A Design of Threaded Messages for Team Chat Platform

Yujin Zhang

January 2018

Problem

Team chat web applications are being widely used. However, current solutions on the threaded messages feature have room for improvement. Threaded messages are useful since a team can have several conversations going on in a chat channel, but one conversation may drown as new messages are posted and new topics are raised. An ideal solution of threaded messages should make it easy for users to keep up and stay engaged in certain conversations while keeping up with the flow of messages in the whole channel. Therefore, I proposed, and are going to implement, my design of threaded messages upon a basic web chat application.

Background

Let's analyze the existing solutions of threaded messages from three popular web team chat application: Slack, Rocket.Chat, and Flowdock.

Slack adopts a forum-liked style. Each thread is leaded by an existing message. All replies to that thread is hidden in the main channel, and can only be seen by opening a new panel. Users' feedback on this design is controversial [1] since it's inconvenient to follow what is going inside a thread unless we keep the thread panel always opened.

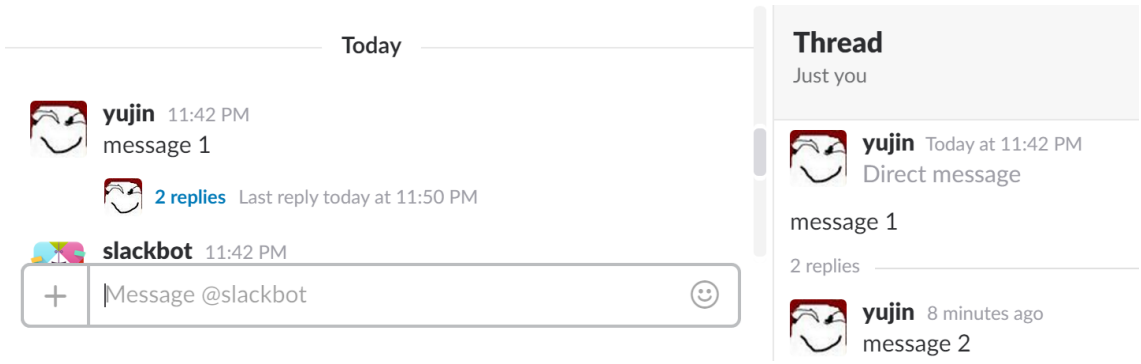


Figure 1. Threaded messages in Slack

Rocket.Chat adopts the email thread pattern. When user replies to a thread, all previous messages in the thread will be automatically append to the end of this reply. As a thread grows longer, a reply may occupies too much space. In addition, when a thread is not interleaved by other messages, users may not need to see the context being duplicated for every reply.

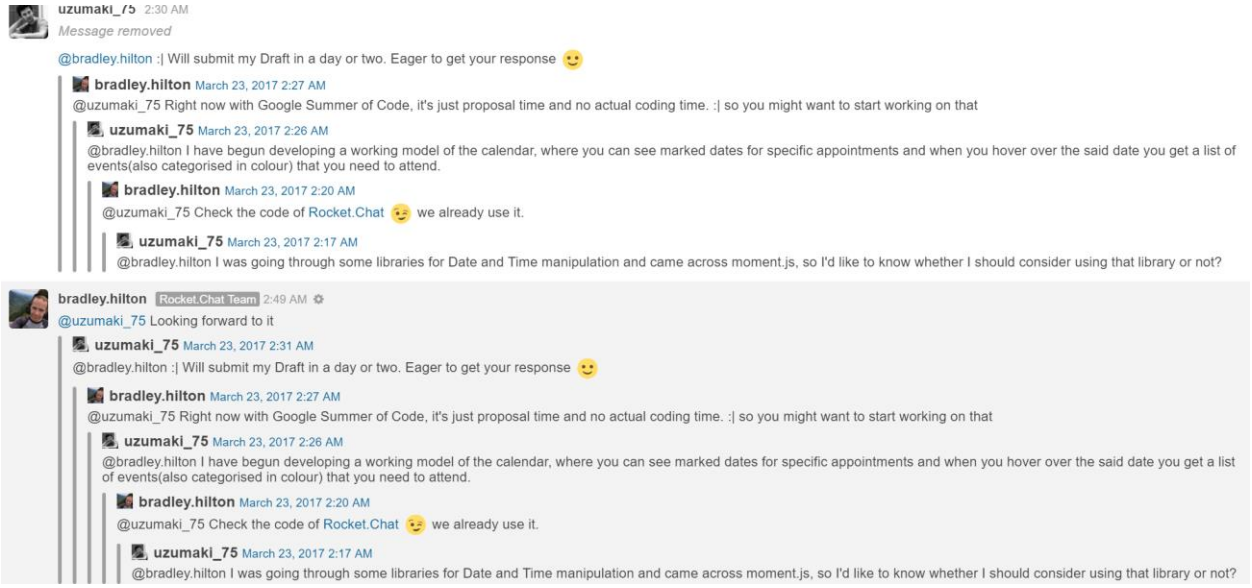
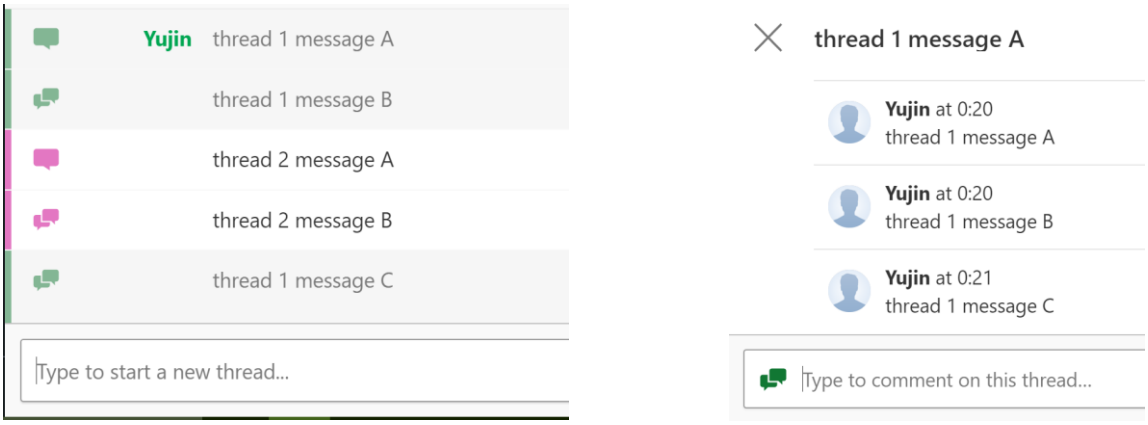


Figure 2. Threaded messages in Rocket.Chat

Flowdock displays messages of different threads by highlighting them with different colors. To reply to a thread, the user clicks the thread icon of any message belong to that thread, and then enter a new view with only messages of this thread displayed. In this case, the user loses the flow of messages in the whole channel when they are engaging in a thread, and hence have to switch between thread view and regular view back and forth.



Channel View **Thread View**

Figure 3. Threaded messages in Flowdock

Solutions

Several problems need to be considered in order to build a user-friendly threaded messages feature: (1) How to indicate if a message is in a thread and how to create a new thread (2) How to reply to a thread (3) How to clearly display the context of a thread

(1) How to indicate if a message is in a thread and how to create a new thread

Here I mainly adopt Flowdock's design, which received positive comments, and may make it color-blind friendly if I have time. Labels are added to the left of messages that are belongs to thread. Colors (and maybe textures) are assigned to labels by the system, so that each thread is mapped with a color. For messages that are not in any thread, the places for their thread labels will be empty. If a user single-click on the empty space for the thread label of a message, a new thread containing only this message will be created.

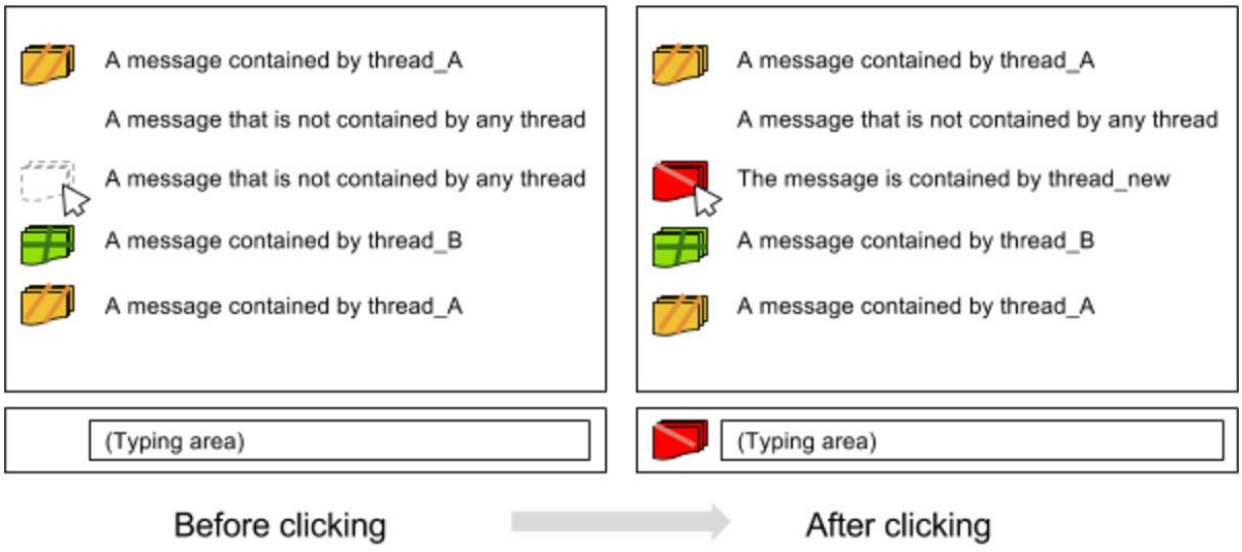


Figure 4. How to create a new thread and how messages in threads are labelled.

(2) How to reply to a thread

If a user single-clicks the thread label of a previous message, that thread label will be copied to user's input area. If he then sends out the message with this thread label, he is replying to the thread of the message that he clicked before. A user will always have at most one label in his typing area, and the position of the label is fixed. There will also be a cross mark on the right of this label, so that the user will be able to delete it. In this way, users can keep track of the flow of messages in the whole channel while engaging in a thread.

Figure 5. How to reply to a thread.

(3) How to clearly display the context of a thread

If the user double-clicks the thread label of a message, we will show a filtered view which only contains messages belong to this thread. The user can still send messages in this view, and the label of this thread will always be in his typing area, so that the message he sent will automatically become a reply to this thread. This view can be used when the user wish to highly focus on a certain thread.

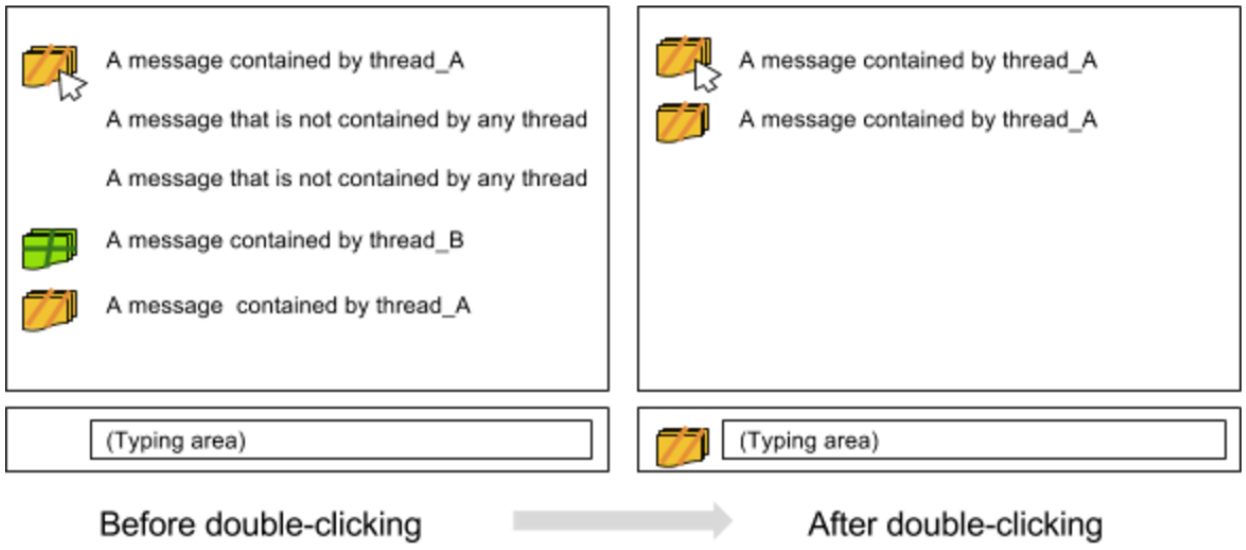


Figure 6. How to get the filtered view of a thread.

Deliverables

- A basic chat platform which enable users to send and view messages.
- Generate thread label in input area and make thread replying works properly.
- Label threaded messages with proper colors.
- Enable user to change and delete the thread label in typing area.
- Provide filtered view of messages of a thread and enable user to switch between views.
- Create proper documentation and tests.