

HIFF FORMAT SPECIFICATION

HYPERMEDIA INTERCHANGE FILE FORMAT

The Hypermedia Interchange File Format™ (HIFF™) standard is a specification for comprehensively describing the form and contents of documents created in and by object-oriented, multimedia or hypermedia application programs. Its purpose is to facilitate the transfer of such documents for use in other environments with similar capabilities.

This document details the HIFF specification.

Revision history

Version 1.0 — *17 July, 1990:*

This is the initial release of the HIFF specification.

Who can use HIFF

HIFF is an open standard. Anyone may develop and distribute products that write or read documents in the HIFF format.

This document is © **Copyright The HyperMedia Group, 1990**. You may copy and redistribute it freely, provided you do so only in its entirety, including this section.

The names Hypermedia Interchange File Format and HIFF are trademarks of The HyperMedia Group. You can obtain a royalty-free license to use these trademarks by certifying that your product is fully compatible with published HIFF standards. (See *Contacts* information below.)

The products, algorithms, and techniques developed by The HyperMedia Group and its licensees for writing and reading HIFF files are proprietary and may be used only by those who have purchased or licensed them from The HyperMedia Group or its authorized resellers or relicensors.

Contacts

For information about HIFF, or to make suggestions or proposals regarding future versions of HIFF, please address correspondence to:

The HyperMedia Group
HIFF Specification
5900 Hollis Street, Suite O
Emeryville, CA 94608

AppleLink, Connect, MCI: HMG

Overview

The fundamental concept of HIFF is to serve as a central “hub” or *lingua franca* connecting a variety of hypermedia environments. The process of transferring a document from one environment to another is either a two or three step process:

- **Exporting** the document from its source environment;
- **Transporting** the resulting HIFF file physically or electronically, if necessary, to a computer where the target environment is available;
- **Importing** the HIFF data into the target environment.

Software that creates a HIFF file from a source document is termed an *exporter*. Software that creates a new document from a HIFF file in a target environment is termed an *importer*. The transport step, if required, is outside the scope of HIFF, except that HIFF assumes that *no automated modification (such as character set mapping) is applied to the file’s data during a transport process, with the sole exception of adapting line delimiters to the conventions of a particular environment.*

Once a document has been exported to HIFF form, the resulting HIFF file can, in theory, be imported into any number of target environments that have an importer available.

Design philosophy

The specification for HIFF draws heavily and unabashedly on existing standard formats for document interchange, such as the Rich Text Format (RTF) developed by Microsoft Corporation, the Tag Image File Format (TIFF) established by Aldus Corporation and Microsoft Corporation, and the Drawing Exchange Format (DXF) from Autodesk, Inc. In theory, though not yet in practice, any of these or other standard formats might be used to represent elements of data within the larger HIFF structure.

HIFF files are presently specified to be in readable (and, optionally, editable) text format, limited to the printable ASCII character set to maximize their transportability among environments.

Wherever possible, HIFF derives its terms for object types, properties and property values from existing hypermedia environments. In choosing terms for HIFF, the general rule of thumb was to adopt the term used in the earliest generally available product that incorporated a particular object type property or value. For example, while the terms *card* (used by HyperCard from Apple Computer, Inc., and SuperCard from Silicon Beach Software, Inc.) and *page* (used by HyperPAD from Brightbill-Roberts, Inc. and ToolBook from Asymetrix Corp.) both refer to approximately the same object type, HIFF documents exported from all these environments would use *card* as the standard term for that object type. It is the responsibility of HIFF importers to map terms appropriately for their environment.

EXPORTERS AND IMPORTERS

Responsibilities of a HIFF exporter

The responsibilities of a HIFF exporter, regardless of the environment from which it is exporting, include:

- Examine a selected document, created in a source environment that the exporter is equipped to interpret, to determine its form and content. (In addition to individual documents, an exporter may operate on selected sets of

documents. Each document, however, will yield a separate HIFF file.)

- Itemize and describe all (or a user-selected subset of) the objects comprising the document, their properties, and the values of those properties. Data or content (e.g. the text of a field) is considered a separate object or property (depending on context) for this purpose.
- Order the descriptions in a hierarchical structure paralleling the structure of the source document.
- Render the structured description into HIFF text format, with all objects, properties and values described in readable words where appropriate, and in hexadecimal text form where the data is of an essentially binary nature.
- Write the resulting description to a plain text file.

Responsibilities of an importer

The responsibilities of a HIFF importer include

- Determine from the header information in a text file whether it contains HIFF data in a version the importer is equipped to recognize.
- If so, create a new document in the target environment.
- Proceed object-by-object through the descriptions in the file. If an object has an analog in the target environment, create that analog in the new document; set analogous properties for the object to analogous values.
- Ignore and pass over all object, property, and value descriptions that do not have an analog in the target environment or that are unknown to the interpreter.
- Report to the user any anomalies or incompletely-translated elements in the new document that may require further attention to make the new document fully functional.

What HIFF does *not* translate to a common format

The HIFF format, as presently specified, does not attempt to translate any of the following into a common form; instead it carries them, if at all, in their original form. Translation, if

possible, is left to source-specific tools that may be available in the importer.

Scripting languages — Object scripts are presently exported unchanged from their native form in their creator environment. It is the responsibility of a HIFF importer to identify the source scripting language and version of a HIFF file, and either to employ its own cross-compiler for translating scripts to its environment, or to “comment out” any imported scripts that it cannot identify as executable in its environment.

Graphics— Rather than defining its own interchange format for graphics, HIFF relies on existing standardized graphics formats that can be incorporated within the HIFF framework, such as the Macintosh PICT format and the Windows bitmap format. Any graphic object in a HIFF file is required to have a property that specifies the format in which its image is encoded. It is the responsibility of a HIFF importer to provide or be able to access tools to interpret any standardized formats that the interpreter supports.

“Externals” — Machine-specific extensions to a hypermedia application may be exported as unmodified binary data if a user desires. Importers, however, are not expected to translate them or install them into the target environment. An importer may include facilities for mapping calls to externals from a source environment to comparable externals or native features in its target environment.

Resources — Data structures for information that are defined not by a hypermedia application but by the operating environment in which it exists, such as fonts, cursors and sounds, are either omitted or (generally at user option) exported in their native form into a HIFF file. It is the responsibility of an importer to determine if exported resources are relevant in its environment, and if so, to provide the capability to install them. As with externals (which are a special instance of the general category resources), an importer may include facilities for mapping uses of resources from a source environment to comparable resources or alternative structures in its target environment.

Audio-visual data — Video images and motion sequences, analog audio or digitized audio stored in other than a standard resource format, external device control capabilities (other than by standard script calls to an input or output port where supported by a source environment), and event time sequence information (e.g. SMPTE time code) are presently beyond the scope of HIFF.

SPECIFICATION FORMAT

Typographic conventions

Example contents of a HIFF file in this document appears in this typeface. Lines of example text too long to be printed as a single line on this page are wrapped around and indented like the following:

```
    This is a single line of HIFF information
      too long to be shown here on one line.
      Its continuation lines are indented.
    This is a second line of HIFF text
```

The newline character(s) is/are represented here as <CR> or the word *newline*, and understood to mean either <CR> or <CR><LF>.

Terms

In this document, the following terms are used as defined below:

- *Alphabetic character* — the letters A...Z and a...z, and the underscore character _ (ASCII 95 decimal) .
- *Numeric character* — the numerals 0...9, the character - precedes a numeral, and the character . (period) when it appears between two numerals.
- *Alphanumeric character* — either an alphabetic or numeric character.
- *Nonalphabetic character, nonnumeric character, nonalphanumeric character* — any printable ASCII character (i.e. in the range 32 to 126 decimal) not belonging to the respective sets defined above.
- *Backslash* — the character \ (ASCII 47 decimal)

- *Open-brace* — the character { (ASCII 123 decimal)
- *Close-brace* — the character } (ASCII 125 decimal)

Scope

This version of the HIFF specification defines the general structure and syntax for a HIFF file, and defines keywords (see below) sufficient to export HyperCard version 1.0 through 1.2.5 stacks, plus some additional keywords anticipated to be necessary as the standard is extended to accommodate other environments. Further versions are anticipated to define keywords for exporting from other environments.

THE HIFF FILE

Form

A HIFF file is a plain, unformatted ASCII text file as defined by the computer and operating system environment where it is created. In environments that employ the concepts of resources or resource forks, a HIFF file is not expected to have any resources or resource fork, and no significant information is stored in such structures.

The standard extension for a HIFF file in DOS and OS/2 environments is .HIF, though HIFF importers should also automatically recognize files with the extension .TXT. The standard filetype in the Macintosh environments (and any other using similar filetype conventions) is HIFF, though HIFF importers should also automatically recognize files of filetype TEXT. There is no specified file creator for HIFF files.

A HIFF file uses only the standard ASCII printable character set, and the standard control codes for <CR>, <LF> and <TAB>. All other control codes and characters are represented in “escaped” format as detailed below, and interpreted according to the file’s `character_set` property.

Newline characters are significant structure information within a HIFF file, however HIFF importers are insensitive to whether <CR>, <LF>, <CR><LF> or <LF><CR> is used to delimit lines, so

long as their usage is consistent throughout a file. Line delimiters may be translated freely to conform to a particular environment's convention for convenience in reading and editing the file with a text editor.

Information that has no meaningful text representation is encoded in hexadecimal form, two characters per byte, with 16 bytes per line followed by a newline. The final line may contain fewer than 16 bytes, and is concluded with a newline. HIFF does not add any length or error-correction information to the raw data. Here is an example of the format:

```
000001000300050009f8100820084008
2008100809f805000300010000000000
0007000a
```

A HIFF file begins with a header, followed by descriptions of zero or more objects, and concluded with the `\end_HIFF` control word. A minimal HIFF file would be:

```
HIFF (tm)
HyperMedia Interchange File Format -- TM, The
    HyperMedia Group
\HIFF_version {1.0}
\end_HIFF
```

If scripts of objects exist in both text and compiled forms, it is the text form that is exported to the HIFF file.

Syntax

Information in a HIFF file is either **header information**, a **control word**, a **control symbol**, or an **object description**. Object descriptions consist of an **object_type** declaration followed by zero or more **property_type** declarations, each followed by zero or more characters of **property_value** information. Control words and symbols, object and property types, and defined property values, are collectively termed **keywords**.

The characters `\`, `{`, and `}` are defined, sometimes in conjunction with a newline, as special characters used to indicate the start and grouping of control words and symbols, object and property declarations, and property values. Where these characters appear in the data for other than their special

HIFF purposes, they are respresented in “escaped” form as defined below.

The {, and } characters are used to indicate a **group**, which may be either data comprising a property value, or properties associated with an object. Structures or collections of objects larger than one object are not presently grouped by HIFF.

Defined HIFF words are **case-insensitive**. Thus \end_HIFF is equivalent to \end_hiff or \end_Hiff.

Blank lines (i.e. lines containing only a newline character) are not significant within HIFF except when contained within a property value. They may be included as desired to enhance readability, or omitted.

“Escaped” character format

Characters outside the printable ASCII set (including <CR>, <LF> and <TAB>) are respresented in escaped format. Each escaped character is represented by a group of four characters. The first character is a backslash, the second character is an apostrophe, and the third and fourth characters are the hexadecimal value of the character, padded with a leading zero if needed. The specially defined printable characters \, {, and } are also represented in escaped format by adding a leading backslash. In summary:

<u>Character</u>	<u>Escaped character</u>
Any below ASCII 32, except <CR>, <LF> or <TAB>	\'nn — where nn = the hex ASCII value 00 > 1f, left-padded with a zero if needed, e.g. \'02 or \'1c
Any above ASCII 127	\'nn — where nn = the hex ASCII value 7f > ff e.g. \'80 or \'a5
\	\\
{	\{
}	\}

Header information

The first 9 characters of a valid HIFF file must be

HIFF (tm)
followed by a newline.

The second nonblank line of a HIFF file must be
HyperMedia Interchange File Format -- TM, The
HyperMedia Group
followed by a newline.

The third nonblank line of a HIFF file must contain the control
word `\HIFF_version` and a value. The value for files conforming
to this version of the specification is 1.0, thus:
`\HIFF_version {1.0}`

Following this header, the next nonblank line begins the object
descriptions.

Control words

A HIFF control word is structured as a backslash followed by a
string of alphabetic characters and delimited by a space or an
open-brace. The HIFF control words presently defined include:

<i>Control word</i>	<i>Meaning / Defined values</i>
<code>\HIFF_version</code>	Version of the HIFF specification to which a file conforms; currently 1.0
<code>\end_HIFF</code>	End of a HIFF file; no significant data beyond this line

Control symbols

A HIFF control symbol is structured as a backslash followed by a
single nonalphabetic character. Presently no HIFF control
symbols are defined, however the structure is reserved.

Hierarchical structure

A HIFF file describes zero or more objects that are assumed to be
related in a hierarchical structure. Objects are described in a

hierarchy parallel to that in which the source application structures them.

Once the first object at a given level has been described, all objects belonging that object (such as all card buttons or card fields belonging to a particular card) are described before any further descriptions of objects at the given level. This pattern is recursive for each level of objects being described. An example of such a structure is:

```
document
  background 1
    non-card object 1 of background 1
    card 1 of background 1
      object 1 of card 1 of background 1
      object 2 of card 1 of background 1
      .
      .
      .
      object n of card 1 of background 1
    card 2 of background 1
      object 1 of card 2 of background 1
      object 2 of card 2 of background 1
      .
      .
      .
      object n of card 2 of background 1
    .
    .
    .
    card n of background 1
  background 2
  .
  .
  .
end-of-document
```

Objects belonging to lower hierarchical levels are assumed by HIFF to belong to the immediately-preceding higher-level object capable of “owning” an object of the type being described.

At a given level, all objects of a particular object type are described in sequence before any other object types at that level are described. The sequence used is whatever sequence the source application interprets the objects to be in. The order in which object types are described at a given hierarchical level may be arbitrary.

Thus, objects are grouped by their sequence in the HIFF file , which reflects the source document's implicit hierarchy. No additional punctuation or formatting is used, for example, to make explicit a grouping of objects that all reside on a particular card or background.

Default hierarchy: A HIFF file is assumed to have the following object hierarchy if no `object_hierarchy` property is included in the file:

```
file
  resources
  parent_file_information
  backgrounds
    background objects
  cards
    card objects
```

OBJECTS, PROPERTIES, AND VALUES

Following is a listing of the objects, properties and values defined in this version of HIFF, and their definitions. Note that exporters (and their users) are free to include or omit any class of object or property when creating a HIFF file (though an exporter should theoretically be capable of exporting full detail of a source document), and importers are free to ignore any class of object or property unrecognized by them, and to use default values for any properties whose values are not supplied by the exporter.

Objects

<u>Object</u>	<u>Meaning and notes</u>
<code>\file</code>	The entire hypermedia document as defined by the creator application.
<code>\parent_file</code>	Any other document, by the same creator, known to be in the message inheritance path of the current document. In HyperCard 1.x, the "Home" stack. Note that resource chains are beyond the scope of the <code>parent_file</code> object.

<code>\resType resource</code>	A resource, of type <code>resType</code> , whose data structure is wholly incorporated in the current document. HIFF does not distinguish between resources stored in a separate resource fork of the file (e.g. HyperCard) and those not so separated (e.g. SuperCard). HIFF considers each type of resource as a separate object type, declared by placing a backslash in front of the type name used by the source environment (e.g. <code>\ICON</code>) This object type is followed by a space and the keyword <code>resource</code> to distinguish such objects from non-resource objects with an identical name (e.g. <code>MENU</code>).
<code>\menu</code>	Reserved for future definition.
<code>\menu_item</code>	Reserved for future definition.
<code>\window</code>	The screen window within which an associated group of backgrounds and cards is displayed.
<code>\background</code>	An object of defined area which may contain other objects that appear as common elements in one or more <code>card</code> instances of the background.

<code>\background_graphic</code>	A graphic image object attached to a background. One property must define the form used to enclose the image (e.g. <code>\macpict</code> to represent a Macintosh PICT format image).
<code>\background_button</code>	An object whose primary purpose is the initiation of some action, attached to a background.
<code>\background_field</code>	An object whose primary purpose is to contain, display, and/or allow the entry of text, attached to a background.
<code>\card</code>	An object of defined area that is one instance of a background and that may contain other objects.
<code>\card_graphic</code>	A graphic image appearing on a card layer of the source document. This object type must have one property defining the form used to enclose the image (e.g. <code>\macpict</code> to represent a Macintosh PICT format image).
<code>\card_button</code>	An object whose primary purpose is the initiation of some action, attached to a card.
<code>\card_field</code>	An object whose primary purpose is to contain, display, and/or allow the entry of text, attached to a card.
<code>\background_field_data</code>	Text information attached to a particular card instance of a <code>background_field</code> .

Properties

In the tables that follow, where the *Meaning/Defined values* column is left blank for an entry, the meaning is assumed to be obvious from the name of the property. If no values are supplied, they are assumed to be either arbitrary text data (e.g. a name) or boolean data (`true` or `false`) obvious from the context. If the name and version of a hypermedia application appears in the column, the *Defined values* are those first defined for the property by that version of that application in its published documentation.

Implicit \HIFF_id property

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
\HIFF_id	The unique identification of an object description within the HIFF file.	1

1. This property is assumed implicitly for every object encoded in a HIFF file, though it may be encoded explicitly at the option of the exporter. It is a sequential numbering of all objects described in the file, beginning with 1 for the initial `\file` object, and incremented by 1 for each subsequent object described. Files subject to editing in which objects descriptions may be removed or added should include explicit `\HIFF_id` numbering. This property may be useful in resolving object ownership ambiguities, when used in conjunction with the `\parent_object` property.

File

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
\creator	The name of the application which created the source document, e.g. HyperCard.	1

<code>\creator_version</code>	The version of the creator application which created the source document. If the source application has a <code>version</code> property or equivalent, the value will be in the format returned by the application.	2
<code>\H_resolution</code>	The number of the source environment's finest units of horizontal measure that equal the standard unit of measure defined by the property <code>H_resolution_unit</code> .	
<code>\H_resolution_unit</code>	Presently, only <code>inch</code> is defined	
<code>\V_resolution</code>	The number of the source environment's finest units of vertical measure that equal the standard unit of measure defined by the property <code>V_resolution_unit</code> .	
<code>\V_resolution_unit</code>	Presently, only <code>inch</code> is defined	
<code>\character_set</code>	The standard character set to which text information associated with objects (e.g. names, scripts, data properties) is mapped. <code>ansi</code> — ANSI standard <code>mac</code> — Apple Macintosh <code>pc</code> — IBM PC <code>pca</code> — IBM PS/2 code page 850	3

<code>\object_hierarchy</code>	Reserved for future definition.	
<code>\name</code>	The name by which the document's file is known to the file system of the source environment.	4
<code>\caption</code>	The name by which the document is identified to the user, if different from the file system name.	
<code>\cantModify</code>	Whether the document is read-only or can be changed. Either <code>true</code> or <code>false</code> .	
<code>\size</code>	The size, in bytes, that the source application or its operating system reports for the document.	
<code>\freesize</code>	The reduction in size, in bytes, that the source application reports would be made available if it compacted / compressed the document's data. Useful primarily for user feedback.	
<code>\external_list</code>	A list of external commands and functions (e.g. HyperCard XCMD resources) included in the file, one per line, with each line containing a comma-separated list of the resource's type, name, ID, and size, e.g. XCMD, Flash, 26, 128	5

1. The HIFF name of an application is the name appearing with the primary emphasis on the product's packaging or documentation, omitting any trademark or similar "bugs" and any descriptive phrases.
2. Note that if the document contains objects, properties, or script commands which were added by editing the document in a later version of the application, the importer may not recognize them.
3. A character set specification at the file level can theoretically be overridden by a property specified for a particular object containing text information, but this ability is not presently supported.
4. Throughout HIFF, `\name` property values (and `\id` property values) are in the format that HyperCard terms the "short" name or ID, that is the name or ID data by itself, with no qualifying words or punctuation.
5. This information may, to some degree, duplicate information encoded as `\resource` objects. It is provided because it is likely that users will choose to omit code resources from HIFF files, but script importer/translators will need to identify calls to externals that are no longer available in their original form, so that they can replace them or map them to alternatives available in the target environment.

Parent_file

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\name</code>	The name by which the parent file is known to the file system of the source environment.	

\resource_list	<p>A list of resources included in the parent file, one per line, with each line containing a comma-separated list of the resource's type, name, ID, and size, e.g.</p> <p>ICON, Arrow, 26, 128</p>
\handler_list	<p>A list of handlers that the parent file places in the message path of the source document, one per line.</p>
\function_list	<p>A list of functions that the parent file places in the message path of the source document, one per line.</p>
\sequence	<p>Reserved for future definition.</p>

Resource

<u>Property</u>	<u>Meaning / Defined values</u>	<u>Notes</u>
\name		
\id	as identified in the source environment	
\size		
\data	in hex format	

Window

<u>Property</u>	<u>Meaning / Defined values</u>	<u>Notes</u>
\name		1

<code>\left_top</code>	horizontal, vertical — in the source environments standard units of screen position measure
<code>\size</code>	width, height
<code>\cantModify</code>	
<code>\number_backgrounds</code>	Total number of backgrounds in the window
<code>\number_cards</code>	Total number of cards in the window
<code>\script</code>	

1. Where documents have only one window, or no defined window structure, the window name and the file name are the same.

Background

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\name</code>		
<code>\number</code>		
<code>\ID</code>		
<code>\background_size</code>	height,width — in the units defined by the file's <code>H_resolution</code> and <code>V_resolution</code> properties.	
<code>\cantDelete</code>		
<code>\script</code>		

Background_graphic

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
-----------------	-------------------------------------	--------------

<code>\type</code>	<p>The standard graphics format in which the graphics data is encoded.</p> <p><code>macpict</code> — Macintosh PICT format.</p> <p><code>wbitmapn</code> — Windows bitmap type n.</p> <p><code>wmetafilen</code> — Windows metafile type n.</p>	
<code>\rect</code>	<p><code>left,top,right,bottom</code> — rectangle of the graphic's bounding frame.</p>	1
<code>\layer</code>	<p>Reserved for future definition.</p>	2
<code>\data</code>	<p>hex data representing the graphic object.</p>	

1. The encoded data in the graphic description may define a smaller frame positioned within that defined by the `rect` property.
2. Throughout HIFF, the `\layer` property is reserved for defining the visual layering hierarchy for all objects belonging to a parent object, considered as a group without regard to object type.

Background_button

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\name</code>		
<code>\number</code>		
<code>\ID</code>		
<code>\rect</code>		
<code>\layer</code>	Reserved for future definition.	
<code>\style</code>		

<code>\showName</code>		
<code>\hilite</code>		
<code>\autoHilite</code>	HyperCard 1.x	
<code>\textFont</code>	The font name (not number) used for the font by the source environment. See notes in <code>background_field</code> object regarding styled text.	
<code>\textSize</code>		
<code>\textStyle</code>		
<code>\textAlign</code>		
<code>\visible</code>		
<code>\icon</code>	The number of the icon resource which is attached to the button.	1
<code>\icon_data</code>	A copy of the data property value of the icon resource.	2
<code>\script</code>		

1. HyperCard uses the value 0 (zero) to indicate no icon is attached to the button. HIFF maps this value to a null or empty value to avoid confusion with any icon resource that happens to be numbered 0.
2. This can be somewhat redundant if the icon resource exists in the source document (rather than elsewhere in a resource chain). However, it also provides for capture of the images of icons that are in parent files or elsewhere in a resource chain, allowing the button graphic to be reconstructed even in the absence of the resource. And it provides flexibility for handling environments where icons are managed differently than in the resource model used by Macintosh.

Background_field

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
\name		
\number		
\ID		
\rect		
\layer	Reserved for future definition.	
\style		
\lockText	HyperCard 1.x	
\showLines	HyperCard 1.x	
\wideMargins	HyperCard 1.x	
\autoTab		
\data_format	Reserved for future definition.	1
\textFont	The font name (not number) used for the font by the source environment.	1
\textSize	in points	
\textHeight	in points	
\textStyle	HyperCard 1.x	
\textAlign	HyperCard 1.x	
\visible		
\script		

1. The text properties associated with fields in this version of the HIFF specification are oriented toward HyperCard 1.x fields limited to a single text style for the entire contents. However the \data_format property reserved for future definition will enable specification of an encoding scheme for styled text, such as a subset of Microsoft's RTF.

Card

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\name</code>		
<code>\number</code>		
<code>\ID</code>		
<code>\cantDelete</code>		
<code>\script</code>		

Card_graphic

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\type</code>	See <code>\background_graphic</code> .	
<code>\rect</code>		
<code>\layer</code>	Reserved for future definition.	
<code>\data</code>		

Card_button

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\name</code>		1
<code>\number</code>		
<code>\ID</code>		
<code>\rect</code>		
<code>\layer</code>	Reserved for future definition.	
<code>\style</code>		
<code>\showName</code>		
<code>\hilite</code>		
<code>\autoHilite</code>		

\textFont
\textSize
\textStyle
\textAlign
\visible
\icon
\icon_data
\script

1. See the description of \background_button properties for notes that apply also to the \card_button object.

Card_field

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
\name		1
\number		
\ID		
\rect		
\layer	Reserved for future definition.	
\style		
\lockText		
\showLines		
\wideMargins		
\autoTab		
\textFont		
\textSize		
\textHeight		
\textStyle		

<code>\textAlign</code>	
<code>\visible</code>	
<code>\script</code>	
<code>\data_format</code>	Reserved for future definition.
<code>\data</code>	Contents of the field.

1. See the description of `\background_field` properties for notes that apply also to the `\card_field` object.

Background_field_data

<i>Property</i>	<i>Meaning / Defined values</i>	<i>Notes</i>
<code>\number</code>	The background field , identified by number, in which to place this data.	
<code>\data_format</code>	Reserved for future definition.	
<code>\data</code>	Contents of the field.	

POSSIBLE FUTURE ENHANCEMENTS

HIFF is intended to be an extensible, expandable format. It is anticipated that future HIFF specification releases will formally incorporate additional object types, properties and property values to more fully support a variety of hypermedia and multimedia applications and environments. Among those anticipated (though by no means an exhaustive list) are:

- SuperCard, from Silicon Beach Software
- Plus, from Spinnaker Software
- HyperPad, from Brightbill-Roberts, Inc.
- HyperCard 2.0 and future versions, from Apple Computer, Inc.