

Online: wrestling Copy / Cut / Paste *to usability*

Michael Meeks <michael.meeks@collabora.com>

General Manager at Collabora Productivity

@michael_meeks, mmeeks #libreoffice-dev irc.freenode.net



*"Stand at the crossroads and look; ask for the
ancient paths, ask where the good way is, and
walk in it, and you will find rest for your souls..." -
Jeremiah 6:16*



Overview

Web copy/paste technologies

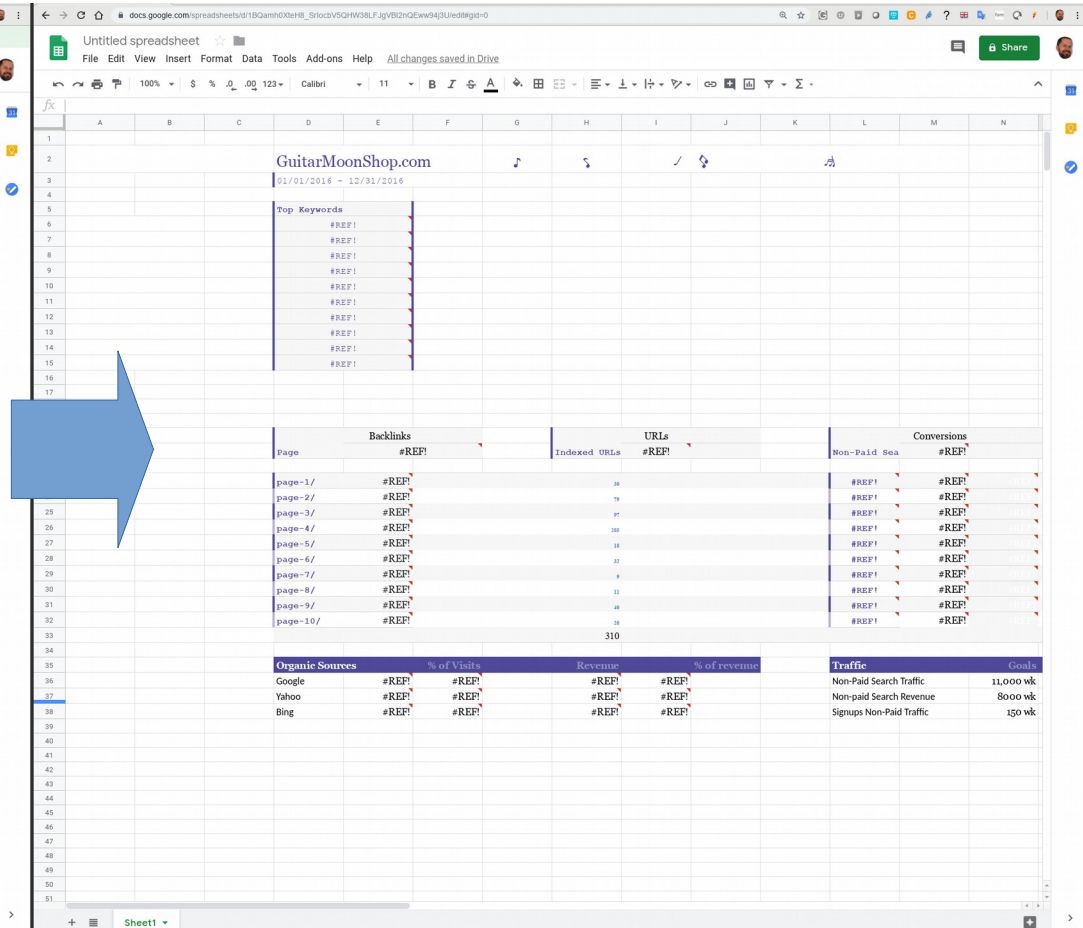
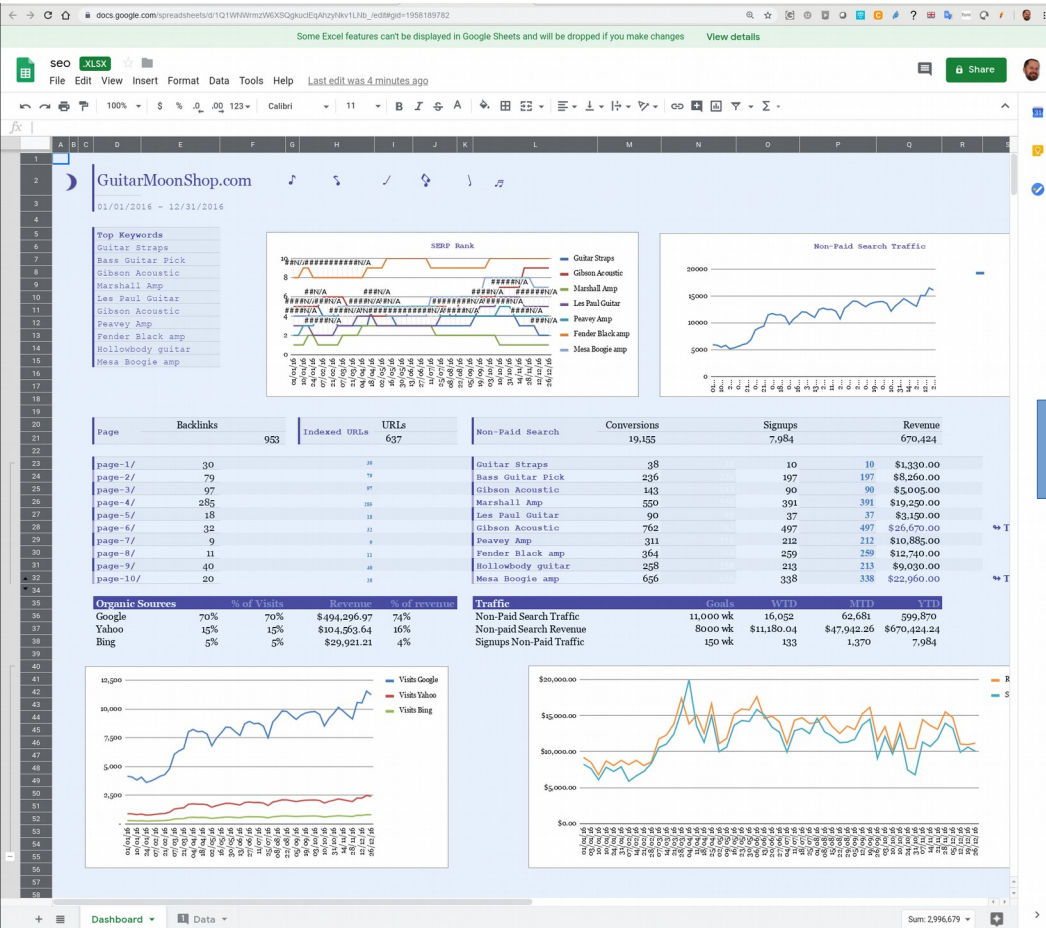
- Baroque and paranoid
- Riddled with inelegant back-compatibility
- Layers of un-necessary and unhelpful 'security' features.

Comparison / before and after

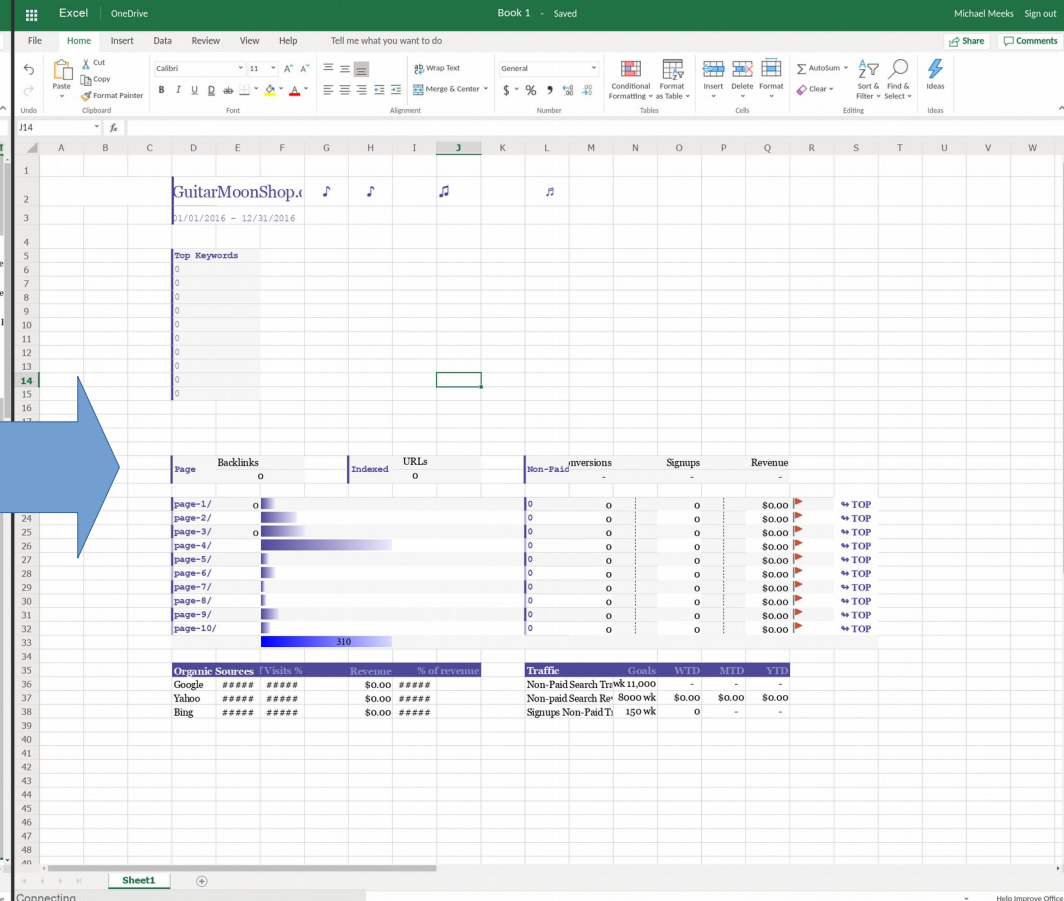
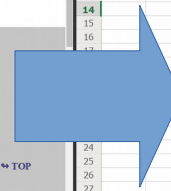
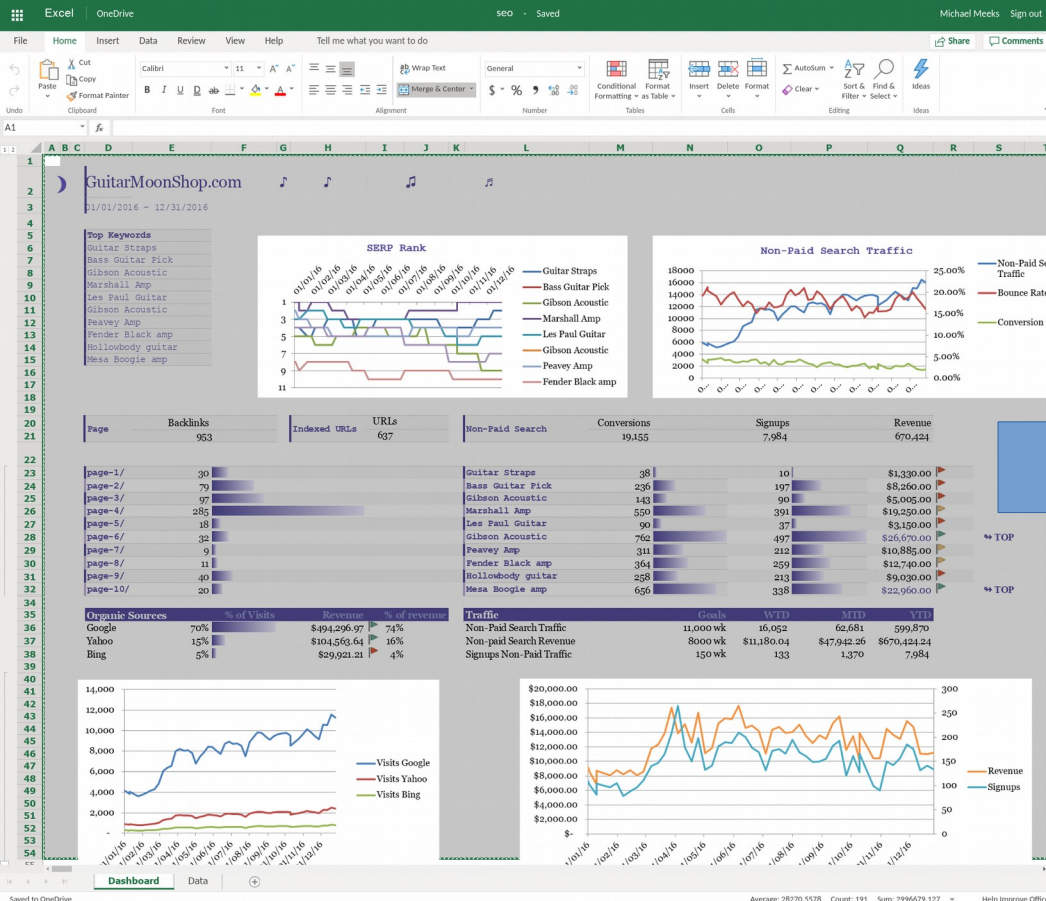
How we improved things.

Beforehand:
the Competition &
the Problem

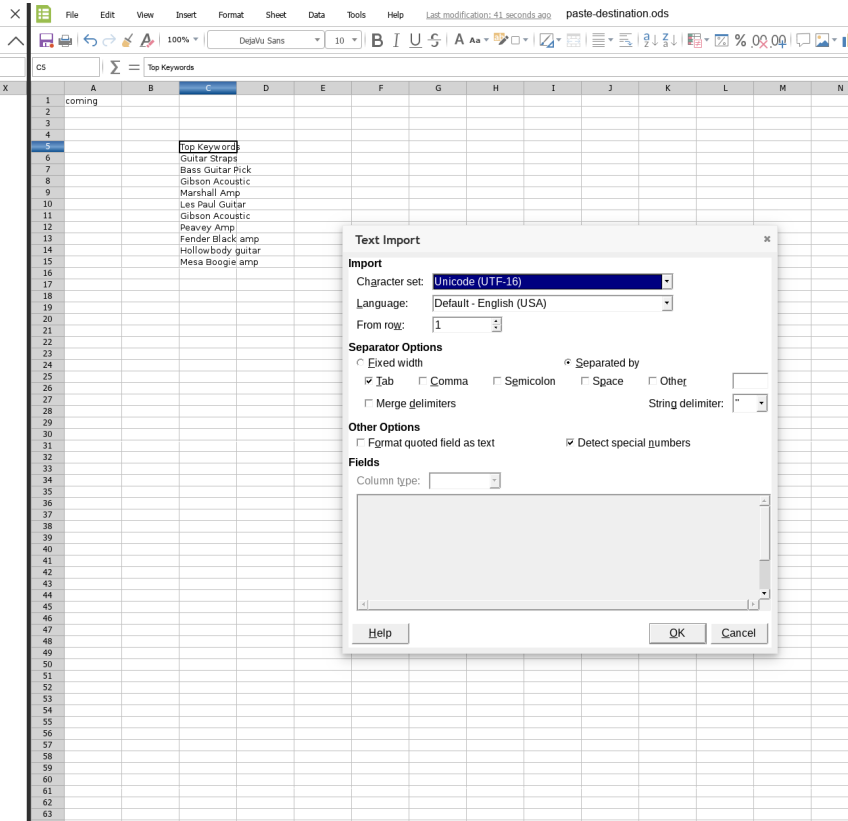
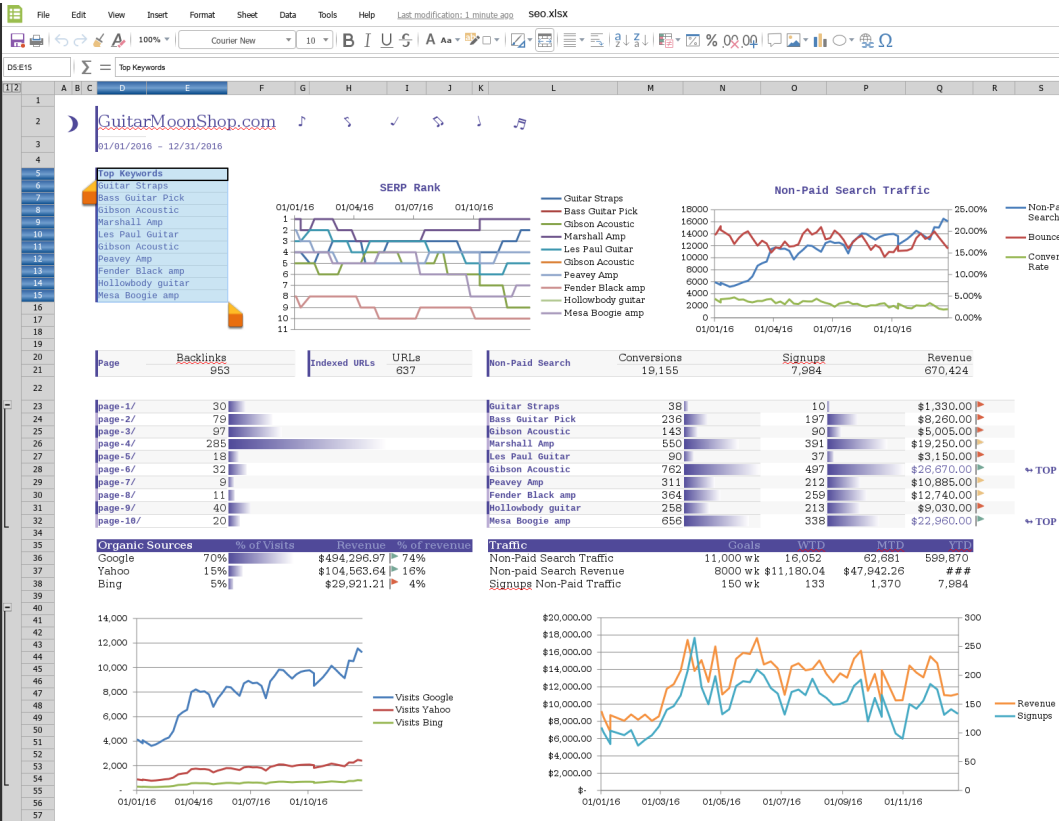
Chrome / GDocs spreadsheet copy/paste



Office.com / Excel copy / paste ...



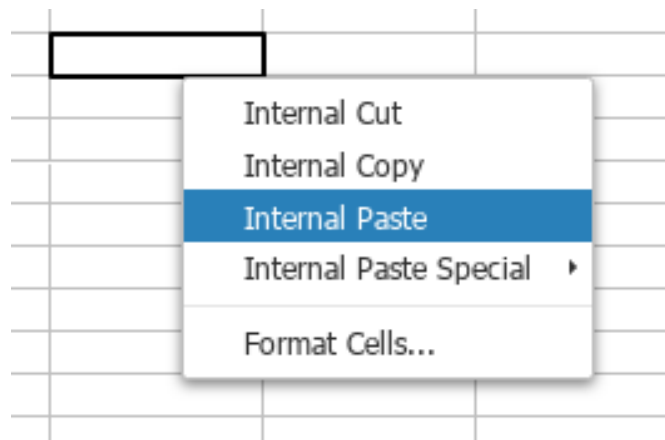
Online – old-style copy/paste: text → CSV import dlg.





High fidelity 'internal' shortcut ...

- A large proportion of copy/paste is for shuffling / pruning bits of the current document.
- So – initial context-menu / copy-paste short-circuited internally
 - Doing all the work inside the Online server.
- Particularly important for eg. large spreadsheet cell area movement.
 - This requires formula re-writing.
- Leaving plain-text for external use.



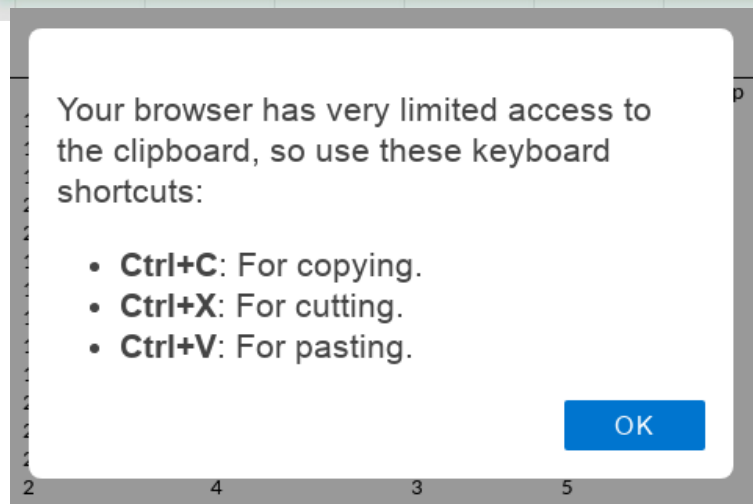
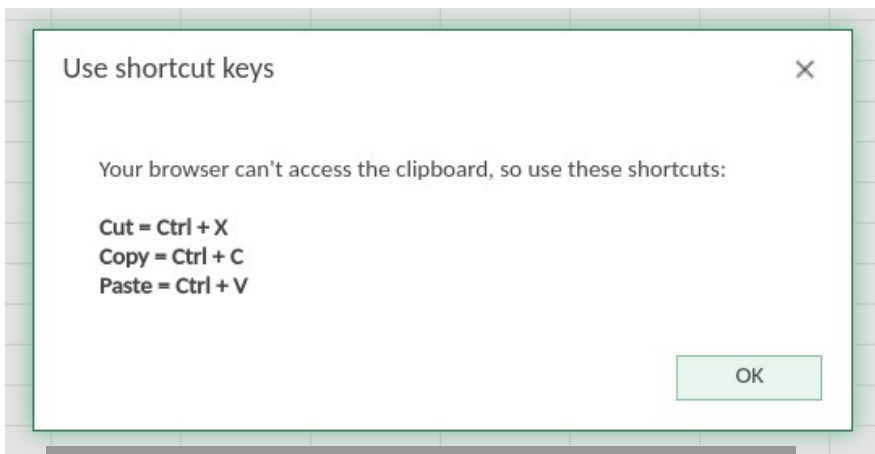
Web Clipboard technologies.



Web copy/paste is baroque & paranoid beyond belief.

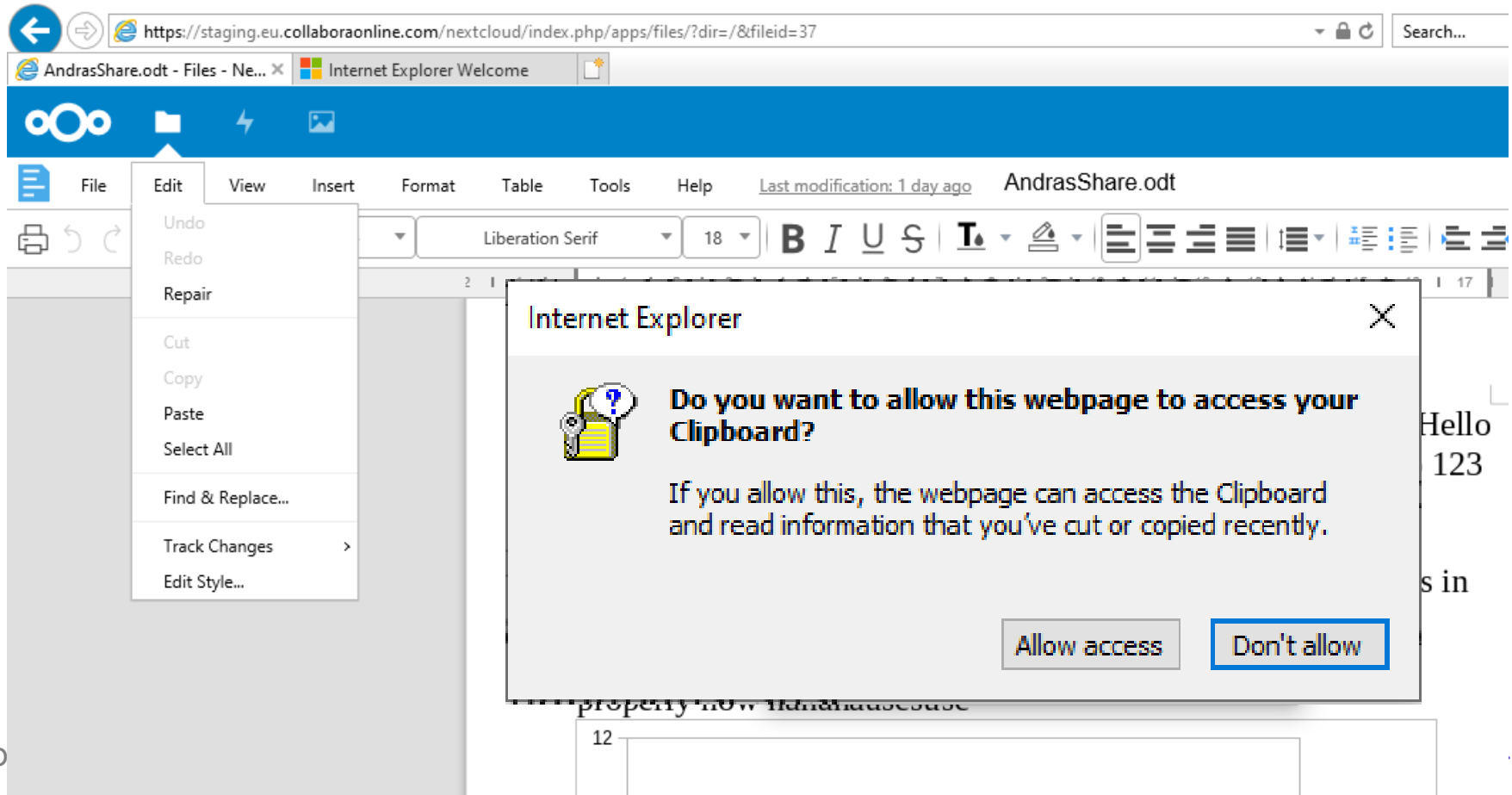
APIs for copy/paste are very mixed

- In many browsers copy/cut can put ~anything on the clipboard ~whenever they like.
 - But they have to work-around layers of obsolete 'security' foo.
- 1. Find/focus/select content-editable
 - 2. `document.execCommand('copy')`
 - 3. catch events if you get them
 - 4. tweak your content-editable,
 - 5. retry from 1.





IE11 – had a great initial approach:





Google Chrome / Docs – joined up thinking ?

Google Chrome's

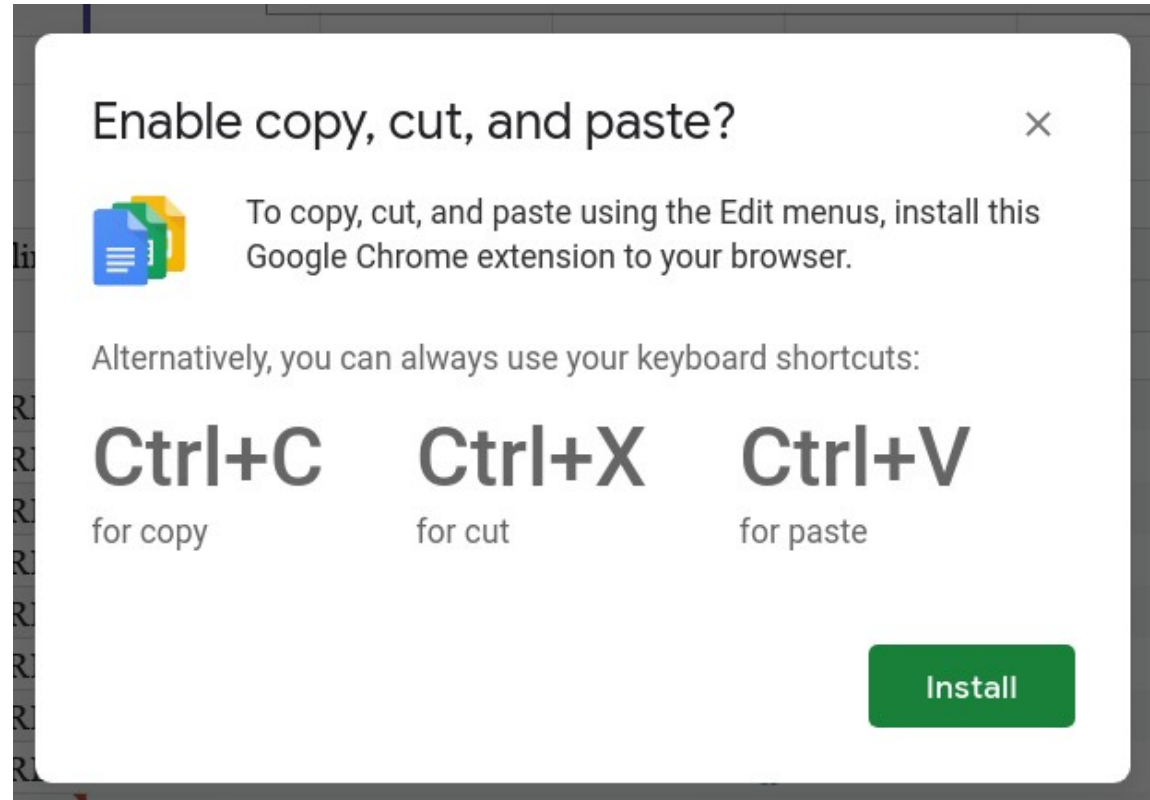
- Security team hardened by many issues → so rightly paranoid.

Google Doc's

- Users cannot copy/paste using context menus → make a plugin for that.

The Security situation ?

- Snooping is indeed bad:
 - IE11 solution was a good one.
 - Signed sites you trust ... ?





Other web security (mis-)features.

Piles of Baroque nonsense

- All schemes revolve around a hidden / transparent / off-screen / textarea / contenteditable.
- JQuery → a great idea for cross-platform support.
 - iOS: JQuery foo's click' isn't.
 - It is a wrapped touch event
 - Which doesn't have the popup security context we need.
 - Breaks only iOS
 - needs `` links

More Piles of Baroque nonsense

- IE11
 - Can only do HTML copy/paste out of a hidden content-editable
 - You have to fill / empty the HTML out of that manually in a very odd sequence.
- Secure Copy/paste types:
 - text/plain, text/html, text/rtf
 - Image/*
 - but in reality unpacked platform Bitmap
 - Getting meta-data through: text/html ...



Mobile Web clipboard problems ...

Pressing Ctrl-C is harder ...

- iOS has a nice button on the keyboard
- Android does not
 - But GBoard does
 - But GBoard converts HTML→Text and includes random <head> tags.
 - GBoard creates text input behavioural horrors of its own.

Luckily we can detect if we succeeded ?

- Not trivially:
 - return values from `execCommand` is unreliable.
 - So set incrementing global serial of successful c/c/p operations
- iOS
 - Amazing ... if security context is not perfect: everything appears to succeed.
 - You just successfully push data to a clipboard no-one else can see (not even you).



Fundamentals: Synchronous & no negotiation

Synchronous Web clipboard APIs

- Are all synchronous
 - They all assume that all the data you want to copy/paste is already in the browser.
 - Horrible for us, VDI clients, and other large / server hosted applications
- When we get Ctrl-X
 - Need all the data-ready to put to the OS clipboard.

All data at once ...

- All (sensible) OS APIs look like
 - “tell me what formats you support”
 - “later (if someone pastes) – I’ll get back to you to ask for the (few) they actually want.
 - Why ? 10^9+ cells in a spreadsheet.
- Not the web:
 - All formats must be inserted synchronously

Consider paste-as PNG of spreadsheet ranges

Improving things ...



Users: KISS – for the simple case ... and then ...

Plain text

- As you select 'simple' selections – these are pushed KIT → browser as HTML

Try to explain the sorry situation to users:

- Necessary to click-again after download to copy / cut → for security.

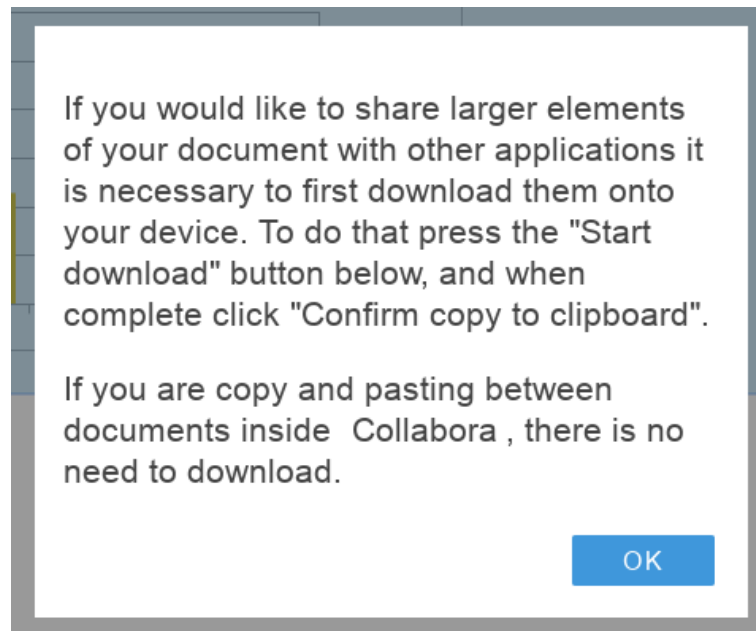
Paste to another Online instance / window ...

- Shortcut, no need to download via a custom meta-origin.



Complex cases

- Push an interesting text to the clipboard:
 - “To paste outside Online, please first click the 'download' button”***





HTML with meta-file bits ...

```
"<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
```

```
<html>
```

```
  <head>
```

```
    <title>Stub HTML Message</title>
```

```
    <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
```

```
    <meta name="origin" content="%2Flool%2Fclipboard%3FWOPISrc%3Dfile%253A%252F%252F%252Fopt%252Flibreoffice%252Fonline%252Ftest%252Fdata%252Fhello-world.ods%26ServerId%3Dc2bde695%26ViewId%3D0%26Tag%3D6da096542b9602ad"/>
```

```
  </head>
```

```
  <body lang="en_US" dir="ltr">
```

```
    <p>To paste outside Collabora Online,  
    please first click the 'download' button</p>
```

```
  </body>
```

```
</html>
```

When we get paste data

- If it has 'our' meta then:
 - If that is for the same document-id, part, view etc.
 - Short-circuit to uno:Paste
 - Job done (as before)
- Otherwise download the data



Download & re-up-load to paste

Paste of 'our' data:

- Async Download source data
(as all mime types)
- Async Up-load data
(as all mime types)
set it to Kit clipboard.
- Send an .uno:Paste to Kit.

'Download' of data:

- Async Download of (just) HTML
 - Base64 embedded images.

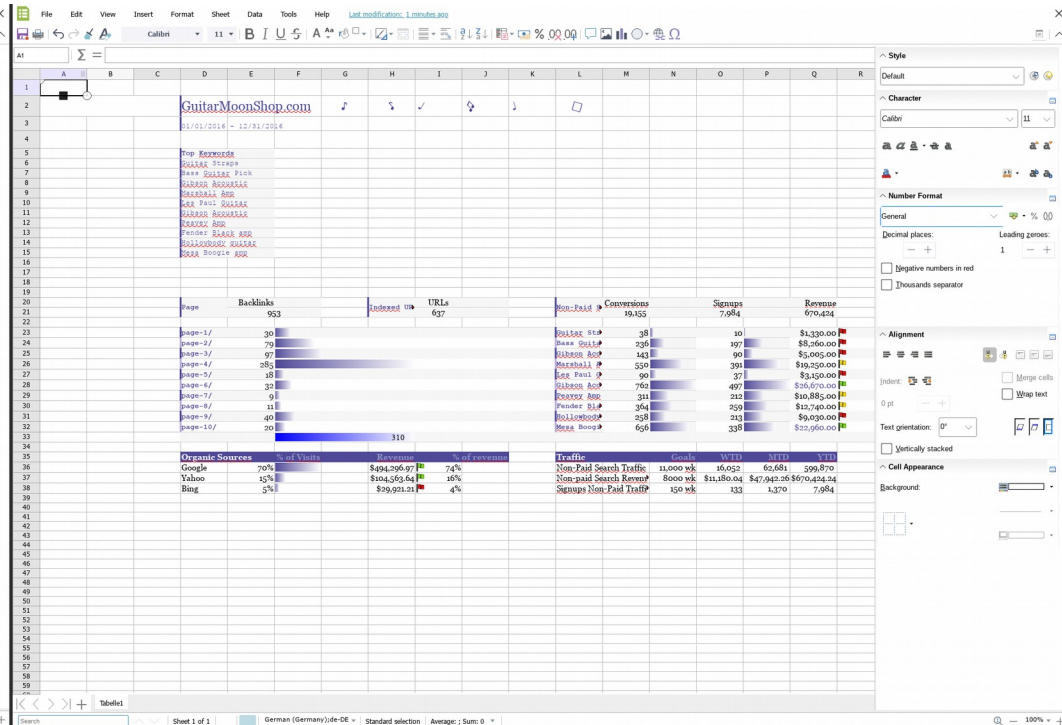
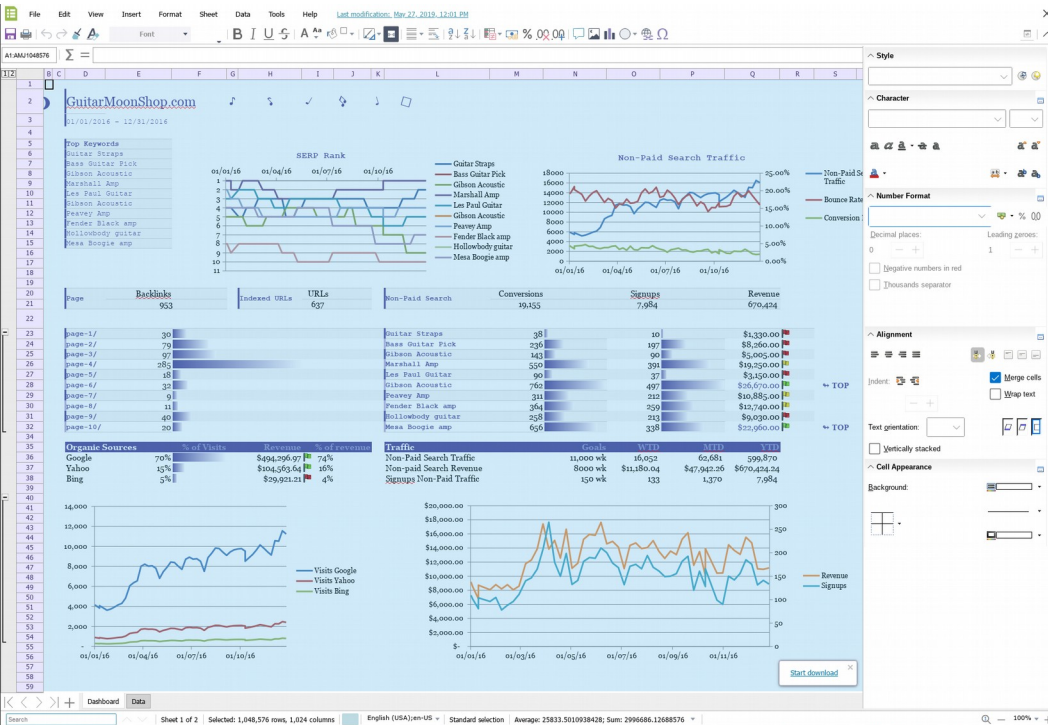
Infrastructure to make this work

- New web */clipboard/* end-point
 - Connects to the relevant Kit process to serve & set the data.
- Kills final need for 'X' libraries → safer ...

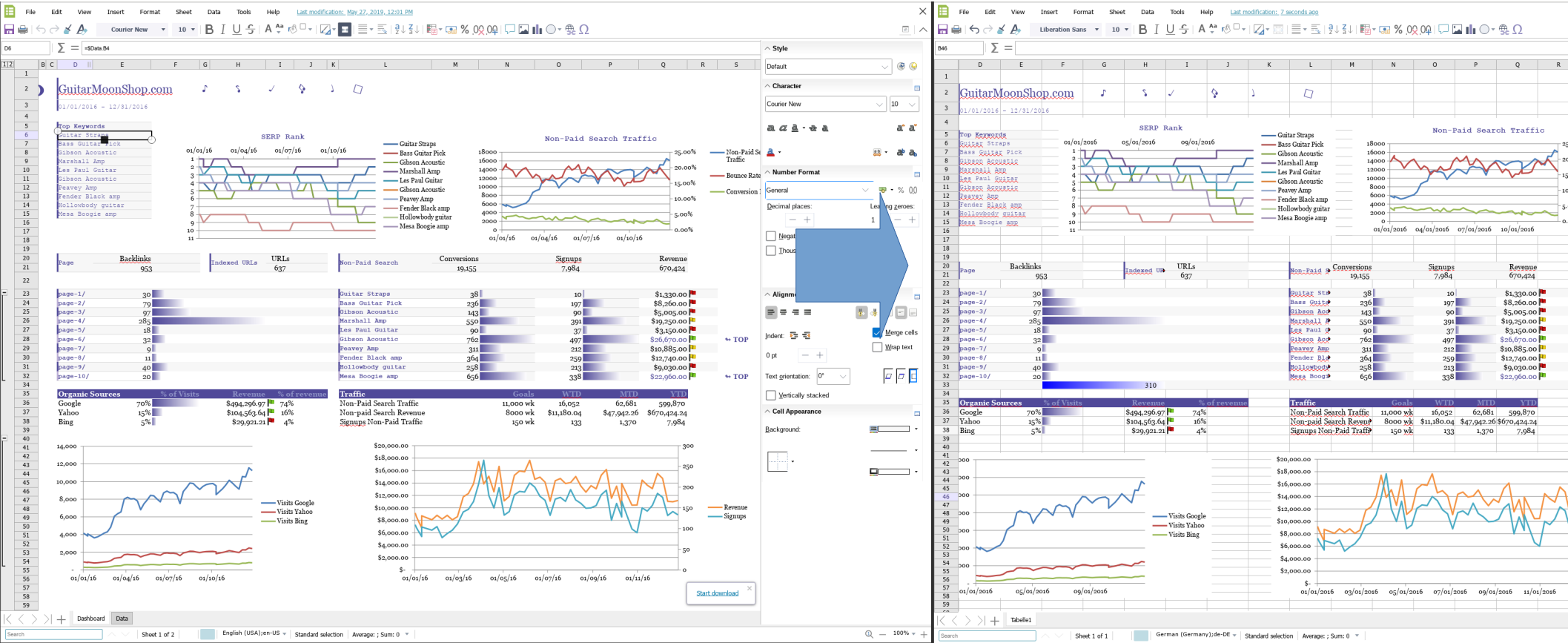
Gotchas

- Dialog paste / paste-keys
- Initially empty Kit clipboard → no 'Paste'
- Closing a Document → loose its clipboard ?
 - New synchronous Kit shutdown state machine... urk.

Afterwards ...



Afterwards + explicitly copying charts:





Other miscellaneous details

Mac / Safari

- Pasting *from* Safari – we get RTF !
 - Lovely, powerful, rich-text ...
- Unfortunately – doesn't contain images.
- Unfortunately – by design can't tell who is sending you the data...
 - So don't accept RTF on the Mac

Security

- We push a new clipboard access key every 2 minutes
 - We accept this + previous key
 - → between 2 & 4 minutes to copy/paste.
- Or you can disable copy/paste with the relevant WOPI-like property

Text/plain – regression ...

- Web can't easily convert HTML → text !?
- Will need to send text/plain too.



Conclusions

- Our codebase has a fantastic heritage & rich copy-paste code
- But connecting it was an epic – primarily fighting web api
- Online can now copy rich types – leading in its class.
- Thanks to: Ashod, Szymon, Marco, Miklos, Aron and others...
- Thanks to all of our customers & partners who make this possible.
- Questions ?

Oh, that my words were recorded, that they were written on a scroll, that they were inscribed with an iron tool on lead, or engraved in rock for ever! I know that my Redeemer lives, and that in the end he will stand upon the earth. And though this body has been destroyed yet in my flesh I will see God, I myself will see him, with my own eyes - I and not another. How my heart yearns within me. - Job 19: 23-27