Sun*

Getting Started + Introduction Git

Outline

1. Introduction to Version Control
2. Introduction to Git

1. Introduction to Version Control

1. About Version Control System

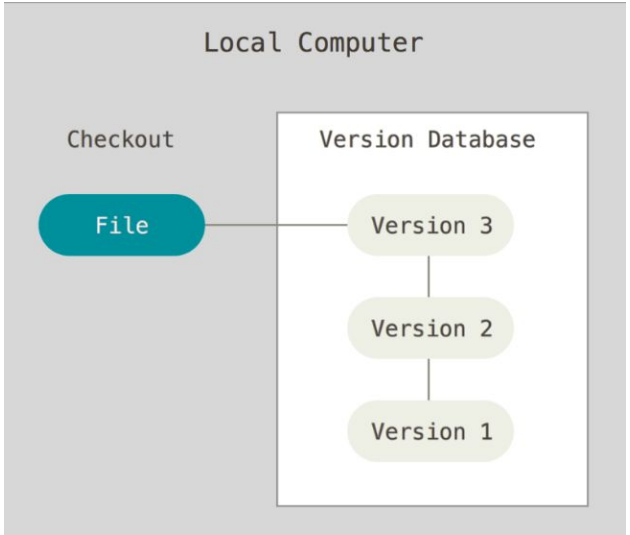
- Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.
- Using a Version Control System (VCS) you can easily recover.
- Version Control System include:
  - Local Version Control Systems
  - Centralized Version Control Systems
  - Distributed Version Control Systems

1. Introduction to Version Control

2. Local vs. Centralized vs. Distributed VCSs

- **Local Version Control Systems**

  Keep all the changes to files under revision control



1. Introduction to Version Control

2. Local vs. Centralized vs. Distributed VCSs

- **Centralized Version Control Systems**

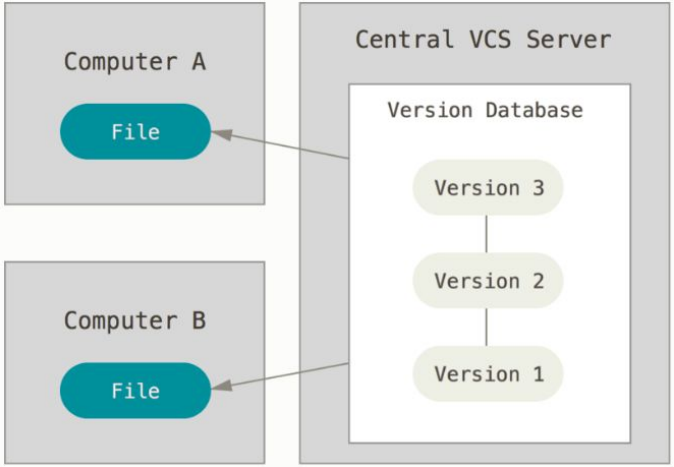
  Collaborate with developers on other systems.

  Have a single server

  that contains all the versioned files,

  and a number of clients

  that check out files from that central place



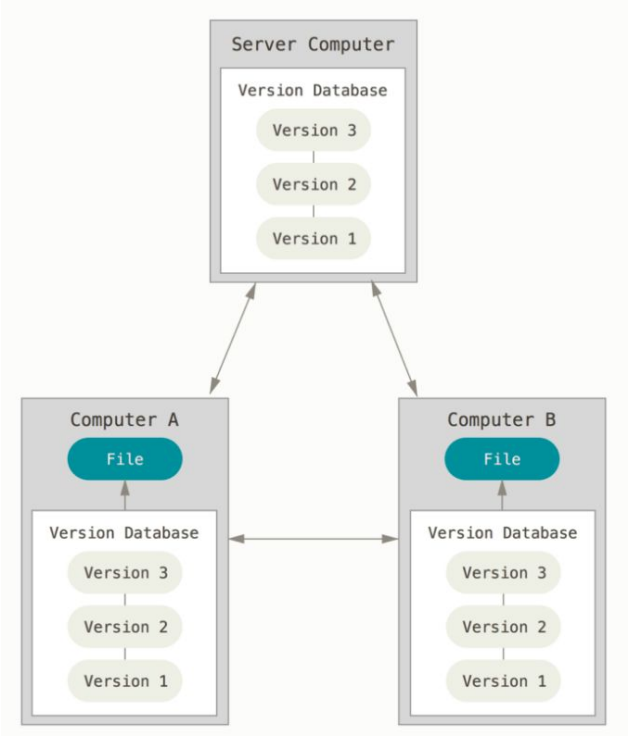
1. Introduction to Version Control

2. Local vs. Centralized vs. Distributed VCSs

- **Distributed Version Control Systems**

  Fully mirror the repository, if any server dies, and these systems were collaborating via it, any of the client repositories can be copied back up to the server to restore it.

  Every checkout is really a full backup of all the data.



2. Introduction to Git

2. History of Git

- Git began with a bit of creative destruction and fiery controversy.
- In 2002, the Linux kernel project began using a proprietary DVCS called BitKeeper.
- In 2005, BitKeeper broke down, and the tool's free-of-charge status was revoked.

This prompted the Linux development community to develop their own tool based on some of the lessons they learned while using BitKeeper. Some of the goals of the new system were as follows:

- Speed
- Simple design
- Strong support for non-linear development (thousands of parallel branches)
- Fully distributed
- Able to handle large projects like the Linux kernel efficiently (speed and data size)

Since its birth in 2005, Git has evolved and matured to be easy to use and yet retain these initial qualities. It's incredibly fast, it's very efficient with large projects, and it has an incredible branching system for non-linear development.

2. Introduction to Git

2. Differences between Git and other VCSs

Git	VCSs
- **Setting up repository**: stores repository in .git directory in top directory of your project - stores a reference to that snapshot	- **Setting up repository**: CVS require setting up CVSROOT - store information as a list of file-based changes made to each file over time

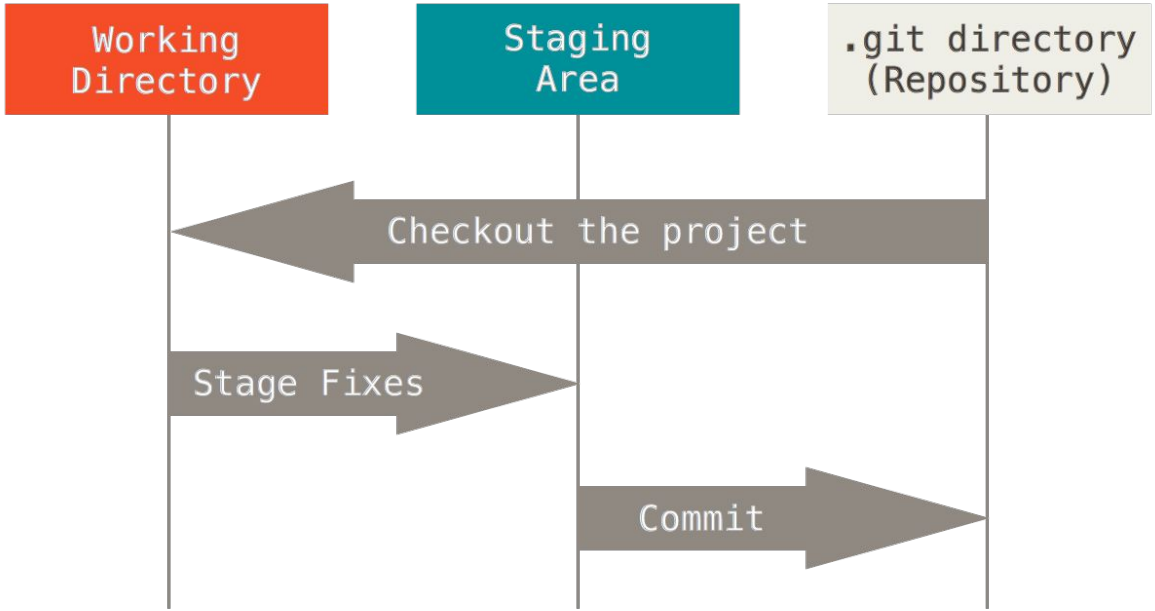
2. Introduction to Git

3. Git States

$ git add

$ git commit

$ git checkout



2. Introduction to Git

4. Installing Git

4.1 Installing on Linux

- If you're on Fedora for example, you can use dnf:

```
$ sudo dnf install git-all
```

- If you're on a Debian-based distribution like Ubuntu, try apt-get:

```
$ sudo apt-get install git-all
```

2. Introduction to Git

4.2 Installing on Mac

- Install the Xcode Command Line Tools

- Install it via a binary installer. An OSX Git installer is maintained and available for download at the Git website, at http://git-scm.com/download/mac.

- You can also install it as part of the GitHub for Mac install. Their GUI Git tool has an option to install command line tools as well. You can download that tool from the GitHub for Mac website, at http://mac.github.com.

2. Introduction to Git

4.3 Installing on Windows

- Just go to http://git-scm.com/download/win and the download will start automatically. To get an automated installation you can use the **Git Chocolatey package**. Note that the Chocolatey package is community maintained.

- Installing GitHub for Windows. You can download this from the GitHub for Windows website, at http://windows.github.com.

2. Introduction to Git

4.4 Installing from Source

- If you do want to install Git from source, you need to have the following libraries that Git depends on: autotools, curl, zlib, openssl, expat, and libiconv.

- Install:

```
$ tar -zxf git-2.0.0.tar.gz
$ cd git-2.0.0
$ make configure
$ ./configure --prefix=/usr
$ make all doc info
$ sudo make install install-doc install-html install-info
```

After this is done, you can also get Git via Git itself for updates:

```
$ git clone git://git.kernel.org/pub/scm/git/git.git
```

2. Introduction to Git

5. First-Time Git Setup

- The first thing you should do when you install Git is to set your user name and email address

```
$ git config --global user.name "John Doe"
$ git config --global user.email johndoe@example.com
```

- Now that your identity is set up, you can configure the default text editor that will be used when Git needs you to type in a message. If not configured, Git uses your system's default editor.

- Check your settings:

```
$ git config --list
user.name=John Doe
user.email=johndoe@example.com
color.status=auto
color.branch=auto
…
```

You can also check what Git thinks a specific key's value is by typing git config <key>:

```
$ git config user.name
John Doe
```

Q&A

Thank you