Redmine - Feature #2180

Lookup custom fields

2008-11-13 19:37 - Jorge L. Cangas

Status:	New	**Start date:**	2008-11-13
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Custom fields	**Estimated time:**	0.00 hour
Target version:			
Resolution:			

Description

-I need a custom field of type 'lookup', i.e., it can reference some other table. So I would add a 'customer' lookup field to the issues in order to link an issue with a customer. Of course, customer plugin will be instaled previously in order I can reference it.

- When I define the lookup field, I name it something like 'customer_id', so Redmine shows me the allowed referenced table in a list box, I select 'customer'. Then I select a 'display column' for it, and I select 'customer.name'

- When I'm will fill this customer field in an issue, I can write the customer name, with autocomplete, of course :), and Redmine stores de customer_id in the issue record and shows me the customer name in the issue form.

I think this feature opens a lot of posibilities for Redmine be more 'smart' when we use plugins

Related issues:

Related to Redmine - Feature #2096: Custom fields referencing system tables (...	**Closed**	**2008-10-27**

History

#1 - 2008-11-13 20:24 - Eric Davis

You can just add a customer_id field to issues via your plugin and then associate issues belongs_to customer via vanilla Rails.

See my Budget Plugins' issue monkey patch to see how I related an Issue to a Deliverable (plugin's model).

#2 - 2008-11-17 11:23 - Jorge L. Cangas

Eric Davis wrote:

> You can just add a customer_id field to issues via your plugin and then associate issues belongs_to customer via vanilla Rails.
>
> See my Budget Plugins' issue monkey patch to see how I related an Issue to a Deliverable (plugin's model).

I'm sorry, maybe I don't tell it well: Is not **my** plugin. Is about any plugin. I pretend to use some plugin in a new way, as a user. Your hint is ok for me as developer (I'm) but in my request I'm thinking more as a final user of redmine: he don't want 'descend' to the code :). By the way, thanks for the hint.

#3 - 2010-08-04 01:44 - Terence Mill

The idea of referencing custom field's from tables bia lookup fieled is nice. How is it implemented at the moment for value lists? Wouldn't you like to have a key-value list, where key is the reference and lookup column and value the displayed value to the user?
This way you could change order, rename items and insert values not only at the list end and will keep the reference and integrity of entities (tickets,..) referencing the display value via the key value.

The table being used for custom fields declaration could look like this:

- Column 1: Custom field name/id like seen in ticket label or another entity (projects, etc.)
- Column 2: Custom field type (number,string,date, regex, etc.)
- Column 3: Field value display name - the name which is shown to use as value name
- Column 4: Field Key name - the value's identity which is referenced as foreign key when selected as value in entities (tickets, projects..etc)

In this table you could insert as many custom fields with key-value lists which can be changed with little refactoring capabilities.

In another step restful xml ws or wiki pages can be used as source for providing and maintaining custom field input via table structure.

#4 - 2010-08-26 00:34 - Eric Davis

- Priority changed from Urgent to Normal

#5 - 2011-03-23 11:37 - Toshi MARUYAMA

- Category set to Custom fields

#6 - 2013-03-25 22:19 - Dipan Mehta

+100. This is very important as we are evaluating to extend Redmine for helpdesk.

The Request Tracker ticketing system has something called an External Fileds .

A potential use case could be to hook to external systems - e.g.

1. Lookup on the external Inventory systems to know the list of machines that can apply for the given context,
2. List of versions of external platforms -e.g. Ruby, or Chrome or even list of stable Linux versions,
3. List external system Bug numbers, Test cases of external systems etc.
4. Most importantly, this can link different configuration option options available in the context.

How should this be implemented?

Terence Mill wrote:

> The idea of referencing custom field's from tables bia lookup fieled is nice. How is it implemented at the moment for value lists? Wouldn't you like to have a key-value list, where key is the reference and lookup column and value the displayed value to the user?
> This way you could change order, rename items and insert values not only at the list end and will keep the reference and integrity of entities (tickets,..) referencing the display value via the key value.

This is absolutely perfect.

> The table being used for custom fields declaration could look like this:
>
> - Column 1: Custom field name/id like seen in ticket label or another entity (projects, etc.)
> - Column 2: Custom field type (number,string,date, regex, etc.)
> - Column 3: Field value display name - the name which is shown to use as value name
> - Column 4: Field Key name - the value's identity which is referenced as foreign key when selected as value in entities (tickets, projects..etc)
>
> In this table you could insert as many custom fields with key-value lists which can be changed with little refactoring capabilities.
>
> In another step restful xml ws or wiki pages can be used as source for providing and maintaining custom field input via table structure.

Again right on!

I think calling upon external DB of the source directly would be very wrong so would be for external application
to touch Redmine DB directly. The best possible way would be to expose this custom-field through the REST API to add, insert and modify the list of possible values.

Now one can write any script on ones' own that fetches the right set of list from whatever appropriate place and run a cron-type job to keep updating this!

One of the other hurdle is, that if the old value (once valid) is already referenced in one of the issue, but it is no longer valid for new version, such deprecated values needs to be in the DB but there needs to be a way to mark such values as deprecated ones.

This is really a powerful idea - must pursue.

#7 - 2013-04-10 14:17 - Dipan Mehta

Add related #9734, #6717

#8 - 2021-06-09 10:31 - Alexandr Chernyaev

+1