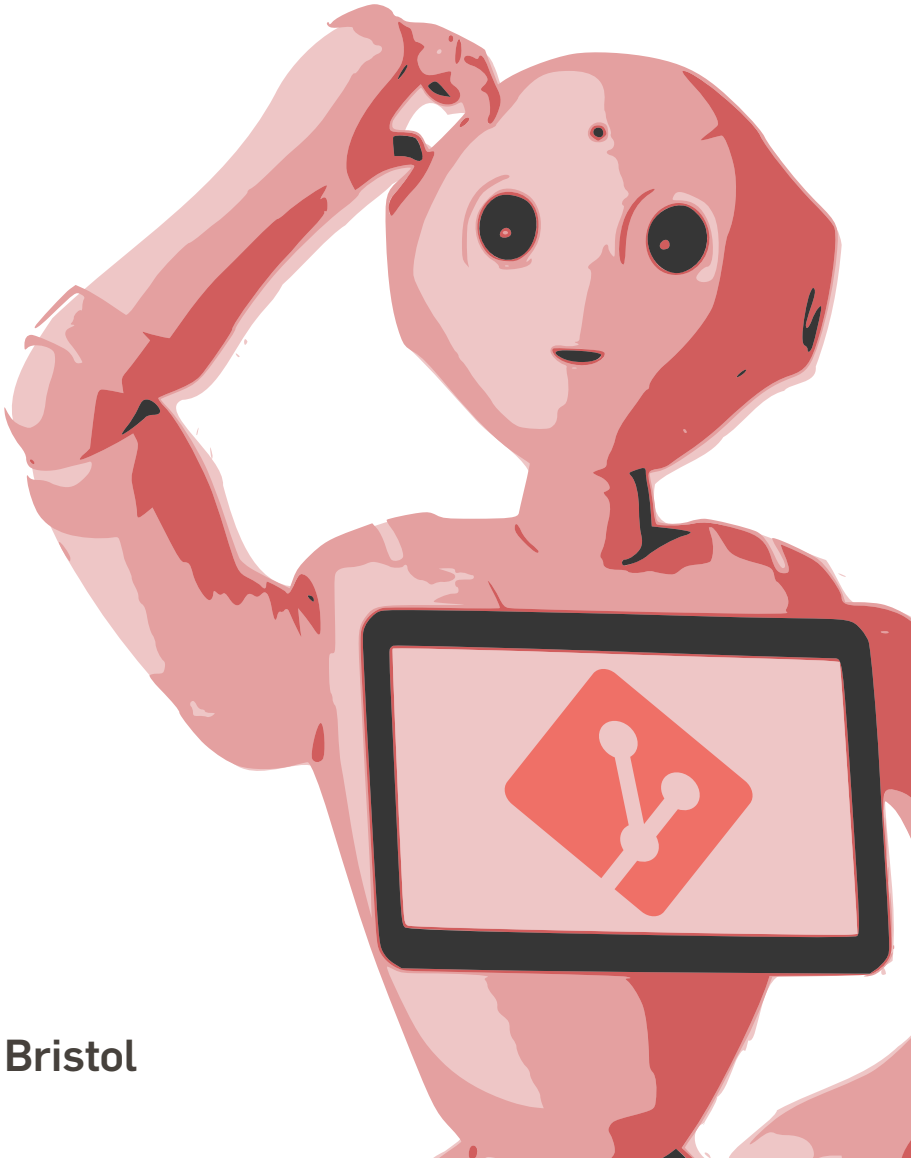
You can download the sources of this presentation here:

github.com/severin-lemaignan/lecture-software-engineering

Software Engineering

FARSCOPE workshops



Séverin Lemaignan

Bristol Robotics Lab
University of the West of England/University of Bristol

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package
- How to organise your code on your hard-drive

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package
- How to organise your code on your hard-drive
- What *Filesystem Standard Hierarchy* means

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package
- How to organise your code on your hard-drive
- What *Filesystem Standard Hierarchy* means
- What is markdown

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package
- How to organise your code on your hard-drive
- What ***Filesystem Standard Hierarchy*** means
- What is markdown
- The differences between the GPL, MIT, BSD,... licenses

TODAY'S OBJECTIVES

At the end of the day, you should know more about:

- code versioning with git
- What does "compiling code" ***really*** means (but you remember from last time, right?)
- The difference between a `dll` and a `lib`
- (and what are `dll`s)
- What is **CMake**
- What is a `deb` package and how to create one?
- How to write and distribute a Python package
- How to organise your code on your hard-drive
- What ***Filesystem Standard Hierarchy*** means
- What is markdown
- The differences between the GPL, MIT, BSD,... licenses
- More git: code sharing, conflict resolution, branching

TODAY'S OBJECTIVES

Looks daunting?

- one of the main challenges of software engineering is the lack of 'structured' approach to it. **Mostly know-how!**
- today's aim is to give you a broad overview + get the key terminology, so that you can successfully navigate this space
- experience (eg, Google + stackoverflow, mainly) will do the rest
- We will focus on Linux, as most of the concepts are more mature and better enforced in this developer friendly environment. However, to a large extend, **the same principles apply to any operating system & programming environment**

GIT

Re: Kernel SCM saga..

From: Linus Torvalds
Date: Thu Apr 07 2005 - 23:41:58 EST

- **Next message:** Evgeniy Polyakov: "Re: [Fwd: Re: connector is missing in 2.6.12-rc2-mm1]"
- **Previous message:** David S. Miller: "Re: [Fwd: Re: connector is missing in 2.6.12-rc2-mm1]"
- **In reply to:** Chris Wedgwood: "Re: Kernel SCM saga.."
- **Next in thread:** kfogel: "Re: Kernel SCM saga.."
- **Messages sorted by:** [date] [thread] [subject] [author]

On Thu, 7 Apr 2005, Chris Wedgwood wrote:
>
> *I'm playing with monotone right now. Superficially it looks like it*
> *has tons of gee-whiz neato stuff... however, it's *agonizingly* slow.*
> *I mean glacial. A heavily sedated sloth with no legs is probably*
> *faster.*

Yes. The silly thing is, at least in my local tests it doesn't actually seem to be _doing_ anything while it's slow (there are no system calls except for a few memory allocations and de-allocations). It seems to have some exponential function on the number of pathnames involved etc.

I'm hoping they can fix it, though. The basic notions do not sound wrong.

In the meantime (and because monotone really _is_ that slow), here's a quick challenge for you, and any crazy hacker out there: if you want to play with something _really_ nasty (but also very _very_ fast), take a look at kernel.org:/pub/linux/kernel/people/torvalds/.

First one to send me the changelog tree of sparse-git (and a tool to commit and push/pull further changes) gets a gold star, and an honorable mention. I've put a hell of a lot of clues in there (*).

I've worked on it (and little else) for the last two days. Time for somebody else to tell me I'm crazy.

Linus

(*) It should be easier than it sounds. The database is designed so that you can do the equivalent of a nonmerging (ie pure superset) push/pull with just plain rsync, so replication really should be that easy (if somewhat bandwidth-intensive due to the whole-file format)

Brian Harrys blog

Everything you want to know about Visual Studio ALM and Farming

The largest Git repo on the planet

05/24/2017 by Brian Harry MS // 59 Comments

3217 0

It's been 3 months since I first wrote about our efforts to scale Git to extremely large projects and teams with an effort we called "Git Virtual File System". As a reminder, GVFS, together with a set of enhancements to Git, enables Git to scale to VERY large repos by virtualizing both the .git folder and the working directory. Rather than download the entire repo and checkout all the files, it dynamically downloads only the portions you need based on what you use.

A lot has happened and I wanted to give you an update. Three months ago, GVFS was still a dream. I don't mean it didn't exist – we had a concrete implementation, but rather, it was unproven. We had validated on some big repos but we hadn't rolled it out to any meaningful number of engineers so we had only conviction that it was going to work. Now we have proof.

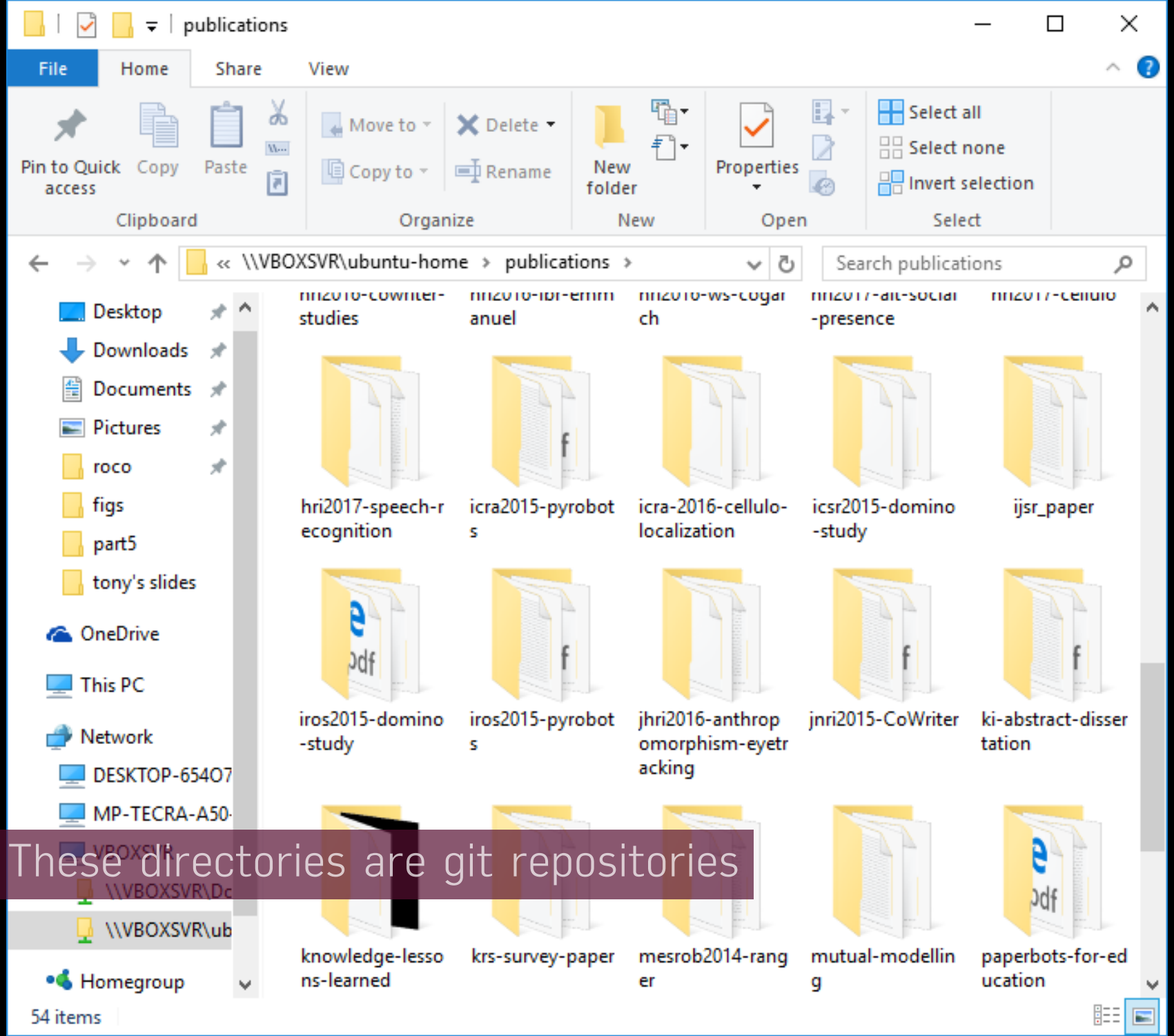
Today, I want to share our results. In addition, we're announcing the next steps in our GVFS journey for customers, including expanded open sourcing to start taking contributions and improving how it works for us at Microsoft, as well as for partners and customers.

Windows is live on Git

Over the past 3 months, we have largely completed the rollout of Git/GVFS to the Windows team at Microsoft.

As a refresher, the Windows code base is approximately 3.5M files and, when checked in to a Git repo, results in a repo of about 300GB. Further, the Windows team is about 4,000 engineers and the engineering system produces 1,760 daily "lab builds" across 440 branches in addition to thousands of pull request validation builds. All 3 of the dimensions (file count, repo size and activity), independently, provide daunting scaling challenges and taken together they make it unbelievably challenging to create a great experience. Before the move to Git, in Source Depot, it was spread across 40+ depots and we had a tool to manage operations that spanned them.

As of my writing 3 months ago, we had all the code in one Git repo, a few hundred engineers using it and a small fraction (<10%) of the daily build load. Since then, we have rolled out in waves across the engineering team.



These directories are git repositories

jnri2015-CoWriter

File Home Share View

Pin to Quick access | Copy | Paste | Move to | Copy to | Delete | Rename | New folder | Properties | Select all | Select none | Invert selection

Clipboard | Organize | New | Open | Select

« publications › jnri2015-CoWriter ›

Search jnri2015-CoWriter

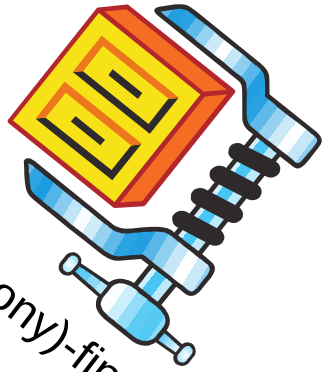
Desktop
Downloads
Documents
Pictures
roco
figs
part5
tony's slides
OneDrive
This PC
Network
DESKTOP-654O7
MP-TECRA-A50-
VBOXSVR
\\VBOXSVR\Dc
\\VBOXSVR\ub
Homegroup

Name	Date modified	Type	Size
.git	11/5/2014 1:29 PM	File folder	
figures	11/5/2014 12:39 PM	File folder	
lemaignan2014being.pdf	11/5/2014 1:29 PM	PDF File	5,11
library.bib	11/5/2014 12:39 PM	BIB File	7
library.bib.bak	11/5/2014 12:39 PM	BAK File	7
main.aux	11/5/2014 1:29 PM	AUX File	
main.bbl	11/5/2014 1:16 PM	BBL File	
main.blg	11/5/2014 1:16 PM	Performance Mon...	
main.log	11/5/2014 1:29 PM	Text Document	3
main.new.tex	11/5/2014 12:39 PM	TEX File	5
main.out	11/5/2014 1:29 PM	OUT File	
main.pdf	11/5/2014 1:29 PM	PDF File	5,11
main.tex	11/5/2014 1:29 PM	TEX File	1
Makefile	11/5/2014 12:39 PM	File	
sig-alternate.cls	11/5/2014 12:39 PM	CLS File	5
[illegible]	[illegible]/2014 12:39 PM	PY File	

16 items

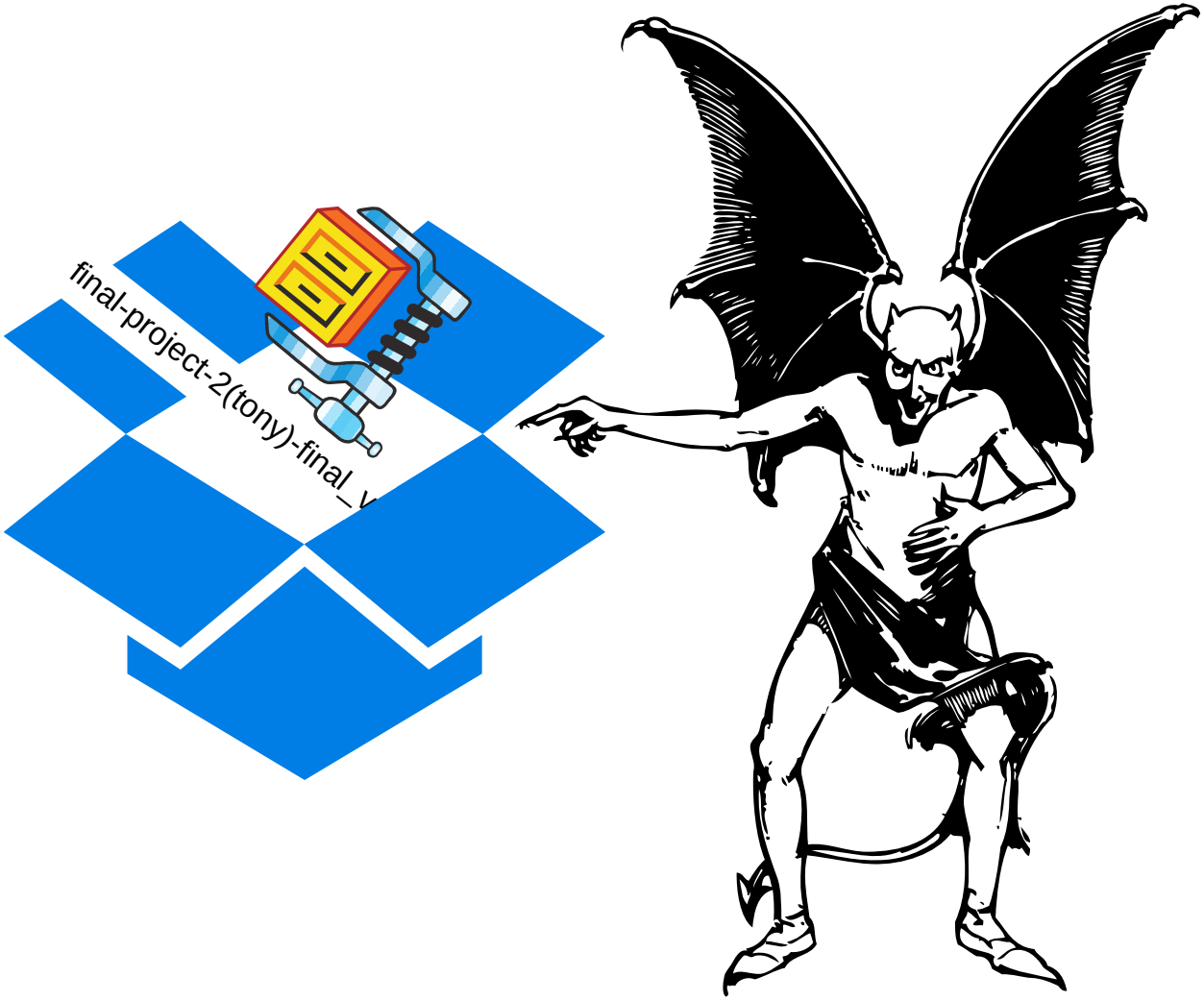
They look boringly normal

A few roundtrips later with teammates...



final-project-2(tony)-final_version3.zip

final-project-2(tony)-final_v


final-project-2(tony)-final_v

facebook

Search

Home Profile Account

Get Started
Wall
Hidden Posts
Info
Listings
Photos
Dan's Welcome Page
Discussions
Edit

About Edit

7
people like this

Add to My Page's Favorites

Tony King B.

Real Estate · Toronto, Ontario Edit Info

Edit Page

Wall Tony King B. · Most Recent

Share: Status Photo Link Video

Write something...

Tony King B.
Paul, you can take my change below:

```
using namespace std;
using namespace cv;
HeadPoseEstimation::HeadPoseEstimation(const string& face_detection_model, float focalLength) :
    focalLength(focalLength),
    opticalCenterX(-1),
    opticalCenterY(-1)
{
    // Load face detection and pose estimation models.
    detector = get_frontal_face_detector();
    deserialize(face_detection_model) >> pose_model;
```

51 Impressions · 0% Feedback
Tuesday at 2:25pm via re2social · Like · Comment

Tony King B.
SVN is really cool, but I like Facebook better!

51 Impressions · 0% Feedback
Tuesday at 2:25pm via re2social · Like · Comment

Admins (4) [?] See All

Use Facebook
Promote with an Ad
View Insights
Suggest to Friends

You

Tony King B. likes this.

Quick Tips

Get more people to like your Page with Facebook Ads today!

Get More Connections

Sample Ad

The text of your ad will go here.

Like · JP Zeni likes this.


facebook
Search
Home
Profile
Account
Edit Page
Tony King B
Real Estate · Toronto
Most Recent
Admins (4)
See All
take my change below:
namespace std;
namespace cv;
detection_model,
focalLength(focalLength),
to Friends
this.
Quick Tips
Get more to like your Page with Facebook
Get More Connections
Get Started
Wall
Hidden Posts
Info
Listings
Photos
Dan's Welcome Page
Discussions
Edit
About
Edit
SVN is cool, but I like Face er!
Sample Ad
The text of your ad will go here.
7
people like this
51 Impressions · 0% Feedback
Tuesday at 2:25pm via re2social · Like · Comment
Like · JP Zeni likes this.
Add to My Page's Favorites

We can do better!

We can do better!

git is essentially about recording the history of files

We can do better!

git is essentially about recording the history of files

(and who did what)

We can do better!

git is essentially about recording the history of files

(and who did what)

(and sharing as well – we'll talk about it tomorrow)

CODE VERSIONING

WHY VERSIONING?

- The history of your development/document
- Compare the current code with an older version
- Roll-back to previous versions (think 'undo on steroids')
- Experiment without losing anything
- Trace who did what (at the level of the line of code)
- Annotate your workflow (important milestones, etc)
- Avoid catastrophes!

ATOMIC COMMITS

The single most important concept (because it requires to think about development/writing in terms of **functional units**):

Atomic commit

A (typically small) commit that represent a **single, coherent & complete** functional change.

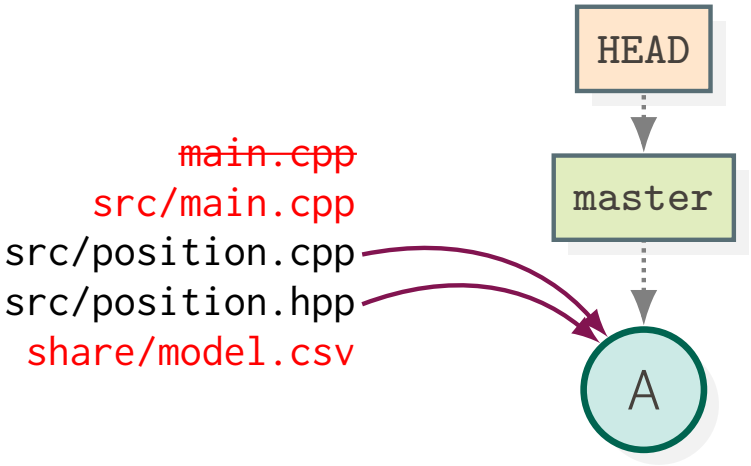
ATOMIC COMMITS

The single most important concept (because it requires to think about development/writing in terms of **functional units**):

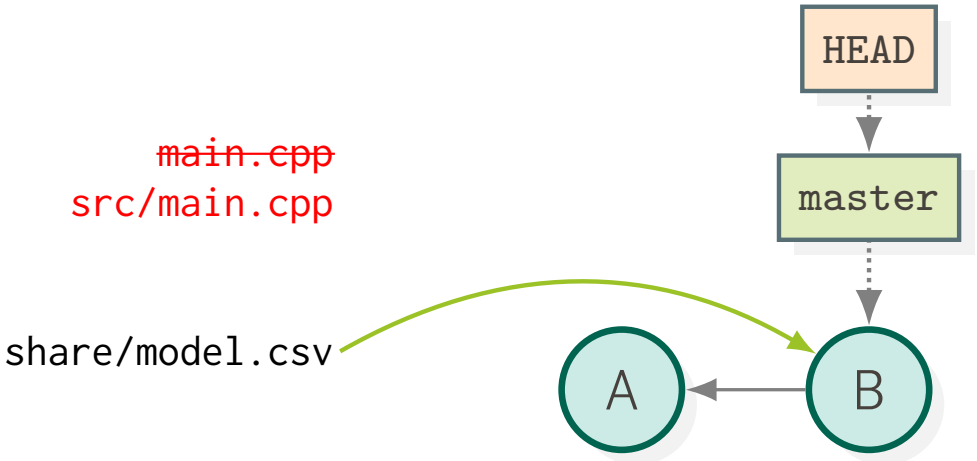
Atomic commit

- Easy to understand the change
- Debugging made easy (`git bisect`)
- Collaboration made easy (less, smaller conflict)
- Easy to write a useful commit message

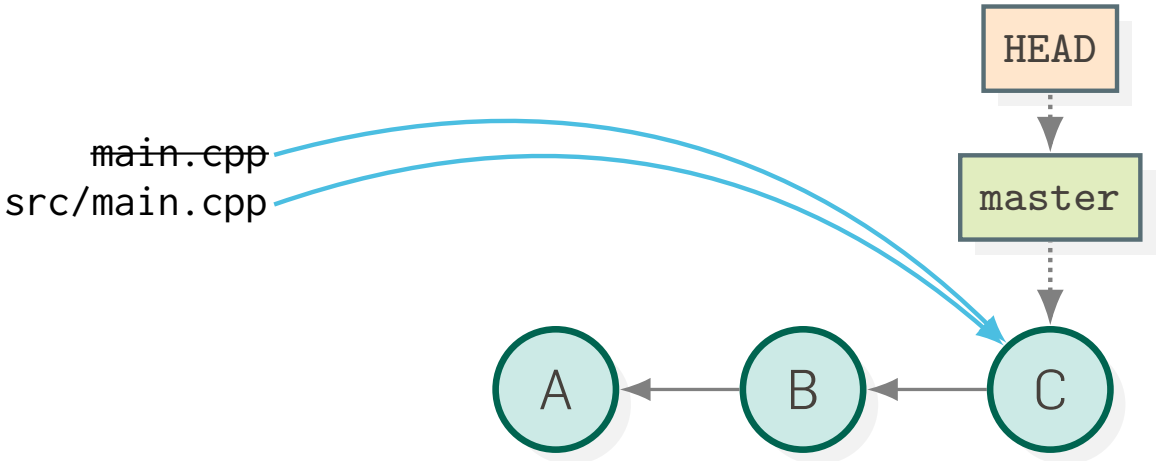
~~main.cpp~~
src/main.cpp
src/position.cpp
src/position.hpp
share/model.csv




```
git add src/position.*
git commit -m"Fix computation of position (float->double)"
```



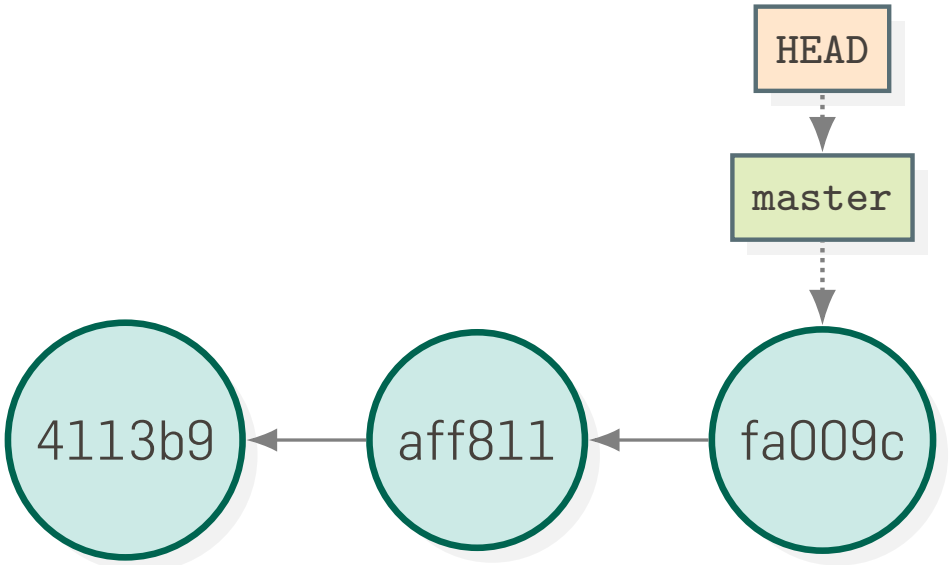

```
git add share/model.csv
git commit -m"Re-trained model with 52 more participants"
```




```
git rm main.cpp
git add src/main.cpp
git commit -m"Move main.cpp to src/"
```

LOG

```
$ git log

commit fa009cd7fca05b0b61170b20cf76a5f72b8843c2
Author: Severin Lemaignan <severin.lemaignan@brl.ac.uk>
Date:   Tue Nov 24 16:48:22 2020 +0000

    Move main.cpp to src/

commit aff81119459d9193c09effef1c150c4f7eac08dc
Author: Severin Lemaignan <severin.lemaignan@brl.ac.uk>
Date:   Tue Nov 24 16:48:02 2020 +0000

    Re-trained model with 52 more participants

commit 4113b9b6e6bbc8de532ad90153e0059cb5819de7
Author: Severin Lemaignan <severin.lemaignan@brl.ac.uk>
Date:   Tue Nov 24 16:47:46 2020 +0000

    Fix computation of position (float->double)
```


HEAD
master
4113b9
aff811
fa009c

THE STAGING AREA

But why do we have to manually tell Git what files to add or remove?

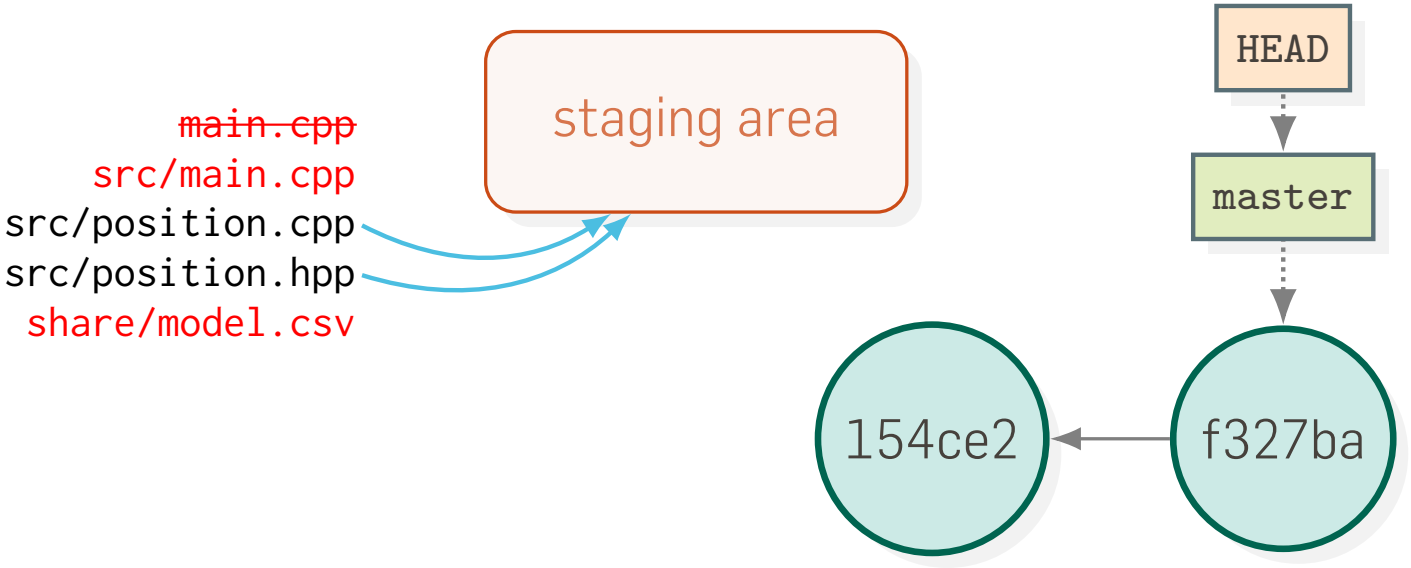
THE STAGING AREA

No “commit all changes” by default (well, you can, actually...)

$\Rightarrow$ Help thinking in terms of atomic commits!

THE STAGING AREA

Preparing a commit consists in filling the **staging area** (or **index**) with the list of changes:



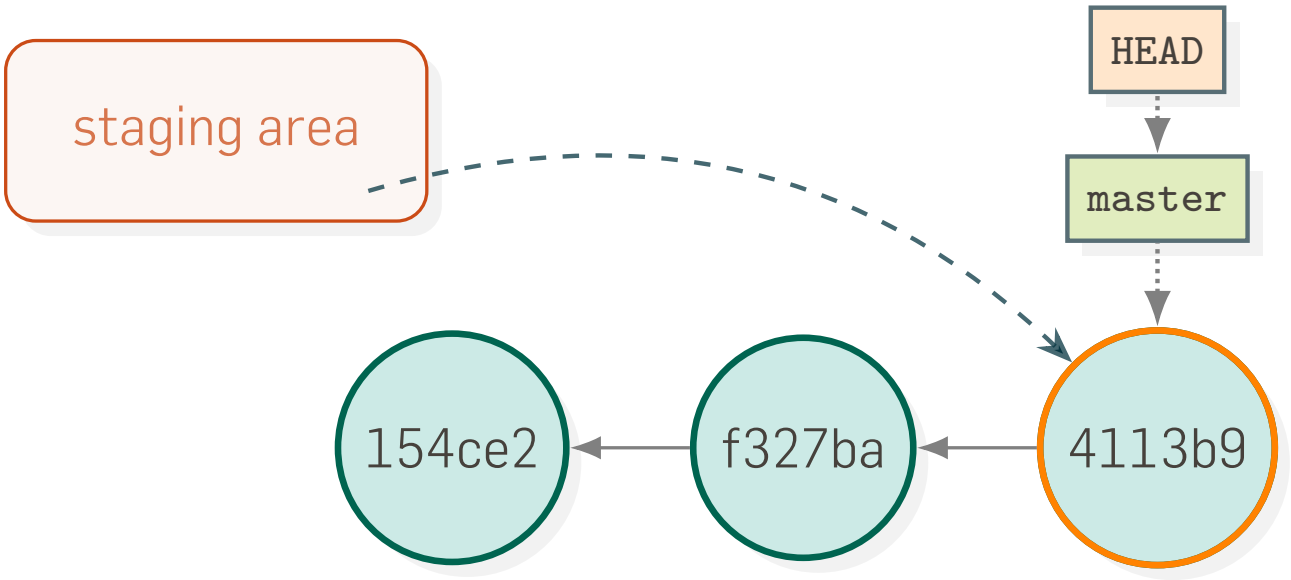

```
git add, git rm
git mv
git add -p
...
```

THE STAGING AREA

Preparing a commit consists in filling the **staging area** (or **index**) with the list of changes:

~~main.cpp~~
src/main.cpp

share/model.csv




```
git commit
```

TO SUMMARIZE...

The first time:

```
$ mkdir my_repo && cd my_repo
$ git init
```

Then:

```
# make some changes...
$ git add <files>
$ git commit -m"<commit message>"
# make some changes...
$ git add <files>
$ git commit -m"<other commit message>"
# That's it!
```

TO SUMMARIZE...

Plenty of tools to help with that, including GUI (one of the best one is the one that comes with VSCode).

My two favourite ones: `tig` and `gitk`

```
$ sudo apt install tig gitk
$ gitk &
$ tig status
```

Exercise: turn your `robomaze` pathfinder into a git repo, and commit the C++ and Python A* as two commits.

WHAT SHOULD BE TRACKED?

Short answer: **everything you care about in your project**

WHAT SHOULD BE TRACKED?

Short answer: **everything you care about in your project**

(you can left out temporary files, automatically generated files, etc → `.gitignore`)

WHAT SHOULD BE TRACKED?

Short answer: **everything you care about in your project**

(you can left out temporary files, automatically generated files, etc → `.gitignore`)

However, versioning is **less useful for binary files**:

- no line-by-line tracking of changes
- every single change creates a whole copy: repo size might grow quickly!

Binary files include images, archives (zip files), **PDF**, **most office document (docx/xlsx/pptx)**

WHAT SHOULD BE TRACKED?

Short answer: **everything you care about in your project**

(you can left out temporary files, automatically generated files, etc → `.gitignore`)

However, versioning is **less useful for binary files**:

- no line-by-line tracking of changes
- every single change creates a whole copy: repo size might grow quickly!

Binary files include images, archives (zip files), **PDF**, **most office document (docx/xlsx/pptx)**

For documents, you might want to consider alternative like markdown.

jnri2015-CoWriter

File Home Share View

Pin to Quick access | Copy | Paste | Move to | Copy to | Delete | Rename | New folder | Properties | Select all | Select none | Invert selection

Clipboard | Organize | New | Open | Select

« publications › jnri2015-CoWriter ›

Search jnri2015-CoWriter

Desktop
Downloads
Documents
Pictures
roco
figs
part5
tony's slides
OneDrive
This PC
Network
DESKTOP-654O7
MP-TECRA-A50-
VBOXSVR
\\VBOXSVR\Do
\\VBOXSVR\ub
Homegroup

Name	Date modified	Type	Size
.git	11/5/2014 1:29 PM	File folder	
figures	11/5/2014 12:39 PM	File folder	
lemaignan2014being.pdf	11/5/2014 1:29 PM	PDF File	5,11
library.bib	11/5/2014 12:39 PM	BIB File	7
library.bib.bak	11/5/2014 12:39 PM	BAK File	7
main.aux	11/5/2014 1:29 PM	AUX File	
main.bbl	11/5/2014 1:16 PM	BBL File	
main.blg	11/5/2014 1:16 PM	Performance Mon...	
main.log	11/5/2014 1:29 PM	Text Document	3
main.new.tex	11/5/2014 12:39 PM	TEX File	5
main.out	11/5/2014 1:29 PM	OUT File	
main.pdf	11/5/2014 1:29 PM	PDF File	5,11
main.tex	11/5/2014 1:29 PM	TEX File	1
Makefile	11/5/2014 12:39 PM	File	
sig-alternate.cls	11/5/2014 12:39 PM	CLS File	5
[illegible].connect.p	[illegible]/2014 12:39 PM	PY File	

16 items

What should I track here?

BUILDING CODE

COMPILING CODE IN C++

```
/*
 * Everyone's favourite: "Hello, World!"
 */

#include <iostream>

using namespace std;

int main(void)
{
    cout << "Hello, World!" << endl;
    return 0;
}
```

COMPILING CODE IN C++

```cpp
/*
 * Everyone's favourite: "Hello, World!"
 */

#include <iostream>

using namespace std;

int main(void)
{
    cout << "Hello, World!" << endl;
    return 0;
}
```

```
$ g++ hello.cpp -ohello
```

COMPILING CODE IN C++

```cpp
/*
 * Everyone's favourite: "Hello, World!"
 */

#include <iostream>

using namespace std;

int main(void)
{
    cout << "Hello, World!" << endl;
    return 0;
}
```

```
$ g++ hello.cpp -ohello
```

```
$ ./hello
Hello, World!
```

COMPILING CODE IN C++: THE MAIN STAGES

1. Pre-processing
2. Compilation
3. Assembly
4. Linking

These four steps are transparently performed one after the other by your favourite compiler.

COMPILING CODE IN C++: PRE-PROCESSING

```
/*
 * "Hello, World!": A classic.
 */

#include <iostream>

using namespace std;

int main(void)
{
    cout << "Hello, World!" << endl;
    return 0;
}
```

Pre-processor ***directives*** start with `#`

→ `#include <iostream>` is replaced by the content of that file.

COMPILING CODE IN C++: COMPILATION

```
$ g++ -S hello.cpp
```

```
main:
.LFB1493:
        .cfi_startproc
        pushq       %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq        %rsp, %rbp
        .cfi_def_cfa_register 6
        leaq        .LC0(%rip), %rsi
        leaq        _ZSt4cout(%rip), %rdi
        call        _ZStlsISt11char_traitsIcEERSt13basic_ostrea
        movq        %rax, %rdx
        movq        _ZSt4endlIcSt11char_traitsIcEERSt13basic_os
        movq        %rax, %rsi
        movq        %rdx, %rdi
        call        _ZNSolsEPFRSoS_E@PLT
```

COMPILING CODE IN C++: ASSEMBLY

```
$ g++ -s hello.cpp
$ hexdump a.out
0000000 457f 464c 0102 0001 0000 0000 0000 0000
0000010 0003 003e 0001 0000 07b0 0000 0000 0000
0000020 0040 0000 0000 0000 1128 0000 0000 0000
0000030 0000 0000 0040 0038 0009 0040 001b 001a
0000040 0006 0000 0005 0000 0040 0000 0000 0000
0000050 0040 0000 0000 0000 0040 0000 0000 0000
0000060 01f8 0000 0000 0000 01f8 0000 0000 0000
0000070 0008 0000 0000 0000 0003 0000 0004 0000
0000080 0238 0000 0000 0000 0238 0000 0000 0000
0000090 0238 0000 0000 0000 001c 0000 0000 0000
00000a0 001c 0000 0000 0000 0001 0000 0000 0000
00000b0 0001 0000 0005 0000 0000 0000 0000 0000
00000c0 0000 0000 0000 0000 0000 0000 0000 0000
00000d0 0b78 0000 0000 0000 0b78 0000 0000 0000
00000e0 0000 0020 0000 0000 0001 0000 0006 0000
...
```

COMPILING CODE IN C++: LINKING

The linker copies (and re-arrange) the machine code of the static dependencies (***static libraries***) into the executable.

That's what the `-l` flag is used for:

```
$ g++ cool_app.cpp -ocool_app -lcv_core -lcv_highgui -lcv_videoproc
```

MODERN COMPILER INFRASTRUCTURE

For instance, LLVM has a front-end for C/C++ called `clang` and has many backends (like `emscripten` to create the new `wasm` binaries for consumption by the web browsers)

LIBRARIES

A library is a collection of pre-compiled functions that might get called by an executable. ***Libraries are not executable*** by themselves.

Why libraries?

- to modularise your code
- to make it easier to reuse

LIBRARIES

A library is a collection of pre-compiled functions that might get called by an executable. ***Libraries are not executable*** by themselves.

Why libraries?

- to modularise your code
- to make it easier to reuse

Two main kinds:

- Static libraries, whose code is ***copied*** into the executable by the linker. Extensions: `.a`, `.lib`
- Dynamic libraries, whose code is ***loaded by the operating system*** at runtime. They are also called ***shared libraries***. Extensions: `.so`, `.dll`, `.dylib`

STATIC VS DYNAMIC LIBRARIES

Take 5 min and try to list 2 advantages for the static libraries on one hand, and the dynamic libraries on the other hand.

STATIC VS DYNAMIC LIBRARIES

Advantages of static libraries:

- application can be certain that all its libraries are present
- libraries are the correct version (on Linux, distributions and package managers handle that for dynamic libraries)
- single executable: simpler distribution and installation
- only need to copy (and load into memory) the parts that are needed

STATIC VS DYNAMIC LIBRARIES

Advantages of static libraries:

- application can be certain that all its libraries are present
- libraries are the correct version (on Linux, distributions and package managers handle that for dynamic libraries)
- single executable: simpler distribution and installation
- only need to copy (and load into memory) the parts that are needed

Advantages of dynamic libraries:

- executables smaller because no need to copy the libraries' code
- prevent redundant code in the system
- allows the libraries to be easily updated to fix bugs and security flaws without updating each of the applications

TWO TOOLS TO EXPLORE LIBRARIES

`nm` lists the symbols provided by a shared library:

```
$ nm libgazr.so
[...]
000000000002d040 W _ZN4dlib9impl_fhog8init_hogIfNS_33memory_manager_stateless_kerne
000000000027e940 b _ZN4dlibL23OBJECT_PART_NOT_PRESENTE
0000000000010d00 t _ZN9__gnu_cxx12__to_xstringINSt7__cxx1112basic_stringIcSt11char_
                 U _ZN9_IplImageC1ERKN2cv3MatE
                 U _Znam@@GLIBCXX_3.4
0000000000011370 T _ZNK18HeadPoseEstimation12drawFeaturesERKSt6vectorIS0_IN2cv6Poir
0000000000011b80 T _ZNK18HeadPoseEstimation12intersectionEN2cv6Point_IfEES2_S2_S2_R
0000000000011c40 T _ZNK18HeadPoseEstimation4poseEm
0000000000014500 T _ZNK18HeadPoseEstimation5posesEv
0000000000011b30 T _ZNK18HeadPoseEstimation8coordsOfEm14FACIAL_FEATURE
[...]
```

TWO TOOLS TO EXPLORE LIBRARIES

C++ signatures returned by `nm` are **mangled**. You can demangle them:

```
$ nm libgazr.so | c++filt
[...]
000000000027e940 b dlib::OBJECT_PART_NOT_PRESENT
0000000000010d00 t std::__cxx11::basic_string<char, std::char_traits<char>, std::al
                 U _IplImage::_IplImage(cv::Mat const&)
                 U operator new[](unsigned long)@@GLIBCXX_3.4
0000000000011370 T HeadPoseEstimation::drawFeatures(std::vector<std::vector<cv::Poi
0000000000011b80 T HeadPoseEstimation::intersection(cv::Point_<float>, cv::Point_<f
0000000000011c40 T HeadPoseEstimation::pose(unsigned long) const
0000000000014500 T HeadPoseEstimation::poses() const
0000000000011b30 T HeadPoseEstimation::coordsOf(unsigned long, FACIAL_FEATURE) cons
[...]
```

TWO TOOLS TO EXPLORE LIBRARIES

`ldd` lists the dependencies to shared libraries:

```
$ ldd estimate_head_pose
linux-vdso.so.1 (0x00007fff32387000)
libgazr.so (0x00007f3f1822f000)
libopencv_imgcodecs.so.3.2 => /usr/lib/x86_64-linux-gnu/libopencv_imgcodecs.so.3.2
libopencv_core.so.3.2 => /usr/lib/x86_64-linux-gnu/libopencv_core.so.3.2 (0x00007f3
libdlib.so.18 => /usr/lib/libdlib.so.18 (0x00007f3f178a8000)
libstdc++.so.6 => /usr/lib/x86_64-linux-gnu/libstdc++.so.6 (0x00007f3f1751a000)
libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007f3f17302000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f3f16f11000)
libblas.so.3 => /usr/lib/x86_64-linux-gnu/libblas.so.3 (0x00007f3f16ca4000)
liblapack.so.3 => /usr/lib/x86_64-linux-gnu/liblapack.so.3 (0x00007f3f16406000)
libopencv_calib3d.so.3.2 => /usr/lib/x86_64-linux-gnu/libopencv_calib3d.so.3.2 (0x0
libopencv_imgproc.so.3.2 => /usr/lib/x86_64-linux-gnu/libopencv_imgproc.so.3.2 (0x0
libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007f3f157c7000)
[...]
```

HOW TO MAKE & USE LIBRARIES?

Code source of a pathfinding tool for our robots:

```
controller.cpp # contains the entry point: int main(...)
ui.cpp
astar.cpp
```

```
$ g++ controller.cpp ui.cpp astar.cpp -opathfinding_ui
```

`astar.cpp` contains the actual pathfinder, and might be useful for many other projects. How to turn it into a library?

HOW TO MAKE & USE LIBRARIES?

First, we need to extract the **API** of our library in a **public header** `astar.hpp`:

```
#ifndef _PATHFINDING_HPP
#define _PATHFINDING_HPP

#include <tuple>
#include <vector>
#include <array>

const unsigned int MAP_WIDTH = 100;
const unsigned int MAP_HEIGHT = 100;

typedef std::tuple<unsigned int, unsigned int> Position;
typedef std::vector<Position> Path;
typedef std::array<bool, MAP_WIDTH * MAP_HEIGHT> Map;

class AStar {

    AStar(std::shared_ptr<const Map> map);
    Path find(size_t goal_x, size_t goal_y);
}
#endif
```

The header contains the **declarations** of our classes, structures, functions, but not the **definitions** (the definitions are in `astar.cpp`).

HOW TO MAKE & USE LIBRARIES?

Next, compile the library:

```
$ g++ -fPIC -shared astar.cpp -olibastar.so
```

HOW TO MAKE & USE LIBRARIES?

Finally, use it:

```
$ g++ controller.cpp ui.cpp -lastar -opathfinding_ui
```

Exercise: extract the A* algorithm from your C++ robomaze client and turn it into a library.

ORGANISING YOUR CODE

principle of least surprise

Make people feel at home when they interact with your project!

REPOSITORY LAYOUT

Try to follow as much as possible the **Filesystem Hierarchy Standard** (FHS). Mainly:

```
src/        # source
include/    # *public* headers
etc/        # configuration files
share/      # data
doc/        # documentation
README
LICENSE
```

NO build artifacts!!

no binaries (except possibly in `share/`)

REPOSITORY LAYOUT

Try to follow as much as possible the **Filesystem Hierarchy Standard** (FHS). Mainly:

```
src/        # source
include/    # *public* headers
etc/        # configuration files
share/      # data
doc/        # documentation
README
LICENSE
```

`README` (or better, use markdown: `README.md`): what is the project about? who is the target audience? how to install? how to get started?

BTW, THE LINUX FILESYSTEM...

```
$ tree -d /
/
 boot   # linux image, grub, initramfs...
 dev    # block devices
 etc    # configuration files
 home   # users' home directories
 mnt    # mount point for eg external devices
 opt    # non-distribution software (eg, ROS)
 proc   # process information _pseudo-file_ system
 tmp    # RAM-mounted temporary space
 usr    # User system resources -- most binaries and lib are here
     bin
     include
     lib
     local
     sbin
     share
     src
 var # variables: log files, local sockets...
```

Read more about these here: Linux Filesystem Hierachy

EXAMPLE 1

What would you change?

```
my_proj/
  controller.cpp
  ui.cpp
  ui.hpp
  astar.cpp
  astar.hpp
  ui.conf
```

EXAMPLE 1

```
my_proj/
  src/
    controller.cpp
    ui.cpp
    ui.hpp
    astar.cpp
  include/
    astar.hpp
  etc/
    ui.conf
  README.md
  LICENSE
```

README.MD EXAMPLE

```
Better pathfinder
=================

![doc/screenshot.png](Screenshot of the provided UI)

This is a really better pathfinder. Check the [publication](http://link...).

Pre-requisites
--------------

- dependency 1
- dependency 2

Installation
------------

```
mkdir build && cd build && cmake .. && make install
```

Usage

```

MARKDOWN CHEAT SHEET

Titles/sections:

```
Title
=====
Subtitle
--------
### Section
#### Subsection
```

Links/images:

```
[Click here](otherpage.md)
![Image caption](myimage.png)
```

Formatting:

```
**Bold**
*Italics*
_Underlined_
~~Strike through~~
`code`
```

Lists:

```
- item (you can use * or + as well)
- item
- item

1. item
2. item
3. item
```

Syntax-highlighted code blocks:

```
```python
    # this is a Python code block
```
```

(tables are also possible)

Exercise:

1. clone this repo:

   ```
   git clone https://github.com/severin-lemaignan/robomaze-cpp.git
   ```

2. re-organise it to follow the FHS and best practises
3. (bonus) try to compile `astar.cpp` as a shared library

EXAMPLE 2: YOU TAKE OVER AN EXISTING PROJECT

```
joe@doe:/usr/robot-planning$ ls -alh
drwxr-xr-x  2 joe  root    4.0K Sep 22 10:40 .
drwxr-xr-x 12 root root    4.0K Sep 22 10:36 ..
-rwxrwxr-x  1 joe  joe     8.8K Sep 22 10:40 pathfinding_ui
-rw-rw-r--  1 joe  joe     2.1K Sep 22 10:39 compile.sh
-rw-rw-r--  1 joe  joe     1.8K Sep 22 10:39 compile.bat
-rw-rw-r--  1 joe  joe      134 Sep 22 10:39 readme-first.txt
-rw-rw-r--  1 joe  joe      895 Sep 21 21:38 controller.cpp
-rw-rw-r--  1 joe  joe     1.2K Sep 21 20:27 controller.hpp
-rw-rw-r--  1 joe  joe     5.8K Sep 22 10:40 controller.o
-rw-rw-r--  1 joe  joe       50 Sep 19 10:36 controller.ini
-rw-rw-r--  1 joe  joe     2.3K Sep 20 09:31 astar.cpp
-rw-rw-r--  1 joe  joe      230 Sep 20 10:32 astar.hpp
-rw-rw-r--  1 joe  joe     4.3K Sep 22 10:40 astar.o
-rw-rw-r--  1 joe  joe     7.3K Sep 21 10:40 astar.so
-rw-rw-r--  1 joe  joe     6.7K Sep 20 11:13 core.cpp
-rw-rw-r--  1 joe  joe     7.1K Sep 22 10:40 core.o
-rw-rw-r--  1 joe  joe     6.1K Sep 22 10:40 core.a
-rw-rw-r--  1 joe  joe     6.0K Sep 21 16:22 core.lib
```

EXAMPLE 2: YOU TAKE OVER AN EXISTING PROJECT

Points that can be improved:

- Developping in `usr/` is a bad practice
- Rename files to be more descriptive
- Change the layout to follow the FHS (eg `controller.ini` to `etc/controller.ini`)
- Use a buildsystem (like CMake) instead of relying on ad-hoc scripts
- Add a `README` and a `LICENSE`
- Public headers should be moved to `include/`

BUILD SYSTEMS

BUILD SYSTEM

Use and provide a build system!

- Windows-only ⇒ a Visual Studio solution is ok
- MacOS-only ⇒ a XCode project is ok

In all other cases, go for a cross-platform build system like **CMake** or **Meson**.

```
$ sudo apt install cmake
$ sudo apt install cmake-curses-gui # a super useful cmdline GUI for CMake
```

EXAMPLE OF A CMAKE FILE: CMAKELISTS.TXT

Create a file called `CMakeLists.txt` at the root of the `robomaze-cpp` project:

```
cmake_minimum_required(VERSION 3.10)
project(robomaze-pilot VERSION 1.0)

# specify the C++ standard
set(CMAKE_CXX_STANDARD 14)
set(CMAKE_CXX_STANDARD_REQUIRED True)

find_package(cpprestsdk REQUIRED) # one external dependency

# First, the library
add_library(astar SHARED src/astar.cpp)
target_include_directories(astar PUBLIC include)

# then, the executable, which depends on the library's target
add_executable(robomaze-client src/controller.cpp)
target_include_directories(robomaze-client PUBLIC include)
target_link_libraries(robomaze-client PUBLIC astar cpprestsdk::cpprest)
```

CONFIGURING AND COMPILING WITH CMAKE

```
$ cmake .
-- The C compiler identification is GNU 9.3.0
-- The CXX compiler identification is GNU 9.3.0
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found ZLIB: /usr/lib/x86_64-linux-gnu/libz.so (found version "1.2.11")
-- Found OpenSSL: /usr/lib/x86_64-linux-gnu/libcrypto.so (found version "1.1.1f")
-- Configuring done
-- Generating done
-- Build files have been written to: robomaze-pilot/build
```

CONFIGURING AND COMPILING WITH CMAKE

```
$ make
Scanning dependencies of target astar
[ 25%] Building CXX object CMakeFiles/astar.dir/src/astar.cpp.o
[ 50%] Linking CXX shared library libastar.so
[ 50%] Built target astar
Scanning dependencies of target robomaze-client
[ 75%] Building CXX object CMakeFiles/robomaze-client.dir/src/controller.cpp.o
[100%] Linking CXX executable robomaze-client
[100%] Built target robomaze-client
$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  CMakeLists.txt  include
    libastar.so  LICENSE  Makefile  README.md  robomaze-client  src
```

CMake **configures** the project, and generate **Makefiles** that are consumed by `make` to actually compile the project.

Note that `make` and Makefiles are independent from CMake. CMake is a Makefiles **generator**. It can also generate VS solutions, etc.

OUT-OF-TREE BUILDING

```
robomaze-client/
  src/
    controller.cpp
    astar.cpp
  include/
    astar.hpp
  README.md
  LICENSE
  CMakeLists.txt
```

OUT-OF-TREE BUILDING

When compiling the project, create a sub-directory `build` and perform an **out-of-tree** build:

```
$ mkdir build && cd build
$ cmake ..
$ make
```

OUT-OF-TREE BUILDING

```
robomaze-client/
  build/
    ... # lots of compilation artifacts
  src/
    controller.cpp
    astar.cpp
  include/
    astar.hpp
  README.md
  LICENSE
  CMakeLists.txt
```

The `build/` directory can be deleted at any point as it contains only generated files.

INSTALLING CODE

Once the application/library is compiled, we normally want to install it in the system, to be generally available. CMake can add an `install` **target** to the Makefiles:

```
cmake_minimum_required(VERSION 3.10)
project(robomaze-pilot VERSION 1.0)

# [...same code as before...]

# install everything
install(TARGETS astar DESTINATION lib)
install(FILES include/astar.h DESTINATION include)
install(TARGETS robomaze-client DESTINATION bin)
```

INSTALLING CODE

```
$ mkdir build && cd build
$ cmake ..
$ make install
Scanning dependencies of target astar
[ 25%] Building CXX object CMakeFiles/astar.dir/src/astar.cpp.o
[ 50%] Linking CXX shared library libastar.so
[ 50%] Built target astar
Scanning dependencies of target robomaze-client
[ 75%] Building CXX object CMakeFiles/robomaze-client.dir/src/controller.cpp.o
[100%] Linking CXX executable robomaze-client
[100%] Built target robomaze-client
Install the project...
-- Install configuration: ""
-- Installing: /usr/local/lib/libastar.so
CMake Error at cmake_install.cmake:47 (file):
  file INSTALL cannot copy file
  "robomaze-pilot/build/libastar.so" to "/usr/local/lib/libastar.so".

Makefile:117: recipe for target 'install' failed
make: *** [install] Error 1
```

INSTALLING CODE

```
$ mkdir build && cd build
$ cmake .. -DCMAKE_INSTALL_PREFIX=$HOME/devel
$ make install
[ 50%] Built target astar
[100%] Built target robomaze-client
Install the project...
-- Install configuration: ""
-- Installing: /home/s-lemaignan/devel/lib/libastar.so
-- Installing: /home/s-lemaignan/devel/include/astar.h
-- Installing: /home/s-lemaignan/devel/bin/robomaze-client
-- Set runtime path of "/home/s-lemaignan/devel/bin/robomaze-client" to ""
```

Your **install prefix** is where you want to install the code you compile yourself. The default (`/usr/local`) is not great (why?). I recommend `$HOME/devel` for instance.

THE ONE SLIDE TO REMEMBER

```
$ mkdir build && cd build
$ cmake .. -DCMAKE_INSTALL_PREFIX=$HOME/devel -DCMAKE_BUILD_TYPE=Release
$ make install
```

...you will type that often!

WHAT ABOUT PYTHON?

Create a `setup.py` file at the root of the project:

```python
import setuptools

setuptools.setup(
    name="pyrobomaze",
    version="1.0.0",
    author="Séverin Lemaignan",
    author_email="severin.lemaignan@brl.ac.uk",
    description="A A* pathfinder to solve the robomaze game",
    url="https://github.com/severin-lemaignan/pyrobomaze",
    install_requires=['requests'],
    package_dir = {'': 'src'},
    packages=['robomaze'],
    scripts=['scripts/pyrobomaze'],
    classifiers=[
        "Programming Language :: Python :: 3",
        "License :: OSI Approved :: MIT License",
        "Operating System :: OS Independent",
    ],
    python_requires='>=3.6',
)
```

WHAT ABOUT PYTHON?

Create a `setup.py` file at the root of the project:

```
$ python3 setup.py install --prefix $HOME/devel
$ pyrobomaze <your robot>
```

- `PYTHONPATH`: where Python looks for libraries
- `export PYTHONPATH=$HOME/devel/lib/python3.7/site-package` (add to your `.bashrc`)
- why no need for `python3` in front of `pyrobomaze`?
  ⇒ shebang: `#! /usr/bin/python3`

Exercise:

1. clone this repo:

   ```
   git clone https://github.com/severin-lemaignan/pyrobomaze.git
   ```

2. re-organise it to follow the FHS and best practises
3. add a setup.py and install the python project locally

VERSIONING

SEMANTIC VERSIONING

Given a version number `MAJOR.MINOR.PATCH`, increment the:

- `MAJOR` version when you make incompatible API changes,
- `MINOR` version when you add functionality in a backwards-compatible manner, and
- `PATCH` version when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the `MAJOR.MINOR.PATCH format`.

Source: `semver` *website*

SEMANTIC VERSIONING

You are the maintainer of `cool_app`, that depends on OpenCV `2.4.11`.

The OpenCV project releases a new version, what should you do...

- ...if the new version is `2.4.12`?

SEMANTIC VERSIONING

You are the maintainer of `cool_app`, that depends on OpenCV `2.4.11`.

The OpenCV project releases a new version, what should you do...

- ...if the new version is `2.4.12`?
- ...if the new version is `2.5.0`?

SEMANTIC VERSIONING

You are the maintainer of `cool_app`, that depends on OpenCV `2.4.11`.

The OpenCV project releases a new version, what should you do...

- ...if the new version is `2.4.12`?
- ...if the new version is `2.5.0`?
- ...if the new version is `2.9.0`?

SEMANTIC VERSIONING

You are the maintainer of `cool_app`, that depends on OpenCV `2.4.11`.

The OpenCV project releases a new version, what should you do...

- ...if the new version is `2.4.12`?
- ...if the new version is `2.5.0`?
- ...if the new version is `2.9.0`?
- ...if the new version is `3.0.0-beta`?

SEMANTIC VERSIONING

You are the maintainer of `cool_app`, that depends on OpenCV `2.4.11`.

The OpenCV project releases a new version, what should you do...

- ...if the new version is `2.4.12`?
- ...if the new version is `2.5.0`?
- ...if the new version is `2.9.0`?
- ...if the new version is `3.0.0-beta`?
- ...if the new version is `3.0.0`?

SOFTWARE LICENSES, OPEN-SOURCE, FREE SOFTWARE

SOFTWARE LICENSES

- **no license** ⇒ default copyright laws apply. You retain all rights to your source code; nobody else may reproduce, distribute, or create derivative works from your work.

SOFTWARE LICENSES

- **no license** ⇒ default copyright laws apply. You retain all rights to your source code; nobody else may reproduce, distribute, or create derivative works from your work.
- **Permissive licenses**: others do essentially whatever they want with your code, as long as they give your attribution. Examples: MIT, BSD

SOFTWARE LICENSES

- **no license** ⇒ default copyright laws apply. You retain all rights to your source code; nobody else may reproduce, distribute, or create derivative works from your work.
- **Permissive licenses**: others do essentially whatever they want with your code, as long as they give your attribution. Examples: MIT, BSD
- **Copyleft licenses**: Derivative work must be made available under the same terms as the original work (*viral licenses*). Example: GPL

SOFTWARE LICENSES

- **no license** ⇒ default copyright laws apply. You retain all rights to your source code; nobody else may reproduce, distribute, or create derivative works from your work.
- **Permissive licenses**: others do essentially whatever they want with your code, as long as they give your attribution. Examples: MIT, BSD
- **Copyleft licenses**: Derivative work must be made available under the same terms as the original work (*viral licenses*). Example: GPL

If you are paid by UWE or UoB, the copyright belongs to the uni.

SOFTWARE LICENSES

- **no license** ⇒ default copyright laws apply. You retain all rights to your source code; nobody else may reproduce, distribute, or create derivative works from your work.
- **Permissive licenses**: others do essentially whatever they want with your code, as long as they give your attribution. Examples: MIT, BSD
- **Copyleft licenses**: Derivative work must be made available under the same terms as the original work (*viral licenses*). Example: GPL

Check **http://choosealicense.com/**

WHAT IF YOU WANT TO USE A GPL LIBRARY?

There is a legal dispute to know whether merely ***linking*** with a library result in a ***derivative work*** (which would then have to be licensed as GPL).

WHAT IF YOU WANT TO USE A GPL LIBRARY?

There is a legal dispute to know whether merely ***linking*** with a library result in a ***derivative work*** (which would then have to be licensed as GPL).

The LGPL (***Lesser GPL***) explicitely allows the usage of the library without putting restrictions on the licensing of the resulting executable.

Open-source vs Free software?

"When we call software "free," we mean that it respects the users' essential freedoms: the freedom to run it, to study and change it, and to redistribute copies with or without changes. This is a matter of freedom, not price, so think of "free speech," not "free beer.""

"Open source is a development methodology; free software is a social movement"

Source: GNU website

PACKAGING

INVESTIGATING THE HELLO PACKAGE

```
$ sudo apt install hello
$ hello
Hello, world!
```

What ***really*** happens when we type `apt install hello`?

INVESTIGATING THE HELLO PACKAGE

The `hello` package is downloaded from the Ubuntu repositories, and installed. Let's investigate:

```
$ apt download hello
$ ar x hello_2.10-1build1_amd64.deb
$ ls
control.tar.gz  data.tar.xz  debian-binary
```

INVESTIGATING THE HELLO PACKAGE

```
$ tar xf control.tar.gz
$ cat control
Package: hello
Version: 2.10-1build1
Architecture: amd64
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Santiago Vila <sanvila@debian.org>
Installed-Size: 108
Depends: libc6 (>= 2.14)
Conflicts: hello-traditional
Breaks: hello-debhelper (<< 2.9)
Replaces: hello-debhelper (<< 2.9), hello-traditional
Section: devel
Priority: optional
Homepage: http://www.gnu.org/software/hello/
Description: example package based on GNU hello
 The GNU hello program produces a familiar, friendly greeting.  It
 allows non-programmers to use a classic computer science tool which
 would otherwise be unavailable to them.
 .
 Seriously, though: this is an example of how to do a Debian package.
 It is the Debian version of the GNU Project's `hello world' program
 (which is itself an example for the GNU Project).
```

DEPENDENCY RESOLUTION

Applications depends on other bits of code! Other libraries, resources, executables, etc.

The package has to store in its metadata this information.

The ***dependency solver*** (on Ubuntu/Debian, `apt`) finds the smaller set of packages to install to satisfy all dependencies.

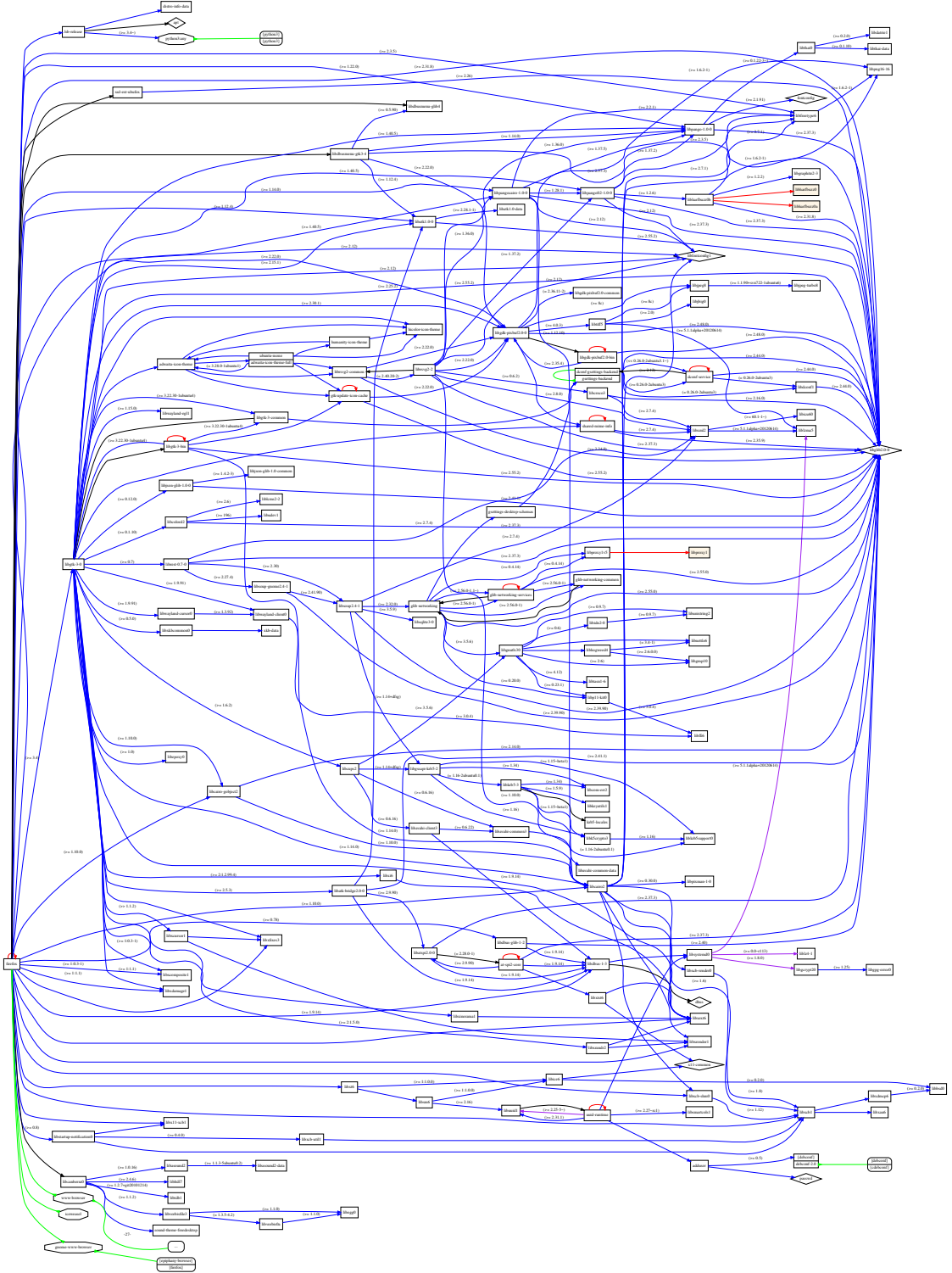
DEPENDENCY RESOLUTION

Applications depends on other bits of code! Other libraries, resources, executables, etc.

The package has to store in its metadata this information.

The ***dependency solver*** (on Ubuntu/Debian, `apt`) finds the smaller set of packages to install to satisfy all dependencies.

```
$ debtree firefox > firefox_deps.dot
$ dot -Tsvg -ofirefox_deps.svg firefox_deps.dot
```



CREATING A PACKAGE

Creating packages is not trivial, and necessitate a good knowledge of the inner working of a application or library.

CREATING A PACKAGE

Creating packages is not trivial, and necessitate a good knowledge of the inner working of a application or library.

The build system (CMake, for instance) already knows a lot!

CPACK

Using cpack to create a package for your A* pathfinder:

```
cmake_minimum_required(VERSION 3.10)

project(robomaze-pilot VERSION 1.0)

set(CPACK_RESOURCE_FILE_LICENSE "${CMAKE_CURRENT_SOURCE_DIR}/LICENSE")
set(CPACK_PACKAGE_VERSION_MAJOR "${robomaze-pilot_VERSION_MAJOR}")
set(CPACK_PACKAGE_VERSION_MINOR "${robomaze-pilot_VERSION_MINOR}")
set(CPACK_PACKAGE_CONTACT "Séverin Lemaignan <severin.lemaignan@brl.ac.uk>")
include(CPack)

# remaining of the file is identical
```

Then:

```
$ cpack ..
$ ls *.deb
robomaze-pilot-1.0.1-Linux.deb
$ sudo apt install ./robomaze-pilot-1.0.1-Linux.deb
```

CPACK

(what about the install prefix?)

```
$ dpkg -L robomaze-pilot
/usr
/usr/bin
/usr/bin/robomaze-client
/usr/include
/usr/include/astar.h
/usr/lib
/usr/lib/libastar.so
```

→ The install prefix is automatically replaced with the system prefix.

WHAT ABOUT PYTHON PACKAGING?

Simpler:

```
$ python3 setup.py sdist bdist_wheel
```

This command should output a lot of text and once completed should generate two files in the dist directory:

```
dist/
  pyrobomaze-1.0.0-py3-none-any.whl
  pyrobomaze-1.0.0.tar.gz
```

You need to create an account on the *Python Index* (**pypi.org**). Then you can upload your package:

```
$ twine upload dist/*
```

That's it! The whole world can access your package.

Dialogs 0.14

```
pip install Dialogs
```

Latest version

Last released: Mar 5, 2015

Handles natural language inputs and outputs on cognitive robots

Meta

License: BSD License (BSD)

Author: Séverin Lemaignan

Project description

1. LAAS-CNRS 2010-2013, EPFL 2013-2015

This module, licensed under the permissive BSD 3-clause, reads on stdin user input in natural language, parse it, call resolution routines when ambiguous concepts are used, and finally generate RDF statements that are an interpretation of the input.

It includes as well a verbalization module that conversely turns RDF statements into a sentence in natural language.

![Overview of the Dialogs pipeline](doc/dialogs_module_simple_small.png)

While not strictly required, it is strongly recommanded to use *dialogs* with a knowledge base that follows the ''KB API'' like [minimalKB](https://github.com/severin-lemaignan/minimalkb/) or [oro-server](http://oro.openrobots.org).

You are welcome to reuse this software for your research. Please refer to the CITATION file for proper attribution in scientific works.

Installation

Simply run:

```
` > pip install dialogs `
```

Usage

You can start to use *dialogs* immediately. For instance, try:

```
` > dialogs -d -p"What are you doing?" > dialogs -d -p"I'm playing with you" `
```

The *-d* flags activates the debug mode, and gives you a complete picture of the different steps: pre-processing, parsing, semantic resolution of the atoms of the sentence, interpretation and verbalization ([read the paper]

LANGUAGE-SPECIFIC PACKAGE MANAGEMENT

How to use our package? **Anyone** can simply type:

```
$ pip3 install pyrobomaze
$ pyrobomaze my_robot
```

→ Anyone can upload Python packages to the **pypi** archive: great to quickly share software packages, without having to bother with source code.

LANGUAGE-SPECIFIC PACKAGE MANAGEMENT

How to use our package? **Anyone** can simply type:

```
$ pip3 install pyrobomaze
$ pyrobomaze my_robot
```

→ Anyone can upload Python packages to the **pypi** archive: great to quickly share software packages, without having to bother with source code.

Many languages offer such a language-specific package management system (**npm** for Javascript, **cargo** for rust, **rubygems** for ruby, etc)

LANGUAGE-SPECIFIC PACKAGE MANAGEMENT

How to use our package? **Anyone** can simply type:

```
$ pip3 install pyrobomaze
$ pyrobomaze my_robot
```

→ Anyone can upload Python packages to the **pypi** archive: great to quickly share software packages, without having to bother with source code.

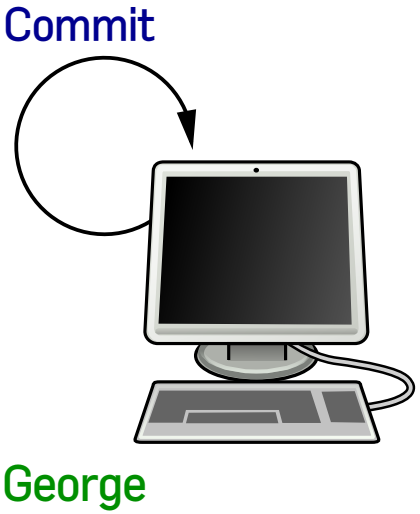
Many languages offer such a language-specific package management system (**npm** for Javascript, **cargo** for rust, **rubygems** for ruby, etc)

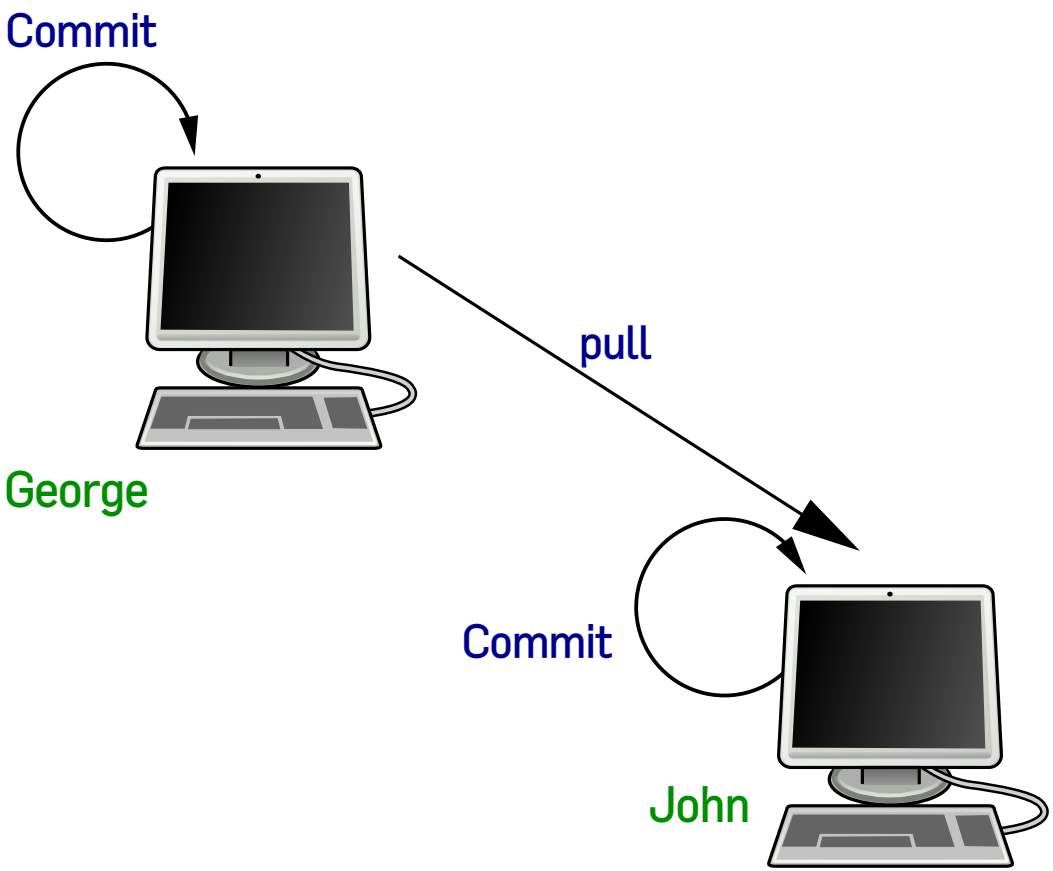
However, no guarantees! In contrast, the acceptance process for Debian is extremly strict → guarantees a level of quality and proper integration.

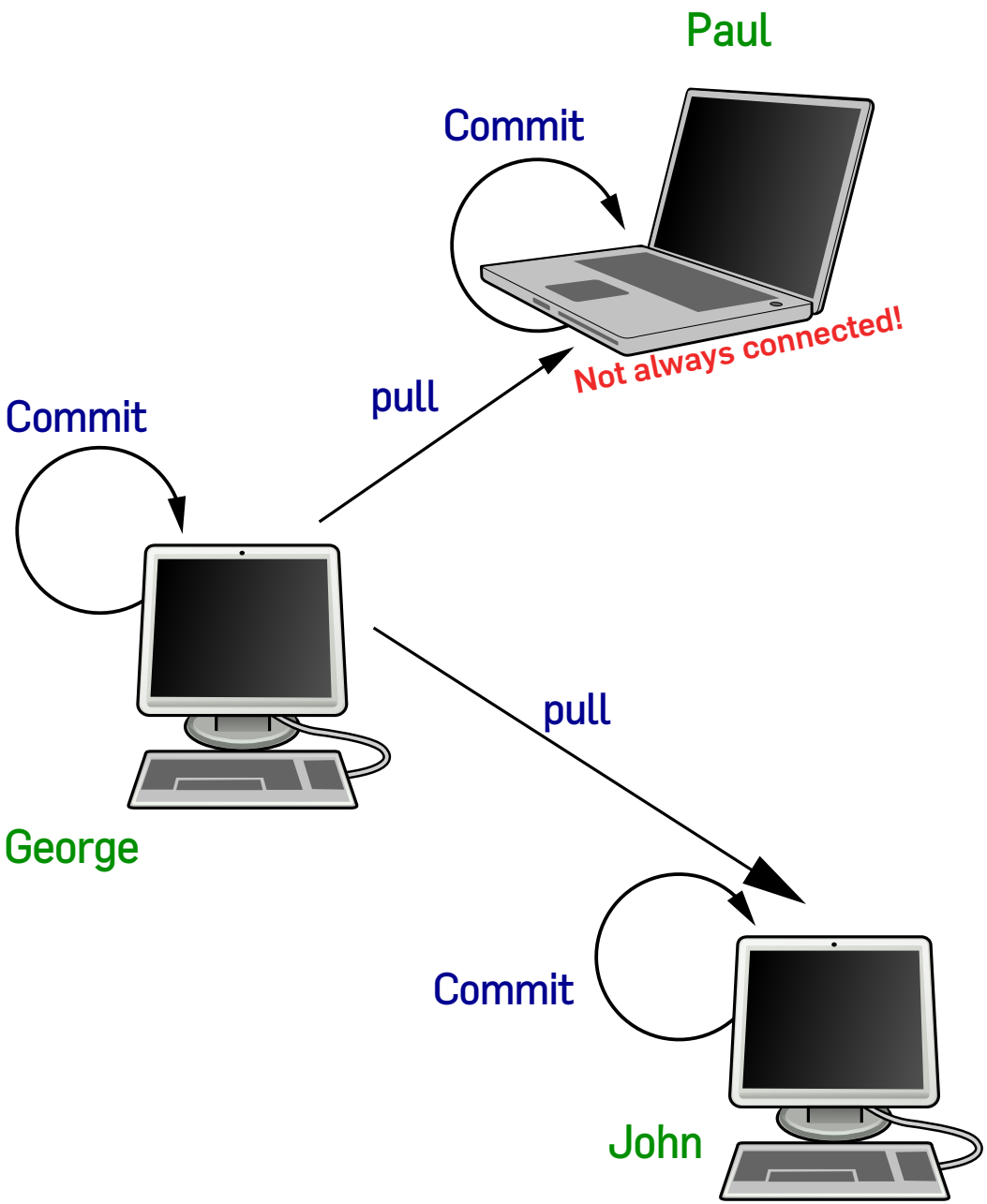
ADVANCED GIT

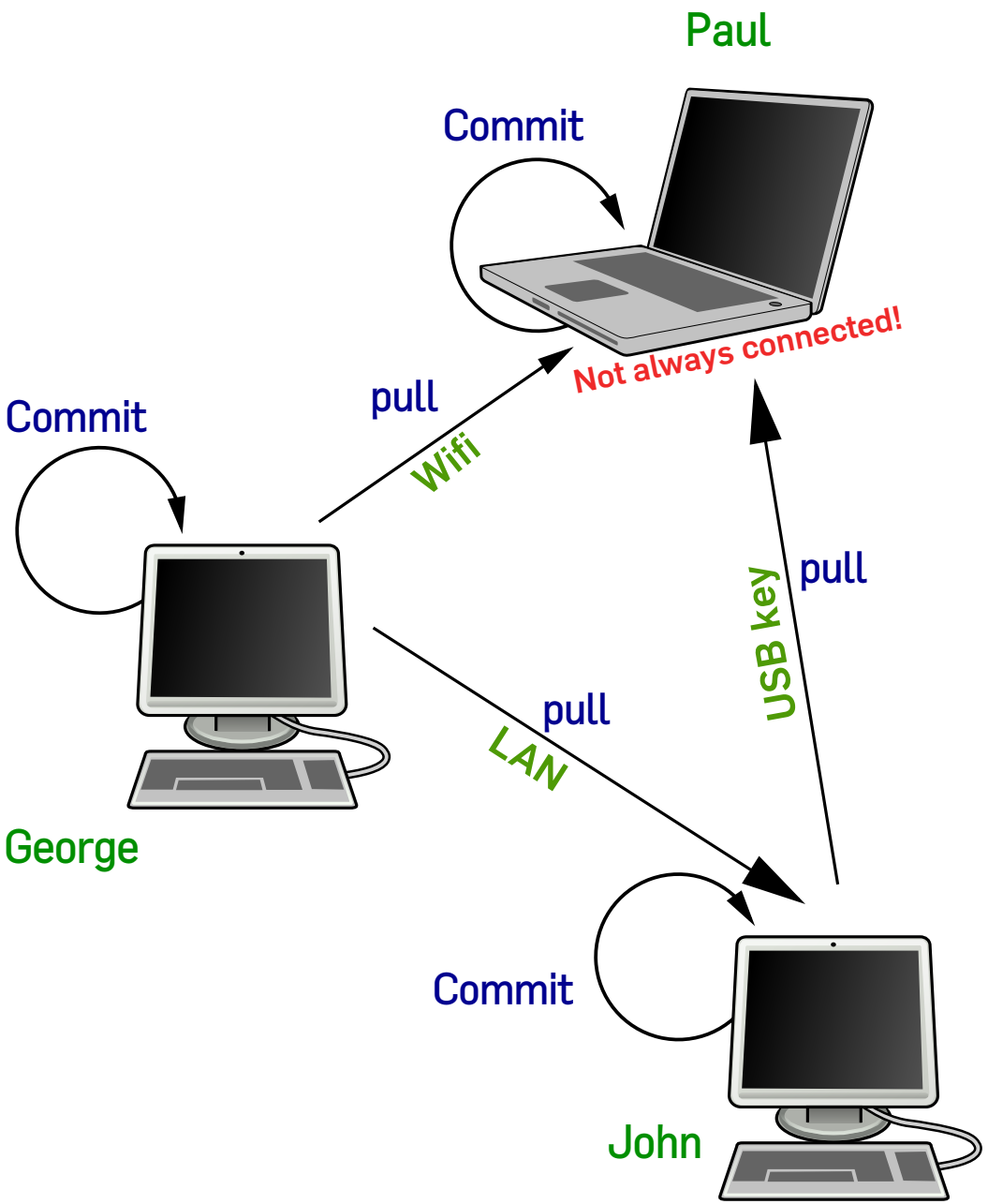
ADVANCED GIT

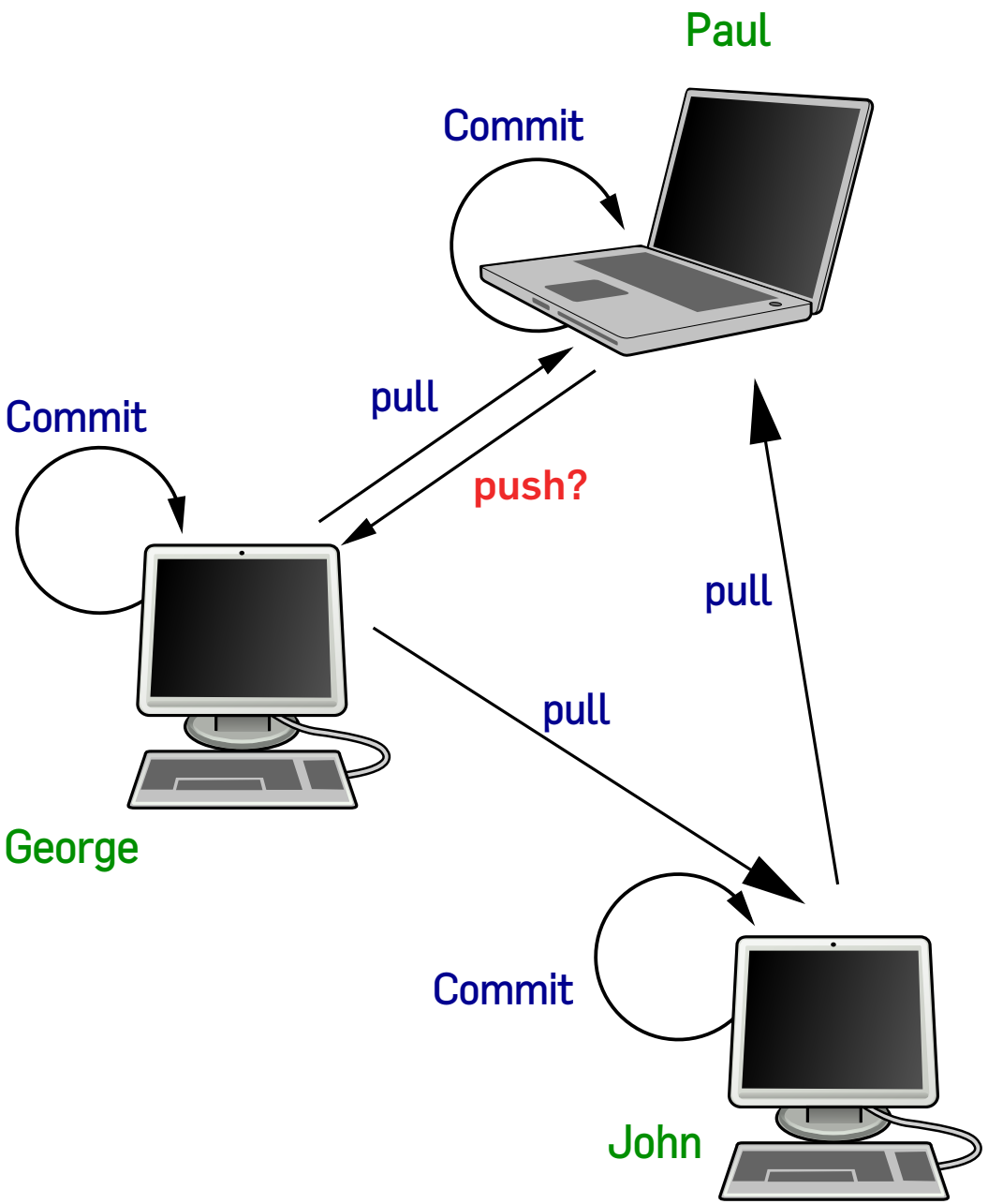
- collaborating
- conflict resolution
- branching

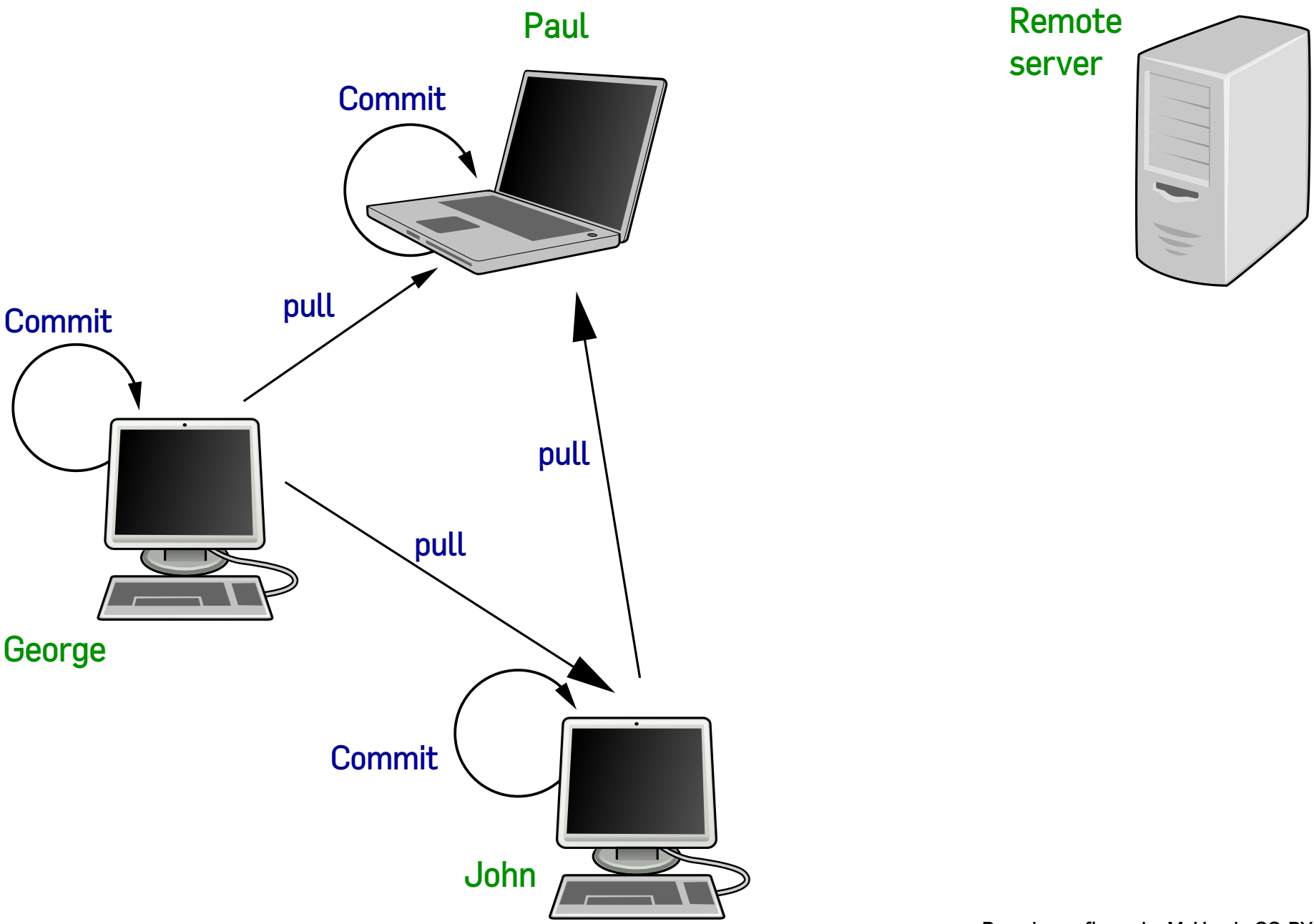


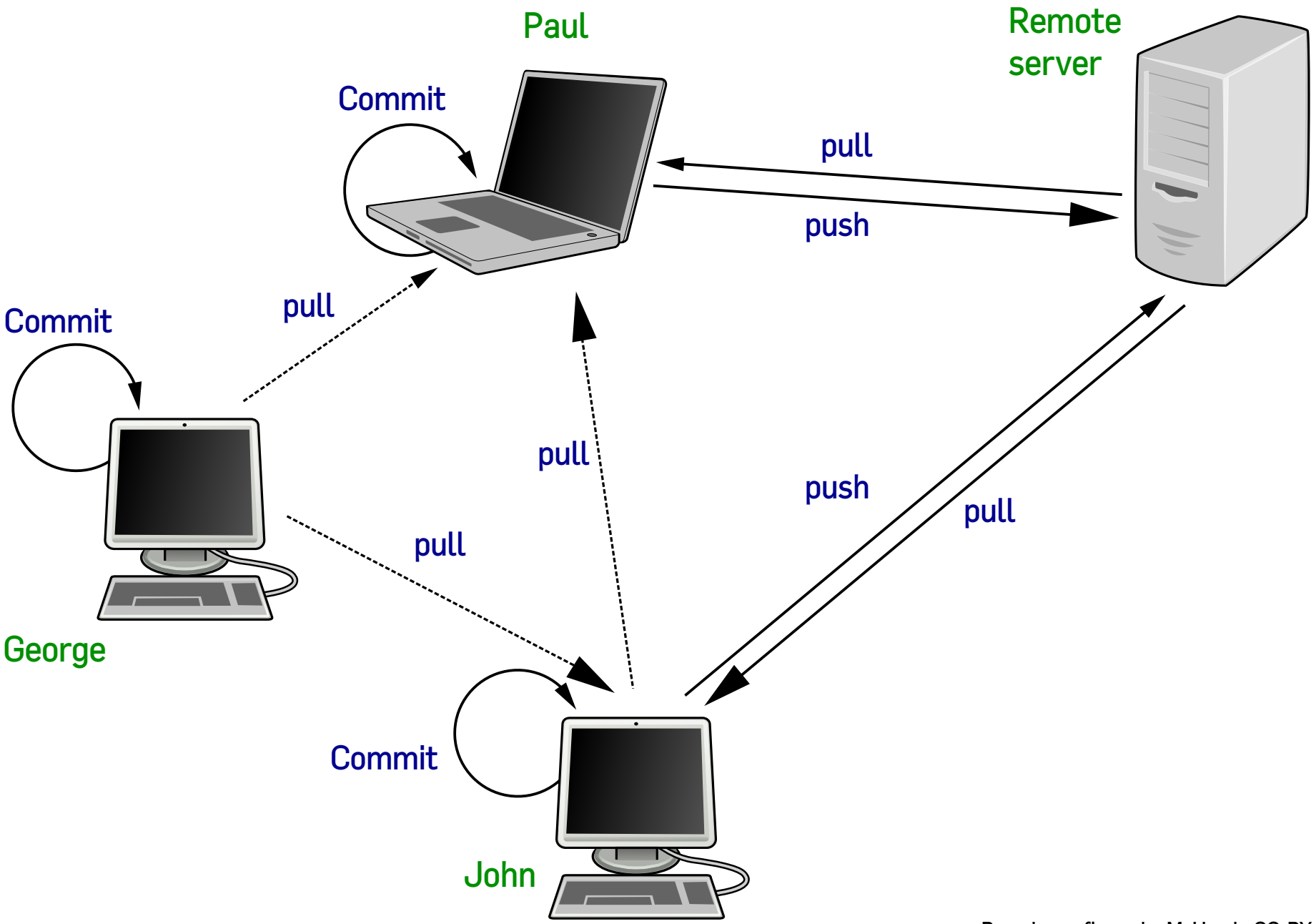


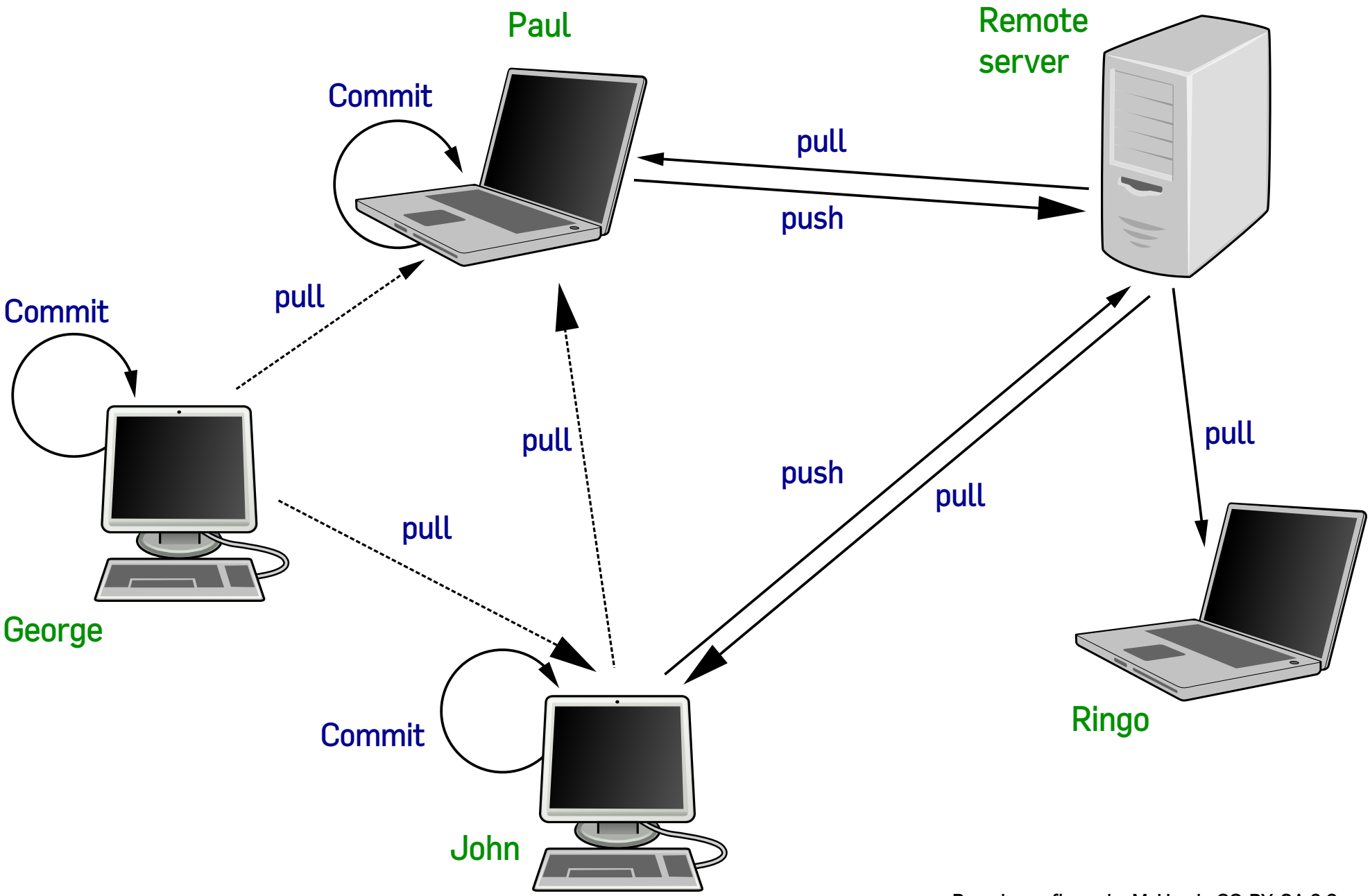


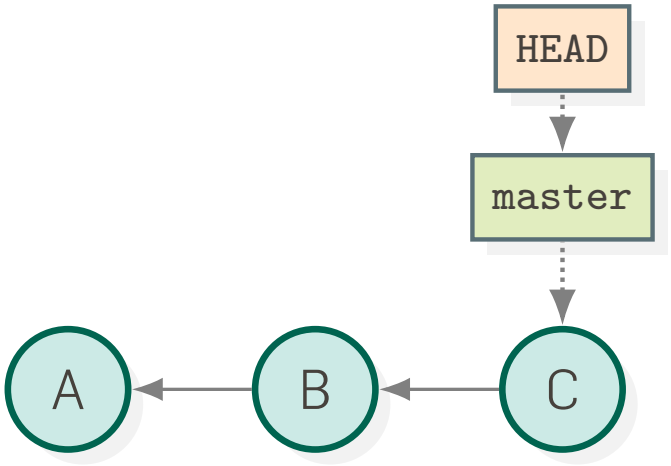


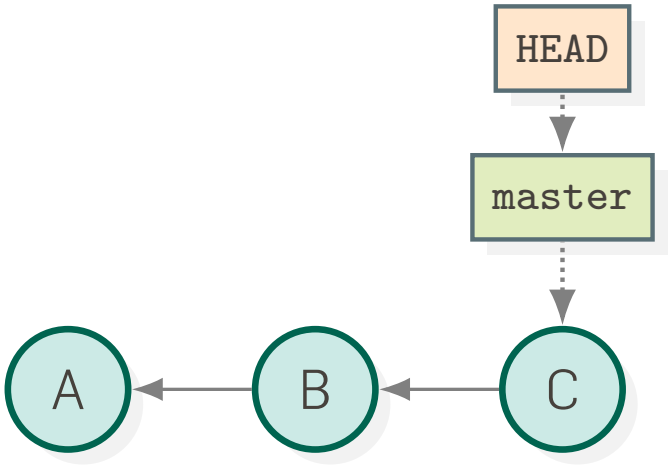




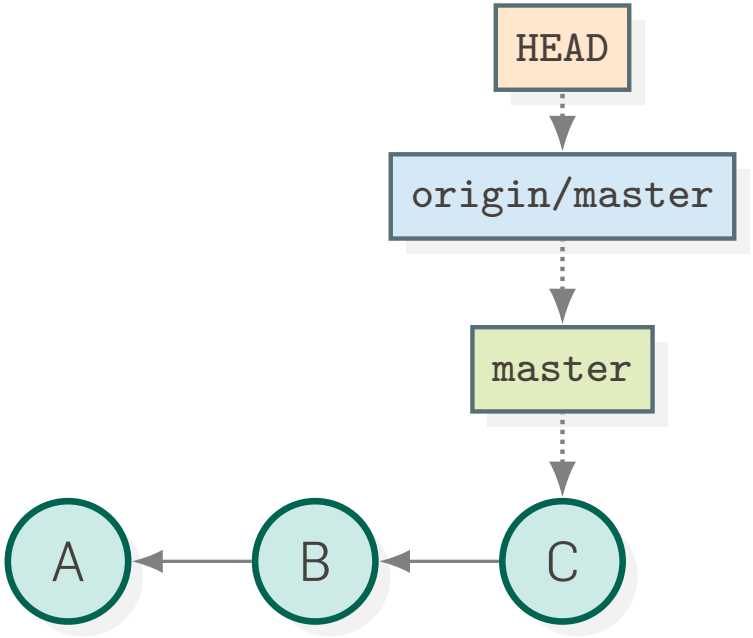





HEAD
master
A
B
C

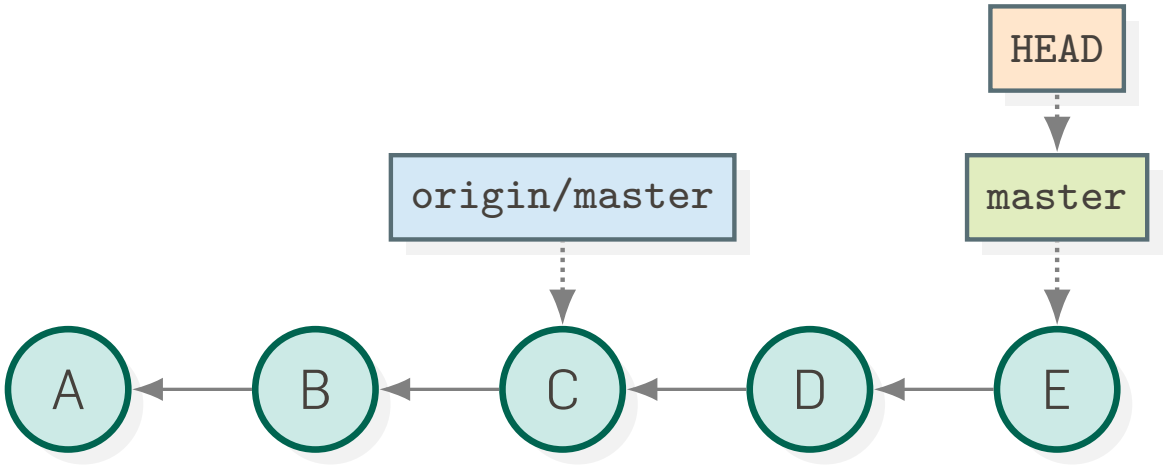


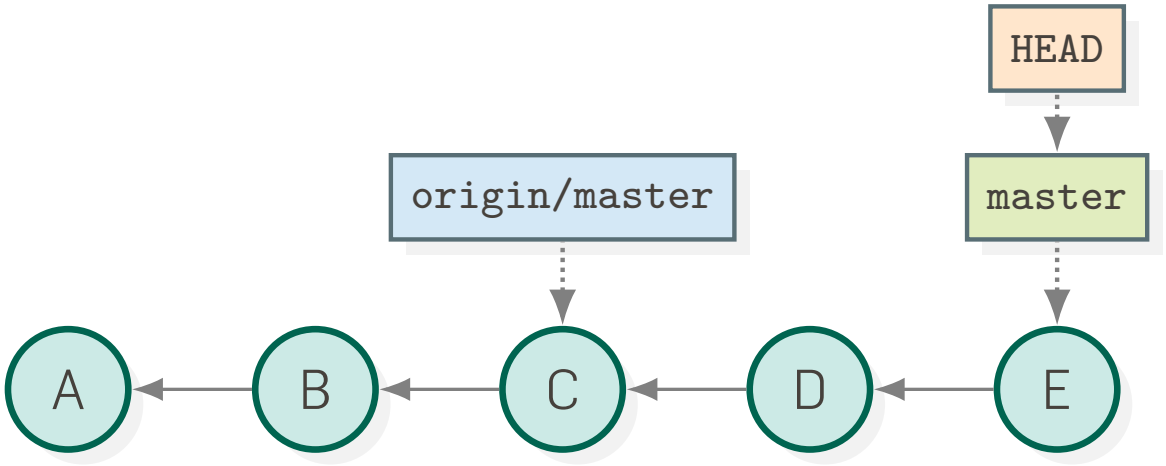

```
git remote add origin git@github.com:user/repo.git
git remote add john-usb E:\john_repo
git remote add ftp-origin ftp://host.xz/path/to/repo.git/
…
```



`git push origin master`

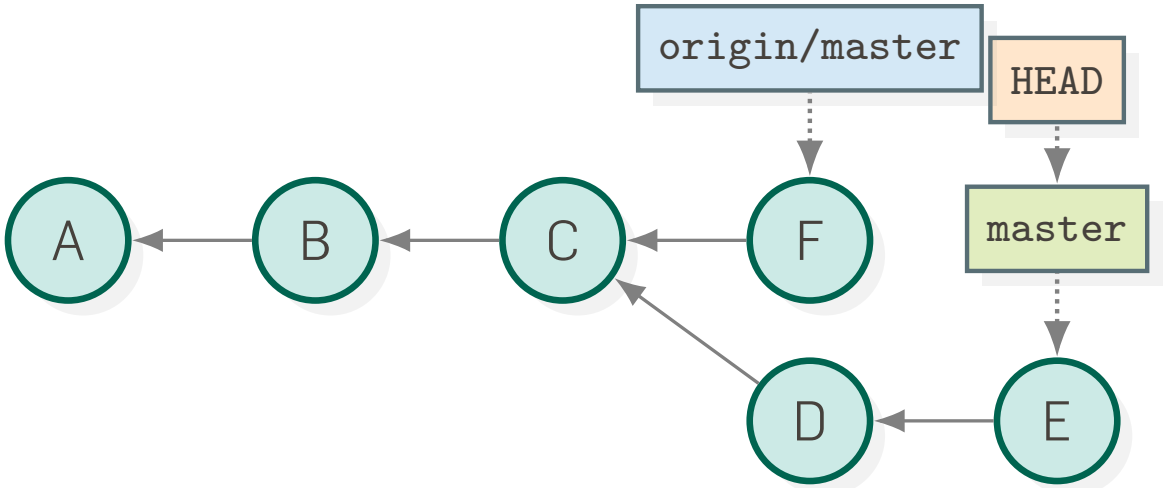
(or simply `git push`)

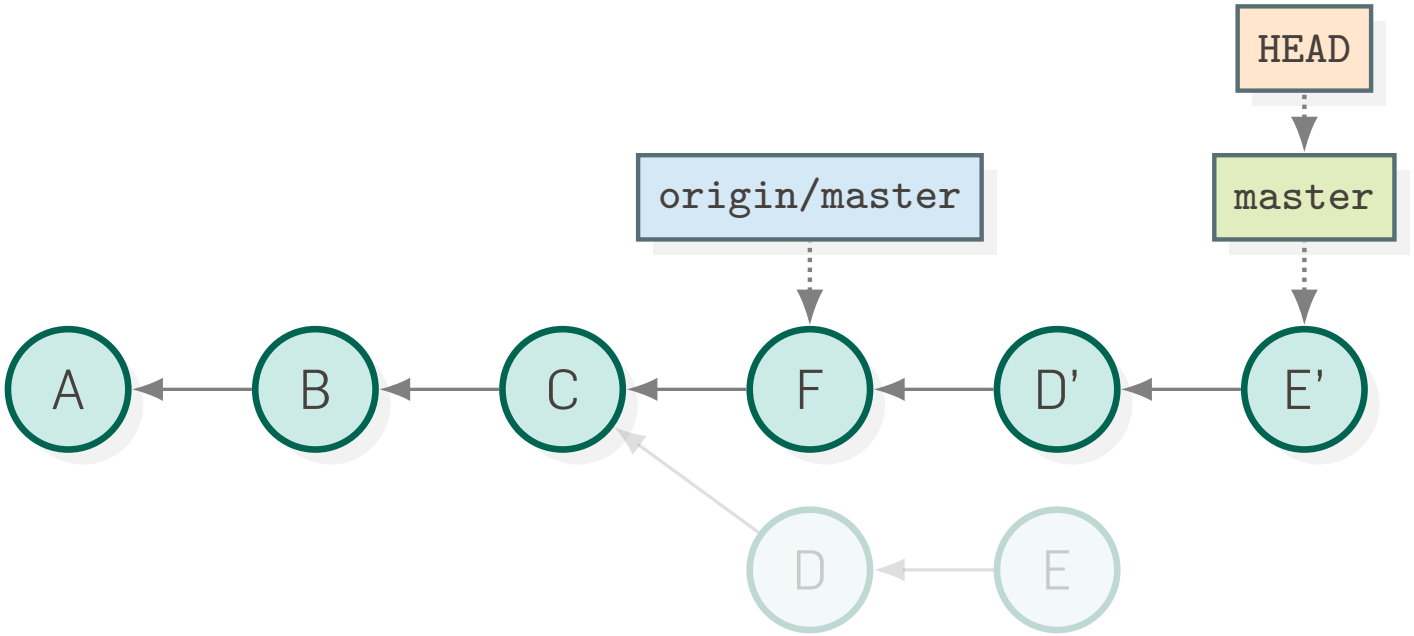

HEAD
origin/master
master
A
B
C
D
E



What happened on our remote? Let's have a look...

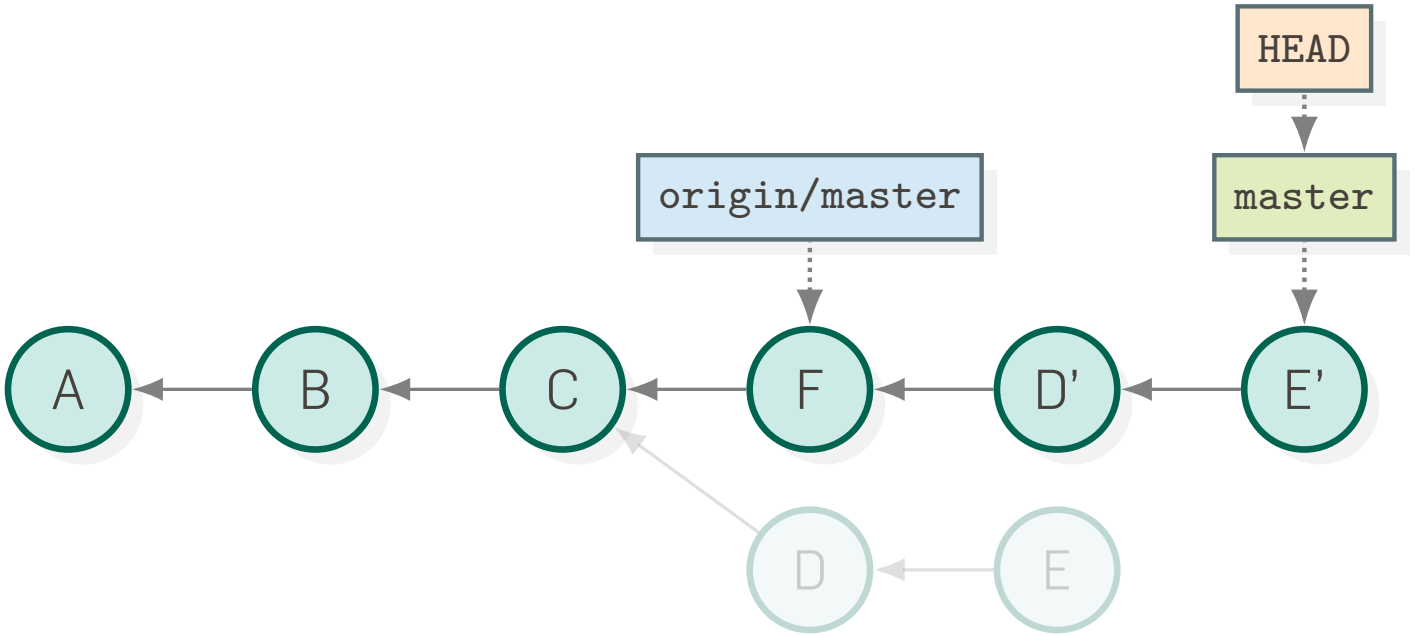
```
git fetch origin
```


origin/master
HEAD
master
A
B
C
F
D
E

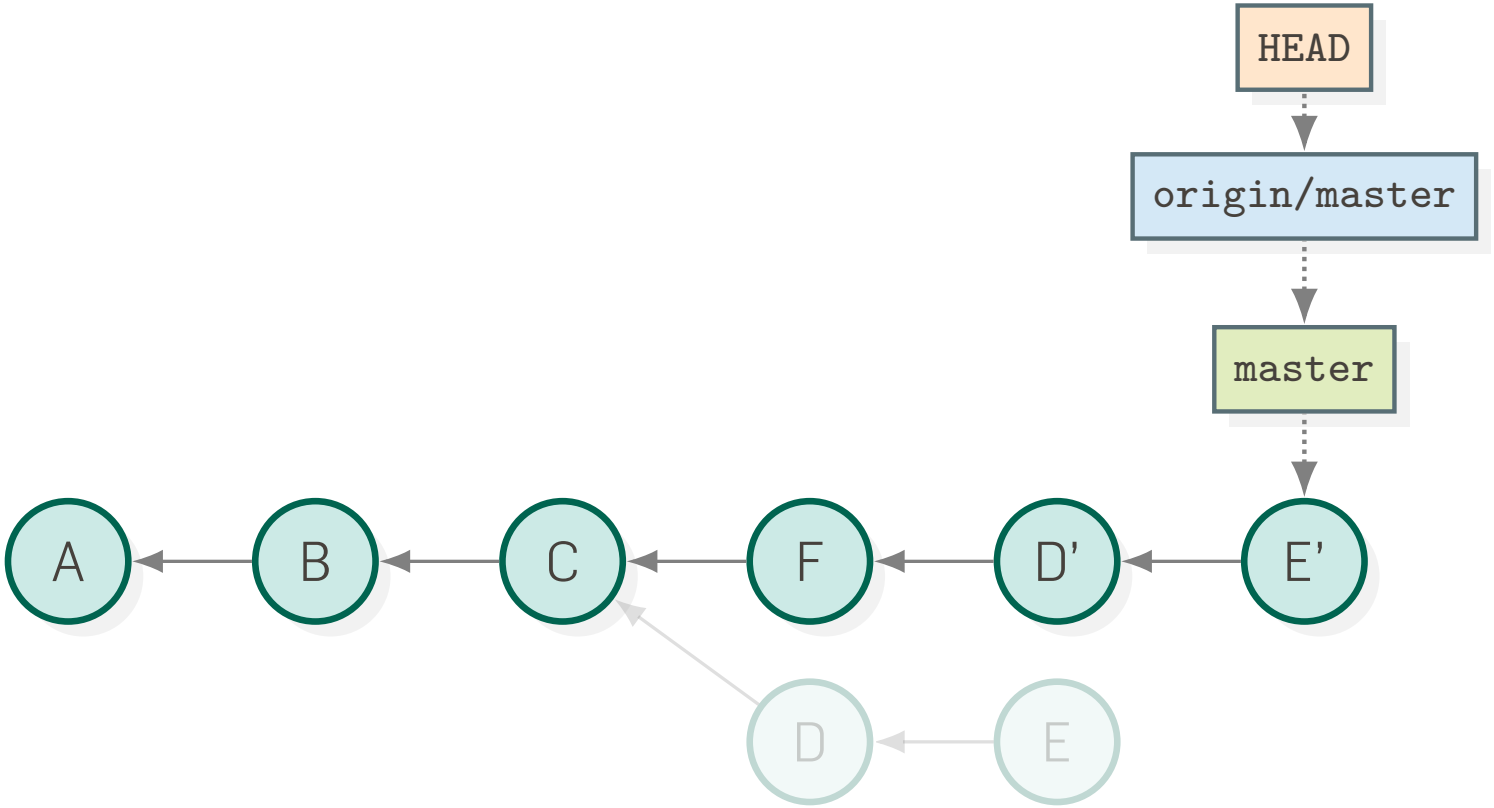


`git rebase origin/master`

(but you don't need it, because...)




```
git pull --rebase
```




```
git push
```

TO SUMMARIZE...

The first time...

```
$ git clone <url>
# for instance,
# git clone https://github.com/user/repo.git
```

Then...

```
$ cd <repo>
# make some changes...
$ git add <files>
$ git commit -m"<commit message>"
# ...
# when you want to share:
$ git pull --rebase # any changes on the remote?
$ git push
```

THE DREADFUL CONFLICTS

THE DREADFUL CONFLICT

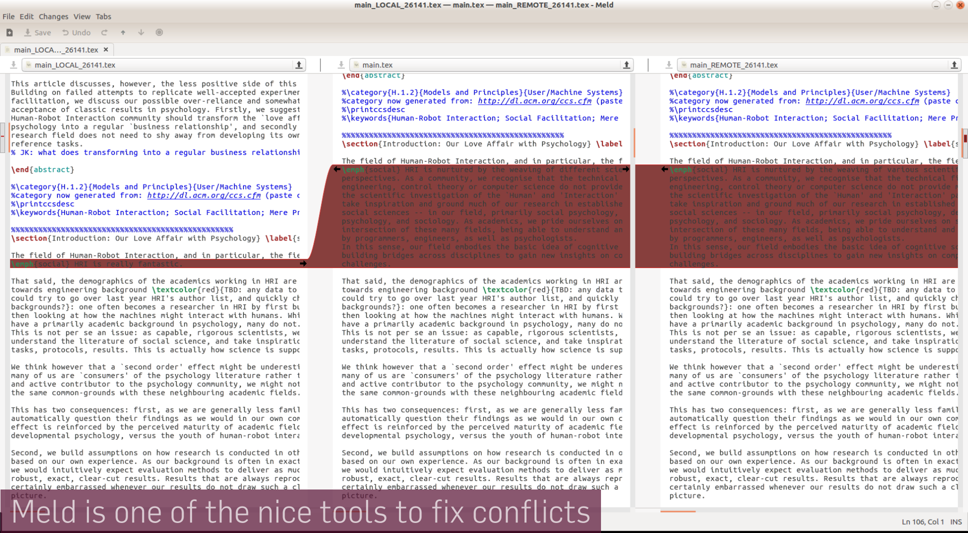
While peacefully editing your last (great) report...

```
$ git pull --rebase john master
First, rewinding head to replay your work on top of it...
Applying: Better terminology
Using index info to reconstruct a base tree...
M        controller.tex
Falling back to patching base and 3-way merge...
Auto-merging controller.tex
CONFLICT (content): Merge conflict in controller.tex
error: Failed to merge in the changes.
Patch failed at 0001 Better terminology
The copy of the patch that failed is found in: .git/rebase-apply/patch

When you have resolved this problem, run "git rebase --continue".
If you prefer to skip this patch, run "git rebase --skip" instead.
To check out the original branch and stop rebasing, run "git rebase --abort".
```

A conflict happens when two modifications of a given file overlap

Two persons can modify the same file at the same time, as long as they do not work on the same region of the file.

```
$ git pull --rebase john master
# conflict!
$ git mergetool
```



```
This article discusses, however, the less positive side of this
Building on failed attempts to replicate well-accepted experimer
facilitation, we discuss our possible over-reliance and somewhat
acceptance of classic results in psychology. Firstly, we suggest
Human-Robot Interaction community should transform the `love aff
psychology into a regular `business relationship', and secondly
research field does not need to shy away from developing its owr
reference tasks.
% JK: what does transforming into a regular business relationshi

\end{abstract}

%\category{H.1.2}{Models and Principles}{User/Machine Systems}
%category now generated from: http://dl.acm.org/ccs.cfm (paste c
%\printccsdesc
%\keywords{Human-Robot Interaction; Social Facilitation; Mere Pr

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\section{Introduction: Our Love Affair with Psychology} \label{s

The field of Human-Robot Interaction, and in particular, the fie
\emph{social} HRI is really fantastic.

That said, the demographics of the academics working in HRI are
towards engineering background \textcolor{red}{TBD: any data to
could try to go over last year HRI's author list, and quickly ch
backgrounds?}: one often becomes a researcher in HRI by first bu
then looking at how the machines might interact with humans. Whi
have a primarily academic background in psychology, many do not.
This is not per se an issue: as capable, rigorous scientists, we
understand the literature of social science, and take inspiratic
tasks, protocols, results. This is actually how science is suppc

We think however that a `second order' effect might be underesti
many of us are `consumers' of the psychology literature rather t
and active contributor to the psychology community, we might not
the same common-grounds with these neighbouring academic fields.

This has two consequences: first, as we are generally less famil
automatically question their findings as we would in our own cor
effect is reinforced by the perceived maturity of academic fielc
developmental psychology, versus the youth of human-robot intera

Second, we build assumptions on how research is conducted in oth
based on our own experience. As our background is often in exact
we would intuitively expect evaluation methods to deliver as muc
robust, exact, clear-cut results. Results that are always reproc
certainly embarrassed whenever our results do not draw such a cl
picture.
```

```
\end{abstract}

%\category{H.1.2}{Models and Principles}{User/Machine Systems}
%category now generated from: http://dl.acm.org/ccs.cfm (paste
%\printccsdesc
%\keywords{Human-Robot Interaction; Social Facilitation; Mere

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\section{Introduction: Our Love Affair with Psychology} \label

The field of Human-Robot Interaction, and in particular, the f
\emph{social} HRI is nurtured by the weaving of different scie
perspectives. As a community, we recognise that the technical
engineering, control theory or computer science do not provide
the scientific investigation of the `Human' and 'Interaction'
take inspiration and ground much of our research in establishe
social sciences -- in our field, primarily social psychology,
psychology, and sociology. As academics, we pride ourselves on
intersection of these many fields, being able to understand an
by programmers, engineers, as well as psychologists.
In this sense, our field embodies the basic idea of cognitive
building bridges across disciplines to gain new insights on co
challenges.

That said, the demographics of the academics working in HRI ar
towards engineering background \textcolor{red}{TBD: any data t
could try to go over last year HRI's author list, and quickly
backgrounds?}: one often becomes a researcher in HRI by first
then looking at how the machines might interact with humans. W
have a primarily academic background in psychology, many do no
This is not per se an issue: as capable, rigorous scientists,
understand the literature of social science, and take inspirat
tasks, protocols, results. This is actually how science is sup

We think however that a `second order' effect might be underes
many of us are `consumers' of the psychology literature rather
and active contributor to the psychology community, we might n
the same common-grounds with these neighbouring academic field

This has two consequences: first, as we are generally less fam
automatically question their findings as we would in our own c
effect is reinforced by the perceived maturity of academic fie
developmental psychology, versus the youth of human-robot inte

Second, we build assumptions on how research is conducted in o
based on our own experience. As our background is often in exa
we would intuitively expect evaluation methods to deliver as m
robust, exact, clear-cut results. Results that are always repr
certainly embarrassed whenever our results do not draw such a
picture.
```

```
\end{abstract}

%\category{H.1.2}{Models and Principles}{User/Machine Systems}
%category now generated from: http://dl.acm.org/ccs.cfm (paste c
%\printccsdesc
%\keywords{Human-Robot Interaction; Social Facilitation; Mere Pr

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\section{Introduction: Our Love Affair with Psychology} \label{s

The field of Human-Robot Interaction, and in particular, the fie
\emph{social} HRI is nurtured by the weaving of various scientif
perspectives. As a community, we recognise that the technical fi
engineering, control theory or computer science do not provide m
the scientific investigation of the `Human' and 'Interaction' pa
take inspiration and ground much of our research in established
social sciences -- in our field, primarily social psychology, de
psychology, and sociology. As academics, we pride ourselves on s
intersection of these many fields, being able to understand and
by programmers, engineers, as well as psychologists.
In this sense, our field embodies the basic idea of cognitive sc
building bridges across disciplines to gain new insights on comp
challenges.

That said, the demographics of the academics working in HRI are
towards engineering background \textcolor{red}{TBD: any data to
could try to go over last year HRI's author list, and quickly ch
backgrounds?}: one often becomes a researcher in HRI by first bu
then looking at how the machines might interact with humans. Whi
have a primarily academic background in psychology, many do not.
This is not per se an issue: as capable, rigorous scientists, we
understand the literature of social science, and take inspiratic
tasks, protocols, results. This is actually how science is suppc

We think however that a `second order' effect might be underesti
many of us are `consumers' of the psychology literature rather t
and active contributor to the psychology community, we might not
the same common-grounds with these neighbouring academic fields.

This has two consequences: first, as we are generally less famil
automatically question their findings as we would in our own cor
effect is reinforced by the perceived maturity of academic fielc
developmental psychology, versus the youth of human-robot intera

Second, we build assumptions on how research is conducted in oth
based on our own experience. As our background is often in exact
we would intuitively expect evaluation methods to deliver as muc
robust, exact, clear-cut results. Results that are always reproc
certainly embarrassed whenever our results do not draw such a cl
picture.
```

Meld is one of the nice tools to fix conflicts

SOCIAL CODING: GITHUB WORKFLOW

This repository Search Pull requests Issues Gist

morse-simulator / **morse**

Unwatch 28 Unstar 145 Fork 74

Code Issues 69 Pull requests 2 Wiki Pulse Graphs Settings

The Modular OpenRobots Simulation Engine http://morse-simulator.github.io/ — Edit

4,174 commits **10** branches **53** releases **30** contributors

Branch: **master** New pull request New file Find file HTTPS https://github.com/morse Download ZIP

adegroote [doc] Document more middleware addition Latest commit `38fa4af` 3 days ago

addons	[builder] fix few occurrences of removed method	a year ago
bin	[bin/morse_sync] Make sure to call it with the same python exec than ...	6 days ago
bindings	[bindings] Prepare for 1.4	7 days ago
config	[cmake] python 3.5 exists since September 2015	a month ago
data	[human] Removes all the code and support for the legacy human avatar	a month ago
doc	[doc] Document more middleware addition	3 days ago
examples	[builder] Rename Environment.set_simulator_frequency in Environemnt.s...	8 days ago
src	[mw/mavlink] Make sure to match only message of the 'good' type	3 days ago
testing	[builder] Rename Environment.set_simulator_frequency in Environemnt.s...	8 days ago
tools	[human] Removes all the code and support for the legacy human avatar	a month ago
.gitignore	add scene.*blend and eclipse files to gitignore	3 years ago
.mailmap	Added a mailmap to group variants of dev emails	3 years ago
.travis.bash	[travis] Upgrade the architecture for trusty	4 months ago
.travis.yml	[travis] Upgrade the architecture for trusty	4 months ago
AUTHORS	[doc] Fix several typos in credits	7 days ago
CITATION	[doc] Added a CITATION file	3 years ago
CMakeLists.txt	[bin/morse_sync] Make sure to call it with the same python exec than ...	6 days ago

GitHub

Atlassian Bitbucket Features Pricing Find

MakeHuman

ACTIONS

Clone
Compare
Fork

NAVIGATION

Overview
Source
Commits
Branches
Pull requests 1
Downloads

Séverin Lemaignan / MakeHuman

Source

default | MakeHuman /

blendertools			
buildscripts			
docs			
makehuman			
maketarget-standalone			
.hgeol	23 B	2014-02-03	Ensure use of LF native line endings for all text files, to avoid careless windows developers changing the line endings.
.hgignore	574 B	2014-03-18	merge with stable
.hgtags	47 B	2014-03-15	Cleanup hgtags
README	1.5 KB	2014-03-23	Add url to development tracker for dev status to readme

```
MakeHuman
=========

Makehuman is a completely free, innovative and professional software for the
modelling of 3-Dimensional humanoid characters. This is the official source
repository of the MakeHuman project.

Official website: http://www.makehuman.org
Development status: http://bugtracker.makehuman.org

License
-------

MakeHuman's source code and its mesh data is distributed freely under the AGPL3
license (see license.txt). Content created using the MakeHuman application is
released under the liberal CC0 license. For more details, refer to these pages:

* https://www.makehuman.org/doc/node/the_makehuman_application.html
* https://www.makehuman.org/doc/node/makehuman_mesh_licenses.html

Licenses for dependencies are included in the licenses folder.

Instructions
------------
```

BitBucket

gitlab.com

GitLab

Back to Group

- Project
- Activity
- Files
- Commits
- Network
- Graphs
- Milestones
- Issues 712
- Merge Requests 52
- Labels

darby

GitLab.org / GitLab Community Edition

Search in this project

master | gitlab-ce | Download zip

Name	Last Update	Last Commit > 6ae806b1 – Merge branch 'fix-link-to-2fa' into 'master'	History
app	a day ago	Achilleas Pipinellis Fix link to 2fa help page. Closes #2055	
bin	2 months ago	Robert Speicher Remove Guard	
config	3 days ago	Marin Jankovski Merge branch 'set-omniauth-full-host' into 'mast…	
db	about 23 hours ago	Marin Jankovski Check if session_expire_delay column exists bef…	
doc	a day ago	Marin Jankovski Merge branch 'master' of gitlab.com:gitlab-org/g…	
docker	7 days ago	Job van der Voort Merge branch 'chef-docker' into 'master'	
features	6 days ago	Stan Hu Add support for destroying project milestones	
lib	2 days ago	Jacob Vosmaer Don't stop if database.sql.gz already exists	
log	4 years ago	gitlabhq init commit	
public	about a month ago	Dmitriy Zaporozhets Replace old logo with new one	
scripts	28 days ago	Kamil Trzciński Added missing packages required by docker builds	
spec	2 days ago	Douwe Maan Merge branch 'rs-security-spec-speed' into 'master'	
tmp	about a year ago	Robert Speicher Make sure important directories exist in git	
vendor/assets	9 days ago	Dmitriy Zaporozhets Add nice scroll for sidebar	

GitLab – open-source You can install it on your own server

This repository Search Pull requests Issues Gist

morse-simulator / **morse**

Unwatch 28 Unstar 145 Fork 74

Code Issues 69 Pull requests 2 Wiki Pulse Graphs Settings

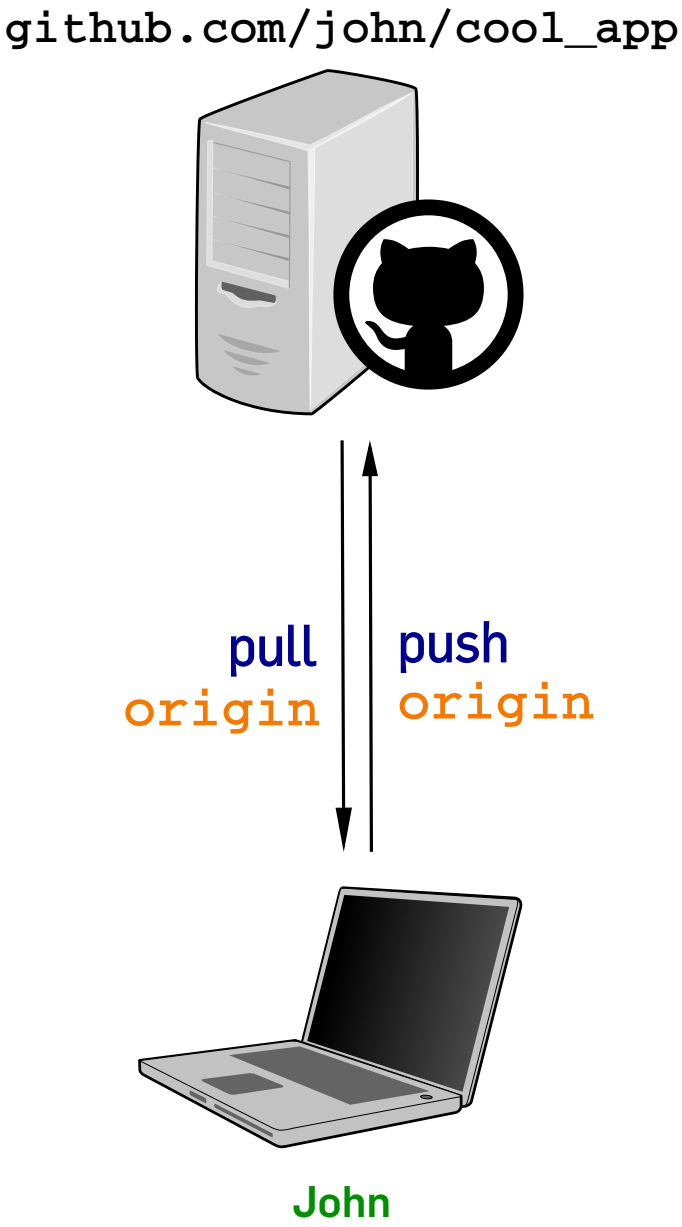
The Modular OpenRobots Simulation Engine http://morse-simulator.github.io/ — Edit

4,174 commits **10** branches **53** releases **30** contributors

Branch: **master** New pull request New file Find file HTTPS https://github.com/morse Download ZIP

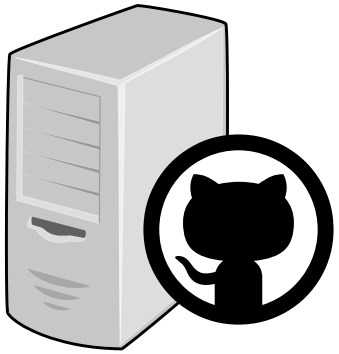
adegroote [doc] Document more middleware addition Latest commit `38fa4af` 3 days ago

addons	[builder] fix few occurrences of removed method	a year ago
bin	[bin/morse_sync] Make sure to call it with the same python exec than …	6 days ago
bindings	[bindings] Prepare for 1.4	7 days ago
config	[cmake] python 3.5 exists since September 2015	a month ago
data	[human] Removes all the code and support for the legacy human avatar	a month ago
doc	[doc] Document more middleware addition	3 days ago
examples	[builder] Rename Environment.set_simulator_frequency in Environemnt.s…	8 days ago
src	[mw/mavlink] Make sure to match only message of the 'good' type	3 days ago
testing	[builder] Rename Environment.set_simulator_frequency in Environemnt.s…	8 days ago
tools	[human] Removes all the code and support for the legacy human avatar	a month ago
.gitignore	add scene.*blend and eclipse files to gitignore	3 years ago
.mailmap	Added a mailmap to group variants of dev emails	3 years ago
.travis.bash	[travis] Upgrade the architecture for trusty	4 months ago
.travis.yml	[travis] Upgrade the architecture for trusty	4 months ago
AUTHORS	[doc] Fix several typos in credits	7 days ago
CITATION	[doc] Added a CITATION file	3 years ago
CMakeLists.txt	[bin/morse_sync] Make sure to call it with the same python exec than …	6 days ago

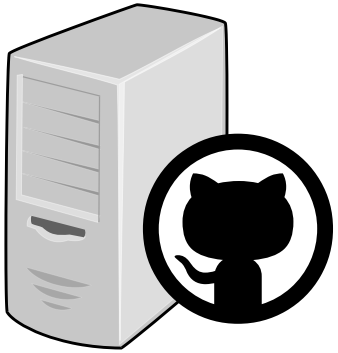

github.com/john/cool_app
pull
origin
push
origin
John

FORK
(a clone, really)

github.com/john/cool_app

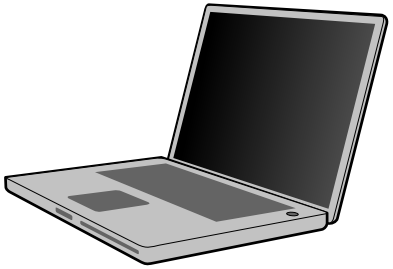


github.com/paul/cool_app

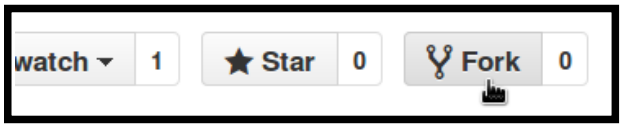


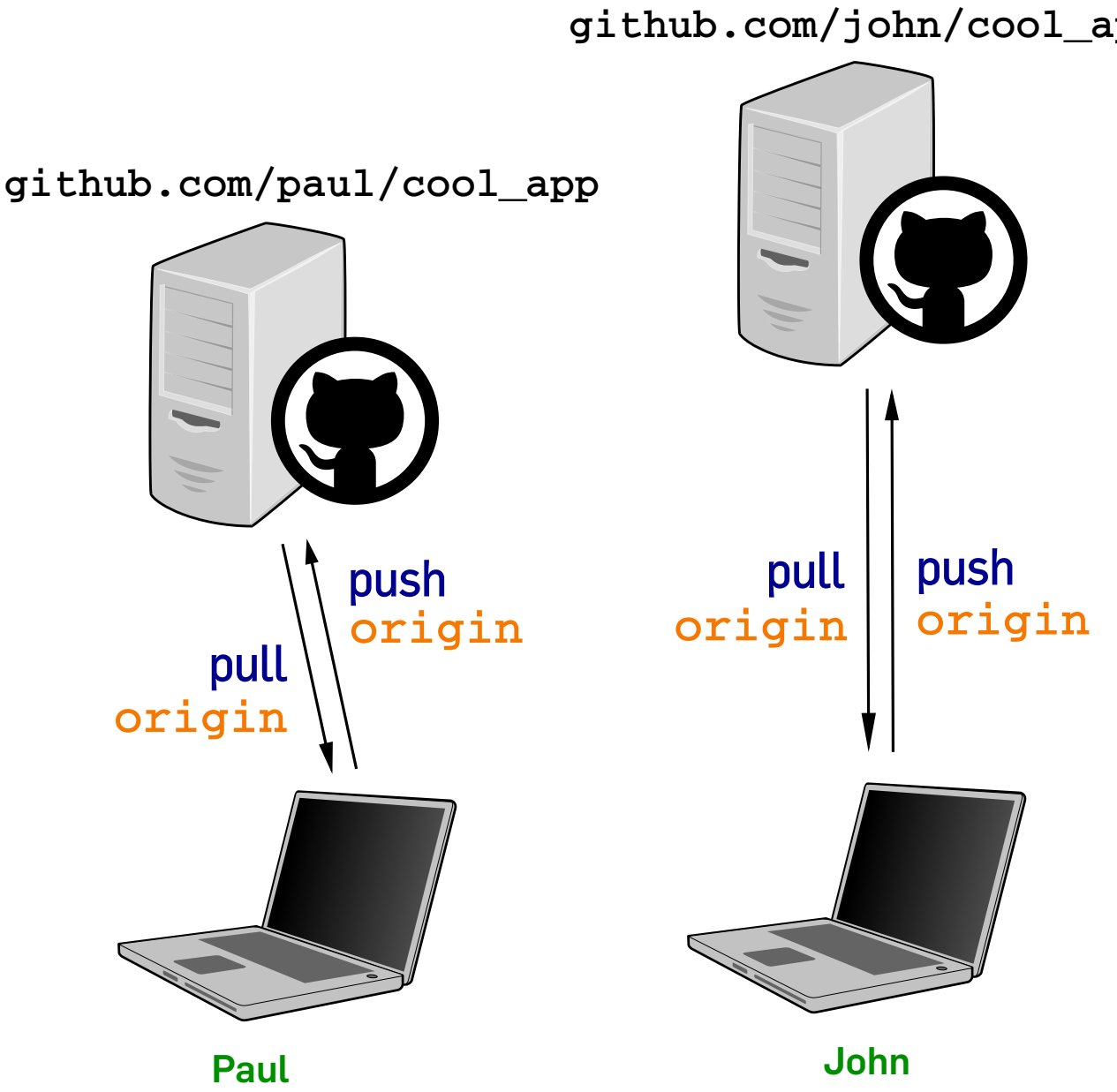
pull
origin

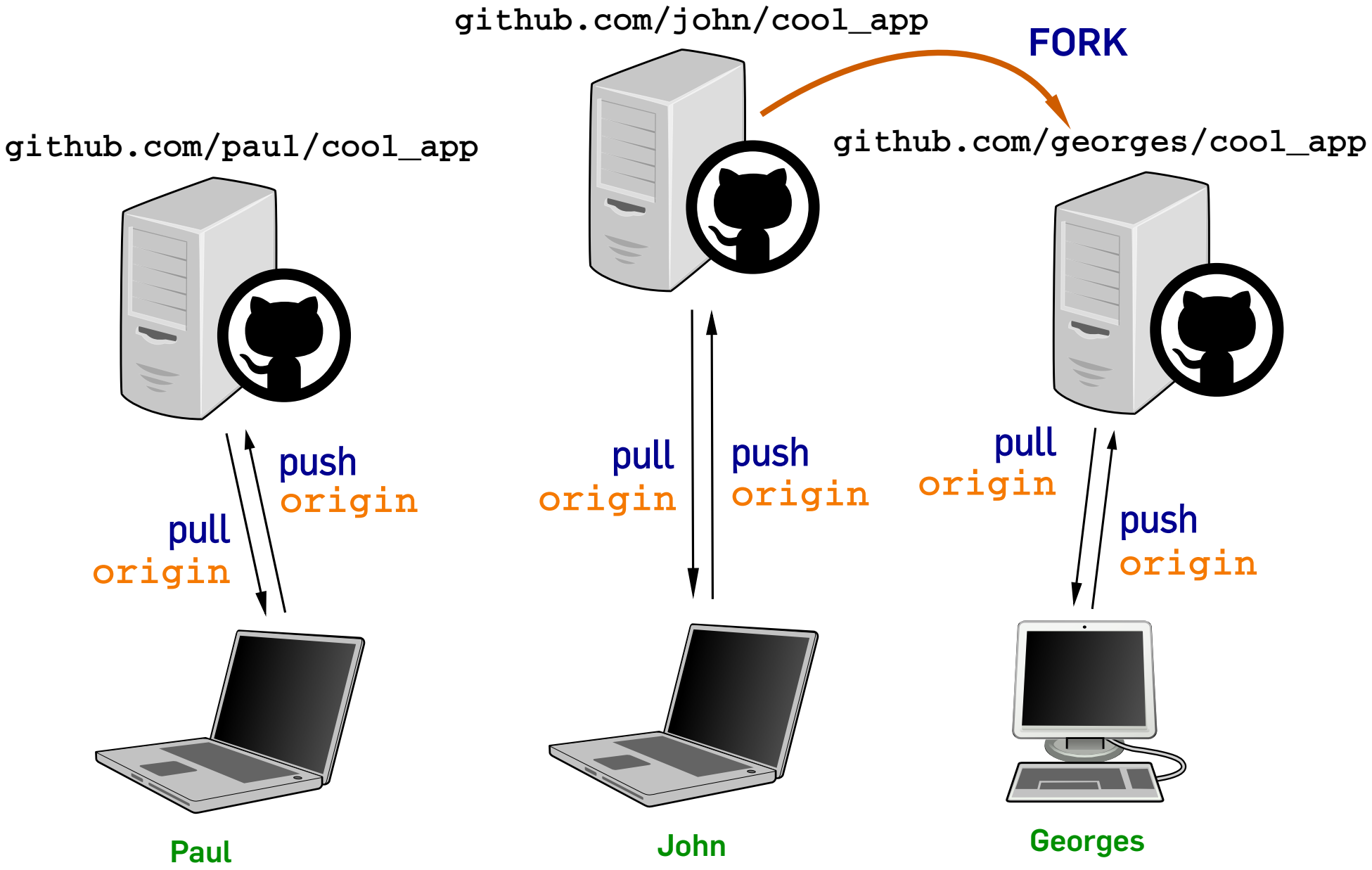
push
origin

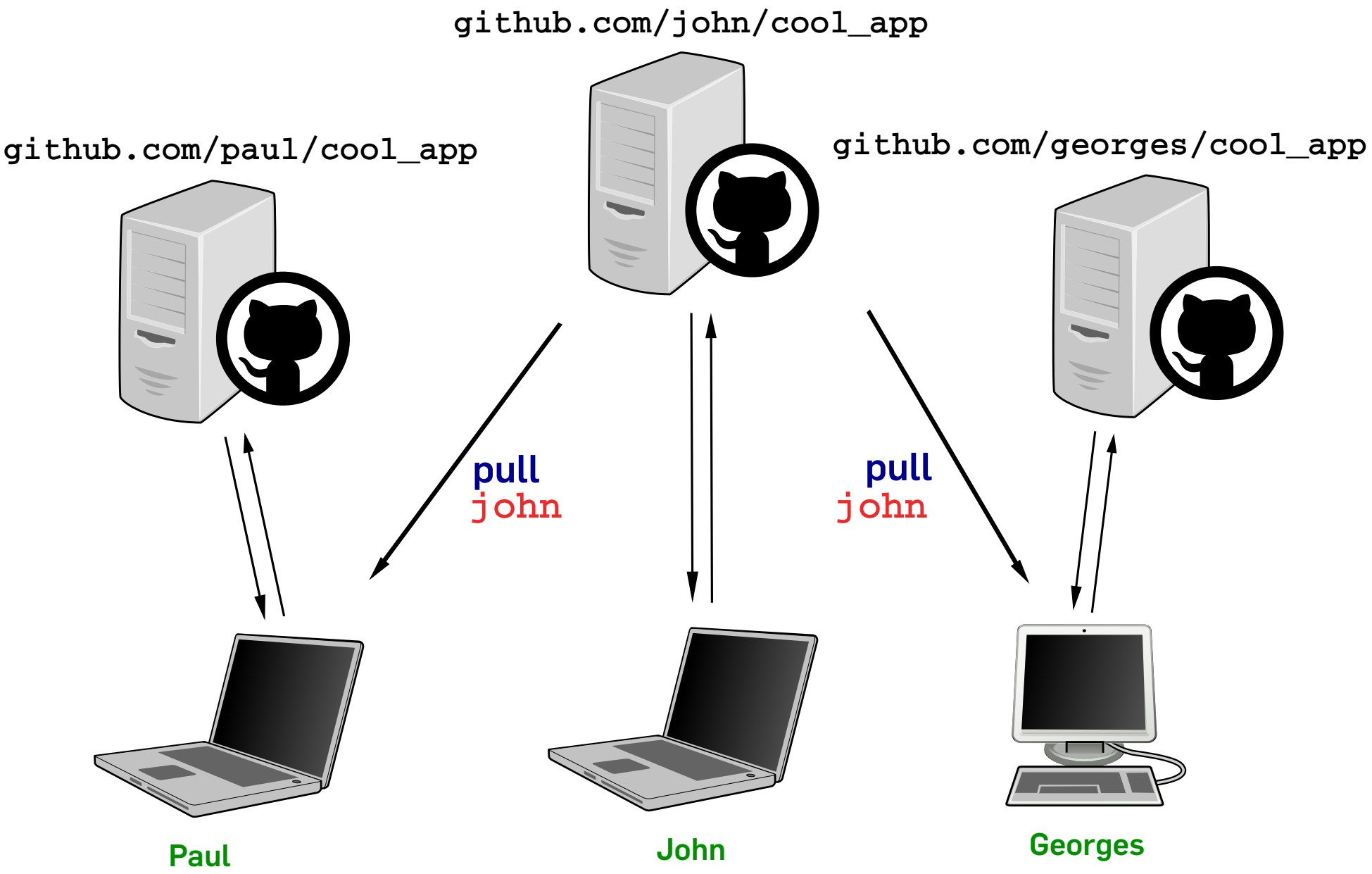


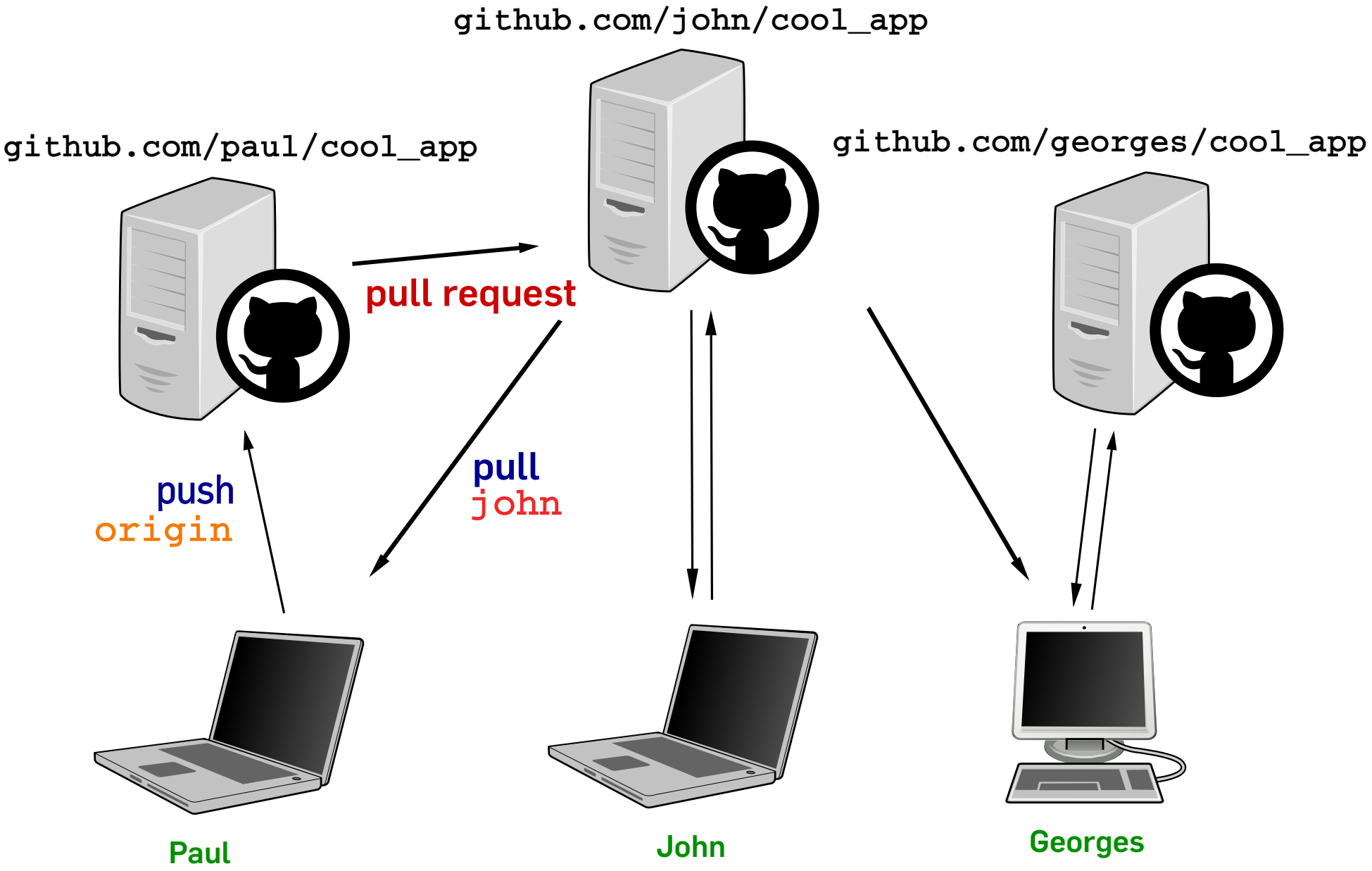
John



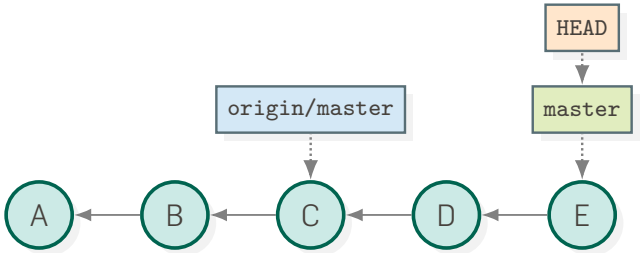

github.com/john/cool_app
github.com/paul/cool_app
push
origin
pull
origin
pull
origin
push
origin
Paul
John

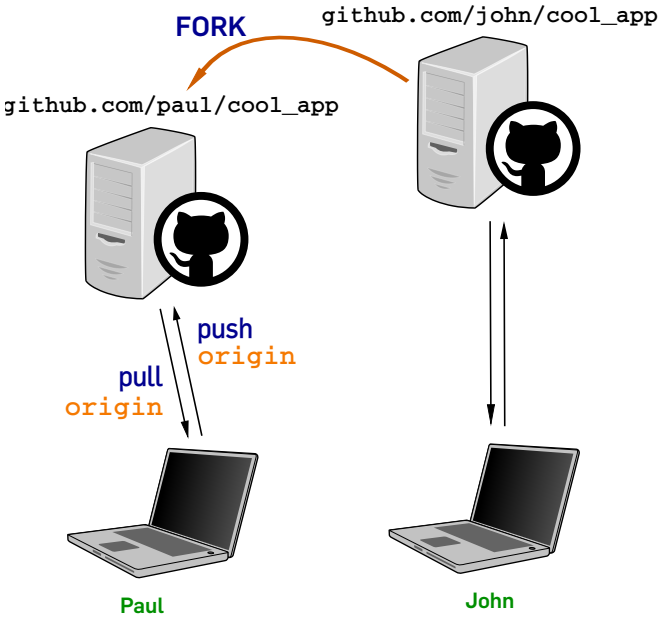

github.com/john/cool_app
FORK
github.com/paul/cool_app
github.com/georges/cool_app
pull
origin
push
origin
pull
origin
push
origin
pull
origin
push
origin
Paul
John
Georges


github.com/john/cool_app
github.com/paul/cool_app
github.com/georges/cool_app
pull
john
pull
john
Paul
John
Georges


github.com/john/cool_app
github.com/paul/cool_app
github.com/georges/cool_app
pull request
push
origin
pull
john
Paul
John
Georges

WHAT HAPPENED EXACTLY?



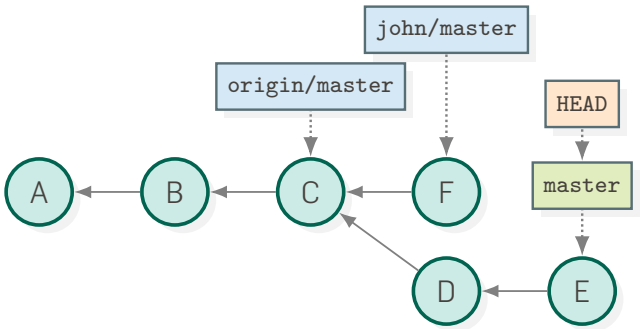


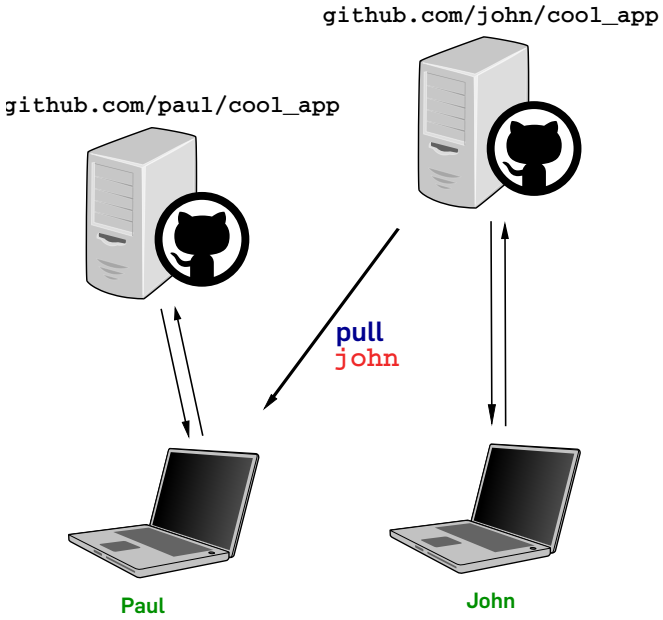
After forking on GitHub, Paul runs

`git clone https://github.com/paul/cool_app.git`

and he adds few local commits

WHAT HAPPENED EXACTLY?



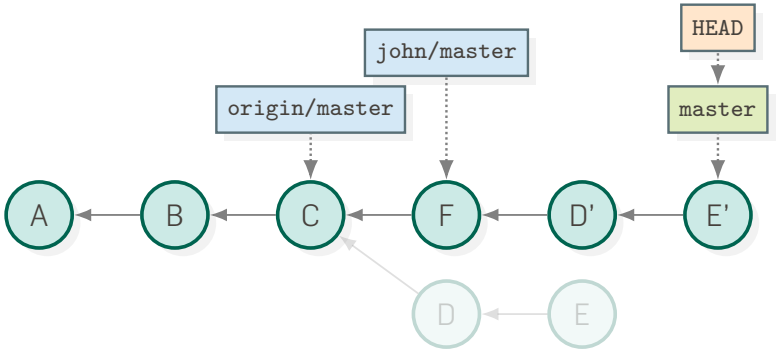


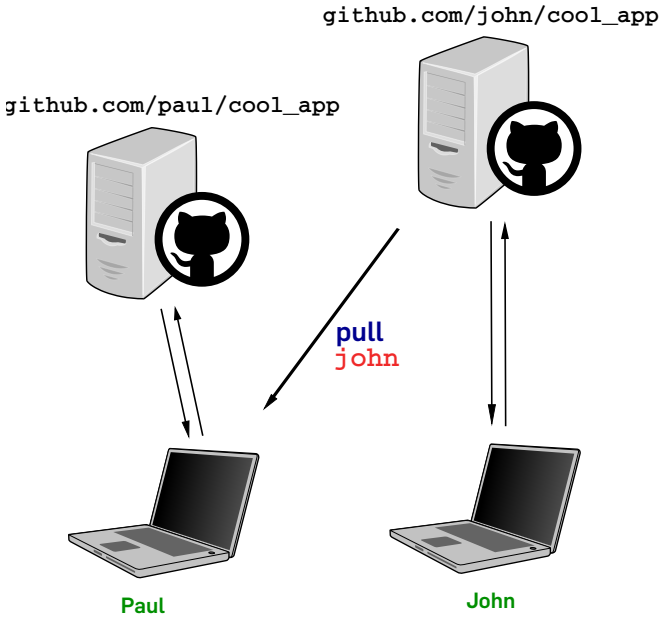
He would like to propose his changes to John

First, he needs to get the latest changes from John:

```
git add remote john https://github.com/john/cool_app.git
git fetch john
```

WHAT HAPPENED EXACTLY?



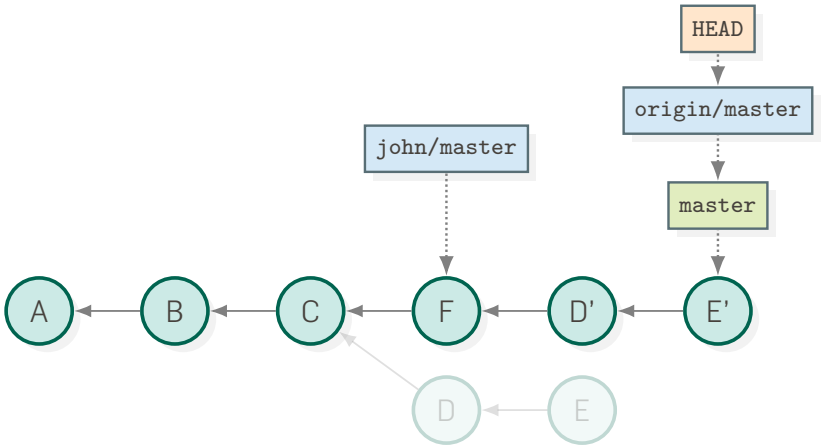


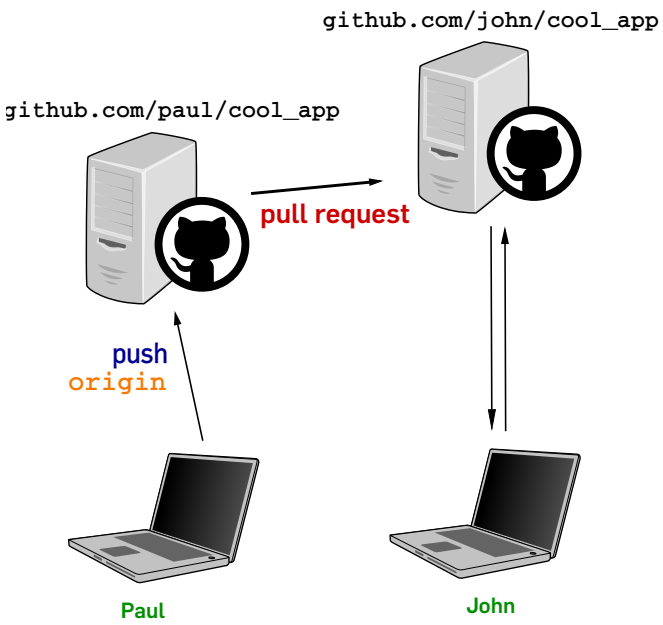
Paul rebases his `master` branch on John's one:

`git rebase john/master`

(actually, Paul would simply run `git pull --rebase john master`)

WHAT HAPPENED EXACTLY?



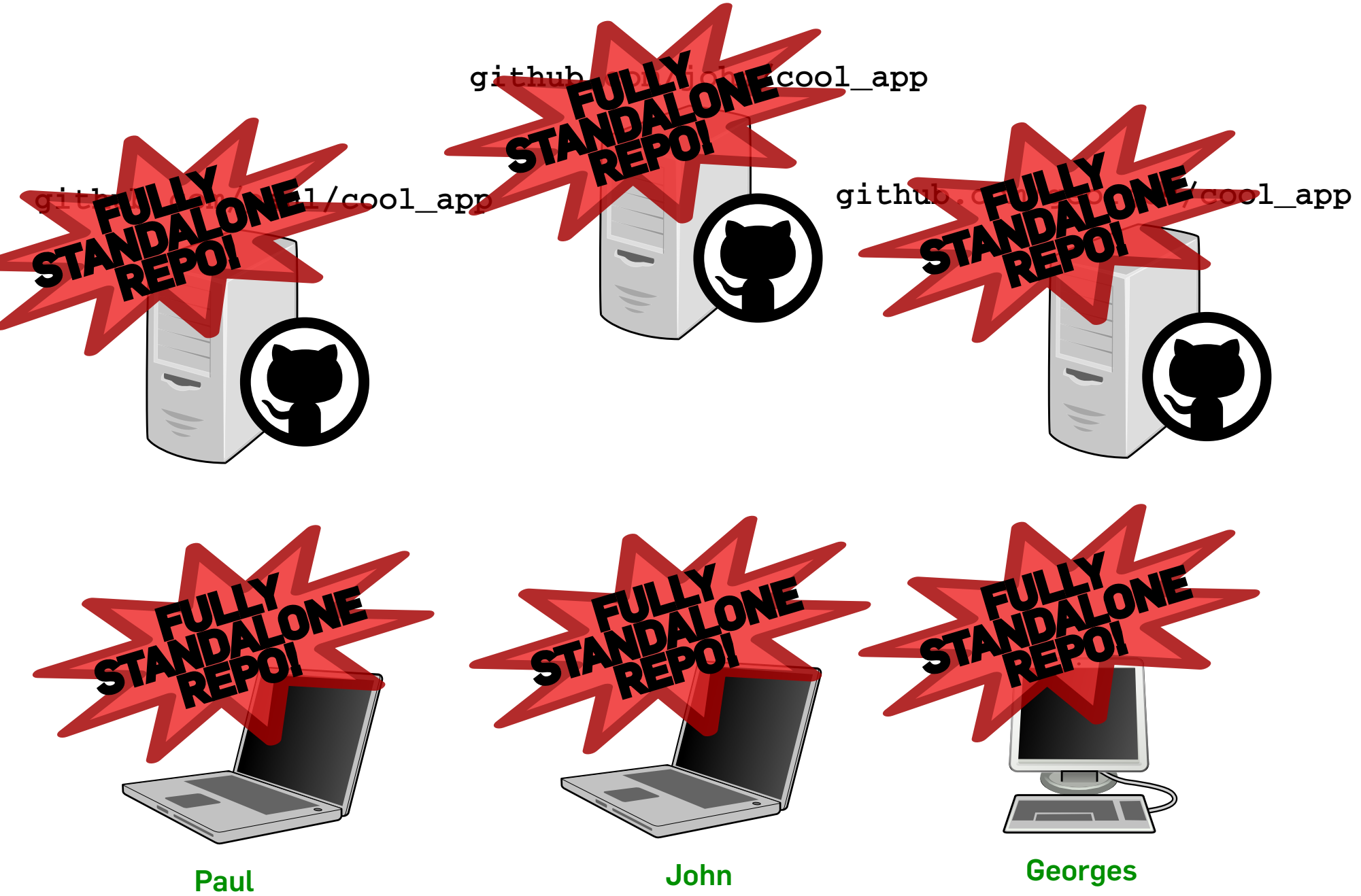


He pushes his commits to his own GitHub account:

`git push`

...and finally press the “Create a pull request” button in GitHub.

(what happens next on John's side is a story for another day :-)
But to make it short, he can press "Merge pull request" on his GitHub account if he is happy with the pull-request!)


FULLY STANDALONE REPO!
FULLY STANDALONE REPO!
FULLY STANDALONE REPO!
FULLY STANDALONE REPO!
FULLY STANDALONE REPO!
FULLY STANDALONE REPO!
Paul
John
Georges

GITLAB@BRL

We have our own 'GitHub': **git.brl.ac.uk**

If you can not yet login, drop an email to **itonline@uwe.ac.uk** and ask for access.

GitLab Projects Groups More Search or jump to...

You pushed to `production` at Séverin Lemaignan / msc-dissertations-mgt 42 minutes ago

Create merge request

Projects

New project

Your projects 47 Starred projects 0 Explore projects

Filter by name... Last updated

All Personal

M	Séverin Lemaignan / **msc-dissertations-mgt** Maintainer	0	0	Updated 47 minutes ago
B	Séverin Lemaignan / **BotAtWork** Maintainer	0	0	Updated 17 hours ago
B	Séverin Lemaignan / **brl-it-needs** Maintainer	0	0	Updated 6 days ago
S	Séverin Lemaignan / **social-interactions-game** Maintainer	0	0	Updated 1 week ago
P	Séverin Lemaignan / **pycozmo** Maintainer	0	0	Updated 1 week ago

GitLab@BRL

GitLab Projects Groups More Search or jump to...

B BotAtWork

- Project overview
  - Details
  - Activity
  - Releases
- Repository
- Issues 14
- Merge Requests 0
- CI / CD
- Operations
- Analytics

Collapse sidebar

BotAtWork

Project ID: 328

Star 0 Fork 0

68 Commits 1 Branch 0 Tags 24.4 MB Files 24.5 MB Storage

master bot-at-work / History Find file Web IDE Clone

Add LICENSE
Séverin Lemaignan authored in 30 seconds
812ca6f2

README MIT License Add CHANGELOG Add CONTRIBUTING Enable Auto DevOps Add Kubernetes cluster Set up CI/CD

Name	Last commit	Last update
assets	Small changes to textures	17 hours ago
doc	[doc] Basic README + 2 screenshots	22 hours ago
python	WIP uploading images to the server and setting the image on ro...	1 week ago

GitLab@BRL

COMMIT HYGIENE

“Show me the project history, I’ll tell you what coder you are”

- **Commit often!** Push when needed (or at the end of day)

Because commits are local (ie, private), **do commit often**: **mistakes are ok** as you can fix them before sharing with others.

COMMIT HYGIENE

“Show me the project history, I’ll tell you what coder you are”

- **Commit often!** Push when needed (or at the end of day)
- Write useful messages (no “`Fixed bug`” or “`New file`”)
- First line of commit messages < 72 characters

COMMIT HYGIENE

“Show me the project history, I’ll tell you what coder you are”

- **Commit often!** Push when needed (or at the end of day)
- Write useful messages (no “`Fixed bug`” or “`New file`”)
- First line of commit messages < 72 characters
- Tag important commits!

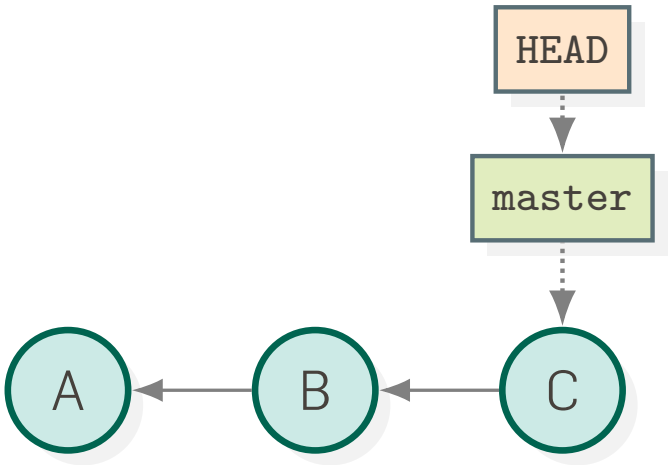
Notably, GitHub (amongst others) interpret tags as **releases** of your code.

one repo = one thing

make plenty of repos!

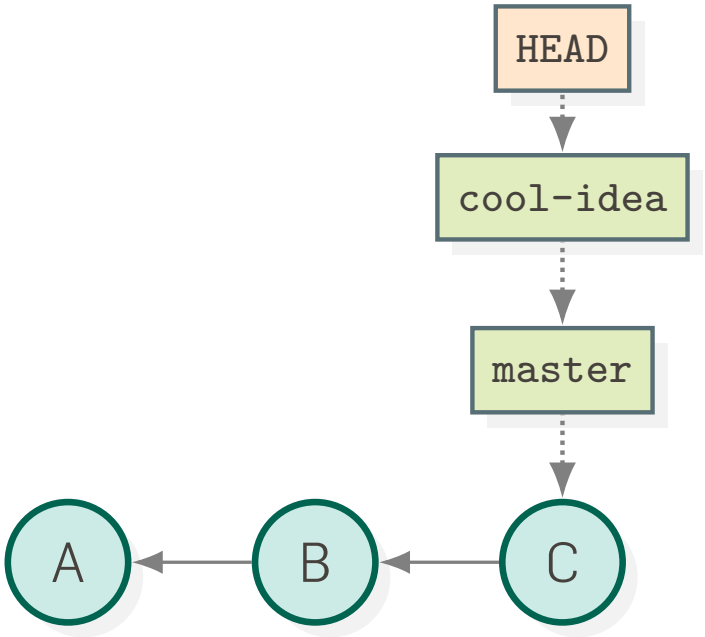
WORKING WITH BRANCHES

BRANCHES



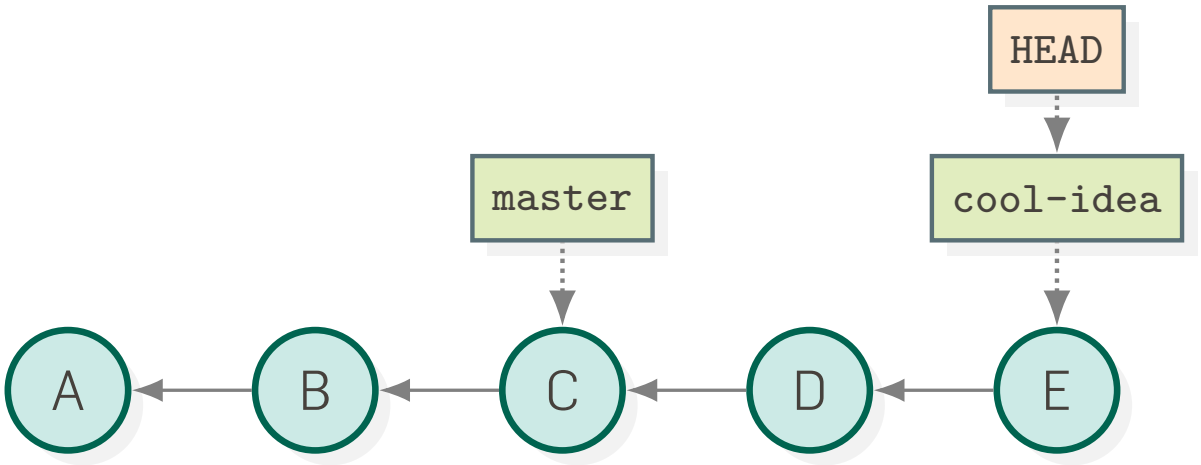
What if...?

BRANCHES

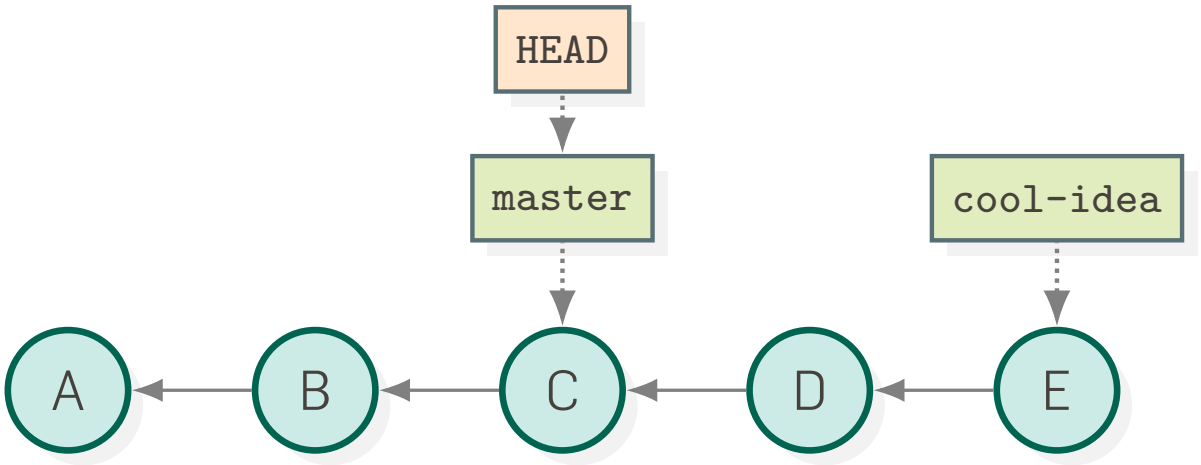



```
git checkout -b cool-idea
```

BRANCHES



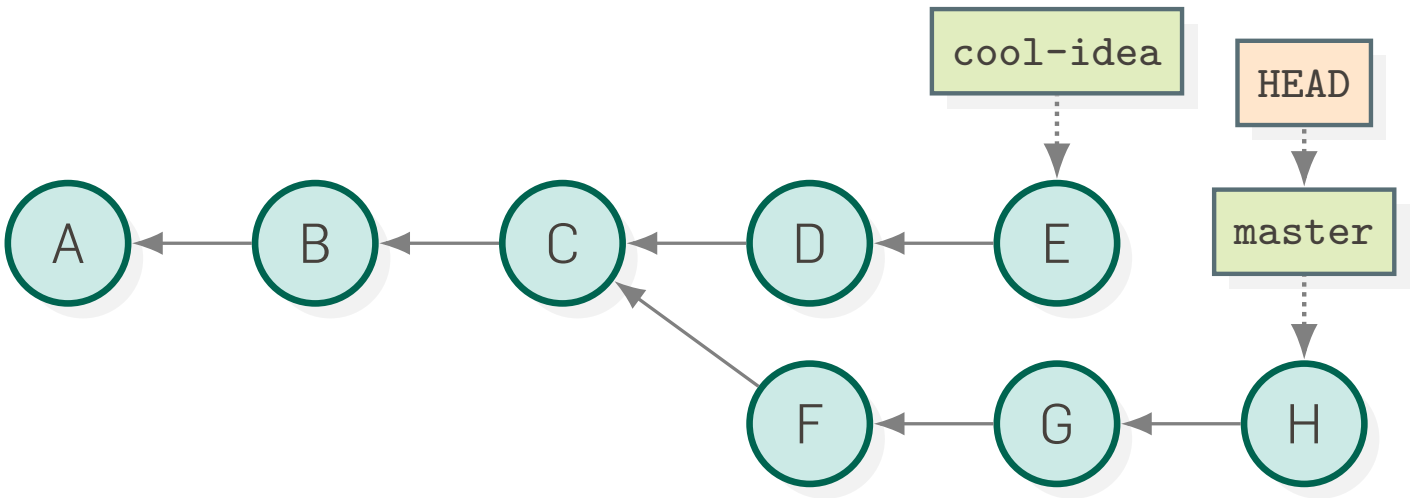
BRANCHES



Let go back to serious stuff!

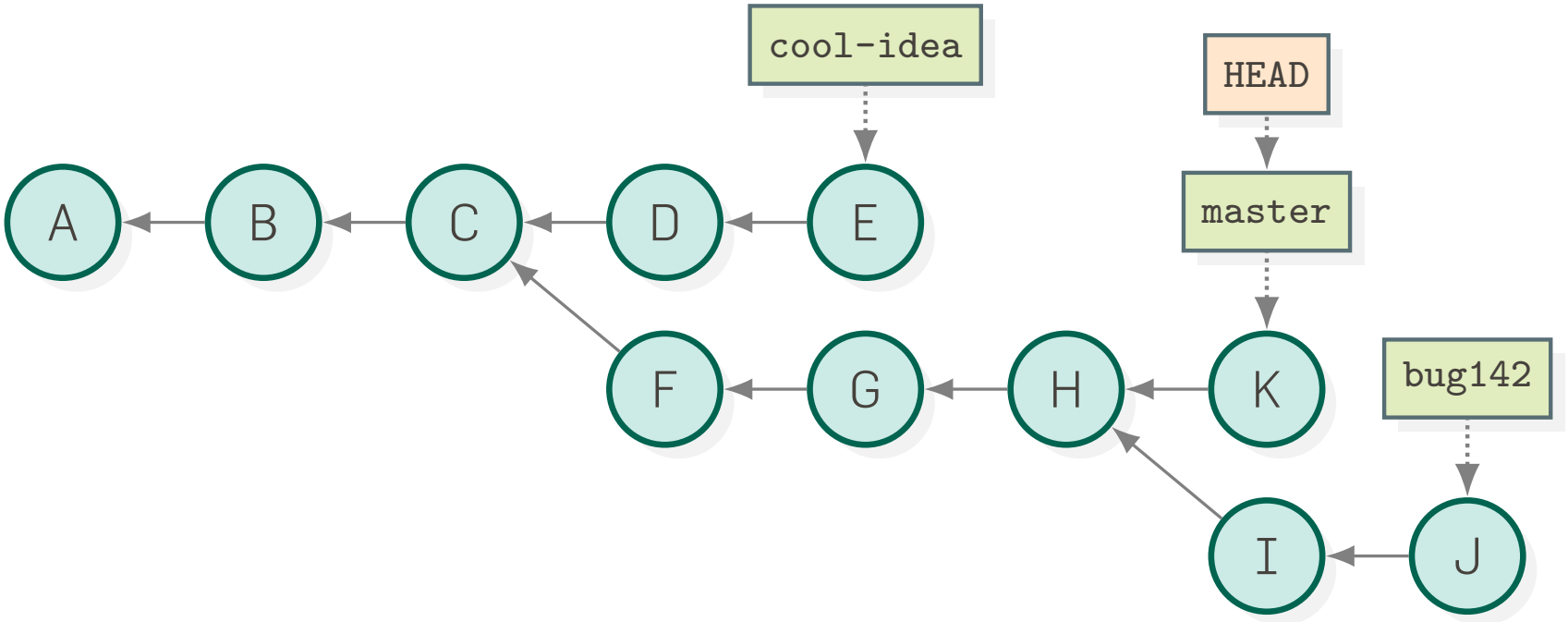
`git checkout master`

BRANCHES



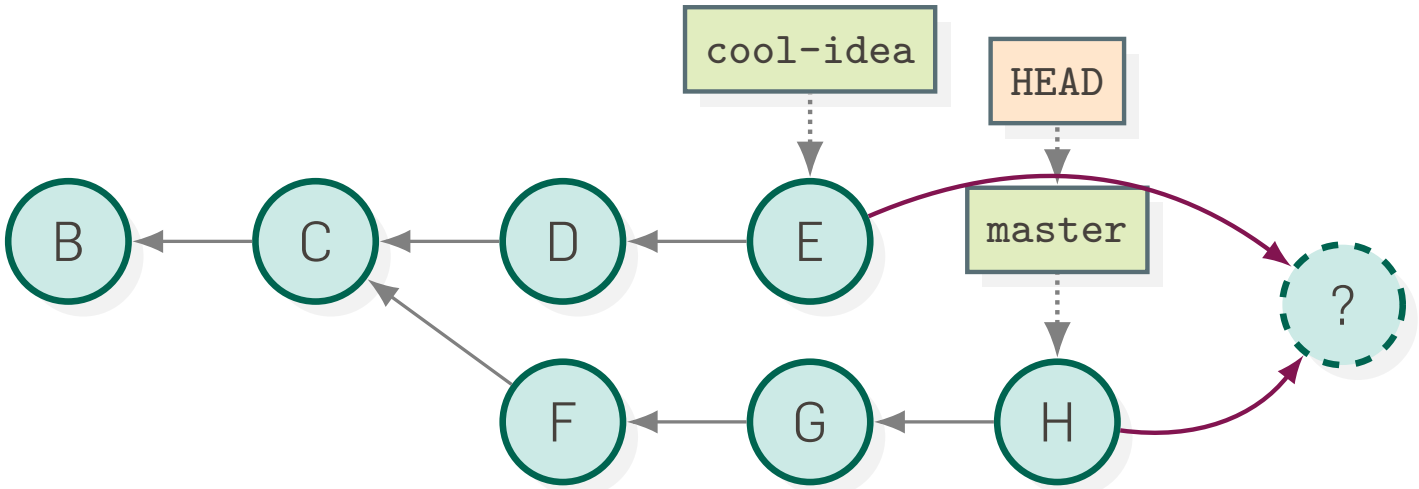
The branch name is an alias for the tip of the current branch

BRANCHES



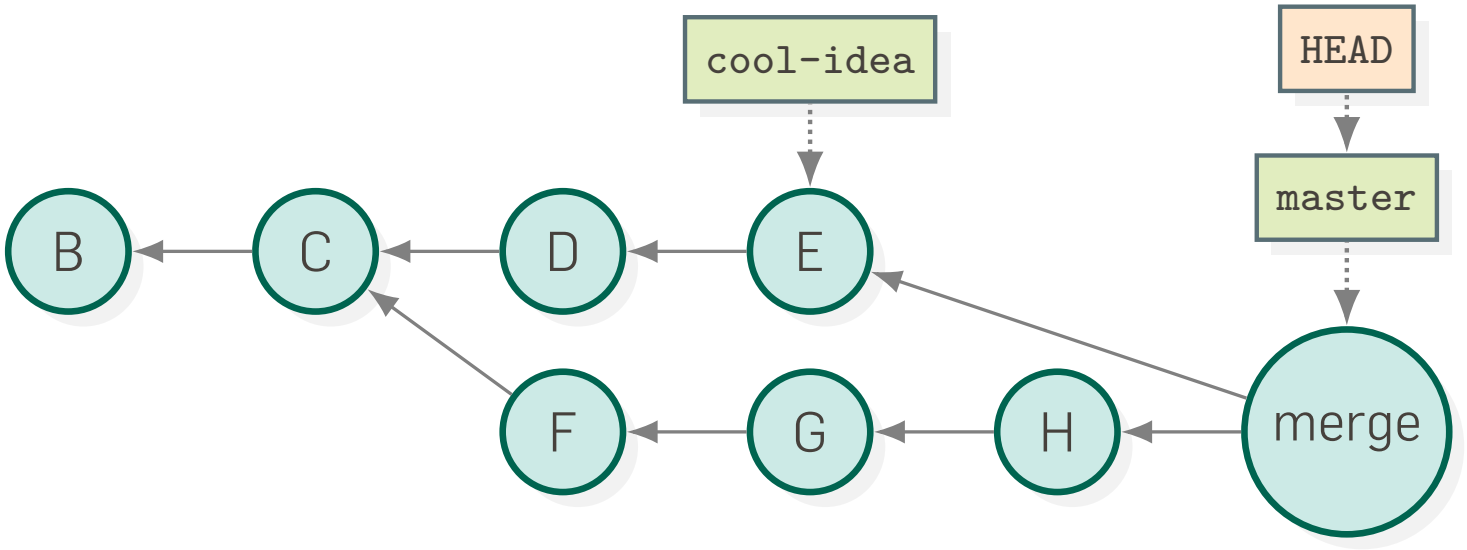
⇒ branches are very cheap
+10 of them at a given time it not uncommon

MERGING BRANCHES



Two options: **merging** and **rebasing**

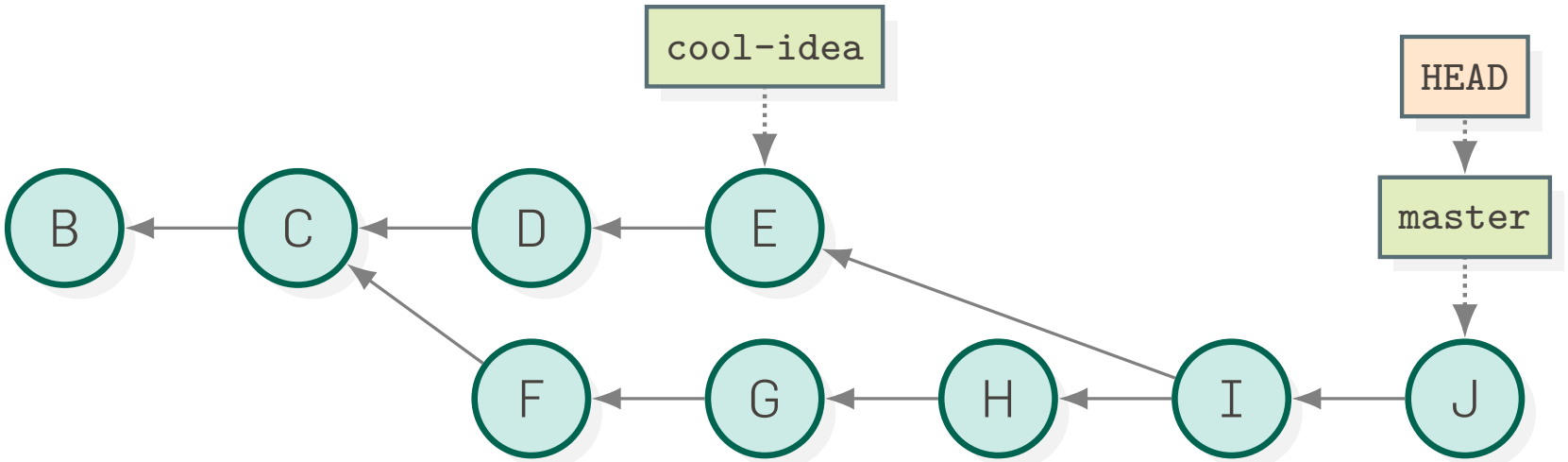
MERGING BRANCHES



Merging

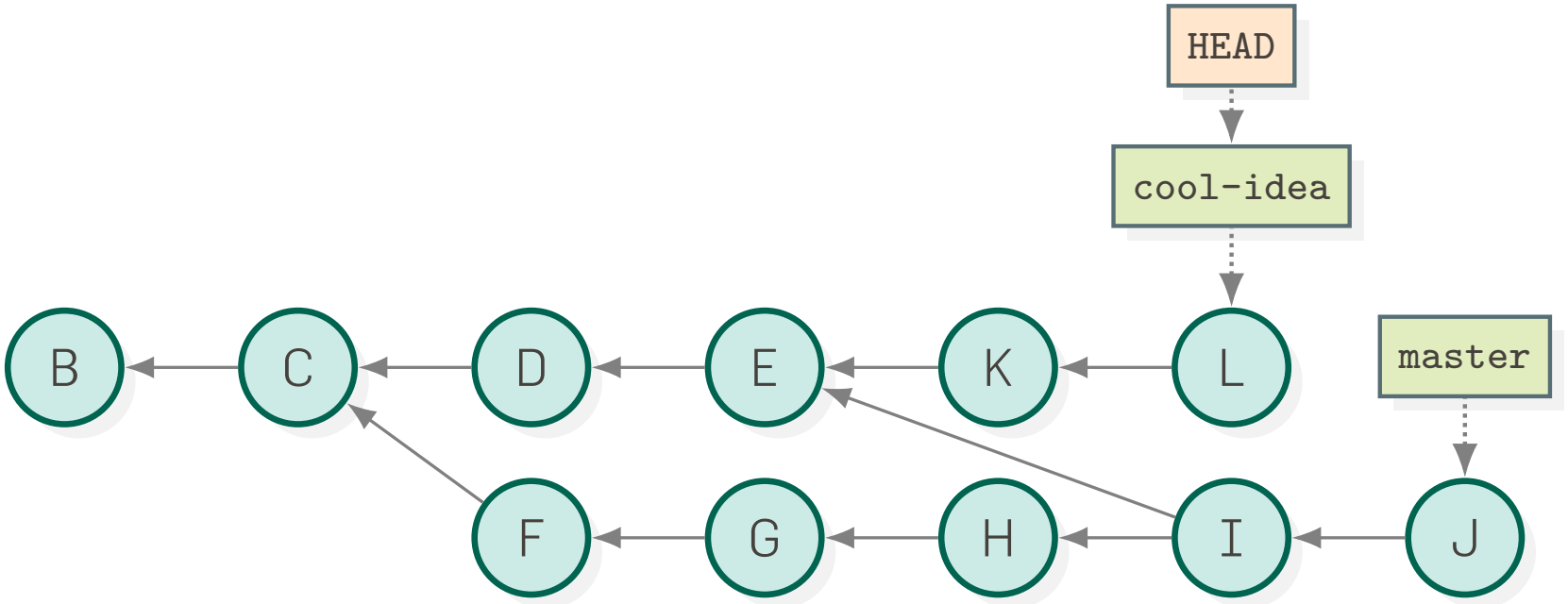
```
git merge cool-idea
```

MERGING BRANCHES




```
git commit
```

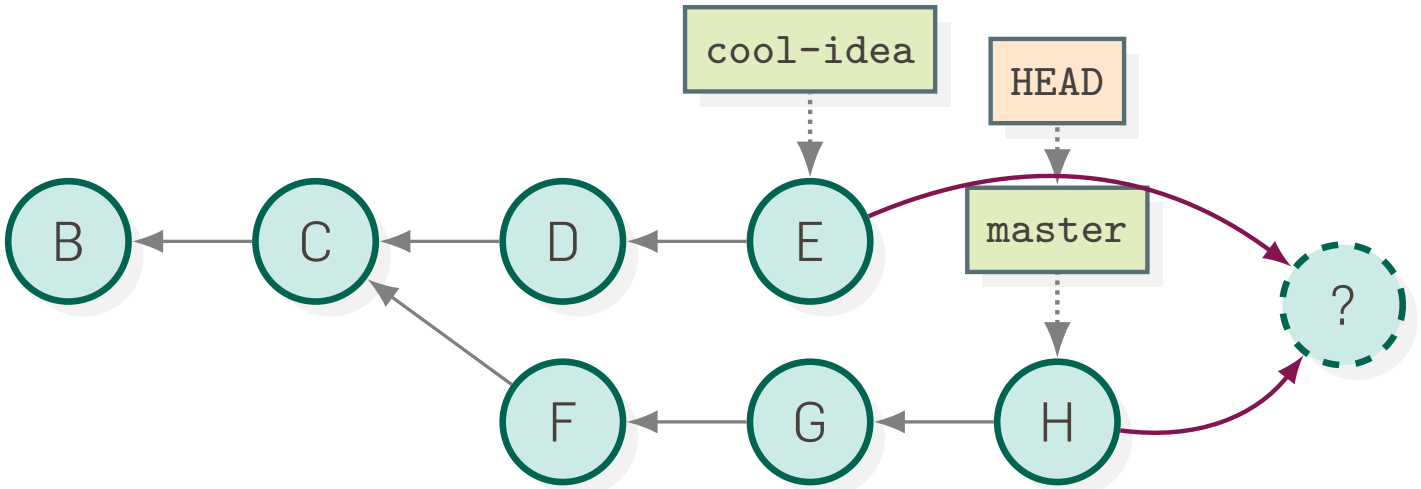
MERGING BRANCHES



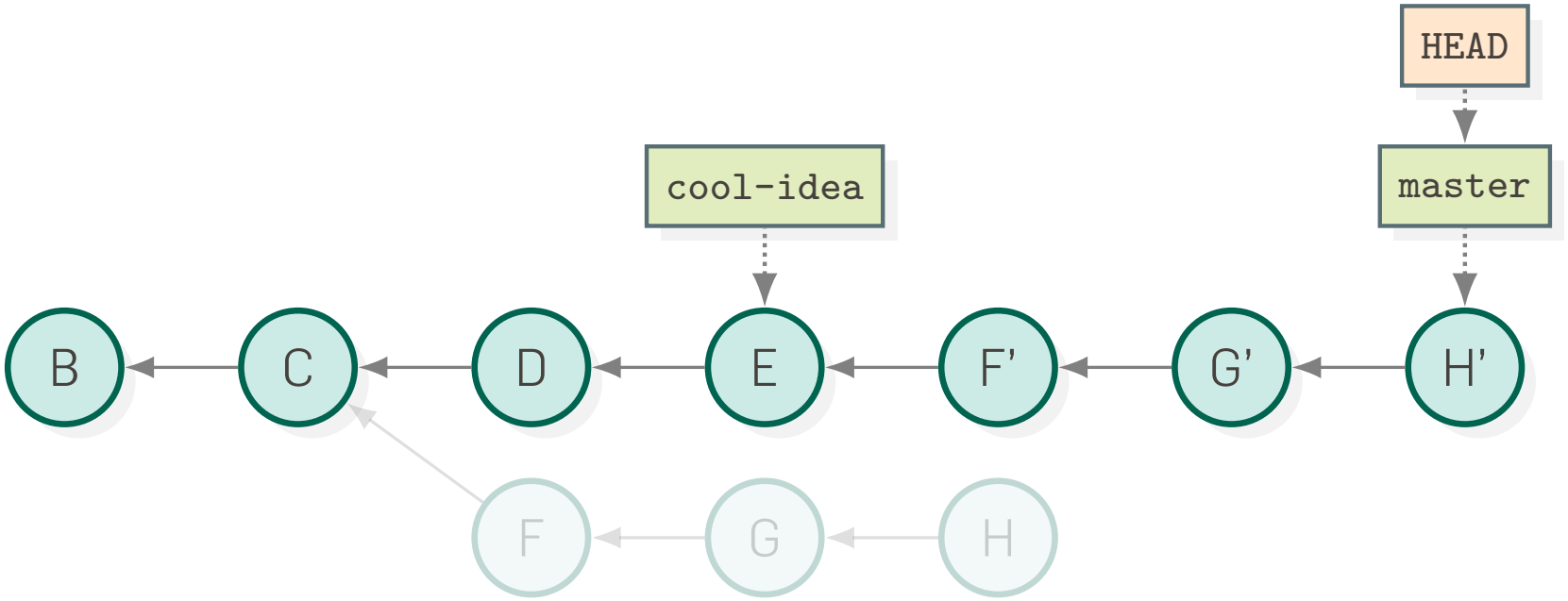

```
git checkout cool-idea
git commit
```

...etc.

REBASING BRANCHES



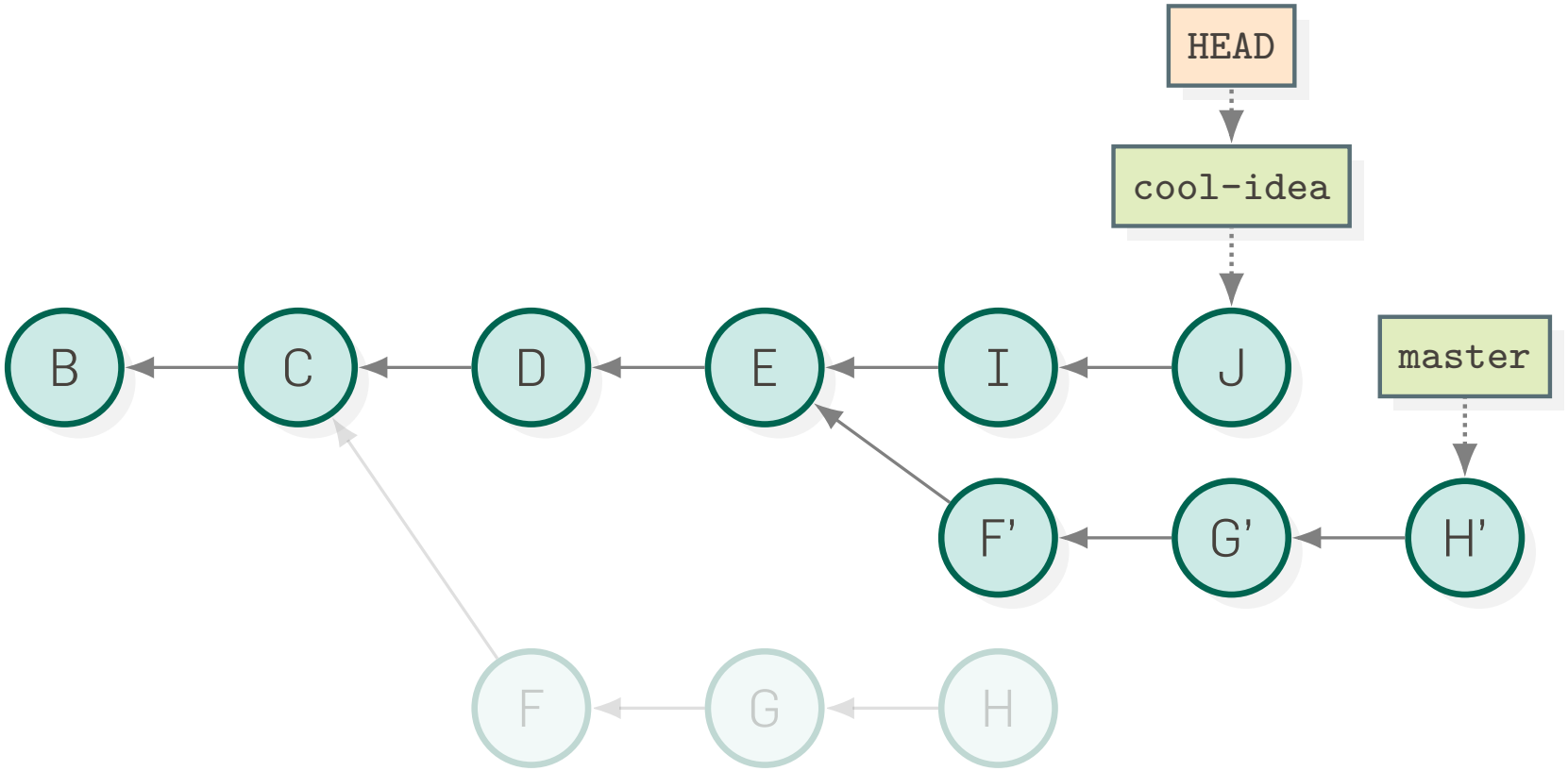
REBASING BRANCHES



Rebasing

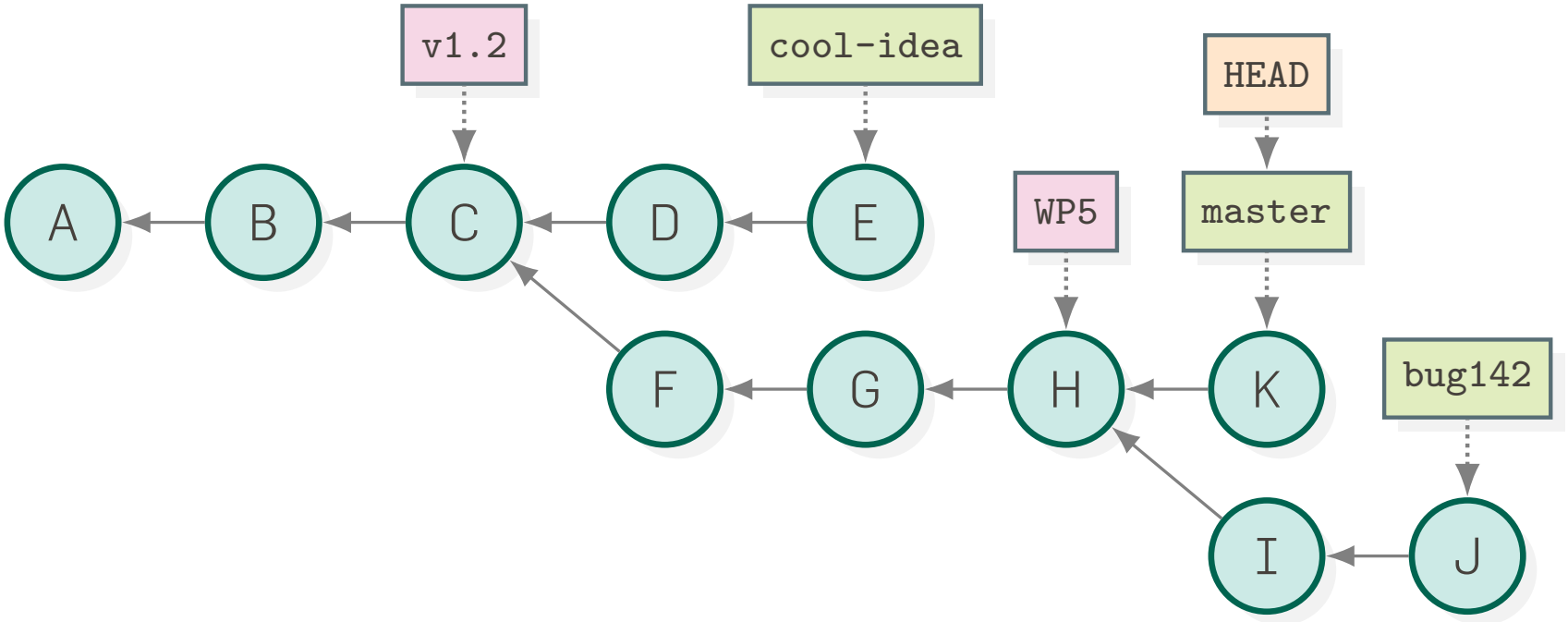
```
git rebase cool-idea
```

REBASING BRANCHES




```
git checkout cool-idea
git commit
```

MORE COMMIT ALIASES: TAGS



Label important commits/milestones

```
git tag v1.2
git tag WP5
```

TO SUMMARIZE...

```
# where are we?
$ git branch
master
# make some changes...
$ git add <files> && git commit -m"<commit message>"
# start working on something new?
$ git checkout -b new-idea
$ git branch
new-idea
# work in that branch for a while
$ git add <files> && git commit -m"<commit message>"
# back to master
$ git checkout master
#...
# rebase master on new-idea: new-idea is now in master
$ git rebase new-idea
```

THE ONE SLIDE TO REMEMBER

GIT CHEAT SHEET

To start...
...from scratch: `git init`
...from existing repo: `git clone <url>`

Prepare commits:
`git add`
`git rm`
`git add -p` (partial files)

Commit:
`git commit`

Create branch:
`git checkout -b <branch>`

Jump between branches:
`git checkout <branch>`

"Import" another branch:
`git rebase <other_branch>`

Add a remote source:
`git remote add <name> <url>`

What's new on a remote?
`git pull <remote> <branch>`
(`git pull` alone ≡ `git pull origin master`)

Share stuff on a remote:
`git push <remote> <branch>`
(`git push` alone ≡ `git push origin master`)

Repo state
`git status`

Repo history
`git log`

Who did what?
`git blame`

I've lost everythg!
`git reflog`

A FEW COOL GITHUB STUFF TO FINISH

Besides bugtracking, project homepages and wikis,GitHub integrates with many third-party services & tools:

- **Travis CI** or **AppVeyor** for continuous integration

[sensors] Added an 'encoders' level to the velocity sensor #541

Edit

Open **severin-lemaignan** wants to merge 1 commit into `morse-simulator:master` from `severin-lemaignan:encoders`

Conversation 4 | Commits 1 | Files changed 3 | +134 −3

severin-lemaignan commented on 29 May 2014 — The Modular OpenRobots Simulation Engine member

This new abstraction level for the velocity sensor that returns encoder ticks instead of linear/angluar speeds

[...]

Add more commits by pushing to the **encoders** branch on **severin-lemaignan/morse**.

All checks have failed
1 errored check

Hide all checks

✕ **continuous-integration/travis-ci** — The Travis CI build could not complete du… Details

This branch has conflicts that must be resolved
Use the command line to resolve conflicts before continuing.

Merge pull request or view command line instructions.

Labels

None yet

Milestone

Notifications

Unsubscribe

You're receiving notifications because you authored the thread.

2 participants

Lock conversation

A FEW COOL STUFF TO FINISH

+ GitHub integrates with many external services & tools:
    - **Travis CI** or **AppVeyor** for continuous integration
    - **zenodo**: associate a DOI to your repository
    - **ReadTheDocs**: generate and publish on-line documentation

That's all for today, folks!

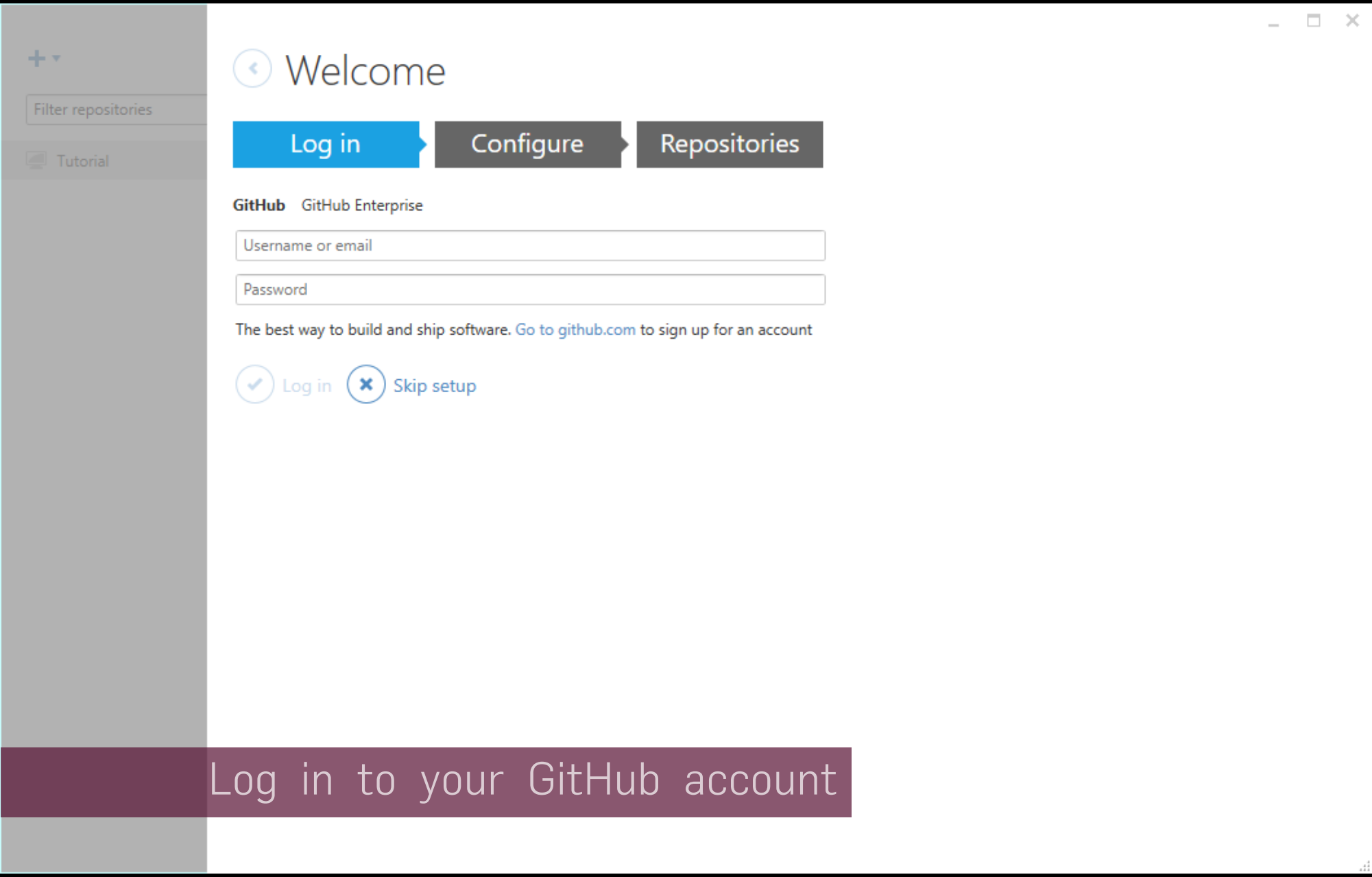
Slides:
github.com/severin-lemaignan/lecture-software-engineering

ADDITIONAL MATERIAL

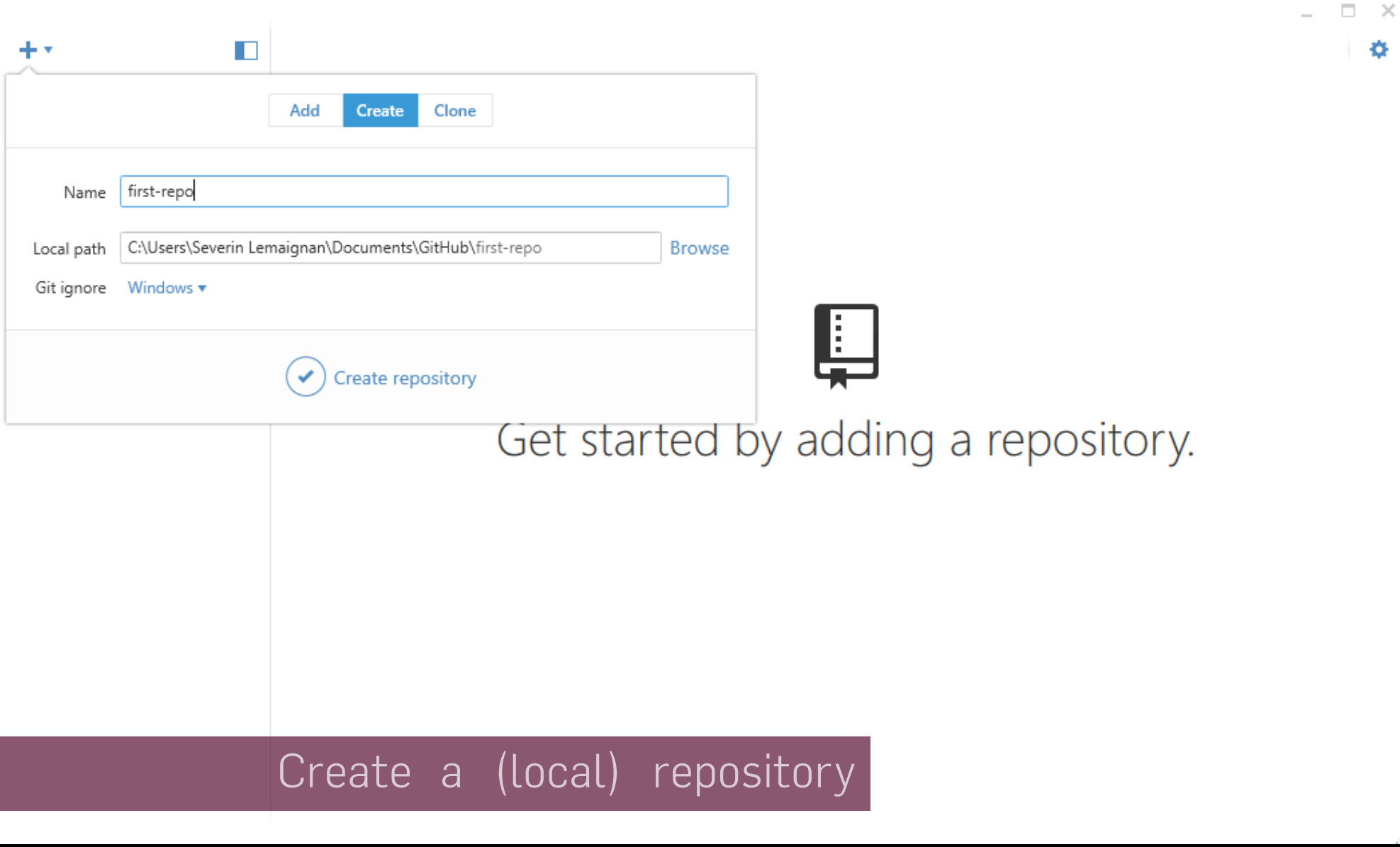
- git from a GUI

Viewed from a GUI (macOS & Windows)
GitHub Desktop Walkthrough

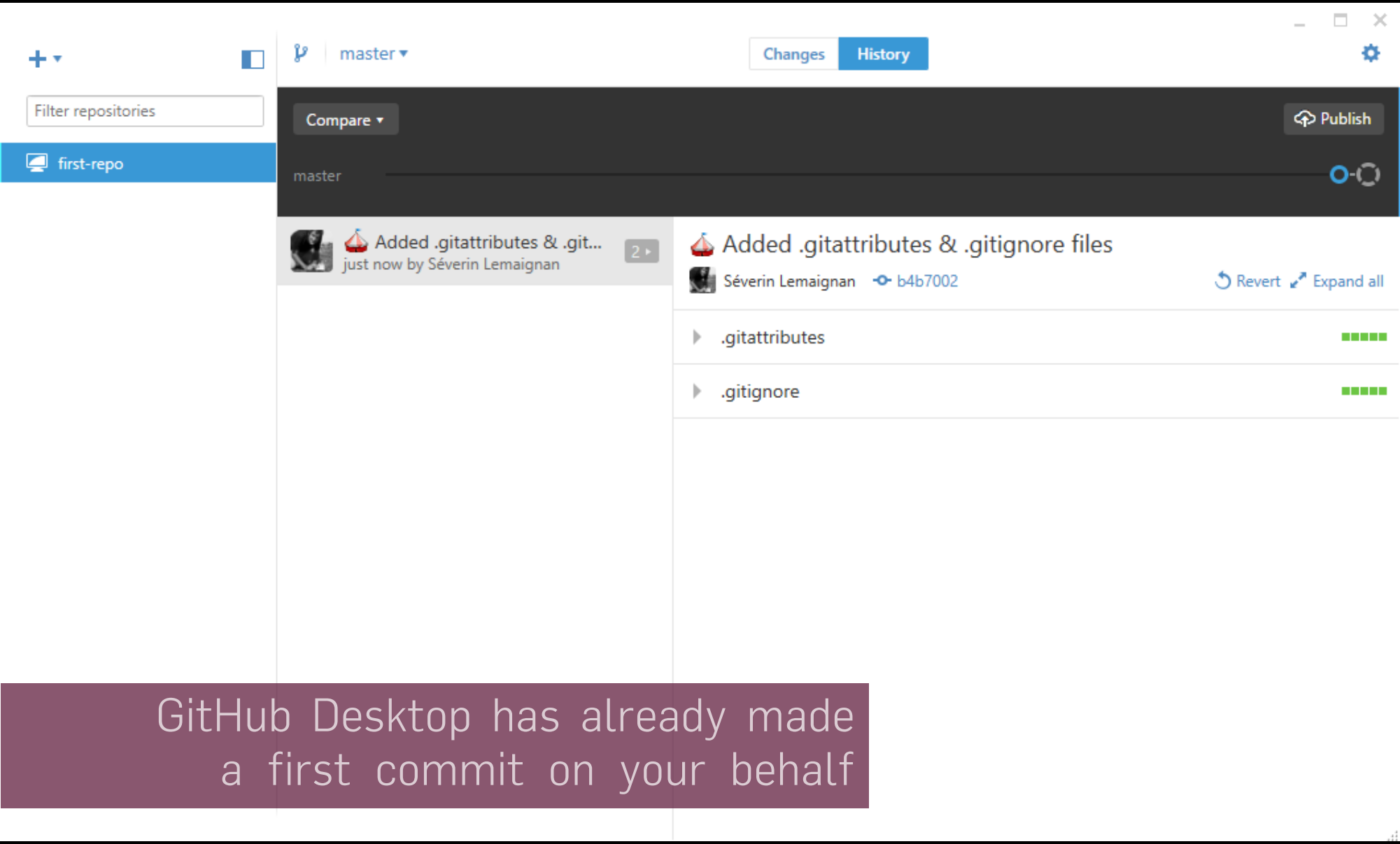
https://desktop.github.com/



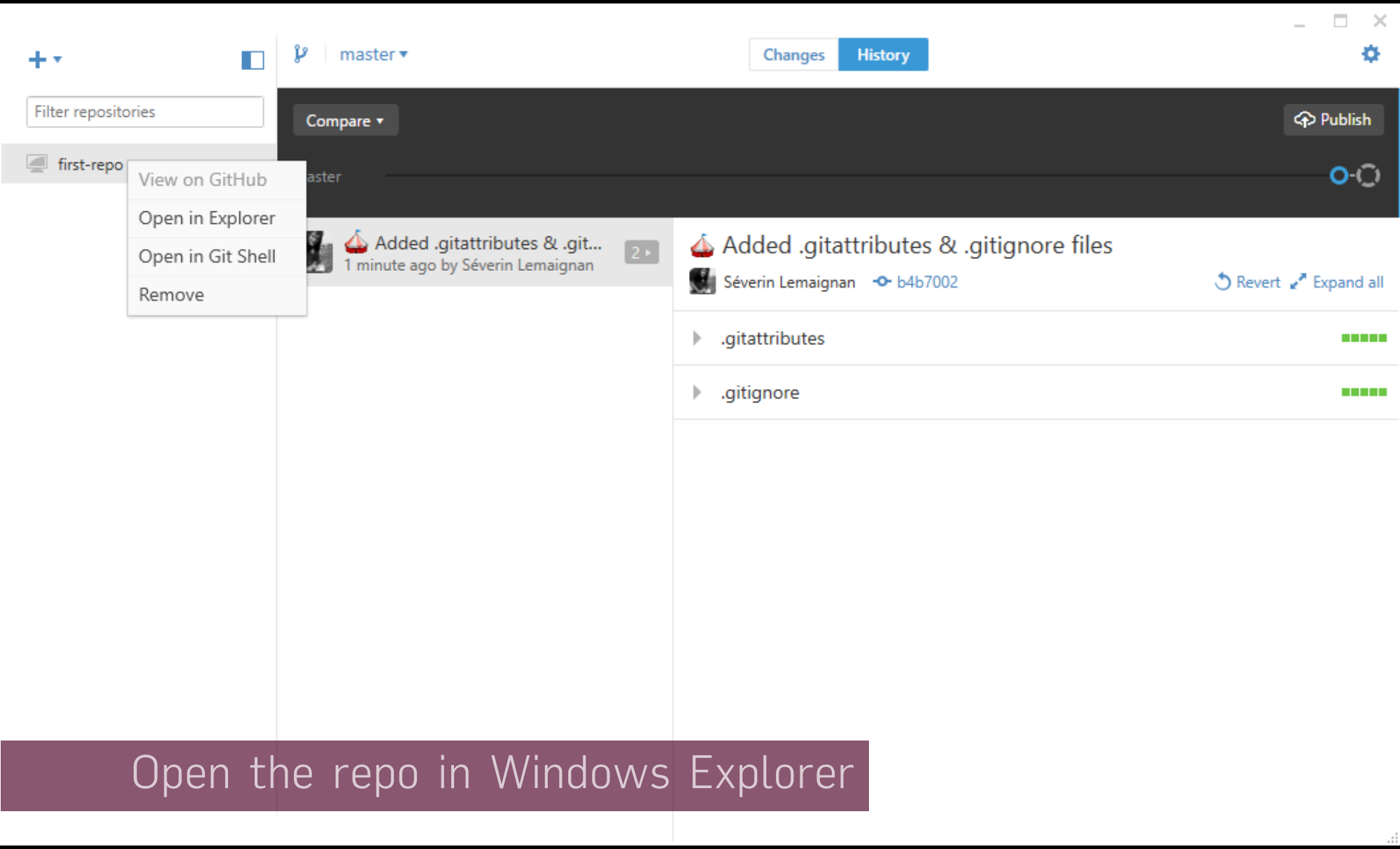
Log in to your GitHub account



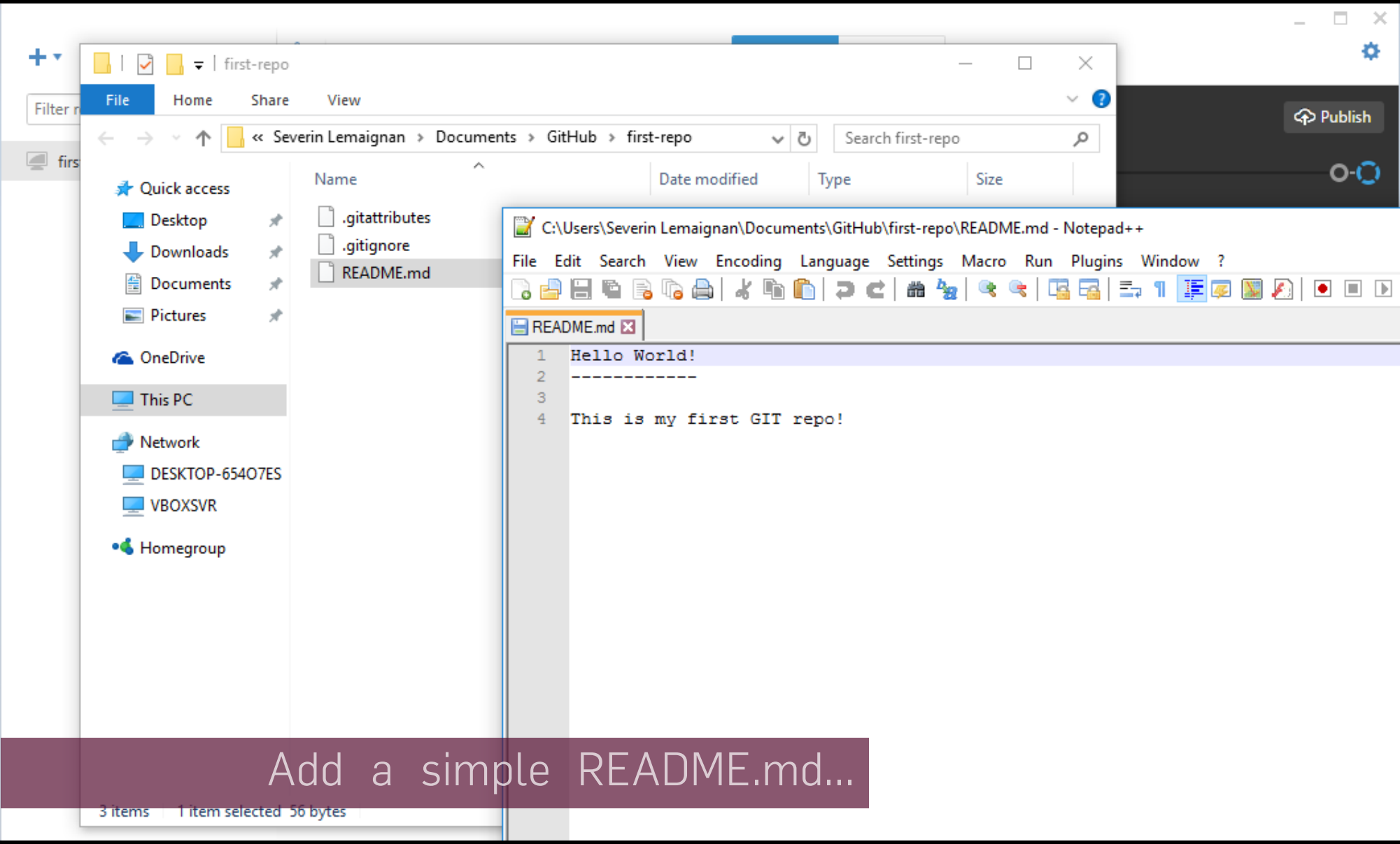
Create a (local) repository



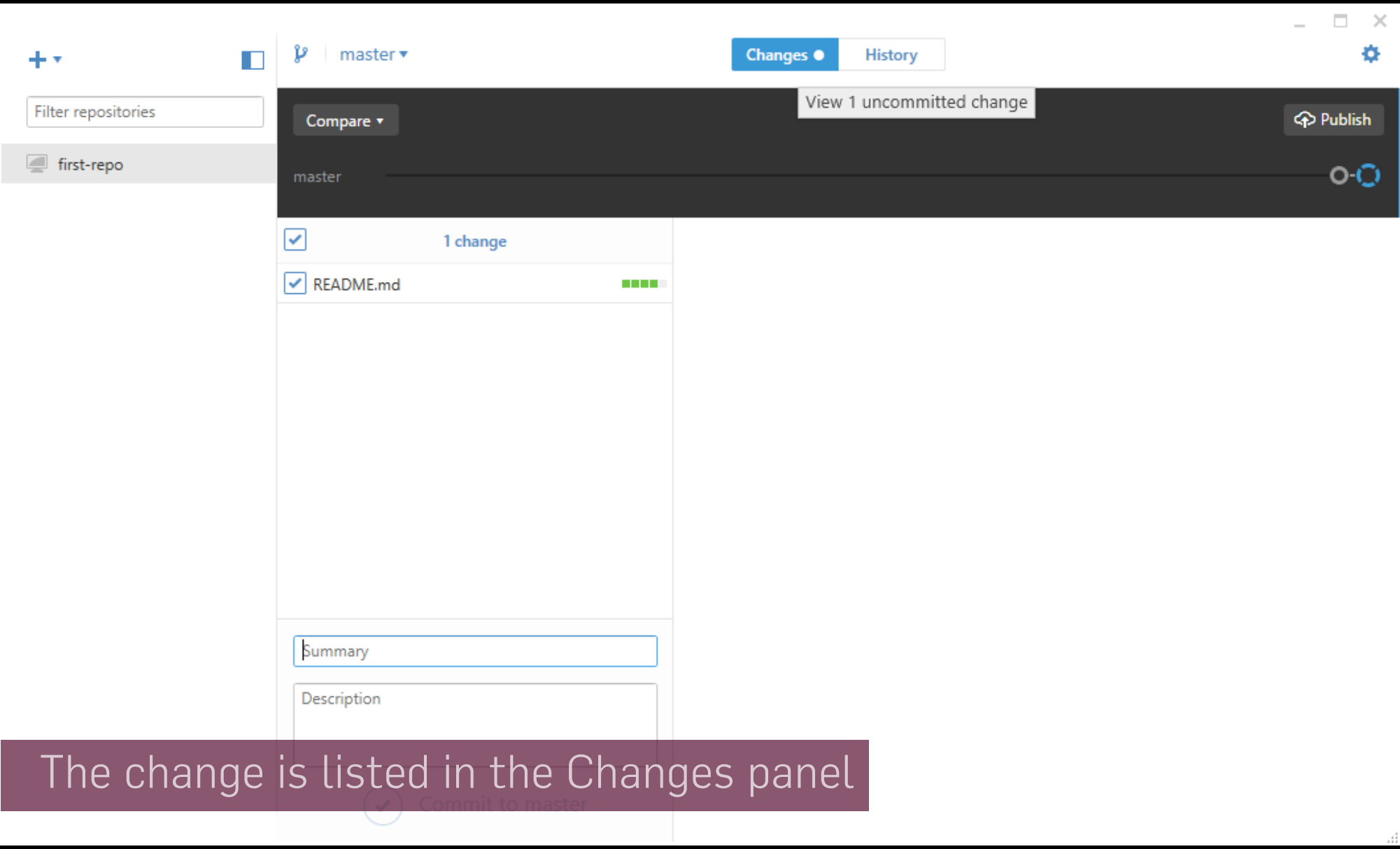
GitHub Desktop has already made a first commit on your behalf



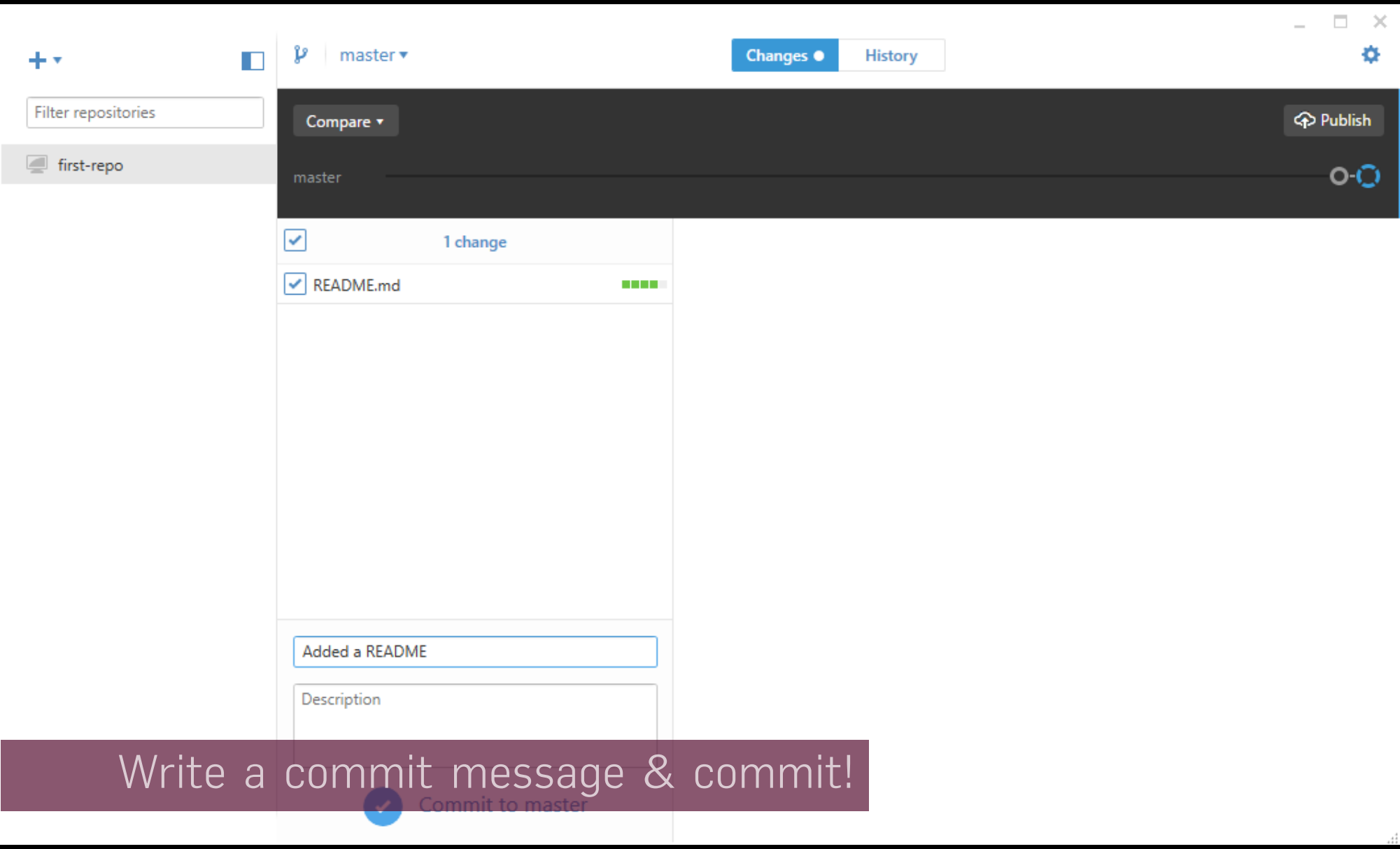
Open the repo in Windows Explorer



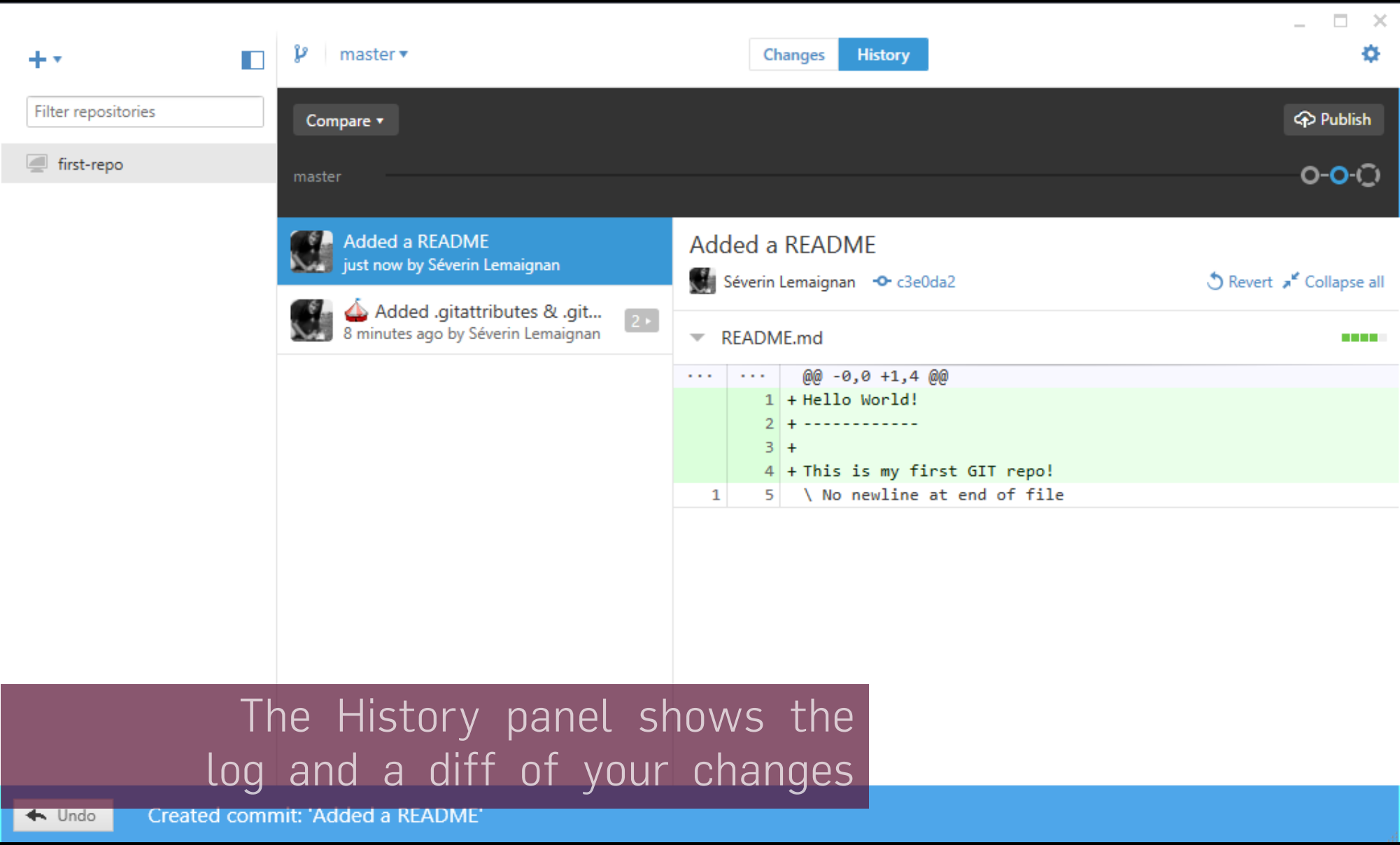
Add a simple README.md...



The change is listed in the Changes panel



Write a commit message & commit!



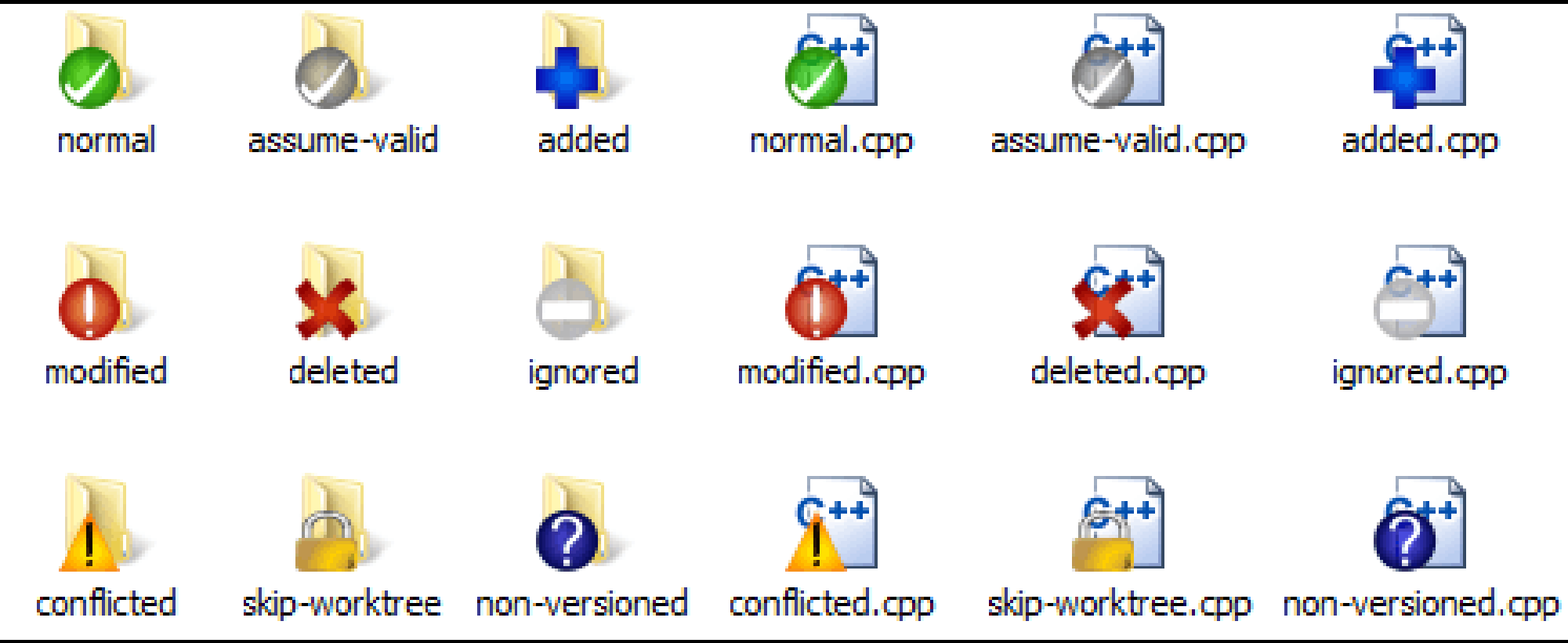
The History panel shows the log and a diff of your changes

Viewed from a GUI

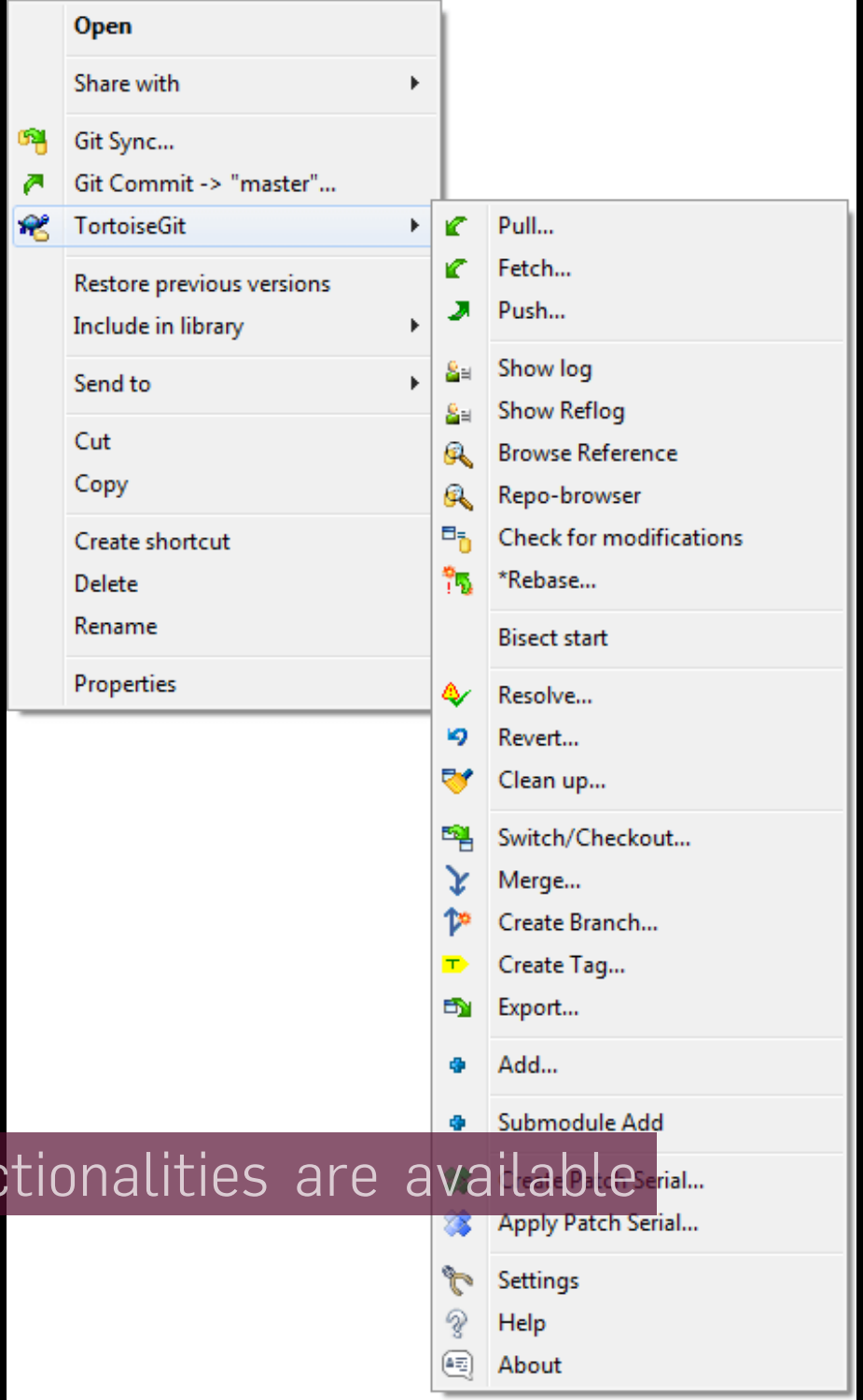
Tortoise GIT

https://tortoisegit.org/

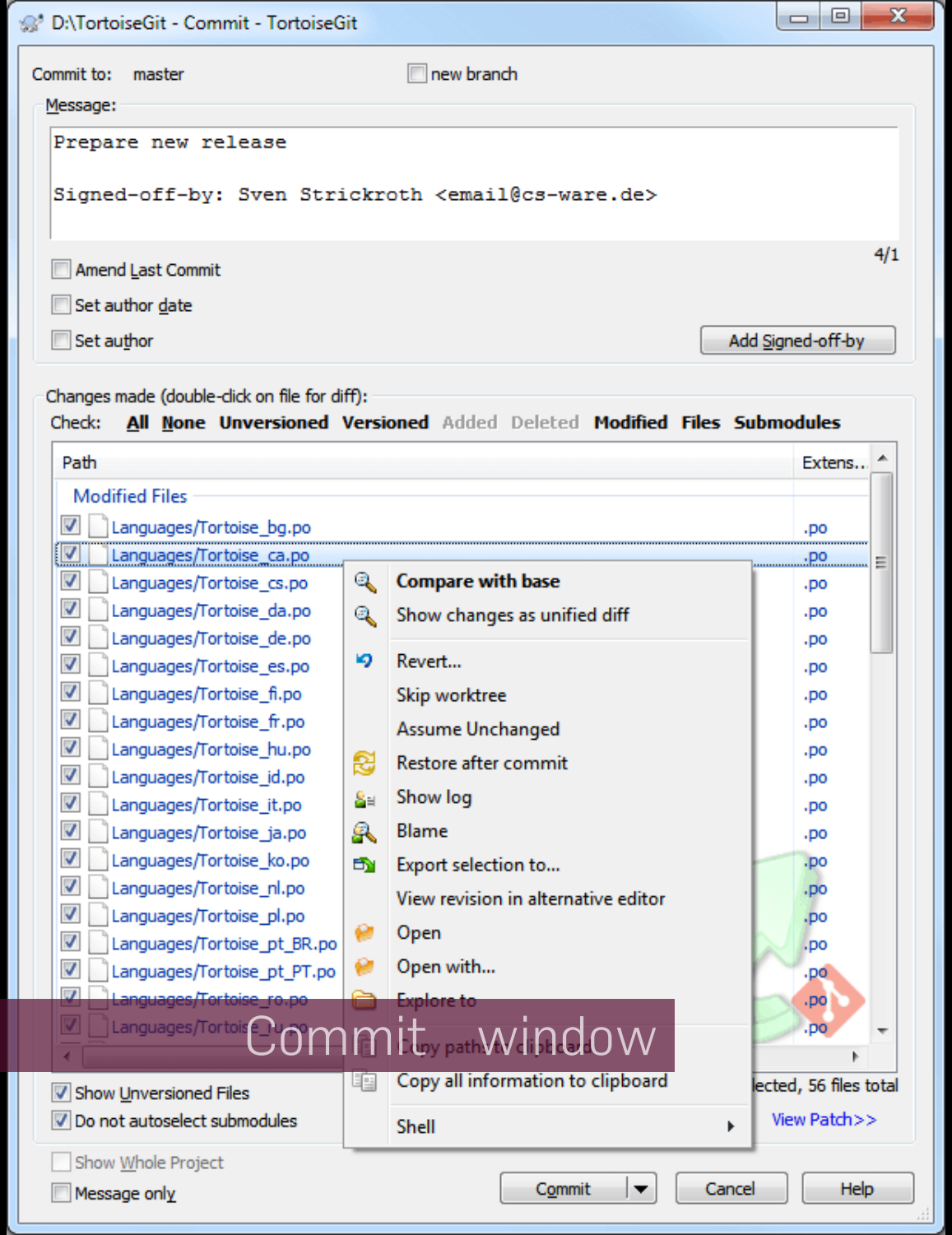
Direct interaction in the Windows explorer



Files' status appear as icons



All the functionalities are available



Commit window

Git branches viewed from a GUI...

Filter repositories

first-repo

master

Changes

History

Publish

Name numerical_coordinates

From branch master

Create new branch

Testing the rendering of a grid
4 hours ago by Séverin Lemaignan

Added a basic main()
5 hours ago by Séverin Lemaignan

Initial commit -- just a README
5 hours ago by Séverin Lemaignan

oved grid rendering with coordinates

verin Lemaignan da25158

Revert Collapse all

main.cpp

```
@@ -16,7 +16,9 @@ int main(int argc, char** argv) {
 16  16      while(!done) {
 17  17
 18  18          char i = 0;
 19     -        cout << "-------" << endl;
     19 +        cout << "  A B C " << endl;
     20 +        cout << " -------" << endl;
     21 +        cout << "1";
 20  22
 21  23          for(auto pos : positions) {
 22  24              i++;
@@ -24,7 +26,8 @@ int main(int argc, char** argv) {
 24  26
 25  27              if (i % 3 == 0) {
 26  28                  cout << "|" << endl;
 27     -                cout << "-------" << endl;
     29 +                cout << " -------" << endl;
     30 +                cout << i/3 + 1;
 28  31              }
 29  32          }
```

We can easily create a new branch

numerical_coordinates ▾ Changes ● History

Filter repositories

first-repo

Update from master | View branch | Publish

master ▾

numerical_coordinates

Switch to numerical coordinates
just now by Séverin Lemaignan

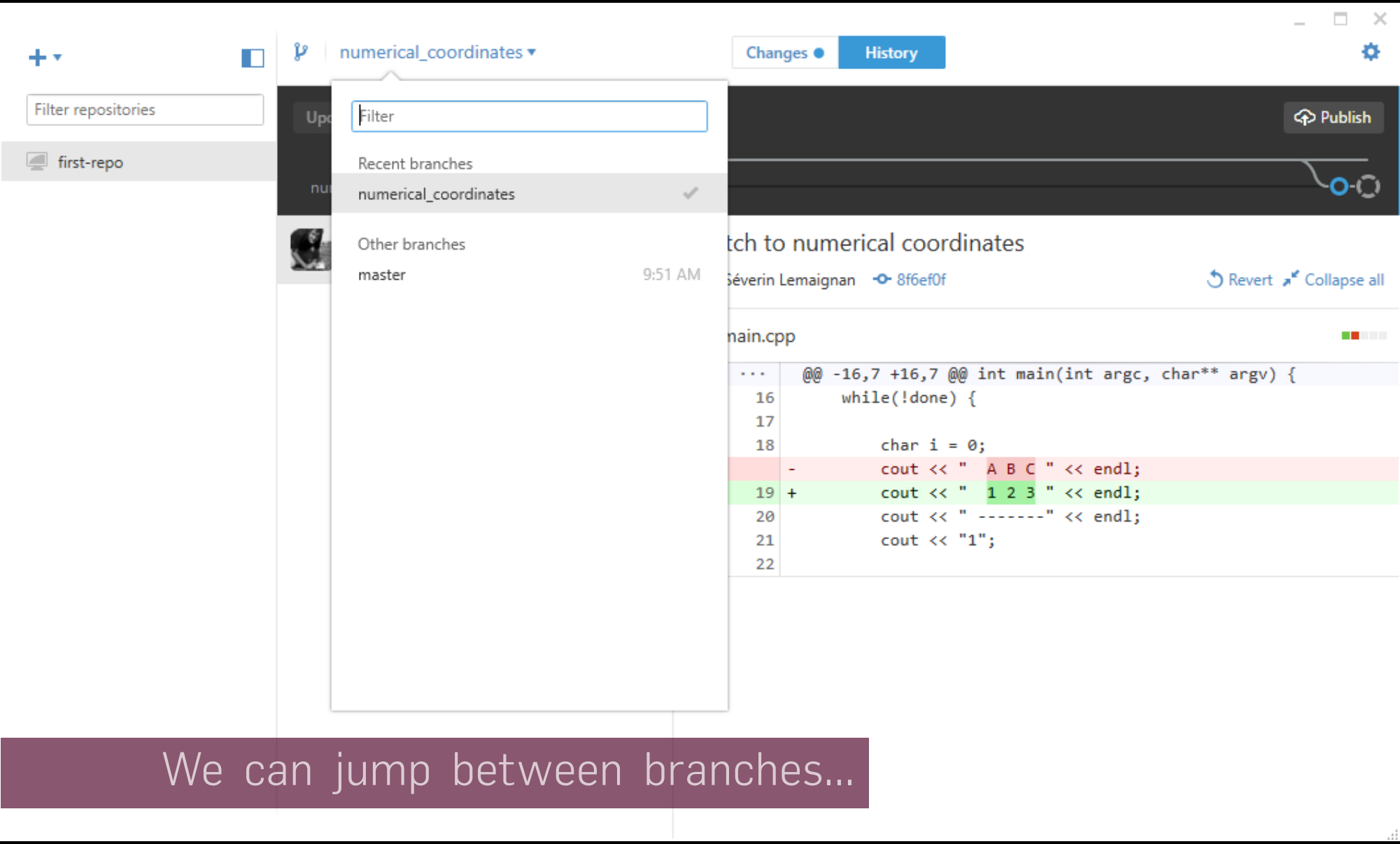
Switch to numerical coordinates

Séverin Lemaignan 8f6ef0f Revert Collapse all

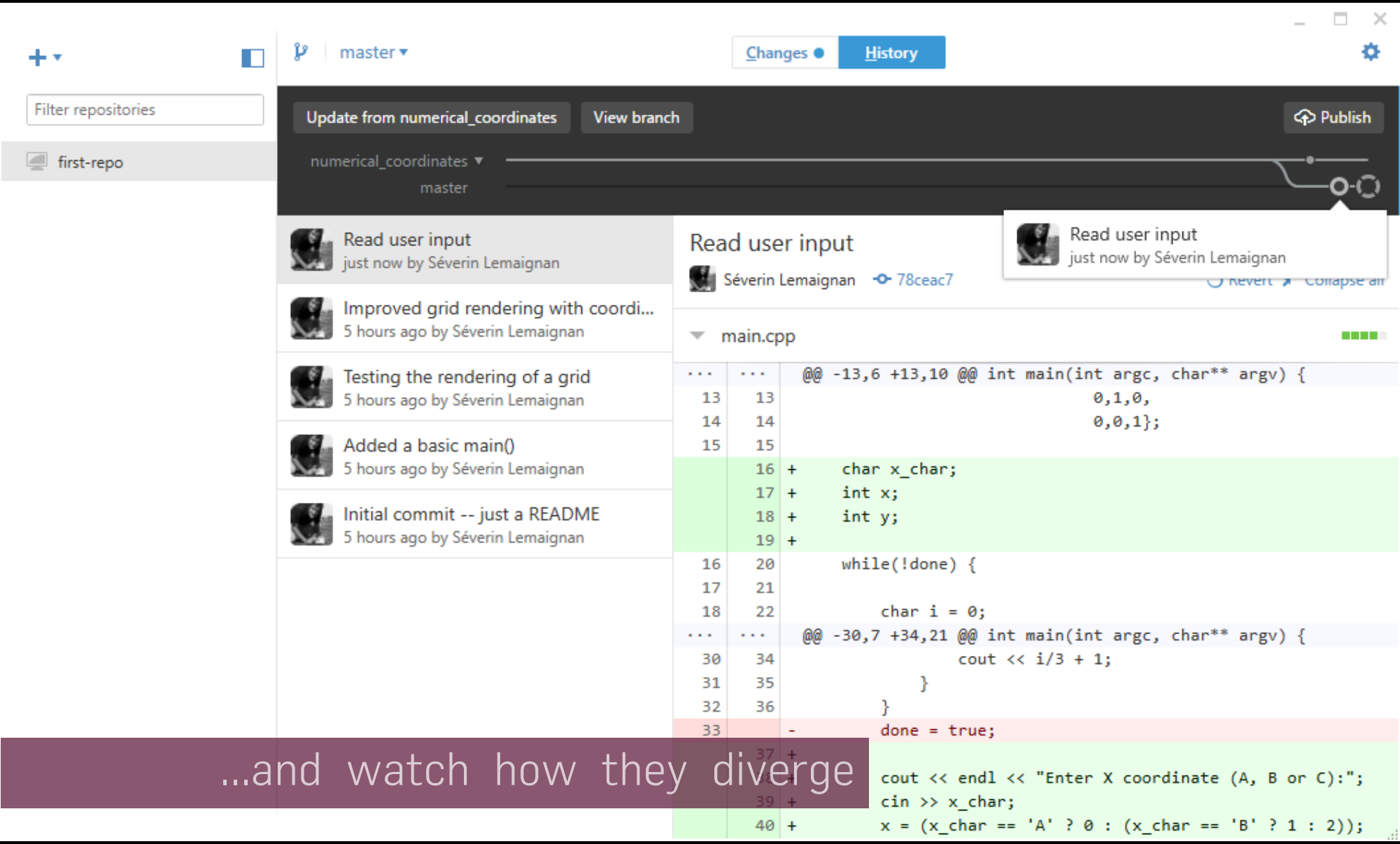
main.cpp

```
...  ...  @@ -16,7 +16,7 @@ int main(int argc, char** argv) {
16   16       while(!done) {
17   17
18   18           char i = 0;
19   -           cout << "  A B C " << endl;
     19 +        cout << "  1 2 3 " << endl;
20   20           cout << " -------" << endl;
21   21           cout << "1";
22   22
```

We can compare numerical_coordinates with master (click on View branch for the full history)



We can jump between branches...



...and watch how they diverge

numerical_coordinates

Changes | History

Filter repositories

first-repo

Update from master | View branch | Publish

Merge 1 commit from master into numerical_coordinates

numerical_coordinates

Switch to numerical coordinates
35 minutes ago by Séverin Lemaignan

Switch to numerical coordinates

Séverin Lemaignan 8f6ef0f

Revert Collapse all

main.cpp

```
@@ -16,7 +16,7 @@ int main(int argc, char** argv) {
 16  16      while(!done) {
 17  17
 18  18          char i = 0;
 19     -        cout << "  A B C " << endl;
    19  +        cout << "  1 2 3 " << endl;
 20  20          cout << " -------" << endl;
 21  21          cout << "1";
 22  22
```

We switch back to numerical_co-ordinates and merge in master

numerical_coordinates ▾ | Changes ● | History

Filter repositories

first-repo

Compare ▾ | Publish

numerical_coordinates

Merge branch 'refs/heads/master' int...
just now by Séverin Lemaignan

Switch to numerical coordinates
36 minutes ago by Séverin Lemaignan

Improved grid rendering with coordi...
5 hours ago by Séverin Lemaignan

Testing the rendering of a grid
5 hours ago by Séverin Lemaignan

Added a basic main()
5 hours ago by Séverin Lemaignan

Initial commit -- just a README
5 hours ago by Séverin Lemaignan

Merge branch 'refs/heads/master' i...
just now by Séverin Lemaignan

Merge branch 'refs/heads/ma

Séverin Lemaignan 64344d4 | Revert | Collapse all

main.cpp

```
@@ -13,6 +13,10 @@ int main(int argc, char** argv) {
                                0,1,0,
                                0,0,1};

+     char x_char;
+     int x;
+     int y;
+
      while(!done) {

          char i = 0;
@@ -30,7 +34,21 @@ int main(int argc, char** argv) {
                  cout << i/3 + 1;
              }
          }
          done = true;
+
+         cout << endl << "Enter X coordinate (A, B or C):";
+         cin >> x_char;
+         x = (x_char == 'A' ? 0 : (x_char == 'B' ? 1 : 2));
```

The merge commit is reflected in the history of the branch