please create a user friendly site!!!

manthony

Hi, I'm sorry to hear you weren't satisfied with the site. Developing code is a technical task so, while Bitbucket does strive to make things as easy as possible, it would be unlikely we could eliminate the need to learn for everyone. If you could provide more information about what you found too technical, we might be able to help you further.

Set up Git and Mercurial

To use Bitbucket, you need to install a DVCS tool on the computer where you write your code. Typically, this computer is a machine physically close to you like your home or work computer. This is your local machine or system. You also might write or deploy code to a remote machine – for example a lab computer or a server in a data center. You may also need a DVCS tool on that machine too. This tutorial refers to the typical case, your local system, but the instructions are the same for both cases.

Linux or Mac User?

Mac users see this page.

Linux users see this page.

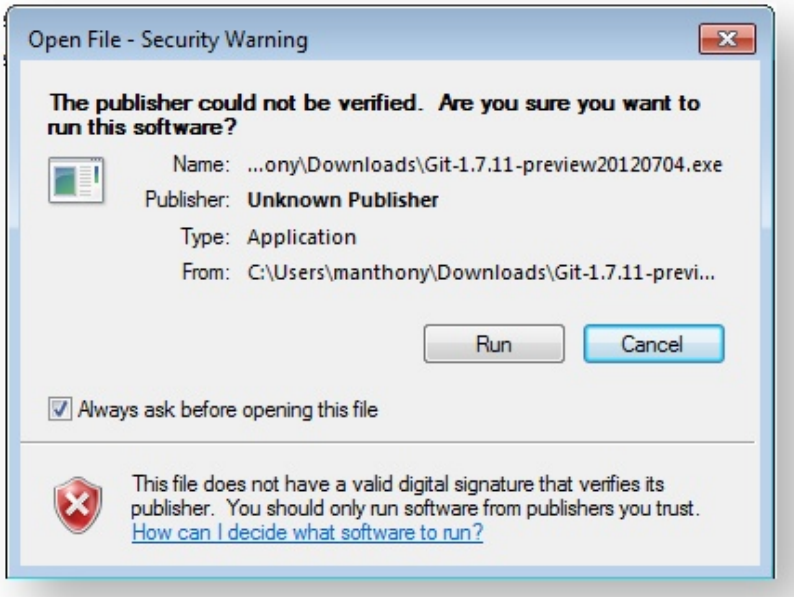
Bitbucket supports two DVCS tools, Git and Mercurial. These tools run on all modern operating systems. For Git, Bitbucket supports 1.6.6 or later and Mercurial version 1.7 or later. Mercurial also requires (depends on) the Python programming language. The installation process takes care of making sure you get the correct version of Python.

Since you can use both Git and Mercurial on the same machine, this page shows you how to install both because you need both to complete this tutorial. If you already have these tools installed, skip the instructions and go to the next step in the tutorial.

Step 1. Install Git for Windows

Do the following to install Git on your Windows machine:

1. Download the Git for Windows installer package.
   You can either **Run** the installer directly from your browser's **File Download** dialog or you can **Save** the file to a folder on your computer. These instructions assume you run the installer when prompted by the downloader.



If you downloaded the file to a folder on your local machine, you can also click the downloaded file's icon to run the installer. When you've successfully started the installer, you should see the **Git Setup** wizard screen:

The version shown on your screen may be different than the one shown here of course. That is ok as long as you are installing a Git version 1.6.6 or later.

2. Press **Next** to move to the next page of the wizard.

   The setup displays the license agreement.

3. Press **Next** to accept the license agreement and continue.

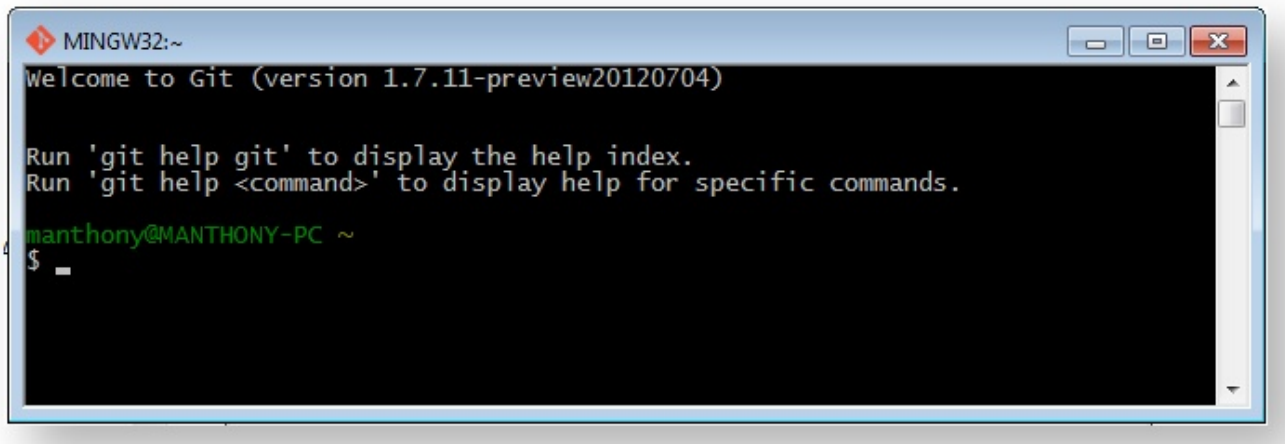
   For this setup, use all the default setup values recommended by installer.

4. To accept all the defaults, press **Next** on each page of the dialog that comes after the license agreement.
5. Press **Finish** on the final page of the dialog to complete the installation.

   The system opens the release notes. You might want to skim them.

6. Close the release notes.
7. Open the **Git Bash** window by choosing **Start > All Programs > Git > Git Bash**.

   The system opens a command window:



Using Git Bash, you can simply enter the `git` command lines that appear elsewhere in the Bitbucket documentation. This documentation assumes you are familiar with a bash shell. If you want to use the Git GUI instead of Git Bash, you can, you'll need to learn that on your own though.

8. Configure your username using the following command:

   ```
   git config --global user.name "FIRST_NAME LAST_NAME"
   ```

9. Configure your email address using the following command:

   ```
   git config --global user.email "MY_NAME@example.com"
   ```

Step 2. (Optional) Install the Git credential helper on Windows 7 or 8

Bitbucket supports pushing and pulling over HTTP to your remote Git repositories on Bitbucket. Every time you interact with the remote repository, you must supply a username/password combination. Instead of supplying the combination with every HTTP call, you can store these credentials in your OSX keychain provided you have the **git-credential-winstore** helper added to Git.

The helper asks for your username/password on the first Git operation and then stores the credential. Future operations won't require you to supply a username/password combination. To install the helper, do the following in your Windows 7 or 8 (.NET4.0 required) environment:

1. Download the git-credential-winstore application.
2. Make a note of where you downloaded the application.
3. Start the Git Bash shell.
4. In the shell, change directory to the directory with the git-credential-winstore download.
5. Install the credential helper by entering the following at the command line.

   ```
   ./git-credential-winstore -i /bin/git
   ```

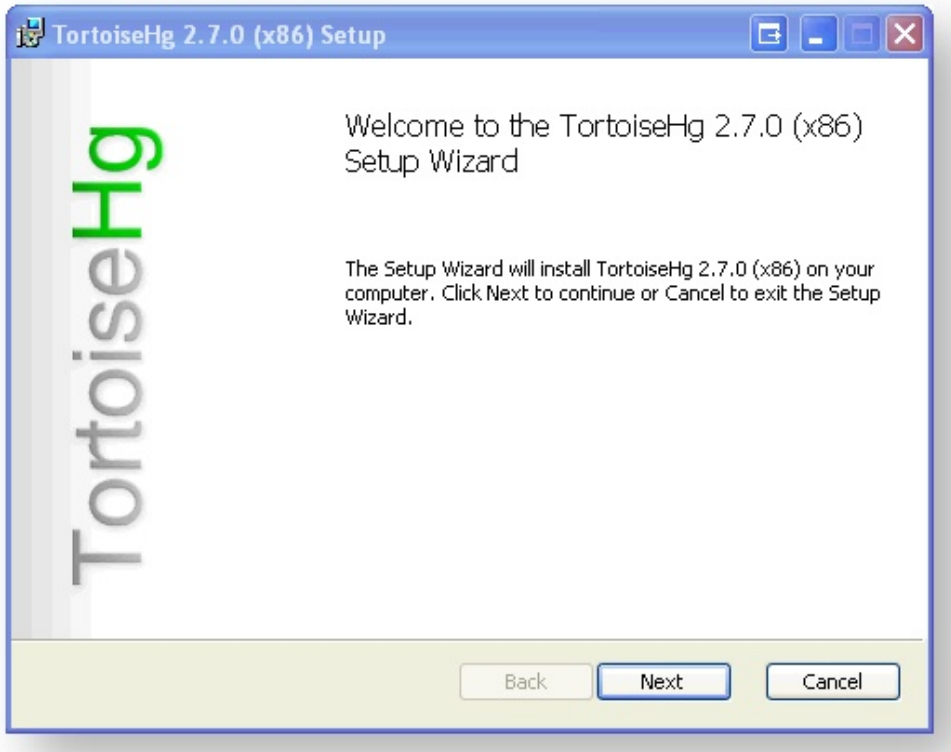
   The command updates your global Git configuration.

Step 3. Install Mercurial

TortoiseHg 2.2.1 is the Microsoft Windows version of Mercurial.

1. Download the all-in-one installer (MSI version) from the Bitbucket repository (shameless promotion) .

   If you downloaded the file to a folder on your local machine, you can also click the downloaded file's icon to run the installer. When you've successfully started the installer, you should see the **TortoiseHg Setup** wizard screen:

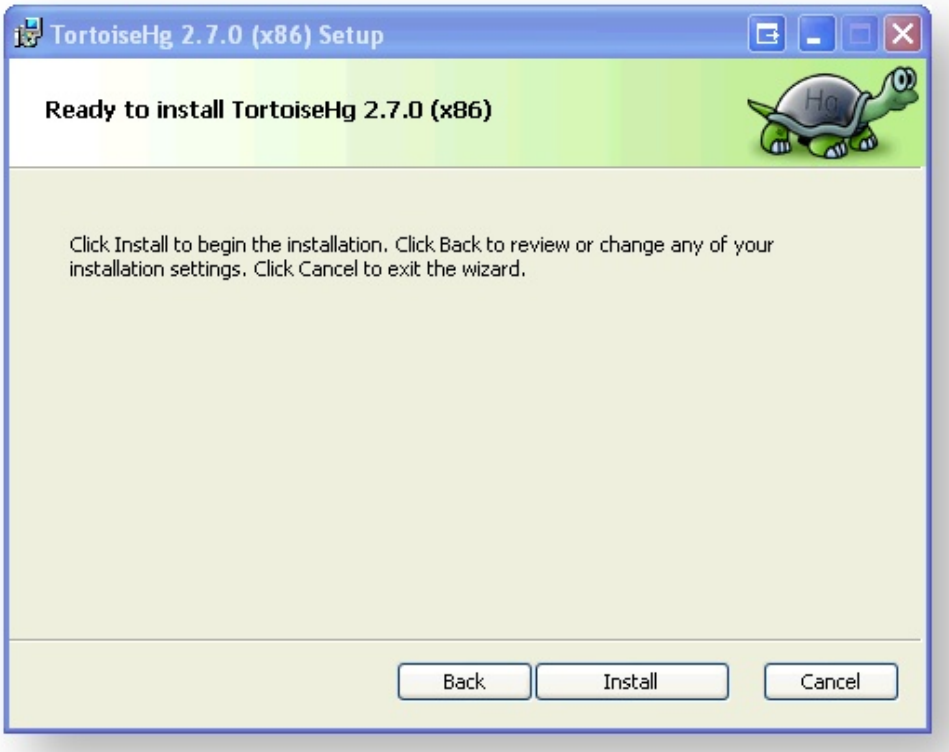


   The version shown on your screen may be different than the one shown here of course. That is ok as long as you are installing a TortoiseHg that supports Mercurial version 1.7 or later.

2. Press **Next** to move to the next page of the wizard.
   The setup displays the license agreement.

3. Press **Next** to accept the license agreement and continue.
   For this setup, use all the default setup values recommended by installer.

4. To accept all the defaults, press **Next** on each page of the dialog that comes after the license agreement until the wizards prompts you to install.



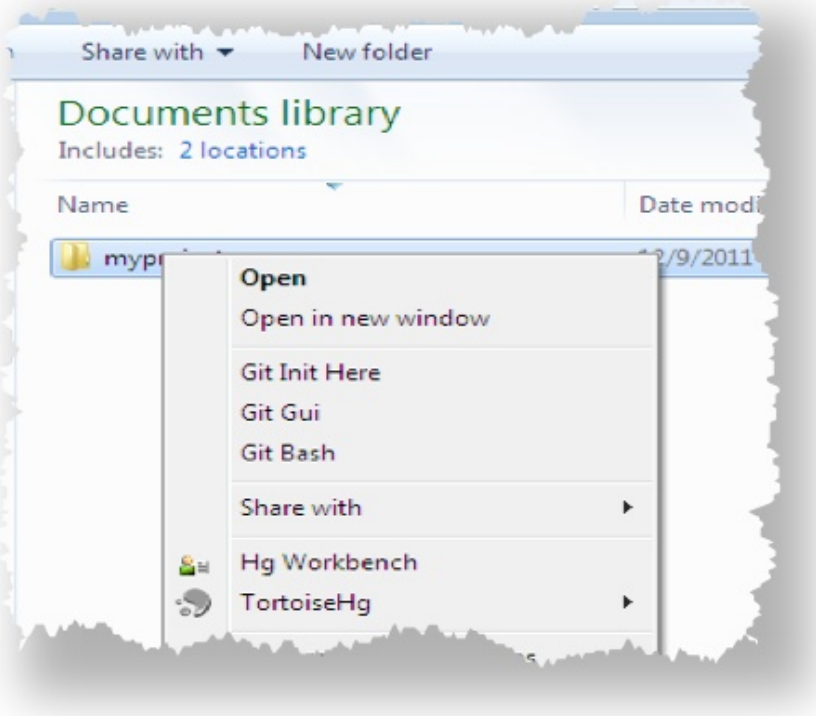
5. Press **Install** to install the software.
6. Press **Finish** on the final page of the dialog to complete the installation.

   You may need to restart your system for the installation to take effect.

Do the following to configure your global default username and email:

1. Right-click in your Desktop to open the context menu.

   The system displays the right hand context menu:



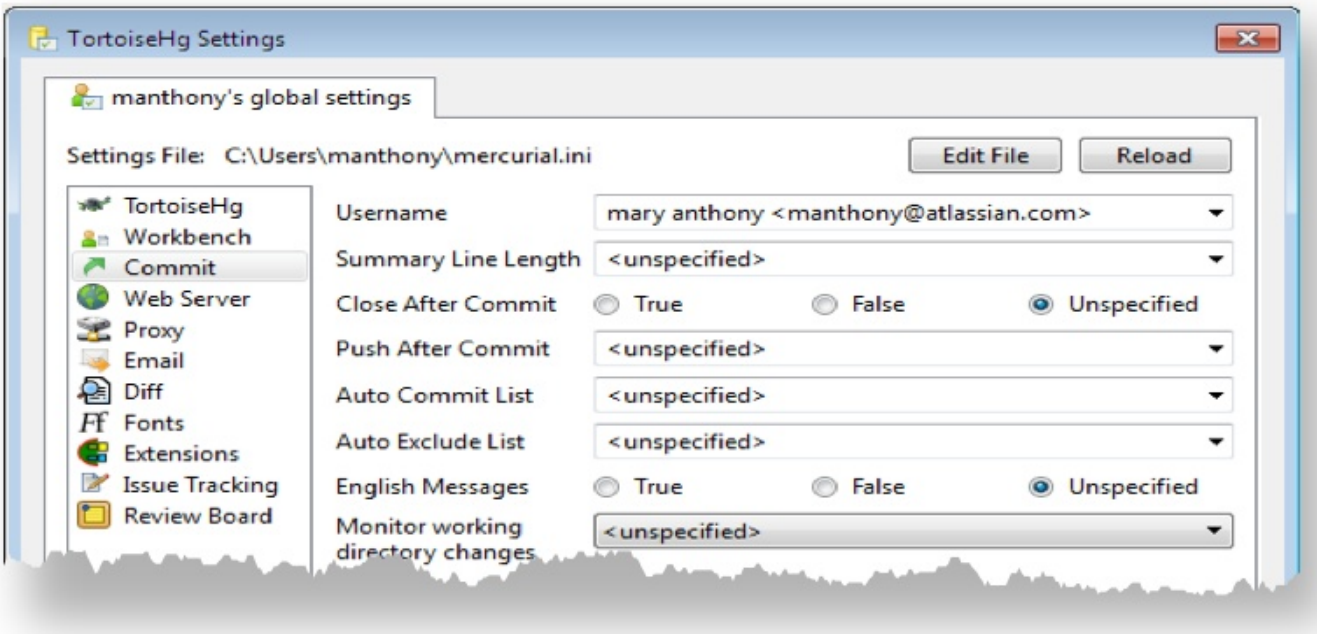
2. Click the **Hg Workbench** item.

   The system opens the **TortoiseHg Workbench** application.

3. Choose **File > Settings** to open the **TortoiseHg Settings** dialog.
4. Locate the **Commit** section on the left hand side and click it.
5. Fill in the **Username** value using the following format:

   firstname lastname<email>

   When done the dialog looks similar to this:



6. Press **OK** to save your changes.

Step 4. Install PuTTYgen and configure PuTTY

PuTTYgen is a free RSA and DSA key generation tool. You'll learn more about RSA, DSA, and key generation later in this tutorial. If you don't have PuTTYgen installed, do the following:

1. Download the proper version for your system.

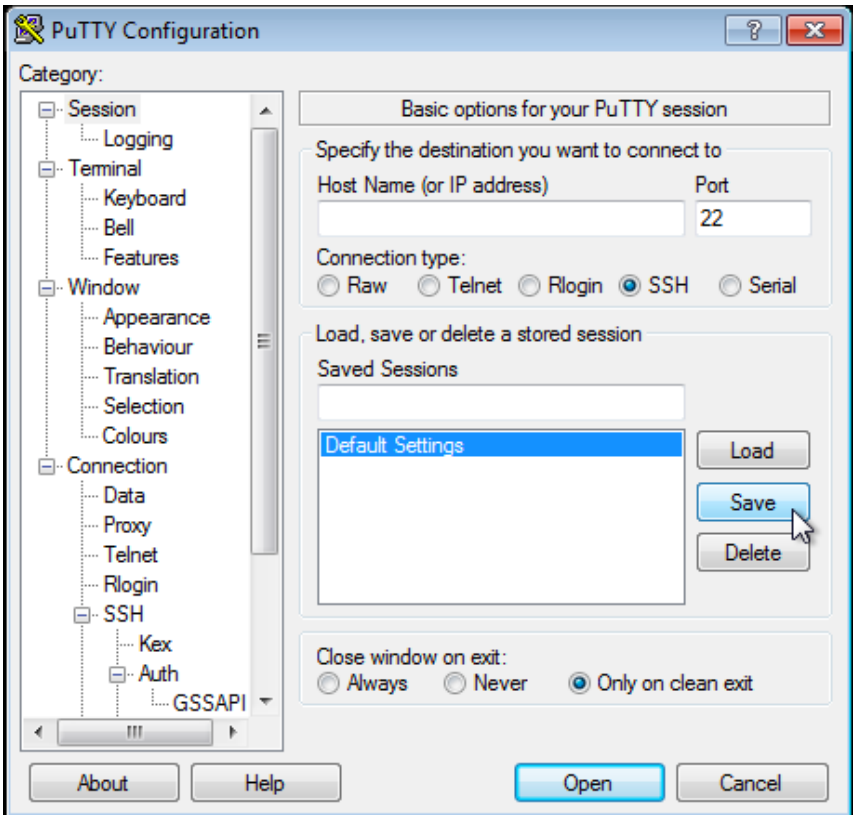
   The PuTTYgen utility is a single executable file.
2. Move the `puttygen.exe` executable to the `C:\Program Files\TortoiseHg` folder.
3. Start Putty.

   The **PuTTy Configuratio**n dialog displays. Use this dialog to configure your PuTTy sessions.
4. Under the **Session** node**,** select **Default Settings** and press **Load**.

   This allows you to edit the **Default Settings** session configuration.
5. Under the **Connection** node, click **SSH**.

   The **Options controlling SSH connections** display.
6. Check **Enable compression**.

   This option can improve performance of a low-band connection.
7. Click the **Session** node**,** select **Default Settings** and press **Save**.



8. Click the **Close** button (red x).

Next Steps

The next step is to Create an Account and a Git Repo

Like Kamran Mackey and 3 others like this

33 Comments

Anonymous

echt ein super tutorial, welches mal schritt für schritt alles erklärt!!!

Vielen Dank!

Anonymous

What he above me wrote, but then in English 🙂

manthony

Danke and you as well in English!

Anonymous

Bitte = You're Welcome

Muhammad Moosa

I am feeling happy over my decision of switching from github. This tutorial is very easy to follow.

Muhammad Moosa

Please update the tutorial contents and add TortoiseGit also.

TortoiseGit

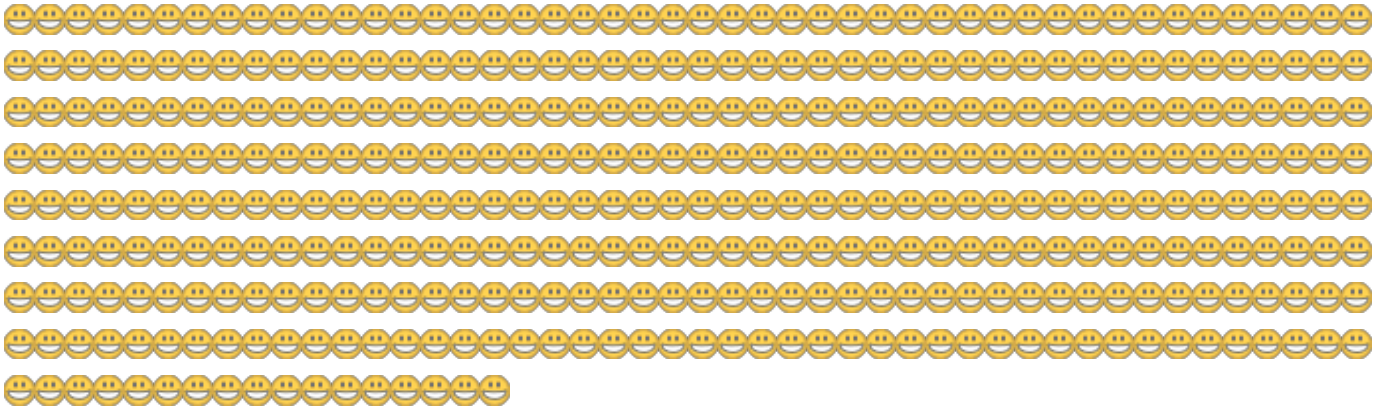
manthony

Muhammad,

I'll bring up your request to our project manager. I'm not full time on the bitbucket documentation so we have to balance enhancement requests against my other responsibilities.

Mary

Anonymous



Anonymous

There is also TortoiseGIT that makes working with GIT bit easier, you might want to have a look at that and add it to documentation, however TortoiseHG steps probably work for that one too.

manthony

Good reference. Msygit is used in this tutorial because I can get "double-work" out of the Git examples — they work in both a Linux env and a GitBash on Windows env.

Anonymous

How about using Tortoise SVN doesn't it work ?

manthony

That wouldn't surprise me. I think TortoiseSVN is only intended to work with SVN. I could be wrong there. Why do you think it should work with Git or Mercurial as well?

Anonymous

Mary, I understand the desire to clean up the comments on this page, however the negative connotations of editing is not a good p.r. strategy. If you'll notice, this is why Amazon, eBay and others don't remove comments, even if they are negative. People tend to ignore such rhetoric when they see it, and move on. Also, the Google and Bing search engines like the continual posts and updates and rank you higher because they see updated relevant information.

manthony

You know, I hadn't considered that view point. I remove comments regularly. Typically, I remove them because 1) I've incorporated the information or 2) answered a question or 3) they have aged out. Rarely, do people make rude comments but those I do remove and spam of course.

Are Amazon and eBay good examples of comparable use cases? Don't know. These pages are documentation. Their goal is to get readers information quickly and accurately. The majority of feedback the company gets on comments is that long comments threads are distracting. And, in some cases, misleading. Readers typically ask us to remove them.

There is a push within the Company to turn comment off all together on documentation. I don't agree with that position. Sometimes, a page is wrong and a customer has a comment that can help others until I get around to incorporating. Why is there a push to turn them off? The long comment threads are often cited as a reason. (For myself, I want to do a usability test on the whole issue.)

As for the positive comments, I tend to leave those behind for selfish reasons. They make me personally happy. There really aren't that many of them. To be fair though, I can remove them on a regular basis as well.

Anonymous

Would it be possible to add a tutorial for Mac OS X. I had a hard time finding how to install git and the rest. I'm not even sure if everything I install will be really useful at this point. A little help would be welcome

manthony

Yes, there are instructions fo Mac OSX.

Lalith

Hi,

I was using the tutorials fine as a first time user until I tried to set up the SSH services later in the tutorials. I just now tried setting up bitbucket on another computer and am on this first step of the tutorials again. I have a question which might explain why Im having issues setting up an SSH service. When installing Git and Mercurial I'm wondering how important my Windows system type is (X86 or X64). My System type is Windows X64 which maybe be an issue in this tutorial? The GIT installer linked here is a X86 version for windows that installs in my C:/Program Files(x86) and the Mercurial basic installation sets up in C:/Program files. As these two programs are in seperate folders in the basic setup, will there be issues in them communicating in an SSH service. Additionally When trying to install PuTTYgen via this tutorial the linking page lists only X86 versions of Puttygen for windows, don't I need a 64x version for my X64 system? Is there a PuTTYgen X64 version?

Thanks for assistance with my questions.

Anonymous

Okay, this is strange. The .gitconfig file is refusing to display its filename. I see the icon, but there's no text in any view of the folder (Win7 x64). Is this a known bug (google doesn't return any results)? Git doesn't seem to have any trouble accessing it, but seeing a file with no name is kinda freaking me out.

manthony

Hmm, this is hard for me to visualize. Can you provide a screen capture showing what you are seeing?

Anonymous

I figured it out–turning off "hide known file extensions" fixed it. Apparently, Windows saw the ".", decided gitconfig was a known "file extension," and hid it.

Anonymous

What does DVCS mean? I've got Mercurial set up, is that a DVCS?

manthony

DVCS is distributed version control system. It is defined on the first page of the tutorial.

Cameron Dunham

Dear Mary,

I have a question about:

Step 1. Install Git for Windows

8. Configure your username using the following command:

git config --global user.name "FIRST_NAME LAST_NAME"

and

9. Configure your email address using the following command:

git config --global user.name "MY_NAME@example.com"

How many users can I have on the same computer?

If I can only have one user on a computer how can I delete a user that is stored in the git bash interface?

manthony

Cameron,

Git has the concept of system, global, and repository specific settings for things like user.name:

- system – every user on a machine and all their repositories
- global - all repositories within a user's environment
- repo - actions within that repo

The first part of this page is a good intro to these concepts:

http://git-scm.com/book/en/Customizing-Git-Git-Configuration

So, if you have multiple user.names you commit under, you would configure the one you use most often as your global user. For repositories where you want to be known as a different user, you would configure that username in the repository's .git/config file. I like to edit the configuration files myself but you can just run the git config command also.

Hope this helps.

Mary

Anonymous

I haven't read all of the tutorial yet, but the clear, complete writing of the first few pages has me convinced to get a Bitbucket account and try it out. I signed up with a competitor yesterday but found the documentation lacking and the process, from what I observed, was too heavy. Thanks for providing documentation that makes Bitbucket readily accessible to everyone.

One little nitpick:

> Since you can use both Git and Mercurial on the same machine, this page shows you how to install both because you need both to complete this tutorial. If you already have one or both of these tools installed, skip the instructions and go to the next step in the tutorial.

The underlined parts sort of contradict each other. You say we need both to complete the tutorial, but then say we can skip installation if we have either one. It makes me think that we could work through the tutorial using either Git or Mercurial, but it leaves me uncertain. Not a big deal, since both are free and downloading them isn't a problem. It just made me wonder.

manthony

Hi, apologies for the slow reply. A very thoughtful comment and you make a good point. I'll correct that. Thank you for pointing it out.

Lynn Krause

Thanks for the tutorial!

I'm trying to get set up using only git, not Mercurial. In Step 4, can I place puttygen.exe somewhere other than the Mercurial folder?

manthony

Sure, you can drop it wherever is convenient.

Lynn Krause

That was quick. Thanks!

Anonymous

That's what she said.

Lynn Krause

ok I was not expecting that. Not on this forum. rofl rofl

Anonymous

Hi all accounts in our team are not accessible using git from windows From the morning. We saw that Bitbucket team did some change in SSH-key using, it cannot been changed from now. Is it possible the reason of the permission reject?

manthony

My suggestion would be you first try, Troubleshoot SSH Issues. If that page does not help you find the issue, please file a report with support@bitbucket.org

Set up Git and Mercurial (Mac OSX)

To use Bitbucket, you need to install a DVCS tool on the computer where you write your code. Typically, this computer is a machine physically close to you like your home or work computer. This is your local machine or system. You also might write or deploy code to a remote machine – for example a lab computer or a server in a data center. You may also need a DVCS tool on that machine too. This tutorial refers to the typical case, your local system, but the instructions are the same for both cases.

Bitbucket supports two DVCS tools, Git and Mercurial. These tools run on all modern operating systems. For Git, Bitbucket supports 1.6.6 or later and Mercurial version 1.7 or later. Mercurial also requires (depends on) the Python programming language. The installation process takes care of making sure you get the correct version of Python.

Since you can use both Git and Mercurial on the same machine, this page shows you how to install both because you need both to complete this tutorial. If you already have these tools installed, skip the instructions and go to the next step in the tutorial.

Windows or Linux User?

Microsoft Windows users see this page.

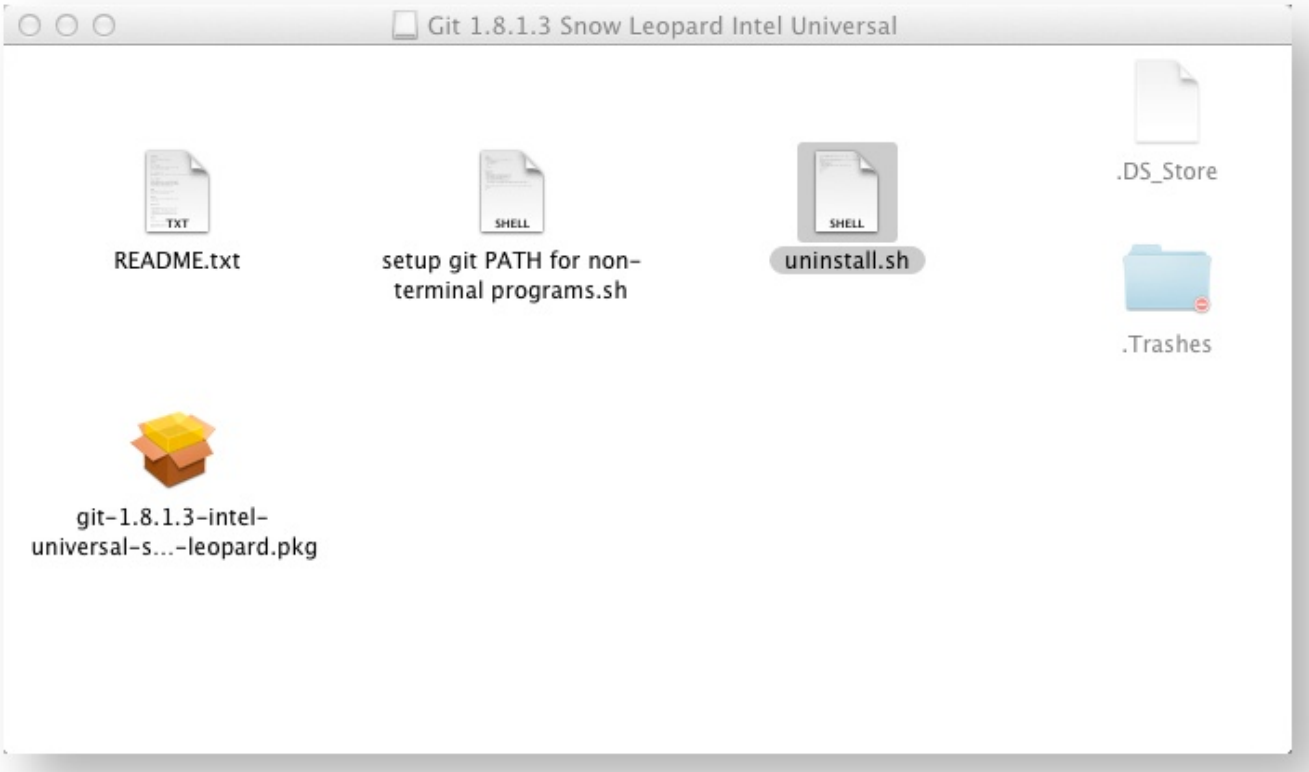
Linux users see this page.

Step 1. Install Git

You can use Git from the command line or you can use one of several GUI-based tools such as Sourcetree. These instructions assume you are using Git from the command-line.

1. Make sure you have root access (sudo) on the system where you want to install Git.
2. Download the Git installer from its official website.

   The installer is a DMG file.
3. Double-click the DMG to expand it.



4. Double-click the PKG file to install it.

   The Git installer launches.
5. Follow the prompts to install Git.
6. Open a terminal on your system.
7. Verify the installation was successful by typing `which git` at the command line.

   ```
   $ git --version
   git version 1.8.1.3
   ```
8. Configure your global username using the following command:

   ```
   git config --global user.name "FIRST_NAME LAST_NAME"
   ```
9. Configure your global email address using the following command:

   ```
   git config --global user.email "MY_NAME@example.com"
   ```

   Git uses the global email address for all your commits. Also, you can set this value per repository (you'll learn more about this later).

Step 2. (Optional) Install the git-credential-osxkeychain helper

Bitbucket supports pushing and pulling over HTTP to your remote Git repositories on Bitbucket. Every time you interact with the remote repository, you must supply a username/password combination. Instead of supplying the combination with every HTTP call, you can store these credentials in your OSX keychain provided you have the **git-credential-osxkeychain** helper added to Git.

The helper asks for your username/password on the first Git operation and then stores the credential. Future operations won't require you

to supply a username/password combination. To install the helper, open a terminal window on your local system and do the following:

1. Check if you have the helper installed by determining if you get a usage statement for it.

   ```
   $ git credential-osxkeychain
   usage: git credential-osxkeychain <get|store|erase>
   ```

   If you receive a usage statement, skip to Step 5. If the helper is not installed, go to the next step.

2. Download the `git-credential-osxkeychain` software to your source with curl;

   ```
   $ curl -O http://github-media-downloads.s3.amazonaws.com/osx/git-credential-osxkeychain
   ```

   This command downloads the source to a local file called `git-credential-osxkeychain`. If you don't have curl installed you can use this link.

3. Move the file to the `/usr/local/bin` directory.

   ```
   $ sudo mv git-credential-osxkeychain /usr/local/bin/
   ```

4. Make the file an executable:

   ```
   $ chmod u+x /usr/local/bin/git-credential-osxkeychain
   ```

5. Configure git to use the helper.

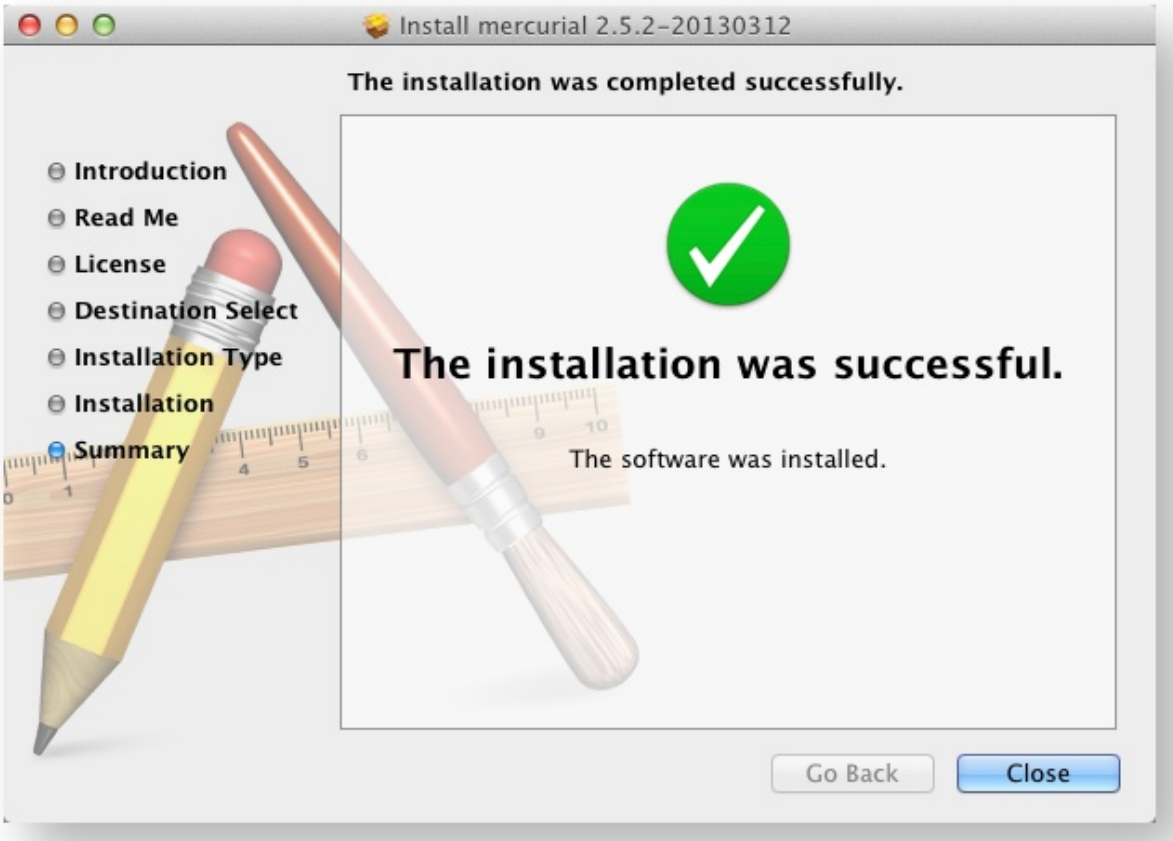
   ```
   $ git config --global credential.helper osxkeychain
   # Set git to use the osxkeychain credential helper
   ```

Step 3. Install Mercurial

You can use Mercurial from the command line or you can use one of several GUI-based tools such as Sourcetree. These instructions assume you are using Mercurial from the command-line.

1. Make sure you have root access (sudo) on the system where you want to install Mercurial.
2. Download the Mercurial installer from its official website.

   The installer is contained in ZIP file.
3. Double-click the ZIP file to expand it.
4. Double-click the MPKG file to run the installer.
5. Follow the prompts to complete the installation.



6. Open a terminal window.
7. Verify the installation was successful by typing the following at the command line.

```
$ hg --version
```

   Hg is the chemical symbol for Mercury and `hg` is the command for mercurial.

8. Determine if you already have a `~/.hgrc` file in your environment by entering the following at the command line:

```
ls ~/.hgrc
```

   If for some reason, you don't have the `.hgrc` file, you should create one yourself using the touch command:

```
touch ~/.hgrc
```

> Files that start with a . (period) are hidden files in Mac OSX. By default, the Finder does not show hidden files. There are several tutorials that show how to show hidden files in finder.

9. Open the Mercurial configuration file `~/.hgrc` using your favorite editor.
10. Add a `username` value to the configuration.

    When you are done, the `~/.hgrc` file looks something like this:

```
[ui]
# Name data to appear in commits
username = Mary Anthony <manthony@atlassian.com>
```

This is default value Mercurial uses, you can also set this for specific repositories (you'll learn more about this later).

11. Save and close the `.hgrc` file.

Next

The next step is to Create an Account and a Git Repo

Like Be the first to like this

18 Comments

Miguel Angel Ivars Mas

Ok. I then understand that my git is fine.

And I think that I have fixed partially credentials error, but once re-installed new git-credential-osxkeychain, as in StackOverflow answer says, when I try to clone repository get another error:

```
fatal: Authentication failed
```

Than seems there are some yet wrong. I searched for a solution and I find this thread: fatal: Authentication failed but recommend me install Homebrew that require install Xcode and for Snow Leopard 10.6.8, which is a size of 4.14Gb! There aren't an alternative solution more simple? And Is mandatory have installed a port (Homebrew or Macports) to continue?

Thank you!

Michael Stassen

You do not *need* any package manager. You can simply install git using the installer provided by git: <http://git-scm.com/>. That said, it appears you already have a working copy of git. The problem is that git has been configured (probably by the installer) with a "credential helper" that is trying and failing to use your keychain for the bitbucket password. In Terminal, enter

```
git config --get-all credential.helper
```

You should get the answer "osxkeychain". The simplest solution is to turn this setting off. This is probably set in your system gitconfig (/usr/local/git/etc/gitconfig). If so,

```
sudo git config --system --unset credential.helper
```

will turn it off (after you enter your login password). If the setting is in your local gitconfig (~/.gitconfig), enter the following in Terminal:

```
git config --local --unset credential.helper
```

Either way, you can verify it is off by entering the "git config --get-all credential.helper" command again and getting no output. Once it's off, the error message will disappear, and you will be prompted for your Bitbucket password whenever you have git interact with Bitbucket, as described in the tutorial.

manthony

Miguel,

Hopefully you are sorted out. I'll be removing this thread in the next day or two.

Cheers,

Mary

Miguel Angel Ivars Mas

Hi Mary and Michael I have fixed error finally. It was related with credential.helper as Michael said. Thank you very much both!

manthony

That is great. I'm glad you got it sorted!

Anonymous

Why get both Git and Mercurial? It sounds as though they do the same thing and I do have limited resources. If bitbucket is a "hosted DVCS service" (whatever that means) - just who does what? Remember, I'm a newbie, I don't have any expectations of offering software to the public in the near future, so why would I want this now? You talk about newcomers to coding, but I'm even newer to developing - I'm just peeping through the door. I came here because the Python tutorial said to. Starting off with an explanation of DVCS and the relationships/rolls of the different players would be helpful.

manthony

Hi,

Thank you for the comment. I understand your frustration. Unfortunately, it is very hard to write one document that everyone at all levels find useful. There is a curve and I try to hit it in the middle with bits on either end to help out those folks. We have some videos to help out people who are investigating: http://www.atlassian.com/dvcs/overview. That should give you an overview. Version control is a developer concept so if you are totally new to developing, you should google around to learn more. This guide assumes that a user has a basic understanding of programming concepts. I'll update the intro to clarify this.

Mary

Jonathan Lafleur

For those who experiment problem with git "comment not found" here is a little workaround :

1. open your ~/.bash_profile:
$ sudo vi ~/.bash_profile

2. add this line to the file:
export PATH=$PATH:/usr/local/git/bin/

3. Save & close the file:
$ ESC, :wq, ENTER

4. Restart your terminal and try :

$ git --version

Hope it helped someone 🙂

Valery Lebedz

Excuse me, but where is step 2? or Mercurial should be the second step?

Thanks.

manthony

Mercurial should be the second step and now it is! Thank you for the catch.

Anonymous

can't find .hgrc file

Anonymous

to open .hgrc open terminal and write

open -a TextEdit .hgrc

Anonymous

Hi everyone, I got this while trying to install Mercurial, any advise??

$ hg --version

'import site' failed; use -v for traceback

Traceback (most recent call last):

File "/usr/local/bin/hg", line 10, in <module>

import os

File "/Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/os.py", line 398, in <module>

import UserDict

File "/Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/UserDict.py", line 84, in <module>

_abcoll.MutableMapping.register(IterableUserDict)

File "/Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/abc.py", line 109, in register

if issubclass(subclass, cls):

File "/Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/abc.py", line 184, in __subclasscheck__

cls._abc_negative_cache.add(subclass)

File "/Library/Frameworks/Python.framework/Versions/7.3/lib/python2.7/_weakrefset.py", line 84, in add

self.data.add(ref(item, self._remove))

TypeError: cannot create weak reference to 'classobj' object

manthony

Sounds like your install failed. Remove the bad installation and try reinstalling Mercurial.

Anonymous

Hi Mary, I tried that already and didn't work. I found a solution in stackoverflow, apparently I'm not the only one...

It consists on emptying the PYTHONPATH, it works, but it's not a good solution, at least for me...

Any help would be appreciated,

Thanks

Here's the link:

http://stackoverflow.com/questions/18090157/python-error-when-cloning-a-repository-with-hg-mercurial/18090158#18090158

Anonymous

I believe the step 2 is wrong. You should swap the order of substeps 3 and 4!

Anonymous

But then you should also use sudo on the chmod command. Better is to change Step 2.3:

```
$ chmod u+x git-credential-osxkeychain
```

Anonymous

You should use this path for part 3 of step 2:

chmod u+x /usr/local/git/bin/git-credential-osxkeychain

Set up Git and Mercurial (Ubuntu Linux)

To use Bitbucket, you need to install a DVCS tool on the computer where you write your code. Typically, this computer is a machine physically close to you like your home or work computer. This is your local machine or system. You also might write or deploy code to a remote machine – for example a lab computer or a server in a data center. You may also need a DVCS tool on that machine too. This tutorial refers to the typical case, your local system, but the instructions are the same for both cases.

Bitbucket supports two DVCS tools, Git and Mercurial. These tools run on all modern operating systems. For Git, Bitbucket supports 1.6.6 or later and Mercurial version 1.7 or later. Mercurial also requires (depends on) the Python programming language. The installation process takes care of making sure you get the correct version of Python.

Since you can use both Git and Mercurial on the same machine, this page shows you how to install both because you need both to complete this tutorial. If you already have these tools installed, skip the instructions and go to the next step in the tutorial.

> This documentation shows you how to install on Ubuntu Linux. If you want to install on Microsoft Windows or Mac OSX, we have instructions for that too.

Step 1. Install Git

Ubuntu uses the apt package management system, which provides the command line utility apt-get and optional graphical interfaces such as Synaptic and Aptitude. We'll use apt-get to install packages, but if you're more comfortable with GUIs, those options are available. Open a terminal window on your local system and do the following:

1. Enter the following command to install Git:

   ```
   sudo apt-get install git
   ```

2. Verify the installation was successful by typing `which git` at the command line.

   ```
   $which git
   /opt/local/bin/git
   ```

3. Configure your username using the following command:

   `git config --global user.name "FIRST_NAME LAST_NAME"`

4. Configure your email address using the following command:

   ```
   git config --global user.email "MY_NAME@example.com"
   ```

Step 2. Install Mercurial

Open a terminal window and do the following:

1. Make sure the `universe` repository is uncommented in the `/etc/apt/sources.list` file.

   To view the file, you can enter `cat /etc/apt/sources.list` at the command line. If the `universe` repo is uncommented the file contains a section similar to the following (example based on Ubuntu 10.10 Maverick Meerkat):

   ```
   deb http://archive.ubuntu.com/ubuntu/ lucid universe
   deb-src http://archive.ubuntu.com/ubuntu/ lucid universe
   deb http://archive.ubuntu.com/ubuntu/ lucid-updates universe
   deb-src http://archive.ubuntu.com/ubuntu/ lucid-updates universe
   deb http://security.ubuntu.com/ubuntu/ lucid-security universe
   deb-src http://security.ubuntu.com/ubuntu/ lucid-security universe
   ```

   If the lines are still commented (have a # prefix), then edit the file and uncomment them.

2. Make sure you have an updated package list by entering the following at the command line:

   ```
   sudo apt-get update
   ```

3. Enter the following command to install Mercurial:

   ```
   sudo apt-get install mercurial
   ```

4. Verify the installation was successful by typing `which hg` at the command line:

   ```
   $ which hg
   /usr/bin/hg
   ```

   Hg is the chemical symbol for Mercury and `hg` is the command for Mercurial.

5. If you don't already have one, create a file named `.hgrc` in your ~ (home) directory.

   This file is the Mercurial global configuration file.

6. Edit the `~/.hgrc` file using your favorite editor.

7. Specify a `username` value.

   When you are done, the contents of the `.hgrc` configuration file look something like this:

   ```
   [ui]
   # Name data to appear in commits
   username = Mary Anthony <manthony@atlassian.com>
   ```

8. Save and close the file.

What's next?

The next step is to Create an Account and a Git Repo

Like Be the first to like this

10 Comments

Anonymous

Step 6 is ambiguous. Do you mean we should manually create a directory called .hg in our home directory (~/)? What should the config file name be?

manthony

Good catch. I think I've fixed the confusion.

Anonymous

the name is .hgrc !

Anonymous

For Ubuntu 12.04 the package git-core is obsolete. You should instead install just git.

manthony

Thanks for the catch. I'll fix that.

Anonymous

What about some instructions for an OS that is far more often being used on live deployment servers in a data center (CentOS).

Anonymous

Any doc for RH-like systems ? Fedora ? CentOS ?

manthony

Not any on the horizon. Were you not able to get started with this?

Anonymous

To start you off.....

As a bare minimum for Fedora 18 simply change "sudo apt-get" for "sudo yum" in the three places above, and (as far as I can see) ignore step 2.1

Anonymous

Isn't it easier to use "hg init" ?

Create an Account and a Git Repo

Now, you get to use Bitbucket! On this page you'll create an account on Bitbucket. Once you have an account, you can create a repository. Here are some quick facts about accounts, repositories, and projects:

- both the username and email on an account must be unique
- each repository belongs to an account (sometimes called a username or user for short)
- the person who created the account is the repository owner
- the repository owner is the only person who can delete the repository
- a user (an account) can have an unlimited number of public and private repositories
- a code project can consist of multiple repositories across multiple accounts but can also be a single repository from a single account

Even if you only intend to work with other people's projects, knowing how to create a repository is useful. Creating a repository is free so there is no reason not to do it.

Who needs an account?

Depending on what you want to do with Bitbucket, you don't even need to create an account. You need an account if you are:

- a new Bitbucket user working through the Bitbucket 101
- a developer who is working on a project whose code happens to be in Bitbucket
- a project owner/leader who wants to share a Bitbucket project (repository) with others
- a developer who wants to contribute to a repository in Bitbucket

Step 1. Create a Bitbucket account

If you already have an account, skip this section and go to the next. You can create an account with the standard sign up or you can create one through OpenID (using an existing Google or Yahoo account for example). Regardless of which sign on method you use, you must supply the following fields:

Field	About what you are supplying
Username	Up to a 30 character username. You can use letters, numbers, and underscores in your username. Your username must be unique across the entire Bitbucket site.

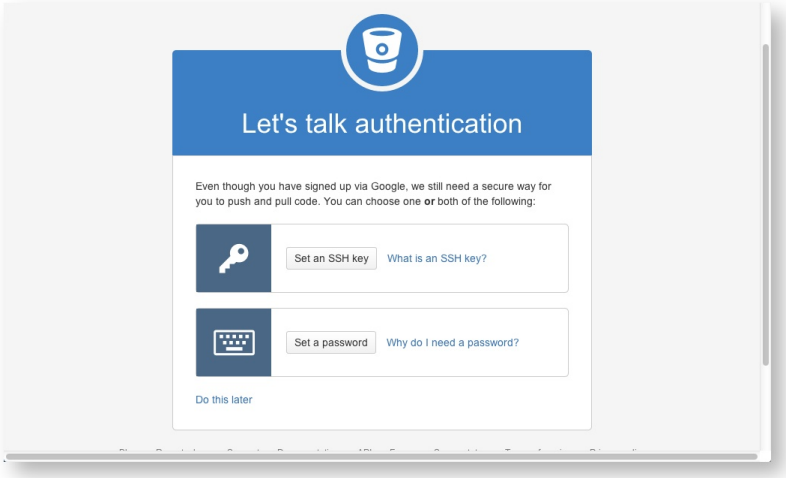
	Bitbucket appends this username to the URL for all the repositories you create. For example, the username **atlassian_tutorial** has a corresponding Bitbucket URL of `https://bitbucket.org/atlassian_tutorial`.
Email address	An email address that is unique across the entire Bitbucket site. The system sends you a confirmation email.
Password	A combination of up to 128 characters. If you are using OpenId the system uses that password. You are responsible for ensuring that your account password is sufficiently complex to meet your personal security standards.

To use the standard Bitbucket sign up, open your browser and do the following:

1. Open https://bitbucket.org/account/signup/ in your browser.
2. Complete the fields on the form.
3. Press **Sign up**.

If you want to sign up using your login from a service, do the following:

1. Open https://bitbucket.org/account/signup/ in your browser
2. Click a service under the **You can also sign up with** heading.
3. Select an account.
4. Follow the provider's prompts to complete the sign up process.
5. When prompted by Bitbucket to authenticate, choose **Set a password**.



SSH is more complex to set up. The tutorial explains setting this up later.

6. Press **Go to dashboard**.

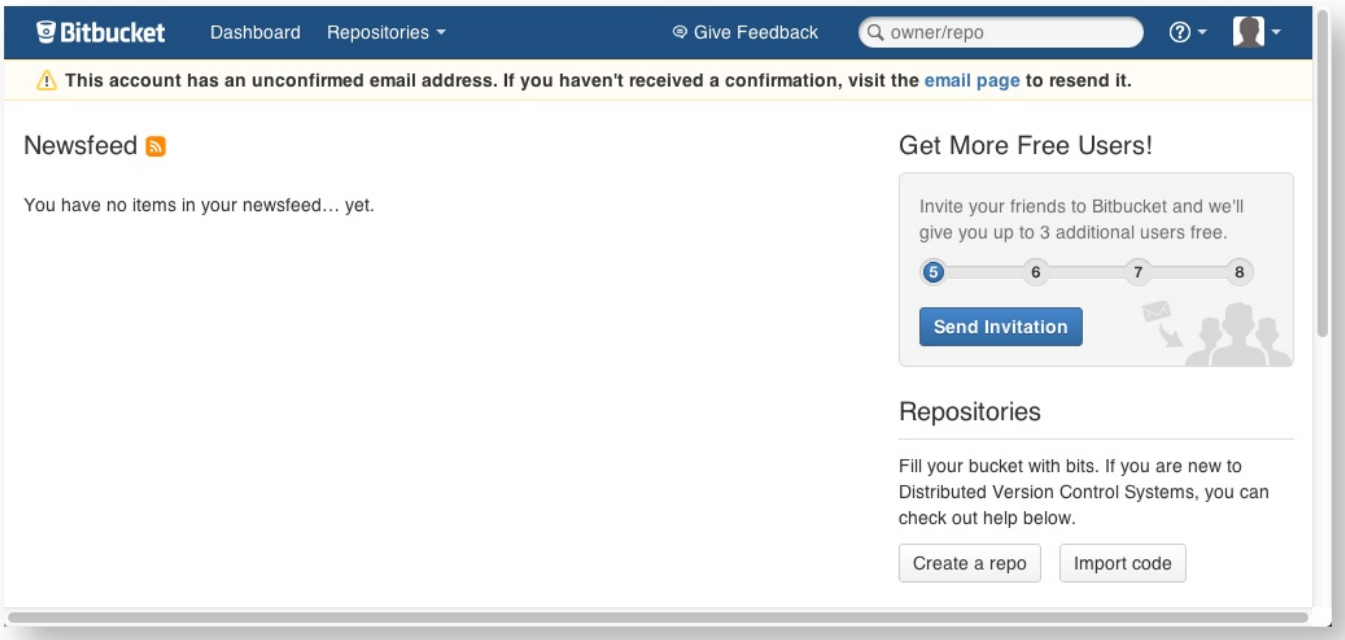
When you are done signing in, Bitbucket places you in your **Dashboard** account. Take a second and look around at the user interface. Across the top of each Bitbucket page is a series of options that let you navigate around Bitbucket. Try displaying the Bitbucket **Keyboard shortcuts** by pressing ? (question mark) on your keyboard.

Step 2. Create a Git repository (repo)

The next step is to create a repository. Initially, the repository you create is going to be empty without any code in it. Do the following to create your repository:

1. Log into Bitbucket (https://bitbucket.org/account/signin/).

The system displays your account **Dashboard** which should be empty if you just created an account.



2. Click the **Create** button at the top of the page.

   The system displays the **Create new repository** page. Take some time to review the dialog's contents. With the exception of the R**epository type**, everything you enter on this page you can later change.
3. Enter `bb101repo` for the **Name** field.

   Bitbucket uses this **Name** in the URL for your repository. For example, the username `atlassian_tutorial` has a repository name `jira-applinks`, the full URL for that repository is `https://bitbucket.org/atlassian_tutorial/jira-applinks`. You can use the URL to quickly navigate to a repository overview.
4. Enter a short **Description**.
5. For **Access level**, leave the **This is a private repository** box checked**.**

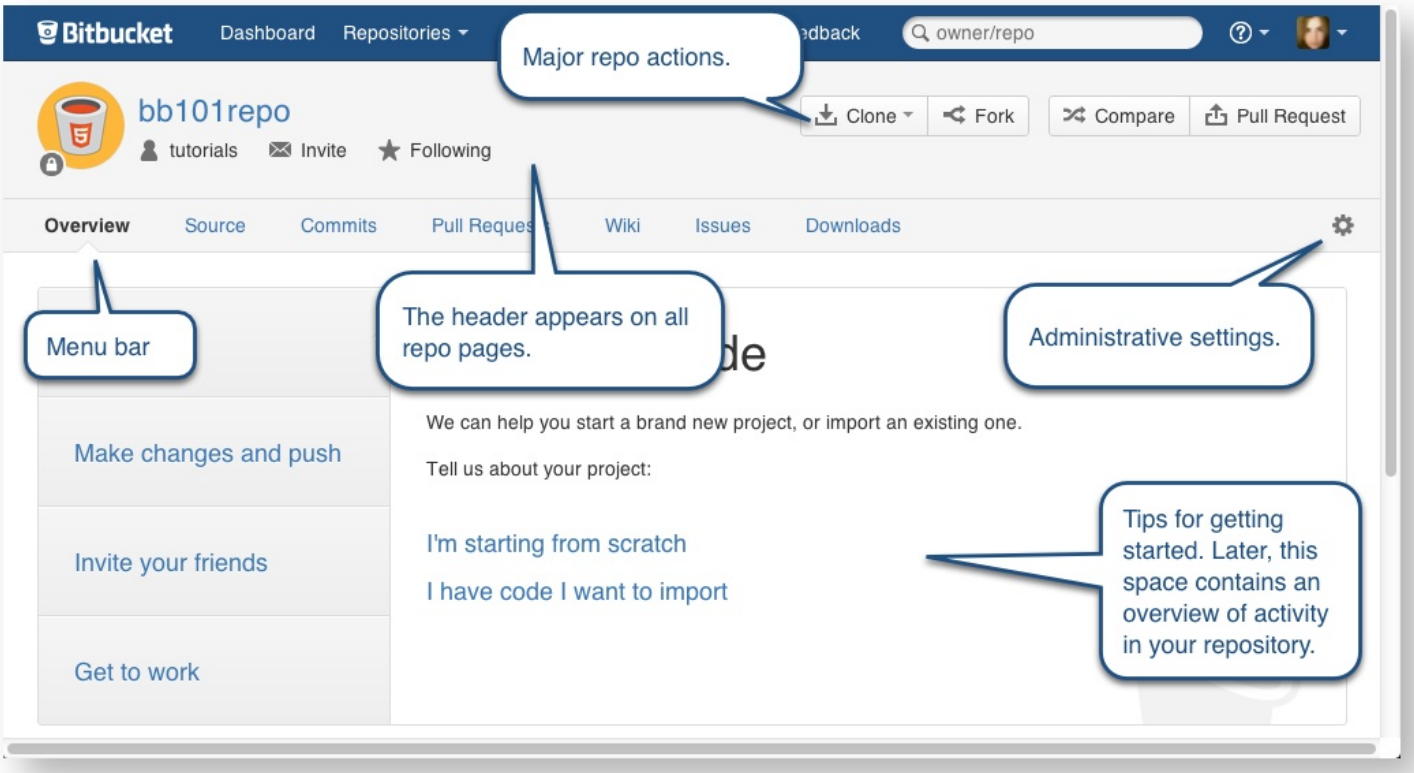
   A private repository is only visible to you and those you give access to (more about this later). If this box is unchecked, everyone can see your repository.
6. Check **Git** for the **repository type**.

   You can't change the repository type once you pick one. For this tutorial, we create a Git repository first. Later we will work with a Mecurial repository.
7. For **Project management**, check both the **Issue tracking** and **Wiki** checkbox.
8. Set the **Language** field to **HTML/CSS**.
9. Click **Create repository**.

   Bitbucket creates your repository and displays the repository **Overview** page.

Step 3. Explore your new repository

Take some time to explore the repository you have just created. You should be on the repository's **Overview** page:



Across the top of each repository is a menu bar that you use to navigate around to your repository's pages. Click items on the bar to see what is behind each one. You can also navigate using the repository shortcuts. To view the shortcuts available:

1. Click somewhere in the Bitbucket browser window to gain focus.
2. Press the ? (question mark) on your keyboard.
   The shortcuts list displays.

Try clicking the **Commits** option on the menu bar. You find there are **No commits recorded yet** (during your repository exploration you probably saw this message as well). You have no commits because you have not created any content for your repository. Your repository is private and you have not invited anyone to the repository. So, the only person who can create or edit the repository's content right now is you, the repository *owner*.

Step 4. Confirm your email address

When you create an account, Bitbucket sends an email to you to confirm your email. Confirming your email allows Bitbucket to automatically match your user account to your future source code commits. Do the following:

1. Open your email client (Gmail, Outlook etc).
2. Find the email confirmation from Bitbucket.
3. Open it.
4. Confirm the address.

Next

Now that you've created your first repository and explored it a bit, you are ready to add content to it.

Like Kamran Mackey and 4 others like this

44 Comments

Anonymous

Is it really needed to have a four step process describing how to confirm your email address?

manthony

An interesting question. I design documentation for novices as they need more help than experts. Also, of course, you might be a native English speaker not all of our users are. We do get support issues on lost confirmation email. Finally, consider over 9,556 people have visited this page. You are the first to comment on the number of steps. That isn't to say your comment is false/bad/whatever. It does imply that for those who found the length of the procedure irksome, only one took the time to comment. So, more than likely an outlier.

Anonymous

@ Mary Anthony [Administrative Account]

You're 100% right.

Personally, I admire your work and how you make it easy to understand for novices.

Some people are inconsiderate, forget that they were once novices too, needed all the help they could get.

Not everyone was born a coder or a computer techie.

Will.

Anonymous

Don't listen; your docs are great and proved useful every step of the way. It's easier for me to skip over something because it's trivial than for me to be staring blankly at the screen because you left something out!

Anonymous

Yup. Probably more than necessary. But I'd much rather have too much information than not enough. Keep up the good work.

Brian Yang

You did really a good job, you are a serious writer and you deserve good comments! leave those guys who cannot understand people's work.

manthony

I'm glad you like the docs Brian! To be fair, your comment reminded me, I really should check with Support; It could be lost confirmation emails are no longer the issue it was in 2012.

Anonymous

Actually I didn't get any confirmation email when creating my first repository!

manthony

You can actually get the system to resend the confirmation.

Anonymous

/owned

Anonymous

Good tutorial!

Anonymous

I'm not a novice and didn't notice the four lines until I'd confirmed my email etc. but blimey, how nice to have full, helpful documentation rather than the short-cutting rubbish you normally find especially in the open-source community. As a professional developer not in need of that help, I'm left with an impression of serious professionalism by such clarity. Well done 🙂

manthony

Wow. Thank you. Made my day.

Muhammad Moosa

@manthony

Really you are doing well. Four step email confirmation is not necessary but the beginner level can not be measured specially in such type of tutorials. So I like your detailed way. There is no under estimation. Bravo.....

manthony

Muhammad thank you for that comment. It is tricky to write one tutorial for such a diverse population as the bitbuckians. Every comment helps me tweak — so I very much appreciate them.

Anonymous

Full documentation is a good thing. I already confirmed my email before reading this tutorial, but I still respect, and appreciate the full documentation.

Anonymous

This is only instructions to create a git repo. How do you create a repo using mercurial?

manthony

You can create a Mercurial repo at any time by selecting Mercurial from the **Create new repository** dialog. As you progress through the tutorial, you will fork a Mercurial repo.

Anonymous

Great tutorial, i like it (where is the like button..., just kidding). There is a minor typo in Step 2, point 3: "has arepository" should be "has a repository".

manthony

Hey thanks on both counts! Typo wasfixed — just kidding was fixed. ;-D

Anonymous

You should be given high praise for taking the large amount of time that I am sure this tutorial took to create to make an excellent tutorial. I truly believe that a more detailed tutorial is of greater value than a quick brief tutorial....Advanced users can quickly skip stuff they know but beginners do not have that option if the information is not there...Some developers have to remember that not everyone using this service would be a git expert or even a command line expert. Similar to what an earlier commenter experienced, I too have come across many open source tools where the documentation is very brief and assumes everyone is a command line expert or is on a Unix/Linux Mac OS/X system where the command line is called a command line...in Windows it is called the Command prompt...Anyways I have found this tutorial excellent...Thank you for making it...Do not listen to the cranky nerds, keep it detailed for the beginners.

manthony

Thank you for taking the time to write this. You pretty much encapsulated my own writing philosophy. The goal is to write to the novice and not hinder the experienced folks. Course, now I have to go convince my team that I did not pay a friend to write a comment for me...or maybe I wrote it in my sleep.  Anyway, thank you very much I appreciated hearing this.

Anonymous

thank you for the tutorial. I came from svn and I was having a lot of difficult to use git. but now everthing is clear.

manthony

You are welcome — and welcome to Git!

Anonymous

Thanks for your efforts in writing/editing this Mary.

The only suggestion I've got would be a clearer picture of you.

Anonymous

What impact does the language field have?

manthony

None. It just is a way of providing information.

Cuitlahuac Hernandez-Santiago

Muchas gracias por tu detallada explicacion. Es muy util para novatos como yo.

Thank's a lot for your detailed tutorial. It is really helpful for novices like me.

---

cesar monges

Great job, this tutorial is very helpful!

Regards from Argentina

manthony

Thank you Cesar from Argentina! Very cool.

Anonymous

Thank you very much

Anonymous

Thank you for your efforts in writing the tutorial. Very neat and detailed.

Anonymous

Thanks for the tutorial, agree that the not-even-a-second it takes to scroll through four email confirmation steps is worth it.

Anonymous

It's easier to confirm your email address if you copy+paste the link into the same browser where you created your BitBucket account. If you signed up with your non-default browser, you'll have to login again when the link uses your default browser.

Neil Dey

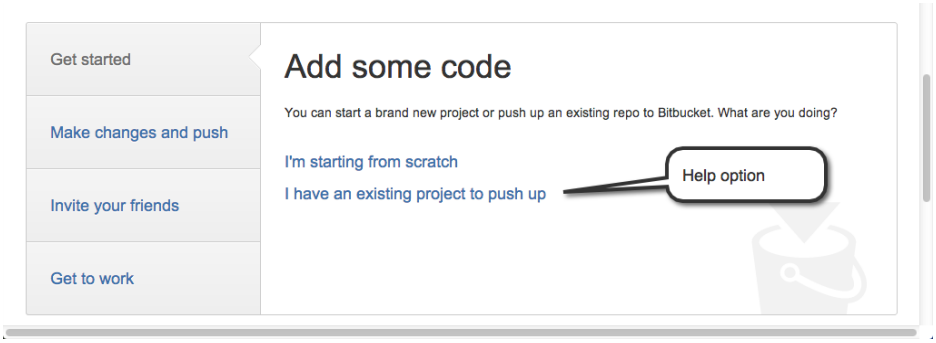
Question: How can I already add the contents of an existing folder to git version control?

Tutorial here covers the case of making a directory and then adding source contents to it. I have some source code in a folder that is path dependent and dont want to move it.

So - How can I just go into my folder and make it a repository?

manthony

Neil, yes the tutorial is very specific because I'm trying to teach many folks the basics. Bitbucket actually provides online help for just your case. Right after you create a repo, the system dumps you into a blank repo with a help panel:



You'd choose the second help and then the system displays the appropriate commands. In this case, they are:

```
cd /path/to/my/repo
git remote add origin ssh://git@bitbucket.org/tutorials/bb101repo.git
git push -u origin --all
```


Anonymous

I'm so impressed with how friendly your docs are. Great job for making it easy and understandable especially for beginners. Regards from the Philippines!

manthony

Thank you always glad to help. And hello from California!

Patrick Boisclair

I created a new repository and added a Group of users. Now when I go to the overview tab, I see the "Get started". When they go there, they see "No commits recorded yet. ". If I make one of them part of the Administrator group, that one can now see the "Get started". What am I missing?

manthony

Well, at this point, someone with write permissions (you, the other admin, or a group member if that group has write) needs to push code to the repo. Just following the Get Started instructions. (Apparently,we only show those instructions to Admins and so I'll see about removing that restriction on the instructions.)

Patrick Boisclair

Just to confirm that the user group mentionned was setup with "Write" access, but no "Admin" access. So it confirms that only Admin users sees the messages on how to upload code.

manthony

Patrick thanks for the confirmation. That is exactly what I thought.

Anonymous

Hi,

I'm getting at the "git push ..." step: "fatal: Could not read from the remote repository."

What step is it likely that I flubbed?

Thanks,

mg

manthony

Hmm, it is hard to diagnose just on the basis of your comment. Check everything you type against the directions on the page. If you still have problems, please paste the command you are trying and the error messages in your comment.

Clone Your Git Repo and Add Source Files

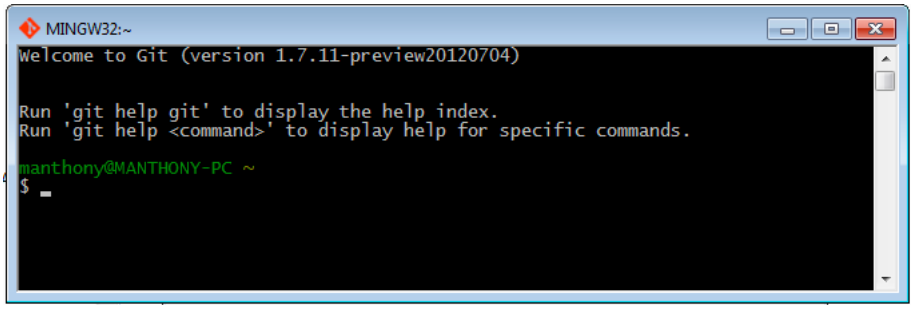
On this page, you'll learn how to add code to your repository. At this point, you have created a `bb101repo` which is a private repository that only you, the *owner*, can see or work in. This is not typical. Usually, you want some help on a project and you won't be the only one working in a repository. However, this is a tutorial and right now you need to learn without being distracted by other people. We'll add more people to your project later.

> **Important reminders about these instructions**
>
> The `git` commands on this page are for a GitBash terminal. However, the commands are nearly identical in a Mac OSX terminal and in an Ubuntu Linux terminal. You shouldn't have any trouble following along but if you do, please comment on the page and we'll make corrections
>
> This tutorial teaches you a little Git first and later a little Mercurial. All the operations that you do with Git have Mercurial equivalents and of course the reverse is also true. You are free to use Mercurial exclusively with this tutorial if you want, but you'll need to do that on your own.

Step 1. Clone your repository to your local system

Open a browser and a Git Bash window (also called a *terminal window*) from your desktop. The terminal window on your local system which is the system you code on as opposed to the remote Bitbucket server. After opening the terminal window, do the following:

1. Navigate to your home (~) directory.



   As you work with code, you will find that you have multiple repositories that you work in. It is a good idea to create a directory to contain all those repositories. Generally, this is your home directory but it can be anywhere you want it to be.

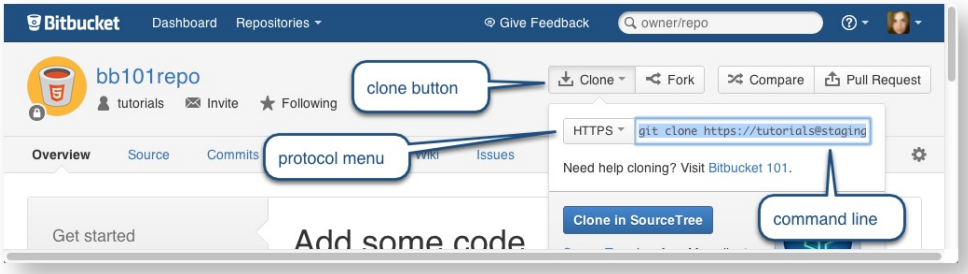
2. Create a directory to contain your repositories.

```
mkdir repos
```

3. If you haven't already done so, go to Bitbucket in your browser and log into your Bitbucket account.
4. Go to your `bb101repo` **Overview** page.
5. Click **Clone button**.

   The system displays a pop-up clone dialog. By default, the clone dialog sets the protocol to **HTTPS**. Leave it there.
6. Click on the command line field that accompanies the protocol.

   When you click on the command field it automatically highlights the command for you.
7. Copy the highlighted `git` command.



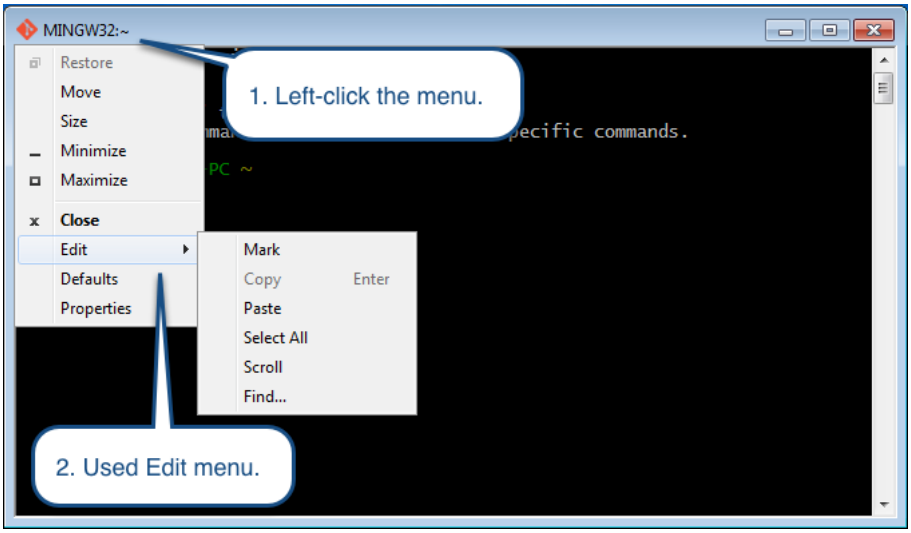
8. Switch back to your terminal window.
9. Change directory to your new `repositories` directory.

```
cd repos
```

10. Paste the command you copied from Bitbucket onto the command line and press **Return**.

    Click to view GitBash cut 'n paste tips...

    Use the menu to go into **Mark** mode and then select text. Pressing **Enter** adds the text to the clipboard. Paste text by pressing `INSERT` on your keyboard or using the menu.



    Git will ask you for the repository password. This is the password you entered when you created your Bitbucket account. If you created an account by linking to Google or Facebook and you are following this tutorial, you should have created a password already.

11. Enter your password.
    Git creates the repository but warns you that you have cloned an empty repository. At this point, your terminal window should look something like this:

```
$ cd ~/repos
$ git clone https://newuserme@bitbucket.org/newuserme/bb101repo.git
Cloning into 'bb101repo'...
Password
warning: You appear to have cloned an empty repository.
```

    You already knew that your repository was empty right? Recall that you have added no source files to it yet.

12. List the contents of your `repos` directory and you should see your `bb101repo` directory in it.

Step 2. Explore your repository and fix a problem

Git said you had cloned an empty repository. This is true, your repository is empty but the directory that git created is not entirely empty. List the contents of your *local repository* – including the hidden files. A local repository is the copy of a repository you have on your local system. After listing the contents of your local repository, you should see something like this:

```
$ ls -al bb101repo/
total 0
drwxr-xr-x 3 manthony staff 102 Dec 14 10:50 .
drwxr-xr-x 3 manthony staff 102 Dec 14 10:50 ..
drwxr-xr-x 9 manthony staff 306 Dec 14 10:50 .git
$
```

The `.git` directory contains special files and directories used by the Git system. For now, you should be aware that it is there.

Notice that Git went ahead and named your repository's directory with the same name as you did when you created it. You could have cloned the repository under a totally different name. For example, you could cloned the repository to a directory called `bb101repo-practice`. The advantage of this name is that it provides a clue about what you were going to do with clone. In fact, it is good not to simply use the repository name when you clone but indicate what you are doing with the clone. Why don't you fix that right now:

1. Remove the repository you just created.

   ```
   rm -irf bb101repo/
   ```

2. Reissue the clone command but this time give Git a name that indicates what you are doing.

   For example, you might want to call the clone `bb101repo-practice`:

   ```
   $ git clone https://newuserme@bitbucket.org/newuserme/bb101repo.git bb101repo-practice
   ```

   Again, Git will tell you that you are cloning an empty repository.

3. List your `~/repos` directory, you should see something similar to the following:

   ```
   $ ls ~/repos
   bb101repo-practice
   ```

Great. Now you are ready to add content to your repository.

Step 3. Create a README, add it locally, and push it to the Bitbucket Server

Bitbucket lets you set a repository's description but you may want to provide detailed information or instructions about your repository. You do this in a README file that is at the root of your repository's code base. There are several formats you can use to create a README. The steps below guide you through creating a plain text README in your local repository, adding the file to tracking, committing your change locally, and pushing the file up to the remote Bitbucket server.

1. Go to your terminal window and navigate to the top level of your local repository.

   ```
   cd ~/repos/bb101repo-practice/
   ```

2. Using your favorite editor, create a `README` file with the following content:

   ```
   Welcome to My First Repo
   ------------------------------
   This repo is a practice repo I am using to learn Bitbucket.
   ```

3. Save the `README` file in your `bb101repo-practice` directory.
4. When you are done, list the contents of your local repository.

   You should see something similar to the following:

   ```
   $ ls -al
   total 8
   drwxr-xr-x 4 manthony staff 136 Dec 14 11:50 .
   drwxr-xr-x 3 manthony staff 102 Dec 14 11:30 ..
   drwxr-xr-x 9 manthony staff 306 Dec 14 11:30 .git
   -rw-r--r-- 1 manthony staff 117 Dec 14 11:50 README
   ```

5. Go ahead and get the status of your local repository.

   The `git status` command tells you about how your project is progressing in comparison to your Bitbucket repository. You should see something like this:

```
$ git status
# 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)
```

You can see that Git is aware that you created a file in your local repository. The status output also shows you the next step, the add.

6. Tell git to track your new `README` file using the `git add` command.

```
git add README
```

You must perform this step before you can commit a file. What happens if you run the `status` command now? Git sees that you have a new file in your local repository and that you can potentially commit it.

7. Commit all the changes you added.

Right now, the only change pending in your local repository is the new README. When you issue the `commit` command you see this:

```
$ git commit -m "adding repo instructions"
[master (root-commit) 12ad229] adding repo instructions
 1 files changed, 3 insertions(+), 0 deletions(-)
 create mode 100644 README
```

Up until this point, everything you have done is local – that is on your local system. It is not visible in your Bitbucket repository until you put it there with the `push` command. You can test this to see if it is true by opening the Bitbucket `bb101repo` **Overview** page in your browser. You should see that the **Overview** looks exactly as it did when you started.

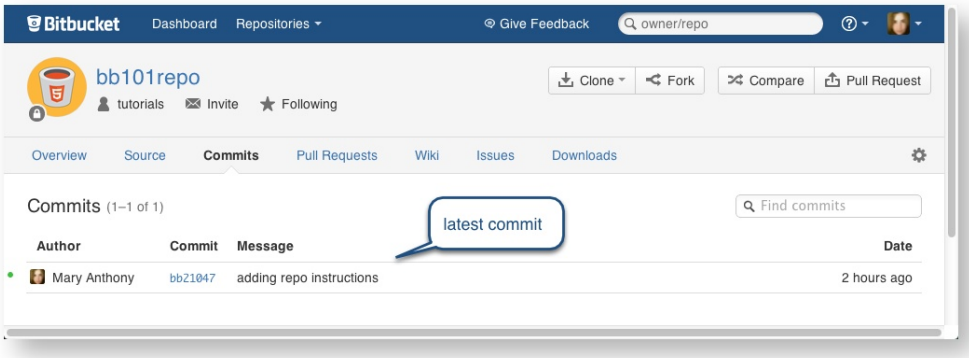
8. Go back to your local terminal window and push your committed changes using the `git push` command.

You should see something similar to the following:

```
$ git push -u origin master
Password:
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 303 bytes, done.
Total 3 (delta 0), reused 0 (delta 0)
remote: bb/acl: newuserme is allowed. accepted payload.
To https://newuserme@bitbucket.org/newuserme/bb101repo.git
 * [new branch]   master -> master
Branch master set up to track remote branch master from origin.
```

9. Go to your Bitbucket **bb101repo** repository in your browser and click the **Commits** item on the menu bar.

You should see a single commit on your repository.



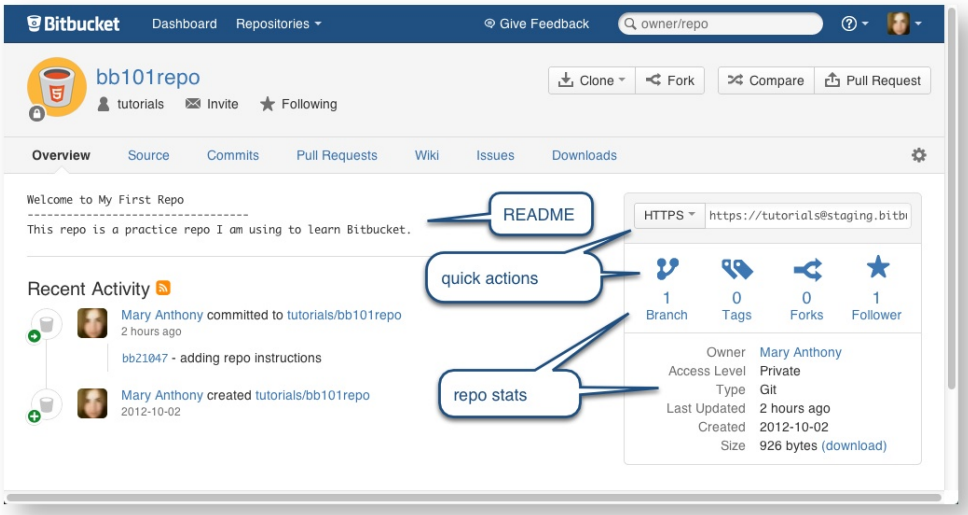
Bitbucket traps a lot of information with your commit and shows it to you. You can see that the **Author** column shows the value you used when you configured the Git global file (`~/.gitconfig`).

10. Select the **Source** option.

You should see that you have a single source file in your repository, the `README` file you just added.

11. Navigate the **Overview** page of your repository.

    You should see the contents of your README displayed.



Remember how the **Overview** page looked when you first created your repo? Take some time and explore a little. Click on buttons and travel down some links.

Next

Now, you have finished the simplest workflow between Bitbucket and a DVCS system. You created a local repository, added a new file to it, and pushed those changes to the Bitbucket repository. At this point you may have a few thoughts in your head such as:

- Hey, I don't consider a README file to be source code.
- Aren't other people supposed to help me with my repository?
- This Git stuff is fine but I want to use Mercurial.

The next page of this tutorial will resolve some of these thoughts because you will use the fork operation and use Mercurial to add code to another user's repository.

154 Comments

Anonymous

I thoroughly enjoyed this tutorial , very well written and the instructions are easy to follow, kudos.

manthony

Thank you for this lovely comment.

Anonymous

yeah I agree with this Anon

Anonymous

And I agree with this Anon agreeing with that Anon

manthony

Anonymous

I also agree with Anon, Anon and that Anon. Oh and also the Anon below this comment.

manthony

😀🙂😀

Anonymous

Hodor?

Anonymous

I find it interesting how all these anons could be the same person which isn't me.

Anonymous

this is the excellent tutorial , very well written with screen shot and all instructions are easy to understand, Shakar

Thanks a lot

Anonymous

Nice work guys! Just getting started with BitBucket and my first git repository every and have been setup without any issues and I can understand what I am doing in the background!

Anonymous

I agree - this is a great tutuorial.

Do you have anything similar for SourceTree? I've read good things about and have it installed but I can't find any documenation to get me started (except the YouTube video which is more marketing focused)

Thanks!

manthony

Glad you like the tutorial. I'm sorry but we don't yet have one for Sourcetree. If you have Sourcetree open you should have the help installed. **Help > Sourcetree Help** that could help but not a tutorial of course.

Anonymous

Neatly explained. Thank you 🙂

Muhammad Moosa

"Excellent" comment from Excellence, for Excellent tutorial and author.

Anonymous

Its clear for all levels of Developers..🙂🙂

Anonymous

Very VERY helpful tutorial, great way to get started and it covers the necessary basics.
Would be nice to add also a windows version with the windows commands for people that don't know the equivalent. 🙂

manthony

Glad you found the tutorial helpful! It always makes my day to hear that. 😀 Can you help me understand your suggestion better? The commands on this page were all executed in a GitBash terminal on a Windows 64 system. When you mean "windows version" do you mean some Windows client other than GitBash? It is perfectly possible a new one was developed that I know nothing about and I love to test new client software.

Anonymous

I would have logged in with my BitBucket account, if I could have, but I won't create another account just so I can give a tip:

You can simply hit the INSERT key in Git Bash to paste stuff. Should fit perfectly with all these no-mouse console-freaks 😃

manthony

Thanks, I have instructions on how to do that in Step 1.10 above. It is in an expand box if the reader needs some tips. Some users need those tips others don't.

Anonymous

thank you Mary!!! Extremely helpful tutorial...

manthony

Glad to help! Have fun out there new Bitbuckian!

Anonymous

I'd have gone with "Bitbucketeer" 😛

manthony

Trendy. I guess I'm a classicist. 😃

Anonymous

I gotta agree with many other comments on this blog.

This is an extremely well documented tutorial. It is "deceptively" simple and elegant. These publishing qualities demand a research, empathy and planning. Well done to you all.

It's fun too!

manthony

This made my morning. Thank you for the holiday gift!

Anonymous

Wonderful tutor , Thanks

manthony

You are welcome!

Anonymous

thanks for taking the time to create this easy to follow tutorial!

manthony

You are welcome!

Anonymous

This is one of the best documentation i have ever seen in my life

manthony

I love this. Course, I'm never going to convince anyone I didn't make the comment anonymously myself. 😛😃

Anonymous

I'm having trouble with the command

```
git add README
```

on Mac OSX terminal. I get the following error:

fatal: Not a git repository (or any of the parent directories): .git

manthony

You'll get this message if you issue a Git command in a file system location where you have not initialized a Git repository. Make sure change directory to a repository or a subdirectory of a repository before issuing the command.

Brian Anderson

I got a little stuck on Step 1 (10) "Paste the command you copied from Bitbucket onto the command line and press **Return**."

I didn't know how to paste into git bash. I tired Ctrl + V, right clicking the black area of the window, but I found I had to right click the actual window for the little menu to come up where you can select Edit and then select Paste.

Your tutorial is great, I just though mentioning something like this would help out.

manthony

Hi Brian,

Thanks for the feedback. I've tweaked that step to include a quick tip on GitBash editing. Let me know what you think.

Mary

Brian Anderson

You are awesome! Thank you so much for having this tutorial!

Anonymous

Hey, I had trouble with $ ls ~/repos

fixed that when I did

$ls ~/ repos

I couldn't sign in because I signed up with my github, what's the trick?

Anonymous

Why mine is like this below and how to fix it? I even tried to rename my file from readme.txt to just readme (as your text editor seems that saved it to readme instead of readme.txt) but no chance.

```
{\rtf1\ansi\ansicpg1252\cocoartf1138\cocoasubrtf510
{\fonttbl\f0\fmodern\fcharset0 Courier;}
{\colortbl;\red255\green255\blue255;}
\paperw11900\paperh16840\margl1440\margr1440\vieww16200\viewh13200\viewkind0
\deftab720
\pard\pardeftab720

\f0\fs24 \cf0 Welcome to My First Repo\
-------------------------------\
This repo is a practice repo I am using to learn Bitbucket.\
}
```

manthony

Hi,

It appears you were working with Word or some other document editor. Your editor added a number of control sequences such as:

```
{\fonttbl\f0\fmodern\fcharset0 Courier;}
```

You should use Notepad to edit text files. Search for code or text editor on Windows.

Mary

Anonymous

just wanted to say thanks for these great tutorials, keep up the good work!

Anonymous

I'm doing this in mercurial and am stuck on step 3.6: the "hg commit -m" command doesn't seem to work. How do I commit in mercurial?

Great tutorial up to this point!

Anonymous

Could I suggest you add a note that Git Bash/Gui must be run with administrative rights in order to git clone to your User directory.

The error given is Permission Denied(publickey) which can be very confusing.

(At least in Windows 8)

manthony

This looks like a SSH issue not a local permissions issue. Did you choose to clone with SSH syntax instead of HTTP by chance? I'll verify. I haven't tested with Windows 8 yet.

Anonymous

Under Step 3 I am a bit confused on how to get from step 2 to 3. Once the README file is created in Notepad, how do we get it into our repository is order to list it?

manthony

You should create the file within the repository directory. In Step3.1 you change into that directory.

Anonymous

Very good indeed. Easy to follow. Thanks.

Anonymous

Very useful! Thank you so much for taking the time to write this out and add very useful images. Will refer to all of these guides as needed,

Cuitlahuac Hernandez-Santiago

Thank's for your work. For non-tech savvy people like me, in Step 3. where it says:

> "2. Using your favorite editor, create a `README` file with the following content:"

would be more comprehensible to say (or something similar):

> "2. Using your favorite editor, create a `README` file with the following content and save the file into the folder
>
> *C:\Users\your_user_name\repos\bb101repo-practice*
>
> where all this process are performed"

manthony

Cuitlahuac,

I've incorporated your suggestion into our instructions. Thank you for the suggestion!

Mary

Teresa Andel

This part is still confusing to me. I have a file created on my local workstation. I cannot find a local folder anywhere named the same as my repo (*bb101repo-practice).* Do I need to create this folder manually? (I thought the clone process would have put something on my machine, but I'm not finding it.

Teresa Andel

I thought I'd post an update to this, as it appears I might have reacted too early. The repos folder did actually get created in my local environment, but to a mapped (home) drive on the network. I found this by listing the files in the ~ folder in the terminal window and recognized where it was at that point. So, it is wherever your home directory is configured.

manthony

Glad you got it sorted!

Peter Le

These are some really great tutorials thanks to the good use of images. I thought I was going to get lost for a second there and I noticed a handy image sitting below the point for me to follow 🙂

Anonymous

```
[user@centos myrepo]$ git clone https://username@bitbucket.org/atcm/myrepo.git .
Initialized empty Git repository in /home/user/murepo/.git/
error: Failed connect to bitbucket.org:443; Operation now in progress while accessing https://username@bitbucket.org/atcm/myrepo.
fatal: HTTP request failed
[user@centos myrepo]$
```

the host is connected to the internet via proxy. when doing the same thig (git clone) from the machine in the same network - there is no problem. what am I doing wrong?

manthony

Hmm, I'm not sure what is going on here. Please file a report with support@bitbucket.org and our Support team will help you solve this.

Anonymous

Was your problem solved ?? I am also facing same problem.

Anonymous

Just pass this command

git config --global http.proxy http://10.3.100.211:8080

Anonymous

I see how to clone a git project. What if I want to grab a single file from the Downloads directory using curl or wget? The URL for a single file does not seem to be working no matter what user credentials I pass?

manthony

You can use a `curl` call to GET a single source file like this:

```
curl https://api.bitbucket.org/1.0/repositories/tutorials/tutorials.bitbucket.org/raw/default/index.html
```

jsolderitsch

Sorry, don't want a source file, I want a file in my Downloads directory in my private repo. The URL that the browser reports is:

https://bitbucket.org/jsolderitsch/bb101repo/downloads/SecureWebinarpdf.pdf

What is the curl command for this?

Thanks

manthony

Unfortunately, we don't have a CURL for this. I've created a ticket for this: iIssue #6915

jsolderitsch

I have some good news to report – this command works:

curl -L -O "https://bitbucket.org/jsolderitsch/bb101repo/downloads/SecureWebinar.pdf" --user jsolderitsch

BitBucket tech support suggested the -L option to curl.

manthony

Thanks! Yes, I just saw this ticket in Support. You can download with this curl call but it isn't part of our REST API. Many people want to POST files to the Downloads area and that is not supported. I'll tweak the ticket.

Ian Barton

I ran into issues while trying to clone into the newly created repo.

I'm using OAuth from Google, so I did not setup a password for my account, and I am unable to clone via the SSH or HTTPS link. I might suggest a note for those using OAuth to determine what to do for that step.

manthony

Hi Ian,

Thanks for the suggestion. I'll add a note.

Mary

Anonymous

Mr. Barton took the word right out of my... mind? I had the same problem and I agree this should be covered here.

David Murray

Mary, great tutorial and documentation! Thank you! For an application to be successful and widely adopted it is important to make entry-points available for all levels of users and knowledge levels. You are making great strides in this direction.

I have some recommendations on how to improve this portion of your documentation from my perspective based upon what I was looking for and what would have helped me:

- I have used Git before intermittently, so I am a little weak on terminology and need something to help translate commands to "plainspeak." The title, "Clone Your Git Repo" was somewhat confusing because what I wanted to do was simply download the source from my repo locally so I could work on it on my computer. After reading through the whole tutorial it is apparent to me that cloning a repository is how you get a "copy" of it, but adding a brief explanation at the beginning of the tutorial saying, "this is how you get a local copy of the repo to work on" would be really helpful.
- Sorry if this one seems very obvious already... I would find it useful if, in the first paragraph, you would mention that this tutorial is based upon a brand new - **empty** - repo as if you were creating a new app from scratch, as I didn't come in at the start of the tutorial series. Instead, I came to this page after copying over an existing (populated) repo from github and my goal was to download the project to my local machine so that I could work on it. I was initially confused as to why Git would warn me that my repo was empty. As I go back through the tutorial I see the sample repo comment for the README, "This repo is a practice repo I am using to learn Bitbucket," and it is a little more obvious, but adding a really quick explanation at the top of the tutorial would have helped me out.
- I see another user has asked why they were getting Git error messages and you indicated it was because they probably had tried executing the command in a non Git initialized directory. It would be helpful if you added a quick note in Step 1 where you run the clone command to copy the repo locally, stating that cloning automatically initializes that project directory with Git, similar to running the Git init command in a directory.

Thank you so much and cheers! This is great stuff. If there is another tutorial that is more aligned with starting out having a project already populated (a new contributor to an existing repo, but trying to learn the basics of bb, Git, etc), please point me to it... maybe a good idea for additional documentation.

Cheers! Keep up the great work!

manthony

David,

Thank you so much for your comments and the compliments! I'll incorporate them as appropriate.

Just as an FYI, I delete the comments I incorporate to avoid confusing other readers.

Mary

David Murray

Awesome! Glad to help. Another thing you do that is excellent is that you are very active in the community and in your responses, which in-turn helps drive up the value of your docs and products.

I wondered if you had a tutorial specific to the idiosyncrasies of getting started as a new contributor to a Rails project... things like... do I still want to change the project folder name since the .rvmrc file is tied to the folder name (and therefore repo folder name)... does that make sense? I can provide a specific example if it helps.

My request is a little more in the line of specific Rails project help, but bb is a nature tie-in because that's where my repo sits.

Anyhow... any thoughts, recommendations... is this something you all would handle or find value in offering?

manthony

David,

Thank you for that comment! My goal is to be the user advocate on the ground if you will. I don't have a tutorial specific to Rails, this is the first time the topic has come up. It might be good to start with a Rails tip page in our FAQ. If you have an example or just a list of what you ran into, I can try and work one up.

Mary

Anonymous

Hello,

Thanks for this tutorial and free private repo service. I am facing clone problem. Can i discuss here? Thanks

Regards

Sohail

manthony

Sohail,

You can discuss it here. Or, you can send an email to support@bitbucket.org and get personal and often faster help.

Mary

Anonymous

Hi Mary,

Thanks for your wonderful tutorial. I am getting an error when i am trying to push the newly created REAME file to the server. Here are the commands that i typed and their output:

```
Spencer@SPENCER-PC /d/repos
$ cd testrepo-practice/
Spencer@SPENCER-PC /d/repos/testrepo-practice (master)
$ git push -u master
warning: push.default is unset; its implicit value is changing in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the current behavior after the default changes, use:

  git config --global push.default matching

To squelch this message and adopt the new behavior now, use:

  git config --global push.default simple

See 'git help config' and search for 'push.default' for further information.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)
fatal: 'master' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
Spencer@SPENCER-PC /d/repos/testrepo-practice (master)
$
```

Please tell me what went wrong. Thanks in advance 🙂

manthony

Spencer,

The first part is Git asking you to set a configuration value for the `push.default` value. The best way to set this for a Git beginner is:

```
git config --global push.default simple
```

Once you set that, try this again:

```
git push -u origin master
```

Mary

Anonymous

Hi,

I'm having the same problem, but it does not go away even I set push default value as you told to do. It still says:

```
$ git push -u origin master

fatal: 'origin' does not appear to be a git repository

fatal: Could not read from remote repository.


Please make sure you have the correct access rights
and the repository exists.
```

Can you advice me? Thanks!

Anonymous

Thank you Mary; these are all is so helpful!

Anonymous

Excellent tutorial, thanks for the clear explanations.

Anonymous

HI. in step two you say:

List the contents of your *local repository* – including the hidden files

maybe a stupid question. but how do i do that?

manthony

You can't learn unless you ask – it is a learner's question. To list the files, you use the **ls** command. To make sure you see hidden files, you use the -a flag. So, at the command line in GitBash you type:

ls -a

And press enter.

Anonymous

"There are a several formats".. 🙂

manthony

Dope slaps herself. Thanks for the catch!

Anonymous

This is a great tutorial. I'm a complete newbie to CVS and am so glad I found this!

I'd like to make a suggestion, though.

I got stuck on these lines:

> Open a browser and a terminal window on your local system (the system you code on). Then, do the following:
>
> 1. Bring up the terminal window on your desktop and navigate to your home (~) directory.

Yes, I'm a total newbie to CVSs, and frankly don't use the command prompt enough to have known right away that "terminal window" is referring to the Git Bash command line window that was installed earlier in the tutorial. The phrase "terminal window" wasn't mentioned earlier during installation of Git Bash (at least I don't **think** it was).

Perhaps it would be helpful if a screenshot of the Git Bash window was added near the quoted lines.This would immediately show the reader what "terminal window" is referring to. (Or add "terminal window" to the instructions for installing Git Bash).

This is really a minor point, I know, but it did trip me up for a bit until I tried entering the instruction's commands in Git Bash.

Otherwise, awesome tutorial!!

manthony

Love this comment because it shows how even very small inconsistencies in language can cause confusion. I fight not to let them into my docs or any copy. It is really hard though to explain to non-writers why it is important without showing examples of user comments like yours. So, thank you.

I've called out terminal window in the intro sentence. Then, I'll drop in a screen cap as you request. Great comment!

Anonymous

I'm very glad I could help!

Anonymous

I'm stuck at step five. Here's my directory listing and the output of the "git status" command.

```
Mark@MARKSLAPTOP-PC ~/repos/bb101repo-practice ((unknown))
$ ls -al
total 3
drwxr-xr-x 4 Mark Administ 0 May 24 06:18 .
drwxr-xr-x 1 Mark Administ 0 May 24 04:47 ..
drwxr-xr-x 1 Mark Administ 4096 May 24 04:48 .git
-rw-r--r-- 1 Mark Administ 123 May 24 06:18 README.txt
Mark@MARKSLAPTOP-PC ~/repos/bb101repo-practice ((unknown))
$ git status
fatal: Not a git repository (or any of the parent directories): .git
Mark@MARKSLAPTOP-PC ~/repos/bb101repo-practice ((unknown))
$
```

As you can see, the current directory is bb101repo-practice, so I don't understand why I'm getting that error. Does the problem have something to do with "((unknown))" after my current path?

manthony

Mark,

The problem does appear to be related to the (unknown). I had never seen this before either. So, I google for git bash unknown and found this on Stackoverflow:

http://stackoverflow.com/questions/16120078/git-bash-always-says-the-branch-is-unknown-in-all-directories

Try some of the solutions there...

Mary

Anonymous

I created a new repo using the GUI and added the readme file from there. I then opened up the Command Line Git and there was no "((unknown))" after the new repo.

I went through the commands in the SO link you provided, but nothing seemed to have any effect.

If you're Googling for answers, then I'll assume I'm on my own in finding a resolution for this. I **could** continue to use both the GUI and the Command Line Gits, but I wasn't able to figure out how to push the new repo and its containing file to Bitbucket from the GUI interface, so I'd have to try to figure that out on my own anyway.

Not an ideal situation in either case...

manthony

You aren't on your own. The first step should always be to google for answers. There is a wide and deep pool of technical knowledge on the net. Getting an answer there is often quickest. Googling hits our own Atlassian Answers database.

If you can't find a solution, you can file a ticket with our support@bitbucket.org. That is a good second step but may take longer as it puts you into a queue.

I'm sorry you are having problems. We will get you an answer.

Anonymous

Thanks Mary... Its really helpful to me

Anonymous

> **BahtiyaR@BHTYR ~/repos/repos-c++ (master)**
> **$ git push -u origin master**
> **Password for 'https://bhtyr@bitbucket.org':**
> **error: src refspec master does not match any.**
> **error: failed to push some refs to 'https://bhtyr@bitbucket.org/bhtyr/repos.git'**

the error is caused by?

manthony

It looks like the repository URL was bad.

erer

Great tutorial, love the dumb-down language 🙂 because I can understand all this stuff, for the most part at least 🙂.

Question though [Platform: Win8 Pro]: When I make a local copy of a repo, it gets copied here ***c:\Users\myuser\repos\test-repo***.

Is there a way to change that folder path to anywhere else? I have a /clients folder in which I would like to copy my repos to.

Thanks in advance.

manthony

Ricardo,

When you clone a repo, it is copied into the directory where you run the command. IOW, if you want it in a different directory, change directory there and then enter the clone command. Hope that makes sense.

Mary

erer

Hello Mary,

Yes, it makes sense. However, it's not intuitive when doing it via the Git Bash window because the commands are different from the CMD/PowerShell windows (DOS based commands). At least I wasn't able to make any sense of the commands of Git Bash explained in this tutorial, and although I'm not a hard core web programmer/developer, I'm a bit familiar with the old DOS commands and that made it SO MUCH easier.

I learned this just these past couple of days.

So yes, all I have to do is navigate to the folder I want to clone the repo to in the CMD/PowerShell prompt, run the "git clone reponame" command, then password and voila, repo cloned to the folder of my preference.

Thanks for your help.

manthony

Ricardo,

Glad you solved it and thanks for the insight into your user experience. The Git Bash window does look like a DOS window but really it uses Msysgit which is explained as:

> MSys is an environment for Windows offering a Unix-type shell and a Perl interpreter. Because many parts of Git are still not builtins programmed in C, but instead shell and Perl scripts, Git for Windows needs such an environment.

This Stackoverflow page explains what you have discovered for yourself. I'll see about explaining this a bit in the tut.

Mary

Anonymous

A comment about this: I'm pretty computer/command prompt savvy, but the ~ location tripped me up (I haven't used *nix seriously since college). A little experimenting revealed that ~ is "C:\Users\<userName>" on my Windows 7 computer. This typically isn't a folder users add files to directly. (They probably want ~/documents or some path straight off the C: drive.) A note about what ~ is in windows would be useful for newbie users.

Anonymous

This tutorial is well written. Thank you.

Anonymous

Good tutorial! Thank you.

Anonymous

Hello, very nice tutorial, it makes this all very easy. However, I have a small problem.

I don't seem to be able to fill in my password at step 1. Is this more occurent, and if so. How could I fix this?

manthony

This is not typical. Keep in mind when you type in your Password, it won't be visible to you. Can you supply any more information? What command line are you supplying? Does the command return any errors.

Anonymous

Mary, excellent work, also in replying to the comments/questions.

manthony

Hey thanks.

Anonymous

hi mary... i simply faced the same problem with anon above where i cannot write my password as im typing and it refuse to appear ? did u get it?

manthony

Hi. Your password shouldn't appear at the command line when you type. This is a security measure of most operating systems. Is it happening someplace other than the command line?

Anonymous

Check firewall

Anonymous

thank's you have saved my time gud tutorial.

Anonymous

anything without great , clear and concise documentation leads to struggle for users.

This is really well documented and awesome. Terrific job guys.

Anonymous

The tutorial explains the process of setting up a bitbucket repo beautifully.You guys at Altassian rock.

Anonymous

I have to right-click (not left click) the GIT menu bar to get the menu drop down. Windows 7 64 bit.

Roy Hornsby

Hi Mary

thanks for your great tutorial and all of your follow up comments

I was following this tut using the terminal on my Mac and when it came to submitting the password I got this: "error: git-credential-osxkeychain died of signal 11". After checking around on the forums I see this is a common error, but I've yet to find a solution that works for me. I've been playing around with SourceTree also and I get the same error there.

Do you have any suggestions how to get around this?

cheers

manthony

Roy,

To enable HTTPS Credential Caching a.k.a 'password credential caching' for https based interaction with Bitbucket, you need to download and install the "osxkeychain credential helper" and tell git to use it. We have that documentation under Stash right now and you can follow the same instructions. I'll have to update our Bitbucket docs within the next day or two.

Mary

Roy Hornsby

Hi Mary, thanks for your prompt reply.

I'm afraid that, as I'm not an experienced terminal user, the document in Stash doesn't make too much sense to me. I'll have a search around and see if I can find some step by step instructions.

cheers - roy

Anonymous

I get an error as below, when trying to clone. Wasn't able to do on IntelliJ, so tried with Cygwin with no luck:

$ git clone https://myuser@bitbucket.org/myuser/test.git
Cloning into 'test'...
error: error setting certificate verify locations:
CAfile: /usr/ssl/certs/ca-bundle.crt
CApath: none while accessing https://myuser@bitbucket.org/myuser/test.git/info/refs
fatal: HTTP request failed

manthony

This tutorial assumes you are using GitBash. I don't test against other clients. That said, I did a search for this error and found this on Stackoverflow.

Anonymous

I have substituted my user name with myuser in above comment. 🙂

Ashish Garg

I am getting an error while trying to clone from bitbucket repository via https:

Using following command:

git clone https://myuser@bitbucket.org/myrepo/myproject.git
Cloning into 'myproject'...
error: Connection time-out while accessing https://myuser@bitbucket.org/myrepo/
myproject.git/info/refs?service=git-upload-pack
fatal: HTTP request failed

Please let me know what am i doing wrong?

Thanks,

Ashish Garg

manthony

Ashish, you should send an email to support@bitbucket.org for this one.

Anonymous

how can I take a print out of this?

File>Print Preview doesn't give the printable version. Is there any printable version of this document?

Thanks

manthony

I'm attaching the PDF I get with the **File > Print** dialog. You don't say what browser you are using. I'm using Chrome which has a **Save as PDF** option. Did you want the entire documentation set or just this page?Clone Your Git Repo and Add Source Files - Bitbucket - Atlassian Documentation.pdf

Anonymous

In create README step, creating a file and saving it in repository is not mentioned properly. As I am a fresher, I found some difficulty over there. Otherwise it is really superb.

manthony

Hi, thanks for the feedback. Can you be more specific about what you had trouble with?

Anonymous

In create README part, in step no 3, you have mentioned save the README file . But you haven't mentioned how to save i.e., command to save. There I found some difficulty.

manthony

How you save would depend on what kind of client software you were using to create the file. Since, there are very many kinds of file editors, I can't be more specific; I assume you could read the documentation for your specific file editing software to learn how to **Save**.

Anonymous

Atlassian Team,

This is really going great - hats off to you for writing such a clear step by step user guide. I am a new comer - and only because of crystal clear guidance here, I am able to use your platform effectively.

I will be selfish if I don't mention this wonderful Tutorial by Lars Vogel which really cleared the concepts about DVCS in astonishingly simple language - thanks to him as well.

I have few suggestions - hope you will consider.

1. You (now) also have a wonderful product SourceTree with excellent GUI.Would you consider updating this user guide to that effect ? Many users would welcome this addition.
2. I dont know if this is the right place to suggest something for SourceTree but since that product too belongs to your company, I am taking liberty to mention it here. Tortoise SVN has very useful shell extensions which really help to use their platform through windows explorer. Would you consider that for SourceTree as well ?

Thanks once again for a wonderful product, platform and best of all, Documentation.

Best Regards,

AAD

manthony

Glad to hear you like our tutorial and our product. Lars' tutorial on DVCS is very well done. We mention it to our users who are unfamiliar with DVCS. Regarding your suggestions:

1. Currently, we don't have plans to change the tutorial to use Sourcetree exclusively. Mostly because the rest of the documentation assumes you are using a command-line. I'll bring it up with the team.
2. I'm not sure if we have plans to support SVN with Sourctree. I'll ask and reply here.

Thanks again for taking the time to comment. We really appreciate our active community.

Anonymous

Thanks for your reply.

>>Currently, we don't have plans to change the tutorial to use Sourcetree exclusively. Mostly because the rest of the documentation assumes you are using a command-line. I'll bring it up with the team.

I understand - this is a huge task. But at least you can mention somewhere in 'BitBucket 101' that such a GUI tool is available. I started searching for GUI tool after seeing the terminal screen in your first tutorial 🙂

>>I'm not sure if we have plans to support SVN with Sourctree. I'll ask and reply here.

Did I fail to convey my point ? I did not expect SourceTree to support SVN. But I wished if SourceTree can have Shell Extensions SIMILAR TO SVN - so that most of the job can be done even without opening SourceTree. Of course I am not sure if SourceTree Developers would like this idea AFTER developing a nice GUI.

Best Regards,

AAD

Anonymous

>>I understand - this is a huge task. But at least you can mention somewhere in 'BitBucket 101' that such a GUI tool is available. I started searching for GUI tool after seeing the terminal screen in your first tutorial 🙂

Please ignore. I can see that it already existed in your 101. Sorry for inconvenience.

Regards,

AAD

Anonymous

Clear, concise tutorial.

Anonymous

Awesome tutorial, straight-forward! Thanks!

Anonymous

I have created local directory repos and when I tried to see in my file system...it is showing me the repos is created in h drive (network drive). When I do pwd in GIt bash it is showing me as /h

Is this expected behavior. Is my local repository in the network drive not in the c drive?

manthony

You can put GitBash or your repos in any drive accessible to you. The C is the standard drive for most setups so that is what we show in our examples.

Anonymous

I did not do any changes in my installation or setup. I expected GitBash will create repos directory in my c drive...but it created in network drive. I never accessed this h drive(network drive) earlier in my daily work.

manthony

It is possible for your Windows administrator to specify a default directory for installing new Windows programs. That might be the situation in your case. If you are concerned, you can talk to your system administrator about this.

Anonymous

I get an error as below, when trying to clone. why and how to solve

Administrator@20130814-2319 /c/repos $ git clone git@bitbucket.org:Edward_Wu/bb101repo.git

Cloning into 'bb101repo'...

Permission denied (publickey).

fatal: Could not read from remote repository.

Please make sure you have the correct access rights and the repository exists.

Thanks,

manthony

Your clone should use an http URL at this point. It would look like this:

```
$ git clone https://newuserme@bitbucket.org/newuserme/bb101repo.git bb101repo-practice
```

If you want to skip ahead and use SSH, you can but make sure you have set up your SSH key and imported it into Bitbucket.

Anonymous

I had the same problem. I verified the address as Edward_Wu suggested, but still did not have any luck.

It turned out that I had created a public key back on my first attempt to go through the tutorials. I went in and deleted that key since I no longer use that computer. I went back to the "Clone" button and re-copied the URL. It worked that time.

Ryan Clamp

I've been having many problems with this unfortunately, on my home computer it works fine however at work I'm having problems.

When trying to clone I get the following:

fatal: unable to access 'https://reponame@bitbucket.org/reponame/lwqa.gi
t/': Failed connect to bitbucket.org:443; No error

I'm not behind a proxy/blocked firewall and can access HTTPS sites normally.

manthony

The call was attempting to connect through port 443 but failed. It could be the port is blocked for some reason. Did you talk to your system administrator? You can also try SSH access to see if that helps. If you try both of those solutions and still have problems, file a support ticket.

Anonymous

Just wanted to post that these directions are phenomenal. I'm a sysadmin who'd learning a bit of coding/version management/etc. This is incredibly helpful and I'm looking forward to utilizing the heck out of git/bitbucket. 🙂

manthony

Hey thanks! You made my day.

Anonymous

Hi,

Thank you for posting this nice tutorial. When I tried to push the commits I got the below error:

```
$ git push -u master
warning: push.default is unset; its implicit value is changing in Git 2.0 from 'matching' to 'upstream'. To squelch this message and maintain the current behavior after the default changes, use:

git config --global push.default matching

To squelch this message and adopt the new behavior now, use:

git config --global push.default upstream

See 'git help config' and search for 'push.default' for further information.

fatal: 'master' does not appear to be a git repository
fatal: The remote end hung up unexpectedly
```

so I issued the below command

```
git config --global push.default upstream
```

but now I am receving the following message

```
$ git push -u master
fatal: 'master' does not appear to be a git repository
fatal: The remote end hung up unexpectedly
```

What did I miss? 🙂

Anonymous

gosh! its embarrassing

the correct command is

```
$ git push -u origin master
```

just noticed after posting the above comment, sorry!

Anonymous

Hi, Is there a way I can write a dos script to pull my "Bit bucket Branch" into my local machine. I need to automate a script which will pull on daily basis before I start writing in my branch.

manthony

Yes, you can write a batch file that does this for you. Then, use the Windows scheduler to run the batch file regularly. Stack Overflow has a lot of queries about this:

http://stackoverflow.com/questions/5401229/how-do-i-execute-several-git-commands-in-a-batch-file-without-terminating-after

Anonymous

You need to explain what you are doing when you "clone" your repository to the local system and how this relates to checking out a local copy of the repositlyr (for those familiar with other version control software). Is it the same thing?

manthony

Cloning is not the same as a "check out" in other version control software. With Git you are always cloning the entire repository. With other version control systems, you can checkout just a single directory.

This documentation does not compare and contrast version control systems. Lots of folks have already done this. Often, very heated and "religious" discussions spring up. Since the goal of this documentation is to teach you about Bitbucket and many of our readers already know DVCS, I don't do a lot of compare and contrast here.

Anonymous

I do not understand where I am supposed to save the readme file in the tutorial!

manthony

You should save it to the root folder of your local tutorial directory.

Anonymous

Hi, everybody. I've some problem like this:

error: SSL certificate problem, verify that the CA cert is OK. Details:
error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed while accessing
https://nurdus@bitbucket.org/nurdusteam_eties/lux.git/info/refs

fatal: HTTP request failed

What do you think about it?

Best regard!

manthony

You don't mention which client you are using. You need to set a flag with Git here:

https://bitbucket.org/site/master/issue/5292/ssl3_get_server_certificate

http://stackoverflow.com/questions/3777075/ssl-certificate-rejected-trying-to-access-github-over-https-behind-firewall/8467406#8467406

Anonymous

Doesn't let me type on the password bit :/

manthony

Really? What step were you on?

Anonymous

manthony, thanks. I'll try to do.

Fork a Repo, Compare Code, and Create a Pull Request

So far, you've worked in a repository you own and you've been the only person working in it. You can only fully experience the power of a DVCS hosted repository by working with others. In this page, you work with a repository you don't own. You'll make a change to a code file, not just a README. You'll use Bitbucket's comparison features to compare your repository with the original.

The tutorial examples so far worked exclusively with Git repositories using `git` commands. In this section, you'll work with a Mercurial repository. If you start working extensively in the host DVCS community, you'll most likely find yourself working with multiple DVC systems. You may even find yourself working with other hosted products, such as GitHub or Kiln. It is good experience to see the workflow you'd use in these types of situations.

Linux or Mac User?

The examples on this page are written for Mercurial using TortoiseHg on **Microsoft Windows 7**. If you want to work in Mac OSX or Linux, see the instructions on this page.

Backgrounder

When you work with another user's public Bitbucket repo, typically you have read access to the code but not write access. This means that you can use a `clone` command to copy the repo but you cannot `push` changes to it. Instead, you use the Bitbucket interface to *fork* a repo. Bitbucket hosts private or public repos. Anyone can fork a public repo. You can fork a private repo where you have permissions.

Forking a repository creates a new repository under your account. This new repository is a copy of the original repo and is called a *fork*. Even though you have a fork, you can still get a change into the original repository, to do this you:

- clone the forked repo from Bitbucket to your local system
- make changes to the local repo
- push the changes to your fork on Bitbucket
- issue a *pull request* against the hosted repo you forked from
- wait for the repo owner to accept or reject your pull request

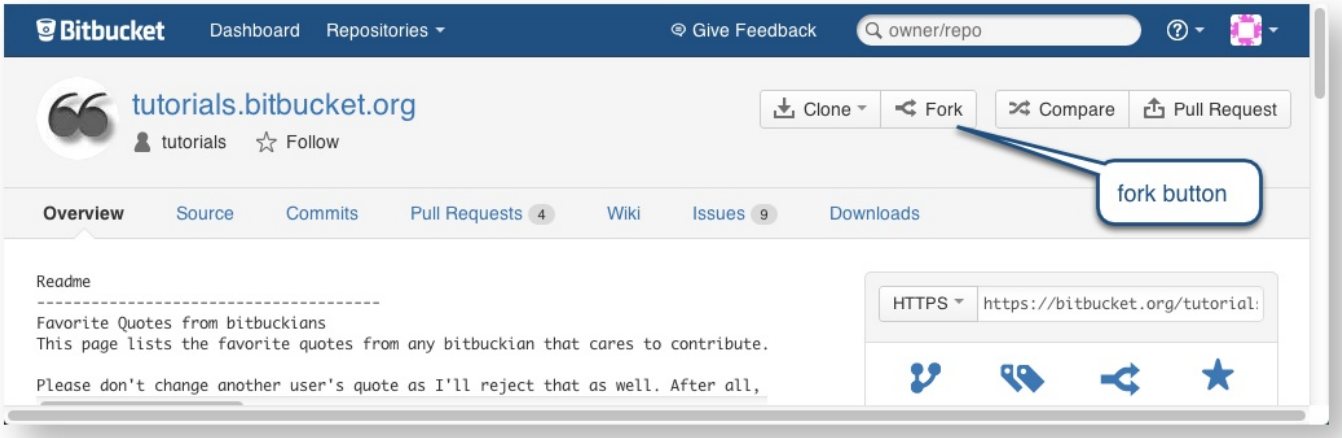
If a repo owner accepts a pull request, Bitbucket pulls your code changes into the original repo and merges them. Bitbucket recommends you work with forks and pull requests even if the repo owner gives you write access to a public repository. While a pull is a DVCS concept, "pull requests" and forks are concepts used only by repository hosting services — like Bitbucket and GitHub. You won't find fork

and pull requests in any Git or Mercurial workflow that doesn't involve a repository hosting service.

Step 1. Fork another user's repo

In this example you'll fork a public repository belonging to a user called `tutorials`.

1. Log into Bitbucket.
2. Locate the **Search** field in the menu bar.
3. Search for the `tutorials/tutorials.bitbucket.org` repository.
4. Click on the search result to go to the repository.
5. Choose the **Fork** button.



   The system displays the fork page.
6. If your account is a member of a Bitbucket team, the page contains a field **Owner**. If you don't belong to a team, there is no Owner field; Continue to the next step.
7. Change the **Name** for example, to `myquotefork`.
8. Enter a **Description** that you think is appropriate.
9. Uncheck **Inherit repository user/group permissions**.

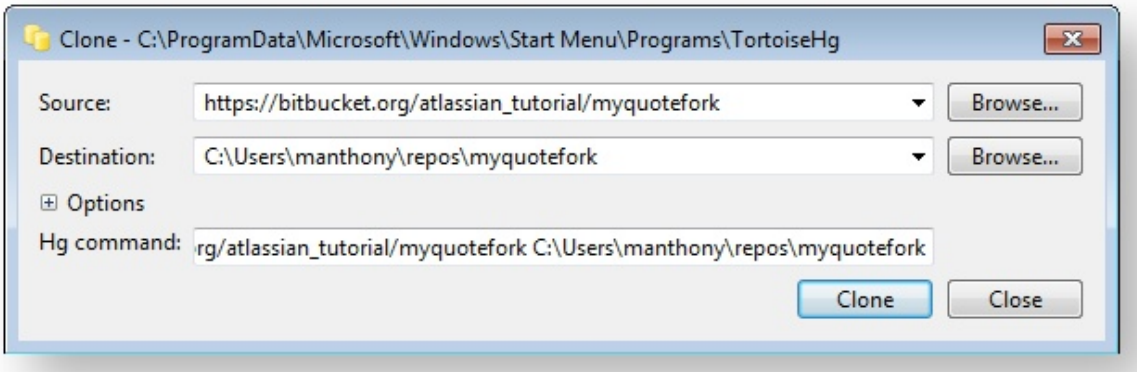
   There are several options for forking. For now, just leave all the remaining options at their defaults.
10. Press **Fork repository**.

Step 2. Clone your fork

1. Start the TortoiseHG Workbench.
2. Choose **View > Show Repository Registry**.
3. Choose **File > Clone Repository**.
4. Enter location of your forked repository in the **Source** field.
5. Enter a destination on your system for your local repository, for example:

   `C:\Users\`*`yourusername`*`\repos\myquotefork`

   When you are done the dialog will look similar to the following:



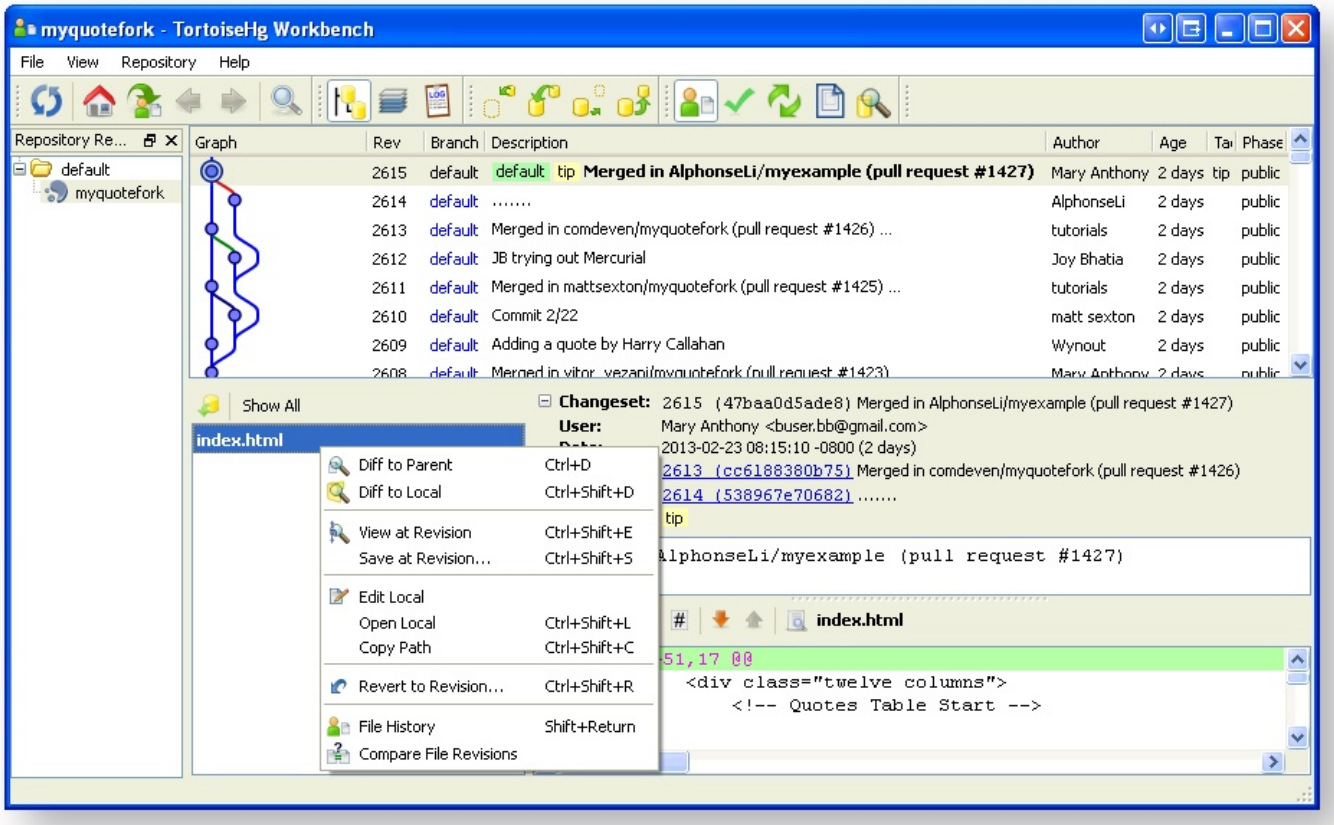
6. Press **Clone**.

   The system clones the repository and displays it in the registry list.

Step 3. Make a change to the repository source

This repository contains a website which, as of this writing, has a single `index.html` file. Bitbucket allows you to host a website in a Mercurial repository. To see the hosted website, go to https://tutorials.bitbucket.org – you may encounter an Untrusted Connection message. Go ahead through to the site. You'll see that the site contains a single page that lists favorite quotes from "bitbuckians" which is just a writer-invented word for users of Bitbucket. Now, it is your turn to record your favorite comedic quote...or just a favorite quote. Do the following to contribute to this repository:

1. Use Google or some other search engine to locate your favorite quote.
2. In the TortoiseHG Workbench, select the `index.html` file.
3. Right-click to display the context menu.



4. Choose **Edit Local**.

   The file is an HTML file.

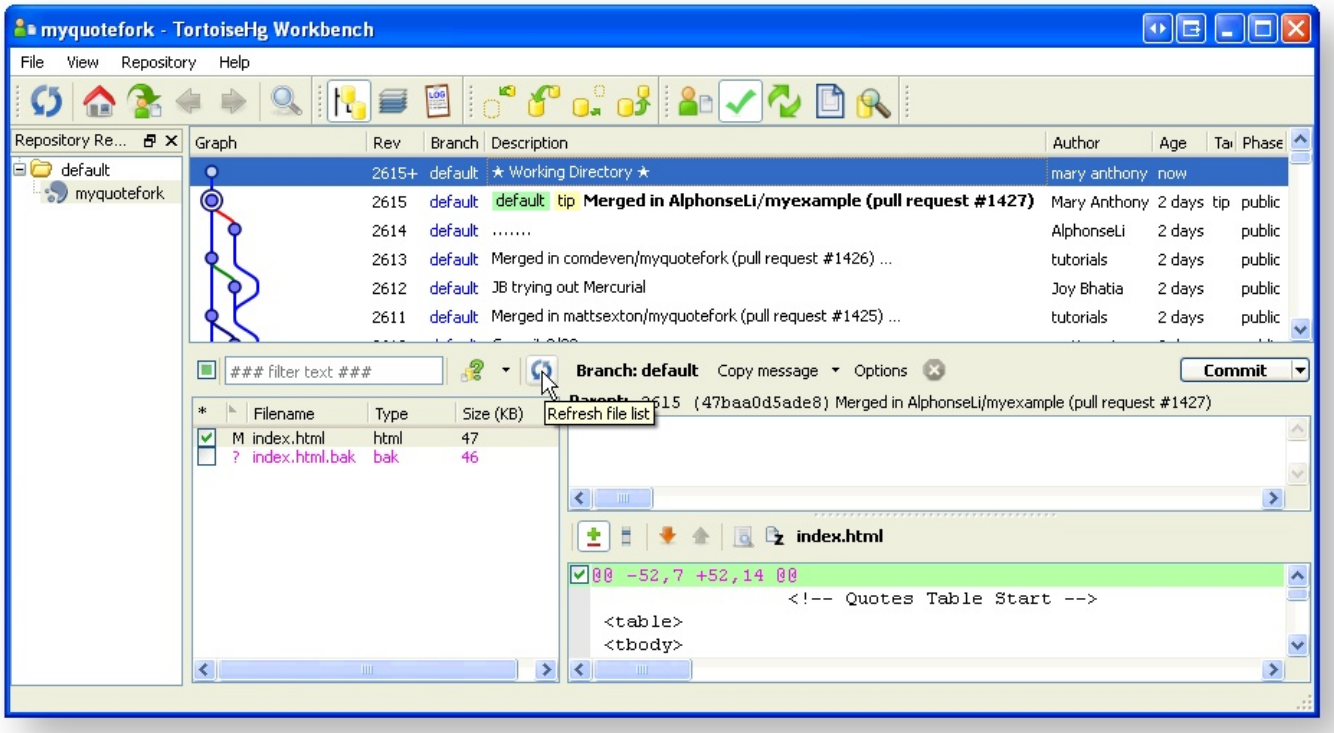
5. Go ahead and include an photo or image using an `<img>` tag and the quote right above the `</table>` tag in the file.

   Here is a sample of what an addition will look like:

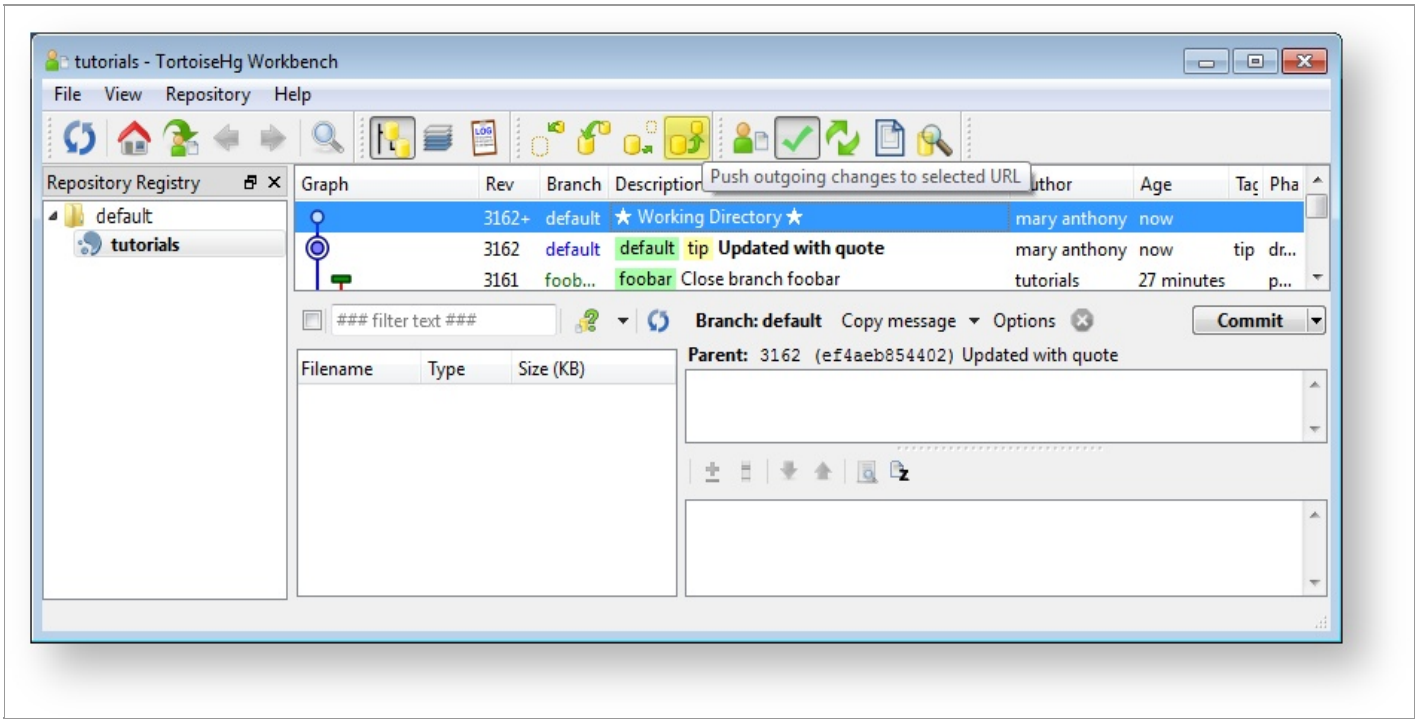
```
<tr>
<td>
    <img src="http://www.sample.com/image.gif"/>
</td>
<td>
    <blockquote>QUOTE_YOU_WANT_TO_ADD</blockquote>
</td>
</tr>
```

   If you are not sure what to do, you can copy and modify another user's entry. You can actually do whatever you like here (ok, within reason). For example, one user cleaned up our HTML — thanks Jon! And another added some nifty CSS – thanks Andrei!

6. Close and save the file.
7. Refresh the **Working Directory**.



8. Enter a commit message in the space provided.
9. Press the **Commit** icon (check symbol) to commit your changes.
10. Press the **Push outgoing changes to selected URL** icon.



The system prompts you to confirm your action.

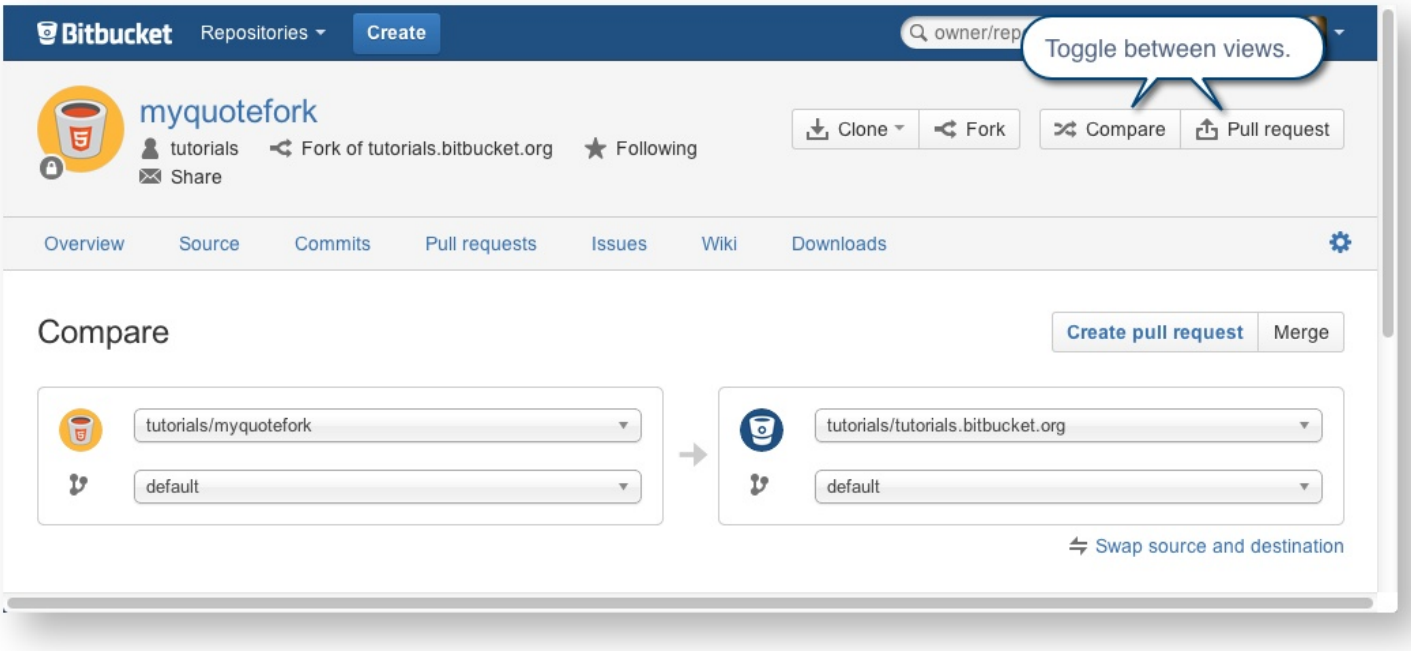
11. Press **Yes**.

The system pushes your changes to the forked repository.

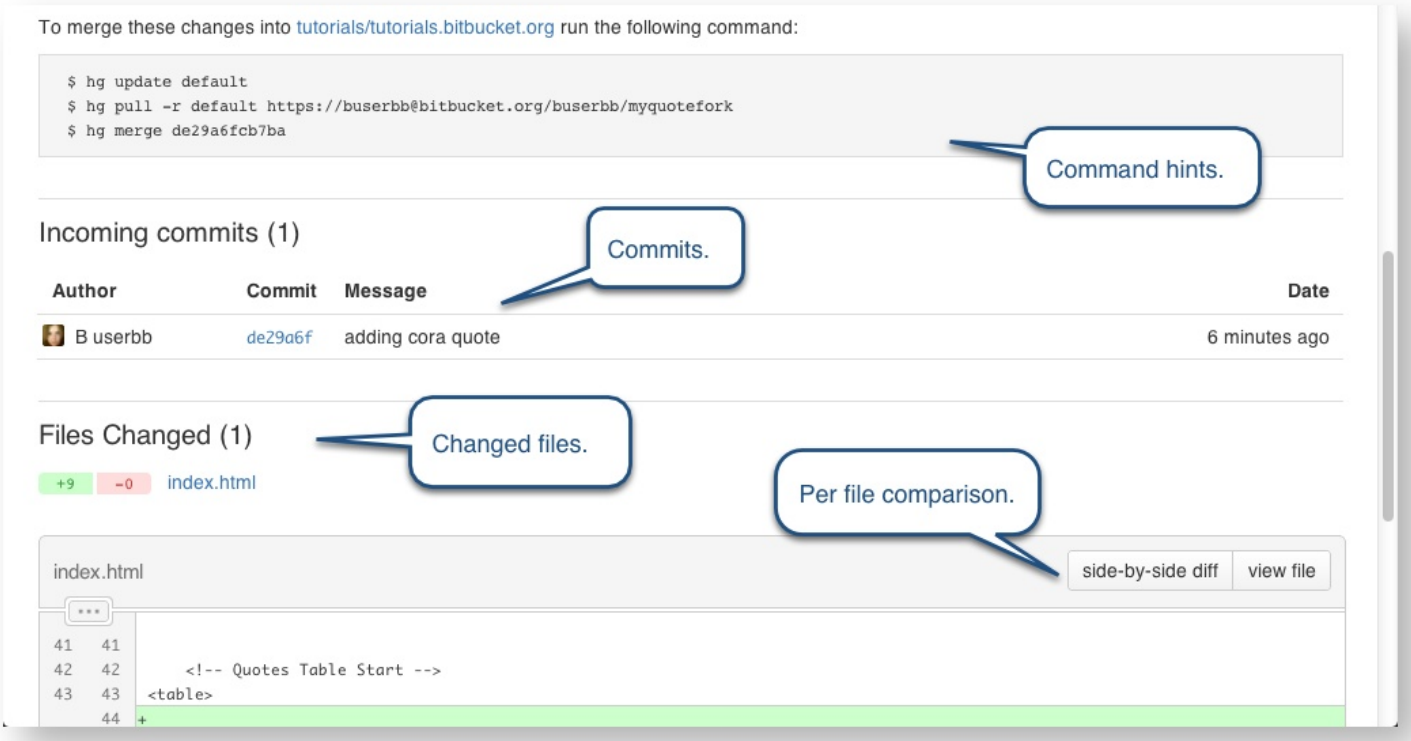
Step 4 . Compare your fork to the original

During the time you were working on your fork, the original repository you forked from may have changed. At this point, you can check that and decide if you need to adjust your fork accordingly. Log into Bitbucket and navigate to your `myquotefork` repository. Forked repositories have a special widget that lets you compare your fork work to the original or to create a pull request. The **Compare** and **Pull Request** buttons toggle between two specialized views available only in forked repositories.

Press **Compare** to reveal the **Compare** view which contains a lot of information. This view allows you to look at your forked repository in comparison to the original repository. You can look at it from two "directions." You can compare your repository against **Incoming** changes from the original. Or, you can compare **Outgoing** changes from your fork to the original.



Keep in mind that the outgoing changes can be from multiple people. How? Well if you have shared your fork repository with others (You'll learn more about how to do this later). Right now, though, you should be the only person on the repository so only your changes appear. Scroll down the view and you will find it includes helpful Mercurial commands:



You can merge your fork into another repository — for example a local copy you may have of the original repository. If you merge locally, you can test your changes before making a pull request through Bitbucket.

The commits section displays the commits pushed from a local repository to the fork in Bitbucket. If there are multiple commits, you see their cumulative changes by file in the **Files Changed** section. You can toggle between the **side-by-side diff** button to see the changes displayed in that format. Or press the **view file** button and see the file.

To see the contents of a specific commit in isolation, select a **Commit** link and the system takes you to the **Commits** page.

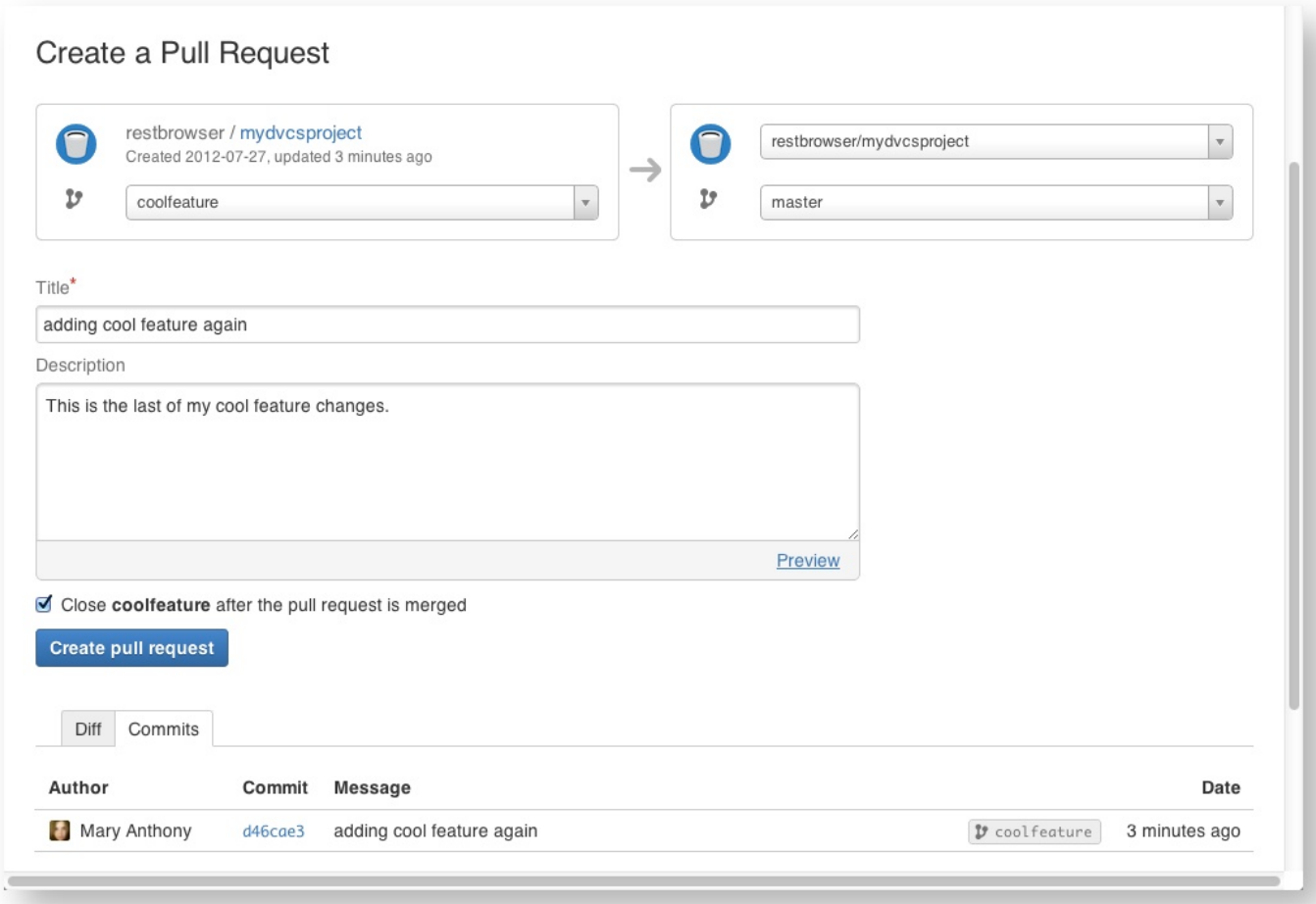
Step 5. Create a pull request

If you haven't already done so, log into Bitbucket and navigate to your `myquotefork` repository. Then, do the following:

1. Press **Create Pull Request**.

   The system displays the request form.
2. Complete the form.

   When you are done it will look something like this:



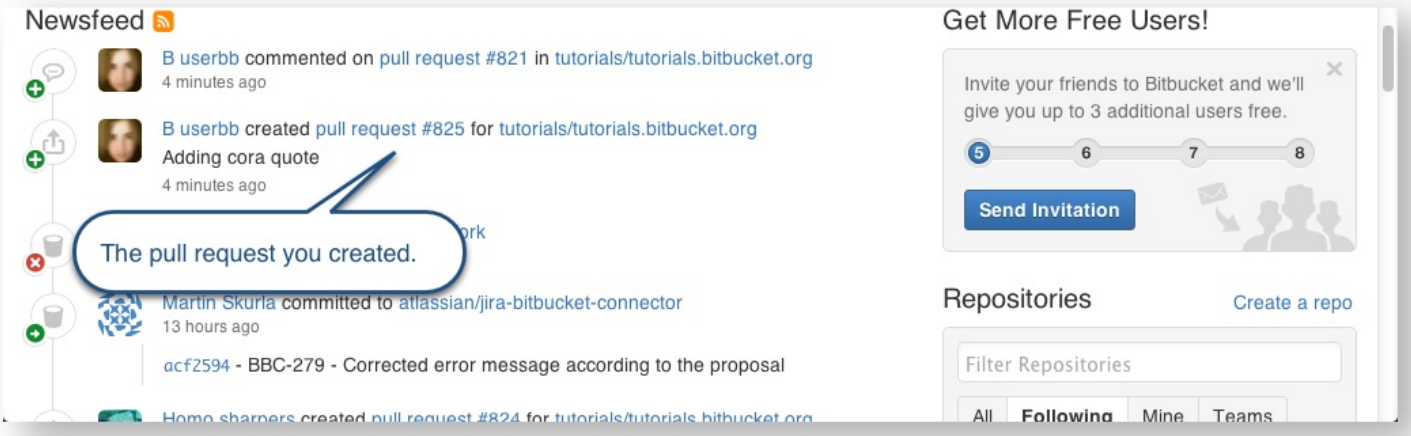
3. Press **Create pull request**.

   The system opens your latest request on the **Pull Request** page *of the original repository*. To see the list of all the pull requests against this repo, click the **Pull Request** tab.

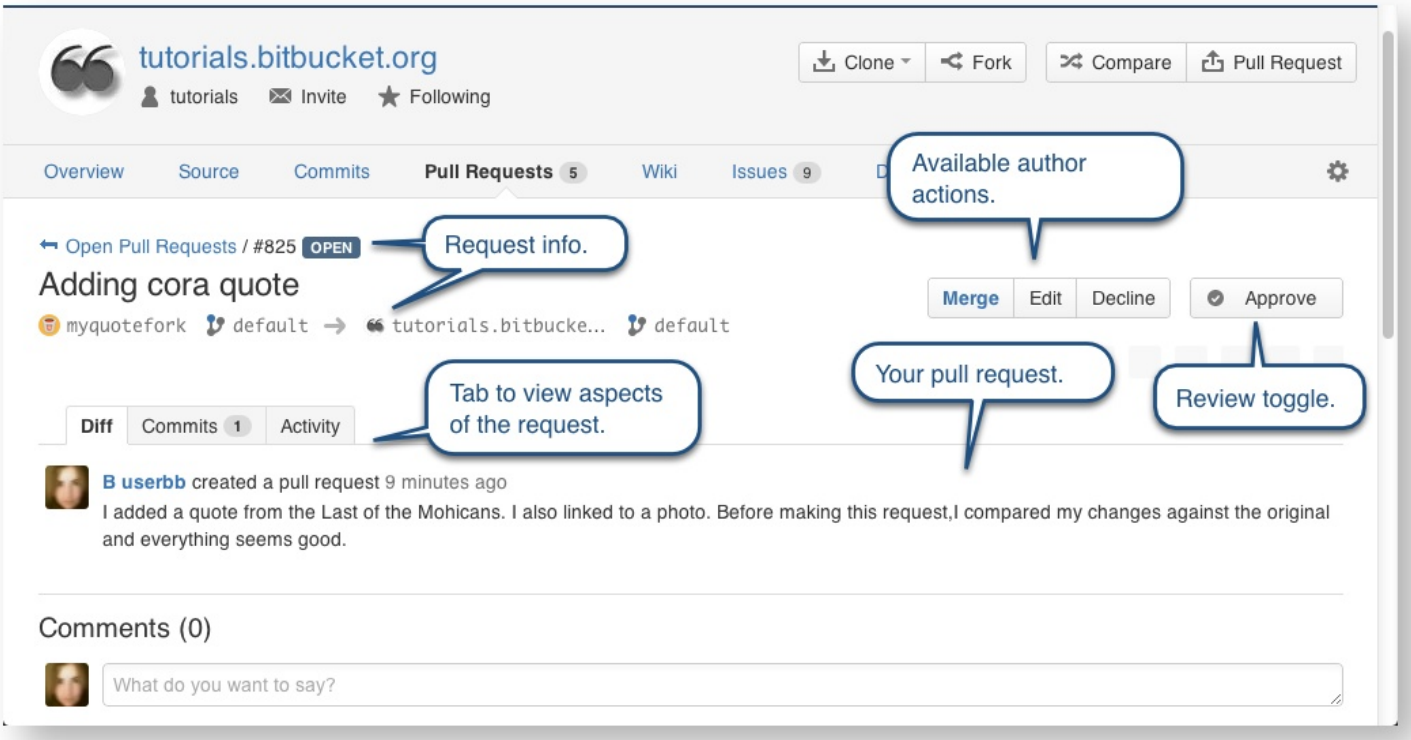
Step 6. Learn what happens to your pull request

After you create a pull request, you can't delete it. Neither can the person that receives your request. The receiver either merges or declines your request. If you delete your fork after you make a request, the receiver can only decline your request because the repository to pull from is gone.

Accepting the pull request is not something you do, it is something done by the repository owner. It is the next step though. When the original repository owner logs into their account, their **Newsfeed** shows your pull request. It also shows your fork from a few days earlier. For example:



The repo owner can click on the request and see your full request. An owner can **Merge**, **Edit**, or **Decline** a request. Owners, and others with access to the repository can **Approve** a request. Clicking **Approve** simply means the approver reviewed the changes in the pull request.



You'll get an email when your pull request is accepted or rejected. For now, continue to the next step.

Next

That was intense. Maybe. Depends on your daily life. In the next section, you learn ways to be social on Bitbucket which includes adding users to your repository.

Like 4 people like this

62 Comments

Sanford Staab

Quite helpful

Muhammad Moosa

Nice tutorial. Here is the point that should be considered timely. Due to a lot of forks and pull requests the tutorial repo size is growing. At this time it is 16.? MB it is due to images. If image links are put rather saving images in repo "image folder" it will be better for future. Tutorial repo should be lesser in size and efficient.

manthony

Thank you for the comments. You are right the quotes repo is, as one user put it, getting ridiculously long. Links versus images is a trade off. Often people link to very slow loading sites which slow the load down considerably. So, that's why images and not links are preferred. I'll see about sweeping through and resizing the images — some of them are very large.

Muhammad Moosa

Backgrounder is very essential and good concept.

Anonymous

Great write-up, and I love the hands-on tutorial concept. The annotated screenshots are also very helpful.

manthony

Super! Thanks for the feedback on the screenshots!

Anonymous

In step 8.7-8.10 it took me a while to find the "**Commit**" and "**Push outgoing changes to selected URL** " icons. A couple of your comments would be a good addition.

Anonymous

I mean adding comment bubbles to the image above.

manthony

Good point. I'll update the Tortoise HG screenshots with this on the next go round.

Anonymous

On the contrary to the previous chapters, this one has really caused problems for me:

- the TortoiseHq Workbench is a different (newer) version, and the things about commit are quite different
- I accidently gave the false user name (The instructions above DID NOT TELL WHICH USER NAME, and I gave TortoiseHq user name, not the bitbucket user name, as it should have been), and it went through, though it did not update the bitbucket repository. And I FIND NO WAY TO CORRECT THAT USER NAME. All change seem to be commited OK, but no changes go to the bitbucket. This one wrong thing mixed everything up, and I cannot correct it !!!

manthony

I'm sorry you had problems with this chapter. You are right, I haven't picked up the latest TortoiseHg version and I need to do a QA against it. I'll bump it up in my todo list.

Anonymous

Any reason why the tutorial clones to tortoisehg just fine but when I go to clone my own I am getting a HTTP Error: 404 (Not Found)?

Marcus Bertrand [Atlassian]

Perhaps you didn't put the .git at the end of the URL (assuming Git). But more than likely the URL may be a little off. You can get an exact copy from the header of the repository after the "git clone" part.

Anonymous

For various reasons I didn't want to use or install mercurial - it would have helped me to have a GIT version of this section too... Though

mercurial sounds so similar I'm guessing I could do the same in GIT following along the mercurial version.

Anonymous

Hi,I was following the tutorial.previous chapter went well for me.But in this chapter while trying to clone(clone your fork) other user repository(which is git repository,it is private and write access shared with me,having one readme file inside it) I am getting the error in TortoiseHg workbench.

HTTP error:404 not found.The url i am trying having the extension .git.I have tried without .git extension also but it is failing.Please help me.

Anonymous

I'm new here, another fan of this awesome tutorial, and just had and fixed this same problem today when trying to clone the repository I'm actually working on. Try GitGUI (or maybe TortoiseGit) instead. The 404 error seemed to be because of trying to clone a Git repo with mercurial tools? Not sure exactly, but try that? Hope it helps. --molly

manthony

Molly thanks for doing a better read of this user's problem. You absolutely hit on the problem. I deleted my response, I'll leave yours behind!

Anonymous

yes,using GitGui i am able to clone the Git repo.Ok this means only mercurial repo we can clone using TortoiseHg.For Git repo cloning we need to use GitGUI.

Thanks a lot.

I have another query that is, now i have forked and cloned the other user repo.Now if he does changes in 2-3 files and updates in his repo.How I am going to update only those files in my cloned repo.do I need to clone the whole repository again in my system.

Belinda Chisholm

How do you delete a fork pls?

manthony

Hi Belinda,

A fork is repository like any other. From Bitbucket, you Deleting a Repository. From your local system, just delete it as you would any other repository.

Mary

Belinda Chisholm

thanks

eden lane

Greetings .
I'm using OAuth.js library and trying to get issues from private repo. When I'm apply more then one filter with the same name, server respons 401 UNAUTHORIZED (more information could be found here : http://stackoverflow.com/questions/15851542/).
Is this bitbucket or OAuth.js bug ?
UPD: Sorry for wrong theme

manthony

Eden,

In case you haven't read it, see OAuth on Bitbucket. The 401 error means your client's credentials were not accepted. Try simplifying your code to just do a simple query. If you still have problems, please send a request to support@bitbucket.org.

Mary

eden lane

Every query with parameters having different names is working fine, so I wrote a request to support.

Thank you for your answer, Mary.

Anonymous

Quite helpfull, this whole 101 is really helping me understand this better.

A little to criticize though, I had some troubles in the step 3 of the tutorial around the step 10, it's really confusing and found out that the option is a bit hidden, a screenshot of how to pull the option menu for "Push outgoing changes to selected URL" will be more helpful.

Other than that, really good job and very helpful.

manthony

Thank you for the comment and the compliment. I've accepted both. 😀

Eric Pittelkau

I'm following along about using bitbucket and fork, and then in step 2 of this page (https://confluence.atlassian.com/display/BITBUCKET/Fork+a+Repo%2C+Compare+Code%2C+and+Create+a+Pull+Request), it starts talking about TortoiseHG. How do I start that program, and what relation does it have to bitbucket?

manthony

Eric,

If you are following the tutorial from start to finish, you should have already installed TortoiseHG. You would start it from the Windows Start menu once it is installed.

Mary

Anonymous

Typo: *You can toggle **bewteen** the* (...) instead of *You can toggle **between** the* (...)

Great tutorials, thanks.

Bitbucket user bartlomiejb

manthony

Thank you and thanks for the catch. This is fixed. :D

Anonymous

fork, clone to local directory, change accordingly to what I want and then create a pull request? the pull request syncs the local changes with the forked bitbucked repository?

manthony

The pull request is a ***request*** to sync the changes. It is up to the repo owner of the destination to accept or deny the request.

Adam Dudu

What's the differences between a new "branch" of one repository and a new "fork" of it ? Thanks.

manthony

A branch is within the repository; a repo can have many branches. A fork is a clone of the entire repository and includes all the branches.

Ebrahim Pasbani

Hi

Is it possible to copy or move some issues from repository to forked one?

Thanks

manthony

We have an issue importer/exporter but it is all or nothing. You can't select a subset. Is it possible what you really want is to Change or transfer repository ownership or Split a repository in two?

Ebrahim Pasbani

I think those links are not suitable for my case.

I fork a repository to resolve some issues.

Where these issues should be?

manthony

Well, if you want to fix an issue by creating a fork and then making a pull request back to the source, you should create the issue on the source repository. Any commits you make on the fork should include that issue tag. When your request is pulled, it will link automatically back to the issue.

Ebrahim Pasbani

Thanks, this is good

Anonymous

Step 1.2 says "Search for and then click through to the `tutorials/tutorials.bitbucket.org` repository."

I do not see anywhere in my Bitbucket account online where to find such repository.

Could you provide some more detailed instructions on where to find it?

Thank you.

Anonymous

I resolved this myself. I entered the path (tutorials/tutorials.bitbucket.org) in the search box in the upper right of the page. Maybe I jumped the gun on saying I couldn't find it, but maybe it could be mentioned to enter the path in the search box, too!

manthony

Thank you for the comment. I'll call out the search more explicitly.

Anonymous

Step 2.3 says, "Choose **File > Clone New Repository**."

The file menu lists "New Repository", and "Clone Repository", but there is no "Clone **New** Repository". I continued fine by selecting "Clone Repository".

Tortoise Hg version info: TortoiseHg version 2.8 with Mercurial-2.6, Python-2.7.3, PyQt-4.9.6, Qt-4.8.4.

manthony

Thank you for the catch. I've updated the instructions.

Anonymous

from Step 2

Step 2. Clone your fork

1. Start the TortoiseHG Workbench.
2. Choose **View > Show Repository Registry**.
3. Choose **File > Clone Repository**.
4. ...

What are the steps for doing this with GIT? Someone named Molly said she found a fix using GIT Gui, however, in GIT GUI, I don't see steps 2 and so on. Mary Anthony said Molly answered the question and subsequently deleted her solution, but Molly's doesn't help me. Just telling me to do something using Git Gui when the language isn't the same in both GUIs is presumptuous. What are the steps of doing this in Git Gui? Is it too much trouble to list the details?

manthony

On the Git side, the tutorial uses Git Bash which is a command line not a GUI. In GitBash, the steps for cloning a forked repository are the same for cloning any repository. You can see those steps illustrated Clone Your Git Repo and Add Source Files. Some vendors offer a GUI interface similar to TortoiseHG. Atlassian, for example, offers Sourcetree. I don't document that here because GitBash is a standard for many shops.

(BTW, I didn't delete Molly's response to the question above, I deleted my own.)

Efrén

What if you have more than one person with each its own fork, and creating branches?

For example:

- **Main** has only a *default* branch and is forked to **main_a** by user "a", and also forked to **main_b** by user "b"
- User "a" then creates a branch *branch_a* in **main_a**, and then pull requests to make the new branch into **Main**
- User "b" can only sync the **main_b** fork with changes from *default*.

If user "b" sees the new branch in **Main** and wants it in **main_b** there is no way to pull request it in that direction

Anonymous

Ok, I should setup SSH for Mercurial first: Set up SSH for Mercurial

You may probably put this SSH setup tutorial before this one?

manthony

The tutorial teaches first with the HTTP protocol and then with SSH. HTTP is simpler for beginners at first. SSH is more complicated and error prone due to the number of systems and steps involved.

Jeffrey Chui

Thank you for making the tutorial!
Very helpful for beginners like me. 🙂

One silly problem I had was figuring out the commit button.
Since there are two commit buttons; one at the top and one at the middle, I couldn't push any changes until I found the second commit button in the middle and realised the one at the menu bar did not actually commit.

Perhaps it would be better to rename the commit button "Commit your changes" and maybe make it red (or a colour that's distinct and easy to see)?

manthony

Jeffrey, which GUI were you looking at? Tortoise Hg or Bitbucket?

Jeffrey Chui

Tortoise Hg (as per tutorial).

Anonymous

An initial Git Bash commit and push was successful with the original repo.

But here I get stuck as step 2 with Tortoise HG cloning where:

File > Settings > Commit > username = name <email@address.com>

and

File > Clone Repository…:

Source = https://bitbucket.org/name/repo

Destination = C:\Users\Name\repos\repo

Hg command: hg clone --verbose -- https://bitbucket.org/name/repo C:\Users\Name\repos\repo

Then clicking clone results:

HTTP Error: 404 (Not Found)
[command returned code 255 Tue Aug 20 15:55:36 2013]

Is there something extra that needs to be done for setting Tortoise up ?

manthony

Check what you are entering agains the screen capture in Step 4. The 404 error says that the **Source** value is wrong.

Anonymous

Yes, I really couldn't see anything wrong with the source value, especially since I copied it from the browser address bar of the repo home page.

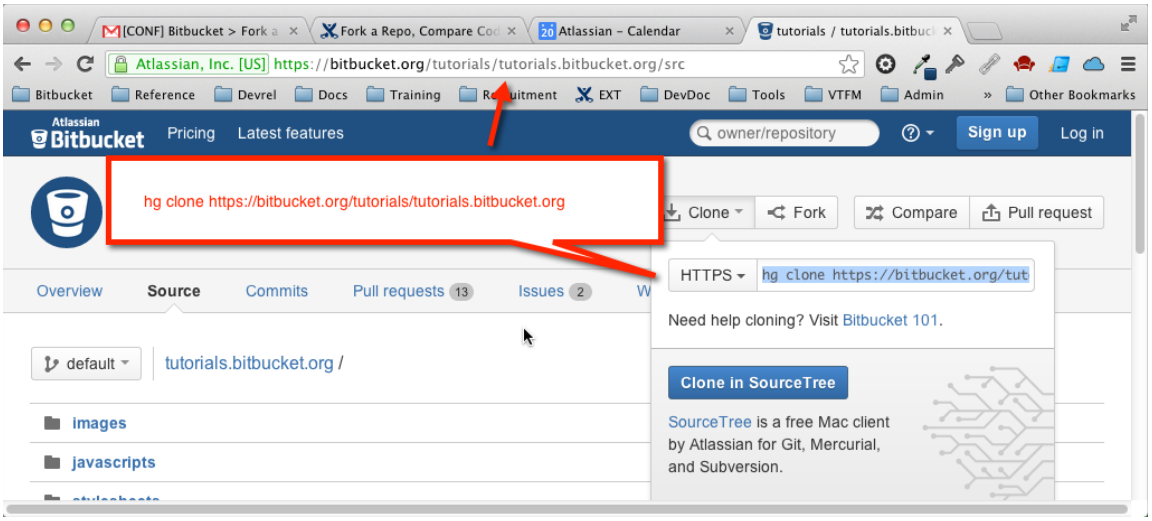
I still didn't manage to do this with Tortoise.

So I installed SourceTree and applied the same source and destination details and it worked.

I was wondering if perhaps Tortoise and SourceTree treat the server key negotiation differently.

manthony

You can't really trust the URL in the browser address bar to be correct. If you are moving fast, you might not notice they are not the same always.



The best place is to copy the URL from the Clone dialog.

Roger Erens

What is the added value of the 'Approve' toggle button? Why would one approve without merging?

Merging a request automatically implies that the request was approved, doesn't it?

The statement that clicking that button "simply means the approver reviewed the changes in the pull request." applies just as well to clicking either the 'merge', 'edit', or 'decline' buttons.

Marcus Bertrand [Atlassian]

This really depends on your specific team's workflow. Merging may not mean that all the requested reviewers have reviewed and approve of the Pull Request. Some teams have a 2 approval minimum before merging, but perhaps they merge a PR here or there without waiting for approvals for hotfixes. In that case, the code wasn't approved, but still got merged anyway. Again, this all depends on how your specific team functions and how many processes you choose to put in place.

Mark Yorkovich

The repo **REALLY** needs to be cleaned up! It's taking **forever** to download it!!!!!

manthony

Yep, I need to restart it again. I'll do that.

Mark Yorkovich

There is a step missing in Step 3.

I couldn't figure out why my changes weren't showing up in my Bitbucket repository. The green checkmark button was depressed in the toolbar, and I clicked the "Push outgoing changes to selected URL" button, but kept getting the message "no changes found" in the Output log.

But after I clicked the button that says "Commit", with the down arrow, right above the message window, my changes were pushed to the server.

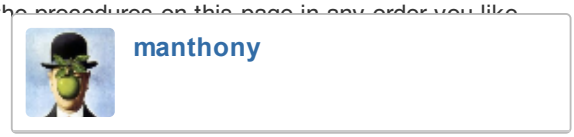
manthony

Mark, Step 3.9 is the commit step and 10 is the push. I can see the instructions for the icons looks a bit mushed up. I'll check the UI to see if it has changed since I last tested this procedure. Thank you for the feedback.

Add Users, Set Permissions, and Review Account Plans

Added by manthony, last edited by manthony on Oct 09, 2012

Forking a repo is perhaps the best way to work with others on Bitbucket. There are other ways to interact and collaborate. For example, you can follow another user or add a user to your repository. You can do the procedures on this page in any order you like.



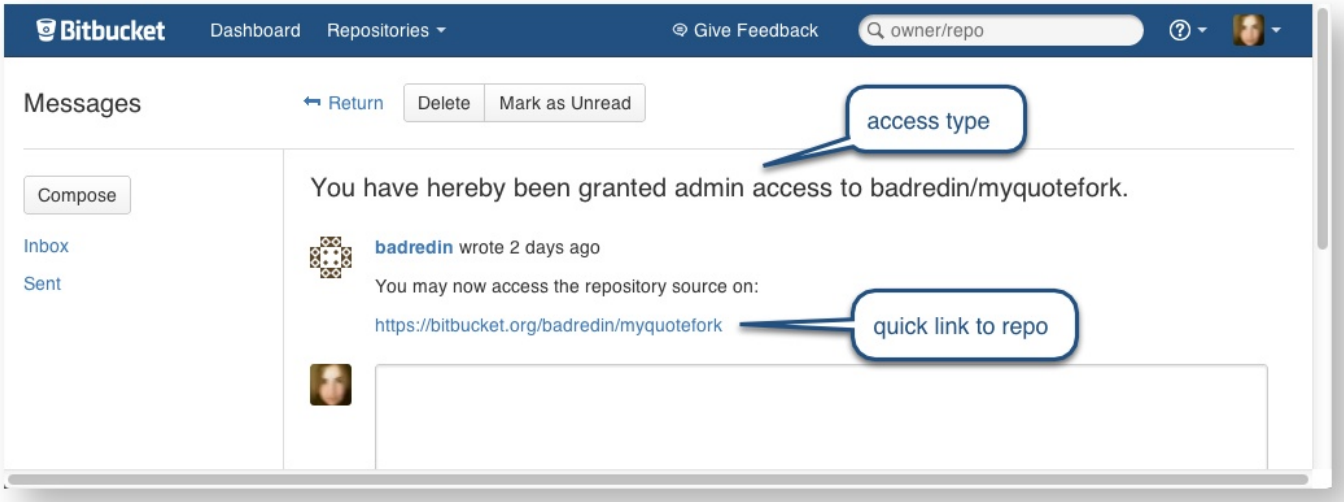
Add users to your repository

If you own a public repository, any Bitbucket user can fork that repository, make changes and send a pull request to you. However, you may want to keep your repository private. In that case, you can add users you know or invite users you think may want to contribute. For both public and private repositories, you can give users the permission to push changes directly to your repository or more, administrate your repository. To do this, you add a user and set their permissions.

1. Go to the **Admin** page for your `bb101repo` repository.
2. Click on **Access Management**.
3. Start typing a username in the field provided.
   The system provides a dropdown list of possible users that match.
   If you don't know a username you want to add, try adding the `tutorials` user or `atlassian_tutorial`.
4. Select a matching name from the dropdown.
5. Select a permission:

Level	This permissions allows a user to do the following:
read	View and fork the repository code. All public repositories grant all Bitbucket users read permissions automatically. Read access on a repository also allows users to create issues, comment on issues, and edit wiki pages.
write	Contribute to the repository by pushing changes directly from a repository on a local machine.
admin	Can do everything a repository owner can except create or delete repositories. This means administrators can: • Change repository settings. • Add, change, and remove user permissions. • Give other users administrator access.

The system adds the user to the **Users** list with the permissions you selected. It also sends an email to the user informing them about the add.



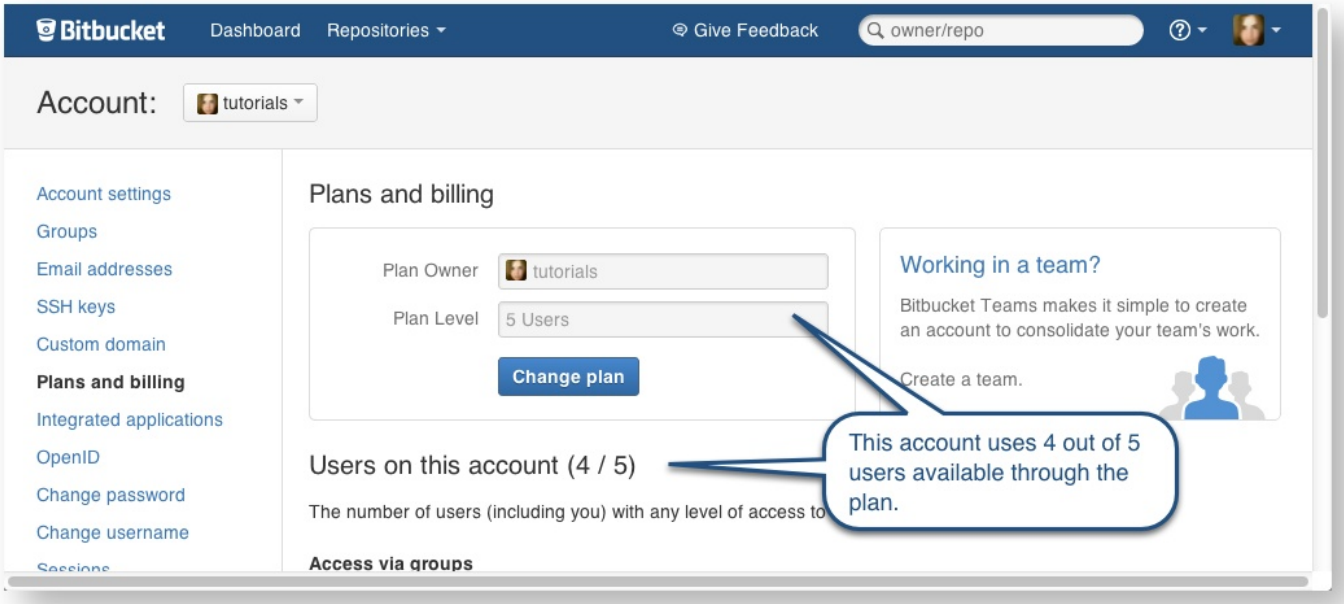
If you make a mistake, delete the user and try again.

Verify Plans and users

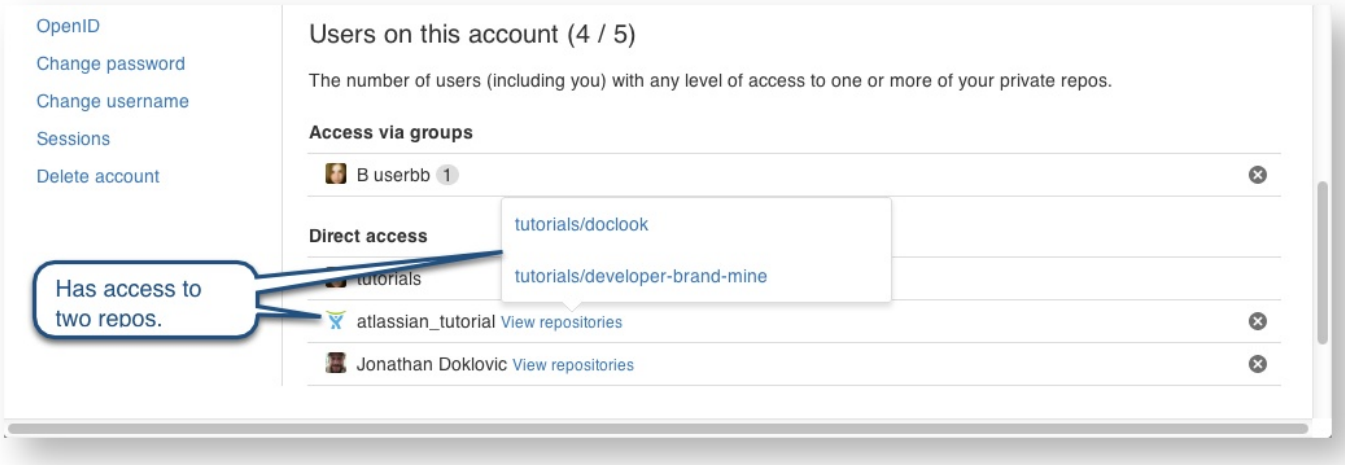
Bitbucket allows everyone with a free account an unlimited number of public and private repositories. You can grant as many users as you want access to your public repositories. Bitbucket plans restrict the number of users *with access to your private repositories*. Users with permissions such as write or admin on public repos don't count. With a free plan, you get up to five users, including yourself, with access to your private repositories.

After adding users to your account, you should review your plan and note which users "count" toward it. To view your plan, do the following:

1. Choose ***avatar*** **> Account**.
2. Choose **Plans and billing** from the left hand panel (or scroll to the page bottom).
   The system displays your current plan status:



3. Scroll down the page and review the users with access:



In this example, four users have access: **B userbb**, **tutorials** (the current account), **atlassian_tutorial**, and **Jonathan Docklovic**. Your plan should show the user you granted access to previously. How would your accounts **Plans and billing** appear if you change `bb101repo` from private to public? Go ahead and try it.

Next

Beyond code contributions, users can also contribute to your repositories by writing documentation in a wiki or making comments through an issue tracker. In the next section, you enable a wiki and an issue tracker on the `bb101repo`.

17 Comments

Anonymous

I created repositories in bitbucket.I tried to send an invite to a friend's email address. I am getting error "Repository no longer exists". I can see the repositores ofcourse.

Anonymous

How can I get more users limit for free? What abou this "Invite your friends to Bitbucket and we'll give you up to 3 additional users free."?

Because I've already invite some but I cna't see any additional users.

Anonymous

You get the additional users when they accept the Invitation

B Johnson

Am I right in thinking this offer has expired? I've invited users via "Invite a friend", they've accepted and joined our project - but no change to user allowance :-\

manthony

Yes, as long as you can see the UI for this offer it is in effect. Please log an issue with support@bitbucket.org and they'll sort you out.

KY

Hi @manthony, may i ask what kind of custom stylesheet is applied to this space? It looked amazing and would like to know how to hide the action buttons and the rest of custom header styles. Thanks.

manthony

Hi,

I'm glad you like the stylesheet. This is an experimental stylesheet for a new style on Atlassian docs. So, that means I don't know if I can give it away wholesale. I'll ask my lead about sharing and get back to you.

At the least, I think I can share how to hide the page navigation menu and page metadata.

```
/***  Visibility In-Page Nav menus **/
#navigation ul.ajs-menu-bar {
    opacity     : 0;
    position    : absolute;
    top         : 0px;
    right       : 10px;
}
#navigation ul.ajs-menu-bar:hover {
    opacity     : 1;
}

/*************************************************
** >> Visibility PAGE METADATA
*************************************************/

.page-metadata {
    position    : absolute;
    top         : 15px;
    opacity     : 0;
}

.page-metadata:hover {
    opacity     : 1.0;
}
```

You can actually turn on **Inspect Element** in your browser to find these classes yourself. Then, it is just the matter of knowing the CSS to add to your stylesheet.

Mary

Anonymous

Can a user with 'write' privileges permanently erase anything from a repository, or can the user only 'add' commits? in other words, can a user with 'write' privileges wipe a commit from the repo?

manthony

Yes, with Git a user could rewrite the history and push that rewrite. You can prevent this by using our Git branch management features. We don't have any such features on Mercurial repos. Mercurial treats history differently than Git.

Anonymous

Is there a way to give a user permissions only on a certain branch and not on the master one?

Say I need the members of the team to be able to commit and push into a development branch and only the team leader to be able to merge the code into master branch.

manthony

Those are called branch permissions. We don't have those yet. You can use forks to do this kind of thing or you could try STASH.

Anonymous

What is "avatar" and where is it located?

manthony

Upper right corner – the avatar picture. You might have a placeholder or it might be a picture you have uploaded.

Anonymous

My repository contains work done for a study project and the course policy demands the repository to be private. I would still like to show the work for example when applying for a job. How should I do this? Create a new user account with only read access to the repository (I cannot know in advance the Bitbucket account of the person reviewing my application or if he even has one).

manthony

You could zip up your project and provide it as a sample when you apply. Though, I would recommend you check with your instructor. If the work you did was on behalf of a commercial firm and mean for a live product, you might be under some agreement never to share that code.

Anonymous

Is there anyway to allow everyone write access to a branch and limit writes to specific users on the default branch?

manthony

We don't have per-branch permissions. You might be able to achieve this workflow with Branch management.

Set up a Wiki and an Issue Tracker

When you add a repository to Bitbucket, you can also enable a wiki and an issue tracker for that repository. The wiki is a simple place to keep documents. The issue tracker is the place to track your projects feature requests, bug reports and other project management tasks. The wiki and issue tracker do not depend on each other. You can choose to set up each one separately. This page leads you through the basics of using the wiki and issue tracker.

Step 1. Configure your wiki

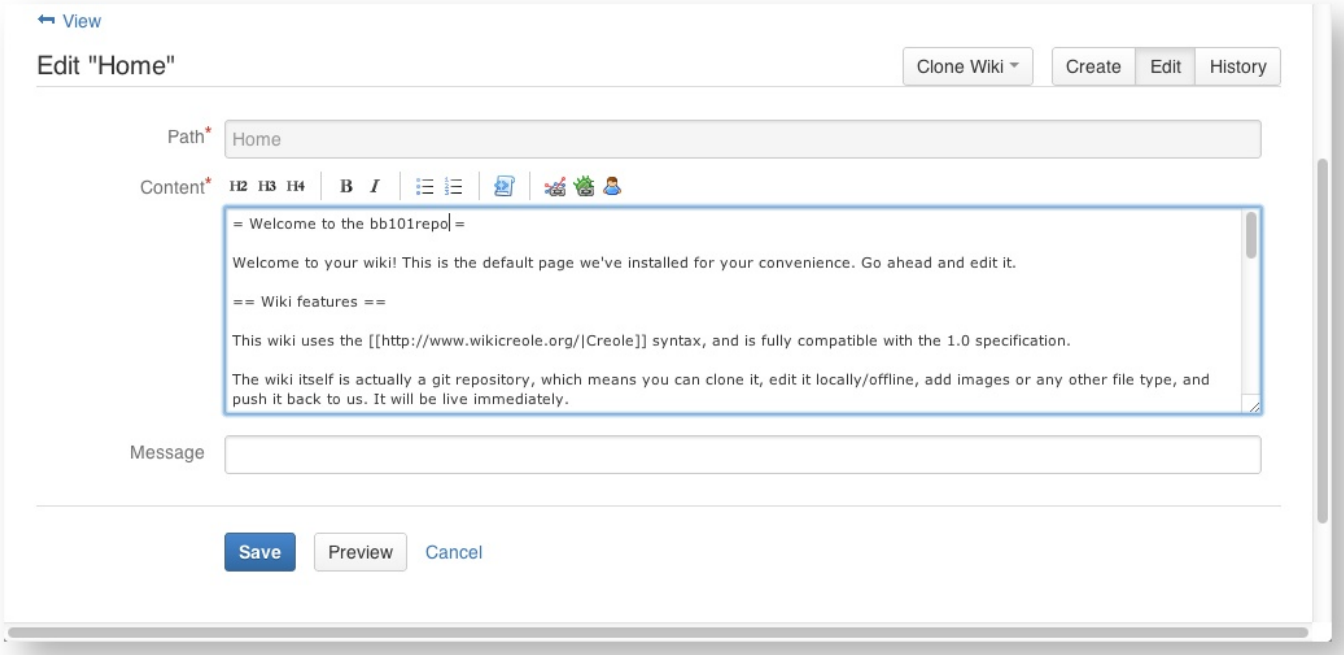
To enable the wiki for your repository, log into Bitbucket and do the following:

1. Navigate to your `bb101repo`.
2. Click the settings button.
   The system displays the **Repository details** page.
3. Choose **Wiki settings** from the left-hand panel.
   The system opens the Wiki page. A private wiki is only visible to people who have permission to see it. A public wiki anyone can view, edit, or create pages.
4. Click **Private wiki**.
5. Press **Save**.
   The system enables the **Wiki** tab for your repository.

Step 2. Update wiki pages

A Bitbucket wiki is a repository like any other, you can clone it and push changes to it. To create content, the wiki uses Creole markup.

1. Click the **Wiki** option on the repository menu bar.
   By default Bitbucket displays the wiki **Home** page.
2. Click **Edit**.
3. Make your changes to the page content.
   You may want to change the `Welcome` heading to say `Welcome to the bb101repo`.



If you want to do something more elaborate but you aren't sure how, take the **Wiki markup** link to view the help. The Bitbucket wiki uses the Creole syntax.

4. Enter a comment in the **Message** text box.

   This is a commit message. What you enter here will appear in the page history above the relevant commit entry.

5. Click **Save**.

   The system displays your home page with the new heading.

Step 3. Configure your issue tracker

To enable the issue tracker for your `bb101repo`, log into Bitbucket and do the following:

1. Navigate to a repository.
2. Click the repository settings button.

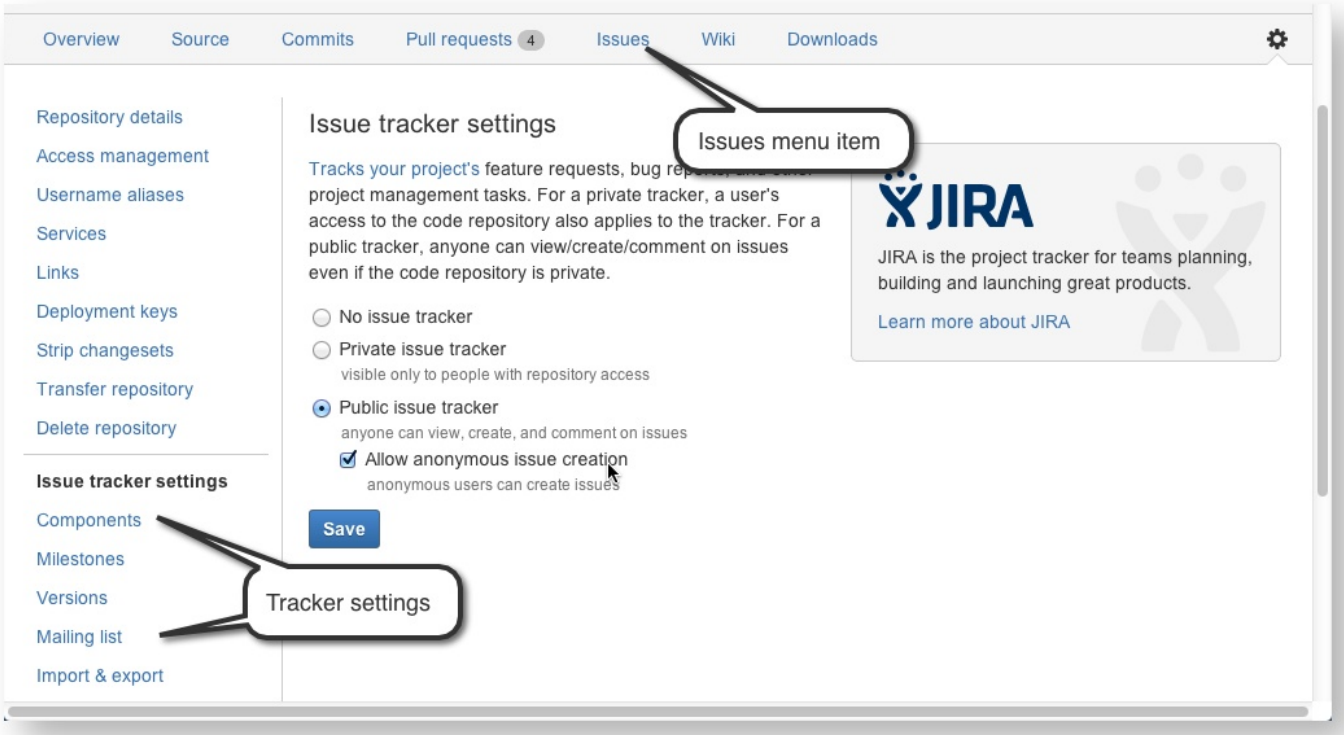
   The system displays the **Repository details** page.

3. Click **Issue tracker settings**.

   The system opens the issue tracker page. A private issue tracker is visible only to people who have permission to see it. A public tracker anyone can view or create issues.

4. Click **Private tracker**.
5. Press **Save**.

   The system enables the **Issue** menu item for your repository. You will see there are now a number of settings available.



6. Click **Components.**

   You can use components to group issues within a project into logical groups.
7. Enter a component, for example, `Site Design` and press **Add**.
8. Add a `Performance` component as well.
9. Ignore the **Milestones** setting.

   A milestone is a subset of a version. It is a point that a development team works towards. Like all settings **Milestones** is optional.

   You won't need it for this tutorial so leave it unset.
10. Click **Versions** and configure the setting by adding a 1.0 value.

    A version helps you schedule and organize your releases and track the release that is affected by a bug.
11. Click **Mailing list** and configure the setting by adding a back up email for yourself.

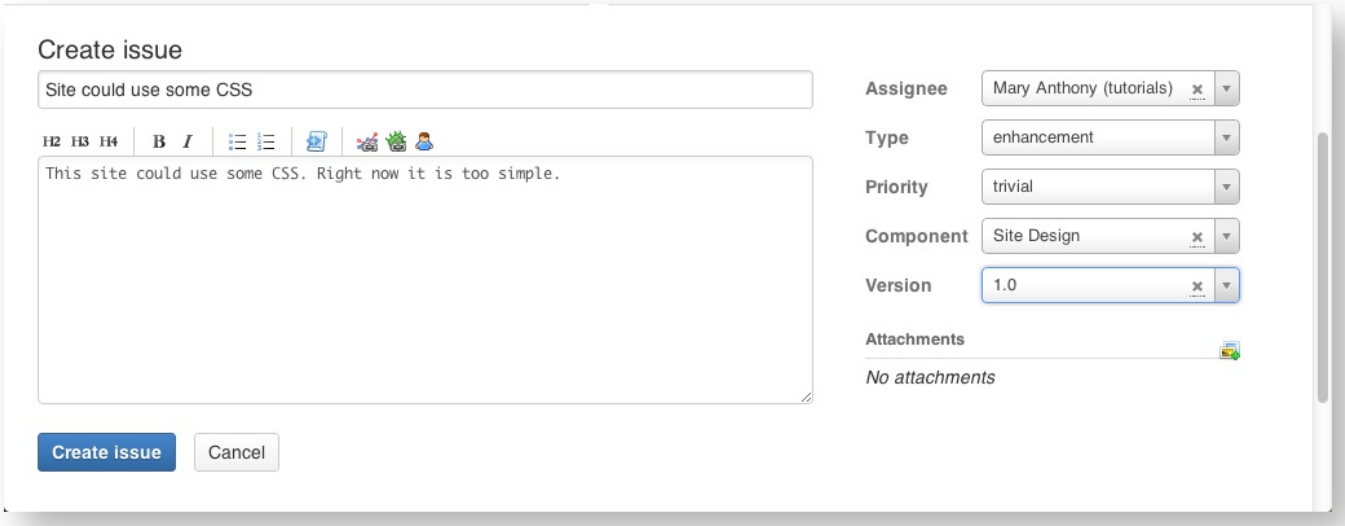
    These are email addresses that receive notification when a user creates an issue.

Step 4. Create an issue

Creating bug reports, improvement requests and tasks is as simple as creating an issue of the appropriate type.

1. Go to the repository's **Issues** tab.
2. Click **Create Issue**.
3. Fill in the fields on the issue form.

   The system displays the create issue panel:



Notice that this issue tracker is not using a milestone value.

4. Press **Create issue**.

Next Steps

Up until this point, you've used HTTPS to access your Bitbucket repos. HTTPS access requires that you give a username and password for each operation. If you are doing a lot of work, this can get annoying. In the next section, you learn how to use secure shell (SSH) to access your Bitbucket repos.

Like 2 people like this

4 Comments

Anonymous

Is there any support for a public wiki in bitbucket? Somewhere that users can report help/tips/tricks?

manthony

You can make your Wiki public. Any user with access can clone and push content changes to it.

Anonymous

It seems that the wiki currently supports Markdown instead of Creole markup, right?

manthony

You can choose which Markup language you want to use in your Wiki.

Set up SSH for Git

Up until this point, you have been using the secure hypertext transfer protocol (HTTPS) to communicate between your local system and Bitbucket. When you use HTTPS, you need to authenticate (supply a username and password) each time you take an action that communicates with the Bitbucket server. You can specify the username in the DVCS configuration file; you don't want to store your password there though where anyone can see it. So, this means you must manually type a password when you use HTTPS with your local repository. Who wants to do that? This page shows you how to use secure shell (SSH) to communicate with the Bitbucket server and avoid having to manually type a password.

Finally, setting up an SSH identity can be prone to error. Allow yourself some time, perhaps as much as an hour depending on your experience, to complete this page. If you run into issues, check out Troubleshoot SSH Issues for extra information that may help you along. You can even skip this whole page and continue to use HTTPS if you want.

Linux or Mac User?

This page shows you how to set up and use a *single default SSH identity* on Windows for a Git repository using GitBash. In the next page, you set up SSH for a Mercurial repository on Windows with TortoiseHg. If you are working on Mac OSX or Linux, a single set of instructions shows you how to setup and identity for either Git or Mercurial in these environments.

Step 1. Read a quick overview of SSH concepts

To use SSH with Bitbucket, you create an SSH identity. An identity consists of a private and a public key which together are a key pair. The private key resides on your local computer and the public you upload to your Bitbucket account. Once you upload a public key to your account, you can use SSH to connect with repositories you own and repositories owned by others, provided those other owners give your account permissions. By setting up SSH between your local system and the Bitbucket server, your system uses the key pair to automate authentication; you won't need to enter your password each time you interact with your Bitbucket repository.

There are a few important concepts you need when working with SSH identities and Bitbucket

- You cannot reuse an identity's public key across accounts. If you have multiple Bitbucket accounts, you must create multiple identities and upload their corresponding public keys to each individual account.
- You *can* associate multiple identities with a Bitbucket account. You would create multiple identities for the same account if, for example, you access a repository from a work computer and a home computer. You might create multiple identities if you wanted to execute DVCS actions on a repository with a script – the script would use a public key with an empty passphrase allowing it to run without human intervention.
- RSA (R. **R**ivest, A. **S**hamir, L. **A**dleman are the originators) and digital signature algorithm (DSA) are key encryption algorithms. Bitbucket supports both types of algorithms. You should create identities using whichever encryption method is most comfortable and available to you.

Step 2. Check if you have existing default Identity

The Git Bash shell comes with an SSH client. Do the following to verify your installation:

1. Double-click the Git Bash icon to start a terminal session.
2. Enter the following command to verify the SSH client is available:

```
manthony@MANTHONY-PC ~
$ ssh -v
OpenSSH_4.6p1, OpenSSL 0.9.8e 23 Feb 2007
usage: ssh [-1246AaCfgkMNnqsTtVvXxY] [-b bind_address] [-c cipher_spec]
[-D [bind_address:]port] [-e escape_char] [-F configfile]
[-i identity_file] [-L [bind_address:]port:host:hostport]
[-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
[-R [bind_address:]port:host:hostport] [-S ctl_path]
[-w local_tun[:remote_tun]] [user@]hostname [command]
```

3. If you have `ssh` installed, go to the next step.

   If you don't have `ssh` installed, install it now with your package manager.

4. List the contents of your `~/.ssh` directory.

   If you have not used SSH on Bash you might see something like this:

   ```
   manthony@MANTHONY-PC ~
   $ ls -a ~/.ssh
   ls: /c/Users/manthony/.ssh: No such file or directory
   ```

   If you have a default identity already, you'll see two `id_*` files:

   ```
   manthony@MANTHONY-PC ~
   $ ls -a ~/.ssh
   .    ..    id_rsa    id_rsa.pub  known_hosts
   ```

   In this case, the default identity used RSA encryption (`id_rsa.pub`). If you want to use an existing default identity for your Bitbucket account, skip the next section and go to create a config file.

Step 3. Set up your default identity

By default, the system adds keys for all identities to the `/Users/`*`yourname`*`/.ssh` directory. The following procedure creates a default identity.

1. Open a terminal in your local system.
2. Enter `ssh-keygen` at the command line.

   The command prompts you for a file to save the key in:

   ```
   manthony@PHOENIX ~
   $ ssh-keygen
   Generating public/private rsa key pair.
   Enter file in which to save the key (/c/Documents and Settings/manthony/.ssh/id_
   rsa):
   ```

3. Press enter to accept the default key and path, `/c/Documents and Settings/manthony/.ssh/id_rsa`, or you can create a key with another name.

   To create a key with a name other than the default, specify the full path to the key. For example, to create a key called `my-new-ssh-key`, you would enter a path like this at the prompt:

   ```
   manthony@PHOENIX ~
   $ ssh-keygen
   Generating public/private rsa key pair.
   Enter file in which to save the key (/c/Documents and Settings/manthony/.ssh/id_
   rsa): /c/Documents and Settings/manthony/My Documents/keys/my-new-ssh-key
   ```

4. Enter and renter a passphrase when prompted.

   Unless you need a key for a process such as script, you should always provide a passphrase.

   The command creates your default identity with its public and private keys. The whole interaction looks similar to the following:

```
manthony@MANTHONY-PC ~
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/manthony/.ssh/id_rsa):
Created directory '/c/Users/manthony/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/manthony/.ssh/id_rsa.
Your public key has been saved in /c/Users/manthony/.ssh/id_rsa.pub.
The key fingerprint is:
e7:94:d1:a3:02:ee:38:6e:a4:5e:26:a3:a9:f4:95:d4 manthony@MANTHONY-PC
manthony@MANTHONY-PC ~
$
```

5. List the contents of `~/.ssh` to view the key files.

   You should see something like the following:

   ```
   $ ls ~/.ssh
   id_rsa  id_rsa.pub
   ```

   The command created two files, one for the public key (for example `id_rsa.pub`) and one for the private key (for example, `id_rsa`).

Step 4. Create a SSH config file

1. Using your favorite text editor, edit an existing (or create a new) `~/.ssh/config` file.
2. Add an entry to the configuration file using the following format:

   ```
   Host bitbucket.org
    IdentityFile ~/.ssh/privatekeyfile
   ```

   The second line is indented. That indentation (a single space) is important, so make sure you include it. The second line is the location of your private key file. If you are following along with these instructions, your file is here:

   ```
    ~/.ssh/id_rsa
   ```

   When you are done editing, your configuration looks similar to the following:

   ```
   Host bitbucket.org
    IdentityFile ~/.ssh/id_rsa
   ```

3. Save and close the file.
4. Restart the GitBash terminal.

Step 5. Update your .bashrc profile file

It is a good idea to configure your GitBash shell to automatically start the agent when launch the shell. The `.bashrc` file is the shell initialization file. It contains commands that run each time your GitBash shell starts. You can add commands to the `.bashrc` file that start the agent when you start GitBash. The folks at GitHub have developed a nice script for this (their script was developed from a post by Joseph M. Reagle Jr. from MIT on the cygwin list). To start the agent automatically, do the following.

1. Start GitBash.
2. Edit your `~/.bashrc` file.

   If you don't have a `.bashrc` file you can create the file using your favorite text editor. Keep in mind the file must be in your ~ (home) directory and must be named exactly `.bashrc`.

3. Add the following lines to the file:

> Chrome and Opera introduce ASCII \xa0 (non-breaking space characters) on paste that can appear in your destination file. If you copy and paste the lines below, copy from another browser to avoid this problem.

```
SSH_ENV=$HOME/.ssh/environment

# start the ssh-agent
function start_agent {
    echo "Initializing new SSH agent..."
    # spawn ssh-agent
    /usr/bin/ssh-agent | sed 's/^echo/#echo/' > "${SSH_ENV}"
    echo succeeded
    chmod 600 "${SSH_ENV}"
    . "${SSH_ENV}" > /dev/null
    /usr/bin/ssh-add
}

if [ -f "${SSH_ENV}" ]; then
     . "${SSH_ENV}" > /dev/null
     ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
        start_agent;
    }
else
    start_agent;
fi
```

4. Save and close the file.
5. Close GitBash.
6. Reopen GitBash.

   The system prompts you for your passphrase:

```
Welcome to Git (version 1.7.8-preview20111206)
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.
Enter passphrase for /c/Documents and Settings/manthony/.ssh/id_rsa:
```

7. Enter your passphrase.

   After accepting your passphrase, the system displays the command shell prompt.
8. Verify that the script identity added your identity successfully by querying the SSH agent:

```
$ ssh-add -l
2048 0f:37:21:af:1b:31:d5:cd:65:58:b2:68:4a:ba:a2:46 /Users/manthony/.ssh/id_rsa (RSA)
```

After you install your public key to Bitbucket, having this script should prevent you from having to enter a password each time you push or pull a repository from Bitbucket.

Step 6. Install the public key on your Bitbucket account

1. Open a browser and log into Bitbucket.
2. Choose ***avatar*** **> Manage Account** from the menu bar.

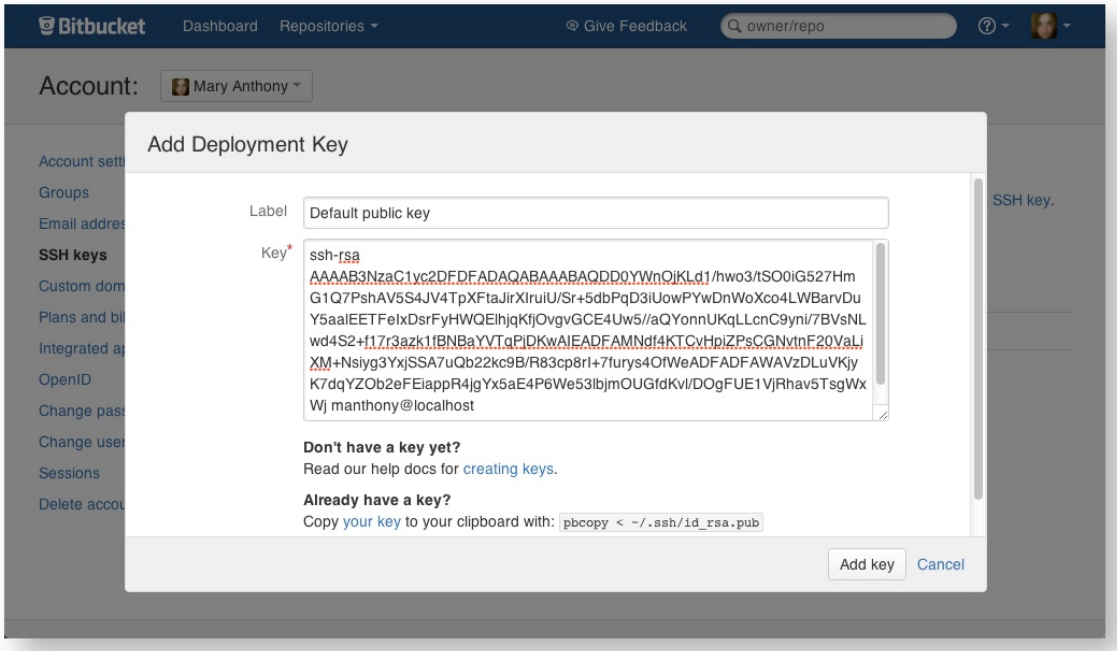
   The system displays the **Account settings** page.
3. Click **SSH keys**.

   The **SSH Keys** page displays. It shows a list of any existing keys. Then, below that, a dialog for labeling and entering a new key.
4. In your terminal window, `cat` the contents of the public key file.

   For example:

```
cat ~/.ssh/id_rsa.pub
```

5. Select and copy the key output in the clipboard.
   If you have problems with copy and paste, you can open the file directly with Notepad. Select the contents of the file (just avoid selecting the end-of-file character).
6. Back in your browser, enter a **Label** for your new key, for example, `Default public key`.
7. Paste the copied public key into the **SSH Key** field.
8. Click the **Add key** button:



   The system adds the key to your account.
9. Return to the terminal window and verify your configuration by entering the following command.

```
ssh -T git@bitbucket.org
```

   The command message tells you which Bitbucket account can log in with that key.

```
conq: logged in as tutorials.
You can use git or hg to connect to Bitbucket. Shell access is disabled.
```

10. Verify that the command returns your account name.

    Click if you got a permission denied (publickey) message.

    The command tests your connection to Bitbucket as a Git user. It first sees if your SSH Agent has an identity loaded. The command then checks if that private key matches a public key for an existing Bitbucket account. You might have either problem.

    To make sure your identity is loaded, enter the following command:

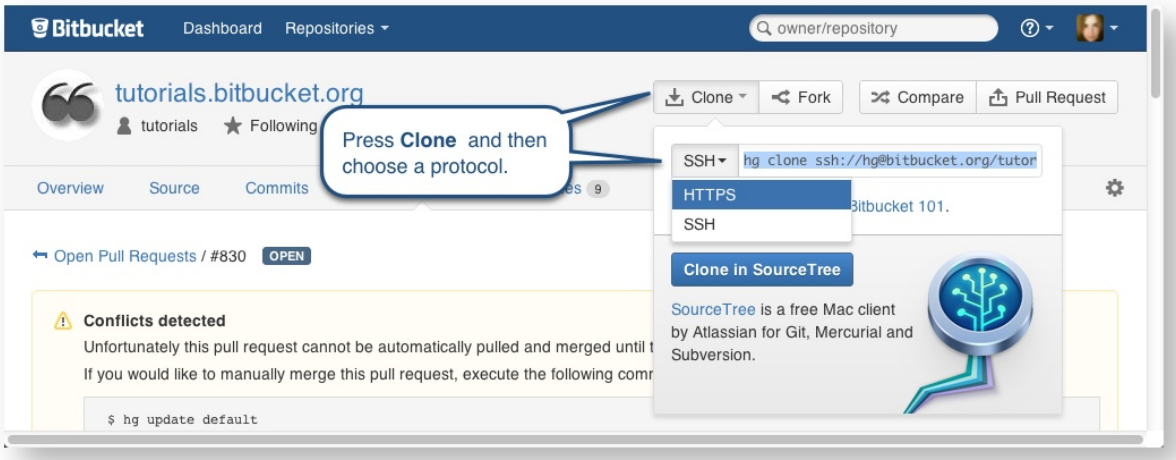
```
ssh-add -l
```

    If the identity isn't loaded, check your work in Step 5 above. If it is loaded, try reinstalling your public key on your Bitbucket account.

    Still having troubles? You can try our Troubleshoot SSH Issues page or email support@bitbucket.org.

Step 7. Configure your repository to use the SSH protocol

The URL you use for a repository depends on which protocol you are using, HTTPS and SSH. The Bitbucket repository **Overview** page has a quick way for you to see the one for your `bb101repo` repo. On the repository's **Overview** page look for the **Clone this repository** line.



Experiment for a moment, click back and forth between the **SSH** and the **HTTPS** protocol links to see how the URLs differ. The table below shows the format for each DVCS based on protocol.

	SSH URL format	**HTTPS URL format**
Mercurial	`ssh://hg@bitbucket.org/`*`accountname`*`/`*`reponame`*`/`	`https://`*`accountname`*`@bitbucket.org/`*`accountname`*`/`*`reponame`*

Git	`git@bitbucket.org:`*`accountname`*`/`*`reponame`*`.git`
	or
	`ssh://git@bitbucket.org/`*`accountname`*`/`*`reponame`*`.git`
		`https://`*`accountname`*`@bitbucket.org/`*`accountname`*`/`*`reponame`*`.git`

In the SSH format, the `accountname` appears *after* `git@bitbucket.org` or `hg@bitbucket.org`. In HTTPS format, the `accountname` *before* `git@bitbucket.org` or `hg@bitbucket.org`.

Go to terminal on your local system and navigate to your `bb101repo-practice` repository. Then, do the following:

1. View your current repository configuration.
   You should see something similar to the following:

```
manthony@MANTHONY-PC ~
$ cat .git/config
[core]
  repositoryformatversion = 0
  filemode = true
  bare = false
  logallrefupdates = true
  ignorecase = true
[remote "origin"]
  fetch = +refs/heads/*:refs/remotes/origin/*
  url = https://newuserme@bitbucket.org/newuserme/bb101repo.git
[branch "master"]
  remote = origin
  merge = refs/heads/master
```

   As you can see, the `url` is using the HTTPS protocol. There are a number of ways to change this value, the easiest way is just to edit the repository's configuration file.

2. Edit the `~/repos/bb101repo-practice/.git/config` file with your favorite editor.
3. Change the `url` value to use the SSH format for that repository.

   When you are done the `origin` section should contain something similar to the following:

```
[remote "origin"]
  fetch = +refs/heads/*:refs/remotes/origin/*
  url = git@bitbucket.org:newuserme/bb101repo.git
```

4. Save your edits and close the file.

Step 8. Make a change under the new protocol

1. Edit the `README` file in your `bb101repo-practice` repository.
2. Add a new line to the file, for example:

```
Welcome to My First Repo
------------------------------
This repo is a practice repo I am using to learn bitbucket.
You can access this repo with SSH or with HTTPS.
```

3. Save and close the file.
4. Add and then commit your change to your local repo.

```
git add README
git commit -m "making a change under the SSH protocol"
```

5. Push your changes.

   The system warns you that it is adding the Bitbucket host to the list of known hosts.

```
manthony@MANTHONY-PC ~
$ git push
$ git push
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 287 bytes, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: bb/acl: newuserme is allowed. accepted payload.
To git@bitbucket.org:newuserme/bb101repo.git
  056c29c..205e9a8 master -> master
```

6. Open the repo **Overview** in Bitbucket to view your commit.

Next Steps

Now, you should set up SSH for Mercurial through the TortoiseHg Workbench. If you are a Mac OSX or Linux user, you should have already worked through one page that covers SSH for both Git and Mercurial.

55 Comments

Anonymous

Step 5 didn't work for me. Problem and resolution described here: http://stackoverflow.com/questions/11918285/my-bashrc-file-not-executed-on-git-bash-startup-windows-7

Anonymous

thanks!

Anonymous

thaaaaaanks, it's very helpful

Anonymous

ok... really thanks...

Lalith

Thanks for your help here anon. I also had problem getting .bashrc to run auto in step 5, but by adding the following lines to Git bash, my .bashrc file was saved in the right file type and runs sucessfully;

```
cd ~
touch .bashrc
notepad .bashrc
```

Anonymous

Thanks, this finally got me unstuck!

Anonymous

It didn't work on my PC (windows7 / 64bit).

How can I work it out?

manthony

Hi,

You can try the script we show in Step 5 above or you can try this script:

```
SSH_ENV=$HOME/.ssh/environment

function start_agent {
     echo "Initialising new SSH agent..."
     /usr/bin/ssh-agent | sed 's/^echo/#echo/' > ${SSH_ENV}
     echo succeeded
     chmod 600 ${SSH_ENV}
     . ${SSH_ENV} > /dev/null
     /usr/bin/ssh-add;
}

# Source SSH settings, if applicable

if [ -f "${SSH_ENV}" ]; then
     . ${SSH_ENV} > /dev/null
     #ps ${SSH_AGENT_PID} doesn't work under cywgin
     ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
         start_agent;
     }
else
     start_agent;
fi
```

which others have reported works in Windows 7. If you still are having problems, please send a note to support@bitbucket.org.

(I'm in the process of expanding my test environments so as to test both scripts in more environments.)

Mary

Anonymous

Hi,

I tried the above script on Win7 32bit. It showed a success message for the script. I had already changed the origin path in the *.git/config* file. I also changed the repo settings from https to ssh. I tried '*ssh -T git@bitbucket.org*' and I got the message '*Permission denied (publickey)*'. So I tried '*ssh-add -l*' and got the message '*Could not open a connection to your authentication agent*'.

Please note that my repo is private. Is this the reason? Thanks in advance 🙂

manthony

See our troubleshooting guide explains how to deal with this.

Anonymous

Hi After Step 5.6

I am not getting this screen after restarting the bash.

```
Welcome to Git (version 1.7.8-preview20111206)
Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.
Enter passphrase for /c/Documents and Settings/manthony/.ssh/id_rsa:
```

Instead I am getting prompt. Can some help what went wrong ?

I am quite new to this and need to configure ssh for bitbucket for running shell script.

Thanks,

Vickram

Anonymous

If your $HOME contains space characters, for example: C:\Documents and Settings\myuser, please use double quote in each ${HOME}, so above script will be as follow:

```
SSH_ENV=$HOME/.ssh/environment

# start the ssh-agent
function start_agent {
echo "Initializing new SSH agent..."
# spawn ssh-agent
/usr/bin/ssh-agent | sed 's/^echo/#echo/' > "${SSH_ENV}"
echo succeeded
chmod 600 "${SSH_ENV}"
. "${SSH_ENV}" > /dev/null
/usr/bin/ssh-add
}

if [ -f "${SSH_ENV}" ]; then
. "${SSH_ENV}" > /dev/null
ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
start_agent;
}
else
start_agent;
fi
```

Anonymous

Great! It works for me on WinXP. Thanks!

Anonymous

I'm hung up on Step 4 and I don't want to skip it. Part 1 says "Using your favorite text editor, edit an existing (or create a new) ~/.ssh/config file." I see no config file in my .ssh directory and I don't understand what I'd put into a new config file if I created one (other than the lines that are indicated to enter in Step 4, Part 2.) I realize this must be really basic.

manthony

Hey no worries, SSH can be a showstopper for very experienced people too. If you don't have the config file, you should create one. Then, you put in the lines that appear in Step 4.2

```
Host bitbucket.org
 IdentityFile ~/.ssh/id_rsa
```

The id_rsa is the default name for the private key. If you are following along in this tutorial, you will have this key. If you keep having problems, send an email to support@bitbucket.org for faster response time and a more personal help.

Anonymous

For any folks (like me) having trouble "creating" the config file, which should be named "config" not "config.txt": when you save the file in NotePad, name the file "config" with the quotation marks. This will keep NotePad from adding the ".txt".

Anonymous

Step 5 also did not work for me and I tried following: http://stackoverflow.com/questions/11918285/my-bashrc-file-not-executed-on-git-bash-startup-windows-7 but the same problem still persists, after typing git push I get a long message that says:

```
warning: push.default is unset; its implicit value is changing in Git 2.0 from
'mathcing' to 'simple'
```

as well as some other stuff that I can't copy and paste. At the end it says

```
conq: invalid command syntax.
fatal: Could not read from remote repository.
Please make sure you have the correct access rights and the repository exists.
```

Anyone know what I'm doing wrong?

Anonymous

I don't see any .git folder in my repository. So how do I know what's in the config file in .git folder when it doesn't exist.

Can somebody please point out to me where I'm going wrong?

manthony

In Windows, the folder is likely to be hidden. Windows default is to hide these types of files in the view. Try googling "show hidden files" for your version of Windows.

Anonymous

Thanks, great tutorial!!!

manthony

U r welcome!

Anonymous

After create .bashrc file and reopen git in step 5,

I've always the error "Could not open a connection to your authentication agent.".

initialization ssh agent is succeeded

someone could help me please?

Brett Crawford

The link above seems to point back to this page. This link to the Troubleshooting SSH Issues page should work.

Anonymous

Is that possible that my problem come from the fact that I'm in 64 bits ?

Because I tried what the site suggested me, but it's still don't work

Anonymous

Well, I couldn't get past step 5 too, so i had to do some googling. Important part for me was:

```
$ eval 'ssh-agent'
```

In the example quotes are wrong and i had to use ` (above TAB key) quotes intead of '.

Then I did :

```
$ ssh-add path/to/my/.ssh/id_rsa
```

and after i restarded Git Bash, it finaly asked me for pass phrase.

Also, I had to open id_rsa.pub with text editor, copy/pasting from Git Bash gave me "Invalid SSH key." error.

Peter Le

Thanks for the info Anon! This helped me as well.*Like*

Anonymous

here's how i solved step 5 on win xp:

```
SSH_ENV="$HOME/.ssh/environment"

function start_agent {
     echo "Initialising new SSH agent..."
     /usr/bin/ssh-agent | sed 's/^echo/#echo/' > "${SSH_ENV}"
     echo succeeded
     chmod 600 "${SSH_ENV}"
     . "${SSH_ENV}" > /dev/null
     /usr/bin/ssh-add;
}

# Source SSH settings, if applicable

if [ -f "${SSH_ENV}" ]; then
     . "${SSH_ENV}" > /dev/null
     #ps ${SSH_AGENT_PID} doesn't work under cywgin
     ps -ef | grep ${SSH_AGENT_PID} | grep ssh-agent$ > /dev/null || {
         start_agent;
     }
else
     start_agent;
fi
```

double qoutes everywhere SSH_ENV is used, not sure if they are needed on the definition but they don't hurt.

Anonymous

This worked for me - I think it was because my home directory ($HOME) referenced my computer username which had a space in it...

Anonymous

Had this problem too. Should upvote this post or something

Anonymous

Step 5 still isn't working for me, when I type in `ssh-agent` I get: socket: Invalid argument.

manthony

From http://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch12_02.htm:

> Make sure the agent points to a valid socket:
>
> ```
> $ ls -lF $SSH_AUTH_SOCK
> prwx --  --  --    1 smith   0 May 14 23:37 /tmp/ssh-smith/ssh-22719-agent|
> ```
>
> If not, your SSH_AUTH_SOCK variable might be pointing to an old socket from a previous invocation of ssh-agent, due to user error. Terminate and restart the agent properly.

I'm sorry for the slow update on this page, I'm waiting on some Windows resources to test and try to duplicate all the issues folks run into.

Alex D

Step5 - When i run the Git bash, it's asking for password for "~ / .ssh/id_rsa" and not for my new created key. How to choose right key in this way?

Anonymous

I solved Step 4/5 with some of the steps over at github's page: (https://help.github.com/articles/generating-ssh-keys), helped me through this. I replaced the id_rsa - since I couldn't get a new key name to work. I was having the same problem as Alex D above.

Anonymous

Step 5 not work.

Anonymous

This says its a windows tutorial but 'ssh-keygen' is not a windows command prompt commadn

manthony

The tutorial assumes you are entering all your commands in a Git Bash terminal window. Git Bash does provide ssh-keygen.

Anonymous

First account works fine. But I can't do a git push to my 2nd bitbucket account. It simply says:

"conq: repository access denied.
fatal: The remote end hung up unexpectedly"

When I try: "ssh -T git@bitbucket.org" it connects to the first account.
How do I get it to connect/use my 2nd ssh key for 2nd account?

manthony

The error message indicates that your second account may not have a SSH public loaded. Did you check the SSH settings?

Anonymous

I have a weird problem - I've passed step 5 successfully. If I make a change and push it, I can see it updating in my panel.

However, git still asks me for the password, albeit *after* uploading the changes...

Isn't it supposed to stop asking me for a password once I setup SSH?

manthony

Hmm, if you have uploaded your key to Bitbucket and your shell is configured correctly, it should only ask you once, each time you start GitBash. You might want to show me the full command input and output...hard to diagnose from a description. The order does seem wrong.

Anonymous

THUMBS UP my friend

Anonymous

Is step 5 necessary if you are not using Git Bash on windows, but using terminal on mac?

manthony

If you are using terminal on Mac, you should use these instructions which not have this step.

Anonymous

So, after a bit of confusion I realized if on Linux disregard all the restarting of GitBash!!! I had to: source ~/.bashrc ...before I could check: ssh -T git@bitbucket.org

Martin Gregory

I wonder why we go to all this trouble to set up gitbash for Windows, when a Windows user would likely rather use a Windows shell? (I know, heaven help them, but still, it's true).

It can be done, using Putty.

manthony

GitBash setup is pretty contained compared with PuTTy. Another strong benefit of GitBash is that it allows me to write command line examples that work across platforms.

Martin Gregory

Yes - I too personally heaps prefer gitbash. However, I have been helping set up an environment for windows users, and they know nothing about unix at all. This being the case, having a purely windows solution is powerful ... which got me wondering why there isn't more discussion about doing this. I agree having to set up putty is a bit messy, but then so is the ssh-agent bashrc when you think about it....

Anonymous

Stucked in step 5. I created the .bashrc file but after restarting GitBash I wasn't asked for my passphrase. When I tried to verify if I have an identity I got the 'The agent has no identities' message. Please help! Thanks.

Dan Walker

My user account has spaces in the username (something I try to avoid but that's how our Windows domain is set up), which causes the .bashrc posted above to fail.

Like "Anonymous" above, I needed to enclose all occurrences of ${SSH_ENV} in double quotes: "${SSH_ENV}".

Rob Emenecker

I can get all of this working, but my question is, isn't the point of using SSH with git so that I don't have to enter a passphrase? I use TortoiseGit and am still prompted from a passphrase at every command when using TortoiseGit. So I end up using a SSH key that is not password protected. Not what I want to do, but I don't want to have to enter the passphrase every instance of running a command via TortoiseGit.

Does anyone have any suggestions?

manthony

What SSH agent are you using Rob?

Rob Emenecker

I am currently just using the ssh client included with msysgit.

I am on a Windows 7 64-bit workstation with mysysgit 1.8.4 and TortoiseGit 1.8.5.0.

I can get everything to work okay using Pagent and TortoisePlink, but still need to enter the password when Pagent first runs. In that instance, I have a Pagent startup shortcut that is set to start in my .ssh directory and then load the keys from that directory, though I still need to enter my passcode(s) when I start it.

This is **not a Bitbucket issue.** It's a configuration issue on my end and I am just trying to get it running as effectively as possible.

manthony

Rob, you may want to google around. I found this http://blog.shvetsov.com/2010/03/making-pageant-automatically-load-keys.html.

Anonymous

There's something I don't get from this tutorial.

Suppose this situation: Bob has an bitbucket account with username "Bob" and a repo called "fum". He has given read-only permissions to "fum" to Fred, who has a bitbucket account with username "BigFred". Fred has a windows PC and his account name on this windows PC is "me". Now Fred wants to setup an ssh key for accessing the fum repository from his windos PC without having to type passwords all the time.

So, where does Fred specify that he wants to use the accountname "BigFred" on bitbucket? It's not in the ssh url, which is git@bitbucket.org:bob/fum.git, it's not "me", it's not inside the ssh key, and it's nowhere in the ~/.ssh folder.

Martin Gregory

Fred has to log into bitbucket as BigFred and supply the ssh keys to bitbucket.

Then Fred uses that same key to log into bitbucket - it's BigFred's key, bitbucket can see that.

Set up SSH for Mercurial

If you are following the tutorial, you have already set up your Git repositories to use SSH. This page has you set up your Mercurial repositories to use SSH as well. If you skipped the Git instructions, you missed the explanation of basic SSH concepts.

Setting up an SSH identity can be prone to error. Allow yourself some time, perhaps as much as an hour depending on your experience, to complete this page. If you run into issues, check out Troubleshoot SSH Issues for extra information that may help you along. You can even skip this whole page and continue to use HTTPS if you want.

GitBash, Linux, or Mac User?

This page shows you how to set up and use a *single default SSH identity* on Windows for a Mercurial repository. If you want set up for a Git repository on Windows using GitBash, there are instructions for Set up SSH for Git. Finally, if you are working on Mac OSX or Linux, a single set of instructions shows you how to setup and identity for both Git or Mercurial in these environments.

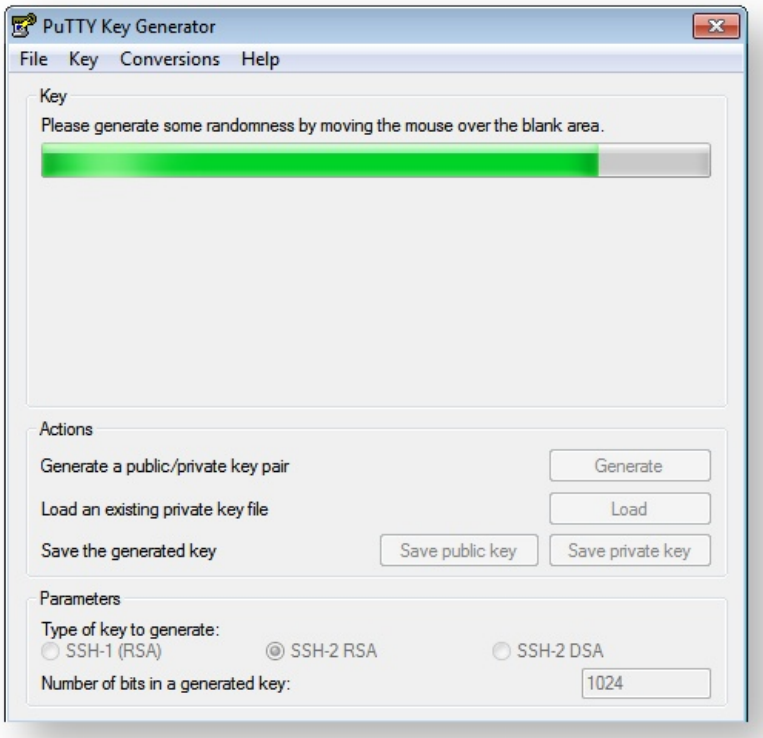
Step 1. Check if you have existing default identity

TortoiseHG relies on the PuTTYgen utility to generate an identity. If you are following along in this tutorial, you should have already installed PuTTYgen. If you used PuTTYgen in the past to generate an identity, you should have an `identity.ppk` file on your system. If you have an existing private key, you can skip Step 2 below and go onto Enable SSH compression for Mercurial. Otherwise, continue to the next step.

Step 2. Create your default identity

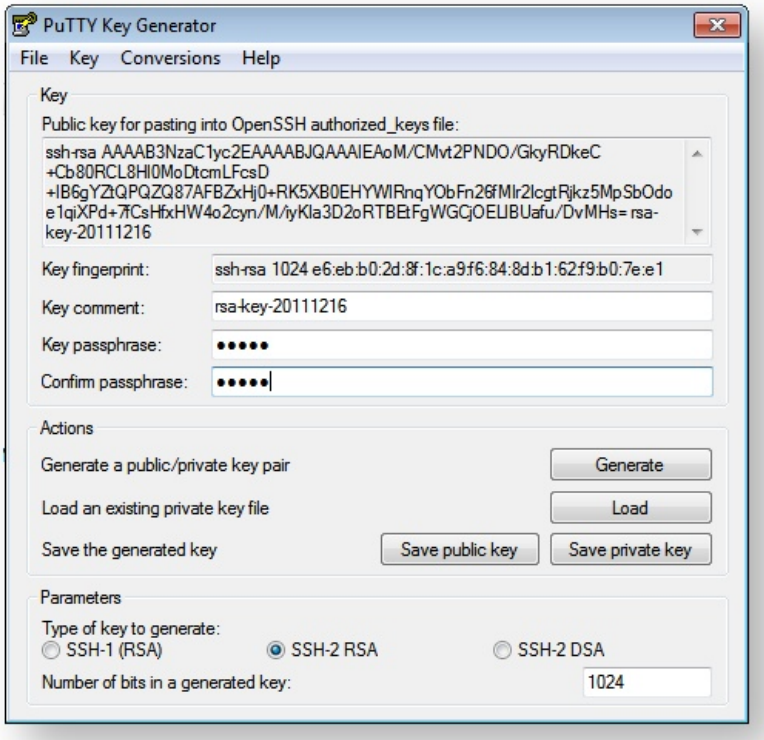
The following procedure creates a default identity with PuttyGen. PuttyGen generates keys in its own format. So, rather than re-use the file you created for use within GitBash, you'll create a different key for TortoiseHG.

1. Locate the `puttygen.exe` executable in your system and double click the icon to start it.
   If you are following along with this tutorial, you installed PuTTYgen in `C:\Program Files\TortoiseHG`. The system opens the **PuTTY Key Generator** dialog.
2. Complete
3. Press **Generate**.
   Following the instructions to generate some randomness.



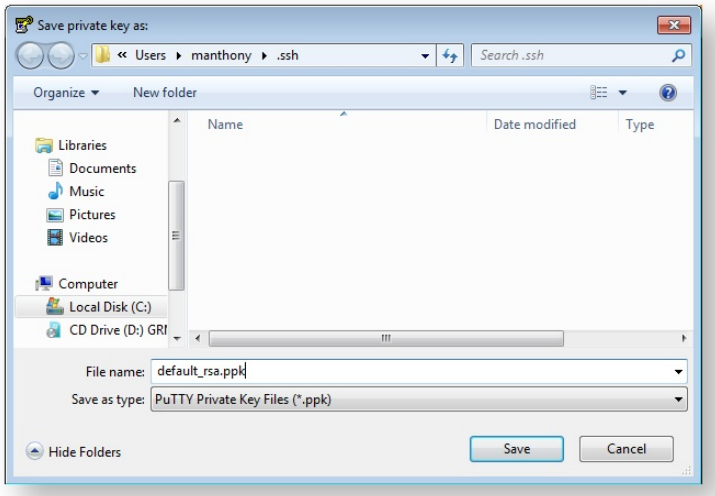
When the generation completes, the system displays the public key and a number of other fields.

4. Enter and confirm a key passphrase.



5. Press **Save private key**.

   The system prompts you for a location to save the file and a file name. By convention, store your key files in a folder called `C:\Users\`*`yourname`*`\.ssh`. and give it a `.ppk` extension.



6. Go ahead and close PuTTYgen.

Step 3. Enable SSH compression for Mercurial

When sending or retrieving data using SSH, Git does compression for you. Mercurial does not automatically do compression. If you are using Mercurial, you should enable SSH compression as it can speed up things drastically, *in some cases*. To enable compression for Mercurial, do the following:

1. Start the TortoiseHg Workbench.
2. Select **File > Settings**.
3. Make sure you have the global settings tab selected.
4. Press **Edit File**.
5. Add the following line to the UI section:

   ```
   ssh = "TortoisePlink.exe" -ssh -2 -batch -C
   ```

   When you are done the file should look similar to the following:

   ```
   [ui]
   # Name data to appear in commits
   username = Mary Anthony <manthony@atlassian.com>
   ssh = "C:\Program Files\TortoiseHg\TortoisePlink.exe" -ssh -2 -batch -C
   ```

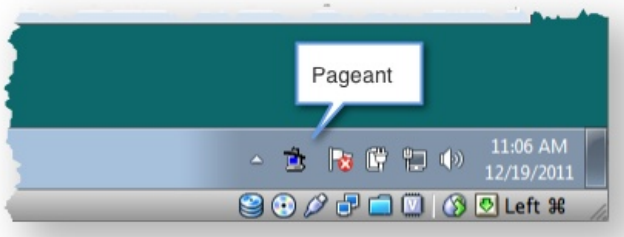
6. Press **Save** to store your settings and close the file.
7. Press **OK** to close the settings dialog.

Step 4. Start Pageant and install your private key

TortoiseHG comes with Pageant which is an SSH authentication agent. You load your keys into Pageant and it automatically authenticates you so you don't need to enter your passphrase. Do the following to load your keys:

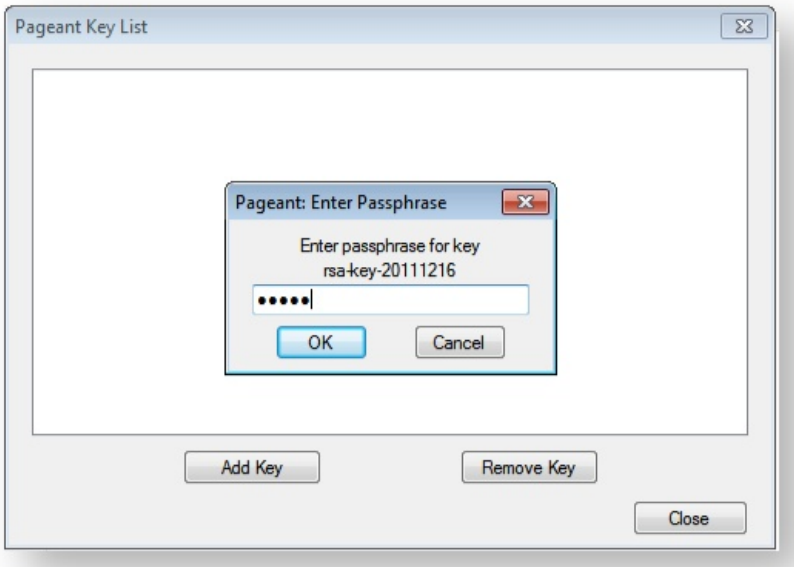
1. Start Pageant by double clicking its icon.

   By default, TortoiseHG installs the Pageant in the `C:\Program Files\TortoiseHG` folder. When it is running, Pageant appears in your system tray:



2. Double-click the Pageant icon to launch the **Pagent Key List** dialog.
3. Click the **Add Key** button.

   The system displays the **Select Private Key File** dialog.
4. Navigate to and open the default key you created previously.
5. Enter the passphrase when prompted:



6. Press **OK**.

   Pageant shows your key in the running list.
7. Press **Close** to close the dialog.

   Pageant continues to run on your system.

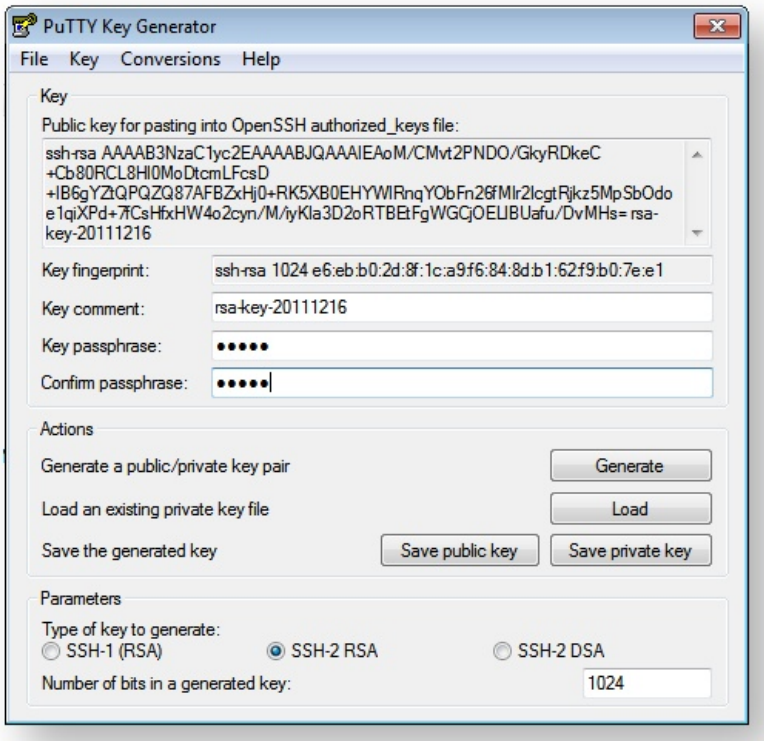
Step 5. Install the public key on your Bitbucket account

1. Open a browser and log in to Bitbucket.
2. Choose ***avatar*** **> Account** from the menu bar.

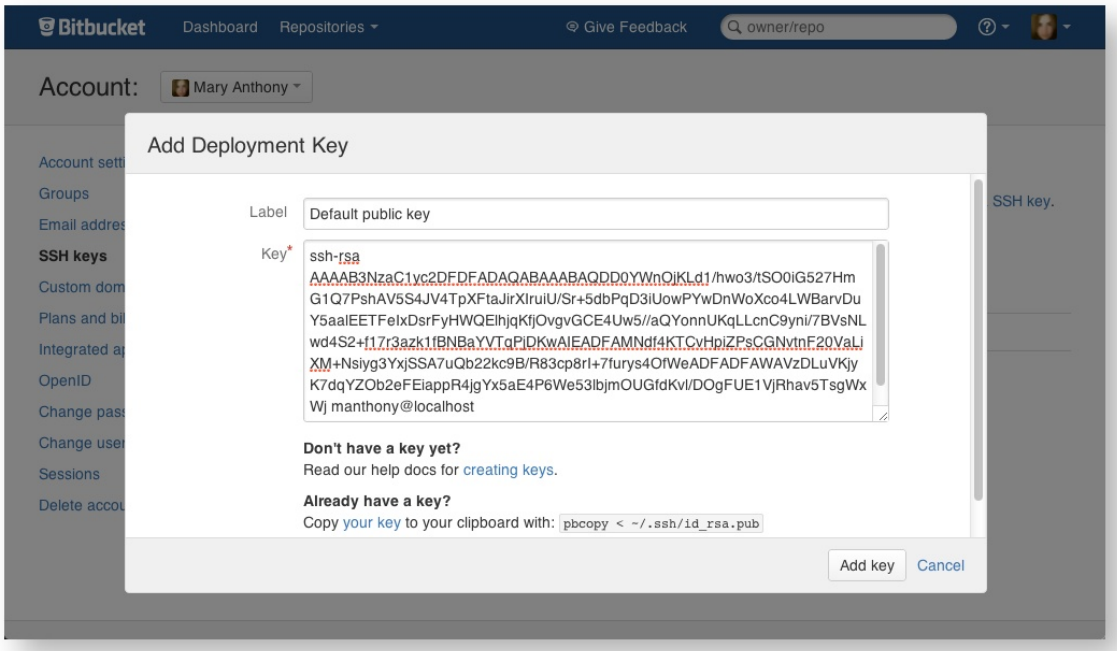
   The system displays the **Account settings** page.
3. Click **SSH keys**.

   The **SSH Keys** page displays. It shows a list of any existing keys. Then, below that, a dialog for labeling and entering a new key.
4. Switch to your local desktop and start the PuTTYgen program.
5. Press **Load**.
6. Navigate to and open your default private key.
7. Enter your passphrase when prompted and press OK.

   The system displays your public key.



8. Select and copy the contents of the **Public key for pasting into OpenSSH authorized_keys file** field.
9. Back in your browser, enter a **Label** for your new key, for example, `Default public key`.
10. Paste the copied public key into the **SSH Key** field:



11. Press **Add key**.

    The system adds the key and it appears in the **SSH Keys** listing.
12. Close PuTTYgen.

Step 6. Configure your local repository to use the SSH protocol

The URL you use for a repository depends on which protocol you are using, HTTPS and SSH. The Bitbucket repository **Overview** page has a quick way for you to see the one for your `myquotefork` repository. On the repository's **Overview** page look for the **Clone this repository** line. Experiment for a moment, click back and forth between the **SSH** and the **HTTPS** protocol links to see how the URLs differ. The table below shows the format for each DVCS based on protocol.

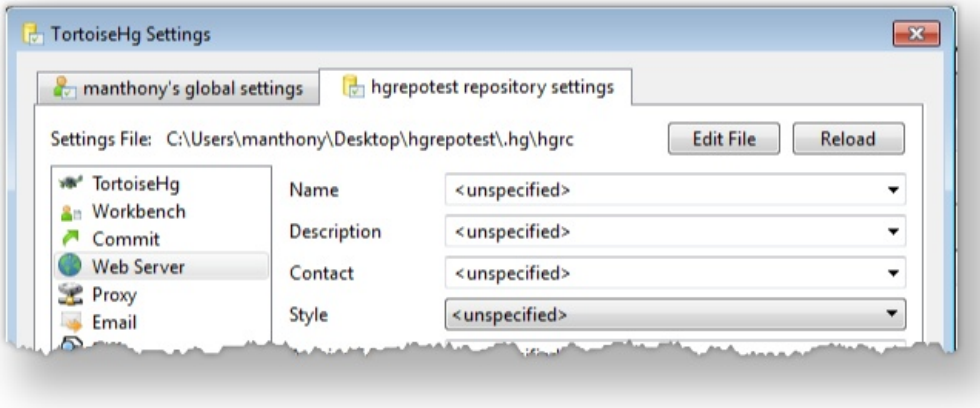
	SSH URL format	**HTTPS URL format**
Mercurial	`ssh://hg@bitbucket.org/`*`accountname`*`/`*`reponame`*`/`	`https://`*`accountname`*`@bitbucket.org/`*`accountname`*`/`*`reponame`

Git	`git@bitbucket.org:`*`accountname`*`/`*`reponame`*`.git`
	or
	`ssh://git@bitbucket.org/`*`accountname`*`/`*`reponame`*`.git`
		`https://`*`accountname`*`@bitbucket.org/`*`accountname`*`/`*`reponame`*`.git`

In the SSH format, the `accountname` appears *after* `git@bitbucket.org` or `hg@bitbucket.org`. In HTTPS format, the `accountname` *before* `git@bitbucket.org` or `hg@bitbucket.org`.

Got to your local system and navigate to your `myquotefork` repository (the only Mercurial repository you've worked with so far). These instructions assume you have added the repository to the TortoiseHG Workbench.

1. Start TortoiseHG.
2. Right click your `myquotefork` repository and choose **Settings**.
   The system displays the **TortoiseHG Settings** dialog with the **myquotefork repository settings** tab active.



3. Press **Edit File**.
4. View your current repository configuration.
   You should see something similar to the following:

```
[paths]
default = https://bitbucket.org/newuserme/myquotefork
```

5. Change the `default` value to use the SSH format for that repository.
   When you are done you should see something similar to the following:

```
[paths]
default = ssh://hg@bitbucket.org/newuserme/myquotefork
```

6. Press **Save** to close the editor.
7. Press **OK** to close the settings dialog.
8. Restart TortoiseHG Workbench so that it uses the new SSH setting.

Step 7. Make a change under the new protocol

1. Edit the `Index.html` file in your `myquotefork` repository.
2. Add a new line to the file.
3. Save and close the file.
4. Add and then commit your change to your local repository.
5. Push your changes to your fork.

   A successful push shows in your TortoiseHG log as follows:

   ```
   pushing to ssh://hg@bitbucket.org/newuserme/myquotefork
   searching for changes
   remote: adding changesets
   remote: adding manifests
   remote: adding file changes
   remote: added 1 changesets with 1 changes to 1 files
   remote: bb/acl: newuserme is allowed. accepted payload.
   [command completed successfully Mon Dec 19 10:49:06 2011]
   myquotefork
   ```

   PuTTY may warn you that the host key is not yet stored. If that happens, press **Yes** to add the bitbucket.org host key.
6. Open the repository **Overview** in Bitbucket to view your commit.

Next Steps

You've completed the 101 tutorial for Bitbucket. At this point, you should have a good beginners knowledge of what you can do in Bitbucket (You should also make sure you have checked https://tutorials.bitbucket.org to see your pull request changes incorporated.) If you are a Mac user, you might want to see SourceTree a Free Git and Mercurial GUI (Mac OSX). Windows users can try Sourcetree Git GUI on Windows.

The rest of the documentation has more topics and information that can help you make the most of Bitbucket. Please let us know what you thought of this tutorial by logging an issue or by sending an email. Of course, you can always contribute by commenting on or editing a page directly.

30 Comments

Anonymous

You may skip the Pageant step if you put the path to the ssh key on the global settings.

Just have it like this instead:

```
[ui]
# Name data to appear in commits
username = Mary Anthony <manthony@atlassian.com>
ssh = "C:\Program Files\TortoiseHg\TortoisePlink.exe" -ssh -2 -batch -C -i C:\Users\<YourAccount>\.ssh\private.ppk
```

manthony

Hi,

I take your point. I was aware of that method. I'll see about adding it to our SSH guide. For beginners, I tend to err on the side of showing the moving parts. There are a couple of reasons for this. First I like a beginner to reach the gross understanding of the component parts of any thing I'm teaching. Second, implementing an advanced method where the mechanism is hidden in a tool's configuration file, is much simpler to explain once a reader has the gross understanding. Later, if that beginner needs to debug a situation, they are better able to troubleshoot.

Mary

Anonymous

By doing it this way without Pageant, you must enter your passphrase for every time.

Anonymous

This was exactly what I was looking for, thanks.

I found that the full path tp TortoisePlink.exe was not needed, and %USERPROFILE% can be used in place of C:\Users\<YourAccount> thus:

```
ssh = "TortoisePlink.exe" -ssh -2 -batch -C -i %USERPROFILE%\.ssh\private.ppk
```

Anonymous

Brilliant!!

Anonymous

As good as I hoped when I started. Thanks–your bosses should give you a raise!!!

Cody Jackson

Actually I was able to get that working. I'm not entirely sure what the issue was, but I realized on GitBash I had about 13 ssh-agent processes running. So I killed all those, and went through the entire setup again and was finally able to get it working.

manthony

I'm glad you got it sorted Cody.

Anonymous

Needed to update the global settings path for Program files when using 64bit windows 8

ssh = "C:\Program Files\TortoiseHg\TortoisePlink.exe" -ssh -2 -batch -C

banterCZ

It still asks for credentials (user/password) in command line and in Tortoise workbench as well. Using 64bit Windows 7.

Anonymous

Thanks!, nice tutorial for a newbie like me.

but when I push from TortoiseHg al last step, I have this output log:

remote: conq: not an hg repository.
no suitable response from remote hg

I searched in google but I not found a good explanation for this error log :$

u know? ty

manthony

It could be your SSH key is not on Bitbucket.

Anonymous

Check that our repository type is of Mercurial type. If you accidentally created a repository for Git, you will obtain this error.

Anonymous

This is a great tutorial but when trying to push data to bitbucket from tortoisehg, this error appears:

"Disconnected: No supported authentication methods available (server sent: publickey)"

Do you have any idea how to fix this?

Thanks

manthony

Make sure Paegent is running and that your key is installed.

Anonymous

In step 4, I tried to use the private key that I had created for use with git (id_rsa) using ssh-keygen. Pageant wouldn't accept it. I had to create a separate key pair for use with Hg using PuttyGen. Do ssh-keygen and puttygen use two different and incompatible private key formats?

Lars

manthony

PuttyGen uses its own format. You'll need convert it. Here is a good article for that. http://meinit.nl/using-your-openssh-private-key-in-putty.

Anonymous

P.S. if so, it would be helpful in step 4 to say explicitly that you can't use the private key that you generated in "Set up SSH for Git".

manthony

Point taken. I added it to the tutorial in Step 2 when I ask people to create a new identity.

Mark Stega

Working through the BB101 tutorial I successfully got SSH to work on the Git repository. Following the next step to get the same access for Hg repositories I was not so successful. I have access to BB according to ssh:

> Mark@CAPTAINNEMO ~
> $ ssh -T hg@bitbucket.org
> logged in as mstega2.
>
> You can use git or hg to connect to Bitbucket. Shell access is disabled.
>
> Mark@CAPTAINNEMO ~
> $

I tried both a new PPK generation and a conversion of the key created when enabling SSH for Git. I've added the private key to Pageant and the public key to BB.

When I try to use the SSH connection to BB from the TortoiseHG Workbench I get a PuTTY Fatal Error - Server refused to start a shell/command. The log contains:

> % hg --repository S:\Solutions\OHI\bb101-QuoteFork push ssh://hg@bitbucket.org/mstega2/ms-tutorialquotefork
> pushing to ssh://hg@bitbucket.org/mstega2/ms-tutorialquotefork
> no suitable response from remote hg
> [command returned code 255 Wed Sep 18 13:15:50 2013]
> bb101-QuoteFork%

I am able to access the repository using HTTPS, just not SSH. I'm not certain where/how to further investigate. Any suggestions are welcome.

manthony

Hmm, have you tried deleting all the keys on BB. Then, just testing with the Paegent generated PPK? (Reducing the variables a bit.)

Mark Stega

I just tried that - No change

I started over from the very beginning (removed/reinstalled TortoiseHg). I don't know what the issue was, but it is gone now; I added Pageant to the Windows start folder (you run shell:startup with Win+R) with the path to the key as a parameter so it starts upon logon (easier than adding the key each time you start Pagaent).

I do understand why you wanted to use PuttyGen to create a new key pair but if you do that then the name of the key in Step 5 should be different than the name used in the Git SSH tutorial.

manthony

Mark glad you got it sorted. You caught bug in the doc. I'm reusing the screen capture for uploading the key to Bitbucket so the label shows up the same in both. I'll fix that. Thank you for the catch.

Anonymous

I don't quite get it why I need to set-up both Git and Mercurial, from what it seems having only one is enough ...

manthony

Well, you don't have to do anything. Since the tutorial is an introduction to Bitbucket and Bitbucket supports both DVCS platforms, I ask you to install both. You never know when you might encounter some cool Mercurial repo.

Course, if you absolutely know you'll never touch Mercurial. Just skip those bits that ask you to do things with it.

Marcus Bertrand [Atlassian]

Depending on how you plan to use Bitbucket, it may be a smart idea to have both available to you. There are many projects out there only in Mercurial or only in Git. It is just the same as probably wanting subversion installed. It only takes a small effort to install, with a big time savings when you need them.

Anonymous

Thanks very much for a great tutorial! I'm a newbie programmer and never would have figured this out without your tutorial.

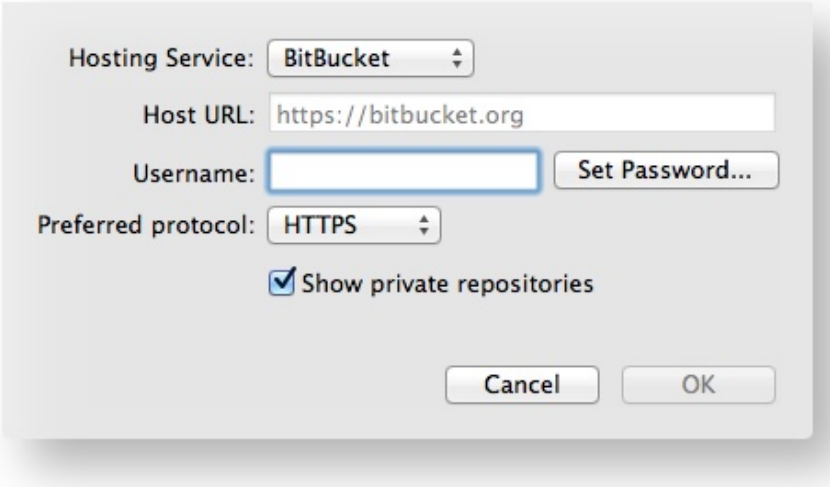
SourceTree a Free Git and Mercurial GUI (Mac OSX)

Atlassian SourceTree is a GUI client you can use to quickly and easily view the status of Git, Mercurial, and Subversion source repositories. SourceTree provides special support for hosted DVCS systems such as Bitbucket and GitHub. Using a single SourceTree client, you can work with local repositories and hosted repositories in multiple source control systems. Why not install SourceTree and give it a try to see if it is a fit for you?

Adding a hosted project in SourceTree

If you haven't already, start SourceTree. Then, do the following:

1. Choose **View > Show Hosted Projects** from the menu bar.
   The system displays the **Hosted Repositories** dialog.
2. Press **Edit Accounts**.
   The system displays a list of the hosted accounts it knows about. If this is the first time you have run this, the list is empty.
3. Press **Add Account**.
4. Complete the dialog:

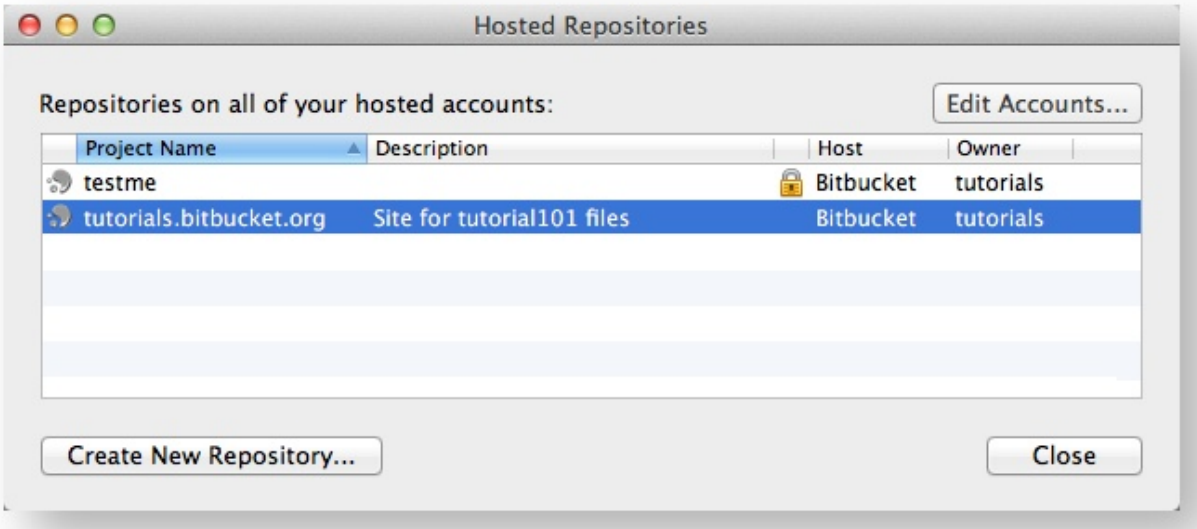


   The **Username** is your account name. The password you provide is your account password. You can choose whether you prefer connecting through HTTPS or SSH.
5. Press **OK**.

SourceTree returns you to the list of hosted accounts. You can add another if you like.

6. Press **Close** when you are done adding accounts.

   The system displays a list of your hosted repositories and repositories your account is following.



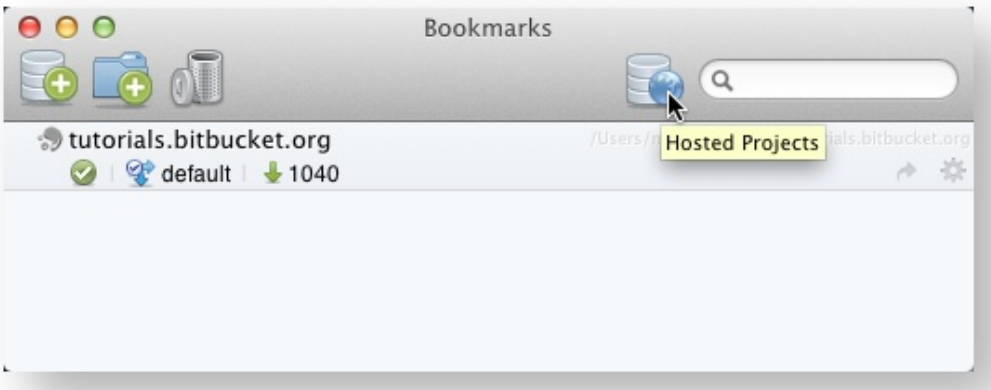
7. Press **Close**.

You'll notice that the SourceTree does not list your hosted repositories on your **Bookmarks** list. The bookmarks list is only for local repositories; you'll need to clone from a hosted repository or create a local repository to add to this list.

Clone a hosted project

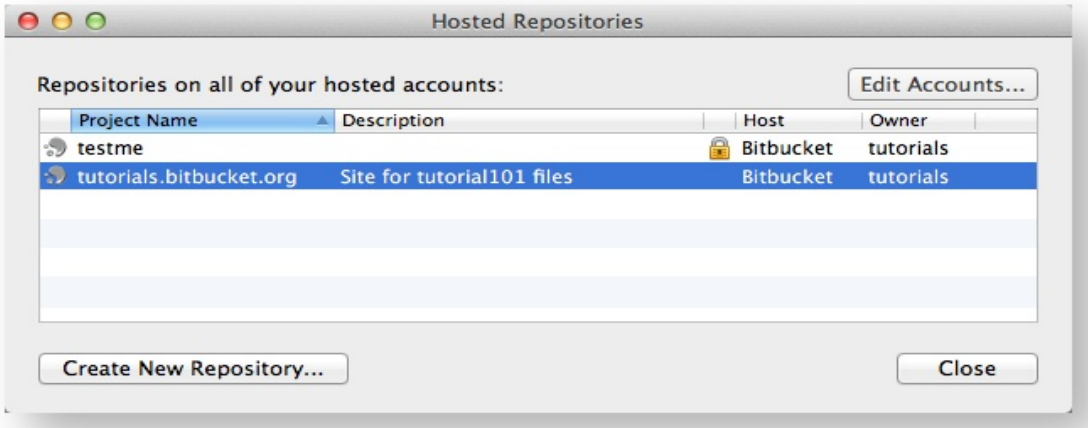
To clone a hosted project to your local machine, you can do the following:

1. Press **Hosted Projects**.



   The system displays your hosted projects.

2. Select a project on the list.
3. Right-click to show the context menu.

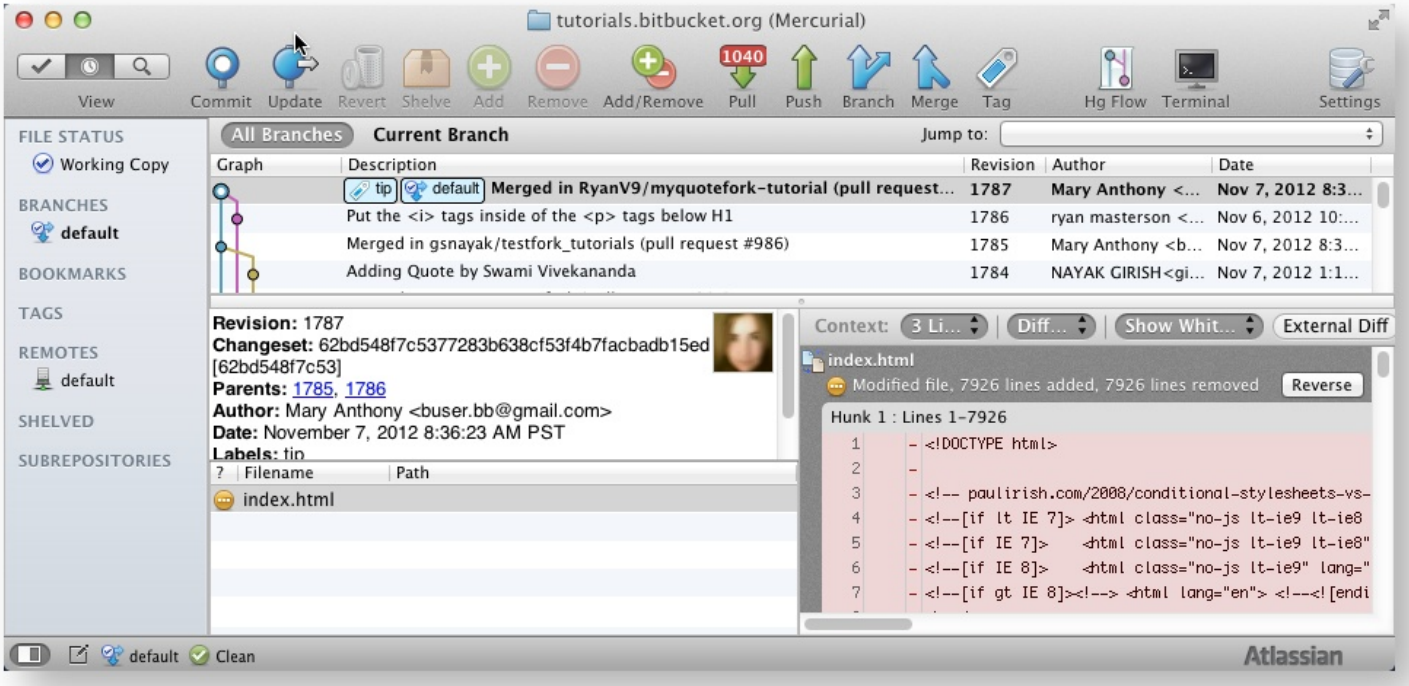


4. Click **Clone**.

   The system adds the clone to your list of bookmarks.

Doing more with SourceTree

There is a lot more you can do with SourceTree as you can see with this repository viewer:



To learn more, try SourceTree yourself or review the Help that comes with it.

7 Comments

Azeem Shaikh

Is there any comparable GUI tool for Windows users?

Kelly Schoenhofen

TortoiseGit is the closest tool for Windows users, IMO.

http://code.google.com/p/tortoisegit/

Anonymous

SourceTree is now available for Windows!!

Anonymous

I prefer Gitbox. Any chance of adding one click support?

manthony

The best chance of getting this is to file an enhancement request.

Anonymous

Tried following this tutorial, but really didn't want to install or learn how to use the command-line Git tools on my Mac when I could be using SourceTree... Is there a tutorial in the works that goes into the same depth as this tutorial but focuses on using SourceTree with Bitbucket?

manthony

Sorry no. You can add your voice to this post on Answers.

Sourcetree Git GUI on Windows

Atlassian SourceTree is a GUI client you can use to quickly and easily view the status of Git source repositories on Windows. SourceTree provides special support for hosted DVCS systems such as Bitbucket and GitHub. Using a single SourceTree client, you can work with local repositories and hosted repositories in multiple source control systems. Why not install SourceTree and give it a try to see if it is a fit for you?

Clone a Git project in SourceTree

If you haven't already, start SourceTree. Then, do the following:

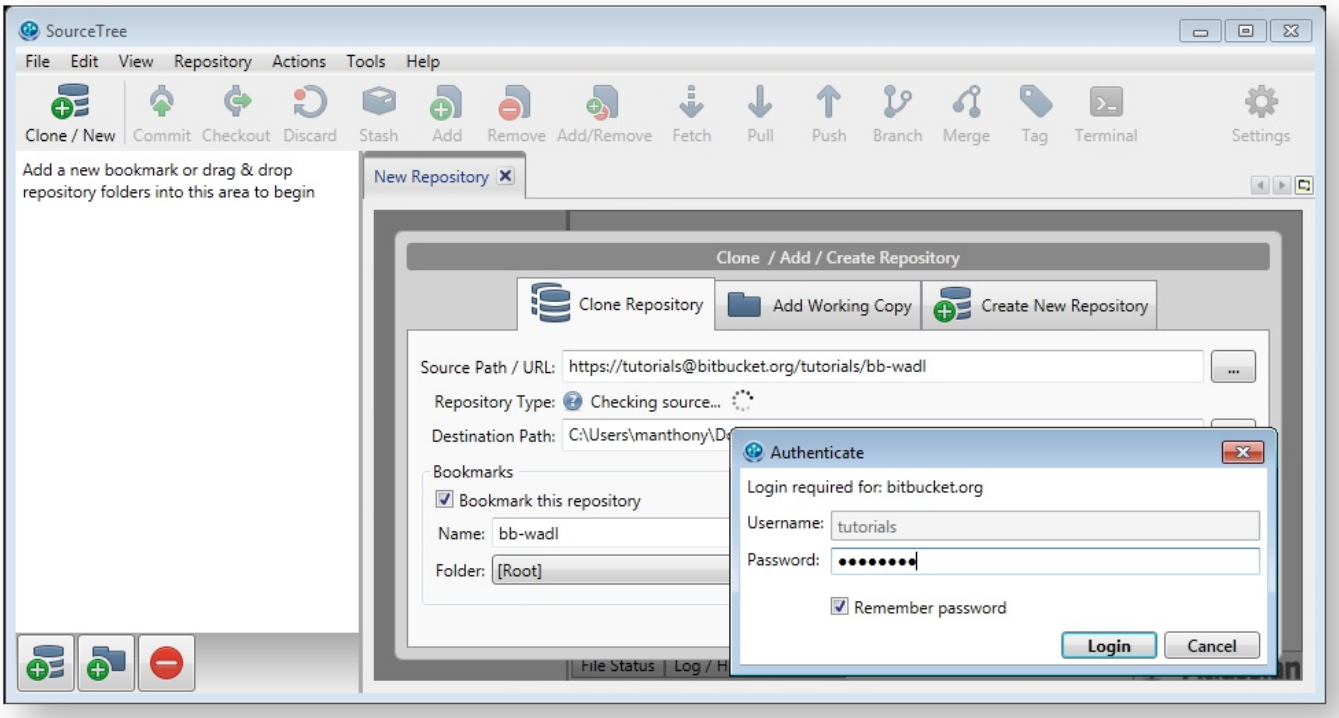
1. Choose **Clone / New** from the menu bar.

   The system displays the **Clone / Add Create Repository** dialog.
2. Enter a **Source Path / URL**.

   You can enter the URL for a remote repository on Bitbucket here. Try entering the URL for your `bb101repo`, for example:

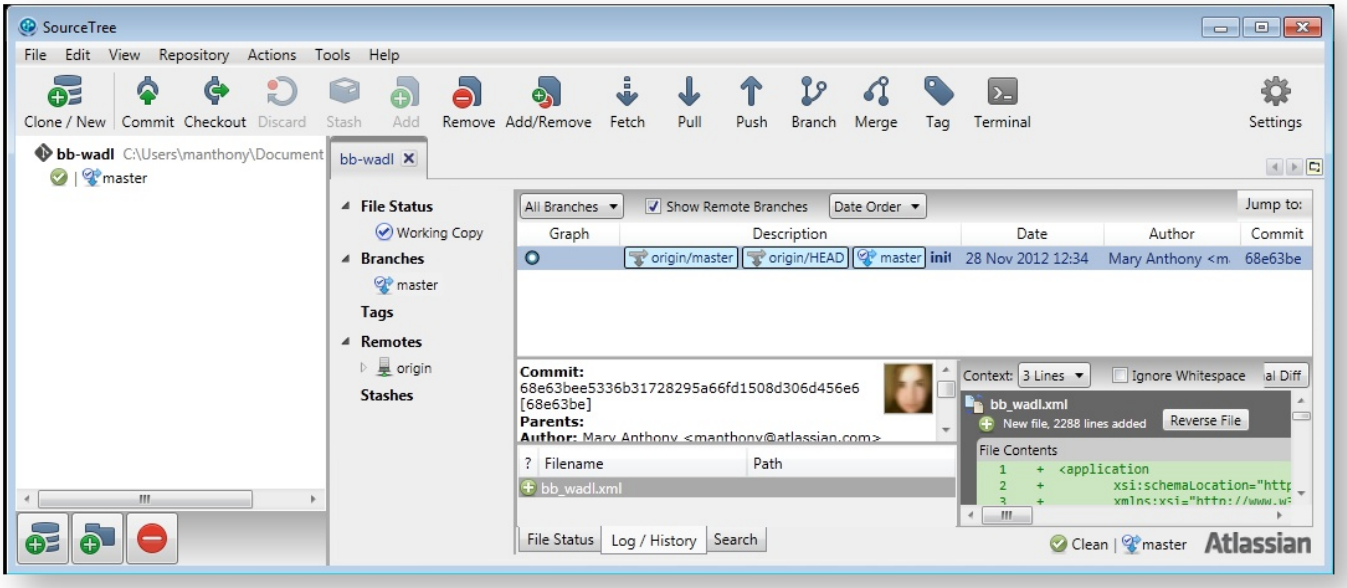
   `https://tutorials@bitbucket.org/tutorials/bb101repo`

   The system requests a username password.



3. Enter your password and press **Login**.
4. Press **Clone**.

   SourceTree returns you to the main window:



   The bookmarks list is only for local repositories; you'll need to clone from a hosted repository or create a local repository to add to this list.