Символы и их свойства.

Лекция 8 (часть 1).

Определение и модификация списка свойств.

Специальности : 230105, 010501

Символы в Лиспе.

Определение. **Символ – это имя, состоящее из букв, цифр, специальных знаков и обозначающее какой-либо предмет, объект, вещь, действие из реального мира.**

В Лиспе символы могут обозначать любые лисповские объекты, включая функции.

Символ является структурным объектом, состоящим из четырех компонент, соответствующих имени, значению, а также связанных с символом определению функции и списку свойств.

Для нового символа интерпретатор резервирует память для возможного значения, определения функции и других свойств. Символы сохраняются в памяти в списке объектов, который содержит как созданные пользователем, так и внутрисистемные символы. В Коммон Лиспе и ряде новых Лисп-систем, в том числе newLISP-tk, возможно использовать несколько различных списков объектов, каждый из которых именуется пакетом или контекстом (в newLISP-tk).

Функции чтения свойств символа в Коммон Лиспе.

Для чтения значений различных системных свойств символа в Коммон Лиспе, а также в библиотеке COMMON.LSP muLISP'a существуют специальные функции, приведенные в таблице 1.

Таблица 1. Функции чтения свойств символа.

Характеристика	Форма представления	Функция чтения
Имя	Набор литер	(SYMBOLP *имя*)
Значение	Произвольный лисповский объект	(SYMBOL-VALUE *имя*)
Определение функции	Лямбда-выражение	(SYMBOL-FUNCTION *имя*)
Список свойств	Список точечных пар	(SYMBOL-PLIST *имя*)

Свойства символов.

В Лиспе с символом можно связать именованные свойства. Свойства символа записываются в хранимый вместе с символом список свойств :

(<имя символа> (<имя свойства 1> <значение свойства 1>) . . .

(<имя свойства N> <значение свойства N>))

Для работы со списками свойств в основных диалектах Лиспа имеются три встроенные функции :

1.Включение свойства в список свойств.

(put <символ> <свойство> <значение свойства>)

2. Просмотр значения заданного свойства.

(get <символ> <свойство>)

3. Удаление заданного свойства из списка свойств.

(remprop <символ> <свойство>)

Показанное описание свойств символов используется при создании динамических баз данных и текстовых редакторов.

Свойства символов в Коммон Лиспе.

В Коммон Лиспе функции наподобие PUT не существует.

Поскольку свойства символов находятся в связанных с символами ячейках памяти, для присваивания значений которым в Коммон Лиспе используется обобщенная функция присваивания SETF, присваивание свойств в Коммон Лиспе осуществляется суперпозицией функций SETF и GET :

(setf (get <символ> <свойство>) <значение свойства>)

Здесь вызов функции GET возвращает в качестве значения ячейку памяти для данного свойства, содержимое которой обновляет произведенный вызов SETF. Присваивание будет работать и в том случае, когда у символа ранее не было такого свойства.

Псевдофункция SETF меняет физическую структуру списка свойств.

Формирование списка свойств (muLISP).

<u>Постановка задачи.</u> Дан символ с некоторым именем *name*. Требуется сформировать список свойств.

Данная задача решается в три этапа. Сначала необходимо задать (ввести) количество свойств, затем ввести список названий свойств и только затем – значения свойств.

```
; Формирование списка свойств символа
; Головная функция формирования списка свойств
(defun f1 (name)
 (print (pack* "Введите количество свойств символа "
               name))
 (f3 name (f2 name (read))))
```

```
; Функция формирования списка названий свойств
(defun f2 (name num)
 ((zerop num) nil)
 (print (pack* "Введите название "
               num
               "-го свойства символа " name " : "))
 (cons (read)(f2 name (- num 1))))
```

```
; Ввод значений свойств
(defun f3 (symb_name prop_name_list)
 ((null prop_name_list) nil)
 (print (pack* "Введите значение свойства "
               (car prop_name_list)
               " символа " symb_name))
 (put symb_name (car prop_name_list)(read))
 (f3 symb_name (cdr prop_name_list)))
```

Здесь в качестве вспомогательной используется функция конкатенации : (pack* *строка1 … строкаN*)

Удаление заданного свойства (muLISP).

Постановка задачи. **Дан список символов lst и некоторое свойство prop. Требуется : удалить свойство prop у всех тех символов списка lst, у которых это свойство имеется.**

Для решения данной задачи воспользуемся функциями get и remprop.

```
; Вариант 1
(defun f4 (lst prop)
 ((null lst) nil)
 (remprop (car lst) prop)
 (f4 (cdr lst) prop))
; Вариант 2 - неправильный
(defun f5 (lst prop)
 ((null lst) nil)
 ((get (car lst) prop)(remprop (car lst) prop))
 (f5 (cdr lst) prop))
```

```
; Подготовка тестовых
; данных
(put one 'Num 'Nechet)
(put two 'Num 'Chet)
(put A 'Sym 'A-lat)
(put B 'Sym 'B-lat)
(put three 'Num 'Nechet)
(put four 'Num 'Chet)
(put C 'Sym 'C-lat)
(put D 'Sym 'D-lat)
```

Результатом вызова (f4 '(one A two B) 'Num) будет удаление свойства Num у символов one и two.

НО ! вызов (f5 '(three C four D) 'Num) не приведет к удалению свойства Num у символа four, поскольку в случае обнаружения искомого свойства у одного из символов списка рекурсивного вызова функции f5 уже не произойдет.

Удаление заданного свойства у символов списка : вариант с использованием LET.

Еще один вариант решения задачи связан с использованием локального определения LET.

```
; Удаление заданного свойства - вариант с использованием
; локального определения LET
(defun f6 (lst prop)
 (let
   ((obj_list lst)
    (property prop))
  ((null obj_list) nil)
  (remprop (car obj_list) property)
  (f6 (cdr obj_list) property)
 )
)
```

Изменение значения заданного свойства (muLISP).

Постановка задачи. **Дан список символов lst и некоторое свойство prop. Требуется : заменить текущее значение свойства prop заданным значением val у всех тех символов списка lst, у которых это свойство имеется.**

```
; Изменение значения заданного свойства
(defun chngprop (lst prop val)
 ((null lst) T)
 ((and (get (car lst) prop)
       (chngprop (cdr lst) prop val))
  (put (car lst) prop val))
 (chngprop (cdr lst) prop val))
; Тестовый набор данных
(put one 'What_is_it Number)
(put two 'What_is_it Number)
(put A 'What_is_it Symbolic_value)
(put B 'What_is_it Symbolic_value)
(setq input_list '(one A two B))
```

Вызов (chngprop input_list 'What_is_it 'Atom) приводит к тому, что у всех элементов списка input_list, для которых определено свойство What_is_it, в качестве нового значения этого свойства будет задано Atom.

Изменение значения свойства на заданное (muLISP).

Постановка задачи. **Дан список символов lst и некоторое значение old_val свойства prop. Требуется : заменить значение old_val свойства prop новым значением new_val у всех тех символов списка lst, у которых это свойство имеется.**

```
; Замена заданного значения заданного свойства новым
(defun chngprop2 (lst prop old_val new_val)
 ((null lst) T)
 ((and (equal (get (car lst) prop) old_val)
       (chngprop2 (cdr lst) prop old_val new_val))
  (put (car lst) prop new_val))
 (chngprop2 (cdr lst) prop old_val new_val))
; Тестовый набор данных
(put one 'What_is_it Number)
(put two 'What_is_it Number)
(put A 'What_is_it Symbolic_value)
(put B 'What_is_it Symbolic_value)
(setq input_list '(one A two B))
```

Вызов (chngprop2 input_list 'What_is_it 'Symbolic_value 'Symbol) приводит к тому, что у всех элементов списка input_list, значением свойства What_is_it которых является Symbolic_value, в качестве нового значения этого свойства будет задано Symbol.

Использование списков свойств символов при разработке текстового редактора (muLISP).

Постановка задачи. Требуется обеспечить выполнение функций редактора для разных режимов работы при нажатии определенных клавиш. С нажатием одной и той же клавиши в разных режимах могут быть связаны разные функции. Отдельные клавиши задействуются только в определенных режимах : F3 – загрузка текста из файла (при активизации главного меню), ↑, ↓, ← и → - перемещение курсора, F2 – сохранение текста в файле (в режиме правки текста).

Решение. Свяжем с символом, соответствующим ASCII-коду каждой из задействованных клавиш, список свойств. Каждому из режимов, где клавиша будет задействована, поставим в соответствие название свойства, а имени вызываемой по нажатию клавиши функции – значение свойства. Для вызова самой функции воспользуемся функционалом MAPC.

Пример для клавиш Enter и Backspace в режиме редактирования (смотри файл muledit.lsp) :

```
(MAPC '(LAMBDA (PAIR)
                    (PUT (ASCII (CAR PAIR)) 'EDITOR (CADR PAIR)))
       '((13 EDITOR-ENTER)
         (8  EDITOR-BACKSPACE)))
```

Литература.

1 Хювенен Э., Сеппянен Й. Мир Лиспа. Т.1. – М.:Мир, 1990. С. 61-63, 168-172 , 326-327.