AUTOCAD & LISP (7)

дипл. инж. Дионис Христов

Селекциони множества

Со AutoLisp може да се селектира група ентитети од цртежот. Групата на селектирани ентитети се вика селекционо множество и има единствено име со помош на кое се манипулира со селекционото множество (Видете го написот во претходиот број за поврзаноста на ентитет името и селекционите множества).

AutoLisp содржи неколку функции што овозможуваат да се манипулира со селекционите множества: додавање на нови ентитети, бришење на ентитети итн.

Функции за работа со селекциони множества

SSGET

(ssget <mode> <t1> <t2> <plist> <fileter>)

SSGET е основана селекциона функција што се користи за селектирање на ентитети од цртежот и како резултат го враќа селекционото множество што ги содржи селектираните ентитети. Оваа функција може да се повика без ниеден или, пак, со еден од аргументите како што е покажано подоле.

Ако се повика без аргументи, SSGET е интерактивна. AutoLisp на комадниот промпт ќе ја покаже пораката Select objects: и корисникот ќе може да ги селектрира елементите на начин кој нему му одговара, со една точка, рамка, прозор, snap или некој друг метод.

Fileter аргументот е листа што содржи еден или повеќе подлисти на групни кодови (за значењето на групните кодови види го написот во претходниот број). Во селекционото множество ќе се вклучат само оние ентитети од цртежот што ги имаат карактеристиките дефинирани со filter листа независно од другите аргументи на SSGET.

Ако SSGET се повика со аргументот mode, таа веќе не е интерактивна. mode е стринг кој го дефинира методот на селекција и автоматски ги селектира ентитетите од цртежот. Подоле се дадени примери за користење на mode аргументот;

(ssget "C" '(t1) '(t2))

ги селектира сите елементи што ја сечат рамката дефинирана со двете точки t1 и t2, а тоа се точките на горниот лев и долниот десен агол од рамката.

(ssget "I")

селектираните елементи ги става во селекционото множество PICKFIRST

(ssget "L")

го селектира последниот ентитет во цртежот

(ssget "P")

го селектира претходно селектираниот елемент

(ssget "P" <filter>)

ги селектира елементите од претходното селекционо множество што ги задоволуваат карактеристиките дефинирани со <filter> листата.

(ssget "W" '(t1) '(t2))

ги селектира ентитетите што се наоѓаат во прозорецот дефиниран со точката t1 како негово горно лево теме и t2 како долно десно теме.

(ssget '(t1))

го селектира првиот ентитет којшто проаѓа низ точката t1

(ssget "WP" '(plist))

ги селектира ентитетите внатре во полигонот дефиниран со серијата темиња во листата plist

(ssget "WP" '(plist) ' (filter))

ги селектира ентитетите внатре во полигонот дефиниран со серијата темиња во листата plist и кои ги задоволуваат карактеристиките дадени со filter листата.

(ssget "CP" '(plist))

ги селектира ентитетите коишто ја сечат или лежат внатре во полигонот дефиниран со серијата на темиња во листата plist.

SSGET треба да се користи со точка како аргумент само во случајот кога во близина на дадената точка нема други ентитети. Во некои случаи, SSGET може погрешно да го селектира ентитетот ако во негова близина има повеќе други ентитети.

(ssget "X" <filter>)

SSGET "X" го пребарува целиот цртеж и го креира селекционото множество што ги содржи сите ентитети (нивните ентитет имиња) кои ги задоволуваат карактеристиките во filter листата. Со помош на оваа функција може да се добијат селекциони множества што ги содржат ентитети од одреден тип, на одредено ниво или пак, со одредена боја.

filter листата е асоцијативна листа слична на онаа која ја дава функцијата ENTGET. Таа одредува кои карактеристики треба да ги имаат ентитетите за да се вклучат во селекционото множество.

(ssget "X" (list (cons 0 "LINE")))

ги селектира сите ентитети од типот LINE (односно сите линии во цртежот). Кодот на група 0 означува тип на ентитет.

(ssget "X" (list (cons 8 "SPRAT")))

ги селектира сите ентитети што се наоѓаат на нивото SPRAT. Кодот на групата 8 означува ниво. Ако во filter листата има повеќе од една точка пар подлииста сите тие треба да се задоволат, односно енитетите да ги имаат тие карактеристики (како да се поврзани со лигичко И).

(ssget "X" (list (cons 0 "LINE") (cons "SPRAT") (cons 61 2)))

ги селектира сите линии со црвена боја кои се на нивото SPRAT. Распоредот на точка пар подлиситите не е значаен односно тие можат да се постават во било кој распоред.

filter листата не може да ги содржи сите кодови на група коишто ги има во асоцијативната листа што ја враќа ENTGET. AutoLisp препознава само едно подмножество од нив. SSGET работи на тој начин што го пребарува целиот цртеж и ги споредува подлистите во filter листата со полињата на ентитетите (главните ентитети). Ако некој ентитет ги поседува сите карактеристиките што ги дефинира filter листата, тој се додава на селекционото множество. Функцијата враќа nil ако ниеден ентитет во цртежот не ги задоволува сите барања. Празна filter листа ги селектира сите ентитети на цртежот.

Следнава табела ги дава кодовите на група, што можат да се употребат во SSGET:

Код	Значење
0	Тип на ентитет
2	Име на блокот (во поинтерот на блокот, INSERT)
6	Тип на линија
7	Стилот на текст за Текст или Атрибути
8	Име на ниво
39	Дебелина (реален број)
62	Боја (256 = "BYLAYER" , 0 = "BYBLOCK")
66	Атрибут флагови (во поинтерот на блокот, INSERT)
210	3Д екструзија (вектор, листа од три релани броја)

SSGET многу често се користи во AutoLisp програмите. Нејзината употреба е многу корисна при работа со блокови. Следниов пример ги селектира сите блокови со име TRAFO што имаат црвена боја и се наоѓаат на нивото ISKLUCENI.

(ssget "X" (list (cons 2 "TRAFO") (cons 8 "IKLUCENI") (cons 62 1)))

Ако во fileter листата се употребат кодови на група што не се прифатени (не се наоѓаат во горната табела) SSGET ќе врати nil. Во сите оние групни кодови каде што треба да се употреби реален број, ако се употреби цел број, ќе дојде до грешка. SSGET не врши автоматско претворање (во овој случај од цел во реален број). На пример:

```
(ssget "X" (list (cons 39 4)))          ;грешка, треба реален број
(ssget "X" (list (cons 39 4.0)))        ;точно
```

Во селекционите множества никогаш нема дупликат на еден ист ентитет, односно ако еден ентитет е веќе во селекционото множество, со негово повторно селектирање не се прави дупла референца.

SSLENGTH

(sslength <selm>)

Функцијата го враќа целиот број што кажува колку ентитети има во селекционото множество selm. Ако бројот на ентитетите е поголем од 32767, тогаш овој број е реален. Ако на цртежот има пет блокови со име TRAFO, следниов пример ќе врати пет:

```
(setq selmno (ssget "X" (list (cons 2 "TRAFO"))))
                                ;го креира селекционото множество
(sslength selmno)               ;враќа 5
```

SSADD

(ssadd [<entime> [<selmno>]])

Со оваа команда се креираат празни селекциони множества и се додаваат нови ентитети во веќе постојните селекциони множества. Ако SSADD се повика без аргументи, тогаш се креира празно селекционо множество. Ако се повика само со името на ентитетот, тогаш се креира ново селекционо множество само со еден ентитет во него. И на крај, ако се повика со ентитет име и селекционо множество, тогаш во selmno го додава ентитетот entime. Важно е дека селекционото множество selmno мора претходно да постои . Ентитетот ќе се додаде во селекционото множество само ако тој не е претходно додаден. Ако тој постои во селекционото множество, целата функција се игнорира.

```
(setq ent (cadr (entsel)))      ;интерактивно селектира
                                 еден ентитет
(setq selmno (ssadd))           ;креира празно селекционо
                                 множество selmno
(ssadd ent selmno)              ;го сместува во selmno
(setq entsle (entnext ent))     ;го селектира следниот ентитет во
                                 цртежот по ent
(ssadd entsle selmno)           ;го сместува и него во selmno
```

SSNAME

(ssname <selmno> <redbroj>)

Оваа функција овозможува пристап до ентитетите во селекционото множество selmno. SSNAME го враќа ентитет името на ентитетот што е на позиција redbroj во selmno. Елементите во selmno почнуваат да се бројат од 0, па натаму. Ако redbroj е поголем од бројот на ентитетите во selmno, функцијата враќа nil. На пример :

```
(setq site (ssget))             ;ги селектира сите ентитети во
                                 цртежот
(setq ent1 (ssname 0 site))     ;го враќа ентитет името на првиот
                                 ентитет во site
(setq ent2 (ssname 2 site))     ;го враќа ентитет името на тртиот
                                 ентиет во site
```

Ако во селекционото множество сакаме да пристапиме до ентитет што е на позиција поголема од 32767 наместо redbroj да биде интегер, треба да биде реален број (наместо 34245, да се употреби 34245.0).

SSDEL

(ssdel <entime> <selmno>)

SSDEL го брише ентитетот entime од селекционото множество selmno и го враќа selmno. Ако ентитетот не е во selmno, се враќа nil. Оваа функција е спротивна на функцијата додавање ентитети SSADD. Важно е да се забележи дека пред да се избрише некој енитет, неговото ентитет име треба да се знае. На пример, да го избришеме петиот ентитет од селекционото множество selmno:

```
(if (not (setq ent (ssname 4 selmno)))
    (ssdel ent selmno))
```

Прво проверуваме дали во selmno има петти ентитет. Ако постои петти ентитет, SSNAME го враќа неговото ентитет име, а ако не, тогаш враќа nil. Ентитет името се запомнува во ent променливата и се користи во SSDEL.

SSMEMB

(ssmemb <entime> <selmno>)

Со оваа функција се проверува дали ентитетот entime го има во селекционото множество selmno. Ако го има, тогаш го враќа неговото ентитет име, ако не тохаш враќа nil. На пример:

```
(setq selmno (ssget "X" (list (cons 2 "TRAFO"))))
(if (ssmemb (cadr (entsel)) selmno) (princ "\nSelektiraniot element e TRAFO."))
```

SSGET креира селекционо множество selmno од сите блокови TRAFO во цртежот. Со entsel (види го написот во претходниот број) интерактивно се селектира еден ентитет од цртежот. Ако ентитетот е во selmno (се проверува со SSMEMB), напиши во нов ред (\n) порака на командниот промпт.

Неколку корисни примери на функциите за селекција

Следнава функција AddSet врши додавање на последниот нацртан ентитет во цртежот во селекционото множество. Функцијата е посебно корисна кога се употребува за симултано цртање и додавање на ентитетите во селекционо множество. Селекционото множество кое е аргумент на функцијата, мора да постои пред да се повика AddSet. Употреба:

```
(addset <selmno> )
```

Празно селекционо множество се креира со помош на функцијата SSADD:

```
(setq ss (ssadd))
(addset ss)
```

Кодот на AddSet е:

```
(defun addst (ss)
; proveruva dali ss e od tipot na
;selekciono mnozestvo
    (if (= type ss) 'PICKSET)
;ako e go dodava posledno nacrtaniot
;entitet na ss
            (ssadd (entlast) ss)
    )
)
```

Rdwset е функција која врши рефреширање (redraw) на оние ентитети што се во селекционото множество. Покрај тоа, кодот на оваа функција е добар пример како се 'пронаоѓа' селекционо множество (да се достапи итеративно до секој негов ентитет и да се изврши некоја акција врз нив).

```
(defun rdwset ( ss / i ss_len)
;proveruva dali ss e od tipot na selekciono mnozestvo
; i dali mu e dolzinata pomala od 32767
 (cond ((and   (= (type ss) 'PICKSET)
               (= (type (setq sslen (sslength ss))) 'INT)
               )
;ako e zadovoleno toa se incijalizira i na -1
               (setq i -1)
; Se povtoruva funkcijata redraw za sekoj entitet vo ss.
; Na mestoto na redraw moze da stoi i druga funcija so
; druga namena koja zema za argument entitet ime.
; Inkrementiranjeto na i ovozmozuva 'proagjanje' na ss.
       (repeat sslen
               (redraw (ssname ss (setq i (1+ i))))
               ) ;kraj na repeat
       )
       ) ; kraj na cond
) ;kraj na defun
```

Rdwset би била пребавна ако се користат селекциони множества со поголем број ентитети од 32767. Затоа се проверува дали должината на ss ја преминува оваа должина.

За крај

Со овој напис ја завршуваме серијата написи за AutoCAD и AutoLisp. Серијата беше наменета за корисници на AutoCAD, како и за идните програмери на AutoCAD апликации. За оние што имаат намера да креираат AutoCAD апликации, AutoLisp е основна алатка (иако не и единствена). Поради својата едноставна структура и прегледност, развојот на апликацијата е краток, а тестирањето лесно. LISP функциите брзо се развиваат и секоја идеја може брзо да се напише и провери. LISP рутините многу го олеснуваат креирањето на комплексните цртежи и ја елиминираат потребата од многубројни избори од менијата и внесување од тастатура.

Останува прашањето дали на корисниците кои AutoCAD го користат како своја работна околина, AutoLisp им е потребен. Сигурно да. Некоја мала LISP рутина што може да се развие за 10 саати, може да заштеди неколку минути по цртеж и да се 'отплати' себеси во смисла на ефикасност и поголема продуктивност. Точно е дека програмирањето е често пати поврзано со многу "мачното" тестирање и дебагирање, со вовлекување во проблеми таму каде што не се очекуваат. Но, на тој начин, исто така, се откриваат сите можности и сета моќ на AutoLisp.