Mass spelling correction

Related:

- Templates Best Practices
- Strings Format Convention

Short version

You just modified an english string in a template or php file in Tiki trunk. The meaning has not changed, so translations should be kept unmodified. Before, the string was "Search engine friendly url Postfilter" and now it is "Search engine friendly URL Postfilter" in tiki-somefile.php

A quick fix in to use the English translation file at http://tikiwiki.svn.sourceforge.net/viewvc/tikiwiki/trunk/lang/en/language.php?view=markup but eventually this needs to be cleaned up with the procedure on this page.

See also http://tikiwiki.svn.sourceforge.net/viewvc/tikiwiki/trunk/doc/devtools/update_english_strings.php?view=markup

Access the root of your Tiki version in your svn tree

cd (blabla)/trunk

1.1.2. Get an up to date language source tree

svn up lang

1.1.3. Execute command

```
perl doc/devtools/mass_wording_corrections.pl 'Search engine friendly url Postfilter'
'Search engine friendly URL Postfilter'
```

Optionally, see what changed in the backup file
diff lang/fr/language.php lang/fr/language.php.bak

1.1.4. If something is wrong, roll back to svn version

rm lang/*/language.php*
svn up lang

1.1.5. Repeat

There may have been other such strings you have corrected

1.1.6. Commit and clean up backups

svn commit tiki-somefile.php lang -m "[ENH] Improved string"
rm lang/*/language.php.back

That's it!

Every technical detail you ever wanted to know, and much more

Due to popular demand (hi Marc!), here is a description of how I did some mass word replacement (in order to properly capitalize buttons).

I will break it down so you understand the process, because it's shown as inspiration, not as a procedure (I expect that all mass capitalisation will be taken care of for the time being).

First, I go to the directory of my cvs repository.

How to mass rename a word in the smarty templates ?

That's the easy part. First, suppose you want to replace all instances of "discuss" into "Discuss". These are in the ".tpl" files in the "templates/" directory, and they are translated, so they appear as "{tr}discuss{/tr}"

How many files are we talking about ?

```
find templates -name "*.tpl" | xargs grep -l "{tr}discuss{/tr}" | wc -l
 7
```

That's not a lot. Let's see the list.

```
find templates -name "*.tpl" | xargs grep -l "{tr}discuss{/tr}"
 templates/tiki-page_bar.tpl
 templates/styles/gemsi/tiki-page_bar.tpl
 templates/styles/moreneat/tiki-show_page.tpl
 templates/styles/neat/tiki-editpage.tpl
 templates/styles/neat/tiki-show_page.tpl
 templates/styles/tiki/tiki-show_page.tpl
 templates/styles/vidiki/tiki-page_bar.tpl
```

Let's replace all "discuss" by "Discuss"

```
find templates -name "*.tpl" | xargs grep -l "{tr}discuss{/tr}" | xargs perl -pi.bak -e
```

```
"s/{tr}discuss{\/tr}/{tr}Discuss{\/tr}/g"
```

I am a cautious and curious person. This did not just replace the strings. It also backuped all modified files with a ".bak" extension. Now I can check what happened.

```
diff templates/tiki-page_bar.tpl templates/tiki-page_bar.tpl.bak
 81c81
 < <span class="button2"><a href="tiki-
view_forum.php?forumId={$wiki_forum_id}&comments_postComment=post&comments_title={$page|esca
pe:"url"}&comments_data={$wiki_discussion_string|escape:"url"}: {"[tiki-
index.php?page="}{$page|escape:"url"}{"|"}{$page|escape:"url"}{"]"}&comment_topictype=n"
class="linkbut">{tr}Discuss{/tr}</a></span>
 ---
 > <span class="button2"><a href="tiki-
view_forum.php?forumId={$wiki_forum_id}&comments_postComment=post&comments_title={$page|esca
pe:"url"}&comments_data={$wiki_discussion_string|escape:"url"}: {"[tiki-
index.php?page="}{$page|escape:"url"}{"|"}{$page|escape:"url"}{"]"}&comment_topictype=n"
class="linkbut">{tr}discuss{/tr}</a></span>
```

Looks good...
I am satisfied that all went well: I remove the backup files (actually, I don't need to, cvs will ignore them)

```
find templates -name "*.bak" | xargs \rm
```

How to avoid breaking the translations

I found no one-liner to do that, so have written a perl script. It just expects two paramaters: the original uncapitalized word "discuss" and the new capitalization, "Discuss". I will not go inside it and describe all it does.
If I type :

```
capitalize_buttons.pl -v Discuss discuss
```

it will verbosely look if there is already a translation for "Discuss" (no luck), so it looks for the translation for "discuss" and creates a new line with a translation for "Discuss", which will be the same as the one for "Discuss", with an Uppercase on the first letter (unless the alphabet does not look like perl knows how to uppercase it, then I just leave as is).
Again, it keeps a backup copy so we can check what changed:

```
diff lang/fr/language.php.bak lang/fr/language.php
2514a2515
> "Discuss" => "Discussion",
```

Right now it's in CVS: doc/devtools/capitalize_buttons.pl
And also the last version I use is in the bottom attachments.

Type :

```
capitalize_buttons.pl --help
```

to have it display how to use it.

What about the "mass" part?

Well, we can just use all of the above and make a bash script of it, which does all in a row. Like this:

```
#!/bin/bash

if [ $# -ne 2 ]
then
  echo "Usage: $0 word Word"
  echo "Mass-replaces word with Word"
  exit 1
fi



oldword=$1
newword=$2
oldwordescaped=`echo $oldword | sed -e 's/\//\\\\\\\//g'`
newwordescaped=`echo $newword | sed -e 's/\//\\\\\\\//g'`

echo -n "files found: "
find templates -name "*.tpl" | xargs grep -l "{tr}$oldwordescaped{/tr}" | wc -l
echo "replace '$1' with '$2' ? (return/l/Ctrl-C)"
read toto

if [ "$toto" = "l" ]
then
  find templates -name "*.tpl" | xargs grep "{tr}$oldwordescaped{/tr}"
  echo "replace '$1' with '$2' ? (return/Ctrl-C)"
  read toto

fi

echo "OK"
find templates -name "*.tpl" | xargs grep -l "{tr}$oldwordescaped{/tr}" | xargs perl -pi.bak
-e "s/{tr}$oldwordescaped{\/tr}/{tr}$newwordescaped{\/tr}/g"

echo "language files ? (return/Ctrl-C)"
read toto
capitalize_buttons.pl -v "$1" "$2"
```

The latest version I use is in the bottom attachments. It is also longer and more complicated to read, and less suitable as explanation support material.

I call it "jml_translate_buttons-v03.bash" and I use it like this:

```
jml_translate_buttons-v03.bash stats Stats
files found: 5
```

```
replace 'stats' with 'Stats' ? (return/Ctrl-C)

OK
language files ? (return/Ctrl-C)

stats --> Stats (given) Stats (auto)
lang/ar/language.php (ar)
-> grep '"Stats"[       ]*=>[   ]*"[^"]*"[      ]*,.*$' lang/ar/language.php | wc -l >
/tmp/result.txt
Nothing to do: translation of Stats is there already (lines found: 1)
[plenty of other lines]
```

What about buttons with more than one word?

It works just the same. The automatic capitalization in language files will only try to Uppercase the first character of the translation.

```
jml_translate_buttons-v03.bash "cancel edit" "Cancel Edit"
files found: 5
replace 'cancel edit' with 'Cancel Edit' ? (return/Ctrl-C)

OK
language files ? (return/Ctrl-C)

cancel edit --> Cancel Edit (given) Cancel edit (auto)
lang/ar/language.php (ar)
[plenty of other lines]
```

What about words translated in php files ?

Well, the above totally ignores the words translated in php files, with "tra('word')". I subsequently wrote jml_translate_buttons-v12.bash. It does all jml_translate_buttons-v03.bash and more, which I let you figure out 🙂

```
#!/bin/bash

language_file_translation_script="jml_translate_buttons-v07.pl"

if [ $# -ne 2 ]
then
  echo "Usage: $0 word Word"
  echo "Mass-replaces word with Word"
  exit 1
fi

if [ `which ${language_file_translation_script}` ];
then
```

```
  echo "${language_file_translation_script} is available. Good."
else
  echo "Error: ${language_file_translation_script} is not available in your path."
  exit
fi



oldword=$1
newword=$2
# The strange 'X's serve to deal with trailing and starting spaces
oldwordescaped=`echo X${oldword}X | sed -e 's/\//\\\\\\\//g' `
oldwordescaped=`echo $oldwordescaped | sed -e 's/\#/\\\\\\\#/g' -e 's/X$//' -e 's/^X//'`
oldwordescaped_perl=`echo X${oldwordescaped}X | sed -e 's/(/\\\\(/g'`
oldwordescaped_perl=`echo $oldwordescaped_perl | sed -e 's/)/\\\\)/g' -e 's/X$//' -e
's/^X//'`
newwordescaped=`echo X${newword}X | sed -e 's/\//\\\\\\\//g' -e 's/X$//' -e 's/^X//'`



echo -n "php files found: "
find . -name "*.php" | xargs grep -l "tra(\(['\"]\)$oldwordescaped\1" | wc -l
echo "replace '$1' with '$2' ? (return/(l)ist/(n)o/Ctrl-C)"
read toto

if [ "$toto" = "l" ]
then
  find . -name "*.php" | xargs grep "tra(\(['\"]\)$oldwordescaped\1"
  echo "replace '$1' with '$2' in php files ? (return/(n)o/Ctrl-C)"
  read toto
fi

if [ "$toto" != "n" ]
then
  find . -name "*.php" | xargs grep -l "tra(\(['\"]\)$oldwordescaped\1" | xargs perl -pi.bak
-e "s/tra\((['\"])$oldwordescaped_perl\1/tra\(\1$newwordescaped\1/g"
fi

echo -n "template files found: "
find templates -name "*.tpl" | xargs grep -l "{tr}$oldwordescaped{/tr}" | wc -l
echo "replace '$1' with '$2' ? (return/(l)ist/(n)o/Ctrl-C)"
read toto

if [ "$toto" = "l" ]
then
  find templates -name "*.tpl" | xargs grep "{tr}$oldwordescaped{/tr}"
  echo "replace '$1' with '$2' in template files ? (return/(n)o/Ctrl-C)"
  read toto
fi

echo "OK"
find templates -name "*.tpl" | xargs grep -l "{tr}$oldwordescaped{/tr}" | xargs perl -pi.bak
-e "s/{tr}$oldwordescaped_perl{\/tr}/{tr}$newwordescaped{\/tr}/g"

echo "insert '$2' translation in language files ? (return/(n)o/Ctrl-C)"
read toto
```

```
if [ "$toto" != "n" ]
then
  ${language_file_translation_script} -i "$1" "$2"
fi
```

What next?

Now we can translate the string "Discuss:" by separating the translation of "Discuss" and ":". This actually makes sense, as it automatically handles the fact that in french, ":" -> " :", and also we can have a single translation for "Discuss:" and "Discuss":
But my script is currently unable to find "Discuss" in the language file when it is given "Discuss:" as input.

I'm on Windows !

Well, you might put your requests for modifications in a table with "text currently in TW" corresponding to "corrected text I'd like" on page Pending text corrections and wait for someone to process them.
Of course, as the process is automated, the text in "text currently in TW" must be an exact cut/paste of the text in language.php