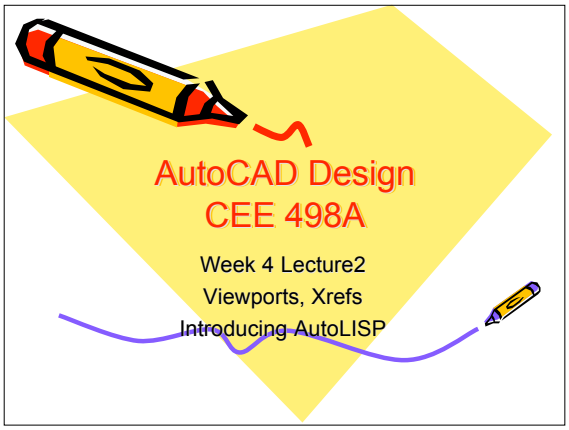

Viewports

- ***vports*** command
- Allows the user multiple windows
- Each window captures specific areas
- Can be scaled and zoomed appropriately
- Demo in class tomorrow

External References

- Create composite drawings creating other saved drawings
- Useful in multi-disciplinary efforts
- Temporarily attached or overlayed
- Unlike a block that is permanently inserted

XREF

- Can: Any operation that treats the xref as a single blocked entity
  - Move, Delete, Rotate
  - Change properties of layer
- Cannot: Any operation that modifies properties of individual components
  - Trim, Explode, Extend, Stretch

The XREF Manager

- Detach: Remove xref
- Reload: Update changes in xref
- Unload: Remove file from link but maintain pointer
- Bind: Permanent insertion: pointer destroyed
- Save Path: Saves pointer to xref

Introducing AutoLISP

- So far : Menu driven and Command line Interfaces
- AutoLISP : Programming Interface
  - Customizing AutoCAD
  - Programming repetitive jobs
- Uses AutoCAD's built in LISP interpreter

Why AutoLISP

- Unit Conversion
- Automatic context sensitive plotting
- Coordinate system transformations
- System calls : Device handling
- File handling
- Database interfacing

AutoLISP

- Code can be entered at command line
  - Results immediately returned
- It can be written in ASCII text files and loaded with extension *.lsp*
  - Load application files at cmd line:
  - *(load "my_file")*
- *acad.lsp* file can be configured to load files automatically

AutoLISP Expressions

(function arguments)
Cmd> (* 2 27)
54

(fun1 (fun2 arguments) (fun3 arguments))
Cmd>(* 2 (+ 5 10))
30

AutoLISP Data Types

- Integers, Reals, Strings
- Lists: points
  - (1.0 0.0 3.0) (1 “ONE”)
- Selection sets: groups of objects
  - > (ssget “my_drg”)
    <Selection set: 1>
- Entities
- File Descriptors

Variables

Symbols refer to data

- Names are not case sensitive
- Should not include ().`’ ” ;
- Cannot be only numeric

Assign values to variables using *setq*

- (setq str “my string”)
  > “my string”
  > !str
  > “my string”

Symbols

Constants: PI = 3.1416

Example

(setq r 5)
(setq a (* pi r r))
What is ‘a’?

AutoLISP so far . . .

- Expressions:
  - *(function arguments)*
  - (* 2 27)
- Data types:
  - *Lists, selection sets*
- Assignment:
  - *(setq x 5)*
- Global assignment:
  - !x

Functions

- Use *defun* to declare functions:

```
(defun myProg()
   ( … )
   ( … )
)
Cmd> (myprog)
```

Functions

```
(defun C:myProg()
   (setq pt1 (getpoint "\nFrom point:") )
   (setq pt2 (getpoint pt1 "\nTo point:") )
   (distance pt1 pt2)
)
Cmd> (load "myprog")
Cmd> myprog
Cmd> From Point: <pick>
To Point: <pick>
Return distance
```

Functions: arguments

```
(defun dtr(degree)
   (/ (* degree pi) 180.0 )
)
Cmd> (dtr 180)
3.14159
```

Function: Syntax

```
(defun <name> <arg list / local var>
   <expr>
   <expr>
     . . .
)
```

Variable Scope

- Local and global
- Global:
  - Available to any expression, in or out of a function
- Local:
  - Variables specifically defined within the scope of a function

Variable Scope

```
(setq x 1)
Cmd> !x
1
(defun fun( / x)
   (setq x 2)
)
Cmd> (fun)
2
Cmd> !x
```

Loops and Conditionals

- *(while expr1 expr2)*
  - expr1: this is true…
  - expr2: … do this
- *(repeat int)*
- *(if expr1 expr2 expr3)*
  - expr1: this is true…
  - expr2: … do this
  - expr3: … else do this
- *(cond (expr1 expr2) (expr3 expr4) …)*
  - expr1:if this is true…
  - expr2: … do this
  - expr3:else if this is true…
  - expr4: … do this

Predicates

- Returns a boolean answer
- *(and expr1 expr2)* e1 and e2 returns non-zero?
- *(> 5 6)* Is this >/</= ?
- *(zerop x)* Is x = 0?
- *(minsup x)* Is x a negative number?

Data Structures

- All data structures are lists
  - A point is a data structure
  - Any entity is represented by a data structure
- Manipulating data structure/ lists

DS Manipulation

- *(append list1 list2)*
  - (append '(1 2) '(3 4)) Returns (1 2 3 4)
- *(apply exp list)*
  - (apply + (1 2 3)) Returns 6
- *(assoc item list)*
  - (assoc 1 '((1 "David") (2 "Allen") …))
- *(car list)*
  - (car '(1 2 3 4)) Returns 1
- *(cdr list)*
  - (cdr '(1 2 3 4)) Returns (2 3 4)

DS Manipulation

- *(cons item list)*
  - (cons 0 '(1 2 3)) Returns (0 1 2 3)
- *(foreach name list exp)*
  - (foreach n '(1 2 3 4) (+ n 1)) Returns (2 3 4 5)
- *(mapcar function list1 list2 …)*
  - (mapcar * '(1 2 3) '(4 5 6)) Returns (4 10 18)
- *(member exp list)*
  - (member 2 '(1 2 3 4)) Returns (2 3 4)
- *(nth n list)*
  - (nth 2 '(1 2 3 4)) Returns 2
- *(subst new old list)*
  - (subst 5 2 '(1 2 3 4)) Returns (1 5 3 4)

Command line calls

- *(command "line" '(1 2) '(2 3) "")*
- Can be used to call any AutoCAD command
- User input using *getx*
  - *(getangle [pt] [prompt])*
  - *(getdist [pt] [prompt]*
  - *(getcorner pt [prompt])*
  - *(getint [prompt]) (Also for reals and strings)*

DXF

- Documented representation of complete contents of an AutoCAD drawing
- Allows transference of drawing data via data structures encoded in text

Selecting DXF Group Codes

(setq en (car (entsel "\n Select entity:")))
(setq enlist (entget en))

Cmd> Select entity: <pick an arc>

```
((-1 . <Entity name: 2b80648>) (0 . "ARC") (5 . "89")
(100 . "AcDbEntity") (67 . 0) (8 . "STR") (100 ."AcDbCircle")
 (10 1.5 0.5 0.0) (40 . 1.58114) (210 0.0 0.0 1.0) (100 "AcDbArc")
 (50 .5.96143) (51 . 3.46334))
```

(-1 . <Entity name:2b80648>) -1 - Entity Name
(AutoCAD Automatically takes care of this)

(0 . "ARC")	0 - Entity Type
(8 . "STR")	8 - Layer Name
(10 1.5 0.5 0.0)	10 - Center Point
(40 . 1.58114)	40 - Radius
(50 .5.96143)	50 - Start Angle Expressed in Radians
(51 . 3.46334)	51 - End Angle Expressed in Radians
(210 0.0 0.0 1.0)	210 - Don't worry about this Extrusion Factor

- *(cdr (assoc 10 enlist))*
  - *Returns?*
- *(cdr (assoc 40 enlist))*
  - *Returns?*
- *(cdr (assoc 8 enlist))*
  - *Returns?*
- *(cdr (assoc 50 enlist))*
  - *Returns?*
- *(cdr (assoc 51 enlist))*
  - *Returns?*

Analyzing Code