The art of developing scientific software

Inga Ulusoy, Scientific Software Center, Institute for Scientific Computing, Heidelberg University

March 2021

What you will learn...

- How to develop scientific software professionally:
  - No more broken code through version control/rigorous code review/testing/...
  - Track your progress and analyze your code
  - Manage your implementation in a way that facilitates further additions to the code
  - Generate reproducible and reliable scientific results
  - Make your contributions as a "scientific software developer" visible

Overview

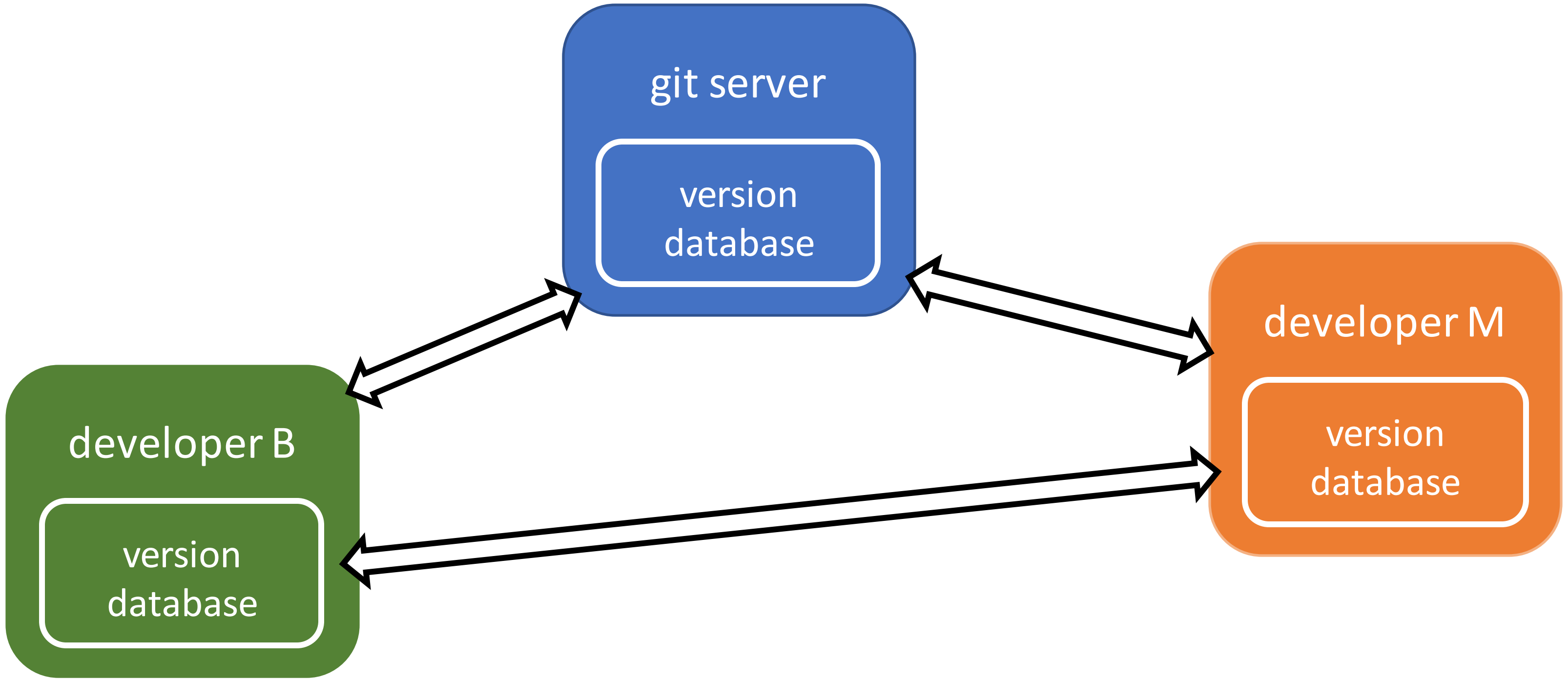
- Unit 1: Introduction to github
- Unit 2: Collaborative software development
- Unit 3: Testing and documenting software
- Unit 4: Continous integration (CI) and publishing code

Unit 1: Introduction to github

- git basics
- Working with repositories on github
- Branching and pull requests
- Code review and merging

git version control system (VCS)

git is a distributed version control system that keeps track of your changes to the software. Distributed means that every developer has a complete copy of the project, including its history.



git version control system (VCS)

- Revert file to previous state
- Revert entire project to previous state
- Review changes over time
- Review and track issues, …

Commit:
A snapshot of your file system
A commit saves the state of your project at that time.

Repository:
A directory containing your project and additional files to communicate with git
This is not equivalent your working directory.

Checkout:
A copy of repository content in your working directory
This can be a specific file, commit, branch, …

Branch:
A new line of development in your project
The branch leaves the main status of the project unchanged.

The three stages of git

modified:
File has been modified but changes are not commited to database yet.

Working directory

staged:
Modified file has been marked to be committed (to be included in the next snapshot).

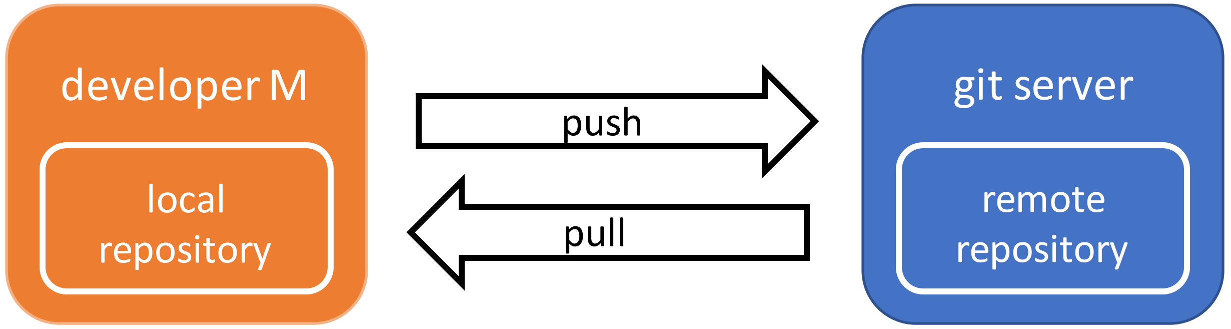
Staging area
(Staging index)

committed:
Modified file has been added to the local database.

.git directory
(repository)

Remote repositories

Remote repositories are hosted on the internet or network. You can set up your own git server or use services such as GitHub, GitLab or Bitbucket.



git workflow

1. `git clone the-repository-URL` to clone an exisiting repository from the remote server or turn a local directory into a git repository through `git init`

2. create a branch in your local repository where you make your changes to the code through `git branch your-branch-name` and then checking out the branch through `git checkout your-branch-name`

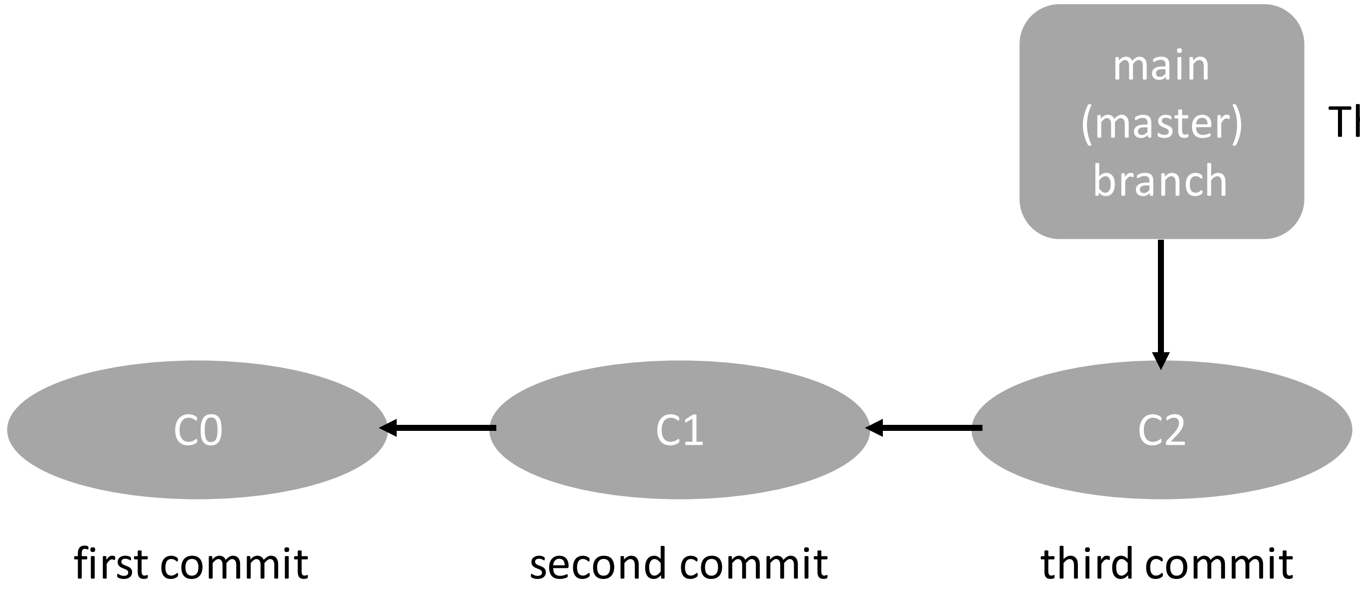
3. make your changes in your branch

4. stage your changes through `git add your-changed-files`

5. commit your changes to your local repository through `git commit –m "your-meaningful-commit-message"`

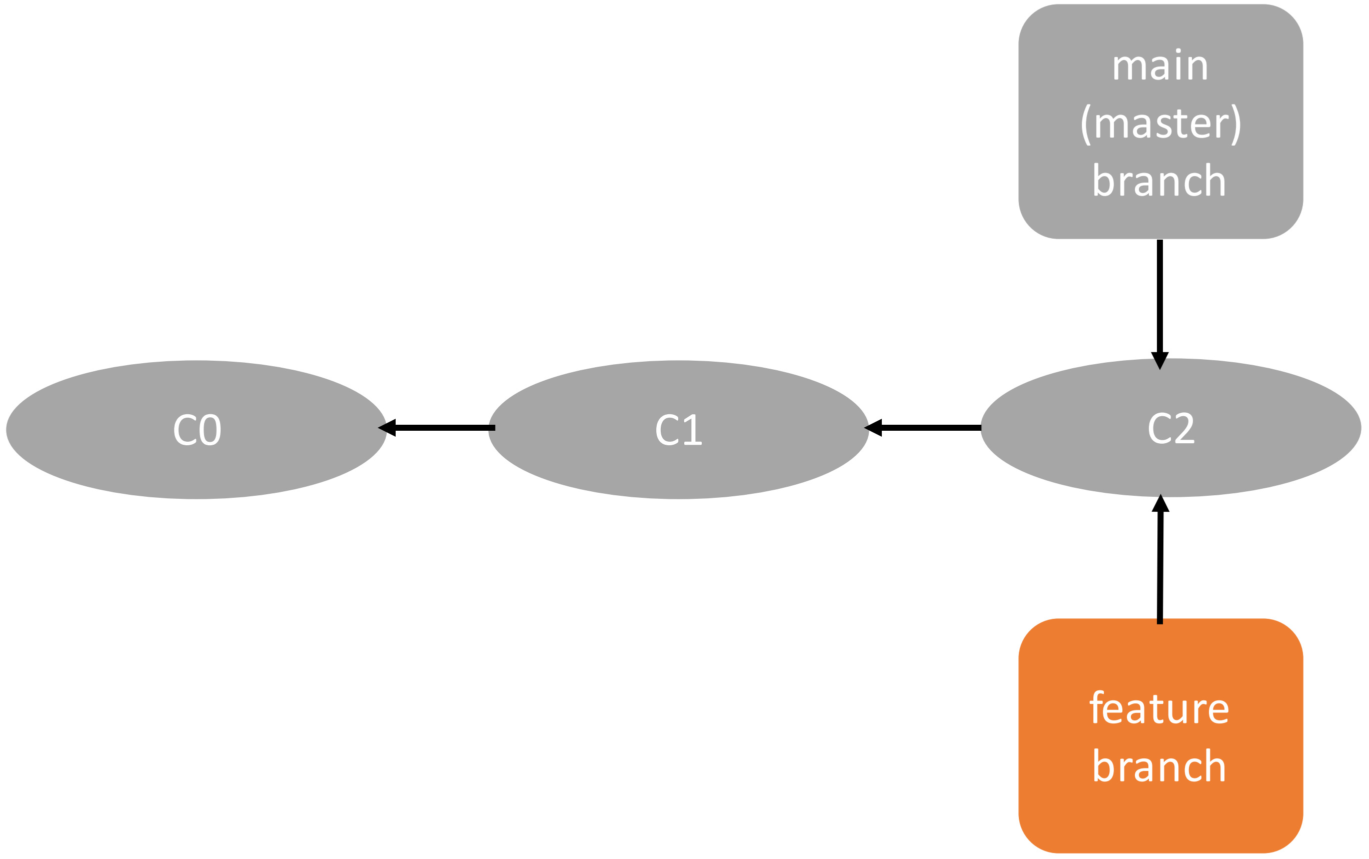
6. push your changes to the remote repository through `git push` or `git push –u origin your-branch-name` if your local branch is not yet initialized on the remote

git commits, branches and merges



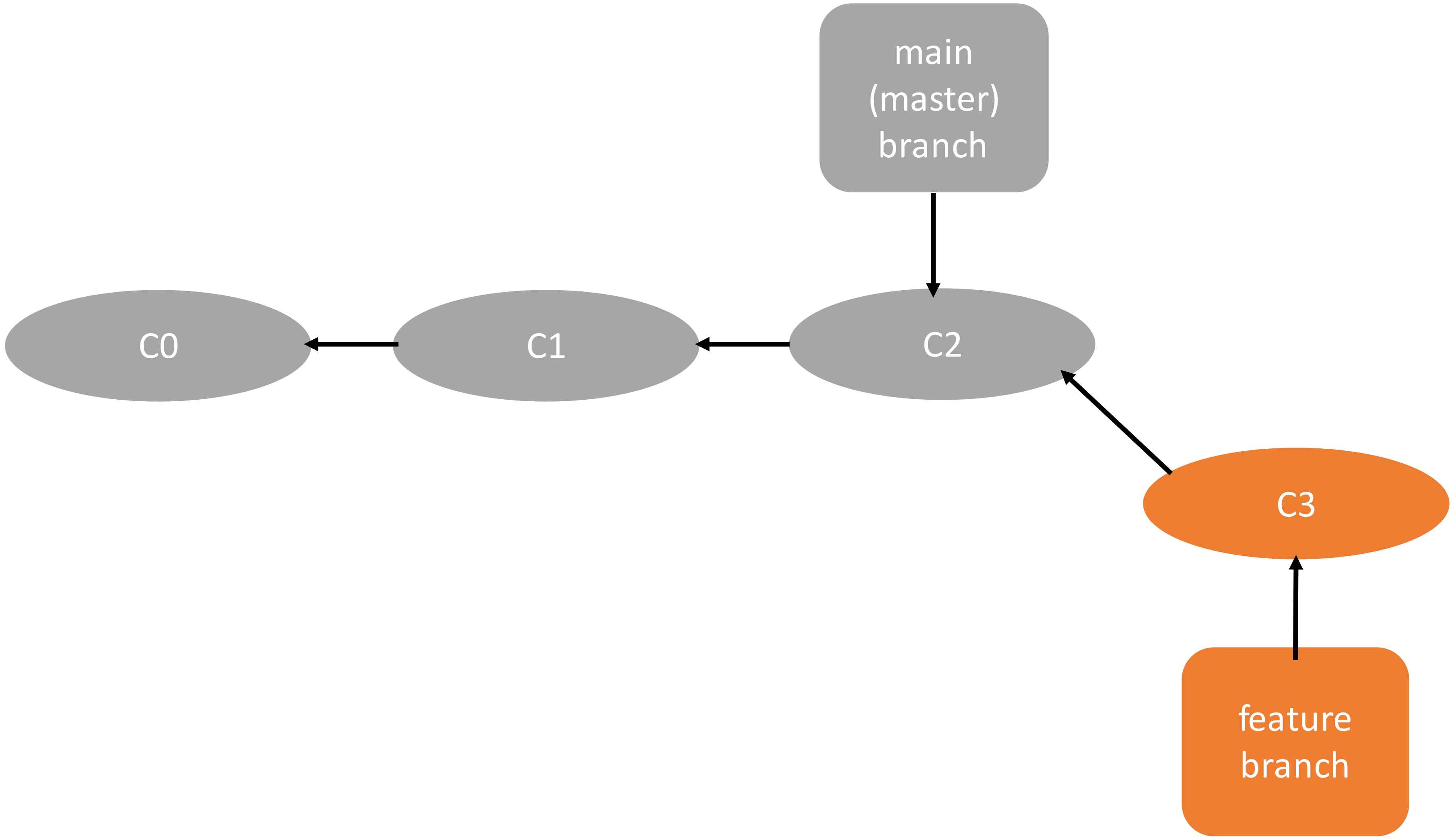
This is the default branch.

git commits, branches and merges

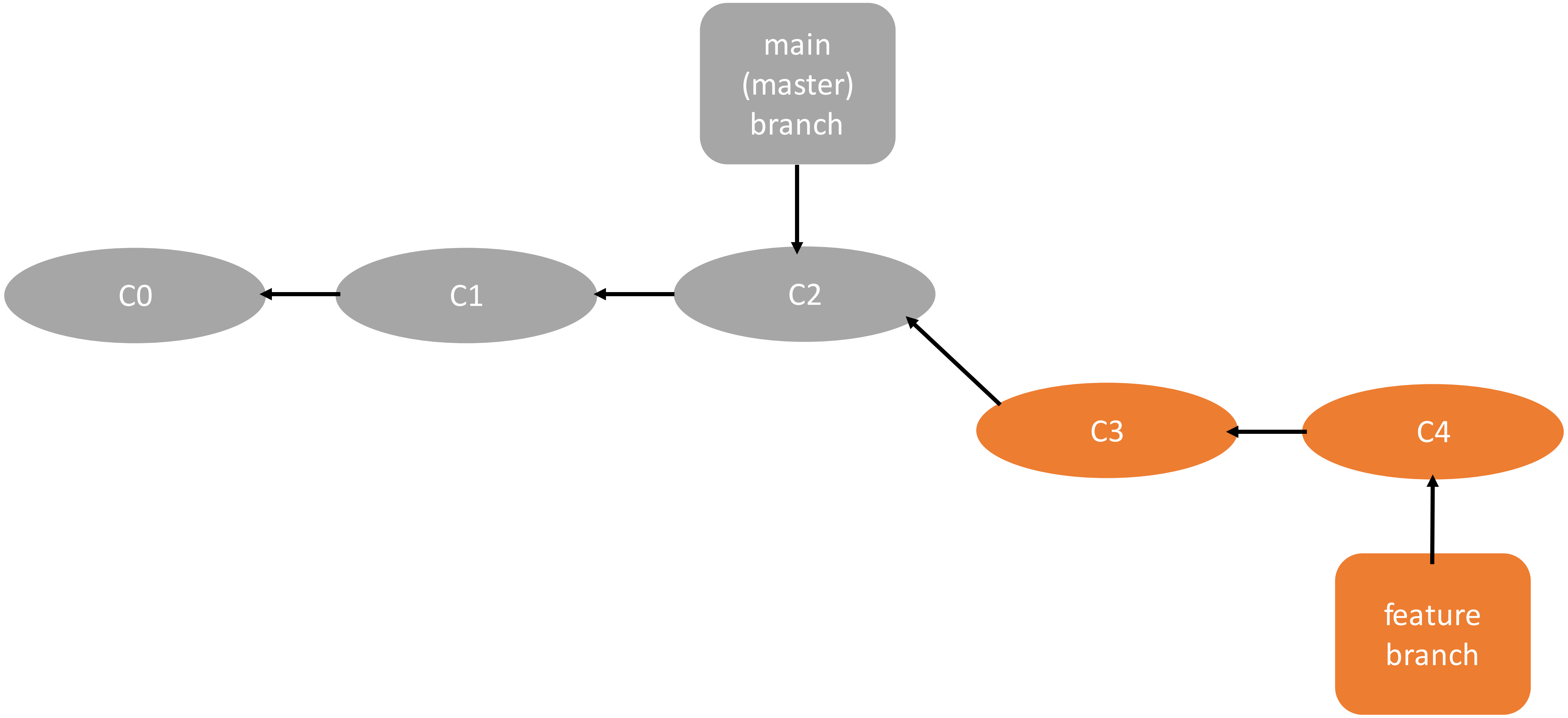


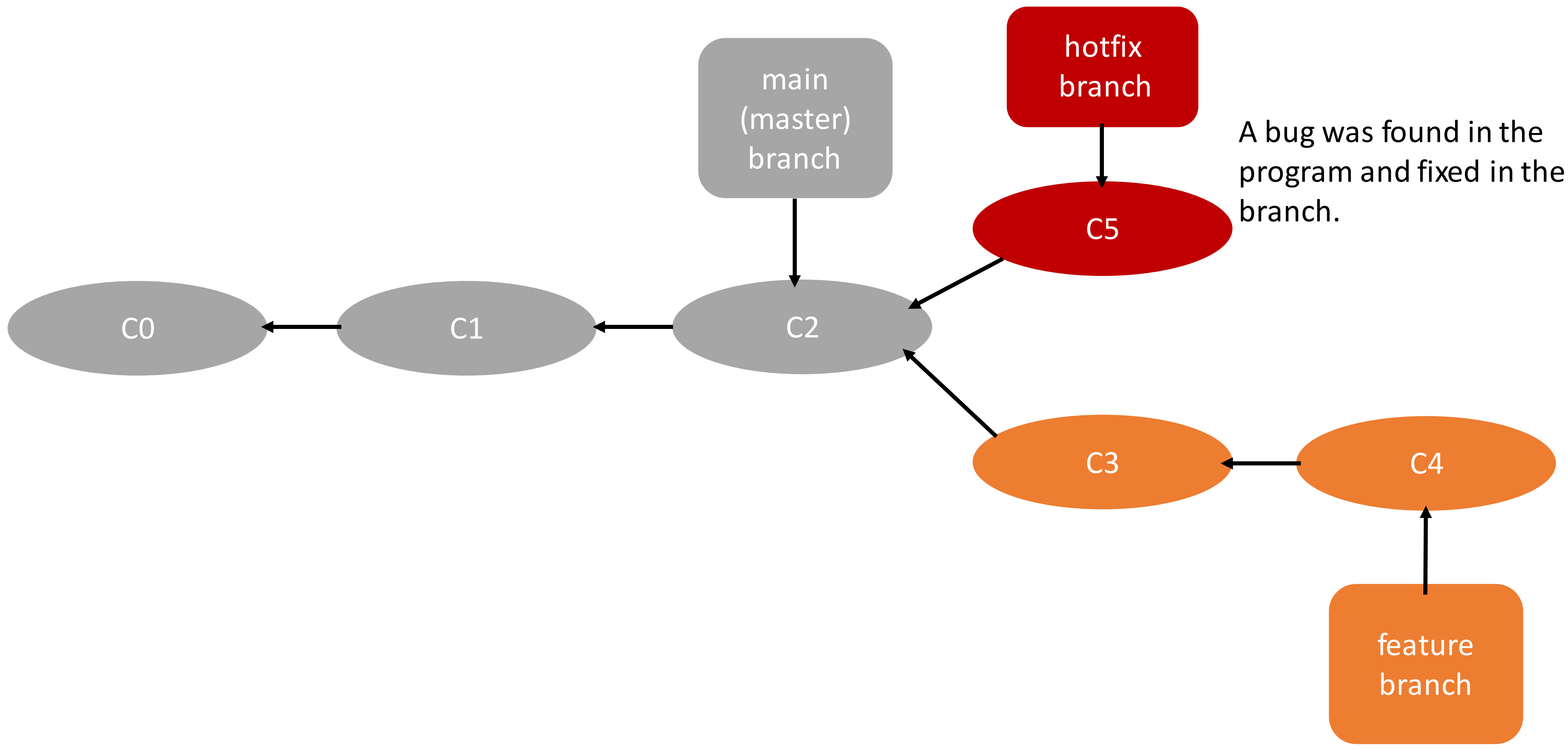
This is the new feature branch.

git commits, branches and merges

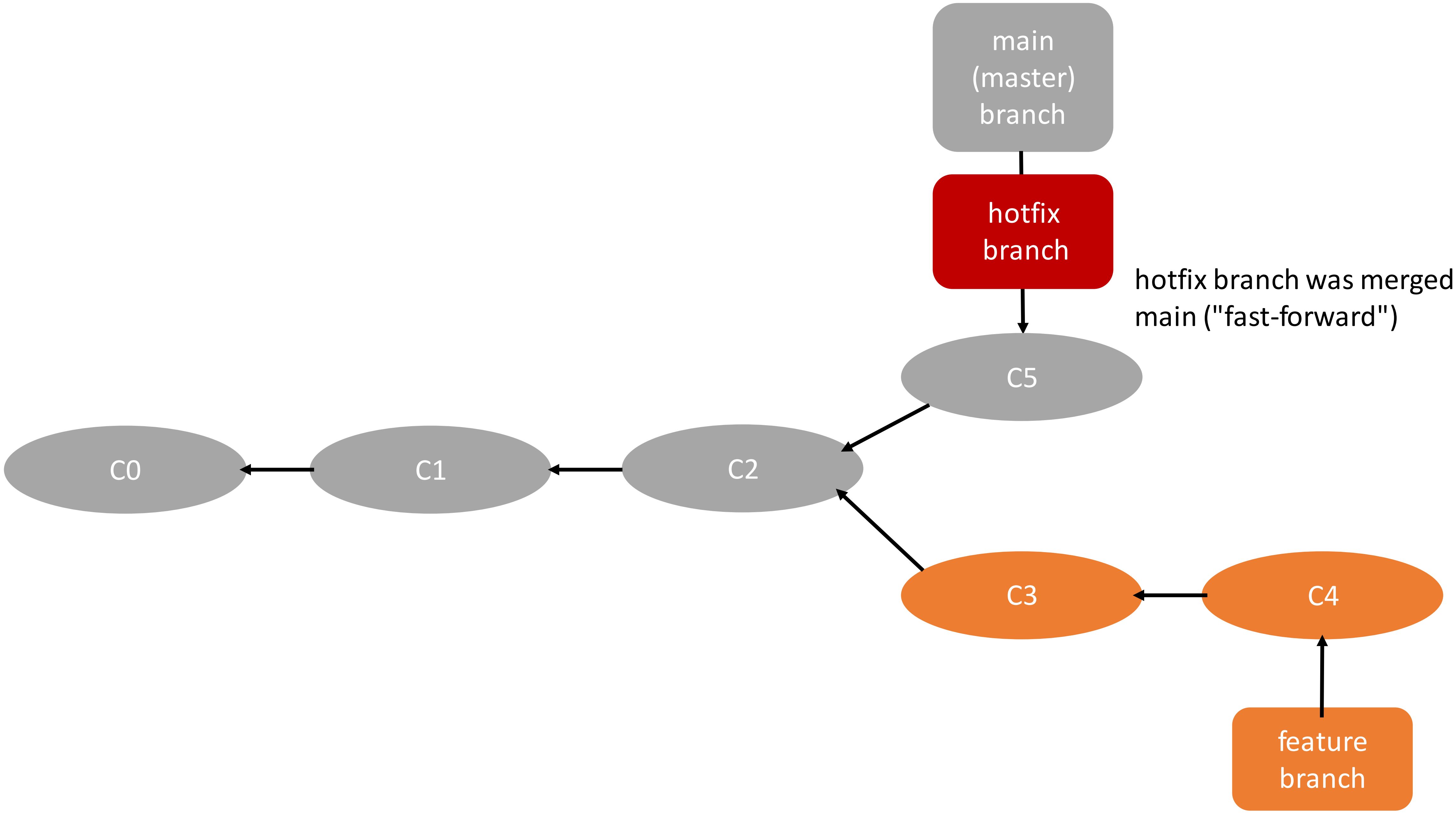


git commits, branches and merges

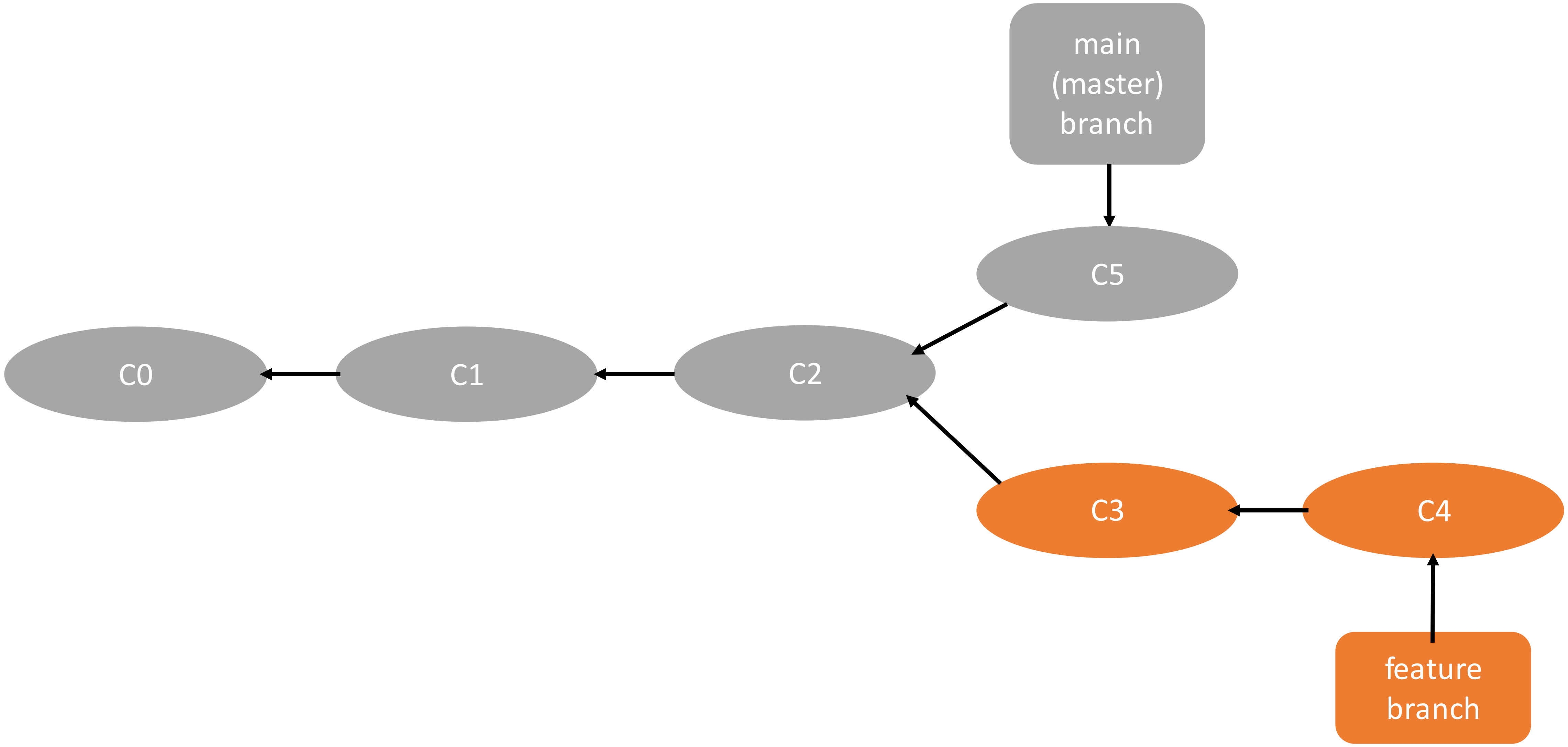


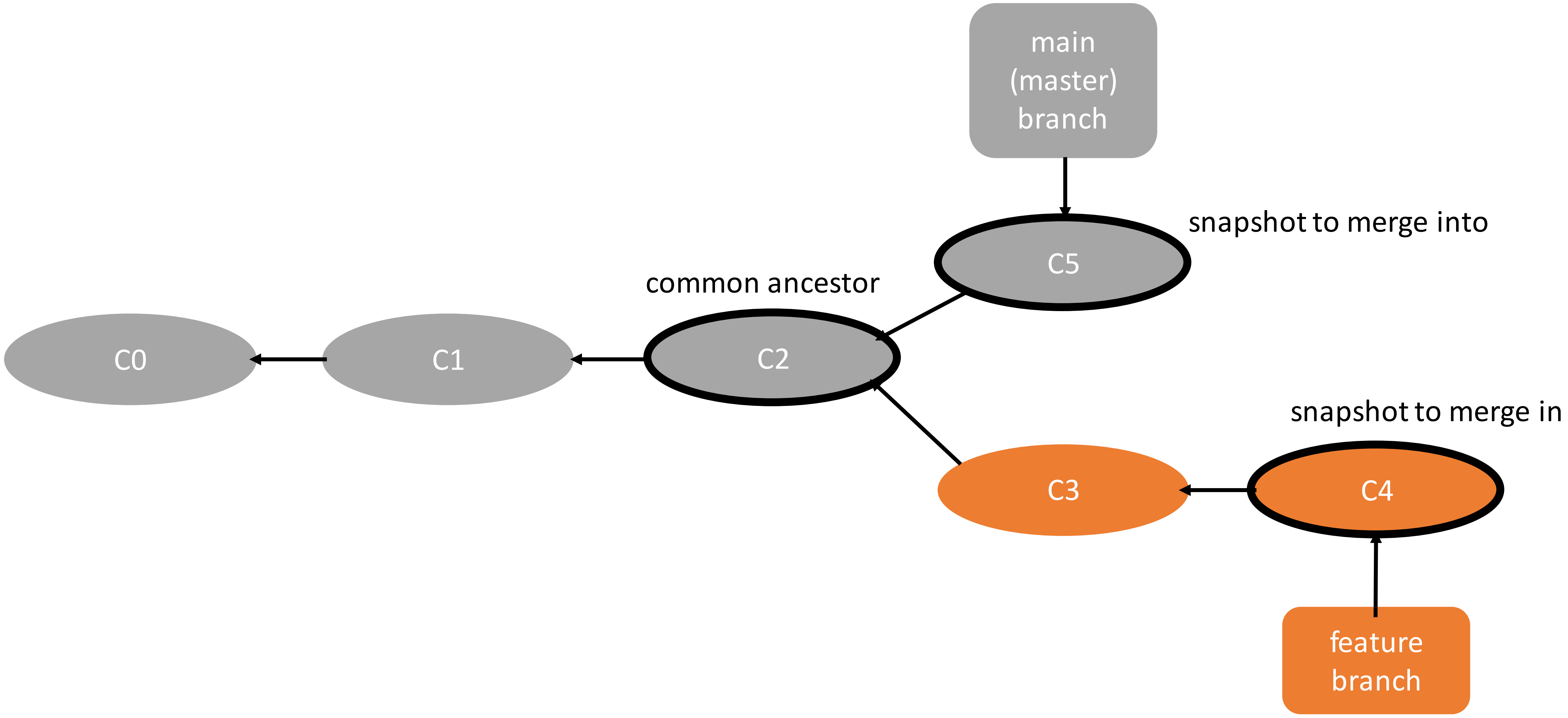


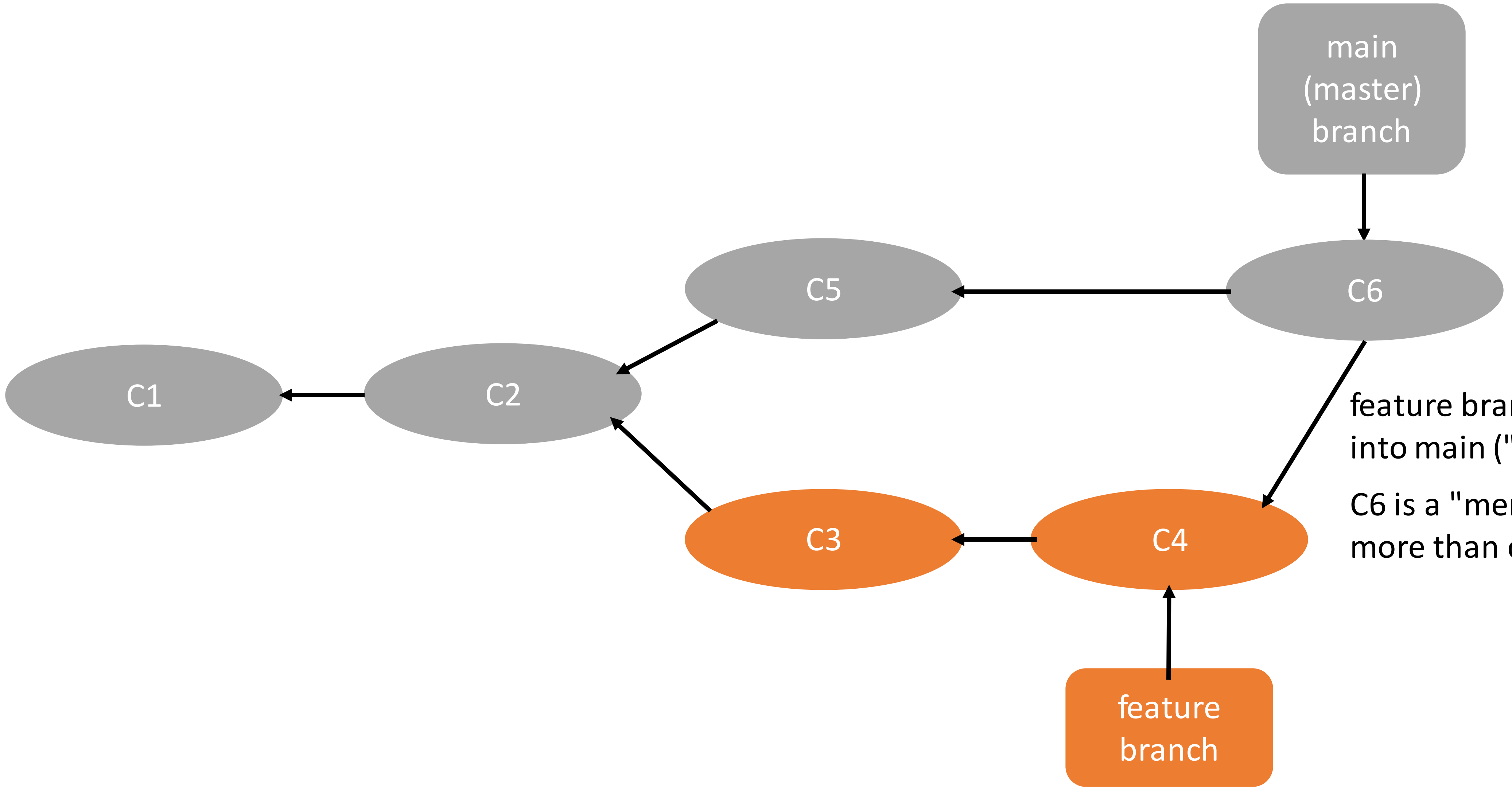
A bug was found in the program and fixed in the hotfix branch.



hotfix branch was merged into main ("fast-forward")


main (master) branch
C5
C0
C1
C2
C3
C4
feature branch


main (master) branch
snapshot to merge into
C5
common ancestor
C0
C1
C2
snapshot to merge in
C3
C4
feature branch



feature branch was merged into main ("recursive")

C6 is a "merge commit" - has more than one parent

git commits, branches and merges

- Not every automatic merge happens without conflict – sometimes you have to manually resolve the conflicts and commit
- To merge a branch using the command line: switch to main using `git checkout main` and then merge `git merge my-branch-to-merge`
- To delete a branch using the command line: `git branch -d my-branch-to-delete`
- To list your branches: `git branch` (the * marks which branch you have currently checked out)
- Display branches on remote: `git branch -r`
- Delete branch on remote: `git push origin –delete my-branch-to-delete`
- Other useful git commands: `git diff` and `git status`

git platforms on the web

- GitHub
- GitLab
- BitBucket
- ...

Differences: GitHub mostly focused on collaboration while GitLab emphasizes features and offers platform for web developers. Also: more private repos on GitLab (same for BitBucket).

We will use GitHub. Please create a user account on www.github.com

github

Using github in your browser does not require you to know all the git commands. Please complete this github learning lab:

https://lab.github.com/githubtraining/introduction-to-github

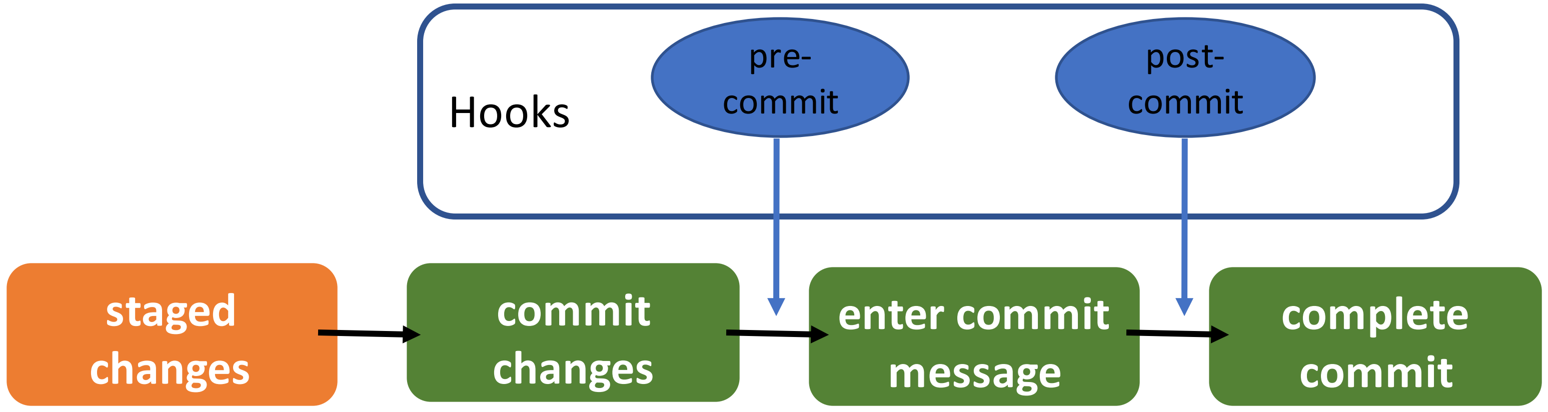
You will learn about commits, branches, merging and pull requests in this learning lab. Pull requests are very useful as they give you a chance to discuss your changes to the code with your collaborators.

Live lesson

- We will work on some code examples in git repositories during the live lessons. You will need git, python and jupyter notebooks installed on your computer. You will need numpy, pandas and seaborn.
- For windows users, it is easiest if you have ubuntu-subsystem installed. Please prepare this before the class starts. Please check the "preparing for the class" document.

git hooks

- hooks are scripts that run automatically upon a particular event
- hooks live in the `.git/hooks` directory (take a look at your .git/hooks directory to see which sample hooks are there. If you want to activate one of them, remove the "`.sample`" extension and make the script executable).
- hooks are local: they are not copied upon `git clone`! Maintaining hooks and ensuring everyone is using the same hooks is tricky.
- *Usually hooks are used to ensure clean commits* (i.e. ensure the commit message contains a certain amount of information, check for stylistic errors, etc).



git hooks

- Use to encourage certain commit policies

pre-commit

- inspect for styling or formatting errors
- prepare commit message

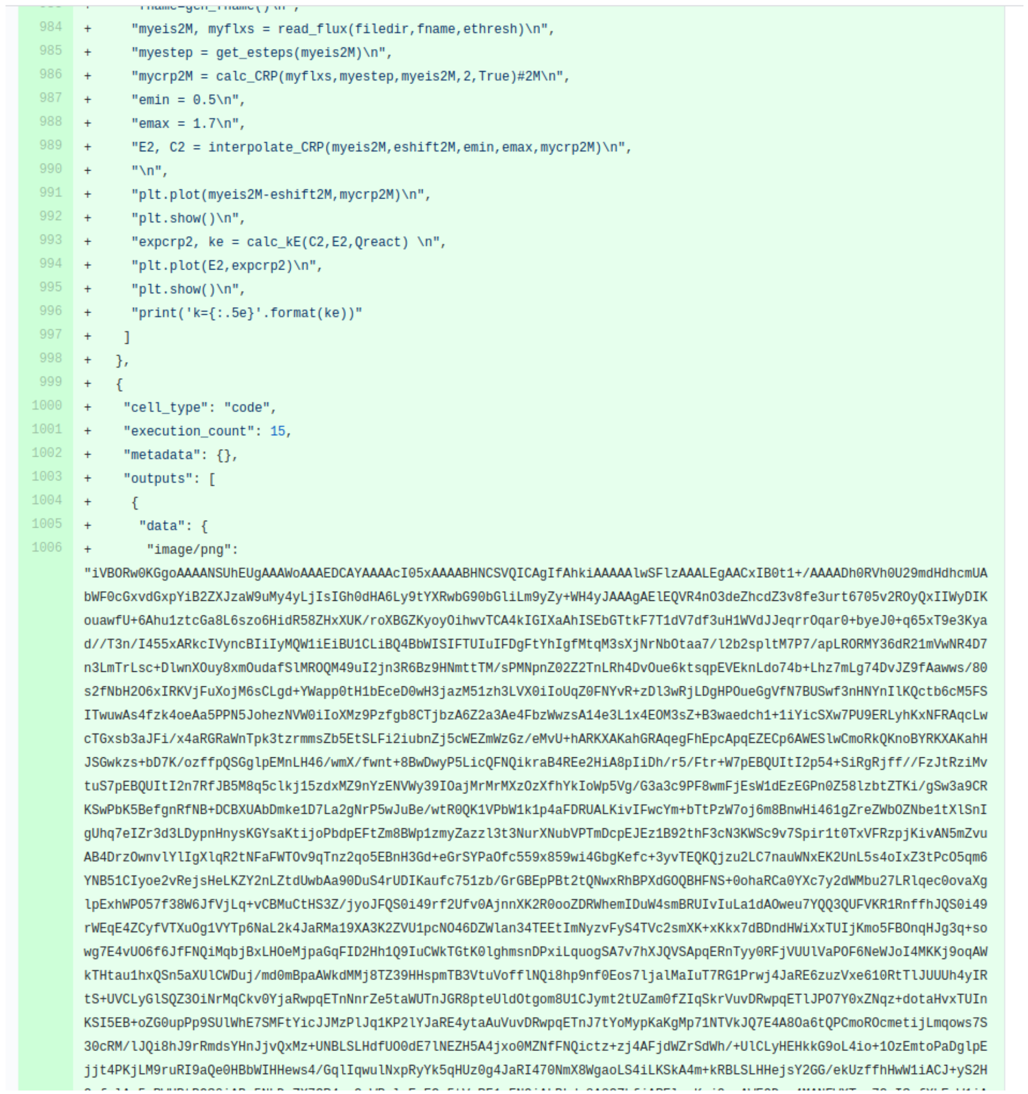
post-commit

- mostly for notification purposes

Jupyter notebooks on github

- We will work with jupyter notebooks today. The diffs that usually show in a pull request, and generally any diffs between commits are rendered as JSON, which is somewhat readable, but the images (binary) lead to problems. There is a variety of tools that address this problem; we will be using pre-commit hooks (nbstripout) to clear the notebooks before committing to the remote repository.

https://nextjournal.com/schmudde/how-to-version-control-jupyter



```
+     "myeis2M, myflxs = read_flux(filedir,fname,ethresh)\n",
+     "myestep = get_esteps(myeis2M)\n",
+     "mycrp2M = calc_CRP(myflxs,myestep,myeis2M,2,True)#2M\n",
+     "emin = 0.5\n",
+     "emax = 1.7\n",
+     "E2, C2 = interpolate_CRP(myeis2M,eshift2M,emin,emax,mycrp2M)\n",
+     "\n",
+     "plt.plot(myeis2M-eshift2M,mycrp2M)\n",
+     "plt.show()\n",
+     "expcrp2, ke = calc_kE(C2,E2,Qreact) \n",
+     "plt.plot(E2,expcrp2)\n",
+     "plt.show()\n",
+     "print('k={:.5e}'.format(ke))"
+    ]
+   },
+   {
+    "cell_type": "code",
+    "execution_count": 15,
+    "metadata": {},
+    "outputs": [
+     {
+      "data": {
+       "image/png":
```

```
"iVBORw0KGgoAAAANSUhEUgAAAWoAAAEDCAYAAAAcI05xAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUA
bWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3deZhcdZ3v8fe3urt6705v2ROyQxIIWyDIK
ouawfU+6Ahu1ztcGa8L6szo6HidR58ZHxXUK/roXBGZKyoyOihwvTCA4kIGIXaAhISEbGTtkF7T1dV7df3uH1WVdJJeqrrOqar0+byeJ0+q65xT9e3Kya
d//T3n/I455xARkcIVyncBIiIyMQW1iEiBU1CLiBQ4BbWISIFTUIuIFDgFtYhIgfMtqM3sXjNrNbOtaa7/l2b2spltM7P7/apLRORMY36dR21D7
n3LmTrLsc+DlwnXOuy8xmOudafSlMROQM49uI6jn3R6Bz9Z9HNmttTM/sPZZ02Z2TnLRh4DvvOue6ktsqpEVEknLdo74b+Lhz7mLg74DvDvAaaws/80
s2fNbH2O6xIRKVjFuXojM6M6sCLgd+YWAbwDP7BUSwf3nHNYnIlKQctb6cM5FS
ITwuwAs4fzk4oeAa5PPN5JohezNVW0iIoXMz9Pzfgb9CTjbzA6e3L1x4EOM3sZ+B3waedch1+iYicSXw7PU9ERLyhKxNFRAqcLw
cTGxsb3aJFi/x4aRGRaWnTpk1tzrmmsZb5EtSLFi1i48aNfrxs0vZ5dGnjTbRnQTJgbPyvEWzGFQIy5OshCExgbnaiFCLw2YJGaHmaYXZNKQX9kCkTtKkFsRhbM0WHrzwhOXqoBC2KKqkJJFg2IEyy2QNMjfYU0IckDx0Ih0N0vuq
JSGwkzs+bD7K/ozffpQSGglpEMnLH46/wmX/fwnt+8BwDwyP5LicQFNQikraB4REe2RFEe2HiA8pIiDh/r5/Ftr+W7pEBQUItI2p54+SiRFJtRziMv
tuS7pEBQUItI2n7RfJB5M8q5clkj15zdxMZ9nYzENVWy39IOajMJ7/gSw3a9CR
KSwPbK5BefgnRfNB+DCBXUAbDmke1D7La2gNrP5wJuBe/wtR0QK1VPbW1k1p4aFDRUALKivIFwcYm+bTtPzW7oj6m8nBwHi461e5gZreZQBTVOUZbqHN3mnS6WD1fmsLmQ6HcTUArSO0pfzXaM8pH1z7i8jCNz7Z
gUhq7eIZr3d3LDypnHnysKGYsaKtijoPbdpEFtZm8BWp1zmyZazzl3t3NurXNubVPTmDcpEJEz1B92thF3cN3KmWSc9v7Spir1t0TxVFRzpjKivAN5mZvu
AB4DrzOwnvlYlIgXlqR2tNFaVcqNu2dlzjxWv7RzAd+bdGnPQ/+P9r7Ja8OtO6Zi1iKWAN6ywlnsYDrnpB16+vJE61n0SjdLk2XX3ZzkNrGuxPfJuZ2d5Ki3r3MfrQtYPDPjFq4gsy58Bo2vs2cR
FDywBbB+NNQhsLWHfIQkqUm2ov8ctNH2v+WWIzPOMEwBUGGq8txKXjMzHbkuZ7rWZJ+MlKBpwAL4o71w5i0tyMpxsxPMvT0K4xwyqzYKdWN+Zdw7HAwbcDh6K25GeRLUFGAYzi+F5K/cXL8G5PM9l0zwrqjnHgkGLmkW7HZEzVMLJHfyFC3z5eh69jP5XO2kk8qCM8fO2QcjQVcx36TN3zjqyl6ZzHe6/VI2a6vxJxS1tnnntkVfM7a7xM
aHbHRkwDyYbE1g2tRIEB7xIAuZbhqF1kH2M9Adu71Fp95cWC9S8IpjOjzVZZ5KgA9ntzVdwgKgzmXh31hG2sYIEZjRDW3hxWu8fVGn8N6N7E4twHtUjTk1AOTVDpIjNw2yTSXXPbDbbdrkNFZ7mQ95Q8YjPrq6FyAvzynzFK2fsD3zmpcMSOZj45WX98ggn09vWoOqLXB7BcIw2o2Cmx9HdbtYjnBw
FQ7l2Nkf7Tv3zFF4OdYHx5Zc8bJE5yfDMvcuI8TtiwVyT4nr3S7f7fw9Xu3QfbeZp48P8IIbeVFsdj1Ap5hlVrm8kxdSTBQRp2ZpczqiQ0vS2gS9zuHxJMz1l9
rZRqTa8TJrMLqcIYpAS4x7p6X3DVq9fZ7sRd5nW5j6rkC7DYYD6ZnsXQ5L2jZR6B2X7kxRIu6Ezd5ZqS4l+r0tbFUp+HaXG5PnKy4WT4Nr7OqDDDXvR7O6ww1B1
NGyRVUG8ywE4wc3E2pS0y8ZrqO8zOzNrNWGJB8k7NCRqWL+nsoEXYGD0xjmCIYYFJ9a9a6p7cP4GmkaY4dmiqYq0JQs2y3XIW5ivXTTcl2B53Ui9s7xW3hRgvAZX3gSpOhDBH2JrSyRQhNTyhhEgzzqSGmJjAWT7JCY7uUhOZAgwYjuhBPb6yPgJd0KCbuvP7tnp5TjXfFHK0P1kRgH3QfAtpqbHHwaO9XWPxs8y0BfRtMuYW69GxJFDXXlJY1XF8BU1H7pNZ3XD6KhmBpWGQzWS
```

Use a package manager for your hooks

https://github.com/kynan/nbstripout

- We will use pre-commit for managing our hooks.

```
pip install pre-commit
```

```
pip install --upgrade nbstripout
```

- Add a .pre-commit-config.yaml file to your repository with the contents:

```
repos:
- repo: https://github.com/kynan/nbstripout
  rev: master
  hooks:
    - id: nbstripout
```

- Run `pre-commit install` to install the hook.
- Feel free to add additional useful hooks.

  https://github.com/pre-commit/pre-commit-hooks