

HyperTIES

References

- [Designing to facilitate browsing: A look back at the Hyperties workstation browser](#)
By Ben Shneiderman, Catherine Plaisant, Rodrigo Botafogo, Don Hopkins, and William Weiland.

Illustrations

- [HyperTIES Diagram](#)
- [HyperTIES Authoring with Emacs](#)

Interesting Features

- **Implementation:**

HyperTIES is written in a combination of C, Forth, NeWS PostScript, Emacs Mocklisp, and HyperTIES Markup Language.

- **Storyboard Interpreter:**

The storyboard interpreter reads in and parses the fields in a storyboard file, and drives the HyperText machinery based on the commands and text therein.

Fields of a Storyboard:

- **Title Field:** A storyboard starts with the line ".title", followed by the storyboard's title on the next line.
- **Synonym Field:** Starts with the line ".synonyms" (or ".synonym"). Followed by the synonyms of the storyboard, each on its own line. The synonym field is optional. It ends with the beginning of the .definition field.
- **Definition Field:** Starts with the line ".definition". It contains text and formatting commands that describes the definition image. It can have text, font changes, pictures and targets, like the .contents field, but it should not generally have reference buttons in it. This field extends to the beginning of the .contents field, or to the end of the file, if the storyboard has no contents.
- **Contents Field:** Starts with the line ".contents". It contains the actual body of the HyperTIES page. Text is automatically formatted and paginated to fit in the window, and the HyperTIES markup language interpreter executes commands and macros whose name begin with a period (".commands").

- **Page Formatter:**

The storyboard interpreter drives a page formatter, whose job it is to decide where things go on the page, and to describe the pages by calling remote PostScript procedures in the NeWS server.

These PostScript procedures describe and manipulate windows and pages.

A HyperTIES window is comprised of a page for the control panel, a page for the definition window, and zero or more pages for the storyboard contents.

The formatter describes a page by building a display list of PostScript code associated with the page in the NeWS server. NeWS executes the display list to draw the page's image.

It puts code to change fonts and show lines of text into the display list. It also creates stamp-pad objects, used to put picture drawing code into the display list, and to render buttons and targets onto the page.

Arguments passed to important functions:

Instantiation (create a stamp-pad):

Instantiate Stamp: object body, class, name, instance ID, filepos

Instantiate Target: object body (args, init), class, name, instance ID, filepos

Display (put down a stamp):

Display Stamp: instance ID, position

Display Target: instance ID, position, size

Display String: string, position

- **Compiling Storyboards into Forth:**

The storyboard interpreter is capable of compiling storyboards into Forth programs that call low level formatting commands to describe pages to the NeWS server.

The interpreter compiles storyboards by formatting them, and printing Forth words to a text output file, capturing the execution of the low level formatting commands in forth function definitions.

HyperTIES can subsequently read in the resulting Forth functions, and compile them into memory. Then they can be efficiently executed, to produce identical pages as the storyboards interpreted from disk, without the overhead of the formatter reading the storyboards and laying out the page.

The memory image of the HyperTIES Forth system, loaded with all the compiled storyboards of a database, can be saved out to disk, so that later sessions can restore the environment. This significantly improves the response time for displaying definitions and contents, without noticeably increasing the startup time.

- **Objects:**

In addition to the storyboard files (with .st0 suffixes), there exist picture files (with .pn0 suffixes) and targets files (with .tn0 suffixes), describing the objects referred to by the author in the storyboard files.

You can think of the objects described in the .pn0 and .tn0 files as stamp pads. When an object is referred to in a storyboard, it is used to stamp a picture or a target down at some location on the page.

The .pn0 and .tn0 files contain an object's class, name, and body.

- **Class:** An object's class defines its behavior. Currently, the class is the name of a PostScript class defined in the NeWS server. In the future, some classes may be totally or partially implemented in the HyperTIES client, in C or C++.
- **Name:** In the storyboard files, an author can refer to an object by its name. The master index maps from the object's name to its location in a .pn0 or .tn0 file.
- **Body:** The interpretation of an object's body, determining its characteristics, is up to its class.

For all the currently implemented classes, the object's body contains PostScript code to be interpreted by the NeWS server. The server executes the body, which pushes onto the stack the parameters for the /new message, sent to the class implemented in the NeWS server, to instantiate the stamp-pad object the first time it is referenced.

- **Data Structures:**

- **Index:**
 - 3 namespaces of index entries:
 - storyboard names (.st0)
 - picture names (.pn0)
 - target names (.tn0)
- **Entry:**
 - name
 - file, offset, length
 - instance header
 - instance ID
 - class
 - width, height
 - NeWS StampDict entry
 - instance ID
 - object
- **Pile:**
 - PileDict
- **Page:**
 - PageDict
 - Canvas
 - EventMgr
 - DL
 - Targets
- **Target:**
 - TargetDict
 - TID
 - Object
 - Ref

- **Classes:**

- **Class Stamp:**
 - **Methods:**
 - /new % name id filepos => Obj

- /compile % SubClassResponsibility
 - /size % - => width height
 - /dlist % {PostScript code ...} => <PostScript code ...>
 - **Instance Variables:**
 - /Name /ID /FilePos /X /Y /Width /Height
- **Class Picture extends Stamp:**
 - **Methods:**
 - /new % source width height name id filepos => obj
 - /compile % x y => x y <PostScript code ...>
 - **Instance Variables:**
 - /Source
- **Class UnitPicture extends Picture, Stamp:**
 - **Methods:**
 - /compile % x y => width height x y gsave translate scale
<PostScript code ...>
grestore
- **Class ScaledRaster extends Picture, Stamp:**
 - **Methods:**
 - /compile % x y => width height x y gsave translate scale
<canvas> imagecanvas
grestore
- **Class Raster:**
- **Class TargetStamp:**
- **Class Target:**
- **Data Files:**
 - Master Index file -- master-index
 - Storyboard file -- .st0
 - Picture file -- .pn0
 - Target file -- .tn0

Interesting Files

- **Python Files:**

Recently written Python code, to convert HyperTIES database to XML, for other programs to interpret.

 - [convert.py](#): Python source file: Convert a HyperTIES database to XML.
 - [XMLUtilities.py](#): Python source file: XML Utilities, like it says.
 - [HyperTIESDatabases.xml](#): HyperTIES databases converted to XML by "convert.py".
- **C Files:**

Used to implement formatting library and second generation markup language interpreter.

 - [fmt.c](#): C source file: NeWS HyperTIES formatter. By Don Hopkins.
 - [fmt.cps](#): C to PostScript header file: HyperTIES C to NeWS interface (generates fmt.h). By Don Hopkins.
 - [fmt.h](#): C header file: NeWS HyperTIES formatter (machine generated from fmt.cps). By Don Hopkins.
 - [master-index.c](#): C source file: Master index database. By William Weiland.
 - [master-index.h](#): C header file: Master index database. By William Weiland.
 - [index.c](#): C source file: HyperTIES index. By William Weiland.

- [index.h](#): C header file: HyperTIES index. By William Weiland.
- [entry.c](#): C source file: Master index entry. By William Weiland.
- [entry.h](#): C header file: Master index entry. By William Weiland.
- [object.c](#): C source file: HyperTIES object. By William Weiland.
- [object.h](#): C header file: HyperTIES object. By William Weiland.
- [tree.c](#): C source file: HyperTIES tree. By William Weiland.
- [tree.h](#): C header file: HyperTIES tree. By William Weiland.
- [primitives.c](#): C source file: HyperTIES primitives. By William Weiland.
- [primitives.h](#): C header file: HyperTIES primitives. By William Weiland.
- [make-index.c](#): C source file: Master index creation utility. By William Weiland.
- [psexec.c](#): C source file: NeWS operating system command executer utility. By James Gosling, Don Hopkins.
- [wrapper.c](#): C source file: Dynamic loader for Forth. By Mitch Bradley.

• **Forth Files:**

Used to implement first generation markup language interpreter and compiler.

- [fmt.f](#): NeWS Forth HyperTIES storyboard interpreter stuff. By Don Hopkins.
- [case.f](#): Dr. Charles Eaker's case statement. By Dr. Charles Eaker.
- [clink.f](#): Utilities for dynamically linking C libraries into Forth. By Mitch Bradley.
- [cload.f](#): Utilities for dynamically loading C libraries into Forth. By Mitch Bradley.
- [ctypes.f](#): Utilities for accessing C types from Forth. By Mitch Bradley.
- [fcall.f](#): Utilities to allow C routines to call Forth routines. By Mitch Bradley.
- [sh.f](#): Unix shell interface for Forth. By Mitch Bradley.
- [random.f](#): Random number generator in Forth. By Mitch Bradley.
- [string-equals.f](#): String comparison in Forth 68k assembler. By Mitch Bradley.

• **NeWS PostScript Files:**

Used to implement display engine, user interface, formatter prototype.

- [variables-doc.txt](#): Outline of HyperTIES NeWS variables and technical notes. By Don Hopkins.
- [fmt.ps](#): NeWS Forth HyperTIES PostScript stuff. By Don Hopkins.
- [stamp.ps](#): NeWS Forth HyperTIES Stamp Pad Class Definitions. By Don Hopkins.
- [target.ps](#): NeWS Forth HyperTIES Target Class Definitions. By Don Hopkins.
- [win.ps](#): NeWS Forth HyperTIES window class definitions. By Don Hopkins.
- [tiemacs.ps](#): NeWS Forth HyperTIES Emacs window class definitions. By Don Hopkins.
- [hookup.ps](#): Setup to make a connection with the NeWS server with psexec. By Don Hopkins.
- [term.ps](#): PostScript terminal emulator. By Don Hopkins.
- [neatwin.ps](#): NeatWindow Class pie menu based window manager. By Don Hopkins.
- [piemenu.ps](#): Pie menu class implementation for NeWS. By Don Hopkins.
- [textcan.ps](#): Scrolling text display and storage facility. By Steve Isaac.

• **Emacs Mocklisp Files:**

Used to implement integrated hypermedia authoring tool in UniPress Emacs.

- [ties.ml](#): MockLisp code: Emacs support for HyperTIES. By Don Hopkins.
- [ht.ml](#): MockLisp code: Emacs support for HyperTIES (later version, rewritten). By Don Hopkins.
- [ties-2.ml](#): MockLisp code: Yet Another HyperTIES Implementation, This Time In Emacs (YAHTITTIE) By Don Hopkins.
- [yahtittie.ml](#): MockLisp code: Yet Another HyperTIES Implementation, This Time In Emacs (YAHTITTIE) (latest version) By Don Hopkins.

• **HyperTIES Markup Language Files:**

Used to script hypermedia pages, macros, styles, templates and conditionals. First generation implemented in Forth, second generation implemented in C.

- [commands-doc.txt](#): List of HyperTies Markup Language commands. By Don Hopkins.
- [ties-doc.txt](#): Outline of HyperTIES technical documentation. By Don Hopkins.
- [todo.txt](#): HyperTIES ToDo List. By Don Hopkins.
- [newdb](#): HyperTIES database directory: Hubble Space Telescope By Catherine Plaisant, William Weiland, Rodrigo Botafogo, Don Hopkins.
- [doc](#): HyperTIES database directory: HyperTIES Documentation. By Catherine Plaisant, William Weiland, Rodrigo Botafogo, Don Hopkins.
- [cookbook](#): HyperTIES database directory: Pie Menu Cookbook. By Don Hopkins.
- [emacs](#): HyperTIES database directory: Emacs HyperTIES Authoring Tool Integration. By Don Hopkins.
- [newsdoc](#): HyperTIES database directory: NeWS Window System Documentation By Don Hopkins.
- [cam](#): HyperTIES database directory: Cellular Automata Machine By Don Hopkins.
- [global](#): HyperTIES database directory: Global Files Shared by Other Databases. By Catherine Plaisant, William Weiland, Rodrigo Botafogo, Don Hopkins.