Laboratory for Percutaneous Surgery

Managing your project in the PerkLab

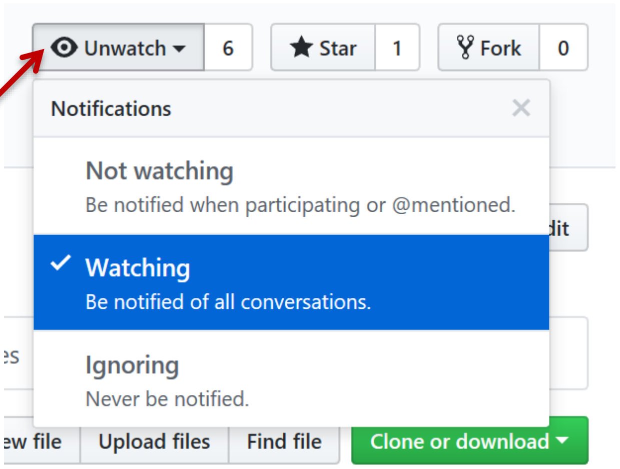
Andras Lasso

Laboratory for Percutaneous Surgery (Perk Lab)
School of Computing
Queen's University, Kingston, ON, Canada

Queen's University

Basic mechanisms

- Set up project space
  - Which repository? Ask your mentor.
    Most commonly: https://github.com/PerkLab private project.
    Make it public when published.
  - Do not ignore email notifications.
    May set to *Not watching*. Avoid *Ignoring*.
- Create README.md
  - Overall goal, approach, high-level plan.
  - Resources: directory on file server, other repositories, etc.
- Set up project board
- Set up regular (weekly) execution control meeting
  - Review what was done vs. planned, check if help needed
    => update issues, add notes.
  - Update short-term plan (2 weeks) => update project board.
  - Check if long-term plan is still OK => update plan if needed.



Not watching
Be notified when participating or @mentioned.

✓ Watching
Be notified of all conversations.

Ignoring
Never be notified.

Documentation

- Frequently changing, temporary information (short-term planning, administration), not for archival => issue tracker, project board
- Software documentation:
  - User manual, Developer manual => few Markdown (.md) files in root folder.
  - API documentation => in source code. Use *doxygen* style for C++, *Sphinx docstring / restructured text* for Python.
- Key papers, presentations create for the project => Doc subfolder in source code repository, in editable format (Word, PowerPoint); avoid large binary files, frequent updates

Issue tracking

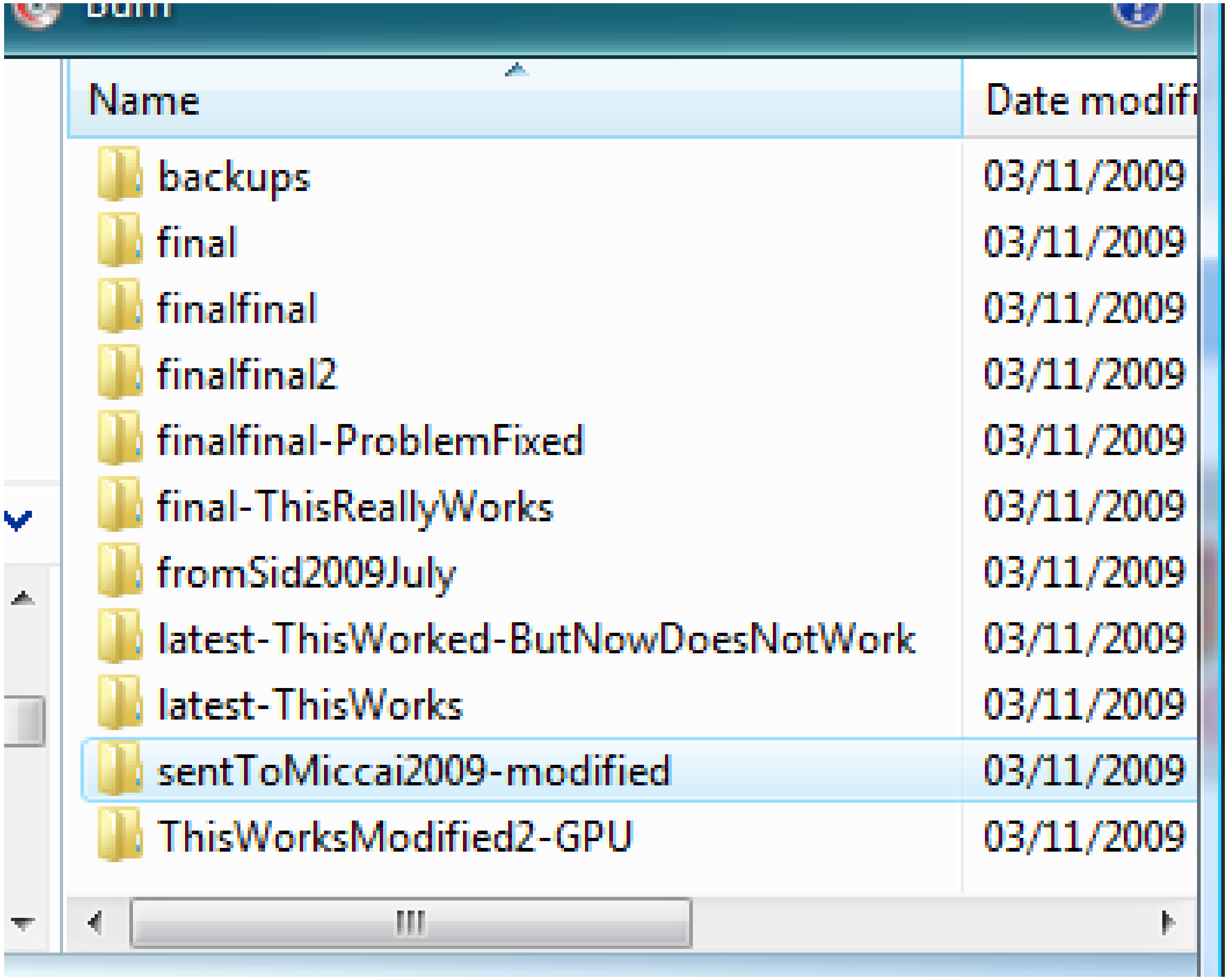
- Tracking tasks: issues, feature requests, enhancement ideas
- Entering a bug report: What did you do? What did you expect? What happened instead? Attach logs, screenshot.
- Add comment whenever making progress
- Refer to issues in source code commits to automatically link them
- @mention people in a comment to get input from someone (instead of writing an email)

GitHub Projects vs. Releases

- Projects:
  - Supports continuous, organic evolution
  - Works across repositories, easy drag-and-drop
  - Columns on the project board
    - **To do**: when tasks are completed, we choose what to do next from this column.
    - **In progress**: currently somebody working on it; make sure somebody is assigned.
    - **Done**: closed issues, no further action is planned.

    Some other useful columns:
    - **To test:** useful for projects where users and developers are different group.
    - **Future:** collection of ideas for the future
  - Example: https://github.com/twbs/bootstrap/projects/13 (we only use in private projects)
- Releases:
  - Supports upfront planning and monitoring progress
  - The way to define deadlines (no deadlines for individual tickets)
  - Files can be attached (e.g., installation package)
  - Example: https://github.com/PlusToolkit/PlusLib/milestone/4

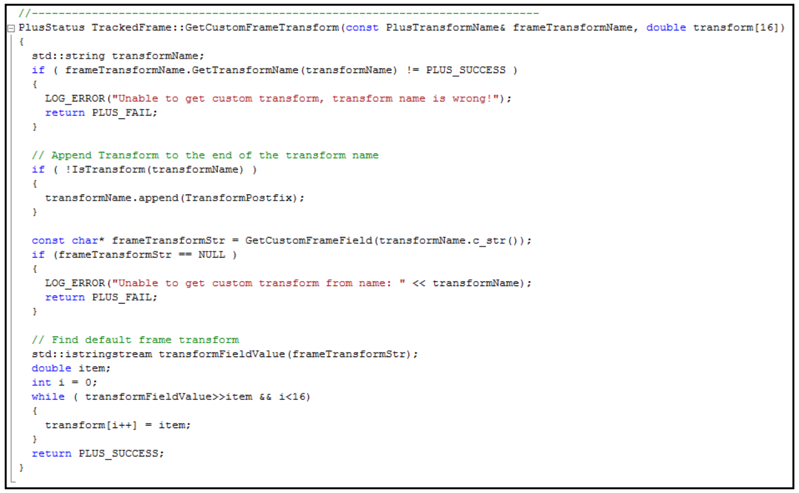
Revision control – the naïve way



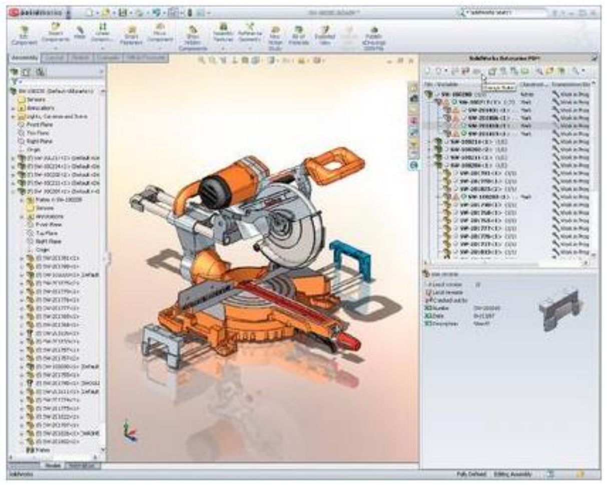
Name	Date modifi
backups	03/11/2009
final	03/11/2009
finalfinal	03/11/2009
finalfinal2	03/11/2009
finalfinal-ProblemFixed	03/11/2009
final-ThisReallyWorks	03/11/2009
fromSid2009July	03/11/2009
latest-ThisWorked-ButNowDoesNotWork	03/11/2009
latest-ThisWorks	03/11/2009
sentToMiccai2009-modified	03/11/2009
ThisWorksModified2-GPU	03/11/2009

What does revision control offer?

- Organized storage of multiple file versions
- Records history of all file changes (who modified, what, when, why)
- Convenient user interface for comparing and merging different file versions
- Allows concurrent modifications by multiple people



```cpp
//---------------------------------------------------------------------------
PlusStatus TrackedFrame::GetCustomFrameTransform(const PlusTransformName& frameTransformName, double transform[16])
{
  std::string transformName;
  if ( frameTransformName.GetTransformName(transformName) != PLUS_SUCCESS )
  {
    LOG_ERROR("Unable to get custom transform, transform name is wrong!");
    return PLUS_FAIL;
  }

  // Append Transform to the end of the transform name
  if ( !IsTransform(transformName) )
  {
    transformName.append(TransformPostfix);
  }

  const char* frameTransformStr = GetCustomFrameField(transformName.c_str());
  if (frameTransformStr == NULL )
  {
    LOG_ERROR("Unable to get custom transform from name: " << transformName);
    return PLUS_FAIL;
  }

  // Find default frame transform
  std::istringstream transformFieldValue(frameTransformStr);
  double item;
  int i = 0;
  while ( transformFieldValue>>item && i<16)
  {
    transform[i++] = item;
  }
  return PLUS_SUCCESS;
}
```



Alternative #1

“I can do the same by saving a copy of my complete source code directory, whenever I have a new version“

FAIL

- ✓ Organized storage of multiple file versions
- ⚠ Records history of all file changes (who modified, what, when, why)
- ⚠ Convenient user interface for comparing and merging different file versions
- ⚠ Allows concurrent modifications by multiple people

Revision control – Myth #2

"I just use Dropbox"

FAIL

- ⚠ Organized storage of multiple file versions
- ⚠ Records history of all file changes (who modified, what, when, why)
- ⚠ Convenient user interface for comparing and merging different file versions
- ✓ Allows concurrent modifications by multiple people

Revision control – Myth #2

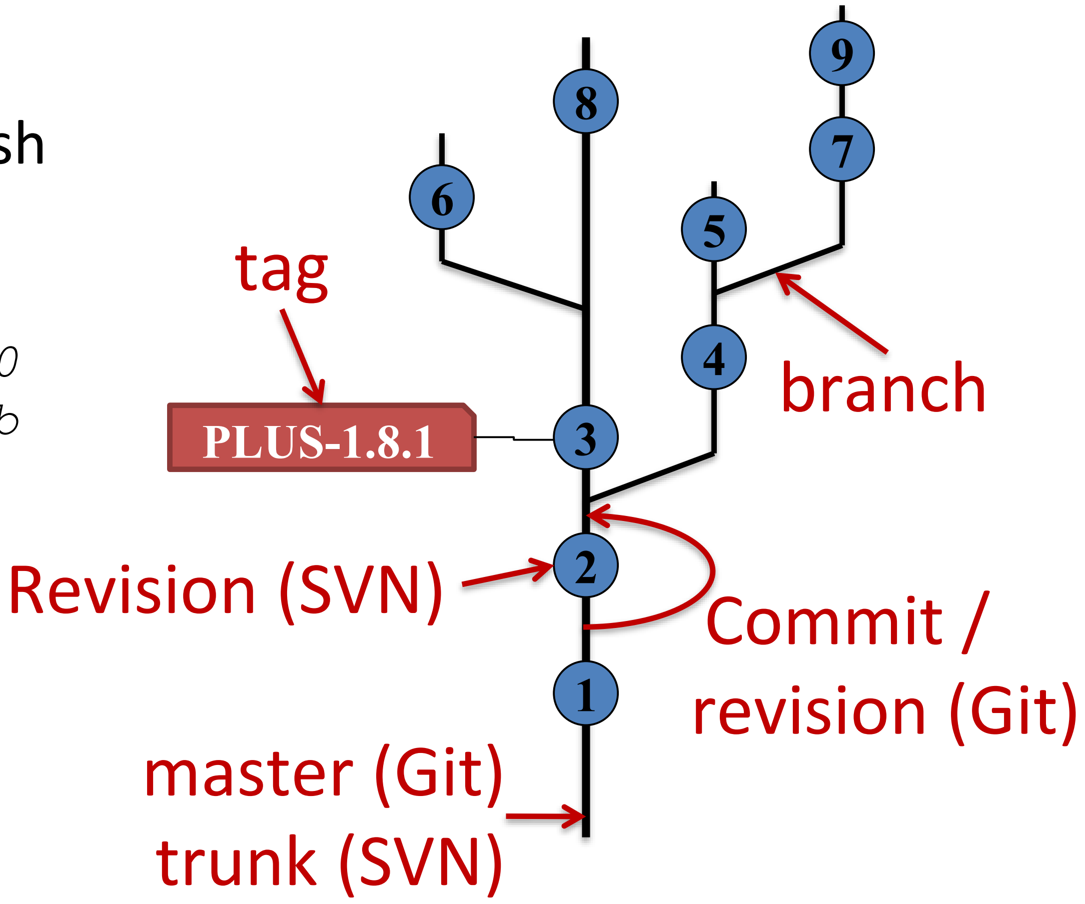
“It is necessary only when there are multiple developers“

FAIL

- ⚠ Organized storage of multiple file versions
- ⚠ Records history of all file changes (who modified, what, when, why)
- ⚠ Convenient user interface for comparing and merging different file versions
- ✓ Allows concurrent modifications by multiple people

Organized storage of multiple file versions

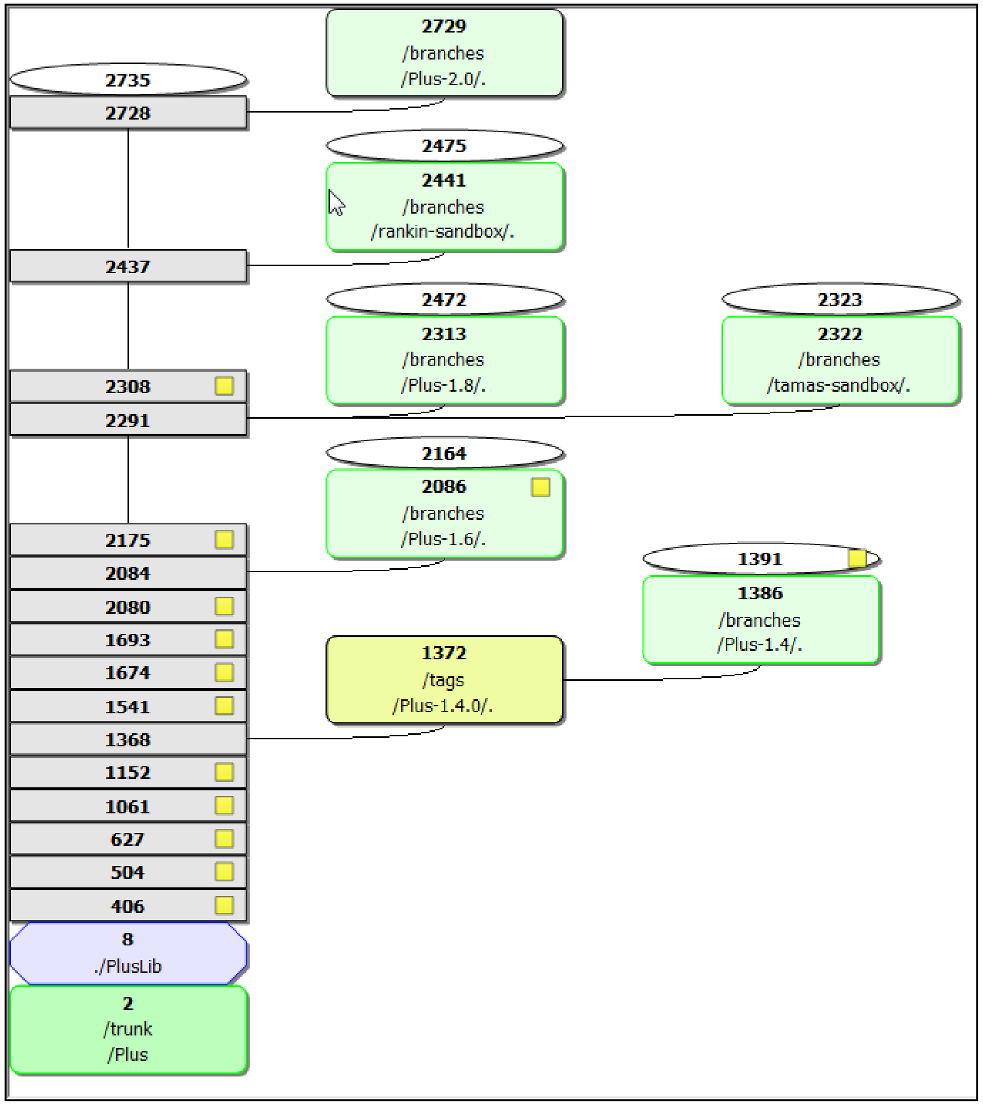
- Numbered revisions/commits:
  - Git: cryptographic hash (SHA-1, 160 bit, 40 character hex string)
    `ca8fbe9085c4f67be170 528b53b8ac3f64dd5acb`
  - SVN: monotonously increasing integers
    `1, 2, 3, …`
- Tags
- Tree structure



Organized storage of multiple file versions

Revision graph

(version tree)



2729 /branches /Plus-2.0/.
2735
2728
2475
2441 /branches /rankin-sandbox/.
2437
2472
2313 /branches /Plus-1.8/.
2323
2322 /branches /tamas-sandbox/.
2308
2291
2164
2086 /branches /Plus-1.6/.
2175
2084
1391
1386 /branches /Plus-1.4/.
2080
1693
1372 /tags /Plus-1.4.0/.
1674
1541
1368
1152
1061
627
504
406
8 ./PlusLib
2 /trunk /Plus

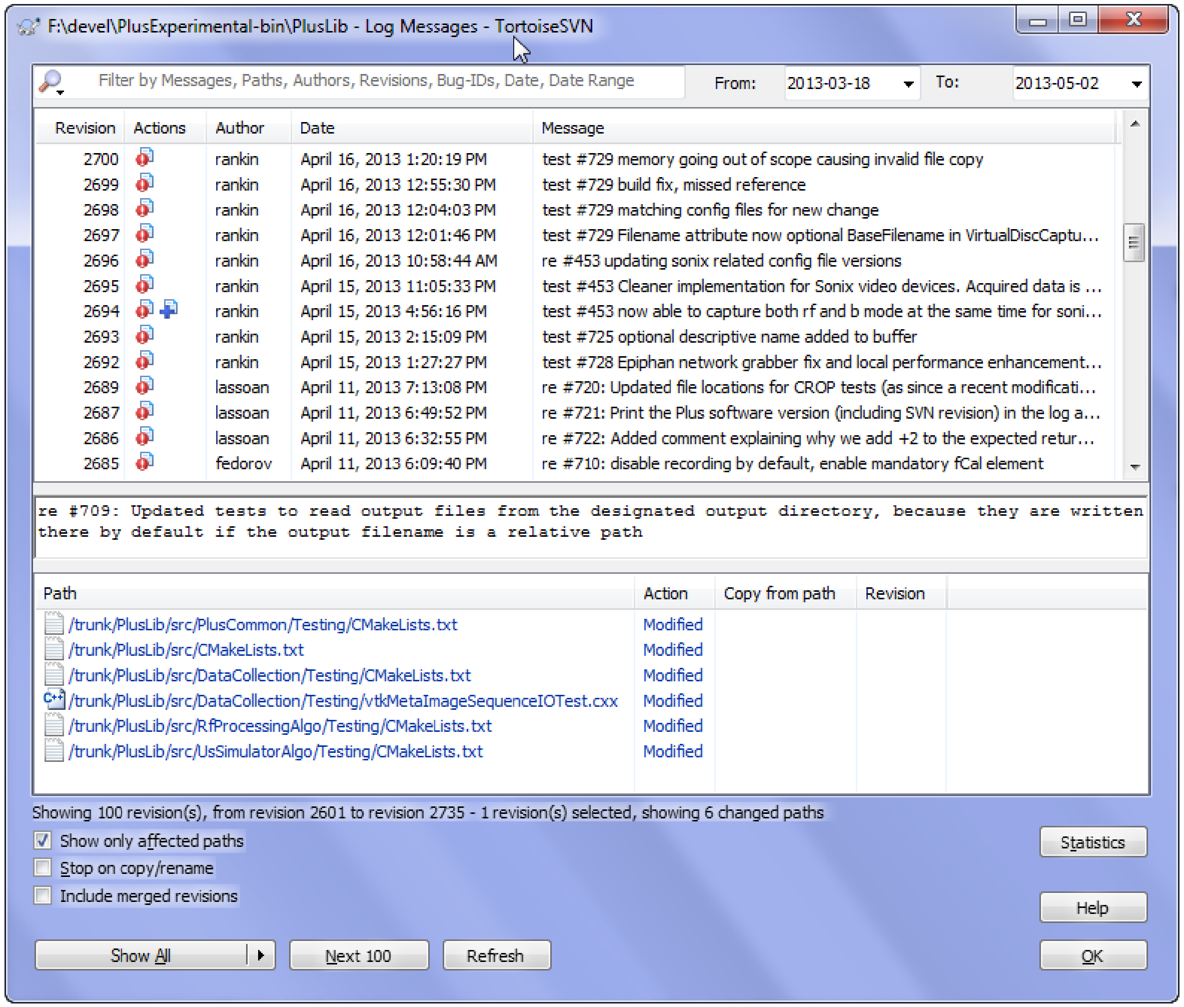
Records history of all file changes

Recorded for each modification:

- What?
- When?
- Who?
- Why?

auto

Show log



F:\devel\PlusExperimental-bin\PlusLib - Log Messages - TortoiseSVN

Filter by Messages, Paths, Authors, Revisions, Bug-IDs, Date, Date Range — From: 2013-03-18 — To: 2013-05-02

Revision	Actions	Author	Date	Message
2700		rankin	April 16, 2013 1:20:19 PM	test #729 memory going out of scope causing invalid file copy
2699		rankin	April 16, 2013 12:55:30 PM	test #729 build fix, missed reference
2698		rankin	April 16, 2013 12:04:03 PM	test #729 matching config files for new change
2697		rankin	April 16, 2013 12:01:46 PM	test #729 Filename attribute now optional BaseFilename in VirtualDiscCaptu...
2696		rankin	April 16, 2013 10:58:44 AM	re #453 updating sonix related config file versions
2695		rankin	April 15, 2013 11:05:33 PM	test #453 Cleaner implementation for Sonix video devices. Acquired data is ...
2694		rankin	April 15, 2013 4:56:16 PM	test #453 now able to capture both rf and b mode at the same time for soni...
2693		rankin	April 15, 2013 2:15:09 PM	test #725 optional descriptive name added to buffer
2692		rankin	April 15, 2013 1:27:27 PM	test #728 Epiphan network grabber fix and local performance enhancement...
2689		lassoan	April 11, 2013 7:13:08 PM	re #720: Updated file locations for CROP tests (as since a recent modificati...
2687		lassoan	April 11, 2013 6:49:52 PM	re #721: Print the Plus software version (including SVN revision) in the log a...
2686		lassoan	April 11, 2013 6:32:55 PM	re #722: Added comment explaining why we add +2 to the expected retur...
2685		fedorov	April 11, 2013 6:09:40 PM	re #710: disable recording by default, enable mandatory fCal element

```
re #709: Updated tests to read output files from the designated output directory, because they are written
there by default if the output filename is a relative path
```

Path	Action	Copy from path	Revision
/trunk/PlusLib/src/PlusCommon/Testing/CMakeLists.txt	Modified		
/trunk/PlusLib/src/CMakeLists.txt	Modified		
/trunk/PlusLib/src/DataCollection/Testing/CMakeLists.txt	Modified		
/trunk/PlusLib/src/DataCollection/Testing/vtkMetaImageSequenceIOTest.cxx	Modified		
/trunk/PlusLib/src/RfProcessingAlgo/Testing/CMakeLists.txt	Modified		
/trunk/PlusLib/src/UsSimulatorAlgo/Testing/CMakeLists.txt	Modified		

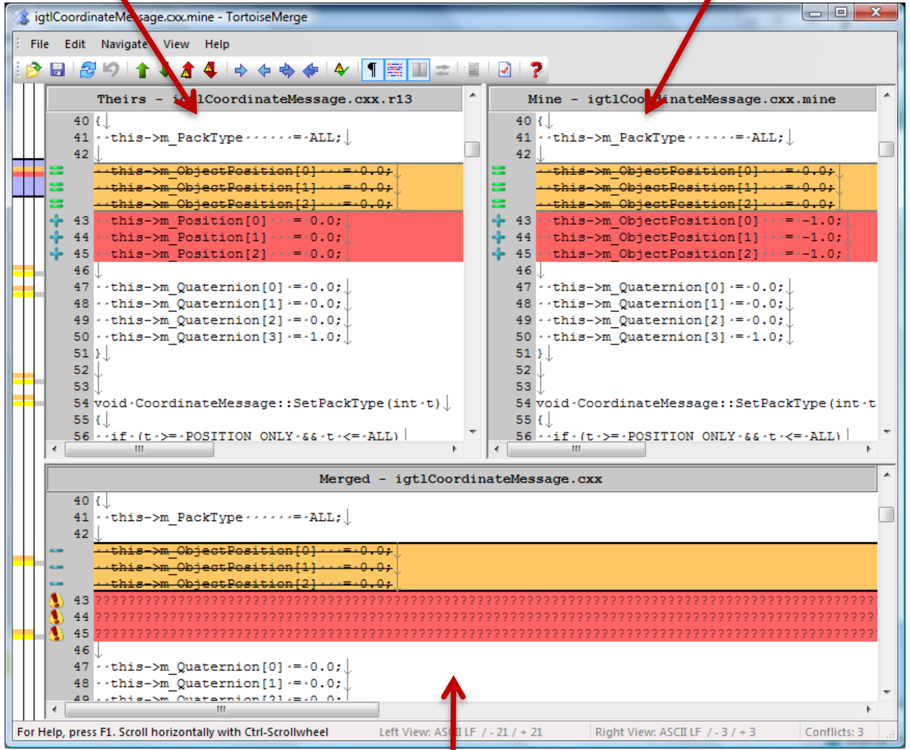
Showing 100 revision(s), from revision 2601 to revision 2735 - 1 revision(s) selected, showing 6 changed paths

- [✓] Show only affected paths
- [ ] Stop on copy/rename
- [ ] Include merged revisions

User interface for compare and merge

Their modifications

My modifications

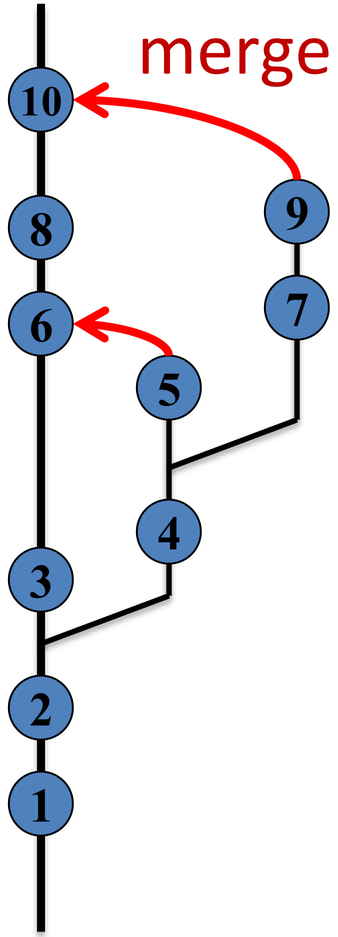

igtlCoordinateMessage.cxx.mine - TortoiseMerge
File Edit Navigate View Help

```
Theirs - igtlCoordinateMessage.cxx.r13
{
  this->m_PackType      = ALL;

~~  this->m_ObjectPosition[0]    = 0.0;~~
~~  this->m_ObjectPosition[1]    = 0.0;~~
~~  this->m_ObjectPosition[2]    = 0.0;~~
  this->m_Position[0]    = 0.0;
  this->m_Position[1]    = 0.0;
  this->m_Position[2]    = 0.0;

  this->m_Quaternion[0] = 0.0;
  this->m_Quaternion[1] = 0.0;
  this->m_Quaternion[2] = 0.0;
  this->m_Quaternion[3] = 1.0;
}


void CoordinateMessage::SetPackType(int t)
{
  if (t >= POSITION_ONLY && t <= ALL)
```

```
Mine - igtlCoordinateMessage.cxx.mine
{
  this->m_PackType      = ALL;

~~  this->m_ObjectPosition[0]    = 0.0;~~
~~  this->m_ObjectPosition[1]    = 0.0;~~
~~  this->m_ObjectPosition[2]    = 0.0;~~
  this->m_ObjectPosition[0]    = -1.0;
  this->m_ObjectPosition[1]    = -1.0;
  this->m_ObjectPosition[2]    = -1.0;

  this->m_Quaternion[0] = 0.0;
  this->m_Quaternion[1] = 0.0;
  this->m_Quaternion[2] = 0.0;
  this->m_Quaternion[3] = 1.0;
}


void CoordinateMessage::SetPackType(int t
{
  if (t >= POSITION_ONLY && t <= ALL)
```

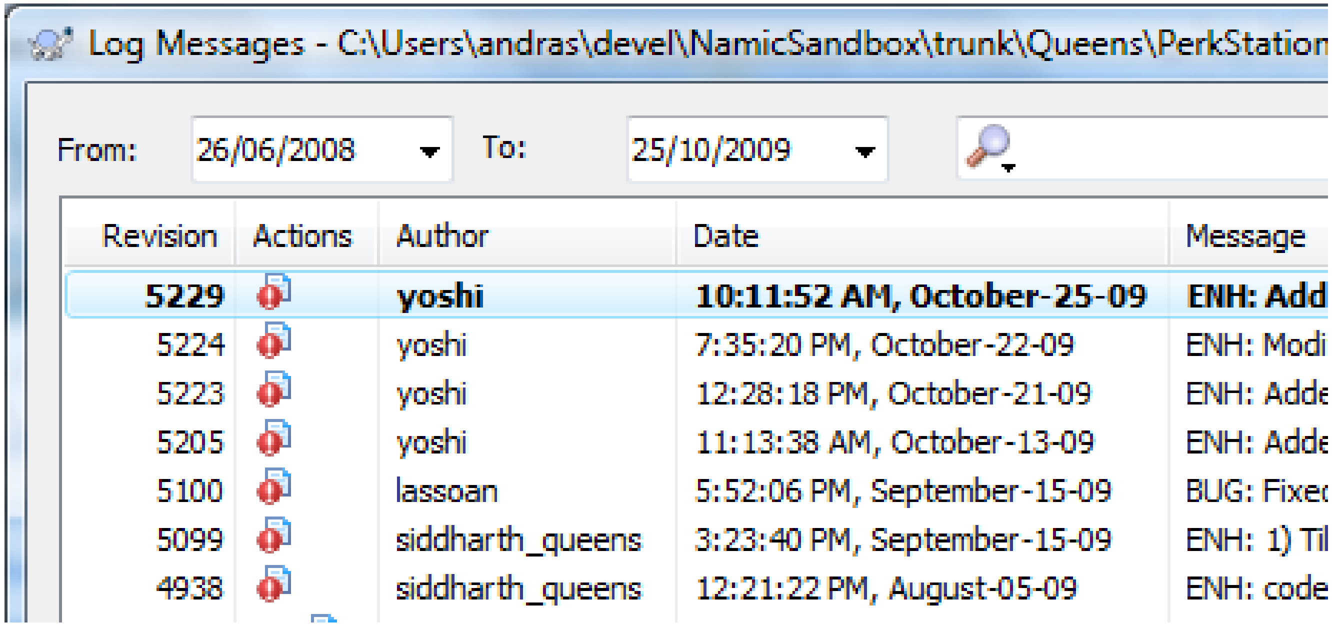
```
Merged - igtlCoordinateMessage.cxx
{
  this->m_PackType      = ALL;

~~  this->m_ObjectPosition[0]    = 0.0;~~
~~  this->m_ObjectPosition[1]    = 0.0;~~
~~  this->m_ObjectPosition[2]    = 0.0;~~
????????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????????
????????????????????????????????????????????????????????????????????????????????????

  this->m_Quaternion[0] = 0.0;
  this->m_Quaternion[1] = 0.0;
```

```
For Help, press F1. Scroll horizontally with Ctrl-Scrollwheel    Left View: ASCII LF  / - 21 / + 21    Right View: ASCII LF  / - 3 / + 3    Conflicts: 3
```



Concurrent modifications by multiple people

- One common server stores all versions
- Automatic merge of trivial changes (if a line is not modified by multiple people)



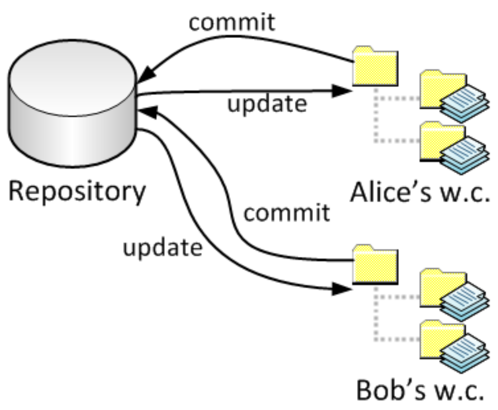
Log Messages - C:\Users\andras\devel\NamicSandbox\trunk\Queens\PerkStation

From: 26/06/2008 To: 25/10/2009

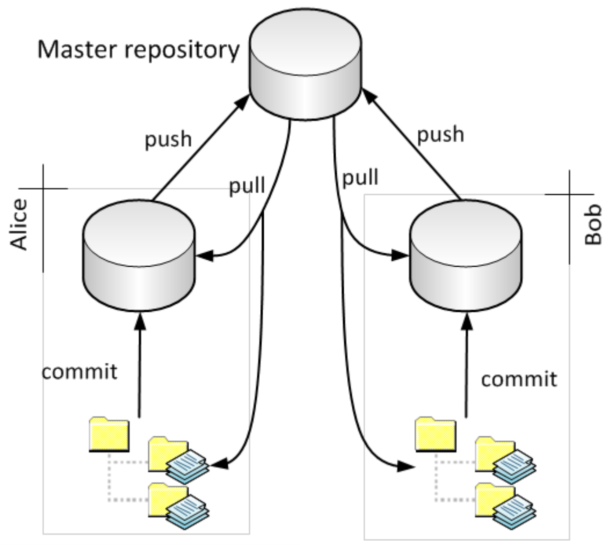
Revision	Actions	Author	Date	Message
5229		**yoshi**	**10:11:52 AM, October-25-09**	**ENH: Add**
5224		yoshi	7:35:20 PM, October-22-09	ENH: Modi
5223		yoshi	12:28:18 PM, October-21-09	ENH: Adde
5205		yoshi	11:13:38 AM, October-13-09	ENH: Adde
5100		lassoan	5:52:06 PM, September-15-09	BUG: Fixe
5099		siddharth_queens	3:23:40 PM, September-15-09	ENH: 1) Til
4938		siddharth_queens	12:21:22 PM, August-05-09	ENH: code

Which one to use? – *git* has won

- Commercial: very good, but expensive (ClearCase, Perforce, Team Foundation Server, etc.)
- Non-commercial: good enough, free



Centralized: **Subversion (SVN)**
=> simpler to use



Distributed: **Git**, Mercurial
=> more flexible, for very distributed projects

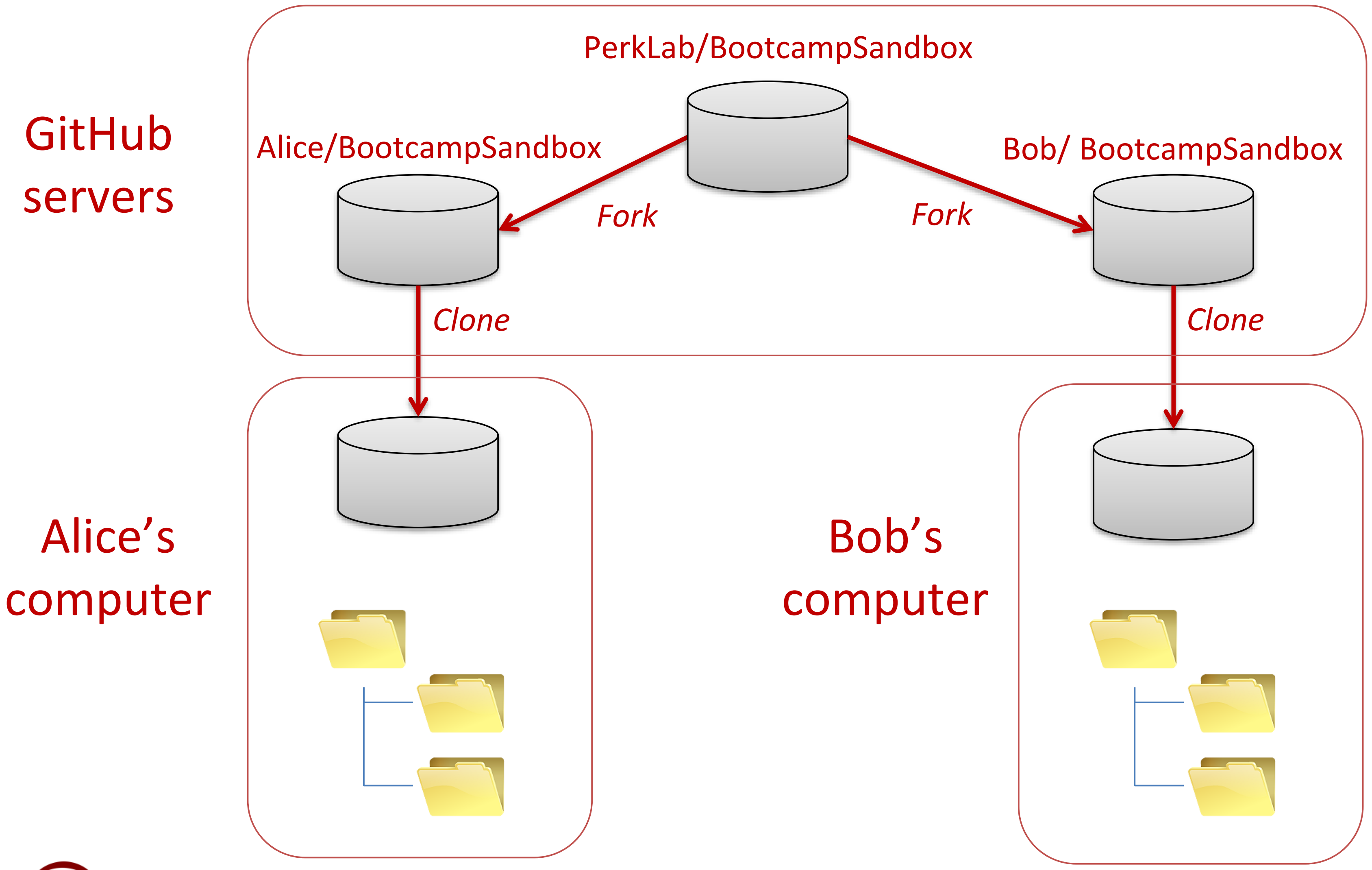
http://pages.cs.wisc.edu/~driscoll/software/vcs/

Setup Git

- Install Git (command-line tool): https://git-scm.com/download/win
- Install TortoiseGit (GUI client): http://tortoisesvn.net/downloads

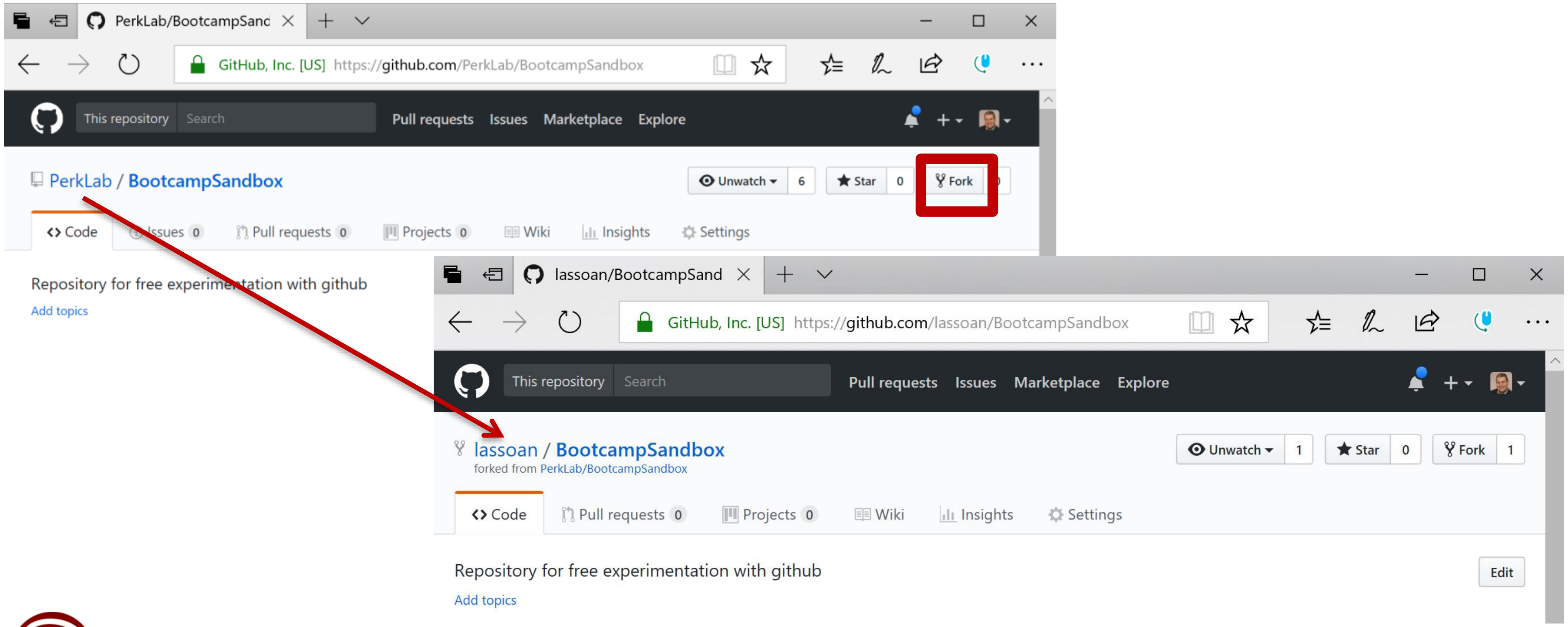
- Optional: for private repositories with SSH authentication, install putty: http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

Create a modifiable local copy of a repository

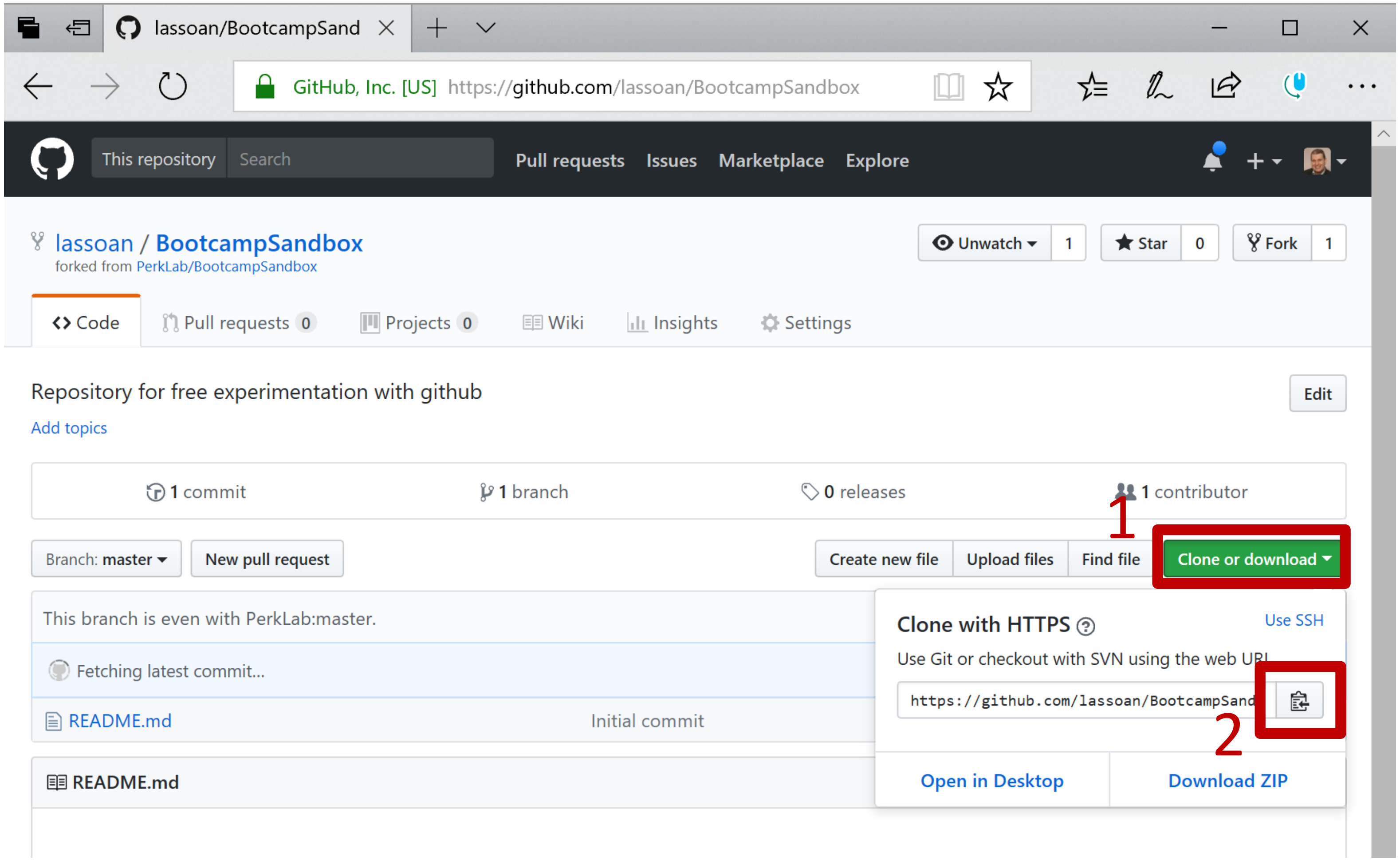


Fork: create copy of official repository

- Open repository webpage: https://github.com/PerkLab/BootcampSandbox
- *Fork* the repository: clone the repository so that you can make changes freely


PerkLab/BootcampSand
GitHub, Inc. [US] https://github.com/PerkLab/BootcampSandbox
This repository
Search
Pull requests
Issues
Marketplace
Explore
PerkLab / BootcampSandbox
Unwatch 6
Star 0
Fork
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
Repository for free experimentation with github
Add topics
lassoan/BootcampSand
GitHub, Inc. [US] https://github.com/lassoan/BootcampSandbox
lassoan / BootcampSandbox
forked from PerkLab/BootcampSandbox
Unwatch 1
Star 0
Fork 1
Edit

Clone: create a local copy of your repository



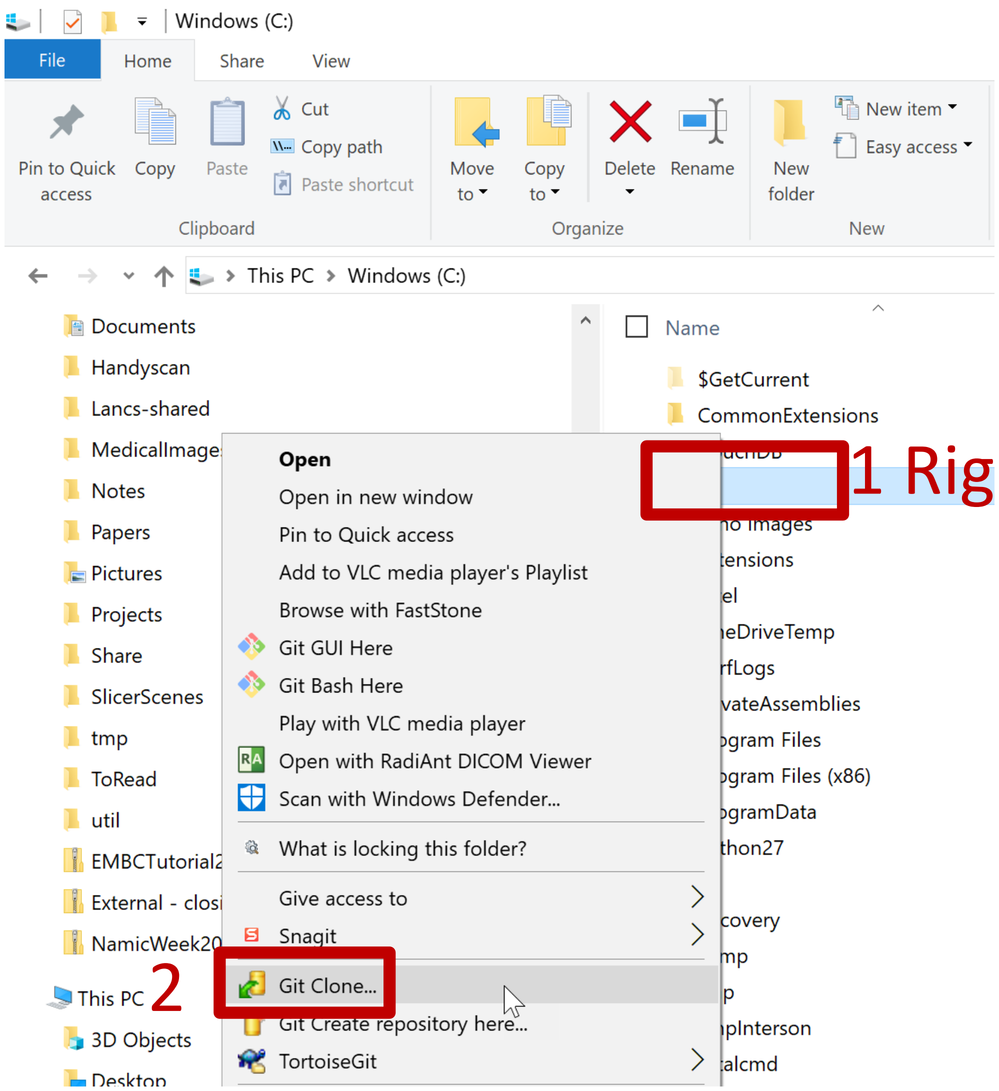
Repository for free experimentation with github

This branch is even with PerkLab:master.

Fetching latest commit...

Use Git or checkout with SVN using the web URL.

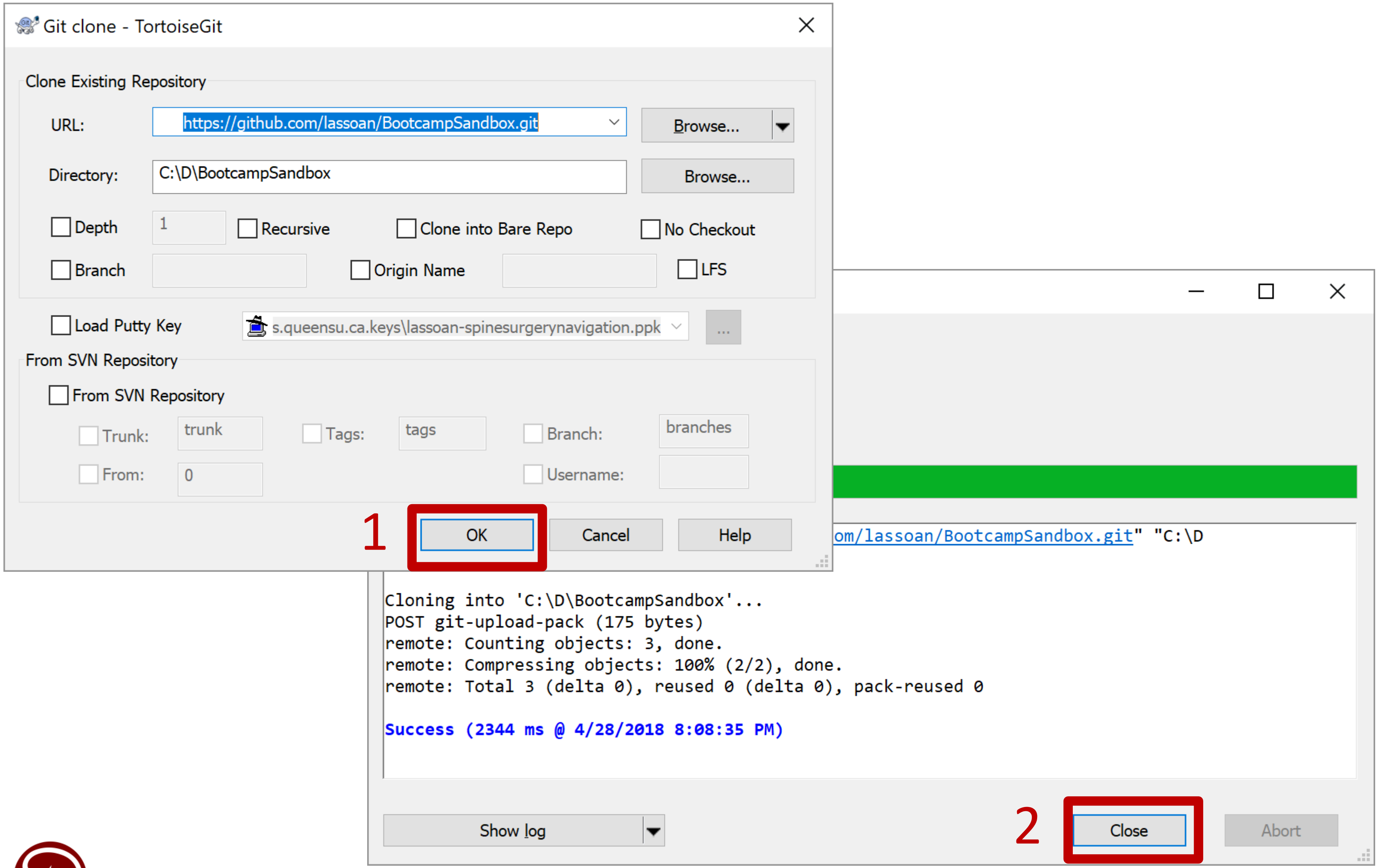
Clone: create a local copy of your repository



1 Right-click on a folder

2

Clone: create a local copy of your repository



Git clone - TortoiseGit

Clone Existing Repository

URL: https://github.com/lassoan/BootcampSandbox.git

Directory: C:\D\BootcampSandbox

☐ Depth 1 ☐ Recursive ☐ Clone into Bare Repo ☐ No Checkout

☐ Branch ☐ Origin Name ☐ LFS

☐ Load Putty Key s.queensu.ca.keys\lassoan-spinesurgerynavigation.ppk

From SVN Repository

☐ From SVN Repository

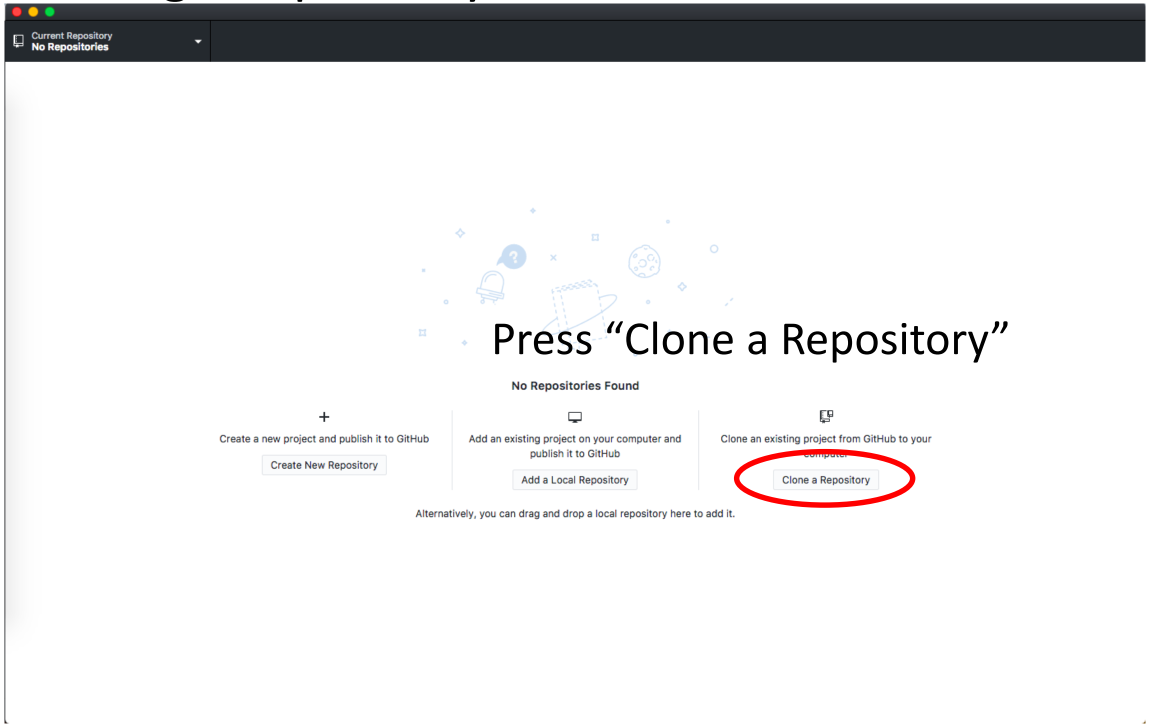
☐ Trunk: trunk ☐ Tags: tags ☐ Branch: branches

☐ From: 0 ☐ Username:

```
om/lassoan/BootcampSandbox.git" "C:\D

Cloning into 'C:\D\BootcampSandbox'...
POST git-upload-pack (175 bytes)
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Success (2344 ms @ 4/28/2018 8:08:35 PM)
```

git clone (GitHub Desktop)

- a.k.a. downloading a repository



Press "Clone a Repository"

No Repositories Found

Create a new project and publish it to GitHub

Add an existing project on your computer and publish it to GitHub

Clone an existing project from GitHub to your computer

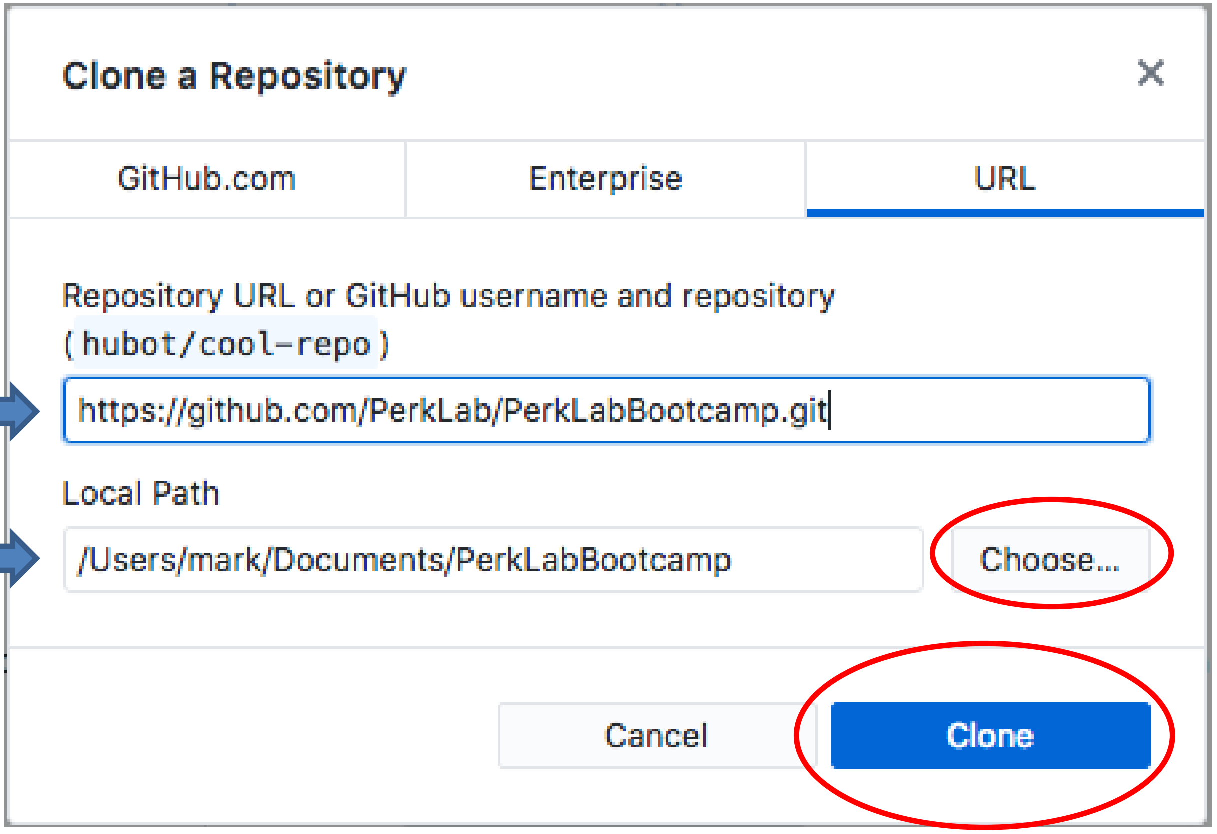
Alternatively, you can drag and drop a local repository here to add it.

- Or, you can use File->Clone Repository

git clone (GitHub Desktop)

Input URL to clone from

Choose where on your PC to clone it to



Clone a Repository

Repository URL or GitHub username and repository (`hubot/cool-repo`)

https://github.com/PerkLab/PerkLabBootcamp.git

Local Path

/Users/mark/Documents/PerkLabBootcamp

git clone (terminal)

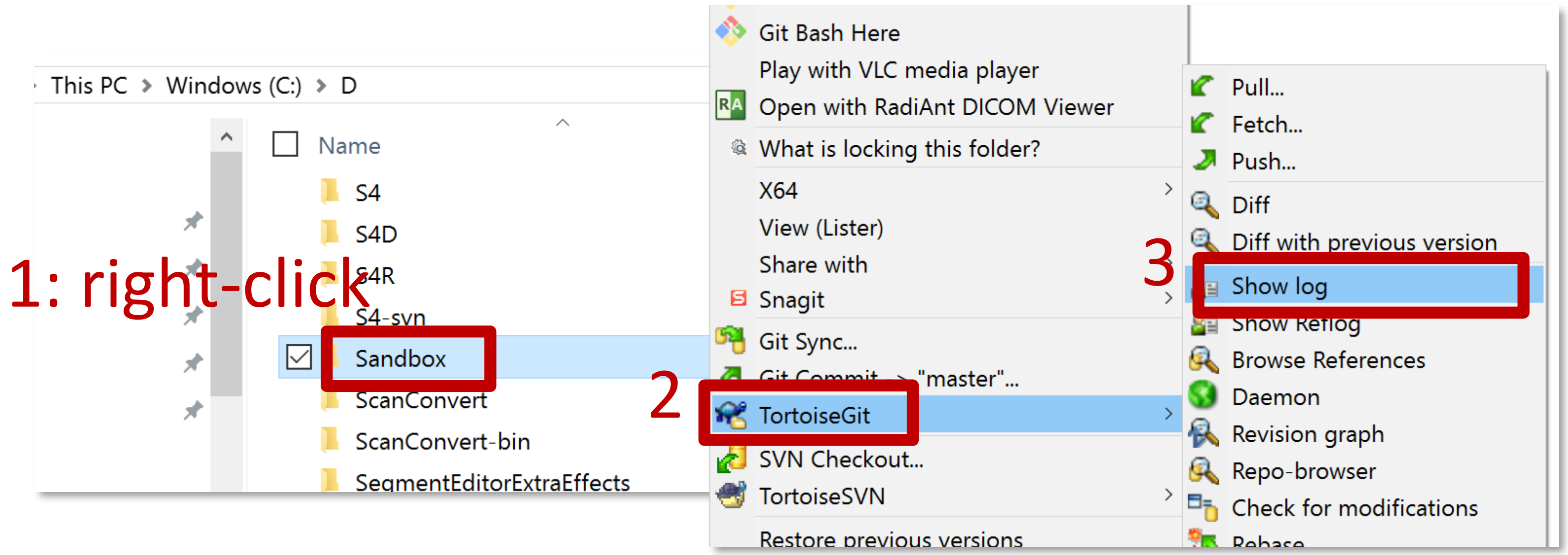
- Use "cd" (change directory) to navigate to where you'd like to clone the repository
- Run the command (changing the url for a different repository):

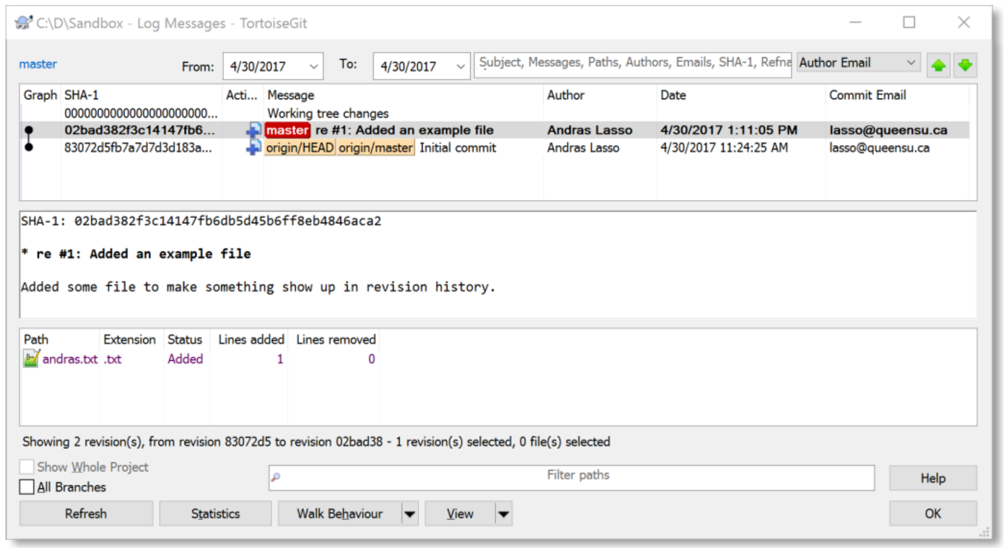
```
git clone <url> <destination>
```

```
git clone https://github.com/PerkLab/PerkLabBootcamp .
```

```
(“.” on Unix means current directory)
```

Review change history




C:\D\Sandbox - Log Messages - TortoiseGit

master From: 4/30/2017 To: 4/30/2017 Subject, Messages, Paths, Authors, Emails, SHA-1, Refna Author Email

Graph	SHA-1	Acti...	Message	Author	Date	Commit Email
	0000000000000000000000...		Working tree changes			
●	02bad382f3c14147fb6...		master re #1: Added an example file	Andras Lasso	4/30/2017 1:11:05 PM	lasso@queensu.ca
●	83072d5fb7a7d7d3d183a...		origin/HEAD origin/master Initial commit	Andras Lasso	4/30/2017 11:24:25 AM	lasso@queensu.ca

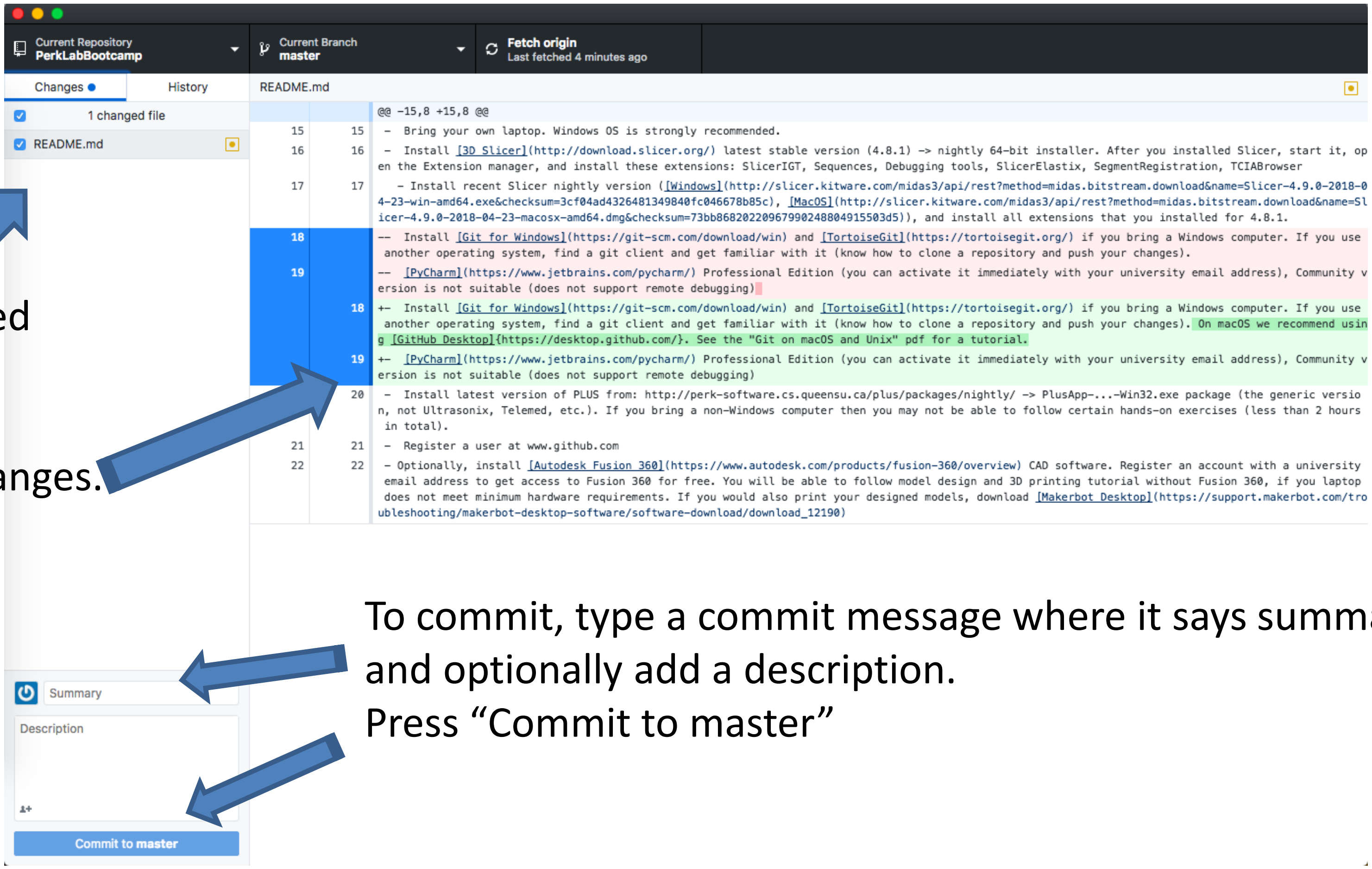
```
SHA-1: 02bad382f3c14147fb6db5d45b6ff8eb4846aca2

* re #1: Added an example file

Added some file to make something show up in revision history.
```

Path	Extension	Status	Lines added	Lines removed
andras.txt	.txt	Added	1	0

Showing 2 revision(s), from revision 83072d5 to revision 02bad38 - 1 revision(s) selected, 0 file(s) selected

git commit (GitHub Desktop)

- a.k.a “saving” your work



```
@@ -15,8 +15,8 @@
 -  Bring your own laptop. Windows OS is strongly recommended.
 -  Install [3D Slicer](http://download.slicer.org/) latest stable version (4.8.1) -> nightly 64-bit installer. After you installed Slicer, start it, op
en the Extension manager, and install these extensions: SlicerIGT, Sequences, Debugging tools, SlicerElastix, SegmentRegistration, TCIABrowser
   - Install recent Slicer nightly version ([Windows](http://slicer.kitware.com/midas3/api/rest?method=midas.bitstream.download&name=Slicer-4.9.0-2018-0
4-23-win-amd64.exe&checksum=3cf04ad4326481349840fc046678b85c), [MacOS](http://slicer.kitware.com/midas3/api/rest?method=midas.bitstream.download&name=Sl
icer-4.9.0-2018-04-23-macosx-amd64.dmg&checksum=73bb86820220967990248804915503d5)), and install all extensions that you installed for 4.8.1.
--  Install [Git for Windows](https://git-scm.com/download/win) and [TortoiseGit](https://tortoisegit.org/) if you bring a Windows computer. If you use
 another operating system, find a git client and get familiar with it (know how to clone a repository and push your changes).
--  [PyCharm](https://www.jetbrains.com/pycharm/) Professional Edition (you can activate it immediately with your university email address), Community v
ersion is not suitable (does not support remote debugging)
+-  Install [Git for Windows](https://git-scm.com/download/win) and [TortoiseGit](https://tortoisegit.org/) if you bring a Windows computer. If you use
 another operating system, find a git client and get familiar with it (know how to clone a repository and push your changes). On macOS we recommend usin
g [GitHub Desktop]{https://desktop.github.com/}. See the "Git on macOS and Unix" pdf for a tutorial.
+-  [PyCharm](https://www.jetbrains.com/pycharm/) Professional Edition (you can activate it immediately with your university email address), Community v
ersion is not suitable (does not support remote debugging)
 -  Install latest version of PLUS from: http://perk-software.cs.queensu.ca/plus/packages/nightly/ -> PlusApp-...-Win32.exe package (the generic versio
n, not Ultrasonix, Telemed, etc.). If you bring a non-Windows computer then you may not be able to follow certain hands-on exercises (less than 2 hours
 in total).
 -  Register a user at www.github.com
 - Optionally, install [Autodesk Fusion 360](https://www.autodesk.com/products/fusion-360/overview) CAD software. Register an account with a university
 email address to get access to Fusion 360 for free. You will be able to follow model design and 3D printing tutorial without Fusion 360, if you laptop
 does not meet minimum hardware requirements. If you would also print your designed models, download [Makerbot Desktop](https://support.makerbot.com/tro
ubleshooting/makerbot-desktop-software/software-download/download_12190)
```

Changed files, click to see changes.

To commit, type a commit message where it says summary, and optionally add a description.
Press “Commit to master”

git commit (terminal)

- First we need to see our changes - use:

```
git diff
```

- Now we want to stage our changes to be committed. To add everything use:

```
git add –A
```

- Or, to add only specific files, use:

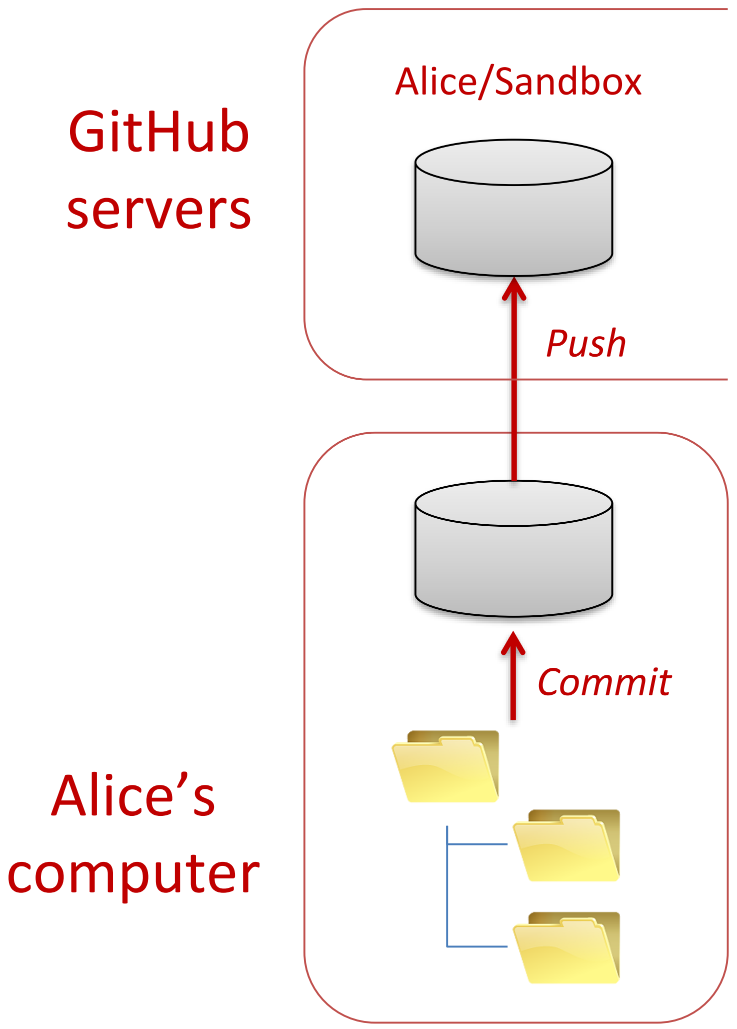
```
git add <filename>
```

- Lastly, we do our commit

```
git commit –m "<commit-message-here>"
```

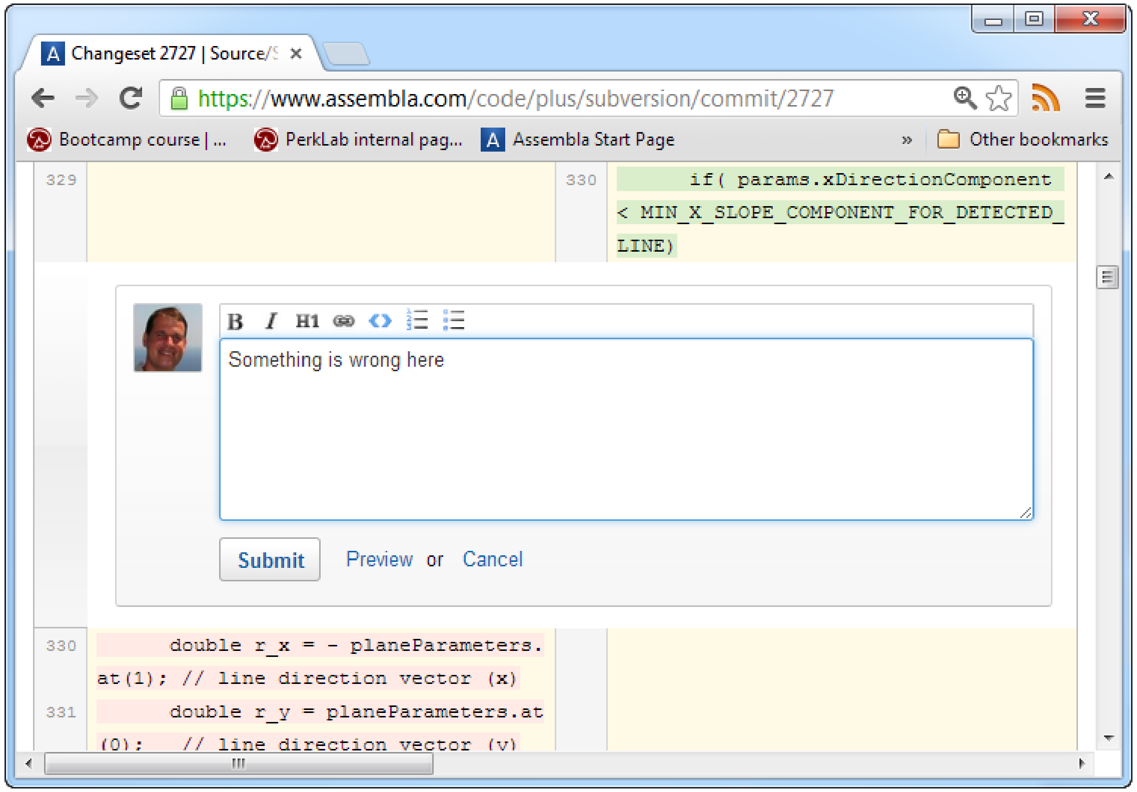
Pull request: get your changes merged into the official repository

- Add/modify/delete files
- Test your modifications (compilation is OK, no crash, etc.)
- Right-click on the revision controlled directory
- Click *Git Commit -> "master"*
- Review your changes
- Enter a *commit message*
  - One-line summary: re #123
  - describe **why** was modified
  - include the issue/ticket number to make the commit show up in the ticket comment list
- Click *Push* to upload changes to GitHub servers to make it available to your collborators



Code review

- Go to the list of commits in Assembla
- Click on the green [+] icon in the source code
- The author will be notified

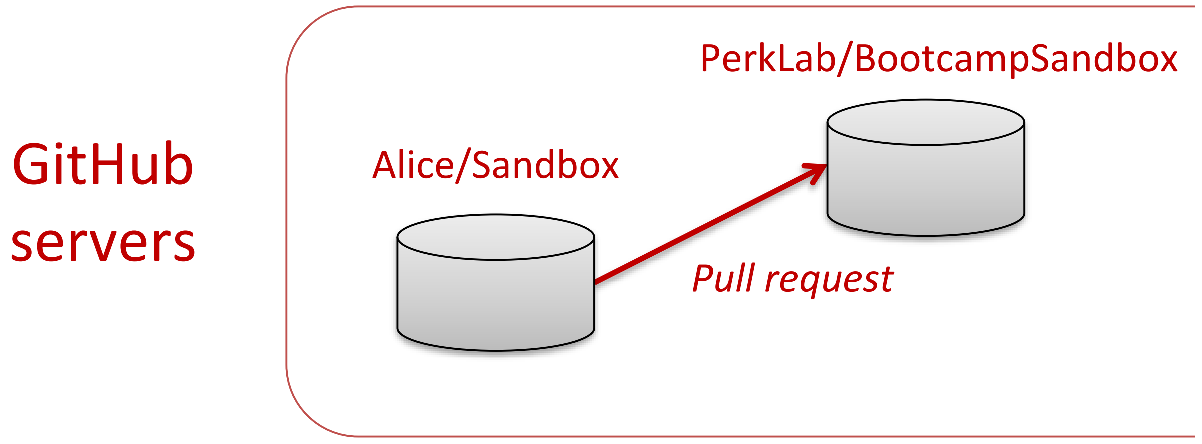


```
if( params.xDirectionComponent < MIN_X_SLOPE_COMPONENT_FOR_DETECTED_LINE)
```

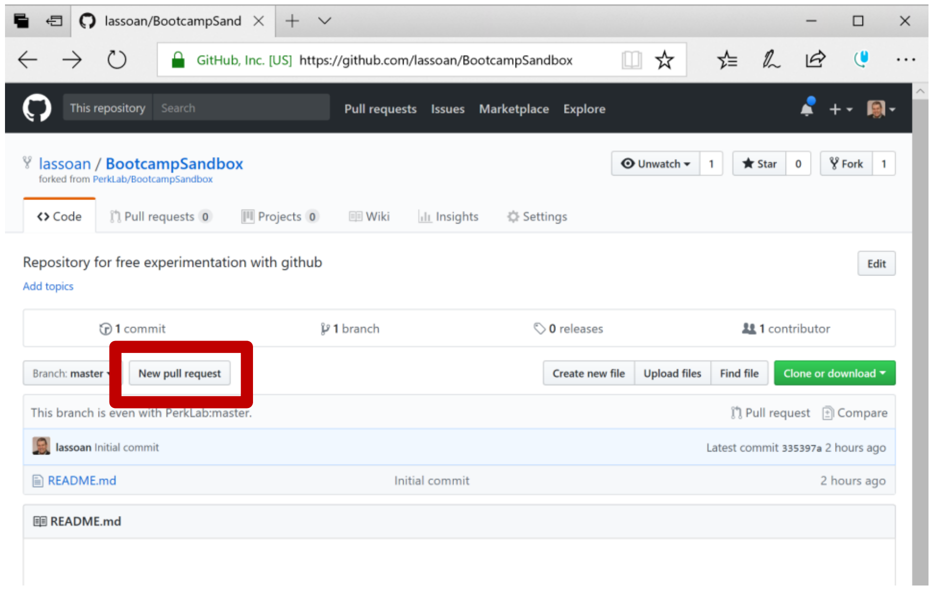
Something is wrong here

```
double r_x = - planeParameters.at(1); // line direction vector (x)
double r_y = planeParameters.at(0);   // line direction vector (y)
```

Pull request: get your changes merged into the official repository



- Open your repository page on GitHub
- Click *New pull request*
- Accept all default parameters
- Provide description
- Click *Create pull request*
- Wait for the official repository maintainer to merge your changes



Repository for free experimentation with github

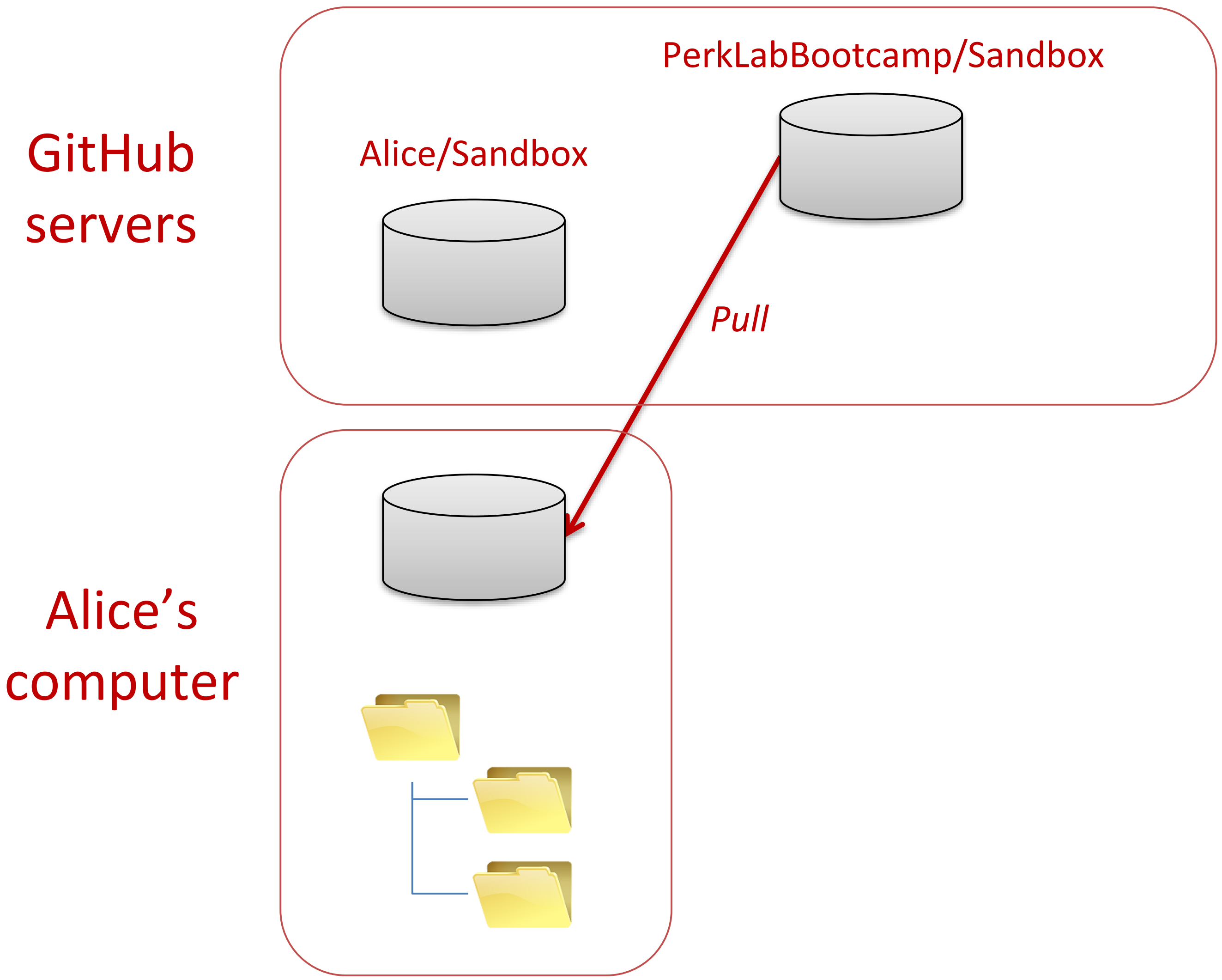
This branch is even with PerkLab:master.

lassoan Initial commit — Latest commit 335397a 2 hours ago

README.md — Initial commit — 2 hours ago

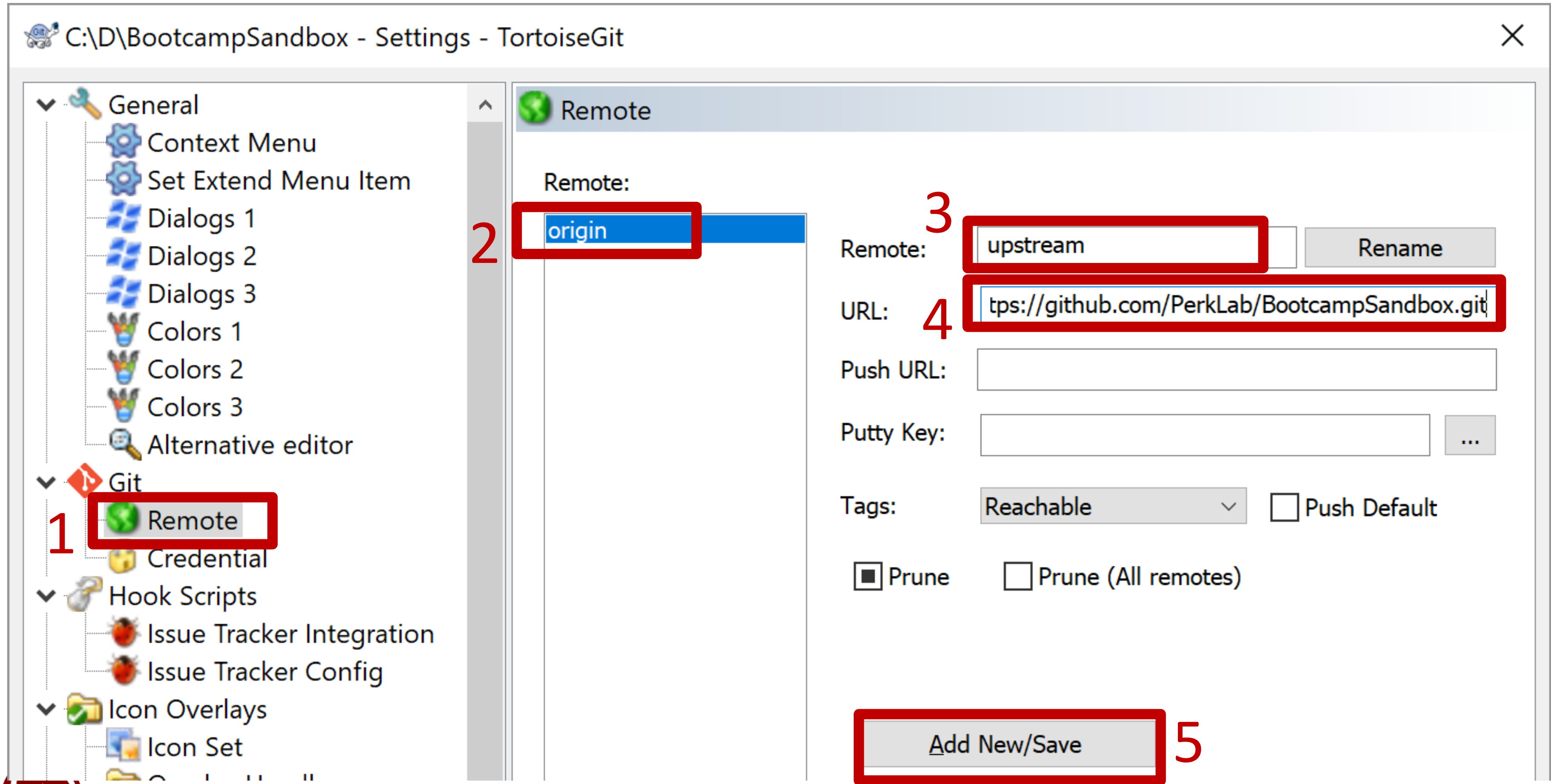
README.md

Pull: get latest changes from the official repository



Pull: get latest changes from the official repository

- Right-click on the revision controlled directory
- Click *TortoiseGit*, click *Settings…*



C:\D\BootcampSandbox - Settings - TortoiseGit

Remote

Remote: origin

Remote: upstream

URL: tps://github.com/PerkLab/BootcampSandbox.git

Push URL:

Putty Key:

Tags: Reachable

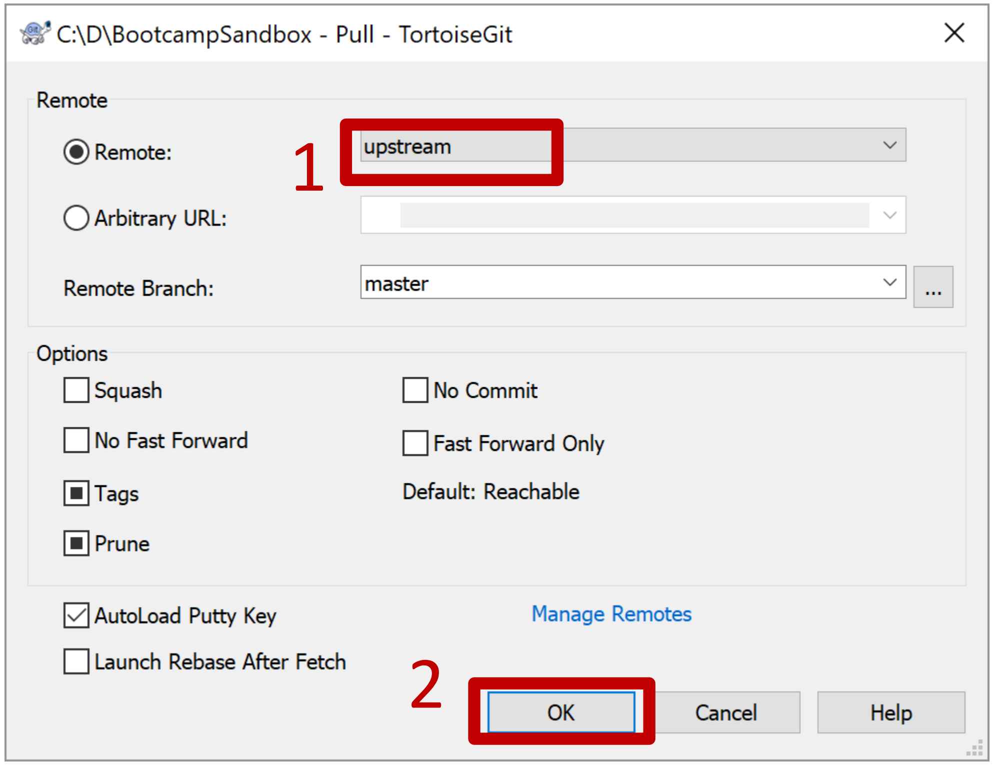
Push Default

Prune

Prune (All remotes)

Pull: get latest changes from the official repository

- *Pull*: *Fetch* and *Merge* changes
- Right-click on the revision controlled directory
- Click *TortoiseGit*, click *Pull...*



C:\D\BootcampSandbox - Pull - TortoiseGit

Remote

(●) Remote: upstream

( ) Arbitrary URL:

Remote Branch: master

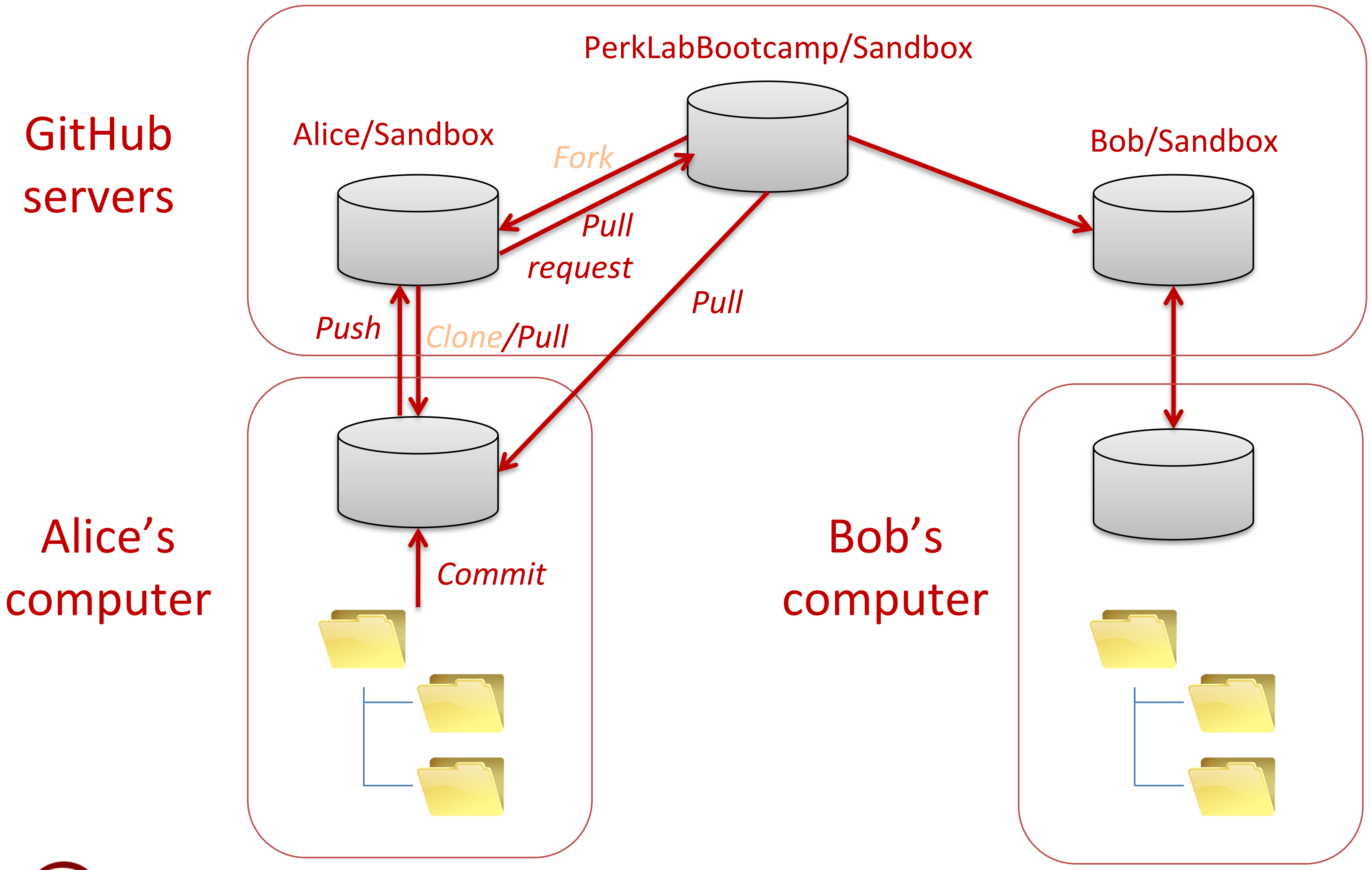
Options

- [ ] Squash
- [ ] No Commit
- [ ] No Fast Forward
- [ ] Fast Forward Only
- [■] Tags
- Default: Reachable
- [■] Prune
- [✓] AutoLoad Putty Key
- [ ] Launch Rebase After Fetch

Recommendations

- **Before committing** any code changes build all and run the automatic tests to make sure there are no regressions
- **Commit:**
  - One fix/enhancement at a time (and not multiple independent developments in one single commit) whenever it's possible
  - Commit comments: **Include the related issue number** (this will automatically link the changeset to the ticket) and describe what did you change (in past tense) and why. Example: e.g.,. in the format:
    `re #123: Added MyClass:MyObject method to allow doing something`
- **After committing** code changes have a look at the dashboard about half an hour later to make sure that all the automatic builds still pass. You can get an automatic notification from CDash about any errors that you introduced if you register yourself on the dashboard.
- Read all notification emails about ticket updates!
  - If you feel that you get too many emails then set project to "Not watching" – then you only get notified about @mentions and events that directly affect you. **Never just blindly ignore/delete any notification emails without reading it**.

Summary


GitHub servers
PerkLabBootcamp/Sandbox
Alice/Sandbox
Bob/Sandbox
Fork
Pull request
Pull
Push
Clone/Pull
Commit
Alice's computer
Bob's computer