What the Cell?!  
Unexpected things happening under the grid

Eike Rathke  
Red Hat, Inc.  
FOSDEM 2017-02-04
Agenda

- About the Speaker
- Accessibility (a11y) is your friend and your foe
- We don’t know where this is
  - So ask infinitely until we do know (which is never)
- Performance examples
- Q&A
About the Speaker

- Eike Rathke, known on the net as erAck
- Based in Hamburg, Germany
- Worked on StarOffice from 1993 to 2000 for Star Division
- Worked on OpenOffice.org from 2000 to 2011 for Sun Microsystems and one other company
- Works on LibreOffice since 2011, employed by Red Hat, Inc.
- Areas of expertise:
  - Calc core, formula compiler and interpreter
  - number formatter/scanner
  - i18n framework, locale data
- Also mentor and knowledge spreader whenever possible
- Web site http://erack.de/
Accessibility (a11y) is your friend

- Provides information to screen readers
- Tells which object is active
- Reads text content of selected objects (edit fields, listbox entries, cells …)
- Listens to object changes
Accessibility (a11y) is your foe

- Listens to object changes
- which can slow things down
- gets in the way at the most inconvenient occasions
EditEngine when typing text in a cell

Cell with text

One line with text.
EditEngine when typing text in a cell

- Cell with text

One line with text.

- Inserting text with paragraph end / newline

One line and inserted paragraph\nwith text.
EditEngine when typing text in a cell

- Cell with text

  One line with text.

- Inserting text with paragraph end / newline

  One line and inserted paragraph with text.

- As the paragraph end (¶) is inserted, a11y gets notified so it could read out that “with text” is now on the next line.
EditEngine when pasting text into a cell

- Cell with text

One line with text.
EditEngine when pasting text into a cell

- **Cell with text**
  
  One line with text.

- **Pasting text with paragraph end / newline**
  
  One line and pasted paragraph
  
  And more with text.
EditEngine when pasting text into a cell

- Cell with text
  
  One line with text.

- Pasting text with paragraph end / newline
  
  One line and pasted paragraph
  And more with text.

- As the paragraph end (¶) is inserted, a11y gets notified
As the paragraph end (¶) is inserted, a11y gets notified
- a11y asks the engine for the new distribution of text
- 0. the text is not reformatted yet
  - 1. engine knows there is text following, appends a line allocating memory
  - 2. due to the internal algorithm, remaining text in this stage is not distributed to the just appended line
  - 3. goto 0.
- ... until due to out of memory (if lucky) the process gets killed
- ... or filling swap grinds the system to halt, tdf#104152
And DOOM it doesn’t go anymore

- The actual fix was to suppress the change notification and postpone/delay the broadcast until the end of the Paste operation, commit `fff0cf20e4020af9e2c134549a5164417fde30b7`

- Rewrote part of the algorithm so that it at least does not loop infinitely in case another scenario hits the same circumstances, commit `7c20d0174c59d46b11fc5029fe3fc0c00f5dc6d0`
Performance #1

- Loading cell formulas from ODF, significant time was spent under ScCompiler::IsValue() calling the number parser SvNumberFormatter::IsNumberFormat()
- Unnecessary because for ODFF only well-formed numbers in an en-US representation have to be recognized, so use rtl::math::stringToDouble() instead

- Test case, before:

<table>
<thead>
<tr>
<th>Ir</th>
<th>Irpc</th>
<th>Callee</th>
</tr>
</thead>
<tbody>
<tr>
<td>9859177</td>
<td>9859</td>
<td>ScCompiler::IsValue</td>
</tr>
<tr>
<td>6246858</td>
<td>6246</td>
<td>SvNumberFormatter::IsNumberFormat</td>
</tr>
<tr>
<td>3496261</td>
<td>3496</td>
<td>SvNumberFormatter::GetStandardIndex</td>
</tr>
</tbody>
</table>

- Test case, after:

<table>
<thead>
<tr>
<th>Ir</th>
<th>Irpc</th>
<th>Callee</th>
</tr>
</thead>
<tbody>
<tr>
<td>298000</td>
<td>298</td>
<td>ScCompiler::IsValue</td>
</tr>
<tr>
<td>248000</td>
<td>248</td>
<td>rtl_math_uStringToDouble</td>
</tr>
</tbody>
</table>

- Factor 33 ...
Performance #2

- Calling `dynamic_cast<foo*>(bar)` 100000 or a million times is slow.
- Repeat: calling `dynamic_cast<foo*>(bar)` 100000 or a million times is slow.
- Commit 7d8196ea2f4ec3634dbad7367345e62c4ea9893d
  - eliminated `SfxSimpleHint` and moved all to `SfxHint`
    - which was derived from `SfxSimpleHint` and ~all places did `dynamic_cast`
  - made 150 `dynamic_cast` superfluous
- Specifically in Calc `ScFormulaCell::Notify()` benefits and doesn’t have to `dynamic_cast` before evaluating the `SfxHintId` and possibly determining it doesn’t have to do anything...
Performance #3

- ScDocument::TrackFormulas() was called too often, due to multiple individual broadcasts
- Instead, track/collect pending formula cells at the end of the bulk broadcast and process them en block
- In the insert rows scenario of tdf#87101 that reduced InstructionsRead by factor 6 and a wall clock speedup by factor 2
Performance #4

- This little innocent piece of code in ScAttrArray::HasAttrib()
  Search( nRow1, nStartIndex );
  Search( nRow2, nEndIndex );

- could be replaced with
  Search( nRow1, nStartIndex );
  if (nRow1 != nRow2)
    Search( nRow2, nEndIndex );
  else
    nEndIndex = nStartIndex;

- because in only 9217 calls out of 36791233 nRow2 differed from nRow1, so only one Search() call was needed
Thank you …

- … for using LibreOffice!
- … for supporting LibreOffice!
- … for hacking LibreOffice!