# GIT environment setup

Background: GIT is a version control system used for Source Code Management (SCM) to keep track of changes in any set of files.

1. Install Node.js: <https://nodejs.org/en/download/>, say in C:\ProgramFiles\Nodejs\

In CMD, enter ‘node –v’; ‘npm –version’ to confirm version installed.

* 1. Download Visual Studio code: <https://code.visualstudio.com/>

2. Install Git: <https://git-scm.com/>, click on Download link. E.g. in same location C:\ProgramFiles\Git\

Ref: <https://www.youtube.com/watch?v=TJgRRwSkGHM>

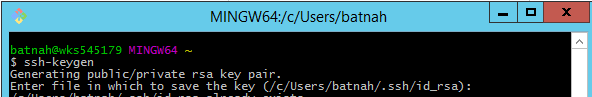
In CMD, enter ‘git –version’ to confirm version installed.

2.1 Configure Git to access Repository.

- Generate Public and private keys

Start->All Programs->Git->Git Bash.

Type ‘$ ssh-keygen’ and hit enter.



Optional if you are specific in creating folder and password. Else, hit enter to use default location e.g: ‘c/users/batna/.ssh/id\_rsa’.

Optional if you require password. Recommend no password. Hit enter.

At this stage, public and private keys are generated in: C:\Users\batnah\.ssh folder.

i.e id\_rsa, id\_rsa\_pub

Edit ‘id\_rsa\_pub’ file to contain valid email and save. Send this file to GIT admin to grant access to repository (i.e. This public Git key will be installed on the GIT server).

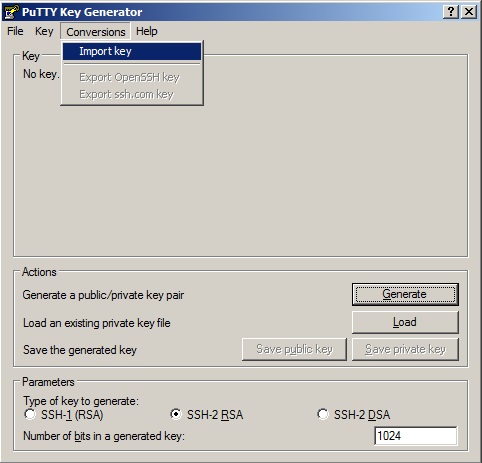
3. Download and install TortoiseGIT (<https://tortoisegit.org/download/>)

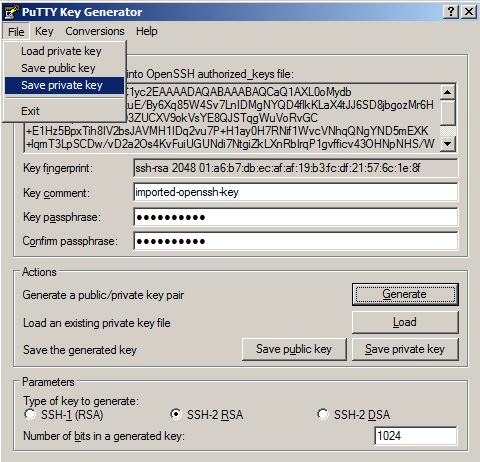
3.1 Configure TortoiseGIT (for both *pull,* *push and Merge* operations):

Prerequisites:

* Git is installed and configured with the appropriate key files
* The public Git key has been installed on the GIT server
* PuTTY is installed with the OpenSSH option

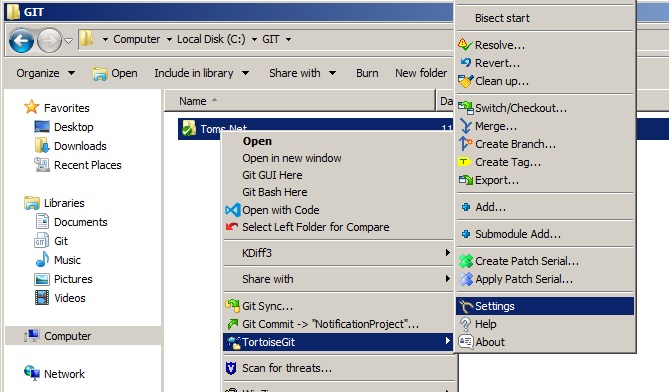
3.1.1 Generate PuTTY Private Key File: Run Start -> All Programs -> TortoiseGit -> PuTTYgen

* Click Import key from the Conversions menu: [](http://scdcdvbsd01/tomsdevwiki/File:PuTTYgen1.jpg)
* Navigate to the folder where GIT saves the generated key files (typically C:\Users\<login>\.ssh)
* Open the id\_rsa file
* (if) Enter the passphrase for the RSA private key file
* Click Save private key from the File menu:

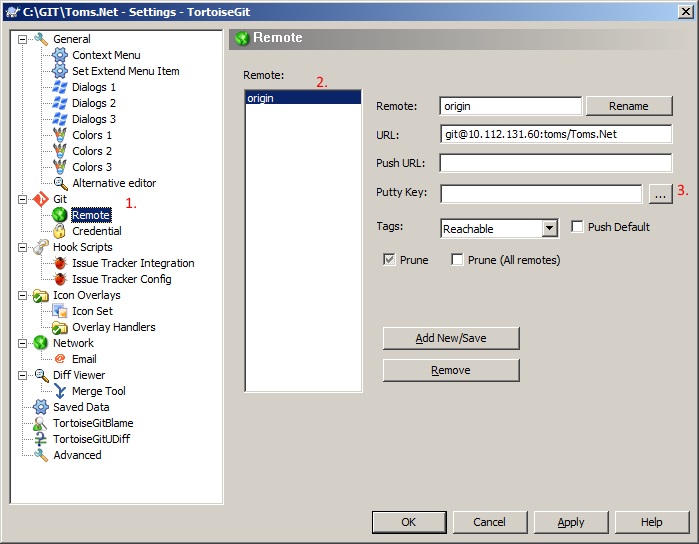
[](http://scdcdvbsd01/tomsdevwiki/File:2PuTTYgen.jpg)

* Give the private key a filename e.g ‘private.ppk’ and the recommended folder to save it in is the same folder as the .ssh folder created by Git i.e. in ‘c/users/batna/.ssh’.

3.1.2 Configure GitTortoise.

Right-click on *CBIS>*  Click on *TortoiseGit*-> Settings[](http://scdcdvbsd01/tomsdevwiki/File:ExplorerMenu.jpg)

Click on *Remote*on the left pane (1.)

[](http://scdcdvbsd01/tomsdevwiki/File:TortoiseGitSettings.jpg)

Click on the origin that you want to configure under the *Remote:* list box (2.)

Click on the ellipses to the right of the input box for *Putty Key*

Select the private key file (.ppk) that was created in the previous steps

Click *OK*to save and close

4. Ensure that above directories in 1, 2 and 3 are available as ‘Environment variables’ on your PC.

(If not Add them to ‘User variables’ or System variables where relevant)

To do, navigate rtClick PC n select Properties> Advanced system settings> Environment Variables.

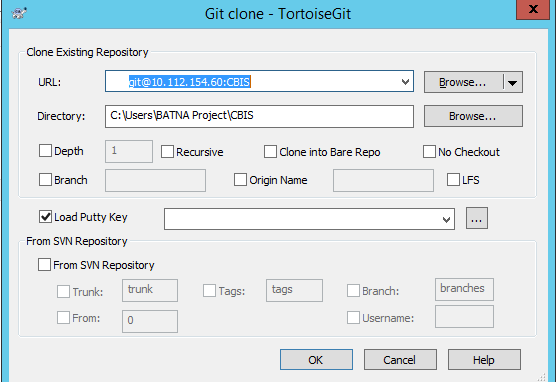
C:\Program Files\Git\bin

C:\Program Files\TortoiseGit\bin

5. Create Project folder and clone repository into it.

Create a ‘Project e.g ABCD’ folder e.g in C:\Users\Batna\Source\Repos\

Rightclick on ABCD> click ‘Git Clone..’



Enter:

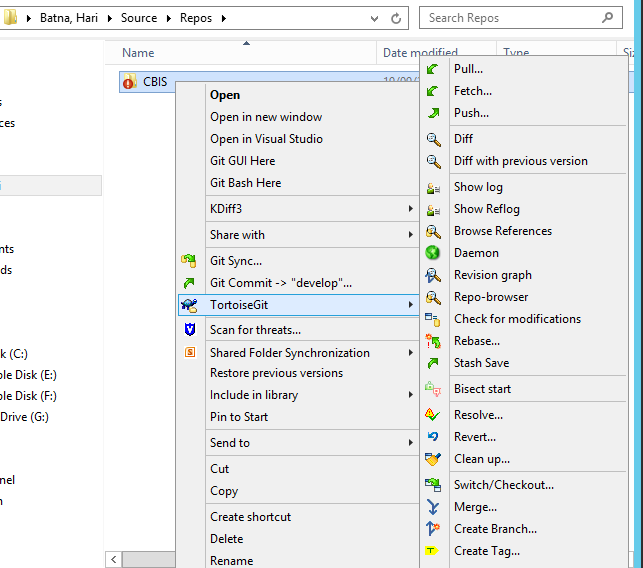
- URl of the GIT Server i.e Source code from cloud ’10.112.154.60:ABCD;

- Load Putty Key in Step 3. Click OK.

This will clone central Repository in the Project folder you created.

Now this will your local copy of the repository where you can work on scripts.

You perform Pull, Push, merge on this folder’s right click options:



Ref: <https://tortoisegit.org/docs/tortoisegit/>

6. Repeat Step 4. i.e Ensure that above directories in 5 are available in ‘Environment variables’:

C:\Users\Batna\Source\Repos\CBIS\protractor\node\_modules\.bin;

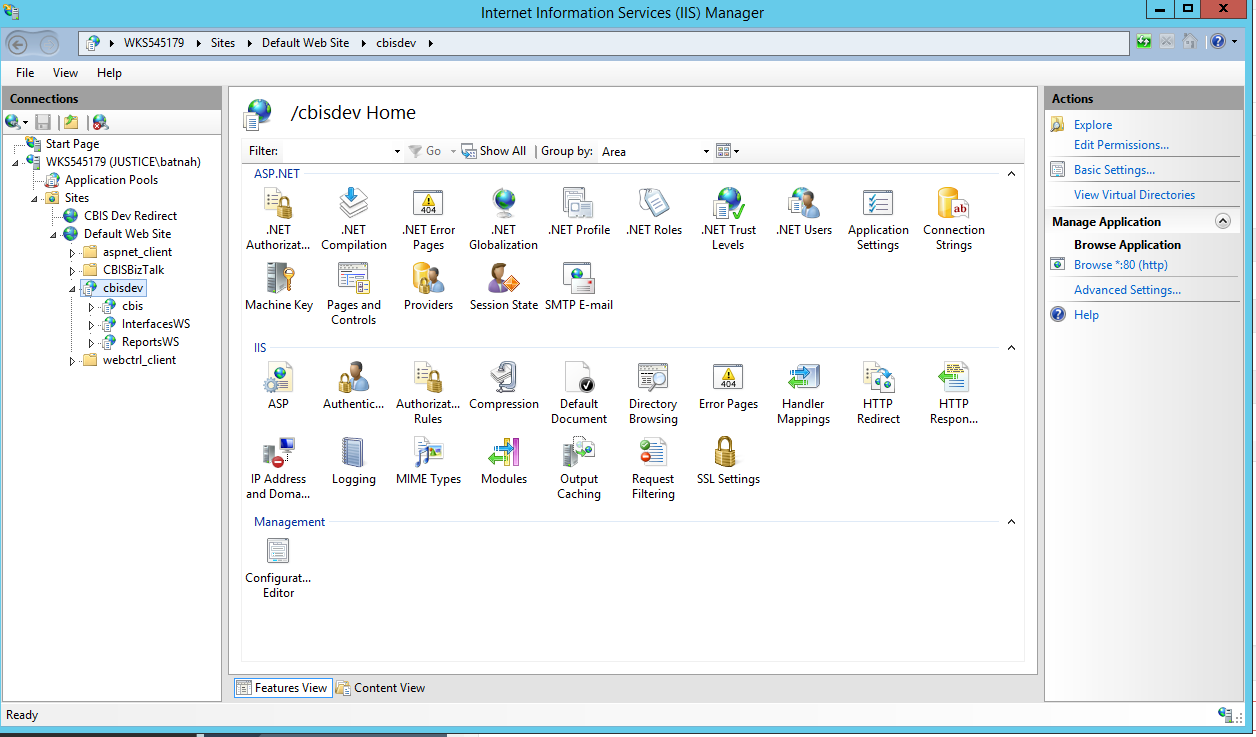
C:\Users\Batna\Source\Repos\CBIS\node

6.1 To confirm that all paths are configured correctly

Run ‘Git CMD’: Start->All Programs->Git->Git CMD.

Enter ECHO %PATH% and hit enter.

7. Web service and its components: Ref to URL in Step 5



# Install Protractor and configure to above paths

Background: Protractor is test automated tool for AngularJS application.

1. Install Node.js: <https://nodejs.org/en/download/>, say in C:\ProgramFiles\Nodejs\

In CMD, enter: ‘node –v’; ‘npm –version’ to confirm version installed.

e.g v8.11.3; 5.6.0 respectively.

2. Install Protractor. This would install both Protractor and Selinium webdriver.

In CMD, enter: npm install –g protractor

enter ‘protractor --version’ to confirm version installed

e.g Version 5.4.1

3. Update selenium webdriver.

In CMD, enter: webdriver-manager update

4. (Optional) Install Jasmin.

In CMD, enter: npm install –g jasmine

5. Create or update conf.js and spec.js file.

- In confi.js file, ensure that spec.js file path is specified.

e.g specs: ['./\*\*/\*.spec.js'] // this indicated all spec.js files will be executed.

- In spec.js file, ensure that test scripts exists.

Note the above file path.

# Start and run tests

1. Start Selinium webdriver

In CMD, enter: webdriver-manager start

Note: Do not close this CommandLine window until test completion. Closing will terminate Protractor

2. Start Protractor. Do this in another CommandLine window

In CMD, enter: C:\Users\batnah> path from above step 5. E.g:

protractor C:\Users\batnah\Source\Repos\CBIS\Application\Web\cbis\App\_Test\E2E\protractor.conf.js

# Example files attached

GIT Bash commands

|  |  |
| --- | --- |
| $ git branch | Lists all branch created |
| $ git checkout –b “BranchName” | Will create a new branch called “BranchName” |
| $ git add. | Add files to the current branch |
| $ git status | Provides summary of which files have changes that are staged for next commit |
| $ git commit –m “BranchName message” | Shows the changes that were committed from staging. –m is to commit with a message. |
| $ git push origin “BranchName” | Push the committed BranchName to mainstream “origin’ |
| $ git pull origin “BranchName” | Pull mainstream ‘origin’ to ‘BranchName’ |

Open Git-Bash:

