Cash-File is a front end to Hash-File which uses a hash table to cache accesses to hash files. This can provide a significant performance improvement in applications which access a small number of keys repeatedly. For example, the Where-Is library module uses this module to achieve acceptable interactive performance.

Cash-File is similar to but not compatible with the LispUsers' module, HASHBUFFER.

All of the code for Cash-File is in a package called Cash-File. Througout this document Lisp symbols are printed as though in a package which uses the packages Cash-File, Hash-File, and Lisp.

Installation

Load CASH-FILE.DFASL and HASH-FILE.DFASL from the library.

Functions

The functional interface is designed to closely resemble that of Hash-File, which was in turn designed to resemble the Common Lisp hash table facility.

(make-cash-file *file-name size cache-size*) [Function]

Creates and returns an empty cash file in *file-name*. *Size* is passed as the *size* argument to `make-hash-file`, while *cache-size* is passed as the *size* argument to `make-hash-table` and determines the maximum number of entries that will be cached.

(get-cash-file *key cash-file &optional default*) [Function]

Just like `get-hash-file` and `gethash`. Retrieves the value stored under *key* in *cash-file* or *default* if there is none. Also returns a second value which is true if a value was found for *key*.

A `setf` method is also defined for `get-cash-file`.

(open-cash-file *file-name cache-size &key direction*) [Function]

Open the existing hash file in *file-name* in *direction* (`:input` or `:io`). *Cache-size* is passed as the *size* argument to `make-hash-table` and determines the maximum number of entries which will ever be cached.

(rem-cash-file *key cash-file*) [Function]

Like `rem-hash-file` and `remhash`. Deletes key from the hash file and the cache. Returns true if and only if there was a value stored under *key*.

(cash-file-p *object*) [Function]

Returns true if and only if *object* is a cash file.

(cash-file-p *object*) ≡ (typep *object* 'cash-file)

(cash-file-hash-file *cash-file*) [Function]

Returns the hash file object which *cash-file* is a front end to.

There are no cash file specific equivalents for `close-hash-file`, `map-hash-file` and `hash-file-count`. For these use the hash file functions on the `cash-file-hash-file`.

Implementation Notes

A queue is maintained to enable cache deletion when the cache is full. This queue is implemented as a list. Each time a key is accessed, it is moved to the head of the queue. The last element of the queue is deleted when a new key is accessed and the queue is full.

Limitations

The cache time is not constant but grows linearly with the size of the cache. For this reason, huge caches are not recommended.