

KnowledgeTree Application Changes

Neil Blakey-Milner
“Lead” developer

Jam Warehouse Open Source Practise
<http://www.jamwarehouse.com/>

KnowledgeTree
<http://www.ktdms.com/>

2005/11/12

Background

- Funded primarily by development contract
- Delivered in four iterations

Visible changes

Bulk Import

- Existing import used ZIP files, but did not allow folder structure inside.
- New functionality allows multiple backends, currently implemented is filesystem and ZIP files

Boolean search

- Current search only allowed searching for documents that had all the given values
- New functionality allows for any/and functionality on subgroups, and any/and functionality within those subgroups.
- Backend supports infinitely nested subgroups.

Tree-like metadata

- Current metadata allowed for lookups or for free-form entry of metadata.
- New functionality allows for lookups to be categorised.

Conditional metadata

- Current metadata could not rely on other metadata fields' values to offer specific lookups.
- New functionality creates groups of fields called fieldsets.
- Conditional fieldsets allow for values of a metadata field to be controlled by one other field.

Metadata versions

- Currently, documents retain the contents of documents at previous versions.
- New functionality saves a snapshot of all document information (including metadata) when any metadata is changed or a new version is checked in.

Editable help pages

- Ability to edit the help pages through an HTML-editor via the web.

Behind the scenes

Upgrade

- Automated single-click upgrade process
- Ordered application of SQL scripts and PHP functions

Permissions (I)

- Users can now create their own permissions
- “Add new folder” is a new built-in permissions
- Backend now supports permissions by user, role, and group on documents and folder instead of just group on folder.
- Single-page permission setup for static permissions

Permissions (II)

- Permission lookups allow for any return of documents to be restricted to documents that a specified user has a specified permission on.
- Descriptor Ids are used to describe combinations of users, groups, and roles in a space-saving way.

Workflow (I)

- Workflow is no longer linear, and is specified in terms of workflow states and workflow transitions.
- Workflow transitions can be guarded to ensure that only users with the given permission on the object can perform the transition.
- Workflow transitions can also be guarded directly on users, groups, and roles.

Workflow (II)

- Workflow use permission descriptors (description of users, roles, and groups) to know who to inform on a state change.

Saved search

- Boolean searches can be saved and reused at a later stage.
- This is useful for system-wide and user-specific searches.
- Re-used to provide “conditions”, which are searches that are checked against documents before certain actions can occur.

Conditions

- Permissions use conditions to allow dynamic permissions – permissions only made available if a condition is met.
- Workflow transitions use conditions to specify a guard conditions, which must be met if the transition is to be allowed.

Partial searches

- Partial searches are searches with named parameters missing.
- Primarily of interest to system searches where the parameters will be filled programmatically.
- But useful in general too.

Robustness (I): Database

- All new code uses request-length transactions (where changes are made).
- All new database storage uses foreign key constraints to ensure no vestigial data or silly mistakes.

Robustness (II): Validation

- Validation of request parameters is now a single-line operation.
- Failed validation rolls back the transaction.
- Failed validation causes one of the following (configurable on the line):
 - Generic error page with message
 - Redirect to a specific page

Plugin-inspired rearchitecture

- Static lists of actions hand-coded in the HTML page is not scalable.
- Assumptions on how the documents are stored are not either.
- Adding an action necessitating numerous files to be edited neither.
- Having to edit actions to have them behave slightly differently, only to have this clobbered when you upgrade is infuriating.

Plugins

- Plugins should be “extract and work”
- Plugins register their changes, and configuration can disable them.
- Plugins require being able to look things up by “name”, not id, since we don't know when our stuff was added.
- Namespaced names are used everywhere to uniquely identify actions, workflows, fieldsets, triggers, and so forth.

Storage

- Storage is now an abstract interface.
- Can be stored in:
 - On disk (currently)
 - In database (next)
 - Compressed on disk (soon)
 - CVS/subversion/&c.

Import

- Import manager handles import types
- Import types:
 - Filesystem
 - Zip
 - CVS/Subversion (sometime)
 - Other DMS systems?

Actions

- Most actions are now registered with the action registry.
- Easy to create.
- Easy to configure to require permissions – both on when listed and when they are called.
- Now just takes on class rather than multiple files.

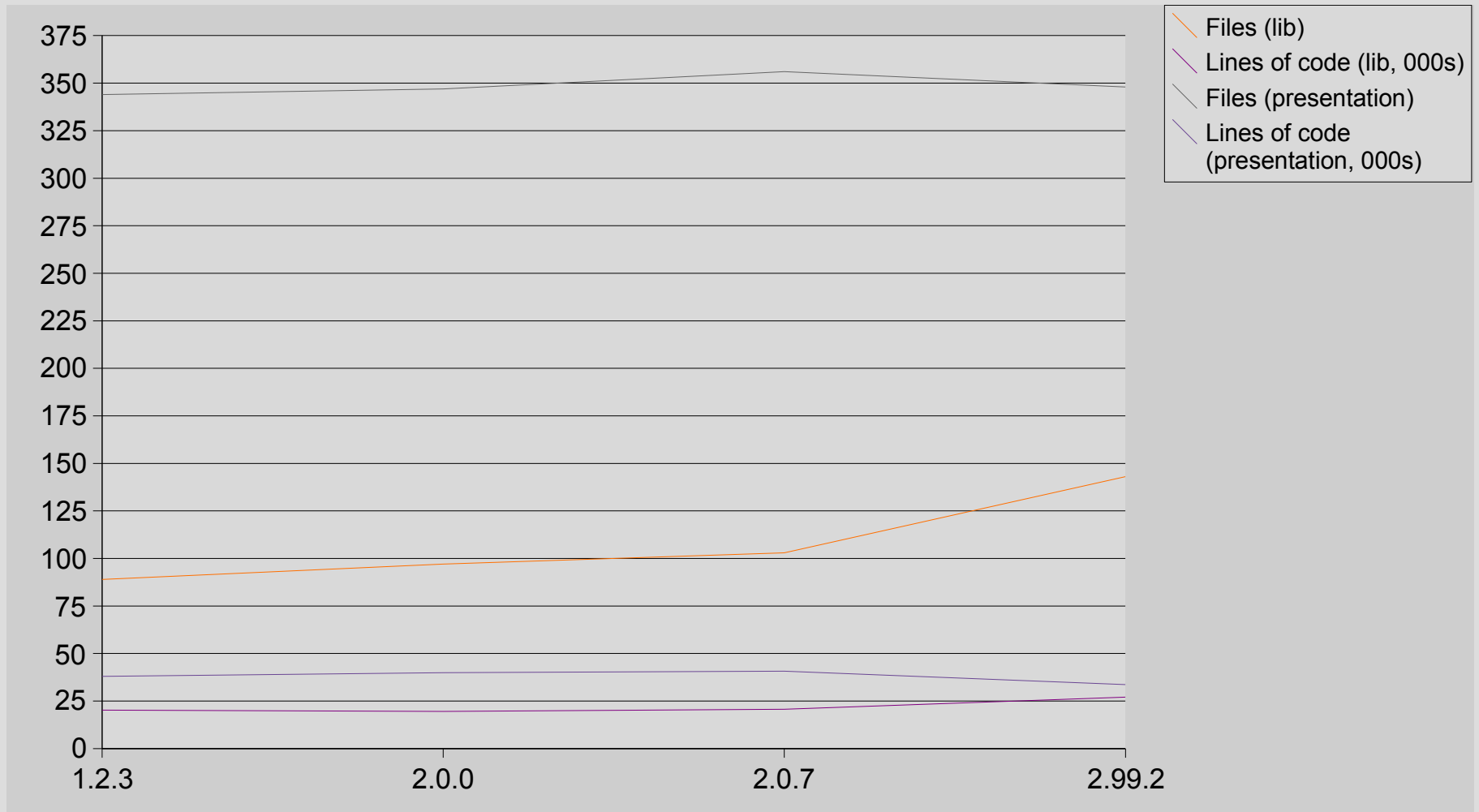
Triggers

- Allow for actions to cause other things from happening, for actions to have additional pre-conditions, and for an action's activity to be subtly altered.
- Triggers available
 - pre-action (can prevent action)
 - scan (can prevent action)
 - transform (add metadata, &c.)
 - validation (can cause rollback)
 - post-validation

Lines of code metrics (I)

	1.2.3	2.0.0	2.0.7	2.99.2
lib	89 files 20.2k lines	97 files 19.6k lines	103 files 20.7k lines	143 files 27.0k lines
presentation	344 files 38.0k lines	347 files 39.9k lines	356 files 40.7k lines	348 files 33.7k lines

Lines of code metrics (II)



Lines of code metrics (III)

- 54 lines for anti-virus scanner
- 78 lines for MP3/OGG ID3 tag extractor
- 169 lines for JPEG image metadata extractor
- 58 lines for a new action to download file contents
- 25 lines for a trigger to convert the downloaded file to PDF.