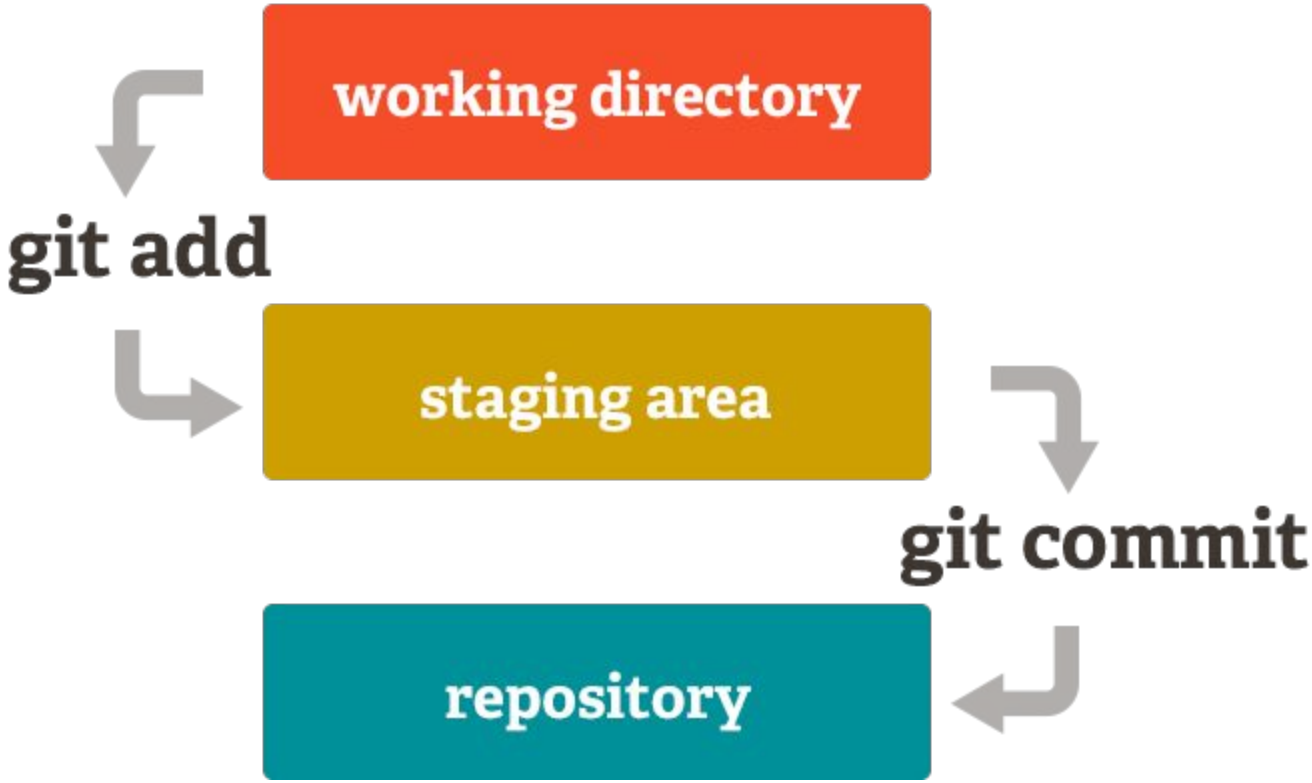
Gerrit Workflow

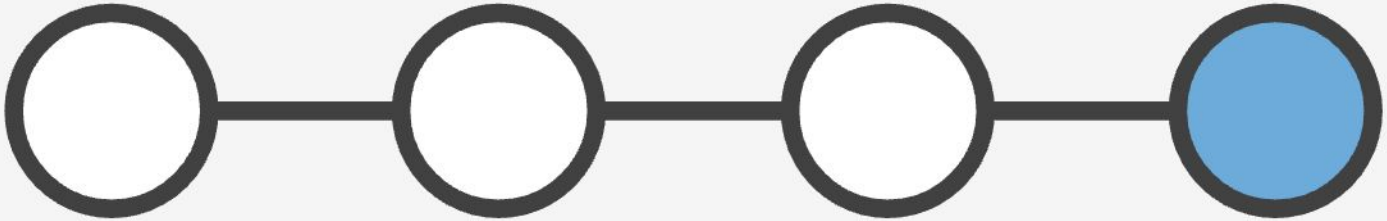
Adam Yi <i@adamyi.com>
26/03/2019

Git

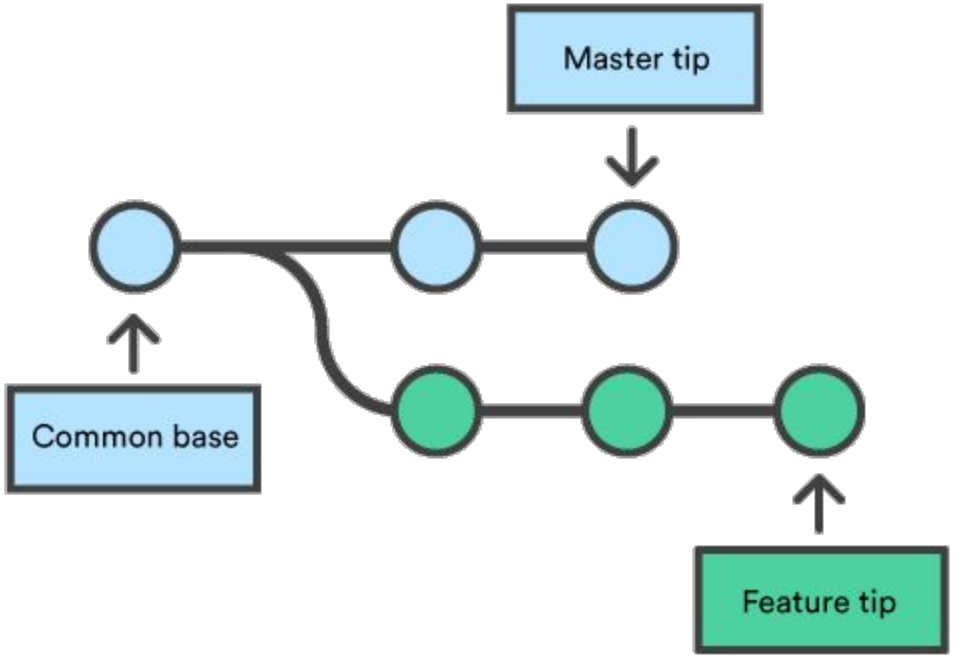


Google Picture Credit: Haydar Ali Ismail

```
commit 87a20b53ceb12bfdf51f8783cdb681f85860c59e
parent 88ecff576ef01a9178713866e2b5debe38dcd24b
author Jeffrey A. Dean <jeff@google.com> 1501883748 -0700
committer TensorFlower Gardener <gardener@tensorflow.org> 1501883997 -0700

Various compiler speedups.  Improves compilation of an image model from
35.6 seconds to 33.3 seconds (average of three runs) (+6.5% improvement).

(1) Avoid extra hash table lookups in HeapSimulator by holding onto the pointer
to the hash table value, rather than looking it up from the same key multiple
times.

(2) In HeapSimulator, reuse operand_buffers_to_free and dead_buffers_to_free
across all instructions, rather than allocating new vectors on every
instructions.

(3) Avoid use of Printf and improve efficiency of string generation for
HloInstruction::ToString and ShapeUtil::HumanStringWithLayout.
PiperOrigin-RevId: 164314222
```

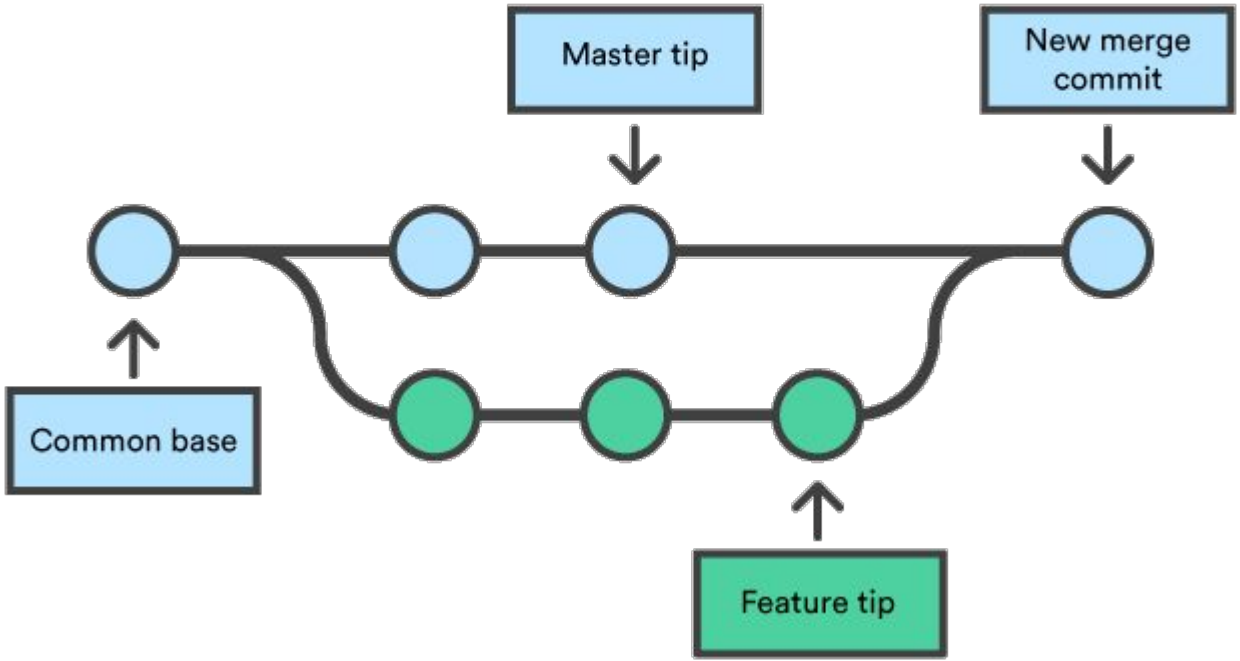


Git Branches



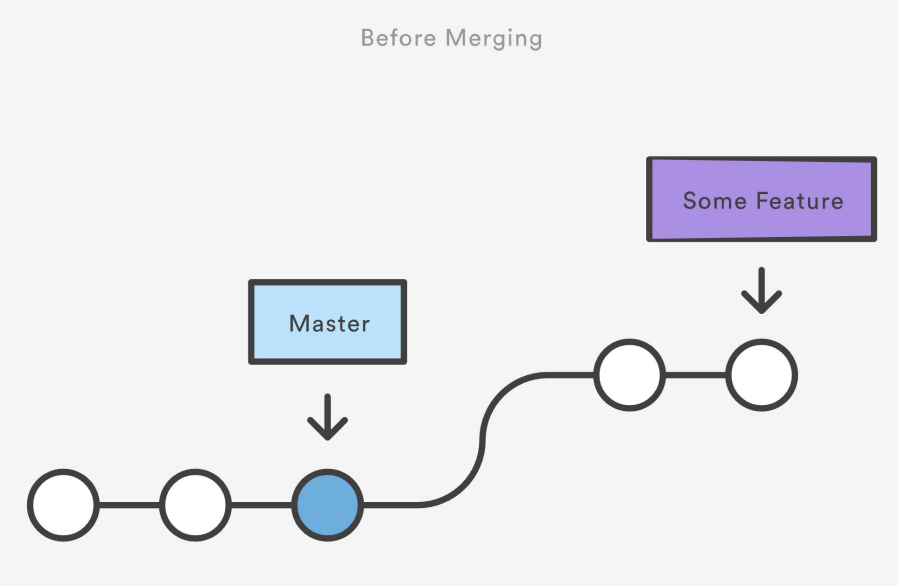
Google Picture Credit: Atlassian

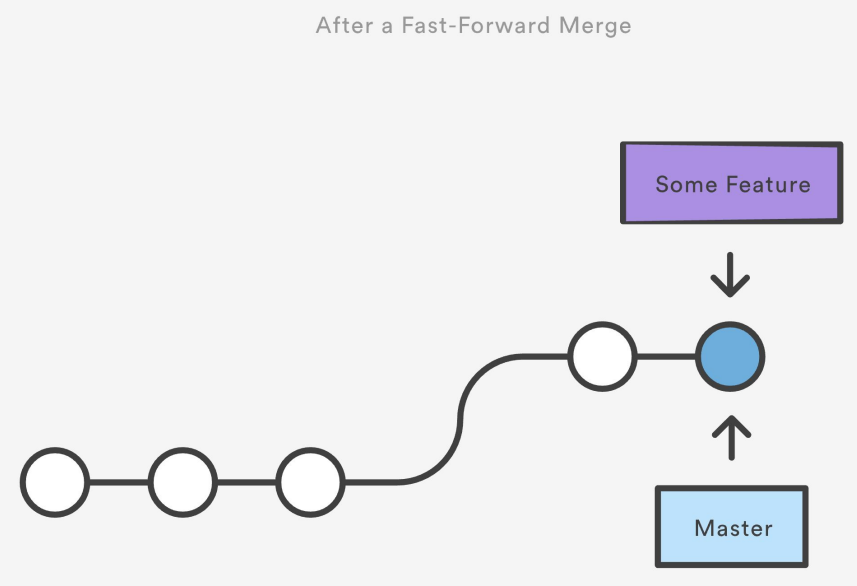
Git Merge



Google Picture Credit: Atlassian

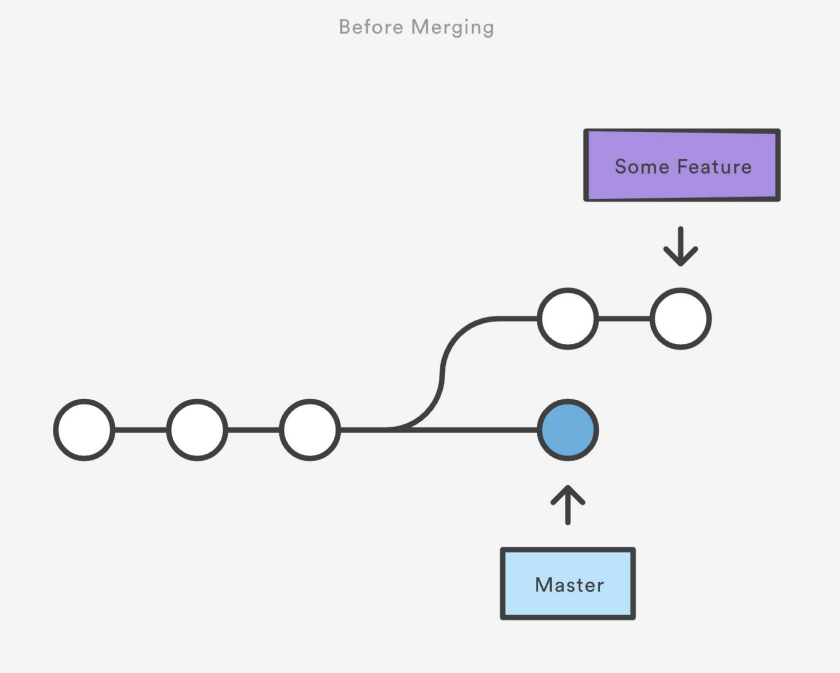
Git Fast Forward Merge

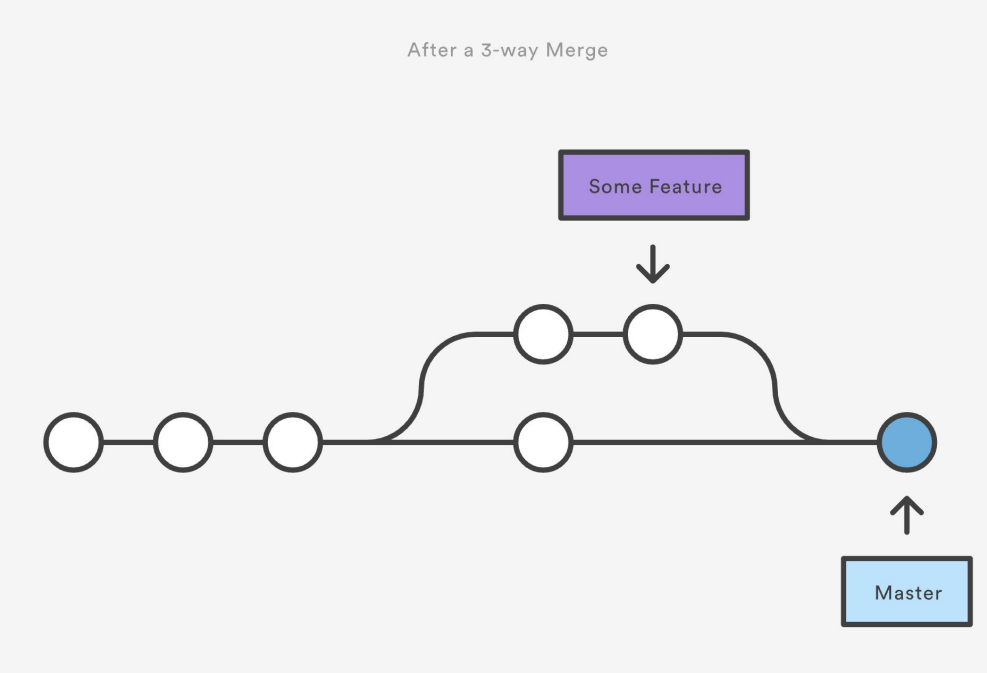




Google Picture Credit: Atlassian

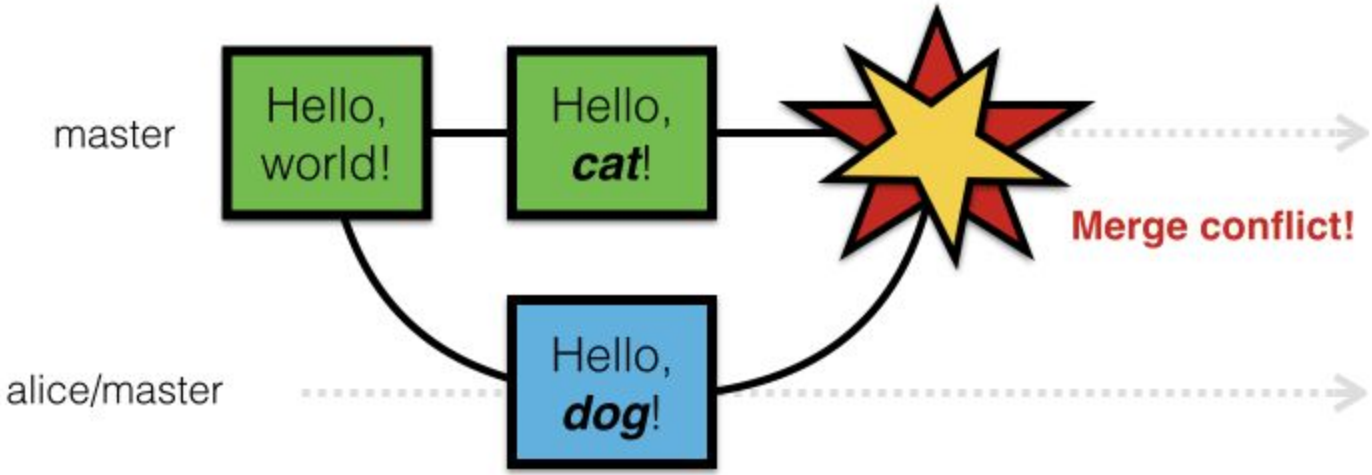
Git 3-Way Merge





Google Picture Credit: Atlassian

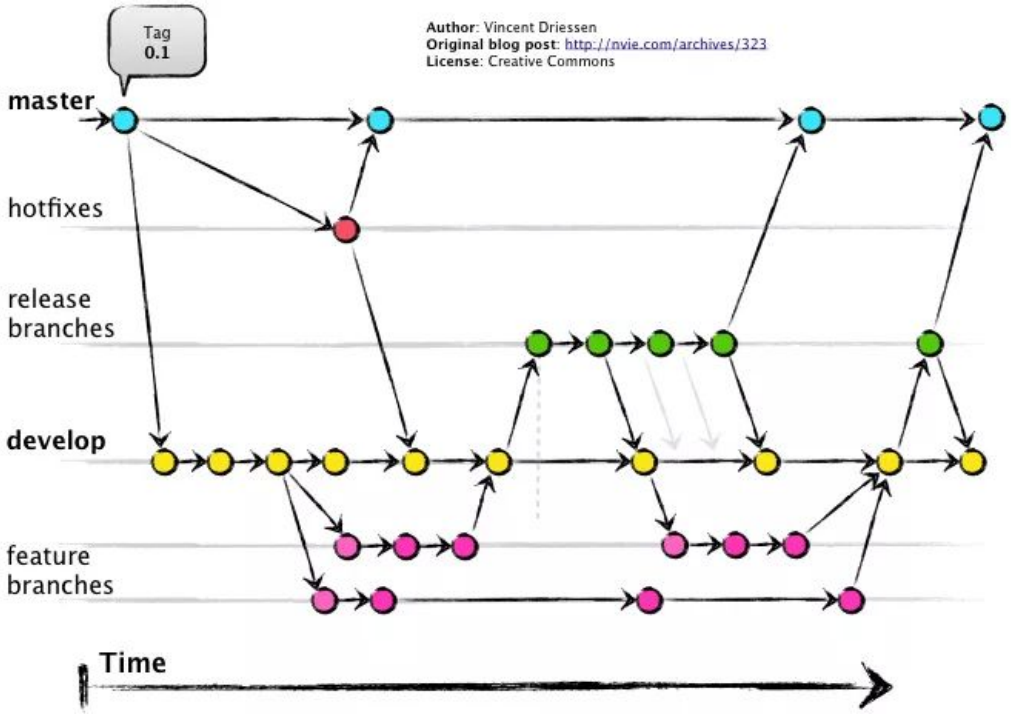
Git Merge Conflict

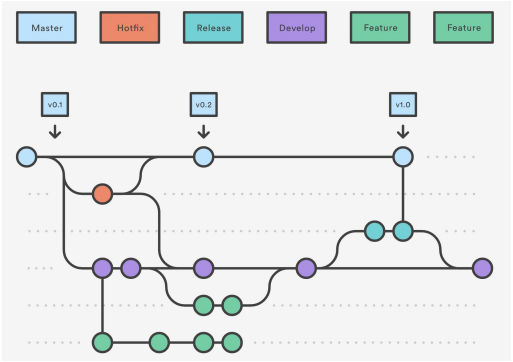



```
Hello world
<<<<<<< HEAD
hello dog
=======
Hello cat
>>>>>>> master
```

Google Picture Credit: Atlassian

Workflow: Git Flow

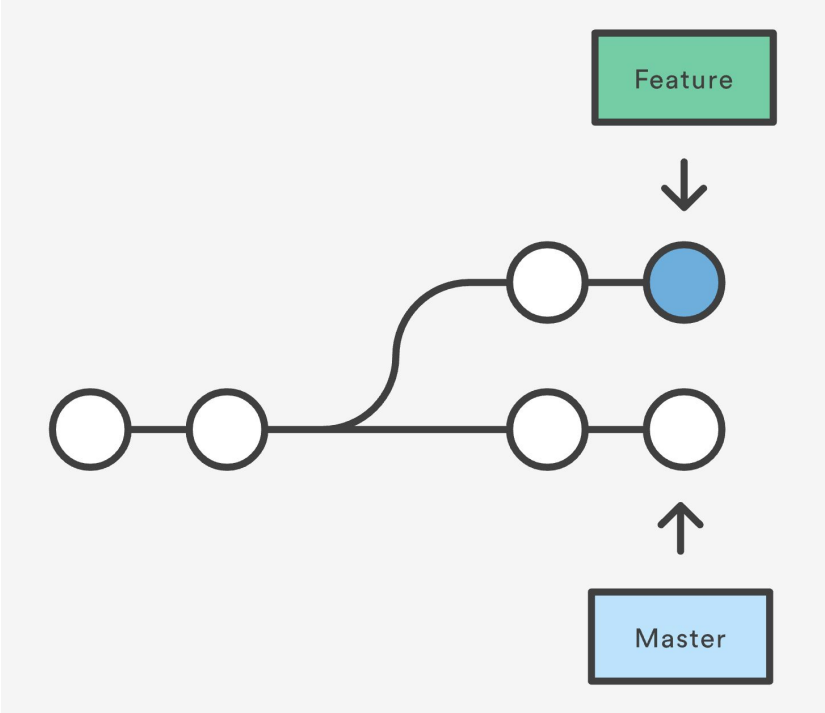


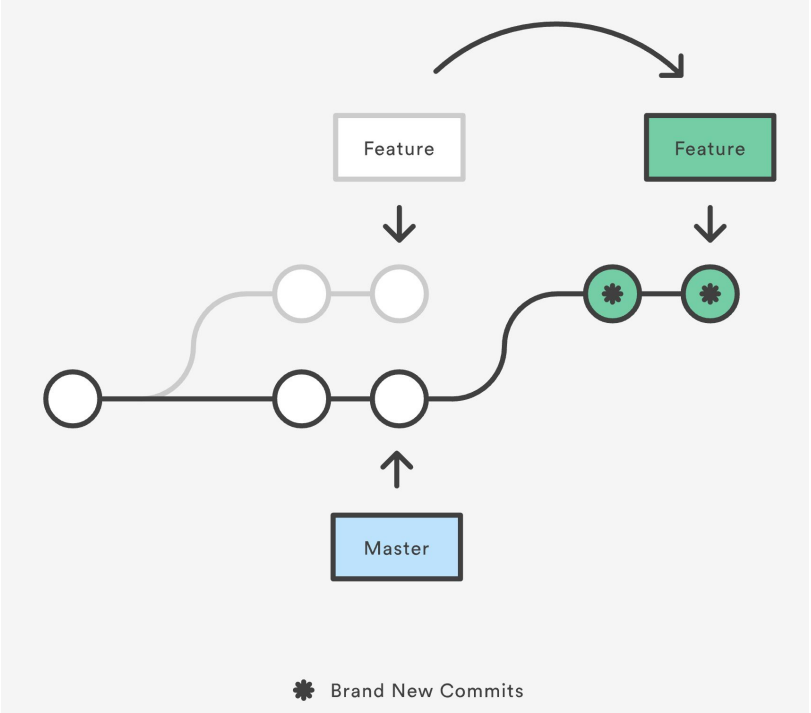


Google Picture Credit: Vincent Driessen / Atlassian

That graph is so painful to watch. Is there something better?

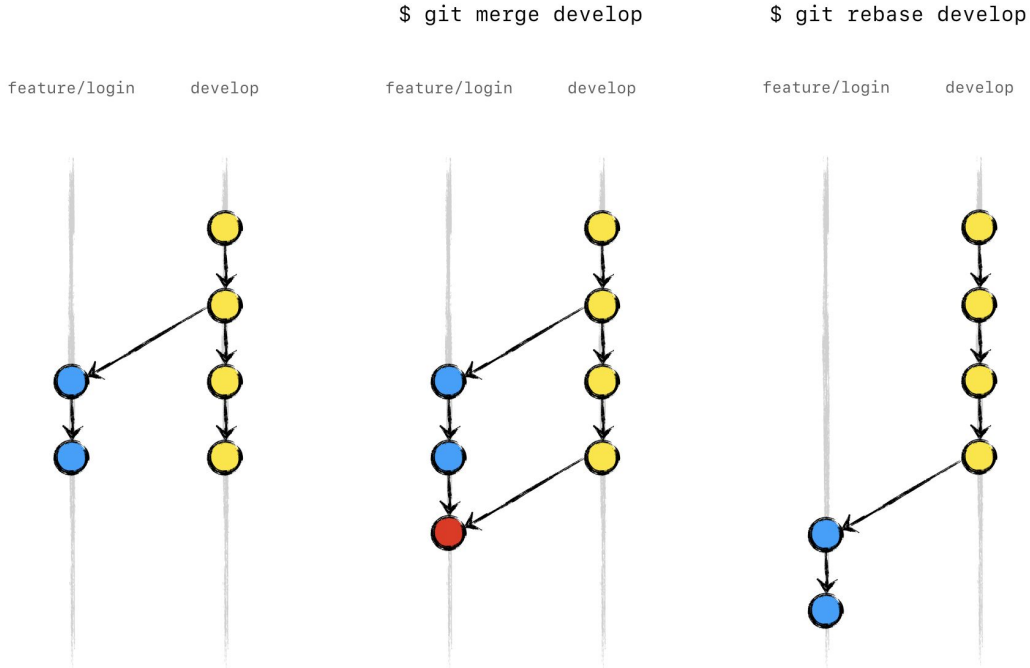
Git Rebase





Google Picture Credit: Atlassian

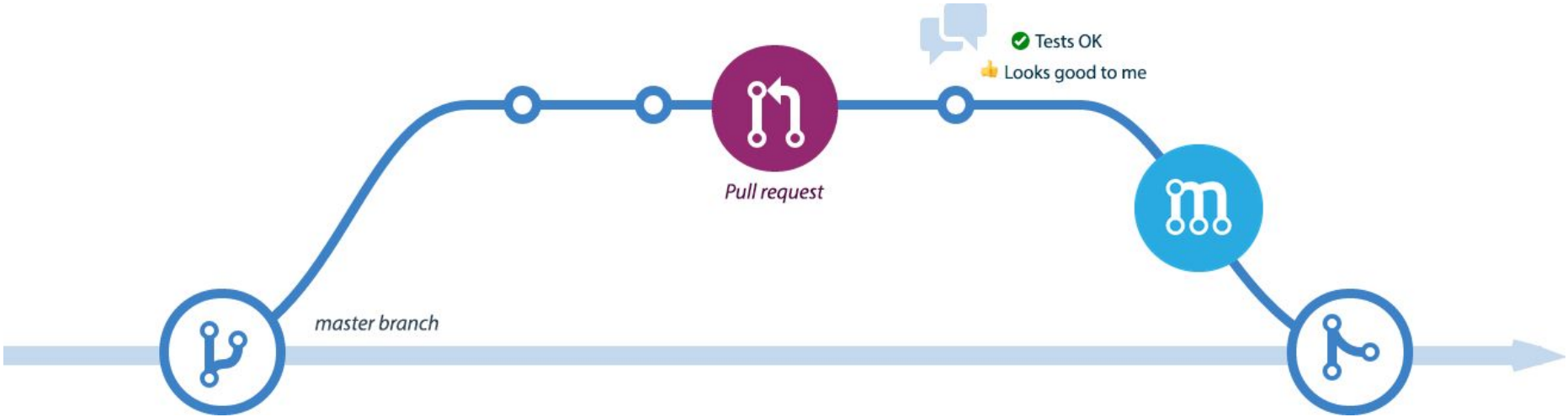
Git Merge vs Git Rebase



Those are band new git commits due to hash changes.
But in essence, we think they are the same changes.

Google Picture Credit: Jeff Kreeftmeijer

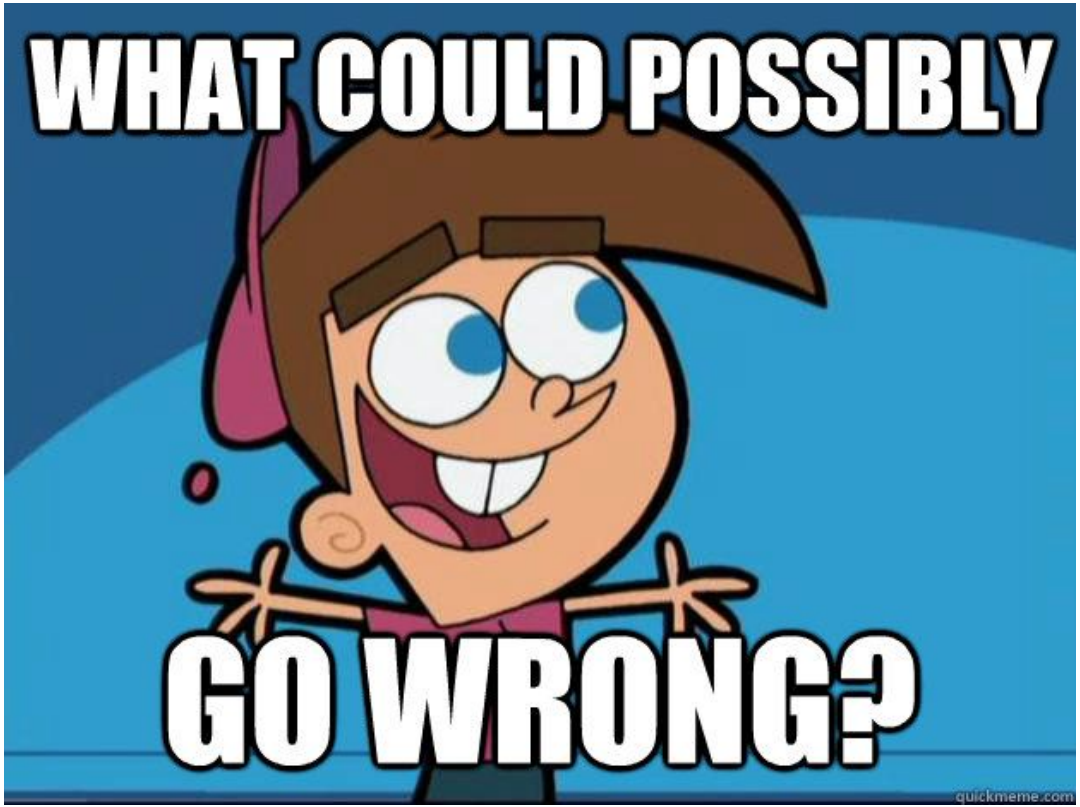
GitHub Workflow - Pull Request

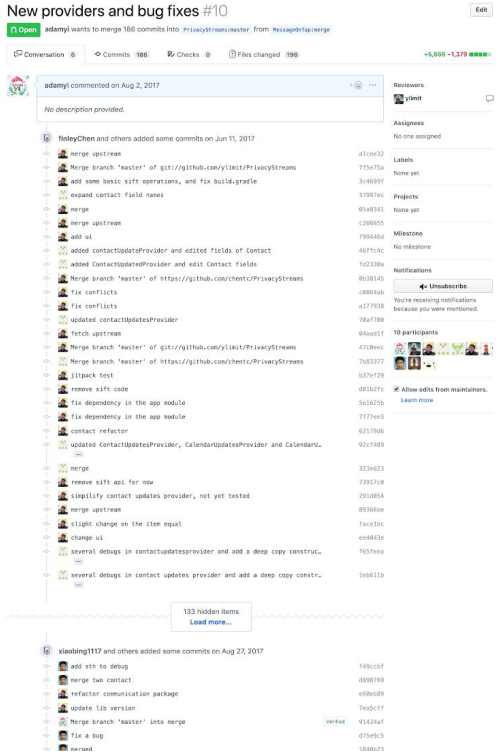


Merge (Fast Forward/3-way) by default
Can also do squash and rebase now.

Google Picture Credit: Julien Danjou

That’s cool.





New providers and bug fixes #10

Open adamyi wants to merge 186 commits into PrivacyStreams:master from MessageOnTap:merge

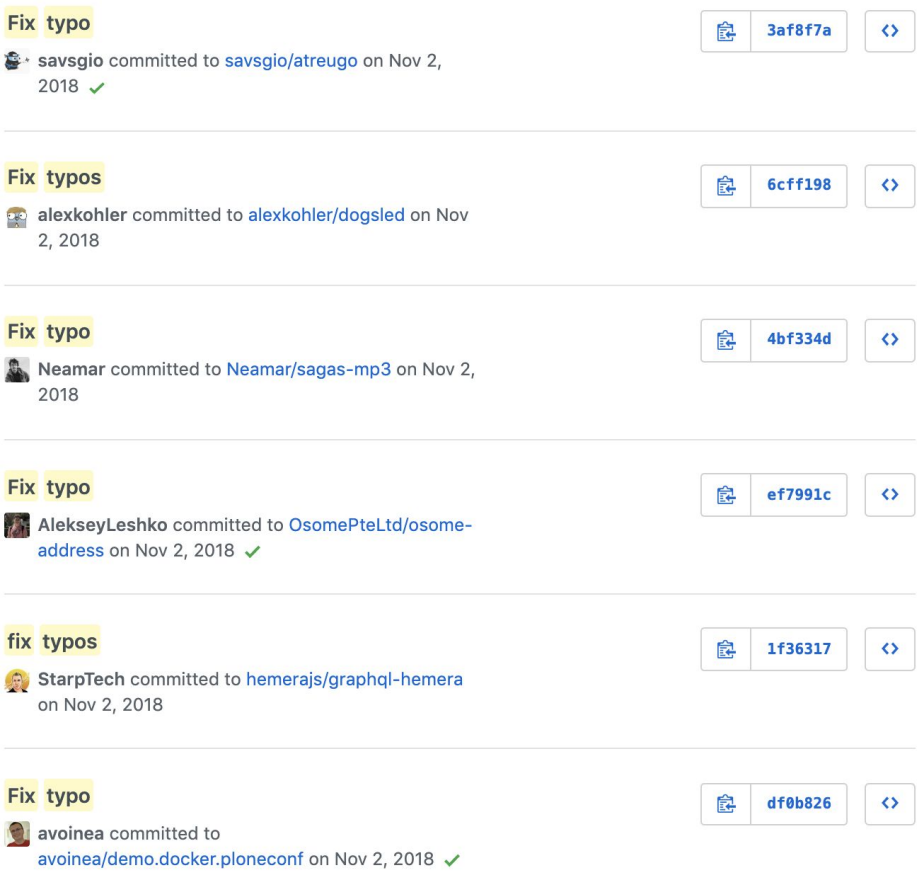
Conversation 6 Commits 186 Checks 0 Files changed 196

adamyi commented on Aug 2, 2017

No description provided.

Huge Branch Single Review

- Impossible for Review
- Uncollaborative



Google

Commits unqualified as a change

- Messy history
- Not every commit works
- Hard to roll back

Introducing Gerrit

Gerrit

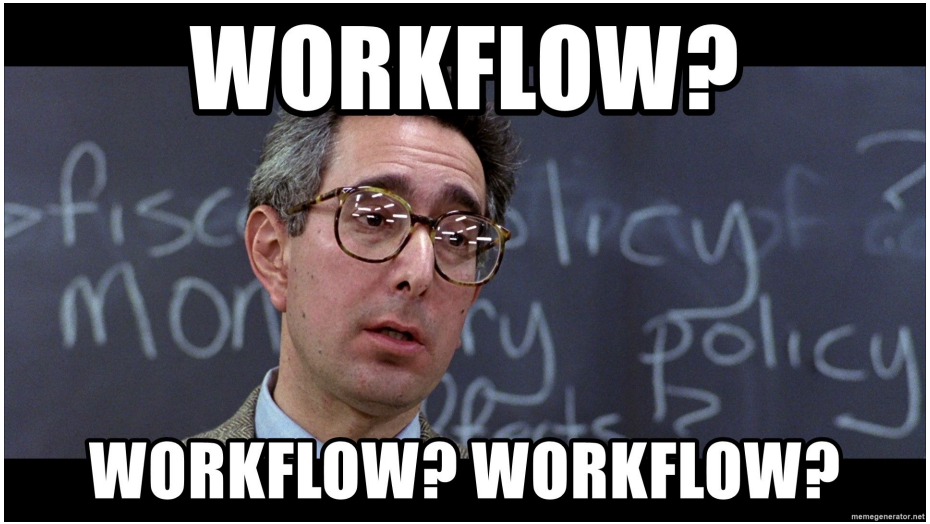
Google internal Piper Version Control workflow

Redesigned for Git

Originally developed for AOSP (Android Open-Source Project)

Used by Android, Chromium, Eclipse, Golang, MediaWiki, LibreOffice, OpenStack, SAP, Qt, BMW Car, etc.

Gerrit Workflow



~~~all hail the code review god~~~

Code-Review

- `+2 Looks good to me, approved`
- `+1 Looks good to me, but someone else must approve`
- `0 No score`
- `-1 I would prefer that you didn't submit this`
- `-2 Do not submit`

Verified

- `+1 It works`
- `0 No score`
- `-1 It doesn't work`

Gerrit: high level workflow

Google Picture Credit: Luca Milanesio

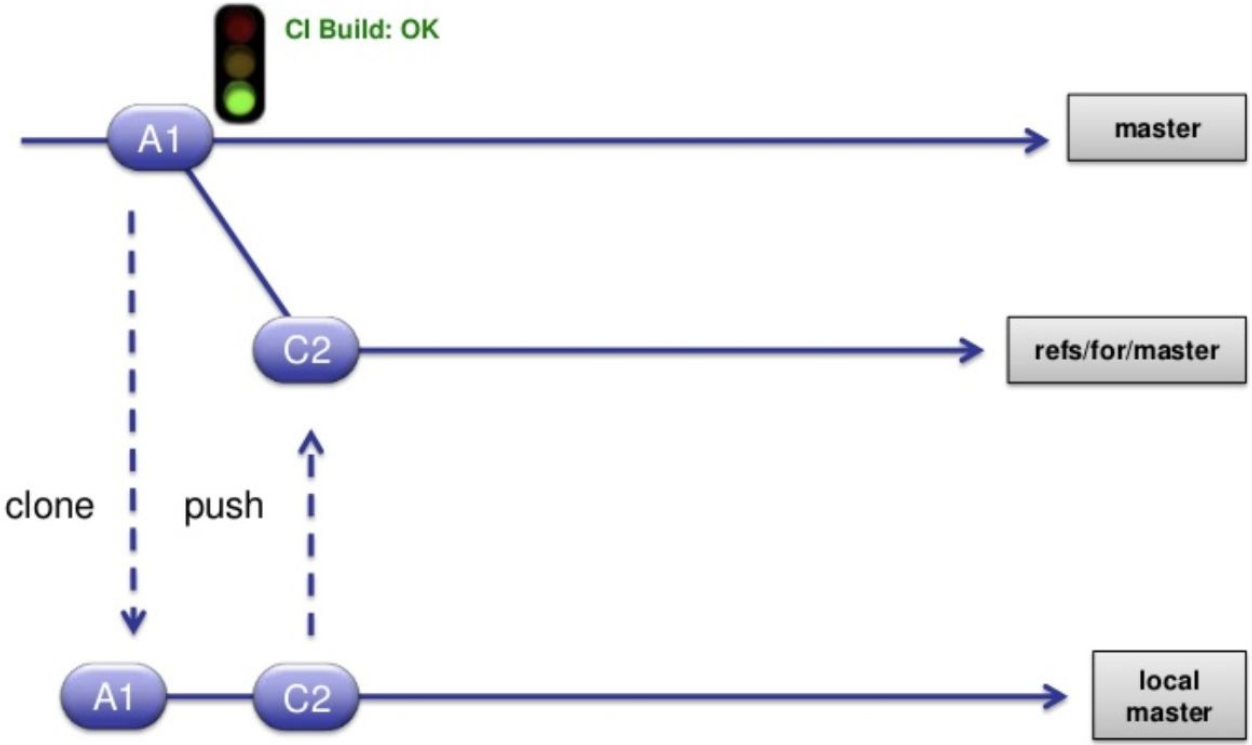
Gerrit: high level workflow

Google Picture Credit: Luca Milanesio

Gerrit: high level workflow

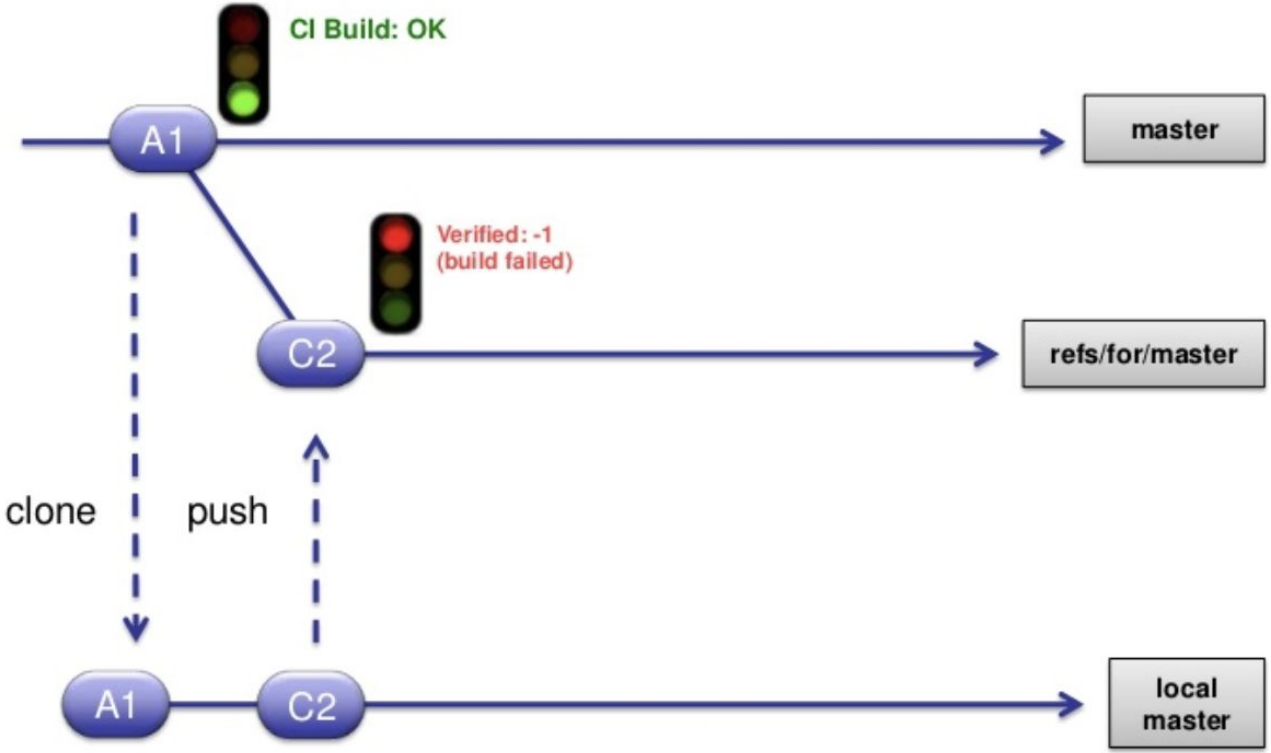
Google Picture Credit: Luca Milanesio

Gerrit: high level workflow



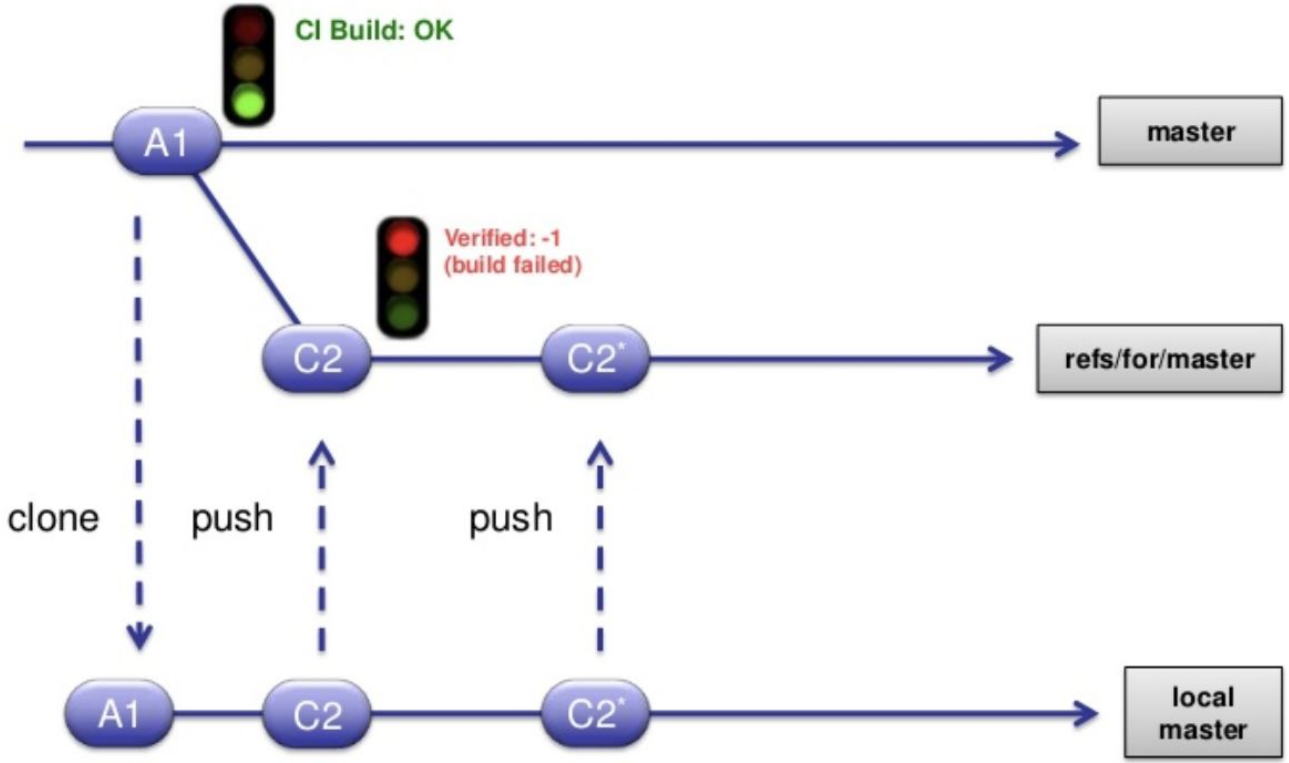
Google Picture Credit: Luca Milanesio

Gerrit: high level workflow



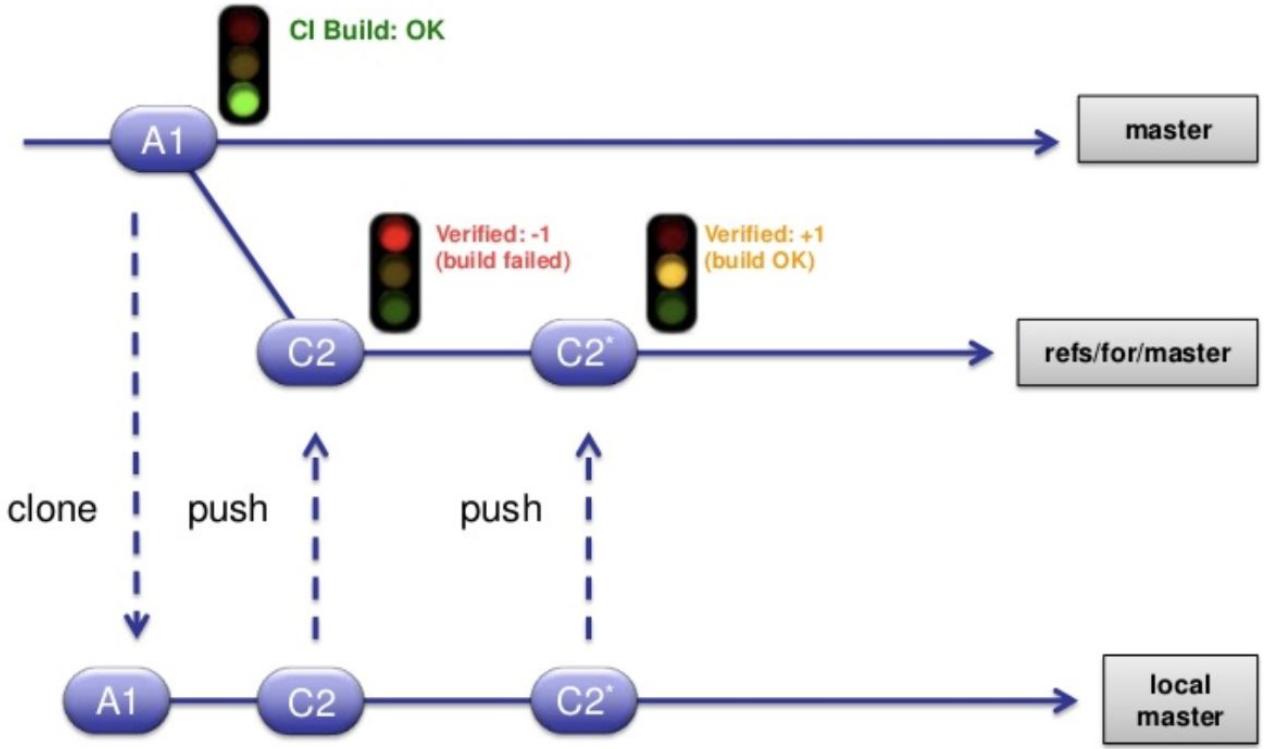
Google Picture Credit: Luca Milanesio

Gerrit: high level workflow



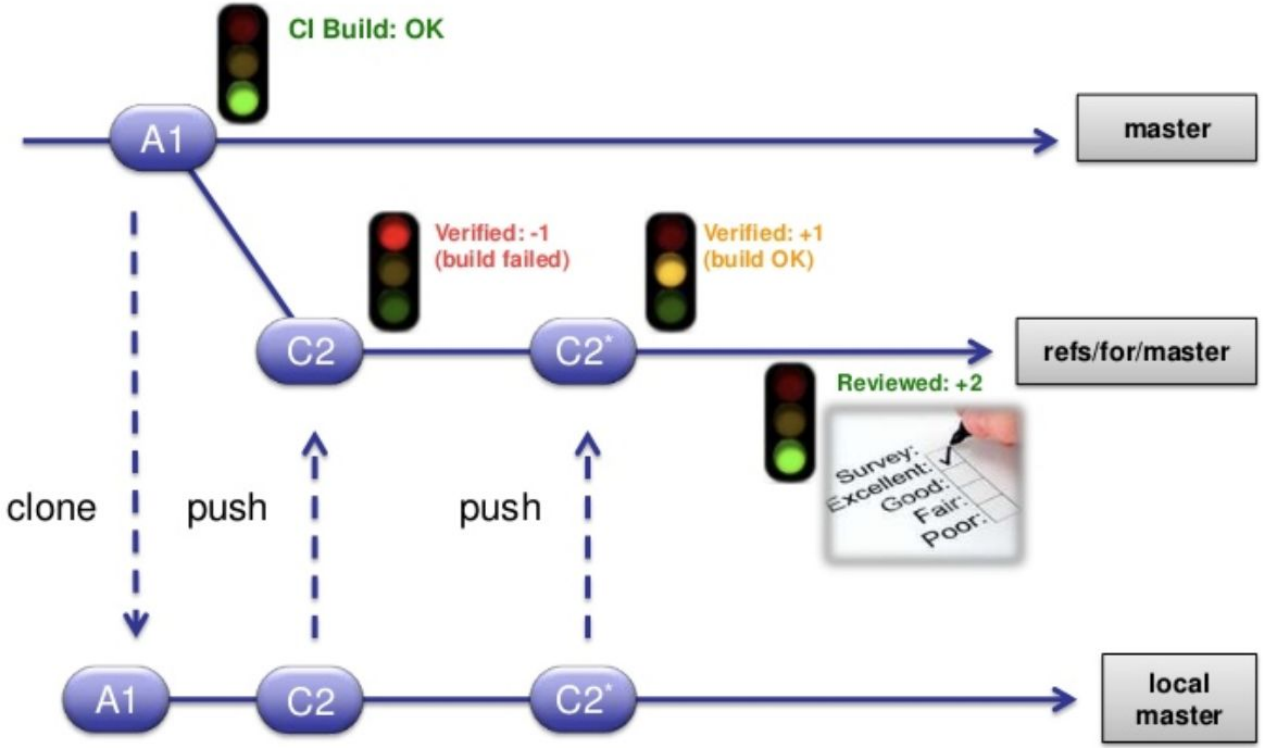
Google Picture Credit: Luca Milanesio

Gerrit: high level workflow



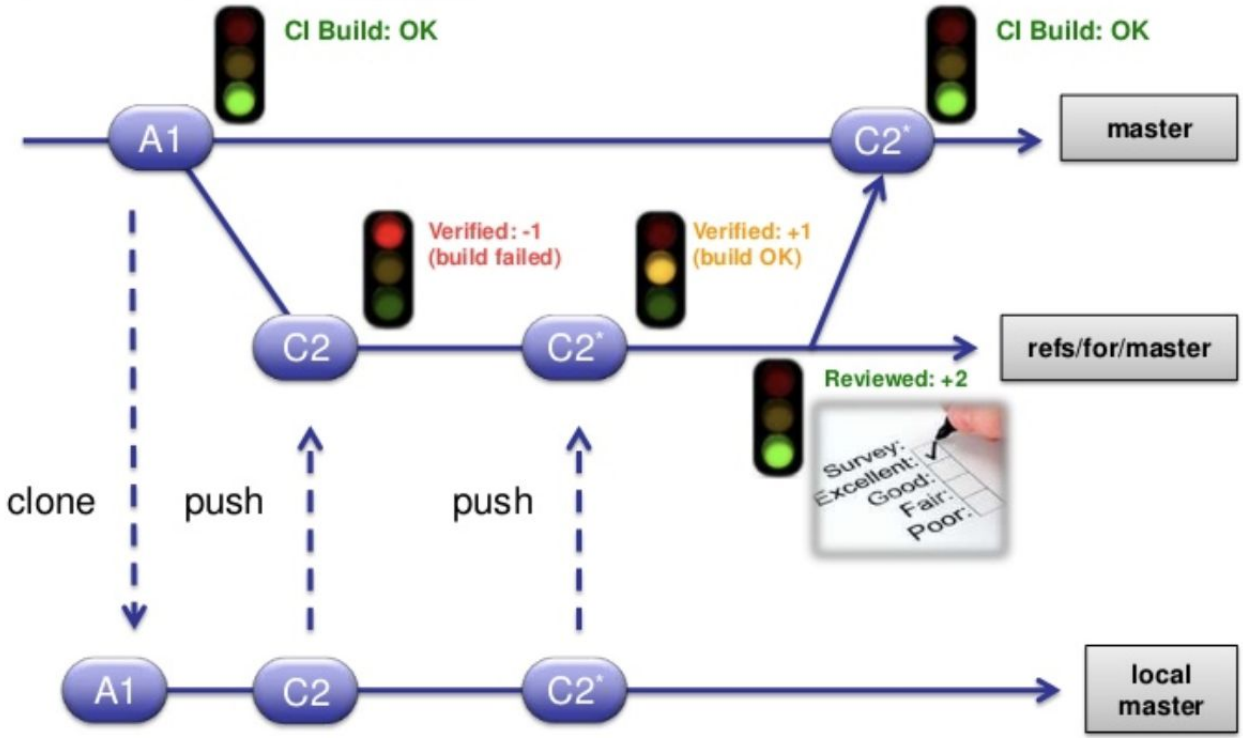
Google Picture Credit: Luca Milanesio

Gerrit: high level workflow



Google Picture Credit: Luca Milanesio

Gerrit: high level workflow

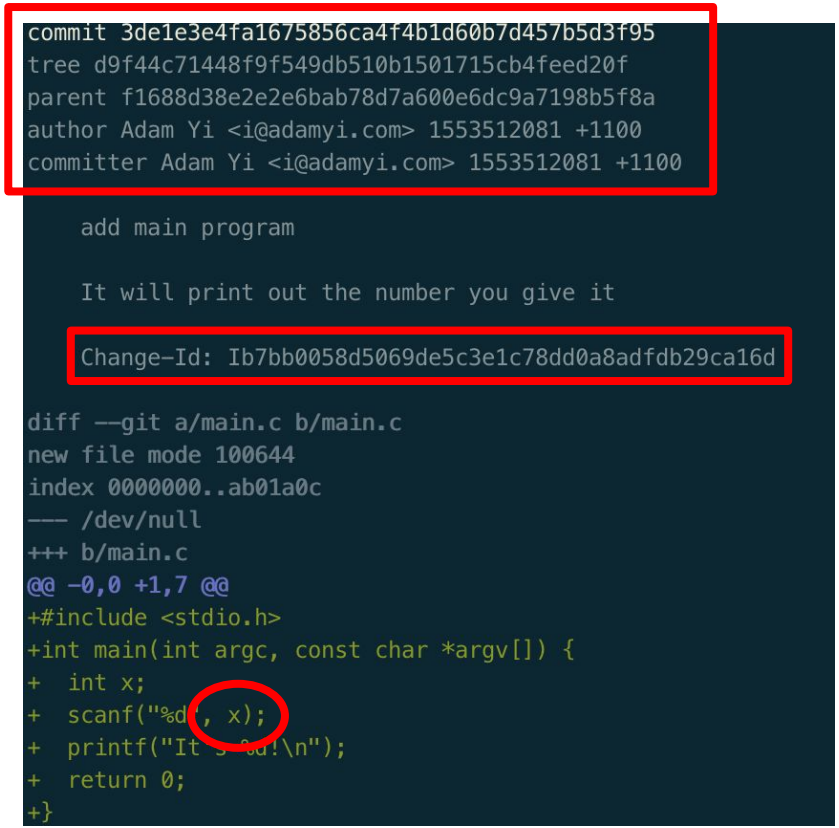


Google Picture Credit: Luca Milanesio

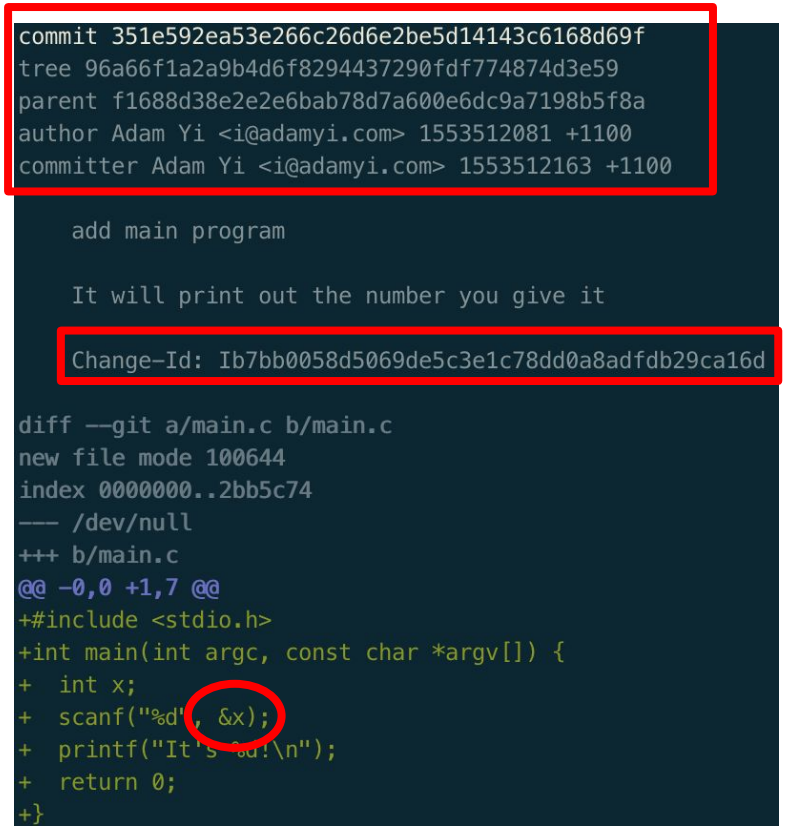
C2 and C2*

- C2 and C2* is the **same Change**
- C2 and C2* have the **same Change-Id** in their commit messages.
- C2 is **Patch Set 1**. C2* is **Patch Set 2**.
- C2 and C2* are completely different **git commits**, because the content is changed.
- Commit is a git thing. Change is a general concept. In Gerrit, a change is a commit. But two different commits can be the same change (e.g. C2 and C2*).
- C2* is a newer version of C2. Therefore we call it Patch Set 2, which stands for the second version of the change.

PS2 is a new version of PS1.



Patch Set 1



Patch Set 2

Google

Code Reviewer: you need an & sign here.

In the meanwhile, master was updated...

```
commit 351e592ea53e266c26d6e2be5d14143c6168d69f
tree 96a66f1a2a9b4d6f8294437290fdf774874d3e59
parent f1688d38e2e2e6bab78d7a600e6dc9a7198b5f8a
author Adam Yi <i@adamyi.com> 1553512081 +1100
committer Adam Yi <i@adamyi.com> 1553512163 +1100

    add main program

    It will print out the number you give it

    Change-Id: Ib7bb0058d5069de5c3e1c78dd0a8adfdb29ca16d

diff --git a/main.c b/main.c
new file mode 100644
index 0000000..2bb5c74
--- /dev/null
+++ b/main.c
@@ -0,0 +1,7 @@
+#include <stdio.h>
+int main(int argc, const char *argv[]) {
+  int x;
+  scanf("%d", &x);
+  printf("It's %d!\n");
+  return 0;
+}
```

Patch Set 2

```
commit 1fcac9bba8a599ea2fafaa82215cfb5579728626
tree 4f351999daf65be3ae7a52f41a2fec619dbca553
parent 01511f8608c7ec592de0710a4e7958e5aca42a9b
author Adam Yi <i@adamyi.com> 1553512081 +1100
committer Adam Yi <i@adamyi.com> 1553514003 +1100

    add main program

    It will print out the number you give it

    Change-Id: Ib7bb0058d5069de5c3e1c78dd0a8adfdb29ca16d

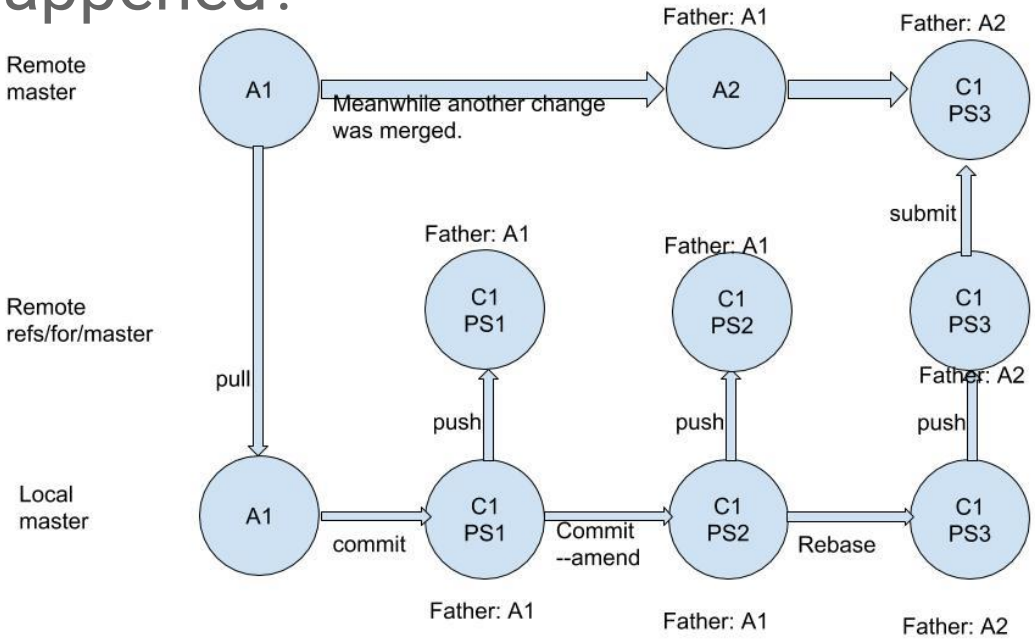
diff --git a/main.c b/main.c
new file mode 100644
index 0000000..2bb5c74
--- /dev/null
+++ b/main.c
@@ -0,0 +1,7 @@
+#include <stdio.h>
+int main(int argc, const char *argv[]) {
+  int x;
+  scanf("%d", &x);
+  printf("It's %d!\n");
+  return 0;
+}
```

Patch Set 3

Google

You can rebase your change to a new PS!

What just happened?



Remember that PS1, PS2, and PS3 are different git commits but they share the same Change-Id. Both PS2 and PS1 are based on A1, yet PS3 is based on A2. There are no dependencies between PS1, PS2, and PS3. In other words, different Patch Sets shouldn't appear in the git history at the same time. There can only be one!

Change-Id

- So how can we maintain the Change-Id?
- There comes **Change-Id hook**!
- It will always append a random new ID if there's no Change-Id in your commit message.
- If there is already one, it won't change it or add one.

Therefore:

- If you do a new commit, it will help you generate one
- If you amend an old commit, the Change-Id is already there and won't be changed unless you manually change it.

Remember all Patch Sets for a single change share one Change-Id; different changes have different Change-Id!

That is the general idea of Gerrit workflow.

However there are multiple different ways you can use git to fulfill the same workflow. It’s really flexible.

For Example

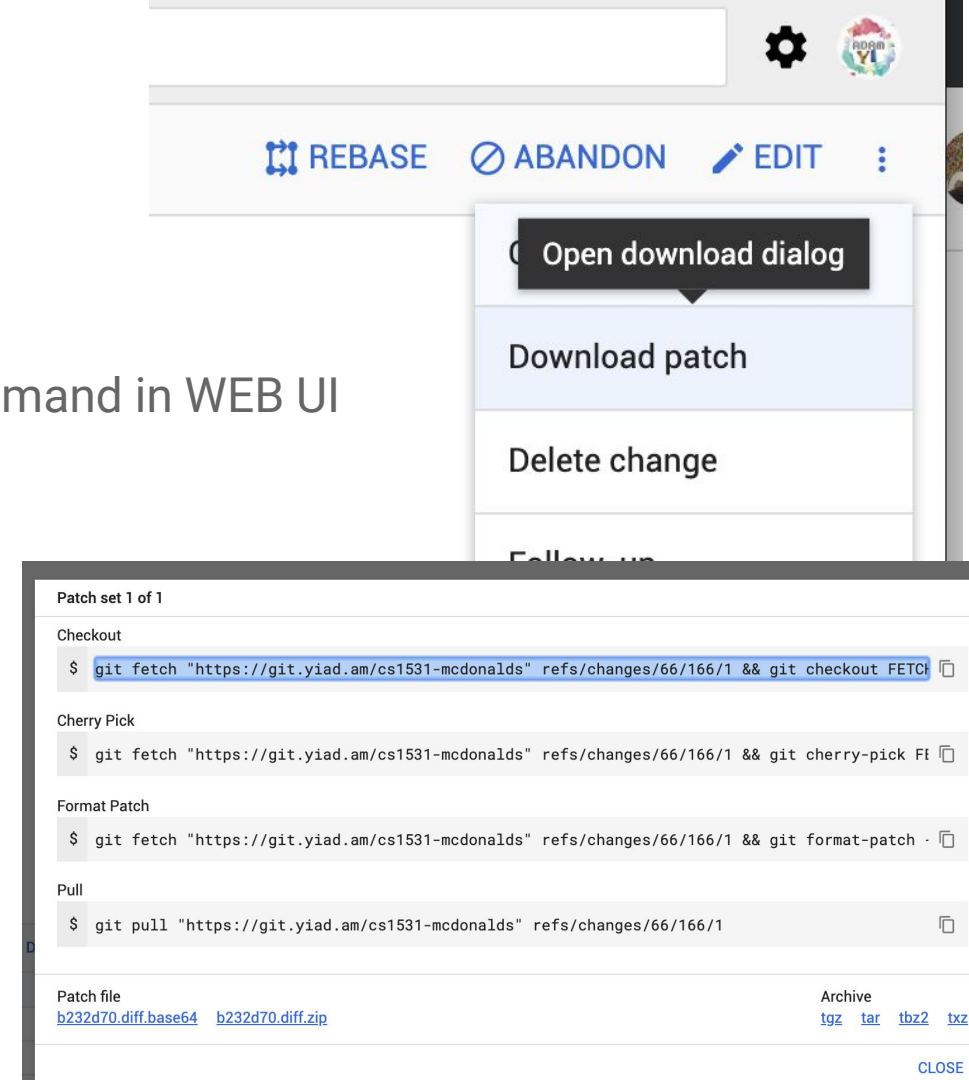
Write new Change

1. Git pull
2. Git checkout --detach master
3. Make your changes
4. Git commit
5. Git push origin HEAD:refs/for/master
6. Git checkout master
7. **Go to WEB UI and invite Reviewers**

For Example

Fetch existing change to local workspace

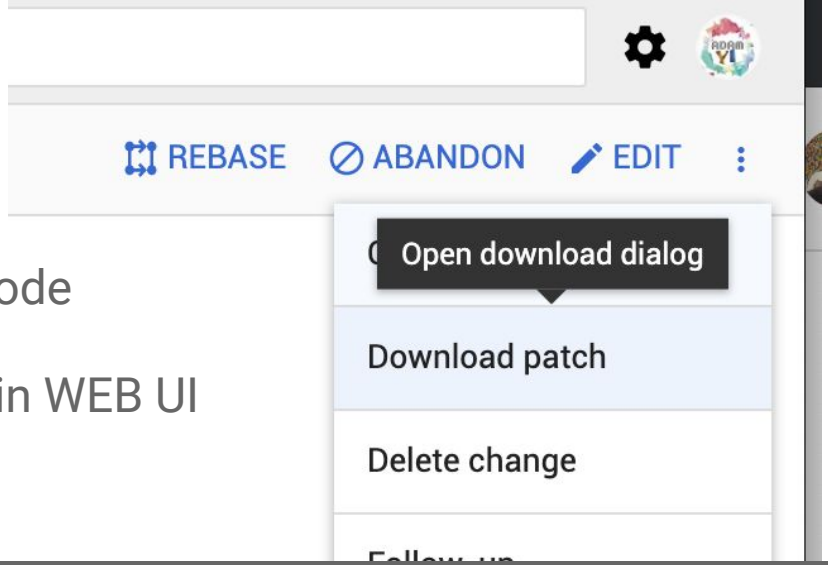
1. Checkout change using provided command in WEB UI
2. Use the code
3. Git checkout master

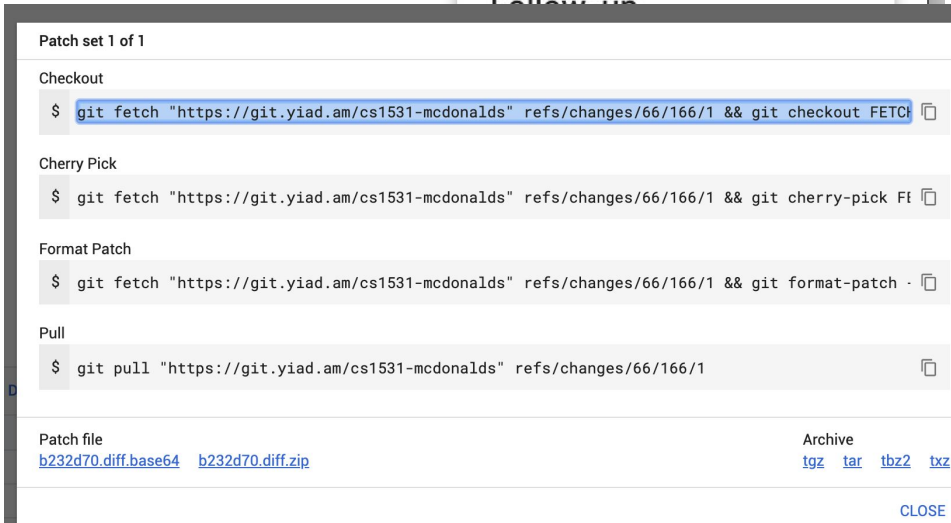


For Example

Upload new PS of existing change by changing code

1. Checkout change using provided command in WEB UI
2. Make your changes
3. Git commit --amend
4. Git push origin HEAD:refs/for/master
5. Git checkout master
6. **Go to WEB UI and make comments**

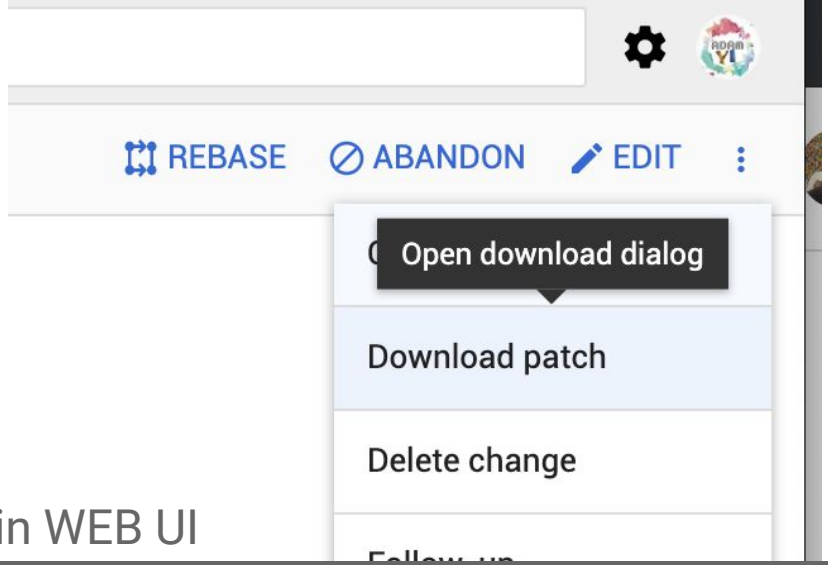


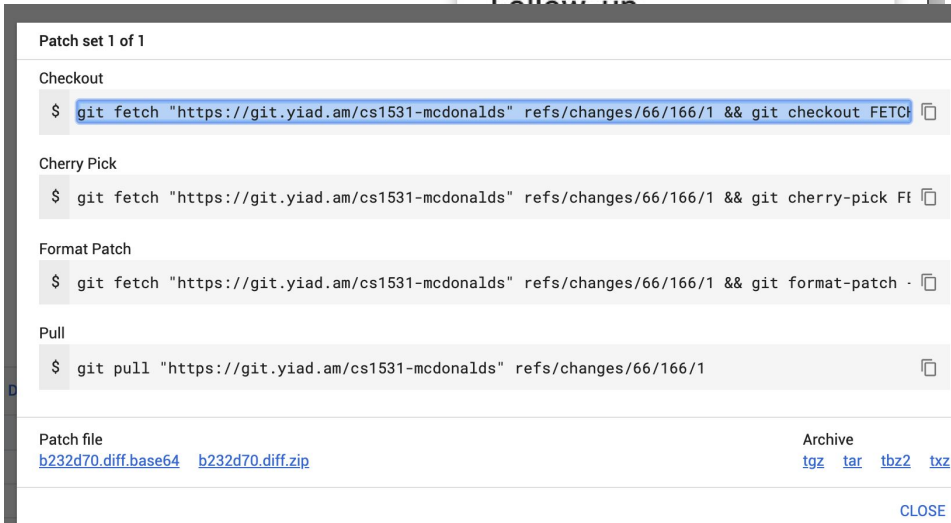


For Example

Upload new PS of existing change by rebasing

1. Git checkout master
2. Git pull
3. Checkout change using provided command in WEB UI
4. Git rebase master
5. Git push origin HEAD:refs/for/master
6. Git checkout master





The lifecycle of a change

https://gerrit-review.googlesource.com/c/plugins/checks/+/217941

Let's take a look at how this is merged.

Merged as 556c840 | 217941: Implement SubmitRule for checks

Updated Mar 23
Owner xchangcheng
Assignee
Reviewers Dave Borowitz, Patrick Hiesel
CC
Repo plugins/checks
Branch master
Parent abab6e9
Topic No topic
Hashtags

Verified +1 xchangcheng
Code-Review +2 Dave Borowitz

```
Implement SubmitRule for checks

This commit implements a new "SubmitRule" for checks.

A check result is blocking if:
* the checker who created the check result is still
applicable to the change.
* the checker who created the check result is enabled.
* the checker who created the check result has expected
"BlockingCondition", i.e. "STATE_NOT_PASSING".
* the check result hasn't passed.

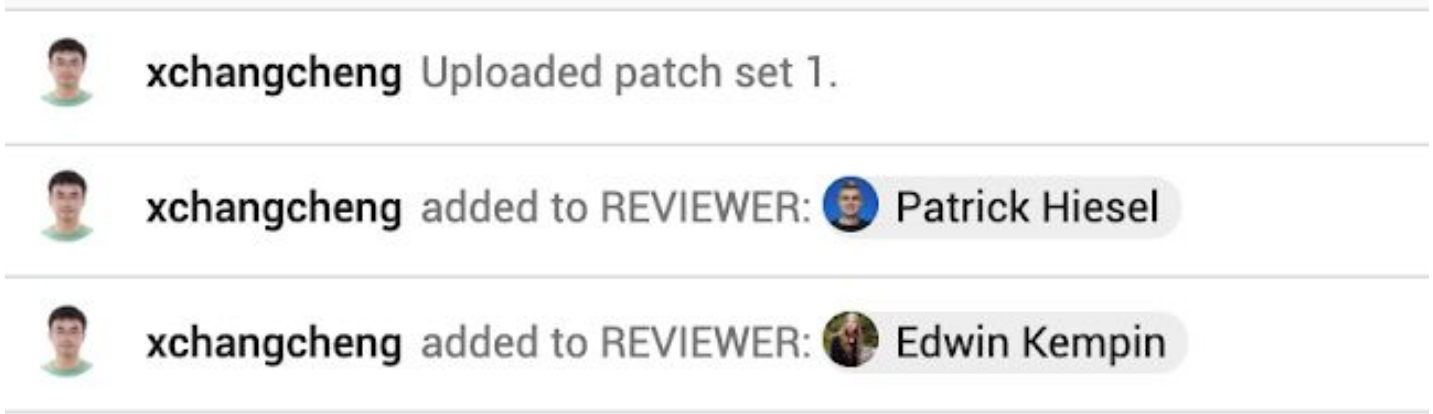
With the parent commit, the above condition is simplified
to verify the "CombinedCheckState" of required checkers
are passing.

Change-Id: I4bc92b09c5db5cdd7da6e958bda98ad973f06e73
```

Files Base → Patchset 12 | 556c840 | NO PATCHSET DESCRIPTION

Commit message
M java/com/google/gerrit/plugins/checks/acceptance/TestModule.java
A java/com/google/gerrit/plugins/checks/rules/ChecksSubmitRule.java
M javatests/com/google/gerrit/plugins/checks/acceptance/CheckerRefsIT.java
A javatests/com/google/gerrit/plugins/checks/acceptance/rules/BUILD
A javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java

We first upload our Change (PS1) and invite reviewers



We got feedback (btw, that is how you do code review)

Patrick Hiesel Code-Review -1

Patch Set 1: Code-Review-1

(16 comments)

Great to see another part of the puzzle fit in!

java/com/google/gerrit/plugins/checks/api/ListChecks.java:

Line 59: This is an API class that should only be called by the REST API servlet and the Java API. The methods you need for this change could go into Checks (in the top-level package) I think.

java/com/google/gerrit/plugins/checks/rules/ChecksSubmitRule.java:

Line 36: Please don't use any API classes here as this is internal code. If there is something you need in these classes, check if it fits into Checkers or Checks (the interfaces we have in the top-level package)

Line 86: Since you don't need backfilling here, could we just use 'Checks#getChecks'

Line 124: I think we will need this more often (e.g. when computing the dashboard state). Does it make sense to move it to a common place?

For example: Checks#getCombinedCheckState(Project.NameKey, PatchSet.Id)

Line 126: nit: applyingCheckers

Line 146: !blockingConditions.equals(ImmutableList.of(BlockingCondition.STATE_NOT_PASSING))

Line 153: If you follow the suggestion of not using API classes, this would be an OK case and just means the checker has not yet reported anything. Since it is blocking, it means we tell the combiner this is 'not started'.

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

Line 47: This can be omitted because there are separate tests that ensure change creation works as intended.

Line 48: nit: The PS-ID contains the change ID, so storing the change ID can be omitted

Line 66: This can be removed since the CheckerOperations are well tested in other places

Line 78: This can be removed since the CheckerOperations are well tested in other places

Line 81: I think it makes sense to test change submission once, but as a general test pattern, testing just the submit rule (which you can get injected into this class) is less overhead and more specific.

For example: This is what IgnoreSelfApprovalRuleIT does.

Line 85: nit: disabledCheckerDoesNotBlockSubmit (here and below)

Line 134: Sandboxed is really slow, is it possible to rewrite this test so that it is not required anymore?

Line 152: Sandboxed is really slow, is it possible to rewrite this test so that it is not required anymore?

Line 169: Please use CheckOperations instead of the API.

We then fix the problems, upload new PS, and comment

xchangcheng Uploaded patch set 2.

xchangcheng Uploaded patch set 3.

xchangcheng

Patch Set 3:

(16 comments)

java/com/google/gerrit/plugins/checks/api/ListChecks.java:

PS1, Line 59: agree. We should try to avoid using API classes directly.

java/com/google/gerrit/plugins/checks/rules/ChecksSubmitRule.java:

PS1, Line 36: Done

PS1, Line 86: Done

PS1, Line 124: Good idea. Done in the previous change of this series.

PS1, Line 126: How about "applicableCheckers"? Done in the previous change of this series.

PS1, Line 146: No. This doesn't work because comparing two different ImmutableList object will always be not equal. It works in Junit assert though.

PS1, Line 153: Ack

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

PS1, Line 47: I guess not? A "createChange" call can fail in a good reason, e.g. the caller doesn't have requried permissions. Different tests could have different setup, so I think it's better for us to verify the change was created correctly.

PS1, Line 48: Yes. That's true. But I would prefer to keep it because it's used by every test blow. It looks not that good if we always do like "testPatchSetId.changeId". Right?

PS1, Line 66: agree.

PS1, Line 78: Done

PS1, Line 81: I considered about this. Yes, it's possible to test against the rule directly. But I can see several problems from that approach.

- Those tests test the "SubmitRule" class directly, which makes them like small unit tests. But the truth is they are not, they are still integration tests which brings up the test server, and all the other setups with Gerrit APIs. So it's kind of mixture.
- There is another problem for those tests. They can't verify the rule can be called by the call correctly. IIUC, those tests will pass even when the SubmitRule isn't set up correctly in the Module.

But it has an advantage that it directly verifies the SubmitRule behavior given an input.

How do you think?

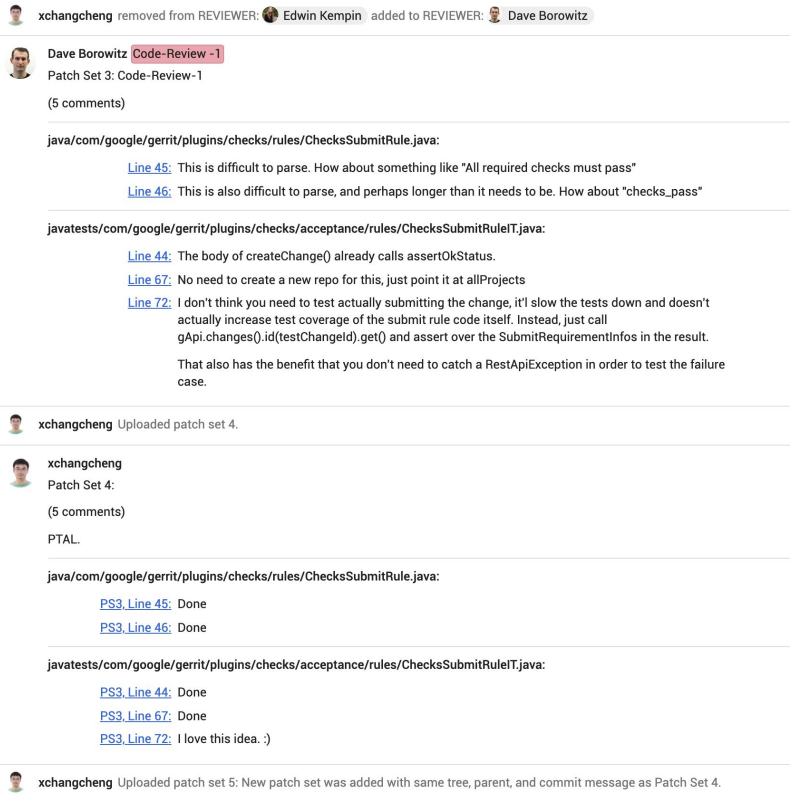
PS1, Line 85: Done

PS1, Line 134: Yes. It's not necessary. I forgot to delete it.

PS1, Line 152: Done

PS1, Line 169: good idea.

We can even invite more reviewers for feedback



xchangcheng removed from REVIEWER: Edwin Kempin added to REVIEWER: Dave Borowitz

Dave Borowitz Code-Review -1

Patch Set 3: Code-Review-1

(5 comments)

java/com/google/gerrit/plugins/checks/rules/ChecksSubmitRule.java:

Line 45: This is difficult to parse. How about something like "All required checks must pass"

Line 46: This is also difficult to parse, and perhaps longer than it needs to be. How about "checks_pass"

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

Line 44: The body of createChange() already calls assertOkStatus.

Line 67: No need to create a new repo for this, just point it at allProjects

Line 72: I don't think you need to test actually submitting the change, it'l slow the tests down and doesn't actually increase test coverage of the submit rule code itself. Instead, just call gApi.changes().id(testChangeId).get() and assert over the SubmitRequirementInfos in the result.

That also has the benefit that you don't need to catch a RestApiException in order to test the failure case.

xchangcheng Uploaded patch set 4.

xchangcheng

Patch Set 4:

(5 comments)

PTAL.

java/com/google/gerrit/plugins/checks/rules/ChecksSubmitRule.java:

PS3, Line 45: Done

PS3, Line 46: Done

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

PS3, Line 44: Done

PS3, Line 67: Done

PS3, Line 72: I love this idea. :)

xchangcheng Uploaded patch set 5: New patch set was added with same tree, parent, and commit message as Patch Set 4.

Rebase change and upload new PS when upstream master changes

xchangcheng Uploaded patch set 6: Patch Set 5 was rebased.

xchangcheng Uploaded patch set 7: Patch Set 6 was rebased.

New reviews from reviewers.
Keep fixing issues mentioned by reviewers.
Keep rebasing when upstream master changes.

And finally...

It’s submitted to master! Hooray!

Dave Borowitz Code-Review -1

Patch Set 8: Code-Review-1

(2 comments)

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

Line 44: @Inject

Line 50: You only need to do this for types which are bound in the plugin modules, not for core Gerrit classes.

xchangcheng

Uploaded patch set 9.

xchangcheng Verified +1

Patch Set 9: Verified+1

(3 comments)

javatests/com/google/gerrit/plugins/checks/acceptance/CheckerRefsIT.java:

PS7, Line 105: sgtm

javatests/com/google/gerrit/plugins/checks/acceptance/rules/ChecksSubmitRuleIT.java:

PS8, Line 44: Done

PS8, Line 50: that's true.

Dave Borowitz Code-Review +2 Patch Set 9: Code-Review+2

xchangcheng Uploaded patch set 10: Patch Set 9 was rebased.

xchangcheng Uploaded patch set 11: Patch Set 10 was rebased.

xchangcheng Verified +1 Patch Set 11: Verified+1

xchangcheng Uploaded patch set 12: Patch Set 11 was rebased.

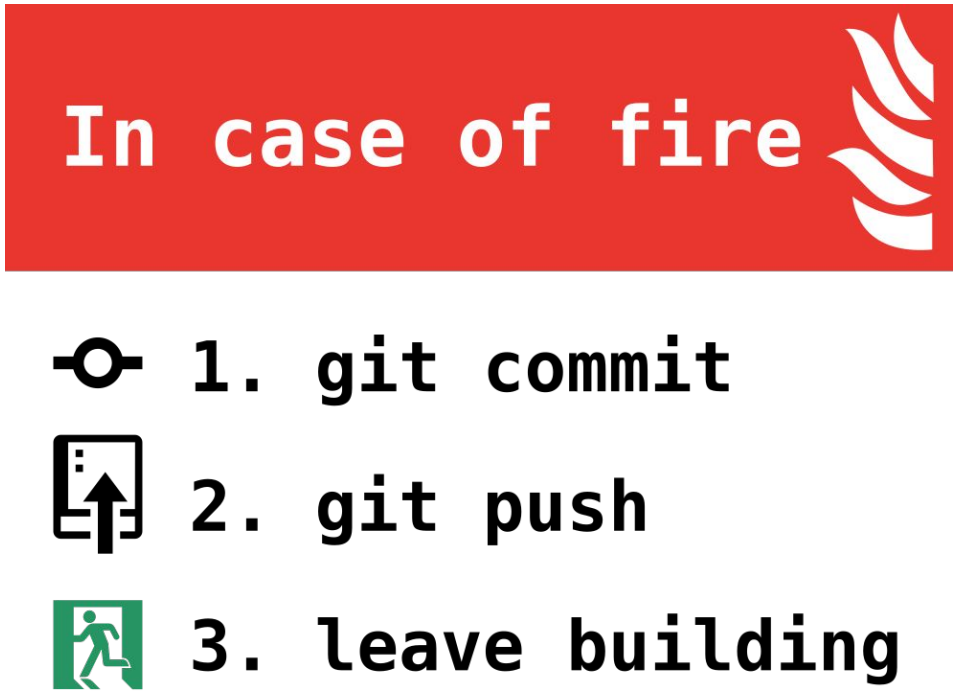
xchangcheng Verified +1 Patch Set 12: Verified+1

Dave Borowitz Change has been successfully merged by Dave Borowitz

Google

Thank You, and good luck coding!

Meanwhile, don't forget to give other people code review ❤️



Adam Yi
<i@adamyi.com>