

VNPAY GIT SEMINAR

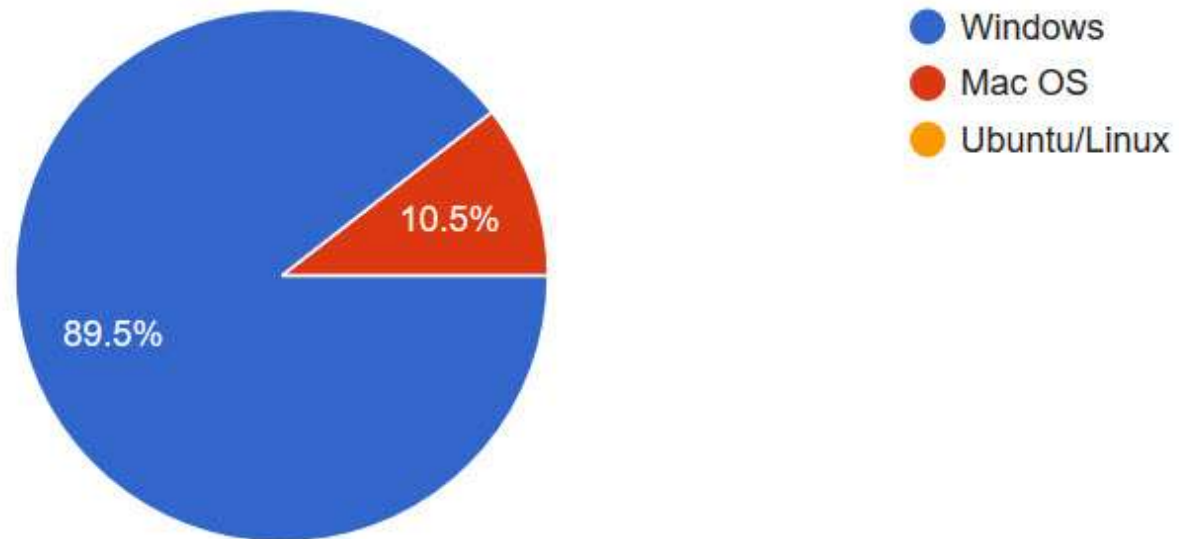


Trình bày: xonv@vnpay.vn

Hà Nội, 2016/04/09

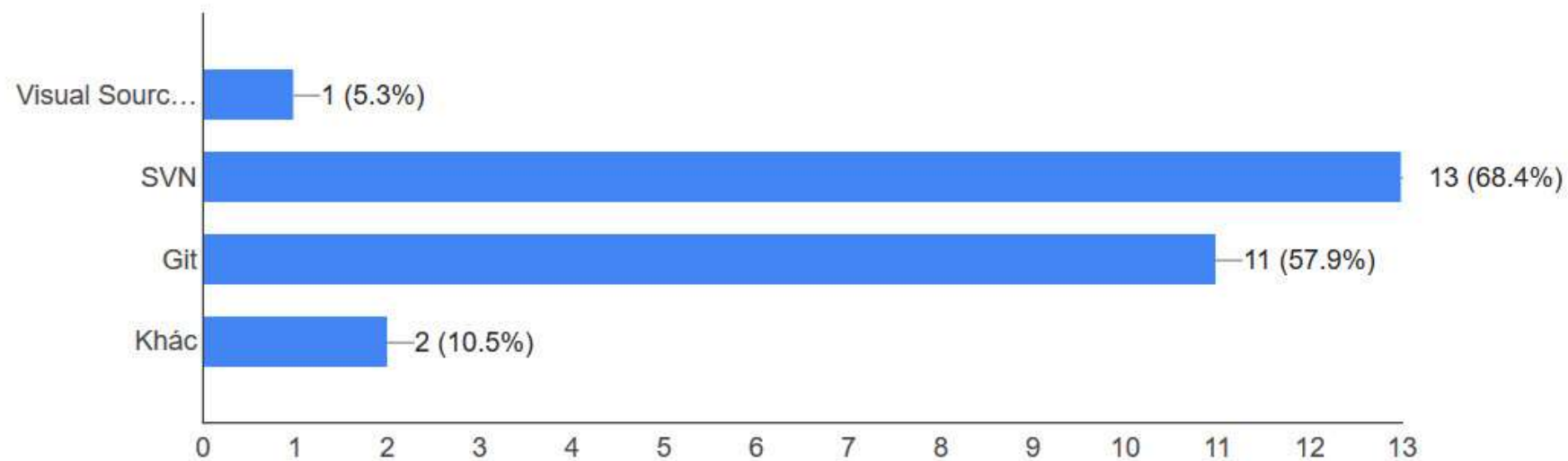
Thống kê nhanh

Bạn sử dụng hệ điều hành nào? (19 responses)



Thống kê nhanh

Bạn đang dùng Source control nào? (19 responses)



Where is git team?

Bạn muốn đạt được điều gì sau buổi Seminar? (16 responses)

Sử dụng được và các bước cần thiết để merge dữ liệu từ các nhánh với nhau

Tìm hiểu thêm cách quản trị GIT thay vì chỉ Push, Pull, Clone

Quản lý công việc và tài liệu được đồng bộ, dễ dàng hơn

Hiểu rõ hơn về Git.

Hiểu biết thêm về GIT, có thể sử dụng GIT

Được giải đáp phần nào về kết hợp giữa công cụ và quy trình phát triển, triển khai, vận hành; thay vì chỉ tập trung vào sử dụng công cụ quản lý phiên bản nào.

quản lý phiên bản tài liệu đơn giản và nhanh chóng

Git khác svn ntn

Tôi muốn đoạt được cái USB 3.0 32Gb

Lí do nên dùng, cách thức sử dụng

Hiểu hơn về Git

Mình muốn biết về cách sử dụng của git, cái lợi và hại của git.

Hiểu rõ được khái niệm GIT là gì và có thể sử dụng được GIT để quản lý Source Code

hiểu chi tiết hơn về GIT

USB 32 GB

Tại sao tôi chọn GIT



Nội dung

1. Giới thiệu về GIT
2. So sánh GIT và Source control khác (Visual SourceSafe, SVN)
3. Tổ chức file trong GIT
4. Sử dụng GIT
5. Các tình huống thường gặp
6. Hỏi đáp

Giới thiệu về GIT

Git được ra đời năm 2005, xuất phát từ nhu cầu của Cộng đồng cũng như những người phát triển hệ điều hành Linux. Git được ra đời với mục tiêu được đặt ra như sau:

- ✓ Nhanh
- ✓ Thiết kế đơn giản
- ✓ Hỗ trợ tốt cho "phát triển phi tuyến tính" (non-linear development)
- ✓ Phân tán toàn diện
- ✓ Có khả năng xử lý các dự án lớn giống như nhân Linux một cách hiệu quả (về mặt tốc độ và khối lượng dữ liệu)

So sánh GIT và Source control khác

Hiện nay chúng ta đang sử dụng phổ biến các công cụ: Visual SourceSafe (Microsoft), Vault (Source gear), SVN và một số công cụ hỗ trợ khác. Đa số các công cụ này đang vướng phải vấn đề:

- ✓ Không hỗ trợ được các IDE đang sử dụng (trừ SVN)
- ✓ Tốc độ chậm hoặc phải đòi hỏi kết nối tới server khi code
- ✓ Khó khăn trong việc quản lý phiên bản phát triển
- ✓ Trong trường hợp server bị sự cố, dự án có thể bị mất lịch sử hoặc sẽ rất khó để khôi phục lịch sử

So sánh GIT và Source control khác

Git đáp ứng tốt các yêu cầu này, ngoài ra Git còn hỗ trợ những tính năng nổi trội khác như:

- ✓ Hỗ trợ tương tác với Git Server thông qua HTTPs/SSH
- ✓ Git Server hỗ trợ quản lý project và người dùng linh hoạt

So sánh GIT vs SVN

- Đánh giá theo kinh nghiệm của bản thân
- Đánh giá theo Google.com

Đánh giá theo kinh nghiệm của bản thân

- Tốc độ
- Khả năng quản lý phiên bản
- Bảo mật (troll)
- Dễ sử dụng

Đánh giá theo Google.com

- Google.com => hỏi Google

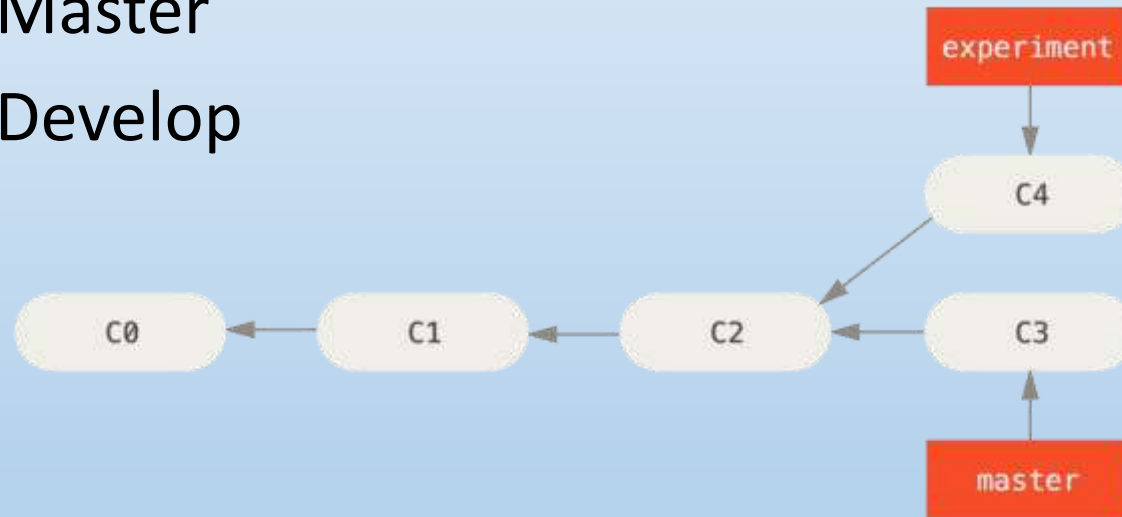
Một số khái niệm trong git

- Branch (Phân nhánh)
- Merge
- Rebase

Branch

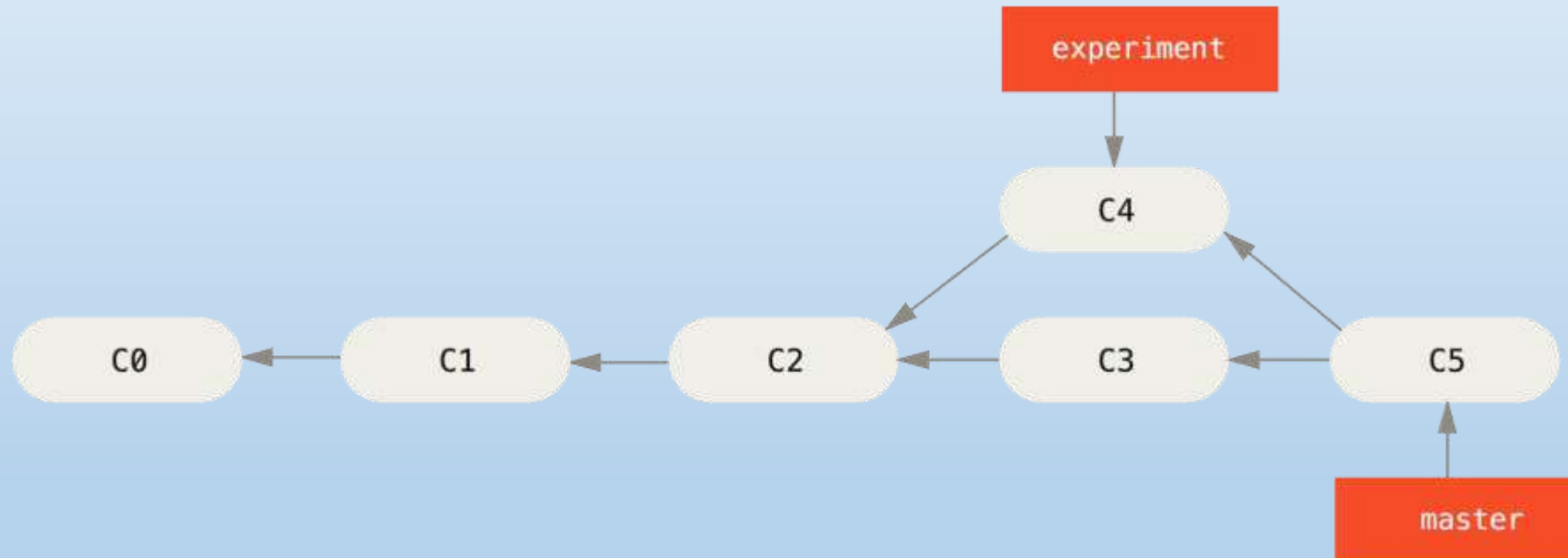
Phân nhánh có nghĩa là bạn phân tách ra từ luồng phát triển chính và tiếp tục làm việc mà không sợ làm ảnh hưởng đến luồng chính.

- ✓ Nhánh Master
- ✓ Nhánh Develop

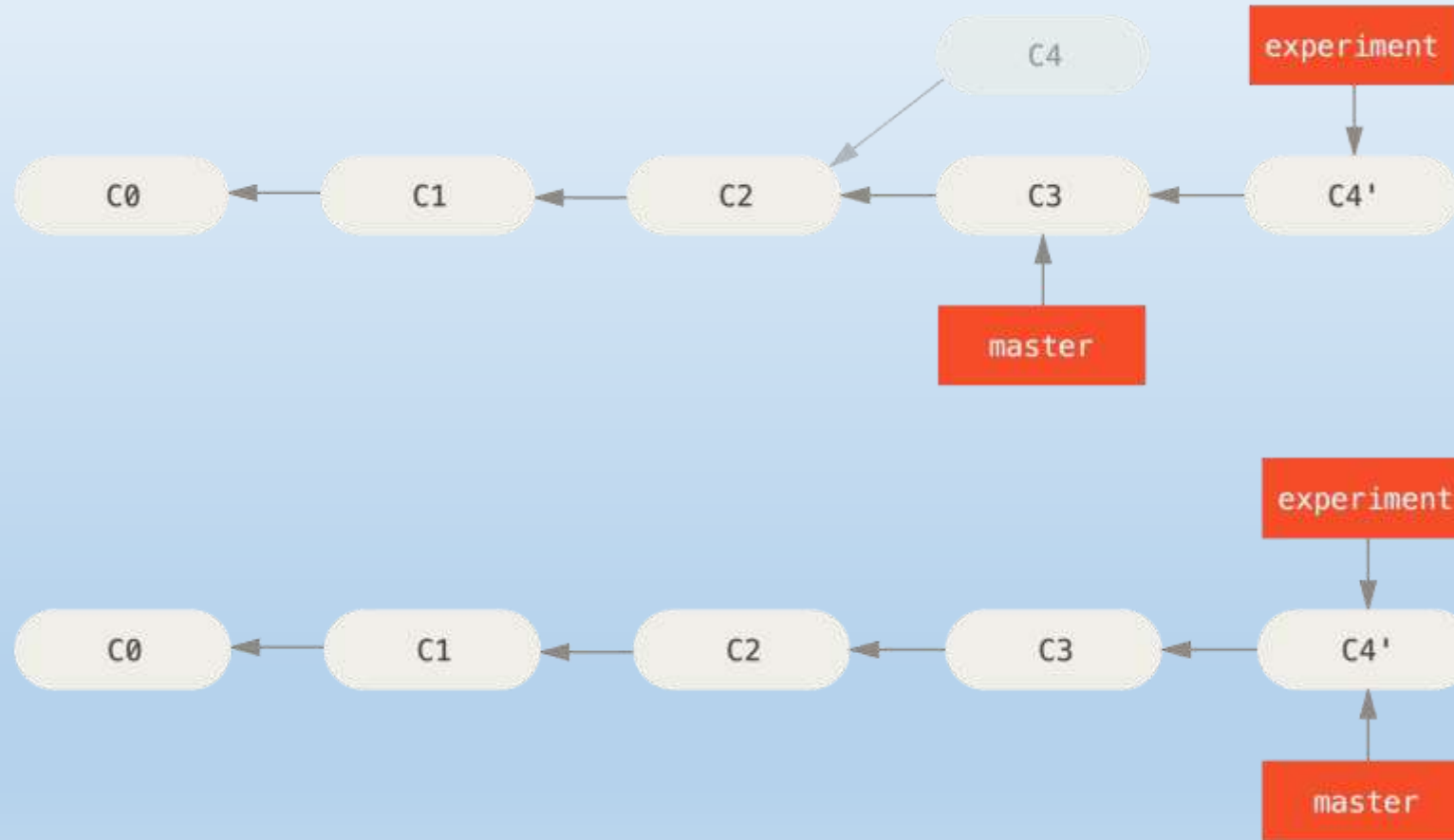


Merge

- Merge được tích hợp thay đổi giữa các nhánh.



Rebase

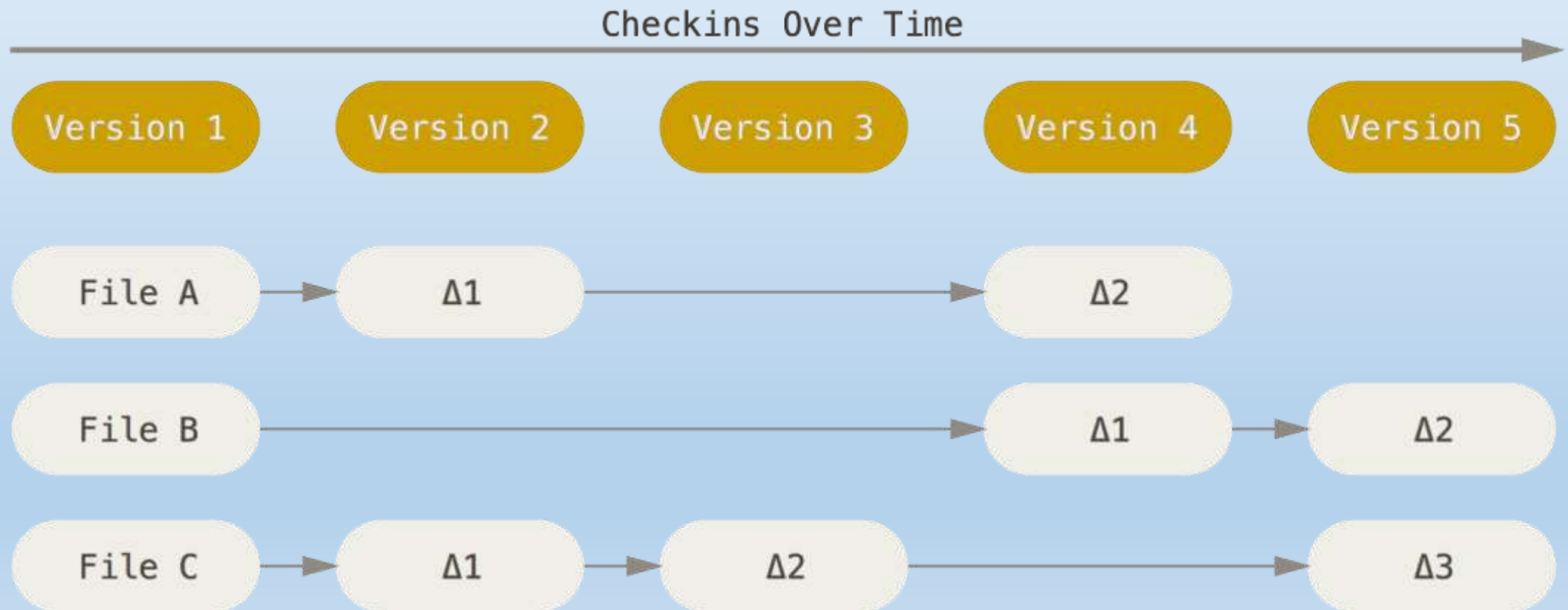


Tổ chức file trong GIT

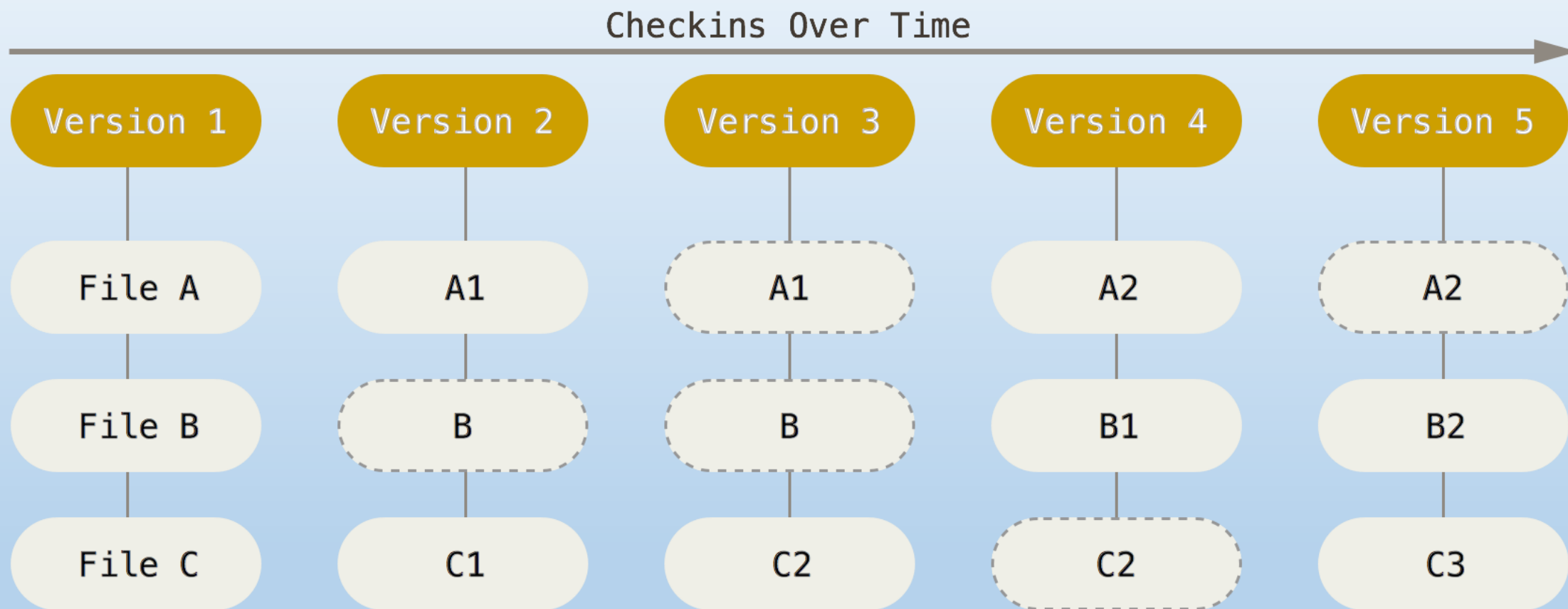
1. Snapshots, Not Differences
2. Phần lớn các thao tác diễn ra tại máy local
3. Git mang tính toàn vẹn
4. 03 trạng thái của file

1. Snapshots, Not Differences

- ✓ Không lưu nếu file không có sự thay đổi
- ✓ Khi file có thay đổi, git sẽ cập nhật thêm mới trạng thái vào CSDL git
- ✓ Chi tiết của dự án theo từng lần commit



Snapshots. Chi tiết của dự án theo từng lần commit



2. Phần lớn các thao tác diễn ra tại máy local

- Việc tương tác với project chủ yếu diễn ra trên local. Không cần phải có kết nối tới máy chủ vẫn làm việc được
- Tương tác với máy chủ thông qua lệnh PUSH, PULL
- Bản thân máy local cũng như một Repository. Khi cần thiết có thể push code lên một server khác mà vẫn giữ nguyên được lịch sử của project.

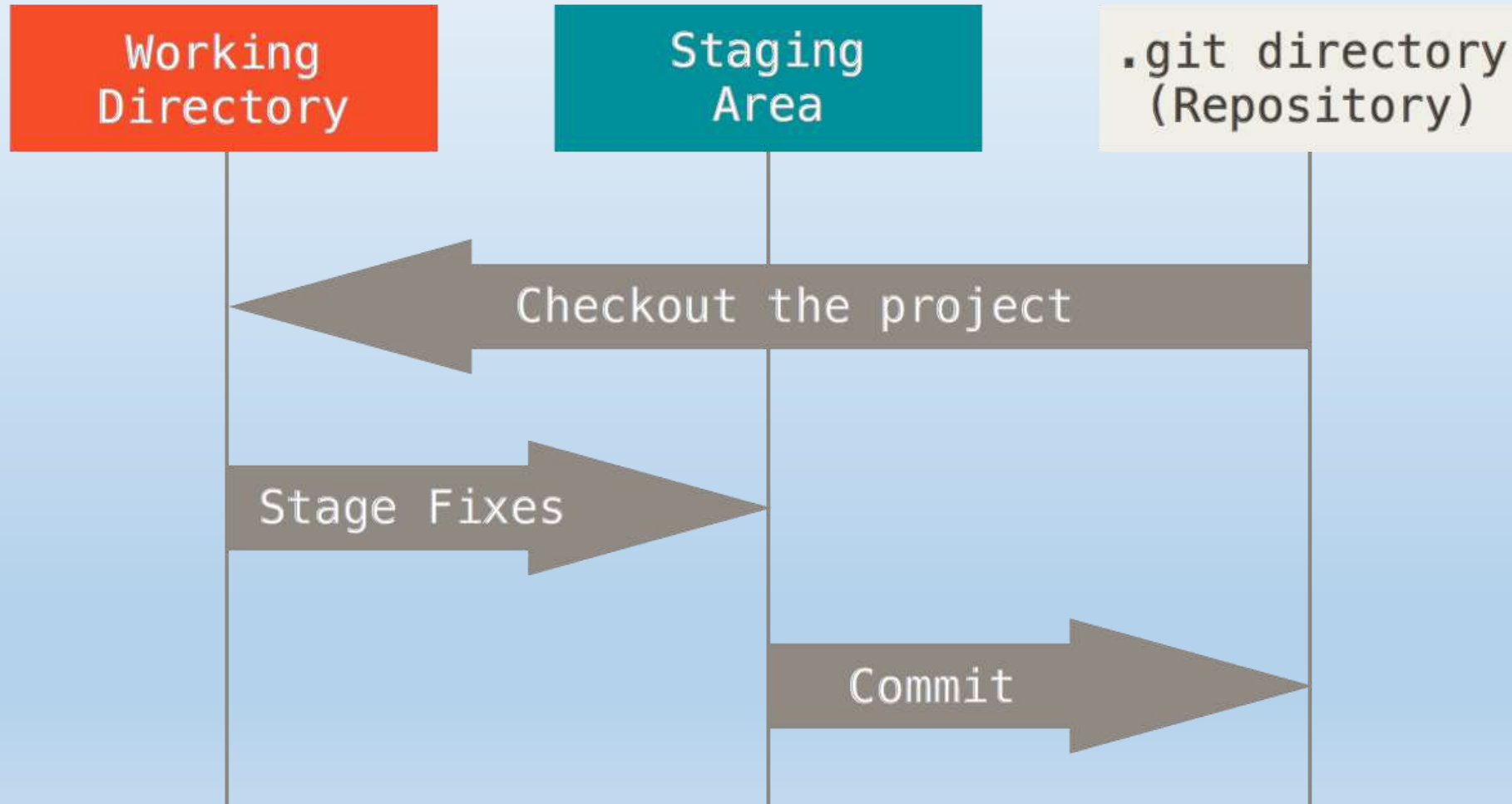
3. Git mang tính toàn vẹn

- Mọi thứ trong Git được "băm" (checksum or hash) trước khi lưu trữ và được tham chiếu tới bằng mã băm đó.
- Việc thay đổi nội dung của một tập tin hay một thư mục sẽ được git phát hiện
- Bạn không thể mất thông tin/dữ liệu trong khi truyền tải hoặc nhận về một tập tin bị hỏng (mà Git không phát hiện ra).

03 trạng thái của file

- Committed: Nội dung thay đổi đã được lưu trong CSDL git
- Modified: File đã bị thay đổi nội dung
- Staged: File được đánh dấu để Commit

03 trạng thái của file



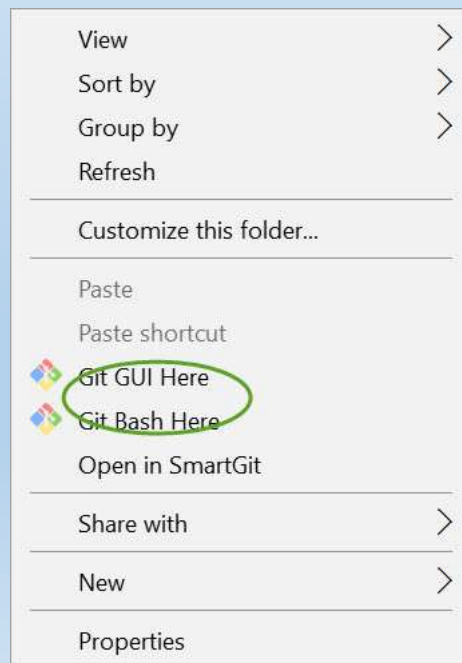
Sử dụng GIT

- Cài đặt GIT
- Những lệnh phổ biến (hay sử dụng)
- Hướng dẫn sử dụng SmartGit

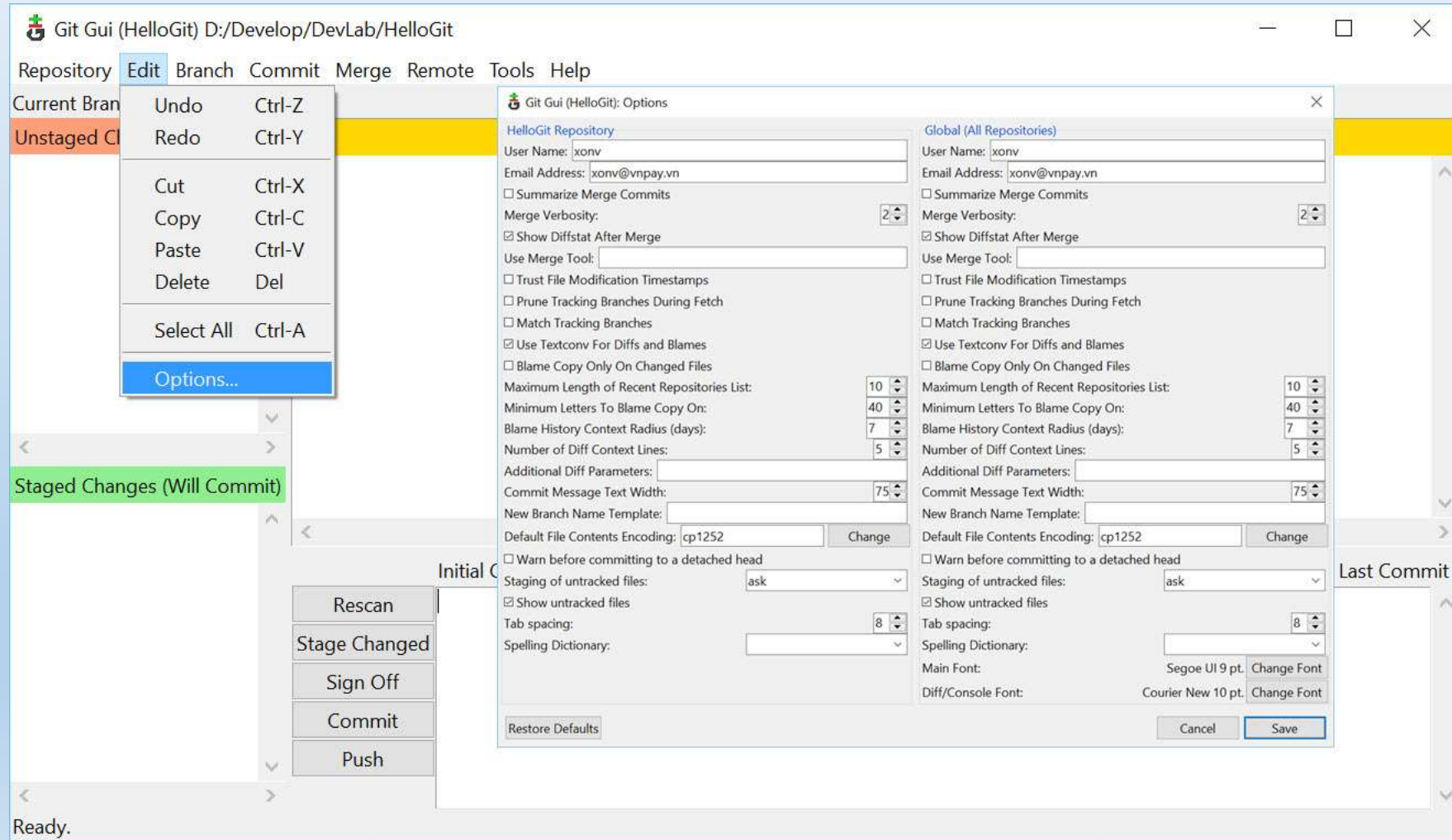
Cài đặt GIT

- Ubuntu/MacOS => Google
- Windows:

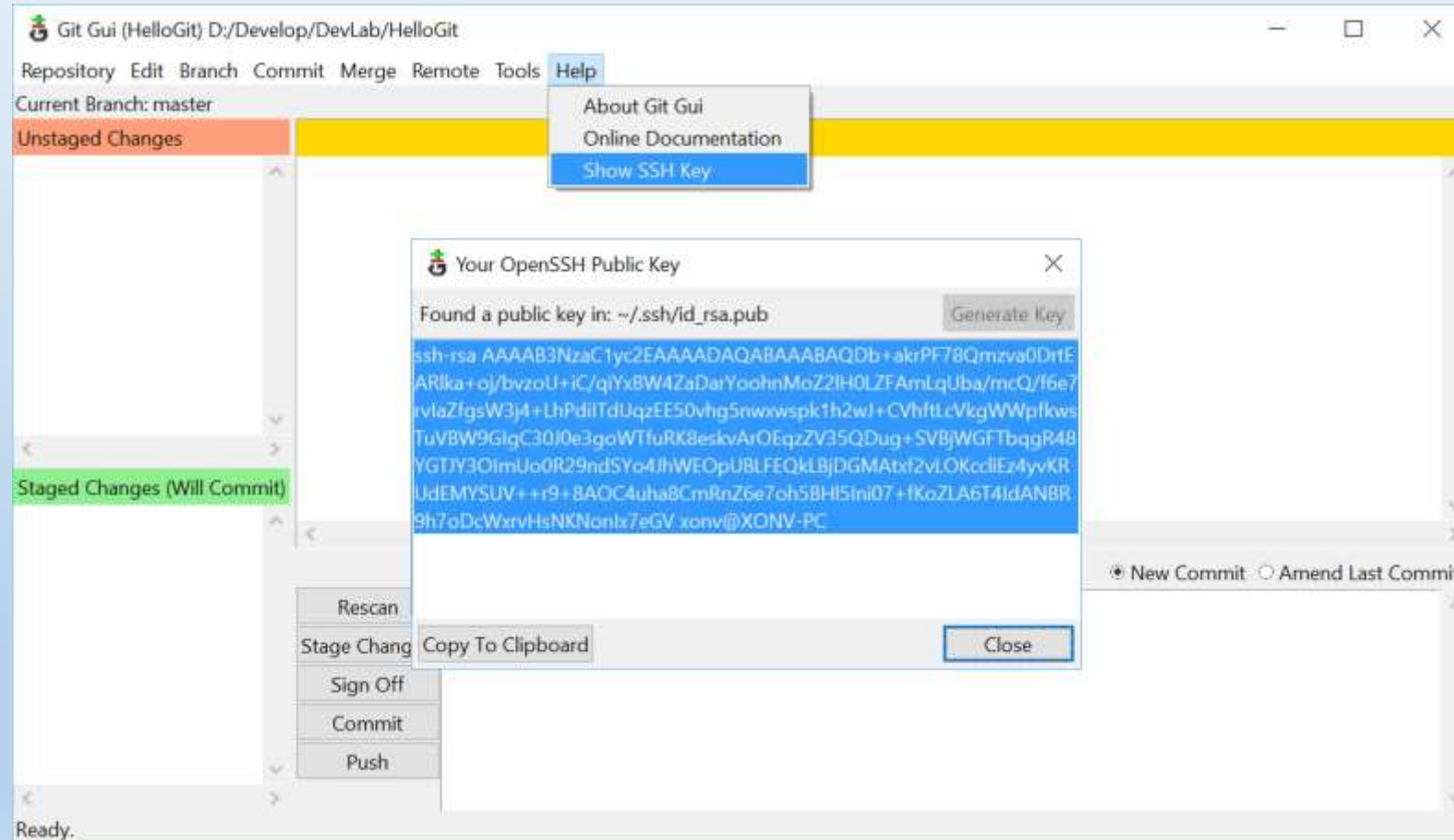
Vào trang: <https://git-scm.com/> để tải bộ cài đặt và làm theo hướng dẫn.



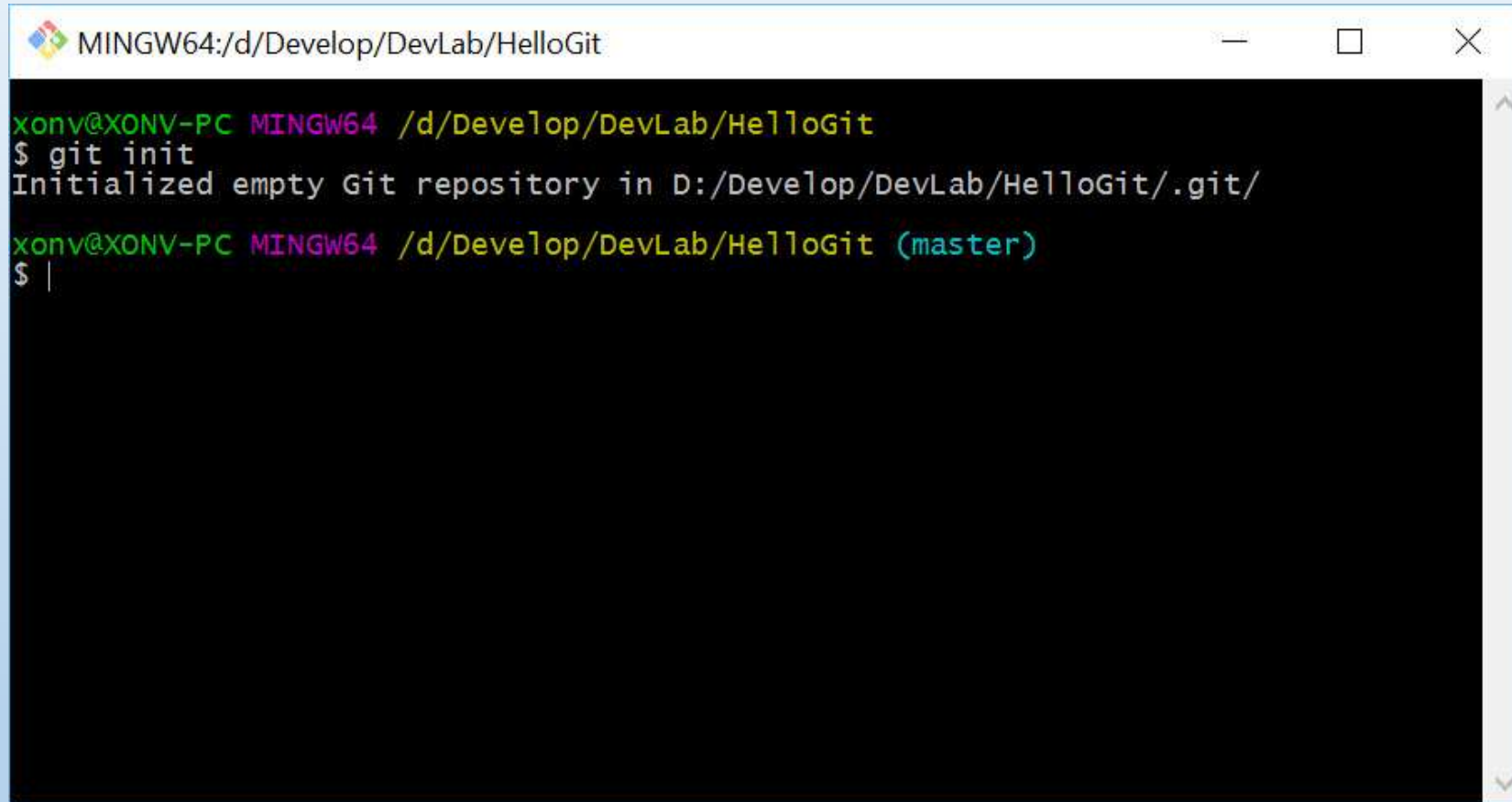
Git GUI – Cấu hình user/email



Git GUI – Cấu hình SSH-Key



Git Bash



```
MINGW64:/d/Develop/DevLab/HelloGit
xonv@XONV-PC MINGW64 /d/Develop/DevLab/HelloGit
$ git init
Initialized empty Git repository in D:/Develop/DevLab/HelloGit/.git/
xonv@XONV-PC MINGW64 /d/Develop/DevLab/HelloGit (master)
$ |
```

Những lệnh phổ biến (hay dùng)

git init	git add	git checkout
git clone	git commit	git merge
git push	git status	git rebase
git pull	git log	git tag
		git branch

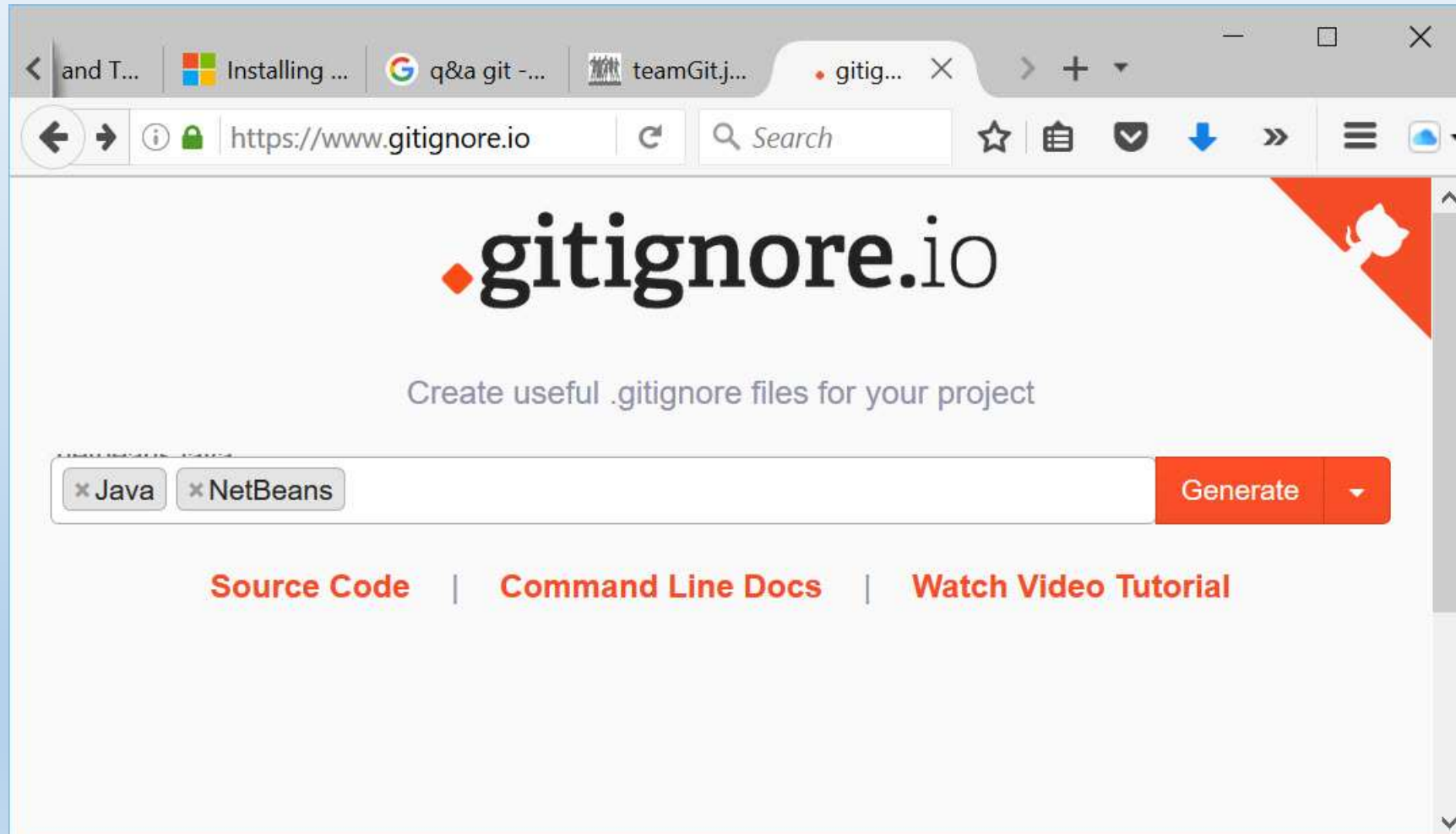
Chú ý **.gitignore**

.gitignore để báo cho git bỏ qua các file được chỉ định, không cập nhật nó lên server, thường là các file do IDE sinh ra khi chạy chương trình.

Tạo file .gitignore:

- Cách 1: <https://www.gitignore.io/>

Giao diện <https://www.gitignore.io/>



File .gitignore cho NetBeans + Java

```
# Created by https://www.gitignore.io/api/netbeans,java

### NetBeans ###
nbproject/private/
build/
nbbuild/
dist/
nbdist/
nbactions.xml
.nb-gradle/

### Java ###
*.class

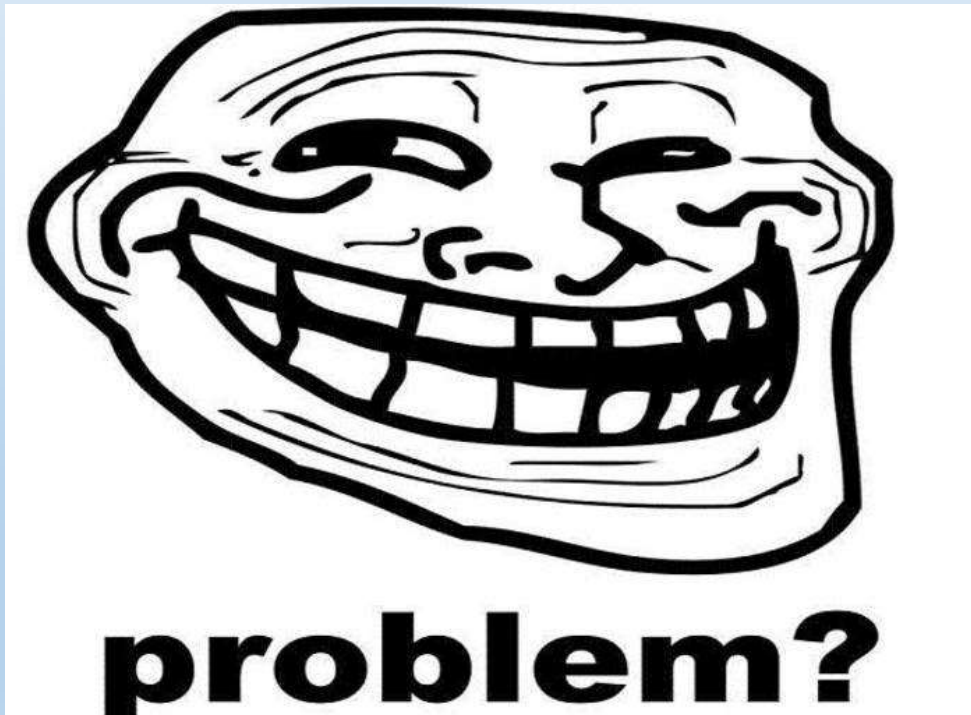
# Mobile Tools for Java (J2ME)
.mtj.tmp/

# Package Files #
*.jar
*.war
*.ear

# virtual machine crash logs, see http://www.java.com/en/download/help/error\_hotspot.xml
hs_err_pid*
```

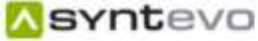

Cách 2

- Copy file từ cách 1 (file đã có ở project khác)



Giới thiệu Smartgit


(Danh sách git client: <https://git-scm.com/download/gui/win>)









SmartGitSmartCVSSmartSynchronizeContactBlog

DownloadPurchaseWhat's NewDocumentationDistributed ReviewsGitHubSVN Bridge

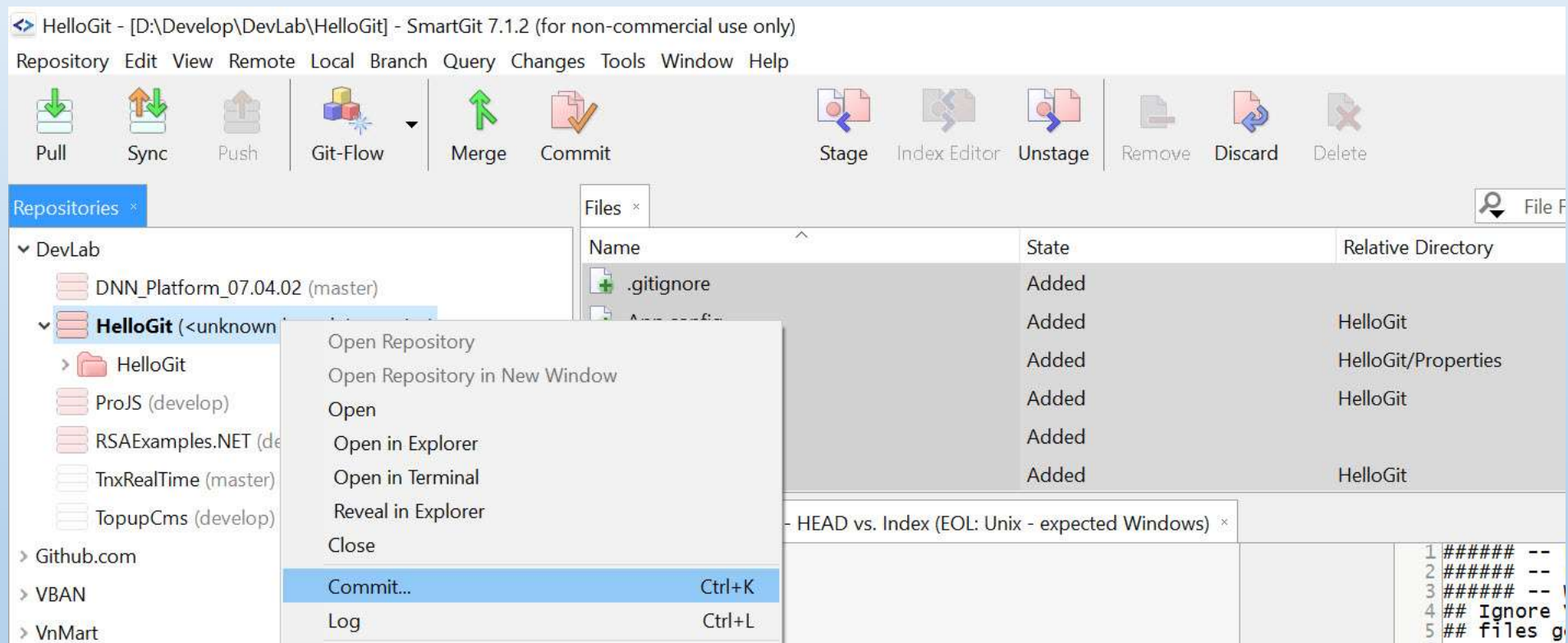
Download SmartGit 7.1.2

 **What's New**
Detailed change log

Other versions:
[→ Next Version Preview](#) [Archive](#)

 Windows 7+	 Mac OS 10.9 - 10.11	 Linux
<div> Download Installer with JRE</div> <div>Size: 77,373,223 MD5: 50e28f2c2bf866a6004f3e9cb2bf64a5</div>	<div> Download Archive (dmg)</div> <div>Size: 79,228,951 MD5: b41220d3c83357599ae96d7516d3bc4d</div>	<div> Download Archive (tar.gz)</div> <div>Size: 17,704,367 MD5: c3eec679b8e59ca9a7e9ed18d2f62883</div>
<div>Portable Archive (7z)</div> <div>Size: 70,119,212 MD5: 8c07cd6eef9be3e1759388d11d2f25c5</div>		<div>Debian package (deb)</div> <div>Size: 17,703,248 MD5: e3a9be10fbae89845b8fd4b97b37d9b9</div>
<div>Installer without JRE (requires installed JRE 1.7+)</div> <div>Size: 12,605,908 MD5: 671b11ef7ca62ead667d2db8acf78854</div>		
The installer with JRE and the portable bundle both come with a Java Runtime Environment (JRE). If you want to use a different JRE which is already present on your system, get the installer without JRE.	The OS X bundle already contains the necessary JRE.	If there is no JRE present on your system yet, use the package manager of your distribution to install Java, e.g. from OpenJDK, or download from oracle.com .

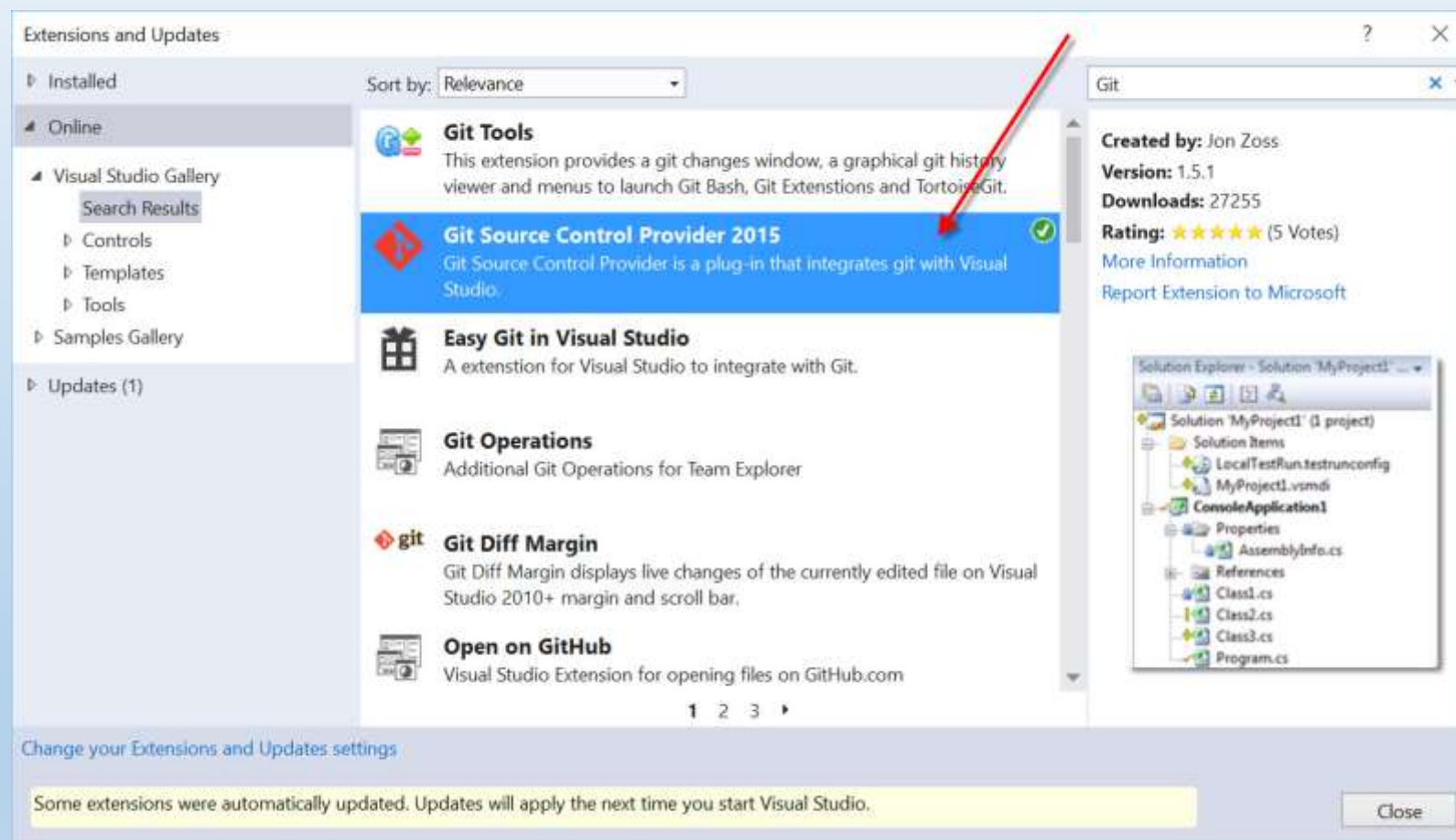
Sử dụng SmartGit



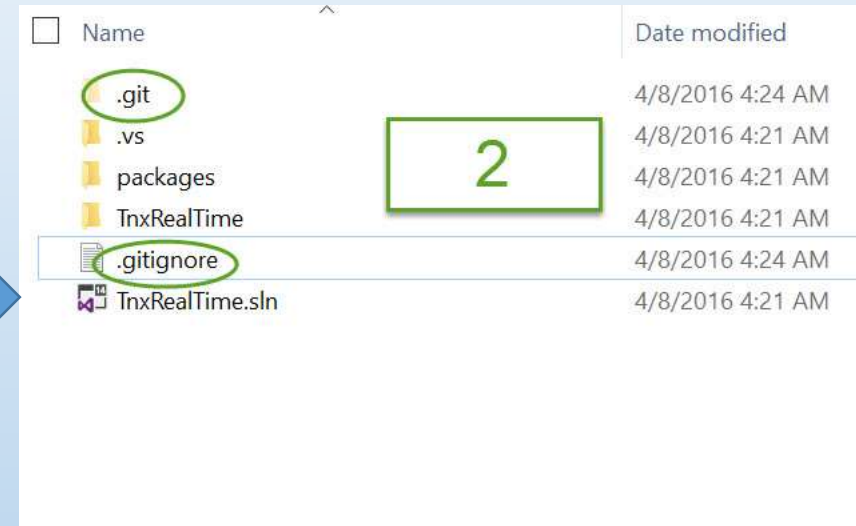
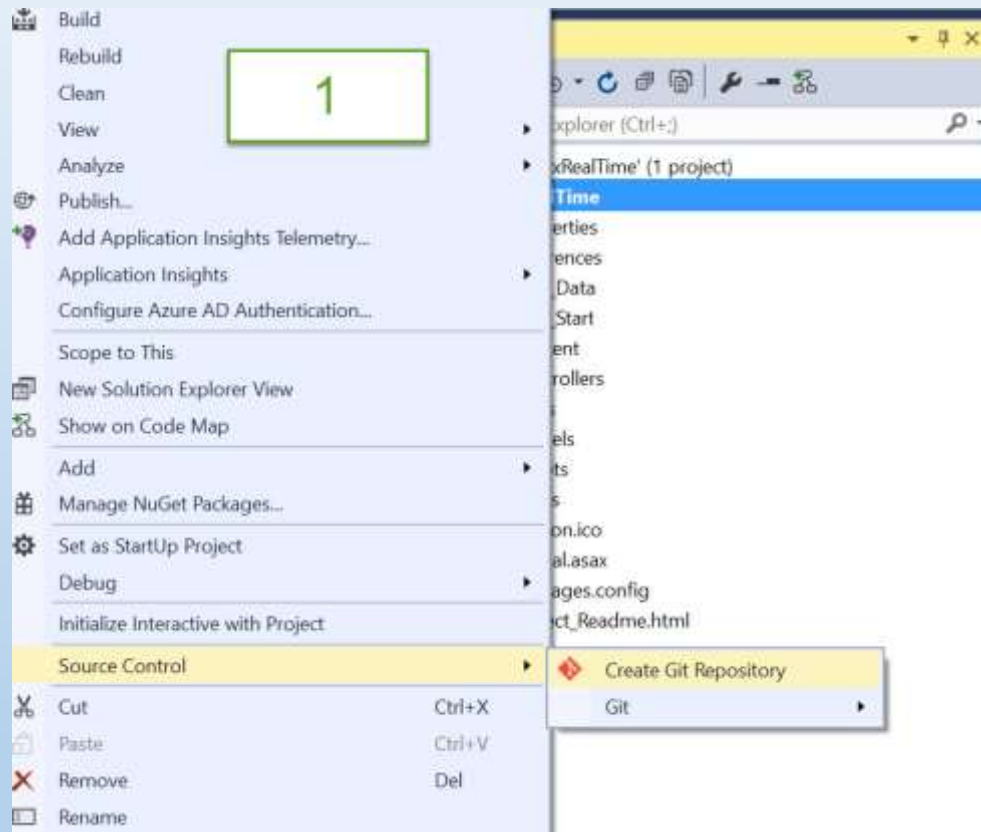
Tích hợp với IDE

- Tích hợp với Visual Studio
- ✓ Tích hợp với Netbeans
- ✓ Tích hợp với Eclipse
- ✓ Tích hợp với Xcode

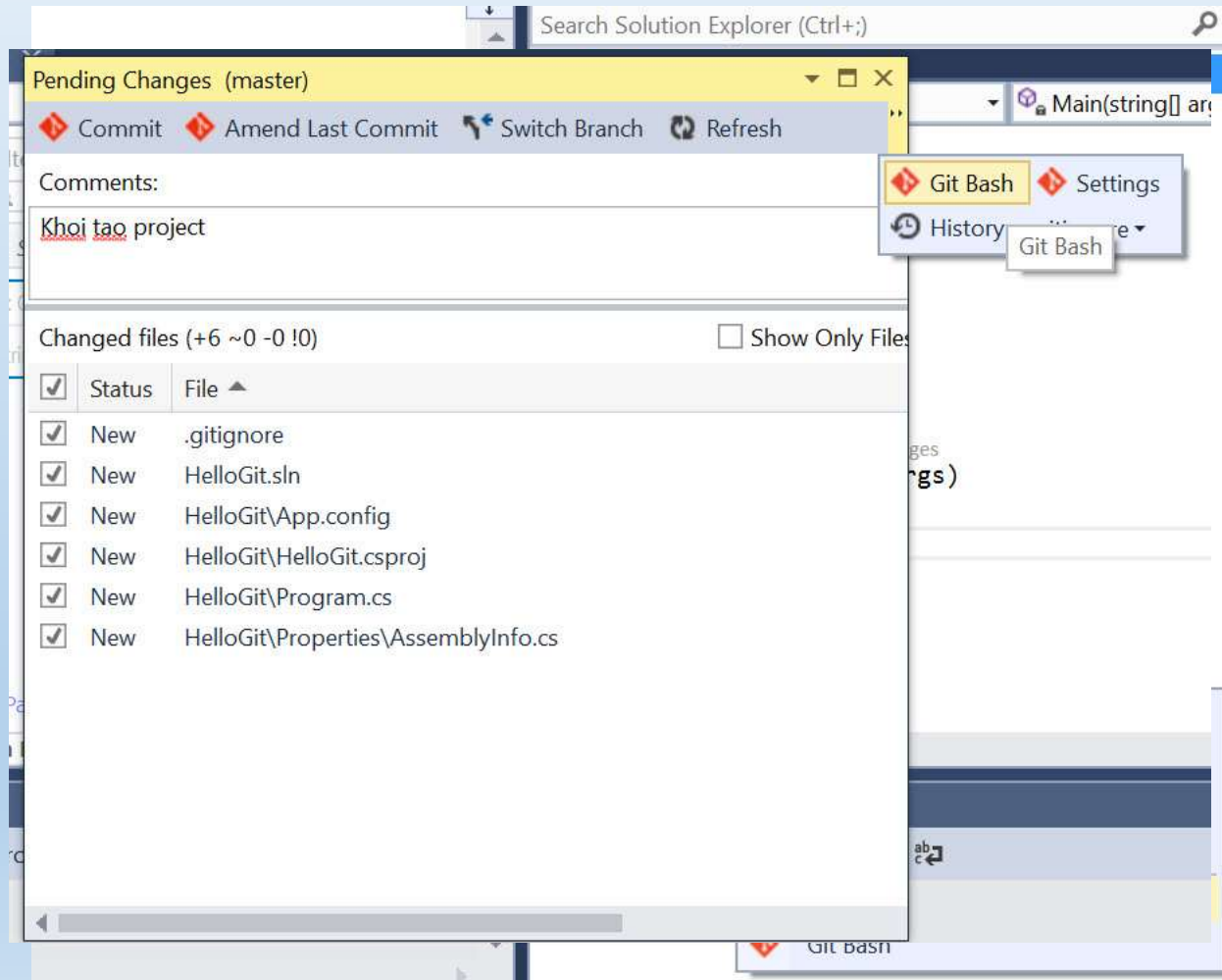
Tích hợp với Visual Studio



Tạo một Repository



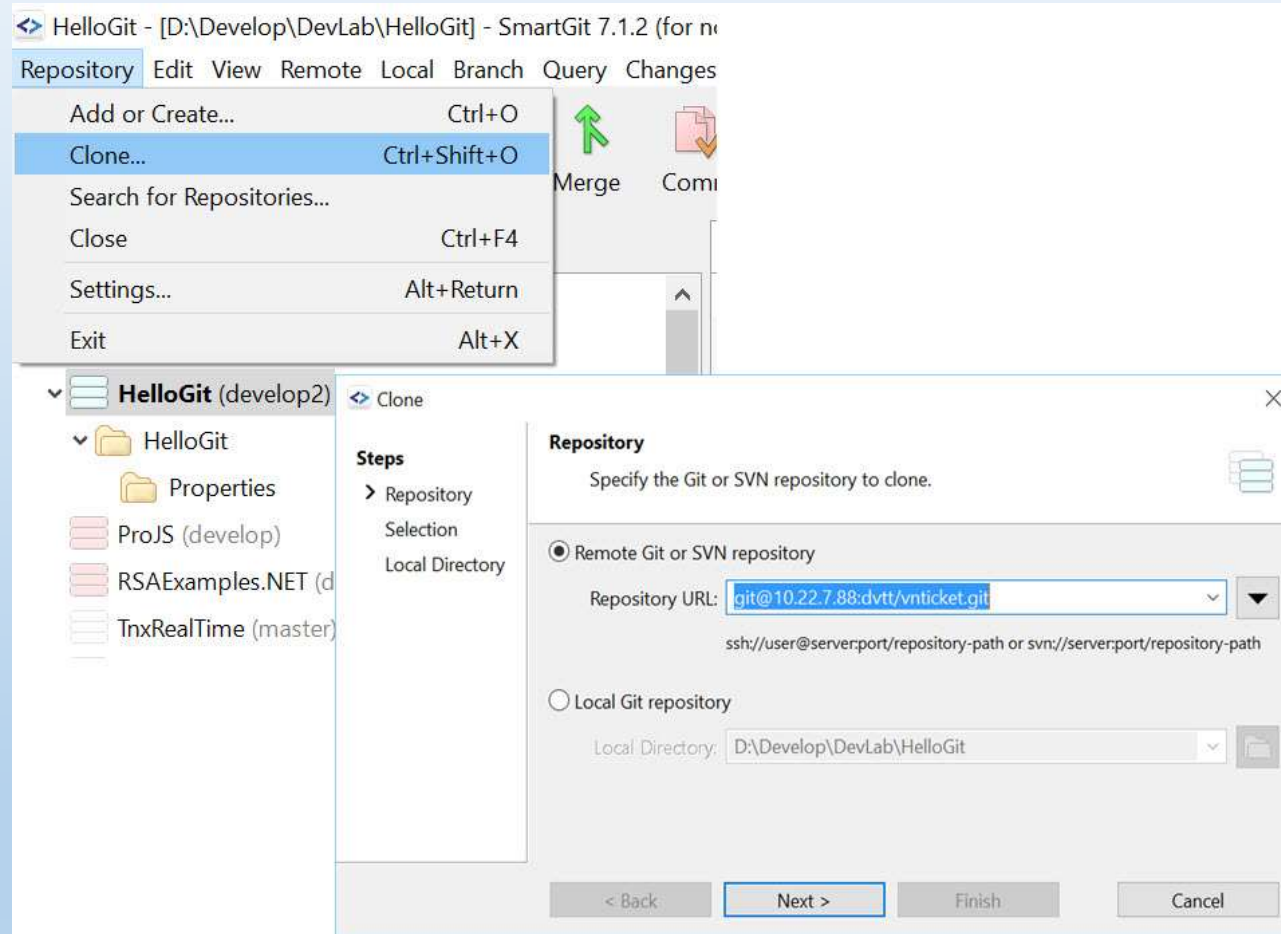
Commit code



Một số tình huống thường gặp trong thực tế

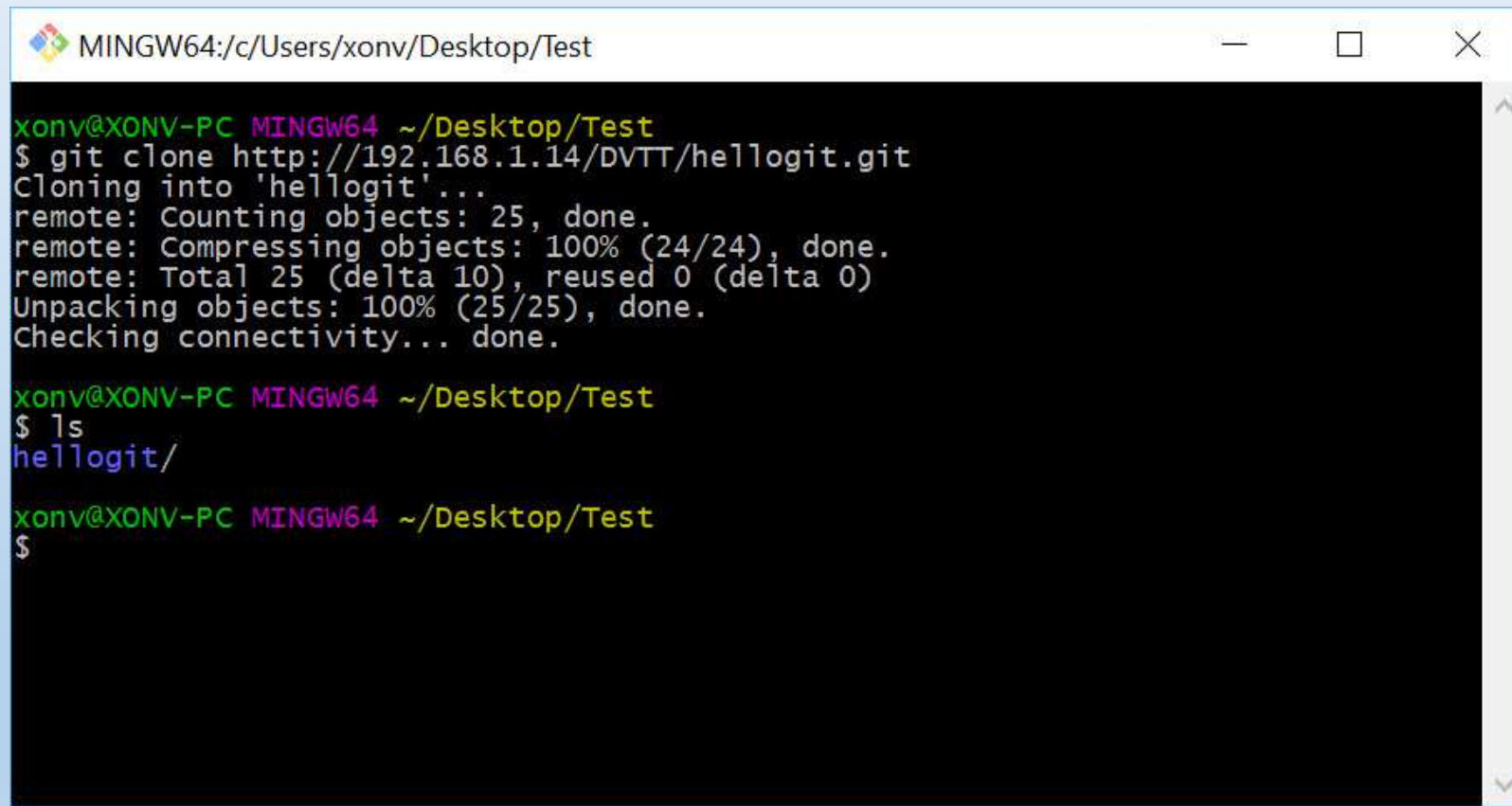
- Cập nhật .ssh key
- Clone code
- Xem lịch sử
- Rollback code
- Conflict code
- Deploy code – Tạo tag đánh dấu phiên bản
- Tạo nhánh
- Merge nhánh

Clone code - smartgit



Clone code - command

Cú pháp: `git clone [url]`



```
MINGW64:/c/Users/xonv/Desktop/Test

xonv@XONV-PC MINGW64 ~/Desktop/Test
$ git clone http://192.168.1.14/DVTT/hellogit.git
Cloning into 'hellogit'...
remote: Counting objects: 25, done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 25 (delta 10), reused 0 (delta 0)
Unpacking objects: 100% (25/25), done.
Checking connectivity... done.

xonv@XONV-PC MINGW64 ~/Desktop/Test
$ ls
hellogit/

xonv@XONV-PC MINGW64 ~/Desktop/Test
$
```

Xem lịch sử

Message

origin → develop cap nhat lan 2

origin → master init project

SHA [294d26dd](#)
by [xonv](#), 04/08/2016 04:15 PM

init project

Files (8) * Comments

Name	Modification	Relative Directory	Re
AssemblyInfo.cs	Added	HelloGit/Properties	
HelloGit.csproj	Added	HelloGit	
HelloGit.sln	Added	.	
Program.cs	Added	HelloGit	
Util2.cs	Added	HelloGit	
Util			

1
2
3
4
5
6
7
8

Show Changes F4
Compare with Local... Shift+F4
Save As... Ctrl+S
Log Ctrl+L
Blame Ctrl+Shift+L
Copy Relative Path Alt+Shift+Insert
Copy Name

Xem lịch sử

The screenshot displays the Visual Studio Git interface. On the left, the 'Commits (1)' pane shows a list of local branches: 'develop' (pointing to 'origin') and 'master' (pointing to 'origin'). Below this, the 'origin (2) - git@192.168.1.14:D' section is visible. The main 'Commits (2)' pane shows a commit history with two entries: 'origin develop cap nhat lan 2' (SHA: 24aeb9b7) and 'origin master init project' (SHA: 294d26dd). The 'Details' pane on the right shows the commit details for 'cap nhat lan 2', including the SHA, author 'xonv', date '04/08/2016 04:16 PM', and parent '294d26dd'. Below the commit details, a table lists the files changed in the commit:

Name	Modification	Relative Directory
Utils.cs	Modified	HelloGit

The bottom pane shows the 'Changes of Utils.cs (Modified)' with a diff view. The left side shows the original code (lines 3-14) and the right side shows the modified code (lines 8-19). The modified code includes a new method 'CapNhatLan2()' in the 'Utils' class.

```
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace HelloGit
8 {
9     class Utils
10     {
11         //Cap nhat lan 1
12     }
13 }
14
```

```
8 {
9     class Utils
10     {
11         //Cap nhat lan 1
12         private void CapNhatLan2()
13         {
14         }
15     }
16 }
17
18 }
19
```

Tạo nhánh - command

C:\ MINGW64:/d/develop/devlab/hellogit

```
xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
```

```
$ git checkout -b develop2
```

```
Switched to a new branch 'develop2'
```

```
xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop2)
```

```
$ git branch
```

```
develop
```

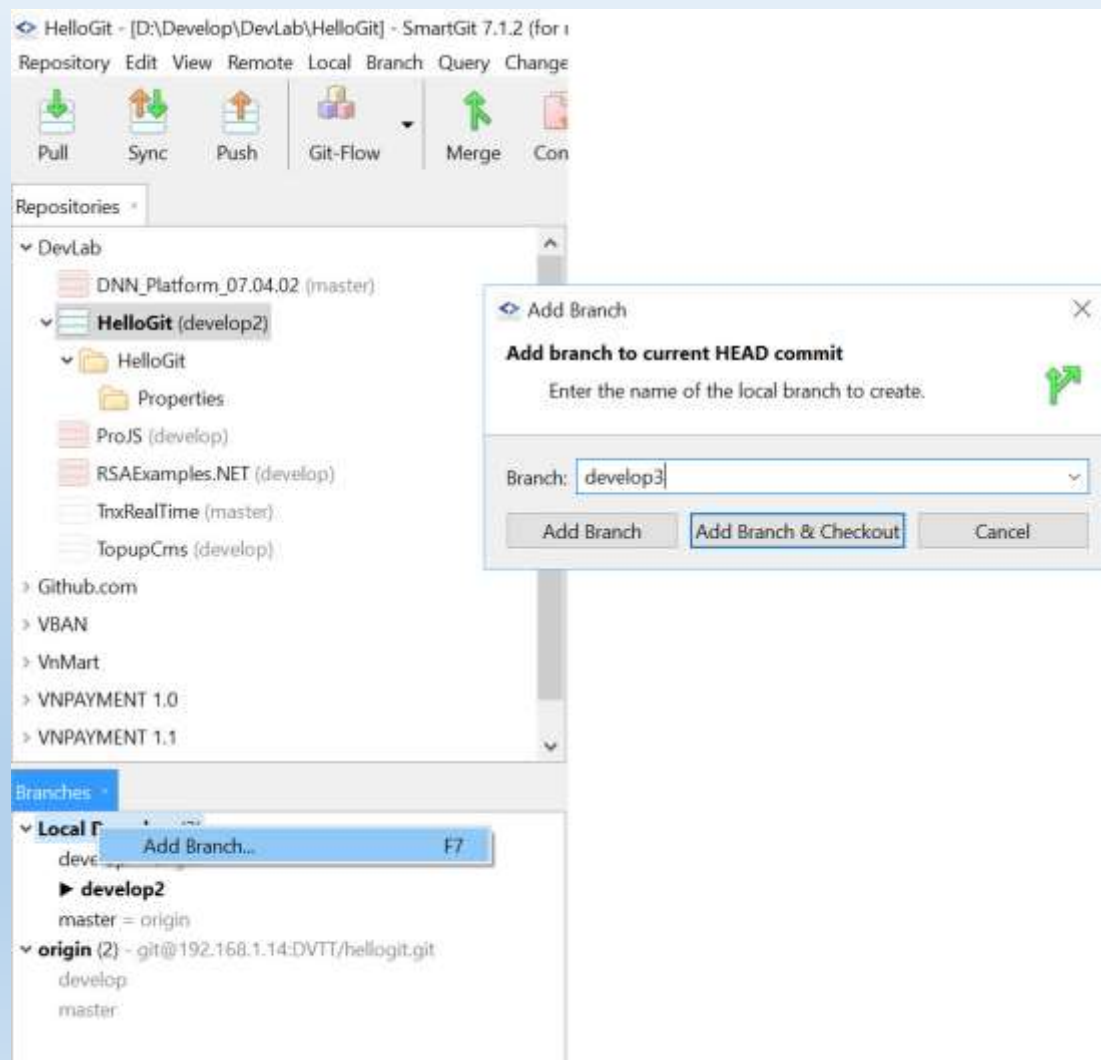
```
* develop2
```

```
master
```

```
xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop2)
```

```
$ _
```

Tạo nhánh - smartgit



Merge nhánh

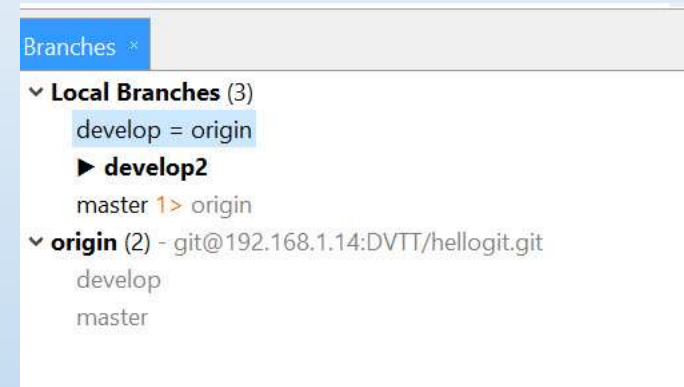
```
C:\> MINGW64:/d/develop/devlab/hellogit

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop2)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (master)
$ git merge develop2
Updating 294d26d..24aeb9b
Fast-forward
 HelloGit/Utils.cs | 5 +++++
 1 file changed, 5 insertions(+)

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (master)
$ git checkout develop2
Switched to branch 'develop2'

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop2)
$
```



Chú ý

➤ Nên:

- Nên hạn chế người có quyền truy cập vào nhánh master
- Những người có quyền master được merge từ nhánh develop <- master
- Code của master tương ứng với phiên bản mới nhất trên production

➤ Nên:

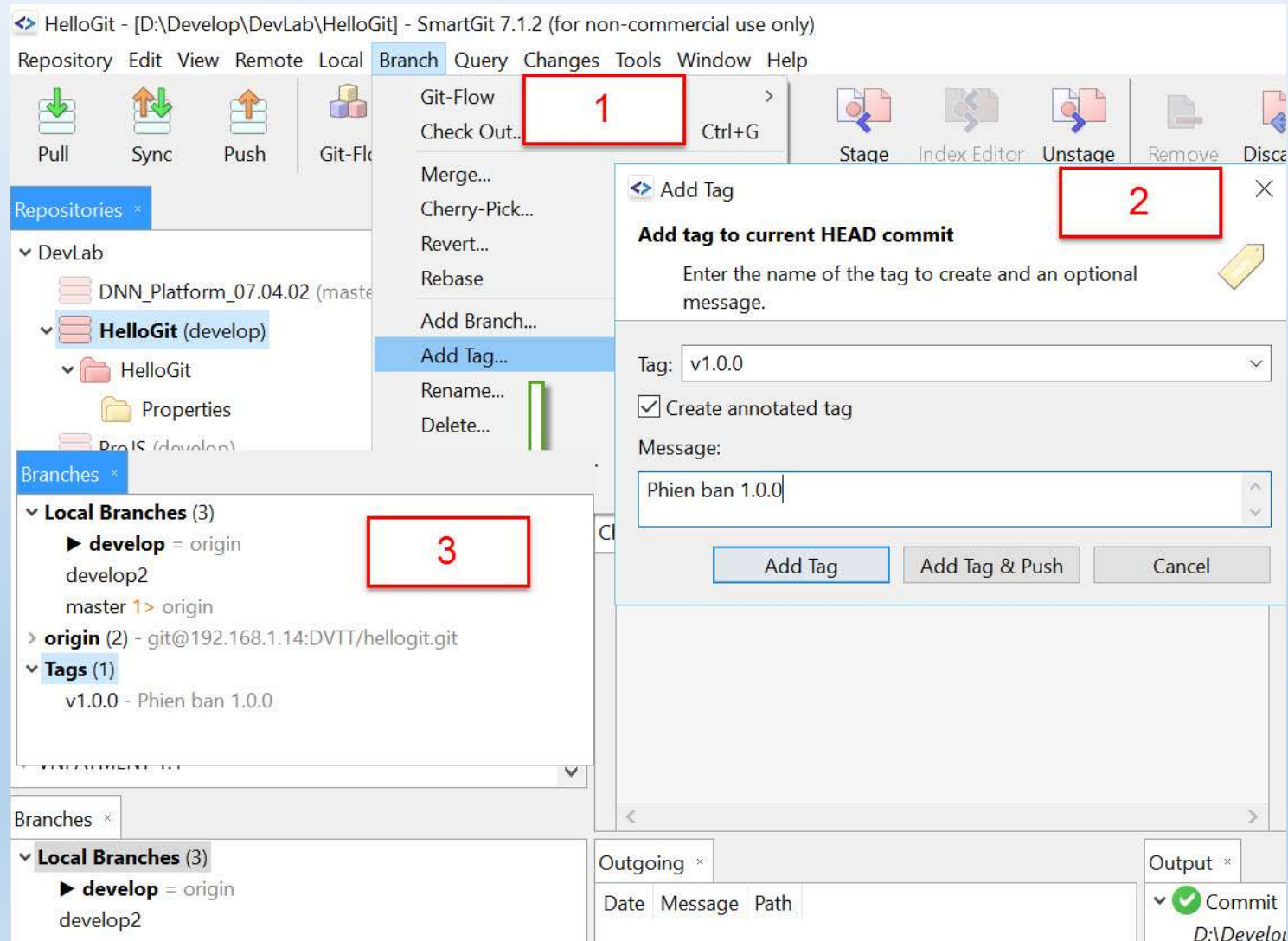
- Không cập nhật code lỗi lên trên server
- Commit trước
- Cập nhật code lên trên server trước khi ra về

Tips: Nếu đã code/fix xong thì commit luôn để hạn chế việc merge code

Rollback code

- Command:
- `git reset --hard [sha1]`
- `git push -f`

Deploy code – Tạo tag đánh dấu phiên bản



git tag tag_name – Tạo tag bằng command

```
CA MINGW64:/d/develop/devlab/hellogit

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$ git status
On branch develop
Your branch is up-to-date with 'origin/develop'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   HelloGit/Utils.cs

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$ git commit -m 'update tag v1.0.0'
[develop 5bdfa01] update tag v1.0.0
 1 file changed, 5 insertions(+)

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$ git tag
v1.0.0

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$ git tag v1.0.1

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$ git tag
v1.0.0
v1.0.1

xonv@XONV-PC MINGW64 /d/develop/devlab/hellogit (develop)
$
```

GIT TEAM! Where are you?



Q&A

- Tại sao sử dụng git mà không phải source control khác? (ưu, nhược của nó với các source control khác). cách thức sử dụng
- Ưu điểm vượt trội của GIT so với SVN là gì? Vì sao chúng ta nên dùng GIT?
- Giới thiệu GIT dùng nhiều cho dev, vậy nghiệp vụ và tester thì hiệu quả thế nào

Q&A

- Cách quản lý source của git như thế nào? Tại sao phải dùng nó?
- Cách tự tạo một Repository Server?
- Có nên thuê dịch vụ git của các nhà cung cấp dịch vụ như github, Bitbucket...? Nếu sử dụng server riêng có đảm bảo được chất lượng không ạ?

Bạn muốn đạt được điều gì sau buổi Seminar? (16 responses)

Sử dụng được và các bước cần thiết để merge dữ liệu từ các nhánh với nhau

Tìm hiểu thêm cách quản trị GIT thay vì chỉ Push, Pull, Clone

Quản lý công việc và tài liệu được đồng bộ, dễ dàng hơn

Hiểu rõ hơn về Git.

Hiểu biết thêm về GIT, có thể sử dụng GIT

Được giải đáp phần nào về kết hợp giữa công cụ và quy trình phát triển, triển khai, vận hành; thay vì chỉ tập trung vào sử dụng công cụ quản lý phiên bản nào.

quản lý phiên bản tài liệu đơn giản và nhanh chóng

Git khác svn ntn

Tôi muốn đoạt được cái USB 3.0 32Gb

Lí do nên dùng, cách thức sử dụng

Hiểu hơn về Git

Mình muốn biết về cách sử dụng của git, cái lợi và hại của git.

Hiểu rõ được khái niệm GIT là gì và có thể sử dụng được GIT để quản lý Source Code

hiểu chi tiết hơn về GIT

USB 32 GB

Tài liệu tham khảo

- Danh sách các tools:
- **1. GIT-SCM để cài git lên trên máy Windows (Máy Ubuntu/Mac thường tích hợp sẵn)**
- <https://git-scm.com/>
- **2. Cài đặt SmartGit làm client. Khi cài chú ý, chọn chế độ Standard để sử dụng miễn phí**
- <http://www.syntevo.com/smartgit/>
- **3. Tài liệu tham khảo**
- Tiếng Việt :
- <https://backlogtool.com/git-guide/vn/>
- Tiếng Anh:
- <https://git-scm.com/book/en/v2>
- Ngoài ra: <https://git-scm.com/book/vi/v1>
- **4. Tích hợp vào IDE**
- - Visual Studio: <https://github.com/jzoss/Git-Source-Control-Provider>
- - Netbeans/Intelij/XCode: Có sẵn
- - Eclipse thường có sẵn (đã kiểm tra trên bản Smars.2) nếu chưa có thì cài plugin:
- <http://www.eclipse.org/egit/>