

NAME

dw – find duplicate words

SYNOPSIS

```
dw [ --help ] [ --keep-lettercase ] [ --list-words ] [ --skip-punctuation ] [ --version ]
    [ --word-pattern '[initial][rest]' ] [ --word-pattern pattern-file ]
    < infile > outfile
or
    infile1 infile2 infile3 ... > outfile
```

DESCRIPTION

dw is a handy tool for finding a common typographical error in documentation. It filters its standard input, or a list of files, printing on standard output only those words that appear two or more times in succession. Each word is prefixed by its filename and line number(s). For *stdin*, the filename is a dash.

A *word* starts with a letter or underscore, and is followed by zero or more letters, underscores, or digits. Letter case is ignored. Program options can modify that behavior.

OPTIONS

Command-line options may be abbreviated to any unique prefix, with one or two leading dashes, and their lettercase is *not* significant. Thus, **-v**, **-V**, and **-VeRs** are equivalent to **--version**.

- help** or **--?** Display a help message on *stdout* and exit.
- keep-lettercase** Preserve input lettercase, overriding the normal behavior of comparing words in a single lettercase.
- list-words** Print each recognized word on a separate line on *stdout*. Use that option to check the word recognition algorithm, as well as to extract text from files that contain punctuation and other material. The word list can often usefully be passed to other text-processing tools, such as **aspell**(1), **grep**(1), **hunspell**(1), **ispell**(1), **sort**(1), **spell**(1), and **uniq**(1).
- skip-punctuation** Collapse to a single space all sequences of characters that are neither whitespace nor word characters. Without that option, the input `sally(sally)sally/sally` does not produce a doubled word report; with it, *three* such instances are diagnosed.
- version** Print the program name, version, and copyright information on *stdout* and exit.
- word-pattern** '[initial][rest]' or **--word-pattern** *file*

Specify an alternate pattern for recognizing the initial and following characters in words. The pattern is given in the following argument as two bracketed character sets, in which dashes represent ranges of consecutive characters. Alternatively, the pattern may be read from a user-supplied file, or from a file in the **dw** installation tree. In the latter case, it is found relative to the location of the executable program, in a directory named something like `../share/lib/dw/dw-2.02`.

Pattern files may contain leading comment lines beginning with a sharp character (**#**): reading terminates after the first noncomment line.

For installed pattern files, a suffix `.pat` is automatically supplied, and the file contents are in Unicode UTF-8 encoding. Such files are named by the *lowercase* English name of the language, or by its ISO 639-1 alpha-2 code. Those codes are often identical to the ISO 3166-1 alpha-2 codes that are used for Internet top-level domain names. Thus, **--word-pattern** `spanish` and **--word-pattern** `es` find equivalent pattern files, `spanish.pat` and `es.pat`, for that language.

The default pattern, similar to that for identifiers in many programming languages, and English-language text, is `[A-Za-z_][A-Za-z_0-9]`. That means an initial letter or underscore, followed by any number of letters, underscores, or digits. Whitespace around the bracketed patterns is ignored.

Any character in the pattern, *except* the character range dash, may be represented by an *octal escape sequence* of the form `\ooo`, `\oo`, or `\o`, where *o* is an octal digit, or by a *hexadecimal escape sequence* of the form `\xhh`, where *h* is a hexadecimal digit in 0–9a–f (ignoring lettercase), or by a *literal escape sequence* of the form `\c` where *c* is any single character, representing itself. The escape character checks are made in that order, so `\x` begins a hexadecimal sequence, rather than being a literal `x`.

Thus, the vowels **aeiou** may be represented by normal characters `aeiou`, by the literal sequence `\a\e\i\o\u`, by the octal sequence `\141\145\151\157\165`, by the hexadecimal sequence `\x61\x65\x69\x6f\x75`, or any mixture thereof.

BUGS AND FEATURES

dw assumes one-byte characters, as used in ASCII and the various ISO Latin-*n* character sets. Thus, it normally does not recognize words that contain multibyte characters, such as accented letters in Unicode UTF-8 encoding. Nevertheless, if you put multibyte characters into the **—word-pattern** value, they will be treated as word constituents, but so will any other characters that have the same leading bytes. While not strictly correct, for many languages, most words are correctly identified, because the UTF-8 encodings group accented letters together.

Here are examples of additional characters needed in word patterns (some glyphs may be lost due to output device and/or font limitations). All of them are recognized by **dw**, with language names in lowercase:

Arabic (ar)	<i>numerous letters and letter variants</i>
Danish (da)	ÉÆØÅ éæøå
default	[A-Za-z_][A-Za-z_0-9]
Dutch (nl)	<i>none normally needed, but sometimes, ligatures IJ ij</i>
English (en)	<i>none normally needed, but rarely, Æ Ĳ æ ĳ</i>
Finnish (fi)	ÉÄÄÖŠŽ éääöšž
French (fr)	ÂÊËÊËÇÎÏÔÛÜŸ Æ äêëêëçîïôôûÿ æ
German (de)	ÄÖÜ äöü ß
Greek (el)	[24 Greek letters, in each case]
Hebrew (he)	22 letters in one case, with 5 word-ending variants, plus apostrophe
Hungarian (hu)	ÁÉÍÓÚÖÜ áéíóúöü, plus long double acute accent on O U o u
Icelandic (is)	ÁÐÉÓÚÝÞ Ö áðéóúýþ
Irish (ga)	ÁÉÍÓÚ áéíóú
Italian (it)	ÈÊÎÏÎÓÔÙ èêîîîóôù
Latin-1	<i>all accented letters from ISO-8859-1 (Latin-1) encoding</i>
Lisp	-%\$:
Maltese (mt)	<i>four accented letters in two cases, plus apostrophe</i>
Norwegian (no)	ÂÊËÊËÖÔÆØÅ âêëêëöôæøå
Polish (pl)	<i>acute accent on C N O S Z c n o s z, ogonek on A E a e, overdot on Z z, and stroke on L l</i>
Portuguese (pt)	ÂÃÇÊÊÍÓÔÕÚ áããçêêíóôõú
Russian (ru)	<i>Cyrillic alphabet</i>
Scots (sc)	<i>none needed</i>

