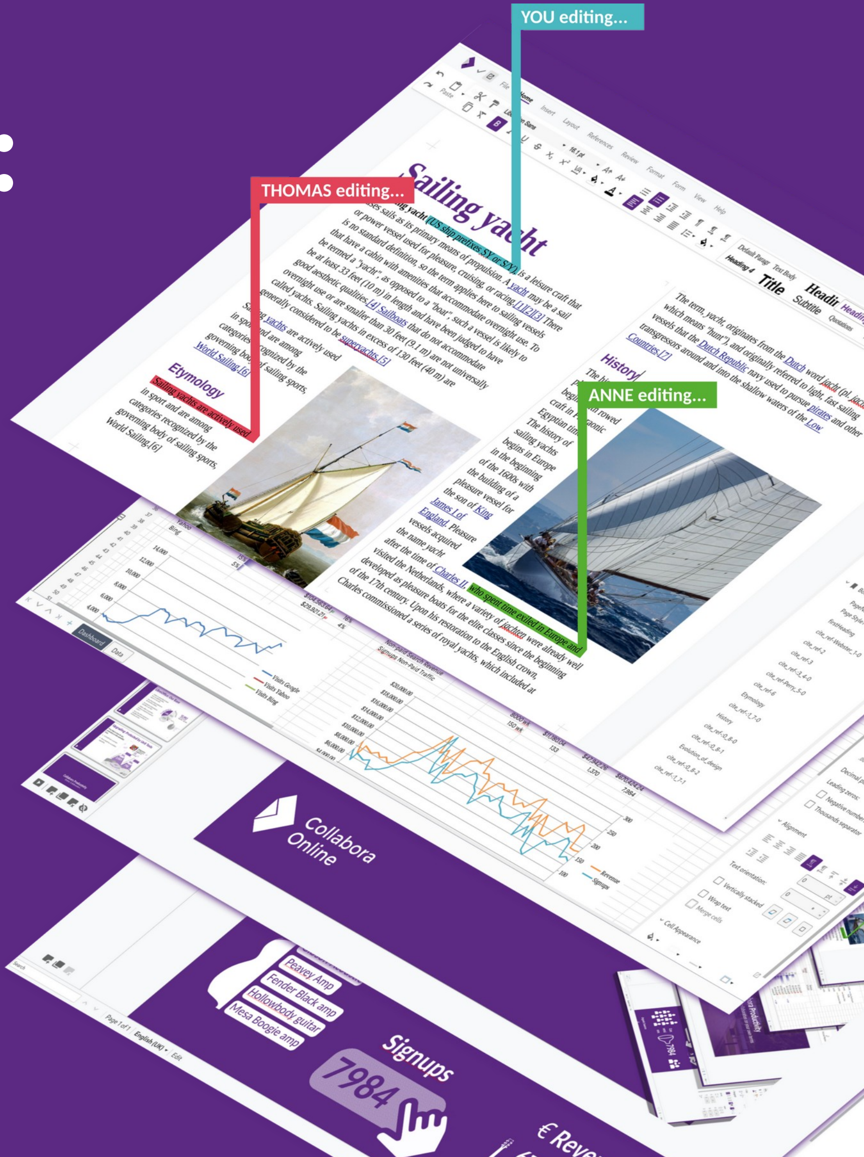
Enhancing Performance: Async Dialogs

Vivekkumar Javiya
Associate Software Developer
vivek.javiya@collabora.com



Collabora Online

LibreOffice Technology

Technical Day
COOL days

Introduction

- Welcome to today's presentation on enhancing performance with asynchronous dialogs.
- We'll delve into the challenges of collaborative dialog experiences and how async dialogs can solve this issue & Enhance Performance.

Technical Day

Collaborative Dialogs Challenge

- Scenario: Multiple users accessing the same dialog simultaneously.
- Issue: Flickering or changes not applied when users close the dialog in quick succession.
- Example:
  - User A opens dialog,
  - User B opens the same dialog,
  - User A does some action in dialog and clicks “OK”
  - User B does some action in dialog and clicks “OK”
  - Expected: changes were applied in both sessions, dialog is closed correctly
  - Wrong result: dialog doesn’t close but starts to flicker, changes are not applied

Collabora Online

Technical Day

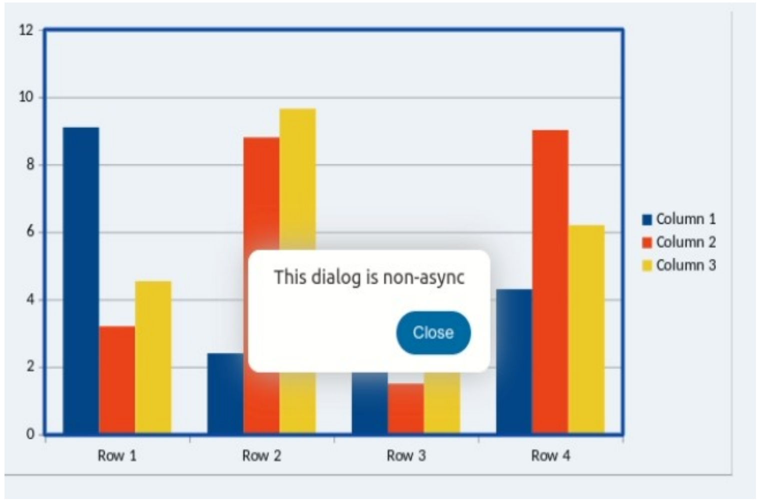
Solution : Async dialogs

- Allowing multiple users to work on the same dialog independently.
- Since the process of dialog is asynchronous, it naturally avoids blocking users to work on same dialogs, thereby indirectly enhancing performance.

Collabora Online

Identify Asynchronous Dialogs?

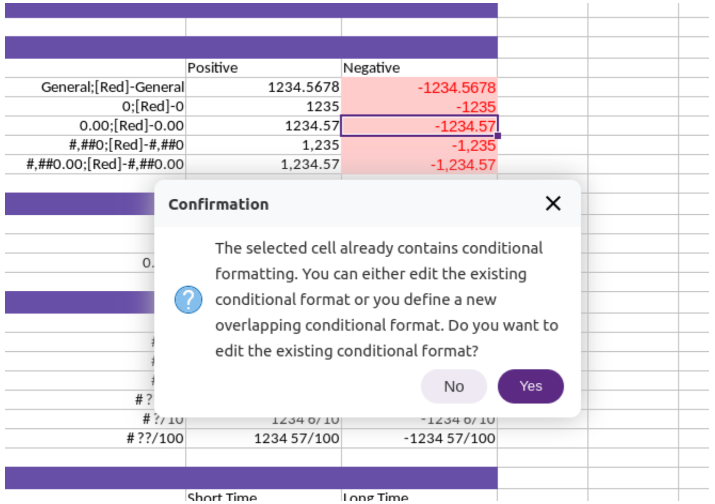
- Ideal way to open in multiple user/screens and test dialogs in parallel.
- Easiest method: In debug mode, display an additional dialog when a non-async dialog is opened, Special thanks to Michael Meeks for the patch!
- From flame graph analysis: It's worth checking the nature of long time-taking dialog boxes.



Technical Day COOL days

Find Code Pointers?

- Copy the dialog title string, typically defined in sc/inc/globstr.hrc, or utilize git grep.
  - Ex: `STR_EDIT_EXISTING_COND_FORMATS` → `The selected cell already contains conditional....`
- Search using the macro of that string to easily locate dialogs in the codebase.



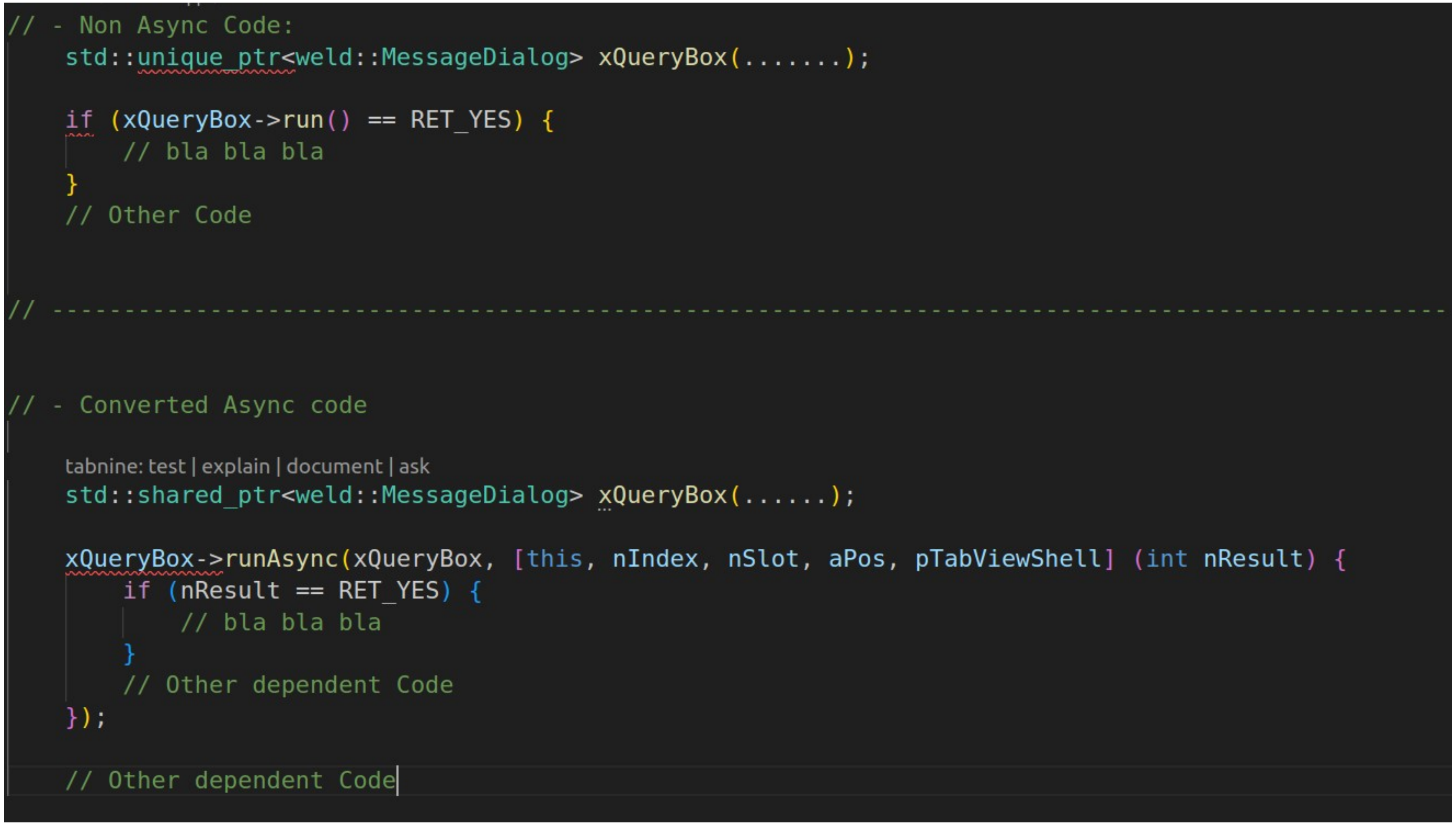
	Positive	Negative
General;[Red]-General	1234.5678	-1234.5678
0;[Red]-0	1235	-1235
0.00;[Red]-0.00	1234.57	-1234.57
#,##0;[Red]-#,##0	1,235	-1,235
#,##0.00;[Red]-#,##0.00	1,234.57	-1,234.57
# ?/10	1234 6/10	-1234 6/10
# ??/100	1234 57/100	-1234 57/100

Confirmation

The selected cell already contains conditional formatting. You can either edit the existing conditional format or you define a new overlapping conditional format. Do you want to edit the existing conditional format?

Collabora Online

Converting Dialogs to Async - Code



```
// - Non Async Code:
    std::unique_ptr<weld::MessageDialog> xQueryBox(.......);

    if (xQueryBox->run() == RET_YES) {
        // bla bla bla
    }
    // Other Code


// ---------------------------------------------------------------------------------------------------


// - Converted Async code

    tabnine: test | explain | document | ask
    std::shared_ptr<weld::MessageDialog> xQueryBox(......);

    xQueryBox->runAsync(xQueryBox, [this, nIndex, nSlot, aPos, pTabViewShell] (int nResult) {
        if (nResult == RET_YES) {
            // bla bla bla
        }
        // Other dependent Code
    });

    // Other dependent Code
```

Technical Day

Converting Dialogs to Async - Code

- First, convert unique_ptr to shared_ptr if it does not already exist.
- Call the runAsync method with the appropriate callback lamda function. Worth checking runAsync method is static or class method.
- For some classes, we have a different method, StartExecuteAsync, instead of runAsync.
- The callback function should accept nResult parameter, which represents the response of the dialog action, Ex. RET_YES , RET_NO.

Converting Dialogs to Async - Code

- You can pass references and variables to a lambda function using a capture list, which can be mentioned in square brackets.`[...]`
- Capturing `this` keyword is often useful to access member variables and member functions of the current object within the lambda.
- It can take a longer time between dialog initialization and calling the callback (the user waits a few seconds). So, we should pass variables and references in the capture list accordingly.

Technical Day

Credits

- Tomaz (tomaz.vajngerl@collabora.com)
- Caolán (caolan.mcnamara@collabora.com)
- Szymon (szymon.klos@collabora.com)
- Pedro (pedro.silva@collabora.com)

Collabora Online

LibreOffice Technology

Technical Day
COOL days

Thank you!

By Vivekkumar Javiya

T S P D

@CollaboraOffice
hello@collaboraoffice.com
www.collaboraoffice.com