Appendix A: Built-in Event Handling

System 7 Pack™ has built-in handlers for the following AppleEvents:

'misc','dosc'

This is the standard 'do script' command. It will execute any 4D command sent with it. An optional parameter 'ACK0', which should be 1 or 0 specifies whether an acknowledgment should be sent after the command executes.

'ISIS','CFIL'
Count Files - returns the number of files.

'ISIS','CFLD'
Count Fields - returns the number of fields in the file specified by the direct object which must be numeric (either long or short integer).

'ISIS','FNAM'
File Name - returns the name of the file specified by the direct object which must be numeric.

'ISIS','FINF'
Field Info - returns the name and type of a field. The file number is specified by the direct object and the field number is given by the key 'FNUM'. Both of these must be numeric. The field name is returned as the direct object of the reply and the field type (an integer as returned by 4D's "type" function) is returned with the key 'FTYP'.

'ISIS','RECS'
Returns the number of records in the selection of the file specified by the direct object.

'ISIS','RECN'
Returns the current record number of the file specified by the direct object.

'ISIS','GETF'
Returns the contents of a field in the current record. The direct object specifies the file number and the field number is given by the key 'FNUM'.

'ISIS', 'ACK0'
This is optionally sent by 4D® as an acknowledgment after a 'do script' command is completed. The direct object is a numeric result code with any 4D® error that occurred when executing the command.

All of these events will return a descriptive error message in the 'errs' parameter if any errors occur. The reply will be one of the following: "Bad File Number", "Bad Field Number", "Bad Record Number", or "No Current Record". In order to accept these AppleEvents, you must execute the command AllowAccess(1).

Appendix B: Sample Programs

Your System 7 Pack™ disk includes several demo programs which illustrate how to use most of the features.

S7P Demo DB is System 7 Pack's on-line reference and demo database. It includes a complete listing of all commands in System 7 Pack in a HyperCard-like format in addition to demos of most of System 7 Pack's features. There are also several useful 4D procedures which you can copy to your own databases. See Appendix D for a description of several functions which create and send some of the core AppleEvents.

Here's a brief description of the menu commands:

•
List Commands: Displays a list of System 7 Pack's commands. Double-clicking on a command name will open up a HyperCard-like information window.

•
Export to Word: Creates a list of System 7 Pack's commands as a formatted MS Word® file and then uses System 7 Pack to open the document in Word.

•
Frontier Demo: Sends a few simple commands to UserLand Frontier™, which must be already running.

•
AppleScript Demo: Allows you to type in a script and execute it with AppleScript.

•
Finder Demo: Demonstrates how to send AppleEvents to the Finder.

•
Excel 4.0: Demonstrates how to send data to Excel® 4.0, request different types of charts in Excel, and convert Excel charts to a picture in 4D.

•
Send & Recv List: Demonstrates how to send and receive arrays using the low-level functions. For the receiver, you must select either your own copy of 4D or another one on the network running this same demo.

•
On Location: Demonstrates how to send AppleEvents to On Location®.

•
ISIS Notes Demo: Demonstrates how to control ISIS Notes™ by sending AppleEvents to look up users, send a note, and send a file.

•
Client/Server menu: Implements a name & address lookup server. Use one of the client applications described below to send lookup requests.

Address Client demonstrates one way 4D® can provide data to a client application. The 4D application provides a minimal address file which responds to AppleEvents (class "ISIS" id "FIND") which request a name lookup. Any text passed via the lookup request will trigger a Search on the name field of the address file. The full address will be returned in an event of class "ISIS" and id "REPL". Address Client is a tiny (28k) stand-alone application which simply sends lookup requests to the 4D® application and displays the results. Address Client Stack is a

HyperCard® 2.1 stack which works exactly the same as the application. FM Pro Client is a sample client written in FileMaker® Pro 2.0. This one sends a different request to 4D and gets back a series of Create Element events to create records containing the results.

Appendix C: Error Codes

All System 7 Pack™ functions will return 0 if the operation is completed successfully. Some applications may return codes not listed here if they're unable to handle an AppleEvent sent to them. Also, any possible system error code can be returned (for a complete list, see Bill Steinberg's "System Errors DA"). Other possible results are:

-1 An invalid signature or target address handle was given
-2 You're not running System 7 or your system doesn't support AppleEvents
-3
The specified application isn't running.
-4
Incompatible array type (PutList or GetList).
-10
Handler for that event is already present (InstallAEVT).
-20
Attempted to replace standard event handler (InstallAEVT).
-30
Attempted to remove event handler we didn't install (IgnoreAEVT).
-35
Couldn't find application to launch on any volume.
-43
Couldn't find document to open or print.
-100
Unable to install AppleEvent handler proc (InstallAEVT).
-108
Not enough memory to launch an application.
-606
Attempt to bring background-only application to front. (BringToFront).
-906
Attempt to send AppleEvent to non-aware application. (usually PrintDoc).
-1700
Incompatible data type in an AppleEvent parameter.
-1701
Parameter not found in the AppleEvent.
-1702-1707
Not a valid AppleEvent or invalid parameter.
-1708
Receiving application couldn't handle that AppleEvent.
-1709
Reply wasn't valid.

-1710
Unknown send mode.
-1711
Wait for reply cancelled by user.
-1712
Timed out waiting for reply.
-1713
Required user interaction but none was allowed.
-1715
Required parameter wasn't accessed.
-1716
Invalid target address.
-1718
Attempted to access reply which hasn't arrived yet.
-1753
General AppleScript error (usually syntax error in script).

Appendix D: AppleEvent Registry

Fully describing all of the standard AppleEvents is beyond the scope of this manual, but here are some of the more common events defined by Apple in the required, core, and miscellaneous standard suites of the AppleEvent registry:

Open Application aevt
oapp
Sent by the finder when an application is opened with no documents. You shouldn't send this event.

Open Document
aevt
odoc
Tells an application to open a list of documents.

Print Document
aevt
pdoc
Tells the application to open and print one or more documents.

Quit
aevt
quit
Tells the application to quit.

Close
core
clos

Closes the specified objects.

Delete
core
delo
Deletes the specified objects.

Do Objects Exist
core
doex
Determines if the specified objects exist.

Get Class Info
core
qobj
Get information about a particular object class.

Get Data
core
getd
Get data from the specified objects.

Get Data Size
core
dsiz
Get the size of specified objects.

Get Event Info
core
gtei
Get information about a particular AppleEvent.

Save
core
save
Save the specified objects.

Set Data
core
setd
Change the specified objects.

Do Script
misc
dosc
Executes commands in the application's specific language.

Evaluate
misc
eval
Evaluates an expression and returns the results.

Here some of the more common class names and property IDs:

Cell
ccel
Column
ccol
Document
docu
File
file
Graphic object
cgob
Menu
cmnu
Paragraph
cpar
Row
crow
Selection
csel
Table
ctbl
Window
cwin
Word
cwor
Best type (property)
pbst
Bounds (property)
pbnd
Class (property)
pbnd
Color (property)
colr
Default type
deft
Font (property)
font
Name (property)
pnam

Point Size
ptsz
Version
vers

The sample database, S7P Reference, includes the following functions which create and send some of the more common core events:

Err:=Create Element(Target;Class;Container;Position)
Creates a new element of the specified class. Specify 0 for container and "" (a null string) for position if no value needs to be given.

Result:=Do ObjectsExist(Target;Object)
Returns TRUE if the specified objects exist.

Err:=Get Text(Target;Object;»data)
Err:=Get Pict(Target;Object;type;»data)
Err:=Get Array(Target;Object;type;»data)
Returns the value of an object or object property. Type should be a 4-letter string specifying the data type to be returned. It should be one of “TEXT”, “LIST”, “PICT”, or “SPIC”. “SPIC” is only used with Excel® 4.0 and is the same as PICT except it will return a color screen picture rather than a dithered print image picture. Data should be a pointer to a variable of the appropriate type.

NOTE: Rather than using a single procedure, we’ve provided separate procedures for the most common data types to make them compatible with the 4D compiler.

Err:=Send Text(Target;Object;Data)
Err:=Send Array(Target;Object;»Array)
Err:=Send Enum(Target;Object;Value)
Changes the value of an object or object property.

Examples:
Err:=Send Text(Excel;Property(“sele”;Obj(“docu”;0;1));”R1C1:R10C1”)
Selects cells R1C1 thru R10C1 in the topmost worksheet.

Err:=Send Array(Excel;ObjNamed(“crng”;Obj(“docu”;0;1);”R1C4:R10C4”);»aList)
Sends an array of numbers to a range in the topmost worksheet.

Err:=Get Pict(Excel;Obj(“chrt”;Obj(“docu”;0;1);1);”SPIC”;»aChart)
Copies the first chart in the topmost worksheet to a color picture.

Err:=Create Data(Excel;”chrt”)
Creates a new chart document.

HasChart:=Do ObjectsExist(Excel;Obj(“chrt”;Obj(“docu”;0;1);1))
Returns TRUE if the topmost worksheet contains a chart.

For more information, see the latest edition of the AppleEvent Registry, available from Apple Computer, Inc and the Excel Software Development Kit, available from Microsoft Press. Many AppleEvent aware applications include a description of the supported events in their manul. You may also be able to get additional information from the software publisher.

Appendix E: Network Access

Before you can send AppleEvents across the network you must configure any machines you wish to be able to access. Any machine which needs to receive remote AppleEvents must have "Program Linking" turned on through the Sharing Setup control panel. In addition, any machine you wish to connect to should have an entry in the Users & Groups file for you (optionally, if security isn't a concern, you can simply turn on the program linking checkbox for guests by double-clicking the <Guest> icon in the Users & Groups control panel). The first time you send an AppleEvent to a program on a remote Macintosh you will be asked to suply a user name and password. If guest access is enabled, you can simply click on the "Guest" button.

Appendix F: Quick Reference

L :=
AddrToString(Target;String1;String2;String3)
L :=
AESend(Target;Class;ID;Text)
L :=
AESendPict(Target;Class;ID;Picture)

AllowAccess(N)
L :=
AppleScript(Text;Reply)
L :=
BringToFront(Signature)
L :=
CreateAERec(AERec)
L :=
CreateAEVT(Class;ID;Target;AEVT)
L :=
CreateXAEVT(Class;ID;Target;N;L;AEVT)
L := Coerce(AERec;Str4)
ospec := Comparison(Str4;container ospec;test ospec;Str4;String)
L :=
CopyDesc(AEVT or Target or OSPEC or AERec)
L :=
DisposeAddress(Target)
L :=
DisposeDesc(AEVT or Target or OSPEC or AERec)
L :=

DoScript(Target;Text)
L :=
Evaluate(Target;Text;Reply)
L :=
FindAppName(Signature;Name)
L :=
FindCreator(Name;Signature)
L :=
FinderOpen(Name);
L :=
Frontier(Text;Reply)
L :=
GetAEInfo(AEVT;NumOfItems;ArrayOfStr4;ArrayOfStr4;ArrayOfLong)

GetAEMessage(Text)
L :=
GetAEPict(Picture Variable)
L :=
GetAERecord(AEVT;Str4;AERec)

GetAEType(Type String)
L :=
GetAliasParam(AEVT ;Str4;String)
L := GetBoolean(AEVT;Str4;aBoolean)
L :=
GetComponent(Str4)
L :=
GetKeyword(AEVT;Str4;Str4)
L :=
GetList(AEVT;Str4;anyArray)
L :=
GetNthDesc(AEVT;integer;Str4;Str4;Long)
L :=
GetNthItem(AEVT;integer;Str4;Str4;Text)
L :=
GetRealParam(AEVT;Str4;aRealNum)
L :=
GetReturnID(AEVT)
L :=
GetShortParam(AEVT;Str4;Str4;L)
L :=
GetTransactionID(AEVT)
L :=
GetPicParam(AEVT;Str4;N;aPicture)

GetReturnAddr(Target)

L :=
GetShortParam(AEVT;Str4;Str4;N)
L :=
GetTextParam(AEVT;Str4;Text)
L :=
HandleAEVT(Class;ID;Name)
L :=
IgnoreAEVT(Class;ID)
L :=
IsRunning(Signature)
L :=
IsVersion3
L :=
Launch(Signature;Name)
L :=
LaunchBehind(Signature;Name)
L :=
ListComponents(Str4;Long;StrArray;TextArray)
L :=
Long(Str4)
L :=
MakeAddress(Signature;Target)
ospec :=
Obj(Str4;ospec ;L)
ospec :=
ObjNamed(Str4;ospec;String)
ospec := ObjX(Str4;ospec;Str4)
L :=
PrintDoc(Signature;Name)

ProcessAEVT
L :=
ProcessList(N;ArrayOfString)
ospec :=
Property(Str4;ospec)
L :=
PutAERecord(AEVT;Str4;AERec)
L :=
PutAliasParam(AEVT;Str4;Name)
L := PutBoolean(AEVT;Str4;aBoolean)
L :=
PutKeyword(AEVT;Str4;Str4)
L :=
PutList(AEVT;Str4;Str4;anyArray)
L :=
PutLongParam(AEVT;Str4;Str4;L)

L :=
PutObject(AEVT;Str4;ospec)
L :=
PutPicParam(AEVT;Str4;N;aPicture)
L :=
PutRealParam(AEVT;Str4;aRealNum)
L :=
PutShortParam(AEVT;Str4;Str4;N)
L :=
PutTextParam(AEVT;Str4;Text)
L :=
SendAppleEvent(AEVT;ReplyAEVT;L;L)
L :=
SetComponent(Str4)
L :=
QuicKeys(Name)
L :=
QuitApp(Signature)
L :=
S7Version(Name)
L :=
SelectAddress(Name,Signature;Target)
L :=
SendWithReply(Target;Class;ID;Text;Reply)

SetTimeOut(N)
L :=
StringToAddress(String1;String2;String3;Target)
L :=
System7

L:
a Long Integer value
N:
an Integer value
Name:
a string variable or field
Signature:
4 character application signature
Class,ID:
4 character strings
Text:
a Text variable or field
Reply:
a Text variable or field
AEVT:
a Long Integer representing an AppleEvent
AERec:
a Long Integer representing an AERecord
ospec:
a Long Integer representing an Object Specifier

Target:
a Long Integer representing a target address
ospec:
an object specifier

Appendix G: Version History

Version 1.0 - Dec. 1991 Initial release.

Version 2.0 - Feb. 1992 Added Frontier & QuicKeys support, Built-in AppleEvent handlers.

Version 3.0 - May 1992 Added low-level commands. Many new commands added but compatibility maintained with 2.0. Procedure Editor now lists commands in logical groupings. Demo programs completely rewritten.

Versoion 3.1 - June 1992 Added AE Record support, needed for Claris Resolve.

Version 3.1.1 - June 1992 Fixed problem with StringToAddress

Version 3.2 - July 1992 Added Long() utility function & demos for Excel 4.0

Version 3.26 - Sep. 1992 Added LaunchBehind command. Added many new commands for examining AppleEvents and provide more access to internal features for custom extensions to S7P.

Version 3.3 - Nov. 1992 Compatibility with 4D® Version 3.0.

Version 3.4 - Apr. 1993 Compatibility with 4D® 3.0.3's new AppleEvent handling & support for AppleScript.

Version 3.4.2 - June 1993 Minor compatibility changes for final release of 3.0.3 & 3.0.4. Maintains compatibility with 4D 2.2.3 but removes 3.0.1/3.0.2 compatibility.

Version 3.4.3 - August 1993 Adds FinderOpen command for cleaner application launching. PutList now accepts a 2 dimensional array. PutAlias now supports remote aliases.

Version 3.5 - Sep. 1993 Added GetBoolean & PutBoolean commands. Improved AppleScript and Open Scripting Architecture support. AppleScript command now returns the script result. SetComponent & GetComponent commands provide a way to change the default scripting language. ListComponents can get information about components of any type.

Version 3.6 - Nov. 1993 Added Coerce, ObjX, and Comparison commands to simplify certain tasks in Now Up-To-Date and FileMaker Pro.