

[Suares & Co](#) [Tarieven](#) [Contact](#) [GNU/Linux](#) [QwikZite](#) [Portfolio](#) [Technieken](#) [Publicaties](#)

Technieken

Op deze pagina vindt u een aantal problemen waarvoor Suares & Co oplossingen zoekt.

Technology

On this page some solutions we found for some uncommon issues.

Creating a hunspell dictionary for use as spellcheck in Open Office

25-10-2008

Cool! OpenOffice/LibreOffice uses hunspell as spellchecker. This means that it is possible to create your own wordlist and affix file for your language, if that doesn't exist. Well, for papiamentu - Curaçao's native language - such dictionary doesn't exist. But I had a hard time finding out how to create such files, so I am documenting it here...

Installing hunspell on Ubuntu 8.04

Of course, you need hunspell installed. I am sure you had that already, and if not, the following might help:

```
sudo apt-get install hunspell hunspell-tools
```

The wordlist

A wordlist is just a list of words. It's the base for the dictionary. A wordlist can have 8000 words, or 100.000, or whatever seems reasonable for your language. Papiamentu has about 30.000 words at the moment.

Here's a very simple example wordlist:

```
love
lover
lovers
beer
beers
office
open
opened
chair
```

Save that lists as wordlist, just for now. Make sure there's a newline at the end of the file or else that last character will be eaten!

The Dictionary File

A dictionary file is just a wordlist preceded by the number of words. So this will make a dictionary file (sorted, too, and duplicates removed):

```
wc -l wordlist > yourlang.dic
sort wordlist | uniq >> yourlang.dic
```

14-05-2013 [Vagrant hangs at "Waiting for VM to boot. This can take a few minutes..."](#)

10-02-2011 [Zentyal/Ebox, slowness, and WDxxEARS Advanced Format](#)

01-09-2010 [Ebox 1.4 and HP Color LaserJet 2600N](#)

20-07-2009 [Firefox on LTSP as Local Apps \(Thin Client, Ubuntu\)](#)

03-07-2009 [ClamAV on Dapper](#)

21-06-2009 [Converting an Access database to mysql with mdbtools](#)

14-05-2009 [Zimbra, Kmail, Maildir convert, Kaddressbook imports and other Annoyances](#)

11-04-2009 [Installing Scalix 11.4 on Ubuntu 8.04 Hardy \(32bit\)](#)

10-04-2009 [Site slow after installing gzip compression: Transparent gzip and zlib.output_compression in PHP and Content-Length](#)

30-03-2009 [GOsa, Gonicus, openldap and ubuntu 8.10 \(gosa2\)](#)

15-03-2009 [Zimbra, Kmail, Maildir convert, Kaddressbook imports and other Annoyances](#)

16-08-2008 [CSS-only inline inplace edit \(no javascript\) in Ruby on Rails](#)

15-08-2008 [Using AJAX and link_to_remote with tabnav widgets in Ruby on Rails.](#)

07-08-2008 [Fixing plone-site on Ubuntu 8.04 Hardy](#)

07-08-2008 [Removing sendmail dependencies from spamass-milter on Ubuntu Dapper when using postfix](#)

24-07-2008 [Monitoring X Traffic with xmon on Ubuntu](#)

23-06-2008 [Thin Can DBE61 from Artecgroun on Hardy LTSP](#)

18-06-2008 [Avoiding Xgl on Ubuntu Hardy LTSP thin clients](#)

23-05-2008 [A box in a box in a box, part I: Compiling a Win4Lin enabled kernel \(for the unsupported WTS 3.0\)](#)

23-05-2008 [A box in a box in a box, part II: Installing Damn Small Linux with a Win4Lin-enabled kernel as a Kernel Virtual Machine](#)

23-05-2008 [A box in a box in a box, part III: Installing Win4Lin and Microsoft Windows 98 on DSL-n](#)

23-05-2008 [Installing Ubuntu 5.10 Breezy as Virtual Machine with qemu or kvm](#)

24-03-2008 [IBM Lotus Forms \(Formerly Workplace Forms, PureEdge\)](#)

14-03-2008 [ltsp-build-client and apt-mirror](#)

14-02-2008 [Adding a second hard disk to winxp under qemu on ubuntu linux.](#)

14-02-2008 [Trouble with raid and lvm on Ubuntu Dapper](#)

28-10-2007 [Installing jEdit on Ubuntu](#)

22-02-2006 [auth_ldap for Apache 1.3: a patch to allow anonymous binds to work properly.](#)

How to create the Affix File

An affix file is ehmm... quite complex. It took me a couple of hours to understand that it can just be an empty file... if you have a relatively small wordlist (let's say less than 100.000).

Create an empty affix file:

```
touch yourlang.aff
```

In the Right Place

On Ubuntu 8.04, the dictionaries are kept in `/usr/share/myspell/dicts/`. So copy our language there:

```
sudo cp yourlang.* /usr/share/myspell/dicts/
```

Testing the Hunspell

```
hunspell -d yourlang
```

This should give you a prompt. Enter a word and you will see a result:

```
Hunspell 1.1.9
love
*
```

This means, that the word 'love' is spelled correctly according to your language.

Now try a non-existing word:

```
bove
& bove 1 0: love
```

This means that 'bove' is not a word, but 'love' comes close.

More Affixion

You got to read the [manual](#) on the affix file format. [Here](#) is a less comprehensible but more comprehensive one. Just a small example:

```
SET UTF-8

SFX P Y 1
SFX P 0 s
```

It says the rule 'P' adds an 's' behind a word without removing any characters. It's a totally bogus rule for the english language, but it'll work fine with our example dictionary.

I also added the SET UTF-8 because in papiamentu, my accents got garbled.

Munch the .dic and the .aff

Munching will apply the affix rules to the dictionary, and produce a smaller dictionary. In fact, it will replace 'lover' with 'lover/P' and remove 'lovers'. One word less. Because the rule will discover these two words and decide that it's more efficient to use 'lover/P'. It'll also remove 'beers' and replace 'beer' with 'beer/P'. Munching will also add the wordcount at the beginning of the .dic file.

```
munch yourlang.dic yourlang.aff > yournewlang.dic
mv yourlang.dic yourlang.dic.old
mv yournewlang.dic yourlang.dic
sudo cp yourlang.* /usr/share/myspell/dicts/
```

Now test it again:

```
hunspell -d yourlang
Hunspell 1.1.9
love
*
```

```
lover
*
```

```
lovers
+ lover
```

Ah! 'love' and 'lover' as expected, and 'lovers' has 'lover' as base.. that's what the '+' means. But remember, for small wordlists, the affix file can be empty!

For the spellchecker to guess the correct 'suggestions', a frequency list of characters is needed. You can produce one like this:

```
tr -d "\n" < words \
| while read -n1 char; \
do echo $char; \
done \
| sort | uniq -c | sort -rn
```

That will give you a list like this:

```
14177 a
11060 i
10470 e
10172 o
9771 n
8638 s
```

You can take it a step further:

```
tr -d "\n" < words \
| while read -n1 char; \
do echo $char; \
done \
| sort | uniq -c | sort -rn \
| sed "s/^.* //" \
| tr -d "\n"
```

This will give you, for example:

```
aieonstrkldmuphbgáfveòcóiyéwñzjùüABCSKIHEPLGTFxMJRqYXVDç
```