Collabora Productivity

Dialog tunneling in LibreOffice Online

By Pranav Kant

Software Engineer at Collabora Productivity

pranavk@collabora.com

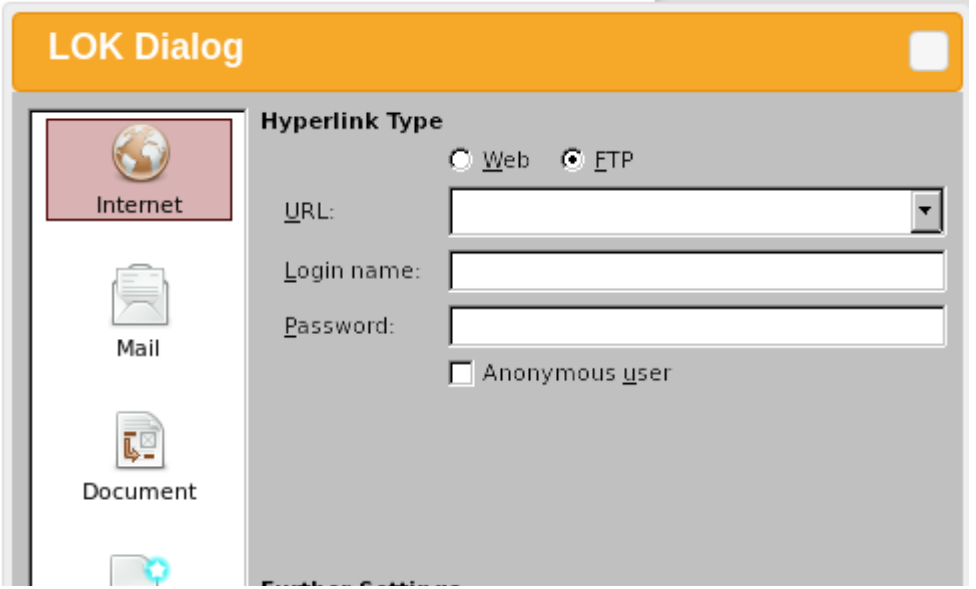
How to get more features?

Approach #1

- Write a LibreOfficeKit API for each and every set of features
- Identify all the places where state is changed in LO core
  - And invoke LOK callbacks to notify LOK clients
- Augment existing UNO commands to make them LOK compatible
- Handle any LOK corner cases in LibreOffice core
- And then implement all of it in Javascript (LOK client)
- Months and months of work per feature
  - With increasing LOK conditional code in LO core

Or

- Reuse the work as much as possible
- Less LOK conditional code
  - Easier to maintain
- Less work overall per feature



Overview

- Request to open the dialogs in the core
- Paint them
- Send the pixels to browser
- Forward mouse/key events from browser to core
- Invalidate and render the updated dialog

Diving Deep

Implementation

- UNO commands used as Identifiers
- Each SfxBaseModel classes (in sd, sc, sw) implement IDialogRenderable
  - Contains the LOK dialog related methods
  - Handles mouse/key events
  - Handle dialog child mouse/key events
  - And LOK dialog related callbacks
    - Notifies dialog invalidation, close

Painting into VD

- Not possible to get the size of the dialog easily before it’s realized
- Provide a larger surface to draw the dialog into
- Return the width and height in paintDialog call
- Trim the original surface to the size of the dialog

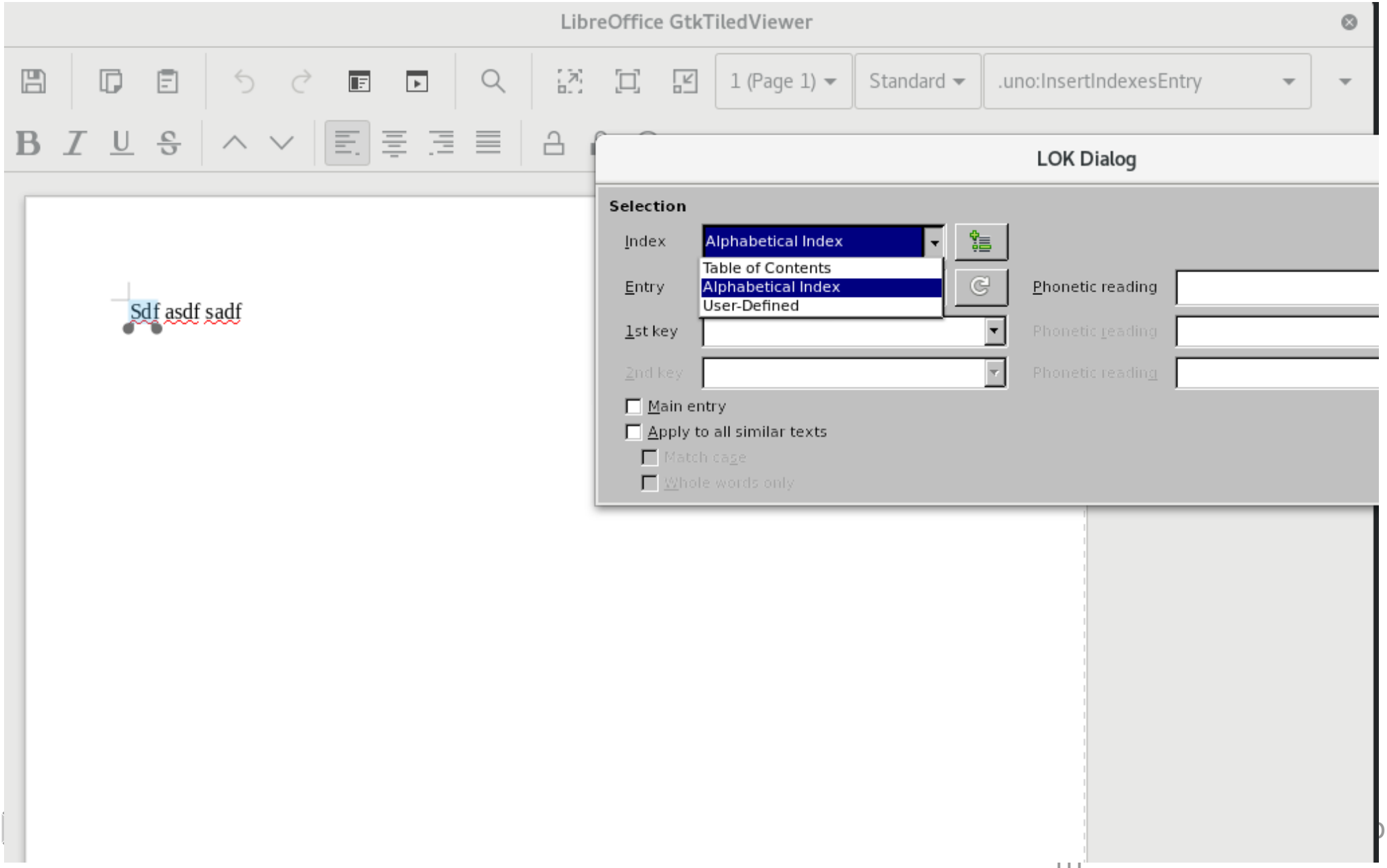
Invalidation

- Dialog consists of several widgets (each vcl::Window)
- Each one is invalidated separately
- But we want per dialog invalidation only
- We catch invalidation of each control in “Control” base class
  - And invalidate the parent dialog
- Suppress the duplicate invalidation callbacks in CallbackFlushHandler
- Supress invalidation during painting
  - Set/unset dialog painting flags before/after dialog painting

Invalidation

- Invalidation callbacks are fired from sfx2 module
  - All view shell objects in sfx2
  - Many LOK helper methods already available
- But the invalidation happens in vcl/
  - And vcl/ does not (should not) depend on sfx2, so cannot fire callbacks directly from vcl/
- Register IDialogRenderable (implemented by SfxBaseModel of each module) with vcl/
  - IDialogRenderable contains methods that can then fire callbacks

Child dialog windows / floating windows



Child dialog windows / floating windows

- PaintActiveFloatingWindow in IDialogRenderable
  - Broadcast coordinates through a callback
    - LOK_CALLBACK_DIALOG_CHILD

{ "dialogId": "IndexEntryDialog",

"action": "invalidate",

"position": "77, 54" }

Demo

What’s still left to do? (Future plans)

- Modeless dialogs only for now
  - Only 7 modeless ones in writer
- Modal dialogs stop the world
  - Which is undesirable in Online
- Some modal dialogs can easily be modeless
- Turn them to modeless

Collabora Productivity

Any Questions?

By Pranav Kant

pranavk@collabora.com