

EPFL, CIVIL-127

Programming and software development for engineers



Questions, comments, feedback:

- Forum
- alok.menghrajani@epfl.ch or alexandre.alahi@epfl.ch
- Teaching Assistants
 - Valentin and Weijiang
- Student Assistants
 - Johanna and Mehdi

2D arrays: plain list

```
1  WIDTH, HEIGHT = (5, 4)
2  grid = [0] * (WIDTH * HEIGHT)
3
4  def offset(x, y):
5      if x < 0 or x >= WIDTH or y < 0 or y >= HEIGHT:
6          raise Exception("invalid offset", x, y)
7      return x * HEIGHT + y
8
9  def get(x, y):
10     return grid[offset(x, y)]
11
12  def set(x, y, v):
13     grid[offset(x, y)] = v
14
15  def printGrid():
16     for y in range(HEIGHT):
17         for x in range(WIDTH):
18             print(get(x, y), end=" ")
19         print()
20
21  set(4, 3, "A")
22  set(1, 2, "B")
23  printGrid()
```

2D arrays: plain list

Result:

```
0 0 0 0 0
0 0 0 0 0
0 B 0 0 0
0 0 0 0 A
```

2D arrays: list of lists

```
1 grid = [[0] * 4] * 5
2 print(grid)
```

Result:

```
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

2D arrays: list of lists

```
1  WIDTH, HEIGHT = (5, 4)
2  grid = [[0] * WIDTH] * HEIGHT
3
4  def get(x, y):
5      return grid[y][x]
6
7  def set(x, y, v):
8      grid[y][x] = v
9
10 def printGrid():
11     for y in range(HEIGHT):
12         for x in range(WIDTH):
13             print(get(x, y), end=" ")
14         print()
15
16 set(4, 3, "A")
17 set(1, 2, "B")
18 printGrid()
```

Don't do this! ⚠

2D arrays: list of lists

Result:

```
0 B 0 0 A
0 B 0 0 A
0 B 0 0 A
0 B 0 0 A
```

We have an incorrect behavior.

2D arrays: list of lists

```
1  WIDTH, HEIGHT = (5, 4)
2  grid = [[0] * WIDTH for _ in range(HEIGHT)]
3
4  def get(x, y):
5      return grid[y][x]
6
7  def set(x, y, v):
8      grid[y][x] = v
9
10 def printGrid():
11     for y in range(HEIGHT):
12         for x in range(WIDTH):
13             print(get(x, y), end=" ")
14         print()
15
16 set(4, 3, "A")
17 set(1, 2, "B")
18 printGrid()
```

2D arrays: list of lists

Result:

```
0 0 0 0 0
0 0 0 0 0
0 B 0 0 0
0 0 0 0 A
```

2D arrays: dict

```
1  WIDTH, HEIGHT = (5, 4)
2  grid = {}
3
4  def get(x, y):
5      if (x, y) not in grid:
6          return 0
7      return grid[(x, y)]
8
9  def set(x, y, v):
10     grid[(x, y)] = v
11
12  def printGrid():
13     for y in range(HEIGHT):
14         for x in range(WIDTH):
15             print(get(x, y), end=" ")
16         print()
17
18  set(4, 3, "A")
19  set(1, 2, "B")
20  printGrid()
```

2D arrays: dict

Result:

```
0 0 0 0 0
0 0 0 0 0
0 B 0 0 0
0 0 0 0 A
```

pip install <package>

- Do leverage existing libraries
- <https://docs.python.org/3.12/library/index.html>
 - [pprint](#)
 - [itertools](#)
 - [functools](#)
 - [datetime](#)
 - ...
- [colorful](#) and [colorama](#)
- [numpy](#)
- [scipy](#)
- ...

git

git ≠ GitHub

- git
 - open source
 - command line tool
 - 3rd party graphical interfaces
- GitHub
 - Owned by Microsoft
 - Collaboration tool, builds on top of git
 - Loads of features (pull requests, issue tracking, forums, wiki, pages, CI/CD, etc.)

What is git?

- distributed version control system
- enables collaboration
- enables iterative changes and rollbacks
- resolves conflicts

Resources

- git man pages
- Pro Git (free book): <https://git-scm.com/book/en/v2>
- Lots of free resources and tutorials

Creating a repo

```
1 $ mkdir my-first-git-repo
2 $ cd my-first-git-repo
3 [my-first-git-repo]$ git init
4 Initialized empty Git repository in /[...]/my-first-git-repo/.git/
```

working directory vs staging area vs tree

- working directory → files on your disk
- staging area → changes you want to include in your next commit
- tree → store of all the commits

An initial file

```
1  [my-first-git-repo]$ echo "Code flows like rivers of change," > haiku.txt
2  [my-first-git-repo]$ ls
3  haiku.txt
4  [my-first-git-repo]$ git status
5  On branch main
6
7  No commits yet
8
9  Untracked files:
10   (use "git add <file>..." to include in what will be committed)
11   haiku.txt
12
13  nothing added to commit but untracked files present (use "git add" to track)
```

Staging

```
1  [my-first-git-repo]$ git add haiku.txt
2  [my-first-git-repo]$ git status
3  On branch main
4
5  No commits yet
6
7  Changes to be committed:
8    (use "git rm --cached <file>..." to unstage)
9    new file:   haiku.txt
```

Initial commit

```
1 Initial commit
2 # Please enter the commit message for your changes. Lines starting
3 # with '#' will be ignored, and an empty message aborts the commit.
4 #
5 # On branch main
6 #
7 # Initial commit
8 #
9 # Changes to be committed:
10 #       new file:   haiku.txt
11 #
```

Initial commit

```
1  [my-first-git-repo]$ git commit
2  [main (root-commit) 17d2441] Initial commit
3    1 file changed, 1 insertion(+)
4    create mode 100644 haiku.txt
5  [my-first-git-repo]$ ls
6  haiku.txt
7  [my-first-git-repo]$ git status
8  On branch main
9  nothing to commit, working tree clean
10 [my-first-git-repo]$ git log
11 commit 17d2441d2c49913d9f022dd49f727f70010ce0c1 (HEAD -> main)
12 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
13 Date:   Fri Feb 7 10:42:24 2025 +0100
14
15     Initial commit
```

git hashes

- Every object in the tree is identified by a hash
- An object's hash is (conceptually):

```
hash(diff + parent hashes + commit message, author, time, ...)>
```

- where `hash()` is a one way function
 - and diff is the content change
- git is fast at showing you the state of the repo for a given hash or computing the delta between two hashes.

Making changes

```
1  [my-first-git-repo]$ echo "Branches diverge, merge," >> haiku.txt
2  [my-first-git-repo]$ echo "History preserved" >> haiku.txt
3  [my-first-git-repo]$ echo "Moonlight dances on the soft clouds" > haiku2.txt
4  [my-first-git-repo]$ git status
5  On branch main
6  Changes not staged for commit:
7      (use "git add <file>..." to update what will be committed)
8      (use "git restore <file>..." to discard changes in working directory)
9      modified:   haiku.txt
10
11  Untracked files:
12      (use "git add <file>..." to include in what will be committed)
13      haiku2.txt
14
15  no changes added to commit (use "git add" and/or "git commit -a")
16  [my-first-git-repo]$ git diff
17  diff --git a/haiku.txt b/haiku.txt
18  index bd34e77..e95b648 100644
19  --- a/haiku.txt
20  +++ b/haiku.txt
21  @@ -1,3 @@
22     Code flows like rivers of change,
23  +Branches diverge, merge,
24  +History preserved
```

Making changes

```
1 [my-first-git-repo]$ git add haiku.txt
2 [my-first-git-repo]$ git add haiku2.txt
3 [my-first-git-repo]$ git commit -m 'Finish both haiku'
4 [main 9cc0f57] Finish both haiku
5   2 files changed, 3 insertions(+)
6   create mode 100644 haiku2.txt
7 [my-first-git-repo]$ git log
8 commit 9cc0f57593d59a1e8faec42e6be10be8b3fd6c2c (HEAD -> main)
9 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
10 Date:   Sat Feb 8 15:35:38 2025 +0100
11
12     Finish both haiku
13
14 commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
15 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
16 Date:   Fri Feb 7 10:42:24 2025 +0100
17
18     Initial commit
```

git log --oneline

```
1 [my-first-git-repo]$ git log --oneline
2 9cc0f57 (HEAD -> main) Finish both haiku
3 17d2441 Initial commit
```

git log <filename>

```
1 [my-first-git-repo]$ git log haiku2.txt
2 commit 9cc0f57593d59a1e8faec42e6be10be8b3fd6c2c (HEAD -> main)
3 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
4 Date: Sat Feb 8 15:35:38 2025 +0100
5
6     Finish both haiku
```

git blame

```
1  [my-first-git-repo]$ git blame haiku.txt
2  ^17d2441 (Alok Menghrajani 2025-02-07 10:42:24 +0100 1) Code flows like rivers of change,
3  9cc0f575 (Alok Menghrajani 2025-02-08 15:35:38 +0100 2) Branches diverge, merge,
4  9cc0f575 (Alok Menghrajani 2025-02-08 15:35:38 +0100 3) History preserved
```

git show <ref>

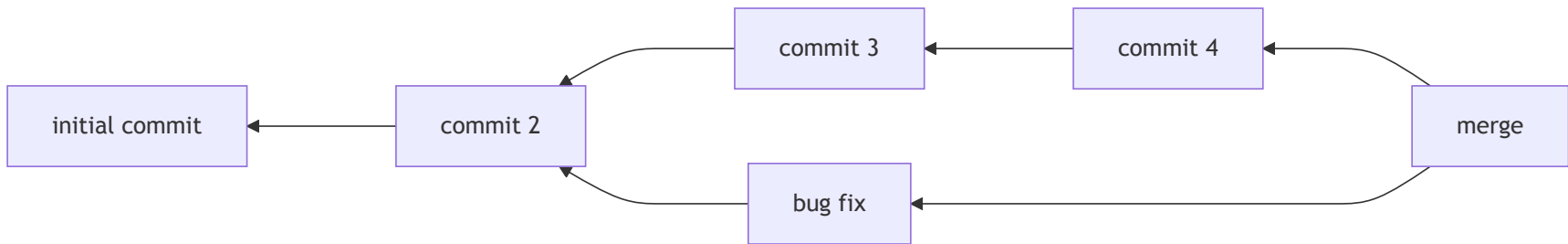
```
1 [my-first-git-repo]$ git show 17d2441d2c49913d9f022dd49f727f70010ce0c1
2 commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
3 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
4 Date:   Fri Feb 7 10:42:24 2025 +0100
5
6     Initial commit
7
8 diff --git a/haiku.txt b/haiku.txt
9 new file mode 100644
10 index 0000000..bd34e77
11 --- /dev/null
12 +++ b/haiku.txt
13 @@ -0,0 +1 @@
14 +Code flows like rivers of change,
```

git show <short hash>

```
1  [my-first-git-repo]$ git show 17d2
2  commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
3  Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
4  Date:   Fri Feb 7 10:42:24 2025 +0100
5
6      Initial commit
7
8  diff --git a/haiku.txt b/haiku.txt
9  new file mode 100644
10 index 0000000..bd34e77
11 --- /dev/null
12 +++ b/haiku.txt
13 @@ -0,0 +1 @@
14 +Code flows like rivers of change,
```

branches and merges

Directed Acyclic Graph



Create a branch

```
1  [my-first-git-repo]$ git switch -c fix-typo
2  Switched to a new branch 'fix-typo'
3  [my-first-git-repo]$ git branch
4  * fix-typo
5     main
```

Add commits to the new branch

```
1  [my-first-git-repo]$ sed -i '' -e 's/preserved/preserved./' haiku.txt
2  [my-first-git-repo]$ git diff
3  diff --git a/haiku.txt b/haiku.txt
4  index e95b648..d640618 100644
5  --- a/haiku.txt
6  +++ b/haiku.txt
7  @@ -1,3 +1,3 @@
8   Code flows like rivers of change,
9   Branches diverge, merge,
10  -History preserved
11  +History preserved.
12  [my-first-git-repo]$ git add haiku.txt
13  [my-first-git-repo]$ git commit -m 'Add missing punctuation'
14  [fix-typo 72ac719] Add missing punctuation
15  1 file changed, 1 insertion(+), 1 deletion(-)
```

Merge the branch

(happy case)

```
1  [my-first-git-repo]$ git switch main
2  Switched to branch 'main'
3  [my-first-git-repo]$ git merge --no-ff --no-edit fix-typo
4  Merge made by the 'ort' strategy.
5    haiku.txt | 2 + -
6    1 file changed, 1 insertion(+), 1 deletion(-)
7  [my-first-git-repo]$ git branch -d fix-typo
8  Deleted branch fix-typo (was 72ac719).
```

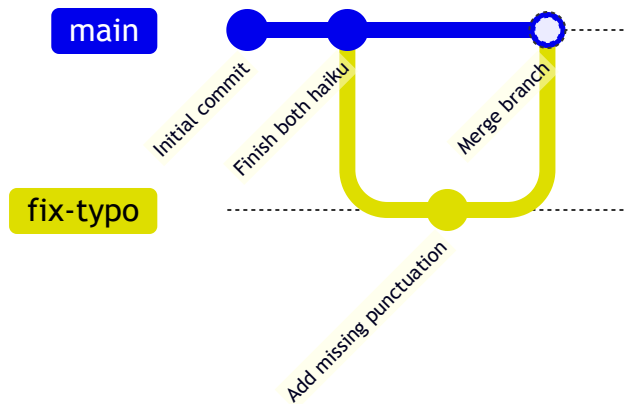
Merge the branch

```
1  [my-first-git-repo]$ git log
2  commit 23642369645a9a71983159d0c305abc44d5b9fd1 (HEAD -> main)
3  Merge: 9cc0f57 72ac719
4  Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
5  Date:   Sat Feb 8 16:22:37 2025 +0100
6
7      Merge branch 'fix-typo'
8
9  commit 72ac719503bc0dae974fb66d1e918a9d24daa5df
10 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
11 Date:   Sat Feb 8 16:21:25 2025 +0100
12
13      Add missing punctuation
14
15  commit 9cc0f57593d59a1e8faec42e6be10be8b3fd6c2c
16 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
17 Date:   Sat Feb 8 15:35:38 2025 +0100
18
19      Finish both haiku
20
21  commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
22 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
23 Date:   Fri Feb 7 10:42:24 2025 +0100
24
25      Initial commit
```

Merge the branch

```
1 [my-first-git-repo]$ git log --graph
2 *   commit 23642369645a9a71983159d0c305abc44d5b9fd1 (HEAD -> main)
3 | \   Merge: 9cc0f57 72ac719
4 | |   Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
5 | |   Date:    Sat Feb 8 16:22:37 2025 +0100
6 | |
7 | |       Merge branch 'fix-typo'
8 | |
9 | *   commit 72ac719503bc0dae974fb66d1e918a9d24daa5df
10 | /   Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
11 | |   Date:    Sat Feb 8 16:21:25 2025 +0100
12 | |
13 | |       Add missing punctuation
14 | |
15 *   commit 9cc0f57593d59a1e8faec42e6be10be8b3fd6c2c
16 |   Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
17 |   Date:    Sat Feb 8 15:35:38 2025 +0100
18 |
19 |       Finish both haiku
20 |
21 *   commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
22 |   Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
23 |   Date:    Fri Feb 7 10:42:24 2025 +0100
24 |
25 |       Initial commit
```

Merge the branch



HEAD

```
1 [my-first-git-repo]$ git show HEAD
2 commit 23642369645a9a71983159d0c305abc44d5b9fd1 (HEAD -> main)
3 Merge: 9cc0f57 72ac719
4 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
5 Date: Sat Feb 8 16:22:37 2025 +0100
6
7 Merge branch 'fix-typo'
8 [my-first-git-repo]$ git show HEAD~2
9 commit 17d2441d2c49913d9f022dd49f727f70010ce0c1
10 Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
11 Date: Fri Feb 7 10:42:24 2025 +0100
12
13 Initial commit
14
15 diff --git a/haiku.txt b/haiku.txt
16 new file mode 100644
17 index 0000000..bd34e77
18 --- /dev/null
19 +++ b/haiku.txt
20 @@ -0,0 +1 @@
21 +Code flows like rivers of change,
```


Merge with conflicts

- `brA` and `brB` branch off `main`
- you commit changes to `brA`
- you commit changes to `brB`
- you merge `brA` into `main`
 - this merge will not conflict
- you merge `brB` into `main`
 - this merge will conflict

Automatic conflict resolution

```
1  [my-first-git-repo]$ git switch -c brB
2  Switched to a new branch 'brB'
3  [my-first-git-repo]$ git switch -c brA
4  Switched to a new branch 'brA'
5  [my-first-git-repo]$ sed -i '' -e 's/Code/git/' haiku.txt
6  [my-first-git-repo]$ git commit -a -m "Replace 'Code' with 'git'"
7  [brA ae8ec80] Replace 'Code' with 'git'
8    1 file changed, 1 insertion(+), 1 deletion(-)
9  [my-first-git-repo]$ git switch brB
10 Switched to branch 'brB'
11 [my-first-git-repo]$ sed -i '' -e 's/History/Everything/' haiku.txt
12 [my-first-git-repo]$ git commit -a -m "Replace 'History' with 'Everything'"
13 [brB 2cfc68b] Replace 'History' with 'Everything'
14    1 file changed, 1 insertion(+), 1 deletion(-)
15 [my-first-git-repo]$ git switch main
16 Switched to branch 'main'
```

Automatic conflict resolution

```
1  [my-first-git-repo]$ git merge --no-ff --no-edit brA
2  Merge made by the 'ort' strategy.
3  haiku.txt | 2 + -
4  1 file changed, 1 insertion(+), 1 deletion(-)
5  [my-first-git-repo]$ git merge --no-ff --no-edit brB
6  Auto-merging haiku.txt
7  Merge made by the 'ort' strategy.
8  haiku.txt | 2 + -
9  1 file changed, 1 insertion(+), 1 deletion(-)
10 [my-first-git-repo]$ cat haiku.txt
11 git flows like rivers of change,
12 Branches diverge, merge,
13 Everything preserved.
14 [my-first-git-repo]$ git branch -d brA
15 Deleted branch brA (was ae8ec80).
16 [my-first-git-repo]$ git branch -d brB
17 Deleted branch brB (was 2cfc68b).
```

Manual conflict resolution

```
1  [my-first-git-repo]$ git switch -c brB
2  Switched to a new branch 'brB'
3  [my-first-git-repo]$ git switch -c brA
4  Switched to a new branch 'brA'
5  [my-first-git-repo]$ sed -i '' -e 's/git/Life/' haiku.txt
6  [my-first-git-repo]$ git commit -a -m "Replace 'git' with 'Life'"
7  [brA 94c0e50] Replace 'git' with 'Life'
8      1 file changed, 1 insertion(+), 1 deletion(-)
9  [my-first-git-repo]$ git switch brB
10 Switched to branch 'brB'
11 [my-first-git-repo]$ sed -i '' -e 's/change,/change!/' haiku.txt
12 [my-first-git-repo]$ git commit -a -m "Change punctuation"
13 [brB 63848e8] Change punctuation
14      1 file changed, 1 insertion(+), 1 deletion(-)
15 [my-first-git-repo]$ git switch main
16 Switched to branch 'main'
```

Manual conflict resolution

```
1  [my-first-git-repo]$ git merge --no-ff --no-edit brA
2  Merge made by the 'ort' strategy.
3  haiku.txt | 2 +-
4  1 file changed, 1 insertion(+), 1 deletion(-)
5  [my-first-git-repo]$ git merge --no-ff --no-edit brB
6  Auto-merging haiku.txt
7  CONFLICT (content): Merge conflict in haiku.txt
8  Automatic merge failed; fix conflicts and then commit the result.
9  [my-first-git-repo]$ head -n1 haiku.txt
10 <<<<<< HEAD
11 Life flows like rivers of change,
12 =====
13 git flows like rivers of change!
14 >>>>>> brB
15 [my-first-git-repo]$ git restore --ours haiku.txt
16 [my-first-git-repo]$ head -n1 haiku.txt
17 Life flows like rivers of change,
18 [my-first-git-repo]$ git restore --theirs haiku.txt
19 [my-first-git-repo]$ head -n1 haiku.txt
20 git flows like rivers of change!
```

Manual conflict resolution

```
1 [my-first-git-repo]$ sed -i '' '1s/./Life flows like rivers of change!/' haiku.txt
2 [my-first-git-repo]$ git add haiku.txt
3 [my-first-git-repo]$ git commit -m 'Manually merged brB'
4 [main c7f3e24] Manually merged brB
5 [my-first-git-repo]$ git log --oneline --topo-order HEAD~2..HEAD
6 8982205 (HEAD -> main) Manually merged brB
7 63848e8 (brB) Change punctuation
8 eab1ee3 Merge branch 'brA'
9 94c0e50 (brA) Replace 'git' with 'Life'
```

Manual conflict resolution

```
1  [my-first-git-repo]$ git show 898220580ee69f4535131210861c3d4ae73271df
2  commit 898220580ee69f4535131210861c3d4ae73271df (HEAD -> main)
3  Merge: eab1ee3 63848e8
4  Author: Alok Menghrajani <441307+alokmenghrajani@users.noreply.github.com>
5  Date:   Sat Feb 8 17:22:34 2025 +0100
6
7      Manually merged brB
8
9  diff --cc haiku.txt
10 index a2b1fae,6052095..96082fb
11 --- a/haiku.txt
12 +++ b/haiku.txt
13 @@@ -1,3 -1,3 +1,3 @@@
14 - Life flows like rivers of change,
15 -git flows like rivers of change!
16 ++Life flows like rivers of change!
17   Branches diverge, merge,
18   Everything preserved.
```

git: more commands

Working with remotes

- `git remote -v`
- `git fetch -p`
- `git branch -a`
- `git push origin HEAD`

git commit --amend

⚠️ rewriting history

- Mutate most recent commit
- Edit a commit message

git reflog

- Operate on the local history (reference logs)

git cherry-pick

Move HEAD

⚠ uncommitted changes can be lost

- `git reset --soft`
- `git reset --hard`
- `git restore <ref>`
- `git restore <ref> <files>`

Rebase

- `git rebase <ref>`
- `git rebase -i <ref>`

git add

- `git add -p`
- `git add .`
- `git commit -a`

git revert <commit>


```
git rm <filename>
```

```
git mv <filename>
```

Searching

- `git log -S <string>`
- `git grep <string>`

git tag

- A shareable bookmark
- `git tag -a v1.1 -m "say something about this version..."`
- `git tag` to list tags

Stash

- `git stash`
- `git stash list`
- `git stash pop`

Tips for authoring commits

- Each commit should be as small as possible
- Single cohesive idea
- Split large commits into small commits by dividing large problems into smaller pieces

Tips for writing commit messages

- Explain the gist of the change in the title
- Explain the reason for the change in the description
- Any other information which might be useful to you or someone else looking at the commit out of context