French Connection 2017 Meetup

On 2017-10-26, Arnaud Héritier and Hervé Boutemy met one day to work on general Maven organisation improvement proposals.

Work done:

- Wiki:
  - tree structure simplification done: see Index
  - page tags started: maven-dev, plugin-dev & maven-user
  - Added proposal pages properties, to have in Proposals index page an automatic index with backlog-like classification (draft/wip/done /abandoned)
- *good lunch in a little restaurant* 🙂
- Git:
  - aggregator: looked at Sling study and WIP with Google repo (~~SLING-7164~~ - Set up the sling aggregator repository RESOLVED)
  - Jenkins: sconnolly videos in progress (see episode 1), expecting gitpubsub integration in addition to ASF Organization scanning
  - Doxia git repo history is not so clean (strange independant branches): is it a strong issue?
    - later hbo analysis: just a little mess in the ancient Doxia 1.0/1.0-alpha times, nothing serious, history is full clean for 1.x
  - preparing new Git repos for content: looked at Sling migrate-svn-to-git.sh script which splits full svn2git mono-repo to a collection of repos based on `git filter-branch --subdirectory-filter ${module_orig}`
  - idea: create a git repo for dist-tool-plugin => 2017-10-29 maven-dist-tool ready, with migration script
- jansi-native Windows improvement issue #11: let's try Apache Help Wanted to find a contributor with required knowledge
- build pom vs consumer pom:
  - think big, start small: should not require Maven core updates, just use flatten-maven-plugin with a "standard" configuration
    - this standard configuration can start easy: just remove build, reporting and profiles
    - publish simplified model documentation
    - prove consumer pom deployment as default pom, and build pom as attached artifactspecial case:
      - when packaging=pom, default pom remains build pom
    - then discuss more aggressive simplification in consumer pom (like parent flatten or any other remaining element)
  - is a key enabler to add more build features in future Maven version, with appropriate pom enhancements in `project/build` section
    - think big, start small: starting with the pom enhancements that are waiting for for many years as magic properties (like encoding, bytecode version, ...)
- build performance enhancement (aka incremental build, or build avoidance, or...)
  - need to track per mojo what are the inputs and the outputs (some configuration being a type of input), then a framework to track rebuilds
  - prerequisite: monitor performance, to get real figures of where time is going and what would be useful to optimize or not (stay simple)
    - would permit to check performance of a plugin mojo version after version
    - or check performance when changing Maven core version
    - or track what is re-done when building 2 times without modifying anything, then should be avoided
    - use Chromium trace viewer to render?