# Un tr() oublié ?

*Qt by Nokia*

Par Jasmin Blanchette - traducteur : Thibaut Cuvelier - Qt Quarterly

Date de publication : 18 février 2009

Dernière mise à jour : 1 août 2011

Le mécanisme de Qt pour l'internationalisation (**tr()**) est facile à comprendre, facile à utiliser, et facile à mal employer. Cet article donne quelques astuces pour s'assurer que toutes les chaînes d'une application, visibles pour l'utilisateur, passent à la moulinette **tr()**, et que **lupdate** les trouve. Les développeurs monolingues ne sont pas laissés sur le banc : cet article traite aussi du suédois.

Cet article est une traduction autorisée de **Forgot a tr()?**, par Jasmin Blanchette.

**Commentez**

I - L'article original..... 3
II - Habituels blunders..... 3
I-A - Oublier un tr()..... 3
I-B - Utiliser tr() sur une variable..... 3
I-C - Utiliser QT_TR_NOOP() sans tr()..... 4
I-D - Utiliser tr() sur un objet..... 4
I-E - Oublier un Q_OBJECT..... 4
I-F - Préparer le QTranslator trop tard..... 4
I-G - Fonction virtuelle ou macro ?..... 4
III - Grep, première solution..... 5
IV - Moquons-nous du suédois..... 5
V - Là où tr() ne suffit plus..... 7
VI - Divers..... 7


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

# I - L'article original

Qt Quarterly est une revue trimestrielle électronique proposée par Nokia à destination des développeurs et utilisateurs de Qt. Vous pouvez trouver les **versions originales**.

Nokia, Qt, Qt Quarterly et leurs logos sont des marques déposées de *Nokia Corporation* en Finlande et/ou dans les autres pays. Les autres marques déposées sont détenues par leurs propriétaires respectifs.

Cet article est la traduction de l'article **Forgot a tr()?** de Jasmin Blanchette paru dans la Qt Quarterly Issue 3.

Cet article est une traduction d'un des tutoriels écrits par **Nokia Corporation and/or its subsidiary(-ies)** inclus dans la documentation de Qt, en anglais. Les éventuels problèmes résultant d'une mauvaise traduction ne sont pas imputables à Nokia.

# II - Habituels blunders

[BLUNDER:] From Middle English *BLUNDEREN*, **to go blindly**, perhaps from Old Swedish *BLUNDRA*, **have one's eyes closed**, from Old Norse *BLUNDA*.
American Heritage Dictionary
[BLUNDER :] De l'anglais moyen, *BLUNDEREN*, **s'avancer en aveugle**, peut-être du vieux suédois *BLUNDRA*, **avoir les yeux fermés**, du vieux norrois *BLUNDA*.

La fonction **tr()** est double par nature : en effet, il s'agit, et d'un marqueur pour **lupdate**, et d'une fonction C++. Oublier cette dualité, c'est ouvrir la porte aux bourdes. Quelques-unes se manifestent lors de la compilation par des erreurs, d'autres sont rattrapées par **lupdate**, et le peu qui reste n'est pas détecté, et résulte en un message sur notre traqueur de bogues.

C'est cette dernière catégorie qui va nous intéresser. Nous commencerons par regarder d'où peut venir le problème, et nous y présenterons des solutions.

Il faut impérativement que tous ces problèmes soient corrigés bien avant que les fichiers *.ts* soient envoyés en traduction.

## I-A - Oublier un tr()

Il est facile d'oublier d'enfermer toutes les chaînes visibles à l'utilisateur dans un **tr()**. Ni **lupdate**, ni le compilateur C ++ ne peuvent distinguer les chaînes visibles à l'utilisateur des autres. Seul **QPainter** peut le savoir, mais il est alors trop tard : "*Cannot open %1*" n'est pas reconnaissable en tant que "*Cannot open "C:\index.html"*"

## I-B - Utiliser tr() sur une variable

Les arguments de **tr()** doivent être littéraux, comme ceci :

```
QPushButton *ok = new QPushButton( tr("OK"), this );
```

Au contraire, ceci ne fonctionnera jamais :

```
QString str = "OK";
QPushButton *ok = new QPushButton( tr(str), this );
```

Il est sûr que votre chaîne passera par **tr()**, mais pas par **lupdate**, auquel cas votre traducteur japonais ne la verra pas. Même si "*OK*" convient pour une majorité des langues, cela n'ira pas en japonais.


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

Un exemple plus sophistiqué peut utiliser des objets temporaires :

```
tr( QString("Cannot open %1").arg(fileName) )
tr( "Cannot open " + fileName )
```

Ce code ne fonctionnera pas. Tout à l'inverse de celui-ci :

```
tr( "Cannot open %1" ).arg( fileName )
```

## I-C - Utiliser QT_TR_NOOP() sans tr().

La macro *QT_TR_NOOP()*, et sa cousine de longueur plus respectable *QT_TRANSLATE_NOOP()*, sont très rarement utilisés, mais, quand elles le sont, elles ne le sont pas correctement. À l'inverse de **tr()** et de **translate**, ces macros n'effectuent aucune traduction, et marquent simplement les chaînes pour la traduction.

```
const char *strings[2] =
{
    QT_TR_NOOP("OK"),
    QT_TR_NOOP("Cancel")
};
for ( int i = 0; i < 2; i++ )
{
    QButton *but = new QPushButton( strings[i], this );
}
```

Le seul résultat de ce code est une charge supplémentaire de travail pour votre traducteur japonais, et un faux sentiment de sécurité. La meilleure solution est d'éviter ces "marrantes" macros

## I-D - Utiliser tr() sur un objet

**tr()** est une fonction membre statique. La façon la plus normale de l'utiliser hors de *myWidget* est *myWidget::tr()*. Le compilateur va laisser *myWidget->tr()* passer, mais **lupdate** n'est pas assez intelligent pour déduire le type de *myWidget*. Depuis Qt 3.1, **lupdate** émet un avertissement en ce cas ("*Cannot invoke tr() like this*").

## I-E - Oublier un Q_OBJECT

Très souvent, quelqu'un vient jusqu'à mon bureau (quel effort), et dit que **tr()** ne fonctionne pas. Je perds une demi-heure à essayer d'isoler le problème, à recompiler Qt en mode débug, pour découvrir, finalement, que l'appel à **tr()** a été effectué dans une classe sans la macro *Q_OBJECT*, nécessaire pour les signaux, les slots, les propriétés, et **tr()**. Sans elle, **tr()** utilise le nom de la classe de base comme contexte, alors que **lupdate** utilise toujours le nom de la classe en question. Les dernières versions de **lupdate** reconnaissent ce problème, et émettent un avertissement (*"Class MyClass lacks Q_OBJECT macro"*).

## I-F - Préparer le QTranslator trop tard

Très souvent, quelqu'un vient jusqu'à mon bureau (quel effort), et dit que **tr()** ne fonctionne pas. Je perds une demi-heure à essayer d'isoler le problème, à recompiler Qt en mode débug, pour découvrir, finalement, que l'objet **QTranslator** était installé dans l'objet *qApp* **après** que la fenêtre principale de l'application soit créée ! Soyons très prudents.

## I-G - Fonction virtuelle ou macro ?



*La croyance que **tr()** est une macro, ou une fonction membre virtuelle, est très répandue. La vérité est que **tr()** est une fonction statique, et, donc, non virtuelle, et certainement pas une*


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

*macro. Chaque classe utilisant la macro Q_OBJECT a les fonctions **tr()** et **trUtf8()** déclarées et implémentées automatiquement.*

Vous pourriez vous demander pourquoi **tr()** n'est pas implémentée une bonne fois pour toutes dans **QObject**, en utilisant **className()** pour fournir un contexte. La raison est la suivante :
Soient A une classe dérivée de **QObject**, et B, un dérivé de A. Supposons que le code source de A contienne ceci :

```
setText( tr("Hello") );
```

La chaîne "*Hello*" est, pour **lupdate** dans le contexte A. Cependant, quand ce code est exécuté pour une instance de B, **className()** retourne B, c'est pourquoi **tr()** essayerait de la traduire dans le contexte B, ce qui est tout à fait erroné.

## III - Grep, première solution

Le premier outil à utiliser pour trouver les erreurs 1, 2 et 3 est *grep*. Les programmeurs Windows peuvent aussi essayer *findstr*, ou un des nombreux ports de *grep*.

La commande suivante va trouver les chaînes qui ne sont pas entourées par un **tr()**.

```
grep -n '"' *.cpp | grep -v 'tr('
```

L'option *-n* demande ) *grep* d'afficher les numéros de ligne. L'option *-v* inverse la sélection. Ainsi, la commande précédente pourrait se lire :
"*Trouver toutes les lignes qui contiennent ´"´, mais pas ´tr()´*".
Cette approche trouve réellement tous les appels manquants à **tr()**, mais aussi de trop nombreuses chaînes innocentes, qui n'ont aucun besoin de traduction, comme les noms de widgets. Nous verrons bientôt une meilleure méthode pour mettre en évidence ces lacunes.

Le problème 2, utiliser **tr()** sur une variable, est plus facile à mettre en évidence :

```
grep -n 'tr(' *.cpp | grep -v '"'
```

Elle est l'opposé de la précédente :
"*Trouver toutes les lignes qui contiennent ´tr(´ mais pas ´"´*".

Si l'application utilise *qApp->translate()*, utilisez ces commandes, en remplaçant **tr()** par **translate**.

L'utilisation malencontreuse de QT_TR_NOOP peut être facilement évitée en regardant le code. Cette commande trouve tous les fichiers à regarder de plus près :

```
grep -l 'QT_TR' *.cpp
```

Pour de petits projets, *grep* est souvent suffisant. Des projets plus grands bénéficieront de la méthode dite "du chef suédois", présentée ci-dessous.

## IV - Moquons-nous du suédois

Belbo orders Abu to change all words, make each "a" become "akka" and each "o" become "ulla", for a paragraph to look almost Finnish.
Umberto Eco
Belbo ordonne à Abu de changer tous les mots, en remplaçant tous les "a" par "akka", et tous les "o" par "ulla", pour que le paragraphe ressemble à du finnois.


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

La méthode la plus pratique pour mettre en évidence les appels manquants à **tr()** est de lancer l'application avec une traduction. Mais que faire si aucune n'est disponible ?

La réponse à ce problème prend la forme d'un personnage, le chef suédois, du Muppet Show. L'idée est d'utiliser un programme pour traduire l'application originale (anglaise ou française) en la langue parlée par le chef suédois, ce que nous appelons le *Mock Swedish* (moquons-nous du suédois). L'application est alors lancée avec cette traduction, et tous les mots anglais qui apparaissent non traduits à ce point sont causés par une mauvaise utilisation de **tr()**.

Voici des exemples de traduction de l'anglais ou du français et leurs équivalents.

| Anglais | Français | Suédois |
|---|---|---|
| &File | &Fichier | &Feele-a |
| Big pink pig | Gros cochon rose | Beeg peenk peeg |
| Qt Quarterly | Qt trimestriellement | Qt Qooerterly |
| Internationalization | Internationalisation | Interneshuneleezeshun |
| Do you want to save %1? | Voulez-vous sauvegarder %1 ? | Du yuoo vunt tu sefe-a %1 ? Bork Bork Bork ! |

L'avantage du chef suédois est que le résultat a un air indiscutablement étranger, ce qui rend la tâche facile de repérer les chaînes non traduites, et, cependant, l'application est toujours utilisable par les testeurs, qui peuvent deviner le sens du suédois.

Il n'y a, en fait, aucune raison technique de préférer un chef suédois à un vendeur de hot-dogs allemand ou un sushi-eur du Japon. Toute langue à l'aspect suffisamment étranger et encore lisible peut être utilisée.

Voici un résumé des différentes modifications.

| Anglais | Français | Suédois |
|---|---|---|
| -an | -en | -un |
| a- | a- | e- |
| -ew ; -ow | -uif | -oo |
| en- | en- ; in- | ee- |
| f | f | ff |
| -i | -ien | -ee |
| -o | -o ; -on | -oo |
| -tion | -tion | -shun |
| -ou | -ou ; -oi | -uu |
| v | v | f |
| w | w | v |
| j | j | i |

Si un mot ne subit pas de transformation radicale, il est toujours possible de lui ajouter un ou plusieurs "bork", ou bien l'écrire en verlan, ou bien de droite à gauche, ou bien en inversant la casse, ou bien en retirant les voyelles, ou bien les consonnes ... Les variations sont infinies.

*Ce type de suédois ne souffre aucune ressemblance avec du vrai suédois, à moins que le texte d'entrée soit du danois, du norrois ou du suédois.*


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

## V - Là où tr() ne suffit plus

**lupdate** supporte les fichiers sources en C++ et les fichiers *.ui* de Qt Designer. Quelques applications stockent leurs chaînes, visibles pour l'utilisateur, dans d'autres endroits : une base de données, des fichiers de configuration, des scripts, etc. Il y a quelques approches qui peuvent être utilisées pour intégrer ces ressources dans un cycle de **lupdate**, **Qt Linguist**, **lrelease**.

La manière la plus simple est un programme qui extrait les chaînes d'où elles peuvent venir et génère un fichier *.ts*, qui pourra ensuite être traduit avec Qt Linguist et converti en *.qm*. **QApplication** supporte plusieurs *.qm* simultanés, il n'y a donc aucun problème à combiner cette technique avec les autres. Son problème est qu'elle dépasse l'algorithme de fusion utilisé par **lupdate**.

Une alternative est d'écrire un programme qui génère du faux code C++, qui peut ensuite être lu par **lupdate**. Voici un exemple d'un tel code.

```
#if 0
qApp->translate( "CustomerDB", "Agency" );
qApp->translate( "CustomerDB", "Company" );
qApp->translate( "CustomerDB", "Foreign" );
qApp->translate( "CustomerDB", "International" );
#endif
```

Cette approche ajoute une conséquente complication à votre système de compilation, mais s'assure que vous puissiez bénéficier des capacités de fusion de **lupdate**.

## VI - Divers

Ceci a été écrit pour Qt3. Cependant, la majorité du contenu reste d'actualité pour Qt4.

J'adresse ici de chaleureux remerciements à **lrmatDen**, **kinji1**, **yan** et **buggen25**, pour leur aide lors de la traduction, et à **matrix788** pour sa rapide relecture !

Au nom de toute l'équipe Qt, j'aimerais adresser le plus grand remerciement à Nokia pour nous avoir autorisé la traduction de cet article !


Copyright ® 2002 Jasmin Blanchette. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.