Introduction to Git for Evolutionary Biologists

OR

ON

THE ORIGIN OF REPOSITORIES

BY MEANS OF VERSION CONTROL,

ON

THE ORIGIN OF REPOSITORIES

BY MEANS OF VERSION CONTROL,

OR THE

ON

THE ORIGIN OF REPOSITORIES

BY MEANS OF VERSION CONTROL,

OR THE

PRESERVATION OF FAVOURED COMMITS IN THE STRUGGLE AGAINST MERGE CONFLICTS.

ON

THE ORIGIN OF REPOSITORIES

BY MEANS OF VERSION CONTROL,

OR THE

PRESERVATION OF FAVOURED COMMITS IN THE STRUGGLE AGAINST MERGE CONFLICTS.

By MICHAEL MAY, F.R.S.
FELLOW OF THE ROTHFELS SOCIETY

Git is a *version control system* used originally for collaborative programming.

Git is a *version control system* used originally for collaborative programming.

GitHub is a git repository hosting service. There are others (e.g., *Bitbucket*)!

Git is a *version control system* used originally for collaborative programming.

GitHub is a git repository hosting service. There are others (e.g., *Bitbucket*)!

There are lots of good GUIs for using Git (I like GitKraken), but *most* interaction with git involves just a few basic commands that can be handled in Terminal/Command Line.

Desktop

Sourcetree

axosoft
GitKraken

Git involves *remote repositories*, which everyone has access to, and *local repositories,* which are (potentially modified) copies of the remote on one user's computer.

Git involves *remote repositories*, which everyone has access to, and *local repositories,* which are (potentially modified) copies of the remote on one user's computer.

Users share their changes by *pushing* their local changes to the remote repository.

Git involves *remote repositories*, which everyone has access to, and *local repositories,* which are (potentially modified) copies of the remote on one user's computer.

Users share their changes by *pushing* their local changes to the remote repository.

Users are required to ensure that their local changes do not conflict with the remote repository.

Git involves *remote repositories*, which everyone has access to, and *local repositories,* which are (potentially modified) copies of the remote on one user’s computer.

Users share their changes by *pushing* their local changes to the remote repository.

Users are required to ensure that their local changes do not conflict with the remote repository.

Merge conflicts are resolved on the local repository before *pushing* can occur.

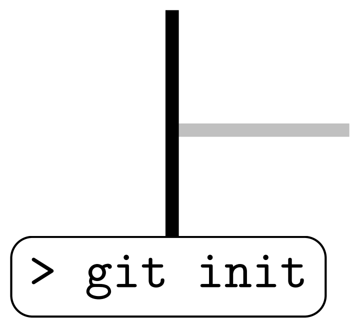
First, an initial repository emerges from the primordial soup.

First, an initial repository emerges from the primordial soup.

Initialize a repository.

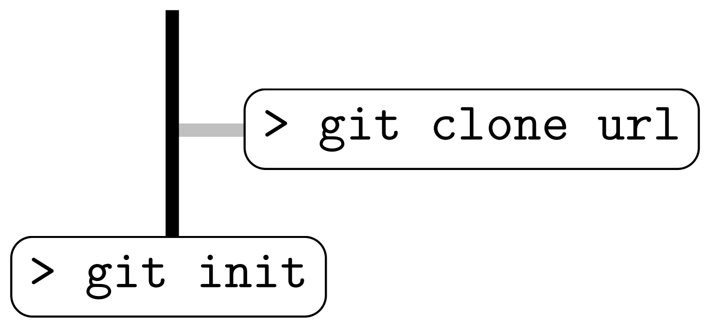
```
> git init
```

Local populations bud off from the main lineage.

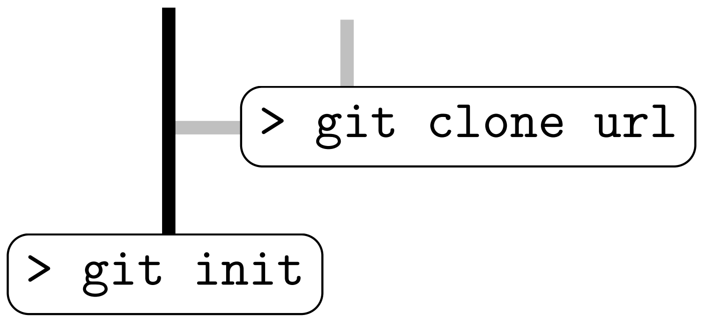


Local populations bud off from the main lineage.

Make a local version of the remote repository using git clone <url>

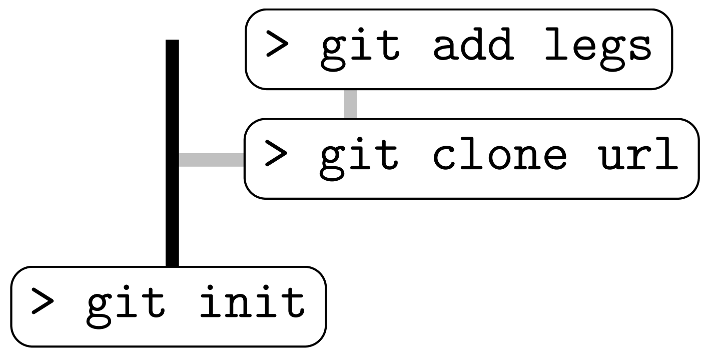


The repository evolves for a little while, as *local repositories* accumulate mutations.



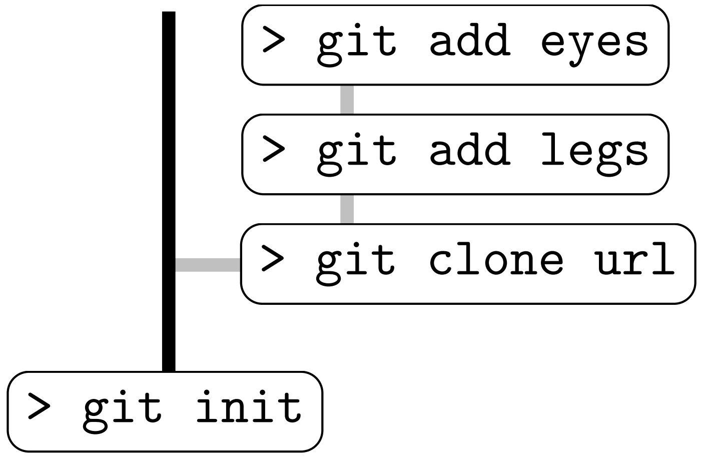
The repository evolves for a little while, as *local repositories* accumulate mutations.

Add your changes to the local repository using git add <filename> or git add .



The repository evolves for a little while, as *local repositories* accumulate mutations.

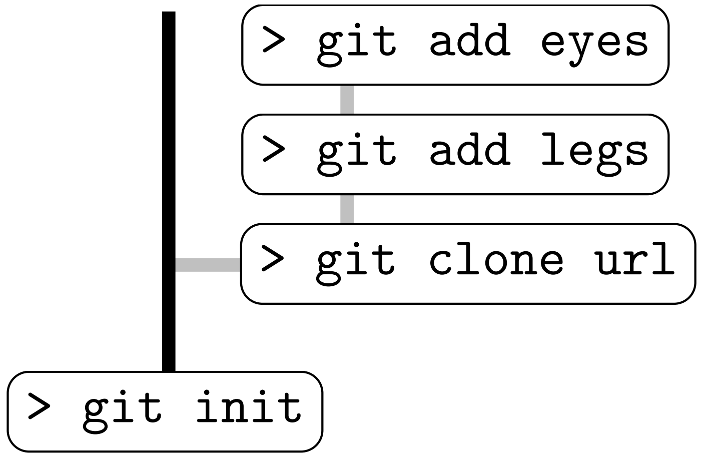
Add your changes to the local repository using git add <filename> or git add .



The repository evolves for a little while, as *local repositories* accumulate mutations.

Add your changes to the local repository using git add <filename> or git add .

A “change” means any NEW file or a change to an EXISTING file.

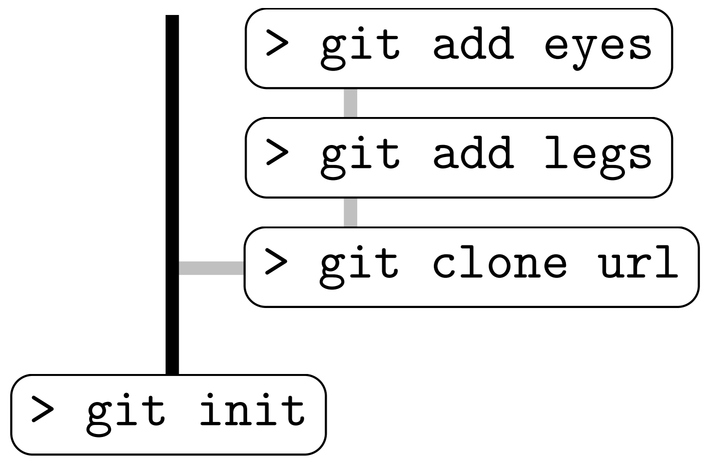


The repository evolves for a little while, as *local repositories* accumulate mutations.

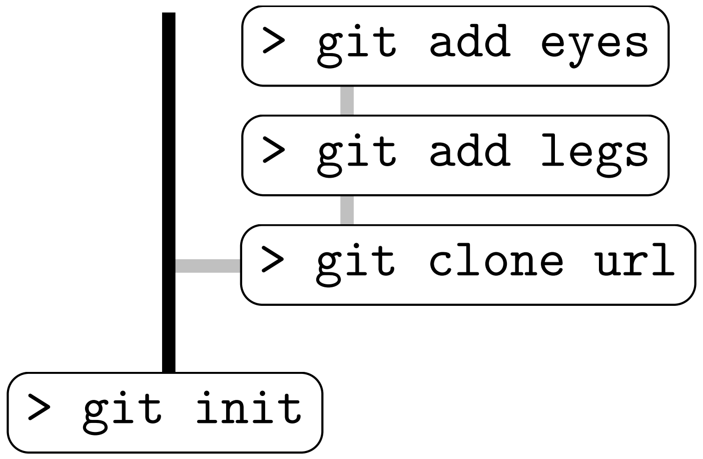
Add your changes to the local repository using git add <filename> or git add .

A “change” means any NEW file or a change to an EXISTING file.

Changes only every occur in local repositories.

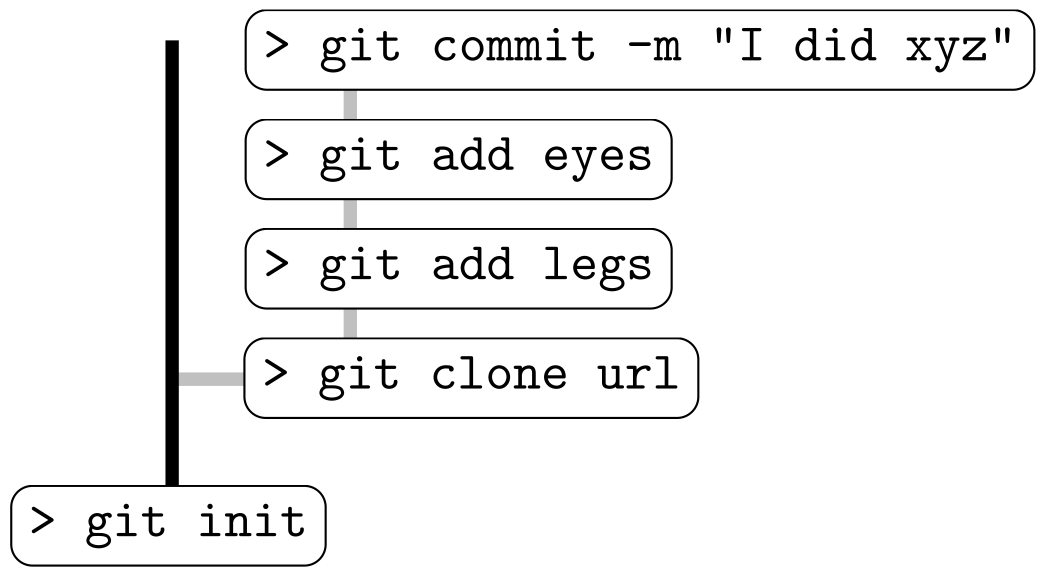


Mutations become fixed within populations.

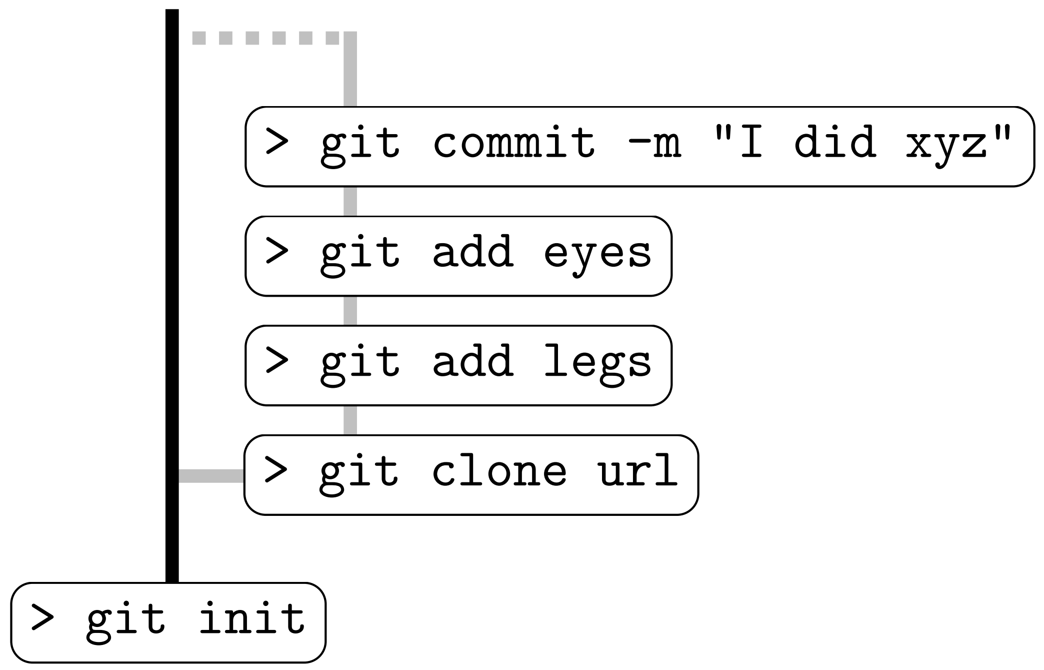


Mutations become fixed within populations.

Commit your changes to your local repository.

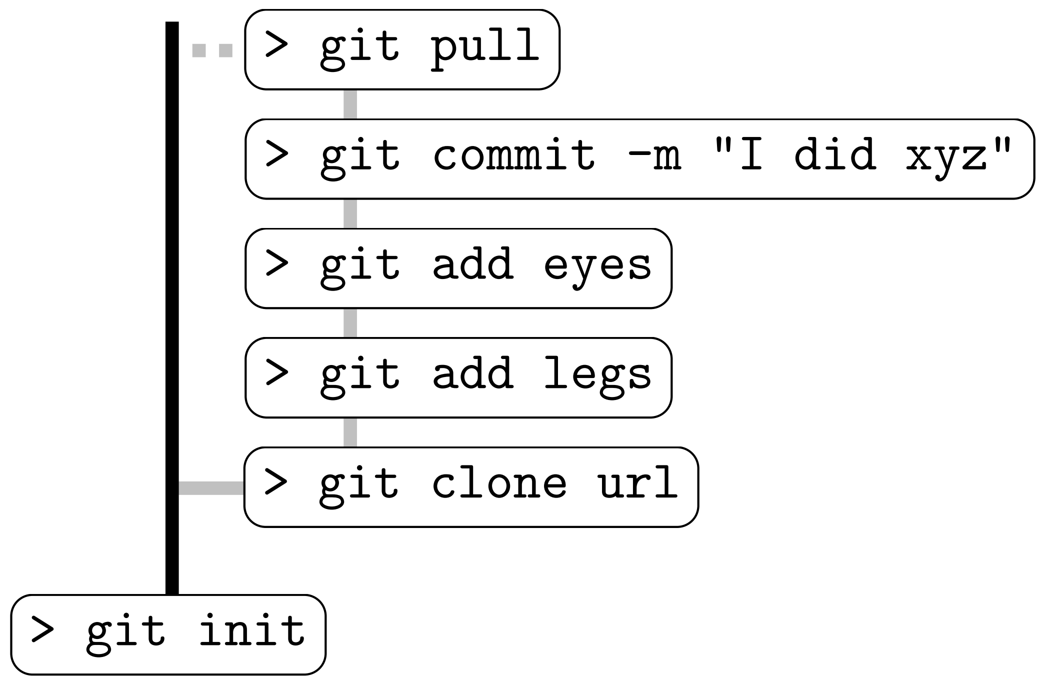


Repos can only hybridize if they have not evolved any reproductive isolation!



Repos can only hybridize if they have not evolved any reproductive isolation!

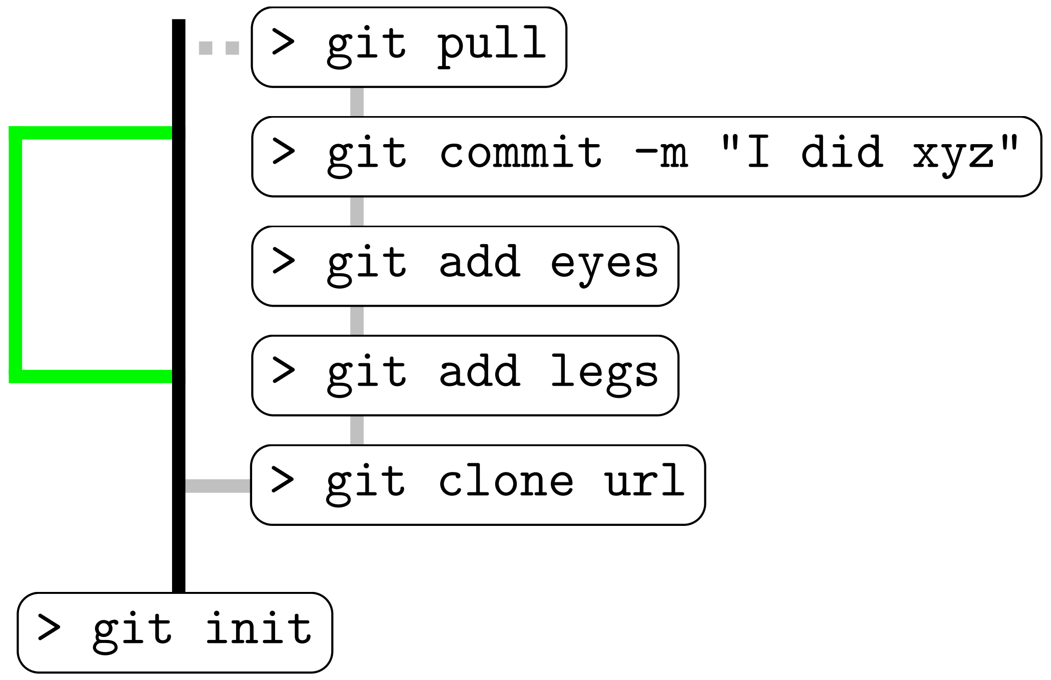
Make sure your local repository does not conflict with the remote.



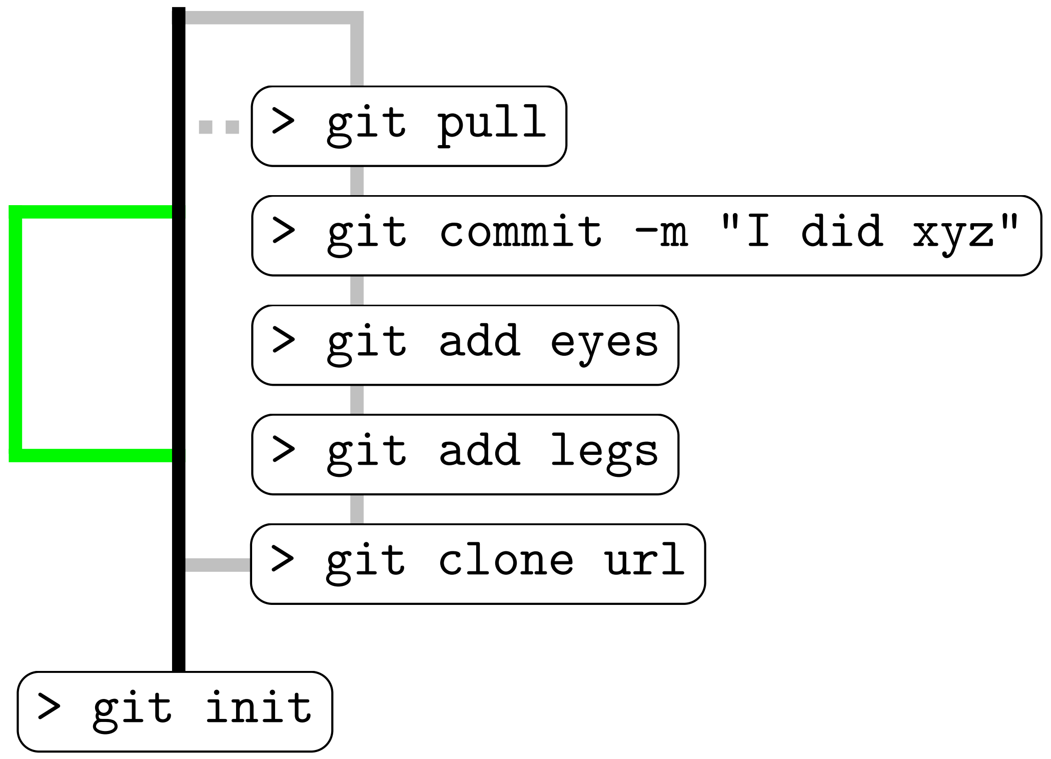
Repos can only hybridize if they have not evolved any reproductive isolation!

Make sure your local repository does not conflict with the remote.

Deal with any merge conflicts manually as they arise.

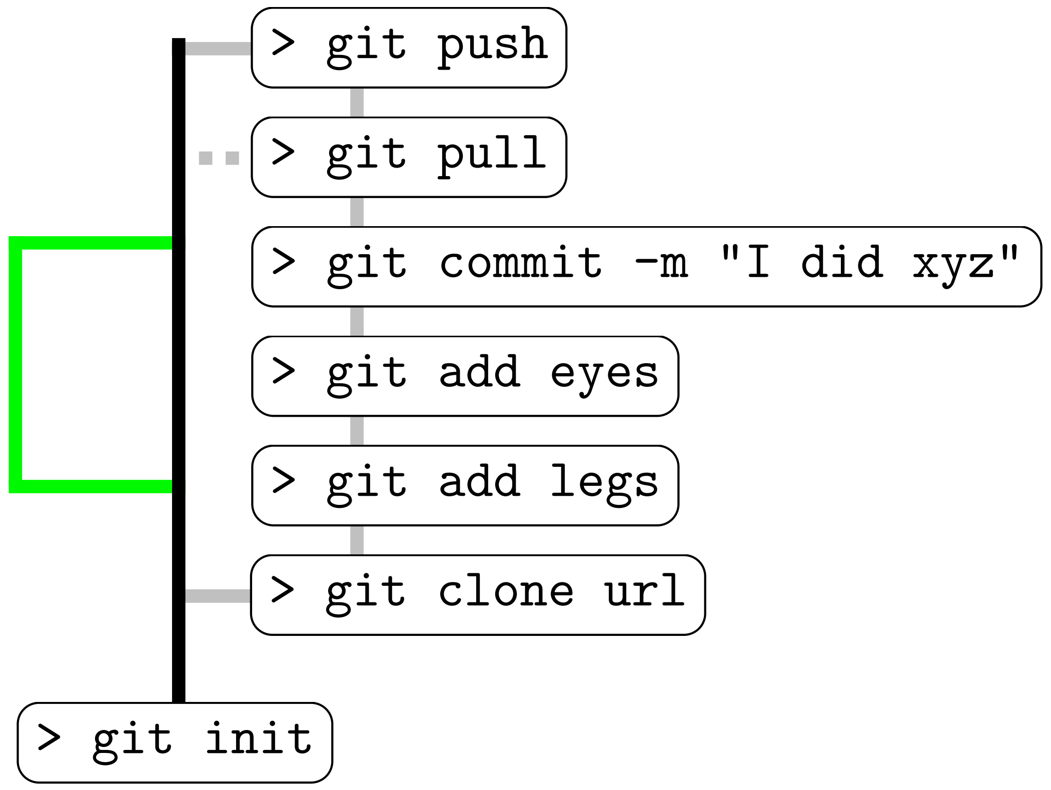


Repos that are not reproductively isolated may fuse.

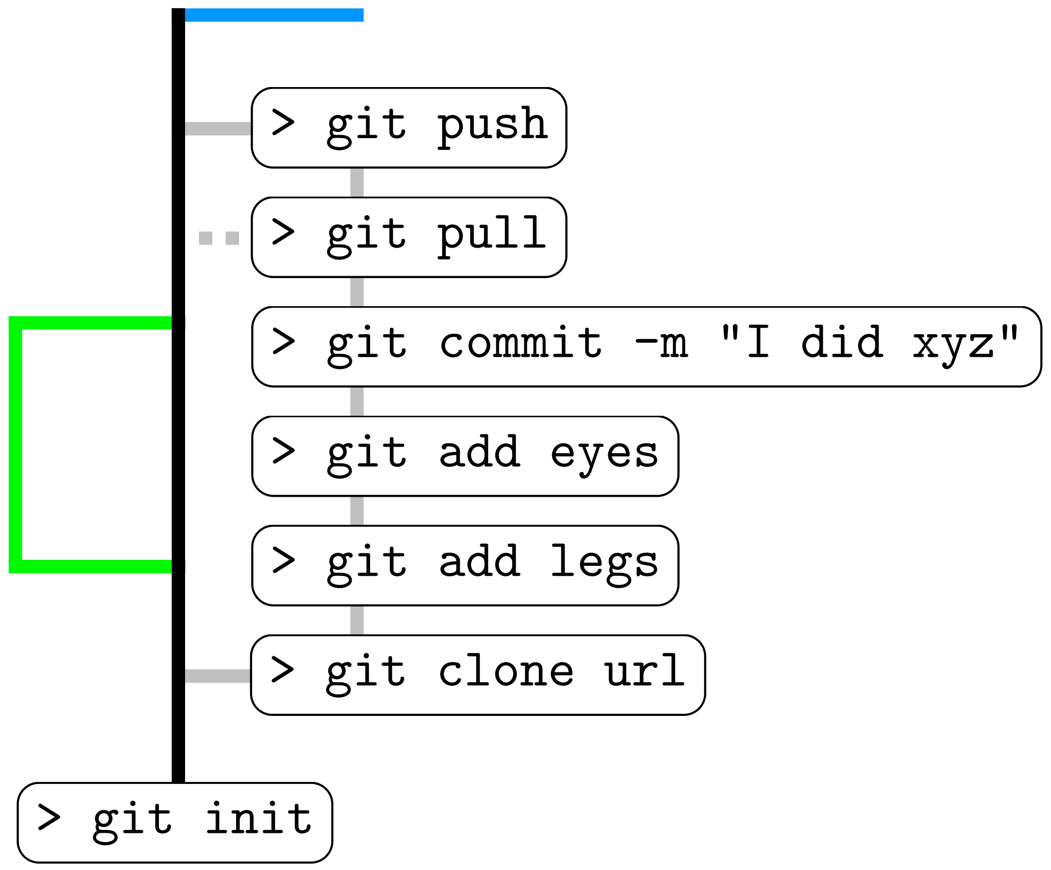


Repos that are not reproductively isolated may fuse.

Push your local changes to the remote repository.

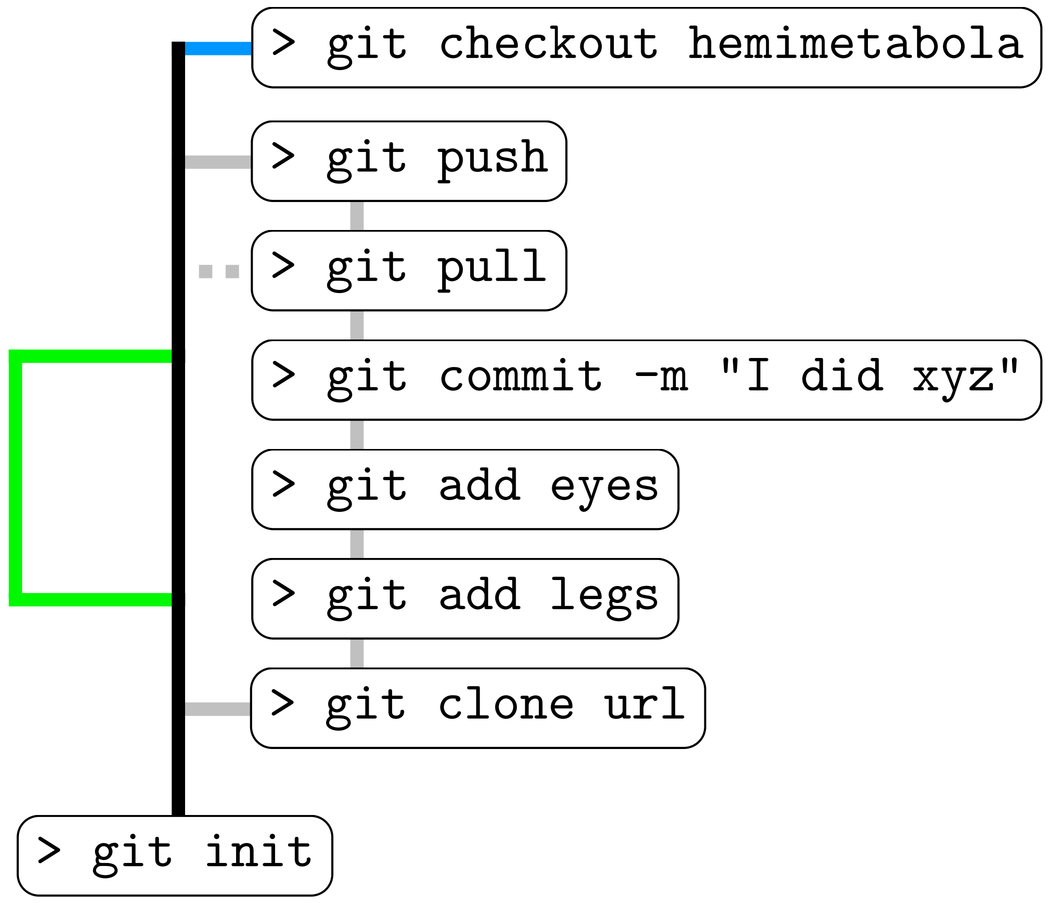


Repos that are very different may become new branches.



Repos that are very different may become new branches.

Create a new branch with git checkout <branch name>.

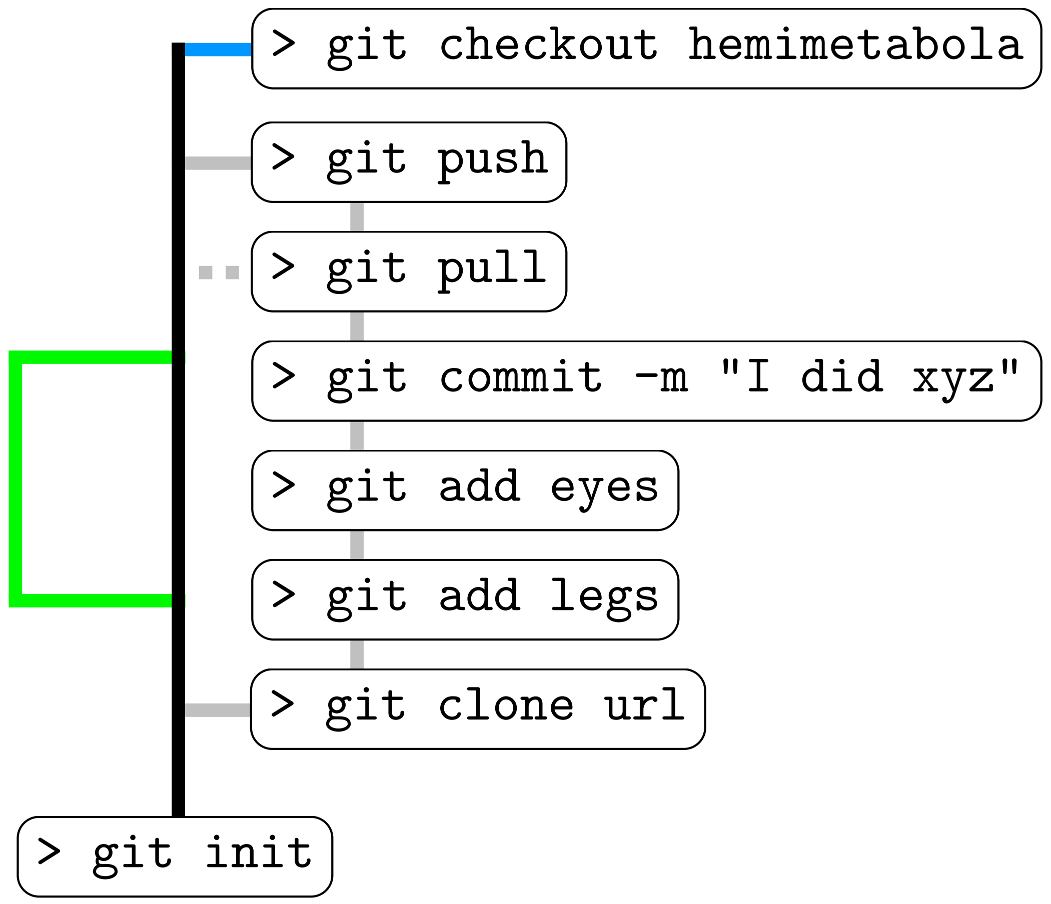


```
> git checkout hemimetabola
> git push
> git pull
> git commit -m "I did xyz"
> git add eyes
> git add legs
> git clone url
> git init
```

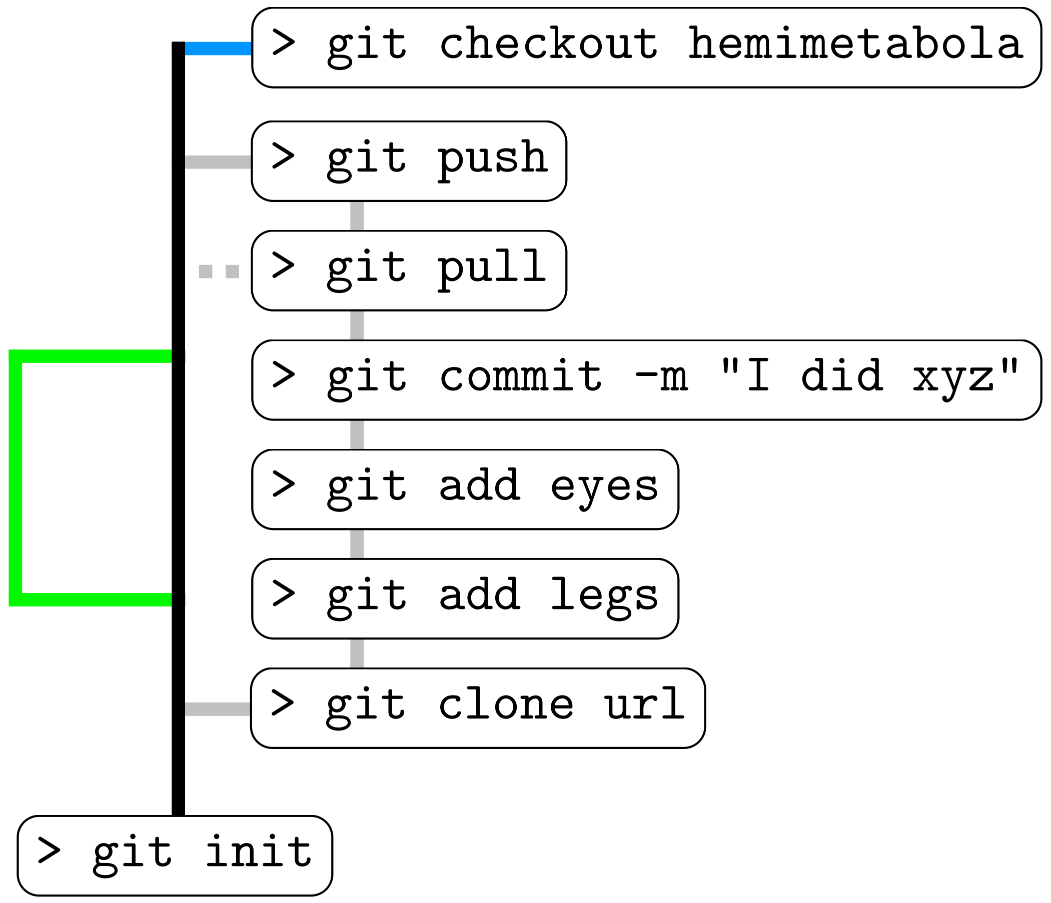
Repos that are very different may become new branches.

Create a new branch with git checkout <branch name>.

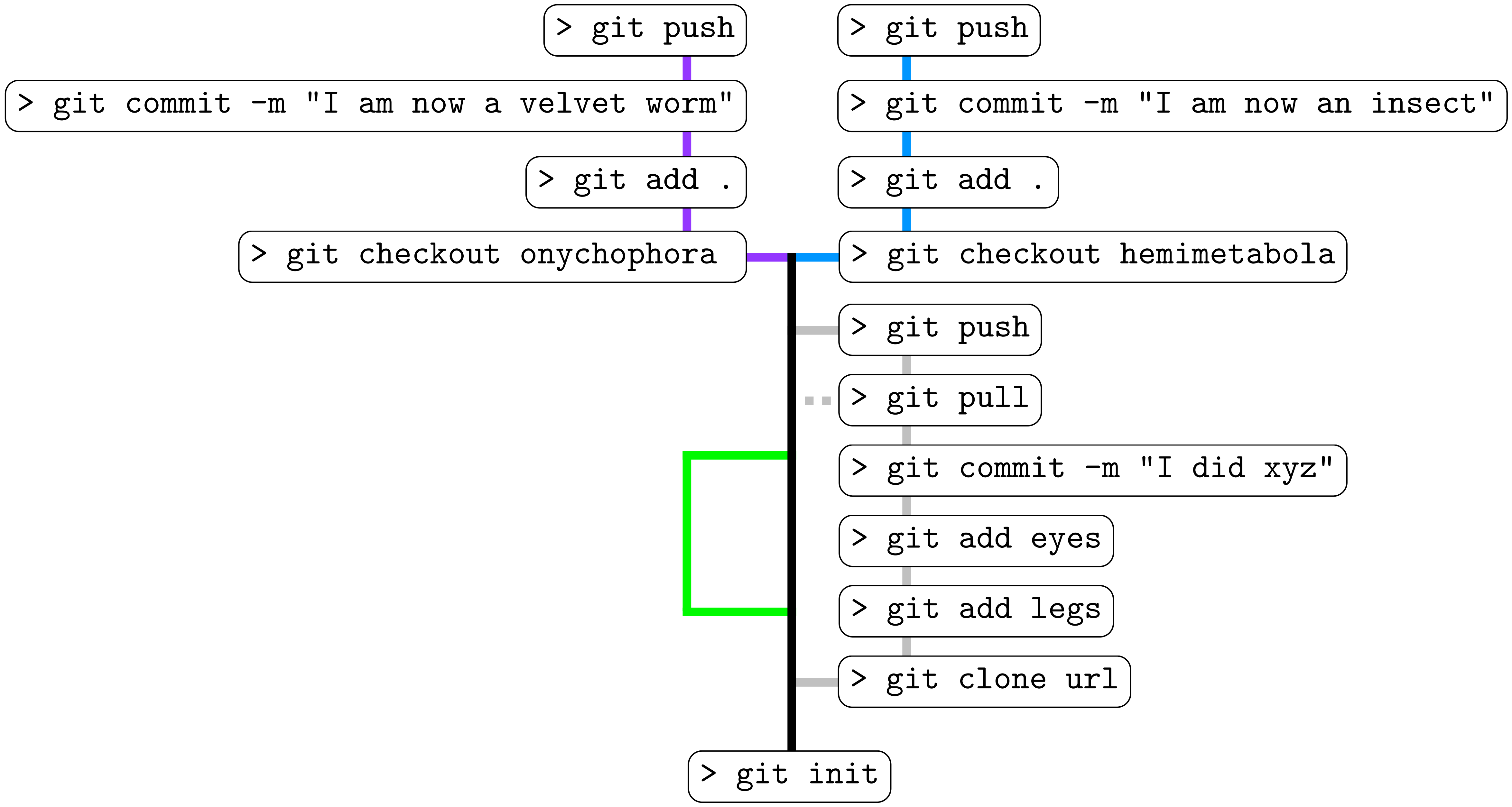
Create a new branch if you want people to collaborate on new features that aren't in the "main" lineage!



Branches can merge to create new lineages with features of both ancestors.

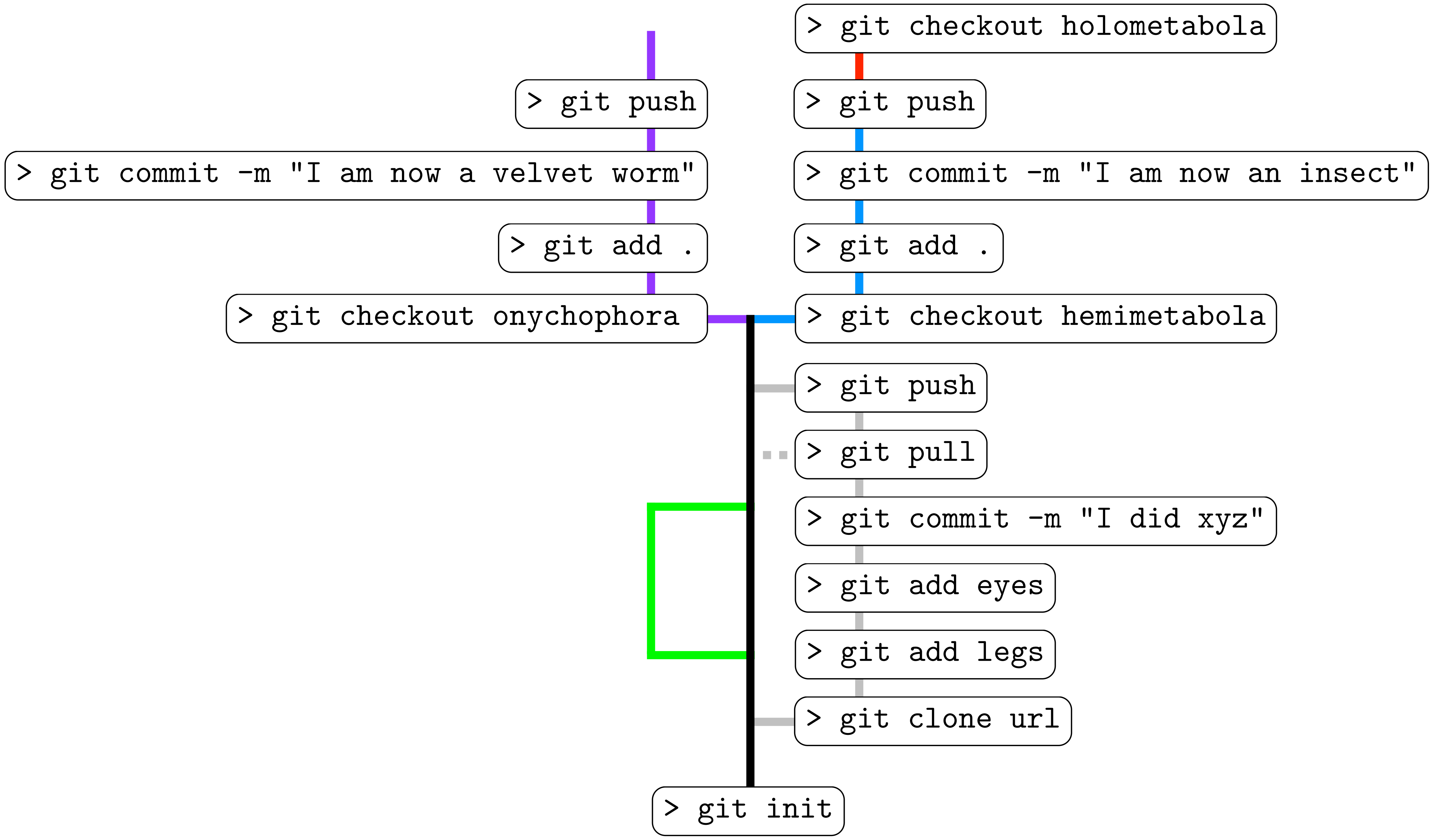


Branches can merge to create new lineages with features of both ancestors.



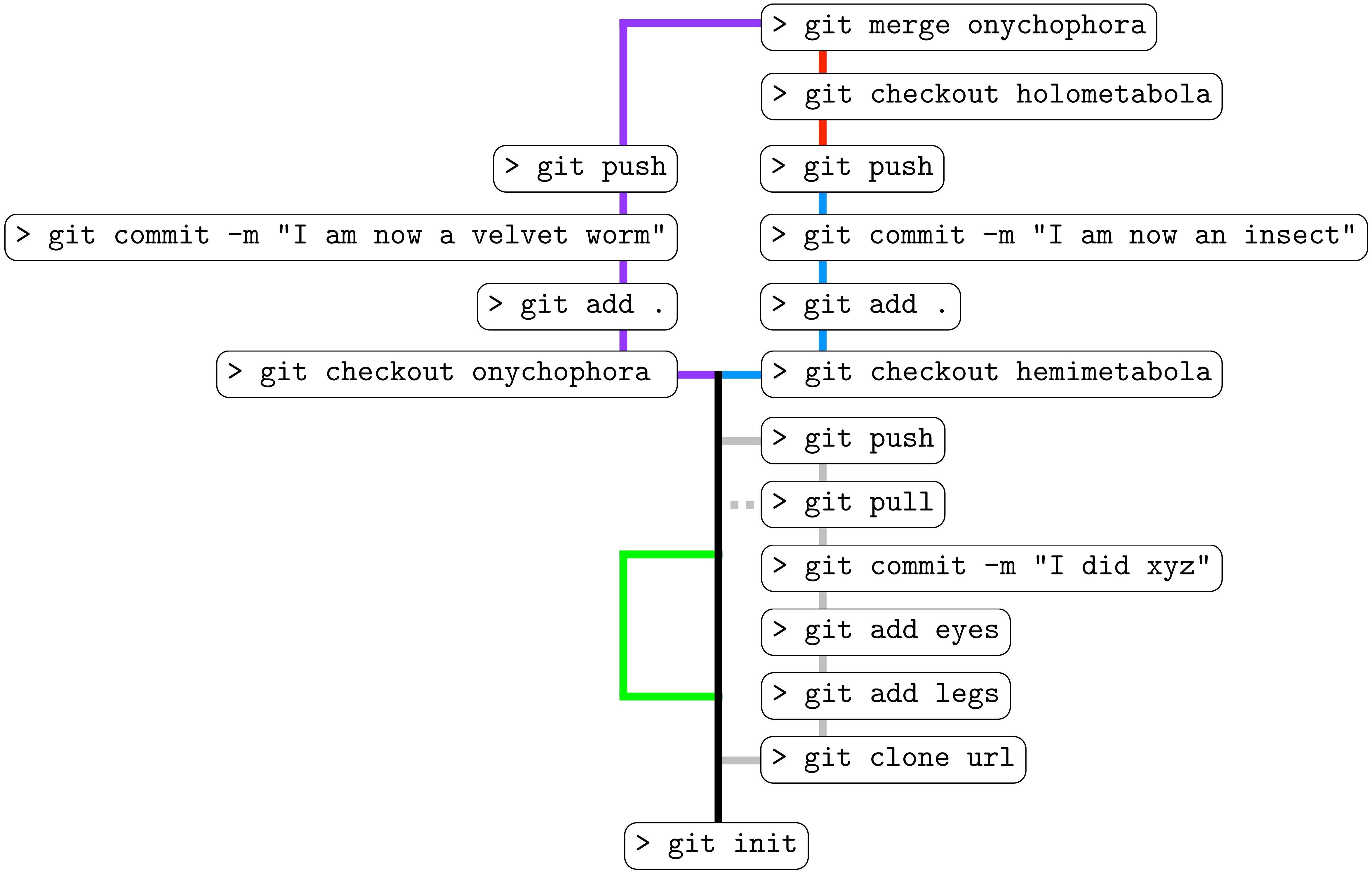
> git push
> git commit -m "I am now a velvet worm"
> git add .
> git checkout onychophora
> git push
> git commit -m "I am now an insect"
> git add .
> git checkout hemimetabola
> git push
> git pull
> git commit -m "I did xyz"
> git add eyes
> git add legs
> git clone url
> git init

Branches can merge to create new lineages with features of both ancestors.



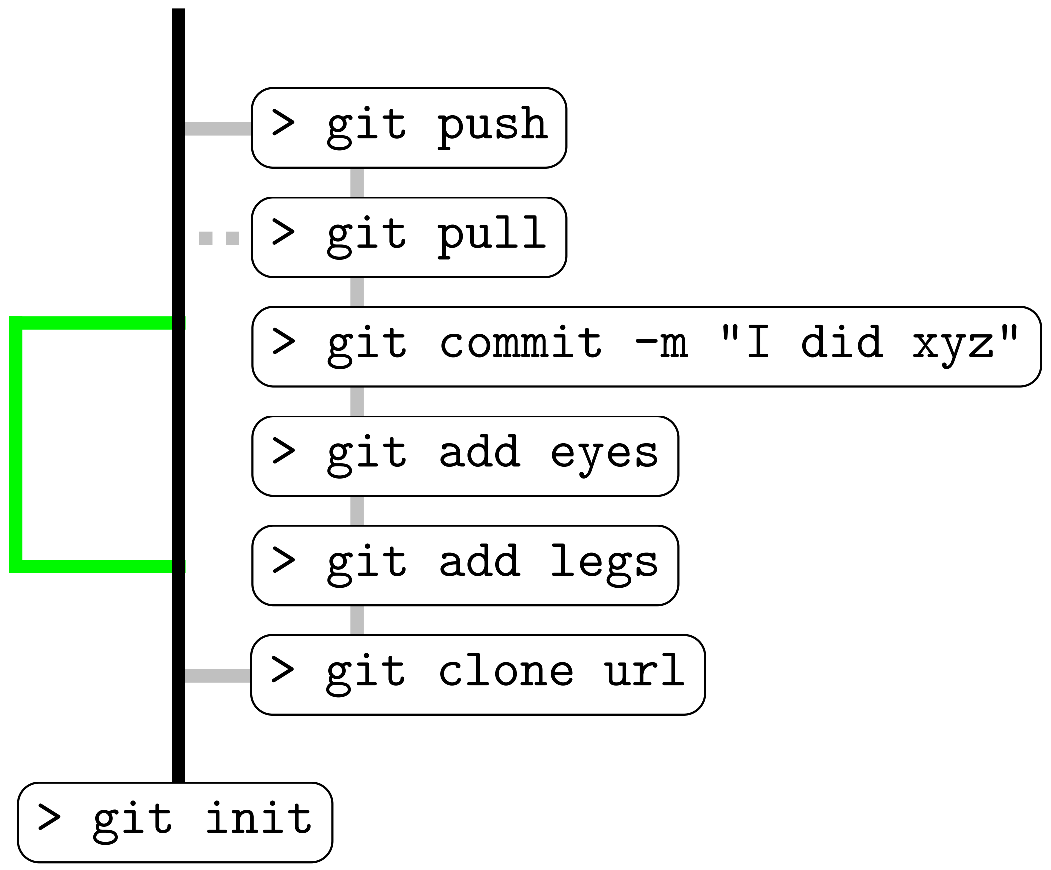
> git checkout holometabola
> git push
> git push
> git commit -m "I am now a velvet worm"
> git commit -m "I am now an insect"
> git add .
> git add .
> git checkout onychophora
> git checkout hemimetabola
> git push
> git pull
> git commit -m "I did xyz"
> git add eyes
> git add legs
> git clone url
> git init

Branches can merge to create new lineages with features of both ancestors.


> git merge onychophora
> git checkout holometabola
> git push
> git push
> git commit -m "I am now a velvet worm"
> git commit -m "I am now an insect"
> git add .
> git add .
> git checkout onychophora
> git checkout hemimetabola
> git push
> git pull
> git commit -m "I did xyz"
> git add eyes
> git add legs
> git clone url
> git init

Large repositories can get pretty complicated.

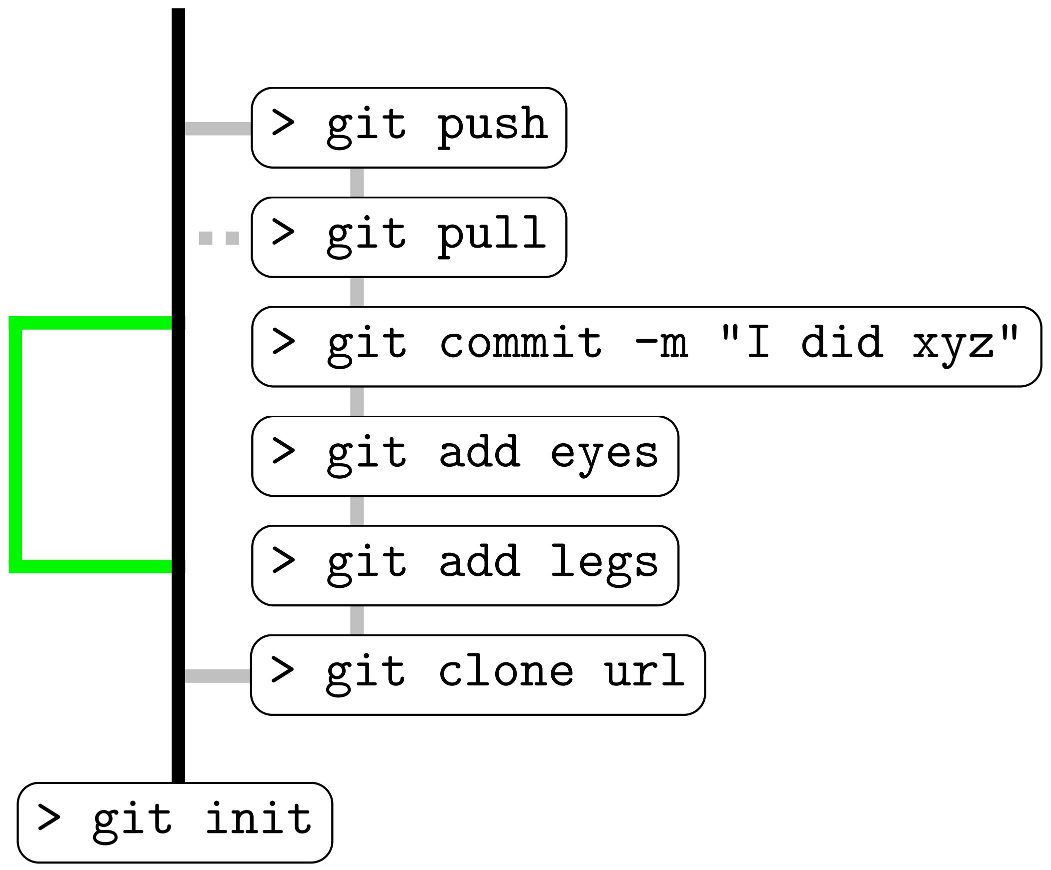
But your everyday workflow is pretty simple:



Large repositories can get pretty complicated.

But your everyday workflow is pretty simple:

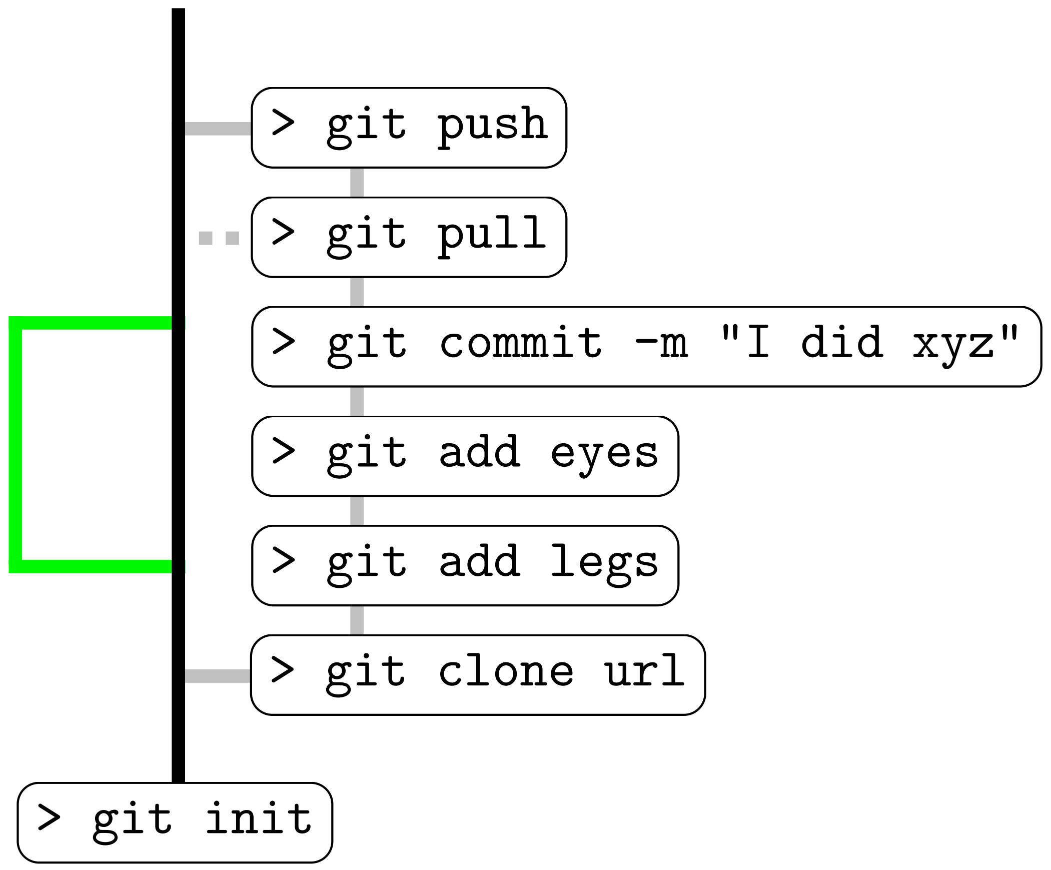
1. Make some changes (add or change files).



Large repositories can get pretty complicated.

But your everyday workflow is pretty simple:

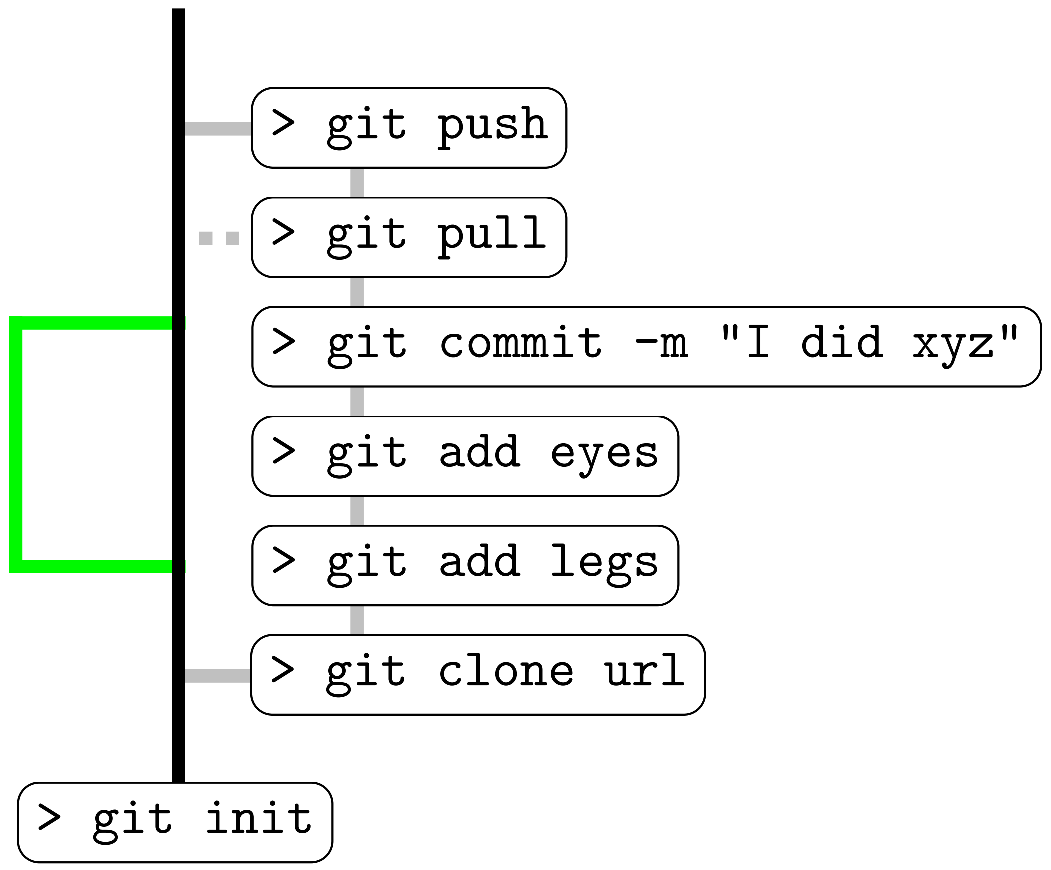
1. Make some changes (add or change files).
2. Use *git add .* to include all your changes.



Large repositories can get pretty complicated.

But your everyday workflow is pretty simple:

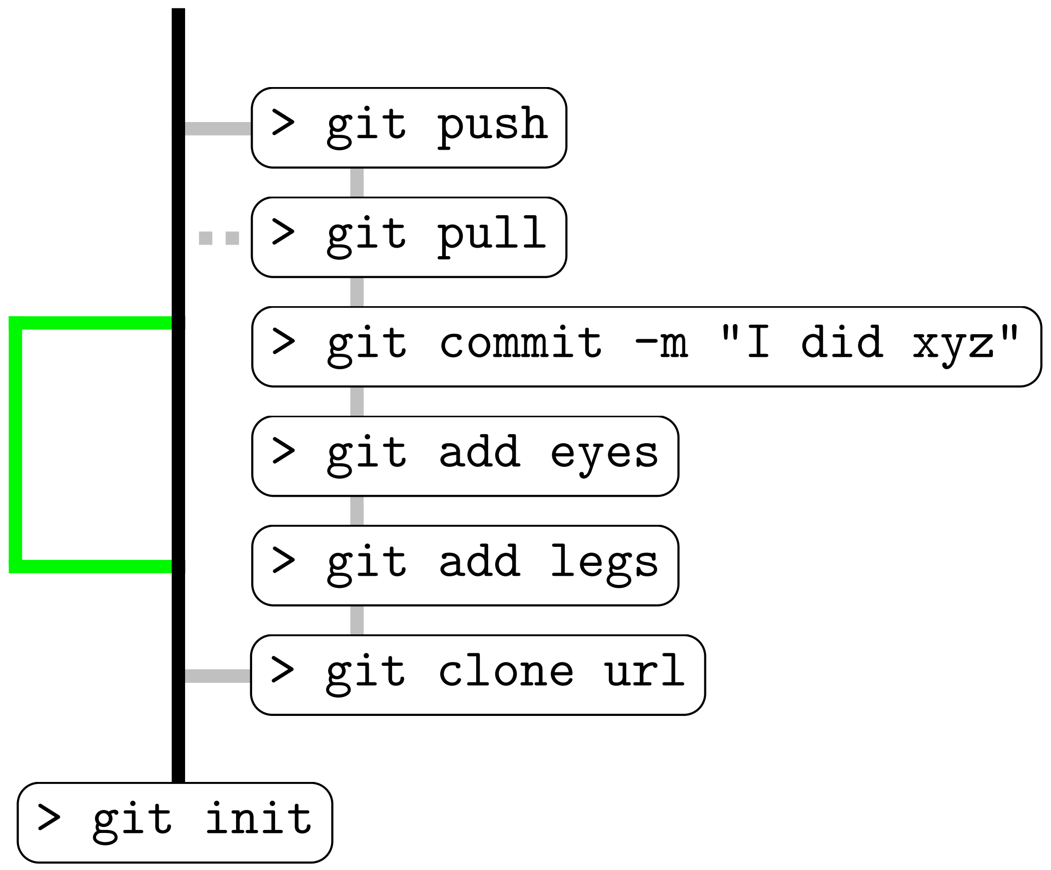
1. Make some changes (add or change files).
2. Use *git add .* to include all your changes.
3. Use *git commit -m “message”* to fix the changes.



Large repositories can get pretty complicated.

But your everyday workflow is pretty simple:

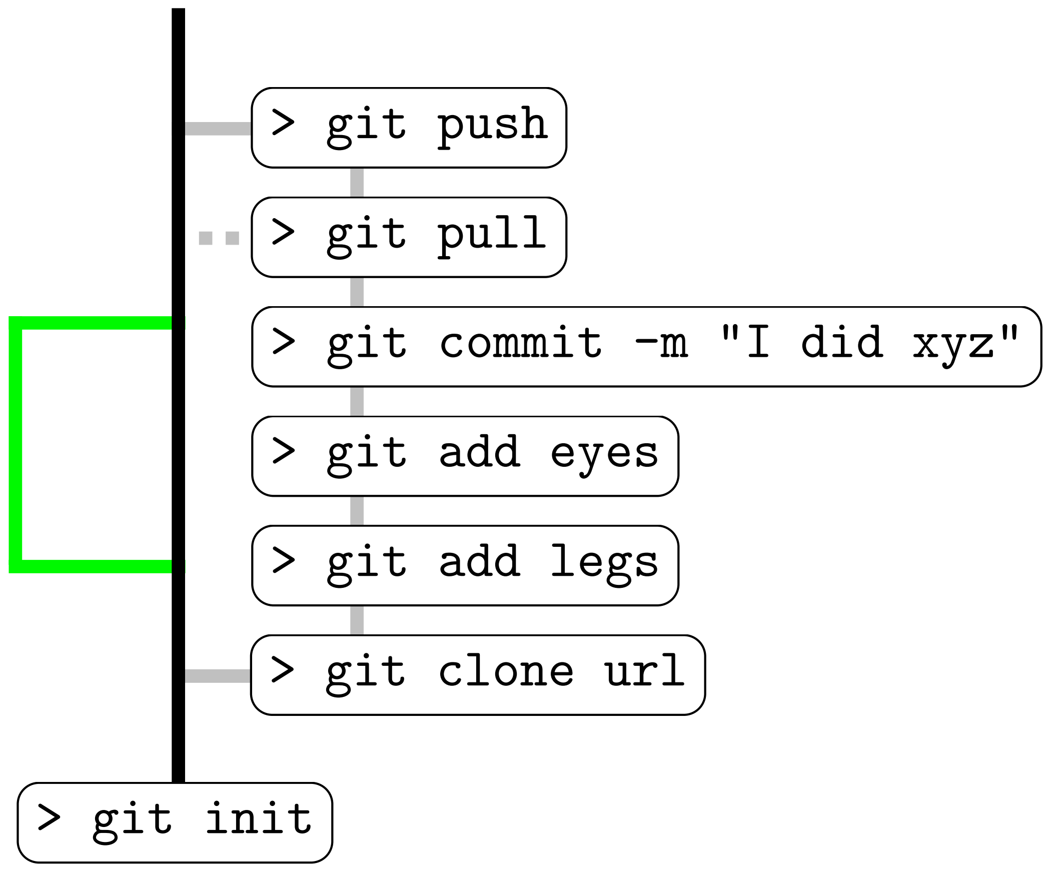
1. Make some changes (add or change files).
2. Use *git add .* to include all your changes.
3. Use *git commit -m "message"* to fix the changes.
4. Use *git pull* to sync local and remote repositories.



Large repositories can get pretty complicated.

But your everyday workflow is pretty simple:

1. Make some changes (add or change files).
2. Use *git add .* to include all your changes.
3. Use *git commit -m "message"* to fix the changes.
4. Use *git pull* to sync local and remote repositories.
5. Use *git push* to send your local changes to the remote.

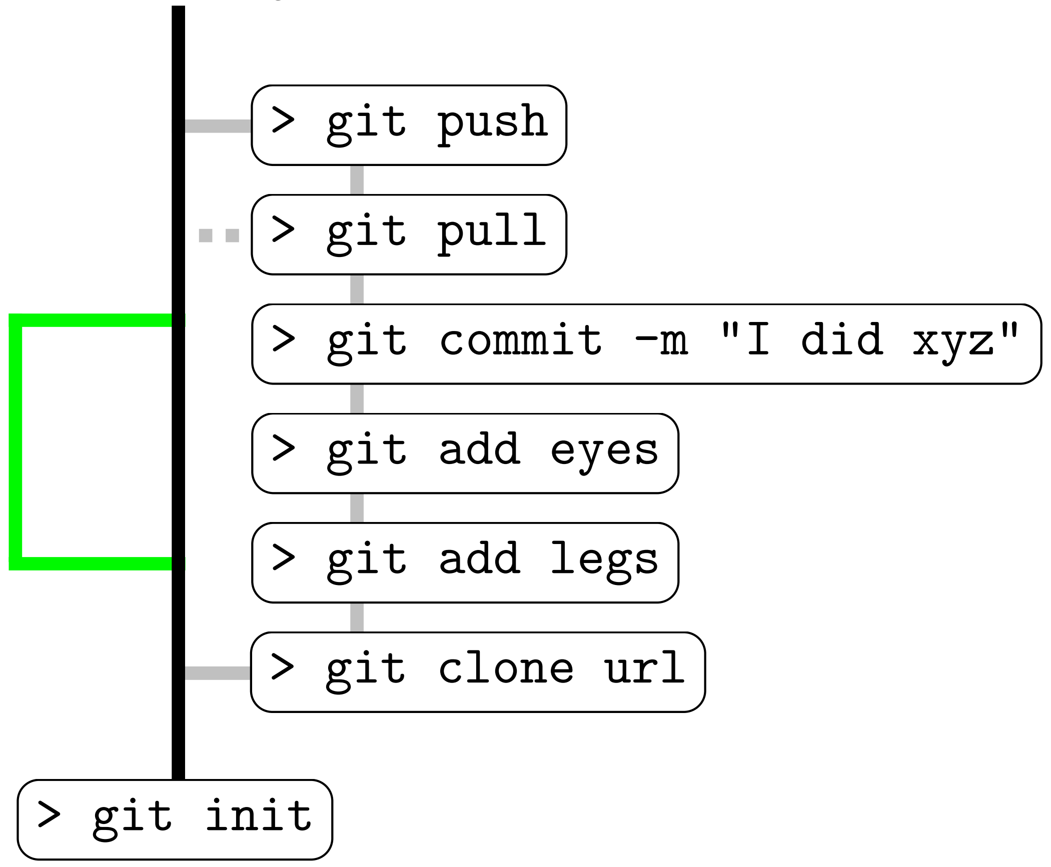


Large repositories can get pretty complicated.

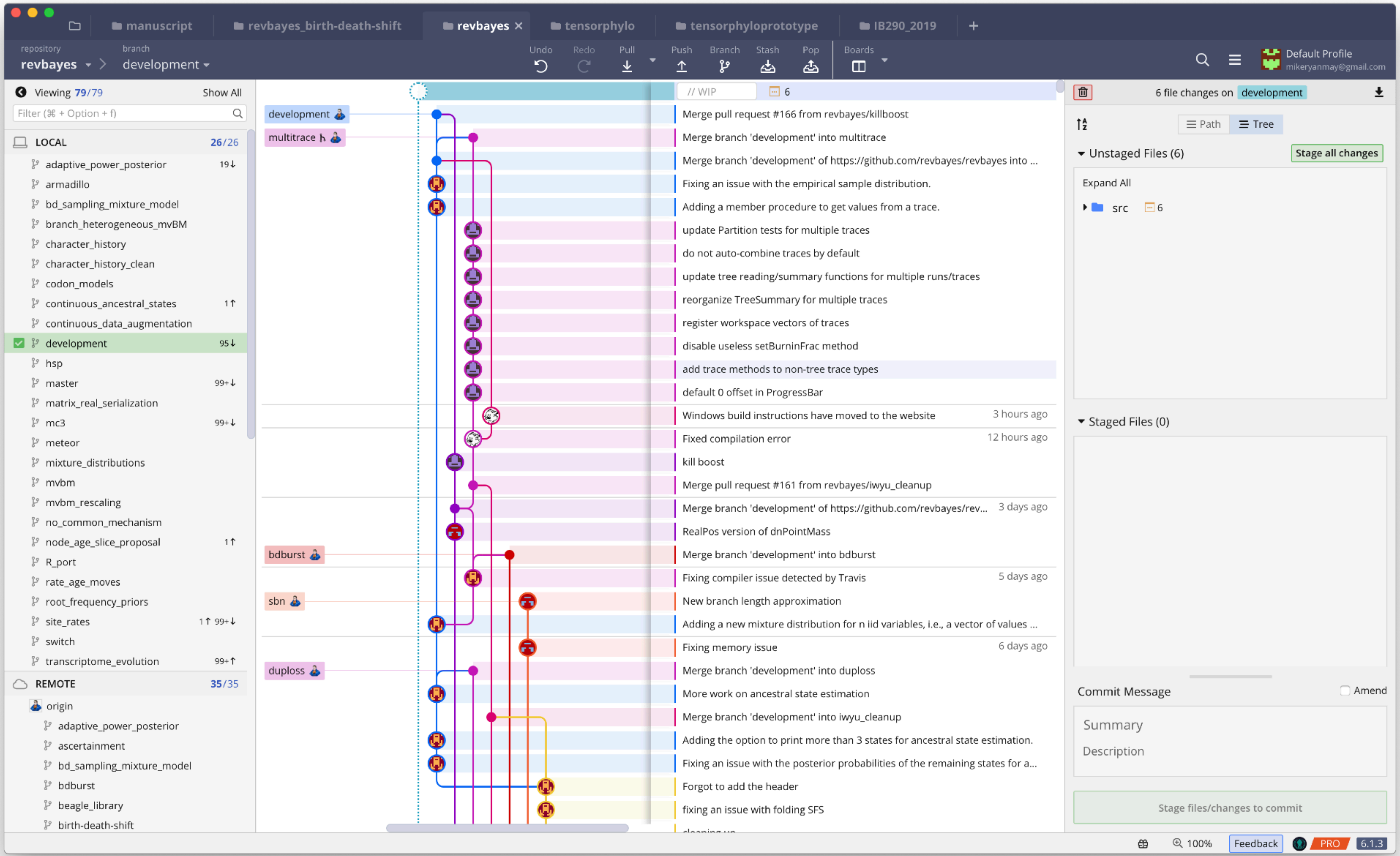
But your everyday workflow is pretty simple:

1. Make some changes (add or change files).
2. Use *git add .* to include all your changes.
3. Use *git commit -m “message”* to fix the changes.
4. Use *git pull* to sync local and remote repositories.
5. Use *git push* to send your local changes to the remote.

Commit early and often! Don’t be shy!



Use a GUI!



repository: revbayes
branch: development

Viewing 79/79 Show All

Filter (⌘ + Option + f)

LOCAL 26/26

adaptive_power_posterior	19↓
armadillo	
bd_sampling_mixture_model	
branch_heterogeneous_mvBM	
character_history	
character_history_clean	
codon_models	
continuous_ancestral_states	1↑
continuous_data_augmentation	
development	95↓
hsp	
master	99+↓
matrix_real_serialization	
mc3	99+↓
meteor	
mixture_distributions	
mvbm	
mvbm_rescaling	
no_common_mechanism	
node_age_slice_proposal	1↑
R_port	
rate_age_moves	
root_frequency_priors	
site_rates	1↑ 99+↓
switch	
transcriptome_evolution	99+↑

REMOTE 35/35

origin
- adaptive_power_posterior
- ascertainment
- bd_sampling_mixture_model
- bdburst
- beagle_library
- birth-death-shift

```
// WIP                                                              6
development   Merge pull request #166 from revbayes/killboost
multitrace    Merge branch 'development' into multitrace
              Merge branch 'development' of https://github.com/revbayes/revbayes into ...
              Fixing an issue with the empirical sample distribution.
              Adding a member procedure to get values from a trace.
              update Partition tests for multiple traces
              do not auto-combine traces by default
              update tree reading/summary functions for multiple runs/traces
              reorganize TreeSummary for multiple traces
              register workspace vectors of traces
              disable useless setBurninFrac method
              add trace methods to non-tree trace types
              default 0 offset in ProgressBar
              Windows build instructions have moved to the website        3 hours ago
              Fixed compilation error                                      12 hours ago
              kill boost
              Merge pull request #161 from revbayes/iwyu_cleanup
              Merge branch 'development' of https://github.com/revbayes/rev...  3 days ago
              RealPos version of dnPointMass
bdburst       Merge branch 'development' into bdburst
              Fixing compiler issue detected by Travis                     5 days ago
sbn           New branch length approximation
              Adding a new mixture distribution for n iid variables, i.e., a vector of values ...
              Fixing memory issue                                          6 days ago
duploss       Merge branch 'development' into duploss
              More work on ancestral state estimation
              Merge branch 'development' into iwyu_cleanup
              Adding the option to print more than 3 states for ancestral state estimation.
              Fixing an issue with the posterior probabilities of the remaining states for a...
              Forgot to add the header
              fixing an issue with folding SFS
```

6 file changes on development

Path Tree

Unstaged Files (6) Stage all changes

Expand All

src 6

Staged Files (0)

Commit Message Amend

Summary

Description

Stage files/changes to commit

100% Feedback PRO 6.1.3