Homophones is a utility to build and interrogate a database of homophonic word sets derived from the Merriam-Webster database. Due to inconsistancies in the pronunciations strings in Webster, the homophonic database does not contain all possible homophonic word sets (just over 1200 unique sets as of this release) and there are a small number of errors.

USAGE

```
Homophones [-b] [-d] [-r] [-h database] [-w dictionary]
```

-b *build* a new Homophone database. If the `Homophones.{D,L}` files exist, then they will be augmented, otherwise a new database is generated. Generally, you want to delete the `*.{D,L}` files before using this option.

-d *dump* the contents of the database in a plain text format. The entries come out in hash order so piping this through 'sort +1' is helpful, e.g.:

```
> Homophones -d | sort +1
['aE] a ae eh
[e] a i
```

```
['aH] aah ah
['aHv] ab of
['aE-bel] abel able
[e-'bil-et-eE] ability ibility
[e-'kaEd-eE-en] acadian akkadian
[ik-'sept] accept except
[ik-'sep-ter] accepter acceptor
[,ak-le-'maE-shen] acclamation acclimation
...
```

-r use *raw* (non-ASCII) format when printing pronunciation strings. This affects both the homophone lookup loop and the -d (*dump*) option.

-h use *database* for the homophone database file, defaults to '`Homophones`'.

-w use *dictionary* for looking up pronunciations, defaults to '`Webster-Dictionary`'.

With no arguments, **Homophones** goes into a `scanf` loop reading words, extracting their pronunciations from Webster and looking them up in the database:

```
> Homophones
birth
['berth] berth birth
wait
['waEt] wait weight
awl
['oGl] all awl
bowl
['boEl] bole boll bowl
^D
>
```

Homophones prints nothing if a matching pronunciation isn't found.

INSTALLATION

As configured, **Homophones** assumes the homophones database, `Homophones.{D,L}`, is in the

`/usr/local/lib/homophones` directory. You can change this assumption using the `-h` option or by changing the line:

```
#define DATABASE "/usr/local/lib/homophones/Homophones"
```

in `Homophones.m` and do a '`make`'. You can test out Homophones without installing the database files by doing '`Homophones -h Homophones`' which will look for them in the connected directory.

IMPLEMENTATION

The homophone routines extract the pronunciation strings (all senses) from Webster and then break them up into individual pronunciations (e.g. \'a(e)r, 'e(e)r, 'aHr\ becomes \'a(e)r\, \'e(e)r\ and \'aHr\). Any 'incomplete' entries are discarded. Incomplete entries are those, ignoring leading or trailing non-alphabetics, that begin or end with a hypen (e.g. \ik-'sept, ak- also ek-\ becomes \ik-'sept\). Any 'garbage' entries are discarded as well (empty strings, single characters, digits, spaces, etc.).

The homophone database file consists of (non-ASCII) pronunciation strings as keys (NXAtoms) and Storage objects as

values. The Storage objects each contain of two or more NXAtoms which are the words that share the pronunciation string key.

The database file is generated using the HashFile object, an alternate interface to the db(3) routines that makes a database file look exactly like a HashTable object (should be available from the same archives as Homophones). The files `libHashFile.a` and `objc/HashFile.h` are from the HashFile module.

AUTHOR

Christopher Lane (`lane@sumex-aim.stanford.edu`)
Symbolic Systems Resources Group
Knowledge Systems Laboratory
Stanford University