

Version control with GIT

A workshop by [@goodbytes](#)

How do you share code
today? zip? Dropbox?
Facebook?

How do you deploy a
website today?

FileZilla - Connected to 127.0.0.1

File Edit Transfer View Queue Server Help

Address: localhost User: filezilla Password: ***** Port: 21 Quickconnect

Command: PASV
 Response: 227 Entering Passive Mode (127.0.0.1,8,225)
 Response: 227 Entering Passive Mode (127.0.0.1,8,224)
 Command: APPE AsyncSocketEx.cpp
 Response: 150 Connection accepted, restarting at offset 10752
 Command: APPE ControlSocket.cpp
 Response: 150 Connection accepted, restarting at offset 10752

Local Site: D:\cvsroot\FileZilla\source\

Remote Site: /source/

Filename	Filesize	Date	Time	Berechtigungen	Owner / Group
..					
h AsyncSocketExLayer.h	6789	29.06.2003	13:40	-rw-r--r--	ftp ftp
c AsyncSslSocketLayer.cpp	45468	29.06.2003	13:48	-rw-r--r--	ftp ftp
h AsyncSslSocketLayer.h	6939	29.06.2003	13:47	-rw-r--r--	ftp ftp
c CommandQueue.cpp	23041	29.06.2003	13:48	-rw-r--r--	ftp ftp
h CommandQueue.h	3005	29.06.2003	13:48	-rw-r--r--	ftp ftp
c ControlSocket.cpp	14775	29.06.2003	13:49	-rw-r--r--	ftp ftp
h Download	5694	29.06.2003	13:49	-rw-r--r--	ftp ftp
c Add to Queue	2068	29.06.2003	13:49	-rw-r--r--	ftp ftp
c Download as...					
Open					
View / Edit					
Create Directory					
Delete					
Rename					
File attributes...					

Selected 5 files with 113636 bytes.

Local Filename	Size	Direction	Remote Filename	Host	Status
D:\cvsroot\FileZilla\source\ControlSoc...	14775	-->	/ControlSocket.cpp	localhost:21	
0:00:05 elapsed	0:00:01 left	93%			13824 bytes (512 B/s)
D:\cvsroot\FileZilla\source\AsyncSock...	35940	-->	/AsyncSocketEx.cpp	localhost:21	
0:00:06 elapsed	0:00:26 left	42%			15360 bytes (716 B/s)
D:\cvsroot\FileZilla\source\FileZilla.dsp	21672	-->	/FileZilla.dsp	localhost:21	
D:\cvsroot\FileZilla\source\FileZilla.dsw	2793	-->	/FileZilla.dsw	localhost:21	

Ready Queue: 73 KB

Throw all files on ftp and
let's hope it works?



Let's GIT to work



Joeri Claes @joericlaes

35m

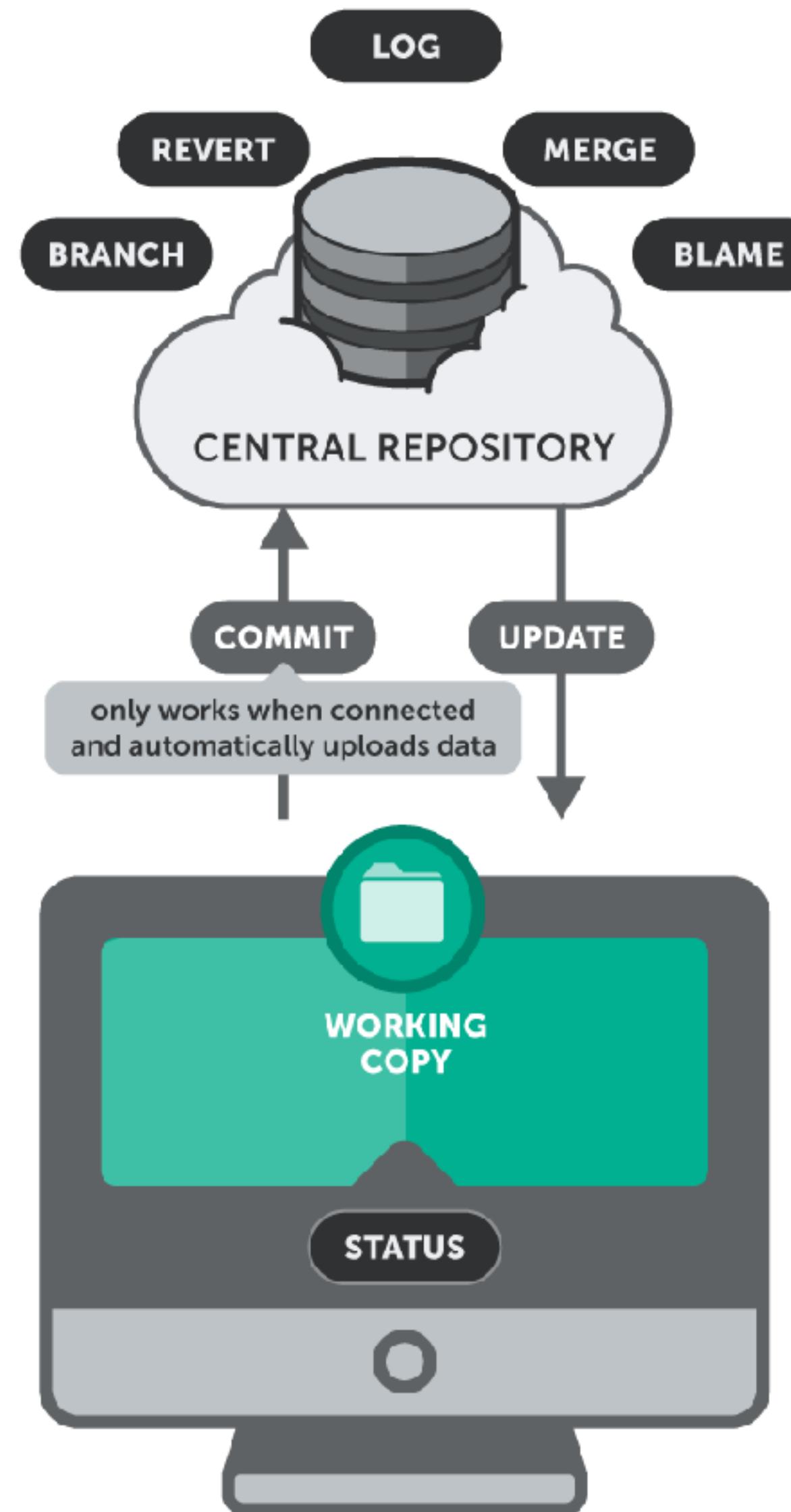
Dat git & command-line belangrijk zijn!
Zonder kennis van die twee kan je hier op
stage niets komen doen :)

...

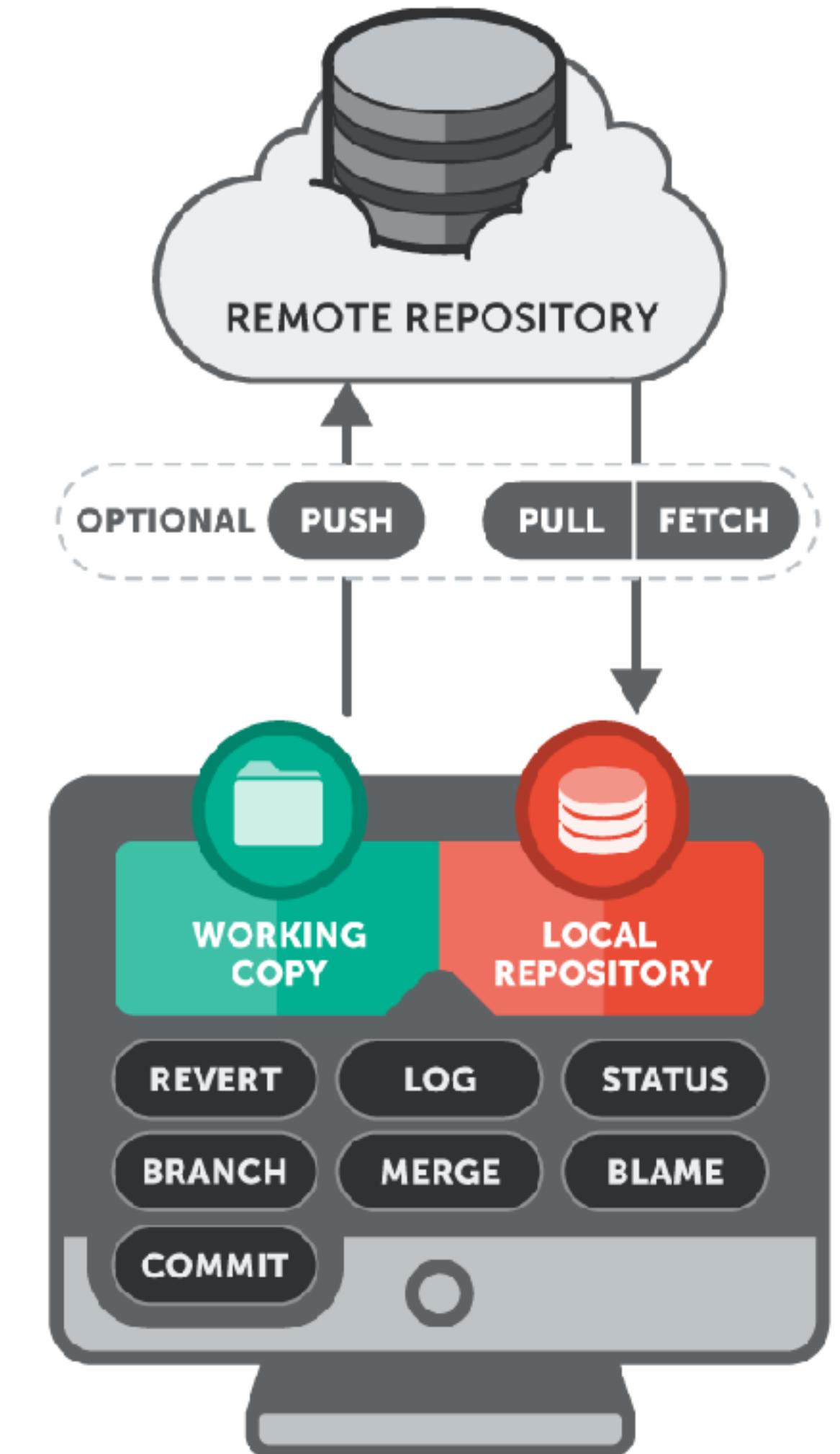
Why GIT

- fool-proof coding workflow
- collaboration
- reliable deployment
- backups
- decentralized (e.g. Subversion is centralized)

SUBVERSION



GIT



First, install git (CLI)

- instructions are here <http://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
- if you are on windows, this may be handy way to install git and a graphical client in one go:
<https://windows.github.com/>
- we'll start with the CLI or *command line interface*

core concepts

- clone
- add / commit
- push / pull
- branching / checkout
- merge / rebase



git



github
SOCIAL CODING

Atlassian
Bitbucket

clone

- copy a repository to your local development system
- usually done when setting up a project
- `git clone git@github.com:iamgoodbytes/2imd-webtech2-labs.git somedirectory`

add & commit files

command	what it does
git add index.html	add the file index.html to version control
git commit -m “commit message”	commit the changes
git status	show what files are ready to be committed or what changes are pending

commit

	Thomas De Bock	15acd28	Bugfix: getOverbooking function doesn't crash when not giving parameters
	Thomas De Bock	e657d50	Basic password mailing functionality
	Joris Hens	d143cfcc	Renamed setup fee to service retainer
	Joris Hens	01755e1 M	Merge branch 'master' of bitbucket.org:goodbytes/belcham-development-intranet
	Joris Hens	b094edc	added payment method to invoice
	Thomas De Bock	ad930b6 M	Merge branch 'master' of bitbucket.org:goodbytes/belcham-development-intranet
	Thomas De Bock	4ce9c94	Fixing billing over multiple months, with start and end date reductions working

bad commit messages

“fixed bug”

- tip: avoid git commit -m as it puts you in a poor mindset forcing you to write short and unclear commit messages

good commit messages

Redirect user to the requested page after login

<https://trello.com/path/to/relevant/card>

Users were being redirected to the home page after login, which is less useful than redirecting to the page they had originally requested before being redirected to the login form.

- * Store requested path in a session variable
- * Redirect to the stored location after successfully logging in the user

- <https://robots.thoughtbot.com/5-useful-tips-for-a-better-commit-message>

pushing & pulling

command	what it does
<code>git pull [alias] [branch]</code>	fetch+merge latest changes from a repository
<code>git push <i>origin</i> <i>master</i></code>	push your commits to the repository at alias <i>origin</i> and to the branch <i>master</i>

push

- when you have commits that are ready to be shared with your team or repository, you can push them
- push requires a remote and a branch selection
- if you just write **git push**, remote *origin* and branch *master* will be selected for you
- **git push origin master**

pull

- pull will grab the latest version of a branch in your repository (fetch) and merges these changes with your current branch
- git pull is short for **git pull origin master**

remotes

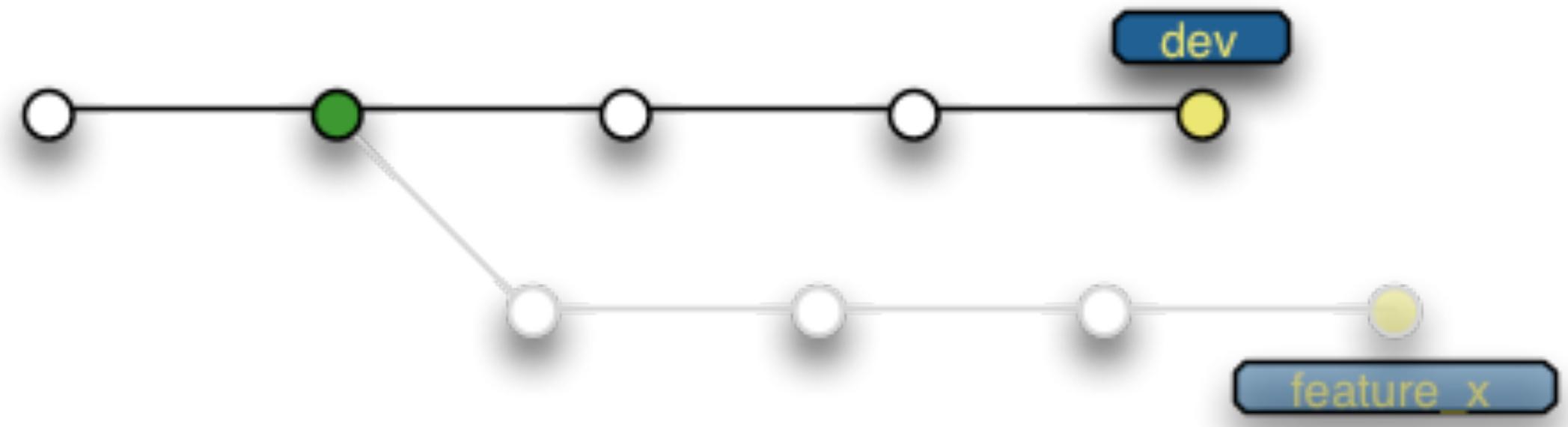
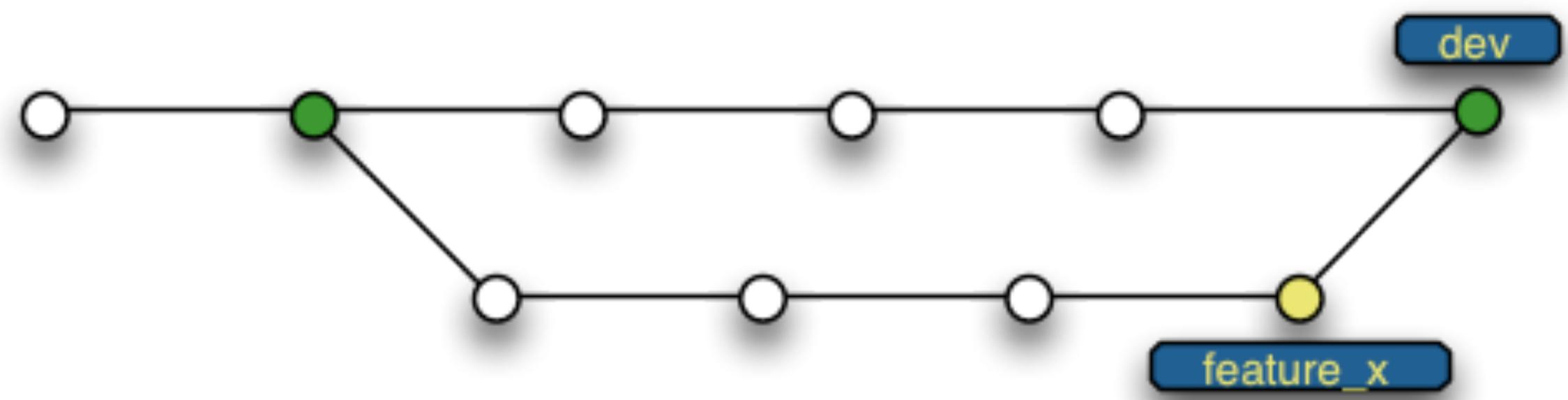
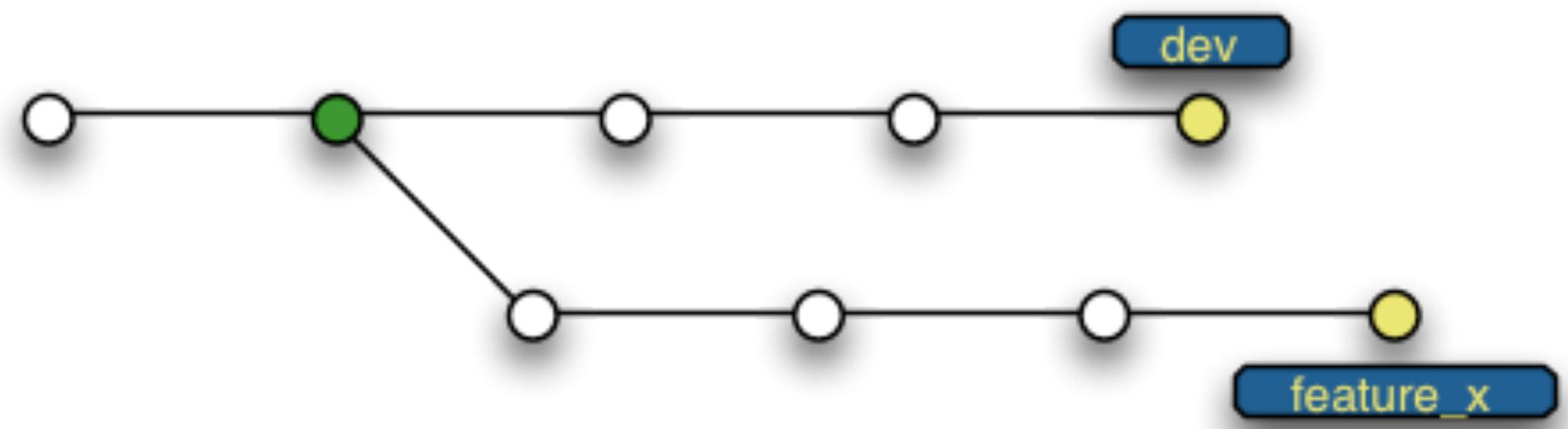
- git needs to know where your files live, that place is known as a remote
- most of the time we'll probably use a service like github or bitbucket to host our repositories
- when you clone a repository, a remote is created automatically, but sometimes you'll want to add remotes manually
- <https://help.github.com/articles/adding-a-remote/>

branching

- **PROBLEM:** *Version 1.1 of your web app is online. You are halfway through working on 1.2 but there are still bugs and missing features. An urgent bug surfaces in version 1.1 that requires fixing immediately.*
- We can't simply fix the bug in the file (e.g. index.php) where we are working on version 1.2 and ship all that. Why not? What are our options?

branching

- use branches religiously
 - new features
 - bug fixes
 - new ideas
 - ...



simple daily git workflow

git pull

pull all the changes from the remote repository

```
git checkout -b branch-name-here
```

create a new branch for your bug/feature/issue

DO YOUR WORK HERE

keep it in small chunks, the smaller your commits the better, in case things go wrong

git add .

and any new files you've created

git status and/or **git diff**

see the changes you're going to commit

```
git commit -m "Detailed message here"
```

make the commit with a nice detailed message

git checkout master

switch back to the master branch when the feature is done, your tests pass right?

```
git merge branch-name-here
```

update the master branch to update the master with all your changes

git push

send your changes up to the remote repository



branching

command	what it does
git checkout -b 0.1	create and switch to a branch called 0.1
git checkout 0.1	switch to a branch called 0.1
git branch	show the branch we are currently on

merge

- merge your own branch with another (e.g. the master) branch
- you have tested your code right?
- to merge, switch to the branch where you want to merge the changes (typically your production or *master* branch), then issue the following command
- `git merge branchname`

rebase

- rebase can be useful if you want to bring a local branch up to date with your master branch
- rebasing will fast forward your work on top of the latest version of the master if you use the command below
- compared to merging, rebasing doesn't create a new commit, which can keep your git history cleaner
- `git checkout yourbranch`
`git rebase master`

merge conflicts

- merge conflicts can occur if two branches you are trying to merge have conflicting code, e.g. by editing the same line of code with different contents
- it's impossible for git to know which version of the code you want to keep, so we'll have to handle that manually
- edit the conflicting file(s) manually and commit them to resolve conflicts
- <https://help.github.com/articles/resolving-a-merge-conflict-from-the-command-line>

```
1 <!-- Author: Marina Pushkova (marina@githowto.com) -->
2 <html>
3   <head>
4     <!-- no style -->
5   </head>
6   <body>
7     <===== HEAD
8       <h1>Hello, World! Life is great! (BRANCH A)</h1>
9
10      <h1>Hello, World! Life is great! (Branch B)</h1>
11    >>>>> branch_b
12   </body>
13 </html>
```

This is your current branch !

The code from other branch

```
git mergetool
// change your files and commit again
```

Kaleidoscope

header.php 2 +

A prosto-old > header.php B prosto > header.php

1	<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 2 <html xmlns="http://www.w3.org/1999/xhtml" <?php language_attri 3 <head profile="http://gmpg.org/xfn/11"> 4 <script type="text/javascript">var _sf_startpt=(new Date()) 5 <meta http-equiv="Content-Type" content="<?php bloginfo('ht 6 <title><?php wp_title('«', true, 'right'); ?><?php bl 7 <link rel="stylesheet" href="<?php bloginfo('template_direc 8 <link rel="stylesheet" href="<?php bloginfo('template_direc 9 <link rel="stylesheet" href="<?php bloginfo('template_direc 10 <link rel="stylesheet" href="<?php bloginfo('stylesheet_url 11 <link rel="pingback" href="<?php bloginfo('pingback_url') 12 <?php if (is_singular()) wp_enqueue_script('comment-re 13 <link rel="pingback" href="<?php bloginfo('pingback_url'); 14 <?php if (is_singular()) wp_enqueue_script('comment-repl 15 <?php wp_head(); ?> 16 <link rel="stylesheet" media="only screen and (-webkit-min 17 <script type="text/javascript" language="javascript" > 18 Cufon.replace('.logo span, .single h1, h2, h3, h4 , h5, 19 </script> <script type='text/javascript' src='<?php bloginfo('templat 20 <script type="text/javascript" language="javascript" > 21 <script type="text/javascript" language="javascript" > 22 Cufon.replace('.cocon, h1, .page_title h2 ', { fontFami 23 </script> 24 </head> 25 <body <?php if (get_post_meta(\$post->ID, "page_isportfolio", tr 26 <div class="wrapper"> 27 <div class="header"> 28 <div class="wrap">	1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 2 <html xmlns="http://www.w3.org/1999/xhtml" <?php language_att 3 <head profile="http://gmpg.org/xfn/11"> 4 <script type="text/javascript">var _sf_startpt=(new Date()) 5 <meta http-equiv="Content-Type" content="<?php bloginfo('ht 6 <title><?php wp_title('«', true, 'right'); ?><?php bl 7 <link rel="stylesheet" href="<?php bloginfo('template_direc 8 <link rel="stylesheet" href="<?php bloginfo('template_direc 9 <link rel="stylesheet" href="<?php bloginfo('template_direc 10 <link rel="stylesheet" href="<?php bloginfo('stylesheet_url 11 <link rel="pingback" href="<?php bloginfo('pingback_url') 12 <?php if (is_singular()) wp_enqueue_script('comment-re 13 <?php wp_head(); ?> 14 <link rel="stylesheet" media="only screen and (-webkit-min 15 <?php if (get_option(SHORTNAME."_cufondisable") == '' & 16 <script type="text/javascript" language="javascript" > 17 Cufon.replace('.logo span, .single h1, h2, h3, h4 , h5, 18 </script> <script type='text/javascript' src='<?php bloginfo('templat 19 <script type="text/javascript" language="javascript" > 20 <script type="text/javascript" language="javascript" > 21 <script type="text/javascript" language="javascript" > 22 Cufon.replace('.cocon, h1, .page_title h2 ', { fontFami 23 <?php } ?> 24 </head> 25 <body <?php if (get_post_meta(\$post->ID, "page_isportfolio", tr 26 <div class="wrapper"> 27 <div class="header"> 28 <div class="wrap">
---	--	--

Git tools

- The command line (cmder on windows!)
- Github for Windows/Mac
- Tower for Mac
- Mergetools like FileMerge, Kaleidoscope, P4Merge (free), Diffmerge (free)
- <https://git-scm.com/downloads/guis>
- Learn the command line first, before you switch to a graphical tool

```
1. jorre@imac-jorre: /Applications/MAMP/htdocs/_thomasmore_courses/2imd-webtech2-labs (zsh)
→ 2imd-webtech2-labs git:(master) git status
# On branch master
nothing to commit, working directory clean
→ 2imd-webtech2-labs git:(master) git --help
usage: git [--version] [--help] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p|--paginate|--no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

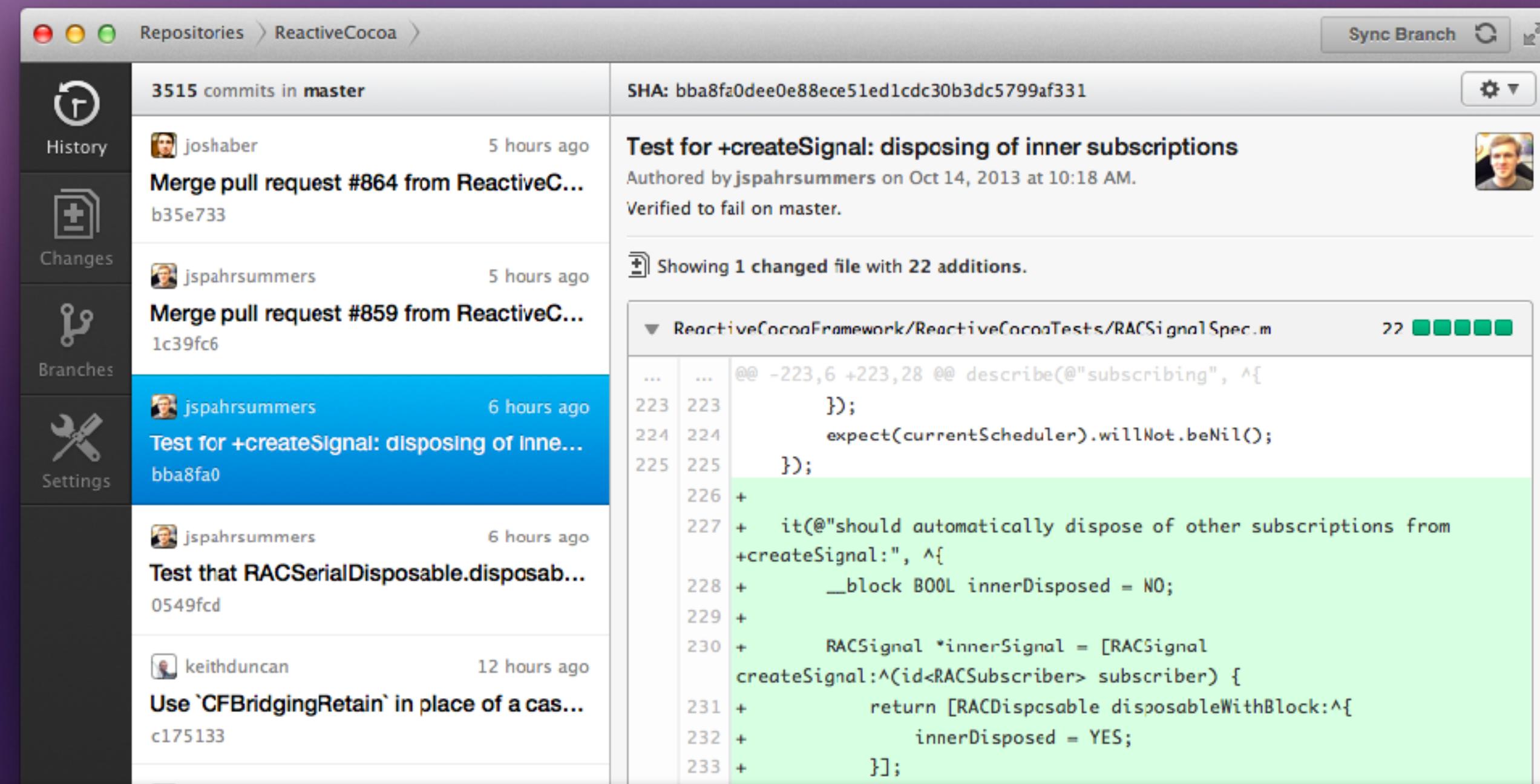
The most commonly used git commands are:
add      Add file contents to the index
bisect   Find by binary search the change that introduced a bug
branch   List, create, or delete branches
checkout Checkout a branch or paths to the working tree
clone    Clone a repository into a new directory
commit   Record changes to the repository
diff     Show changes between commits, commit and working tree, etc
fetch   Download objects and refs from another repository
grep    Print lines matching a pattern
init    Create an empty Git repository or reinitialize an existing one
log     Show commit logs
merge   Join two or more development histories together
mv      Move or rename a file, a directory, or a symlink
pull   Fetch from and merge with another repository or a local branch
push    Update remote refs along with associated objects
rebase  Forward-port local commits to the updated upstream head
reset   Reset current HEAD to the specified state
rm      Remove files from the working tree and from the index
show    Show various types of objects
status  Show the working tree status
tag     Create, list, delete or verify a tag object signed with GPG

'git help -a' and 'git help -g' lists available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
→ 2imd-webtech2-labs git:(master)
```

The easiest way to use GitHub on Mac.

[Download GitHub for Mac](#)

OS X 10.7 or later



✓ Clone repositories

✓ Browse history

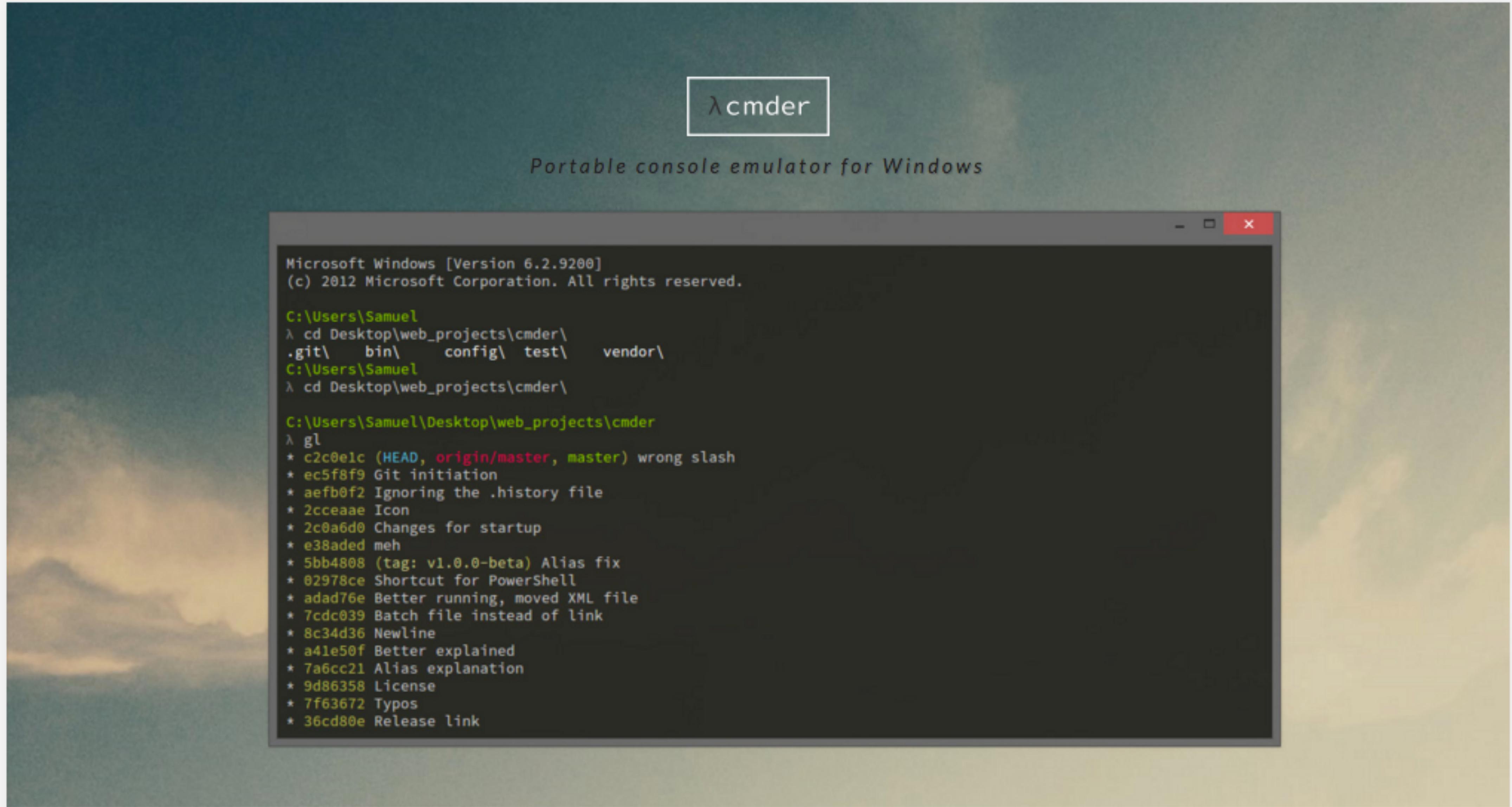
✓ Commit changes

✓ Branch code

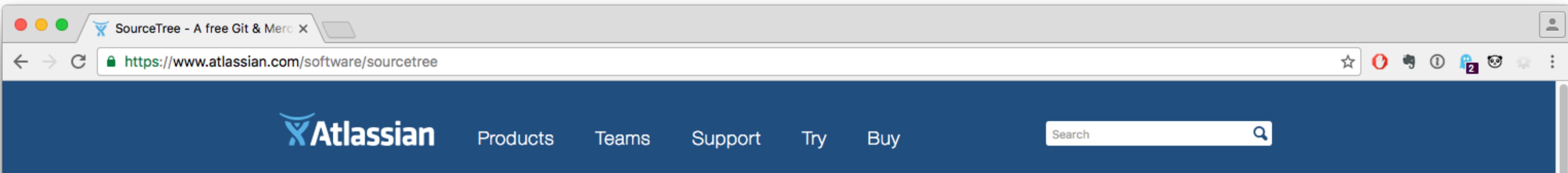
✓ Share code on github.com

Sharing code should be as simple as possible.

That's why we created GitHub for Mac.



Cmder is a software package created out of pure frustration over the absence of nice console emulators on Windows. It is based on amazing software, and spiced up with the Monokai color scheme and a custom prompt layout. Looking sexy from the start.

[Download free](#)

Harness the power of Git and Hg in a beautifully simple application

[Download for Mac OS X](#)

Also available for Windows

A screenshot of the SourceTree application window titled 'sourcetree-website (Git)'. The window shows a commit history graph on the left and a detailed list of commits on the right. The commits are listed in chronological order from top to bottom. Each commit includes the author, date, and a brief description. The commit history graph shows various branches and merges.



Join the SourceTree Beta Program

Want access to new features and improvements before they are in production? Sign up for the Beta program to try new features, provide feedback and engage with the SourceTree Team.

[Sign up now](#)

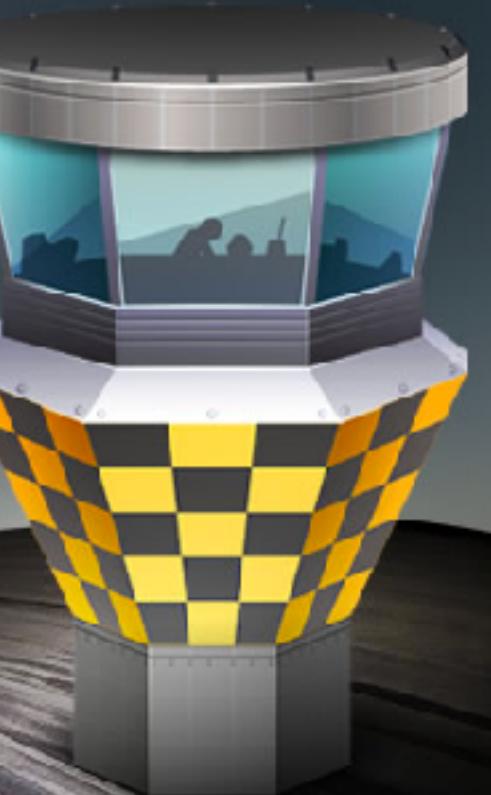
TOWER

the most powerful Git client for Mac

Download Trial

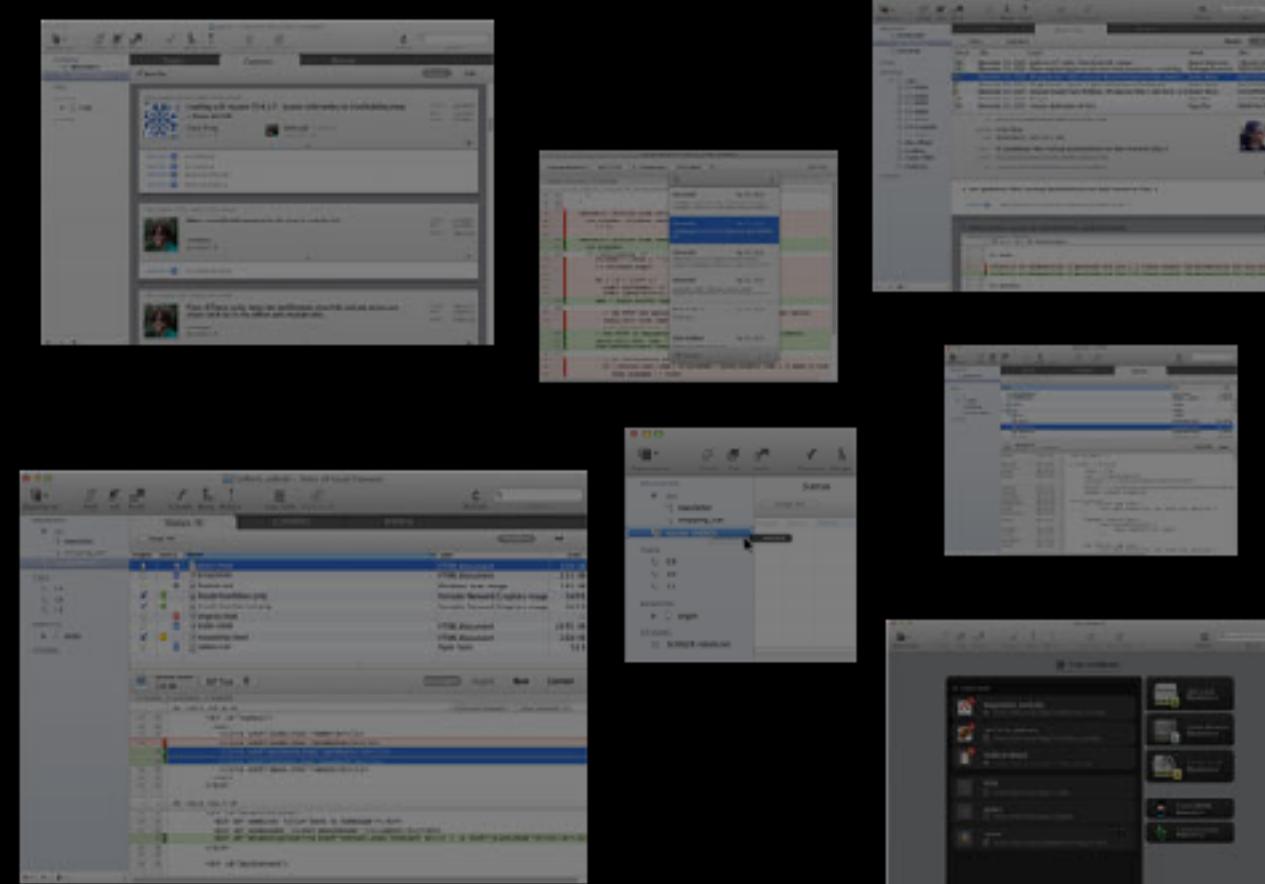
Buy Now

Free for 30 days. All features. Version 1.6.2 [release notes](#)
Download trial for Mac OS X 10.5



Version control with Git - made easy.

Git is the future of version control. But using it on the command line can be difficult.
Make your life easier with Tower - the most powerful Git client for your Mac.



Screencast

Tower in 2 minutes.

From Beginner...

Learn Git with Tower. With its **easy-to-use interface** you'll be up to speed in no time. Tower takes the pain out of Git and makes complex tasks **simple**.

...to Master

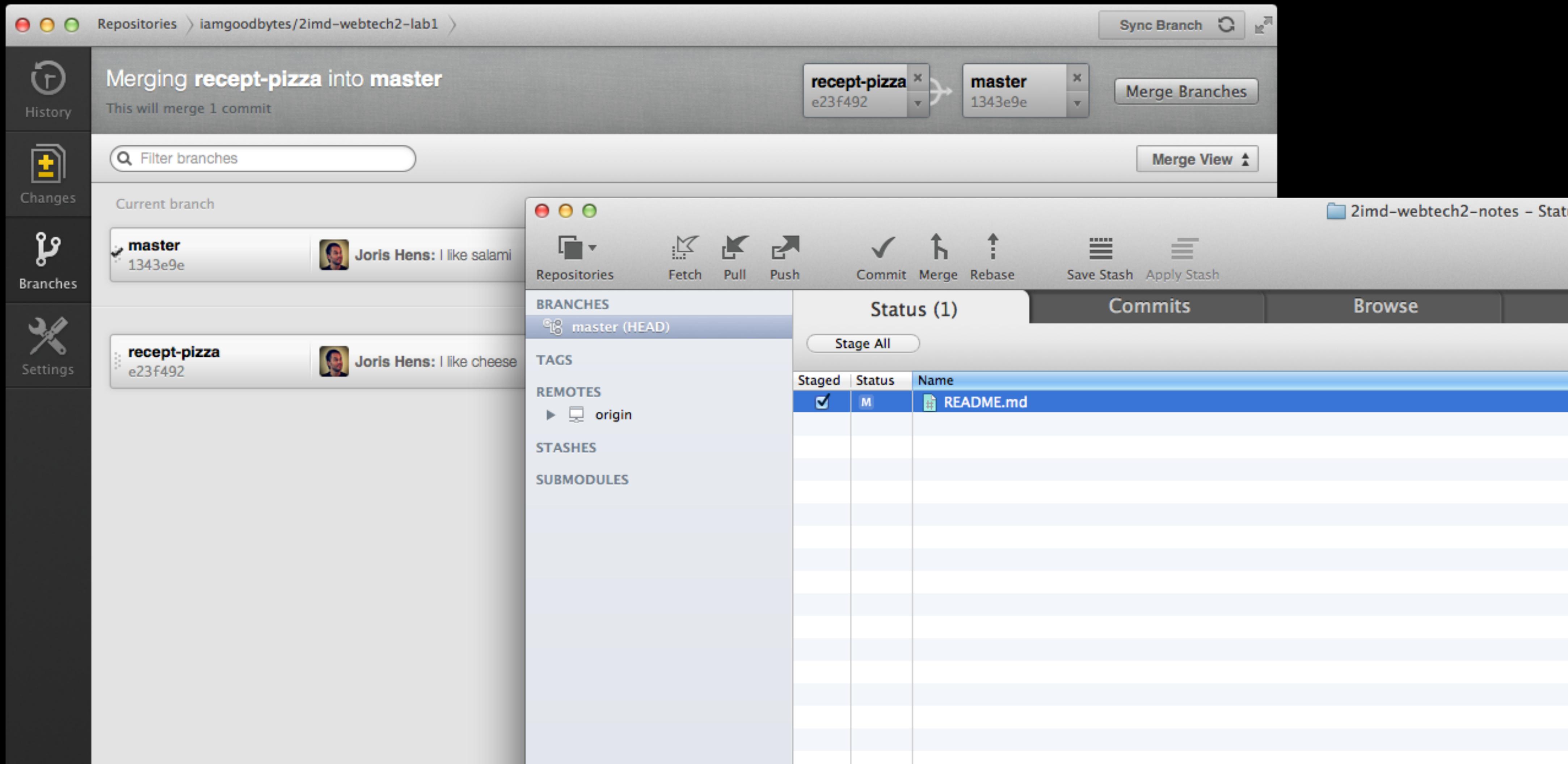
Tower has all of Git's **advanced features** waiting for you: single line staging, submodule support, file history...
Increase your productivity - without missing Git's power.

From Confidence

to Productivity

Git is the Future

same flow, different UI



pull requests

- contribute to projects on GitHub
- a pull request notifies a project owner that you want to submit changes to their project
- always try to use a feature branch when creating pull request
- **Assignment: Lab1**

pull requests

1. **fork** the repository you want to contribute to on [github.com](#) (this will create a copy of the project on your own GitHub account)
2. (if you want you can add a git remote called *upstream* that references the original project in order to pull in updates. Run `git fetch upstream` and `git merge upstream/master` to do so)
3. make changes in a feature branch and publish the branch when ready
4. go to GitHub and issue your pull request
5. the repository owner will evaluate and merge your code

deployment

- overview of how real life deployment works
- example: revisebeta and revise
- *no more FTP!*

Course notes

- clone + pull them from github
- be an active contributor to this course via pull requests and earn extra credits / contribute by adding interesting links
- <https://github.com/iamgoodbytes/2imd-webtech2-notes>

Pull request tryout?

Help keep the course notes up to date

1. *remove the part about “Advanced Javascript” and replace it with “ES6”*
2. *add some interesting links about learning ES6*
3. *remove the part about Angular, replace it with React, add some links*

To pass this level you...

- can explain all core Git concepts to a newbie
 - GIT workshop for the IMD designers, who's in?
- can use Git to manage a real word project by using branches
- can contribute to a project by using pull requests on GitHub.com
- know what a centralised vs a distributed version control system is
- have successfully finished your Lab1 assignment (see toledo)
- have completed the intro at <https://try.github.io>
- have completed <https://nodeschool.io/> git-it (npm install -g git-it)

GIT + GITHUB : VERSION CONTROL + SOCIAL CODING

- » GET GIT [COMPLETED]
 - » REPOSITORY [COMPLETED]
 - » COMMIT TO IT
 - » GITHUBBIN
 - » REMOTE CONTROL
 - » FORKS AND CLONES
 - » BRANCHES AREN'T JUST FOR BIRDS
 - » IT'S A SMALL WORLD
 - » PULL, NEVER OUT OF DATE
 - » REQUESTING YOU PULL, PLEASE
 - » MERGE TADA!
-

HELP

EXIT

THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



GitHub Education

[Stories](#)[Events](#)[Student pack](#)[Classroom guide](#)[Contact us](#)[Request a discount](#)

Student Developer Pack

The best developer tools, free for students



Learn to ship software like a pro



26k



8,856

There's no substitute for hands-on experience, but for most students, real world tools can be cost prohibitive. That's why we created the GitHub Student Developer Pack with some of our partners and friends: to give students free access to the best developer tools in one place so they can learn by doing.

[Get your pack](#)



1.1 · Got 15 minutes and want to learn Git?

Git allows groups of people to work on the same documents (often code) at the same time, and without stepping on each other's toes. It's a distributed version control system.

Our terminal prompt below is currently in a directory we decided to name "octobox". To initialize a Git repository here, type the following command:

→ **git init**



TryGit—1300x310

Press enter to submit commands

> | https://try.github.io

My Octobox Repository

Advice



Directory:

A folder used for storing multiple files.

Repository:

A directory where Git has been initialized to start version controlling your files.