Dependencies Management and Consequences (15:00 UTC)

Summary
When we add technologies to Tiki, things are usually fine and problems are or can be solved along the way and in time for the next release. However, those dependencies have their own life and changes in them will have an impact on Tiki. For example:

- Text-editing tools that were once integrated and then abandoned by the developer
- Bootstrap 3 to Bootstrap 4 upgrade impacted upgrading from Tiki18- to Tiki19+
- We are actually holding back Composer upgrade from v1 to v2. Done: https://gitlab.com/tikiwiki/tiki/-/merge_requests/785
- ...

Instead of complaining about the past, let's have an open discussion about the importance of re-thinking the process that validates the addition of such external technologies into Tiki.
This is a very interesting topic, and it can get quite broad.

1. Should we code ourselves or leverage existing code? If we pick existing code: How to pick a software library?
2. Is the component optional? Use it if you like it like Diagrams or something that affects everyone, like Bootstrap?
3. If it affects everyone, there is also the decision of when: When to make major changes

This is a little off topic but fits in here better than elsewhere i think, i've been offered funding to work on updating our wysiwyg editor to CKEditor 5 which has lots of nice new features (mainly they want drag and drop image addition, which would be nice!)

Too late for 23.0 but maybe could be done for 23.1? (or definitely 24.x)

ok, let's talk about license and text syntax like the CommonMark syntax.

The original document is available at https://tiki.org/item11952