SoundSoftware Code Site - Feature #14
tags for projects

2010-10-01 05:39 PM - Chris Cannam

Status:	New	**Start date:**	2011-11-11
Priority:	Urgent	**Due date:**	
Assignee:	Luis Figueira	**% Done:**	85%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

The ability to associate free tags (e.g. "vamp") with projects would be handy.

And some ability to visualise and browse them.

There is a "tagging plugin" in the Redmine plugins list, but it's not clear whether you can use this for tagging projects themselves (the summary mentions tagging issues and wiki pages).

Subtasks:

Bug # 321: Hiding "My Projects" only useful if hidden state is persistent	**Closed**
Bug # 322: Pagination broken in projects list	**Closed**
Bug # 323: Filtering does not work in test sites	**Closed**
Bug # 324: Filter "widget" needs to be closer to project table	**Closed**
Bug # 325: Text in My Projects list too small	**Closed**
Bug # 327: Edits ignored in tag edit field while autocomplete menu visible	**New**
Bug # 331: Not apparent how to remove tag filters	**Closed**
Bug # 332: Textual search ignored if tags present in filter	**Closed**
Bug # 330: Tag autocomplete in project filter area should _not_ have Add New...	**Feedback**
Feature # 336: My projects should also list their tags	**Closed**
Bug # 337: Pagination links disappear when list refreshed after search	**Closed**
Bug # 338: Project filter yielding no results does not update results list	**Closed**
Bug # 339: Changing tags in filter box should regenerate results list immediately	**Closed**
Bug # 340: Textual search matches not highlighted in description	**Closed**
Bug # 341: Clicking on a tag does not show tag in filter area	**In Progress**
Feature # 343: Need helpful line of text under Tags field in Settings	**Closed**
Bug # 365: Tag autocomplete broken on New Project page	**Closed**
Bug # 352: Cannot add tags on New Project Page	**Closed**
Feature # 459: Warn project managers that their project is untagged!	**New**
Bug # 559: Autocomplete not working when creating a new project	**Rejected**

History

#1 - 2010-12-01 09:23 AM - Chris Cannam

- Assignee set to Luis Figueira

#2 - 2010-12-14 12:14 PM - Chris Cannam

- Priority changed from Normal to High

#3 - 2011-08-12 04:34 PM - Chris Cannam

Some notes on tag representation in the database:

http://www.pui.ch/phred/archives/2005/04/tags-database-schemas.html
http://www.pui.ch/phred/archives/2005/06/tagsystems-performance-tests.html

http://stackoverflow.com/questions/334183/what-is-the-most-efficient-way-to-store-tags-in-a-database
http://stackoverflow.com/questions/20856/how-do-you-recommend-implementing-tags-or-tagging/20871#20871

#4 - 2011-08-13 11:09 AM - Chris Cannam

The third representation mentioned in the article at http://www.pui.ch/phred/archives/2005/04/tags-database-schemas.html (referred to as the "Toxi" schema) seems probably the most appropriate.

Note that this schema, which uses a separate tag-labels table and a join table to connect tags to objects, allows you to use the same tag vocabulary with more than one sort of object (e.g. with projects and issues) if you want to.

This suggests that we should take another look at the plugins that already exist for tagging, because even if a plugin has only been designed to permit tagging issues (say), it might still turn out to have a schema suitable for tagging other objects.

#5 - 2011-08-15 12:32 PM - Chris Cannam

And indeed, https://github.com/ixti/redmine_tags/ appears to provide in the basic design for tags to be applied to objects of any model.

#6 - 2011-08-15 03:02 PM - Chris Cannam

... using http://agilewebdevelopment.com/plugins/acts_as_taggable_on

#7 - 2011-08-15 03:21 PM - Chris Cannam

Hm, one problem with a completely generic tagging implementation like this -- it has to record the taggable class in the db for every taggable-tag relation added.

e.g. I install the redmine_tags plugin and add two tags to one issue ("thin", "thick") and one tag to another ("thin"). The relevant db tables subsequently contain:

```
mysql> select * from tags;
+----+-------+
| id | name  |
+----+-------+
|  1 | thick |
|  2 | thin  |
+----+-------+
2 rows in set (0.00 sec)

mysql> select * from taggings;
+----+--------+------------+-----------+-------------+--------------+---------+---------------------+
| id | tag_id | taggable_id | tagger_id | tagger_type | taggable_type | context | created_at          |
+----+--------+------------+-----------+-------------+--------------+---------+---------------------+
|  1 |      1 |          4 |      NULL | NULL        | Issue        | tags    | 2011-08-15 15:10:28 |
|  2 |      2 |          4 |      NULL | NULL        | Issue        | tags    | 2011-08-15 15:10:28 |
|  3 |      2 |          3 |      NULL | NULL        | Issue        | tags    | 2011-08-15 15:16:22 |
+----+--------+------------+-----------+-------------+--------------+---------+---------------------+
3 rows in set (0.00 sec)
```

That's a hell of a lot of data for three tags, far more than the alternative model in which we have a dedicated join table for each taggable type. That model would give us e.g. an issue_tags table containing something like

```
+----+----------+--------+
| id | issue_id | tag_id |
+----+----------+--------+
|  1 |        4 |      1 |
|  2 |        4 |      2 |
|  3 |        3 |      2 |
+----+----------+--------+
```

which is far smaller than the above, but which would require a separate table for each taggable type (i.e. statically limited set of taggables).