

Document Foundation Planet - Latest News

- [Miklos Vajna: Per-user track changes recording in Writer](#) (2025/04/02 11:33)

Writer has the concept of recording tracked changes or not: if recording, typing into a document or deleting content will create tracked changes of type insertion or deletion. So far this was a per-document setting, but now individual users can enable or disable this as they wish. This work is primarily for Collabora Online, but the feature is available in desktop Writer as well. Motivation¶ When Alice keeps typing and Bob enables change tracking, then surprisingly the typed characters of Alice will form a tracked insertion, which is surprising, since that was not the case a second ago and Alice didn't do anything other than typing. Giving users a choice if they enable recording for just this user or for all users fixes this problem. Results so far¶ Here is how the per-user (technically per-view) tracked changes recording looks like: Per-view tracked changes recording As you can see, the user on the left has recording turned on and this doesn't influence the user on the right, while this was not possible before. How is this implemented?¶ If you would like to know a bit more about how this works, continue reading... :-)

As usual, the high-level problem was addressed by a series of small changes. Core side: cool#11226 sw per-view redline on: support this-view <-> all-views transition cool#11226 sw per-view redline on: state for the per-view and per-doc commands cool#11226 sw per-view redline on: allow both per-view and per-doc Related: cool#11226 sw per-view redline on: fix ratio buttons of is-show cool#11226 sw per-view redline on: add view-aware getter cool#11226 sw per-view redline on: move the setter to the model cool#11226 sw per-view redline on: add new flag in SwViewOption cool#11226 sw layout xml dump: show some more view options Online side: cool#11226 sw per-view redline on: add tri-state UI in the notebookbar Want to start using this?¶ You can get a development edition of Collabora Online 25.04 and try it out yourself right now: try the development edition. Collabora intends to continue supporting and contributing to LibreOffice, the code is merged so we expect all of this work will be available in TDF's next release too (25.8).

- [Official TDF Blog: Video: Document Freedom Day with the Nepalese LibreOffice community](#) (2025/04/02 10:07)

On March 26, we celebrated Document Freedom Day. Mike Saunders from The Document Foundation, the non-profit behind LibreOffice, gave an online talk about the importance of open standards and free software: Please confirm that you want to play a YouTube video. By accepting, you will be accessing content from YouTube, a service provided by an external third party. YouTube privacy policy If you accept this notice, your choice will be saved and the page will refresh. Accept YouTube Content

- [Official TDF Blog: LibreOffice project and community recap: March 2025](#) (2025/04/01 13:41)

Here's our summary of updates, events and activities in the LibreOffice project in the last four weeks – click the links to learn more... We started the month by posting some extra videos from the open source tracks of the LibreOffice Conference 2024 in Luxembourg. Check them out! Then we announced that LibreOffice is taking part in the Google Summer of Code 2025. New features will come to the suite thanks to independent developers who will use the summer to learn new things. Our LibreOffice 25.2 “New Features” video has subtitle translations in 18 languages, thanks to our worldwide localisation communities. Please confirm that you want to play a YouTube video. By accepting, you will be accessing content from YouTube, a service provided by an external third party. YouTube privacy policy If you accept this notice, your choice will be saved and the page will refresh. Accept YouTube Content Next, we posted an update on the German state of Schleswig-Holstein moving 30,000 PCs to

LibreOffice. In March, we recorded Episode #2 of the LibreOffice podcast – this time looking at design, UI and UX in free and open source software. (Also available on PeerTube). Please confirm that you want to play a YouTube video. By accepting, you will be accessing content from YouTube, a service provided by an external third party. YouTube privacy policy If you accept this notice, your choice will be saved and the page will refresh. Accept YouTube Content On March 26 we celebrated Document Freedom Day 2025, which raises awareness about the problems of proprietary standards, and encourages people to move to open standards like the Open Document Format. In terms of the suite, there were two minor updates to LibreOffice in March – 25.2.2 and 24.8.6. All users are recommended to stay up-to-date. Then we posted a translation from our Spanish blog, about how LibreOffice Base and Firebird work really well together. And finally, we announced the two proposals for the LibreOffice Conference 2025 – Luxembourg and Budapest. TDF Members are currently voting to decide where the event will be held. Keep in touch – follow us on Mastodon, Bluesky, X (formerly Twitter), Reddit and Facebook. Like what we do? Support our community with a donation – or join our community and help to make LibreOffice even better!

- [Official TDF Blog: LibreOffice Conference 2025: Location proposals](#) (2025/03/31 14:23)

The Document Foundation (TDF) has received two different proposals for the organisation of the LibreOffice Conference 2025. TDF Members will receive an email asking them to cast a vote and decide which will be the final venue. Budapest Full application here City: Budapest, the capital of Hungary and a former part of the Austro-Hungarian Empire, is well known for its stunning landscape, with the Danube River and surrounding hills, its Belle Époque architecture, and its vibrant atmosphere. The city offers numerous attractions, including the medieval Buda Castle district and its famous thermal spas. As a popular tourist destination, Budapest boasts a wide range of accommodations, cafés, parks, and a bustling nightlife with many pubs and entertainment venues. Additionally, Budapest serves as Hungary’s administrative, cultural, and educational centre, with a population of around one million. The city is home to numerous museums and academic institutions. Entity: The LibreOffice Conference 2025 in Budapest would be co-organized by the FSF.hu Foundation and the ELTE University Faculty of Informatics. The FSF.hu Foundation, established nearly 25 years ago, was created to support the localization and promotion of FLOSS in Hungary. In addition to handling financial matters, FSF.hu has offered to issue visa invitation letters. A list of countries requiring a visa for travel to Hungary can be found on the Hungarian Ministry of Foreign Affairs website. Dates: The expected date for the conference is September 1 – 5, 2025. Venue: The conference venue will be located in the heart of Budapest, at ELTE University’s Faculty of Informatics, one of Hungary’s leading universities. The venue, near by the Danube River offers a spacious university campus with numerous large and small rooms, computer labs, and open spaces ideal for community gatherings, meeting the typical needs of a LibreOffice Conference. The venue provides a reliable Wi-Fi connection suitable for a conference of this scale, and the auditoriums used for the event will be wheelchair accessible. Additionally, the university can provide personnel for video recording during the main conference days. Team: The organizing team consists of experienced contributors from the local LibreOffice community, including Annabella Szép, Anikó Kelemenné Husi, Gábor Kelemen, András Tímár, Miklós Vajna, Balázs Varga, and Attila Szűcs. Each of them has extensive experience in developing, testing, or teaching LibreOffice. This dedicated team shares a deep commitment to the project, working together effectively as a well-coordinated group. From ELTE Faculty of Informatics, Ágnes Erdősné Németh is responsible for managing the conference venue. Gábor Kelemen, head of the organizing team, will oversee visa and sponsor relations. After many years of collaboration and contributing to LibreOffice’s growth, we now hope to be awarded the opportunity to host the LibreOffice 2025 Conference in Budapest. Luxembourg Full application here City: Belval is the new technology and University campus being developed next to Esch-sur-Alzette, the second largest city in Luxembourg, and is optimally positioned in Europe to provide visibility to LibreOffice and its community as well as for engaging with the various

EU institutions it hosts. Being also bordering with France, Belgium and Germany it facilitates the participation to the conference of organisations within the greater region creating even more opportunities to promote LibreOffice and the complementary Open Source software that can form the foundations for the technological autonomy/sovereignty that many countries are now looking to achieve. Entity: TDF will be the legal entity handling financials and sponsorships directly. Dates: To guarantee that all participants will have a hotel room on campus we selected the week starting the 29th of September 2025. Venue: As it has been proven to be an excellent location, we will be hosting it again in last year's venue, within the Belval campus of the University of Luxembourg, and we will work to get even more support from ministries, local organizations, and of course the university itself. More information about the location can be found here. Team: The organisation of the conference is a joint effort made by several local volunteers and organisations coordinated by Paolo Vecchi with the support of the Digital Learning Hub and 42 School which are providing the venues. So those are the two proposals, and as mentioned, TDF Members will receive an email asking them to cast a vote and decide which will be the final venue. Why be a member of TDF?

- [Official TDF Blog: LibreOffice Base and Firebird - a special relationship](#) (2025/03/28 10:14)
(Translated from the Spanish original.) Juan C. Sanz writes: The Firebird database is distinguished by its unique features within the LibreOffice Base compatible database ecosystem. Why do I consider Firebird to be special? Because it is the only database engine that supports all possible forms of connection in Base and also allows the creation of both embedded, external and server databases directly from LibreOffice, without having to use specific tools. LibreOffice offers the following Firebird connection options: Embedded database Standalone database file (no server required) Database server via internal driver Additionally, like other database servers, it is possible to establish a connection via JDBC or ODBC connectors. These connectors are available free of charge and as open source software on the official Firebird website. Advantages of multiple connection options. Firebird offers several ways of connection that represent important advantages: Embedded database: The embedded or internal database consists of a *.odb file containing all the database facilities (table view, query designer, forms, reports, macros) together with the data. This option is especially easy and accessible for users with little database experience. Simply enable the experimental features of LibreOffice to start using it. It is ideal for learning basic database concepts and the Base tool. External database file: For advanced users looking to work more rigorously, it is recommended to migrate from embedded databases to external files. This type of connection does not require additional installations; a new file can be created using the Connect to an existing database option, rather than the Create a new database option (the nomenclature can be confusing). This connection method offers greater security by storing the data in a separate FDB file from the Base ODB file. In addition, modifications are saved instantly, which reduces the risk of data loss in the event of computer crashes or failures. In the long term, external Firebird files can be connected to servers without modification, as long as the versions are compatible. Transformation between different versions of the FDB file can be done easily and smoothly. Firebird database server: In this case, we will have a Base file with the functionalities of forms, query designer, reports and macros and the data will be hosted in a Firebird database server. The server is a software that does not have to be installed in a special computer, it could even be installed in the same computer that we use to connect to it. This type of connection provides specific advantages: Allows simultaneous access by multiple users Facilitates permissions management and data access control Provides greater speed in the delivery of information The connection to the server via the internal driver provides a fast and easy way to configure connection. In any case, ODBC and JDBC connections also work without problems, although, as they require an intermediate connector, they are usually slower. For all these reasons, I consider that Base and Firebird are an ideal combination.
- [Official TDF Blog: Announcement of LibreOffice 25.2.2 and LibreOffice 24.8.6](#) (2025/03/27 13:27)

Berlin, 27 March 2025 – The Document Foundation announces the availability of LibreOffice 25.2.2, the second minor release of the recently announced LibreOffice 25.2 family [1], and LibreOffice 24.8.6, the sixth minor release of the LibreOffice 24.8 family [2], for Windows (Intel, AMD and ARM), macOS (Apple Silicon and Intel) and Linux. LibreOffice is the best office suite for users who want to retain control over their individual software and documents, thereby protecting their privacy and digital life from the commercial interference and the lock-in strategies of Big Tech. All LibreOffice releases can be downloaded from www.libreoffice.org/download/. What makes LibreOffice unique is the LibreOffice Technology Platform, the only one on the market that allows the consistent development of desktop, mobile and cloud versions – including those provided by companies in the ecosystem – capable of producing identical and fully interoperable documents based on the two available ISO standards: the open ODF or Open Document Format (ODT, ODS and ODP) and the proprietary Microsoft OOXML (DOCX, XLSX and PPTX). The latter hides a huge number of artificial (and unnecessary) lock-in complexities that create problems for users convinced they are using a standard format. Products based on LibreOffice Technology are available for all major desktop operating systems (Windows, macOS, Linux and ChromeOS), mobile platforms (Android and iOS) and the cloud. For enterprise-class deployments, TDF strongly recommends the LibreOffice Enterprise family of applications from ecosystem partners – for desktop, mobile and cloud – with a wide range of dedicated value-added features and other benefits such as SLAs and backports of security patches for several years: www.libreoffice.org/download/libreoffice-in-business/. English manuals for LibreOffice 25.2 and LibreOffice 24.8.6 can be downloaded from books.libreoffice.org/en/. End users can get first-level technical support from volunteers on the user mailing lists and the Ask LibreOffice website: ask.libreoffice.org. Download LibreOffice Both LibreOffice 25.2.2 and LibreOffice 24.8.6 are immediately available from www.libreoffice.org/download/. LibreOffice 25.2.2 is targeted at power and tech-savvy users, while LibreOffice 24.8.6 is targeted to users who don't need the latest features and prefer a version that has undergone more testing and bug and regression fixes. LibreOffice is the only office suite designed to meet the actual needs of the user – not just their eyes. It offers a range of interface options to suit different user habits, from traditional to modern, and makes the most of different screen sizes, optimising the space available to put the maximum number of features just a click or two away. It is also the only software for creating documents (that may contain personal or confidential information) that respects the user's privacy, ensuring that the user can decide if and with whom to share the content they create, thanks to the standard and open format that is not used as a lock-in tool, forcing periodic software updates. All this with a feature set that is comparable to the leading software on the market and far superior to that of any competitor. LibreOffice users, free software advocates and community members can support The Document Foundation and the LibreOffice project with a donation at www.libreoffice.org/donate. [1] Fixes in RC1: <https://wiki.documentfoundation.org/Releases/25.2.2/RC1>. Fixes in RC2: <https://wiki.documentfoundation.org/Releases/25.2.2/RC2>. [2] Fixes in RC1: <https://wiki.documentfoundation.org/Releases/24.8.6/RC1>. Fixes in RC2: <https://wiki.documentfoundation.org/Releases/24.8.6/RC2>.

- [Michael Meeks: 2025-03-26 Wednesday](#) (2025/03/26 15:17)
Crit-sit stand-up call, catch-up with Dave, last-stage interview. Published the next strip: building for maintainability Monthly all-hands call, sales call.
- [Michael Meeks: 2025-03-25 Tuesday](#) (2025/03/25 21:00)
Planning call, sync with Karen, last-round interview. Monthly mgmt meeting, larger partner meeting. Some recreational hacking in the evening.
- [Michael Meeks: 2025-03-24 Monday](#) (2025/03/24 21:00)
Mail chew, admin catch-up, 1:1 calls much of the day. PCC meeting in the evening.

- [Michael Meeks: 2025-03-23 Sunday](#) (2025/03/23 21:00)
Use-up breakfast, packed the car, collected Janine, and off to All Saints Exeter for M's baptism - wonderful service - lovely to have Janine & Barbara & Greer there too. Good talk(s) from Paul Sutton. Met & talked to lots of interesting people of varying age supporting M. Out for a roast lunch together at a nearby pub. Admired some bridges over the river, before dropping M. and collecting bits & pieces for easter break. Long drives back via a fine dinner with R&A in Kennilworth. Sleep.
- [Michael Meeks: 2025-03-22 Saturday](#) (2025/03/22 21:00)
Out to buy some breakfast, dropped babes to have brunch with M. Shopping, wandered through town with J. in the sun - lovely, Greggs & lazed by the cathedral: still super-tired somehow. Met up with the babes, back home & drove out to Exmouth walked along the beach - beautiful views from near the lifeboat station; ice-cream. Back for ad-hoc dinner with Janine at our AirBnB - fun.
- [LibreOffice QA Blog: QA/Dev Report: February 2025](#) (2025/03/10 12:21)
General Activities LibreOffice 25.2.0 was announced on Feb 6. Three weeks later, LibreOffice 25.2.1 was announced on Feb, 27 LibreOffice 24.8.5 was announced on Feb 20 Olivier Hallot (TDF) improved the descriptions of new Calc functions shown in the UI, added a Help button to the Data Provider dialog, added help pages for new Calc functions CHOOSECOLS(), CHOOSEROWS(), VSTACK() and HSTACK(), added a help page for Calc's Data Provider and improved help for Paste Special as well as labels and business cards Tomaž Vajngerl (Collabora) continued working on PDF 2.0 support and refactored graphics and animation handling code in VCL toolkit Miklós Vajna, Rashesh Padia, Darshan Upadhyay, Gökay Şatır, Attila Szűcs, Szymon Kłos (Collabora) worked on LOKit used by Collabora Online. Szymon also improved the user experience of the Currency dropdown by removing the need to click an OK button Andras Timar (Collabora) fixed an issue with importing WEEKNUM() functions from XLSX files, made Excel style cell reference syntax be respected in non-English UIs and made it so in read-only documents one can't invoke the Search and Replace dialog, reset cell attributes or fill down cells Xisco Faulí (TDF) implemented new Calc functions CHOOSECOLS(), VSTACK() and HSTACK(), made UNIQUE() case-insensitive like its counterpart in Excel, added a couple of dozen automated tests, upgraded many dependencies and fixed a crash Michael Stahl (allotropia) fixed rendering of overlapping tracked formatting and deletions in imported DOCX files, fixed losing tracked changes when paragraph has a frame anchored to it, fixed truncation of tables in sections split across pages and improved compatibility with MS Word in the case of hidden text Mike Kaganski (Collabora) fixed an issue with the Alt+X Unicode conversion command when following a combining character, fixed Calc's INFO() function giving unexpected results with some arguments, made BASIC's Shell() function more robust and implemented a compatibility option for MS Word's "Underline Trailing Spaces". He also did many code cleanups and optimisations Caolán McNamara (Collabora) fixed sheet identifiers going out of sync sometimes with XLSX export, fixed crashes, fixed many issues found by static analysers and did code cleanups and optimisations Stephan Bergmann (allotropia) worked on the WASM build. He also adapted the code to compiler changes and did code cleanups Noel Grandin (Collabora) made it faster to load and display XLS and XLSX files with lots of conditional formatting. He also did many code cleanups and optimisations, especially in the area of graphics handling Justin Luth (Collabora) fixed an Excel compatibility issue with frozen cell zones, fixed unwanted empty paragraphs appearing in headings in DOCX files, fixed tabstops missing from paragraph styles in DOC import and made DOCX metadata compatible with MS Word (Word deviates from the OOXML specification in this area) Michael Weghorn (TDF) continued cleaning up and reorganising accessibility-related code, made Sidebar, Quick Find and editable comboboxes more accessible, fixed a visual glitch when resizing the window in certain cases affecting Qt-based UIs, fixed an issue with pasting non-latin text from Firefox or Thunderbird affecting Qt-based UIs and fixed crashes and build issues on Android. He also worked on using native widgets in Qt

Uls Balázs Varga (allotropia) optimised the speed of Calc's SubTotal functions, fixed a data loss issue affecting text box controls and fixed locking down of "Use hardware acceleration" options not always working Patrick Luby enabled native full screen mode on macOS, helped Sahil in polishing the UI theming rework and fixed macOS and iOS build issues Oliver Specht (CIB) implemented support for read protection in RTF files, fixed multi-line Docvariable fields being broken in imported DOCX files, made it so border distance in styles gets applied to tables in imported RTF files, fixed renaming list styles causing disconnection from the paragraph style, implemented support for repeated table headers in RTF import, fixed character properties getting wrongly extended in RTF import and fixed unwanted clearing of object state after visiting Impress/Draw options László Németh improved the inline headings and smart justify features and worked on DOCX support for hyphenate-keep feature Ilmari Lauhakangas (TDF) reduced the size of Karasa Jaga SVG icon theme by simplifying graphics Christian Lohmaier (TDF) improved the Windows build setup Jonathan Clark (TDF) implemented exact and at-least line spacing for CJK text grid in Writer, fixed DOC/DOCX compatibility issues related to CJK grid and fixed a kashida justification issue in Writer Sahil Gautam (allotropia) continued polishing the Libreoffice Theme rework Andreas Heinisch made it possible to use the Delete key to remove bitmaps in the Area tab of various dialogs Chris Sherlock did code cleanups, documentation and refactoring in VCL toolkit Armin Le Grand (Collabora & allotropia) continued polishing item handling and Cairo Linux rendering reworks Björn Michaelsen did refactoring in Writer code Tibor Nagy (allotropia) made it possible to insert AutoText and enable spell checking in sections that are editable in read-only documents and made the PDF export of table caption elements conform to accessibility standards Jean-Pierre Ledure worked on the ScriptForge library Áron Budea (Collabora) fixed an issue causing certain presentations with embedded media to fail to open with PowerPoint after saving to PPTX via command line Adam Seskunas converted a database test from Java to C++ Rafael Lima did cleanups in item handling Jaime Pujantell (Collabora) fixed unwanted anchoring of a shape to the page when inserted to a DOCX file and made it so the page number is added when saving/loading PDF pages as images Alexandre Sena Coelho fixed ambiguous sorting in SQL Query Wizard by including table names in ORDER BY clause Robin Candau and René Engelhard fixed PDF import breakage due to changes in poppler version 25.02.0 Mohamed Ali implemented right-to-left brochure printing in Draw / Impress Manish Bera improved thread handling in WebDav code Samuel Mehrbrodt (allotropia) made it so turning off a colour AutoFilter drops the filter settings Thorsten Behrens (allotropia) made mouse-as-pen status changes be reflected in real time into live Impress slideshows and made it so cli and Firebird intl DLLs are code signed Kohei Yoshida upgraded mdds and liborcus libraries Jim Raykowski fixed Writer bookmarks list getting corrupted after sorting and deleting actions, made it so reminder objects will be skipped when copying and pasting text in Writer, fixed inability to deal with font listboxes after increasing font size on the system and made Navigator respect change tracking visibility in the case of deleted headings Gülşah Köse (Collabora) fixed an issue causing XLS files with command buttons roundtripped as XLSX to not open in Excel Julien Nabet synchronised Star Database Connectivity (SDBC) API with JDBC 4.3 Bayram Çiçek (Collabora) fixed a pivot table issue when exporting to XLSX Mohit Marathe (allotropia) fixed unwanted table border lines shown in a certain PPTX file Pranam Lashkari (Collabora) fixed OOXML import of formulas containing delimiters Michael Meeks (Collabora) improved thread handling code Kudos to Ilmari Lauhakangas for helping to elaborate this list. Reported Bugs 536 bugs, 69 of which are enhancements, have been reported by 330 people. Top 10 Reporters Eyal Rozenberg (28) Justin L (26) Aron Budea (19) Gabor Kelemen (allotropia) (14) Jeff Fortin Tam (12) Mike Kaganski (8) Radish (8) wodsfort (7) Telesto (6) Buovjaga (6) Triaged Bugs 465 bugs have been triaged by 60 people. Top 10 Triagers m_a_riosv (76) Buovjaga (70) V Stuart Foote (44) Xisco Faulí (35) Heiko Tietze (27) raal (22) Mike Kaganski (19) Justin L (18) Aron Budea (15) Regina Henschel (12) Resolution of resolved bugs 354 bugs have been set to RESOLVED. Check the following sections for more information about bugs resolved as FIXED, WORKSFORME and DUPLICATE. Fixed Bugs 144 bugs have been fixed

by 34 people. Top 10 Fixers Oliver Specht (13) Michael Weghorn (8) Olivier Hallot (7) Justin Luth (7) Armin Le Grand (Collabora) (6) Mike Kaganski (6) Balazs Varga (5) Tibor Nagy (5) Xisco Fauli (5) Noel Grandin (4) List of critical bugs fixed tdf#164949 Crash on Clone Formatting when selecting more than one table cell (Thanks to Oliver Specht) tdf#165099 CRASH: selecting an animation after slideshow (Thanks to Mike Kaganski) List of high severity bugs fixed tdf#120397 FILESAVE doesn't save all the text in text box control (Thanks to Balazs Varga) tdf#153131 Copy causes Calc to Freeze on Windows 11 with Speech Recognition (comment 58) (workaround: comment 73) (Thanks to Michael Weghorn) tdf#160252 Editing a conditional format from the Manage dialog changes the range / creates a new one (Thanks to Armin Le Grand (Collabora)) tdf#165295 REPORTBUILDER - Report builder freezes when creating a report (Thanks to Caolán McNamara) List of crashes fixed tdf#164072 LibreOffice crashes when deleting all comments (debug) (Thanks to Michael Weghorn) tdf#164620 CRASH: selecting all and deleting (Thanks to Noel Grandin) tdf#164949 Crash on Clone Formatting when selecting more than one table cell (Thanks to Oliver Specht) tdf#165099 CRASH: selecting an animation after slideshow (Thanks to Mike Kaganski) tdf#165420 Shell(Empty) crashes (Thanks to Mike Kaganski) List of performance issues fixed tdf#134864 Calc takes a time for XLSX file opening (so many condition formatting rules in the file) (Thanks to Noel Grandin) List of old bugs (more than 4 years old) fixed tdf#118465 RTF import does not repeat header / repeat heading / repeat as header row for table (Thanks to Oliver Specht) tdf#120397 FILESAVE doesn't save all the text in text box control (Thanks to Balazs Varga) tdf#126824 Most strings in Calc Data Provider shown in English (Thanks to Olivier Hallot) tdf#128186 Create Native macOS Full Screen Mode (Thanks to Patrick Luby) tdf#133146 Allow [Delete] shortcut to open "Delete Bitmap" dialog in Paragraph Style > Area (Thanks to Andreas Heinisch) tdf#134864 Calc takes a time for XLSX file opening (so many condition formatting rules in the file) (Thanks to Noel Grandin) tdf#90293 Unify drawing object rotation access by single click (Thanks to Oliver Specht) WORKSFORME bugs 51 bugs have been retested by 25 people. Top 10 testers V Stuart Foote (10) Buovjaga (6) m_a_riosv (4) Dieter (3) Regina Henschel (3) BogdanB (3) Aron Budea (2) Olivier Hallot (2) Samuel Mehrbrodt (allotropia) (2) Michael Weghorn (2) DUPLICATED bugs 96 bugs have been duplicated by 26 people. Top 10 testers Buovjaga (17) Xisco Faulí (16) V Stuart Foote (14) m_a_riosv (13) Michael Weghorn (6) Gabor Kelemen (allotropia) (3) Saburo (3) Regina Henschel (3) Timur (2) Julien Nabet (2) Verified bug fixes 22 bugs have been verified by 12 people. Top 10 Verifiers Buovjaga (4) Stéphane Guillou (stragu) (4) m_a_riosv (3) Xisco Faulí (2) Gerald Pfeifer (2) Telesto (1) BogdanB (1) Julien Nabet (1) Heiko Tietze (1) Ming Hua (1) Categorized Bugs 270 bugs have been categorized with a metabug by 27 people. Top 10 Categorizers V Stuart Foote (52) Eyal Rozenberg (45) Aron Budea (38) BogdanB (27) Roman Kuznetsov (25) Jeff Fortin Tam (23) Heiko Tietze (10) Stéphane Guillou (stragu) (7) Buovjaga (4) jan d (4) Regression Bugs 37 bugs have been set as regressions by 10 people. Top 10 m_a_riosv (9) raal (8) Xisco Faulí (7) Buovjaga (4) V Stuart Foote (3 ...

- [allotropia: ZetaJS: Combining Writer & Calc](#) (2025/03/06 10:30)

We've added a great new Vue.js-3 ZetaJS demo (source)! It showcases word processing and spreadsheets inside a single web app. Calc is being used as a data source for an HTML app, filling letter templates in Writer. You can even upload custom data spreadsheets or document templates! And have you seen the nice Writer toolbar, all done with Vue.js? We've also updated the existing demos, showcasing Chrome PWA support with the Ping Monitor demo - just click the little install button at the top-right of the address bar, to get the Ping Monitor installed on your desktop! Talks Meanwhile, our team was giving some great talks about our work for ZetaOffice and LibreOffice. Why not check out the recordings during your lunch break? ZetaJS & ZetaOffice Moritz Duge - OSXP Paris: LibreOffice as Web Component - moving customized office workflows into the browser Moritz Duge - 38C3 CCC congress: LibreOffice WASM & JS - Blending a C++ FOSS into a web app

Stephan Bergmann: LOWA, In Need Of a VCL Plug (description) Thorsten Behrens - Distributed real-time collaboration for Writer - a first prototype (description) FOSDEM LibreOffice DevRoom talks Balazs Varga - Introducing Glow Effect for texts in shapes (description) Gabor Kelemen - Testing the QA instructions (description) Sarper Akdemir - LibreOffice's Python API: Working around limitations of the Pythonic approach (description) Gabor Kelemen - Exploring the deprecated parts of LibreOffice API (description) News clippings Look, we made some headlines! TheRegister was following up some earlier coverage about the WebAssembly port, after Thorsten gave Liam a demo during FOSDEM. Read up the full article here. Next up In case you're around, meet us in two weeks at the FOSSAsia Summit in Bangkok, where Sarper Akdemir will give an update over our work. Dates are March 13-15. If you're based in Europe, you might instead enjoy Thorsten's talk at the Chemnitz Linux Days (Germany) from March 22-23. Looking forward to meet you there! Feedback appreciated! Please subscribe to our Newsletter or on Mastodon and let us know how you liked ZetaJS and the demos! If you're playing with the code leave a star at the ZetaJS repo or if you hit any issues please file a report on GitHub. Or just leave a comment and let us know directly - thanks for reading!

- [LibreOffice Design Blog: New Templates For You - Your Feedback Matters!](#) (2025/03/03 13:33)

By Ndidì Folasade Ogboi For the past two months, I've been working on adding more templates to LibreOffice Writer as part of my Outreachy project. My goal has been to create functional templates that users need the most. I created these templates based on what you told us in our survey and your response was incredible!...

- [Marius Popa Adrian: Firebird 5.0.2 minor release is available](#) (2025/02/26 10:57)

Firebird Project is happy to announce general availability of Firebird 5.0.2 — the latest minor release in the Firebird 5.0 series. This minor release offers bug fixes as well as a few improvements, please refer to the Release Notes for the full list of changes. Binary kits for Windows, Linux, MacOS and Android platforms are immediately available for download.

- [LibreOffice Dev Blog: Understanding the existing code to provide better patches](#) (2025/02/17 10:17)

LibreOffice inherits a gigantic code base from its ancestors, StarOffice and OpenOffice. Here I discuss some notes for the newcomers on how to better understand the existing LibreOffice code, and improve the patches. Studying the Existing Code As said, LibreOffice is a huge code base, containing ~10 million lines of mostly C++ code. There are different assumptions, conventions and coding styles across ~200 modules that LibreOffice has. Therefore, it is important to first, study the existing code, through reading and debugging LibreOffice source code, to understand the things that it does, and the way you can implement your ideas, including bug fixes and adding new features. And although implementing some ideas seem to be straightforward at first sight, it is meaningful to study the details. Quality Assurance Point of View First of all, you should understand the thing that you want to implement. No matter if it is a bug, a new feature, or just an EasyHack, you should understand what is requested, what works and what does not work. This requires careful reading of the Bugzilla pages. User Point of View Then, you should try to run LibreOffice to understand the exact place in the application where you want to change. LibreOffice user interface has thousands of dialog boxes, so you need to make sure that you understand the thing that you want to do. Developer Point of View And at last, you get into implementing something in the code. Here are some questions that you can ask yourself about the details, when reading the existing code: Why this statement is here, in the first place? (detail-oriented view) You can use git blame to see the last author of a specific line You can use git log to study the details by knowing the commit hash What can this part of code actually do? Can I see its effect? git log Or, you may be interested in the code behavior in the big picture: What does the code do as a whole? (holistic view) There are many other statements, functions and other constructs in the code. What do they do? What is the overall goal of the code? Can I test that in action? You can do some small changes, before even getting

into implementing your idea: What happens if I remove it? (small changes) Does the removal prevent the code from working? Is it incomplete, or does it actually do something useful, which will be absent if I remove it? Then, you can work on the actual implementation. Ask yourself: How can I implement the idea in its simplest form? (straightforward change) Does it have side effects? How can I make sure every thing else works as before? How can I write a test for it? After understanding some of the basic details about the way things work, you may go into improving your implementation. How can I make it better? (sophisticated change) Can I make the code more robust where it is brittle? Can I complete the code where it is incomplete? Final Notes These were the questions to give you some ideas of some of the underlying complexities in the code. You can start from small changes to become familiar with these complexities, and then grow to do more complex stuff in the code. We have various different EasyHacks in LibreOffice, with different difficulty levels. If you are interested in coding, you can always find something that fits you, and grow gradually. You can read more in these links: TDF Wiki: EasyHacks Blog posts on some (solved) difficultyInteresting EasyHacks

- [LibreOffice QA Blog: QA/Dev Report: January 2025](#) (2025/02/11 16:18)

General Activities Olivier Hallot (TDF) added help pages for new Calc functions TOROW(), TOCOL(), WRAPROWS(), WRAPCOLS(), EXPAND(), TAKE() and DROP(), added dark mode support to the help interface, improved help for PDF/UA, did cleanups in the Xapian-based search in online help, added help for tables styles in Writer and improved help related to printing Dione Maddern added a help page for Cell Appearance Sidebar deck Stanislav Horáček did some cleanups in help Gábor Kelemen (allotropia) added a detailed list of allowed PDF password characters into help and improved the developer tools for finding unneeded includes and UI strings that might need to be translatable Tomaž Vajngerl (Collabora) continued working on PDF 2.0 support and document themes and fixed an Excel compatibility issue with empty values of defined names Miklós Vajna, Andras Timar, Henry Castro, Gökay Şatır, Attila Szűcs, Szymon Kłos and Pranam Lashkari (Collabora) worked on LOKit used by Collabora Online Xisco Faulí (TDF) implemented new Calc functions, TOCOL, TOROW, WRAPCOLS, WRAPROWS, TAKE, DROP, EXPAND and CHOOSEROWS, added support for setuptools and pip in Python scripting, upgraded many dependencies, added some unit tests and did many code stability improvements Michael Stahl (allotropia) continued improving the correctness of HTML import regarding formatting and fixed issues with table splitting in Writer's layout Mike Kaganski (Collabora) fixed an issue with opening newly-created database forms, fixed Basic isNumeric() function giving incorrect results, fixed an installation issue affecting Active Directory setups on Windows, fixed issues with allowed characters in file name when exporting as PDF, fixed wrong number of results being reported when going over 1000 while executing Find All in Calc, fixed inability to pass a Date object to an UNO API method, fixed an issue with handling of Variant types in Basic, made handling of conditional formatting with colour conditions more robust when moving columns, made intercepting .uno:Open command work again, fixed a crash related to regular expressions in Basic and made SQL queries handle negative values Caolán McNamara (Collabora) fixed crashes, fixed many issues found by static analysers and did code cleanups and optimisations Stephan Bergmann (allotropia) worked on the WASM build. He also adapted the code to compiler changes and did code cleanups Noel Grandin (Collabora) improved the speed of inserting rotated images to Writer. He also did many code cleanups and optimisations Justin Luth (Collabora) fixed DOCX import issues with frames before tables getting anchored to a table cell instead of an empty paragraph and missing header properties in page styles Michael Weghorn (TDF) continued cleaning up and reorganising accessibility-related code, did refactoring in Linux printer code and fixed some crashes. He also worked on using native widgets in Qt UIs Balázs Varga (allotropia) fixed import of cropped vector graphic objects in PPTX files, improved warnings related to allowed characters in the PDF password input dialog, made it possible to show or hide the text in some password dialogs (more to be included), fixed broken cropped SVG files in PPTX import and made it so the size values in Position and Size and Crop tabs in Image Properties dialog are synchronised Patrick Luby fixed

artifacts showing in animated GIFs with Skia UI rendering on macOS, added Quick Look plugins for .od* files on macOS and made it so the Start Center menubar is shown in the default menubar on macOS Oliver Specht (CIB) made it so the table context menu in Draw/Impress includes hyperlink actions, made scrolling while selecting less hasty, made it so Ctrl+scrollwheel changes the slides per row setting when in View - Slide Sorter in Impress, made the status of numbered and bulleted list toggle state visible in toolbars and menus in Impress/Draw, made it possible to open the Edit Field dialog in read-only Writer documents and fixed losing chart number formatting when copying and pasting the chart Heiko Tietze (TDF) added a confirmation dialog when deleting all comments in Writer László Németh fixed loss of images anchored to page in subdocuments of Writer master documents and made bookmark boundary mark labels look cleaner in Writer Ilmari Lauhakangas (TDF) improved the layout of help and did cleanups in its CSS styles Christian Lohmaier (TDF) improved the Windows build setup Eike Rathke (Red Hat) added support for English (Guyana) Jonathan Clark (TDF) added support for Mongolian while enabling vertical text options for it, made the script type assignment algorithm in the context of mixed Western and Asian text more robust, implemented vertical CJK printing for all fonts on Windows and fixed borders of merged cells in Calc vanishing when changing sheet direction to right-to-left Sahil Gautam (allotropia) continued polishing the Libreoffice Theme rework Andreas Heinisch added support for importing inserted text tag "ins" from HTML, made it so the Edit... button in Writer's Index dialog is disabled, if no concordance file has been selected and added first and secondary keys to the tooltips of index fields Chris Sherlock did code cleanups in VCL Laurent Balland did fixes in Lights, Focus, Forestbird, Yellow Idea and Vivid Impress templates Armin Le Grand (Collabora) did refactoring in item handling Björn Michaelsen did refactoring in Writer code David Gilbert added a readme for PDF import code Tibor Nagy (allotropia) fixed a PDF export accessibility issue and made the Formatting toolbar visible in sections that are marked as editable in read-only documents Jean-Pierre Leduc worked on the ScriptForge library Ahmed Hamed added a category to store favorite functions in Calc's Function Wizard and Functions Sidebar deck Áron Budea (Collabora) fixed unwanted cell formatting reset upon changing language on a selection in Calc Adam Seskunas ported a Java test to C++ Rafael Lima made solver's Sensitivity Report prettier and did cleanups in item handling Jaume Pujantell (Collabora) fixed unneeded duplication of slide master when exporting to PPTX Skyler Grey (Collabora) made the iOS app use desktop clipboard code Kudos to Ilmari Lauhakangas for helping to elaborate this list. Reported Bugs 418 bugs, 50 of which are enhancements, have been reported by 246 people. Top 10 Reporters Justin L (20) Gabor Kelemen (allotropia) (18) Aertx (12) Eyal Rozenberg (11) Telesto (10) Aron Budea (9) Michael Otto (8) Jeff Fortin Tam (8) Mike Kaganski (7) Mihai Vasiliu (7) Triaged Bugs 362 bugs have been triaged by 67 people. Top 10 Triagers BogdanB (45) Buovjaga (37) raal (37) Heiko Tietze (28) m_a_riosv (26) V Stuart Foote (21) Mike Kaganski (16) Aron Budea (11) Roman Kuznetsov (11) Xisco Faulí (10) Resolution of resolved bugs 347 bugs have been set to RESOLVED. Check the following sections for more information about bugs resolved as FIXED, WORKSFORME and DUPLICATE. Fixed Bugs 157 bugs have been fixed by 29 people. Top 10 Fixers Mike Kaganski (17) Xisco Fauli (12) Olivier Hallot (11) Jonathan Clark (8) Oliver Specht (7) Patrick Luby (7) Michael Weghorn (7) Balazs Varga (6) Noel Grandin (5) Michael Stahl (5) List of critical bugs fixed tdf#164185 View -> Boundaries is turned off by default making it impossible to move image + caption frame (Thanks to Ilmari Lauhakangas) List of high severity bugs fixed tdf#164127 [Crash] Crash on returning to dialog window after switching to document while editing Basic-IDE dialog controls (Thanks to Michael Weghorn) tdf#164640 List bullets formatting changed (Thanks to Noel Grandin) tdf#164855 Crash while centering table contents (Thanks to Balazs Varga) tdf#35361 [feature request: macOS] Support Apple Quick Look plugin (Thanks to Patrick Luby) List of crashes fixed tdf#156348 Crash if change in formatting in Writer by converting text to table with field variable (Thanks to Michael Stahl) tdf#159377 CRASH at undo at after pasting table in footer (swlo!SwFormatFootnote::SetNumStr+0x3e26:) (Thanks to Michael Stahl) tdf#160770 Crashes on second access of regex matches

without VBA support option (Thanks to Mike Kaganski) tdf#163335 Linux (qt6): crash whenever selecting text using cursor or keyboard going from right to left (Thanks to Michael Weghorn) tdf#164127 [Crash] Crash on returning to dialog window after switching to document while editing Basic-IDE dialog controls (Thanks to Michael Weghorn) tdf#164130 LibreOffice Calc crashes when doing a lookup in a sheet with a space in its name (Thanks to Henry Castro) tdf#164179 Crash when switching the Short Name in Bibliography Entry dialog (Thanks to Vojtěch Doležal) tdf#164620 CRASH: selecting all and deleting (Thanks to Noel Grandin) tdf#164621 CRASH: pasting content (Thanks to Oliver Specht) tdf#164783 Libreoffice crashes when clicking on grid form column header or in empty space below rows, gtk3+a11y (Thanks to Michael Weghorn) tdf#164855 Crash while centering table contents (Thanks to Balazs Varga) tdf#164899 [CRASH] LO crashes upon opening file with macro when the Tabbed interface is used (Thanks to Michael Weghorn) List of performance issues fixed tdf#137848 Inserted image slow (15 seconds, expected 3) (Thanks to Noel Grandin) tdf#164853 unusual copy seen in find_if (Thanks to Caolán McNamara) List of old bugs (more than 4 years old) fixed tdf#105083 Impress: The numbered/bulleted list toggle button and menu items aren't highlighted when a numbered/bullet list is active (Thanks to Oliver Specht) tdf#117946 Impress: Slide Sorter: Ctrl+mouse wheel should change slides per row (Thanks to Oliver Specht) tdf#121119 Loss of image anchored to page in a writer master document (Thanks to László Németh) tdf#130672 base sql query parameter with negative value fails (Thanks to Mike Kaganski) tdf#132770 Underline text using INS tag from HTML document do not appear (Thanks to Andreas Heinisch) tdf#137848 Inserted image slow (15 seconds, expected 3) (Thanks to Noel Grandin) tdf#34837 Merged Cell's borders vanishes when changing sheet direction to (Right-To-Left) (Thanks to Jonathan Clark) tdf#35361 [feature request: macOS] Support Apple Quick Look plugin (Thanks to Patrick Luby) tdf#37507 Vertical scrolling with mouse cursor is too fast to control (Thanks to Oliver Specht) tdf#41775 Don't remove all menus when no windows are open - keep Tools and Help (Thanks to Xisco Fauli) tdf#50743 FORMATTING: Highlighting scrolls automatically (Thanks to Oliver Specht) tdf#66791 FORMATTING: Incorrect application of "Asian text font" for quotation marks when the paragraph contains a mixture of western and asian characters (Thanks to Jonathan Clark) tdf#94193 Installer forces AD domain users in Administrators group to run as Administrator, otherwise custom actions are disallowed during execution stage and not completed (Thanks to Mike Kaganski) WORKSFORME bugs 54 bugs have been retested by 29 people. Top 10 testers BogdanB (11) raal (5) V Stuart Foote (4) Andreas Heinisch (3) Timur (3) Buovjaga (3) Regina Henschel (2) Aron Budea (2) m_a_riosv (2) Eduardo (1) DUPLICATED bugs 66 bugs have been duplicated by 26 people. Top 10 testers Buovjaga (8) Aron Budea (7) BogdanB (6) V Stuart Foote (6) Gabor Kelemen (allotropia) (5) m_a_riosv (5) Jonathan Clark (3) Roman Kuznetsov (3) Timur (2) Justin L (2) Verified bug fixes 22 bugs have been verified by 13 people. Top 10 Verifiers Buovjaga (4) Gerald Pfeifer (3) BogdanB (2) Aron Budea (2) Piotr Osada (2) Xisco Faulí (2) Michael Weghorn (2) Timur (1) Alex Thurgood (1) Regina Henschel (1) Categorized Bugs 354 bugs have been categorized with a metabug by 28 people. Top 10 Categorizers Roman Kuznetsov (128)...

- [Jean Hollis Weber: Calc, Impress, Draw 24.2 guides published](#) (2025/02/03 06:19)

Print editions of several LibreOffice 24.2 user guides were published in 2024. You can buy them from Lulu.com. Free PDFs, as always, are available from the LibreOffice website. Calc (May 2024) Impress (July 2024) Draw (August 2024) Writer (March 2024)

- [LibreOffice Dev Blog: Custom message boxes using VCL Weld](#) (2025/01/30 15:01)

When you want to interact with users, sometimes simple dialog boxes are sufficient: a simple yes or no, or some info box. But in other cases, you may need more complex message boxes. Here I discuss how to use VCL Weld to create a custom one. Simple Message Box You can create a simple message box, using predefined templates like Info box using a code snippet like this: `std::unique_ptr<weld::MessageDialog>`

xInfoBox(Application::CreateMessageDialog(pParent, VclMessageType::Question, VclButtonsType::YesNo, u"Are you sure?"_ustr));
 xInfoBox->run(); And, this is the result, which is very simple, without any title bar: Yes / No message box There are other predefined types, which can be used in different scenarios: enum class VclMessageType { Info, Warning, Question, Error, Other }; But, if you want custom message boxes, you should be using weld mechanism, with its CreateBuilder function. Custom Message Boxes Below is the code from the source code sfx2/source/doc/QuerySaveDocument.cxx, which is inside sfx2 (framework) module. This dialog box is accessible across different modules, including Writer, Calc and Draw/Impress. Let's look into the code: short ExecuteQuerySaveDocument(weld::Widget* _pParent, std::u16string_view _rTitle) { ... std::unique_ptr<weld::Builder> xBuilder(Application::CreateBuilder(_pParent, u"sfx/ui/querysavedialog.ui"_ustr)); std::unique_ptr<weld::MessageDialog> xQBox(xBuilder->weld_message_dialog(u"QuerySaveDialog"_ustr)); xQBox->set_primary_text(xQBox->get_primary_text().replaceFirst("\$ (DOC)", _rTitle)); return xQBox->run(); } The code is using a UI file, named sfx/ui/querysavedialog.ui to create a message dialog, and then change the title of it. QuerySaveDialog If you look into the include file, include/vcl/weld.hxx inside Builder class, you may see functions like weld_... that are suitable to find various different UI elements from the UI, by mentioning the element ID. For example, to find a label with the ID equal to lable_id, you do this: std::unique_ptr<weld::Label> m_pTextLabel label = m_xBuilder->weld_label(u"label_id"_ustr) Result This is the result, when you try to close an unsaved document. QuerySaveDialog running Alternative Ways This is not the only way you can create nice dialog boxes using VCL weld mechanism. There are some predefined message boxes that look nice which use weld mechanism, and are available for use via relevant C++ classes. An interesting one here, is the QueryDialog, which is created by a factory method design pattern. It uses a predefined dialog, using cui/uiconfig/ui/querydialog.ui as the UI file, and it contains a nice stock image! You can test it easily, by modifying a LibreOffice example, minweld. IMPL_LINK_NOARG(TipOfTheDayDialog, OnNextClick, weld::Button&, void) { VclAbstractDialogFactory* pFact = VclAbstractDialogFactory::Create(); auto pDlg = pFact->CreateQueryDialog(getDialog(), u"Tips"_ustr, u"Tip of the day"_ustr, u"Are you sure you want to see the next tip of the day?"_ustr, false); sal_Int32 nResult = pDlg->Execute(); pDlg->disposeOnce(); if(nResult == RET_YES) { ++m_nCounter; m_pTextLabel->set_label(u"Here you will see tip of the day #"_ustr + OUString::number(m_nCounter) + "."); } } Assuming that you have a working build of LibreOffice, you can simply run the minweld workbench by invoking: ./bin/run minweld The result looks much more interesting: QueryDialog Final Words The possibilities are endless! It only depends on your ideas and understanding of the user's needs and requirements. It would be good if you look into what design team does to understand the design process: LibreOffice design team blog TDF Wiki: Design and User Experience team

- [Marius Popa Adrian: Firebird Docker Images Now Under the Firebird Organization](#) (2025/01/29 19:11)

We are pleased to announce the successful migration of Firebird Docker images to their new home:<https://github.com/FirebirdSQL/firebird-docker>The images are now published on Docker Hub at<https://hub.docker.com/r/firebirdsql/firebird>Thanks to Adriano dos Santos Fernandes for his invaluable contributions and improvements throughout this process.

- [LibreOffice QA Blog: LibreOffice 25.2 RC2 is available for testing](#) (2025/01/17 12:52)

LibreOffice 25.2 will be released as final at the beginning of February, 2025 (Check the Release Plan) being LibreOffice 25.2 Release Candidate 2 (RC2) the forth and last pre-release since the development of version 25.2 started in mid Juny, 2024. Since the previous release, LibreOffice 25.2 RC1, 104 commits have been submitted to the code repository and 55 issues got fixed. Check the release notes to find the new features included in this version of LibreOffice. LibreOffice 25.2 RC2 can be downloaded for Linux, macOS and Windows, and it will replace the standard installation.

In case you find any problem in this pre-release, please report it in Bugzilla (You just need a legit email account in order to create a new account). For help, you can contact the QA Team directly in the QA IRC channel or via Matrix. LibreOffice is a volunteer-driven community project, so please help us to test it we appreciate it! Happy testing!! ...

- [LibreOffice Dev Blog: Outlook for the new year 2025](#) (2025/01/16 14:29)

Happy new year 2025! I wish a great year for you, and the global LibreOffice community. Now that we are now in 2025, I briefly discuss the year 2024 and outlook for 2024 in the development blog. LibreOffice Conference 2024, Luxembourg At The Document Foundation (TDF), our aim is to improve LibreOffice, the leading free/open source office suite that has millions of users around the world. Our work is community-driven, and the software needs your contribution to become better, and work in a way that you like. My goal here, is to help people understand LibreOffice code easier, and eventually participate in LibreOffice core development to make LibreOffice better for everyone. In 2024, I wrote 22 posts around LibreOffice development in the dev blog (4 of them are unpublished drafts). Outlook For the New Year Focus of the development blog for 2025 in this blog will be: Introducing new EasyHacks Describing user interface creation with VCL Explaining LibreOffice architecture Explaining Python interaction with LibreOffice I have written about some of these topics in 2024. Therefore, this year I will try to expand the previous writings and provide new articles about them. For example, creating user interfaces using VCL with the help of glade interface designer will be one of important things to discuss. You can give feedback by writing a comment here, or sending me an email to hossein AT libreoffice DOT org. We provide mentoring support to those who want to start LibreOffice development. You are welcome to contact me if you need help to build LibreOffice and do some EasyHacks via the above email address. Also, you can always refer to our Getting Involved Wiki page: TDF Wiki: Getting Involved in LibreOffice Development Let's hope a great year for LibreOffice (and the world) in 2025.

- [Marius Popa Adrian: A sad day for the Firebird Project](#) (2025/01/13 13:15)

Helen Borrie, a key figure in the Firebird relational database project and a longtime contributor at IBPhoenix, passed away on January 2, 2025. Her contributions were essential to Firebird's creation and its development over the past 25 years. Read the rest of the official announcement

- [Marius Popa Adrian: Jaybird 6.0.0 released](#) (2025/01/13 13:06)

We're happy to announce the first release of Jaybird 6, Jaybird 6.0.0.

- [LibreOffice Design Blog: Results from a survey about Writer templates](#) (2025/01/13 10:06)

By Ndidi Folasade Ogboi LibreOffice Writer has long been a trusted tool for users worldwide, offering an open-source solution for documents. But what happens when we take a step back and look at the user experience? How do templates fit into the workflows of users, what makes a great template and where do users want LibreOffice writer to improve?...

- [LibreOffice QA Blog: QA/Dev Report: December 2024](#) (2025/01/09 13:08)

General Activities LibreOffice 24.8.4 was announced on December 19 Olivier Hallot (TDF) improved the warning in Help when JavaScript is not active and did many cleanups in help pages Dione Maddern created a help page for Alignment Sidebar deck Alain Romedenne improved and updated help for ScriptForge libraries Bogdan Buzea improved some UI labels, improved help for superordinate object settings and cached spreadsheet formulas and did many code cleanups Tomaž Vajngerl (Collabora) continued working on PDF 2.0 and PDF/A-4 support Miklós Vajna, Rashesh Padia, Attila Szűcs, Bayram Çiçek, Szymon Kłós, Marco Cecchetti, Pranam Lashkari, Hubert Figuière (Collabora) worked on LOKit used by Collabora Online Xisco Faulí (TDF) worked on crash report analysis tools, upgraded many dependencies and did many code stability improvements Michael Stahl (allotropia) made style name handling more robust, improved the handling of hidden frames after recent changes

and improved the correctness of HTML import regarding formatting Mike Kaganski (Collabora) added an application-wide Viewer mode where all files are opened in read-only state while all editing tools are disabled, dropped all code specific to Windows 7, 8 and 8.1 while also making use of new possibilities such as handling long Windows paths with wildcards, made the Unix document mailer script future-proof in case the attach parameter is disallowed in mailto URLs, fixed an issue with calculating minimum heights for menus, fixed an issue with the number format being reported incorrectly in Writer tables, fixed an issue with multi-selection in Calc showing an incorrect cell format and preventing change of format, fixed inability to edit doubles in Basic IDE's Watch window and fixed an issue preventing the use of points for custom image height/width in the PNG export dialog Caolán McNamara (Collabora) improved dark mode support, fixed crashes and fixed many issues found by static analysers and fuzzers and did code cleanups Stephan Bergmann (allotropia) fixed an issue with emailing multiline messages on Unix and worked on the MAR updater and WASM build. He also adapted the code to compiler changes and did code cleanups Noel Grandin (Collabora) improved saving time of XLSX files with lots of conditional formatting, improved the speed of processing styles when opening DOCX files and worked alongside Michael Stahl in making style name handling more robust. He also did many code cleanups and optimisations Justin Luth (Collabora) fixed a DOCX compatibilityMode import and export issue, fixed an issue with imported area fill images not being saved with their associated documents, fixed a Calc comment copying crash and fixed an issue with tables of contents in DOCX files misbehaving when the printer list has been disabled Michael Weghorn (TDF) did a big reorganisation in accessibility-related code to make it easier to work with, continued working with Cambalache developer (UI editing app) and did various accessibility fixes. He also worked on using native widgets in Qt UIs Balázs Varga (allotropia) improved and expanded Writer's accessibility warnings, fixed saving "Fit height to text" property of drawing objects to PPTX, fixed laying out of text in SmartArt objects found in PPTX files and implemented support for soft edge and glow effects in text frame objects in PPTX files Patrick Luby made resizing windows on macOS appear smoother, implemented jumping the view to the proportional location in the document when Option-clicking the scrollbar on macOS (instead of just advancing a single screen/page), implemented support for native macOS full screen mode, fixed an issue causing a long delay in opening the Print dialog on macOS when objects with transparency were present and fixed macOS printing issues related to page settings in Calc and brochures Jim Raykowski did a big rework of macro organiser dialogs, reducing them from five to one Oliver Specht (CIB) improved support for VML textboxes in DOCX files and fixed an issue with paragraph spacing of bullets in Impress Heiko Tietze (TDF) added visual feedback into the status bar for when AutoCalculate is active in Calc László Németh continued polishing support for inline headings in Writer documents Ilmari Lauhakangas (TDF) did code cleanups after the decision to remove support for Windows 7, 8 and 8.1 from version 25.8 Christian Lohmaier (TDF) improved the Windows build setup Eike Rathke (Red Hat) improved date input detection in Calc and helped finish ODF 1.4 support for EASTERSUNDAY function Jonathan Clark (TDF) improved the detection of Asian scripts in text runs adjacent to weak punctuation characters or explicit direction marks Sahil Gautam (allotropia) continued polishing the Libreoffice Theme GSoC project Andreas Heinisch implemented support for pasting HTML strikethrough formatting, made it so the choice of "Link" when inserting an image is remembered during a session and fixed an issue with dashed lines sometimes becoming solid in imported graphics Chris Sherlock did code cleanups in VCL Vasily Melenchuk (CIB) fixed an issue with unwanted background fill in placeholders in PPTX files Laurent Balland replaced a binary DocBook template with an ODF one Xuan Chen fixed shading issues in custom shapes in PPT files Armin Le Grand (Collabora) worked on a renovation of graphics rendering on Linux with Cairo library Björn Michaelsen did refactoring in Writer code Ariel Darshan implemented support for autorepeating slides in windowed mode in Impress Samuel Adesola made it possible to access Writer's view layout options via the View menu Marc Mondesir fixed touchpad scrolling for Slides and Pages panes in Impress and Draw Andrei Alin fixed a ReadLine API function not always stripping line-ending

characters André Herbst fixed an issue with canceling a cell dragging operation leaving behind a visual glitch David Gilbert implemented support for clipping stroke paths in imported PDFs Tibor Nagy (allotropia) fixed an issue with overflowing text in Impress presenter notes getting cut off from printing, fixed presenter notes not being relayouted when changing paper size in the print dialog and similarly for changing orientation for handouts Mohit Marathe continued polishing the new Comments Sidebar deck Jean-Pierre Ledure worked on the ScriptForge library Kudos to Ilmari Lauhakangas for helping to elaborate this list. Reported Bugs 400 bugs, 60 of which are enhancements, have been reported by 249 people. Top 10 Reporters Eyal Rozenberg (13) Justin L (12) Gabor Kelemen (allotropia) (9) Alex Kemp (9) Roman Kuznetsov (8) gplhust955 (7) Robert Großkopf (7) Anna (7) Óvári (7) Mike Kaganski (6) Triaged Bugs 365 bugs have been triaged by 69 people. Top 10 Triagers Buovjaga (77) BogdanB (62) m_a_riosv (31) V Stuart Foote (20) Mike Kaganski (15) zcrhonek (14) Dieter (13) raal (11) Heiko Tietze (10) Michael Weghorn (away) (8) Resolution of resolved bugs 371 bugs have been set to RESOLVED. Check the following sections for more information about bugs resolved as FIXED, WORKSFORME and DUPLICATE. Fixed Bugs 138 bugs have been fixed by 31 people. Top 10 Fixers Balazs Varga (13) Mike Kaganski (10) Bogdan Buzea (9) Caolán McNamara (6) Patrick Luby (6) Michael Weghorn (6) Jonathan Clark (6) Andreas Heinisch (4) Justin Luth (4) Heiko Tietze (3) List of high severity bugs fixed tdf#155211 Regression: dashed lines become solid when breaking imported SVG / exporting to SVG (Thanks to Andreas Heinisch) tdf#163033 Crash when attempting to save a COPY of a sheet-with-comments from a now-closed spreadsheet (Thanks to Justin Luth) tdf#164093 Crash when clicking on the Sidebar Tab menu button for SB deck selector pop-up menu with AT active on Windows (Thanks to Michael Weghorn) tdf#164417 Autofiltered XLSX with dates cannot be opened in MSO (Thanks to Balazs Varga) List of crashes fixed tdf#163033 Crash when attempting to save a COPY of a sheet-with-comments from a now-closed spreadsheet (Thanks to Justin Luth) tdf#163221 Hovering the mouse over a Basic dialog will make it grow or crash (Dialog Editor) (Thanks to Noel Grandin) tdf#163948 Crash upon startup after enabling Notes Pane (Thanks to Sarper Akdemir) tdf#164075 crashtesting: assert on import of rtf exported from LibreOffice (Thanks to Justin Luth) tdf#164093 Crash when clicking on the Sidebar Tab menu button for SB deck selector pop-up menu with AT active on Windows (Thanks to Michael Weghorn) tdf#164098 Typing tatweel character leads to a crash (Thanks to Jonathan Clark) tdf#164299 Pasting from Calc to Impress in HTML format crashes LO (Thanks to Oliver Specht) tdf#164359 Crash on double-click of level 2 word in Impress (Thanks to Miklos Vajna) List of old bugs (more than 4 years old) fixed tdf#113015 “Online updates” checks the updates of extensions installed too (Thanks to Bogdan Buzea) tdf#124954 HELP for recalculating of formulas on opening of a spreadsheet needs updating (Thanks to Bogdan Buzea) tdf#127937 Provide feedback of AutoCalculation at the status bar (Thanks to Heiko Tietze) tdf#128957 UI Can’t set custom image height/width in PNG Options Window in points (Thanks to Mike Kaganski) tdf#131332 HELP: replace custom date formats with ISO 8601 to stop promoting ambiguous formats (Thanks to Bogdan Buzea) tdf#132111 Initial cell’s number format in Writer table is reported wrong (Thanks to Mike Kaganski) tdf#135320 FILEOPEN PPTX: effect similar to soft edges (“inward soft edges”) not shown (Thanks to Balazs Varga) tdf#135628 SendSimpleMailMessage bodytext with new lines ends up as multiple recipients (Thanks to Stephan Bergmann) tdf#138615 Insure Windows wildcards work properly with long paths (Thanks to Mike Kaganski) tdf#42989 FORMATTING: Selecting Multiple Cells with Different Formats Show as Same Format and Can’t Be Changed as a Group (Thanks to Mike Kaganski) tdf#61358 UI: Remember state of option “Insert image from file -> Linked” (Thanks to Andreas Heinisch) tdf#62845 Option for “Document Viewer Mode” (read-only mode by default) required. (Thanks to Mike Kaganski) tdf#66791 FORMATTING: Incorrect application of “Asian text font” for quotation marks when the paragraph contains a mixture of western and asian characters (Thanks to Jonathan Clark) tdf#79298 FORMATTING: Copy/paste: importing of strikethrough attribute doesn’t work (Thanks to Andreas Heinisch) tdf#85428 Imported PDF displays extra-long lines for shaded area (Thanks to Dr. David Alan Gilbert

) tdf#88226 Excessive text in Presentation Notes is not printed (Thanks to Tibor Nagy) WORKSFORME bugs 44 bugs have been retested by 20 people. Top 10 testers Buovjaga (9) BogdanB (8) V Stuart Foote (4) m_a_riosv (4) Robert Großkopf (3) Telesto (2) John (2) Xisco Faulí (1) Regina Henschel (1) Alan (1) DUPLICATED bugs 63 bugs have been duplicated by 20 people. Top 10 testers Mike Kaganski (11) Buovjaga (11) BogdanB (8) V Stuart Foote (7) m_a_riosv (7) Timur (4) Gabor Kelemen (allotropia) (2) Eyal Rozenberg (2) zcrhonek (2) Roman Kuznetsov (1) Verified bug fixes 28 bugs have been verified by 14 people. Top 10 Verifiers Buovjaga (7) BogdanB (5) Stéphane Guillou (stragu) (4) Lars Jødal (2) raal (1) Gerald Pfeifer (1) Eyal Rozenberg (1) Mihai Vasiliu (1) steve (1) lol (1) Categorized Bugs 1175 bugs have been categorized with a metabug by 27 people. Top 10 Categorizers BogdanB (1024) Roman Kuznetsov (36) Eyal Rozenberg (25) V Stuart Foote (21) Dieter (10) Buovjaga (8) Aron Budea (7) Timur (6) Stéphane Guillou (stragu) (6) Telesto (5) Regression Bugs 58 bugs have been set as regressions by 16...

- [Miklos Vajna: Ignoring the paragraph margin at the top of pages in Writer](#) (2025/01/08 08:53)

Writer has the concept of paragraph margins and page margins, but what happens when you combine the two? It turns out the expectation is that sometimes the top paragraph margin is ignored in this case. We'll see two cases where the behavior of Writer is now improved to better match Word in this regard. This work is primarily for Collabora Online, but the feature is available in desktop Writer as well. Motivation¶ As described in a previous bugreport, there was a first problem where Word ignored the top paragraph margin of a document, but Writer did not. A recent bugreport then pointed out that the first implementation went too far and now a wanted top margin was ignored. This led to a set of conditions which now does a decent emulation of Word's rules in this regard. Results so far¶ Here is the old Writer render result for a document where the top margin should be ignored: Bugdoc: old Writer render And here is the new Writer render result for a document where the top margin is ignored: Bugdoc: new Writer render Finally, the reference render result, showing the ignored top paragraph margin: Bugdoc: reference render As you can see, now the unwanted top paragraph margin is omitted at page top. How is this implemented?¶ If you would like to know a bit more about how this works, continue reading... :-)) As usual, the high-level problem was addressed by a series of small changes: tdf#160952 sw: ignore top margin of para on non-first pages with newer DOCX tdf#164095 sw: fix missing top margin on paragraph after changing page style Want to start using this?¶ You can get a development edition of Collabora Online 24.04 and try it out yourself right now: try the development edition. Collabora intends to continue supporting and contributing to LibreOffice, the code is merged so we expect all of this work will be available in TDF's next release too (25.2).

- [LibreOffice QA Blog: LibreOffice 25.2 RC1 is available for testing](#) (2025/01/03 13:07)

LibreOffice 25.2 will be released as final at the beginning of February, 2025 (Check the Release Plan) being LibreOffice 25.2 Release Candidate 1 (RC1) the third pre-release since the development of version 25.2 started in mid June, 2024. Since the previous release, LibreOffice 25.2 Beta1, 175 commits have been submitted to the code repository and 76 issues got fixed. Check the release notes to find the new features included in this version of LibreOffice. LibreOffice 25.2 RC1 can be downloaded for Linux, macOS and Windows, and it will replace the standard installation. In case you find any problem in this pre-release, please report it in Bugzilla (You just need a legit email account in order to create a new account). For help, you can contact the QA Team directly in the QA IRC channel or via Matrix. LibreOffice is a volunteer-driven community project, so please help us to test it! we appreciate it! Happy testing!! ...

- [LibreOffice Design Blog: LibreOffice Themes will replace the color customization](#) (2024/12/20 12:55)

Since the first implementation of a dark color theme we continuously improved the customization of LibreOffice. In a GSoC projects this year,

Sahil Gautam made it possible to not only change the application colors but also what is defined by the operating system respectively the desktop environment....

- [Miklos Vajna: Editeng RTF export: fixing a lost paragraph style](#) (2024/12/04 10:34)

Impress shape text doesn't have much support for styles, e.g. the default UI in Writer gives you a paragraph style dropdown, and you don't get the same in Impress. Still, a paragraph style is attached to bullets based on their outline level, and Impress has a View → Outline menu item to give you that styled text you can copy. Pasting that to Writer started to lose styles recently and it's now fixed to work again. This work is primarily for Collabora Online, but the feature is available in desktop Impress as well. Motivation¶ As described in a previous commit, I had a case where lots of not needed paragraph styles were exported to RTF in case an Impress document had enough master pages. The idea was to only export actually used paragraph styles, to avoid wasting CPU power. Turns out filtering out paragraph styles has to happen at two locations: in the style table to assign an index to a paragraph style when referring to those styles The problem was that unused styles were removed from the style table, but not from the style → index mapping, so as soon as you had both used and unused paragraph styles, the declared and the referred style indexes didn't match anymore. Results so far¶ Here is a sample paste result in Writer, where you can see that the text doesn't have a custom paragraph style: Bugdoc: old Writer paste And here is the same paste, now with paragraph styles restored: Bugdoc: new Writer paste As you can see, now the pasted text has paragraph styles. How is this implemented?¶ If you would like to know a bit more about how this works, continue reading... :-) The bugfix commit was editeng RTF export: fix broken offsets into the para style table. The tracking bug was tdf#163883. Want to start using this?¶ You can get a development edition of Collabora Online 24.04 and try it out yourself right now: try the development edition. Collabora intends to continue supporting and contributing to LibreOffice, the code is merged so we expect all of this work will be available in TDF's next release too (25.2).

- [Chris Sherlock: The mess that is the VCL](#) (2024/11/29 22:58)

Let me count the ways, in no particular order and in no way exhaustive:OutputDevice is the base class for printing, windowing and PDFs. It doesn't just do output. OutputDevice has GetOutDevType() because the base class needs to know what child class is using it. Ugh. OutputDevice drawing primitives not only draw, but they record a metafile. There are literally functions that turn off drawing and just let it record the metafile. I made an attempt at separating the concerns, but it got nowhere. VCL relies on DrawingLayer and DrawingLayer relies on the VCL. There is a concept of a VirtualDevice, which is derived from OutputDevice. VirtualDevice does a bunch of things, but one of which is alpha-handling. In OutputDevice, there is a member which is a VirtualDevice. Each drawing function in Outputdevice calls upon the correlated drawing function in this member VirtualDevice. Bitmaps don't get modified via the Bitmap class. Instead, you have to use BitmapInfoAccess, BitmapReadAccess and BitmapWriteAccess. I'm still puzzling out why these are separate classes. Bitmaps are transformed in SalGraphics indirectly via OutputDevice. Except when they aren't, in which case it fails, whereby OutputDevice tries an alternative way via SalGraphics. Otherwise, it tries its own poor man approach at drawing the bitmap. Consequently, often times you bypass the platform optimized ways of doing things, because its not been implemented. Fonts are lazy loaded from OutputDevice. There is no central font manager. To get the fonts, you have to go through SalGraphics. To get a SalGraphics, you need to initialize a lot of stuff not related to fonts. Font caching is done from OutputDevice. Lazily. Font data is updated for all frames. Frames are a concept needed for Windows. Frames are not a concept needed by Printers and VirtualDevices, or even PDFs. Note that Printers, VirtualDevices and PDFs all inherit from OutputDevice. OutputDevice converts between "logical" units and display units. It's a nightmare to know what each function needs what sort of units. For the mapping between units, I refer you to vcl/source/gdi/mapmod.cxx and

vcl/source/outdev/map.cxx There is tools and basegfx. They do the same thing, though basegfx is considerably better written. You have Size and B2DSize, Point and B2DPoint, Polygon and B2DPolygon, PolyPolygon and B2DPolyPolygon. OutputDevice must handle it all. Gradient handling is sort of half baked in OutputDevice, much of gradient handling is done in other modules. Font substitution is truly, truly weird. PhysicalFontSelect::FindFontFamilyByAttributes() has clearly got a bug in it - (e.g. ImplFontAttrs::None == ((nSearchType ^ nMatchType) & ImplFontAttrs::Rounded an XOR?) and it is a truly strange weighting scheme. Yes, I did try to untangle that beast with proper unit tests, but gave up after being told I was being unreasonable. There is VCL, canvas, cppcanvas and drawinglayer. drawinglayer is way better than VCL, but we are stuck with VCL for everything. Consider the following Window hierarchy: WorkWindow inherits from SystemWindow, which inherits from Window. Window holds an OutputDevice to do stuff. WindowOutputDevice derives from OutputDevice. This is needed because OutputDevice often needs to know if it is doing Window operations, via WindowOutputDevice. Try untangling this in your head. Text layout is its own beast, and has its own set of classes. A lot of text layout is worked out in OutputDevice. Text layout is done via OutputDevice::ImplLayout(). I present to you the ImplLayout function signature: `std::unique_ptr<SalLayout> ImplLayout(const OUString&, sal_Int32 nIndex, sal_Int32 nLen, const Point& rLogicPos = Point(0, 0), tools::Long nLogicWidth = 0, KernArraySpan aKernArray = KernArraySpan(), std::span<const sal_Bool> pKashidaArray = {}, SalLayoutFlags flags = SalLayoutFlags::NONE, vcl::text::TextLayoutCache const* = nullptr, const SalLayoutGlyphs* pGlyphs = nullptr, std::optional<sal_Int32> nDrawOriginCluster = std::nullopt, std::optional<sal_Int32> nDrawMinCharPos = std::nullopt, std::optional<sal_Int32> nDrawEndCharPos = std::nullopt) const;`

- [allotropia: Precision-engineering for JavaScript \(2024/11/26 09:00\)](#)

This post is about recent improvements for ZetaJS, the JavaScript wrapper library for ZetaOffice's WebAssembly version of LibreOffice: There is something of a mismatch between the UNO type system and the JavaScript types used by zetajs. For example, JavaScript only has a single number type for both integer and floating point values, while UNO has a whole slew of different numeric types (BYTE, SHORT, UNSIGNED SHORT, LONG, UNSIGNED LONG, FLOAT, DOUBLE) that all map to that one JavaScript type. Similarly, the different UNO sequence<T> types all map to JavaScript arrays, where information about the UNO element type T is lost. Normally, that's not an issue. When you call a UNO method that returns a LONG, you get a number just like when you call a UNO method that returns a DOUBLE, and your JavaScript code then has a number to work with, and that's all. Similarly, when you call a UNO method that returns a sequence<LONG>, you get an array of numbers you can work with, just like when you call a UNO method that returns a sequence<DOUBLE>. And when you then call a UNO method that takes a sequence<LONG> as an argument, you pass in an array of numbers, and the zetajs runtimes figures out how to dress that array up as a UNO sequence<LONG>, and all is well. However, one place where UNO's insistence on more precise typing gets in the way is the UNO ANY type. It is not just a means to transport any kind of UNO value, it also carries precise type information. A UNO ANY value that contains a LONG of value 1 is something different than a UNO ANY value that contains an UNSIGNED LONG of value 1. And a UNO ANY value that contains a reference of type `css.uno.XInterface` to some UNO object is something different than a UNO ANY value that contains a reference of type `css.lang.XComponent` to the same UNO object. Again, most of the time, those precise distinctions are irrelevant to most of the code. When you call a UNO method that returns an ANY, and you know that that ANY value must contain a LONG, you just want to get a JavaScript number out, regardless of what precise numeric UNO type was encoded in that ANY value. Similarly, when you call a UNO method that returns an ANY that must contain a `css.uno.XInterface` reference, you just want to get some JavaScript object that you can do further UNO method calls on (or null), regardless of what precise UNO interface type was encoded in that ANY value. And when you then call a UNO method that takes an ANY that must contain a

LONG, you want to just pass in a JavaScript number, and the zetajs runtime shall figure out how to dress that up as a UNO ANY containing a LONG (or throw an exception, if you passed something that just can't be dressed up accordingly). But, sometimes, you need more fine-grained control. There might be a UNO method that takes an ANY argument and behaves completely differently when you pass it a LONG of value 1 or an UNSIGNED LONG of value 1. But when you call that UNO method with the JavaScript number 1, zetajs will always dress that up as a UNO ANY of type LONG for you, never as an UNSIGNED LONG. To solve that issue, the zetajs UNO binding also has the notion of a zetajs.Any JavaScript type, which records a value along with its precise UNO type. You can thus pass either a new zetajs.Any(zetajs.type.long, 1) or a new zetajs.Any(zetajs.type.unsigned_long, 1) when you call that picky UNO method. Now, when a UNO method returns an ANY value, the zetajs binding used to be conservative: You might want to know exactly what UNO type it contains (even though, most of the time you don't actually care), so it always returned those wrapped zetajs.Any objects that carry the precise contained UNO type. But that lead to awkward code. When you call e.g. x.nextElement() to get a UNO ANY that contains a reference to another UNO object, you had to unwrap that first (with zetajs.fromAny) before you could do any further calls on the obtained UNO object: zetajs.fromAny(x.nextElement()).doSomething(). But you know that this call to x.nextElement() will return an ANY containing an interface reference, and you don't care about the exact UNO interface type—you just want to do another method call on the obtained object. So, recently (in Let zetajs return unwrapped ANY representations), the zetajs binding was changed so that it now always returns unwrapped UNO ANY values: x.nextElement() no longer returns a zetajs.Any wrapper (on which you would need to call zetajs.fromAny first), it directly returns the relevant JavaScript object. And the resulting overall code looks way better: x.nextElement().doSomething(). When, in the other direction, you pass something into a UNO method that takes an ANY argument, you still have the same options you had before: Either, you simply pass the JavaScript number 1, and zetajs figures out for you that that should be dressed up as a UNO ANY of type LONG, or you want to be picky and pass in either a new zetajs.Any(zetajs.type.long, 1) or a new zetajs.Any(zetajs.type.unsigned_long, 1). And when it comes time that you do want to be picky about the ANY values that you obtain as return values from UNO method calls, there's now a \$precise way to do that: x.\$precise.nextElement() (and same for any other UNO method call) will always give you back a wrapped zetajs.Any value. See the updated The zetajs UNO Mapping for all the details.

- [LibreOffice Dev Blog: VCL weld: create LibreOffice GUI from design files](#) (2024/11/22 17:07)

LibreOffice uses VCL (Visual Class Library) as its internal widget toolkit to create the graphical user interface (GUI) of LibreOffice. Here I discuss how to use UI files designed with Glade interface designer to create LibreOffice user interfaces with a framework called weld, which is part of LibreOffice core source code. Creating a Minimal VCL Weld Application In my previous blog post, you can find out about the structure of a minimal VCL application. Please refer to the below blog post to see how a Window is created in VCL, and how it can be used as a test workbench called minvcl. You can run it with ./bin/run minvcl after you build LibreOffice. VCL application in its minimal form Here I discuss how to go further, and create user interface with Glade interface designer, and do most of the things without writing code. VCL Weld Mechanism In order to simplify user interface creation in LibreOffice, experienced LibreOffice developer, Caolán, has introduced a mechanism to load UI files created with Glade interface designer, and use them as if they are UI files for each and every GUI framework that LibreOffice supports: from GTK itself to Qt, Windows, macOS and even the so-called gen backend that only requires the X11 library on Linux. To illustrate how the VCL weld mechanism works, I have added a minimal example, minweld, as a test workbench. The structure of the code is very similar to the previous example, minvcl, but there are some changes in the code. In the new code, UI is created from a .ui file that is designed visually with Glade interface designer. The .ui file is an XML file which contains placement of widgets that should be displayed on the screen. The complete code for minweld is available in

the LibreOffice core source code repository, which can also be viewed online: OpenGrok: LibreOffice core source: `vcl/workben/minweld.cxx` Glade UI File In `minweld`, I have used an existing Glade UI file, `tipofthedaydialog.ui`. This is the user interface for displaying a tip of the day in LibreOffice at startup. Heiko, the TDF design mentor, has discussed this dialog box in detail before: Easyhacking: How to create a new “Tip-Of-The-Day” dialog But, you can assume that it is a simple `.ui` file, that one can create with Glade. Here, we use it to create our own user interface in C++.

You may use any other `.ui` file that you have created with almost the same code. Tip of the day displayed at LibreOffice startup This UI file is found in `cui/ui/config/ui/tipofthedaydialog.ui`, and `minweld` loads it. This is how it looks when you open it in Glade interface designer: `tipofthedaydialog.ui` in Glade user interface designer Let’s look into the specifics of `minweld.cxx`. Header Includes Headers are almost the same, but here we use `vcl/weld.hxx` instead of `vcl/wrkwin.hxx`. Therefore, you can see this line in the code: `#include <vcl/weld.hxx>` Then we have the C++ code for the application. The `TipOfTheDayDialog` class is defined with: `class TipOfTheDayDialog : public weld::GenericDialogController { public: TipOfTheDayDialog(weld::Window* pParent = nullptr); DECL_LINK(OnNextClick, weld::Button&, void); private: std::unique_ptr<weld::Label> m_pTextLabel; std::unique_ptr<weld::Button> m_pNextButton; sal_Int32 m_nCounter = 0; }; ... }` As you can see, `TipOfTheDayDialog` inherits from `weld::GenericDialogController`, and not `Application` class as before. Also, `TipOfTheDayDialog` constructor receives a parent of type `weld::Window*`, which is `nullptr` now. The reason is that there is no parent window in this example. Using `weld::` prefix is also done for other types of widgets that we use in LibreOffice. For example, we use `weld::Button` to denote a push button in LibreOffice, or in any application that is created with the `vcl::weld` mechanism. Class Constructor This is the code for the `TipOfTheDayDialog` constructor. Here, we initialize two member variables, `m_pTextLabel` and `m_pNextButton` which point to a label and a button, respectively. We will interact with these two in our code. There are string literals like `lbText` and `btnNext`, which are the IDs of those widgets in Glade. The IDs should be unique for linking to specific variables in the code. `TipOfTheDayDialog::TipOfTheDayDialog(weld::Window* pParent) : weld::GenericDialogController(pParent, u" cui/ui/tipofthedaydialog.ui" _ustr, u"TipOfTheDayDialog" _ustr), m_pTextLabel(m_xBuilder->weld_label(u"lbText" _ustr)), m_pNextButton(m_xBuilder->weld_button(u"btnNext" _ustr)) { m_pNextButton->connect_clicked(LINK(this, TipOfTheDayDialog, OnNextClick)); }`


One last step is linking the events with functions in the code. You may do that with the `LINK` macro. In the last line, `connect_clicked` activates `OnNextClick` from the class `TipOfTheDayDialog`, whenever `m_pNextButton` is clicked. Event Handler This is the implementation of the event handler. It should be started with `IMPL_LINK` macro, in the form of `IMPL_LINK_NOARG(Class, Member, ArgType, RetType)`. The code is straightforward: It increases a counter which is initially zero, and displays it alongside a text: `IMPL_LINK_NOARG(TipOfTheDayDialog, OnNextClick, weld::Button&, void) { ++m_nCounter; m_pTextLabel->set_label(u"Here you will see tip of the day #" _ustr + OUString::number(m_nCounter) + "."); }` With a call to `set_label` function, `m_pTextLabel` is updated every time that you click on “Next Tip” button. Running the Example You may run the example after you have built LibreOffice from sources. Then, you may simply invoke: `./bin/run minweld` The result is a little bit different from the `tipoftheday` dialog in LibreOffice, as it does not use a picture. But, it has a nice feature: if you click on “Next Tip”, it will show a text and a counter that goes up whenever you click on it again. Final Notes You may look into the original “tip of the day” dialog box in `cui/source/dialogs/tipofthedaydlg.cxx`, which is more complex than the one that we created here, as it reads some data from the configuration and uses images. But, the idea is the same. Inherit a class from `GenericDialogController`, define and link variables to the widgets with their IDs, add event handlers. Now, the application with VCL graphical user interface is ready to use! This is somehow similar to the way one creates dialog boxes with Qt and other widget toolkits. On the other hand, the VCL `weld` mechanism is different in the way that it uses such a toolkit to create UI on the fly. Therefore, if you choose a desired VCL UI plugin, then it will use that specific library for creating user interface. For example, you can

run minweld example with Qt this way: `export SAL_USE_VCLPLUGIN=qt5 ./bin/run minweld` The minweld example in Qt (light theme) You may also run it with GTK3 UI, this way: `export SAL_USE_VCLPLUGIN=gtk3 export GTK_THEME=Adwaita:light # For light/dark theme ./bin/run minweld` The minweld example in GTK3 (light theme) I hope that this explanation was helpful for you to understand the basics of GUI design and implementation in LibreOffice. You can try doing small improvements in LibreOffice GUI by looking into the EasyHacks that with the tag "Design": TDF Wiki: EasyHacks categorized by "Design" as the required skill We welcome your code submissions to improve LibreOffice. If you would like to start contributing to LibreOffice, please take a look at our video tutorial: Getting Started (Video Tutorial)

- [Marius Popa Adrian: Read free book "Detailed New Features Of Firebird 5": available as HTML and as PDF \(2024/11/21 09:05\)](#)

Dive deep into the revolutionary features of Firebird 5.0 with this comprehensive guide written by database expert Denis Simonov and edited by Alexey Kovyazin. This book offers an in-depth exploration of the significant advancements that make Firebird 5.0 a pivotal release in the world of relational databases. Whether you're a seasoned database administrator, a curious developer, or an IT

- [LibreOffice Dev Blog: Notebookbar part 1: custom widgets for the tabbed interface \(2024/11/14 15:04\)](#)

Notebookbar, or tabbed interface is an attempt to modernize LibreOffice user interface. In these series, I try to explain the implementation in LibreOffice code. In the first part, I discuss custom Glade widgets that are building blocks of Notebookbar user interface. Building LibreOffice From Sources If you haven't built LibreOffice from sources before, you can refer to can refer to this tutorial: Getting Started (Video Tutorial) The next sections assume that you have a working build environment. Custom Widgets in Glade Catalogs Notebookbar implementation consists of .ui files, configuration files and C++ implementation. Let's look into the user interface files. First time that you clone LibreOffice source code, and try to open a Notebookbar UI file like this, you may see error: `$ glade ./sc/uiconfig/scalc/ui/notebookbar.ui` You may see an error, which indicates that a required catalog related to LibreOffice is not available. Glade error To fix this issue, you have to know that Notebookbar uses custom widgets that with the Glade interface designer. These custom widgets are available from a Glade catalog with the name of LibreOffice. Inside `sc/uiconfig/scalc/ui/notebookbar.ui`, you may see these two lines: `<requires lib="gtk+" version="3.20"/> <requires lib="LibreOffice" version="1.0"/>` Glade catalogs are xml files with the keyword `glade-catalog` inside them, so we can search for this keyword: `$ git grep -l glade-catalog extras/source/glade/libreoffice-catalog.xml.in extras/source/glade/makewidgetgroup.xslt` The .in files is an input file in which the build process creates the final xml file out of it. Searching for `glade-catalog` inside the build folder results: `$ grep -lr glade-catalog ... instdir/share/glade/libreoffice-catalog.xml` As you can see, the result goes inside the folder `instdir/share/glade/`, so to be able to use the catalog, you should add this folder to the glade catalog search path. One of the easiest ways to do this, is to add it via Glade interface itself. Use  (hamburger menu), go to "Glade Preferences", and add `instdir/share/glade/` to the "Extra Catalog & Template paths". Then, reload a notebookbar UI file, and the error should go away. This setting is saved inside `~/.config/glade.conf` configuration file. If you want to get a preview of the UI file, you need to set the environment variable first: `$ export GLADE_CATALOG_SEARCH_PATH=$PWD/instdir/share/glade $ glade-previewer -f sw/uiconfig/swriter/ui/notebookbar.ui` Custom Widgets for the Notebookbar Inside the Glade custom catalog `instdir/share/glade/libreoffice-catalog.xml`, you can see 10 custom widgets: `$ grep "glade-widget-class" instdir/share/glade/libreoffice-catalog.xml <glade-widget-class title="Notebookbar ToolBox" name="sfxlo-NotebookbarToolBox" generic-name="Notebookbar ToolBox" parent="GtkToolbar" icon-name="widget-gtk-toolbar"> <glade-widget-class title="Notebook switching tabs depending on context" name="sfxlo-NotebookbarTabControl" generic-name="NotebookbarTabControl" parent="GtkNotebook" icon-name="widget-gtk-notebook"/> <glade-widget-class title="Horizontal box hiding children depending on its priorities" name="sfxlo-PriorityHBox" generic-name="PriorityHBox" parent="GtkBox" icon-name="widget-gtk-`

```

box"/> <glade-widget-class title="Horizontal box hiding children depending on its priorities" name="sfxlo-PriorityMergedHBox" generic-
name="PriorityMergedHBox" parent="GtkBox" icon-name="widget-gtk-box"/> <glade-widget-class title="Box which can move own content to
the popup" name="sfxlo-DropdownBox" generic-name="DropdownBox" parent="GtkBox" icon-name="widget-gtk-box"/> <glade-widget-class
title="Box which can hide own content" name="VclOptionalBox" generic-name="VclOptionalBox" parent="GtkBox" icon-name="widget-gtk-
box"/> <glade-widget-class title="Vertical box hiding children depending on context" name="sfxlo-ContextVBox" generic-name="ContextVBox"
parent="GtkBox" icon-name="widget-gtk-box"/> <glade-widget-class title="Managed Menu Button" name="svtlo-ManagedMenuButton" generic-
name="ManagedMenuButton" parent="GtkButton" icon-name="widget-gtk-button"/> <glade-widget-class title="NotebookBar Toolbar Addons"
name="NotebookBarAddonsToolMergePoint" generic-name="ShowText" parent="GtkToolButton" icon-name="widget-gtk-toolbutton"/> <glade-
widget-class title="NotebookBar MenuItem Addons" name="NotebookBarAddonsMenuItemMergePoint" generic-name="ShowText"
parent="GtkMenuItem" icon-name="widget-gtk-menuitem"/>

```

The previous xml shows the custom widgets that are building blocks of building Notebookbar. Let's look into each of them, based on their title and names. Notebookbar widgets In the next picture, you can see the notebookbar in LibreOffice, and compare it to what is visible in Glade user interface designer. As you can see, not everything is visible in the designer. Specifically, icons and text are not visible in the designer but are visible in the final application. Notebookbar in LibreOffice Main Widget 1. Notebookbar Tab Control: This widget has the name sfxlo-NotebookbarTabControl, and is the primary widget for Notebookbar. It can change the set of visible tabs based on the user context. Its parent class is GtkNotebook and provides context-sensitive tab switching. Container Widgets 2. NotebookbarToolBox: This widget is named sfxlo-NotebookbarToolBox, its parent class is GtkToolbar. It can contain toolbar elements. NotebookbarTabControl 3. Priority Horizontal Box: This widget has the name sfxlo-PriorityHBox, and its parent class is GtkBox. It is the horizontal box hiding children depending on its priorities. In this way, lower priority widgets becomes hidden to give the more important widgets room to be displayed on a screen that is not big enough to show all the available elements. 4. Priority Merged Horizontal Box: This widget has the name sfxlo-PriorityMergedHBox, and its parent class is GtkBox. It is the "horizontal box hiding children depending on its priorities". This widget is also related to the previous one for creating more room for important widgets, but it is used inside the PriorityHBox. 5. Optional Box: This widget has the name VclOptionalBox, and its parent class is GtkBox. This "box which can hide own content", is a widget that is useful for creating small areas dedicated to a specific purpose. For example, you may see Home-Section-Clipboard, which is used to define an area for clipboard related tasks inside Home tab. 6. Contextual Vertical Box: This widget has the name sfxlo-ContextVBox and is a "vertical box hiding children depending on context" and its parent class is GtkBox. It provides a box that can act based on the context, showing and hiding its children accordingly. You may look into sw/uiconfig/swriter/ui/notebookbar_single.ui, which provides an example use. Here is the correct control hierarchy, as depicted and described in the TDF Wiki: Correct Notebookbar controls hierarchy Menu Widgets 7. Dropdown Box: This widget has the name sfxlo-DropdownBox, its parent class is GtkBox and is a "box which can move own content to the popup". This is also useful where the space for the tabbed interface is not big enough. The menu, is what you can see in "File" and "Help" menu in every notebookbar in LibreOffice tabbed interface. Please note that only 1 GtkBox child should be inside it, so that the popup works properly. In fact, the above diagram shows the correct usage. 8. Managed Menu Button: This widget has the name svtlo-ManagedMenuButton, and its parent class is GtkButton. It is a "Managed Menu Button". It provides a button that opens a dynamic menu which is populated according to the context. Custom Widgets for the Extensions 9. NotebookBar MenuItem Addons: This widget has the name NotebookBarAddonsToolMergePoint, and its parent class is GtkToolButton. Specifically, LibreOffice extensions can use it for adding additional tools to the notebookbar. 10. NotebookBar MenuItem Addons: This widget has

the name `NotebookBarAddonsMenuMergePoint`, and its parent class is `GtkMenuItem`. This is also used for adding extra items into the notebookbar. Final Notes You can find useful information about Notebookbar in the design team blog: [Easyhacking: How to set up your environment](#) [DIY UI: How to create your own Notebookbar](#) [Evolving Past the Restrictions of Toolbars](#) And at last, these are some useful Wiki articles around Notebookbar in the TDF Wiki: [TDF Wiki - Notebookbar](#) [TDF Wiki - Notebookbar implementation](#) [TDF Wiki - Create new dialog in Impress](#)

- [allotropia: Announcing ZetaOffice, a new LibreOffice Technology product for web, mobile & desktop](#) (2024/11/08 10:59)
Hamburg and Bolzano, November 8th, 2024 – During the two-day annual South Tyrol Free Software Conference, allotropia software GmbH today announces beta versions of its new product line “ZetaOffice”. ZetaOffice is a new set of applications, libraries and services, all powered by the LibreOffice Technology stack. Featured among its products is ZetaJS, an innovative browser-based plugin, with unique programmability & embeddability – the perfect tool for complex office editing, process automation and line-of-business applications in the web. Additionally, leveraging the unique portability and flexibility of the LibreOffice Technology stack, ZetaOffice will be available in bit-by-bit identical versions (allowing for perfect interoperability and feature parity) also for open-source-based mobile operating systems (Android, and derived OS), as well as for all relevant desktop operating systems (Windows, macOS, Linux – via flatpak and snapcraft). “We’re very excited being able to offer powerful, data-sovereign Open Source office functionality on even more platforms today”, says Thorsten Behrens, owner and managing director of allotropia software. “In particular our innovative, WASM-based browser version of LibreOffice will be a game-changer for every web developer in need of processing, analysing or integrating with office documents.” “This could not have come at a better time”, says Michiel Leenaars, director of strategy at philanthropic investor NLnet Foundation. “It is long overdue but certainly in the wake of the recent geo-political developments, we all recognise the urgent need for Europe to regain its technological independence when it comes to core technologies – as boring as these may seem. ZetaOffice shows that Europe has the talent and capacity to break with the past and create new paradigms and use innovation and collaboration to save the day.” “ZetaOffice is the perfect addition to our portfolio of tools for document and business process automation”, says Uli Brandner, CEO and owner of CIB Group. “With solutions like CIB flow for workflow modeling and CIB coSys for high-quality template management, CIB Group already offers powerful digitalization tools. As demand grows to bring proven applications to the web and stay on the cutting edge of technology, ZetaOffice stands out as an innovative solution precisely tailored to our customers’ needs.” A detailed blog post, including links to beta versions of the software, is available [here](#). For the products, please refer to our website at [zetaoffice.net](#). ZetaOffice and the team at allotropia thanks the European Commission’s Next Generation Internet initiative/NGI Zero for its financial contribution to the development of this software. About ZetaOffice: ZetaOffice is a product line based on LibreOffice Technology, comprising of desktop LTS products for classical office productivity requirements; a browser-native version based on WebAssembly for fast, client-side integration and automation of office technology; and an upcoming mobile app widget, for deep integration in mobile line-of-business applications. ZetaOffice is focused on speed, superb embeddability, excellent inter-product as well as Office compatibility, and geared towards digital-sovereign & data protection needs. About ZetaJS: ZetaJS is a JavaScript library, available via the npm package manager, to enable developers to quickly & conveniently embed ZetaOffice WebAssembly in web applications. ZetaJS makes available the entire gamut of the LibreOffice programmability interfaces, providing a web-native component for JavaScript developers to deeply embed an office suite into their web apps. In contrast to classical cloud-office setups, ZetaJS can be used as an integral, client-side part of any web application – permitting users to interact with office documents as part of a larger application framework, with very low latency. That way, e.g. direct integration for editing, suggestions or running calculations in

complex spreadsheets can be provided. Similarly, it's trivially easy to implement direct, client-side rendering and export of office documents into PDF or HTML – all via a self-hostable, digital-sovereign Open Source solution. About allotropia software GmbH: The company allotropia software GmbH provides services, consulting and products around LibreOffice and related opensource projects. Founded in 2020 by long-time developers of the project, its stated mission is to make LibreOffice shine – in as many different shapes and forms as necessary to serve modern needs towards office productivity software. allotropia software GmbH is headquartered in Hamburg, Germany at the birthplace of the OpenOffice/LibreOffice project. For more information, visit allotropia.de, or follow fosstodon.org/@allotropia on Mastodon and LinkedIn: www.linkedin.com/company/allotropia-software-gmbh

- [allotropia: Launching ZetaJS for ZetaOffice](#) (2024/11/08 10:58)

Today allotropia has launched the ZetaOffice range of products at the SFSCON in South Tyrol. ZetaOffice is a LibreOffice Technology built & designed for professional use in the browser, on the desktop and on mobile. We are excited to additionally announce a massively improved way for which LibreOffice Technology can be used fully client-side on the web. As an additional building block, we have developed the ZetaJS wrapper, which enables convenient embedding and automating WASM (WebAssembly) builds of ZetaOffice via JavaScript. With that, all of the LibreOffice Technology APIs and features are available to web applications – and by leveraging WASM, which runs ZetaOffice client-side, no server or cloud services are needed. All processing is taking place on the client browser, which minimizes latencies & load (of course, a minimal static delivery of web application code, assets and the WASM binary is still needed, but that's extremely light-weight).
Examples Let's look at some simple examples to give you an idea, how easy ZetaOffice integration is. All comprise of an HTML and a JavaScript file. A ZetaOffice WASM build will automatically be included from the following URL. To replace it with a custom WASM build see `config.sample.js` of each demo.

https://cdn.zetaoffice.net/zetaoffice_latest/ Next you need to upload the `zetajs/` folder onto a webserver of your choice, which sets the following HTTP headers (see developer.mozilla.org for further details): `Cross-Origin-Opener-Policy "same-origin"Cross-Origin-Embedder-Policy "require-corp"` So back to the example code. The HTML files for all examples embed ZetaOffice and some JavaScript loading code. Please check the actual JavaScript file for the code interacting with ZetaOffice. Lets have a look at the `simple.html` (see live). ZetaOffice displays its content using an HTML canvas. So in line 14 we initialize this canvas. Currently a list of attributes like is needed for the canvas. But we will migrate those attributes to the ZetaJS wrapper, so they won't be needed anymore in the HTML code. `<canvas id="qtcanvas" contenteditable="true" oncontextmenu="event.preventDefault()" onkeydown="event.preventDefault()" style="height:100%; width:100%; border:0px none; padding:0;"/>` The `Module` variable on line 30 passes the information needed to initialize WASM binaries. First is the canvas. And second is an array of JavaScript files which will be executed in the main Web Worker running the WASM binary. Web Workers are a process like feature of the browsers WASM runtime environment. We pass the ZetaJS wrapper and a file with custom JavaScript code, in this example the `simple.js`. You may need to ensure, that the `zeta.js` is reachable under the given URL path. Line 33 to 39 preload the `soffice.js` file to ensure, it's not being blocked by the browsers origin policy when loaded from a foreign origin. Line 42 triggers a website resize event, to make ZetaOffice display nicely inside the canvas. This can be done more precise, as shown in the more complex demos. But for the start the resize event will be triggered after a fixed interval. And finally the `soffice.js` document is finally loaded which triggers the start of the WASM binary. Second is the `simple.js` file. It's running inside the same Web Worker as the WASM binary to enable interaction. When running in Chromium / Google Chrome you will find a dropdown list labeled "top" at the upper left of the "Console" tab in the developer tools. There you can select the `em-pthread_1` Web Worker to debug code in the `simple.js` file. Inside the `simple.js` you will find pretty much the same code as when controlling a LibreOffice running naively on Linux,

Windows or any other native OS. It is using LibreOffice's UNO interface. Most existing examples using UNO via Python or Basic can be easily moved to JavaScript. The control flow is being passed by the `Module.zetajs.then` which gets called as soon as the WASM binary is loaded. It passes the `zetajs` object from which we first get the common `com.sun.star` object (do not confuse its abbreviation `css` with HTML CSS). In the lines 11 to 21 we get some control objects via UNO, which allow us to trigger the load of an example office document `example.odt` which is embedded in the WASM binary. `Module.zetajs.then(function(zetajs) { function getTextDocument() { const css = zetajs.uno.com.sun.star; const context = zetajs.getUnoComponentContext(); const desktop = css.frame.Desktop.create(context); let xModel = desktop.getCurrentFrame().getController().getModel(); if (xModel === null || !zetajs.fromAny(xModel.queryInterface(zetajs.type.interface(css.text.XTextDocument)))) { xModel = desktop.loadComponentFromURL('file:///android/default-document/example.odt', '_default', 0, []); } const toolkit = css.awt.Toolkit.create(context);` Line 27 is where the actual application logic starts. In this simple example we get a cursor object from the document to insert the text string here! at the top. In the final section from line 32 to 38 each paragraph of the office document becomes colored in a random color. `const xText = xModel.getText(); const xTextCursor = xText.createTextCursor(); xTextCursor.setString("string here!"); } { const xModel = getTextDocument(); const xText = xModel.getText(); const xParaEnumeration = xText.createEnumeration(); for (const next of xParaEnumeration) { const xParagraph = zetajs.fromAny(next); const color = Math.floor(Math.random() * 0xFFFFFFFF); xParagraph.setPropertyValue("CharColor", color); }` This other simple-examples/ show you a little more interesting tasks you can do with the same basic techniques as shown here. While the HTML files are all the same, the `simple_key_handler.js` (see live) shows you how to register to ZetaOffice event handlers. And finally `rainbow_writer.js` (see live) uses this to implement a small tool coloring text as you write it. More Complex Examples The next big step is in the `standalone/` (see live) example. It adds a nice loading animation and shows you how to pass messages between the WASM Web Worker and the browsers main thread, handling the HTML page. This is being used to implement some simple controls on the HTML page for formatting text inside ZetaOffice. The demo is build as a npm package and can be run according to the contained `README.md`. Don't forget to pass an URL to the `soffice_base_url` variable as explained above! Additional examples are `vuejs3-ping-tool/` (see live) and `letter-address-tool/` (see live). The `vuejs3-ping-tool/` is again a npm package, and show-cases how to automatically fill spreadsheets documents with values, displaying them in nicely animated Calc charts. The other `letter-address-tool/` example gives you an impression how to connect ZetaOffice with external data sources to automatically create letters from templates, and export the result as office document or PDF file. Please share your feedback as a comment in the blog, or use the GitHub issue tracker for suggestions or bugs in the code!

- [Miklos Vajna: Handling page captures for Writer TextBoxes](#) (2024/11/08 07:58)

Writer TextBoxes provide the user with shapes that can have complex geometry and complex content. There is also a feature to capture shapes inside page boundaries: now the two features interact with each other better. This work is primarily for Collabora Online, but the feature is available in desktop Writer as well. Motivation¶ As described in a previous post, Writer implements the TextBox feature with a pair of objects: a Draw shape (with complex geometry) and a (hidden) Writer TextFrame, providing complex content. To avoid wrapping problems, the underlying TextFrame always has its wrap type set to "through", i.e. text may wrap around the Draw shape, but the hidden TextFrame is always ignored during text wrapping. In most cases this provides the expected behavior, because the user sees one object, so wrapping around at most one object is not surprising. However, there is also an other feature, that shapes may be captured inside page frames: if their position would be outside the page frame, Writer corrects this, so they are not off-page. This also makes sense, so it can't happen that your document has a shape that is hard to find, due to a silly position. The trouble comes when these two are combined: the Draw shape's position gets adjusted to be

captured inside the page frame, but the TextFrame's wrap type is "through", and objects with this wrap type are an exception from the capturing mechanism, so the position of the two shapes get out of sync. Results so far¶ The problem is now solved by improving the layout, so in case the TextFrame is actually part of a Draw shape + TextFrame pair (forming a TextBox), then we calculate the effective wrap type of the TextFrame based on the wrap type of its Draw shape, so either both objects are captured or none, which results in consistent render result. Here is a sample document where all margins are configured to be equal, but capturing corrected the Draw shape (and not the TextFrame): Bugdoc: old Writer render And here is the same document, with consistent positioning: Bugdoc: new Writer render As you can see, now the rendered margins actually equal, as wanted. How is this implemented?¶ If you would like to know a bit more about how this works, continue reading... :-)

The bugfix commit was `sw textbox: capture fly when its draw object is captured`. The tracking bug was `tdf#138711`. Want to start using this?¶ You can get a development edition of Collabora Online 24.04 and try it out yourself right now: try the development edition. Collabora intends to continue supporting and contributing to LibreOffice, the code is merged so we expect all of this work will be available in TDF's next release too (25.2).

- [Miklos Vajna: Per-paragraph semi-transparent shape text in Impress](#) (2024/10/04 06:22)

The SVG export in Impress now supports a per-paragraph setting to handle semi-transparent shape text, while previously this was only possible to control at a per-shape level. This work is primarily for Collabora Online, but the feature is available in desktop Impress as well. Motivation¶ As described in a previous post, Impress already had the capability to have semi-transparent shape text, but the SVG export of this for the case when not all paragraphs have the same setting was broken. Transparency in SVG can be described as a property of a group (`<g style="opacity: 0.5">...</g>`) and it can be also a property of the text (`<tspan fill-opacity="0.5">...</tspan>`). The SVG export works with the metafile of the shape, so when looking for meta actions, it tries to guess if the transparency will be for text: if so, it needs to use the `tspan` markup, otherwise going with the `g` markup is OK. What happened here is that meta action for a normal text started, so the SVG export assumed the text is not semi-transparent, but later the second line was still transparent, so we started a group element, and this resulted in a not even well-formed XML output. Results so far¶ The relevant part of the test document is simple: just 3 paragraphs, the second one is semi-transparent (and also has a bullet, as an extra): Bugdoc: original Impress render Once this was exported to SVG, this resulted in a non-well-formed XML, so you got this error in a web browser: Bugdoc: old SVG render Once tweaking the transparency mask writer to check if text started already, we get the correct SVG render: Bugdoc: new SVG render How is this implemented?¶ If you would like to know a bit more about how this works, continue reading... :-)

The bugfix commit was `SVG export: fix handling of semi-transparent text inside a list`. The tracking bug was `tdf#162782`. Want to start using this?¶ You can get a development edition of Collabora Online 24.04 and try it out yourself right now: try the development edition. Collabora intends to continue supporting and contributing to LibreOffice, the code is merged so we expect all of this work will be available in TDF's next release too (25.2).

- [Björn Michaelsen: Nothing ever happens](#) (2024/09/19 23:30)

Nothing ever happens. And nothing ever happens, nothing happens at all The needle returns to the start of the song And we all sing along like before -- Del Amitri, Nothing Ever Happens In my last post on Libreoffice I promised to talk about Writer changes once in a while, but then ... nothing ever happened. However, given that I had an annoying motorcycle accident in the meantime that turned out much more persistently annoying than originally thought, I think I have a bit of an excuse. So ... what did happen? For one, I fixed quite a few regressions with my name on them, but ... is there much to talk about here? Mostly not: If you look at the fixes, they are often oneliners fixing something that seems rather obvious in retrospect. The more tricky question is: how did these get in in the first place? Its hard for me to say that, as the introducing commits

are from even longer ago. One thing is certain though: Often a unittest would have caught them, so whenever possible, I tried to create a reproducer adding such a test with the fix. To anyone writing bug reports: Creating minimal reproduction test is hugely valuable in this -- not just for finding the issue, but also as a starting point for a regression test. So if a bug bugs you and it is missing a minimal reproduction scenario, adding one is a great way to move this forward. Oh, and maybe verifying a bugfix, if someone provided a fix and the friendly bot say affected users are "encouraged to test the fix and report feedback". While doing these fixes, I stumbled over Noel suggesting to speed up bookmarks in writer which is of course great, but I noticed that the code could be optimized a bit more as the bookmarks of a document are now sorted by their starting position (which was one of the first changes I made back on OpenOffice.org about more than a decade ago). Thus we can use bisectional search on the bookmarks here, which should be even faster. Now, it would be great if the discussion on this between Noel and me would be available for others to learn from, wouldn't it? The cool thing is: it is. All discussion happened on gerrit in the comments so if you want to learn about bookmark in Writer and how to maybe speed them up for documents that have a lot of them, that is a great starting point! Is there anything to add? Well maybe the following: Currently the bookmarks starting at the same position are currently not sorted. If one would sort them by their end position, the bisectional search could maybe cover even more? This would also remove one extra loop of logic and make the code simpler and easier to read. The performance improvement is likely irrelevant -- esp. since there will be not that many documents with lots of bookmarks starting at the same position. The simpler code might be worth it though. So why wasn't it done? It still can be tried in a follow-up, but speaking about regressions earlier: This has some obscure regression risk, because if we change the order of bookmarks starting at the same position from undefined to something ordered by the end position it might impact a lot of code using bookmarks. The function in question might actually be faster, but other functions (e.g. the inserting of new bookmarks) might actually be slower. So ... this is left as an exercise to the reader. Comments? Feedback? Additions? Most welcome here on the fediverse !

From:

<https://wiki.tromjaro.alexio.tf/> - **TROMjaro wiki**

Permanent link:

<https://wiki.tromjaro.alexio.tf/doku.php?id=news:planet:documentfoundation>

Last update: **2021/10/30 11:41**

