Built-in AppleEvent Handlers

System 7 Pack™ installs several AppleEvent handlers which allow certain applications, such as Frontier™, to request information from 4D®. In addition, it installs a handler for the 'do script' AppleEvent which allows another application to execute most 4D® commands. This event handling is OFF by default. Call AllowAccess(1) to allow these events to be processed. When these events are disabled, the sending application will get back an error message “AppleEvents not allowed at this time.”

If you’re using a pre-3.0 version of 4D®, use extreme caution when sending any command to 4D® which could change the current record or selection. In particular, moving to a different record while a record is being modified in an input layout could cause changes to be lost. For this reason, I strongly suggest that AllowAccess(0) should be called before doing a Modify Selection or any other command which modifies or creates records. In version 3.0 or later of 4D, all AppleEvent handlers run in a separate process called “AEHandler”, so AppleEvents can change the selection or current record without affecting any other process. Since one process is used for all AppleEvent processing, a series of AppleEvents can act on a current record or selection.

For a more complete description of the AppleEvents handled by System 7 Pack™, see Appendix A.

Sending AppleEvents

System 7 Pack™ includes several high-level commands which make it easy to send AppleEvents.

Before you can send an AppleEvent, you must create a target address. The address can be selected from a list of programs on the network using the SelectAddress command, or created with the commands MakeAddress (if the program resides on your machine) or StringToAddr.

Once you have a target address, the easiest way to send an AppleEvent is by using the high-level command AESend, which lets you specify the class and ID as well as a message which can consist of up to 32K of text. You can also send a picture instead of text by using AESendPict.

If the receiving application will return any meaningful information in response to the AppleEvent, you can use the command SendWithReply instead. NOTE: SendWithReply isn’t intended to be used when sending a user-defined event to another 4D® 2.2.3 database, although it can be used to send one of the pre-defined events which return information. SendWithReply can be used to send an event to a 4D® 3.0 database which returns a reply.

If you need to send an event with more than one parameter or additional data types such as file aliases, object specifiers, or arrays, you must use the low-level interface.

Sending an AppleEvent with the low-level functions requires three steps: First, you must create the event using CreateAEVT. Next, you must add any needed parameters using commands such

as PutObject, PutAliasParam, PutTextParam, or PutList. Finally, you must use SendAppleEvent to actually send the event. When you're finished, you must dispose the event and the reply using DisposeDesc. For more information, see the section "Low Level Interface."

Example 1: Sending a "do script" using high-level commands (the DoScript & Frontier commands can simplify this even more)

```
$Err:=MakeAddress("LAND";$Addr)
if ($Err=0)
   $Err:=AESend($Addr;"misc";"dosc";"message("hello")")
end if
```

Example 2: Using the low-level interface to send a "Get Data" event and access the reply

```
  $Err:=MakeAddress("DFB0";$Addr)
if ($Err=0)
  $Err:=CreateAEVT("core";"getd";$Addr;$myAEVT)
  if ($Err=0)
    $Err:=PutObject($myAEVT;ObjNamed("cMPV";0;"MPVariable"))
    if ($Err=0)
      $Err:=SendAppleEvent($myAEVT;$myReply;kAEWaitReply;-1)
      if ($Err=0)
        $Err:=GetTextParam($myReply;"----";Result)
        $Err:=DisposeDesc($myReply)
      end if
    end if
    $Err:=DisposeDesc($myAEVT)
  end if
  $Err:=DisposeDesc($Addr)
end if
```

Receiving AppleEvents

In order to receive an AppleEvent, you must install a handler using the function HandleAEVT. A handler is a normal 4D® procedure which will be executed when an AppleEvent of the specified class and ID is received.

System 7 Pack™ provides several commands which allow an AppleEvent handler to extract information from the received event. To obtain the address of the sending application, use GetReturnAddr. The resulting target address can be used to send an event back to the same application. To determine the data type of the direct object, use GetAEType.

The easiest way to extract the direct object if it is text or a numeric value is by using GetAEMessage. To obtain a picture that was sent using AESendPict, use GetAEPict. NOTE: GetAEPict should only be used to extract a picture that was sent via AESendPict. In all other cases, you should use the low-level function GetPicParam.

If the received event contains additional parameters or uses other data types, you must use the low-level functions, such as GetTextParam, GetAliasParam, or GetList.

Note for 4D v3.0.x: All AppleEvent handlers run in a single process called Apple Event Manager. This process has no window associated with it unless a procedure explicitly creates a window. You must never call MESSAGE from such a procedure without first creating a window. Because of the new AppleEvent handling in 4D v3.0.3 and later, you can now send a reply by using the low-level commands and using -1 to specify the AppleEvent.

Example 1: Using the high-level functions to receive text & send acknowledgement

```
GetAEMessage($theMessage)
GetReturnAddr($who)
Alert($theMessage)
$Err:=SendAEVT($who;"TEST";"DEMO";"OK, I got the message!")
$Err:=DisposeAddress($who)
```

Example 2: Using the low-level functions to receive an array and create new records

```
$Err:=GetList(0;"----";anArray)
if ($Err=0)
   ArrayToSelection(anArray;[File1]Field1)
End if
```

Example 3: Waiting for an event to be received (Note that in 4D 3.0.x you must use interprocess variables.)

```
HandleAEVT("aevt";"xxxx";"SetFlag")
◊EventReceived:=False ` the event handler will change this
while (Not(EventReceived))
   idle
   ProcessAEVT
end while

` Procedure SetFlag -- executed in response to AppleEvent
◊EventReceived:=True
```