**Source Control**

**Questions to companies:**

* What is the current source control server for undergoing projects?
* Is the new project developed on single or multiple platforms?
* How many people will interact with the revision repository of the new project?
* Are developers going to work in one site or different sites?
* Is this position responsible for the setup of source control server?
* What protocol had been applied on existing Git server?
* Are we using internal server or external hosting server like GitHub?
* Is code review needed?
* Why are there **two shell interfaces** for Git, Cygwin & msysGit?

Cygwin is a Linux shell, msysGit is a Windows shell.

* **Cygwin** **VS** **msysGit**?

|  |  |
| --- | --- |
| Cygwin | msysGit |
| **Support Chinese input and display perfectly** | Need specific configuration |
| Command line shell | **Provide visual shell TortoiseGit which integrated msysGit and windows file management** |

* **Traits of distributed revision control system** comparing with centralized revision control system.
* Revision repository is placed under the root of workspace.
* To secure repository, clone the repository and enable auto push.
* Editable username and email are not secure. Server repository can set up authentication (Gerrit).
* Why are **alias** used in Git?

Make it easier for CVS & SVN users to use CVS/SVN alike command in Git.

* Primary commands of **creating repository**

git init / git add / git commit / git config

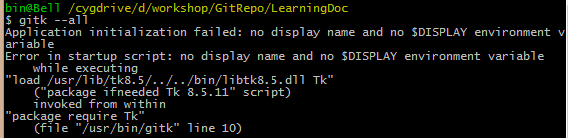
* Benefits of **Stage** in Git
  + Stage makes it possible to get rid of dependency upon internet
  + Much faster to commit to local stage rather than remote server
  + Easier to perform revert operations
* **Traits** of Git **repository** implementation
  + Tree structure
  + Pointer reference
  + Stage
  + SHA1 hash
* Why not use **index** to mark submitting in Git?

Submitting is not in sequence in distributed revision control system, only universal unique number can identify submitting occurred at any time in anywhere.

* **Visual user tool** for Git
  + Why not use gitk, gitg, qgit? – They requires Linux or QT environment.
  + Why choose TortoiseGit? – Familiar to SVN users; free SW in windows; easy to port SVN repository to Git.
  + Install **TortoiseGit**
    - Read and follow setup instruction in <http://code.google.com/p/tortoisegit/wiki/SetupHowTo>, prerequisites: Windows Installer and Git for Windows. Check if the computer already installed Windows Installer: run->command window->msiexec.
    - Download installation package in <http://code.google.com/p/tortoisegit/wiki/Download> and execute it.
    - Refer to instruction in Youdao notes.
  + Install **gitk**
    - Much easier way is installing it in the Cygwin installation, just tick the “tcl –tk: Tcl X11 toolkit” in the package list.
    - The other way is downloading ActiveTcl Community Edition at <http://www.activestate.com/downloads#4> which is for windows and free.
    - Run the .exe to perform installation.
    - An error during installation, not sure to what extent it would affect the function, note down it here for future reference:



* Troubleshooting of running **gitk**
  + Run “$ gitk –all” in Cygwin, below error is displayed:



To solve this issue, install “XInit” in Cygwin installation by selecting the “XInit” package. Refer to Youdao notes for details.

* **Protocols** supported in Git and the corresponding URL. SSH, GIT and local are intelligent protocols.



* A bare repository will be initiated under **Cygwin root drive** “C:\cygwin64” if only relative path is specified.



* **Ssh: could not resolve host name**



The issue occurred when push updates to remote repository. Reason is Cygwin recognises only paths in Cygwin format, the path of the remote repository was <file:///path/to/repo/shared.git>, later it was changed to C:\cygwin64\path\to\repo\shared.git when I clone repository for user2 in TortoiseGit.

To solve the issue, just specify the path in Cygwin:

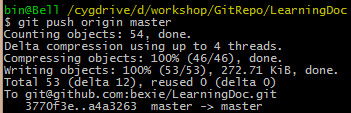


* Disable non-fast-forward: $ git –git-dir=/path/to/repo/shared.git config \ receive.denyNonFastForwards true (failed in trial)
* Source control server management tool **Gitolite**. – to be studied later.
* Create unique entry for users to visit repository server
* Manage users’ access rights to repositories or branches
* Gerrit – code review server constructed on top of source control server
* Download the .war file at <http://gerrit-releases.storage.googleapis.com/index.html>, then unzip it in winrar.
* More info refer to **Chapter 32**.
* **Host** Git project in an external dedicated hosting site
  + Quick to set up and easy to start project on
  + No server maintenance or monitoring
  + Best choice for open source project as people can find it easily
  + Most popular one **GitHub** – support open source projects and also private project
    - <https://github.com>, bexie
    - **Deploy key/cert** in GitHub as soon as account created, only then you can push local repository to GitHub. Refer to next subject of how to generate key/cert. open “settings” -> “deploy keys” -> “add deploy key”, paste the content of C:\cygwin64\home\bin\.ssh\id\_rsa.pub into the editor then save. Then deploying public key in a host server is finished. Note: different repository maintains its own public key list in GitHub, remember to deploy key for new repository, the existing key can be used for multiple repositories.
    - To change the URL format of the remote repository

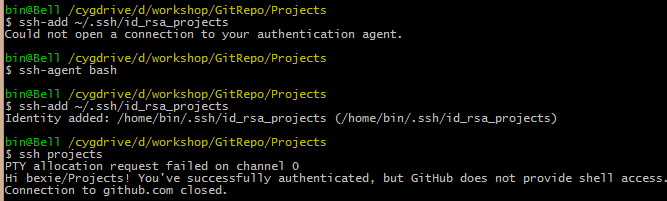


* + - **Push local repository to GitHub**, remember to pull first if there is something in server to be merged.

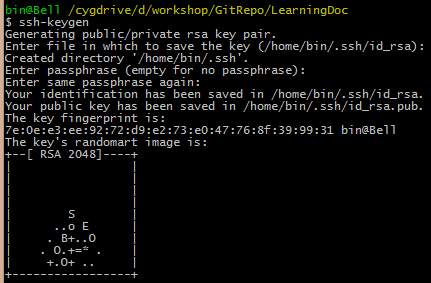
$git add origin git@github.com:bexie/LearningDoc.git



* + - **Manage multiple keys** in client and remote server. Generate new private/public key pair. Add public key into corresponding GitHub repository. Add the new key into SSH agent otherwise it displayed error message when trying to push anything to server.

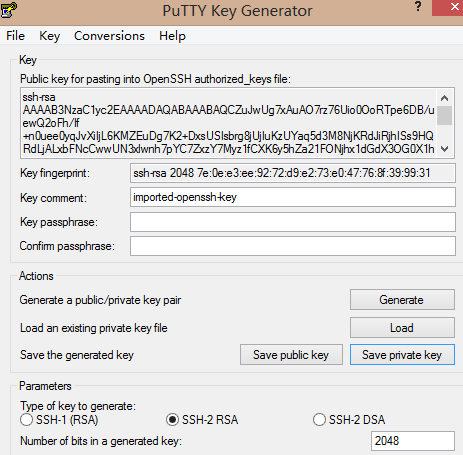


* Set up **key/certificate** for SSH/HTTP connection to remote Git server
  + **Public/private key pair** for SSH protocol
    - Check if folder “.ssh” exist under C:\cygwin64\home\bin, if not, then go to next step.
    - Generate public/private key pair by executing “$ ssh-keygen” in Cygwin console:

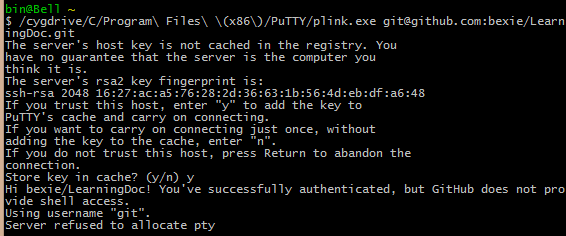


Or directly use PuTTY keygen to generate public/private key pair.

* + - Set PuTTY as SSH client in Git by “export GIT\_SSH=/cygdrive/C/Program\ Files\ \(x86\)/PuTTY/plink.exe”.
    - Launch PuTTY, load the private key generated above and save it as .ppk key.



* + - Launch Pageant from PuTTY folder, execute context menu “add key” in Pageant, load the .ppk private key we saved in above step. This step is to be performed whenever SSH connection is required after restarted computer.
    - To **establish the trust of public key** deployed in remote server, execute below command, it is required just once:



* **Certificate** for HTTP protocol – to be studied
* Source control tool cooperates with requirement and bug track tool? Redmine?

**Chapter 5/6/7/8/9/10 to be read thoroughly to understand details of branches.**

**Chapter 7 – Git Reset -- unread.**

**Chapter 8 – Git Checkout – brief read.**

**Chapter 9 – Resume progress – brief read.**

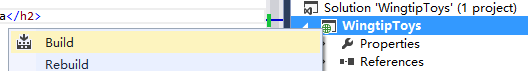
**.NET**

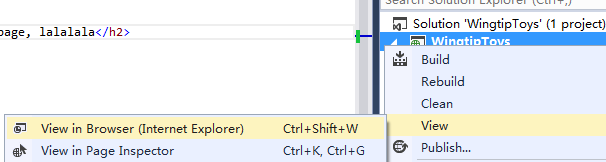
* How to get a sample project for learning?

Open the “new project” dialog from menu, there is a link to get samples online.



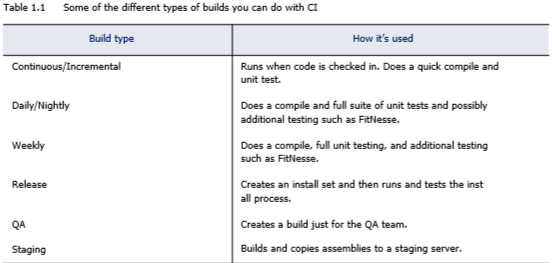
* Common and most easiest operation on a sample project are build and view:





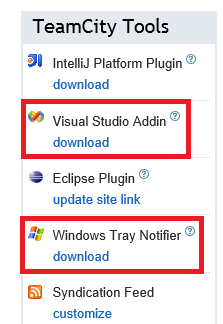
**CI tool – TeamCity**

* **Questions to companies:**
  + What is the CI process? Are there daily build? Weekly build? Weekly release?
* What does CI do in general?
* Automated process that builds, tests, analyzes and deploys an application to help ensure it functions correctly, follows the best practices and is deployable
* Multiple integrations per day
* The centerpiece of software development
* The work in a CI server can be triggered by changes in source control server after developer committed code changes, or be triggered at certain intervals or events
* **Builds** supported in CI system

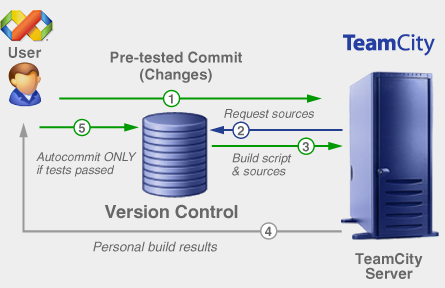


Continuous build is most important one and is triggered when code is checked in to repository.

* Reasons to use CI in development
  + Reduced risks
  + Deployable software
  + Increased Project visibility
  + Fast incremental builds
* **Essential tools** in CI process
  + Source control tool (SVN, Git) – studied and noted in above pages
  + Continuous integration server (TFS, Hudson, TeamCity) – driven by configuration file
  + Feedback mechanism – common method is through a website
  + Build manager (MSBuild, NAnt) – do actual build and controlled by XML configuration file, there is GUI tool to make the maintenance of the configuration file easier
  + Unit test framework (MSTest, NUnit) – run the unit test and generate result to text/XML file which CI server can fetch then display in feedback mechanism
* **Desired tools** in CI process
  + Code analysis (FxCop, StyleCop, NCover) – generate analysis report that can be fetched by CI server and then be integrated with build report
  + Other testing tool (Seleneium, FitNesse)
* **Pre-tested commit**
  + To ensure the code is **pre-tested before checking in to remote repository**, we install a visual studio plugin in TeamCity. Go to “Bella” -> “TeamCity Tools” in the right, and install these two:



* Visual studio express edition does not support TeamCity addins, replaced it with visual studio 2012 ultimate. Key: YKCW6-BPFPF-BT8C9-7DCTH-QXGWC
* To enable the addin, we must connect it with the VCS so that it can perform committing to the remote repository when the new code built and tested successfully.
* The workflow of the addin:



* Encountered connection problem when login the TeamCity addin in VS as the server URL <http://localhost:89> was not recognized, just change the host to local ip <http://192.168.1.7:89>.
* Git is not supported in this addin by default, so give up this trial.

