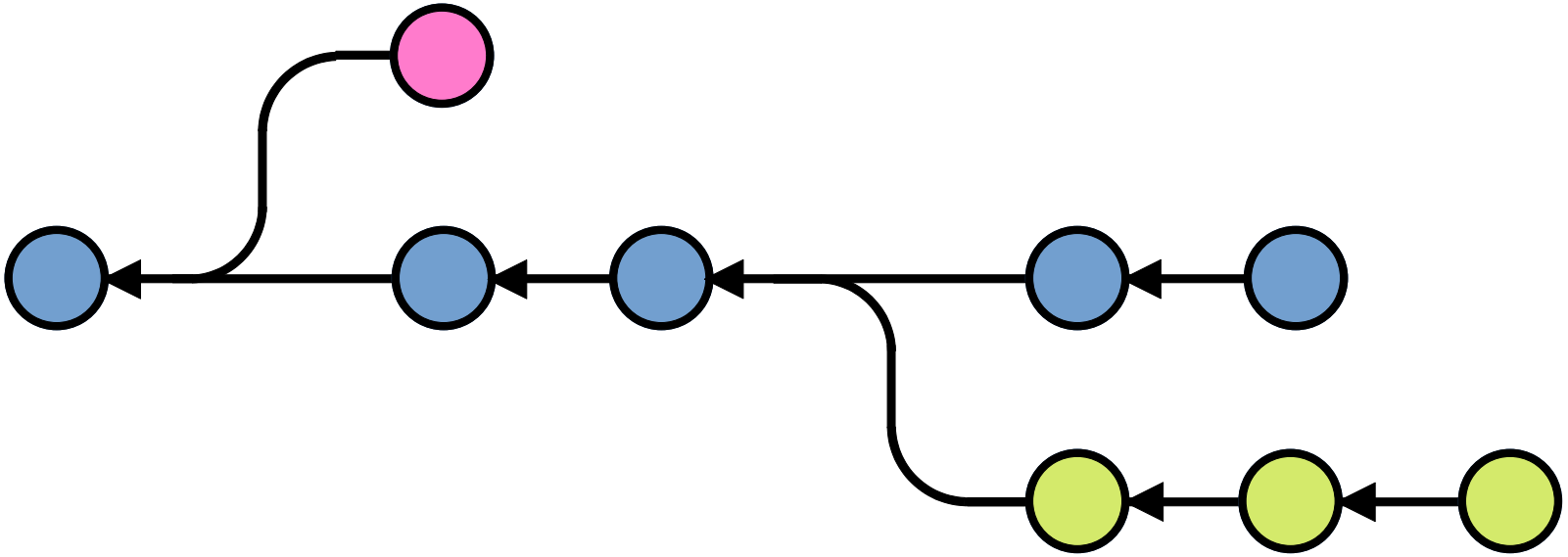
FLATIRON INSTITUTE
Center for Computational Quantum Physics

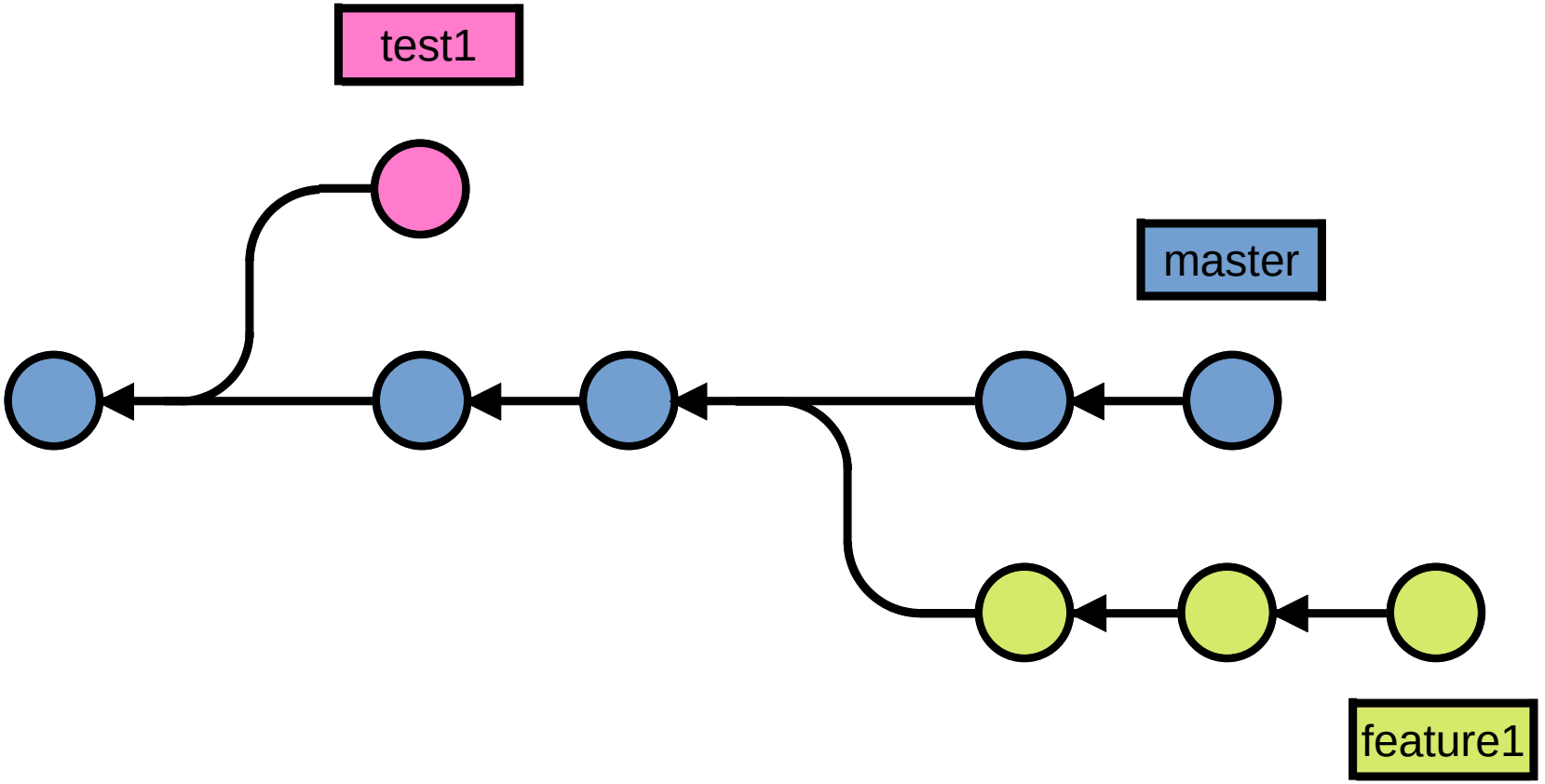
git basics

Maxime Charlebois
CCQ, Flatiron Institute

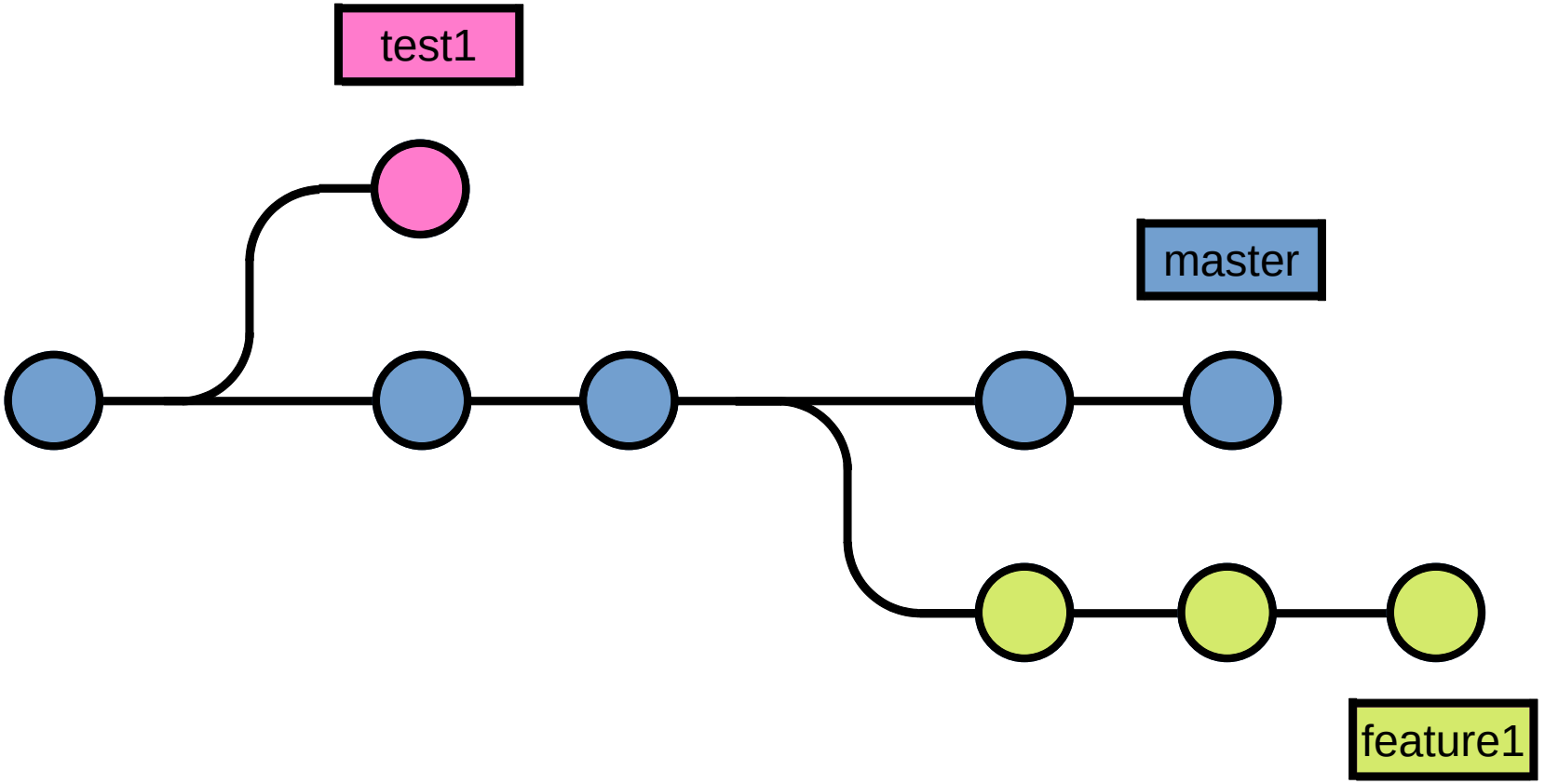
git



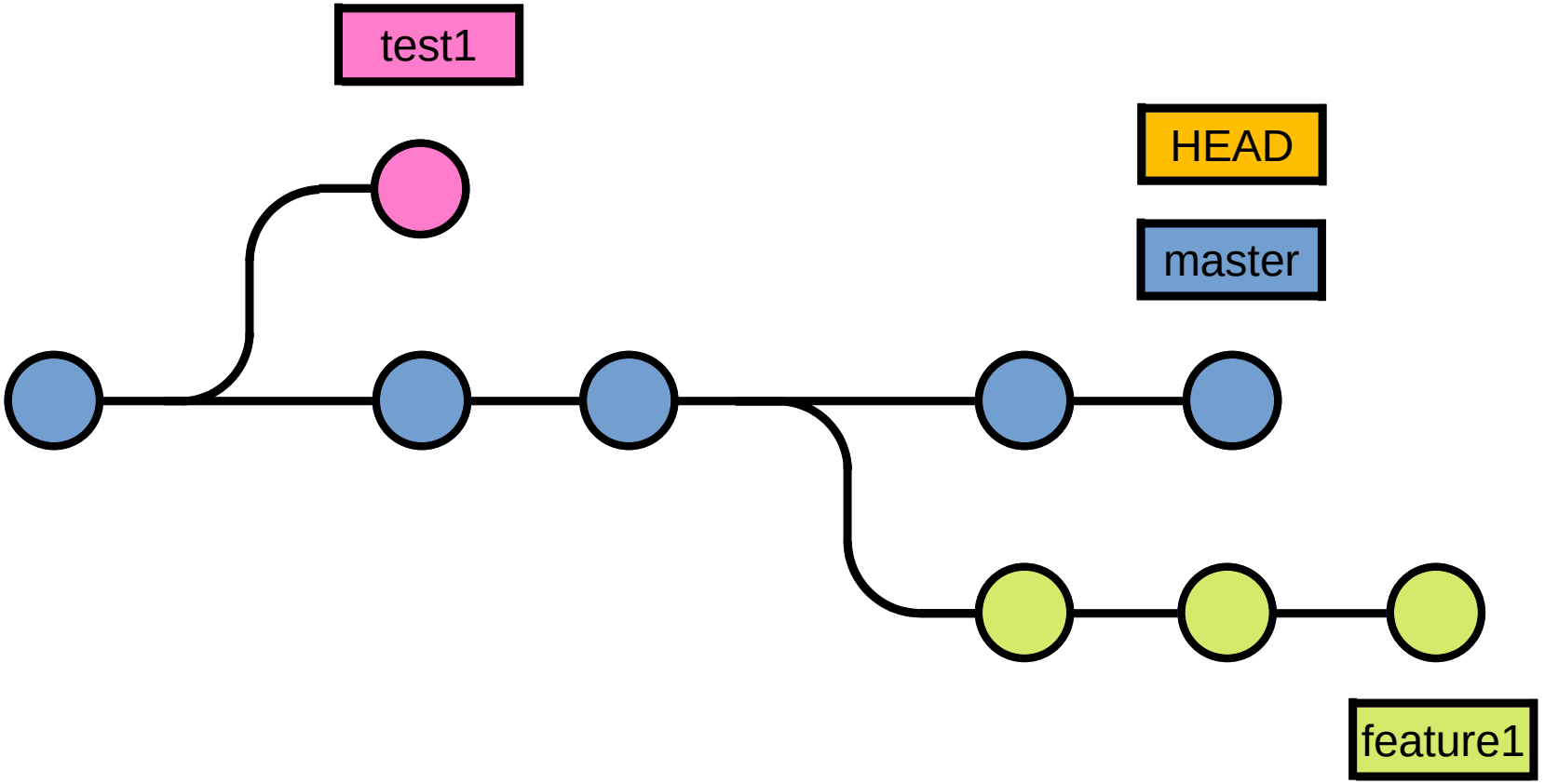
git



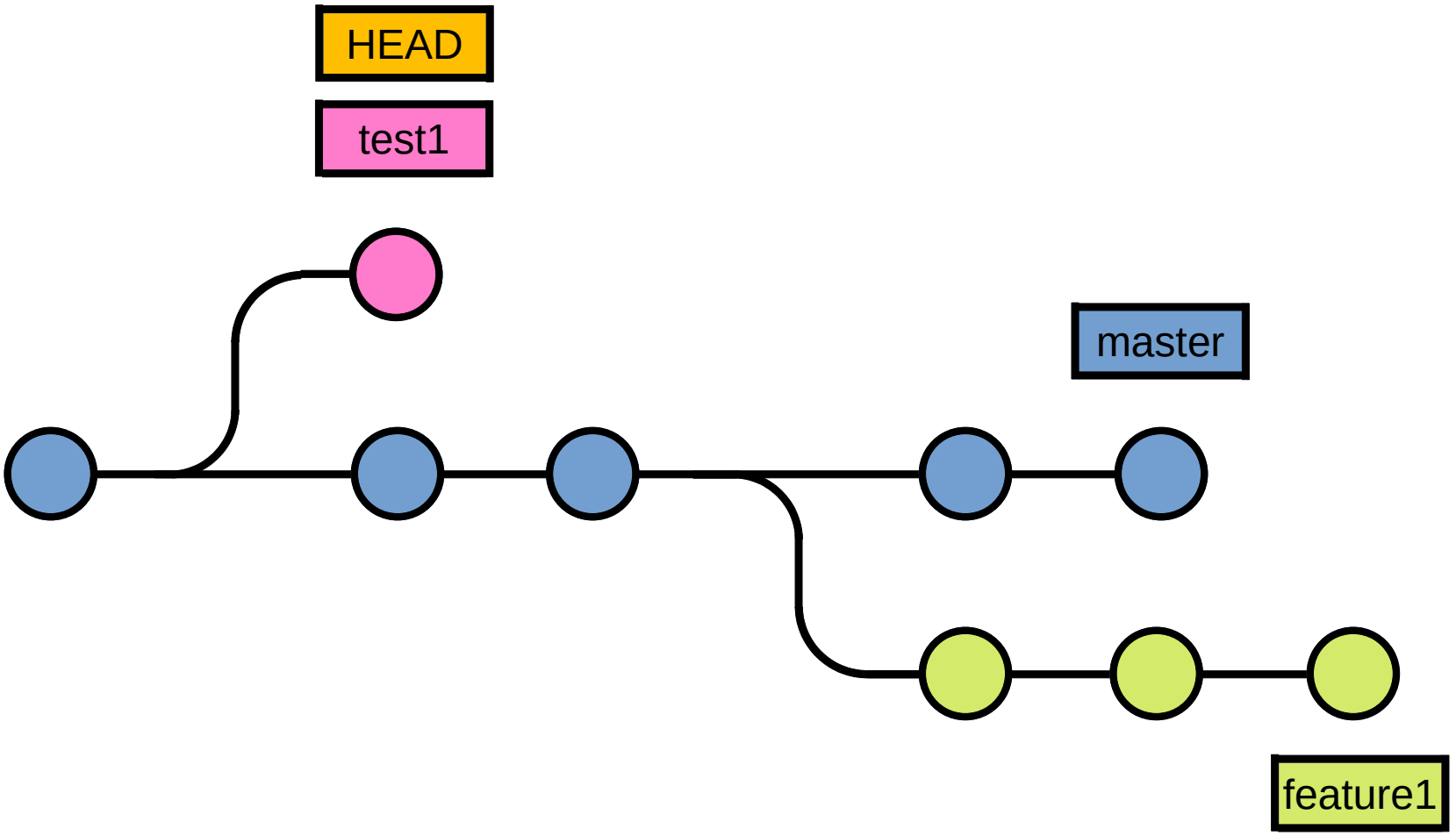
git



Navigation

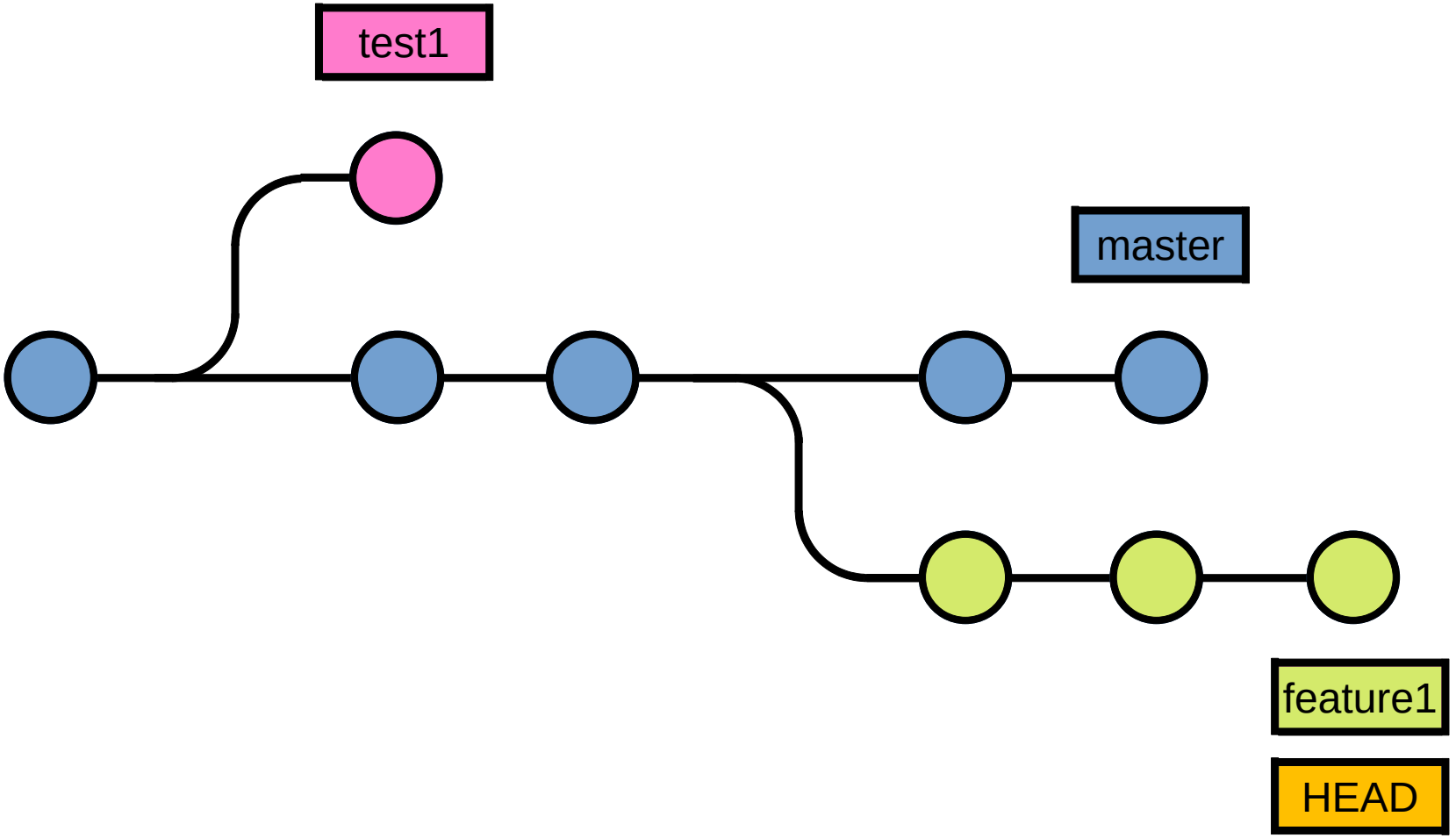

test1
HEAD
master
feature1

git checkout



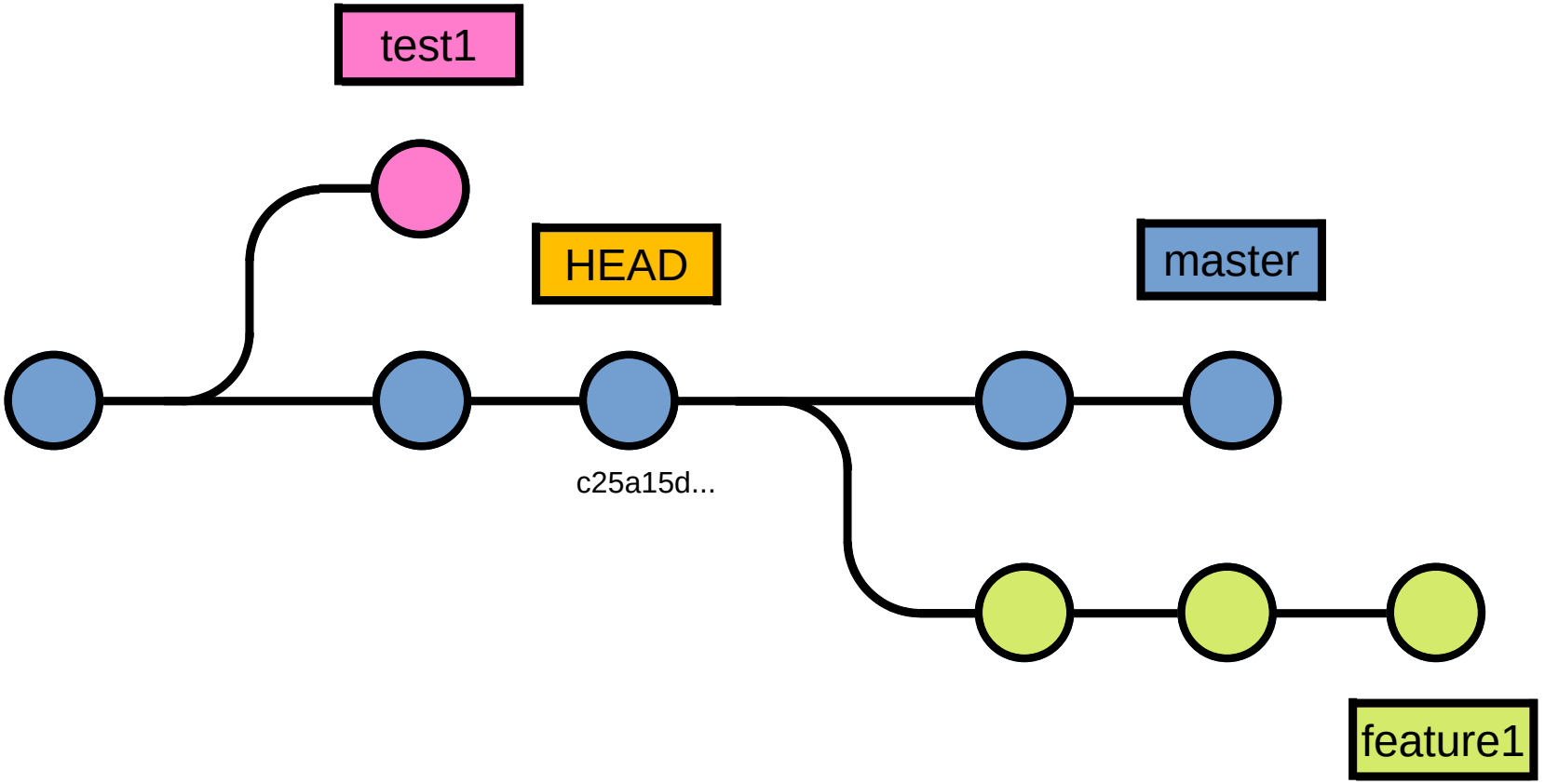

```
$ git checkout test1
```

git checkout



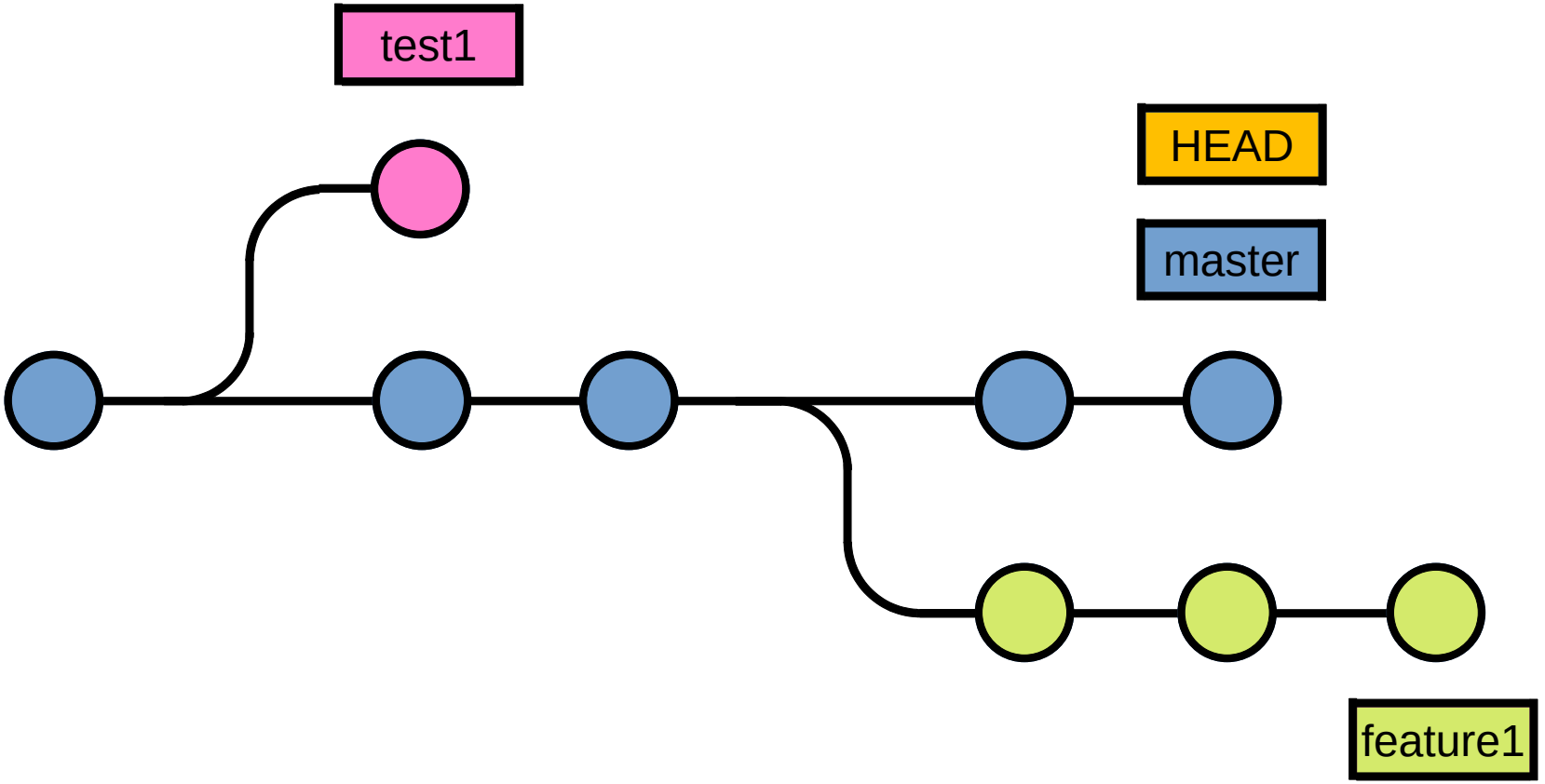

```
$ git checkout feature1
```

git checkout



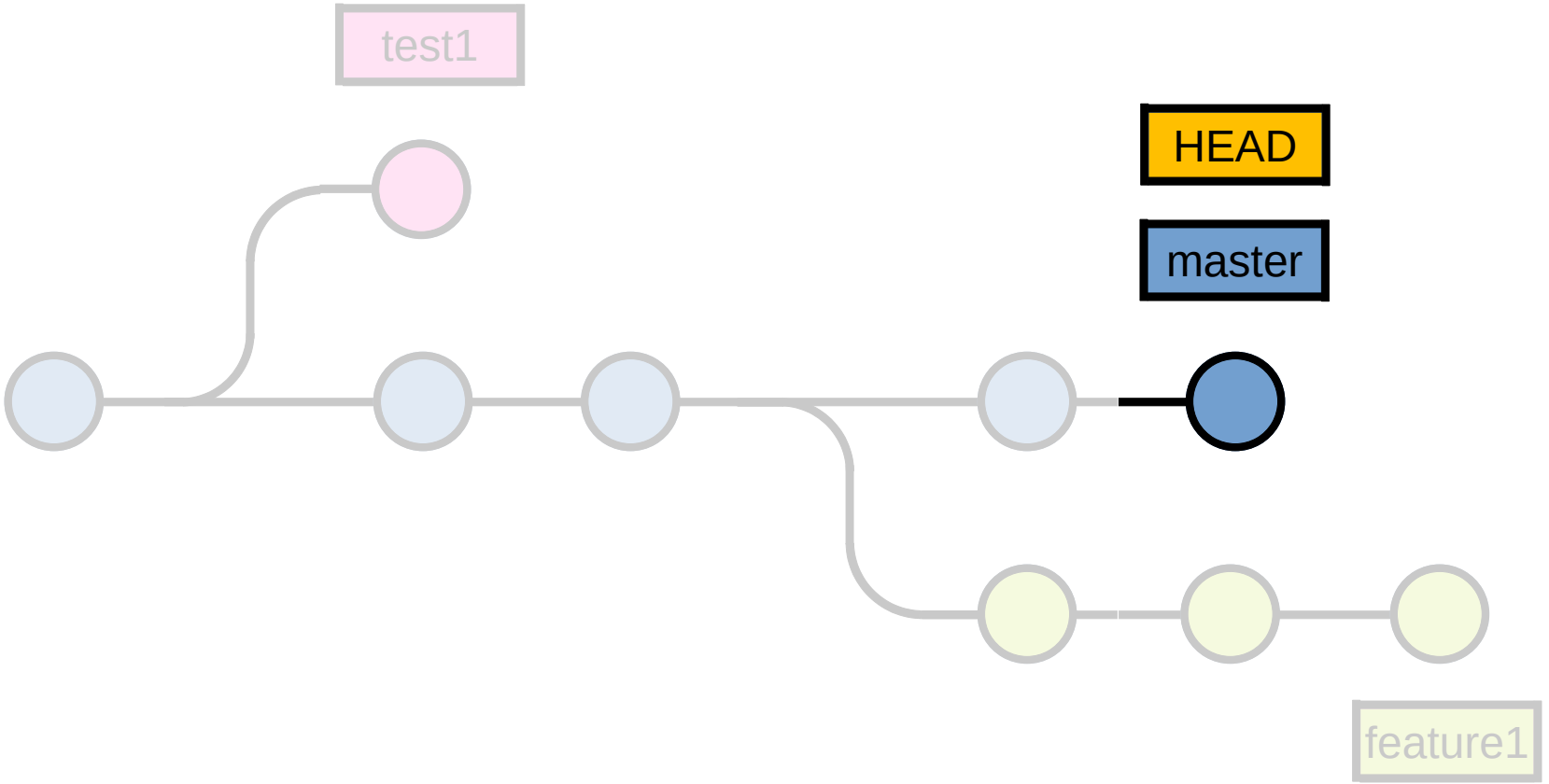

```
$ git checkout c25a15d
```

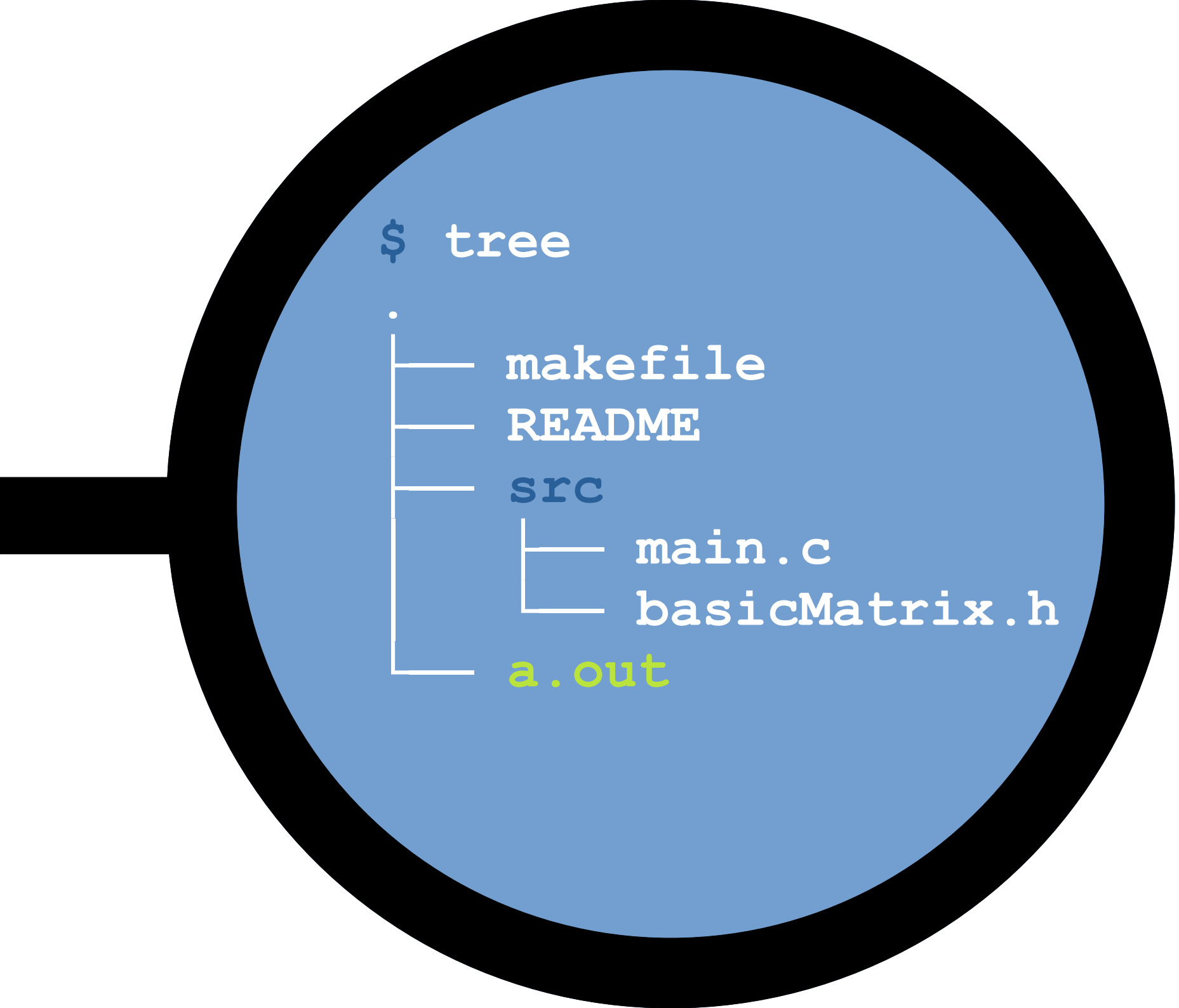
git checkout



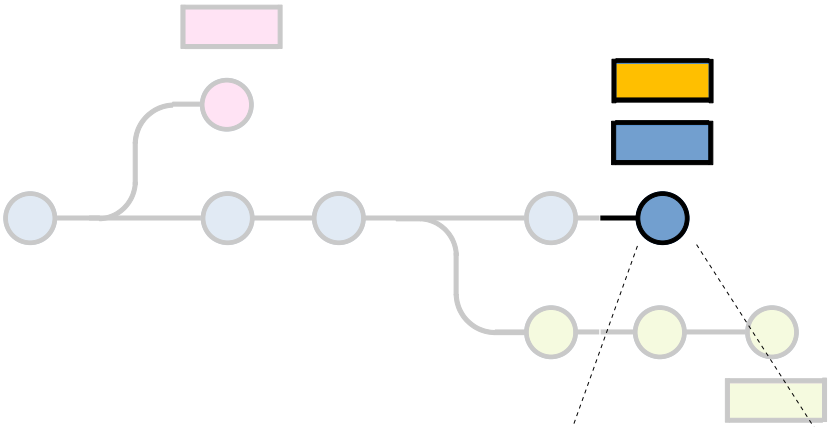

```
$ git checkout master
```

Editing


test1
HEAD
master
feature1


$ tree
.
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out

git status



work directory

```
$ tree
.
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

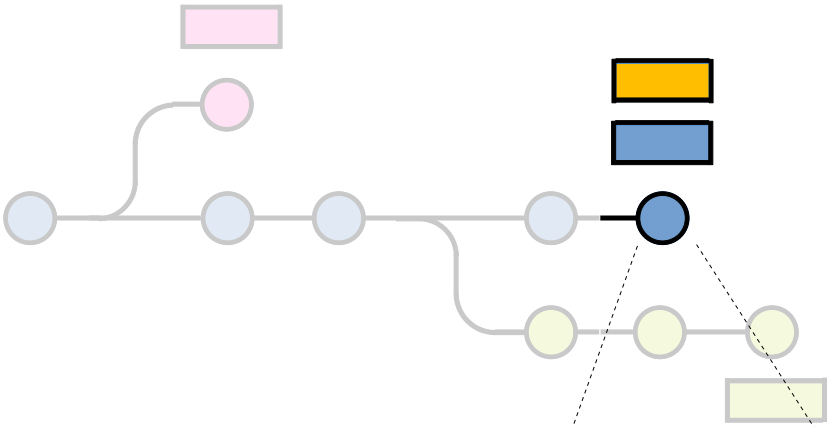
staging area

```
$ git status

On branch master
nothing to commit,
working directory clean
```

Specific commit

```
.
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

git status



work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

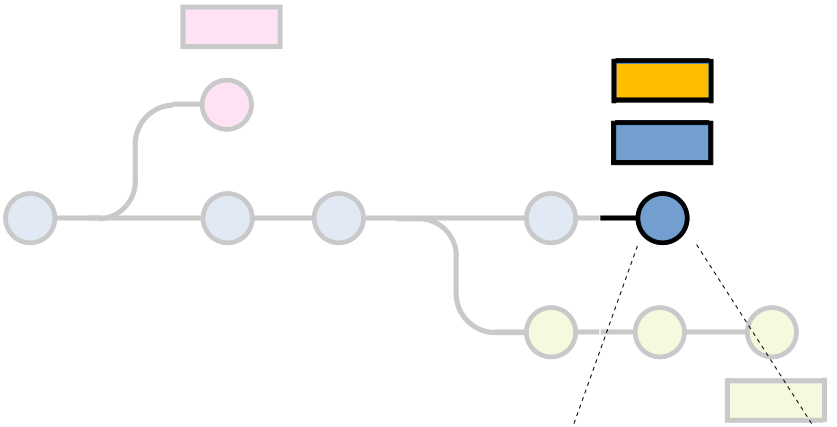
staging area

```
$ git status

On branch master
Untracked files:


    LICENSE
```

Specific commit

```
.
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

git add



```
$ git add LICENSE
```

work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

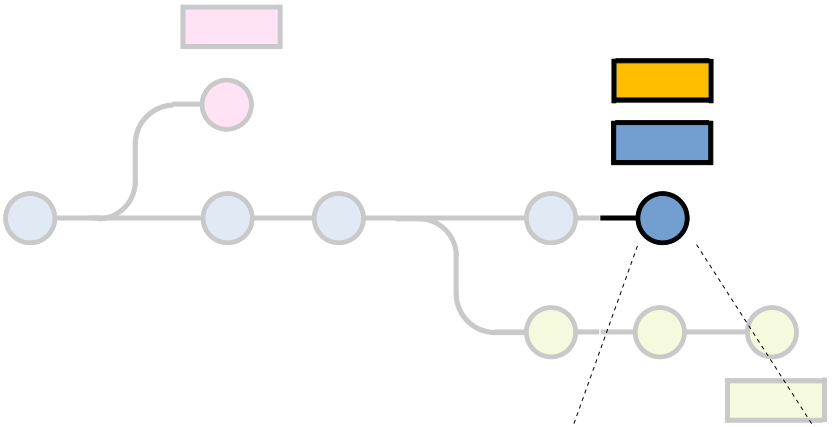
staging area

```
$ git status

On branch master
Changes to be committed:


    new file:    LICENSE
```

Specific commit

```
.
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

git rm



```
$ git rm a.out
```

work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

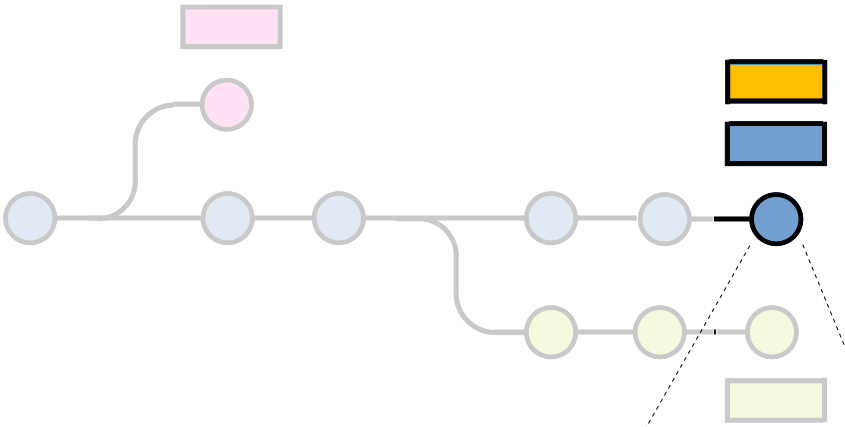
staging area

```
$ git status

On branch master
Changes to be committed:

    deleted:    a.out
    new file:   LICENSE
```

Specific commit

```
.
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
└── a.out
```

git commit



$ git commit

work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

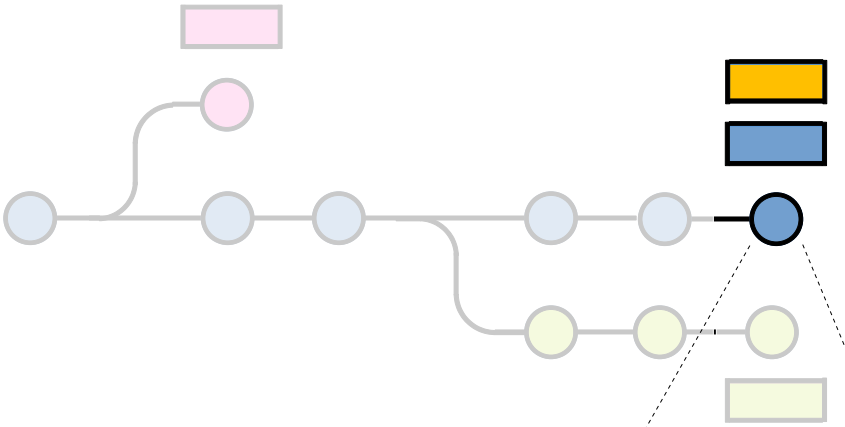
staging area

```
$ git status

On branch master
Changes to be committed:

   deleted:    a.out
   new file:   LICENSE
```

Specific commit

```
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

git commit



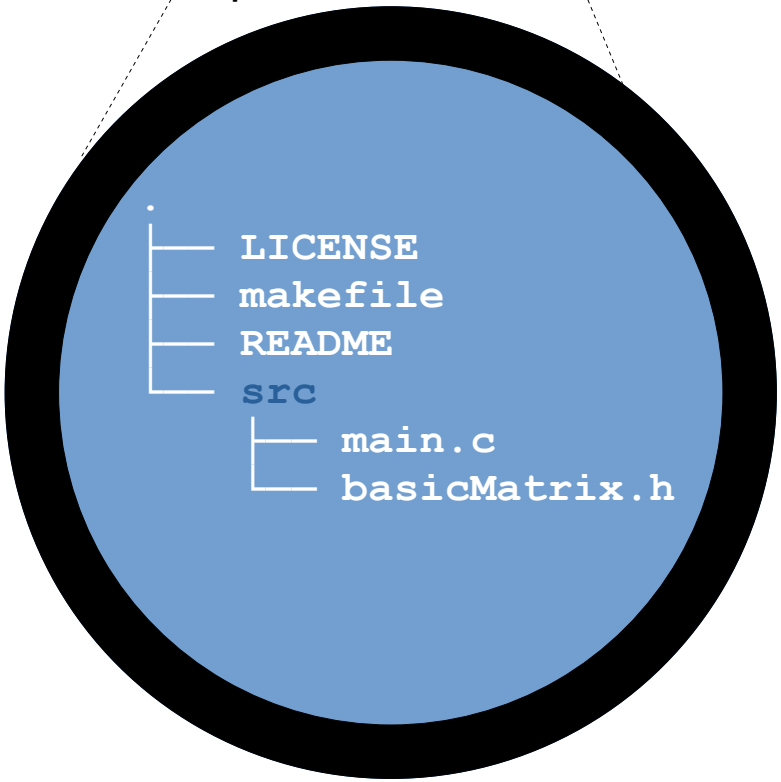
work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

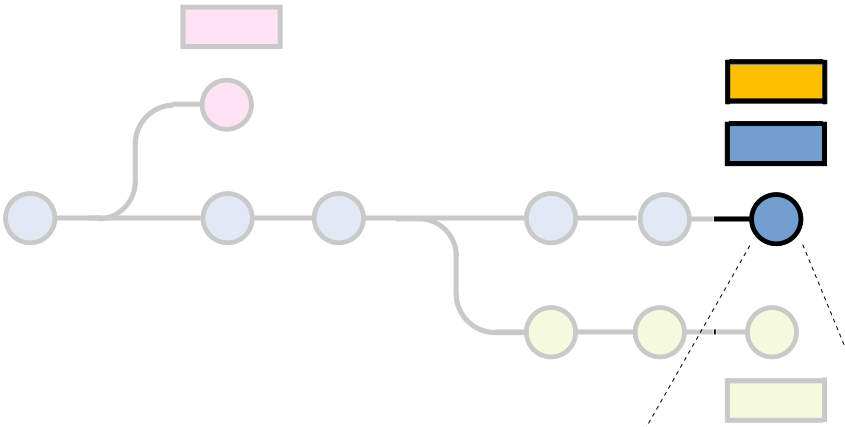
staging area

```
$ git status

On branch master
nothing to commit,
working directory clean
```

Specific commit



.gitignore



work directory

```
$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

staging area

```
$ git status

On branch master
Untracked files:


    a.out
```

Specific commit

```
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

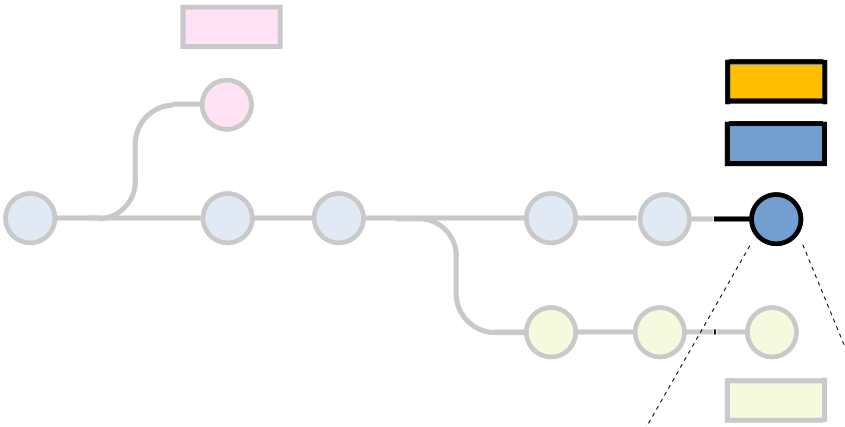
.gitignore

work directory

```
$ ls -a
.
..
.git
LICENSE
makefile
README
src
a.out


$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

staging area

```
$ git status

On branch master
Untracked files:


    a.out
```



Specific commit

```
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

.gitignore

work directory

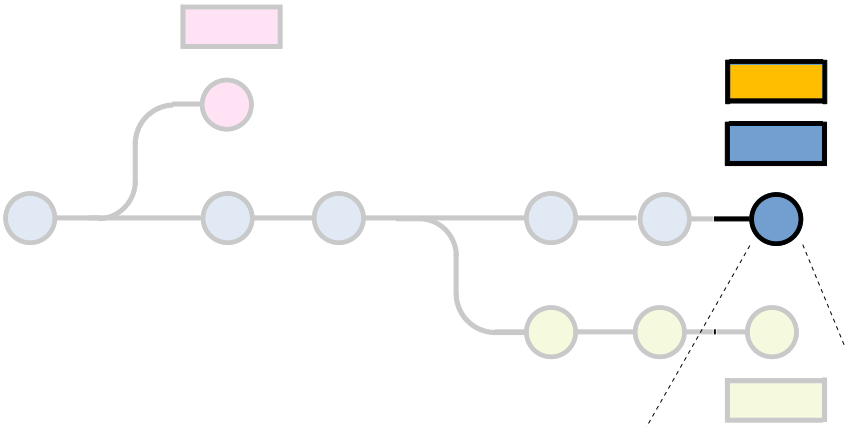
```
$ ls -a
.
..
.git
.gitignore
LICENSE
makefile
README
src
a.out

$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

a.out

$ git add .gitignore

staging area

```
$ git status

On branch master
Untracked files:


    .gitignore
```



Specific commit

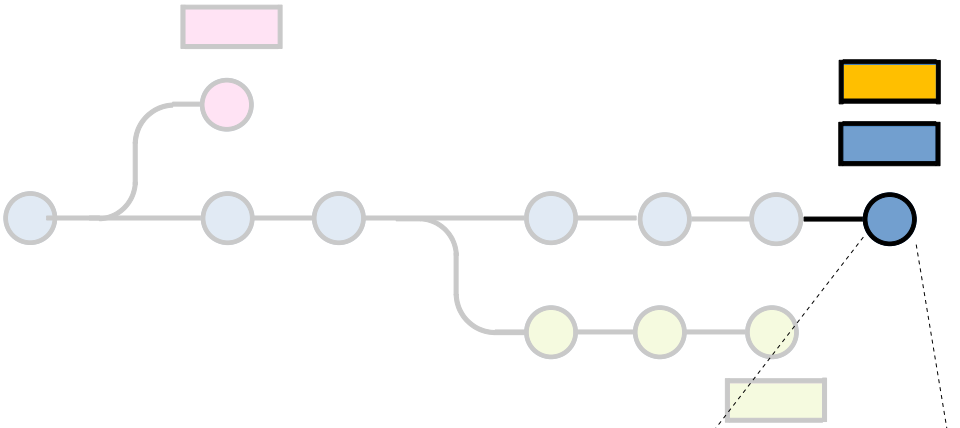
```
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

.gitignore

work directory

```
$ ls -a
.
..
.git
.gitignore
LICENSE
makefile
README
src
a.out

$ tree
.
├── LICENSE
├── makefile
├── README
├── src
│   ├── main.c
│   └── basicMatrix.h
└── a.out
```

a.out



$ git commit

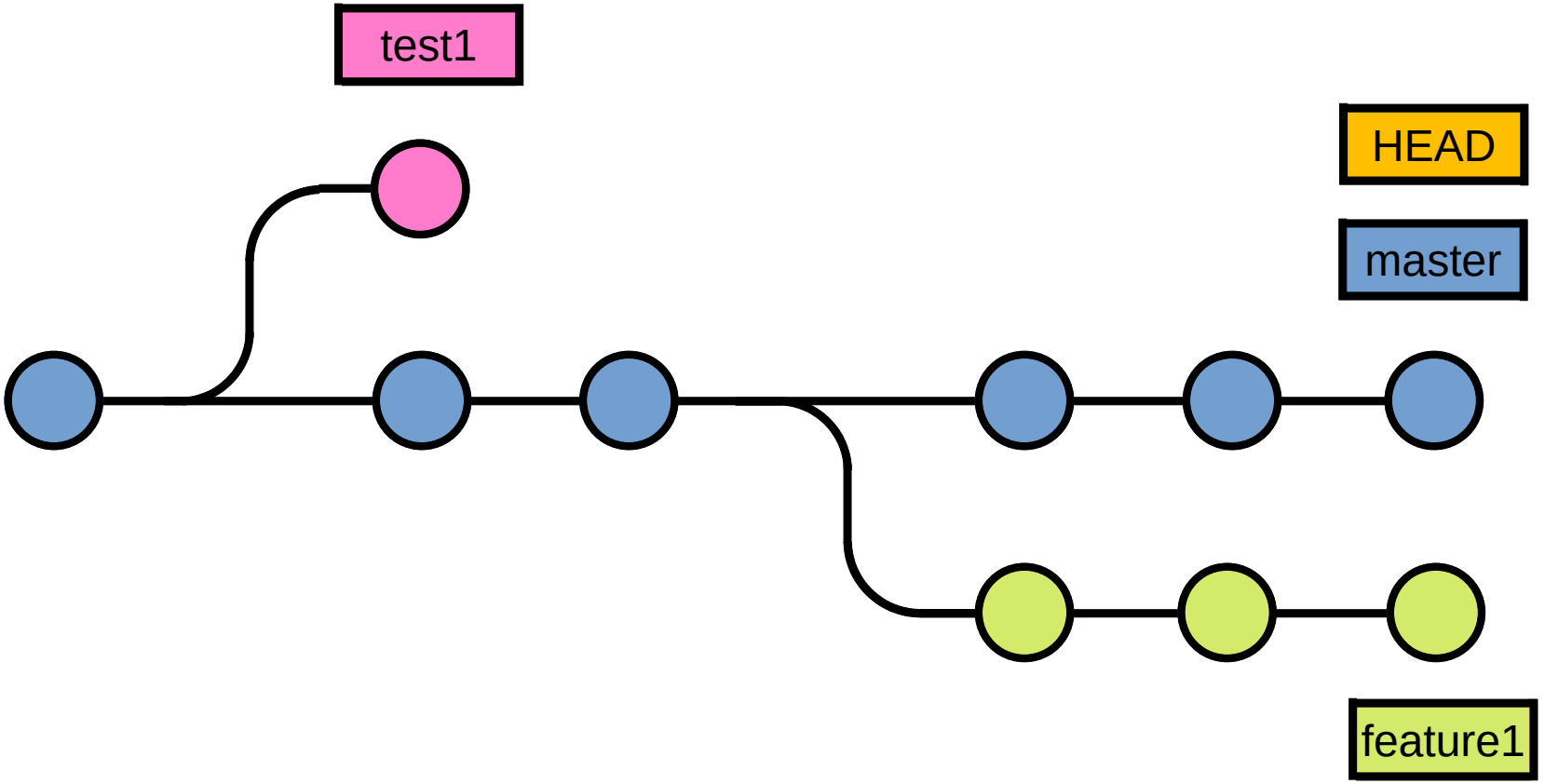
staging area

```
$ git status

On branch master
nothing to commit,
working directory clean
```

Specific commit

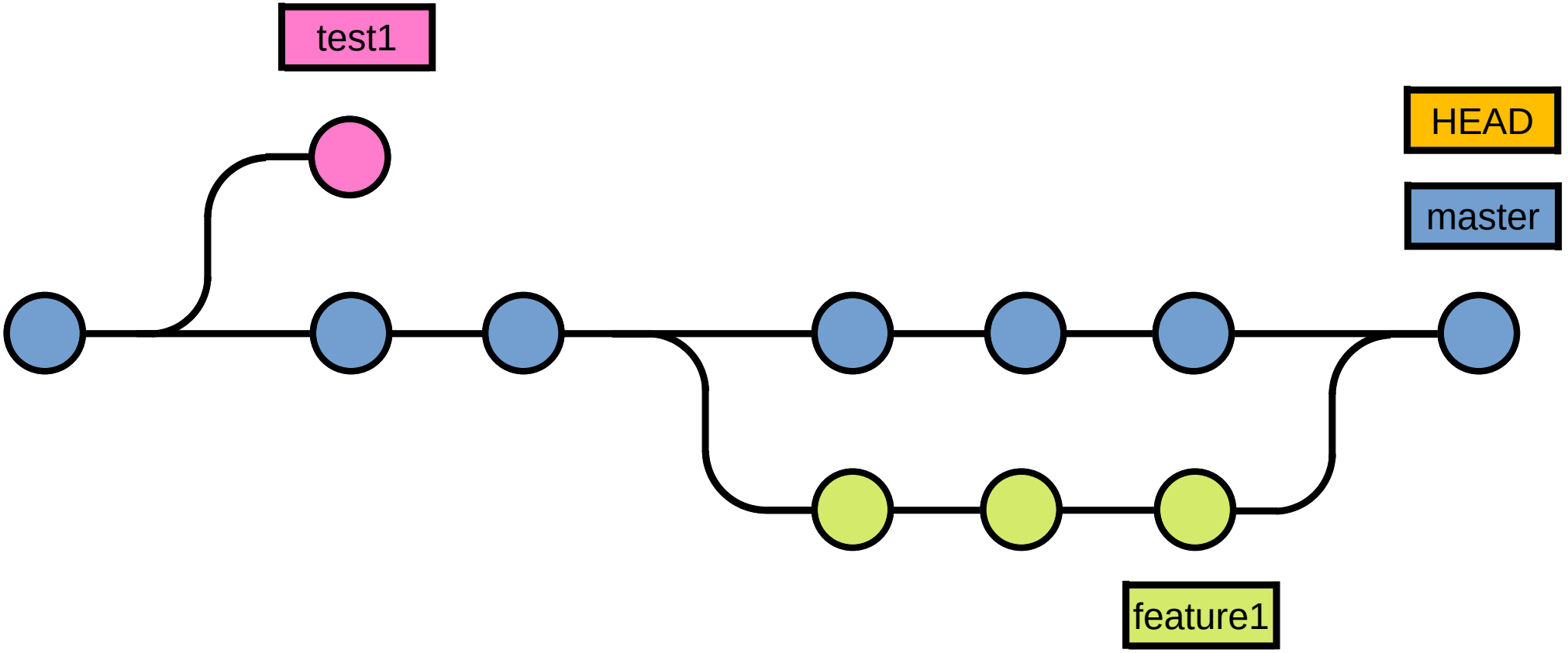
```
.
├── LICENSE
├── makefile
├── README
└── src
    ├── main.c
    └── basicMatrix.h
```

git merge




```
$
```

git merge




```
$ git merge feature1
```

Hands on

Hands-on
https://bit.ly/343MXTt

```
$ git status

fatal: Not a git repository (or any
of the parent directories): .git
```

Hands-on

```
[user]
email = mcharlebois@flatironinstitute.org
name = Maxime Charlebois
```

```
$ git config --global --edit
```

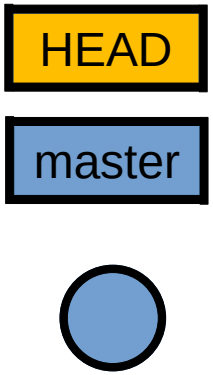
Hands-on


```
$ git init
$ git status
```

Hands-on

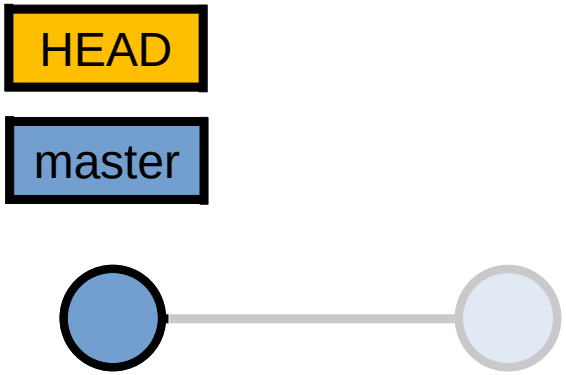

```
$ date >> file1
$ git status
$ git add file1
$ git status
```

Hands-on



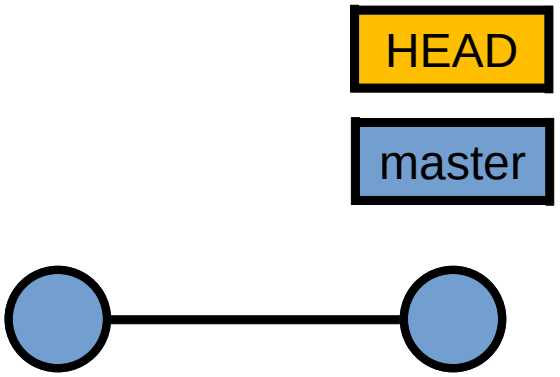

```
$ git commit -m "first commit"
$ git status
```

Hands-on



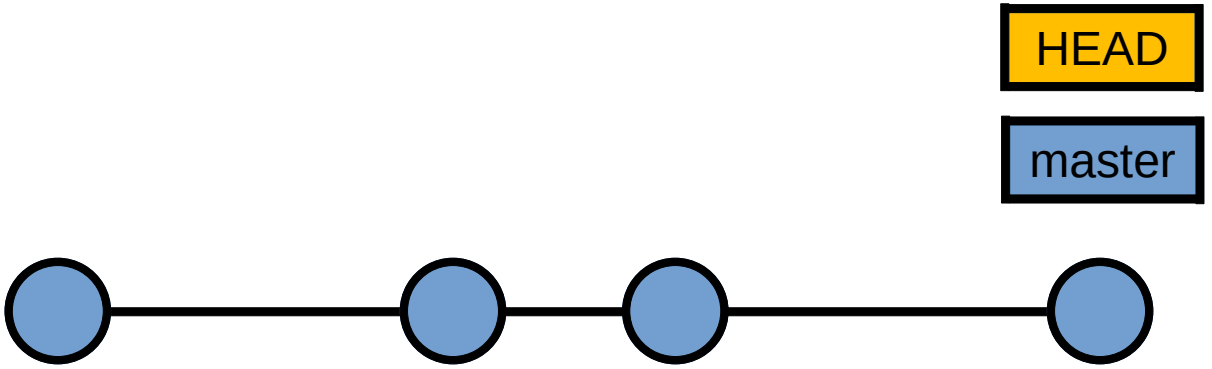

```
$ date >> file1
$ cat file1
$ git status
$ git diff
$ git add file1
$ git status
$ git diff
```

Hands-on



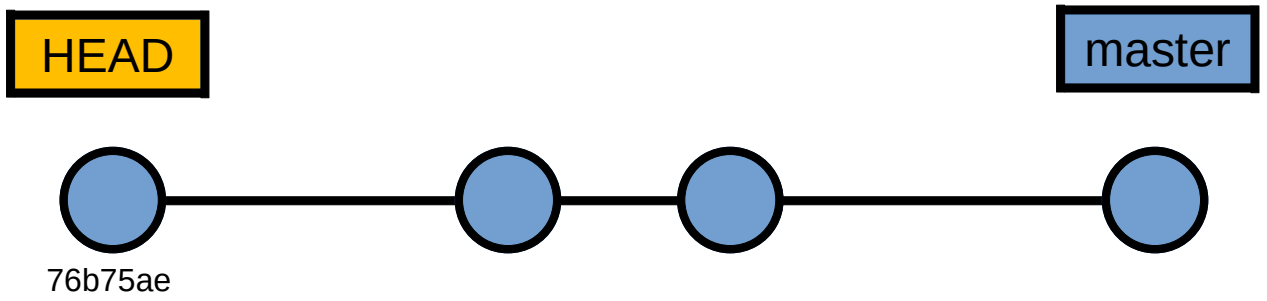

```
$ git commit -m "second commit"
```

Hands-on




```
$ date >> file2
$ git status
$ git diff
$ git add .
$ git status
$ git diff
$ git commit -m "third commit"

$ date >> file1
$ git commit -am "4th commit"
```

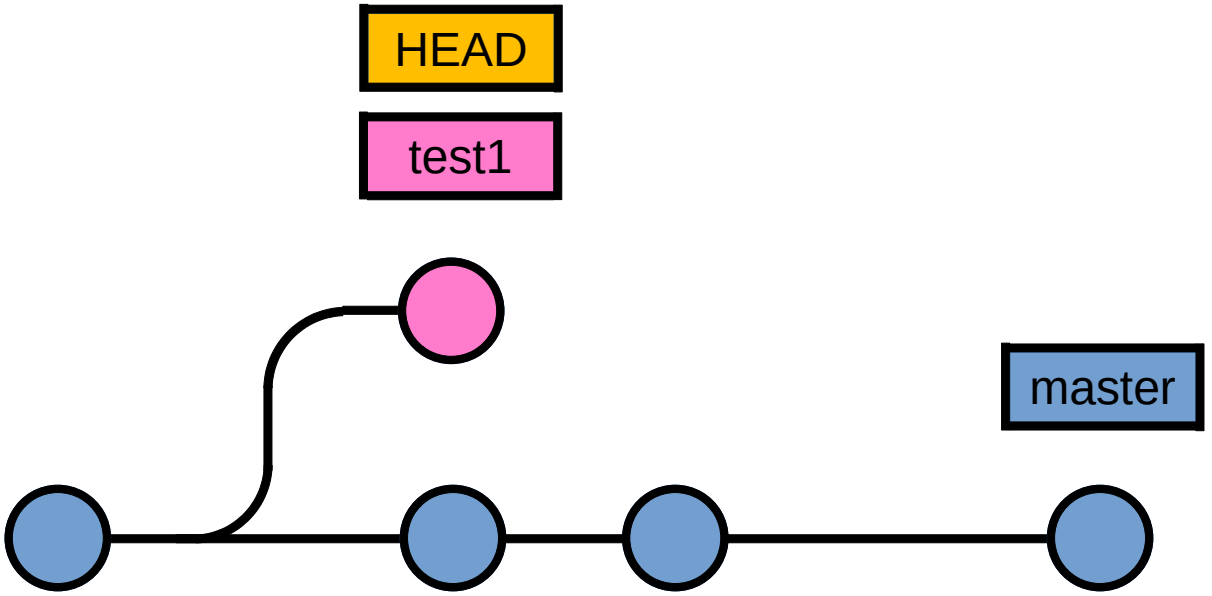
Hands-on



!! BEWARE, commit number will be different in your case !!

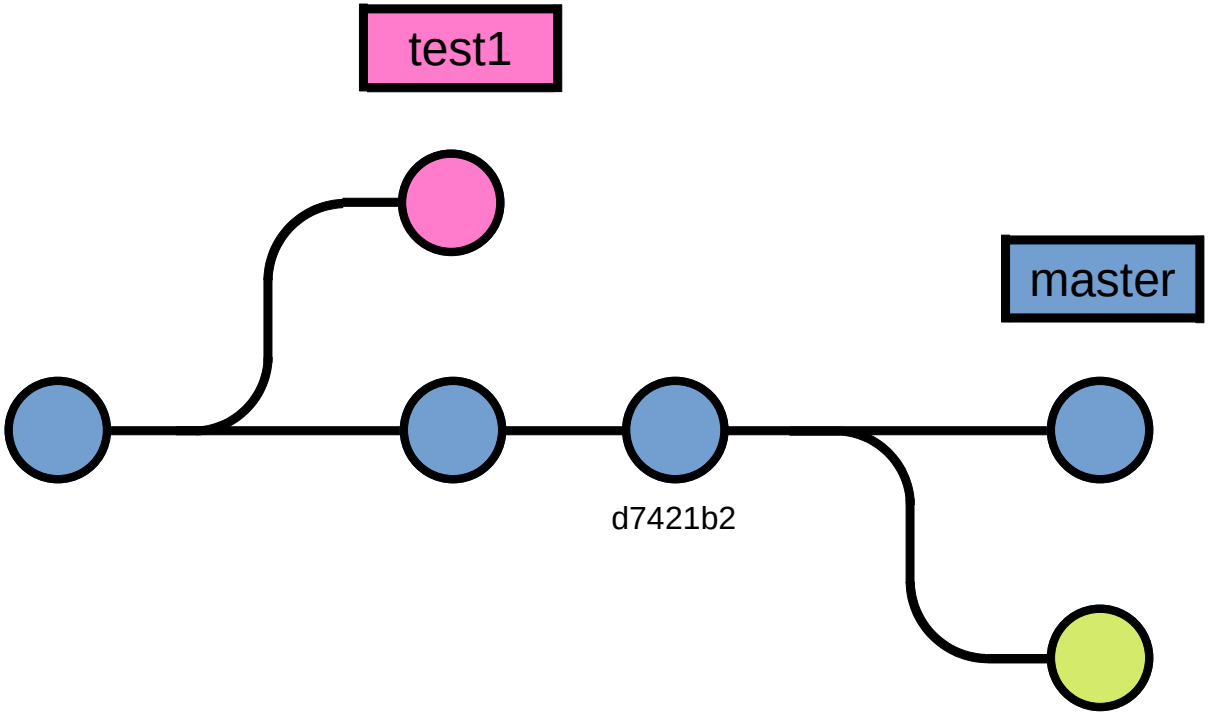
```
$ git log
$ git chekcout 76b75ae
```

Hands-on




```
$ git checkout -b test1
$ ls -a >> tmpList
$ git add tmpList
$ git commit -m "test branch"

$ git log
```

Hands-on

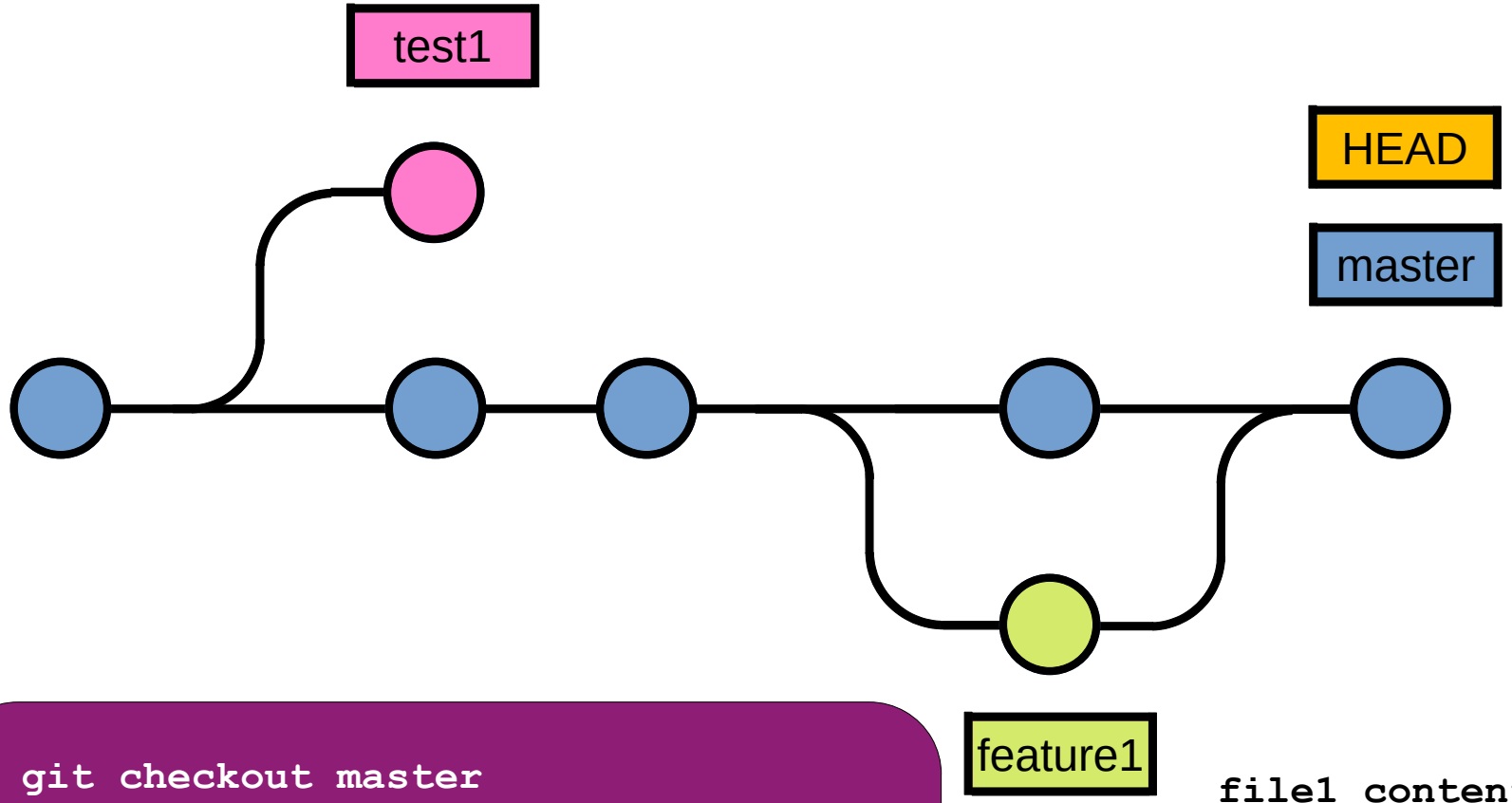



```
$ git log
$ git checkout d7421b2
$ git status
$ git checkout -b feature1
$ date >> file1
$ git add file1
$ git commit -m "creating conflict"
```

!! BEWARE, commit number will be different in your case !!

git merge

```
!!! CONFLICT in file1 !!!
please solve
```



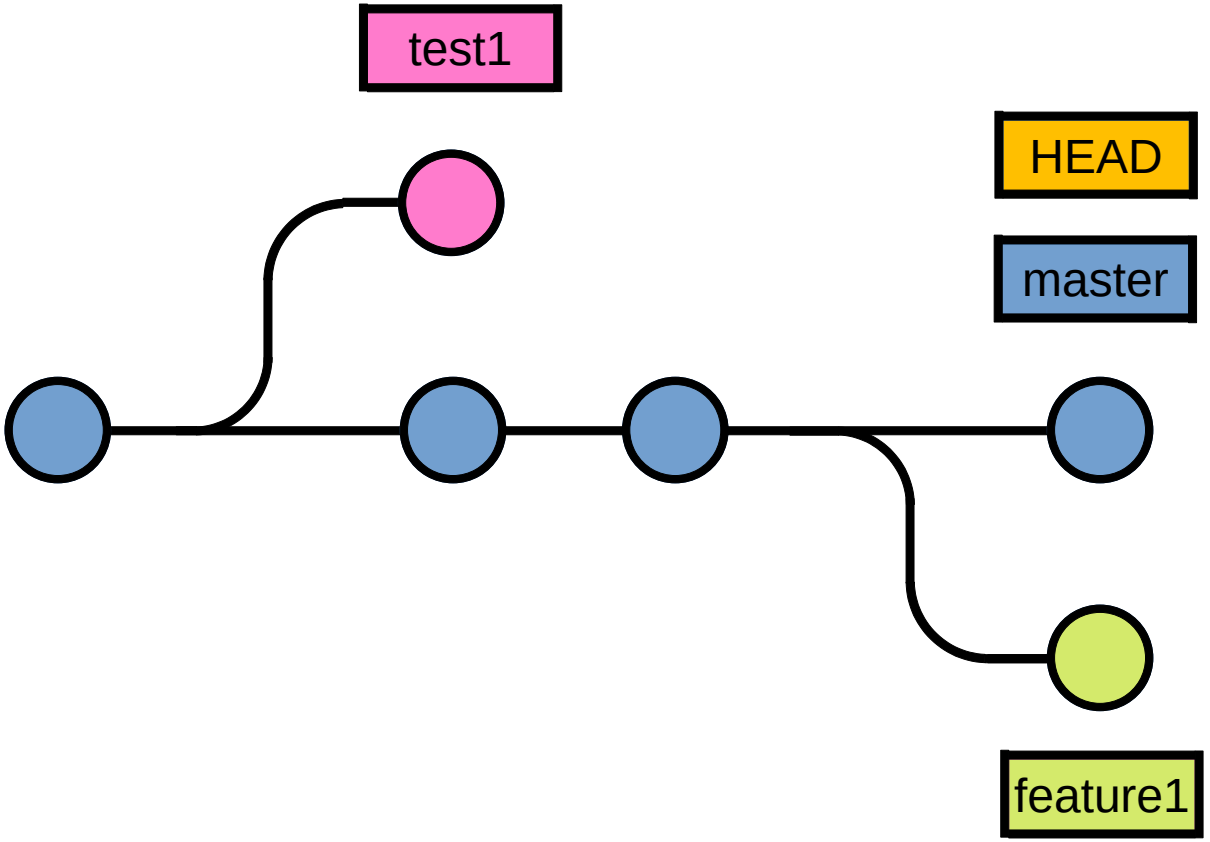

```
$ git checkout master
$ git merge feature1
```

file1 content:

```
Fri Oct 25 17:12:30 EDT 2019
Fri Oct 25 17:12:30 EDT 2019
<<<<<<< HEAD
Fri Oct 25 17:12:30 EDT 2019
=======
Fri Oct 25 17:13:15 EDT 2019
>>>>>>> feature1
```

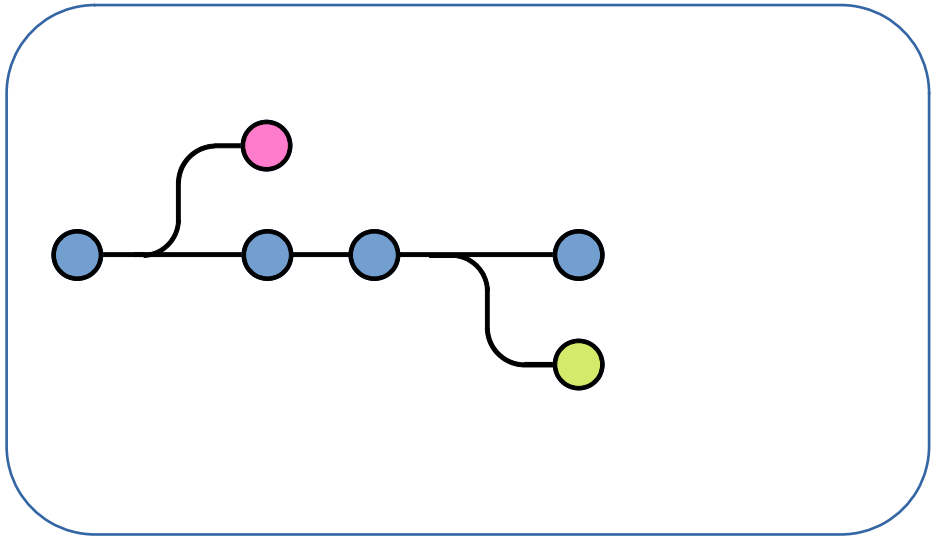
Remote

git remote



git remote

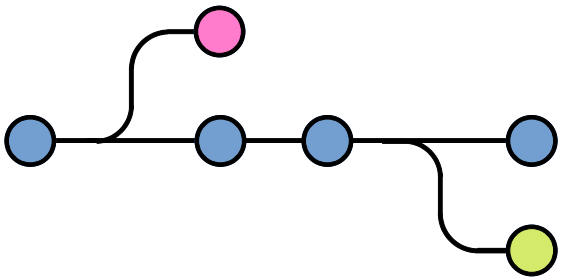
local



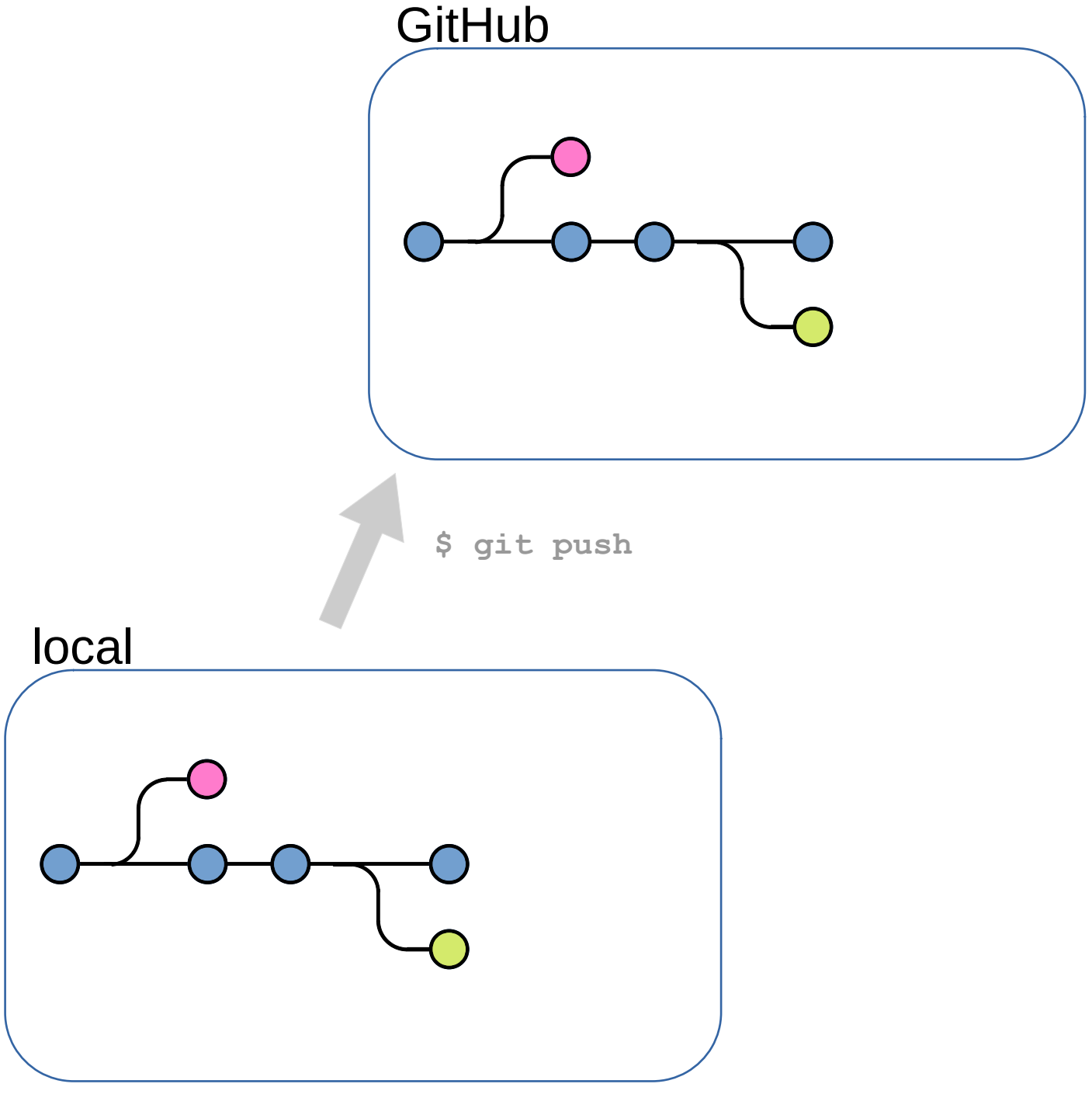
`git remote`

GitHub

local

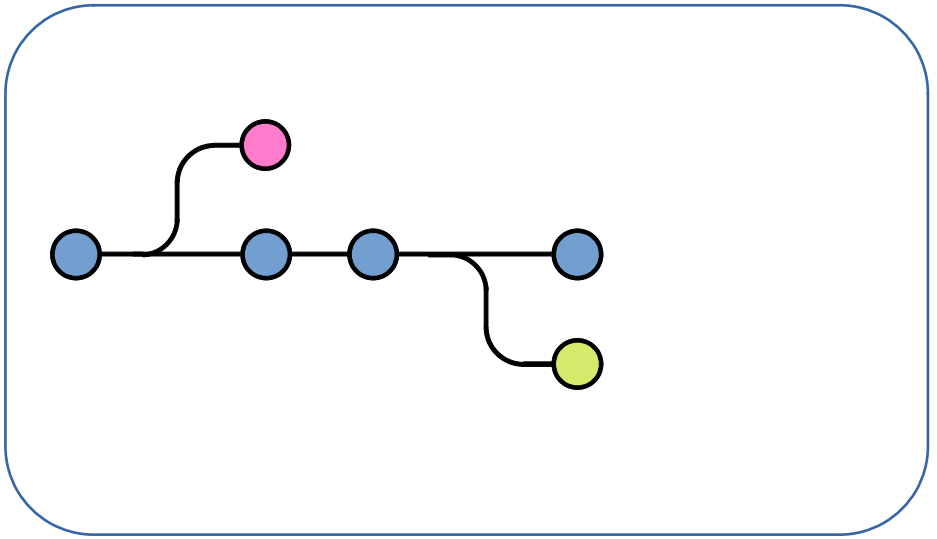


git push

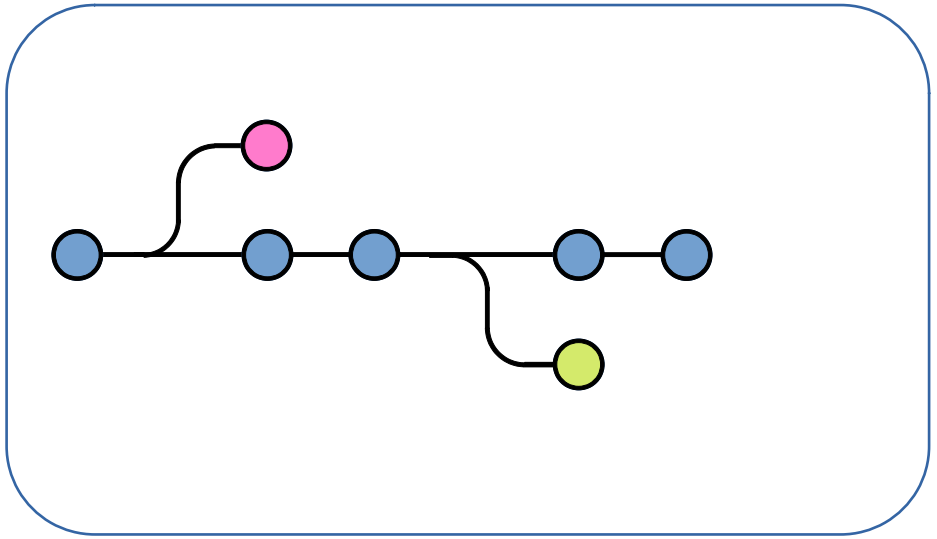


`git push`

GitHub

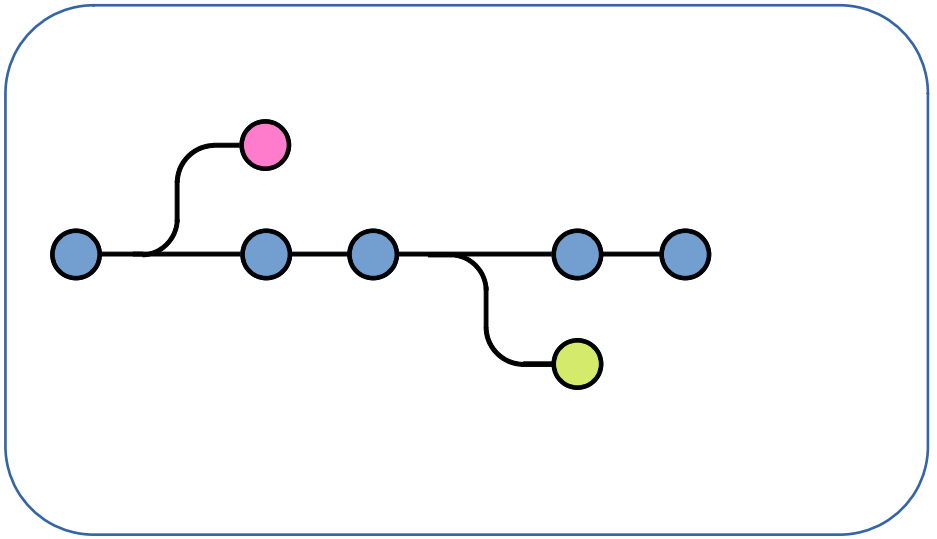


local



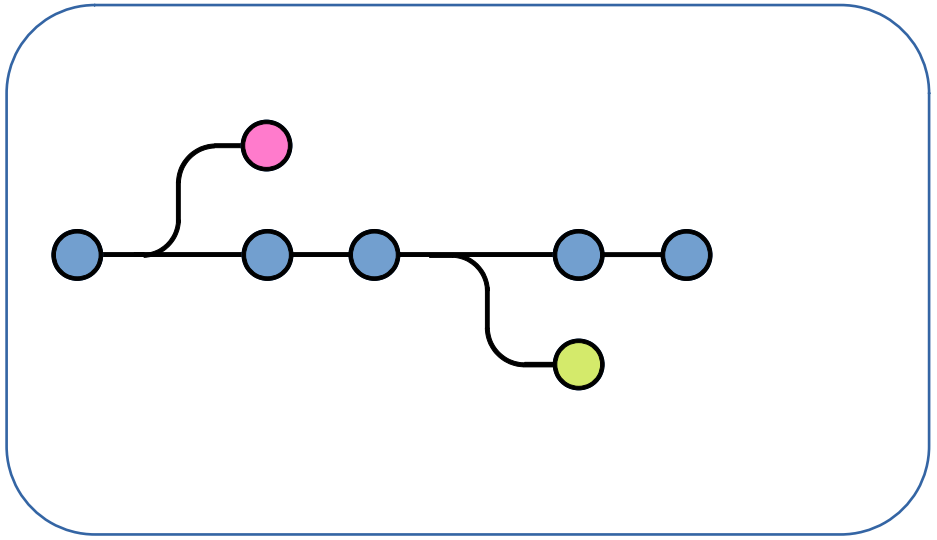
git push

GitHub



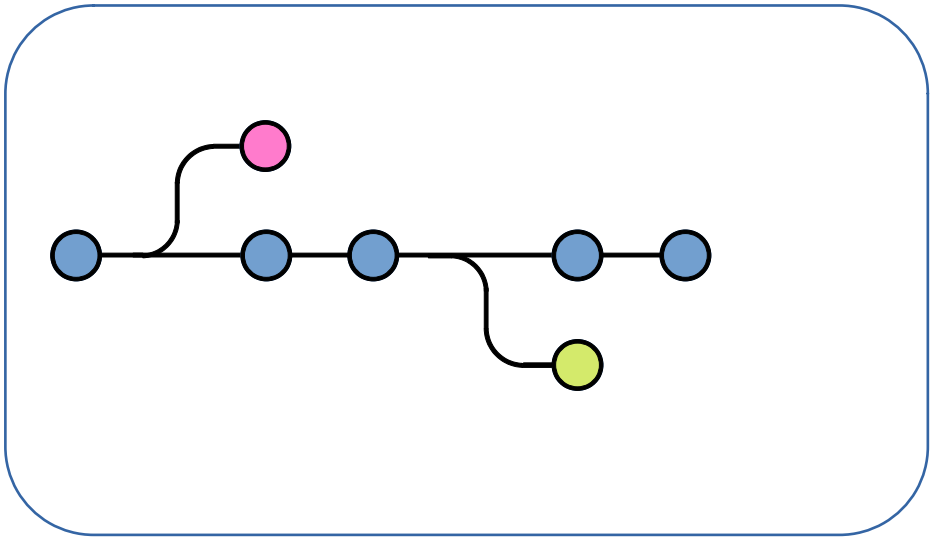
$ git push

local

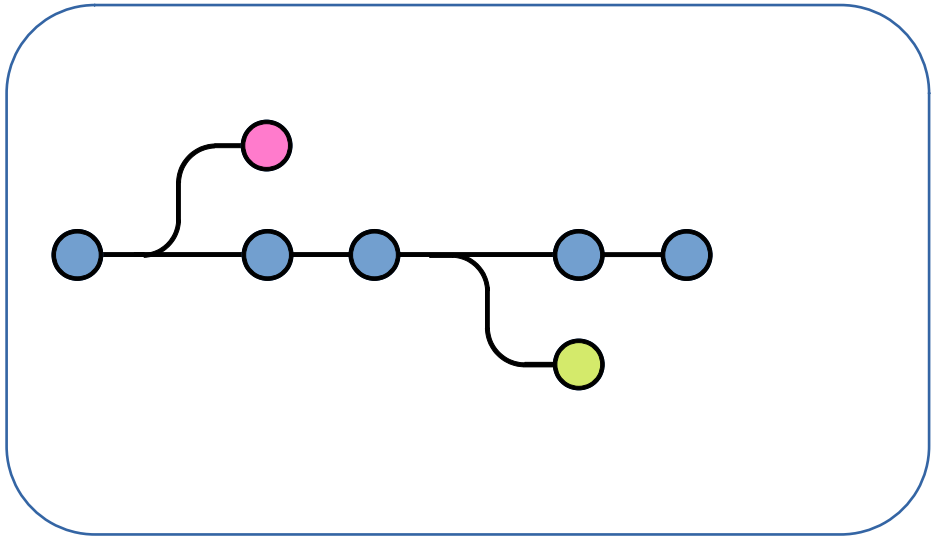


`git clone`

GitHub



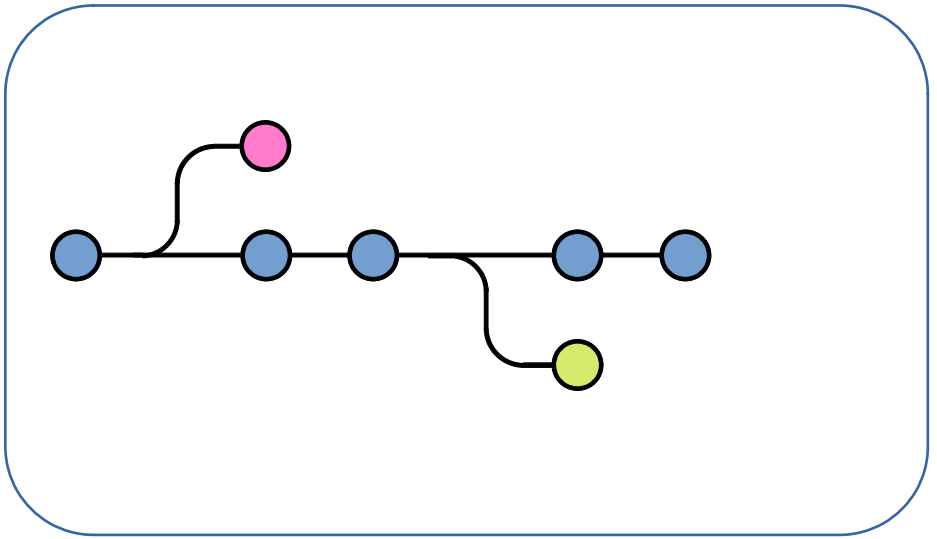
local



co-worker computer

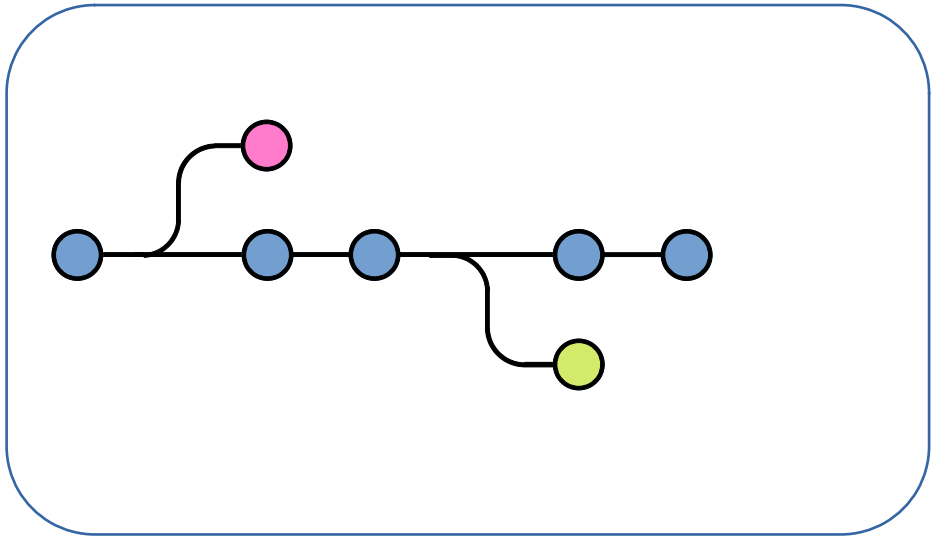
git clone

GitHub

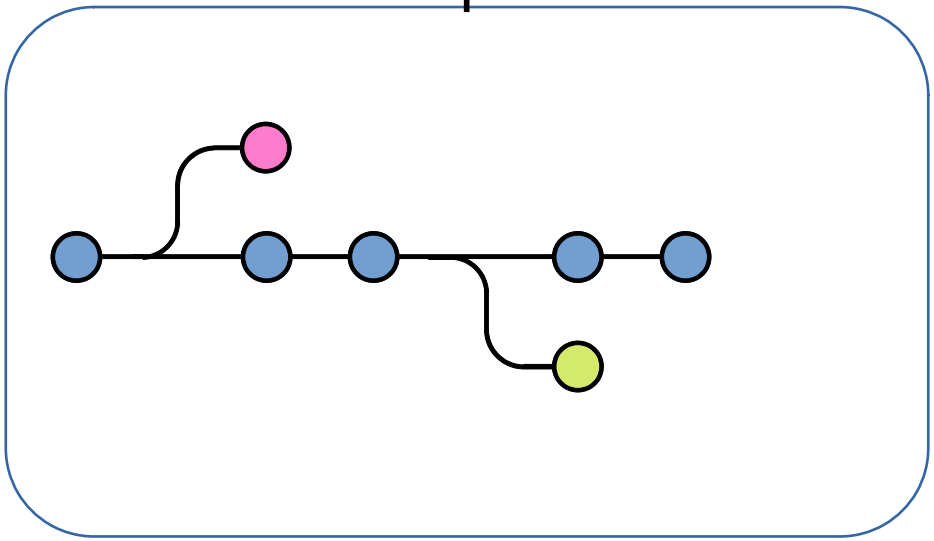


$ git clone

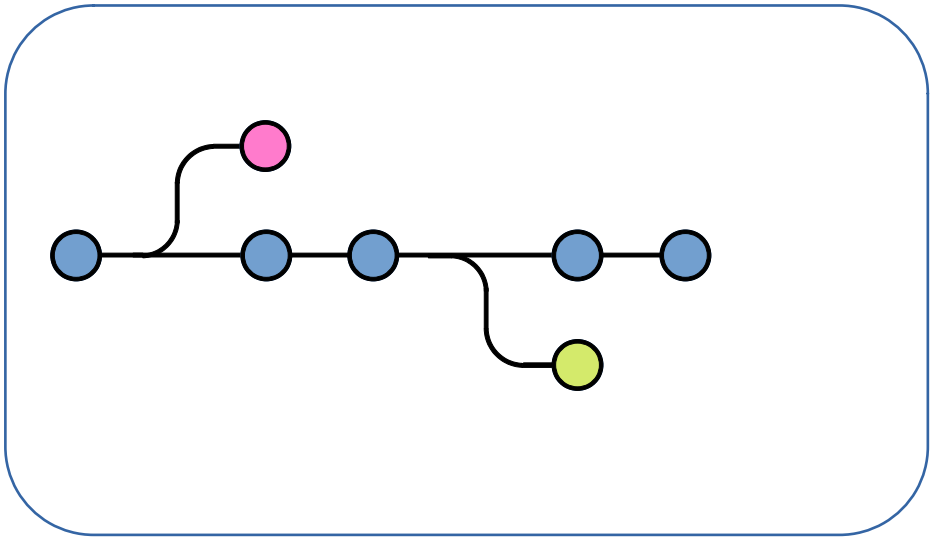
local



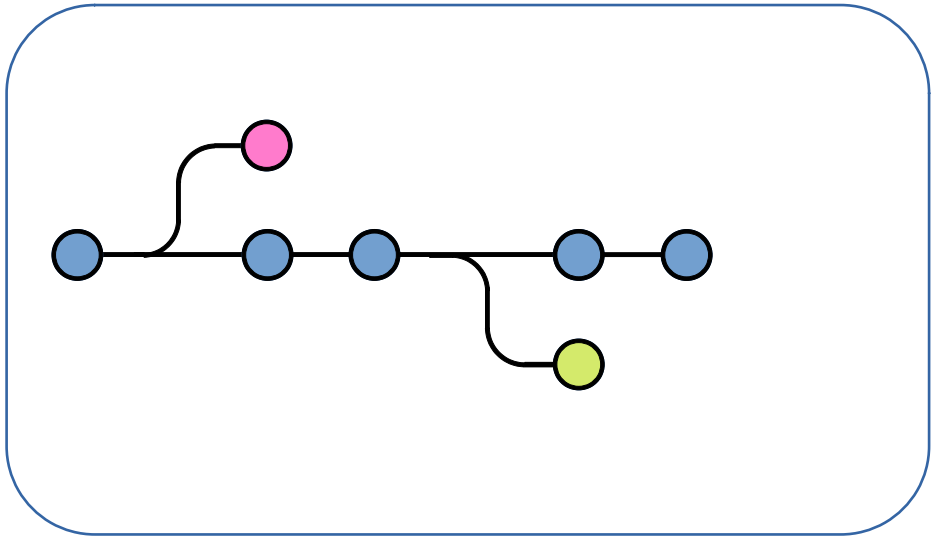
co-worker computer



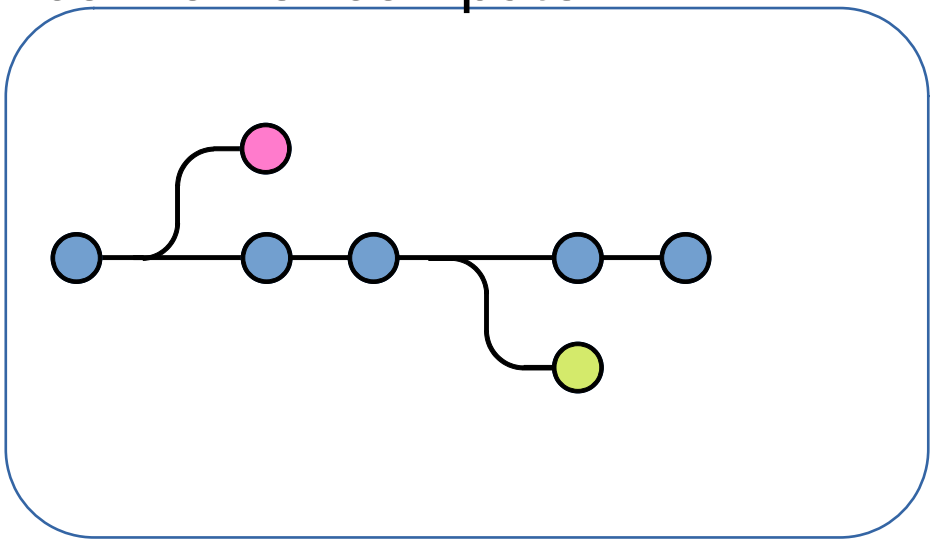
GitHub



local

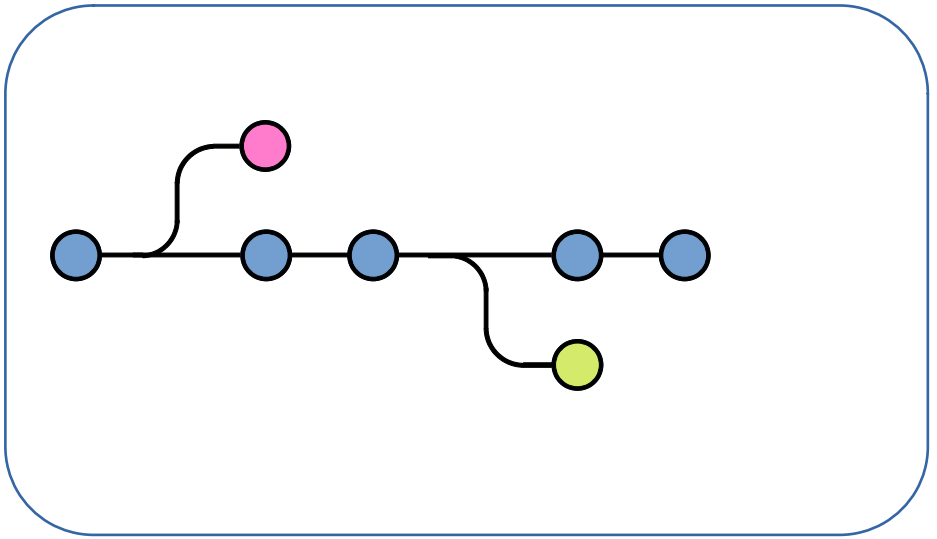


co-worker computer

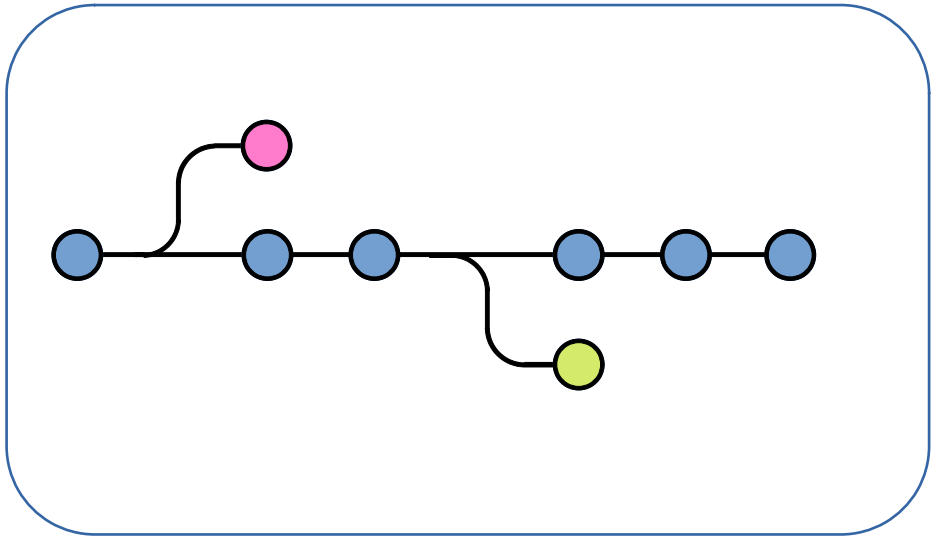


one day later

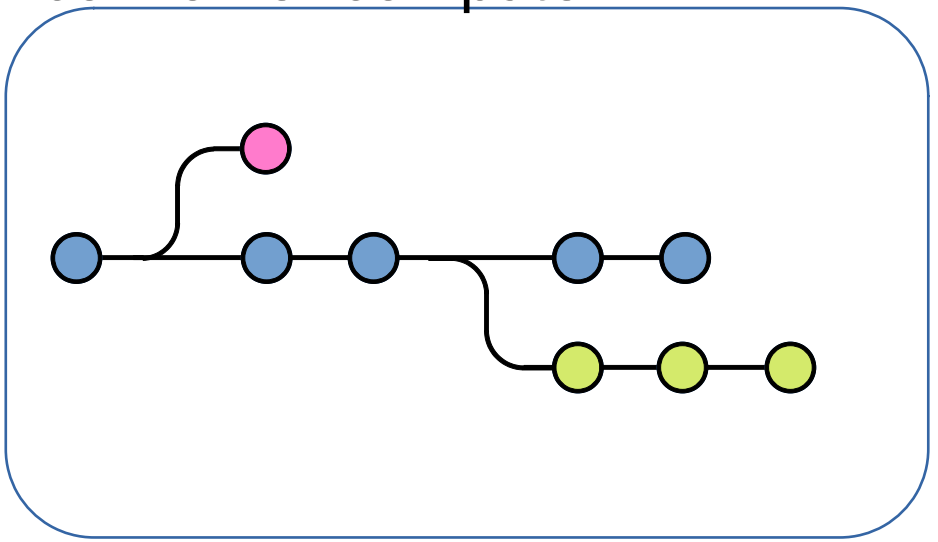
GitHub



local

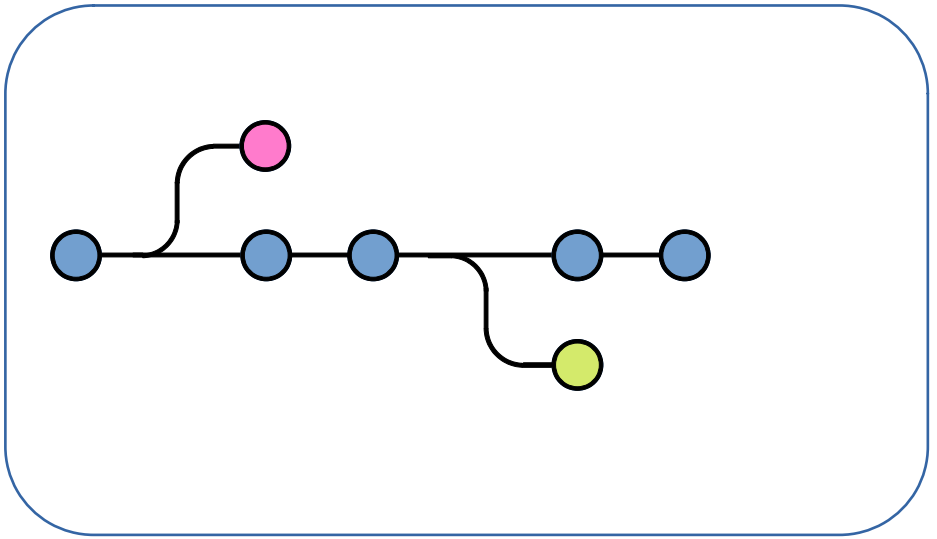


co-worker computer



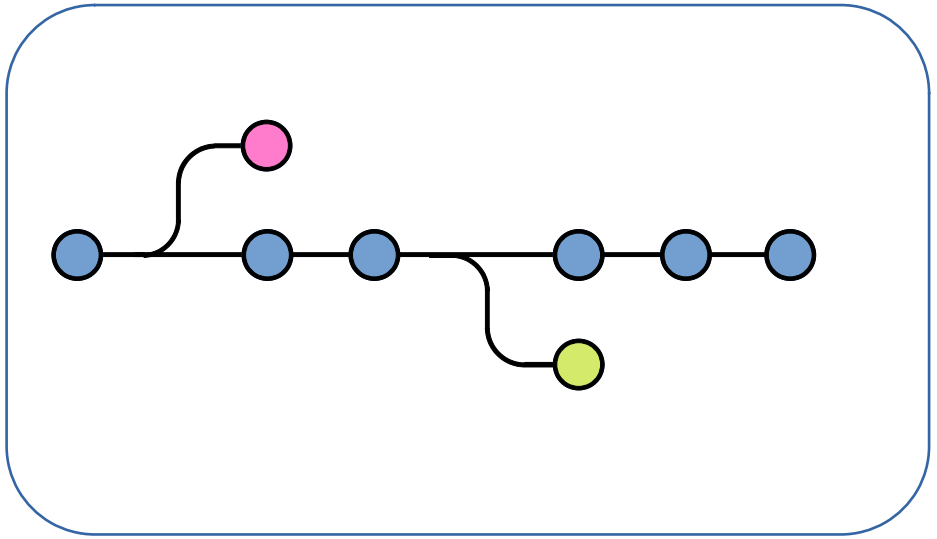
git pull/push

GitHub

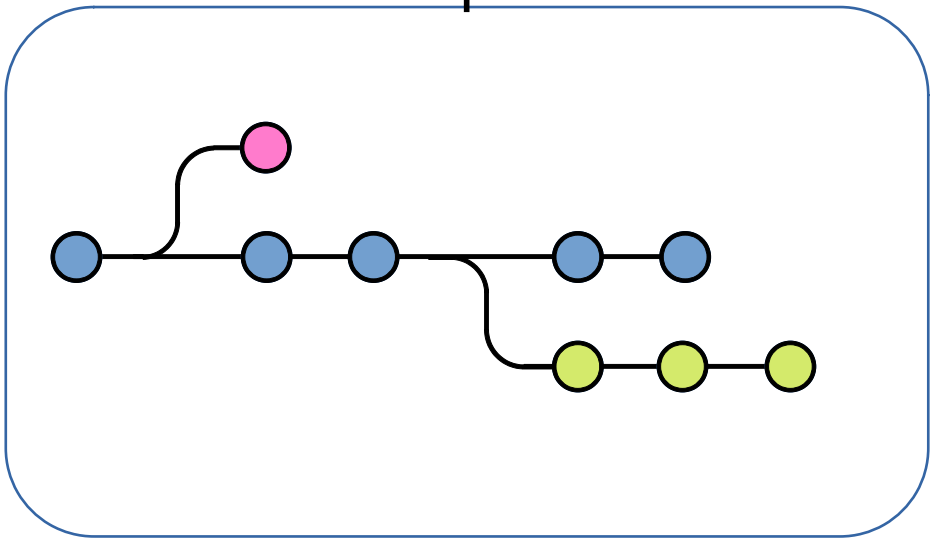


$ git pull

local

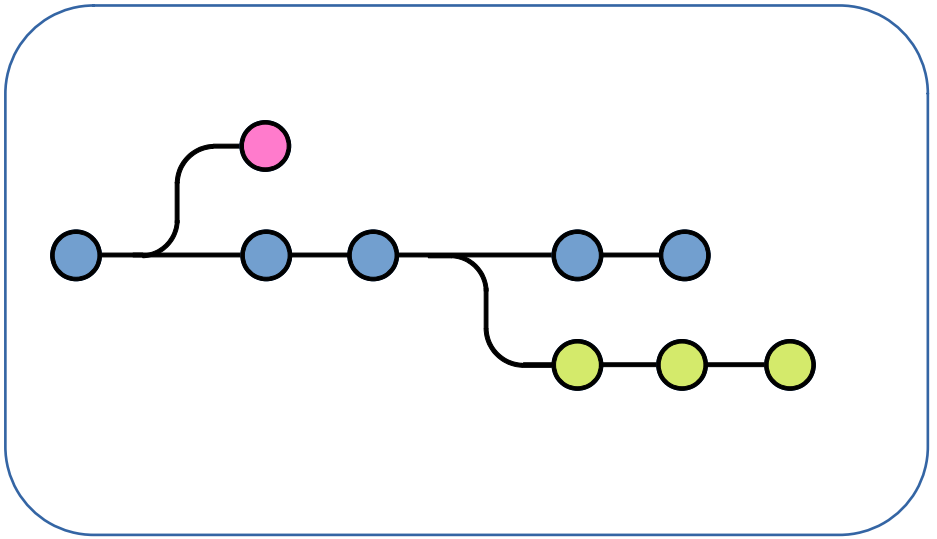


co-worker computer



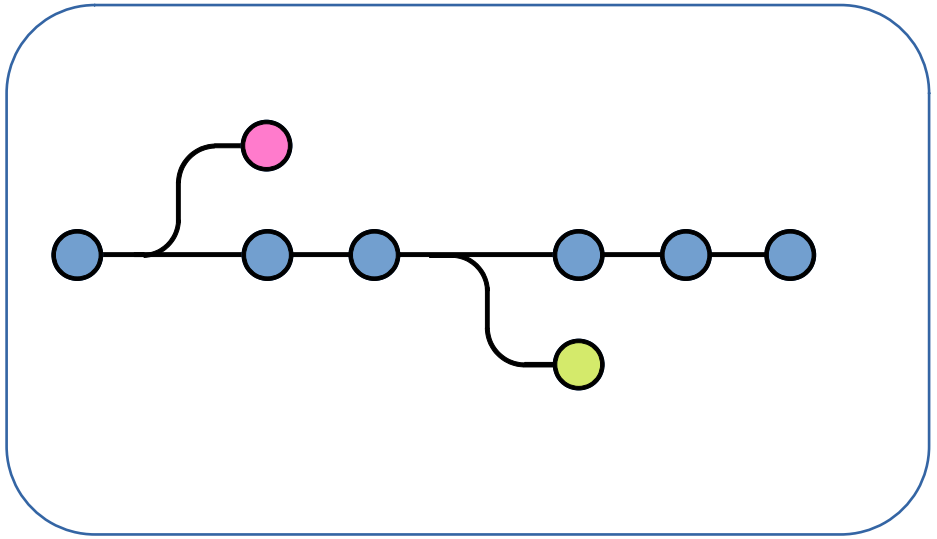
git pull/push

GitHub

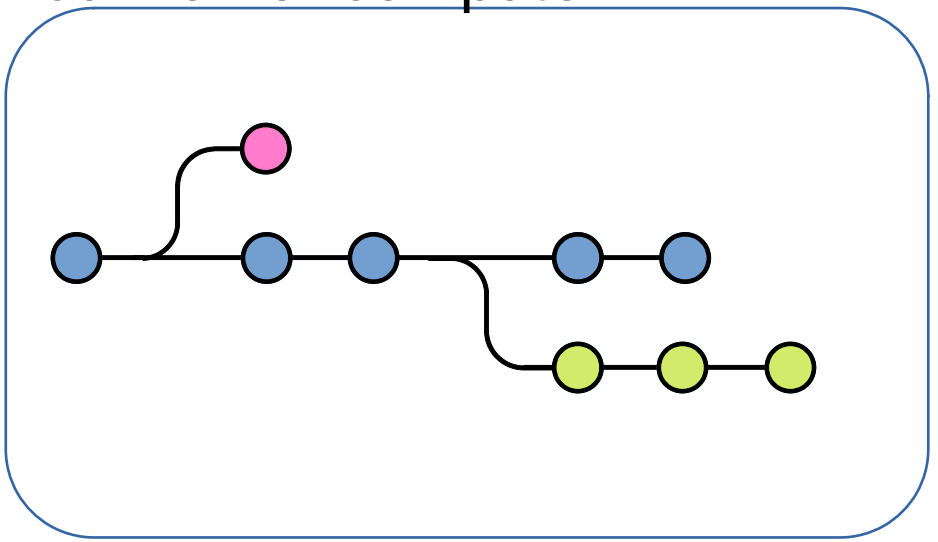


$ git push

local

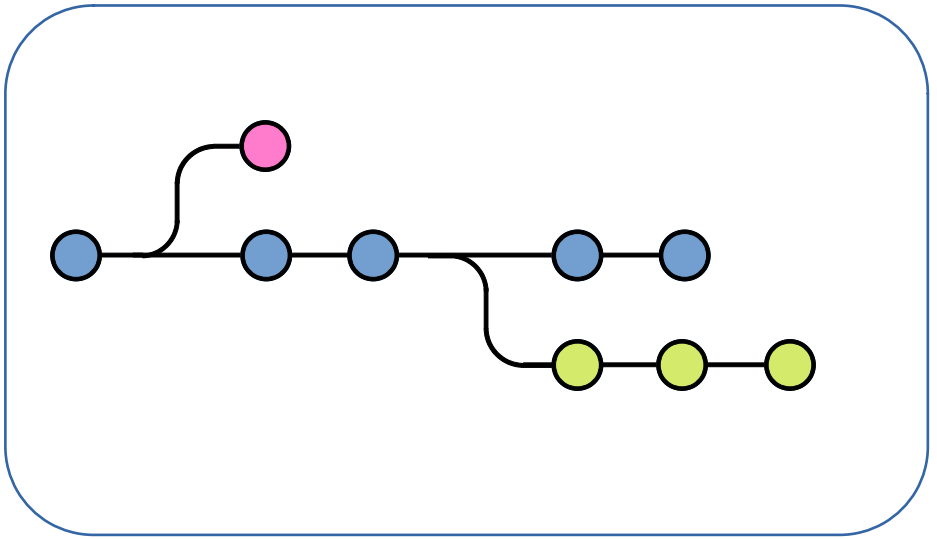


co-worker computer



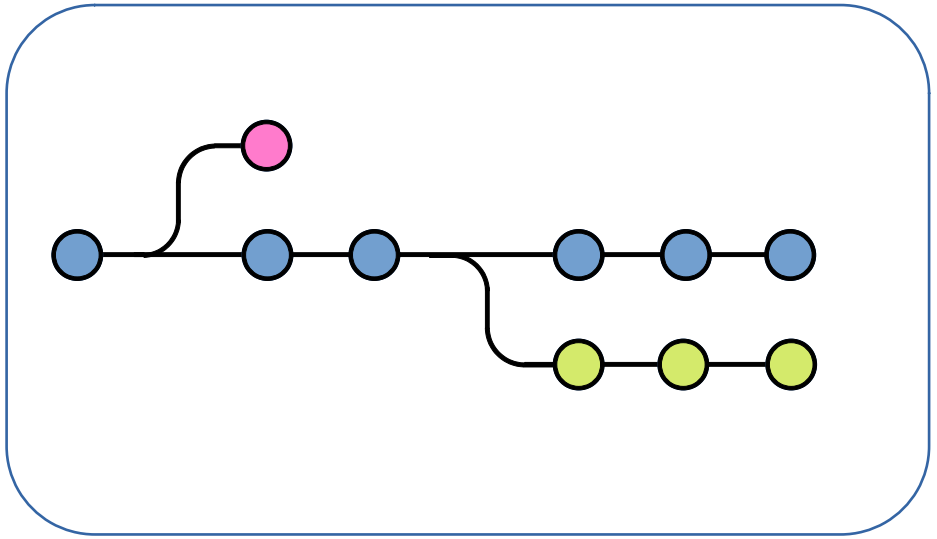
git pull/push

GitHub

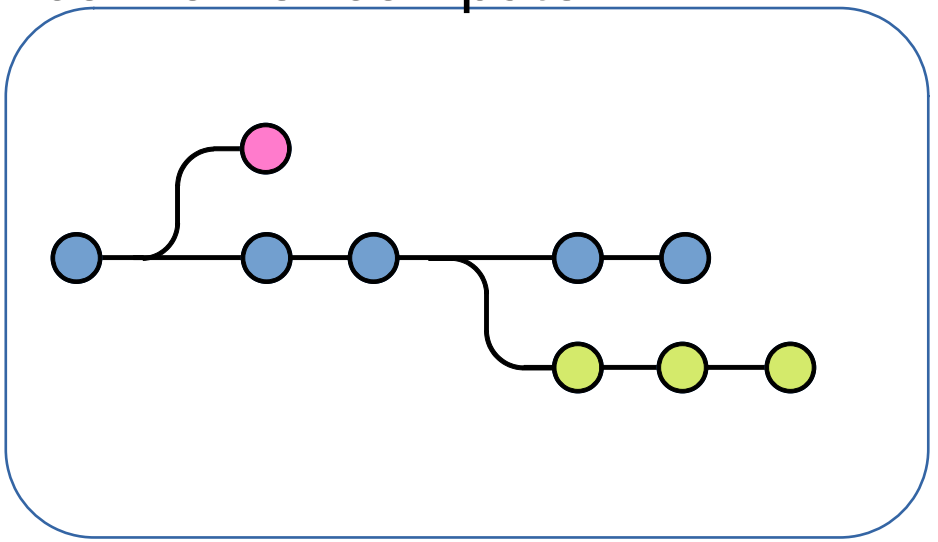


$ git pull

local

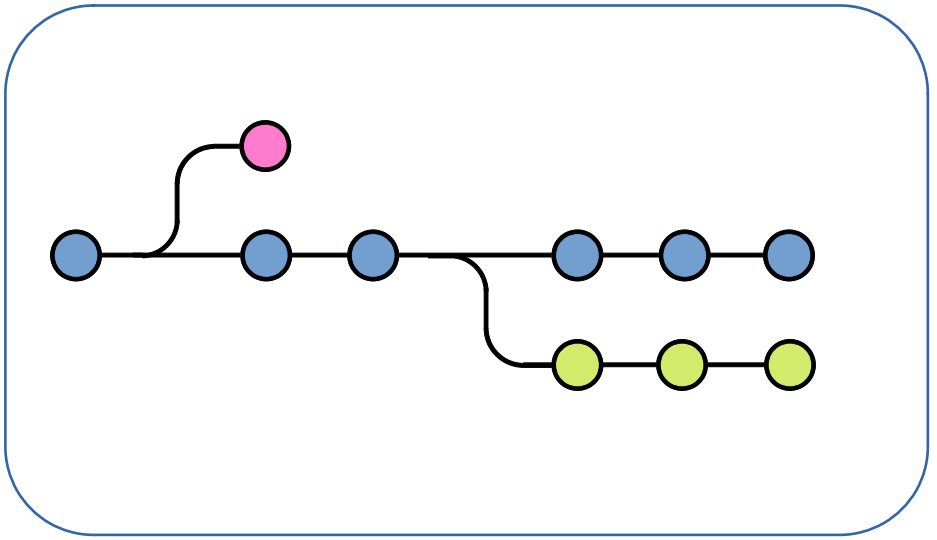


co-worker computer



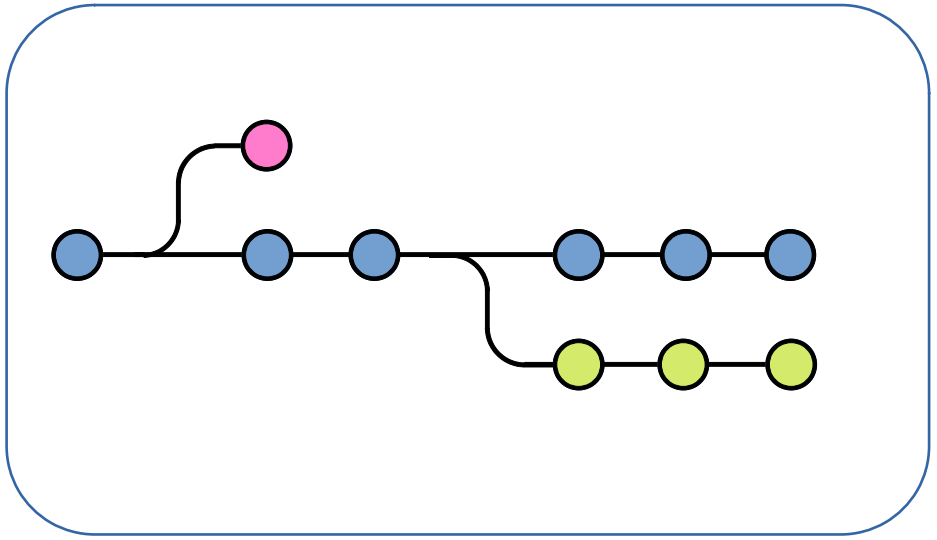
git pull/push

GitHub

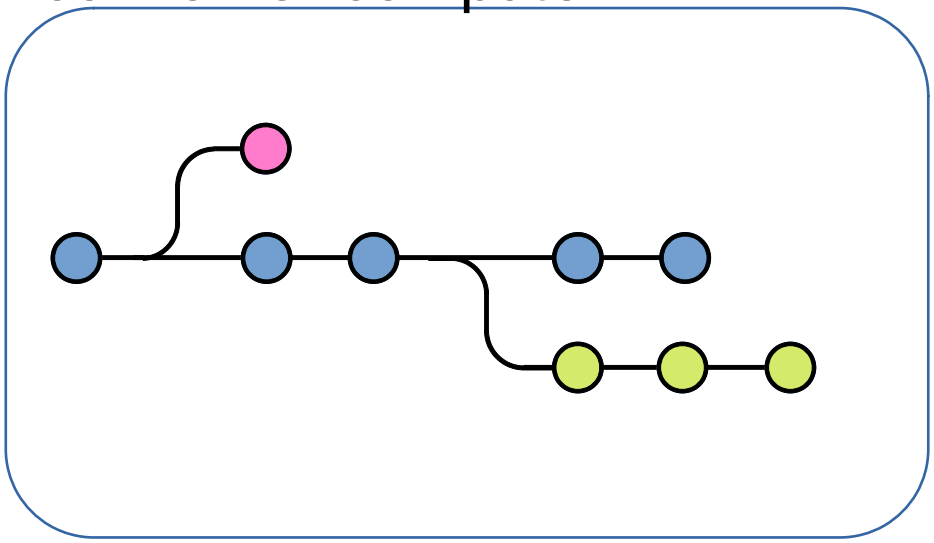


$ git push

local

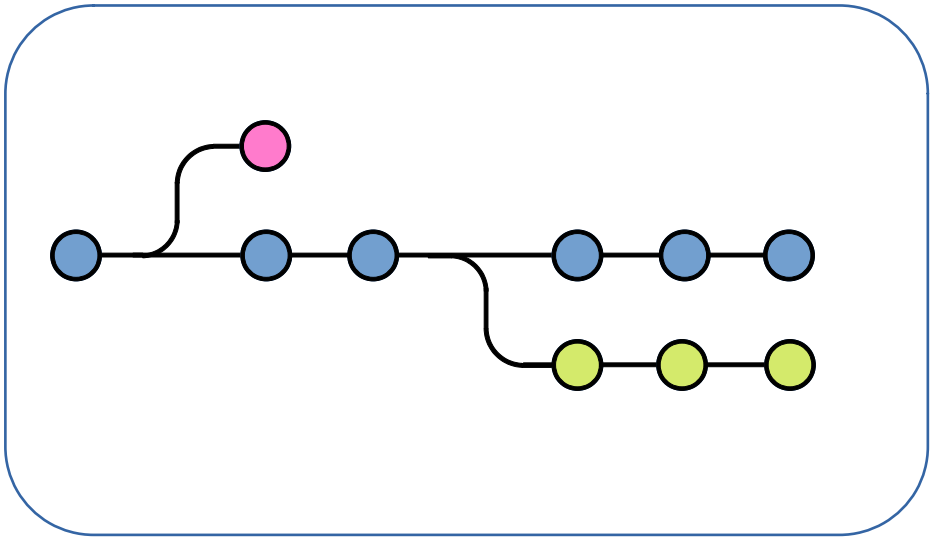


co-worker computer



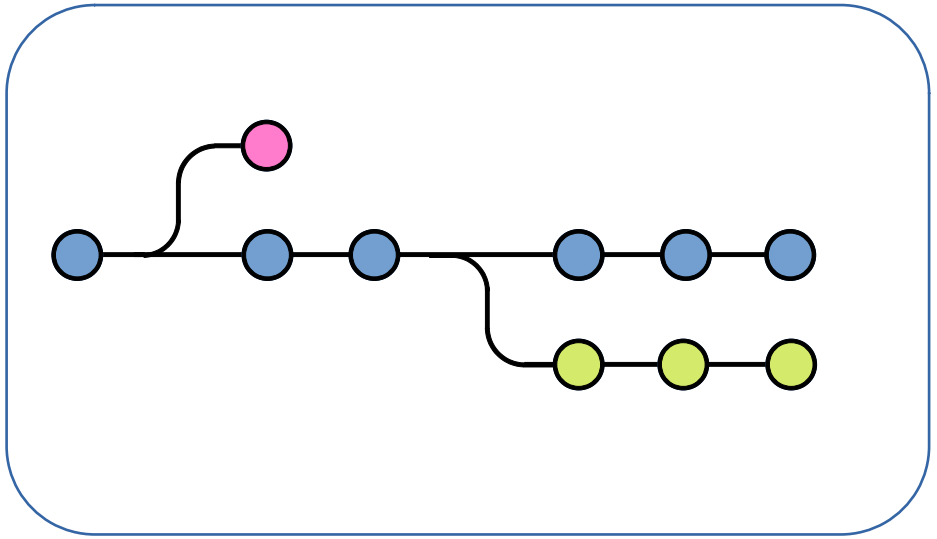
git pull/push

GitHub

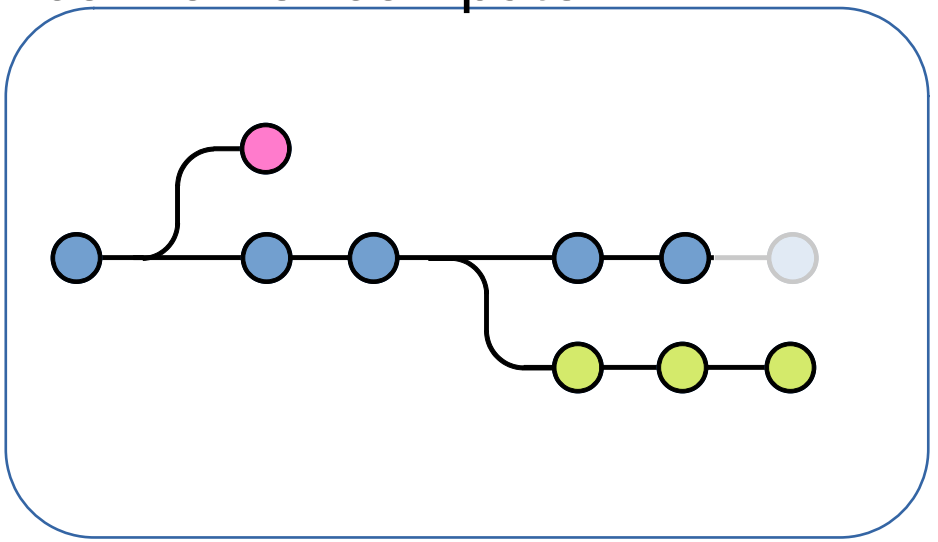


$ git fetch

local

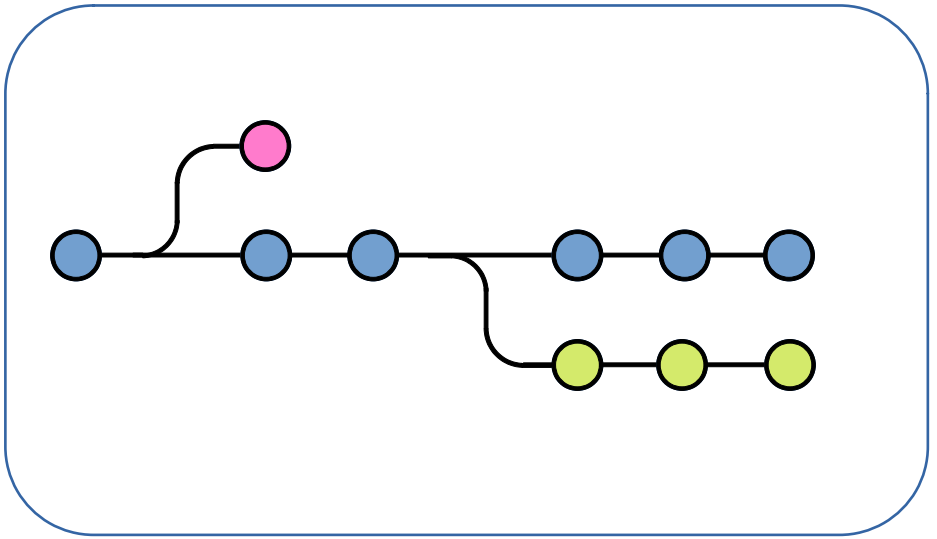


co-worker computer



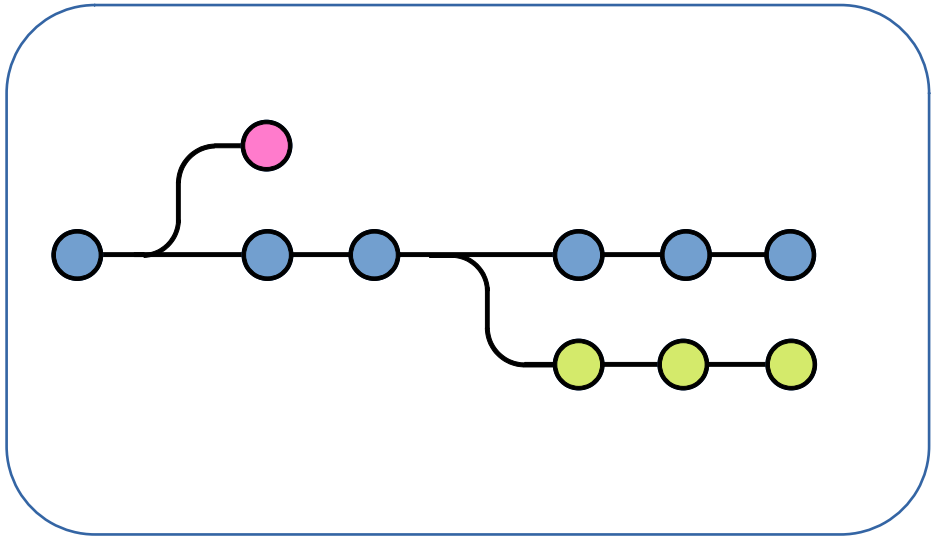
git pull/push

GitHub

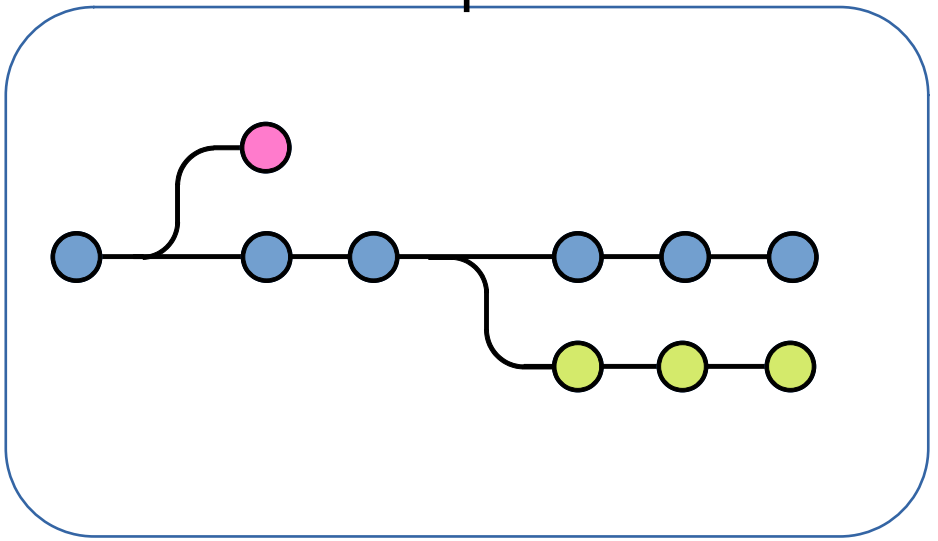


$ git pull

local



co-worker computer



etc

`git commit -m "..."`

Explicit messages

```
$ git commit -m "[mpi] Added parrallelization to Monte Carlo sampling"
$ git commit -m "[mkl] BUG, not working yet, do not use."
```

Bad messages

```
$ git commit -m "error 2"
$ git commit -m "."
```

Reference

onlywei.github.io/explain-git-with-d3/#checkout-b

Visualizing Git Concepts with D3

This website is designed to help you understand some basic git concepts visually. This is my first attempt at using both SVG and D3. I hope it is helpful to you.

Adding/staging your files for commit will not be covered by this site. In all sandbox playgrounds on this site, just pretend that you always have files staged and ready to commit at all times. If you need a refresher on how to add or stage files for commit, please read Git Basics.

Sandboxes are split by specific git commands, listed below.

Basic Commands		Undo Commits	Combine Branches	Remote Server	
git commit	git checkout	git reset	git merge	git fetch	git push
git branch	git checkout -b	git revert	git rebase	git pull	git tag

`git branch name` will create a new branch named "name". Creating branches just creates a new tag pointing to the currently checked out commit.

Branches can be deleted using the command `git branch -d name`.

Type `git commit` and `git branch` commands to your hearts desire until you understand this concept.