

# Software Development Methodologies

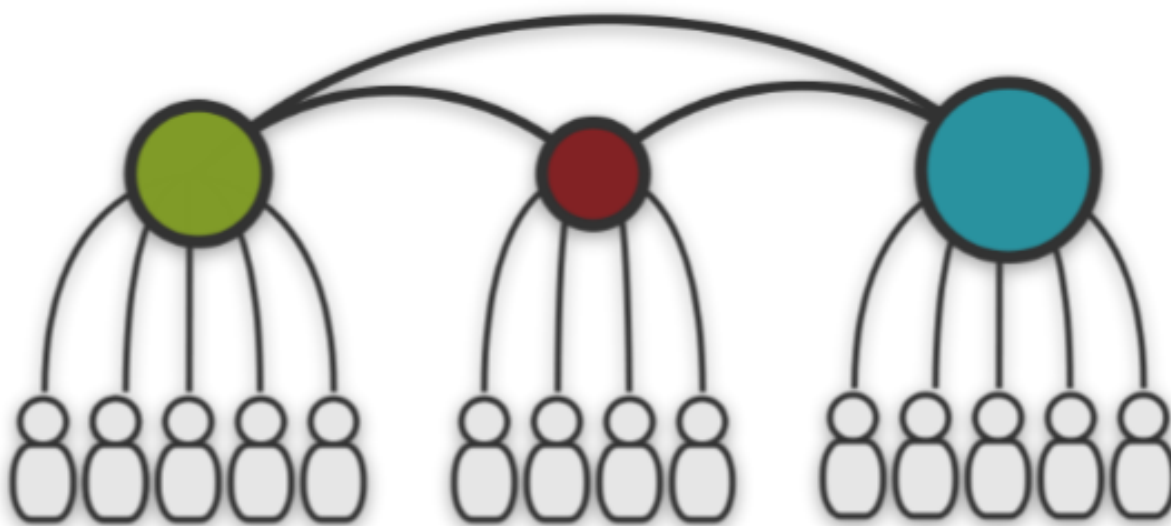
## Lecture 5 - Version Control 2

SOFTENG 750 2013-03-25

# If VCSs were Airlines...



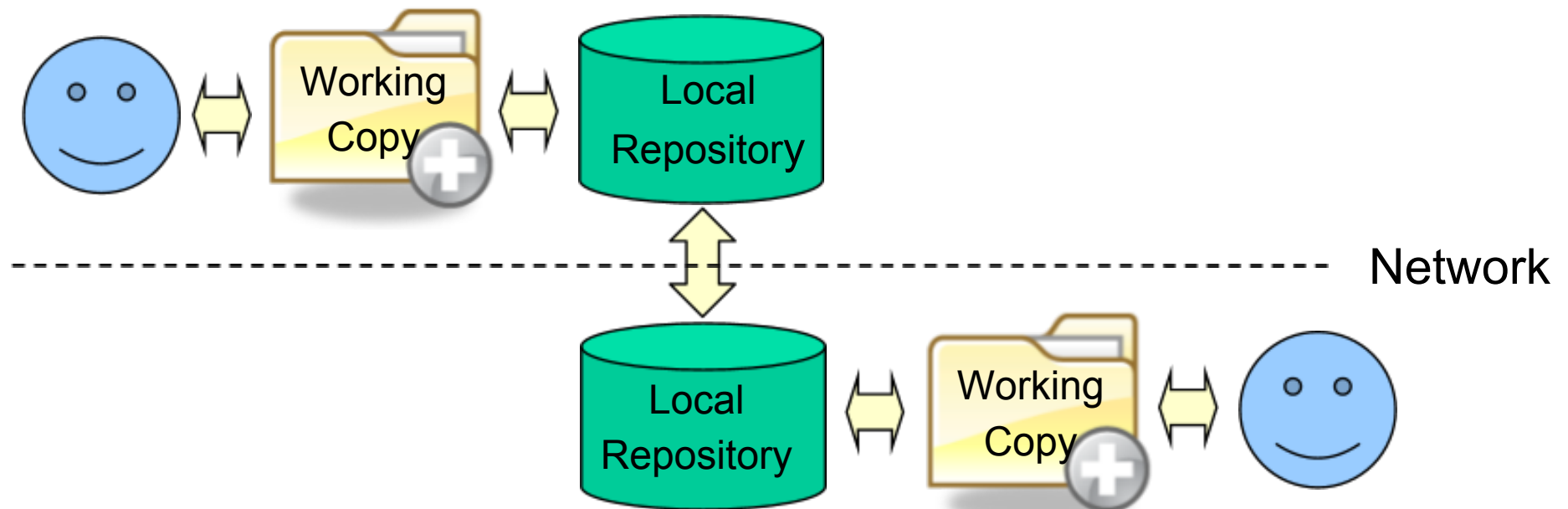
# Distributed Version Control



# Distributed Version Control

Every developer has their own local repository  
(a.k.a. “decentralized version control”)

1. Developers work on their working copy
2. Developers commit changes of the working copy to their own local repository first
3. Changes can be exchanged between repositories  
 (“pushed” and “pulled”)



# Push and Pull

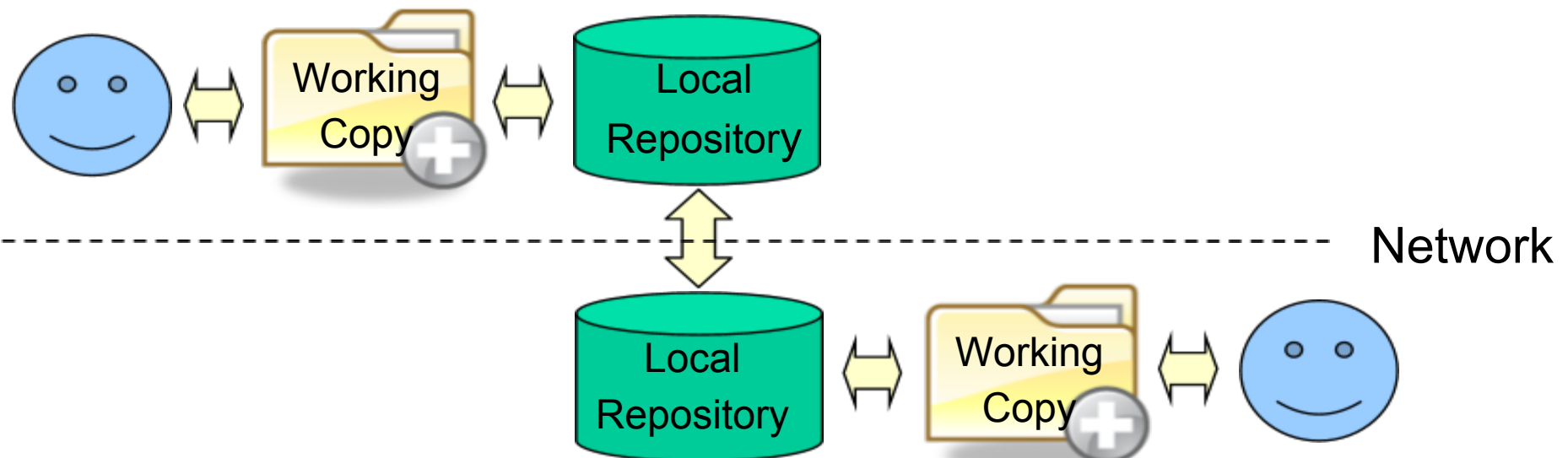
## Push

- Once developers have committed versions on their local repository, they can push them to another repo
- Versions are pushed from local branches into corresponding remote branches
- Like “commit” from one repo to another



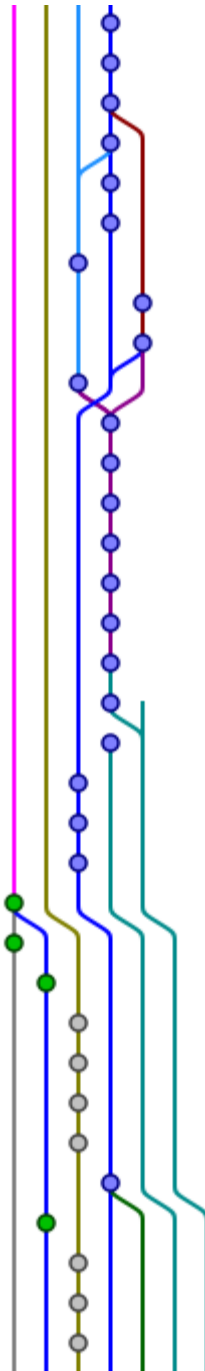
## Pull

- Latest versions are pulled from remote branches and put into the corresponding local branches
- Like “update” from one repo to another



# Branches

- Created by branching off or cloning existing branch (**parent** branch and **child** branch)
- Parent and child have a common history and can be merged later
- Typically branches will have a common **ancestor** (e.g. sometimes all branches start with a trunk)
- Typically new branches are only visible locally
- Need to be explicitly pushed or pulled to be visible in other repositories
- Having many temporary branches can be confusing



# Distributed VCS

## Advantages

1. Versioning can be done locally  
(does not depend on central repository)
  - Good if you don't have Internet connectivity
  - Good if you don't have access to the main repo
  - Good for bigger changes that involve many steps

2. Branching is easier
  - Easy to clone a repository
  - Branching can be done locally

3. Merging is easier
  - Merging can be done locally
  - Because history of a branch is kept locally, changes can be easier merged back into ancestor



## Disdvantages

### 1. Makes it easier for people not to integrate their work

- All versioning can be done locally
- Easier to forget to push
- Related problem: branches becoming “forks”



### 2. Memory consumption of local repository

- Up-to-date local repository contains same data as remote repository
- All branches, versions, files (even old ones)...





# Mercurial (hg)



# Mercurial

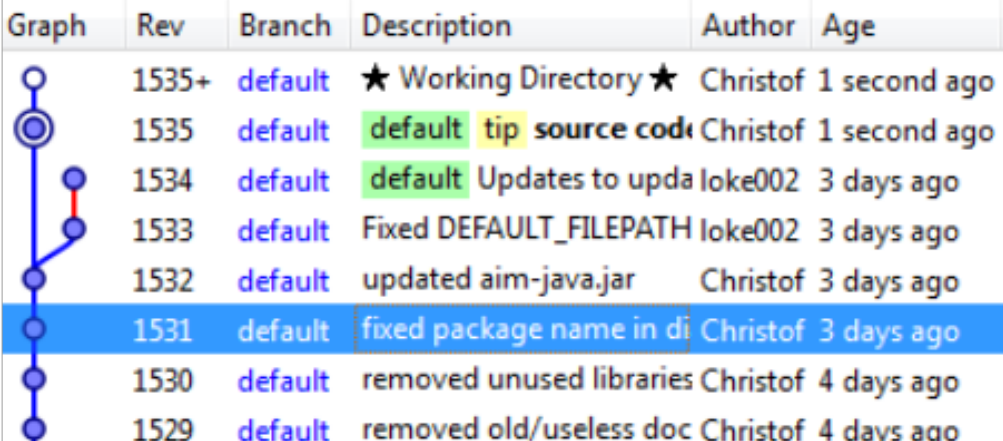










- Open-source project, started around 2005
- Used for many open-source projects
- Every developer has a repository, which is a folder
- Repo folder contains working copy,  
and a subfolder .hg which contains the version data
- Versions are identified locally by natural numbers  
and globally by hash values,  
e.g. 5c240805ac2d9530b780cbd514574af398c0cdd6
- Good tool support (TortoiseHg)
- Fairly easy to use

# Working with Hg

1. Start by **cloning** existing repo, or **creating** new one
  - New repo has only “default branch” (like trunk)
  - After cloning you have local copies of all branches of parent repo
2. **Pull** to load new versions from parent repo into local repo

- Does not change working copy unless you **update**
- Pulled versions are put in separate branch from your local versions



Graph	Rev	Branch	Description	Author	Age
	1535+	default	★ Working Directory ★	Christof	1 second ago
	1535	default	default tip source code	Christof	1 second ago
	1534	default	Updates to upda loke002	loke002	3 days ago
	1533	default	Fixed DEFAULT_FILEPATH	loke002	3 days ago
	1532	default	updated aim-java.jar	Christof	3 days ago
	1531	default	fixed package name in di	Christof	3 days ago
	1530	default	removed unused libraries	Christof	4 days ago
	1529	default	removed old/useless doc	Christof	4 days ago

3. Modify working copy and **commit** to create new versions in your local repo

# Hg Pull

**Pull** regularly to stay up to date.

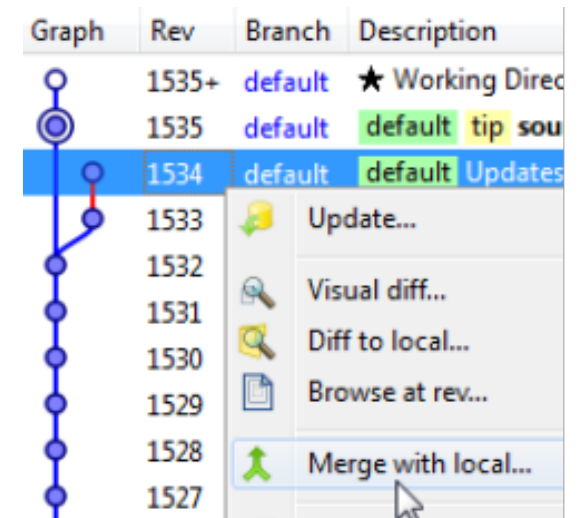
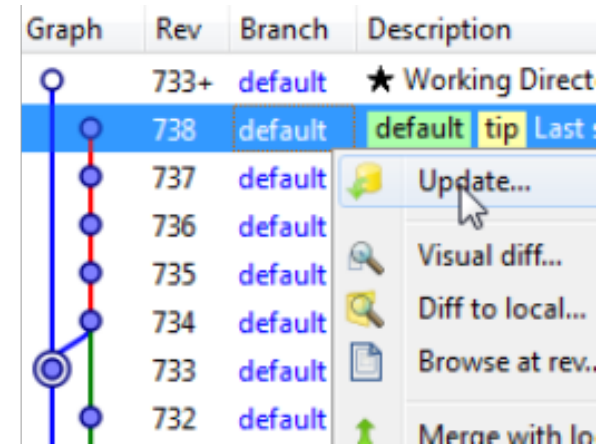
Have you committed local versions on some branch?

1. If no, you can **update** to the latest pulled version

- Changes in working copy are merged with pulled version
- Unless you choose to

“discard local changes”

2. If you have committed local versions on some branch, they should be **merged** with pulled versions on same branch

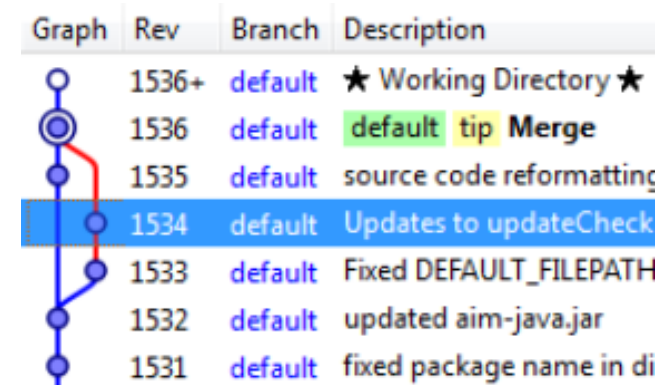
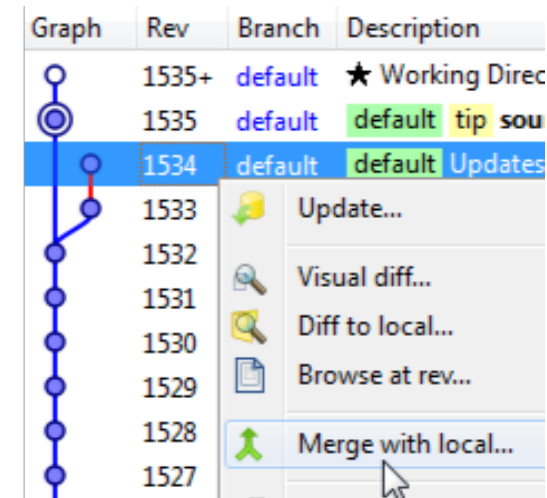


# Hg Push

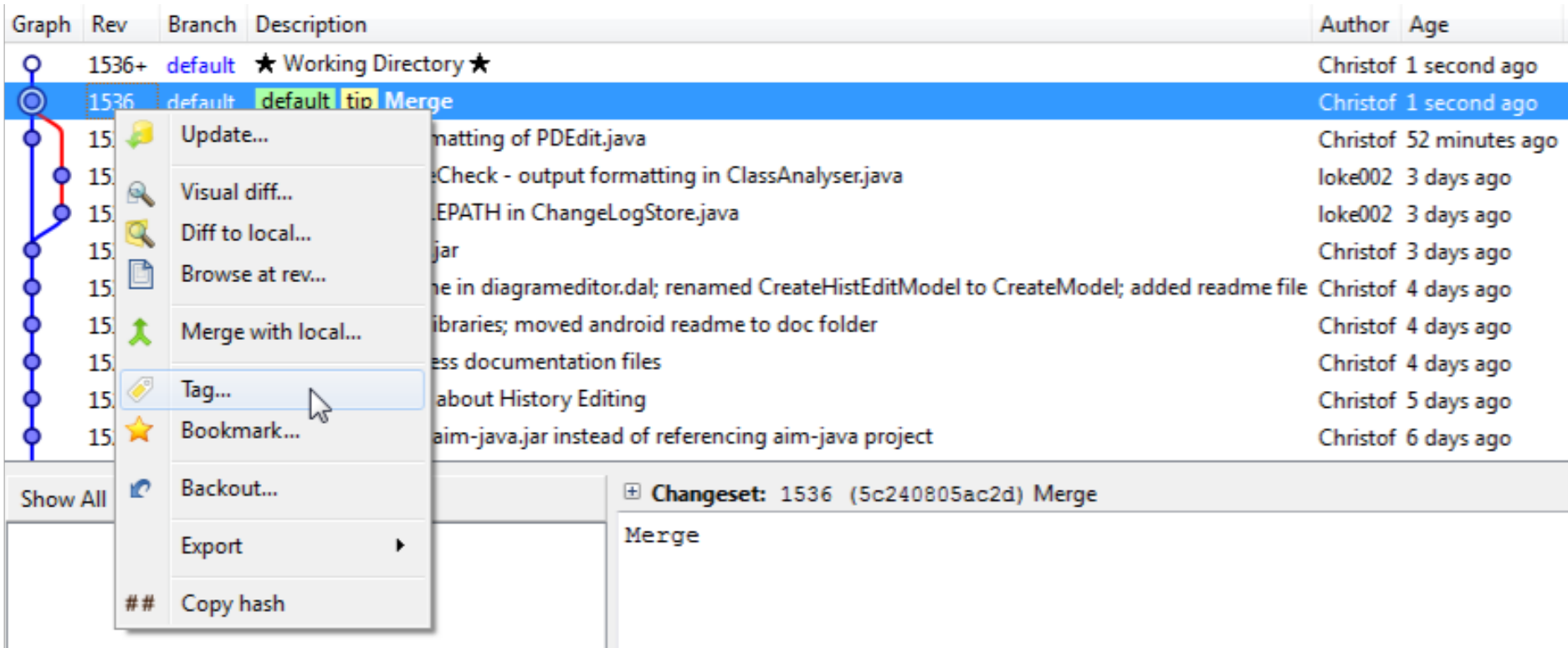
**Push** regularly to integrate your changes.

Have others committed versions on a remote branch that you have committed to locally?

1. If no, **push** will succeed and the local versions will be in the remote repo
2. If yes, i.e. others have committed versions on a branch you have committed to locally:
  - You need to **merge** your versions with their versions
  - When local branches and corresp. remote branches are merged, **push** succeeds



# More Hg Functions



The screenshot shows the TortoiseHg interface. On the left is a graph view of the commit history. The main area is a table of revisions. A context menu is open over the selected revision (1536), listing various actions like 'Update...', 'Visual diff...', 'Diff to local...', 'Browse at rev...', 'Merge with local...', 'Tag...', 'Bookmark...', 'Backout...', 'Export', and 'Copy hash'. The 'Export' option has a right-pointing arrow. Below the table, a 'Changeset' summary is visible for revision 1536, which is a 'Merge'.

Graph	Rev	Branch	Description	Author	Age
	1536+	default	★ Working Directory ★	Christof	1 second ago
	1536	default	default tip Merge	Christof	1 second ago
	15		formatting of PEdit.java	Christof	52 minutes ago
	15		Check - output formatting in ClassAnalyser.java	loke002	3 days ago
	15		EPATH in ChangeLogStore.java	loke002	3 days ago
	15		jar	Christof	3 days ago
	15		ne in diagrameditor.dal; renamed CreateHistEditModel to CreateModel; added readme file	Christof	4 days ago
	15		libraries; moved android readme to doc folder	Christof	4 days ago
	15		ess documentation files	Christof	4 days ago
	15		about History Editing	Christof	5 days ago
	15		aim-java.jar instead of referencing aim-java project	Christof	6 days ago

Changeset: 1536 (5c240805ac2d) Merge  
Merge

- **Backout**: undo the changes in a version
- **Browse** the files as seen in other revisions
- **Export** changes as patch files
- Note: older versions 1.x of TortoiseHg look different

git



- Created by Linus Torvalds
- Open-source project, started around 2005
- Used for many open-source projects
- Generally similar to Mercurial, but different terminology
  
- Fast cloning, branching and merging
- Lots of things that can be changed by users, e.g. history can be changed a-posteriori
- Personal experience as compared with Mercurial: more powerful and faster, but harder to use





# Git Branches



- The default branch is called master
- There are different types of branches:
  - Local branches (e.g. master)
  - Remote branches (e.g. origin/master)
  - Tracking branches (Special type of local branches; local copies of remote branches)
- You always work in a **local branch**
- You never work in **remote branches** (they are somewhere else)
- To work with a remote branch, you need to create a corresponding **tracking branch** locally

# Important Operations on Branches



- `git branch newbranch existingbranch`  
Create new branch from an existing branch
- `git branch --track branchx origin/branchx`  
Create local tracking branch for a remote branch
- `git checkout branchname`  
Make a branch appear in the repo folder
- `git pull` Merge latest changes from remote branch to local tracking branch (could ask to resolve conflicts)
- `git push` Merge latest changes from local tracking branch to remote branch (complains if not up-to-date)
- `git merge source`  
Merge branch source into the current working copy

# Example Session

```
git clone clut002@genoupe.se.auckland.ac.nz:/var/git/pdstore
cd pdstore // clone the repo and go into it
echo "hello" > newfile.txt
git add newfile.txt // mark the new file for addition
git commit -a

git branch --track ProjectX origin/ProjectX // new tracking
// branch

git checkout ProjectX // checkout tracking branch
git pull // update tracking branch

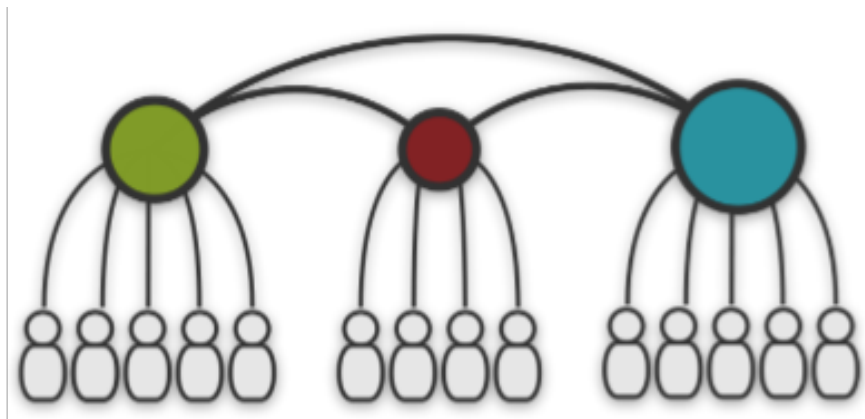
git merge master // merge changes of master to here
git push // send changes to remote branch
```



# Today's Summary

Distributed VCSs: every user has a full repository with versions (not just a working copy)

- Versions are committed to local repository first
- Versions can be **pushed** from local to remote repository
- Versions can be **pulled** from remote to local repository
- Both may require merging and conflict resolution



# Quiz

1. What is the main difference between centralized and distributed version control?
2. What does hg pull do?
3. Name two advantages of distributed version control.

```
#include <math.h>
#include <sys/time.h>
#include <X11/Xlib.h>
#include <X11/keysym.h>
double L, o, P
, _dt, T, Z, D=1, d,
s[999], E, h= 8, I,
J, K, w[999], M, m, O
, n[999], j=33e-3, i=
1E3, r, t, u, v, W, S=
74.5, l=221, X=7.26,
a, B, A=32.2, c, F, H,
int N, q, C, v, D, U;
Window z; char f[82]
; GC k; main(){ Display*d=
XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k,XCreateGC (e,z,0,0),BlackPixel(e,0))
; scanf("%1f%1f%1f",y +=n,w+y, y+s)+1; y +=); XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400,
0,0,WhitePixel(e,0) ).KeyPressMask); for (XMapWindow(e,z); ; T=sin(O)){ struct timeval G={ 0,dt*1e6}
; K= cos(j); N=le4; M+= H*; Z=D*K; F+= *P; r=E*K; W=cos(O); m=R*W; H=K*T; O+=D* *F/ R+d/K*E*; B=
sin(j); a=B*T*D-E*W; XClearWindow(e,z); t=T+E+ D*B*W; j+=d* *D-*F*E; P=W*E*B-T*D; for (o+=(I=D*W+E
*T*B,E*d/K *B+v+B*K*F*D)*; pcy; ){ T=p[s]+1; E=o-p[w]; D=n[p]-L; K=D*m-B*T-H*E; if (p [n]+w[ p]+p[s
] == 0)K <fabs(W-T-r-I*E +D*F) [fabs(D-t *D+Z *T-a *E)> K)N=le4; else{ q=W/K *4E2+2e2; C= 2E2+4e2/ K
*D; N=1E4&& XDrawLine(e ,z,k,N ,U,q,C); N=q; U=C; } +=p; } L+= * (X*t +P*M+m*1); T=X*X+ 1+1+M *M;
XDrawString(e ,z,k ,20,380,f,17); D=w/1+15; i+=(B *1-M*r -X*Z)*; for(;; XPending(e); u +=CS!=N){
XEvent z; XNextEvent(e ,&z);
++*((N=XLookupKeysym
((z.xkeycode))-1)?
N-L? UF-N?4 E&
J& u: &h); --{
DN -N? N-DT ?N==
RT?4u: & W&h:4J
); } m=15*F/1;
c+=(I=M/ 1,1*H
+I*M+a*X)*; H
-A*r+v*X-F*1+(
E=-1-X*4.9/1,t
-T*m/32-I*T/24
)/S; K=F*M+(
h* 1e4/1-(T+
E*s*T*E)/3e2
)/S-X*d-B*A;
s+=.63 /1*d;
X+=( d*1-T/S
* [.19*E +a
*.64+J/1e3
)-M* v +A*
Z)*; l +=
K *_; W=d;
sprintf(f,
"%5d %3d"
"%7d",p -1
/1.7, (C=9E3+
O*57.3)40550, (int)l); d=T*(.45-14/1+
X-a*130-J* .14)*_/125e2+F*^v; P=[T*(47
*I-m* 52-E*94 *D+*.38+u*.21*E) /1e2+W*
179*v)/2312; select(p=0,0,0,6G); v=(
W*F-T*(.63*m-I*.086*m*E*19-D*25-.11*
)/107e2)*_/; D=cos(o); E=sin(o); } }
```

International Obfuscated  
C Code Contest (ioccc.org)  
- Carl Banks 1998

A flight simulator.