This series of lessons deal with the fundamentals of mac programming. Remember that mac programming is **event driven**....all those mouse-clicks and pressing of keys... they make things happen.. such as a menu is drawn down, an inactive window is selected and activated..a menu item is hi-lited..etc. The operating system keeps a record of events...named
appropriately **eventrecord**. The next few lessons will deal with standard procedures which take care of all these events. Think of procedures as sections of code which you create and use over and over again...modular procedures which can be inserted into other programs.These lessons are very important. Take them slowly and subsequent lessons will not seem confusing...go through these lessons too fast and what follows will make little or no sense.

The first procedure is called **handleevent**. It uses a case statement to identify those things that can happen and what should be done as a result.

These procedures are fairly standard and are found in virtually all mac programs. We will cover these same procedures over the next few lessons. So if it doesn't make sense right away don't worry. Do ask questions. Some I will be able to answer, many i will not.

```
PROCEDURE HandleEvent(the event:eventrecord);       {this proc handles new events it checks the
                                                    eventrecord kept by the operating system. to see
if there is one waiting you use GetNextEvent}


begin

case theevent.what of                               {case of event and what should be done}

mousedown   :domousedown(theevent);                   {mousebutton pushed}
keydown       :dokeypress(theevent);                  {key pressed down}
autokey       :dokeypress(theevent);                  {key held down}
updateevt     :doupdate(theevent);                    {window need updating, it has been resized}
activateevt  :doactivate(theevent)                    {window made active or inactive}
end
end;                                                {end of proc handleevent}
```

This procedure leads to other procedures as you will see. Take for instance **"domousedown"**, that is a procedure which will be invoked if the eventrecord shows a mousedown, like wise dokeypress, doupdate, and doactivate arealso procedures.

These Pascal lessons are prepared specifically for use by members of the Arizona Macintosh Users Group. Please feel free to use and distribute them. Macintosh is a trademark of Apple Inc. Turbo Pascal is a trademark of Borland International Inc.

These procedures are shown below, one at a time.

```
PROCEDURE domousedown (theevent:eventrecord);            {deals with mouse clicks in different areas}

var
location        : integer;
thewindow   : windowptr;
mloc            : point;
wloc            : integer;

begin

mloc:= theevent.where;                                   {get mouse postion}
wloc:= findwindow(mloc,thewindow);                       {get window,window location}


case wloc of                                             {handle mouse locations}

inmenubar       :handlemenu(menuselect(mloc));               {inside the menubar}
incontent       :handleclick(thewindow,mloc);                {inside the window}
ingoaway        :handlegoaway(thewindow,mloc);               {on the go away box}
ingrow          :handlegrow(thewindow,mloc);                 {in the grow box}
indrag          :dragwindow(thewindow,mloc,dragarea);        {in the drag bar}
insyswindow     :systemclick(theevent,thewindow);            {in a desk accessory window}

end {of case wloc}
end; {of proc mousedown}
```

In the above procedure, 'domousedown', a case statement is used once again.The cases are inmenubar, incontent, ingoaway (the close box - the upper left of a window), ingrow (the resize box at the lower right of a window) etc. The result of each case item is after its colon. If your intuition told you these are probably procedures too, then give yourself a pat on the back! yes the terms handlemenu, handleclick, handlegoaway etc., are also procedures. Look them over. We won't cover them rightaway but get used to the terms. You will see these terms often. What we wil do instead is cover all the procedures required to fulfill handleevent.

The second procedure required by handleevent is dokeypress. This procedure is invoked by either keydown or autokey, where keydown is simple a key is pressed once...and autokey is where a key is held down.

These Pascal lessons are prepared specifically for use by members of the Arizona Macintosh Users Group. Please feel free to use and distribute them. Macintosh is a trademark of Apple Inc. Turbo Pascal is a trademark of Borland International Inc.

```
---------------------------------------------------------------------
procedure DoKeypress(theEvent : EventRecord);       { handles keypress (keyDown, autoKey) event}

var
 KeyCh : Char;

begin

 if (theEvent.modifiers and cmdKey) <> 0 then begin            { menu key command }
   KeyCh := Chr(theEvent.Message and charCodeMask);        { decode character }
   HandleMenu(MenuKey(KeyCh))                                 { get menu and item}
 else SysBeep(1)                                             { do *something*  }
end;                                                         { of proc DoKeypress }


---------------------------------------------------------------------
```

Next is the procedure doupdate.

```
---------------------------------------------------------------------
procedure DoUpdate(theEvent : EventRecord)                { handles window update event}

var
 SavePort,theWindow    : WindowPtr;

begin
 theWindow := WindowPtr(theEvent.Message);             { find which window     }
 if theWindow = MainPtr then begin                      { only update ours      }
   SetCursor(CursList[watchCursor]^^);                 { set cursor to watch   }
   GetPort(SavePort);                                  { save current grafport }
   SetPort(theWindow);                                 { set as current port   }
   BeginUpdate(theWindow);                               { signal start of update}

 { and here's the update stuff! }

 ClearWindow(theWindow);                                 { do update stuff }

{ now, back to our program...}

   EndUpdate(theWindow);                                   { signal end of update  }
   SetPort(SavePort);                                  { restore grafport  }
   SetCursor(Arrow)                                    { restore cursor }
 end
end; { of proc DoUpdate }
---------------------------------------------------------------------
```

Followed by doactivate

```
---------------------------------------------------------------------
```

These Pascal lessons are prepared specifically for use by members of the Arizona Macintosh Users Group. Please feel free to use and distribute them. Macintosh is a trademark of Apple Inc. Turbo Pascal is a trademark of Borland International Inc.

```
PROCEDURE  DoActivate(theEvent : EventRecord);              {handles window activation event}

var
 I              : Integer;
 AFlag          : Boolean;
 theWindow : WindowPtr;

begin
 with theEvent do begin
   theWindow := WindowPtr(Message);                          { get the window}
   AFlag := Odd(Modifiers);                                 { get activate/deactive }
   if AFlag then begin                                      { if it's activated...  }
     SetPort(theWindow);                                    {   make it the port }
     FrontWindow := theWindow;                               {   know it's in front }
     DrawGrowIcon(theWindow);                                { set size box}
   end
   else begin
     SetPort(ScreenPort);                                   { else reassign port }
     if theWindow = FrontWindow                              { if it's in front }
       then FrontWindow := NIL                               { ...then forget that}
   end;
   if theWindow = MainPtr then begin                          { if it's our window }
     SetItemState(EM,1,not AFlag);                           { update edit cmds }
     for I := 3 to 6 do
       SetItemState(EM,I,not AFlag);
     SetItemState(EM,8,AFlag);                               { update Quit command }
     for I := IM to DM do                                    { update other menus}
       SetItemState(I,0,AFlag);
     DrawMenuBar                                             {update menu bar }
   end
 end
end; { of proc DoActivate }
```

Thats it for the first round of procedures. Two of the handleevent procedures require even more procedures (domousedown and dokeypress).
There is a lot of terminology in this lesson, much of it is explained in the comments...what is not explained we will have to take as is and find out more about it later.

Good luck! and hang in there!

These Pascal lessons are prepared specifically for use by members of the Arizona Macintosh Users Group. Please feel free to use and distribute them. Macintosh is a trademark of Apple Inc. Turbo Pascal is a trademark of Borland International Inc.