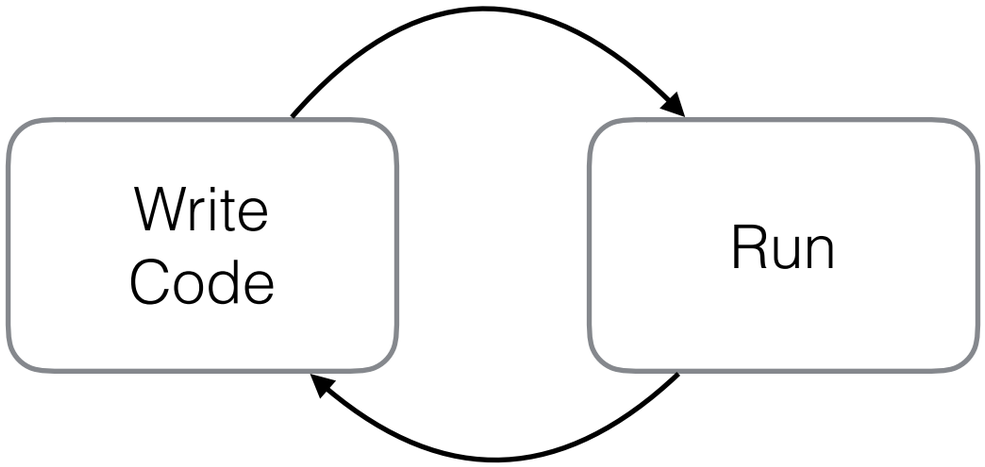
Lecture 4

Implementation

3 February 2017


Write
Code
Run

- Your code is an important communication tool that tells other developers what you are doing.
- Comments should supplement your code as a communication tool (e.g., the rationale)

```
float toc (float x)
{
    // substract 32 from x, multiple
    // the result by 5, then divide by 9.
    return 0.5555*(x-32);
    // return the result
}
```

```
float farenheit_to_celcius (float temperature_in_farenheit)
{
    float temperature_in_celcius;
    temperature_in_celcius = 0.5555*(temperature_in_farenheit-32);

    return temperature_in_celcius;
}
```

```
float farenheit_to_celcius (float temperature_in_farenheit)
{
    float temperature_in_celcius;

    // C = 5*(F-32)/9
    // Uses 0.5555 as an approximation to 5/9 since division is
    // expensive and we do not need high accuracy
    temperature_in_celcius = 0.5555*(temperature_in_farenheit-32);

    return temperature_in_celcius;
}
```

- Don’t try to be terse
- Make your code self-explanatory
- Comments are for high-level descriptions, rationales, API docs
- Use English and don’t be afraid of long names

Week 2 [Jan 16]

Part 1 - Good code, bad code: Toward production quality code

[Slides]

Overview: Good code conforms to an accepted coding standard and follows good coding practices such as avoiding 'magic numbers'. Adding lengthy comments does not necessarily produce good code. One practice that is hard to learn but very important is to stay within the same level of abstraction inside a method.

Topics: code quality

Review your CS2103 Materials Again

Reso

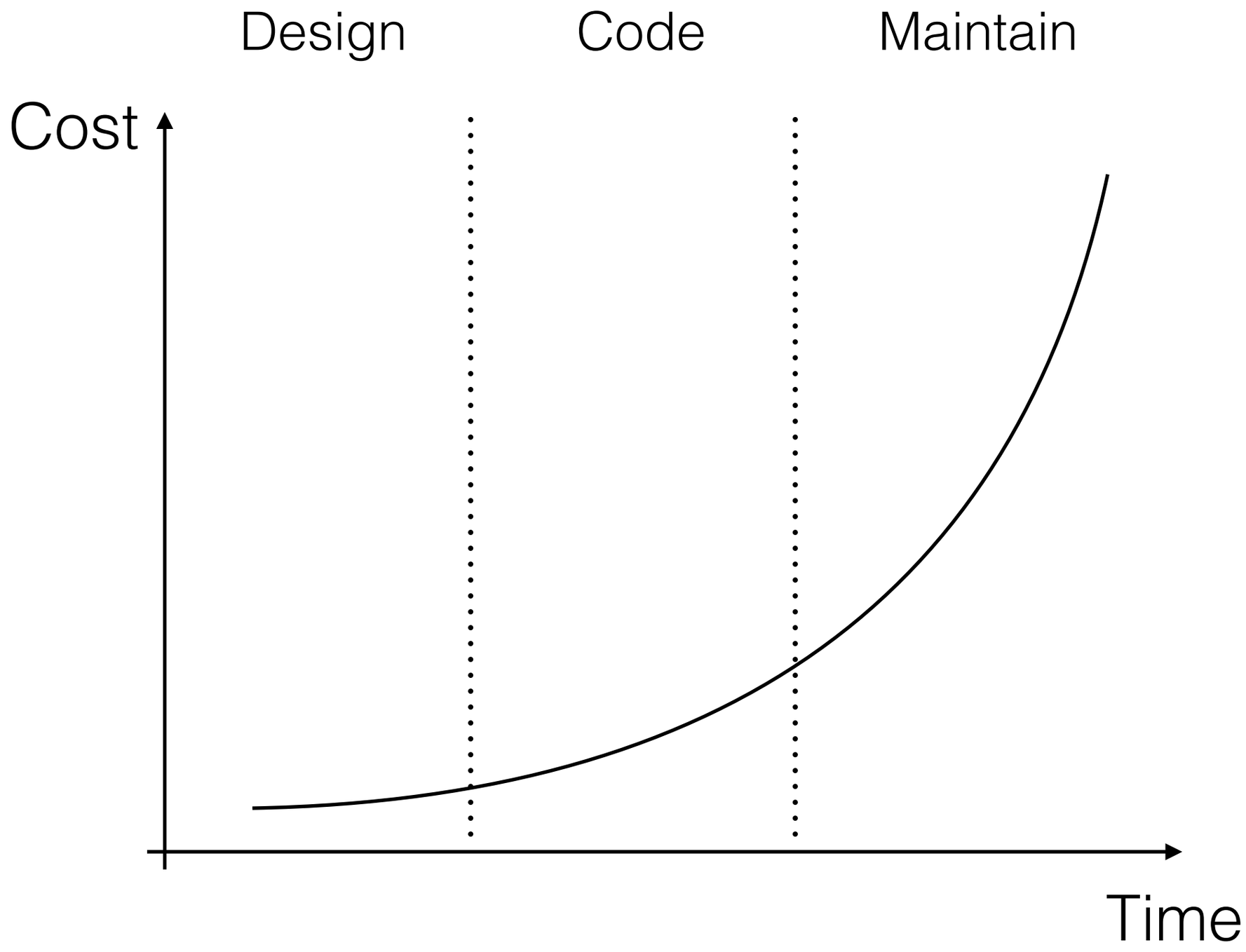
- Interesting side reading
  - [Article] Wikipedia entry on many styles of code indentation
  - [Blog post] Learning from Apple's #gotofail Security Bug - How following a coding style could have prevented a major security bug.
  - [Blog post] Why coding style matters - A view from a practicing software developer
  - [Blog post] Understanding your own code - Blogger Eli Bendersky says 'Code that isn't readable is as bad as, or worse than code that doesn't work'
  - [Web article] Top 9 qualities of clean code
  - [Developer opinions] What makes a good programmer good | What makes a good programmer | How do I train myself to code faster and with fewer bugs? | What is 'good code'?
- Humor: The six most common species of code (checkout number 6!), Commenting style, XKCD on code quality

- Make sure your code is
  - correct,
  - easy to change/extend,
  - readable by human, and
  - a pleasure to maintain.

when I have to edit legacy code..

“Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.”

— John Woods, 1991 (?)



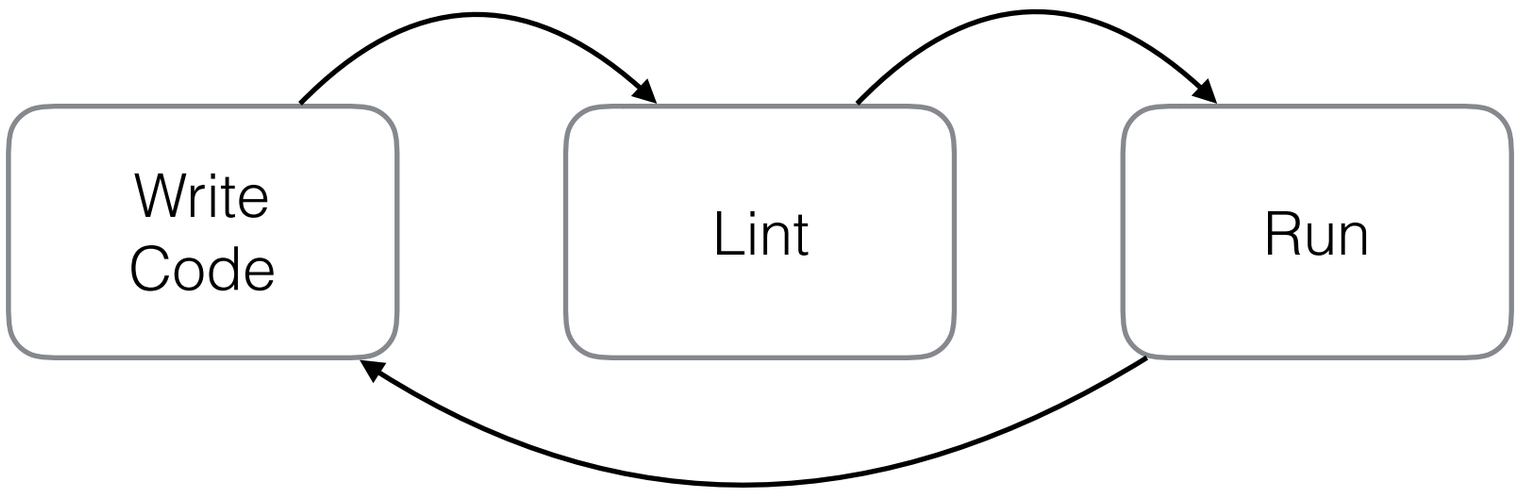
Boehm’s Curve

- Think of your code on GitHub as a public, permanent, show case of your ability as a software engineer.

- Always strive to improve the quality of your code:

  - read

  - practice


Write
Code
Lint
Run

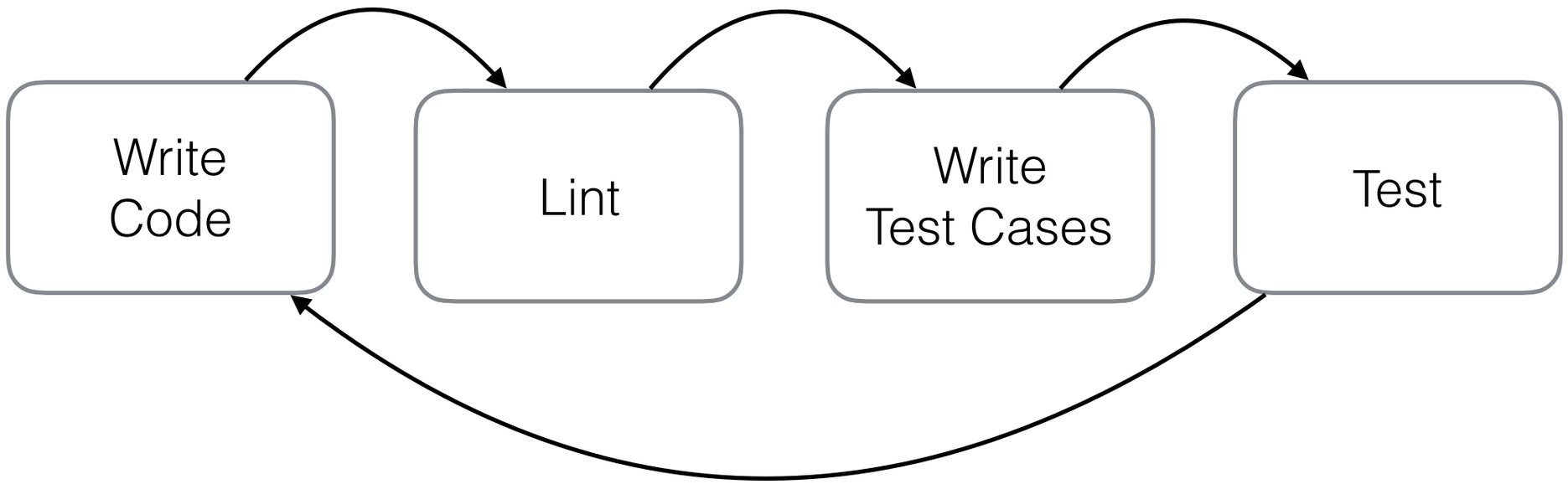
Linting

- Static analysis of code for potential error
- Some checks for styles as well
- There is at least one lint tool for every programming language I use.

Formatter

- Re-indent your code consistently based on a style
- Your team should agree on a coding style
- Don’t underestimate the importance of indentation.

```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer signedParams,
                                 uint8_t *signature, UInt16 signatureLen)
{
    OSStatus        err;
    SSLBuffer       hashOut, hashCtx, clientRandom, serverRandom;
    uint8_t         hashes[SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN];

                ■■■■

    if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
        goto fail;
        goto fail;
    if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
        goto fail;

        err = sslRawVerify(ctx,
                           ctx->peerPubKey,
                           dataToSign,                          /* plaintext */
                           dataToSignLen,                   /* plaintext length */
                           signature,
                           signatureLen);
        if(err) {
                sslErrorLog("SSLDecodeSignedServerKeyExchange: sslRawVerify "
                    "returned %d\n", (int)err);
                goto fail;
        }

fail:
    SSLFreeBuffer(&signedHashes);
    SSLFreeBuffer(&hashCtx);
    return err;

}
```

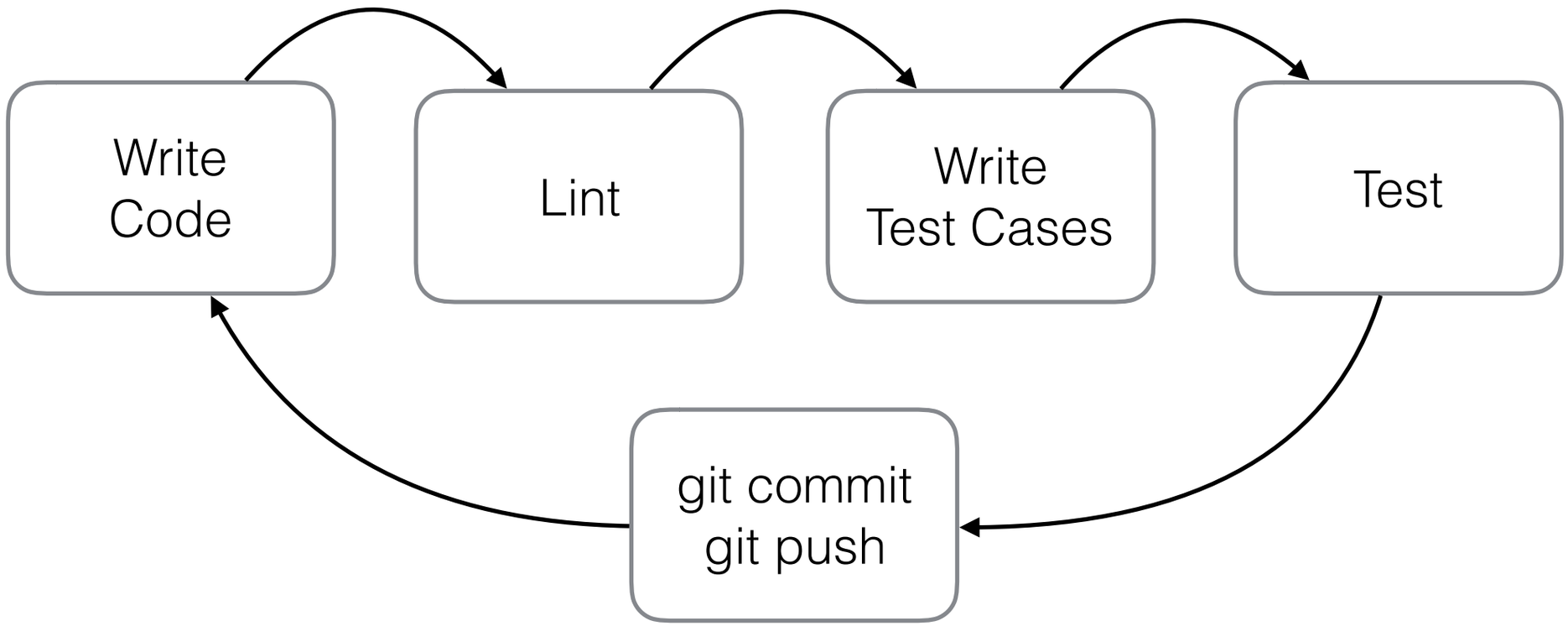

Write
Code
Lint
Write
Test Cases
Test

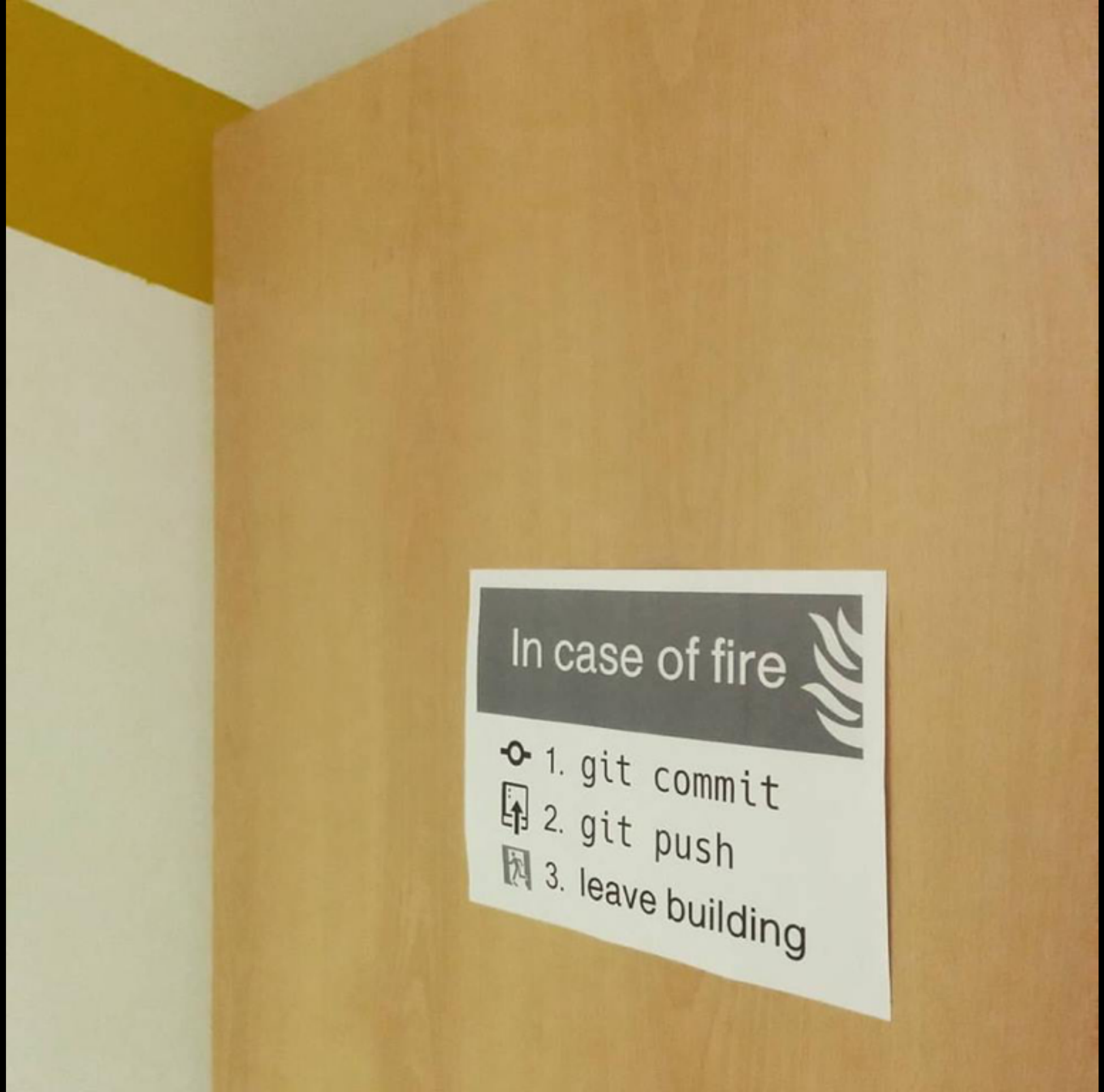
Testing Tips

- The “Main Success Scenario” is easy
- Testing corner / unexpected cases is harder but more important

Testing Tips

- Automate as much as possible
- Isolate as much as possible (using pretend objects)


Write
Code
Lint
Write
Test Cases
Test
git commit
git push


In case of fire
1. git commit
2. git push
3. leave building

- Commit message is another important communication tool
- Be specific about what has changed and why

Commits on Jan 25, 2017


UPDATED calendar status strings ...
weitsang committed 9 days ago
bbe5f3f


Removed hardcoded calendar
weitsang committed 9 days ago
da49694


ADDED storing of calendar information upon success
weitsang committed 9 days ago
2f0c013


FIXED added auth_token to MISTService APIs
weitsang committed 9 days ago
78d841d


FIXED added auth_token to all API calls ...
weitsang committed 9 days ago
90f6f56


FIXED task handler when loading of records succeed
weitsang committed 9 days ago
6deb188


FIXED uninitialized recordIds array
weitsang committed 9 days ago
ea2c23f


FIXED inconsistency between json names ...
weitsang committed 9 days ago
d39ba93


FIXED bug with uninitialized pairs
weitsang committed 9 days ago
097759d

1-25
committed 9 days ago
837eb87

Commits on Jan 23, 2017

11:45
committed 11 days ago
601cd58

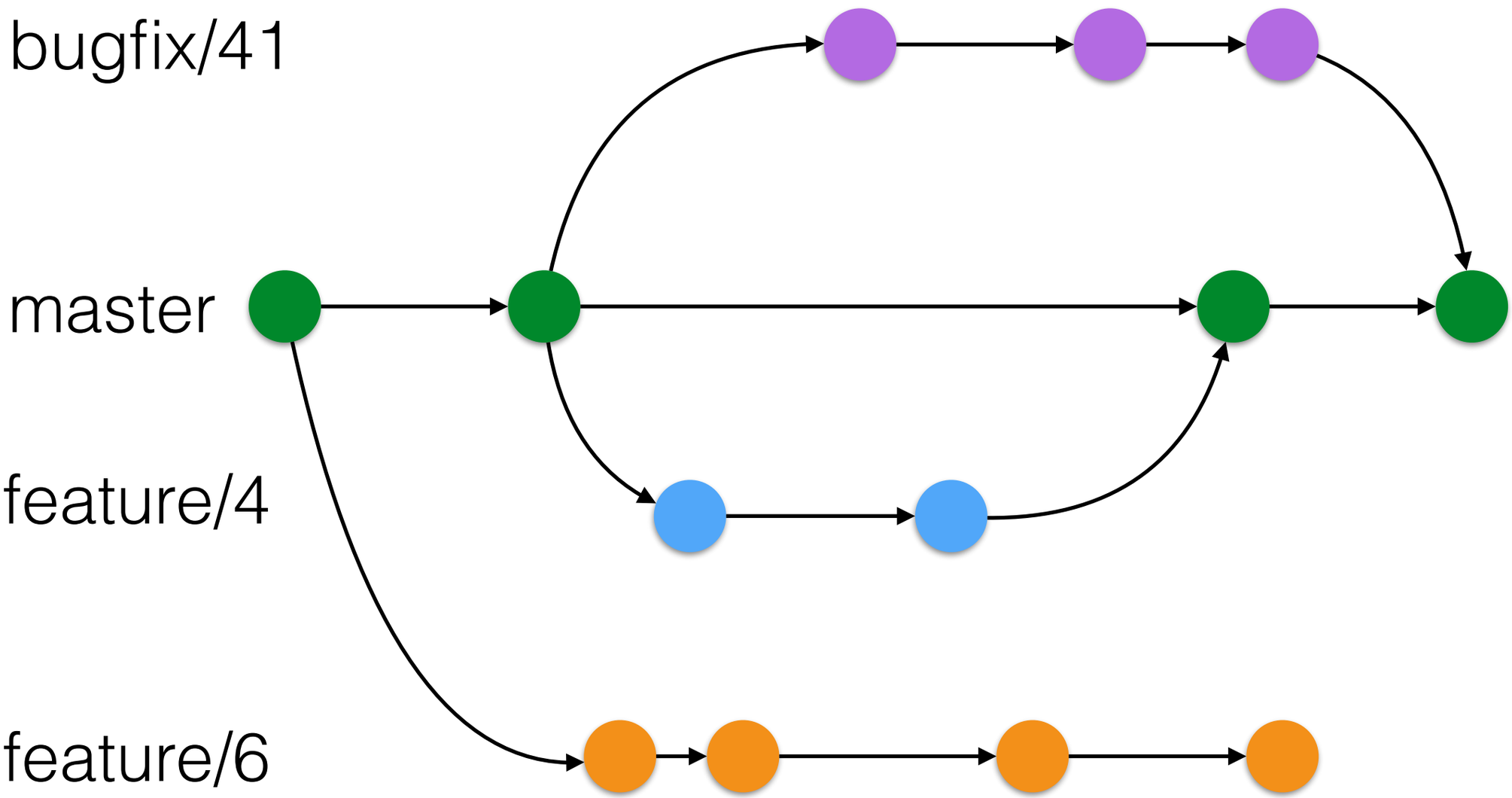
Merge branch 'master' of https://github.com/nus-mmsys/mist-android
committed 11 days ago
b3cf633

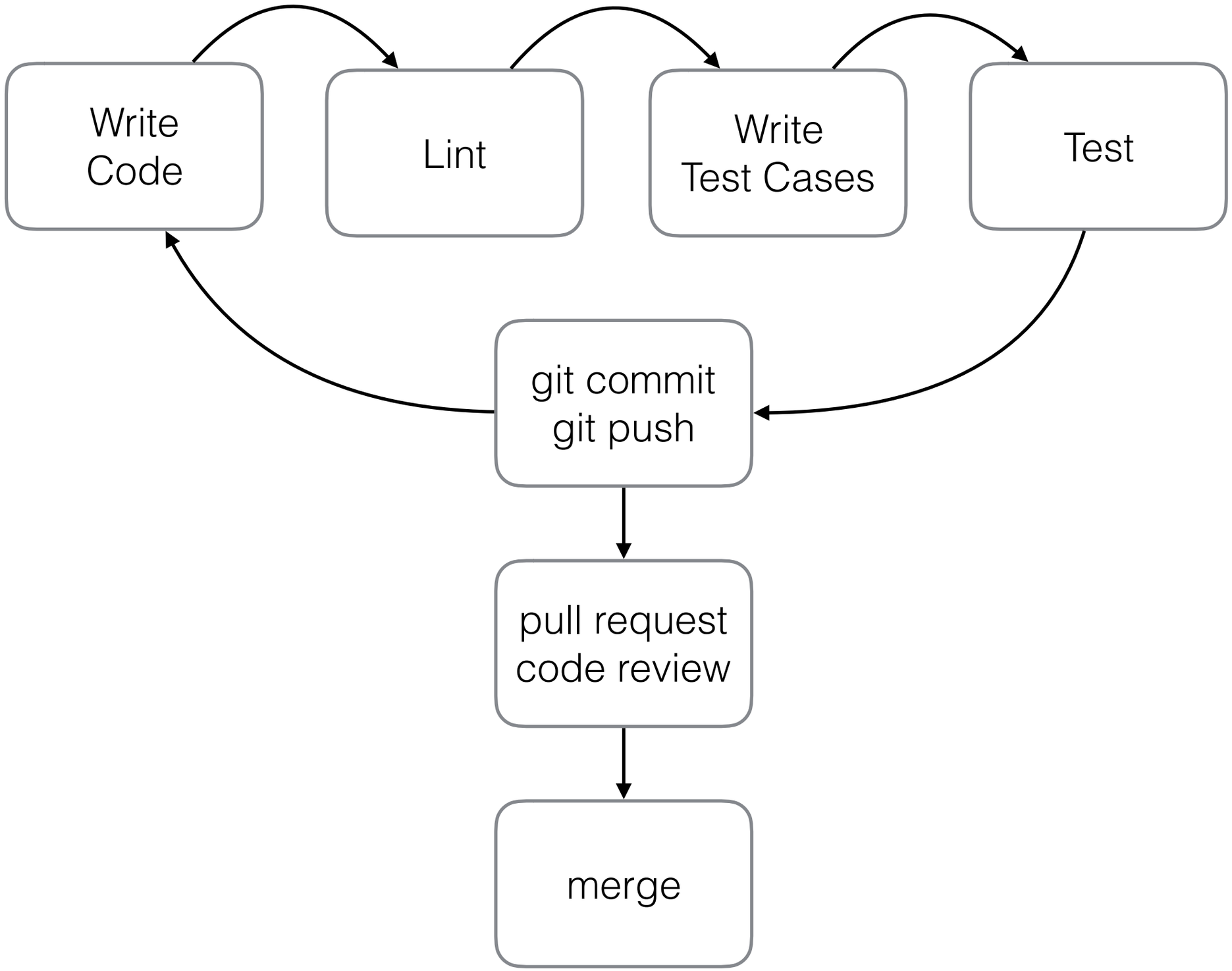
version 1-23-2017
committed 11 days ago
d6ba477

version 1-23-2017
committed 11 days ago
0ecc973

- Beware of “git commit” without specifying the changes
- Commit related changes together
- Commit unrelated changes separately
- Use a GUI or `git add -p`

Git Workflow




Write
Code
Lint
Write
Test Cases
Test
git commit
git push
pull request
code review
merge

Code Review

- "Rigorous inspections can remove up to 90% of errors from a software product before the first test case is run."
-- “Facts and Fallacies of Software Engineering,” Robert Glass

Code Review

- "The average defect detection rate is only 25% for unit testing, 35% for function testing, and 45% for integration testing. In contrast, the average effectiveness of design and code inspections are 55 and 60%."
--- “Code Complete,” Steve McConnell.

Code Review

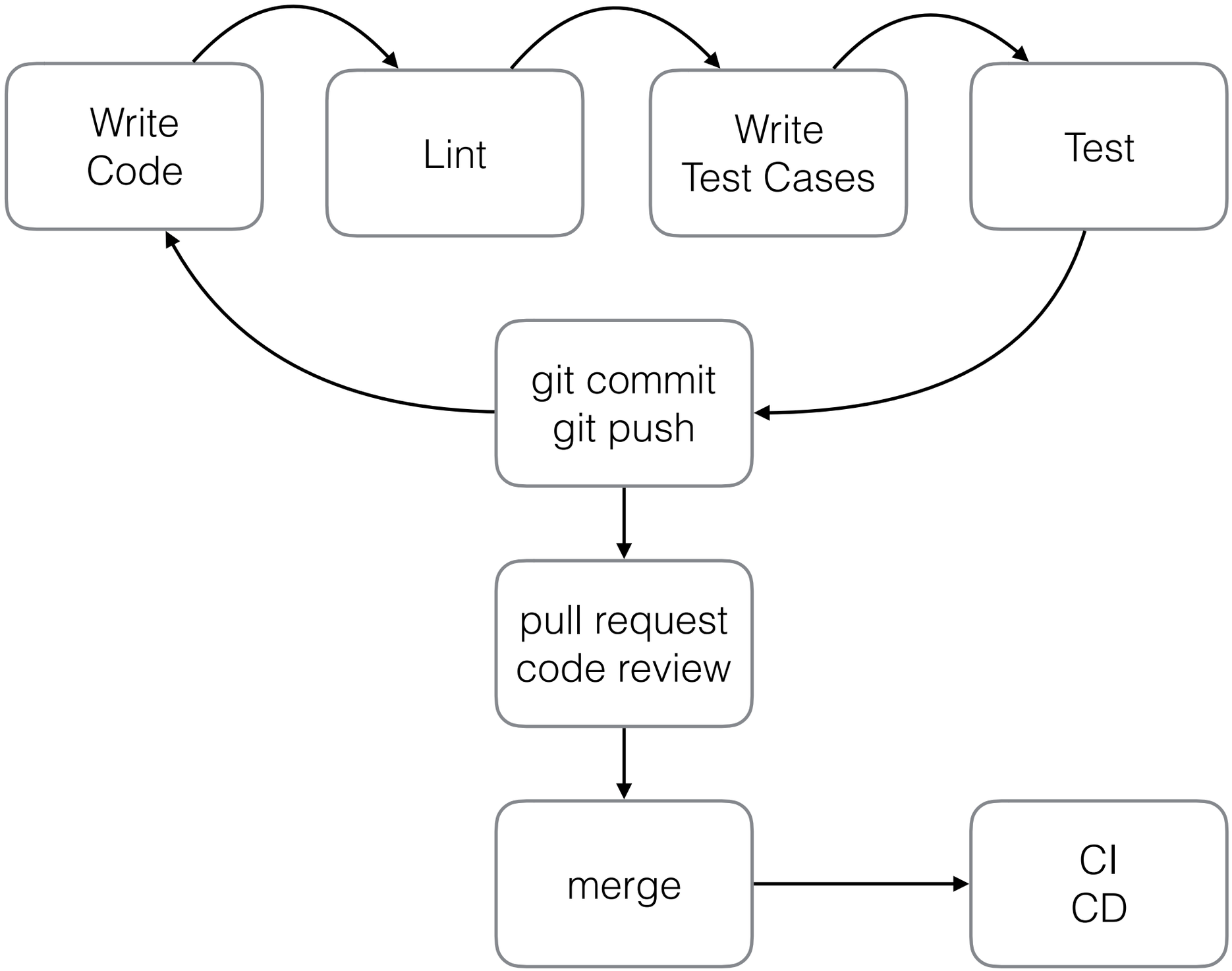
- Another team member should review and approve the pull request for merging
- Treat this seriously
- Document your review on github

Code Review Tips

- Review small chunk at a time
- Make sure code is ready for review
- Not just for styles / readability

Code Review Tips

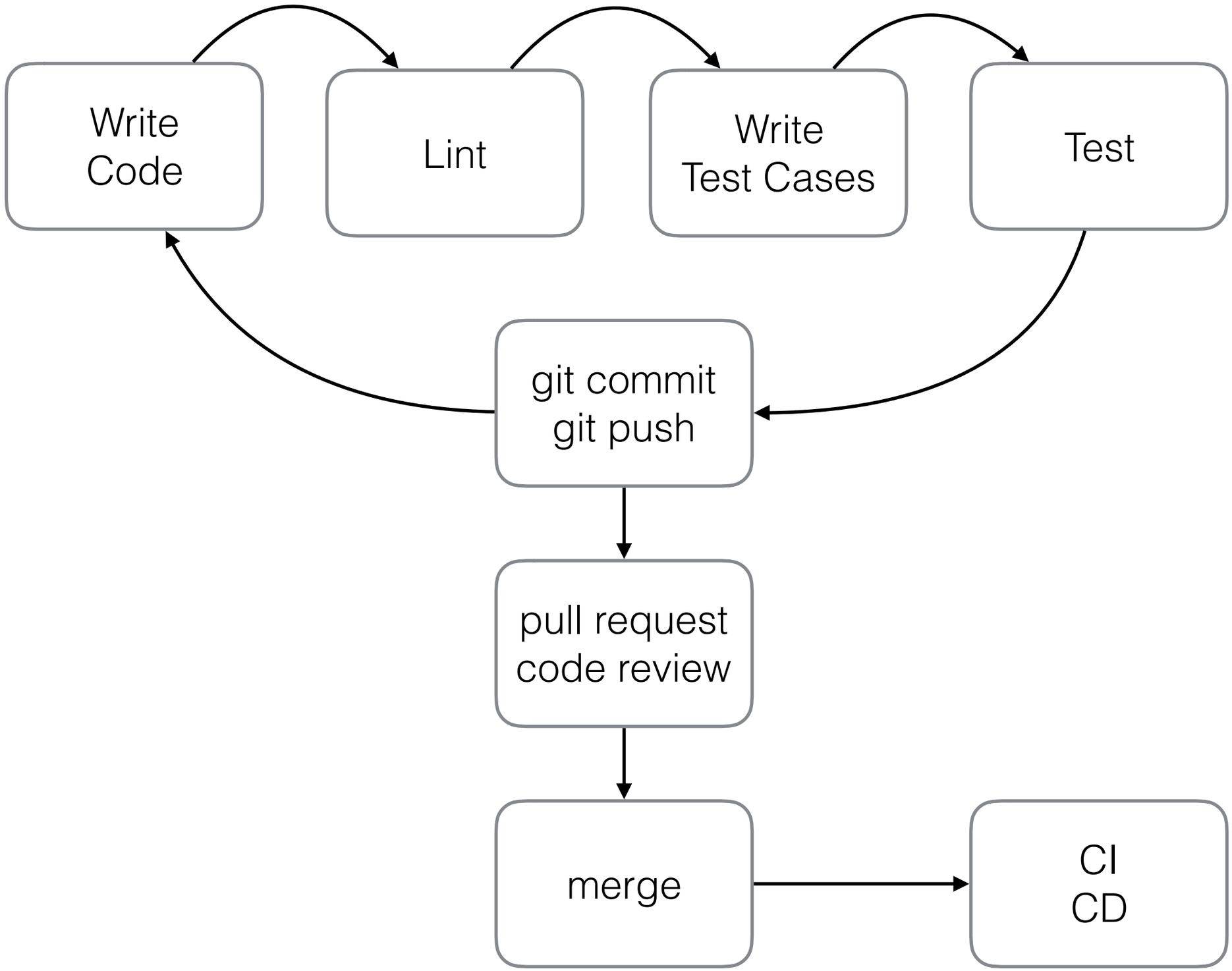
- At least two members are familiar with the working of a component
- Remember: communication in terms of your requirements, diagrams, code, comments, commit messages, etc are important


Write
Code
Lint
Write
Test Cases
Test
git commit
git push
pull request
code review
merge
CI
CD

After I changed one line of code..

Continuous Integration

- Your commit does not mess things up
- Everyone can see what is broken and can fix immediately
- A tight build/test loop


Write
Code
Lint
Write
Test Cases
Test
git commit
git push
pull request
code review
merge
CI
CD