Appendix

Frequently Asked Questions(FAQ)

1) Must I always open a data file with the standard get file dialog?

Both the 'OpenCollection' and 'CreateCollection', as well as the 'DBInit' and 'DBOpen' commands have a optional parameter for the complete file path. No dialog is required if this parameter is used.

2) Does the Jovis splash screen always appear?

The Jovis splash screen is required for non-licensed use. A non-licensed version is the SDK (Software Development Kit) version. If you have commercial or in-house products, you are required to purchase a license, which is $350 USD. This is a one time fee for as many products as you create. There are NO royalties. With a licensed version, the splash screen does not appear.

3) Is the prompt dialog during searching required?

You can set the "Delay" property to zero at anytime using the 'SetProperty' command. This will prevent both the prompt and progress dialogs from appearing.

You can also modify both the progress and prompt dialogs using ResEdit. All of the commands that cause one of these dialogs to appears, includes a parameter for a custom prompt messages.

5) Can I mask the Jovis error dialog and get the error in a variable ?

Yes. You can turn off the error dialog system, and use the "JovisErrorCode" global variable. Using the 'SetProperty' command, set both the "ErrorMsg" and "WarningMsg" properties to empty, or change them to custom script handlers of your own. Keep in mind that you must always check for errors and warnings, this is how Jovis communicates to you.

All of the Jovis error code messages are supplied in a 'Str#' resource. This will allow you to change not only the language, but also the actual wording for the messages. Version 1.0.3 of Jovis is not completely language independent, however, we are working quickly towards this goal and hope to reach it by version 1.5.

6) When Importing or Exporting, can I use a dialog that I have created? Also, what about specifing fields and separators.

Both of these commands can accept a full file path name. You can indicate any of the relation's fields to export, and use any separators you require.

7) What is the difference between the "Relational" and "Architectural" capabilities?

The Architectural commands are very fast due to the fact that there is no overhead in processing. All that you are doing is finding a key and reading the record; it is more of a "nuts-n-bolts" approach. If you have records which are unique, such as pictures, sounds, or text blocks the Architectural capability will be a perfect match. Architectural commands are commonly referred to as "Flat File" commands. However, Jovis adds many enhancements and capabilities that make these commands far more useful and beneficial than ordinary "Flat File" utilitiy routines.

With the relational, there's the added processing of field information and interface features. So, there are some preformance considerations, even though it is very fast. If you're data falls into a tabular format, meaning many records with the same fields, then the relational capability is for you. Also, the relational version should be used if you need to search with criteria and obtain subsets of your database. A subset could be as small as a single field from one record, or hundreds of entire records. The deciding factor will be the data and the way your end-users need to work with it.

8) You have a handler written for getFieldList but you still have to manually list the fields. Is there a handler which will automatically get the field list from the database so you don't have to have a special handler for each and every table. I will have 10 to 15 tables and am trying to cut the code down to where I can use the same script for all the tables. I would like a way to get the data fromthe DB other than having 15 scripts listing the fieldNames.

Try using the "ListFields" command. The field names and their type are returned via a return delimiter list. You'll need to extract the names from this list. For example:

```
function getFieldList2 RelationName
    put jovis("ListFields","myDB",RelationName) into dbFlds
    repeat with x = 1 to number of lines of dbFlds
       put item 1 of line x of dbFlds into item x of fldNames
    end repeat
    return fldNames
end getFieldList2
```

Jovis Error Codes

Error Codes
0 No Error.

Architectural Warnings

5 File is open read-only.
6 Cancel selected by user.
11 Key is not an exact match.
13 Key truncated. Key is larger than the one defined by keyset.
14 Key padded. Key is smaller than one defined by keyset.
18 Memory Low No Reserve. Save and quit application.

Architectural Errors

20 Not enough parameters passed for command to be executed.
21 Not a valid global name.
22 Not a valid data file. Selected file is not a Jovis data file.
23 Not a valid command. Possible typo of command name..
31 No current key was set. The command failed.
32 Duplicate key. Keyset as defined does not allow duplicate keys.
33 Tried to define a key size greater than 255.
34 Tried to define a key size of less than 1 byte.
35 Key is pointed to by a marker and cannot be used.
36 Failed to find newly created key.
37 Null keys not allowed.
40 Current record is not current key's record.
41 No record exists for the current key.
42 A record already exist for this key.
44 Attempted to write a record of less than one byte.
45 Record is not a text record.
46 Keyset not found.
47 Keyset not in memory.
48 Keyset not in freed list.
49 Internal Keyset. Internally created keyset cannot be edited.
50 Keylen change not allowed. The given length for a keyset is not valid.
51 Keyset not defined. Current key does not point to a keyset.
52 Keyset exists. The current key points to a keyset.
53 End of keyset. No more keys exist, or the keyset is empty.
54 Keyset not empty. Attempted to delete a keyset that contains keys.

55 Keyset not allowed. The current keyset does not allow sub-keysets.

56 Keyset name is empty.

57 Keyset ID is invalid.

58 Invalid keyset name.

59 Failed to change keyset name.

60 Duplicate Keyset name. Keyset names must be unique.

61 No current keyset.

62 Can not change a relational record or index.

66 Empty database. No keysets exist.

67 The current keyset is already the top most keyset.

71 Marker not found. No marker by the given name or ID was found.

72 Invalid marker. The marker path has been altered.

73 Duplicate marker name. Marker name already exists.

74 Too many markers.

75 Invalid marker type.

76 No markers found.

77 Invalid marker name.

78 A marker exists for this key.

80 Unable to read requested resource.

81 Database file already open.

82 No resource name. Cannot locate a resource by the given name.

83 No resource ID. Cannot locate a resource by the given ID number.

84 A resource required by the database system cannot be found.

85 Requested resource type does not exist.

86 A resource by the given name or ID already exists.

90 Nothing selected.

91 Invalid coordinates for resource selection list.

93 Insufficient system. Required operating system features not present..

94 Wrong OS version. The currently running version of the operating system is not supported.

95 Wrong version. Attempted to open a database file that is incompatible with the database program.

97 Not enough memory to continue.

98 Memory error. Insufficient memory to continue.

101 Invalid password.

102 Invalid creator type.

110 Not a PICT file. Selected file does not contain PICT information in data fork.

111 Not a valid Pict global name.

112 Invalid rectangle. Rectangle coordinates for PICT are invalid.

113 Not a valid Pict record.

120 The ‘snd ’ has completed.

121 The ‘snd ’ is still playing.

122 Not a valid ‘snd ’ global name.

123 No ‘snd ’ in memory.
124 Not a valid ‘snd’ record.
130 Not a valid BLOB record.
131 No BLOB was selected.
132 Not a valid BLOB global name.
133 Invalid parameter information.

Relational Warnings

200 Cancel selected from dialog.
201 No more records satisfy search criteria.
202 One or more fields were missing or invalid.
203 One or more buttons were missing or invalid.
204 Total selection exceeds 32000 characters.
205 File is opened read-only.
206 Record does not exist.
207 No records selected.

Relational Errors

219 Field does not exist.
220 Field already exists.
221 Field is indexed. Delete index first.
222 Not a valid field type.
223 Field name too long.
224 Field name missing.
225 Field does not exist in this relation.
226 Not a valid field name.
227 Invalid field.
228 Invalid field name. Invalid characters, bad format, or is too long.
229 Hierarchical access not available.
230 Relational records cannot be changed using architectural commands.
239 Index does not exist.
240 Index already exists.
241 Invalid index exists. Delete invalid index first.
242 Invalid Index.
243 Invalid Index name. Invalid characters, bad format, or is too long.
244 Invalid index field name. Invalid characters, bad format, or is too long.
259 Relation name too long.
260 Relation name missing.
261 Relation name already exists.
262 Invalid relation name. Invalid characters, bad format, or is too long.

263 Not a valid relation name.
279 Invalid selection field name.
280 Cannot select more that 8000 fields per record.
281 No selection set for indicated relation.
282 Invalid operator.
283 Unbalanced quotes or brackets.
284 Missing field name.
285 Invalid field name.
286 Missing background field name.
287 Invalid background field name.
288 Missing card field name.
289 Missing operand.
290 Incorrectly placed '('.
291 Unbalanced parenthesis.
292 Incorrectly placed 'NOT' operator.
293 Incorrectly place 'AND' operator.
294 Incorrectly place 'OR' operator.
295 One of the operands must be a field name.
296 Invalid selection row number..
300 File not initialized.
309 Not enough memory for sort.
310 Error while allocating sort table.
319 Global variable name is missing or invalid.
320 Global variable name was not initialized correctly.
321 Global variable name is not a valid Handle.
329 Cannot import more than 1024 fields per record.
330 Cannot export more than 1024 field per record.
331 Number of given fields does not equal number of input field delimiters.
332 Invalid field and/or record delimiter.
339 Not enough parameters.
340 Parameter one is empty.
341 Invalid character.
342 Not a relational data file.
343 Invalid record.
344 Required resources missing.
345 Not enough memory.
346 No property by that name.

Client/Server Errors

400 File not found. Access to the file has been denied from the server.
401 Invalid password. Given password is not valid for the selected entity.

402 Server not responding.
403 Buffer too small. Error at server in transmitting a record.
404 Already at end of record. Error at server in transmitting record.
405 Record locked by another user. Record cannot be updated at this time.
406 File has been closed. File closed at server.
407 Record not locked.
408 File in use. File has been opened in exclusive mode.
409 Already handling maximum number of authorized users .
410 Maximum locks for file reached..
411 File already open. Tried to create a file with same name as one already open at the server.
413 File opened exclusively by administrator.

Circular List Manager

501 Circular list empty.
502 Circular list bad item.
503 Circular list bad data.
504 Circular list invalid nth item.
505 Circular list data not found.
506 Circular list found end of list.

Disk Manager

600 Disk request too large. Disk Mgr.
601 Invalid File Mgr. address. Disk Mgr.
602 Disk allocation too small. Disk Mgr.
603 Disk Deallocation too small. Disk Mgr.
604 Address less than custom area. Disk Mgr.
605 Start disk manager failed. Disk Mgr.

Compression Errors

650 Compress failure.
651 Decompress failure.
652 Compression input too small.
653 Compression input invalid.
654 Compression repeating char are invalid.

Memory Manager

700 Memory allocation error. Mem. Mgr.
701 Memory Deallocation error. Mem. Mgr.

702 Memory Reallocation error. Mem. Mgr.
703 Memory under limit. Mem. Mgr.
704 Memory over Limit. Mem. Mgr.
705 Bad Deallocation. Mem. Mgr.
706 Bad memory. Mem. Mgr.
707 Requested memory greater max allowed.

Document Indexing Errors

750 Document Indexing: Failed to open file.
751 Document Indexing: Initialization failed.
752 Document Indexing: Empty file.

Queue Manager

800 Queue empty. Queue Mgr.
801 Queue full. Queue Mgr.
802 Nil address. Queue Mgr.
803 Bad memory. Queue Mgr.
804 Queue allocation failed. Queue Mgr.

Memory Manager

901 Node in memory is nil. Index Mgr.
902 Node in cache nil. Index Mgr.
903 Node in block nil. Index Mgr.
904 Cache overflow. Index Mgr.
905 Unable to access node. Index Mgr.
906 Cache memory address is empty.

Extended ASCII Table

This appendix lists the character assignments for the 256 single-byte character values used by the Macintosh.

There are 256 possible values of the ASCII character set. The first 128 have been assigned to a standard set of characters and commands used in data processing and communication. These assignments form the ASCII character set. (*ASCII* stands for *American Standard Code for Information Interchange.*)

The remaining 128 values are <u>not</u> assigned in the ASCII standard. Because they have higher numerical values than the first 128 characters, they are often referred to as the high-ASCII characters.

This appendix lists all character values by their decimal equivalent.

Table B-1 lists the first 32 characters, the control characters, which have no printable-character representation, with the standard abbreviation, the hexadecimal notation, and their meaning.

Table B-2 lists the remaining 224 character values with the characters to which they are assigned in the Macintosh Courier font, and the hexadecimal notation for each.

Value	Name	Hex	Meaning
0	NUL	0000	Null
1	SOL	0001	Start of heading
2	STX	0002	Start of text
3	ETX	0003	End of text
4	EOT	0004	End of transmission
5	ENQ	0005	Enquiry
6	ACK	0006	Acknowledge
7	BEL	0007	Bell
8	BS	0008	Backspace
9	HT	0009	Horizontal tab
10	LF	000A	Line feed
11	VT	000B	Vertical tab
12	FF	000C	Form feed
13	CR	000D	Carriage return
14	SO	000E	Shift out
15	SI	000F	Shift in
16	DLE	0010	Data link escape
17	DC1	0011	Device control 1
18	DC2	0012	Device control 2
19	DC3	0013	Device control 3
20	DC4	0014	Device control 4
21	NAK	0015	Negative acknowledge
22	SYN	0016	Synchronous idle
23	ETB	0017	End of transmission block
24	CAN	0018	Cancel
25	EM	0019	End of medium
26	SUB	001A	Substitute
27	ESC	001B	Escape
28	FS	001C	File separator <--Used as field delimiter Reservered for Jovis use
29	GS	001D	Group separator <--Used as header delimiter Reservered for Jovis use
30	RS	001E	Record separator <--Used as record delimiter Reservered for Jovis use
31	US	001F	Unit separator

Value	Char	Hex	Value	Char	Hex	Value	Char	Hex
32	space	0020	66	B	0042	100	d	0064
32	!	0021	67	C	0043	101	e	0065
34	“	0022	68	D	0044	102	f	0066
35	#	0023	69	E	0045	103	g	0067
36	$	0024	70	F	0046	104	h	0068
37	%	0025	71	G	0047	105	i	0069
38	&	0026	72	H	0048	106	j	006A
39	‘	0027	73	I	0049	107	k	006B
40	(	0028	74	J	004A	108	l	006C
41	)	0029	75	K	004B	109	m	006D
42	*	002A	76	L	004C	110	n	006E
43	+	002B	77	M	004D	111	o	006F
44	,	002C	78	N	004E	112	p	0070
45	-	002D	79	O	004F	113	q	0071
46	.	002E	80	P	0050	114	r	0072
47	/	002F	81	Q	0051	115	s	0073
48	0	0030	82	R	0052	116	t	0074
49	1	0031	83	S	0053	117	u	0075
50	2	0032	84	T	0054	118	v	0076
51	3	0033	85	U	0055	119	w	0077
52	4	0034	86	V	0056	120	x	0078
53	5	0035	87	W	0057	121	y	0079
54	6	0036	88	X	0058	122	z	007A
55	7	0037	89	Y	0059	123	{	007B
56	8	0038	90	Z	005A	124	|	007C
57	9	0039	91	[	005B	125	}	007D
58	:	003A	92	\	005C	126	~	007E
59	;	003B	93	]	005D	127		007F
60	<	003C	94	^	005E	128	Ä	0080
61	=	003D	95	_	005F	129	Å	0081
62	>	003E	96	`	0060	130	Ç	0082
63	?	003F	97	a	0061	131	É	0083
64	@	0040	98	b	0062	132	Ñ	0084
65	A	0041	99	c	0063	133	Ö	0085

Value	Char	Hex	Value	Char	Hex	Value	Char	Hex
134	Ü	0086	168	®	00A8	202	**	00CA
135	á	0087	169	©	00A9	203	À	00CB
136	à	0088	170	™	00AA	204	Ã	00CC
137	â	0089	171	´	00AB	205	Õ	00CD
138	ä	008A	172	¨	00AC	206	Œ	00CE
139	ã	008B	173	≠	00AD	207	œ	00CF
140	å	008C	174	Æ	00AE	208	–	00D0
141	ç	008D	175	Ø	00AF	209	—	00D1
142	é	008E	176	∞	00B0	210	“	00D2
143	è	008F	177	±	00B1	211	”	00D3
144	ê	0090	178	≤	00B2	212	‘	00D4
145	ë	0091	179	≥	00B3	213	’	00D5
146	í	0092	180	¥	00B4	214	÷	00D6
147	ì	0093	181	µ	00B5	215	◊	00D7
148	î	0094	182	∂	00B6	216	ÿ	00D8
149	ï	0095	183	∑	00B7	217	Ÿ	00D9
150	ñ	0096	184	∏	00B8	218	⁄	00DA
151	ó	0097	185	π	00B9	219	¤	00DB
152	ò	0098	186	∫	00BA	220	‹	00DC
153	ô	0099	187	ª	00BB	221	›	00DD
154	ö	009A	188	º	00BC	222	fi	00DE
155	õ	009B	189	Ω	00BD	223	fl	00DF
156	ú	009C	190	æ	00BE	224	‡	00E0
157	ù	009D	191	ø	00BF	225	·	00E1
158	û	009E	192	¿	00C0	226	‚	00E2
159	ü	009F	193	¡	00C1	227	„	00E3
160	†	00A0	194	¬	00C2	228	‰	00E4
161	°	00A1	195	√	00C3	229	Â	00E5
162	¢	00A2	196	ƒ	00C4	230	Ê	00E6
163	£	00A3	197	≈	00C5	231	Á	00E7
164	§	00A4	198	∆	00C6	232	Ë	00E8
165	•	00A5	199	«	00C7	233	È	00E9
166	¶	00A6	200	»	00C8	234	Í	00EA
167	ß	00A7	201	…	00C9	235	Î	00EB

Value	Char	Hex
236	Ï	00EC
237	Ì	00ED
238	Ó	00EE
239	Ô	00EF
240		00F0
241	Ò	00F1
242	Ú	00F2
243	Û	00F3
244	Ù	00F4
245	ı	00F5
246	ˆ	00F6
247	˜	00F7
248	¯	00F8
249	˘	00F9
250	˙	00FA
251	˚	00FB
252	¸	00FC
253	˝	00FD
254	˛	00FE
255	ˇ	00FF

** Stands for non-breaking space.

Memory Requirements and Jovis Resources

The implementation of a virtual memory system in Jovis is how portions of different indexes can be in memory at the same time. This, of course, allows for large indexes and significant speed in accessing keys within an index. Indexes which are being accessed at a higher rate will have larger portions in memory than indexes which are not being accessed. If an index is not accessed significantly, it will be entirely removed from memory to allow other indexes to be brought into memory. The minimum amount of memory required by Jovis is 512K. The system will generally manage any size database using this minimum size. Obviously, the more memory you reserve for Jovis' indexes the better it's performance will be. The amount of memory you reserve, should be in proportion to the size of the database you are working with. Some databases have several indexes, but only two or three are accessed with any significance. Hence, a large memory allocation would be under used.

In addition to the index cache system Jovis dynamically uses a portion of the shell application's heap space for general purposes, such as selections, sorting, buffering to and from disk, etc. If there is not enough application heap space for these requirements, low memory warnings and out of memory errors will occur. There is no fixed amount of memory required, however at least another 512k is a general suggestion.

For the client version, the memory requirements are considerably less, due to the fact that the server handles all of the index cacheing. The application heap space requirment is 128K, for network functionality, and about 512K for general processing.

Jovis Resources

Below is a listing of the required Jovis resources. Several of these resources have attributes which are preset, and MUST not be changed. Also, do not change any of the resource IDs that are listed below.

Type	Name	ID
DASW	Jovis Engine	
DASW	Jovis Info	
DITL	Jovis PutFile	2001
DITL	Jovis GetFile	2002
DITL	Jovis OpenFile	4000
DITL	Jovis CreateFile	4001
DITL	Jovis Thermometer	4002
DITL	Jovis Progress	4003
DLOG	Jovis PutFile	2001
DLOG	Jovis GetFile	2002
DLOG	Jovis OpenFile	4000
DLOG	Jovis CreateFile	4001
DLOG	Jovis Thermometer	4002
DLOG	Jovis Progress	4003
STR#	Jovis Ref List	5300
STR#	Jovis Error Codes	5301
XFCN	Jovis	

Single Character Parameters

Char	Name	Example
!	Copyright	Jovis® Copyright © 1997 All Rights Reserved. DAS Works
@	Compilation Date	May 28 1997,19:52:12
%	Version	1.0.3

Example:

```
on mouseUp
   put Jovis("@") into msg  --> get compilation date and time
end mouseUp
```

Accessing a Blob in C

```
/*

   Blob_XFCN.c

   Sample XFCN for accessing Blobs which have been stored/retrieved
   from a Jovis database.  Useing MetroWerks DR11.

*/


#include <A4Stuff.h>
#include <TextUtils.h>
#include <HyperXCmd.h>


/* Usage:
      on mouseUp
         put Blob_XFCN("myBlob") into ErrorResult
      end mouseUp
*/


/* main entry point */
pascal void main( XCmdPtr paramPtr )
{
   Str255   strGloName, str;
   long     L;
   Handle   BlobHdl,UtilHandle;
   char     *cp;
   short    i = 0;

   EnterCodeResource();                // see A4Stuff.h

   ZeroToPas(paramPtr, paramPtr->params[0], strGloName);    // get global name
   if(*strGloName == 0) {              // sanity test
      paramPtr->returnValue = PasToZero(paramPtr, "\perror,invalid global name");
      return;
   }
   UtilHandle = GetGlobal(paramPtr, strGloName);
   HLock(UtilHandle);
   cp = (Byte *)StripAddress(*UtilHandle);
   while( *cp != (char)',' && *cp != (char)'\0' )           // scan up to comma
      cp++; i++;
   cp++; i++;                 // bump past the comma
   BlockMove(cp,&str[1],(GetHandleSize(UtilHandle) - i) -1); // less the null
   *str = GetHandleSize(UtilHandle) - i;
   HUnlock(UtilHandle);
   DisposeHandle((Handle)UtilHandle);
   StringToNum(str,&L);
   BlobHdl = (Handle)L;                // This is the actual Blob handle
   /*
    *
    *    Now proceed using the BlobHdl as needed.
    *
    */


   /* If you want to dispose the BlobHdl Handle yourself, rather than using
      the Jovis 'ClearBlob' command, you can do the following: */

   DisposeHandle((Handle)BlobHdl);
   UtilHandle = PasToZero(paramPtr, "\pJovis");
   SetGlobal(paramPtr, strGloName, UtilHandle);
   DisposeHandle((Handle)UtilHandle);

   paramPtr->returnValue = PasToZero(paramPtr, "\pno error");

   ExitCodeResource();                 // see A4Stuff.h
}
```