Improving Commit Messaging: Developer Insights Survey

Thank you for taking the time to participate in this survey. The purpose of this survey is to gather feedback on a proposed enhancement to the way commit messages are generated in software development.

What is a Commit Message?

In software development, a commit message is a brief description that accompanies a set of changes (a "commit") made to the codebase. When developers make changes to the code—such as adding new features, fixing bugs, or modifying existing functionality—they "commit" these changes to the version control system (like Git).

The commit message serves as a summary of the changes made in that particular commit, helping other developers (and sometimes the original author) understand what was done and why. It is an essential part of the development process, as it provides context and history for the codebase, making it easier to review, track, and maintain the project over time.

Traditionally, a single commit message is used to describe all the changes in a commit, regardless of how many files are modified. This survey explores the potential of enhancing this practice by introducing a two-level commit message system: one message for each file changed and another summarizing all the changes. Your insights will help us understand the challenges developers face with current commit messaging practices and assess the benefits of this new approach. The survey is brief and should take no more than 3 minutes to complete.

* Indicates required question

1. Name

____

2. Gender *

Mark only one oval.

○ Male

○ Female

○ Prefer not to say

○ Other: ____________________

3. Occupation *

Mark only one oval.

○ Graduate Student

○ Undergraduate Student

○ Professor

○ Professional Software Developer

○ Other: ____________________

4. Years of Programming Experience? *

Mark only one oval.

○ Less than 5

○ Betwwen 5 - 10

○ More than 10

○ Other: ____________________

5. Are you currently working in Academia or Industry? *

Mark only one oval.

○ Academia

○ Industry

○ Other

6. Have you ever worked in any Open Source project?

Mark only one oval.

○ Yes

○ No

7. Have you ever encountered situations where the commit message provided was not descriptive enough to understand the changes made? *

Mark only one oval.

○ Yes

○ No

○ Sometimes

8. How often do you find it challenging to track back to specific file changes when only a single commit message is available for multiple file changes? *

Mark only one oval.

○ Frequently

○ Occasionally

○ Rarely

○ Never

9. What are the problems with current commit messages?

Check all that apply.

□ Lack of Detail: Commit messages are often too brief and do not provide enough context or explanation for the changes made.

□ Difficulty in Tracking Changes: With only one commit message for multiple file changes, it can be challenging to track and understand the specific modifications made to individual files within a commit.

□ Inconsistency: Commit messages vary widely in style and format, making it difficult to follow a consistent pattern across the codebase.

□ Ambiguity: Commit messages are sometimes vague or unclear, leaving room for misinterpretation of the changes.

□ Missing Information: Important details, such as why certain changes were made or the impact of the changes, are frequently omitted.

□ Irrelevance: Commit messages occasionally include information that is not directly related to the changes, making it harder to understand the core modifications.

□ Other

10. Would you find it useful to have separate commit messages for each file change in a diff, along with an overall commit message? *

Mark only one oval.

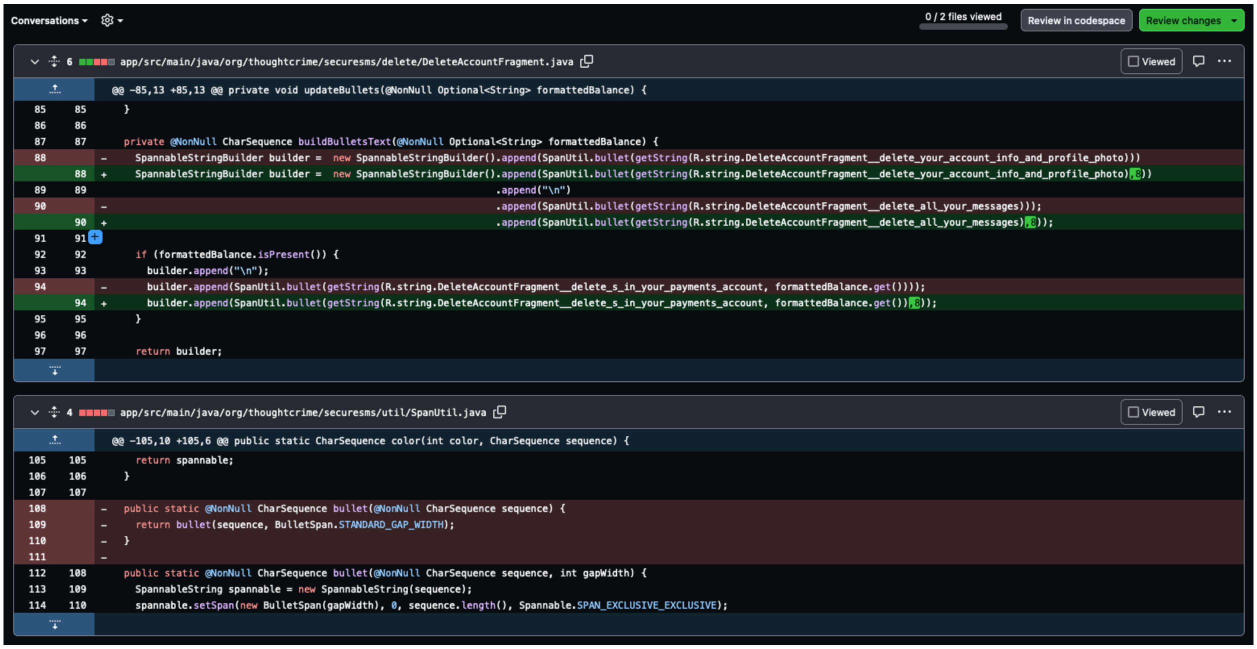
○ Extremely important

○ Somewhat important

○ Not important

Below is an image of a commit that has changed two files. In the traditional approach, we get one commit message for all the file changes, as in this case with two files. However, in some cases, the number of files changed can be greater, such as 4, 6, 8, or more. If you are unable to see the image clearly due to its small size, you can view it at this link: https://github.com/signalapp/Signal-Android/pull/13488/files

Now, based on this, please answer the next question.



6 app/src/main/java/org/thoughtcrime/securesms/delete/DeleteAccountFragment.java

```
@@ -85,13 +85,13 @@ private void updateBullets(@NonNull Optional<String> formattedBalance) {
   }

   private @NonNull CharSequence buildBulletsText(@NonNull Optional<String> formattedBalance) {
-    SpannableStringBuilder builder =  new SpannableStringBuilder().append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_your_account_info_and_profile_photo)))
+    SpannableStringBuilder builder =  new SpannableStringBuilder().append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_your_account_info_and_profile_photo),8))
                                                                  .append("\n")
-                                                                 .append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_all_your_messages)));
+                                                                 .append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_all_your_messages),8));

     if (formattedBalance.isPresent()) {
       builder.append("\n");
-      builder.append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_s_in_your_payments_account, formattedBalance.get())));
+      builder.append(SpanUtil.bullet(getString(R.string.DeleteAccountFragment__delete_s_in_your_payments_account, formattedBalance.get()),8));
     }

     return builder;
```

4 app/src/main/java/org/thoughtcrime/securesms/util/SpanUtil.java

```
@@ -105,10 +105,6 @@ public static CharSequence color(int color, CharSequence sequence) {
     return spannable;
   }

-  public static @NonNull CharSequence bullet(@NonNull CharSequence sequence) {
-    return bullet(sequence, BulletSpan.STANDARD_GAP_WIDTH);
-  }
-
   public static @NonNull CharSequence bullet(@NonNull CharSequence sequence, int gapWidth) {
     SpannableString spannable = new SpannableString(sequence);
     spannable.setSpan(new BulletSpan(gapWidth), 0, sequence.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
```

11. For the diff of a commit shown in the figure above, which format of commit message would you prefer: *

Mark only one oval.

○ A single commit message summarizing all file changes (traditional approach)

○ Separate commit messages for each file change, along with an overall summary

12. Do you think having two-level commit messages (file-specific and overall) would improve the quality and traceability of commit history? *

Mark only one oval.

○ Yes

○ No

○ Unsure

13. If you had to implement a new commit message strategy, would you prefer to adopt the proposed two-level message approach or stick with the traditional single commit message? *

Mark only one oval.

○ I would adopt the two-level message approach

○ I would stick with the traditional single commit message

○ I am open to either approach

14. Any suggestions or feedback?

This content is neither created nor endorsed by Google.

Google Forms