Perils of Doubly-weak Hashtables

European Lisp Symposium 2013
Jason Cornez
CTO, RavenPack

Doubly-weak Hashtable

- Weak Values / Weak Keys
- When value is finalized:

  entry remains; value set to nil
- When key is finalized:

  entry removed

First Attempt

```
(defvar *dictionary* (make-hash-table :weak-keys t
                                      :values :weak
                                      :test 'equal))

(defstruct token name)

(defun get-token (text)
  (let ((key (string-upcase (copy-seq text))))
    (multiple-value-bind (value exists)
        (gethash key *dictionary*)
     (unless exists
        (setq value (make-token :name key))
        (setf (gethash key *dictionary*) value))
     value))))
```

Second Attempt

```
(defvar *dictionary* (make-hash-table :weak-keys t
                                      :values :weak
                                      :test 'equal))

(defstruct token name)

(defun get-token (text)
  (let ((key (string-upcase (copy-seq text))))
    (excl:critical-section ()
      (multiple-value-bind (value exists)
          (gethash key *dictionary*)
       (unless exists
          (setq value (make-token :name key))
          (setf (gethash key *dictionary*) value))
       value)))))
```

Third Attempt

```
(defvar *dictionary* (make-hash-table :weak-keys t
                                      :values :weak
                                      :test 'equal))

(defstruct token name)

(defun get-token (text)
  (let ((key (string-upcase (copy-seq text))))
    (excl:critical-section ()
      (multiple-value-bind (value exists)
          (gethash key *dictionary*)
       (unless value
          (setq value (make-token :name key))
          (setf (gethash key *dictionary*) value))
       value)))))
```

Finally Correct

```
(defvar *dictionary* (make-hash-table :weak-keys t
                                      :values :weak
                                      :test 'equal))

(defstruct token name)

(defun get-token (text)
  (let ((key (string-upcase (copy-seq text))))
    (excl:critical-section ()
      (multiple-value-bind (value exists)
          (gethash key *dictionary*)
       (unless value
          (setq value (make-token :name key))
          (when exists (remhash key *dictionary*))
          (setf (gethash key *dictionary*) value))
       value)))))
```